

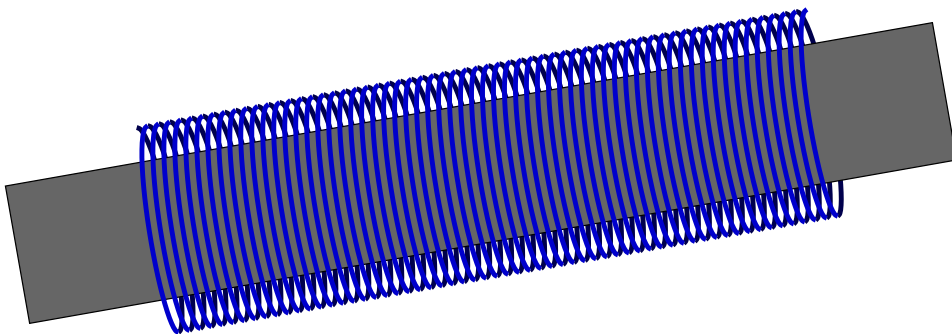
# D1

---

## Boost Converter Design Project: Implementation

---

In the D1 project you will build a power supply that boost 1.5V to up to 12 V. You will study the behaviour of an idealised boost circuit with a simulation. You will use the knowledge gained from the simulation study to develop a C-program that runs on I1 Matto and controls the output voltage through pulse-width-modulation. You will also develop a second C-program that runs on a host PC and interfaces with your embedded program on I1 Matto by serial communication. The user-interface of the hosted C-program will allow for setting a desired output voltage and for displaying the current output voltage. After building your computer-controlled power supply you will tune and evaluate its performance.



## Schedule

---

Preparation Time : See notes *D1: Modelling and Simulation*

Lab Time :  $5 \times 3$  hours

## Items provided

---

Tools :

Components : 100 mm  $\times$  10 mm Ferrite Rod, STP55NF06L MOS-FET 60V/55A, 1N5819 Schottky Diode 40V/1A,  $680\Omega$  3W 5% Resistor,  $150\Omega$  3W 5% Resistor,  $100\Omega$  3W 5% Resistor,  $0.22\Omega$  3W 5% Resistor,  $2 \times 22k\Omega$  0.25 W,  $2 \times 4.7k\Omega$  0.25 W,  $220\mu\text{F}$  63V Electrolyte Capacitor,  $2 \times$  Tactile Switch,  $\approx 4$  cm of 13 mm 3:1 Heatshrink tubing

Equipment : Multimeter, Oscilloscope[1]

Software : avr-gcc, ASCII text editor

## Items to bring

---

Il Matto

Identity card

Laboratory logbook

Toolkit

Version: January 20, 2014

©2012–2014


Klaus-Peter Zauner

Electronics and Computer Science

University of Southampton

Before entering the laboratory you should read through this document and complete the preparatory tasks detailed in section [2](#).

**Academic Integrity** – *If you wish you may undertake the preparation jointly with other students. If you do so it is important that you acknowledge this fact in your logbook. Similarly, you will probably want to use sources from the internet to help answer some of the questions. Again, record any sources in your logbook.*

You will undertake the exercise working with your laboratory partner. During the exercise you should use your logbook to record your observations, which you can refer to in the future – perhaps to write a formal report on the exercise, or to remind you about the procedures. As such it should be legible, and observations should be clearly referenced to the appropriate part of the exercise. As a guide the  symbol has been used to indicate a mandatory entry in your logbook. However, you should always record additional observations whenever something unexpected occurs, or when you discover something of interest.

For each Task you should create a new directory so that you have a working version of every program at the end of the lab. Remember to place comments in your code.

You will be marked using the standard laboratory marking scheme; at the beginning of the exercise one of the laboratory demonstrators will mark your preparatory work and at the end of the exercise you will be marked on your progress, understanding and logbook.

## Notation

This document uses the following conventions:



An entry should be made in your logbook



Care should be exercised

command input

Command to be entered at the command line

*Remarkable text*

A point of note

## 1 Introduction

This project builds on the various skills you have acquired in the exercises C1 to C11. You will build a computer controlled power supply based on a boost converter of the type shown in Figure 1.

