

Introduction

The **MyDoorOpener** project has attracted a surprisingly high number of enthusiasts over the years. This is our third “simplified assembly process” document, which is always getting simpler and simpler with each release. This simplification is highly due to the increased interest in the IoT (Internet of Things) phenomenon, which drives creation of simplified hardware and lower production costs.

For reference purposes, you can still find all documents (past and current):

v3 (DFRobot Xboard Relay) *current*	http://mydooropener.com/downloads/MyDoorOpener-Instructions-3.pdf
v2 (Arduino + DFRobot Relay Shield)	http://mydooropener.com/downloads/MyDoorOpener-Instructions-2.pdf
v1 (Arduino + DFRobot Single Relay Shield)	http://mydooropener.com/downloads/MyDoorOpener-Instructions.pdf

This new even more simplified assembly process brings the project accessibility to a whole new level of simplicity and also affordability as the number of components is drastically reduced.

Again, the objective of this assembly process is to provide an easier way to build the project, particularly for people with more limited knowledge of general electronics.

This doesn't mean you don't need to know anything about electronics ... You still need to have some basic understanding. What it does mean is that you can get around with a simplified components selection/purchase process. For many, this is great news, which makes the project a whole lot more accessible.


Components Supplier







The components for this assembly process can be ordered directly from the following electronics parts suppliers (we are not affiliated with any of them in any way):

Suppliers
http://www.dfrobot.com
http://www.robotshop.com/

Components List










Following is the list of components that should be ordered to build an entry-level setup for the **MyDoorOpener** project (this setup can control 2 doors/devices):




 HOME  STORE  BLOG  WIKI  FORUM  COMMUNITY

Home / Shopping Cart

SHOPPING CART (0.18KG)

Image	Product Name	Model	Quantity	Unit Price	Total
	Xboard Relay Reward Points: 20	DFR0222	1  	\$34.95	\$34.95
	Micro USB Cable Reward Points: 2	FIT0265	1  	\$2.90	\$2.90
	Wall Adapter Power Supply 12VDC 1A (American Standard) Reward Points: 7	FIT0231	1  	\$6.90	\$6.90

Continue Shopping

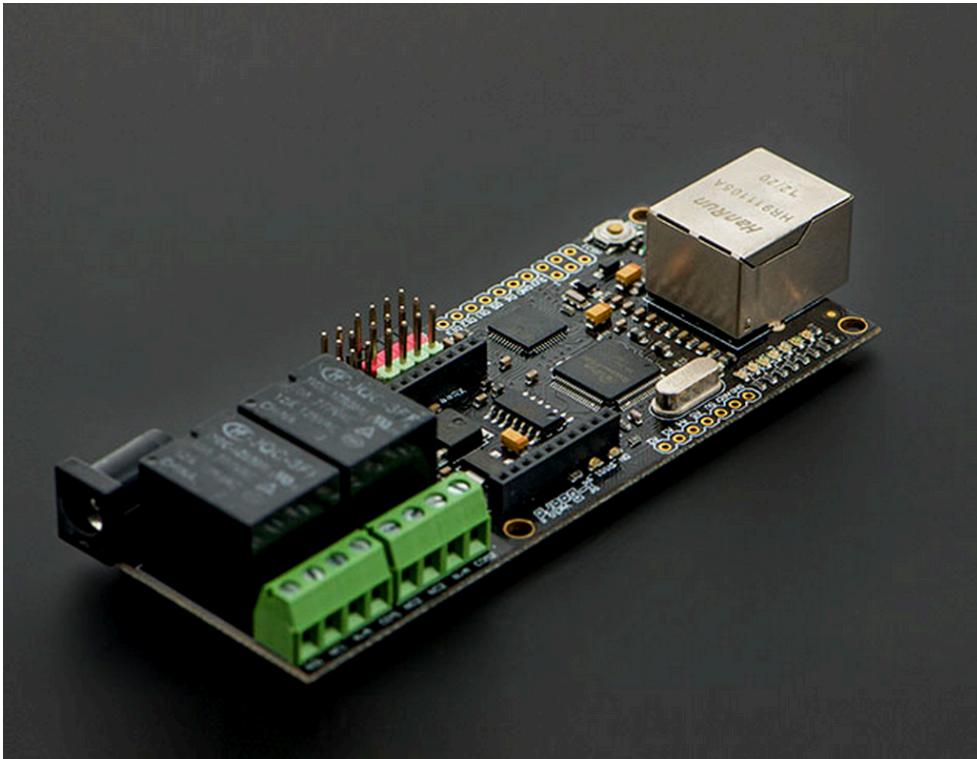
Checkout OR 

Sub-Total: \$44.75
Total: \$44.75

The prices above can give you an indication of rough costing for the project, but those prices may obviously change over time. The same applies for the actual part numbers and their availability. We can't guarantee anything, but we'll try our best to keep this document up to date (please report any discrepancy you find when purchasing your components).

