Cursor (Cur)

Lecture Digital Design (DiD)

Hes·so VALAIS WALLIS

School of Engineering

Orientation: Information and Communication Technology (ISC)
Specialisation: Data Engineering (DE)
Course: Digital Design (DiD)
Authors: Silvan Zahno, Axel Amand, François Corthay, Christophe Bianchi
Date: 01.12.2025
Version: v3.1

Contents

1 Introduction

The goal of this project is to apply the acquired knowledge in a practical example at the end of the semester. It involves controlling a DC motor to precisely move a carriage along a lead screw to predefined positions. This positioning system can be seen in Figure 1.

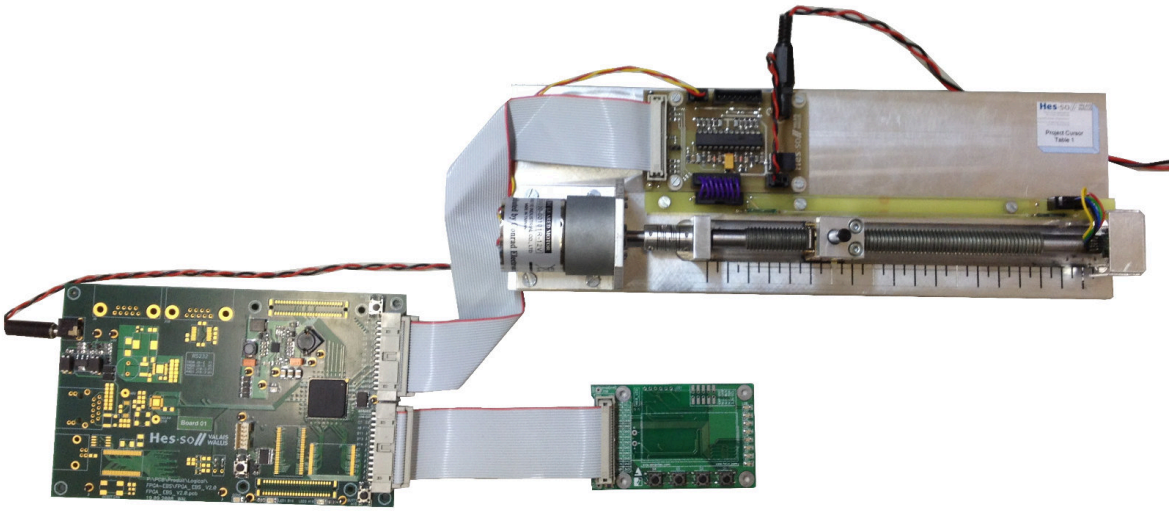

Figure 1 - Hardware setup Cursor (EBS2)

The minimum specifications (see Section 2) must be met, but students are encouraged to implement additional features. For example, an **L**iquid **C**rystal **D**isplay (LCD) screen can be used to display various information.

Implementing extra features will provide additional points in the final evaluation.

2 Specification

2.1 Functions

The basic functions are defined as follows:

- When the **`restart`** button is pressed, the cursor moves to the start position indicated by a reed relay (Section 3.4) located near the DC motor (Section 3.2.1).
- When the Position$_1$ button is pressed, the cursor must first accelerate smoothly to position 1 (p_1), then move at full speed and finally decelerate smoothly to stop at position 1 (p_1). This can be done from the start position or from position 2, see Figure 2.
- When the Position$_2$ button is pressed, the cursor must first accelerate smoothly to position 2 (p_2), then move at full speed and finally decelerate smoothly to stop at position 2 (p_2), see Figure 2.

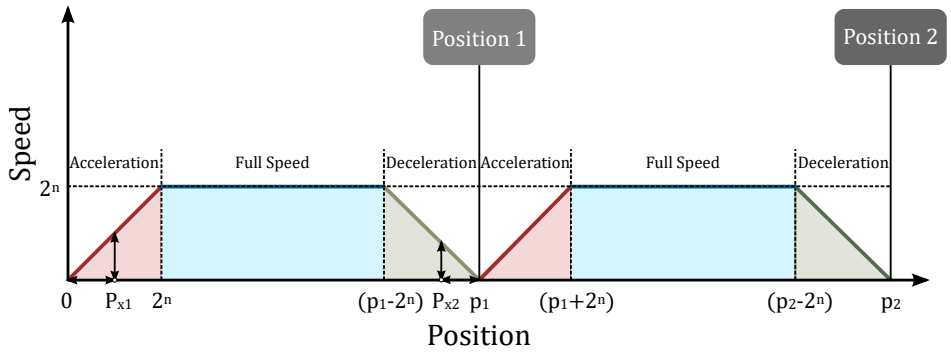

Figure 2 - Cursor speed diagram

The target positions are:

- Position 0 (p_0, reset position) = 3.3 - 3.5 cm following mechanical assemblies
- Position 1 (p_1) = 8 cm
- Position 2 (p_2) = 12 cm

2.2 Circuit

To accomplish the task, the following circuit is provided to move the carriage.

