

# Safety Developer



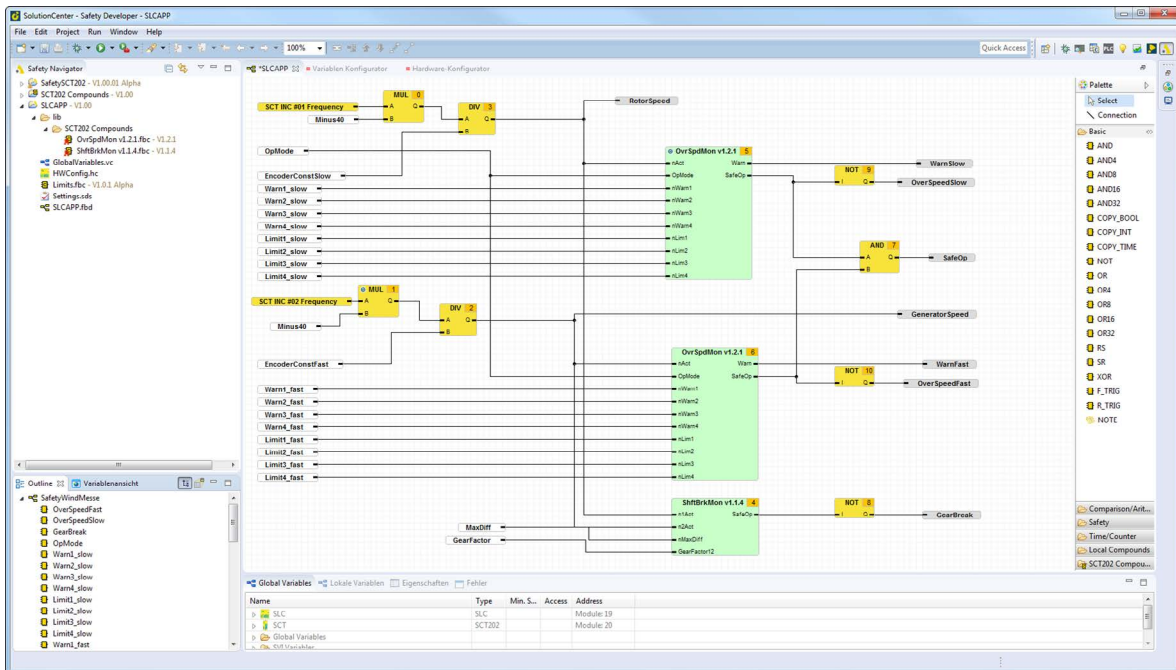
## Safety Developer Engineering Tool

For the more safety-relevant, engineering steps the SolutionCenter contains the Safety Developer that includes all tools required for safety-conformant programming in accordance with EN61508 and b PLCopen. Safety Developer has been developed and certified in close collaboration with TÜV. All methods required for logging and the verification obligation of the machine manufacturer are integral components: password management, fail-safe program transmission, tamper-proof logging on the target device, documentation of the safety program, and all software components used, unique identification of the safety modules, and the programming itself.



- Integrated in the SolutionCenter all-in-one engineering tool
- Full-graphic function plan editor with autorouting
- Certified safety modules in accordance with PLCopen Safety
- Standard modules for logical links, timers and arithmetic operations in accordance with EN61131-3
- Color coding of secure and unsecure signal flow
- Grouping of circuitry parts for repeat use (compound)
- Adjustable test depth for the project translation
- Variable monitoring, value simulation and break points
- Open programming interfaces (PLC, C, C++) for e.g. online monitoring
- Bidirectional exchange of values between secure and non-secure controller
- Unique identification of the safe hardware
- Configuration of the clocked self-monitoring of inputs/outputs
- Communication to safe hardware via Ethernet (M1) or serial
- Certified redundant program download
- Logging of the acceptance state in PDF format
- Direct connection to version management
- Online monitoring of all I/Os also in safe operating mode





## Programming in Accordance with PLCopen Safety

The safety application is programmed in a free-graphic function plan editor (Safety Editor) in accordance with EN61131-3. The module set includes a library of safety modules that have been strictly implemented, tested, and certified in accordance with the PLCopen safety standard. For the logic, additionally required standard modules, such as timers, arithmetic, and logical operations are available. The application can be organized in multiple separated functional units and sub programs in order to structure the program. Unsafe input and output signals from the control system can be added to the safety project via a browser. The execution sequence of the modules is presented graphically and can be corrected by the user. Redundant hardware inputs are summarized via equivalence or antivalence blocks and subsequently presented as a safe signal in the program. The data types, bool, integer, and time, are supported.

### Visibility in the standard program

The transparent exchange of signals functions in both directions. In the Safety Developer you configure which values – irrespective of the states of the safe I/Os – should also be visible in the unsafe world. Thus intermediate results in networks and the status of modules can be presented in a visualization, recorded with the Scope, or evaluated in a PLC program. Thus there are extensive diagnostic possibilities and a high level of operating convenience.

### Safe and unsafe paths

Signals from the standard sequential program and from the I/O modules of the control system can be used in the safety program as unsafe input and output signals. The classification of a signal as safe or unsafe is presented with color coding.

