Semester project

Embedded control of a step-down DC-DC converter

Author:
Martin Specq

Supervised by:
Emilio Maddalena

Professor:
Colin Jones

June 12, 2020

Contents

1 Introduction

1.1 Overview

A step-down DC-DC converter is a power-electronics device that converts a high DC voltage into a low DC voltage. This is also called *Buck converter*. It can be found in many applications such as:

- Mobile chargers: For instance, a mobile phone has to be charged with a DC voltage of 5 V but the power source only provides a 12 V continuous voltage.
- Solar panels: The voltage provided by solar panels is constantly varying according to the ambient brightness. However, it is important for electronic devices to operate at a stable DC voltage. A buck converter would step down the voltage from the solar panel to a constant DC voltage.
- Motherboards: The operating voltage for a CPU is usually just above 1 V while the main voltages provided by the power supply is 3.3 V, 5 V and 12 V.

The efficiency of such converter is often higher than 90 %, which makes its use suitable for applications that require batteries.

1.2 Operation

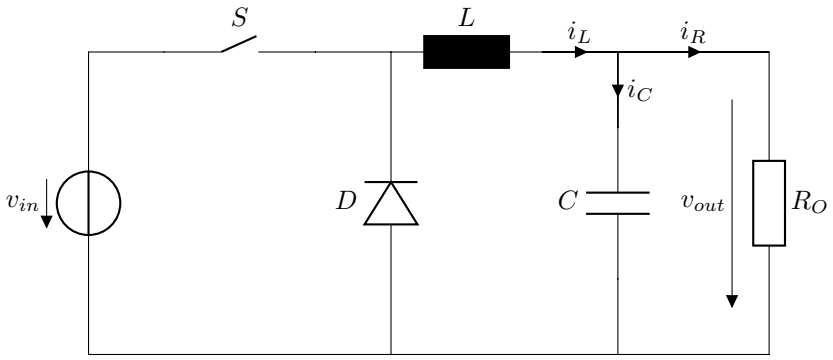


Figure 1: Simplified schematic of a buck converter.

When the switch S is closed, the current flows from the input source v_{in} to the load R_O through the inductor L. The current through the inductor i_L is increases and the output capacitor C charges itself so the output voltage v_{out} also increases. In parallel, the diode D is reverse polarized so no current flows through it. When the switch is open, a decreasing current flows through the diode, the inductor and the output load. The capacitor also discharges itself through the load. Therefore, the inductor current rises when the switch is on and drops when the switch is off. By switching the switch on and off rapidly, an equilibrium is reached. Thereby, the variation of the current through the inductor is equal for the rise and the drop. If the switching frequency is high enough, the current through the inductor never reaches 0. It is called continuous conduction mode (CCM).
In CCM, the output voltage can be theoretically calculated. Let T and D be respectively the period and the duty cycle of the PWM signal controlling the switch. Besides, let $\Delta i_{L,on}$ and $\Delta i_{L,off}$ be the variation of the inductor current when the switch is respectively on and off. It is assumed that the diode and the switch are ideal. Therefore, there is no voltage drop across them.
When the switch is closed:

$$v_{in} - v_{out} = L\frac{\Delta i_{L,on}}{\Delta t}$$

$$\Rightarrow v_{in} - v_{out} = L\frac{\Delta i_{L,on}}{DT} \tag{1}$$

$$\Rightarrow \Delta i_{L,on} = \frac{DT(v_{in}-v_{out})}{L}$$

When the switch is open:

$$\begin{aligned} 0 - v_{out} &= L\frac{\Delta i_{L,off}}{\Delta t} \\ \Rightarrow -v_{out} &= L\frac{\Delta i_{L,off}}{(1-D)T} \\ \Rightarrow \Delta i_{L,off} &= \frac{-(1-D)Tv_{out}}{L} \end{aligned} \tag{2}$$

At the equilibrium holds:

$$\begin{aligned} \Delta i_{L,off} + \Delta i_{L,on} &= 0 \\ \Rightarrow \frac{-(1-D)Tv_{out}}{L} &= -\frac{DT(v_{in}-v_{out})}{L} \\ \Rightarrow DTv_{out} - Tv_{out} &= -DTv_{in} + DTv_{out} \\ \Rightarrow v_{out} &= Dv_{in} \end{aligned} \tag{3}$$

It is clear that the output voltage increases linearly with the duty cycle. This is also interesting to notice that the theoretical efficiency of such converter is 100 % since there is not any resistive component except the output load. In the real world, electric components are not ideal and contain a parasitic resistance which makes the efficiency drop a little bit.
The equation derived previously only describes the system at the steady state, not during transients. Open loop transients generally present oscillations that can lead to high over/undershoots potentially dangerous for the hardware. These oscillations also induce a longer settling time.
Furthermore, equation 3 does not take into account the current drawn by the load with different R_O values which leads to steady state errors. Indeed, a load that draws more current will draw more charges from the output capacitor and thus lower the output voltage. The solution to this problem is to adjust the duty cycle with a feedback loop. This will maintain a constant output voltage no matter the load. To sum up, feedback control is needed. In the work presented here, the goal is to reach a smoother startup and load transient, less over/undershoot, less steady state error and shorter settling time than the open loop response has. Section 2 describes how the circuit is designed and built in order to make control possible. In order to perform model-based control, a mathematical model of the system is derived in section 3. In section 4, the open loop performance is presented and the response of the real system is compared to the one of the derived model. An integral controller is presented and analysed in section 5. Finally, an LQR is compared to a simplified explicit MPC controller using neural networks in section 6.

2 Circuit design

2.1 Specifications

The input is a 15 V DC power supply. The desired output voltage is 5 V. The microcontroller used to control the system is an STM32 nucleo L476RG board whose clock runs at 80 MHz. The sampling frequency is set to 10 kHz so that enough samples are retrieved to capture the dynamics without running out of computation time. The switching frequency is 20 kHz. This value is pretty low for such converter so it has to be considered by choosing the components. It allows to avoid problems that occur at high frequencies like parasitic inductive behaviour. Besides, the higher the frequency, the lower the counter period of the timer and the lower the resolution of the duty cycle. In compensation, the current ripple is pretty high if the switching frequency is low.
Furthermore, the circuit has to contain two output loads driven by a switch that selects either one or the other. Thus, it is possible to see how well the controller handles load switches.

2.2 Main components

This is very important that the system reacts slowly enough to step inputs. Indeed, the more points sampled during transients, the easier for the controller to capture all the dynamics. The state of the system is only known every sampling period and unknown otherwise. If the system reacts too fast compared to the sampling frequency, unwanted behaviour can happen without that the microcontroller even notices them. 10 samples during the rise is a reasonable value. Since the sampling period is 0.1 ms, the components has to be chosen such that the rise time is about 1 ms. The inductor and the output capacitor are the elements that store energy. The more energy they can store, the more time it takes to release this energy and the slower is the dynamics. Therefore, the inductor must have a sufficiently high inductance and the output capacitor a sufficiently high capacitance. The output load resistance has an impact on how much oscillations there are. A higher resistance will tend to induce more oscillations with a larger amplitude. The goal here is to choose a load inducing enough oscillations to make control challenging. Two resistors are chosen in order to use an on off switch that either connects one resistor or both resistors in parallel to the circuit. LTSpice simulations are run with different parameters in order to find the desired startup curve. Eventually, the following values are used:

- L = 10 mH
- C = 56 μF
- R_1 = 100 Ω
- R_2 = 100 Ω

Therefore, the output load is either $R_1 = 100\ \Omega$ or $R_1 // R_2 = \frac{1}{\frac{1}{R_1}+\frac{1}{R_2}} = 50\ \Omega$. Figure 2 shows the result of the simulation with the chosen values.

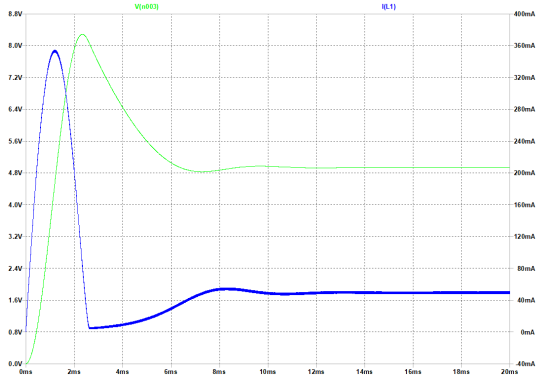


Figure 2: LTSpice simulation: Output voltage (green) and inductor current (blue) versus time during the startup. R_O = 100 Ω, D = 33.79 %.

The diode and the switch are the two remaining components to determine. They have to be as ideal as possible so their voltage drop has to be as low as possible and can change instantaneously. Thus, a Schottky diode is used since it has a volatge drop of about 0.1 V and can handle very large switching frequencies. An N channel MOSFET is chosen to act as a switch since its on-resistance and thus its voltage drop are usually very low.

2.3 Gate driver

A PWM signal is applied between the gate and the source of the MOSFET in order to switch the switch on and off. Whenever the gate source voltage V_{gs} goes from low to high, charges have to be supplied to the gate. This generates a current. However, if this current is too low, the rate at which V_{gs} increases is too slow. The microcontroller is neither capable of supplying enough voltage nor enough current to the gate. Indeed, in order to turn the MOSFET on, a gate source voltage higher than 5 V is needed. However, the maximum voltage that the microcontroller can supply is 3.3 V. In addition to that, it is important to isolate the control circuit from the power circuit because the microcontroller has sensitive components that could be damaged by the power circuit. Here, the gate driver is an optocoupler that takes the PWM signal from the microcontroller in input and outputs an amplified signal with the use of an external power supply. The external power supply can be the main one in addition to a bootstrap circuit. The later is formed by the ceramic capacitor C1, the electrolytic capacitor C2 and the diode D1 on figure 3. The diode has to be added externally so that connecting an external power supply between the terminals 8 and 5 of the optocoupler remains possible. In the latter case, the capacitors C1 and C2 act as bypass capacitors. Resistors R3 and R1 are current limiting resistors while R2 is a pull down resistor so that ghost voltages between the gate and the source are avoided.