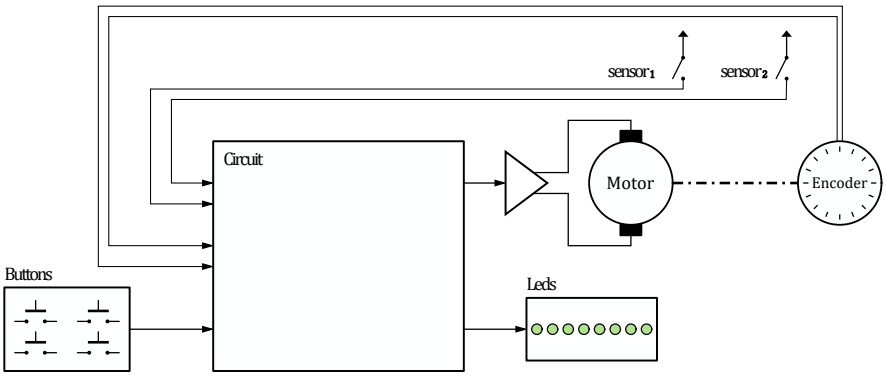

Figure 3 - Cursor circuit

2.3 Circuit

The circuit works as follows:

- The DC motor (Section 3.2.1) is controlled by the three signals motor_{On}, side_1, side_2:
  - The motor is activated when the signal motor_{On} is set to 1 and must be reset to 0 when the motor should not run.
  - Its speed is controlled by a **P**ulse **W**idth **M**odulation (PWM) modulation applied either to the signal side_1 or to the signal side_2 depending on the desired direction of rotation.
- Two reed relays (Section 3.4) sensor_1 and sensor_2 are placed at the ends of the rail [1]. They detect the presence of the cursor carriage by indicating the presence of a magnet with a 1.
- The encoder (Section 3.3) is used to track, respectively count the position of the cursor. Its three outputs, encoder_A, encoder_B and encoder_I, allow tracking the movements of the screw. The A and B outputs alternate between 0 and 1 during movements with a phase shift of 90° between them, which allows determining the direction of movement. The I output generates a 1 pulse at each complete turn of the screw.
- Three buttons are used to control the system: restart, go_1 and go_2. An additional button, button_4, can be used for optional functions. When one of the buttons is pressed, the corresponding signal switches to 1.
- The pins testOut can be used to output additional information from the system, for example to connect them to the **L**ight **E**mitting **D**iodes (LEDs) for debugging or visualization purposes.
- The signal testMode is set to 1 during simulation. This can be used to shorten the counters used for signal generation during tests to speed them up.

The empty toplevel design (**`cursor-toplevel-empty.pdf`**) shows all signals connected to the **F**ield **P**rogrammable **G**ate **A**rray (FPGA) board Figure 4.

Figure 4 - Empty Toplevel Circuit

2.4 Scenario (example)

In Figure 5, three different scenarios are presented. First, the restart button is pressed and the carriage moves at full speed to the start position ($sensor_1$). The other two scenarios go_2 and go_1 move the carriage to the $position_2$ and $position_1$ respectively, a variable PWM signal is applied to the signals $side_2$ and $side_1$ to accelerate and decelerate the carriage.

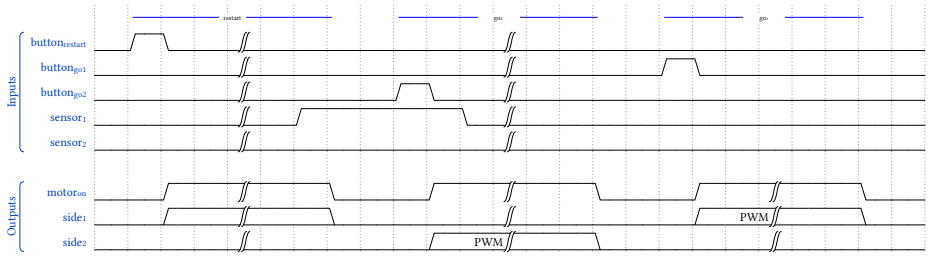

Figure 5 - Scenario cursor

The above scenarios are examples. It is up to the students to complete them.

2.5 HDL-Designer Project

A predefined HDL-Designer project can be downloaded or cloned from Cyberlearn or Github. The file structure of the project looks as follows:

```
did_cursor
+--Board/           # Project and files for programming the FPGA
|   +--concat/      # Complete VHDL file including PIN-UCF file
|   +--ise/         # Xilinx ISE project
+--Cursor/          # Library for the components of the student solution
+--Cursor_test/     # Library for the simulation testbenches
+--doc/             # Folder with additional documents relevant to the project
|   +--Board/       # All schematics of the hardware boards
|   +--Components/  # All data sheets of hardware components
+--img/             # Pictures
+--Libs/            # External libraries which can be used e.g. gates, io, sequential
+--Prefs/           # HDL-Designer settings
+--Scripts/         # HDL-Designer scripts
+--Simulation/      # Modelsim simulation files
```

The path of the project folder must not contain spaces.

