Digital Circuits Theory - Laboratory						
Academic year	Laboratory exercises on	Mode of studies	Field of studies	Supervisor	Group	Section
2020/2021	Wednesday 15:30 – 17:00	SSI	Informatics	KP	1	3

Report from Exercise No 9

Performed on: 21.10.2020

Exercise Topic: Registers

Performed by:

Dawid Grobert

Purpose of the exercises

The aim of the classes was to learn about the registers – very useful digital circuits used to store binary information. Using the knowledge about Flip-Flops, and using a few additional combination gates, our task was to prepare digital circuits based on different register patterns, as well as to design several own solutions for the purpose of getting more familiar with the subject.

1 Description of the first task

Task 1

Obtain a 4-bit parallel-in parallel-out register with loading information in one asynchronous stage.

1.1 Circuit diagram

During the classes I managed to create a particular digital circuit:

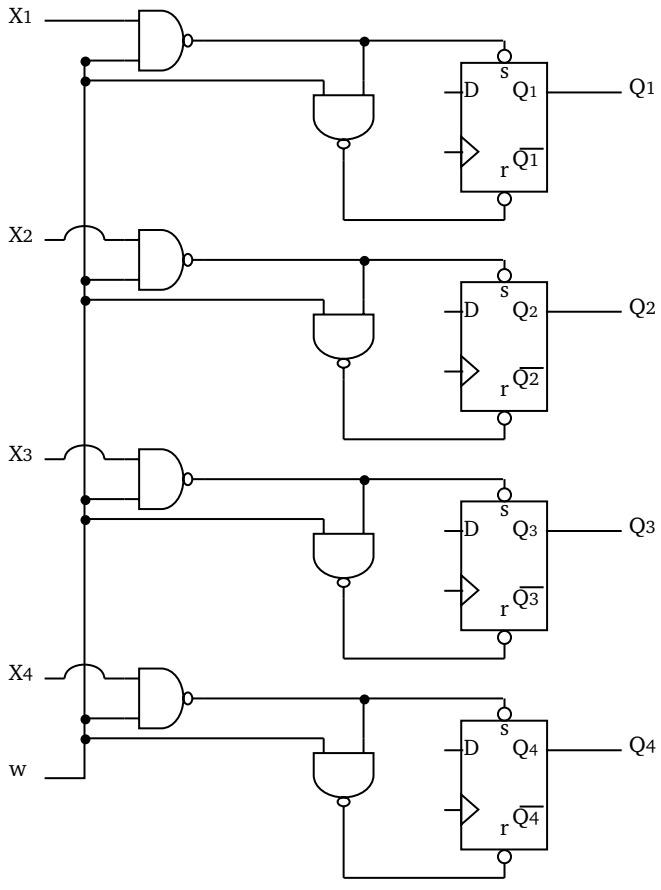


Figure 1: Digital circuit made during laboratories

The w input serves as a kind of "write permission". If w is set to 1, it allows to change the Flip-Flop values, and if w is set to 0, it blocks such a change, so that the values are in some way "read-only". When w is set to 1, the given circuit enters the information from inputs x_1, x_2, x_3, x_4 into the register and the entered information overwrites the previous registered content.

2 Description of the second task

Task 2

Obtain a 4-bit parallel-in parallel-out register with loading information in two asynchronous stages.

Another digital circuit that I managed to prepare had an additional Z input. It was a fundamental difference between the first task where "loading" of information took place in one asynchronous stage, and here in two asynchronous stages. We can see that it reduced the number of gates by half.

