Fast and flexible implementation of reliable control programs with TwinCAT PLC

Programmable logic controls (PLCs) have been known and popular throughout industry for versatile applications for many years. Since 1994, control applications have been programmed according to the IEC 61131-3 standard. TwinCAT PLC software also supports this standard, which describes not only PLC programming languages, but a complete architecture for machine control logic. It describes tasks, blocks of code, data types and programming languages, among other things

The IEC 61131-3 standard currently provides five different programming languages – two textual and three graphical. TwinCAT PLC supports the use of the language of choice or a combination of different languages. The assembler-like programming language Instruction List (IL) is rarely used anymore. Mainly Structured Text (ST) is used, which is similar to the Pascal language and is considered a high-level language. The graphical languages Function Block Diagram (FBD) and Ladder Diagram (LD) are rather traditional PLC programming languages. In addition, there is the Sequential Function Chart (SFC) language, enabling easy graphical programming of complex step chains.

Flexible workflow and fast optimization

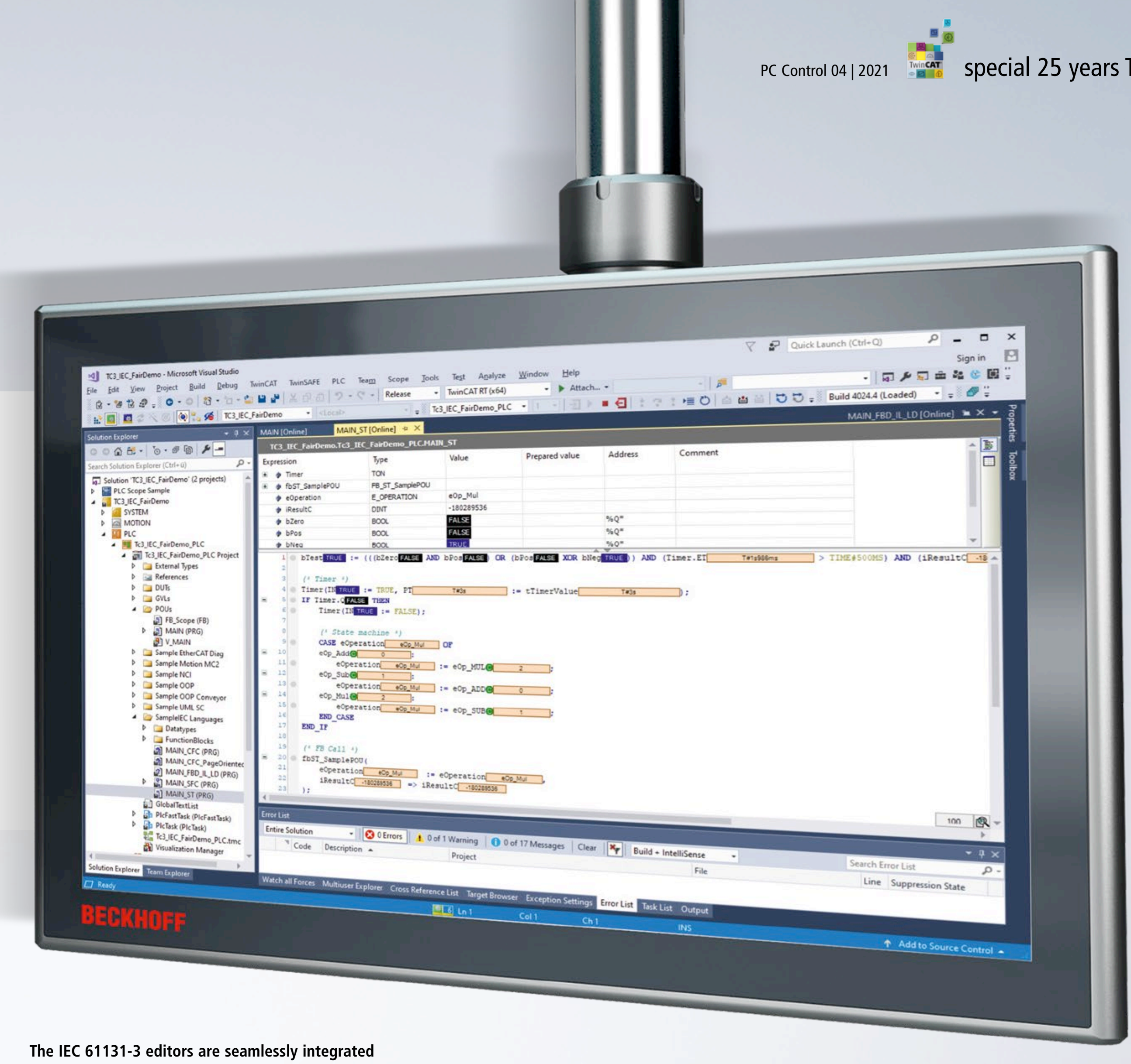
In the TwinCAT XAE engineering environment, various blocks – programs, function blocks or functions – are created in the different languages. The starting point for execution is always a task. This is processed cyclically within a certain time (cycle time). The priority of a task determines whether interruptions in the process may occur or not. Consequently, a higher priority task can – if it is scheduled – interrupt a lower priority task. This kind of preemptive scheduling is supported by the TwinCAT real-time kernel. A task is also responsible for collecting inputs from the PLC via the fieldbus from the sensors and sending outputs from the PLC to the corresponding actuators. For this purpose, each task has a process image that lists all inputs and outputs of the task as well as how they are connected to the inputs and outputs of the hardware in the configuration. With a "stand-alone" PLC, it is possible to separate a PLC project from the actual hardware configuration if required. In this way, a workflow can be established where a PLC developer cannot change the hardware configuration and an electrician cannot change the PLC code.