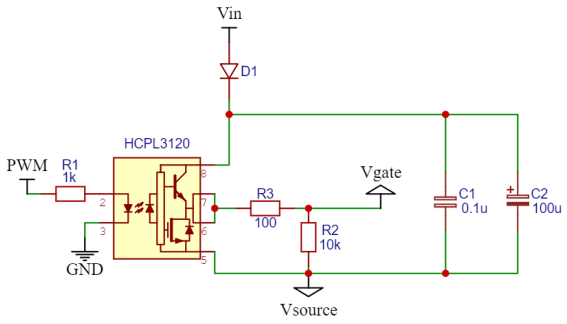


Figure 3: Schematic of the gate driving circuit including a bootstrap circuit.

2.4 Sensing

Measuring the output voltage is simple. A voltage divider is set between the output voltage and the ground in order to scale down the voltage. The divided voltage is then handled by the 12 bits ADC of the microcontroller. The ADC returns values between 0 (0 V) and 4095 (3.3 V). This portion of the circuit is presented on figure 4a. The ADC readings are a linear function of V_{OUT}:

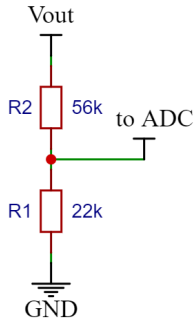
$$V_{ADC} = \frac{R1}{R1 + R2} V_{OUT} = 0.282\ V_{OUT} \tag{4}$$

This scaling factor still has to be adjusted at hand on the real system since the resistors R1 and R2 have an uncertain resistance.
Measuring the current is slightly more complicated. It could be estimated using an observer based on the measured voltage. However, the observer depends on the model that itself depends on the output load. Since it is not known a priori what load is connected at the output, the estimate is biased if the load is not the one used to derive the model. Therefore, a current sensor is integrated on the board. The measured currents do not exceed 400 mA and the maximum voltage that the ADC can handle is 3.3 V.

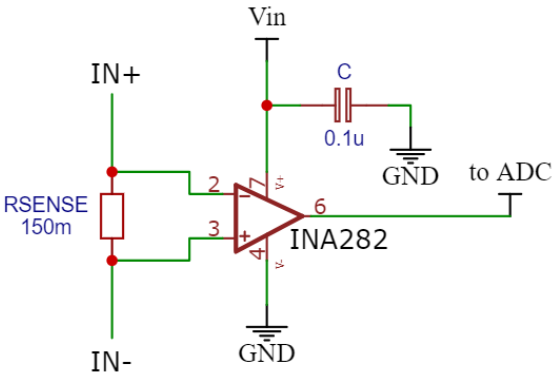
Hence, the sensitivity of the sensor should be ideally:

$$S = \frac{3.3}{0.4} = 8.25\ V/A \tag{5}$$

In order to have a safe margin, 7.5 V/A is chosen. To meet these requirements, a shunt resistor of 150 mΩ is connected in serie with the inductor. The voltage drop across the shunt resistor is then amplified by a current shunt monitor IC. The latter is the INA282 made by Texas instrument. Its gain is 50 V/V. The output of the IC is then connected to the 12 bits ADC of the microcontroller. The voltage drop across the shunt resistor does not exceed 15 mV at the steady state so it can be neglected. The cicuit is shown on figure 4b.



(a) Voltage divider scaling down the output voltage.



(b) Current sensing circuit.

Figure 4

2.5 PCB design

The schematic and the board was designed with Eagle PCB software.

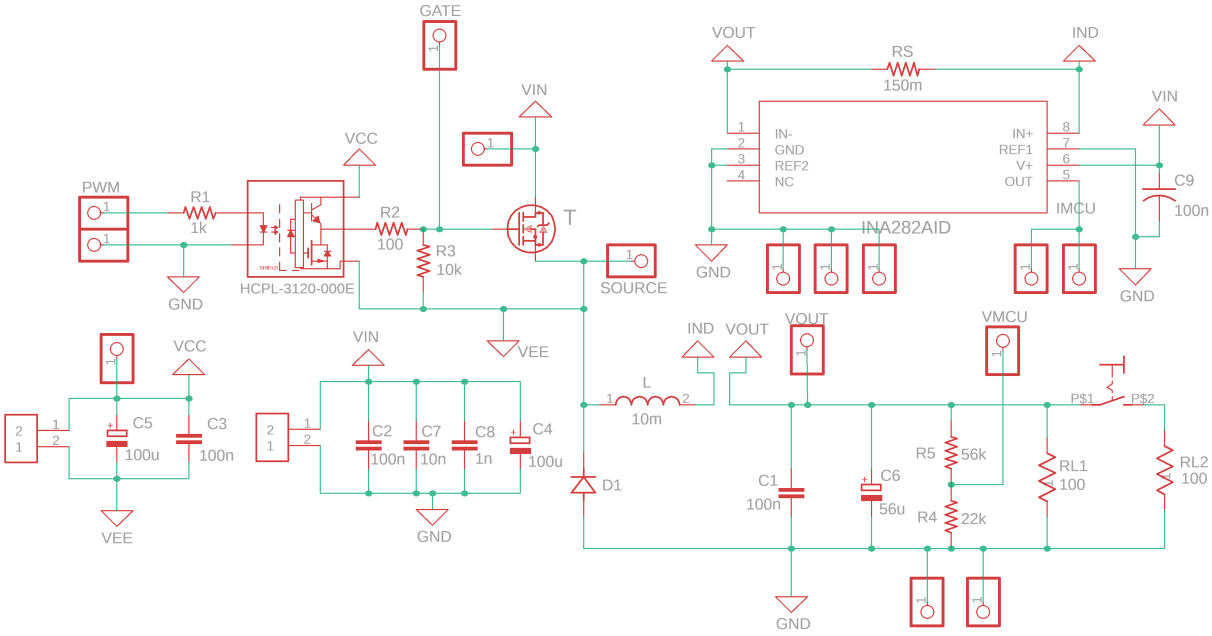


Figure 5: Schematic of the circuit made with Eagle.

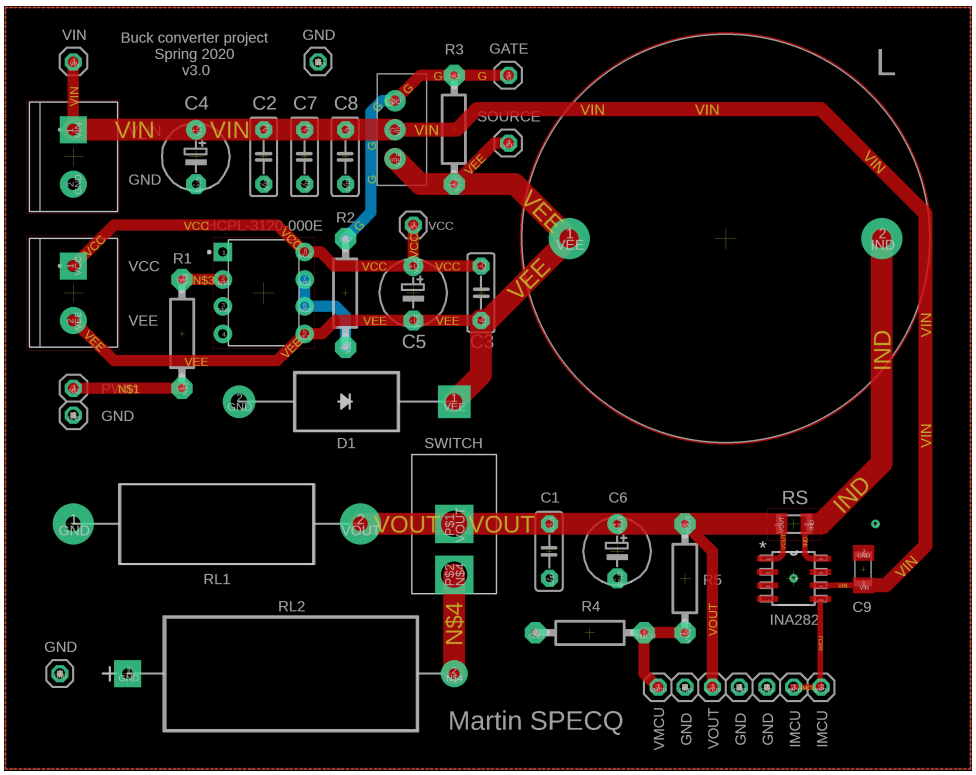


Figure 6: Board layout made with Eagle. The ground planes at the top and at the bottom are hidden.

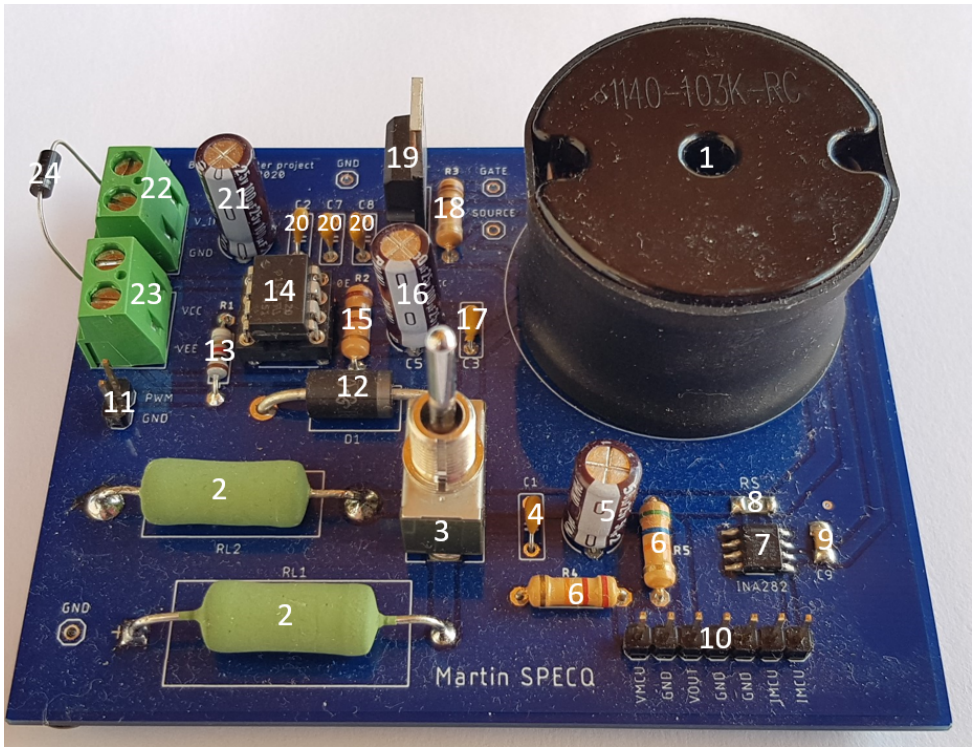


Figure 7: Picture of the final PCB. Each component is described in table 1.