2.1 Circuit diagram

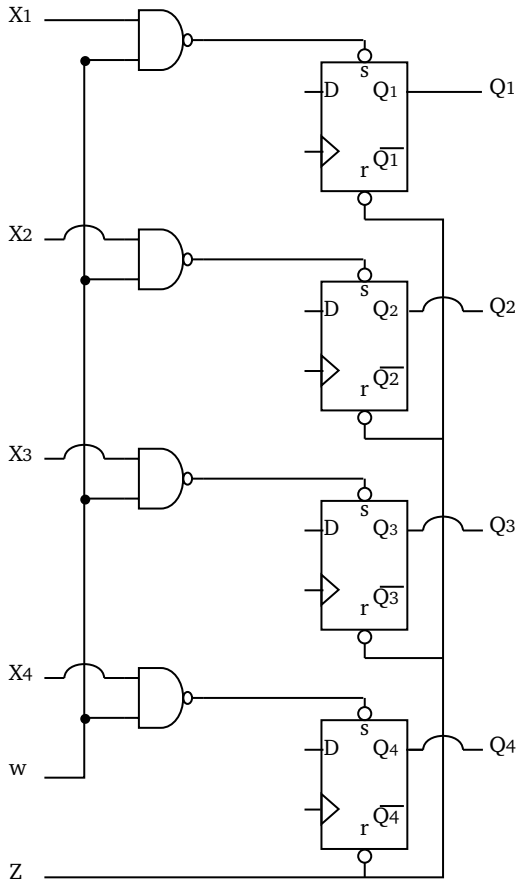


Figure 2: Digital circuit made during laboratories

This time the w signal seemed to work in a similar way – it also blocked and allowed writing to the register. The main difference when entering the data was that the data could not be overwritten so easily. The logical ones could no longer be overwritten with zero. This is where the Z is used, which works in such a way that it resets the saved data by setting all of them to logical zero. Therefore, when entering data, you should first reset the data stored in the Flip-Flop by setting Z to low and then back to high (as long as you do not want to perform logical OR operations on binary numbers).

3 Description of the third task

Task 3

Obtain a 4-bit self-setting ring register counter with the effect of “circling 1”.

This time the asynchronous circuit will no longer be built, but the synchronous circuit, with the circulating one. This means that the system should use the clock, and according to the description it should work in the following way:

$$0001 \longrightarrow 0010 \longrightarrow 0100 \longrightarrow 1000 \longrightarrow 0001$$

The self-correction system in this case makes sure that setting the wrong initial state of the registry will not cause it to fall out of the correct cycle and thus spoil the operation of the digital system. For example setting 0101 should lead to:

$$0101 \longrightarrow 1010 \longrightarrow 0100 \longrightarrow 1000 \longrightarrow 0001$$

