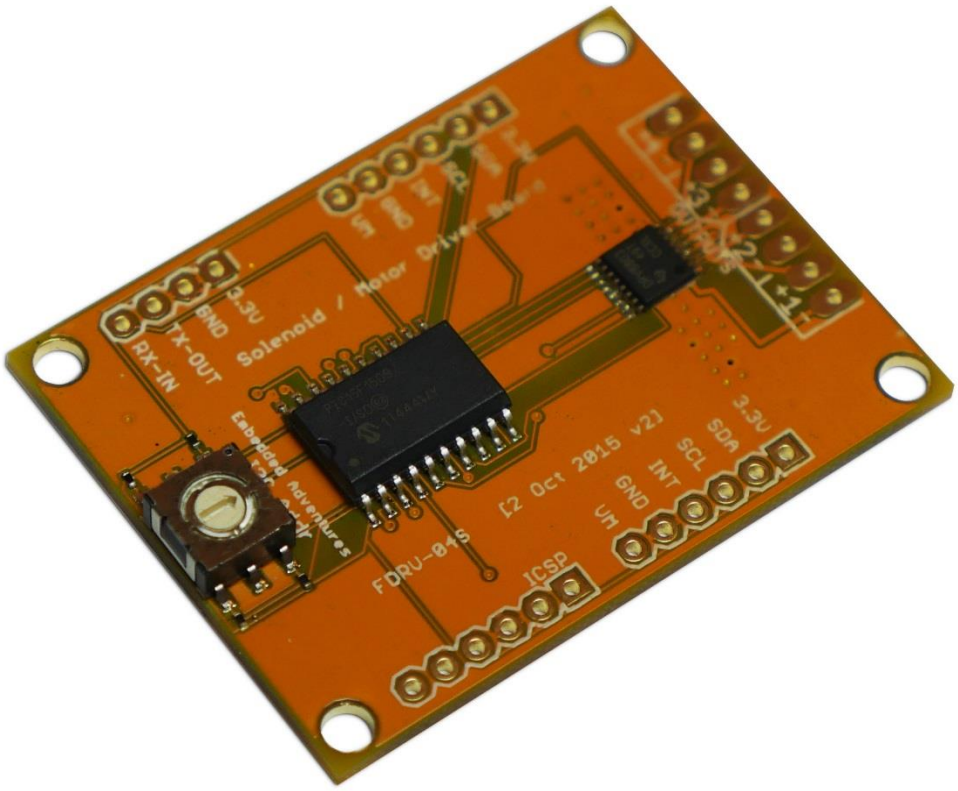
Device: FDRV-04S

This document version: v1

Matches module version: v2 [2 Oct 2015]

Document revision date: 9 November 2015

Description: I2C 4 Device Motor / Solenoid Driver Board



Contents

Introduction

The FDRV-04S is a 4 device high voltage solenoid and motor driver board.

Features

The FDRV-04S features the DRV8803 from TI. This does the actual driving of the motor or solenoid – it can handle up to 2 amps per device connected, and switch up to 60 volts! Behind that we've put a microcontroller that gives us 10 bit PWM control of each output, and some smart software to drive the device at different power levels over time. Each output can have up to three phases that are completely programmable.

For solenoids, they can burn out if held pulled in (ie, powered). Still, you may wish to have the solenoid stay pulled in – imagine the flipper on a pinball machine. The FDRV-04S allows this by having different power levels programmable for certain periods of time. For example, you could have phase 1 at 100% for 50ms and then phase 2 at 10% from then on. The driver board allows this with a simple **start** command and this happens automatically.

Of course, you can control other high voltage / current devices as well – for example 12V RGB LED strands etc, so long as the device can handle all the grounds being connected together. A device that has a +12V connection and R, G and B connections that you need to ground to light up the LEDs won't work. If it has a common ground connection instead, you're good to go.

Hackability

The FDRV-04S is 100% hackable.

At Embedded Adventures, we believe you have the most fun when you have the most control over your hardware. For the FDRV-04S we provide a datasheet, complete schematic and complete source code. After that, it's all up to you. We'd love to hear about the projects you're using it for – send us information and photos to myproject@embeddedadventures.com

That being said, this module has been developed as a standalone device that (all being well) will not require any firmware updates, at least until we get some great suggestions from our customers.