Component	Description
1	Inductor
2	Load resistor
3	On Off switch
4	Output ceramic capacitor
5	Output electrolytic capacitor
6	Voltage divider resistor
7	Current shunt monitor IC
8	Shunt resistor
9	Bypass SMD ceramic capacitor
10	Output pins
11	PWM pins
12	Schottky diode
13	Current limiting resistor of the PWM input
14	Optocoupler
15	Gate current limiting resistor
16	Bootstrap electrolytic capacitor
17	Bootstrap ceramic capacitor
18	Pull down resistor between the gate and the source
19	N channel MOSFET
20	Input ceramic bypass capacitor
21	Input electrolytic buffer capacitor
22	Input voltage connector
23	Gate driver's supply voltage connector
24	Bootstrap circuit's diode

Table 1: Description of the components.

3 System dynamics

In order to perform model-based control, a model of the form $\dot{x} = Ax + Bu$ has to be established. In this case, u is the duty cycle. The state components are related to parts that are able to store energy. In this case, the output capacitor stores energy in the form of electrical charges and the inductor in the form of a magnetic field directly related to the current flowing through it. Thus, the magnitude of the current flowing through the inductor and the magnitude of the voltage across the output capacitor are both components of the state vector. Through this section, the system dynamics is derived with this state representation of the system.

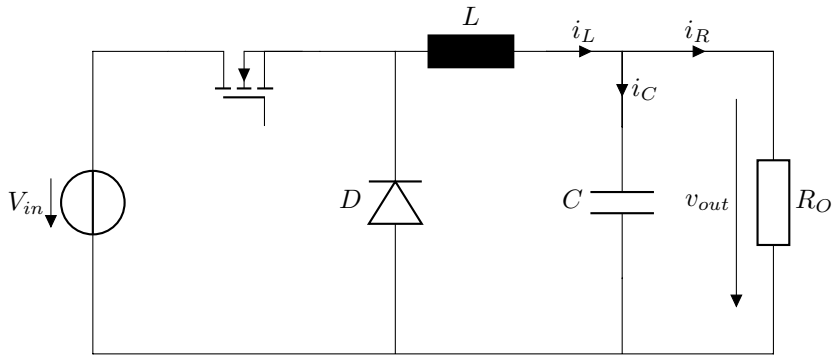


Figure 8: Simplified schematic of a buck converter.

The system has 2 states:

$$\begin{aligned} x_1 &= i_L \\ x_2 &= v_{out} \end{aligned} \tag{6}$$

The input u of the system is the duty cycle of the PWM signal applied between the gate and the source of the MOSFET. The schematic depicted above is not perfect because it only considers ideal components. In order to stick faithful to reality, some parasitic components are added:

1. $R_L = 2\ \Omega$: Resistance in serie with the inductor
2. $R_C = 330\ m\Omega$: Resistance in serie with the capacitor
3. $R_{on} = 5\ m\Omega$: Resistance between the drain and the source of the MOSFET
4. $V_j = 100$ mV: Voltage drop across the diode when it is direct polarized

The system operates in continuous conduction mode. That means that the current through the inductor never reaches 0. Thus, there are 2 operating states during one switching period:

1. The switch is closed: the switching node is connected to the input voltage
2. The switch is open: the switching node is connected to (ground - voltage across the diode)

3.1 Closed switch

When the switch is closed, the diode is reverse polarized and can be replaced by an open circuit. A schematic of the equivalent circuit is shown on figure 9.

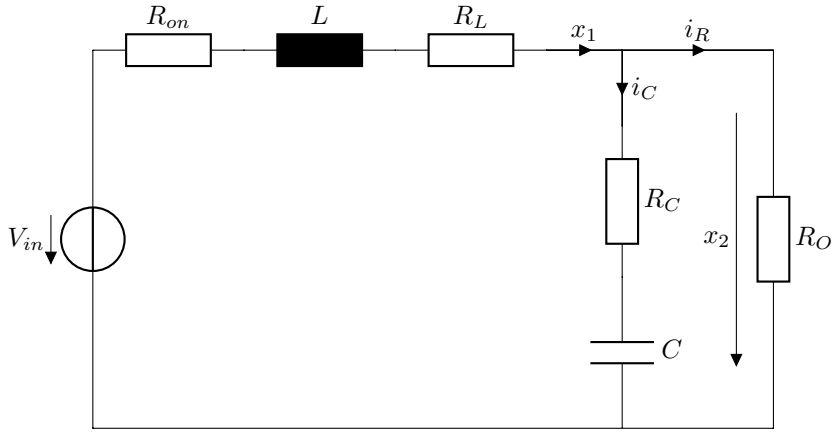


Figure 9: Closed switch: Equivalent schematic.

For x_1, one has:

$$\begin{aligned} V_{in} - x_2 &= (R_{on} + R_L)x_1 + L\dot{x}_1 \\ \Rightarrow \dot{x}_1 &= -\tfrac{R_{on}+R_L}{L}x_1 - \tfrac{1}{L}x_2 + \tfrac{V_{in}}{L} \end{aligned} \tag{7}$$

For x_2, one has:

$$\begin{aligned} x_2 &= R_C i_c + \tfrac{1}{C}\int i_c dt \\ \Rightarrow \dot{x}_2 &= R_C \dot{i}_C + \tfrac{1}{C} i_c \\ \Rightarrow \dot{x}_2 &= R_C(\dot{x}_1 - \dot{i}_R) + \tfrac{1}{C}(x_1 - i_R) \\ \Rightarrow \dot{x}_2 &= R_C\big(\dot{x}_1 - \tfrac{1}{R_O}\dot{x}_2\big) + \tfrac{1}{C}(x_1 - \tfrac{1}{R_O}x_2) \\ \Rightarrow \dot{x}_2 &= R_C\dot{x}_1 - \tfrac{R_C}{R_O}\dot{x}_2 + \tfrac{1}{C}x_1 - \tfrac{1}{R_O C}x_2 \\ \Rightarrow \tfrac{R_O+R_C}{R_O}\dot{x}_2 &= R_C\dot{x}_1 + \tfrac{1}{C}x_1 - \tfrac{1}{R_O C}x_2 \\ \Rightarrow \dot{x}_2 &= \tfrac{R_C R_O}{R_C+R_O}\dot{x}_1 + \tfrac{R_O}{(R_C+R_O)C}x_1 - \tfrac{1}{(R_C+R_O)C}x_2 \end{aligned} \tag{8}$$

By using equation 7 in equation 8, one gets:

$$\begin{aligned} \dot{x}_2 &= \tfrac{R_C R_O}{R_C+R_O}\Big(-\tfrac{R_{on}+R_L}{L}x_1 - \tfrac{1}{L}x_2 + \tfrac{V_{in}}{L}\Big) + \tfrac{R_O}{(R_C+R_O)C}x_1 - \tfrac{1}{(R_C+R_O)C}x_2 \\ \Rightarrow \dot{x}_2 &= \Big(-\tfrac{R_C R_O(R_{on}+R_L)}{(R_C+R_O)L} + \tfrac{R_O}{(R_C+R_O)C}\Big)x_1 - \Big(\tfrac{R_C R_O}{(R_C+R_O)L} + \tfrac{1}{(R_C+R_O)C}\Big)x_2 + \tfrac{R_C R_O V_{in}}{(R_C+R_O)L} \\ \Rightarrow \dot{x}_2 &= \tfrac{-R_C R_O(R_{on}+R_L)C + R_O L}{(R_C+R_O)LC}x_1 - \tfrac{R_C R_O C + L}{(R_C+R_O)LC}x_2 + \tfrac{R_C R_O V_{in}}{(R_C+R_O)L} \end{aligned} \tag{9}$$

3.2 Open switch

When the switch is open, the diode is direct polarized so the circuit can be simplified like on figure 10:
For x_1, one has:

$$\begin{aligned} -V_j - x_2 &= L\dot{x}_1 + R_L x_1 \\ \Rightarrow \dot{x}_1 &= -\tfrac{R_L}{L}x_1 - \tfrac{1}{L}x_2 - \tfrac{V_j}{L} \end{aligned} \tag{10}$$

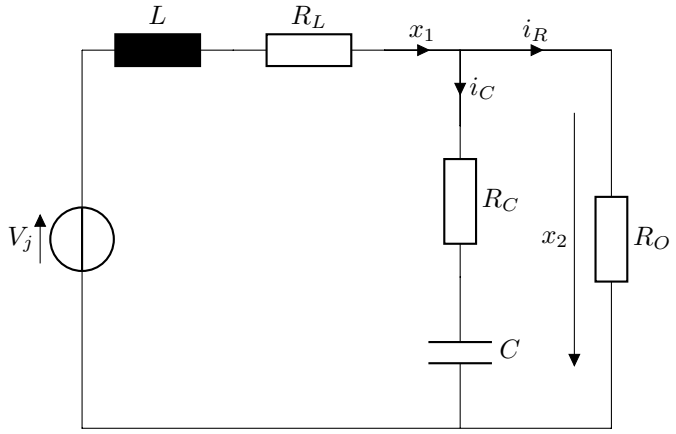


Figure 10: Open switch: Equivalent circuit.