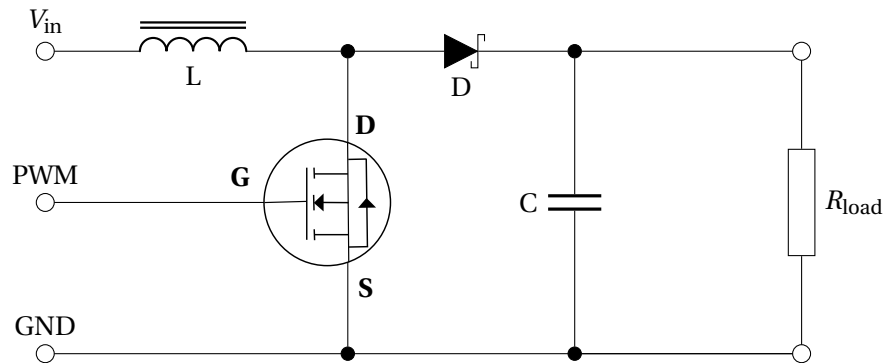


FIGURE 1: Boost circuit schematic with PWM controlled MOS-FET as switch

### 1.1 Outcomes

At the end of the exercise you should be able to:

- Understand the principles of switch-mode voltage conversion
- Design and implement an embedded system that is controlled by a PC<sup>1</sup> through serial communication
- Use a microcontroller to continuously monitor an analogue input signal and respond with appropriate output in real-time
- Implement a closed-loop control-scheme with a microcontroller and evaluate its performance

## 2 Preparation

The preparation for this exercise is described in a separate document: *D1: Modelling and Simulation*; your logbook will be marked in the first lab session according to this separate document. You are expected to read all of the present notes, but you are not required to make additional preparations for the work described here in your logbook.

*You will find a Project Completion Form on the notes page. Some milestones during this project have to be recorded on this form. The demonstrators have to sign off on your entries in the form before you leave the lab. At the assessment session the demonstrators will collect the form from you and fill in the main part of the form during the evaluation of your project.*

<sup>1</sup>Personal Computer

Logbooks will not be marked for the laboratory work. You are expected, however, to keep a record of your work in your logbook. Demonstrators may ask to see evidence for work claimed on the completion form that is not evident in your code (e.g., the tuning of the control circuit parameters), and then your logbook is the obvious place to turn to for such evidence.

## 3 Laboratory Work

### 3.1 Winding a Coil

Wind a coil on your ferrite rod<sup>2</sup>, using Figure 2 as a guide. The coil should be wound tight and all in one direction. Leave at least 5 mm space at the end of the rod and wind at least 60 turns. Leave 10 cm leads at both ends of the coil. You are provided with heat shrink tubing that can be slipped over the coil ends and heated with hot air to secure the windings in place.

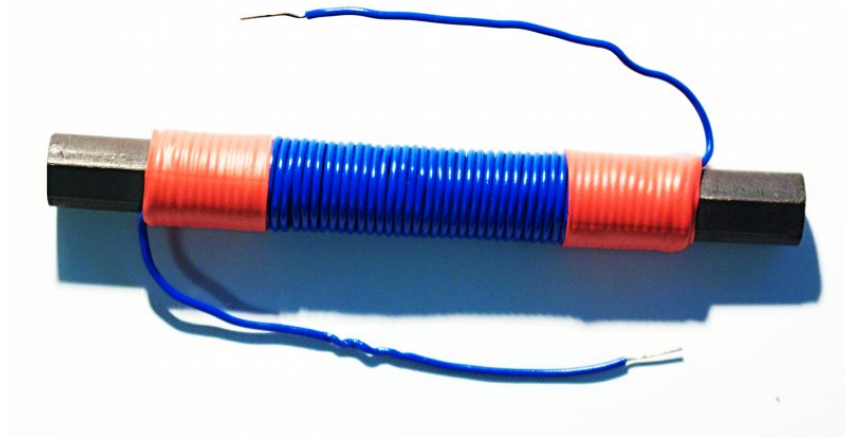


FIGURE 2: Hand-wound coil with 60 turns on 100 mm × 10 mm diameter ferrite rod. Try to avoid damage to the wire insulation (here visible in the foreground) when using the hot-air gun.

If you decide to wind more than one layer: make sure the ends of the coil are thin enough to fit in the heatshrink tube. You can wind the outer layer with a shorter length. Note in your logbook and on the project completion form how many turns you have wound and whether you wound more than one layer.

