# DO's and DON'Ts for Systematically Implementing Late Engineering Changes On Your Project

Project management and automation is quickly becoming the most critical element in the overall design and verification process. The key ingredients are good overall specification development that leverages planning with metric based checkpoints. The mentality of "beginning with the end in mind" allows for optimal resource usage, much higher design quality and realistic schedule estimates. Despite all the best planning, changes in requirements happen up to the last possible day.

Today's design teams typically do not address the problems that cause slips or productivity and quality issues. They tend to focus solely on individual tasks, engine performance, or languages, rather than defining the entire verification challenge, independent of its solution. In fact, most verification plans are merely a set of incomplete discussion notes that atrophy as the project moves forward.

Change management of the design and verification process needs to start with the goals of what needs to be verified. From this, an experienced verification team will have the ability to develop changes to the plan that are complete and inclusive of the intended goals.

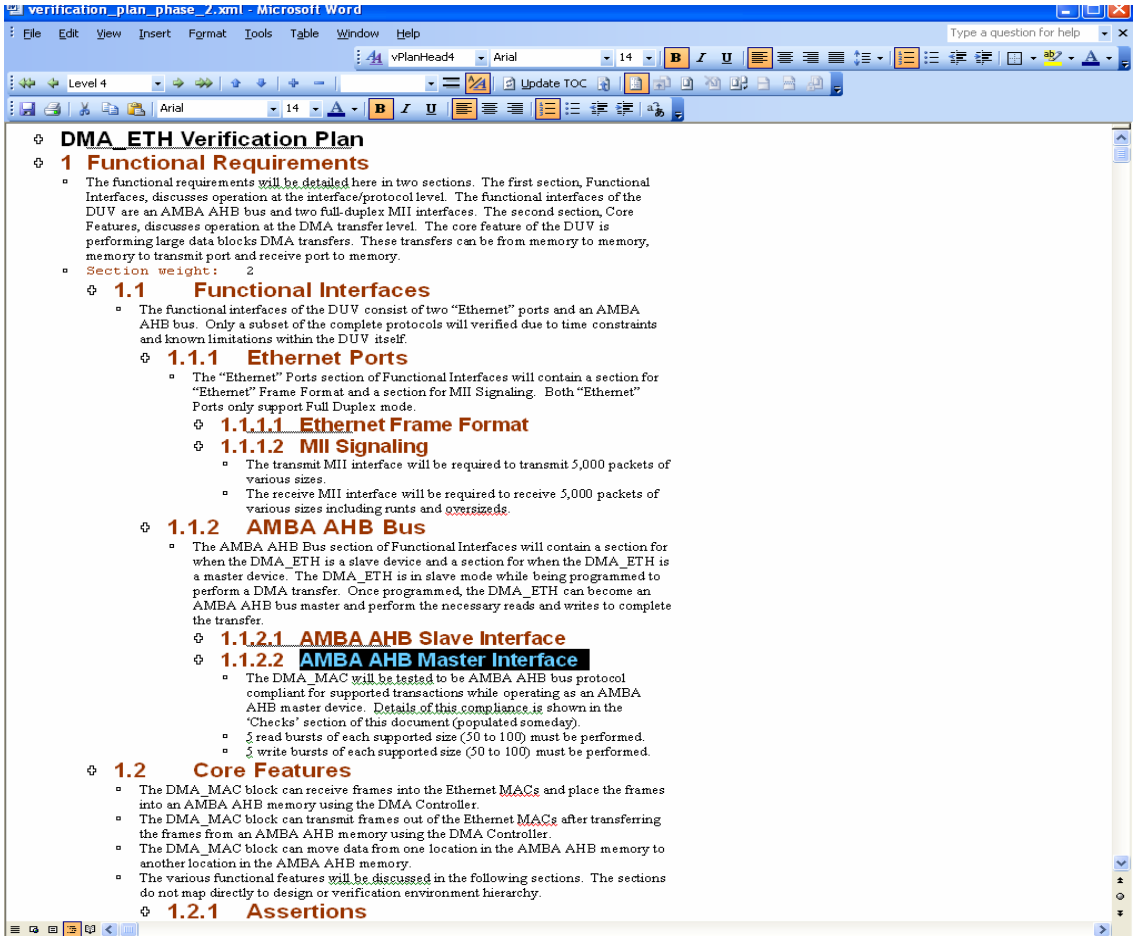


**Figure 1: An example of a Verification Plan: which features we want to verify, by what date, by whom etc.**

What is needed is a system that enables efficient introduction and tracking of changes as well as the right team approach to the required planning and implementation activities. Here are some of the basic do's and don't that you and your team needs to be aware of to introduce project changes effectively:

## Do's

- Must have a plan that is kept always up-to-date and is executable: your total coverage results after regressions will be mapped directly into your plan. This is used as the measure against which progress and completion is measured.