Construction

It's all pre-built! Just add female or male header pins, or solder directly to the board, and away you go.

When switching motors and solenoids, both of which have large coils involved, you will need to put diodes across the coil. The stripe of the diode goes toward the positive connection. As coils are disconnected, they generate a large voltage in the opposite direction – which the diodes will feed back in the coil, effectively damping them. This simple component saves all these nasty voltage spikes killing all the other electronics around it.

While the DRV8803 has some diodes included as part of the design, we include diodes with the module since we don't really trust the ones in the DRV8803. Besides, that means the current has to travel across possibly long wires and through the driver board itself. So, just use the diodes.

We have found that in some circumstances the microcontroller reboots itself if your power supply is not clean. This can be fixed by a 100n capacitor soldered across pins 1 and 20 of the microcontroller on the board. We'll include more bypass capacitors in the next revision of the board.

Connections

The FDRV-04S has a bunch of connection ports.

At the top of the board, you'll find the **output port** to connect your devices.

1	+	Device 1 positive connection
	-	Device 1 ground connection
2	+	Device 2 positive connection
	-	Device 2 ground connection
3	+	Device 3 positive connection
	-	Device 3 ground connection
4	+	Device 4 positive connection
	-	Device 4 ground connection

There are two identical **I2C ports**, so you can daisy chain driver board to control even more devices. The interrupt output will go high to alert you to an error condition, either over-current or over-temperature. This can be discovered by a simple 1 byte I2C read (which clears the interrupt pin).

3.3V	Logic supply. While these boards are designed to run at 3.3V, you can happily run the power and logic at 5V.
SDA	I2C Data
SCL	I2C Clock
INT	Interrupt
GND	Ground
VM	Power for solenoid / motor (8.2V – 60V)

The **ICSP port** is only used for initial programming of the microcontroller and shouldn't need to be used in real life

The **serial port** allows you to interact with the driver board to diagnose issues and debug what's happening.

3.3V	Logic supply
GND	Ground
TX-OUT	Serial debug output
RX-IN	Serial debug input

The serial port runs at 115,200. You can watch this to check what is happening with your

Don't forget to include a diode across the coil with the stripe of the diode towards the positive connection.

Power

The FDRV-04S logic can be powered from 3V to 5V. Just make sure you are interfacing the logic at the same voltage level as the power pin ("3.3V").

The VM pin provides power to the solenoid or motor. This can range from 8.2V to 60V. The DRV8803 module can handle 2 amps of current for any one controller, or 1 amp if driving all four controllers simultaneously.

Pull up resistors

I2C requires the use of pull-up resistors. Since the board is designed to be daisy chained, it does not include pull-up resistors. Many microcontrollers include their own weak pull-ups for I2C. If yours doesn't, you'll need pull-up resistors (4.7k to 10k) pulled up to the logic power level.

Using the driver board

Before use, the driver board needs to be configured. It needs this each time it is powered on.

To configure the board, each controller output you use needs a **setup** command.

The setup command allows you to set what power level should be used for each phase, how many phases are used and how long each phase goes for.

When a **start** command is received, the board will run through the phases as configured, and if there are no endless phases specified, will finish at 0% power level. The output will always idle at 0%.

So, a good example might be our flipper solenoid. We're going to specify two phases. The first phase will be 1023 (100% power), and for 50ms. The second phase will be 102 (10% power) and will run forever. For phases that run forever, simply specify 0xffff (65535).

When a **stop** command is received, the driver will return to an idle state and set the output to 0% power.

I2C protocol

The I2C address of the board is 0x10 + the value of the configuration wheel. So the addresses can be set to 0x10 up 0x1F.

Remember that in the seven bit world of I2C addresses, some microcontrollers specify it as 0x10 (Arduino) and some specify it as if the R/W bit were attached as bit 0 – meaning they specify the address as 0x20.