Use the inductance meter in the lab to measure the inductance of your coil and note the value in your logbook and on the Project Completion Form.

### 3.2 Circuit Construction

#### 3.2.1 Basic boost circuit

Assemble the circuit shown in Figure 3, but reserve space on the breadboard for the two push button switches that will be required later (see Figure 6). The diode is marked with a ring that indicates the

---

<sup>2</sup>Do not drop or knock it, it may break.

cathode. On the circuit diagram the cathode is the line on the tip of the arrow. Your circuit uses a Schottky diode because it can switch faster and there is less voltage drop across a Schottky diode than across a regular silicon diode. Please note that *the diode* you have *has a current limit of 1 A*. If you power the circuit with the bench top supply and short-circuit the output you can damage the Schottky diode<sup>3</sup>.

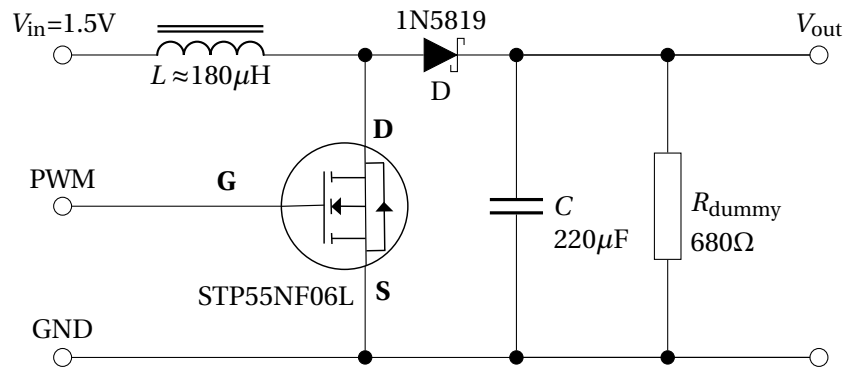


FIGURE 3: Basic circuit for first tests. The terminal PWM is driven by PD7 of the IlMatto board.

The MOS-FET is sensitive to electrostatics: touch a grounded surface (e.g. the metal case of an instrument or PC) before you handle it and then insert it in the bread-board. Do not handle it unnecessarily. The risk of electrostatic damage is much reduced when the MOS-FET is installed in the circuit.

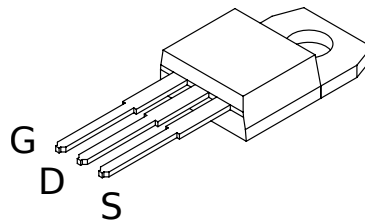


FIGURE 4: Pinout of the STB55NF06L MOS-FET

After you have convinced yourself that your circuit is correctly assembled, the first step is to test whether it can boost the input voltage. Set the bench-top power supply to 1.5 V and current limit to 200 mA. Connect the oscilloscope with a 10× probe  $V_{out}$ . Ground the PWM input (gate of the MOS-FET) of your circuit. Connect the bench-top power supply to your circuit<sup>4</sup>. What do you observe at the oscilloscope?

Compile the `embedded_boost.c` program and download it to the IlMatto board. Make sure you have your linker setup with the options for printf with floating point numbers as you did in C9. The compiler needs the options: `-Wl,-u,vfprintf -lprintf_flt -lm`

Connect GND of Boost circuit (see Figure 3) with IlMatto board. Connect PWM of boost circuit with Pin 7 on Port D of the IlMatto board.

Use the oscilloscope with a 10× probe to monitor the output voltage at  $V_{out}$ . Raise the current limit at the bench-top power supply until the output reaches 15V. Note the current limit in your logbook and on the project completion form.

<sup>3</sup>Diodes with a higher current limit do not fit in the holes of the breadboard.

<sup>4</sup>Do *not* use the Il Mat to board to power your boost circuit!

Do not raise the current limit beyond this for the following parts of this exercise—otherwise you may damage your IlMatto board.



### 3.2.2 Measuring output voltage

The next step is to use the IlMatto board not only for producing the PWM input to the boost converter, but also to monitor the output voltage delivered at the load.