In the project folder **doc/** many important information can be found. Datasheets, project evaluation as well as help documents for HDL-Designer to name just a few.

3 Components

The system consists of 3 different hardware boards, visible in the Figure 1.

- A carriage assembly with a **P**rinted **C**ircuit **B**oard (PCB) board that controls the motor and reads the sensors (Figure 6)
- A FPGA development board (Figure 14 or Figure 15)
- A control board with 4 buttons and 8 LEDs (Figure 16)

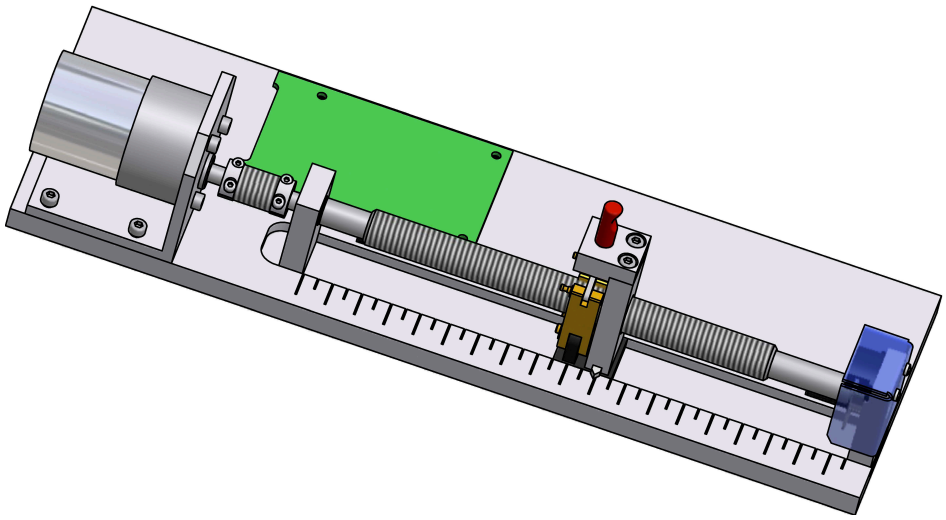

Figure 6 - Cursor carriage assembly

3.1 Carriage

The carriage structure includes the DC motor, the two reed relays Section 3.4 as well as the carriage and the lead screw. The lead screw thread size is M12x1.75, meaning that a distance of 1.75mm is covered per revolution (Figure 7).

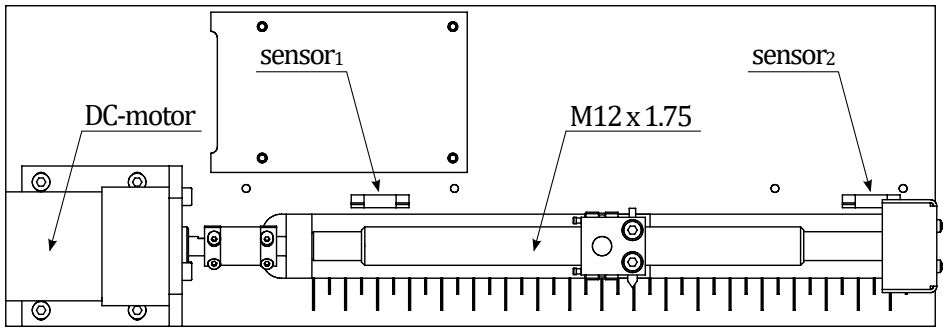

Figure 7 - Detailed assembly of the cursor carriage

3.2 Motor control circuit

The DC motor of the carriage is powered by 12V. The power board has an H-bridge that is controlled by digital signals. On the power board, a 5V regulator generates the voltage powering the FPGA board [2].

3.2.1 Direct current motor

The DC motor is controlled by an H-bridge driver L6207 [3], see figure Figure 8. The maximum switching frequency of the H-bridge is 100kHz. This should be taken into account when creating the PWM signal.

