Project in TFE4152 Design of Integrated Circuits

AS 2024

Fall 2024

1 Introduction

The work is to be performed in groups of two (2) students. You have to sign up to one of the groups on Blackboard together with another student. Please find a partner BEFORE you sign up to the group so that we don't have to spend a lot of time removing people from groups they no longer intend to be in. Feel free to use the "Looking for a project partner"-forum on Blackboard to find someone to work with (either respond to an existing post in the forum or create your own post). You find a link to the forum under *Forums* on the TFE4152 Blackboard page.

Both students must contribute to the project. If your partner is not contributing, please contact Snorre and/or Asta as soon as possible and we will address the issue.

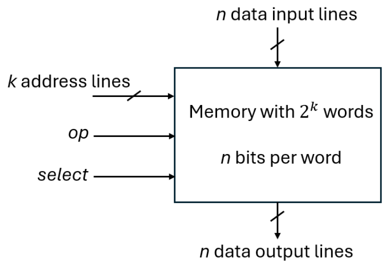
2 The project

You are employed in a start-up IoT-company working on designing its own sensor unit. The unit is in need of a small memory array containing 8 words, with each word having 8 bits, and it is your task to design it. This equates to a 8x8 bit memory, which means it should contain a total of 64 bits (/bitcells). The gpdk 90 nm technology is to be used. Design your memory having knowledge relevant to TFE4152 in mind. Both AimSpice and Verilog simulations in ActiveHDL are required to complete the task.

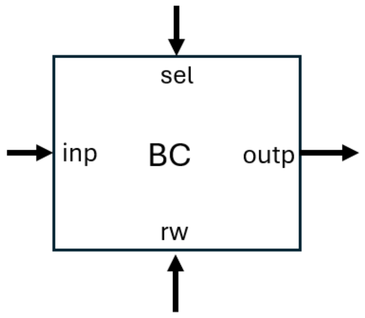
Your work should culminate in a written report, with all material delivered in a single PDF-file. More on that in subsection 3.2.

2.1 Design

Your memory unit should adhere to the block diagram in Figure 1a. Each word in the memory should have a particular address, starting from 0 up to $2^k - 1$. An internal decoder uses the address to select the n-bit word specified, whether a word should be read from or written to the memory. The *op* signal dictates which operation should be run. When the binary *op* input is logic 0, a particular word may be read from memory, depending on the address. $op = 1$ indicates a



(a) Memory unit.



(b) The bitcell (BC).

Figure 1: Block diagram of the memory, and the bitcell. Use $n = 8$, and $k = 3$.

write operation. *select* is used to indicate whether the command (read or write) is valid, and will be used as an input to your FSM (described later). For the memory unit, no additional input- or output-signals should be used. (This also means that no clock signal is used for the memory itself, while such is likely to be included in systems who could use it as a component.)

Each 1-bit memory element, from here on called the bitcell ("BC"), should have input and output signals as shown in Figure 1b. It is the *rw*-signal that decides between the read- or write operation, with $rw = 0$ for read and $rw = 1$ for write. The bitcell is ready for read- or write operation only when the binary value $sel = 1$. The *sel* signal should be created by a decoder, based on the address input signal and the *valid* signal (*valid* is created by your FSM, described further down). The arrows in Figure 1b indicate the directions of the binary signals. Read operations make use of the *outp* signal, whereas write operations get information through the *inp* connection.

2.1.1 There are 3 main design tasks for this project:

The first task is to design a 1-bit memory/bitcell, depicted in Figure 1b, using boolean gates only. The properties of the bitcell should also be demonstrated through AIMSpice and the 90 nm gpdk parameters. Further specifications are given in subsubsection 2.1.2. A possible starting point for the bitcells is to use a simple SR latch based on two 2-input NAND gates, but you are free to choose other designs as well as long as your bitcell is constructed purely from logic gates and interconnects.

The second main task is to construct an 8x8 bit memory built upon the use of 1-bit memory from the first main task as well as an address decoder.

The bitcell as well as the 8x8 bit memory should be built from Boolean gates and interconnect only, and demonstrated by using the Verilog option in ActiveHDL.

The third main task is to design a Finite State Machine, FSM, using gate level Verilog. Your FSM should be able to control the reading of information from, and writing of information to, your memory. You must verify both that the FSM works on its own and as part of the overall memory system. Your boss has already drawn up a state diagram for the FSM, which you can find in Figure 2. Your system is controlled by two external signals *op* and *select*. The signal *op* dictates which operation the memory should run, with $op = 1$ meaning write and $op = 0$ meaning read. The input signal *select* is 1 if the memory has been selected for either read or write, and 0 otherwise. The outputs of the FSM are the *valid* and *rw* signals needed to control your bitcells, as previously described.

As an extra measure to ensure that all values in the memory are stable after a write operation, a mandatory STABLE state is used. For this state, *valid* is set to 0 as there is no valid memory operation, but *rw* remains 1 (indicating a write operation). Note that the FSM in Figure 2 is a Moore machine, which means that the output only depends on the current state (not on the inputs). 'X' is used to indicate 'don't care'.

