

Integration TRACE32-Simulink®

It is now common to perform simulation and verification of designs before committing to hardware. This is why tools such as MATLAB® and Simulink® have made inroads as development software into the control engineering market. It can save a lot of time and effort if the control loop can be tested for the effects of many variables before finalizing the design.

So what is the next step, after the control algorithm has been found through simulation? How is this solution integrated into the control hardware? For this, Simulink enables you to generate code automatically. But can you be sure that the program behaves the same way on the control hardware as in the simulation?

Verification Approach

The Institute of Flight System Dynamics at Technische Universität München came up with an interesting solution during development of a flight control system for a Diamond DA42 (see Figure 3).

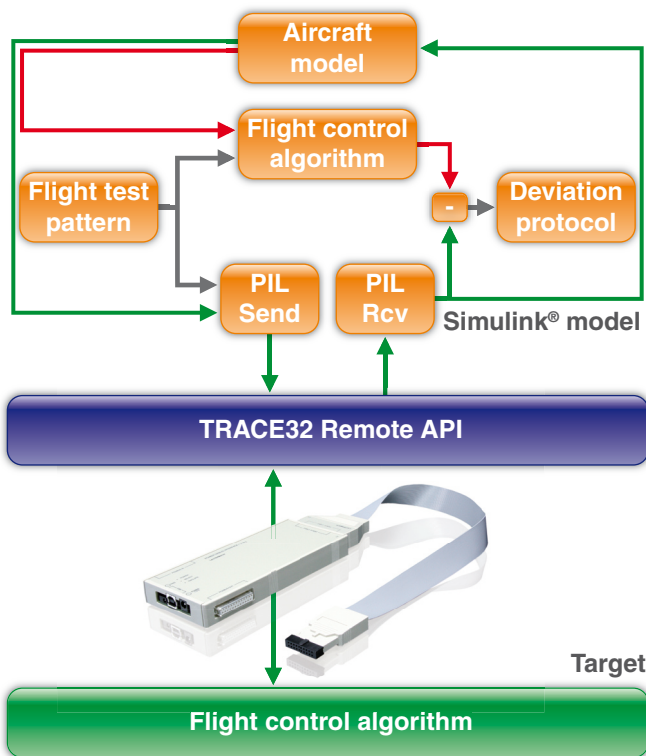


Fig. 1: The real control behavior (green path) and the simulated control behavior (red path) are compared.

After the control algorithms had been created and functionally tested with Simulink, the corresponding program code for the processor of the control hardware was generated from the control blocks using the Embedded Coder. Using a TRACE32 debugger, the generated code was loaded into the control hardware and functionally tested in-situ.

To determine the level of deviation between simulated control behavior (red path) and real control behavior (green path), but above all to confirm the numeric accuracy of the control hardware, a Processor-In-the-Loop simulation (PIL) was chosen (see Figure 1).

Essentially, the PIL simulation is based on the specially developed Simulink blocks "PIL Send" and "PIL Receive". These were designed to implement communication between Simulink and the TRACE32 Remote API.

In each run through, the flight control algorithm performs a single calculation step of the discrete time flight control on the target hardware. The Simulink model provides the necessary input parameters. The values calculated are returned to the Simulink model and there supply the aircraft model. In a parallel calculation, the simulated flight control algorithm computes the same values. The difference is then used to compare the two results.

The testing in the stand resulted in an absolute deviation of 10^{-13} – a high level of consistency that was elegantly and easily proven with this approach.

For more information about the project of the Institute of Flight System Dynamics at the Technische Universität München, go to www.lauterbach.com/intsimulink.html.

TRACE32 Integration for Simulink®

At the Embedded World show February 2012 in Nuremberg/Germany, Lauterbach will be presenting an even closer coupling between Simulink and Lauterbach's TRACE32 debuggers.

Lauterbach has used the property of the Simulink code generation that the code block always begins with a comment line which contains the name and model path for the block. These comment lines are available after the generated code has been loaded into the TRACE32 debugger. These lines allow a simple correlation between the Simulink block and the lines in the source code.

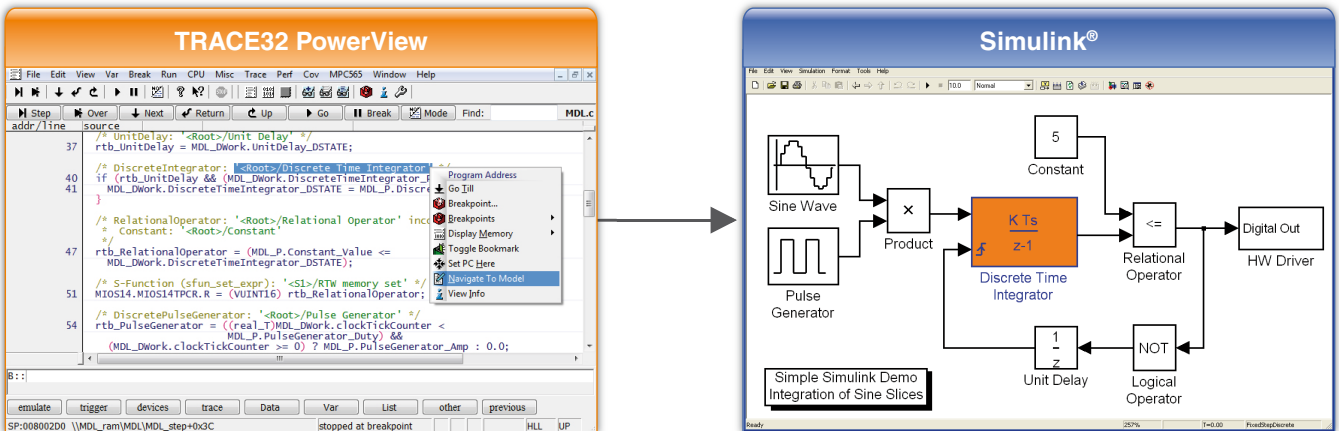


Fig. 2: The block belonging to the selected source code line is marked in Simulink.

Navigation from Simulink® to TRACE32

A global TRACE32 menu and TRACE32 menus for blocks and signals are integrated into Simulink as 'Simulink Customization Menus'.

The TRACE32 debugger can be controlled from Simulink with the help of these menus. The following functions are available:

- Show block code in TRACE32
- Open TRACE32 Variable Watch Window for signals
- Load Simulink build to the TRACE32 debugger
- Set and manage block/signal breakpoints
- Start and stop program on the control hardware

Navigation from TRACE32 to Simulink®

Selecting a section of source code in the TRACE32 debugger marks the corresponding block in Simulink (see Figure 2).

Future

When Simulink Release 2012a is available, further TRACE32 functions will be possible in Simulink. Lauterbach will use the improved functionality of the Simulink rtiostream API to integrate a PIL simulation, data logging, and parameter tuning.

MATLAB® and Simulink® are registered trademarks of The MathWorks, Inc.



Fig. 3: Diamond DA42 (Source: www.diamond-air.at)