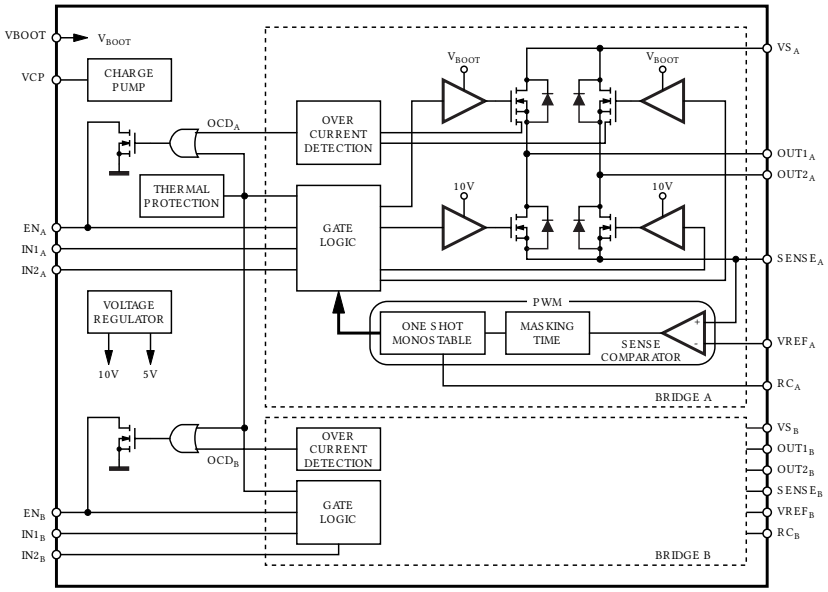

Figure 8 - Block diagram of the L6207N H-bridge circuit [3]

To adjust the speed of the DC motor, a PWM signal must be applied to the $side_1$ or $side_2$ signals, the maximum frequency being 100kHz. The longer the voltage is applied to the motor, the faster it turns. Powering either $side_1$ or $side_2$ controls the direction of the motor.

PWM - Implementation strategy

One strategy is to create a sawtooth counter (0, 1, 2 ... $2^N - 2$, $2^N - 1$, 0, 1, ...) and compare its value to a threshold. The rule is to output '1' when the counter value is less than the threshold, and '0' otherwise. By changing it, the duty cycle of the PWM signal is altered. In this way, the frequency of the PWM signal is defined by the speed of the counter (sawtooth), and the duty cycle is defined by the threshold value.

In the Figure 9, the motor turns slower with the green signal than with the blue signal and than with the red signal.

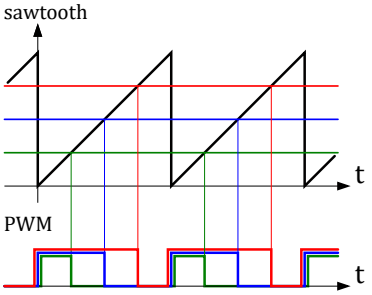

Figure 9 - PWM signals

The more steps (bits) the PWM has, the more precise the speed control is. However, too many steps may result in no notable speed difference due to control systems and motor characteristics. A good basis is to use 8 bits for the PWM resolution, giving 256 speed levels from 0 to 255.

Ramps - Implementation strategies

To create acceleration and deceleration ramps, a common method is to use a trapezoidal speed profile. This profile consists of three phases: acceleration, constant speed, and deceleration as shown under Figure 10.

Multiple strategies exist to create such ramps. One - simpler to implement - method is given under Section 4.

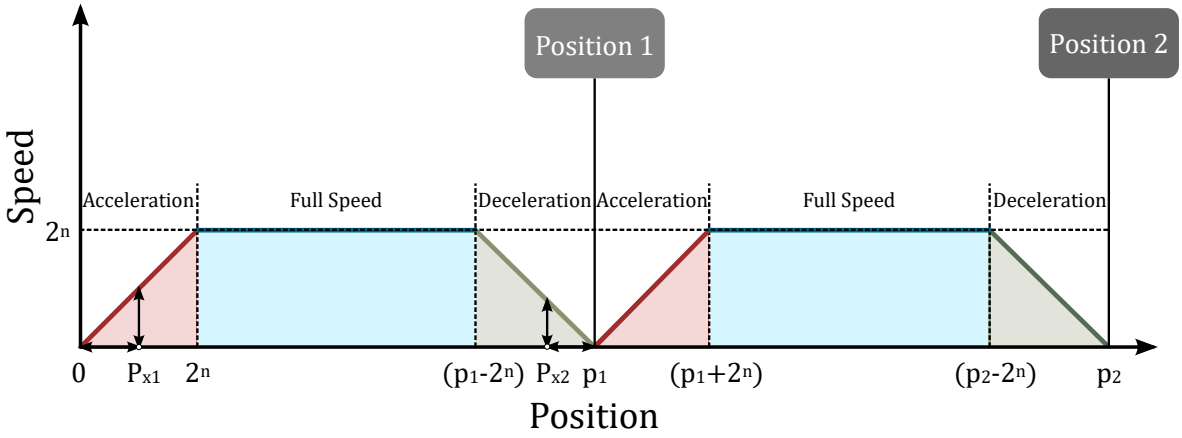

Figure 10 - Cursor speed diagram

3.3 Encoder

The angle of the lead screw can be measured using an incremental encoder. The model used in the assembly is an AEDB-9140-A12 [4] (Figure 11) with 2 feedback channels with 500 **C**ounts **p**er **R**evolution (CPR) (counts per revolution) per channel, represented in the Figure 12.