For x_2, the equation is still:

$$\begin{aligned}
&\Rightarrow \dot{x}_2 = \tfrac{R_C R_O}{R_C+R_O}\dot{x}_1 + \tfrac{R_O}{(R_C+R_O)C}x_1 - \tfrac{1}{(R_C+R_O)C}x_2 \\
&\Rightarrow \dot{x}_2 = \tfrac{R_C R_O}{R_C+R_O}\Big(-\tfrac{R_L}{L}x_1 - \tfrac{1}{L}x_2 - \tfrac{V_j}{L}\Big) + \tfrac{R_O}{(R_C+R_O)C}x_1 - \tfrac{1}{(R_C+R_O)C}x_2 \\
&\Rightarrow \dot{x}_2 = \Big(-\tfrac{R_C R_O R_L}{(R_C+R_O)L} + \tfrac{R_O}{(R_C+R_O)C}\Big)x_1 - \Big(\tfrac{R_C R_O}{(R_C+R_O)L} + \tfrac{1}{(R_C+R_O)C}\Big)x_2 - \tfrac{R_C R_O V_j}{(R_C+R_O)L} \\
&\Rightarrow \dot{x}_2 = \tfrac{-R_C R_O R_L C + R_O L}{(R_C+R_O)LC}x_1 - \tfrac{R_C R_O C + L}{(R_C+R_O)LC}x_2 - \tfrac{R_C R_O V_j}{(R_C+R_O)L}
\end{aligned} \tag{11}$$

3.3 Combination

The input of the system is the duty cycle u. For deriving the dynamics, the average behaviour of $\dot{x}_1$ and $\dot{x}_2$ is used. Thus, one has:

$$\begin{aligned}
&\dot{x}_1 = \dot{x}_1(\text{closed})u + \dot{x}_1(\text{open})(1-u) \\
&\Rightarrow \dot{x}_1 = u\Big(-\tfrac{R_{on}+R_L}{L}x_1 - \tfrac{1}{L}x_2 + \tfrac{V_{in}}{L}\Big) + (1-u)\Big(-\tfrac{R_L}{L}x_1 - \tfrac{1}{L}x_2 - \tfrac{V_j}{L}\Big) \\
&\Rightarrow \dot{x}_1 = -\tfrac{R_L}{L}x_1 - \tfrac{1}{L}x_2 - \tfrac{R_{on}}{L}x_1 u + \tfrac{V_{in}+V_j}{L}u - \tfrac{V_j}{L}
\end{aligned} \tag{12}$$

$$\dot{x}_2 = \dot{x}_2(\text{closed})u + \dot{x}_2(\text{open})(1-u)$$

$$\Rightarrow \dot{x}_2 = u\left(\frac{-R_C R_O(R_{on}+R_L)C+R_O L}{(R_C+R_O)LC}x_1 - \frac{R_C R_O C+L}{(R_C+R_O)LC}x_2 + \frac{R_C R_O V_{in}}{(R_C+R_O)L}\right) +$$

$$(1-u)\left(\frac{-R_C R_O R_L C+R_O L}{(R_C+R_O)LC}x_1 - \frac{R_C R_O C+L}{(R_C+R_O)LC}x_2 - \frac{R_C R_O V_j}{(R_C+R_O)L}\right) \tag{13}$$

$$\Rightarrow \dot{x}_2 = u\Big(\frac{-R_C R_O(R_{on}+R_L)C+R_O L}{(R_C+R_O)LC} - \frac{-R_C R_O R_L C+R_O L}{(R_C+R_O)LC}\Big)x_1 + u\Big(-\frac{R_C R_O C+L}{(R_C+R_O)LC} + \frac{R_C R_O C+L}{(R_C+R_O)LC}\Big)x_2$$

$$+u\Big(\frac{R_C R_O V_{in}}{(R_C+R_O)L} + \frac{R_C R_O V_j}{(R_C+R_O)L}\Big) + \frac{-R_C R_O R_L C+R_O L}{(R_C+R_O)LC}x_1 - \frac{R_C R_O C+L}{(R_C+R_O)LC}x_2 - \frac{R_C R_O V_j}{(R_C+R_O)L}$$

$$\Rightarrow \dot{x}_2 = \frac{-R_C R_O R_L C+R_O L}{(R_C+R_O)LC}x_1 - \frac{R_C R_O C+L}{(R_C+R_O)LC}x_2 + \frac{R_C R_O(V_{in}+V_j)}{(R_C+R_O)L}u - \frac{R_C R_O R_{on}}{(R_C+R_O)L}x_1 u - \frac{R_C R_O V_j}{(R_C+R_O)L}$$

3.4 Linearization

The above equations show that there are bilinearities in the system. That means that there is at least one term where two system variables are multiplied together (e.g. $x_1 u$). Besides, the system is affine because $\dot{x}$ depends on at least one term that does not contain any system variable. At this point holds:

$$\dot{x} = \begin{pmatrix}\dot{x}_1\\ \dot{x}_2\end{pmatrix} = f(x_1, x_2, u) = \begin{pmatrix}f_1(x_1, x_2, u)\\ f_2(x_1, x_2, u)\end{pmatrix} \tag{14}$$

f is affine and contains bilinearities. If it is linearized around the equilibrium point using the first order Taylor serie, one obtains:

$$\dot{x} = f(x,u) \approx f(x_{eq}, u_{eq}) + J_x^f(x_{eq}, u_{eq})(x - x_{eq}) + J_u^f(x_{eq}, u_{eq})(u - u_{eq}) \tag{15}$$

Where J_x^f and J_u^f are respectively the jacobians of f with respect to x and u.
Since $\dot{x}_{eq} = f(x_{eq}, u_{eq}) = 0$, one has:

$$\begin{aligned}\dot{x} &= f(x,u)\\ &\approx A(x - x_{eq}) + B(u - u_{eq})\\ &= Ax + Bu - (Ax_{eq} + Bu_{eq})\\ &\approx Ax + Bu - \dot{x}_{eq}\\ &= Ax + Bu\end{aligned} \tag{16}$$

Where $A = J_x^f(x_{eq}, u_{eq})$ and $B = J_u^f(x_{eq}, u_{eq})$.

The affine term has disappeared and the equation above is linear.

Then, the steady state values are computed. The system has to be linearized about the equilibrium point. Therefore, $\dot{x}_1$ and $\dot{x}_2$ are set to zero:

$$\begin{aligned}0 &= -\frac{R_L}{L}x_{1eq} - \frac{1}{L}x_{2eq} - \frac{R_{on}}{L}x_{1eq}u + \frac{V_{in}+V_j}{L}u_{eq} - \frac{V_j}{L}\\ 0 &= \frac{-R_C R_O R_L C+R_O L}{(R_C+R_O)LC}x_{1eq} - \frac{R_C R_O C+L}{(R_C+R_O)LC}x_{2eq} + \frac{R_C R_O(V_{in}+V_j)}{(R_C+R_O)L}u_{eq} - \frac{R_C R_O R_{on}}{(R_C+R_O)L}x_{1eq}u_{eq} - \frac{R_C R_O V_j}{(R_C+R_O)L}\end{aligned} \tag{17}$$

The desired output voltage is $x_{2eq} = 5$ V so there are 2 unknowns and 2 equations. After solving the equations, one gets:

$$\begin{aligned}x_{1eq} &= \frac{1}{R_O}x_{2eq}\\ u_{eq} &= \frac{R_O V_j + (R_L+R_O)x_{2eq}}{R_O(V_{in}+V_j) - R_{on}x_{2eq}}\end{aligned} \tag{18}$$

Then, the matrices A and B are computed:

$$A = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x_{eq}, u_{eq}) & \frac{\partial f_1}{\partial x_2}(x_{eq}, u_{eq}) \\ \frac{\partial f_2}{\partial x_1}(x_{eq}, u_{eq}) & \frac{\partial f_2}{\partial x_2}(x_{eq}, u_{eq}) \end{pmatrix} = \begin{pmatrix} -\frac{R_L + R_{on} u_{eq}}{L} & -\frac{1}{L} \\ \frac{-R_C R_O R_L C + R_O L - R_C R_O R_{on} C u_{eq}}{(R_C + R_O) LC} & -\frac{R_C R_O C + L}{(R_C + R_O) LC} \end{pmatrix}$$

and: $B = \begin{pmatrix} \frac{\partial f_1}{\partial u}(x_{eq}, u_{eq}) \\ \frac{\partial f_2}{\partial u}(x_{eq}, u_{eq}) \end{pmatrix} = \begin{pmatrix} \frac{V_{in} + V_j - R_{on} x_{1eq}}{L} \\ \frac{R_C R_O (V_{in} + V_j - R_{on} x_{1eq})}{(R_C + R_O) L} \end{pmatrix}$

The state and input values at the equilibrium have been computed using $f(x_{eq}, u_{eq}) = 0$. The linearized version of f is slightly different compared to f. Thus, the steady state values of the linearized model are slightly different, that means that $Ax_{eq} + Bu_{eq} \neq 0$. In order to find the new steady state values, the following system of equations is solved:

$$\begin{pmatrix} A & B \\ C & 0 \end{pmatrix} \begin{pmatrix} x_s \\ u_s \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \cdot \text{ref} \tag{19}$$

Where ref = 5 V is the desired output voltage. With a 100 Ω load resistance, one gets:

$$x_s = \begin{pmatrix} 0.05 \\ 5 \end{pmatrix} \tag{20}$$

and :

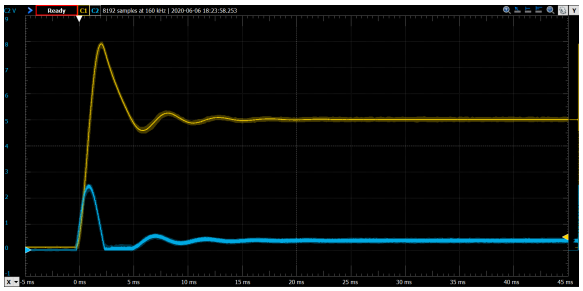
$$u_s = 0.3379 \tag{21}$$

Finally, the state space system is discretized using the zero-order-hold method.