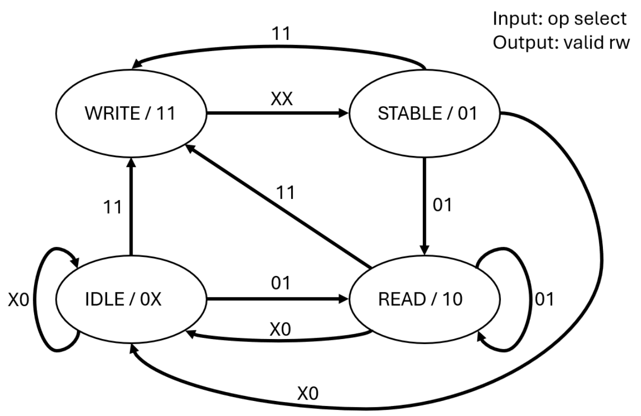


Figure 2: State diagram for the FSM.

You are advised to design your memory in a hierarchical fashion, and include simulations demonstrating the functionalities of any subcomponents in the memory.

Use the following naming convention and order of signals, as presented below:

```
module<name> (i0,i1,i2,i3,i4,i5,i6,i7,adr2,adr1,adr0,op,select,
o0,o1,o2,o3,o4,o5,o6,o7);
```

2.1.2 Further specifications for the memory

These specifications are relevant for your general design and the implementation of it in AIMSpice.

Supply voltage: 0.3V $< VDD <$ 1V

Technology: 90 nm technology
Transistor dimensions: $100\text{nm} < W < 1500\text{nm}$ and $100\text{nm} < L < 300\text{nm}$
You should design your circuit for low power consumption.
Since your memory will be used in a larger system, your memory must adhere to some time constraints. Your colleague has calculated the delay in the rest of your memory system, and concluded that each bitcell can have a maximum read/write time of 3ns.

For the memory design in Verilog, you must fulfil the following requirement: You should only use gate level code, which shows the entire memory as connected components. This goes for both the general memory and the FSM. We recommend designing your memory in a hierarchical fashion, using the logic gates and interconnects to design modules such as flip-flops, decoders and so on that you can then use to build your memory system. Also, no logic gate is allow to have a fan-in above 4.

2.2 Simulation and verification

Aim at demonstrating the writing of a bit pattern to the 8 different lines of the memory (each line containing 8 bits), followed by reading them out. Include *at least* the simple test of writing the pattern 01010101 to one of the words in your memory, and a following read operation of the same. This can be simulated in ActiveHDL.

For the Spice simulations of the bitcell, you should include *at least* demonstrations of the following:

- The main functionalities of the bitcell for all 5 process corners (TT, FF, SS, FS, SF), with each corner tested using -20, 27, and 50 degrees Celsius.
- The leakage power and write/read-time for the TT, FF and SS corners at 27 degrees.

For the FSM, use structural level Verilog to write a testbench. The testbench should at least include the following test:

Write the ASCII equivalent of the initial of the first name (or first part) of one of the members of your group to the first address of your memory unit, and afterwards read out the same pattern. This should be simulated in ActiveHDL.

ASCII to binary conversion can be made using the following link: `https://www.rapidtables.com/convert/number/ascii-to-binary.html`. For example, the letter 'a' has the binary value 01100001, and 's' has the binary value of 01110011.

Suppose that the 8 bit pattern (your chosen letter) is ready in the register whenever there is a valid write operation.

3 Deliverables

3.1 Midterm delivery

To be handed in on Blackboard, deadline Friday 25th of October at 2pm (14:00). Will only be graded pass/fail.

This does *not* need to be a full report. The point of the midterm delivery is to show your work so far, and to help ensure that you are on track for the final project deadline. Must include:

- A brief reflection on your progress so far, and an overview of the remaining work. Which circuits do you have to design/implement? Which simulations are you planning to run? Are there any parts of the work that you have already done that you plan to improve/change?
- Testbench for your chosen bitcell, where you have simulated it's functionality on structural level using only Boolean gates and interconnect using Verilog and ActiveHDL.
- Demonstration of the functionality of your bitcell using AIMSpice. Demonstrate the functionality of the bitcell for the TT, FF, SS, SF and FS corners, for -20, 27, 50 degrees Celsius.
- A short description of each group member's contribution to the work.

Feel free to include anything else you have done in the project so far :-)

3.2 Final Report

To be handed in on Inspera, deadline Friday 22nd of November at 2pm (14:00). This work will be graded A-F, and count as 30% of your final grade in TFE4152.

The final project report should follow the structure and guidelines given in the report template, which will be posted on Blackboard.

The completeness, conciseness and quality of your solutions are important factor for the grading of the final report. Justifying and explaining your design choices is an important part of this. The final report should contain all information needed to be able to reproduce your simulations and results.

3.3 Deadlines

Note that the deadlines are absolute. Handing in too late is an immediate fail, so do not wait until the last minute. It will be possible to upload your report several times, so please upload a version of your work the day before the deadline to ensure that you have handed in something that can be graded. This goes for both the midterm delivery and the final report.

Both deadlines are set to 2pm so that you will be able to contact us and/or Orakeltjenesten if you are having technical issues when trying to hand in your work.

Good luck!