Introduction to Digital Systems (21L)

ELab4: Static Random Access Memory (SRAM)

Start: 26.04.2021 **Deadline:** 12.05.2021, 23:59 **Q&A:** 7.05.2021, 16:15 on Teams

ELab4 movies: https://youtube.com/playlist?list=PL_WyEcCQBujQ-sHsSFSpfEgJZVzR_Ju0V

Please use the draft of the report available on Teams: `Files\ELab4\ELab4_report.docx`

Points to score: 4 + 1 extra

To complete this laboratory you will need to send 3 files:

- a report, that should be downloaded from Teams Files
  and sent in **pdf format** under the name [lastname]_[index]_**report.pdf**
  via Teams Assignment **ELab4: Report**.
  Please do not edit the report in Teams Files, download it locally on your computer.
- Falstad circuit file for **Falstad1 Task**
  named [lastname]_[index]_**F1**.circuitjs.txt
  and sent via Teams Assignment **ELab4 (F1): SRAM and tri-state buffer**.
- Falstad circuit file for **Falstad2 Task**
  named [lastname]_[index]_**F2**.circuitjs.txt
  and sent via Teams Assignment **ELab4 (F2): SRAM (read-only) and multiplex display**.

All the specified files should be sent separately (not zipped) via appropriate Teams Assignments. Do not forget to click the button "Turn in" to send files. Attachments that do not meet the requirements will not be accepted. We hope this information is consistent with the movies, however, if any differences, this document is more valid.

Good luck!

Questions (0.6 pts)

Please answer the questions from the report (it is good to check them before watching the movies, you will save some time).

Falstad tasks (3.4 pts + 1 bonus)

Falstad1. SRAM and tri-state buffer (1 pt + 0.7 bonus)

Basing on the movie (Pt 2) build the circuit, with `SRAM` and `Tristate Buffers`, that can be either in write-to-SRAM mode or in read-from-SRAM mode. In the circuit that you will return on Teams you should store your Student Number. Under address 0 of `SRAM` write the unity digit of your Student Number. E.g. if your Student Number is 8452, then:

Number to store	**SRAM address**	0	1	2	3
8452	**SRAM data**	2	5	4	8

The requirements for your circuit (very similar to the version from the movie)

- data input
  i. as many bits as you need to store digits 0-9,
  ii. the least significant bit (LSB) should be assembled as "the real one" with `Switch`[1], LED indicator, etc.
  iii. the remaining bits should be simple `Logic Inputs`
  iv. data input should be provided on `SRAM` data pins through `Tristate Buffers`
- SRAM address
  i. `Logic Inputs`
  ii. number of address bits should be as low as possible to store your Student Number (number of bits can be changed by right mouse click on `SRAM` in `Edit` option),
- Write/read modes of `SRAM` and enabled/disabled modes of `Tristate Buffers` should be controlled with one `Push Switch`[2].
- indicator: `7-segment LED display` that will display the data that you either selected on the input (in write mode) or the data that you read from `SRAM` (in read mode),
- you can use any additional Falstad elements that you need to accomplish the task,
- name your file [lastname]_[index]_**F1**.circuitjs.txt and send it via Teams Assignment **ELab4 (F1): SRAM and tri-state buffer**,
- and fill the report.

Extra task (0.7p): Instead of simple `Logic Inputs` at the SRAM address, assemble the possibility to automatically read your Student Number

- SRAM Address bits should be connected to a counter
- The counter should be driven
  i. either by a clock of proper frequency (to comfortably read all the values stored),
  ii. either by manual pulses created with a `Push switch`,
  iii. the mode of operation manual/automatic should be set with a `Switch`.
- The counter should reset automatically after reading all the numbers of your Student Number
- Use the same Falstad file [lastname]_[index]_**F1**.circuitjs.txt.

See the next page!

[1] In Falstad element called `Switch` is a bistable switch, that changes its state (short/open) when clicked with a mouse.
[2] In Falstad element called `Push switch` is a momentary switch, that shorts when clicked by a mouse, and opens when click released.

Falstad2. SRAM (read-only) and multiplex display (2.4p)

You start with SRAM memory that has stored the data in:

```
0: 60 66 169 133 133 169 66 60
```

You should paste that data in the properties of SRAM (right-click on SRAM, Edit\Contents) and also change there the numbers of address and data bits as in Figure 1.

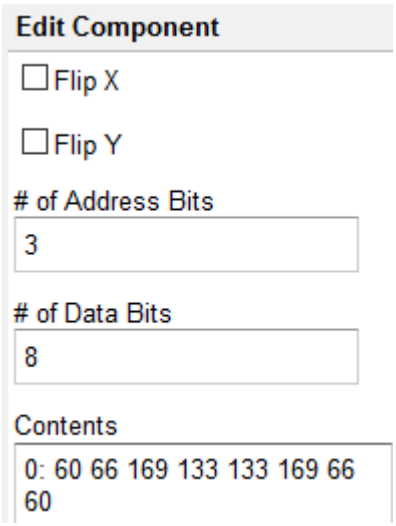

Figure 1. Properties of SRAM that you start with.

The requirements for your circuit:

- to decode the received data you will need an LED array.
- The output of SRAM should be connected to the ROWS of the LED array.
  SRAM D0 to the bottom row, D7 to the top row.
- COLUMNS of the LED array should be connected to hmmm – that's your task – to what? We hope after watching the movies the idea of multiplex display is clear (and you've learned a new component that can be helpful).
- The decoded picture has to be displayed **automatically**, so you will need…
- …a clock signal and something more. The speed of the clock should be adjusted experimentally if needed.
- You can use any Falstad elements that you need.
- Name your file [lastname]_[index]_**F2**.circuitjs.txt and send it via Teams Assignment **ELab4 (F2): SRAM (read-only) and multiplex display**.
- Paste in the report your decoded message.

Extra task (0.3p): Add master reset (with Push switch), which will print the message from the beginning. Use the same Falstad file [lastname]_[index]_**F2**.circuitjs.txt.

We strongly encourage you to fill the **surveys on USOS** – they are fully anonymous and will be available at the end of the semester. Remember that when you won't write your comments, we won't be able to change anything, because we don't know about it!