3.1 Circuit diagram

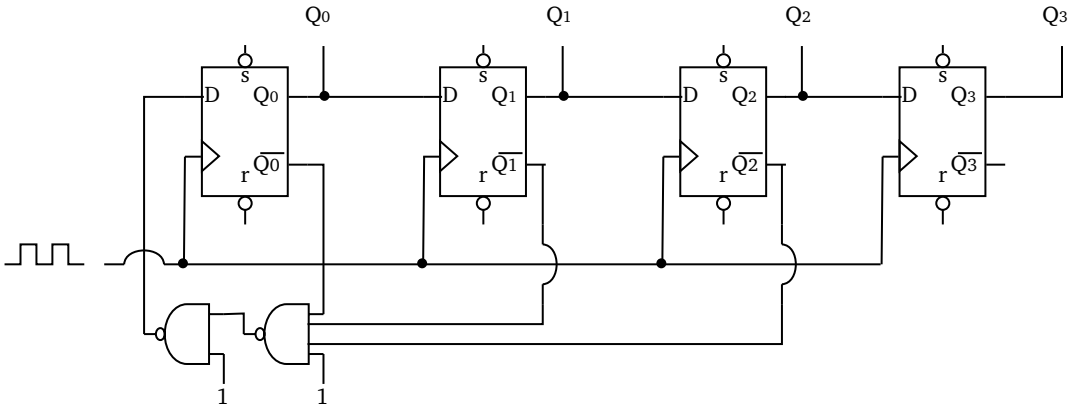


Figure 3: Digital circuit made during laboratories

3.2 Timing Chart

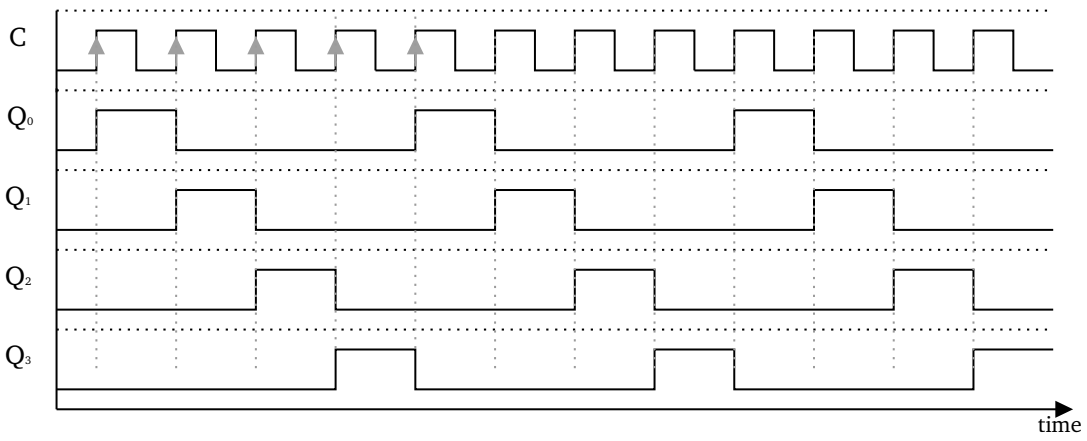


Figure 4: Timing chart of given circuit (see Fig. 3)

4 Description of the fourth task

Task 4

Obtain a 4-bit self-setting ring register counter with the effect of “circling 0”.

This task is very similar to the previous one. The difference is that instead of having a logical one on only one position, and a logical zero on all the others, we have the opposite situation – one logical zero and a logical one on all the others positions.