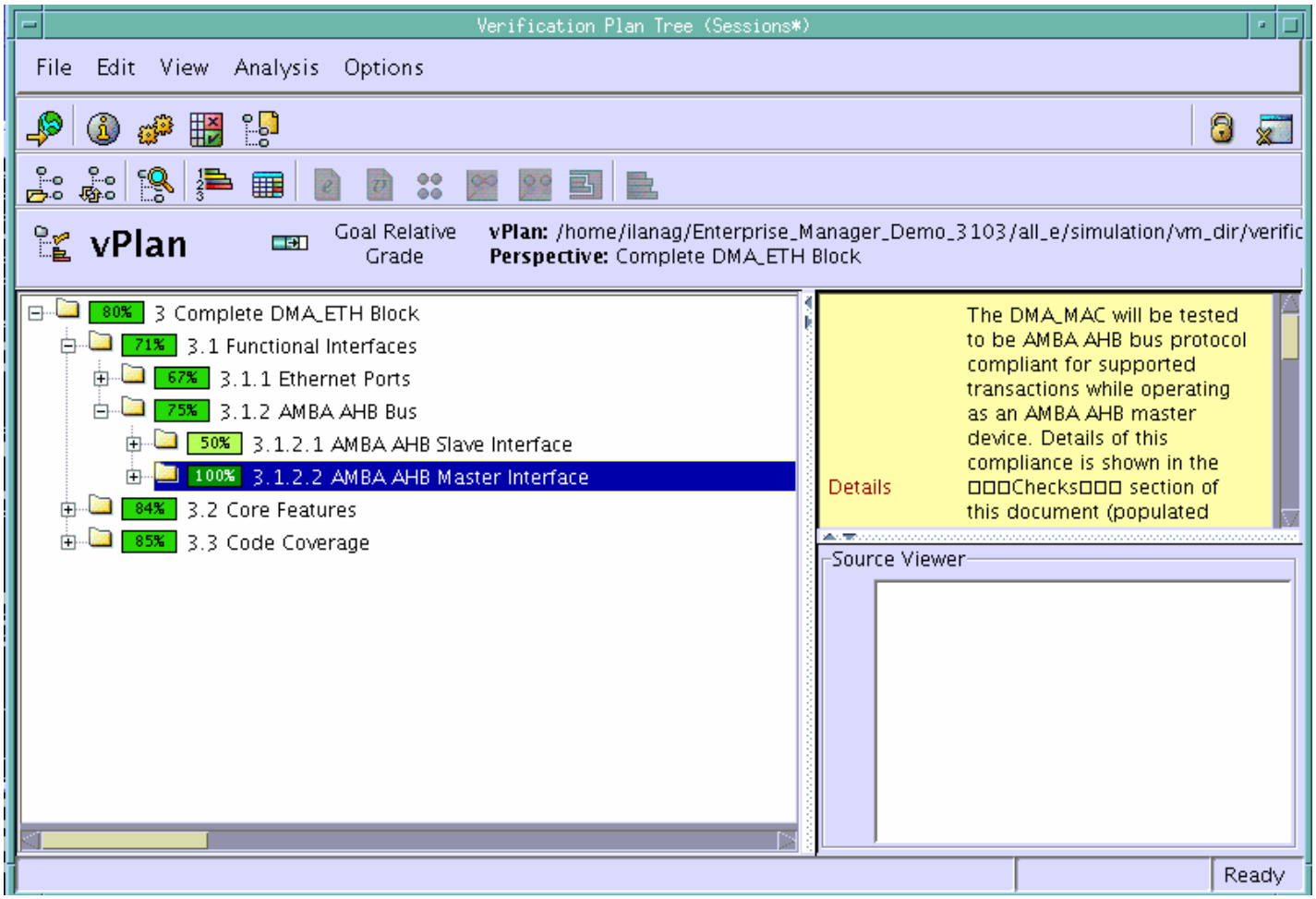


**Figure 2: An executable plan – the coverage grades are mapped to the features in the plan**

- You will need to identify all parts of the system that are impacted by each change and incorporate that into the overall plan before making any changes to the specification.

- Identify an owner of the change process that will be committed to following the process to completion. This person needs to have a good overall view of the project and maintain a high attention to detail.

- Measure the impact of changes by the scope of the change to the verification plan. Especially consider impact to various dependencies outside of your control -- such as any third-party cores; designs that may use third party soft or hard IP; or any dependencies on support from a semiconductor vendor.

- Review the effectiveness and quality of your process on a regular basis, and at the end of each project. Each change forced by errors in the past should be used as a learning experience and categorized for future projects.