The **setup** command works as follows:

Byte	Example value	Meaning
0	0x01	Setup command
1	0x03	Controller (0x01 – 0x04)
2	0x02	Number of phases (0x01 – 0x03)
3	0x03	Phase 1 level (MSB)
4	0xFF	Phase 1 level (LSB)
5	0x00	Phase 2 level (MSB)
6	0x66	Phase 2 level (LSB)
7	0x00	Phase 3 level (MSB)
8	0x00	Phase 3 level (LSB)
9	0x00	Phase 1 length in ms (MSB)
10	0x32	Phase 1 length in ms (LSB)

11	0xFF	Phase 2 length in ms (MSB)
12	0xFF	Phase 2 length in ms (LSB)
13	0x00	Phase 3 length in ms (MSB)
14	0x00	Phase 3 length in ms (LSB)

Phase levels are specified 0x0000 (0% power) to 0x03FF (100% power).

Phase lengths are specified in milliseconds (1s = 1000ms). 0xFFFF has a special meaning in that the phase will run forever.

Each controller output must be configured before use.

Once configured, the **start** can be issued to fire the output.

Byte	Example value	Meaning
0	0x02	Start command
1	0x03	Controller (0x01 – 0x04)

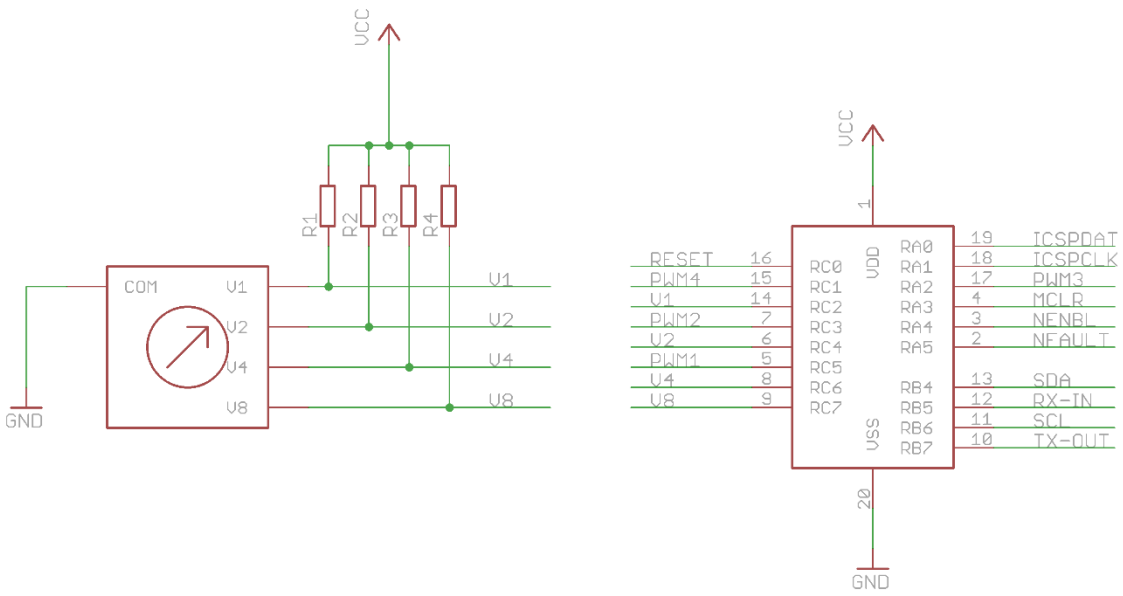
If you want the controller to return to idle (0%), the **stop** can be used. Note that if one of your phases is 0% forever (0xFFFF) then the controller output doesn't need to be stopped, it can simply be retriggered again with another **start** command.

Byte	Example value	Meaning
0	0x03	Stop command
1	0x03	Controller (0x01 – 0x04)