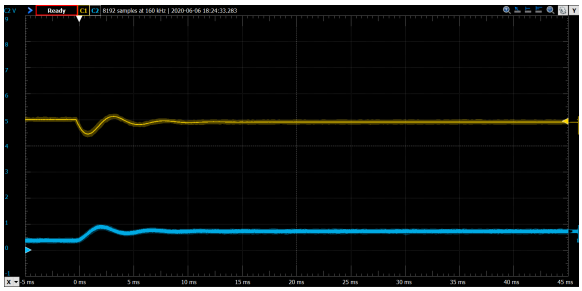
4 Open loop response

The open loop response is obtained by applying a step at the input. The magnitude of the step is the theoretical steady state input previously computed in section 3. All along this work, the performance of the system is evaluated according to the following criterions:

- Settling time: Time it takes for the error $|y(t) - y_{final}|$ between the response y(t) and the steady state response y_{final} to fall to within 2 % of y_{final}.
- Rise time: Time it takes for the response to rise from 10 % to 90 % of the steady state response.
- Undershoot: Once the response has risen, percentage error $|y_{min} - y_{final}|$ between the minimum value of the response y_{min} and the steady state response y_{final}, relative to y_{final}.
- Overshoot: Once the response has risen, percentage error $|y_{max} - y_{final}|$ between the maximum value of the response y_{max} and the steady state response y_{final}, relative to y_{final}.
- Steady state error: Error $|ref - y_{final}|$ between the final value y_{final} and the desired value ref



(a) Startup transient, $R_O = 100\ \Omega$.



(b) Load switch from $R_O = 100\ \Omega$ to $R_O = 50\ \Omega$.

Figure 11: Output voltage versus time during the open loop transients. Yellow (1 V/div): Output voltage, Blue (1 V/div): Current sensor output.

	Startup	Switch 100 Ω to 50 Ω	Unit
Settling time	11.3	4.41	ms
Rise time	0.883		ms
Maximum over/undershoot	58	9.3	%
Steady state error	0	0.07	V

Table 2: Open loop performance.

The characteristics of the open behaviour are given in table 2. One important thing to notice is the steady state error that occurs after a load switch. Indeed, it demonstrates that equation 3 of the introduction is only an approximation of the output voltage. Therefore, feedback control is necessary if a precise output voltage is desired. Moreover, it can be seen on figure 11 that the open loop response presents a high overshoot and oscillations. It could be qualified as a "bad" open loop response which makes further control challenging.
In order to see whether the derived model is accurate, the simulated open loop response is compared to the measured one. The model allows to have negative state values while this is not possible in reality. This introduces a non-linearity into the model and therefore, negative state values have to be avoided for a model-based control perspective. Since the model gives negative states values for the open loop response, the values are set to 0 if they happen to be negative. By doing this, a good fit of the real open loop response can be achieved showing that the model can be used with a lot of confidence. It can be seen on figure 12 that the model fits the actual open loop response pretty well.

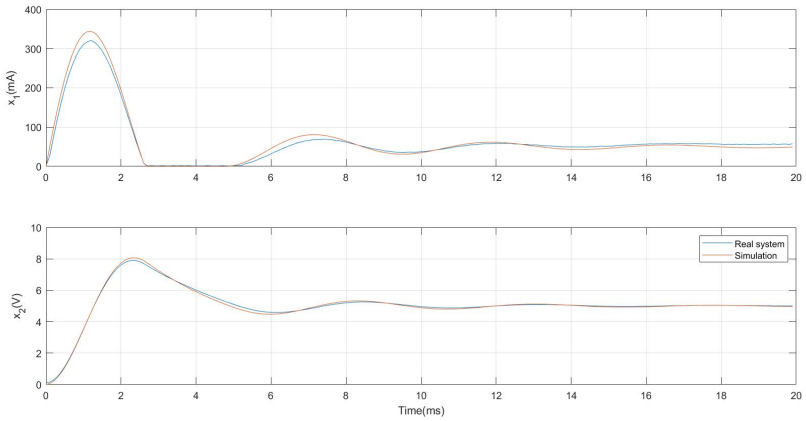


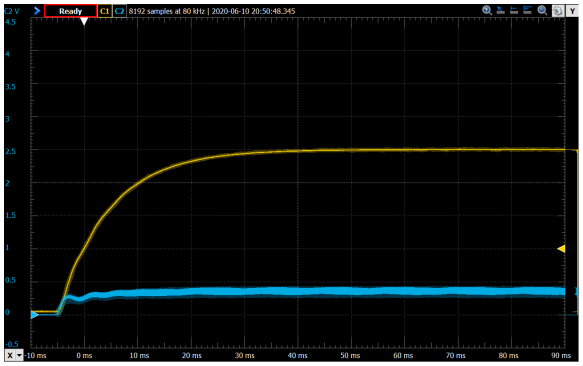
Figure 12: Comparison between the simulated and the measured open loop response.

5 Integral controller

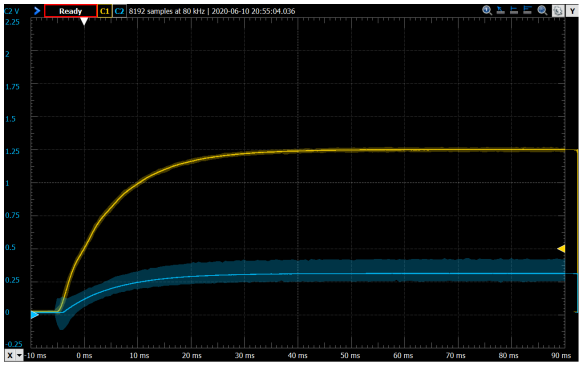
One of the most simple way to control the output voltage is to use a PID controller. The parameters are tuned at hand through an iterative process. Eventually, it turns out that the derivative and the proportional terms are not useful in this case so only an integrator is used. The configuration of the microcontroller is explained later in section 6 table 4. The current sensor output does not have to be connected.
The performance of the integral controller is presented in table 3.

	Startup	Switch 100 Ω to 50 Ω	Unit
Settling time	37.8	6.63	ms
Rise time	20.7		ms
Maximum over/undershoot	0	9.1	%
Steady state error	0	0	V

Table 3: Integrator performance.



(a) Yellow (1 V/div): Output voltage,
Blue (0.5 V/div): Current sensor output.



(b) Yellow (1 V/div): Output voltage,
Blue (0.25 V/div): Duty cycle.

Figure 13: Integrator: Startup transient scope screenshots.

It can be seen on figure 13 that the startup response is smooth without overshoot, oscillations or steady state error. However, the settling time is almost 4 times longer than the open loop one.
Figure 14 shows the response during a load switch. There is a high similarity between the open loop load

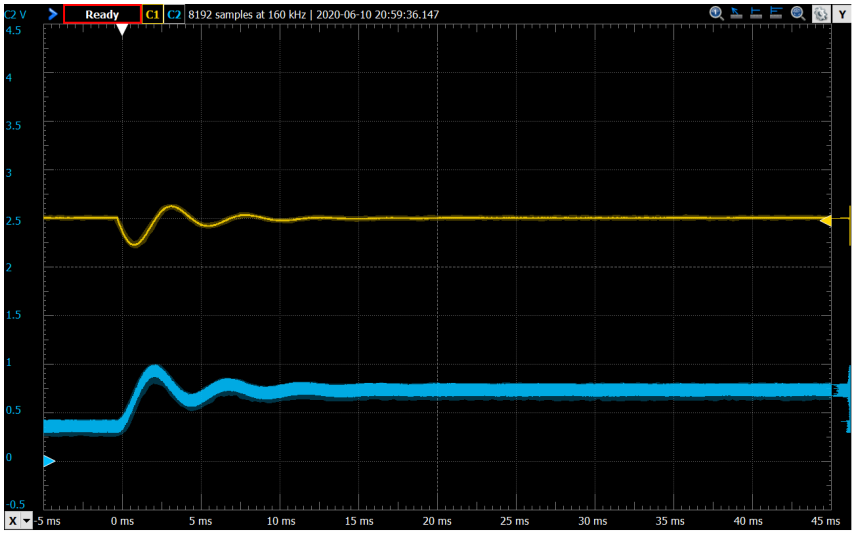


Figure 14: Integrator: Load switch from $R_O = 100\ \Omega$ to $R_O = 50\ \Omega$. Yellow (1 V/div): Output voltage, Blue (0.5 V/div): Current sensor output.

switch and this one except that there is no steady state error with the integrator.
To conclude, the integrator works well in practice because there is low overshoot and no steady state error. Though, it presents a quite poor performance in terms of settling speed. Indeed, the behaviour of the system is not anticipated like it can be with an LQR or an MPC controller.

6 Model-based control

The integral controller used in the previous section works well in practice but is slow compared to the open loop response. For speeding up the startup transient, model-based control has to be used. Since the output load is not known a priori, a nominal load and thus a nominal model is chosen. Here, the nominal load is 100 Ω. The tuning of the parameters is only based on this nominal model. Then, an integrator is added to compensate for modeling errors once the system is settled. The system is considered to be settled when the voltage difference between 2 consecutive samples is lower than 100 mV 100 times in a row. Disabling the integrator during the startup phase allows to avoid overshoot.
In order to show the motivation of using MPC instead of linear control, an artificial constraint is added. Indeed, the inductor current is limited to 200 mA while the real limit imposed by the hardware is 400 mA. If the limit of 400 mA is kept, the constraints on the state are never reached and the motivation of using MPC is lost.