4.1 Circuit diagram

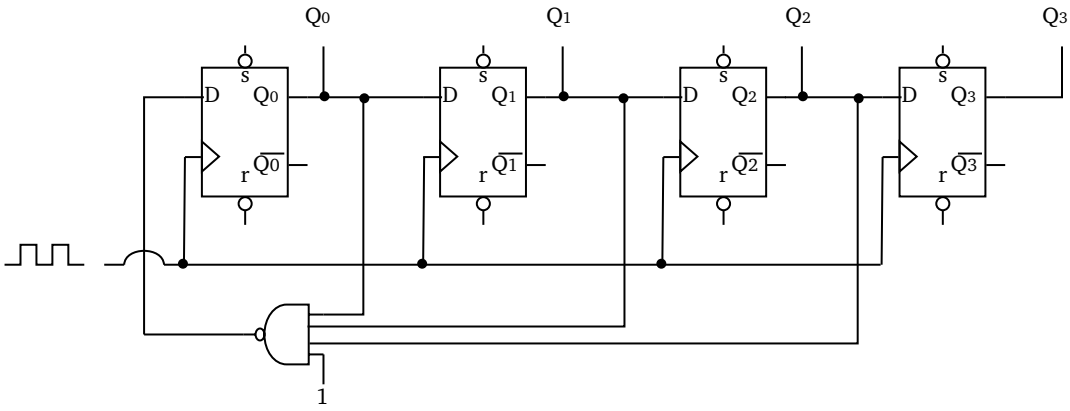


Figure 5: Digital circuit made during laboratories

4.2 Timing Chart

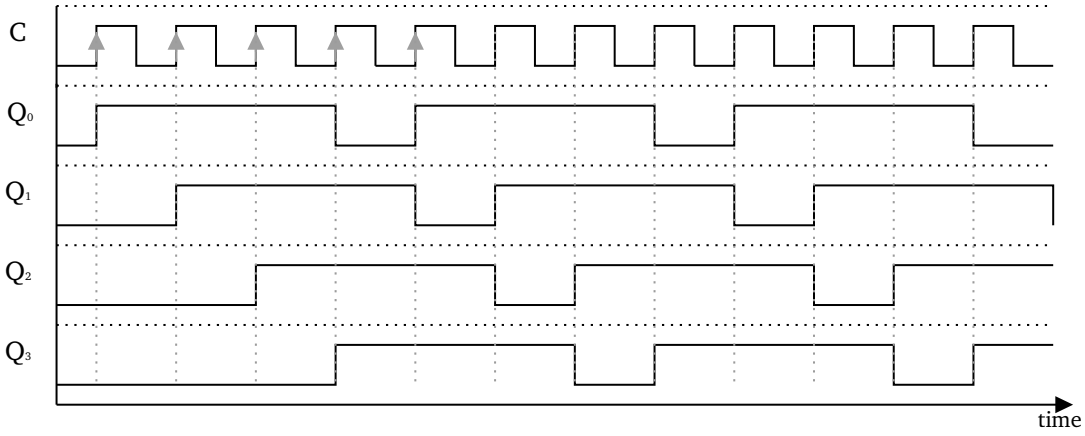


Figure 6: Timing chart of given circuit (see Fig. 5)

In the previous task the self-correction was not noted on the timing chart, so here the system started from an incorrect configuration so we can analyze how it works.

$$0001 \longrightarrow 0011 \longrightarrow 0111 \longrightarrow 1110 \longrightarrow 1101 \longrightarrow 1011 \longrightarrow 0111 \longrightarrow 1110$$

The self-correction is marked in blue. In all aspects, the timing-chart looks like a negated results of a "circling 1".

5 Description of the fifth task

Task 5a

Obtain a 3-bit linear register with the feedback function defined as $Q_1 \bigoplus Q_0$

In the case of this task, due to the lack of available XOR gates at the laboratory, it was realized using NAND gates. It took the most time to build the first digital circuit here, because every next part of the task was based only on changing the inputs of our XOR (created with the help of NANDs) so that it would perform XOR operations on the bits we want. The result of this operation always appears at the very beginning of the next output – in this case marked as Q_0.

Note that in order to perform XOR operations on the Q_1 and Q_0 outputs we connected the inputs of our NANDs specifically to the Q_1, $\overline{Q_0}$, Q_0, $\overline{Q_1}$ outputs.

Warning: In those tasks, it is also worth mentioning about illegal states that can lead to "spoiling" of the system. For example, setting this specific digital circuit ($Q_1 \bigoplus Q_0$) to $Q_0 = 0$, $Q_1 = 0$, will lead to eternal staying in this state ($Q_0 = 0$, $Q_1 = 0$, $Q_2 = 0$), as $Q_1 \bigoplus Q_0$ will always output zero.

