GENERAL DESIGN RULES

Eight rules below are for system design in engineering fields

They are also applicable to many situations in daily life, including **exams**

How can we deal with complex tasks ?

**WE NEED TO BE IN CONTROL OF THE TASK !**

**Rule 1 :** Remember that we are human beings, not machines! Make sure you do not increase the pressure on yourself. You already have enough pressure! Therefore, help yourself! Do not rush! Think!

The goal is controlling the design: **Are you in control ?** If you are in control (you are controlling the pace of the progress and deciding what to do next and when), you have a chance to complete the design successfully and on time since your technical knowledge and team work are sufficient. If you are not in control, the chances of completing the project successfully is slim.

The remaining rules below help you ensure that. Prepare a plan that makes use of the rules below, stick to it and improve it if necessary.

**Rule 2 :** Simplify your task right in the beginning.

Do not deal with complex ideas, choices, decisions and many details in the beginning. Give yourself time to be familiar with what you are going to do. Give yourself a chance to like what you are doing. Do not rush the design. Perhaps, you sleep on it. Remember, if you are not motivated, you will not come up with a design as good as you would like.

Ask yourself these two questions:

i) Am I familiar with the **system** I am going to design?

ii) Am I familiar with the design **process**?

If you answered “No” to any one of them, you need to **help yourself** more. There are three ways to do that:

a) Investigate about similar systems/design processes.

See what others have done similar to what you will do. Do the related work search. If related information is found, use it if allowed.

b) Dealing with a lot of details is frustrating.

To avoid many details in the beginning, use the top-down design method where you conceive of a number of “layers,” each describing the system in different detail. Also, any two neighbor layers are logically related. You order the layers such that upper layers have few details and lower layers have more. You start the design with the top layer and proceed towards lower layers. The bottom layer is the final design.

This approach relies on **abstraction** since upper layers, in their simplicity, abstract the system more. This is also the same idea as thinking in terms of blocks. Thus, think in terms of blocks. On each layer, the number of blocks (the amount of details) is different and increases as we proceed down. Once we define the blocks on a layer, we move down to the layer below and implement **each block** of the upper layer by using **a few subblocks**. Then, we move one layer below and implement each subblock by using a few sub-subblocks, and so on. When, we reach the bottom layer, each sub...subblock is a few gates, wires, etc.
c) It might be a better idea to design just a representative piece of the system. You may not be able to abstract the system. You may not be able to conceive of those layers. You may not be able to think of what each layer should be. You need to gain insight! Thus, rather than thinking about the whole large system, think of a “representative” piece which can later be expanded to the whole system quickly. Examples:

- If you are asked to design a 64-bit ADDer, first consider designing an 8-bit ADDer. Once you understand how the design progresses, you can expand the design to 64 bits easily.
- If you are asked to design a 8-bit up/down counter, first design a 2-digit up/down counter. In fact, you can first design a 2-digit up counter, then a 2-digit down counter, and then you can combine them. Again, once you gain insight into the design, the expansion of the design can be easy.

**Rule 3**: Base your design decisions on a set of design goals.
Do not make arbitrary design decisions: Be consistent. For example, do not just use a few high-speed chips here and a few slower chips there. Become familiar with design goals before you start the design and make use of these goals on each layer. Most commonly used design factors are **speed**, **cost**, **reliability**, **upgradability**, **maintenance**, **size**, **weight** and **power consumption**.

**Rule 4**: Do not go for the design of an optimal system right in the beginning.
Rules 2 and 3 imply that we design a system that works and also satisfies design goals. However, doing both simultaneously especially as a beginner designer can create too much pressure on you. In these circumstances, relax Rule 3 a bit and just use Rule 2. Once you have a working system, then try to optimize it with respect to the factors you have in Rule 3. For example, design an ADDer that works even though it is slow. Then, modify it so it is faster as required by the speed factor. This design and optimize cycle can take a number of iterations, which is fine.

**Rule 5**: Leave room for future expansions (upgrades).
Your design has to include “space” for future upgrades wherever possible. Otherwise, upgrading the system in the future can be quite difficult or impossible.

**Rule 6**: Consider several layers at a time after you become familiar with the system/process.
After you become familiar with the system you are designing and with the design process, consider how each decision you make on the current layer affects the lower layers, in terms of the factors you determine in Rule 3. You now design the system on several layers simultaneously, which is not a top-down design anymore. The advantage of using this rule is that it can avoid wasting time. It is possible that one partitions blocks in a certain way on a layer, assuming they can be easily implemented on the lower layers. But, when those layers are considered later, the designer realizes that is not the case. Then, the designer has to go back up to higher layers and do the partitioning again.

**Rule 7**: If you are stuck, do not know what to do next, stop...! “See where you are.”
If you do not know what to do next, which sub...subblock to work on next, then see the big picture: Move up to the upper layers to focus on those larger blocks and see what is next or which sub...subblock is next to implement.

**Rule 8**: If you cannot explain something, why it happens as it happens, “go for the basics.”
Move down to the lower layers: There things can be explained in a few basic things (simple sub...sub-blocks). For example, if the addition time is too slow, focus on the carryout circuits right away.