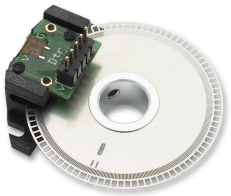

Figure 11 - Encoder AEDB-9140-A12 [4]

Encoders - Implementation strategy

It is possible to use it in two ways:

- By using a single channel, which gives a resolution of $500\frac{\text{pulses}}{\text{revolution}}$ i.e. $1'000\frac{\text{edges}}{\text{revolution}}$ if counting both rising and falling edges.
- By using both channels as a QEI (Quadrature Encoder Interface), counting the rising and falling edges of both channels, increasing the resolution to $2000\frac{\text{edges}}{\text{revolution}}$. This interfacing also allows to detect the rotation direction of the motor and is a common way to control DC motors with feedback encoders in the industry.

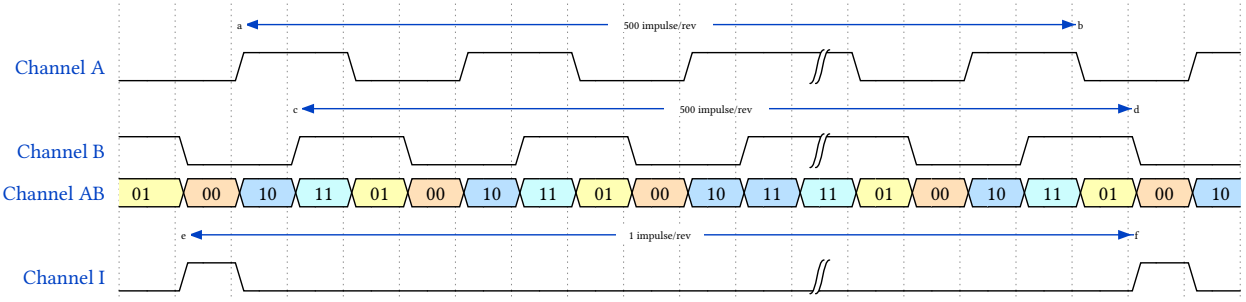

Figure 12 - Signals of incremental encoders

3.4 Reed relay

The reed relay is a switch toggled using electromagnets [1]. When a magnet is near the sensor, the contact closes (Figure 13). 2 reed relays are used (sensor_1 and sensor_2) to identify the left and right limits of the carriage.

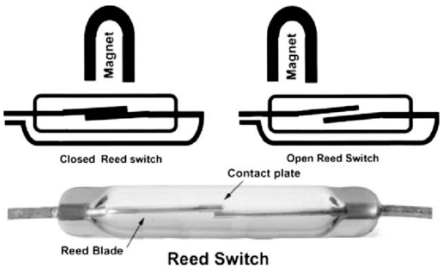

Figure 13 - Reed relay [5]

The signals sensor_1 and sensor_2 are '1' when the contact is closed (magnet nearby), otherwise '0'.

3.5 FPGA board

The main board is the school's FPGA-EBS 2 laboratory development board [6]. It hosts a Xilinx Spartan xc3s500e FPGA [7], [8] and has many different interfaces (**U**niversal **A**synchronous **R**eceiver **T**ransmitter (UART), **U**niversal **S**erial **B**us (USB), Ethernet, etc.). The oscillator used produces a clock signal (**`clock`**) with a frequency of $f_{\text{clk}} = 66\text{MHz}$ [9].

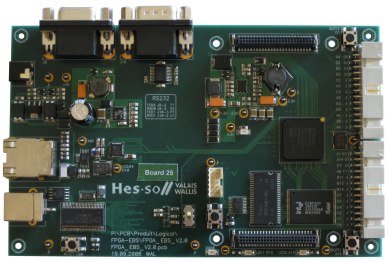

Figure 14 - FPGA board [6]

On the EBS3 board, the oscillator used produces a clock signal (**`clock`**) with a frequency of $f_{\text{clk}} = 100\text{MHz}$, reduced by PLL to $f_{\text{clk}} = 60\text{MHz}$.

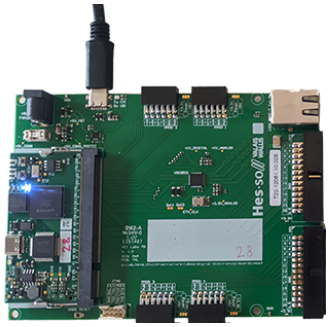

Figure 15 - FPGA board [10]

> The simulators are set by default for the EBS3 boards. To change them, open a testbench block **xxx_tb** and double-click on the **Pre-User** declarations (top left of the page) to modify the **clockFrequency** variable according to the desired clock value.

3.6 Buttons and LEDs

The board with the buttons and LEDs [11] is connected to the FPGA board. It has 4 buttons and 8 LEDs that can be used in the design. This board is equipped with an LCD display [12], [13].

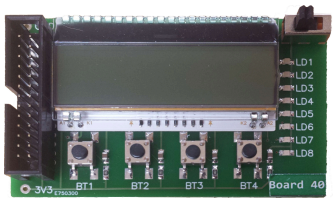

Figure 16 - Button-LEDs-LCD board [11]

4 Motor Ramp Implementation

This chapter is a proposition for implementing acceleration ramps. Other methods exist and can very well be used.