5.1 Circuit diagram

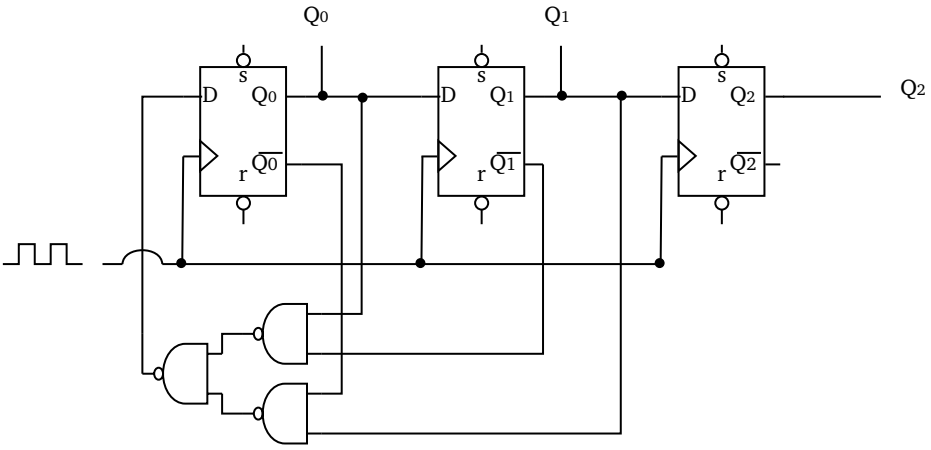


Figure 7: Digital circuit made during laboratories

5.2 Timing Chart

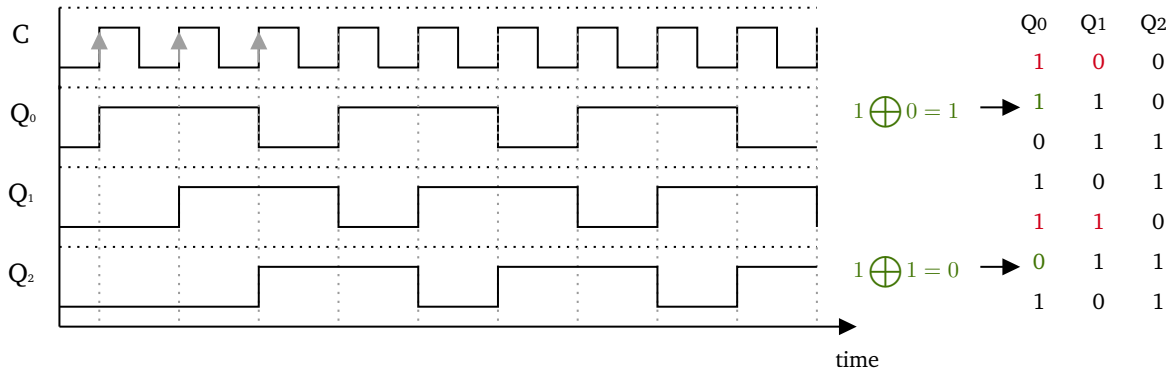


Figure 8: Timing chart of given circuit (see Fig. 7), and table showing exemplary values on the output

Task 5b

Obtain a 3-bit linear register with the feedback function defined as $Q_2 \bigoplus Q_0$

The circuit was almost unchanged compared to the previous task. Similarly, to perform the XOR operations for Q_2, and Q_0 we used the outputs Q_2, $\overline{Q_0}$, Q_0, $\overline{Q_2}$, to connect the NANDs.

It is also worth adding that here an illegal setting would be $Q_0 = 0$, $Q_1 = 0$, $Q_2 = 0$. This setting would last forever.

5.3 Circuit diagram

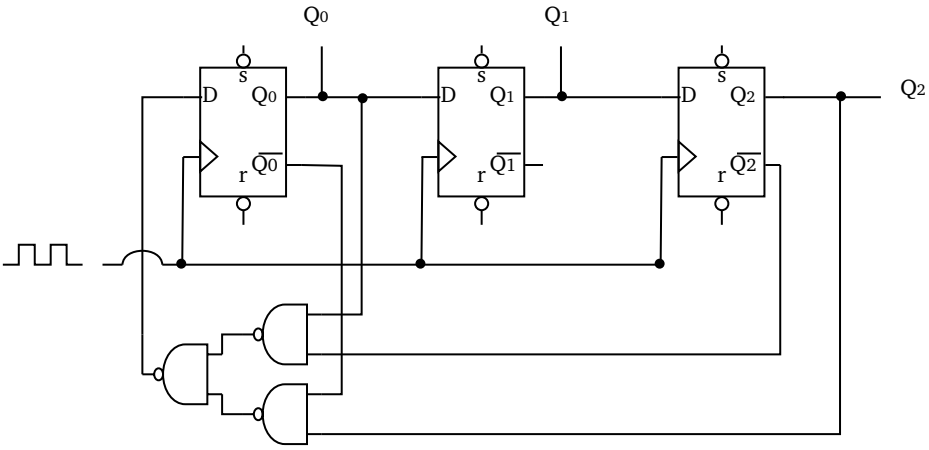
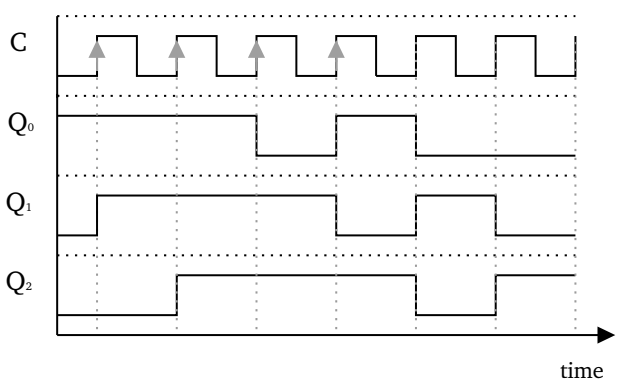