6.1 C implementation

The microcontroller is coded in C. The configuration of the input and output peripherals is presented in table 4. The ADCs are used with Direct Memory Access (DMA). In this case, the ADC is continuously filling a buffer with a frequency of 692 kHz. The use of DMA requests allows to store values from the ADCs directly into the memory without using the CPU. Thus, precious time is saved. Indeed, whenever the callback routine is called, there is no need to start the ADC, make the conversion and get the value. The value is already stored in the buffer. The latter can contains 7 unsigned 16 bits integers. Therefore, it contains the ADC values of the last $\frac{7}{0.692} \approx 10$ μs. Since the callback routine is called every 100 μs, it is reasonable to take the median of the buffer values and assume it as the measured value. By taking the median of the buffer values, noisy samples are avoided.

Pin	**Peripheral**	**Function**
None	Timer 3	Interruption routine
PB3/D3	Timer 2 channel 2	PWM generation
PA10/D2	GPIO output	Timing
PA0/A0	ADC1 channel 5 with DMA	Divided output voltage sampling
PA1/A1	ADC2 channel 6 with DMA	Current sensor output sampling
PA4/A2	DAC output 1	Information extraction (duty cycle, regions...)

Table 4: Pin description.

The code contains almost only integers. Indeed, the execution time of the callback routine is approximately 10 times faster than with double values. Since it would be too imprecise to truncate or round the original numbers, they are multiplied by a high coefficient between one thousand and one million and then rounded. If a multiplication by these numbers is done, the result is then divided by the coefficient.

Even if a direct access to the state values is possible with the embedded sensors, the model is combined with the measurements to estimate the state. The reason for that is explained later when the explicit MPC controller is presented. The estimated state is a weighted sum of the measured state and the inferred state from the model. Thus, the state estimate is computed as follow:

$$x(k) = \alpha\, y(k) + (1-\alpha)(Ax(k-1) + Bu(k-1)) \tag{22}$$

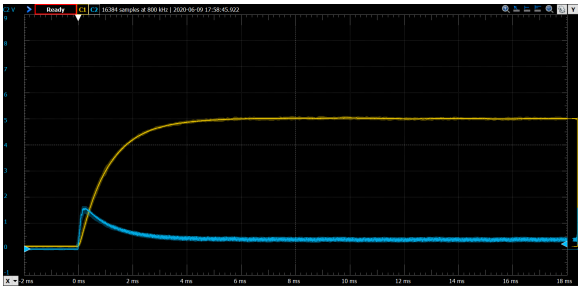
Where y(k) is the measured state at step k and α is a weighting factor such that $0 \leq \alpha \leq 1$.

6.2 LQR

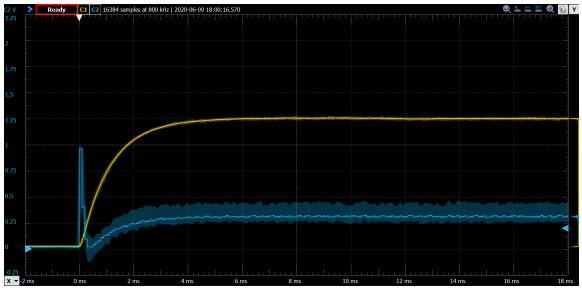
The Q and R matrices are tuned so that, during the startup phase, the inductor current never exceeds 200 mA and the input is never saturated. The resulting values are $Q = \begin{pmatrix} 500 & 0 \\ 0 & 1 \end{pmatrix}$ and $R = 10$.
The weighting factor α is set to 0.5. Since the steady state target is not the origin, the LQR controller has to be "shifted" in the state space and the input space. Therefore, instead of applying $u = -Kx$, $u = -K(x - x_s) + u_s$ is applied. x_s and u_s are the steady state values computed for the nominal model. Hence, if the load is not the nominal load, steady state errors occur. In order to avoid this, the integrator is added.
The scope plots are shown on figure 15 and the comparison of the real response with the simulated one on figure 16. Table 5 reports on the performance of the LQR.

	Startup	Switch 100 Ω to 50 Ω	Unit
Settling time	4.33	5.46	ms
Rise time	2.39		ms
Maximum over/undershoot	0	12.7	%
Steady state error	0	0	V

Table 5: Performance of the LQR.



(a) Yellow (1 V/div): Output voltage,
Blue (1 V/div): Current sensor output.



(b) Yellow (1 V/div): Output voltage,
Blue (0.25 V/div): Duty cycle.

Figure 15: Scope screenshots: Startup transient with the LQR.

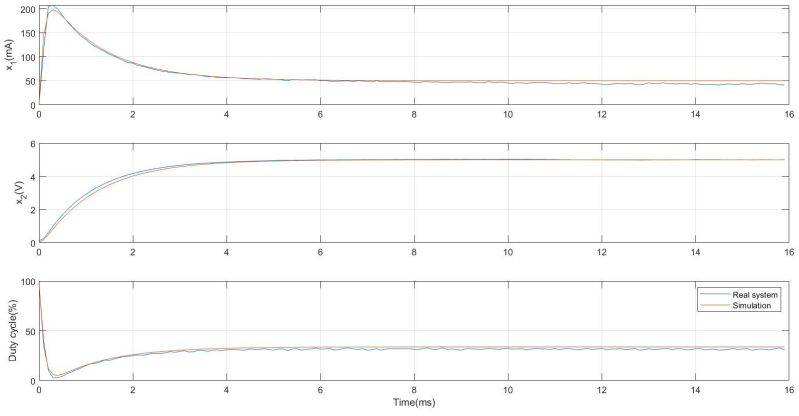


Figure 16: Comparison between the simulated and the measured response of the LQR.

According to the startup phase, the measured response is very similar to the simulated one. This is a good proof that the model is close to the real system. Neither the state nor the input constraints are broken. The settling time is more than 2 times shorter than the one of the open loop response and the transient presents no oscillations and no overshoot.

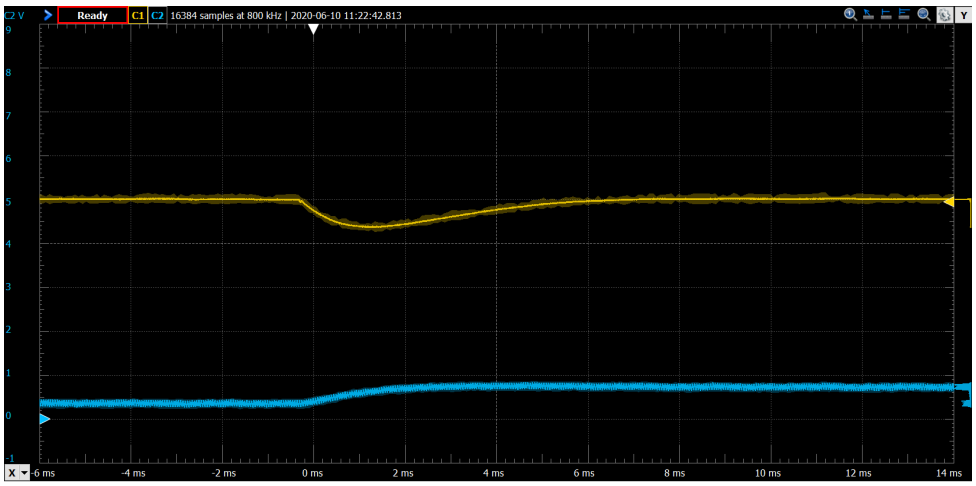


Figure 17: LQR: Load switch from $R_O = 100\ \Omega$ to $R_O = 50\ \Omega$. Yellow (1 V/div): Output voltage, Blue (1 V/div): Current sensor output.

It can be seen on figure 17 that during a load switch, the integrator corrects the steady state error without oscillations but the bump is a bit higher than the one in open loop. The settling time is also a bit longer. Indeed, the LQR is designed for the nominal model with the nominal steady state values. If the load is not the nominal one, the steady state current is wrong and since the origin has been shifted to the nominal steady state target, the LQR does not drive the states to the origin. With an output load of 50 Ω, the steady state current and voltage are respectively 100 mA and 5V. However, the LQR drives the current to 50 mA and the voltage to 5V. This steady state difference of 50 mA induces an error on the input. The resulting steady state error is higher than if the input is held constant no matter the load (open loop). Hence, a higher bump and a longer settling time occurs with the LQR.
The computation time is 9.7 μs, namely 9.7 % of the sampling period.

6.3 Explicit model predictive control using neural networks

First, the feasible set is defined. The lower bound for the voltage and the current is 0 V and 0 mA in order to avoid states where the model is not linear. The lower bound of the duty cycle is obviously 0. The upper bound for the voltage is set to 7V, only 2V above 5V. This enables to have a reasonable voltage range for a further normalization. This normalization is then necessary to train the neural network. The upper bound for the current has already been set to 200 mA. Finally the upper bound for the duty cycle is 1. Thus, the constraints can be written as follow:

$$\begin{aligned} x_{min} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \leq \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 0.2 \\ 7 \end{pmatrix} = x_{max} \\ u_{min} = 0 \leq u \leq 1 = u_{max} \end{aligned} \tag{23}$$

Like for the LQR controller, the origin is shifted to the nominal steady state target. Let $\Delta x = x - x_s$ and $\Delta u = u - u_s$. The state constraints become:

$$\begin{aligned} &\Delta x_{min} \leq \Delta x \leq \Delta x_{max} \\ \Rightarrow &\begin{pmatrix} -0.05 \\ -5 \end{pmatrix} \leq \Delta x \leq \begin{pmatrix} 0.15 \\ 2 \end{pmatrix} \\ &\Delta u_{min} \leq \Delta u \leq \Delta u_{max} \\ \Rightarrow &-0.3379 \leq \Delta u \leq 0.6621 \end{aligned} \tag{24}$$

Now, regulation to the origin can be performed.
The MPC problem can be witten as follow:

$$\begin{aligned} J^*(\Delta x) = \min_{\Delta u_i} \Big(&\textstyle\sum_{i=0}^{N-1} (\Delta x_i^T Q \Delta x_i + \Delta u_i^T R \Delta u_i) + \Delta x_N^T P_{LQR} \Delta x_N \Big) \\ &s.t.\ \Delta x_{i+1} = A\Delta x_i + B\Delta u_i \\ &\Delta x_{min} \leq \Delta x_i \leq \Delta x_{max} \\ &\Delta u_{min} \leq \Delta u_i \leq \Delta u_{max} \\ &\Delta x_N \in X_{LQR} \\ &\Delta x_0 = \Delta x \\ &i = 0, 1, ...,\text{N-1} \end{aligned} \tag{25}$$

Where N is the horizon of the MPC problem, Q the state penalty, R the input penalty, X_{LQR} the invariant set of the LQR control law and P_{LQR} the solution of the associated Riccati equation.

