ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA114

INTEGRATION OF NEW SIEMENS S7-1500 PLC FAMILY IN UNICOS-CPC: ENGINEERING CHALLENGES AND PERFORMANCE EVALUATION

M.Vazquez Muñiz, J. Ortolá Vidal, CERN, Geneva, Switzerland

Abstract

UNICOS-CPC (UNified Industrial COntrol System - Continuous Process Control) framework is a CERN standard solution for the design and implementation of continuous industrial process control applications [1]. This paper reports on the design and test results for the integration of a new PLC platform, the new S7-1500 Siemens PLC (Programmable Logic Controller) series. Special focus is given to the challenges faced during the integration, due to the new software architecture of the PLC, as well as to the early stage of the development and communication interfaces provided by the supplier. The paper shows the openness of the PLC development tool (TIA Portal) and presents a comprehensive evaluation of the PLC-SCADA communication mechanisms, and their integration in UNICOS-CPC.

INTRODUCTION

The first phase of the re-engineering process to support the new SIMATIC S7-1500 series in the UNICOS-CPC framework was initiated as soon as Siemens released a first stable version of its S7-1500 programming environment, the Siemens' Totally Integrated Automation Portal (TIA Portal) version 13. The solution produced in this first stage was based on the previous architecture of the framework implementation to support the SIMATIC S7-300 and S7-400 series and their corresponding programming environment: Siemens' SIMATIC Step 7 [2]. This implementation was fully stable and provided a first architectural approach and a proof of concept in terms of functionality and performance. All the necessary features needed to cope with TIA Portal were re-engineered without making use of any of the new functionality extensions and resources provided by the new programming environment together with the new PLC series (S7-1500). This implementation was successful and very useful to reveal the point where the framework must be redesigned.

After the completion of the first stage, a new UNICOS-CPC version was planned in order to incorporate the new functionality offered by the new technology (e.g. additional communication drivers, optimised data blocks, name variable addressing, increased code and data memory capacity). TIA Portal version 15 was chosen as the target version of the S7-1500 series programming environment to develop this framework version. The new implementation was focused on the re-engineering of major components of the framework, such as the device type definitions and specially the communication mechanism between the PLC and the SCADA layer based on the selection of three available communication drivers: S7 Driver, S7+ Driver and OPC UA. The limitations and design choices are presented in detail in the following sections.

DEVICE IMPLEMENTATION

The UNICOS-CPC framework relies on a library of standard object types that can model the operational behavior of the most commonly used physical devices in an industrial plant, such as sensors (temperature, flow, pressure, etc.) and actuators (heaters, control valves, motors, switches, etc.). Those devices are grouped in different families depending on their functionality and hierarchy in the plant definition: I/O Objects, Interface Objects, Field Objects and Control Objects.

Each object type is modeled and implemented as a single object in the PLC, in the case of Siemens, this functionality is implemented in a function block (FB). This FB will be instantiated, following an object oriented model, as many times as objects of the same type are needed in the control system. Then, for each instance of each object, a data structure is created to contain the information regarding its particular instance configuration. As an example, the structure for a Digital Input (DI) is shown in Fig. 1.

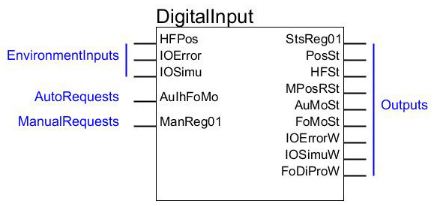


Figure 1: UNICOS CPC Digital input function block diagram.

In the version of UNICOS-CPC framework that supports the S7-300 and S7-400 series, the most common object types (IO Objects and Interface Objects) are instantiated as User Data Type (UDT) structures inside a single multi-instance data block (DB), which contains a data structure for every object instance. This encapsulation is the technical solution chosen to cope with the limitation of most of S7-300 CPUs regarding the maximum number of DBs that can be declared in a single application (maximum 4000 DBs in a S7-319-3 PN/DP CPU). The limits for a S7-1516-3 PN/DP CPU are shown in Fig. 2.