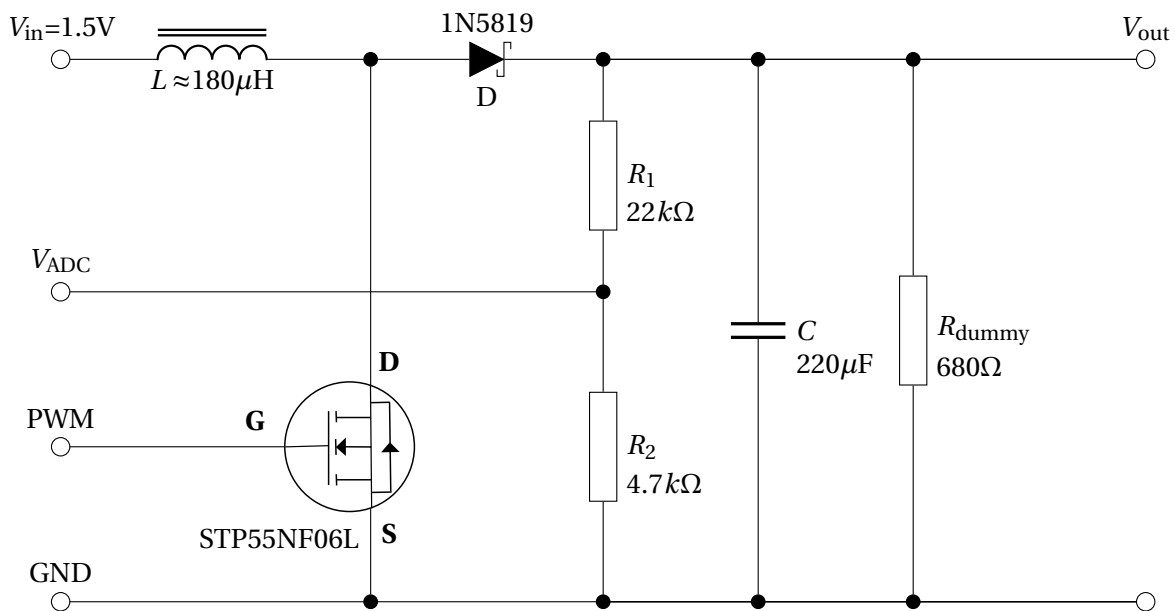


FIGURE 5: Circuit for test with ADC. The PWM terminal is driven by PD7 of the IlMatto board.

Extend your circuit as shown in Figure 5 to include a voltage divider made from  $R_1$  and  $R_2$ .

Connect the oscilloscope at  $V_{ADC}$ . What voltage do you measure? Is this voltage safe for your IlMatto board? Note voltage in your logbook and on the Project Completion Form.



Complete your circuit with the two switchable load resistors  $R_A$  and  $R_B$  as shown in Figure 6. Check which contacts on the switch are connected when it is depressed—your assumptions may be wrong.

You have now completed the circuit you need for this project. The remaining resistors can be used for optional extensions: either to use a second ADC channel to measure the voltage at the input or to measure the current flowing through the coil or the current flowing through the load. For current measurements, insert the  $0.22 \Omega$  resistor in series with the path in which you want measure the current. The use the ADC to measure the voltage across this resistor.

### 3.3 Circuit Testing

This final test should establish whether your circuit works and also provide you with some code that can be a starting point for your software development.

Connect PD0 on the IlMatto board with the USB serial cable signal TXD (orange) and PD1 on the IlMatto board with RXD (yellow) of the USB serial cable. Also connect GND (black).