The Hardware

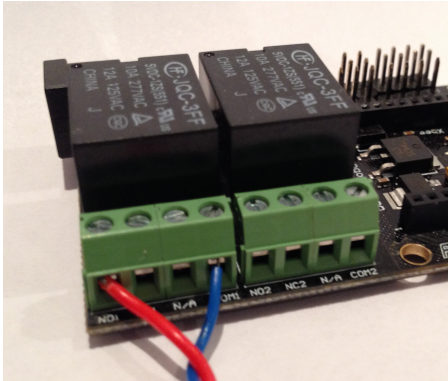
The DFRobot Xboard Relay is a 3-in-1 board that combines an Arduino micro-controller, an Ethernet shield and a Relay shield, all on a single board. It offers 2 controllable relays. Alternatively, you could use a separate Arduino Uno board, combined with a separate Arduino Ethernet shield and a separate Relay shield.



REF: http://www.dfrobot.com/index.php?route=product/product&product_id=837

The Assembly Process

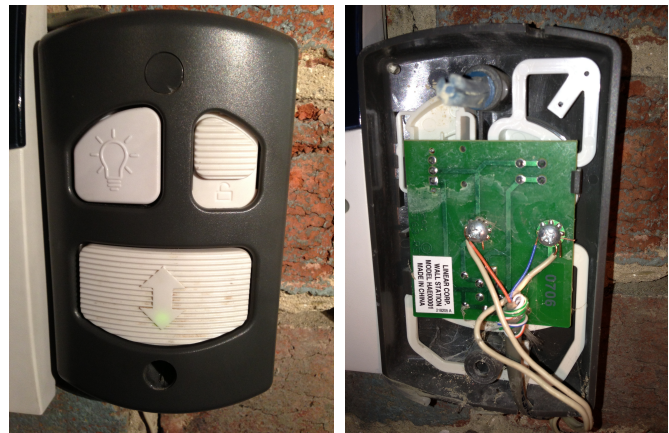
1. Assembly required for opening and closing of door/device.



Connect one of the DFRobot Xboard relays to your existing garage door opener wall button. This will require the installation of 2 low voltage wires going from one of the terminal blocks labelled COMx and NOx, to your existing garage door wall mount button station (the 'x' in COMx and NOx is a value between 1 and 2).

When you open up your garage wall mount button station, you'll typically find 2 screws onto which you'll attach the other ends of the 2 low voltage wires coming from the relay block.

It doesn't matter which terminal from the wall mount button station gets linked with COMx or NOx, for as long as the entire setup can form a closed loop whenever the relay gets triggered.

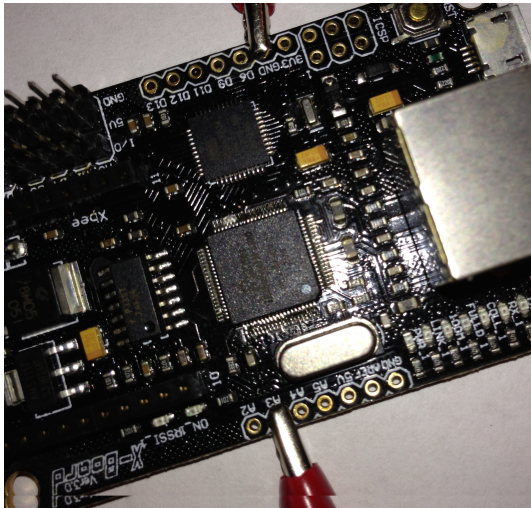


Repeat the above procedure for each door/device you want controlled (up to 2), using a distinct COMx/NOx terminal block for each door/device.

Following is the pin# used for each of the relays on the Xboard Relay:

Device #	Pin for relay
1 (operated via relay @ COM1/NO1)	Digital pin #7
2 (operated via relay @ COM2/NO2)	Digital pin #8

2.Assembly required for status monitoring (opened/closed) of door/device.



Connect the DFRobot Xboard Relay to your status sensing device, typically installed on your door/device. This will require the installation of 2 low voltage wires going from the Xboard Relay's GND pin and one of the board's analog pins. Have those 2 wires form a closed loop, going thru your status sensing device.

When connecting the 2 wires on the DFRobot Xboard Relay, you will need to solder them onto the board (or fasten them using silicone or any other fastener you're comfortable working with).

We recommend using the following pins for your status sensing device(s):

Device #	Pins for sensor
1 (operated via relay @ COM1/NO1)	Analog pin #3, GND
2 (operated via relay @ COM2/NO2)	Analog pin #4, GND

Your status sensing device should be configured so that it connects the GND to the analog pin when the door is closed. When the door gets opened, your status sensing device should disconnect the GND from the analog pin. This is how MyDoorOpener knows the door/device is opened or closed. So, GND attached to analog pin means the door is closed and a broken link means the door is opened.