As a consequence of using the multi-instance DB, the number of DBs declared in any application is drastically reduced, however accessing from the user program to the Input-Output interface of each object instance, which is required to implement the control logic in the application

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA114

CPU blocks	
Number of elements (total)	6000; blocks (OB/FB/FC/DB) and UDTs
DB	
Number range	1 ... 60 999; divided into: Number range available for the user: 1 ... 59 999 and number range for DBs generated by SFC 86: 60 000 ... 60 999
Size, max.	5 MB; the maximum size of the DB is 64 KB with non-tuned block access
FB	
Number range	0 ... 65 535
Size, max.	512 KB
FC	
Number range	0 ... 65 535
Size, max.	512 KB

Figure 2: CPU blocks specification for a S7-1516-3 PN/DP PLC.

becomes more complex. Accessing the object instance data from the multi-instance data block requires adding to the user program the corresponding path into the multi-instance data block to every instance variable. Additionally, as not all object types are instantiated using a multi-instance data block, the user requires previous knowledge of what types are using the mentioned instance mechanism.

The multi-instance data block requires a function block to be instantiated, which then calls all the object instances in the right order. Both function block and its instance data block contain, depending on the application, a defined number of object instances. This FB is used to execute all the necessary actions before or after calling the execution of the corresponding instances per object type. As a result, both code and data blocks needed to implement the object instantiation are significantly reduced. As an example, the structure of these blocks is shown in Fig. 3 for a Digital Input (DI) object.

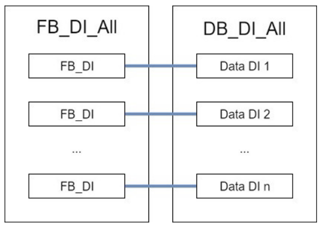


Figure 3: Example of a Digital Input multi-instance object.

The new implementation of UNICOS-CPC for the S7-1500 PLCs no longer requires the use of multi-instance data blocks to instantiate the objects. With this new technology certain limitations are no longer present, such as size of user program and specially the amount of data blocks that can be used in a single PLC application, resulting in a simpler and more intuitive implementation in which every object instance has its dedicated instance data block. In this way, all data relevant to the user program regarding each object instance (input and output interface, internal and temporary variables) is now available in a single data block, simplifying their access from the user program directly by its instance name. An example for the structure of these blocks is shown in Fig. 4 for a Digital Input (DI) object.

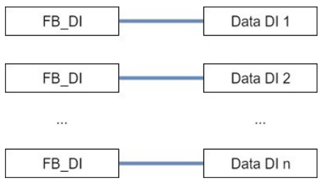


Figure 4: Example of single instances of Digital input objects (still connected to multi-instances communication blocks). Lines show relation between object data.

The functionality of an object type is still implemented in a unique FB, however in order to streamline the code produced and avoid duplication, a new function (FC) is implemented for each object type. This FC has additional functionality, such as establishing and formatting the information to be communicated to the SCADA and executing the call to the periphery access functions responsible for the correct mapping and addressing of objects into peripheral addresses (real IOs of the PLC). This function receives the data block of the instance as a parameter (new functionality offered by S7-1500 and TIA Portal V15) and calls the FB of the object type with that instance as a parameter, reducing significantly the implementation of the instantiation process.