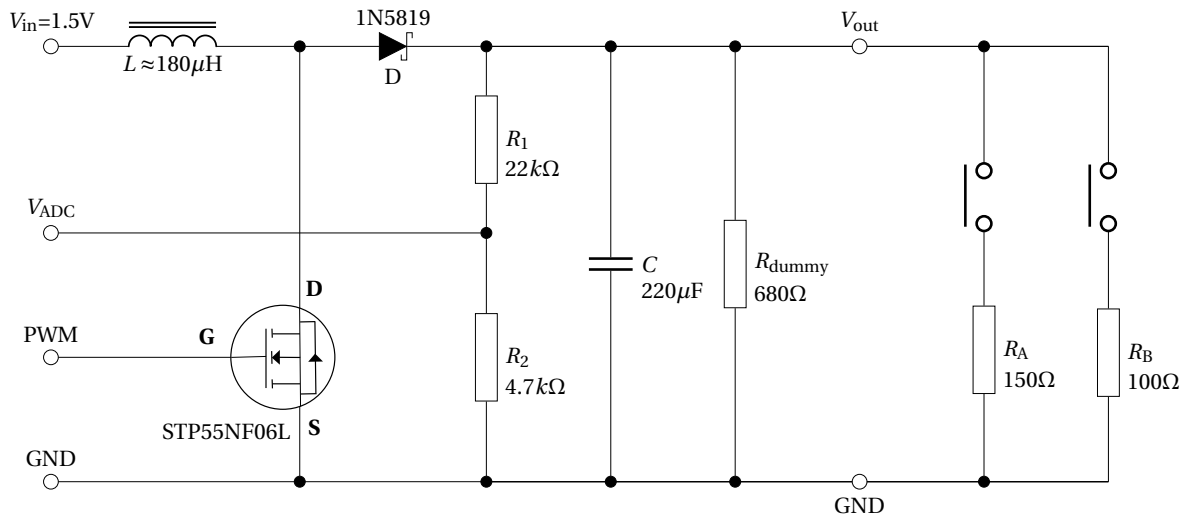


FIGURE 6: Final circuit for tests with varying load. For a high load on the output both load resistors  $R_1$  and  $R_2$  can be switched in parallel.

Open a serial terminal and set the terminal to 9600 baud, one stop bit, no parity, no flow-control. Connect the  $V_{ADC}$  terminal of the boost circuit to the ADC input at PA0 of the IIMatto board. You should now be able to see the voltage measured by the ADC of the IIMatto board as output on the serial terminal. Note the voltage that is shown in your logbook.

### 3.4 Communication Testing

Open hyper terminal or the device manager to find out on which COMPORT the USB cable is connected.<sup>5</sup> Close the terminal so that the COM port is released. Edit the source code of the host program `host_coms_boost.c` and change the `cport_nr` to the COMPORT number.<sup>6</sup>

Copy the files `host_coms_boost.c`, `rs232_d1.h`, `rs232_d1.c`, `timer.h`, and `timer.c` to a folder (or Eclipse project) and compile them into an executable for the host. See the header comment in `host_coms_boost.c` for the command line compilation instructions. You can do this on the command line or as an Eclipse project as you did in C1–C5<sup>7</sup>. Open a DOS terminal to run the compiled program from the command line. Look at the source of the program to see what it can do.

Compile `embedded_coms.c` for the IIMatto board and download it to the board. Test the communication between the IIMatto board and the host program. At the host program (DOS terminal) type “start”. Reset IIMatto, it should send a message. Then send commands (“high”, “low”) to IIMatto, it should respond to them.

<sup>5</sup>On Linux use: `ls -lh /dev/serial/by-id` to see the device name assigned to the FTDI cable.

<sup>6</sup>This is the case for the use of `rs232_d1.h/rs232_d1.c`, if the unmodified library `rs232.h/rs232.c` is used, then the `cport_nr` is an index into a look up table and not a comport number; see line 221 of `rs232.c`.

<sup>7</sup>In case you use Eclipse also for the AVR code, then do not forget that the host program must be a plain C project, not an AVR project.



### 3.5 Writing your own software

Finally compile `embedded_coms_boost.c` and download it to the I1 Matto board. With this in place you should be able to change the PWM output through the “high” and “low” commands. When it works you have all parts in place to get you started. The skeleton code is of course very simple—it is now up to you to improve it and to implement a control strategy. Your final code should allow one to set a desired output voltage at your host program (not a serial terminal!) and communicate this set point to the I1 Matto. The I1 Matto should report the current output voltage back to the user interface at the host. It should also keep the output voltage at the desired set point when the load at the power supply output changes. You want to achieve fast accurate control and a responsive user interface.

### References

- [1] Tektronix. Digital Storage Oscilloscope (TDS1000C). User Manual Rev. A, 2006. URL [https://secure.ecs.soton.ac.uk/notes/ellabs/reference/equipment/std-bench/Tektronix\\_TDS1000C\\_2000C\\_User\\_Manual.pdf](https://secure.ecs.soton.ac.uk/notes/ellabs/reference/equipment/std-bench/Tektronix_TDS1000C_2000C_User_Manual.pdf).