As a rule, a control program, once created, must be "optimized" during the commissioning phase, i.e., small adjustments must be made and errors in the sequence must be eliminated. First, all current values are displayed in the editors of the individual languages. Monitoring the sequence flow also allows engineers to see which lines or networks are currently being traversed, giving a very quick overview of what is happening at any given time. By setting variables, the flow can be specifically overridden and set in a desired direction for test purposes. More complex dependencies between variables can be easily represented graphically with the TwinCAT Scope software oscilloscope. The special feature of a TwinCAT PLC is that a sequence can be changed without having to interrupt the program execution. This feature, known as "Online Change," offers great advantages because, depending on the application, stopping the PLC may either not be permissible at all or else result in significant downtime costs.

If commissioning is carried out with a team of engineers, it is helpful to activate the multi-user functionality for the PLC project in TwinCAT. This functionality supports the workflow in that the system automatically saves all version statuses of a PLC project that are downloaded onto the control system. This means that when a changed version is logged in, a comparison with the currently activated status can be displayed in the TwinCAT Project Compare tool. This helps ensure that already imported changes of the other commissioning engineers are not overwritten during an online change, and it avoids having to perform a manual exchange and adjustment. The multi-user function uses Git in the background to store and merge the version statuses, and due to the deep integration into the engineering workflow, no additional effort is required from the user. The repository can be stored either on the control device itself or on a separate PC.

Assured software quality

Analogous to software development in IT, controls engineers must ensure the quality of PLC software. What does standard IT do to create "good"



TC3_IEC_FairDemo.Tc3_IEC_FairDemo_PLC.MAIN_ST

Expression	Type	Value	Prepared value	Address	Comment
Timer	TON				
fbST_SamplePOU	FB_ST_SamplePOU				
eOperation	E_OPERATION	eOp_Mul			
iResultC	DINT	-180289536			
bZero	BOOL	FALSE		%Q*	
bPos	BOOL	FALSE		%Q*	
bNeg	BOOL	TRUE		%Q*	

```
bTest TRUE := (((bZero FALSE AND bPos FALSE) OR (bPos FALSE XOR bNeg TRUE )) AND (Timer.ET T#1s986ms > TIME#500MS) AND (iResultC -18

(* Timer *)
Timer(IN TRUE := TRUE, PT T#3s := tTimerValue T#3s );
IF Timer.Q FALSE THEN
    Timer(IN TRUE := FALSE);

    (* State machine *)
    CASE eOperation eOp_Mul OF
    eOp_Add 0 :
        eOperation eOp_Mul := eOp_MUL 2 ;
    eOp_Sub 1 :
        eOperation eOp_Mul := eOp_ADD 0 ;
    eOp_Mul 2 :
        eOperation eOp_Mul := eOp_SUB 1 ;
    END_CASE
END_IF

(* FB Call *)
fbST_SamplePOU(
    eOperation eOp_Mul := eOperation eOp_Mul ,
    iResultC -180289536 => iResultC -180289536
);
```

Error List: Entire Solution | 0 Errors | 0 of 1 Warning | 0 of 17 Messages | Clear | Build + IntelliSense

The IEC 61131-3 editors are seamlessly integrated with TwinCAT in the Visual Studio® environment.

software? On the one hand, engineers can attempt to encapsulate functional blocks in the concept phase and reuse them as often as possible – even with slight modifications. The concepts of object-oriented programming (OOP) help here. For this purpose, TwinCAT PLC offers a UML editor that supports class diagrams for planning and implementation as well as state charts. Furthermore, the software applications should be versioned, which means that every change is logged, provided with an author and comment and can be restored at any time. Using the Visual Studio® integrated interface to source code management systems, developers can freely choose their desired system. In addition, integrated static code analysis offers the possibility to guarantee the uniformity and quality of the code with previously defined coding guidelines. The static analysis function can be triggered manually or performed automatically during code generation.

Automated software tests are another component in quality assurance, which can be implemented freely with the existing interfaces in a combination of unit, integration and system tests. Finally, when the software is already executed on the machine and running reliably, the TwinCAT PLC Profiler can be used to optimize the software. Markers are automatically added to the PLC code, and the execution times are measured and analyzed. In this way, for example, a hot-path analysis for identifying program parts with high time consumption and other detailed analyses can be carried out. In sum, TwinCAT Engineering supports the PLC developer along the entire software lifecycle.

Left: Béla Höfig, Product Manager TwinCAT
Right: Dr. Josef Papenfort, Product Manager TwinCAT

More information:
www.beckhoff.com/tc1200