COMMUNICATION

The PLC-SCADA (Siemens' WinCC OA) communication implementation for the S7-300 and S7-400 series is based on the Time Stamp Push Protocol (TSPP) [3]: an in-house development in collaboration with Siemens. TSPP is an event driven communication protocol implemented to address data transfer optimization from the PLC to the SCADA and time-stamped data at source. At the SCADA layer, the protocol built as an additional layer to the existing WinCC OA S7 driver. This protocol supports time stamping of all messages sent and transmission of information in a efficient way allowing significant data throughput. TSPP is responsible for the transmission of process data in a non deterministic way, where data is filtered and stored in a buffer and then sent asynchronously, as well as responsible for the transmission of critical events deterministically. Events are defined as relevant changes in the internal status of every object instance. In order to achieve determinism, TSPP implements an event buffering up to 5000 events and pushes to event buffer to the SCADA every 2 seconds or whenever the buffer is full. The SCADA S7 driver is responsible for the decoding of the event buffer and dispatches the buffer data to the corresponding SCADA data point elements (dpe). In addition, TSPP monitors that the PLC-SCADA connection is alive.

The new S7-1500 PLC controllers support the existing S7 communication (Siemens proprietary 'de facto' communication standard for S7-300 and S7-400 series) together with OPC Unified Architecture (OPC UA). OPC UA is a machine to machine communication protocol for industrial

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA114

automation developed by the OPC Foundation with focus on industrial equipment and systems for data collection and control. It is open: freely available and implementable under GPL 2.0 license, and cross-platform: not tied to one operating system or programming language. In addition, the S7-1500 series support the new Siemens proprietary protocol, the S7+ communication, evolution of the existing S7 protocol.

The performance of those two additional protocols in comparison with the existing TSPP has been analyzed. The test bench is composed of a PLC CPU S7-1516-3 PN/DP in its version 6ES7516-3AN01-0AB0 with an embedded OPC UA server. On the supervision side, a WinCC OA v3.15 SCADA application has been installed on both Windows and a Linux Centos 7 servers. The results shown in this paper are related to the the Centos 7 set up only and in the case of the tests performed with the S7+ driver, WinCC OA V3.16 has been used to enable the auto recognition; an interesting feature supported by WinCC OA with its S7+ driver which allows direct access to published process data in the PLC without any additional configuration in the SCADA side.

In the control layer a PLC application has been deployed, consisting of a configurable set of a maximum of 10000 UNICOS-CPC objects sent to the supervision layer at variable rates, The performance results obtained for a S7-300 CPU is shown in Fig. 5.

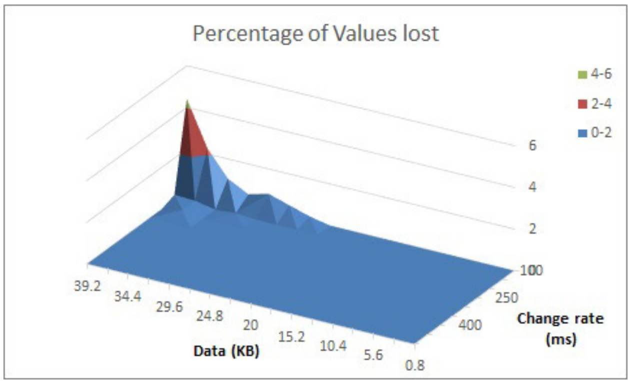


Figure 5: Performance results for the implementation of the TSPP in a S7-319 CPU. The graph shows the percentage of values lost in relation with transmission throughput and sampling time (change rate). Without any lost value the maximum performance at the minimum sampling time (100 ms.) is around 20 Kb/s and therefore an absolute performance of 200 Kb/s

OPC UA Driver

The S7-1500 PLCs are equipped with a built-in OPC UA server. On the SCADA side, the OPC UA client driver available in WinCC OA has been used.

OPC UA provides a series of different mechanisms to interact with process data. Those mechanisms include the browsing of the data (checking what is available in the OPC UA server), reading/writing and subscriptions. The communication is specially interesting when using the subscription mechanism, in which data is monitored inside the PLC and only transmitted on change. This reduces the load on the entire communication chain, from the PLC communication processor, to the SCADA OPC UA client. In this case, both server and client need to be properly configured to support the expected communication load.