You can use any status sensing device you wish, for as long as it respects the above rules. Typically, you'd use contact sensors which should be readily available from your local hardware store:

Typical alarm contact sensors should work great. You can also Google around "contact sensor" or "garage contact sensor" to find good candidates for your project.



Repeat the above procedure for each door or device you want monitored, using a distinct analog pin for each door/device. The GND pin should be shared for all sensors.

This concludes the hardware assembly process.

Software Configuration

1.Download the Arduino IDE software from the Arduino web site, based on the operating system you are using:

Operating System	Arduino Download
Windows	http://arduino.cc/download.php?f=/arduino-1.0.6-windows.exe
Mac OSX	http://arduino.cc/download.php?f=/arduino-1.0.6-macosx.zip
Linux 32bit	http://arduino.cc/download.php?f=/arduino-1.0.6-linux32.tgz
Linux 64bit	http://arduino.cc/download.php?f=/arduino-1.0.6-linux64.tgz

* Note that the version currently supported is **version 1.0.6**. Using any other version of the Arduino IDE is not officially supported.

2.Download the MyDoorOpener project ZIP file:

<https://github.com/yanavery/MyDoorOpener-Arduino/zipball/master>

3.Have the files extracted from the ZIP file so that the directory and file structure is similar to the following:

FAVORITES	Name	Date Modified
All My Files	▼ Arduino	Aug 11, 2010 20:15
AirDrop	▼ libraries	Nov 16, 2012 22:22
Music	▼ Aes256	Apr 19, 2011 18:50
Desktop	aes256.c	Mar 26, 2009 23:32
Downloads	aes256.h	Mar 26, 2009 23:33
Applications	▼ Time	Feb 22, 2010 16:07
Documents	Time.cpp	May 11, 2012 21:37
yavery	Time.h	Jan 11, 2010 20:20
Dropbox	▼ Webduino	May 11, 2012 21:23
	WebServer.h	Feb 10, 2012 4:34
	▼ MyDoorOpener	Today 17:45
	MyDoorOpener.ino	Today 17:35

Respecting the location of libraries is important. If you don't, you will get compiler warnings and/or errors when compiling the project with the Arduino IDE.

4. Open the MyDoorOpener.ino file inside the Arduino IDE. Edit the following lines to match your specific configuration (and save):

Line #	Item	Description
~ 53	IP Address	Change to match the network IP address you want your Arduino to be accessible from. You will need to make sure this network IP address is reserved and never assigned to any other devices on your internal network. This is not the address you will be accessing your Arduino from the iPhone application. You need to configure NAT forwarding on your home router to do so. See http://en.wikipedia.org/wiki/Port_forwarding for more details.
~ 58	Password	Change to the password you want to protect your Arduino with. This password needs to match the password you will be entering in the iPhone application.

5. **Optionally**, you can turn *ON* notifications from MyDoorOpener. By default all notifications are turned *OFF*. So if you don't care for notifications, you can skip this section altogether and move on to the next section of this document.

MyDoorOpener currently supports 2 types of notifications:

Watchdog Open Notification	As soon as a door/device gets opened, a timer is started and after a configurable amount of time, a notification gets fired, unless the door/device gets closed before the timer reaches its expiration.
Open Notification	As soon as a door/device gets opened, a notification gets fired.

Each notification type can be turned *ON* or *OFF* individually. You can get notified for a single type or for both. In all cases, notifications get fired whether the door/device gets opened thru the iPhone application or by any other means. For now, notifications are global, therefore, if turned *ON*, all sensors are observed. You currently can't specify which sensors are observed and which aren't.

To turn *ON* "Watchdog Open Notifications", uncomment the following line (near line ~107). You can also change the default value (5) to the number of minutes you want:

```
#define NOTIFICATIONS_WATCHDOG_MINUTES 5
```

To turn *ON* regular “Open Notifications”, uncomment the following line (near line ~110):

```
#define NOTIFICATIONS_OPEN
```

MyDoorOpener currently supports 3 notification broadcasting mechanisms:

Push Notifications	Will send a push notification to your iPhone. Requires the purchase of the Prowl iPhone application. http://prowlapp.com/
SMS Notifications	Will send an SMS message to your mobile phone, using your mobile carrier’s SMTP to SMS gateway. http://en.wikipedia.org/wiki/List_of_SMS_gateways/
Email Notifications	Will send an email message to a configurable email address, typically using your ISP’s SMTP server.

Each notification broadcasting mechanism can be turned *ON* or *OFF* individually. You can get notified using a single mechanism or all of them, for as long as you configure them appropriately.