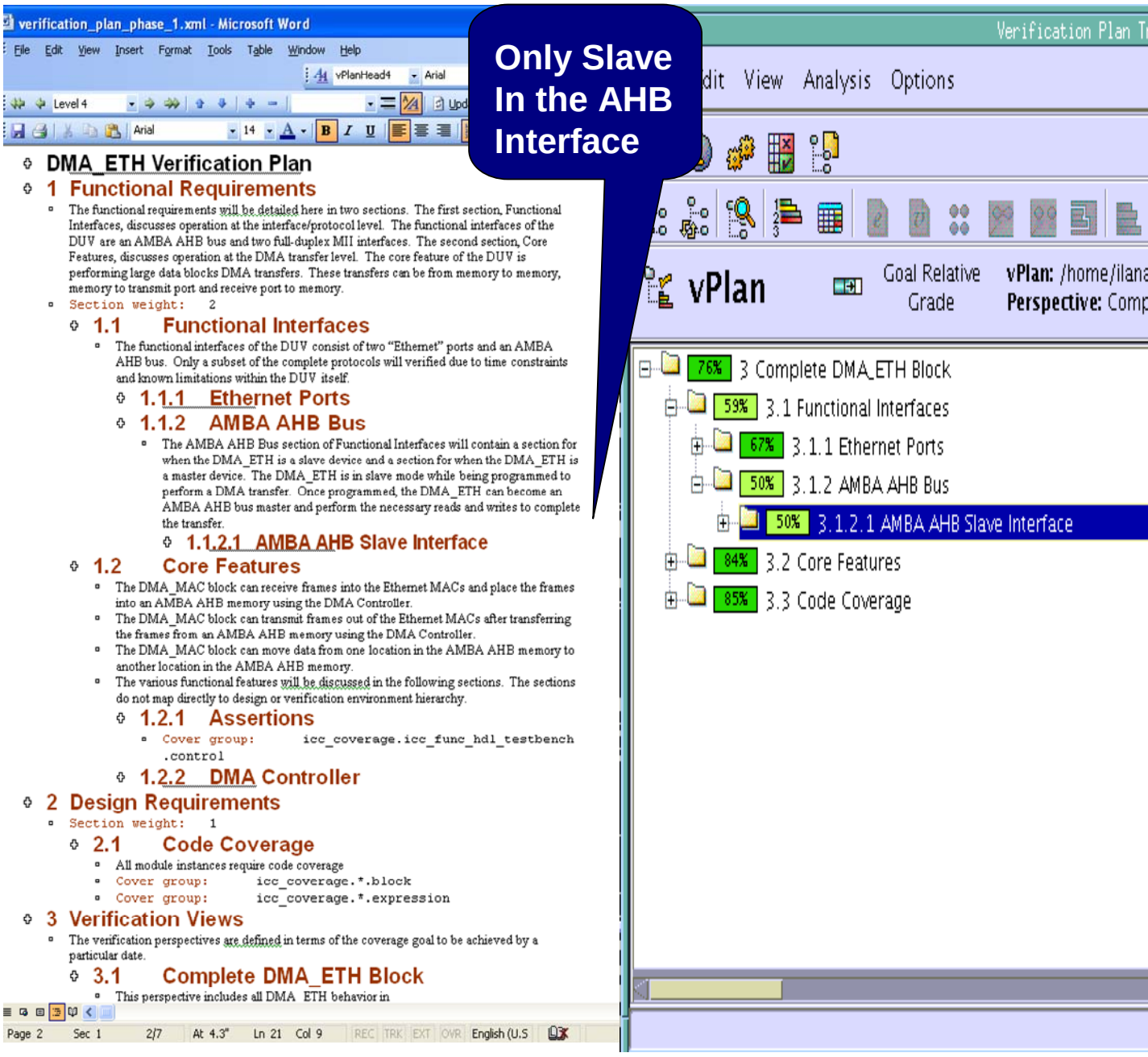


**Figure 3: An example of a verification plan with an AHB interface and slave only – left. On the right – the executable plan that was created by reading in this verification plan**

