Speed control of a model train

1 Objective

We want to pilot an electric train using an FPGA BASYS-3. This board will control the electric motor of the train by a PWM command and the train speed will be measured by reading the counter-electromotive force of the motor using an ADC MCP3002.

In this second part of the project we are interested in the acquisition of the speed command, and the control using PWM, and finally the control with feedback.

2 Acquisition of speed setpoint

The speed setpoint is encoded on 6 bits. The push buttons (ref. 7 of Figure 1) are used to do the acquisition.

The button *up arrow* is used to increase and the button *down arrow* is used to decrease the setpoint value, within the boundaries of the interval $[0, 63]$.

Build the circuit performing the setpoint acquisition. Display the setpoint value on the two 7-segment displays on the right (ref. 4 of Figure 1).

Tips: Reuse the code previously written for the 7-segment display and the button debouncer. Use as much as possible composition.

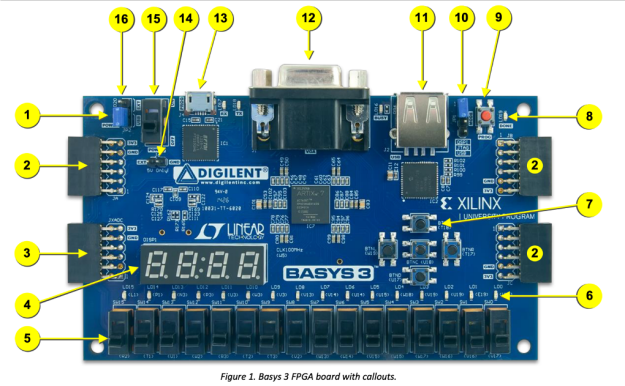

Figure 1. Basys 3 FPGA board with callouts.

Callout	Component Description	Callout	Component Description
1	Power good LED	9	FPGA configuration reset button
2	Pmod port(s)	10	Programming mode jumper
3	Analog signal Pmod port (XADC)	11	USB host connector
4	Four digit 7-segment display	12	VGA connector
5	Slide switches (16)	13	Shared UART/ JTAG USB port
6	LEDs (16)	14	External power connector
7	Pushbuttons (5)	15	Power Switch
8	FPGA programming done LED	16	Power Select Jumper

Figure 1: BASYS 3

3 The Pulse Width Modulation

3.1 PWM in simulation

Generate a PWM signal with a frequency which you choose (of about 20 kHz) whose duty cycle is controlled by the setpoint value in steps of 1/64%.

- If $setpoint = 0$ then the PWM is always inactive (motor not supplied),
- If $setpoint = 63$ then the PWM is always active (motor fully supplied).

Check your PWM signal in simulation.

3.2 PWM - no feedback

To control the train motor in power, we use an extension board Figure 2 which is connected to a PMOD connector (ref. 2 in Figure 1) of your choice.

Send your PWM generated signal to the PWM pin of the PMOD connector.
Ask the teacher for a motor (do not try with the locomotive yet).
Use the K4 connector (Figure 2), and an adaptor module BNC 4mm.

- Supply the board with 18V, between pins GND and $18V$.
- Branch the motor between pins GND and $Moteur$.
- Ask the teacher for a checking before turning the power supply on.

At the end of this step, the digital encoder directly controls the PWM, and thus the speed of the locomotive. So we have an open loop control (no feedback).

3.3 Speed control with feedback

At this point we aim at reading the counter-electromotive force of the motor. The counter-electromotive force gives us the speed of the motor, when the PWM is inactive.

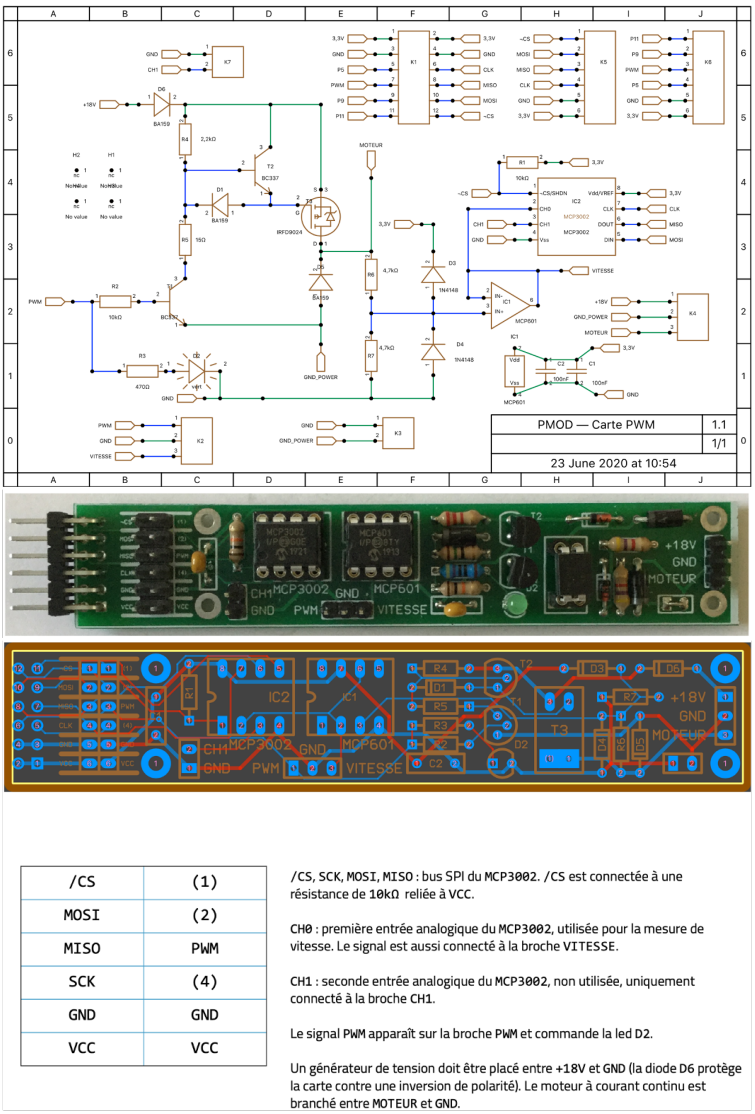

Figure 2: The extension board PWM + ADC MCP3002

Adapt your circuit to periodically stop the PWM during a time long enough to be able to measure the voltage at the motor terminals using the MCP 3002. Build the following cycle of 10ms:

- the first 9ms, the PWM is active (with its normal duty);
- the last 1ms, the PWM is stopped, and the speed is measured.

Reuse your SPI master component to communicate with the MCP3002.

First display the data acquired on the 10 LEDs (ref. 6 of Figure 1), and then with a format of your choice on the two 7-segment diplays on the left.

Build the proportional controller: $command = K * (setpoint - measure)$. You will need to set the setpoint and the measure in equivalent formats. Finally, control your PWM with the computed command.