The penalty matrices Q and R are tuned so that the performance is maximized while keeping the origin in the feasible set. The horizon of the MPC problem is set to 10. A smaller horizon would shrink the feasible set and the origin would not be in the feasible set. In order to avoid this, a higher Q could be set but the performance would decrease. Setting a higher horizon would allow to set a smaller Q and thus slightly increase the performance.
The MPC controller is implemented and simulated on Matlab with the help of the Multi Parametric Toolbox 3. An explicit solution of the MPC problem is then computed. The original explicit solution contains 70 regions. Figure 18 shows the explicit optimizer and the regions as a function of the state.

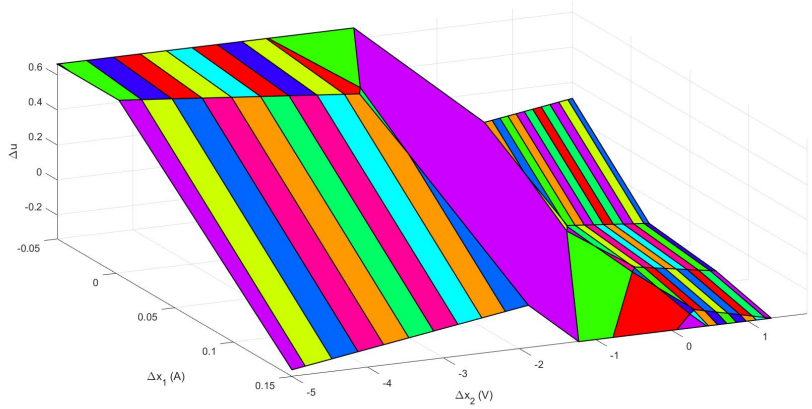


Figure 18: Original explicit optimizer versus state.

Each region is a polytope in the state space to which corresponds to an affine function that outputs the optimizer. Thus, the problem is explicitly solved offline. A lookup table is then coded into the microcontroller. Online, the microcontroller has to know what affine control law it must apply by knowing in which region the state is. In the worst case, it has to check 70 regions before finding the good one. The

microcontroller used for this work is not fast enough to check 70 regions in less than 100 μs (sampling period). Besides, in the MPC theory, it is assumed that sampling and applying the input is simultaneous. This is not possible in reality because of the computation time. However, the computation time has to be as short as possible in order to be as close as possible from the theory. Thus, the number of regions has to be reduced.
Fortunately, it can be seen that some regions are related to very similar affine functions. A simplification would merge the previous mentioned regions in one region. The simplification implemented in this work uses a neural network (NN). The theory behind this simplification method is developed in [1]. The goal here is to test this method in a practical way.
Online, the normalized state is given as the input of the NN and the latter returns the optimizer. The NN is made of 4 layers. The first and the third layers are an affine mapping, the second layer is a parametric quadratic program whose solution is explicitly solved and the last layer projects the given input into the feasible input set. An illustration of the neural network is presented on figure 19.

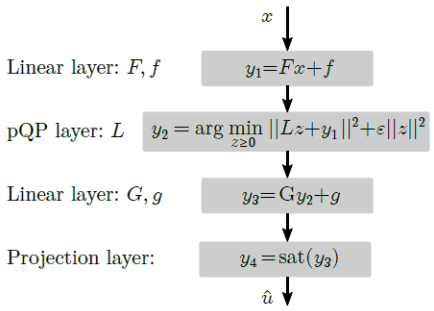
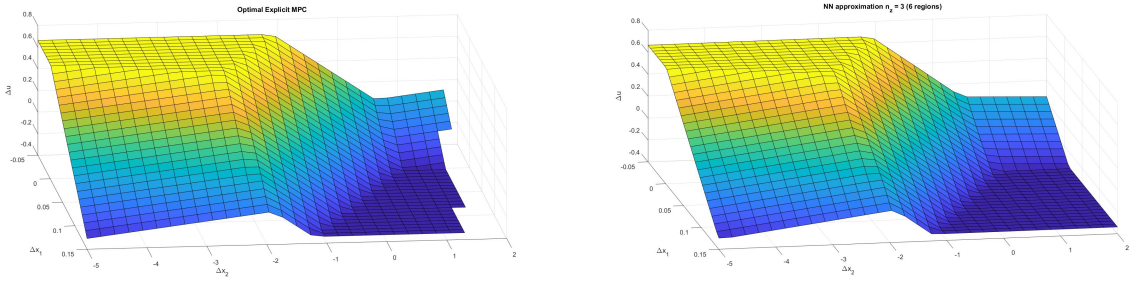


Figure 19: An illustration of the NN eMPC controller approximator. The parameters to be optimized are shown on the left-hand side [1].

After the NN has been trained, the optimizer of the simplified MPC controller is compared to the optimal one. Figure 20 shows that the simplification provides a very good fit of the optimal MPC control law.

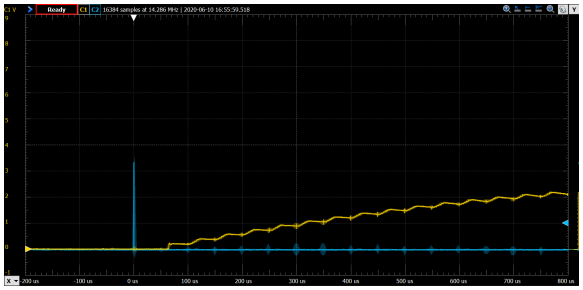


(a) Optimal controller.

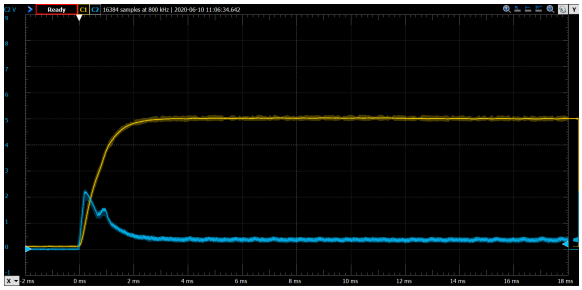
(b) NN simplification.

Figure 20: Optimizer vs state.

According to the state estimation, the method mentioned in subsection 6.1 is used. Indeed, there is a critical phase during which the current is supposed to be stuck at the upper limit. This can be seen on the first plot of figure 23. If only the measurements are used to get the state, the constraints are broken just at the beginning of this critical phase. Figure 21b shows a high current peak above the upper limit. This is due to the fact that there is a delay of about 60 μs between the first time the duty cycle is given and the time at which the current starts increasing. It can be seen on figure 21a. Because of this delay the current is lower than expected at the second sampling step. Thus, the corresponding input is higher than expected and the constraint on the current is violated at step 3. This delay could come from a non-linearity in the system. This is suspected that the diode is not ideal and needs some time before conducting the current.
For this experiment, the weighting factor α of the measurements is set to 0.25. This allows to compensate for the error induced by the delay.



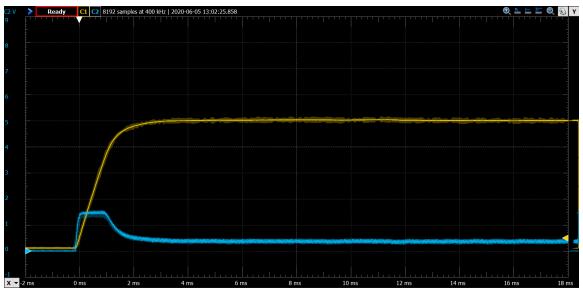
(a) Duty cycle current delay.
Yellow (1 V/div): Current sensor output,
Blue peak (1 V/div): Duty cycle command given.



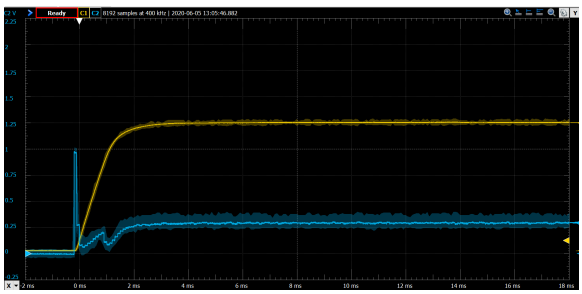
(b) eMPC using only the measurements
Yellow (1 V/div): Output voltage,
Blue (1 V/div): Ourrent sensor output.

Figure 21

Besides, the integrator is only enabled if the state is in the terminal set and, like previously, if the system is settled. Finally, if the state happens to be in no region, a slow LQR control law is applied.
The scope plots on figure 22 demonstrate excellent results. Indeed, the transient is very smooth, without overshoot, oscillations or steady state error. The current successfully hits the upper limit and stays close to it during the critical phase. This results in a very fast settling speed.



(a) Yellow (1 V/div): Output voltage,
Blue (1 V/div): Current sensor output.



(b) Yellow (1 V/div): Output voltage,
Blue (0.25 V/div): Duty cycle.

Figure 22: NN eMPC: Startup transient scope screenshots.