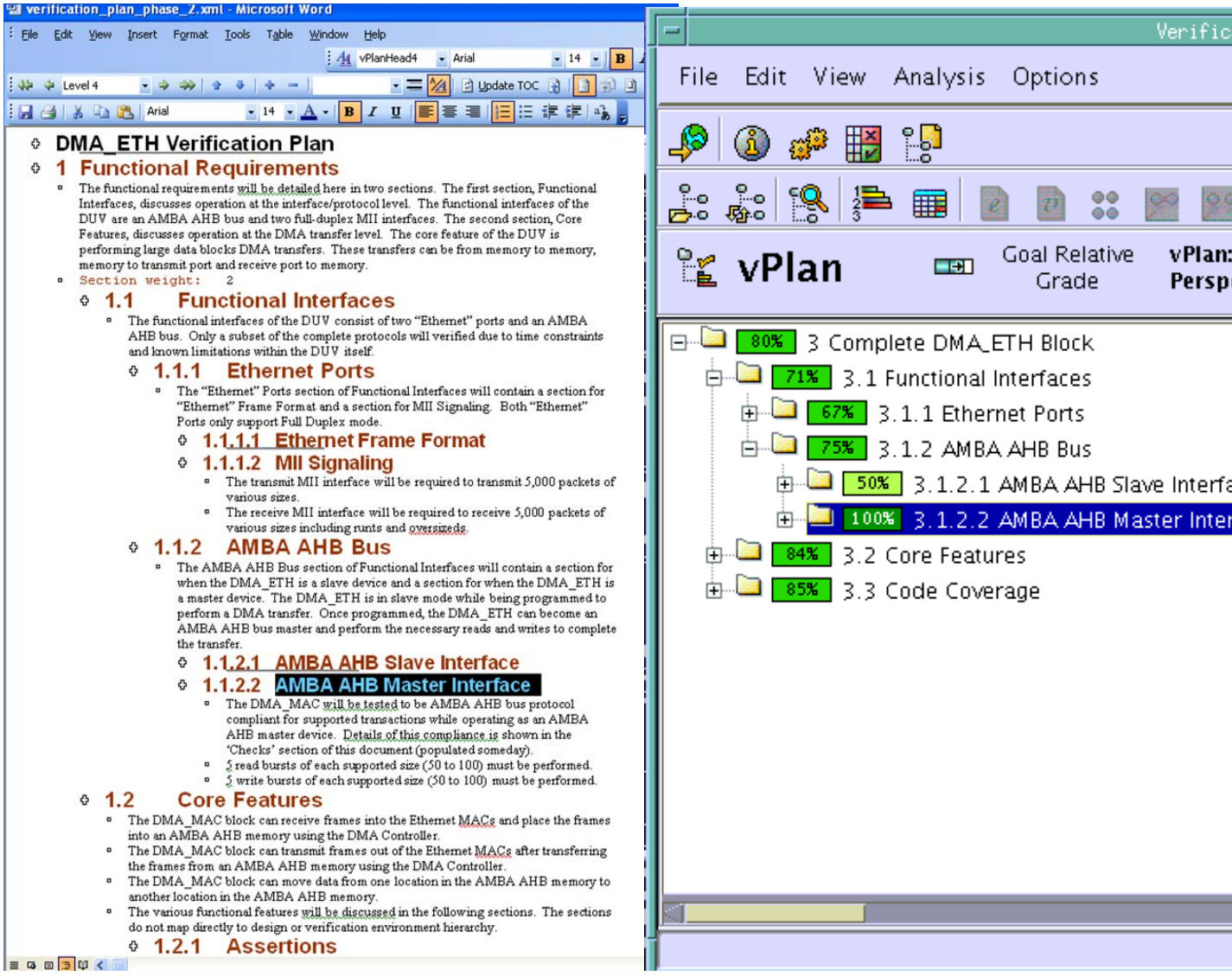


**Figure 4: Lets assume that we want to add a master to the AHB Interface. We will first add it to our verification plan. The executable plan on the right shows the master added with its coverage grade after running the regressions.**

## Don'ts

- Don't assume a change is completed until specific system behaviors exhibiting the change have been verified. Especially take care of changes up and down the entire process that reach fare within other areas of specialization.

- Don't assume one person can implement a change to completion. Effectiveness in Verification is a lot higher with redundancy and cross-checks.

- Don't assume that a "simple change" is the best solution. At times, a simple change may create a "butterfly effect" and morph the system being verified from relatively stable to utterly chaotic. Watch out for the fallacy of simple solutions by reviewing changes with all adjacent groups and the architecture team.

- Avoid short cuts by not updating plans and all necessary metrics of quality. Short cuts may lesson quality, and often impact predictability as well as productivity.

- Don't change multiple parts of the system all at once without first assessing dependencies. In some cases, it may be prudent to verify the design in a few, well-defined iterations.

By following these simple guidelines, "good" planning based on verification managment can be realized. Your organization will benefit as all stakeholders begin to leverage the ability to capture and review the verification plan as you drive you next design to closure.

*Ilana Golan is a Principal Product Engineer at Cadence Design Systems responsible for Verification Planning, Methodology and Management solutions. Ms. Golan is an expert in functional and formal verification technologies, with extensive domain experience in security, storage, networking and telecommunications systems. She has worked with several industry-standard protocols including ARM, PCI-E, PCI and SPI and supported many key customers including Agere, Broadcom, Cisco, NEC and Philips to achieve unprecedented plan to closure.*

*Prior to Cadence, Ms. Golan was a Logic Verification and EDA engineer at Intel, Haifa, where she managed development of a sequential verification tool, received the Division Recognition Award, and the best presentation award at the Design, Test and Technology Conference. Prior to Intel, Ms. Golan served as Lieutenant, Commander of the F16 Flight Simulator instructors, where she led training of air force pilots.*

*Ms. Golan holds a B.Sc. in Computer and Software Engineering from the Technion - Israel Institute of Technology, Haifa, Israel.*