To create acceleration and deceleration ramps, a common method is to use a trapezoidal speed profile. This profile consists of three phases: acceleration, constant speed, and deceleration as shown under Figure 17.

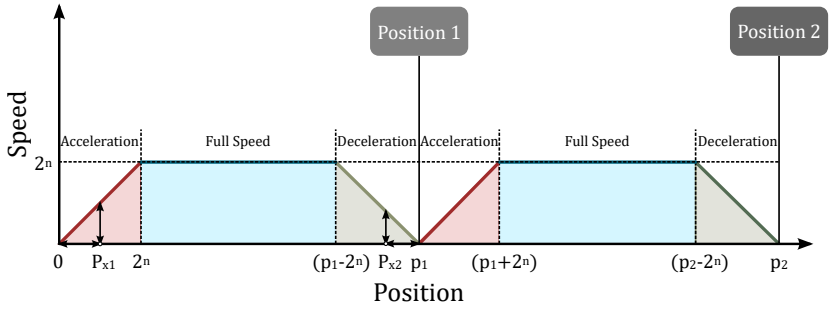

Figure 17 - Cursor speed diagram

Both acceleration and deceleration ramps use the same profile.

4.1 Ramps base concept

Typically, speed profiles are based on the time. This method is often encountered in industrial applications.
However, such systems need a whole feedback loop to precisely measure the true motor speed in time as long as a control loop to allow precise positioning. The motor could slow down due to load variations, frictions, wear, aging... Plus, it would be difficult to define when to begin the deceleraton phase without fully characterizing the system.

To counter this, this ramps proposition is based on the position instead.
Due to the nature of this concept, the system regulates itself automatically (but is not able to fight for load variations).

4.2 Naive ramp profile

To get a complete profile, the acceleration + deceleration phases must complete within the smallest distance available between two target positions. Here, this distance is 4cm from position p1 (80mm) to p2 (120mm).
So, the ramps should use around 1cm (to be visible enough) to under 2cm (to leave time for the full-speed to also kick-in) to complete.

The naive ramp is shown under Figure 18:

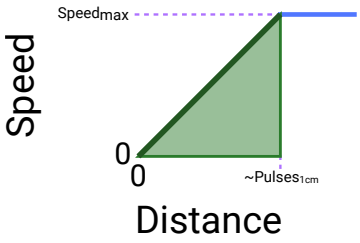

Figure 18 - Naive acceleration ramp

To calculate the speed, a linear equation representing the speed as a function of the position is used:

$$v_x = m * x + b$$

with

$$m = \frac{\Delta y}{\Delta x} = \frac{\text{speed}_{max} - 0}{\sim\text{pulses}_{for\_1\_cm} - 0}$$

$$b = 0$$

To simplify all calculations and not require hardware multipliers and dividers, only power of twos are used as arguments of the function:

- speed_{max} corresponds to the PWM resolution, i.e. 2^{pwmBitNb} with pwmBitNb $= 8$ by default
- $\sim \text{pulses}_{for\_1\_cm}$ depends how pulses are counted:
  - Using only one encoder, i.e. $500\frac{\text{pulses}}{\text{revolution}}$:
    $\sim \text{pulses}_{for\_1\_cm} = \frac{1\text{ cm}}{3.5\frac{\mu m}{\text{pulse}}} = 2'857\text{ pulses} \approx 2^{12} = 4'096\text{ pulses} \rightarrow 4'096\text{ pulses} * 3.5\text{ µm} = 1.425\text{ cm}$
  - Using only one encoder while counting both edges, i.e. $1'000\frac{\text{pulses}}{\text{revolution}}$:
    $\sim \text{pulses}_{for\_1\_cm} = \frac{1\text{ cm}}{1.75\frac{\mu m}{\text{pulse}}} = 5'714\text{ pulses} \approx 2^{13} = 8'192 \rightarrow 8'192\text{ pulses} * 1.75\text{ µm} = 1.425\text{ cm}$
  - Using both encoder signals, i.e. $2'000\frac{\text{pulses}}{\text{revolution}}$ (QEI fashion):
    $\sim \text{pulses}_{for\_1\_cm} = \frac{1\text{ cm}}{875\frac{nm}{\text{pulse}}} = 11'428\text{ pulses} \approx 2^{14} = 16'384 \rightarrow 16'384\text{ pulses} * 875\text{ nm} = 1.425\text{ cm}$

The rest of this proposition is based on the QEI method, the most precise of the three.

With those variables defined, the slope parameters are:

$$m = \frac{\text{speed}_{max} - 0}{\sim\text{pulses}_{for\_1\_cm} - 0} = \frac{2^8}{2^{14}} = \frac{1}{2^6} = \frac{1}{64}$$

$$b = 0$$

This means the slope would take the form : $v_x = \frac{x}{2^6}$.
Division by a power of two can be simplified by a right shift: $v_x = x >> 6$

Since the slope multiplier m is smaller than 1 and the circuit only handles integers, the speed curve will increase only every 2^N pulses. This means the speed will increase in steps, not smoothly. This is acceptable for this application. See Figure 19 for illustration.