Figure 9: Digital circuit made during laboratories

5.4 Timing Chart



	Q0	Q1	Q2
	1	0	0
$1 \bigoplus 0 = 1$ →	1	1	0
	1	1	1
	0	1	1
	1	0	1
$1 \bigoplus 1 = 0$ →	0	1	0
	0	0	1

Figure 10: Timing chart of given circuit (see Fig. 9), and table showing exemplary values on the output

Task 5c

Obtain a 3-bit linear register with the feedback function defined as $Q_2 \bigoplus Q_1$

Again, the only change that has been made is the connection of NAND gates, which have to simulate XOR operation for Q_2 and Q_1. To achieve this, their inputs have been connected – no surprise, to the inputs: Q_2, $\overline{Q_1}$, $\overline{Q_2}$ and Q_1.

Here again, it would be wrong to set values: $Q_0 = 0$, $Q_1 = 0$, $Q_2 = 0$. This is an illegal state.

5.5 Circuit diagram

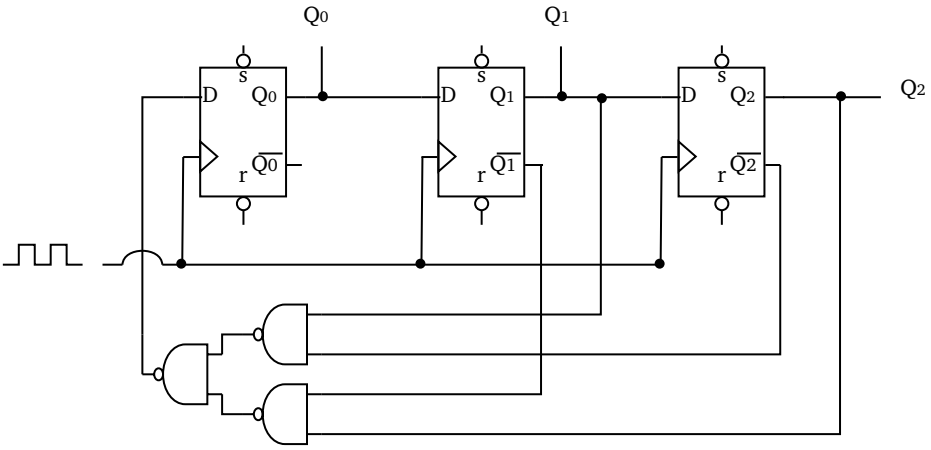
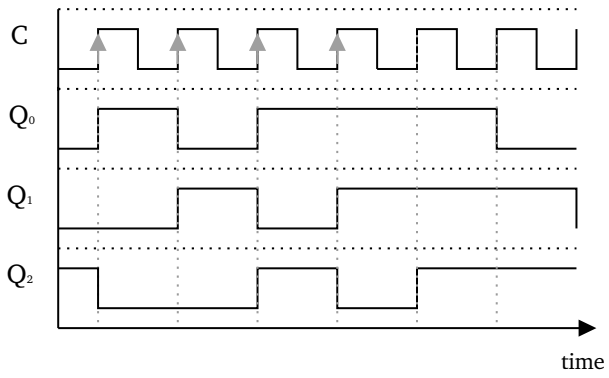


Figure 11: Digital circuit made during laboratories

5.6 Timing Chart



Q0	Q1	Q2
0	0	1
1	0	0
0	1	0
1	0	1
1	1	0
1	1	1
0	1	1

Figure 12: Timing chart of given circuit (see Fig. 11), and table showing exemplary values on the output

6 Conclusions

Luckily, all the tasks were completed in the laboratory. From the very beginning, when I was creating a digital circuit from Task 5 (see page 5), I completely forgot about the illegal states that can spoil it. It is important to be careful and vigilant about building digital circuits. Most of the tasks were carried out without any major problems, and the proper organization of work and cables in Task 5 allowed for quick realization of the next sub-tasks.