Already in the first tests, the preliminary results showed performance issues of the OPC UA server in the S7-1516-3 PN/DP PLC as it failed to communicate the expected values with different configurations. When communicating a large number of values (more than 5000) the performance of the communication was decreasing until a point in which the communication was no longer possible. Due to those reliability an stability issues, OPC UA was the only communication protocol that could not be tested comprehensively.

S7+ Driver

The S7+ Driver is the new Siemens proprietary communication mechanism for the S7-1500 and S7-1200 series and supported by WinCC OA. The communication is based on a subscription mechanism without any kind of data buffering. The maximum number of objects that can be monitored depends on the PLC CPU used. In the case of a S7-1516-3 PN/DP CPU, the maximum number of objects communicated is 8000, therefore the PLC test application was modified accordingly. The test results show that without any lost value the maximum performance at the minimum sampling time (100 ms.) is around 10 Kb/s and therefore an absolute performance of 100 Kb/s. The results of this test are shown in Fig. 6.

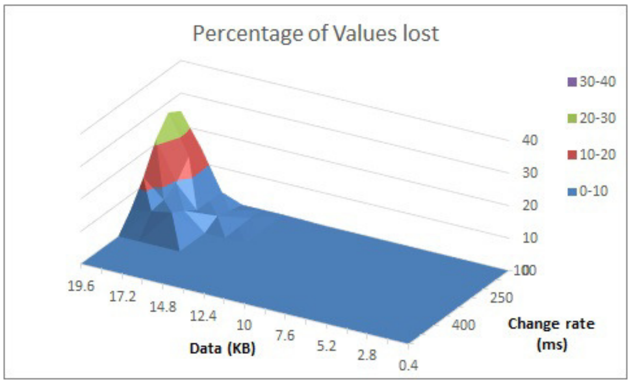


Figure 6: Performance results for the communication test using S7+ communication.

S7 Driver

A test scenario was engineered to test the TSPP mechanism applied to the S7-1500 series in communication with the SCADA layer via the native S7 Driver. The preliminary results demonstrated a significant performance degradation compared to the same mechanism implemented in a S7-300 PLC. The results are shown in Fig. 7.

After further investigations and analysis, the cause of the performance degradation was found to be a change of functionality in a function of the Siemens communication library

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA114

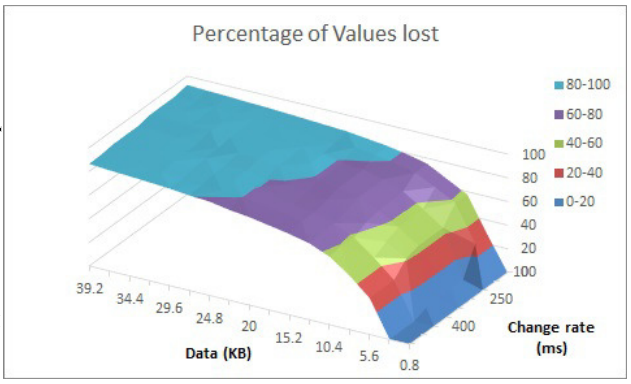


Figure 7: Performance results for the old implementation of the TSPP in a S7-1516-3 PN/DP CPU.

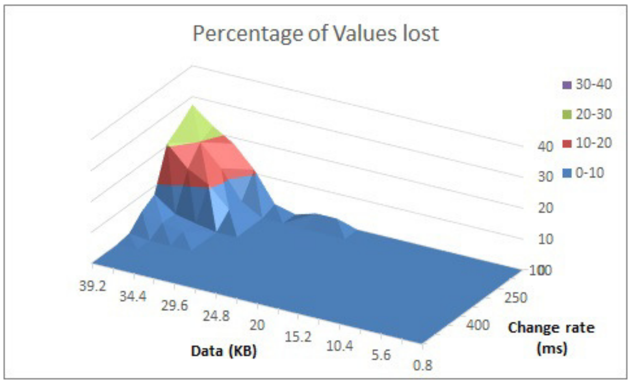


Figure 8: Performance results for the new implementation of the TSPP in a S7-1516-3 PN/DP CPU.