The real system plots are then compared to the simulation ones on figure 23. Once again, the high similarity between these plots underlines the importance of having a good model. It is also interesting to notice that the real and simulated state sequence cross the same regions simultaneously.
Figure 24 shows the trajectory in the state space. With this representation, it is clearly visible that the second samples are very far from each other because the actual current (x_1) is much lower than expected. Since the model account for this error, the current used to compute the control law is not as low and the resulting input is not too high. Thus, the samples get closer later and the constraints are not broken.

	Startup	Switch 100 Ω to 50 Ω	Unit
Settling time	2.73	2.62	ms
Rise time	1.50		ms
Maximum over/undershoot	0	8.3	%
Steady state error	0	0	V

Table 6: Performance of the eMPC controller.

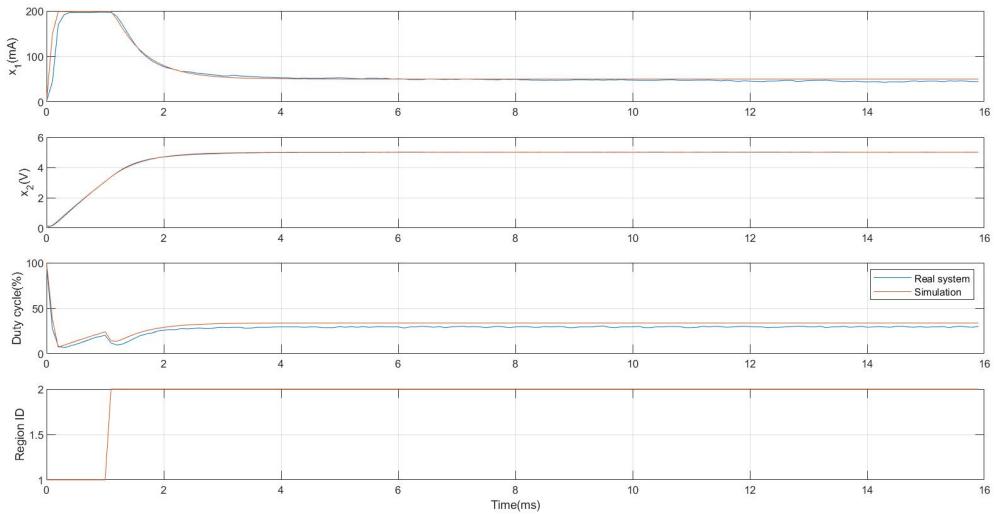


Figure 23: eMPC: Comparison between the simulated and the measured response.

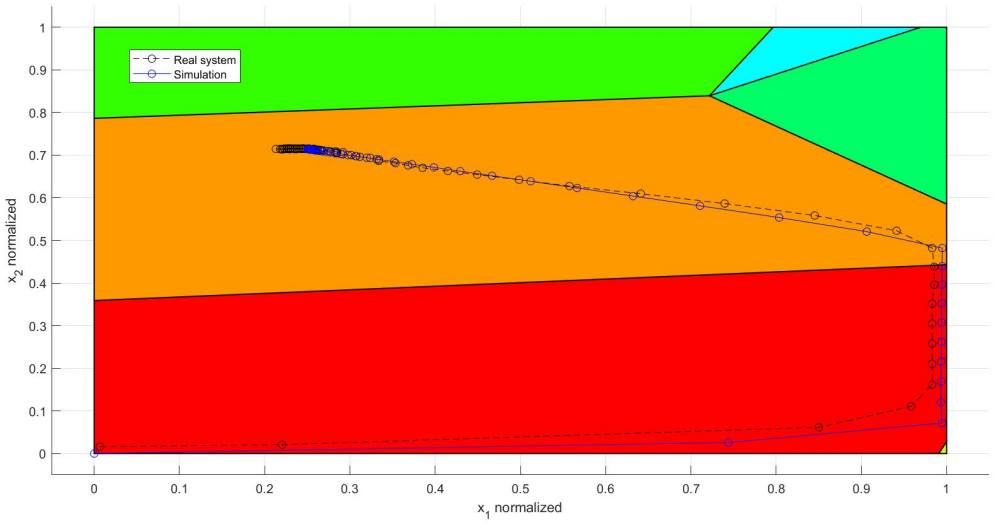


Figure 24: eMPC: State space trajectory.

The response during a load switch is presented on figure 25. The integral gain used here is 4 times higher than the one of the LQR presented in section 6.2. The reason is that the LQR begins to show oscillations with a higher integral gain. An hypothesis is that the LQR is slower than the control law applied here in the terminal set. Therefore, if the integral gains are equal for both controllers, the contribution of the integral term is higher for the LQR than for eMPC controller. Thus, oscillations due to the integral term appear with the LQR. As expected, the settling speed of the eMPC controller is about 2 times higher than the one of the LQR. The performance of the eMPC controller is presented in table 6. One can see that the eMPC controller startup settling speed is about 2 times higher than the one of the LQR. This can be explained by the fact that the current can be hold constant at the upper limit in the eMPC case while it cannot be in the LQR case. In the eMPC case, the output capacitor is more rapidly charged and the voltage rises more rapidly.
The computation time is 30 us. This is reasonable since it represents 30 % of the sampling period.

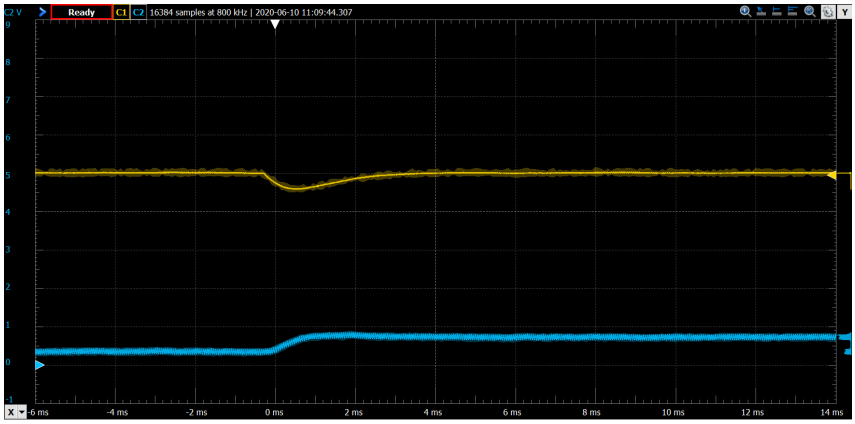


Figure 25: eMPC: Load switch from $R_O = 100\ \Omega$ to $R_O = 50\ \Omega$. Yellow (1 V/div): Output voltage, Blue (1 V/div): Current sensor output.

A comparison between the open loop, the LQR and the eMPC response is presented on figure 26.

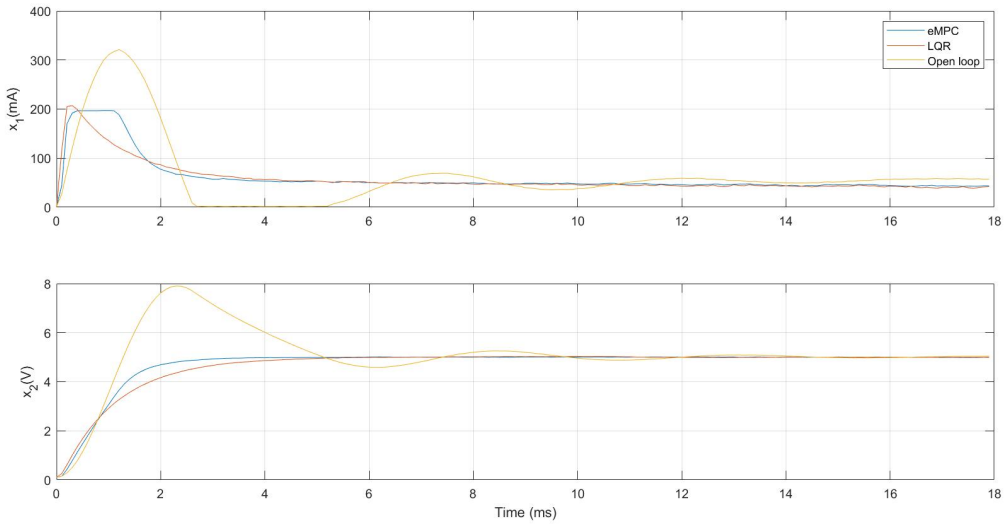


Figure 26: Comparison: Open loop vs eMPC vs LQR in practice.

7 Conclusion

A step-down DC-DC converter has been designed and built from scratch in order to perform model-based control. Then, several control techniques have been tested. Firstly, the components have been chosen by taking into account the requirements imposed by the microcontroller and the hardware. Then, the circuit has been designed, manufactured and soldered. Secondly, a model has been derived and tested in open loop in order to perform further model-based control. The result is excellent since the modeled behaviour is very close to the real one. Thirdly, the implementation of an integral controller demonstrates the practical simplicity of this control technique but shows a quite limited performance. Fourthly, the LQR is compared to a simplified explicit MPC controller using neural networks. Both methods present a much better performance than the open loop system and the integral controller. However, the simplified eMPC controller is able to deal with constraints while the LQR is not. Thus, the simplified eMPC controller is approximately 2 times faster than the LQR which relates the efficiency of the simplification method. Though, it has been shown that the eMPC controller is very sensitive to plant-model mismatch. Therefore, a robust eMPC controller could be implemented in order to overcome that. Moreover, a model augmented with a disturbance estimate could be used instead of the integrator to account for modeling errors and variable loads. All of this is still let for further work.

8 References

[1] E. T. Maddalena C. G. da S. Moraes G. Waltrich C. N. Jones. A neural network architecture to learn explicit mpc controllers from data. 2019.

[2] Andrea Giovanni Beccuti Sébastien Mariéthoz Sébastien Cliquennois Shu Wang Manfred Morari. Explicit model predictive control of dc–dc switched-mode power supplies with extended kalman filtering. 2009.

[3] Prof. C. N. Jones. Model predictive control, epfl. Course slides.