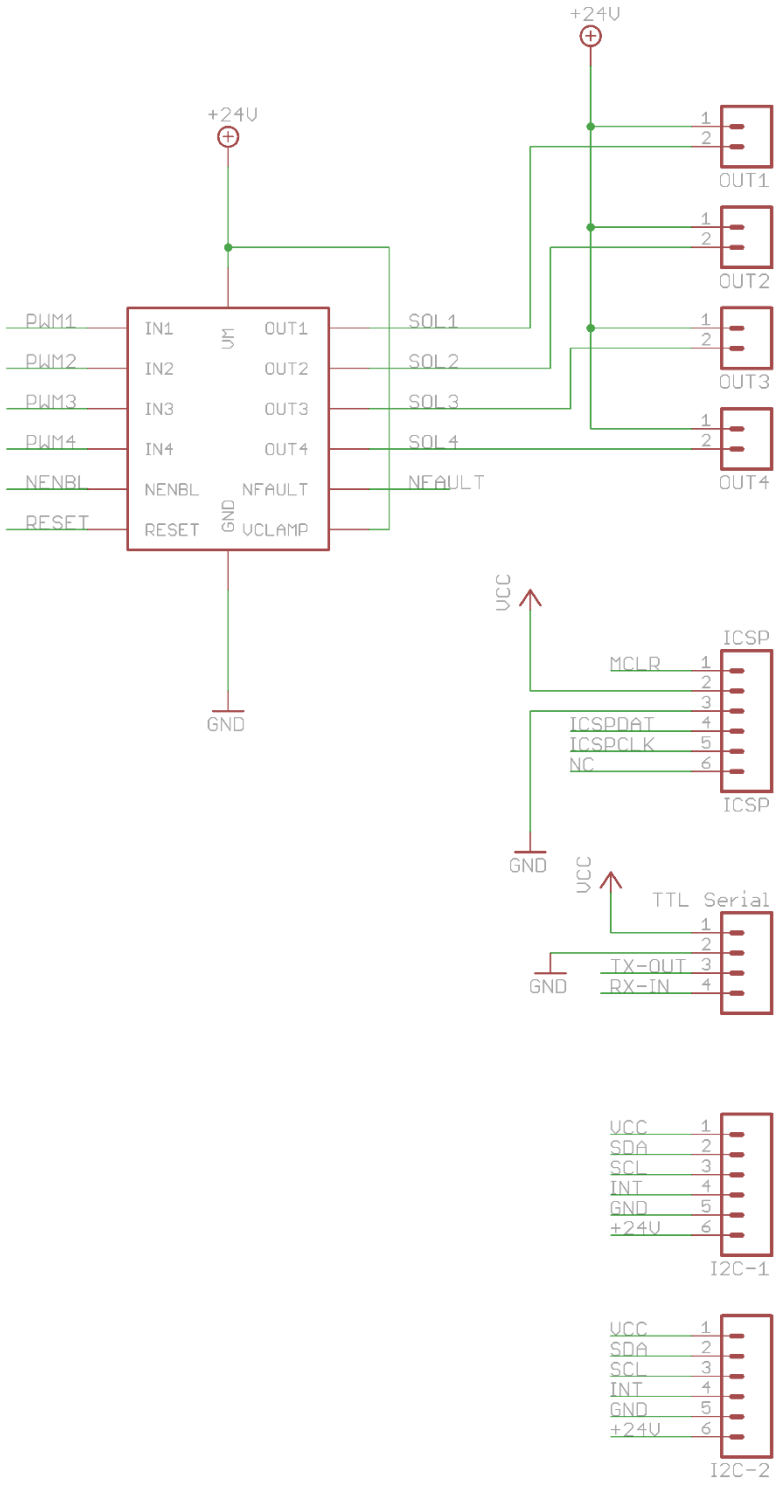
Tricks and traps

Arduino code is available for use, letting you get up and running quickly. It's also a good reference if you're using the driver board with a different microcontroller.