# Safety Developer

## User-specific Templates

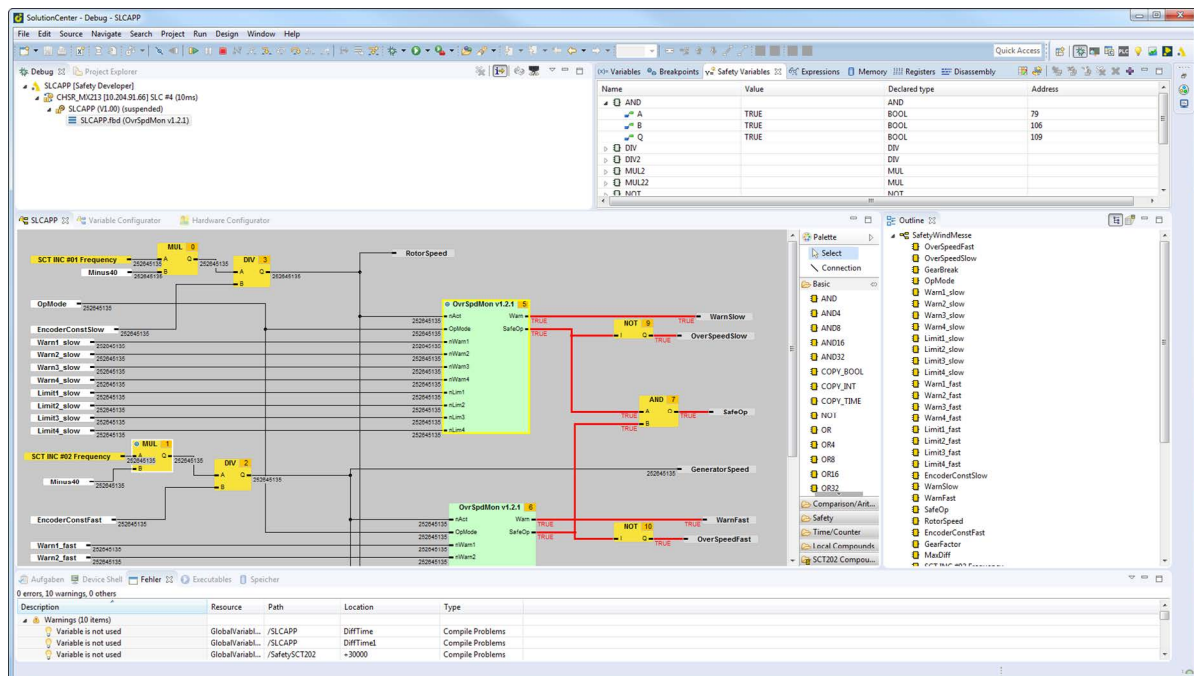
A logical circuit that is structured from a group of basic modules can be put together and given an interface in the project for a function group. Thus separate user-specific function blocks (templates) can be created and used in the project multiple times. These groups or even complete networks can also be transferred from one project into the next project.

## Tracking Changes

In the Safety Developer version management is an integral component – regardless of the logging and verification obligation. The version management database is operated directly from the project navigator. A local history is always kept automatically, which enables reversal of changes even without a genuine version management system. Naturally Undo/Redo functions are also available.

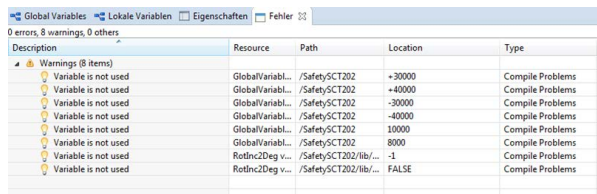
## Modularity

The Safety Developer takes the flexible requirements in today's industry into account through its modular project management. FBD networks, and also additional safe I/O modules of the project, can be activated depending on the expansion stage, signals can be connected to different sources and potentially susceptible equipment, thus a project for maximum machine expansion can be created and tested. Adaptation to the real degree of expansion is achieved through the bringing together of the desired parts. Commissioning of individual machine parts is also possible in this manner.



## Program Verification

The program for execution on the target system is not compiled, but rather is translated into a script that is checked and executed redundantly by the firmware of the target system. Nevertheless the Safety Developer can detect and display possible error sources in the code when the script is generated.



Description	Resource	Path	Location	Type
Warnings (8 items)				
Variable is not used	Global/Variabl...	/SafetySCT202	+30000	Compile Problems
Variable is not used	Global/Variabl...	/SafetySCT202	+40000	Compile Problems
Variable is not used	Global/Variabl...	/SafetySCT202	-30000	Compile Problems
Variable is not used	Global/Variabl...	/SafetySCT202	-40000	Compile Problems
Variable is not used	Global/Variabl...	/SafetySCT202	10000	Compile Problems
Variable is not used	Global/Variabl...	/SafetySCT202	8000	Compile Problems
Variable is not used	Retinc2Deg v...	/SafetySCT202/lib/...	-1	Compile Problems
Variable is not used	Retinc2Deg v...	/SafetySCT202/lib/...	FALSE	Compile Problems

## Logging

The verification obligation is supported in different ways. For logging of acceptance, a project report can be generated that also presents the entire program code graphically. The tamper-proof log book of the safety controller logs each safety-relevant change in the system, such as the download of a changed program. Thus any manipulation can be traced with user name, date, and time.

Optionally the safety program can be stored on the safety controller and can be opened and further processed from the controller. Additional user-specific information, such as author, version history, and additional comments can be stored for each network.

## Hardware Configuration

In addition to the tools for variable selection, programming, and logging, the safe hardware can also be directly configured in the Safety Developer. This includes not only assignment of unique channel names, but particularly also allocation of the controller to the project, the adding of additional safe I/O modules, and specification of test intervals for clocked lines, which then are tested automatically by the hardware for short circuit, cross-connection, and interference voltage.

Safety-relevant, required unique module identification that excludes the possibility of swapping modules after a service deployment is also executed directly in the Safety Developer. Communication between Safety Developer and the control system for program download, diagnosis and configuration is executed conveniently and in broadband via the Ethernet Interface of the M1 control system. Alternatively, communication can also be executed directly with the Safety Controller via a serial RS232 interface, which also enables use of the Safety Controller as a stand-alone solution without a surrounding control system.