Published at the end of 2011, the DO-178C standard mainly differs from its predecessor DO-178B in that it has standard supplements to provide greater scope for using new software development methods. The most important supplements are those on methods for model-based design and model-based verification, which are described in supplement DO-331. These key software design techniques offer great potential for achieving highly efficient software development in the aerospace sector while not only maintaining the high quality and safety requirements for software but actually improving them. This article shows how to use TargetLink together with DO-178C and DO-331 and which aspects to take into account.

**Models: Opening the Door to Innovative Methods**

The ability to represent requirements by models in accordance with DO-331 is a decisive advance in efficient and quality-oriented software development. The shift from purely textual requirements to formalized requirements expressed as models opens up many new options for automated analysis, source code generation, and verification. There are two different types of software requirements according to DO-178B/C and DO-331:

- **High-level requirements (HLR)**
  These describe what the software has to do, but not how it has to do it (i.e., the software is treated as a black box). HLRs are derived from the requirements for the actual system that are defined in the system process, e.g., according to...
Safe Code According to DO-178C

dSPACE’s production code generator, TargetLink, is suitable not only for automotive production projects, but also for projects in civil and military aviation. dSPACE offers a comprehensive workflow description particularly for using TargetLink in DO-178-compliant aerospace projects. It describes how to use a TargetLink-based tool chain to make software certification easier.
for representing the LLRs from which the actual source code is then generated by means of automated code generation. According to DO-331, such models representing LLRs are called design models. They contain the description of the actual functionality and also all necessary detailed information on the software, such as internal data structures, control flow information, and potential fixed-point representations (figure 2).

**From Design Model to Source Code at a Click**

Design models representing requirements (according to DO-331) offer a direct route to creating the software source code — by using automatic code generation instead of manual coding. In terms of quality and reliability, TargetLink by far surpasses human programmers and produces source code deterministically at the click of a button:

- The source code generated by TargetLink is very readable and suitable for reviews. This is ensured by extensive source code commenting and easy-to-understand symbol names, and by using a subset of the C language.
- The code can be traced straight back to the design model. This provides direct traceability between the source code and the associated model from which it was generated.
- The code to be generated with TargetLink is also highly configurable in order to fulfill the coding guidelines, combine the code generated with TargetLink with existing legacy code, and optimally integrate the generated code into the software architecture.

In general, the quality, configurability, and efficiency of the generated code are outstanding TargetLink features that are visible in all application areas.

**ARP4754 (Aerospace Recommended Practice).**

- **Low-level requirements (LLR)**
  
  These describe the internal workings of the software (viewed as a white box), i.e., how it has to do what it does. LLRs are naturally derived from the HLRs. It must be possible to generate the actual source code directly from the LLRs.

  Models can now be used to represent requirements on these two levels (figure 1). Simulink®/TargetLink® models are used particularly often for representing the LLRs from which the actual source code is then generated by means of automated code generation. According to DO-331, such models representing LLRs are called design models. They contain the description of the actual functionality and also all necessary detailed information on the software, such as internal data structures, control flow information, and potential fixed-point representations (figure 2).

**Figure 1: Important development phases according to DO-178C, including the necessary verification steps.**

**Figure 2: Simulink/TargetLink design models are used for direct automatic source code generation with TargetLink.**
TargetLink is well suited for DO-178C-compliant aerospace projects, making it possible to generate high-quality source code at the click of a button. Due to its layout commenting and symbol names, the generated code is highly readable, provides seamless traceability to the requirements and is easy to configure, e.g., to fulfill coding guidelines. TargetLink and its integration with third-party tools offer an ideal environment for verification, simulation, analysis, and tests. From the requirements to the final source code – with TargetLink, users have their DO-178C-compliant development projects under control.

Conclusion

TargetLink is well suited for DO-178C-compliant aerospace projects, making it possible to generate high-quality source code at the click of a button. Due to its layout commenting and symbol names, the generated code is highly readable, provides seamless traceability to the requirements and is easy to configure, e.g., to fulfill coding guidelines. TargetLink and its integration with third-party tools offer an ideal environment for verification, simulation, analysis, and tests. From the requirements to the final source code – with TargetLink, users have their DO-178C-compliant development projects under control.