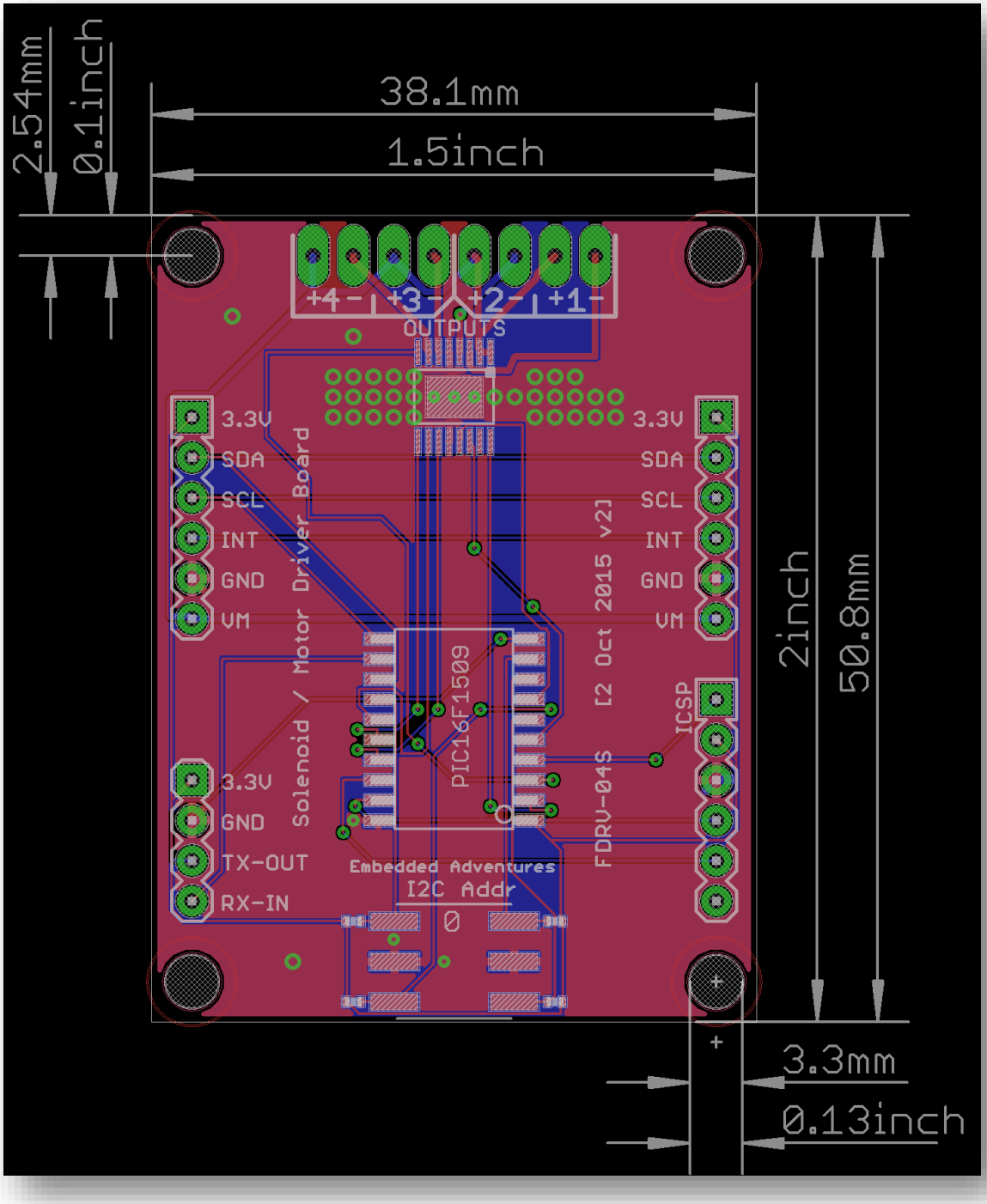
Schematic



The I2C Address Selector drives pull-up resistors low. A PIC16F1509 microntroller is used to output multiple PWM connections at the same time while also communicating over I2C and TTL Serial.


+24V
+24V
PWM1
PWM2
PWM3
PWM4
NENBL
RESET
IN1
IN2
IN3
IN4
NENBL
RESET
GND
VM
OUT1
OUT2
OUT3
OUT4
NFAULT
VCLAMP
SOL1
SOL2
SOL3
SOL4
NFAULT
GND
OUT1
OUT2
OUT3
OUT4
VCC
ICSP
MCLR
ICSPDAT
ICSPCLK
NC
ICSP
GND
VCC
TTL Serial
TX-OUT
RX-IN
GND
VCC
SDA
SCL
INT
GND
+24V
I2C-1
VCC
SDA
SCL
INT
GND
+24V
I2C-2

PCB



Versions

Version	Date	Comments
Version 1	9 Nov 2015	Initial Version for board v2