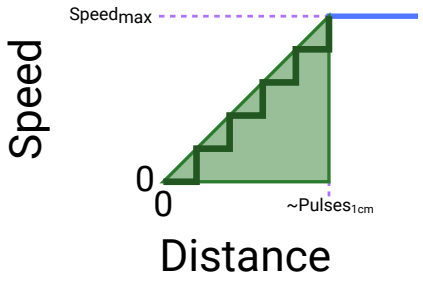

Figure 19 - Naive acceleration ramp - stepped effect

4.3 Naive solution issues and improvements

A problem still lies in this approach: at the beginning of the movement, the speed is 0. The motor is not powered and will not start moving. If the equation depended of the time, the curve would slowly progress and start the motor. Here, the position does not change if the motor is not moving, so the speed remains 0 while X is 0.

To avoid those problems, a minimal offset is added to the equation as shown in Figure 20:

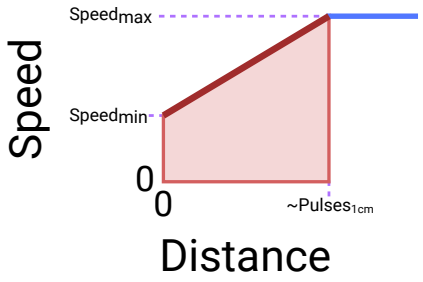

Figure 20 - Modified acceleration ramp

The equation then becomes:

$$v_x = m * x + b$$

with

$$m = \frac{\Delta y}{\Delta x} = \frac{\text{speed}_{max} - \text{speed}_{min}}{\sim\text{pulses}_{for\_1\_cm} - 0}$$
$$b = \text{speed}_{min}$$

speed_{min} depends on "how much power" is required for the motor to move, impacted by: the motors used, their wears, mechanical constraints, PWM implementation, driving electronic ... requiring experimentation.

For the example, speed_{min} of $2^6 = 64$ is used.

With those variables defined, the slope parameters are:

$$m = \frac{\text{speed}_{max} - \text{speed}_{min}}{\sim\text{pulses}_{for\_1\_cm} - 0} = \frac{2^8 - 2^6}{2^{14}} = \frac{192}{16384}$$

Such m value is not easily implementable in hardware without multipliers/dividers. It is possible to try and approximate this factor through power of twos. To keep it simple, the same slope as calculated before is used. An offset is simply added to counter low X positions speed values.

Doing such approximations implies to always re-calculate basis hypothesis once all parameters are set. In this case, since the slope is approximated, one must ensure the acceleration ramp is not becoming "too quick" (see Equation 1).

4.4 Final ramp

The final equation is now:

$$v_x = m * x + b = \frac{x}{2^6} + 2^6 = (x >> 6) + 64$$

This equation should only be applied for the acceleration phase then limited to the maximal PWM value once reached.

Example

Assuming the cursor is at position X = 0 and must reach point P1:

- the acceleration phase is done between $X_{\text{trolley}} - X_0$ until the PWM reaches 255
- follows the full-speed phase until $X_{\text{P1}} - X_{\text{trolley}}$ gives a PWM value below 255
- the deceleration phase begins until the target position is reached, at which point the motor must be stopped

4.5 Hypothesis verification

Since the previous slope has been kept, the motor will be at full speed before reaching the predefined distance as shown in Figure 21 (blue curve):

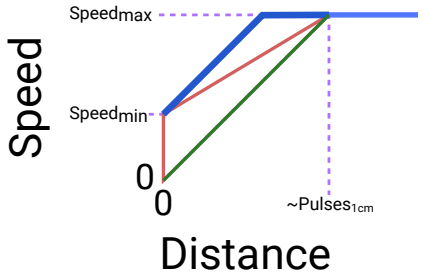

Figure 21 - Final acceleration ramp

This value can be calculated as:

$$x = \frac{v_x - b}{m} = \frac{v_x - b}{\frac{\text{speed}_{max}}{\sim\text{pulses}_{for\_1\_cm}}} = \frac{(v_x - b) * (\sim\text{pulses}_{for\_1\_cm})}{\text{speed}_{max}} = \frac{(2^8 - 2^6) * 2^{14}}{2^8} = \frac{3'145'728}{256} = 12'288 \text{ pulses}$$

$$\rightarrow 12'288 * 875 \text{ nm} = 1.075 \text{ cm}$$

Equation 1 - Final acceleration distance

The minimal distance between two points being 4cm (p1 - p2), the acceleration + deceleration curves take 2.15cm in total, leaving 1.85cm at full speed. Curves never have to be "cut" in order to reach the target positions.

Note that if the speed_{min} would have to be set higher due to motor characteristics, the slope m would have to be reduced accordingly to still fit within the 1cm to 2cm ramp distance requirements.