To turn *ON* “Push Notifications”, uncomment the following line (near line ~117) and provide the required Prowl API key for your iPhone device (near line ~123).

```
#define PUSH_NOTIFICATIONS
```

```
static const char prowlApiKey[] = "paste-your-prowl-provided-api-key-here";
```

To turn *ON* “SMS Notifications”, uncomment the following line (near line ~141), provide the required smtp address for your mobile (near line ~146) and smtp server (near line ~160).

```
#define SMS_NOTIFICATIONS
```

```
static const char smtpToForSms[] = "your-mobile-number@your-carrier-gateway.com";
```

```
static const char smtpServerName[] = "smtp-server.your-isp.com";
```

To turn *ON* “Email Notifications”, uncomment the following line (near line ~155), provide the required smtp server (near line ~160) and email address (near line ~166).

```
#define SMTP_NOTIFICATIONS

static const char smtpServerName[] = "smtp-server.your-isp.com";

static const char smtpToForEmail[] = "your.address@gmail.com";
```

6. Compile the MyDoorOpener.ino project from within the Arduino IDE, using the **Sketch -> Verify/Compile** menu option. Make sure there are no compiler warnings or errors.

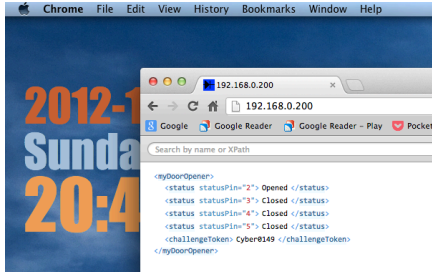
7. Connect your Arduino to your computer using a standard micro USB cable. From the Arduino IDE, choose the **Tools -> Board -> Arduino Leonardo** menu option. Upload the project, using the **File -> Upload** menu option.

8. Once the upload is completed, you can unplug the Arduino from your computer. You will also need to either “reset” the Arduino (using the reset button if one is available on the board) or unplug the AC power from the Arduino for a few seconds. This will have the MyDoorOpener program initiated.

This concludes the software configuration process.

Network Configuration

1. You should first test your installation by trying to access your Arduino with a web browser from within the same internal network. To do so, fire up your favorite web browser and type in the IP address you configured your Arduino to listen to, from step #4 of the software configuration process.



You should see a screen similar to the one on the left if things are working properly (your web browser must be able to display XML in order for this to work).

2. You should now configure your home router to do NAT port forwarding to your Arduino so that you can access it from the internet, using your home router's ISP assigned public IP address instead. This is required because the IP address assigned in step #4 of the software configuration process is an internal IP address, which is not visible from the internet.

The Arduino is listening on port 80 (default) so you will need to input a NAT forwarding rule that will forward to that port, off the IP address you configured in step #4 of the software configuration process.

How NAT port forwarding works is specific from router to router, therefore it is beyond the scope of this document to cover these specifics. You can either look at your home router manual or Google around if you aren't sure how this can be achieved with your particular home router.

3. Another aspect you might want to look into is dynamic DNS. Since your home router most likely gets assigned an IP address by your ISP, it is subject to change from time to time. In order to be shielded from such changes, it is recommended to use a name rather than an IP address. The name will map to an IP address and with dynamic DNS services, that name to IP mapping will get maintained whenever your ISP allocates a new IP address to your home router.

Again, the details on how dynamic DNS works and the intricacies for the many dynamic DNS service providers out there is beyond the scope of this document. You can Google around to get more details on how such service works, the many service providers that offer this service, as well as how it can be configured with your particular home router.

This concludes the network configuration process.

iPhone Configuration

1.Download the iPhone application from the Apple iTunes store:

<http://itunes.apple.com/app/mydooropener/id359774310>

2.Once installed on your iPhone, open the application and click the small configuration gear located at the top right of the main screen.

3.Fill-in the configuration fields as follows:

Controller

Field Name	Description
URL	The URL used to connect to your Arduino from the internet. This is not the internal IP address for your Arduino, but a name or IP address that is visible from the internet. Typically, this will be a URL with the name or IP address of your home router on which you'll have configured a NAT forwarding rule that will forward to your Arduino. This URL must have the form http://xxx.yyy.zzz:port where xxx.yyy.zzz is either a fully qualified name or IP address. The port number is optional.
Password	The password for your Arduino, as configured in the software configuration process. This must match as-is, and is case sensitive.

Doors & Devices

Field Name	Description
Relay Pin Number	Choose the relay pin number based on your setup, typically: COM1/NO1 = Relay pin #7 COM2/NO2 = Relay pin #8
Status Pin Number	Choose the status pin number based on your setup, typically: COM1/NO1 = Status pin #3 COM2/NO2 = Status pin #4

We hope this document was helpful in setting up your **MyDoorOpener** project.

If you have questions or if find inaccuracies in this document, please use our online forums to find answers to your questions or report problems.

Thanks for using MyDoorOpener and for your continued support!