From the time perspective, as the curve depends on the position, it will not look linear. Since low X positions have low motor speeds and so take longer time to move rather than higher X values, the curve will appear rather exponential like in Figure 22.

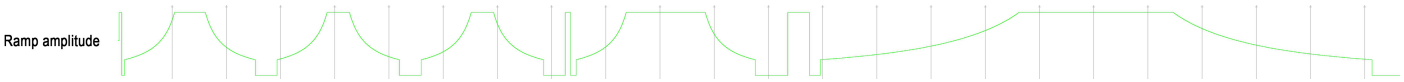

Figure 22 - Motor ramp as a function of time

5 Evaluation

In the **doc/** folder, the file **evaluation-bewertung-cursor.pdf** shows the detailed evaluation scheme, Table 1.

The final grade includes the report, the code as well as a presentation of your system.

Evaluated aspects	Points
Report	**55**
Introduction	3
Specification	5
Project	20
Verification and validation	10
Integration	9
Conclusion	3
Formal aspects of the report	5
Functionality of the circuit	**30**
Quality of the solution	**10**
Presentation	**10**
Total	**105**

Table 1 - Evaluation grid

The evaluation grid already gives indications about the structure of the report. For a good report, consult the document “How to write a project report” [14].

6 First steps

To start the project, proceed as follows:

- Read the specifications and information above carefully.
- Check the hardware and test the pre-programmed program.
- Browse through the documents in the **`doc/`** folder of your project.
- Develop a detailed functional diagram. You should be able to explain the signals and their functions.
- Implement and simulate the different blocks.
- Test the solution on the printed circuit board and find any errors 🕷.

6.1 Tips

Here are some additional tips to avoid problems and time loss:

- Divide the problem into different blocks, use the empty Toplevel document (**`cursor-toplevel-empty.pdf`**) for this. It is recommended to have a balanced mix between the number of components and the size/complexity of the components.
- Analyze the different input and output signals, for this it is recommended to use the data sheets.
- Follow the DiD chapter "Methodology for the development of digital circuits (MET)" when creating the system [15].
- It is recommended to realize the system incrementally, for example:

Don't forget to have fun.

Glossary

***CPR* – Counts per Revolution** 9

***FPGA* – Field Programmable Gate Array** 5, 7, 10, 11

***LCD* – Liquid Crystal Display** 3, 11

***LED* – Light Emitting Diode** 5, 7, 11

***PCB* – Printed Circuit Board** 7

***PWM* – Pulse Width Modulation** 5, 6, 8

***UART* – Universal Asynchronous Receiver Transmitter** 10

***USB* – Universal Serial Bus** 10

Bibliography

[1] "Reed Relay." Dec. 05, 2020. Accessed: Nov. 24, 2021. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Reed_relay&oldid=992433034

[2] Olivier Walpen, "Schematic: Cursor Chariot Power Circuit." 2009.

[3] STMicroelectronics, "Datasheet: DMOS Dual Full Bridge Driver with PWM Current Controller." 2003.

[4] Agilent Technologies, "Datasheet Agilent AEDB-9140 Series Three Channel Optical Incremental Encoder Modules with Codewheel, 100 CPR to 500 CPR." 2005.

[5] "Magnetic-Reed-Switch-Above-Closed-and-open-reed-switch-in-response-to-magnet-placement.Png (850×345)." Accessed: Nov. 24, 2021. [Online]. Available: https://www.researchgate.net/profile/Sidakpal-Panaich-2/publication/51169357/figure/fig1/AS:394204346896388@1470997048549/Magnetic-reed-switch-Above-Closed-and-open-reed-switch-in-response-to-magnet-placement.png

[6] Silvan Zahno, "Schematic: FPGA-EBS v2.2." 2014.

[7] Xilinx, "Spartan-3 FPGA Family." Accessed: Nov. 20, 2021. [Online]. Available: https://www.xilinx.com/products/silicon-devices/fpga/spartan-3.html

[8] Xilinx, "Datasheet Spartan-3E FPGA Family." 2008.

[9] CTS, "Datasheet CTS Model CB3 & CB3LV HCMOS/TTL Clock Oscillator." 2006.

[10] A. Amand and S. Zahno, "FPGA-EBS3 Electornic Technical Documentation." 2022.

[11] Silvan Zahno, "Schematic: Parallelport HEB LCD V2." 2014.

[12] Sitronix, "Datasheet Sitronix ST7565R 65x1232 Dot Matrix LCD Controller/Driver." 2006.

[13] Electronic Assembly, "Datasheet: DOGM Graphics Series 132x32 Dots." 2005.

[14] Christophe Bianchi, François Corthay, and Silvan Zahno, "Comment Rédiger Un Rapport de Projet?." 2021.

[15] François Corthay, Silvan Zahno, and Christophe Bianchi, "Méthodologie de Conception de Cicuits Numériques." 2021.