(BSEND) used by the TSPP mechanism to encapsulate and send information from one single data block. In the BSEND function version for S7-300 and S7-400 series, all data is sent asynchronously to the SCADA. With the new behavior, the data is split in several packets and each packet is sent as long as the previous has been transmitted and a new call to the function is executed. As a result, with only one call to the BSEND function per PLC cycle time, only one packet of data is successfully sent.

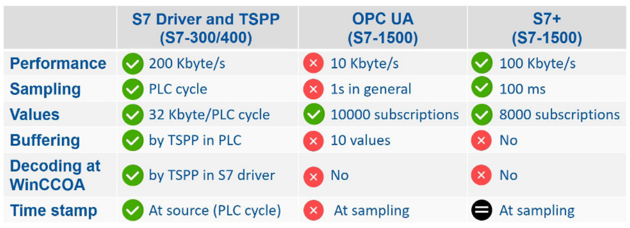
To overcome this issue, a loop calling this function at the end of the PLC cycle is set, with a timeout after which the program will continue even if all the data has not been sent. This extends the PLC cycle, but increases the TSPP performance. Figure 8 shows similar performance as obtained in the S7-300 series performance tests (shown in Fig. 5).

S7-1500 controllers have an optimized data storage. In optimized blocks, all variables (so-called tags) are automatically sorted according to their data type. The sorting ensures that data gaps between the tags are reduced to a minimum and that the tags are stored access-optimized for the processor. Non-optimized blocks are only available for compatibility reasons in S7-1500 and decrease significantly the program execution performance. The use of the TSPP for S7-1500 controllers relies on non-optimized data blocks as full control of data addressing is needed. Even if the use of optimised blocks is possible in the PLC application, the TSPP functions necessarily use non-optimised blocks for the data handling, as the mechanism relies on absolute addressing of data rather than symbolic addressing, a major feature of the S7-1500 controllers. The program execution performance degradation is not analyzed here but it does not impact significantly in the data transmission as this is executed asynchronously.

Results Summary

The test results show a significant performance difference between the three communication protocols as shown in Fig. 9. OPC UA has been discarded for the time being as a communication mechanism for UNICOS-CPC due to its performance deficiencies and the lack of a reliable way to deal with deterministic critical data. The minimum observed sampling time of 1 s. constitutes an important constraint regarding the transmission of events, as a proper buffering mechanism should be implemented together with a way to push the transmission of the events buffer from the PLC to the SCADA deterministically. OPC UA however offers a buffering mechanism for which 10 values are saved internally, in any way insufficient for the correct treatment of events.

Regarding the performance observed on the S7+ communication, no major constraints have been observed regarding the data throughput. Even if the average transmission rate is a factor 2 lower that transmission rates observed with the S7 driver for the S7-300 and S7-400 controllers, it should be enough for process data. On the contrary, S7+ communication does not provide any buffering of data or a deterministic mechanism for event handling and transmission, resulting in additional development for the implementation of such features on both PLC and SCADA layers, with the corresponding impact on the overall application performance as the functionality is not natively supported by PLC system and the S7+ Driver (in the SCADA layer).



	S7 Driver and TSPP (S7-300/400)	OPC UA (S7-1500)	S7+ (S7-1500)
Performance	✓ 200 Kbyte/s	✗ 10 Kbyte/s	✓ 100 Kbyte/s
Sampling	✓ PLC cycle	✗ 1s in general	✓ 100 ms
Values	✓ 32 Kbyte/PLC cycle	✓ 10000 subscriptions	✓ 8000 subscriptions
Buffering	✓ by TSPP in PLC	✗ 10 values	✗ No
Decoding at WinCCOA	✓ by TSPP in S7 driver	✗ No	✗ No
Time stamp	✓ At source (PLC cycle)	✗ At sampling	⊖ At sampling

Figure 9: Performance and functionality comparison of the three communication drivers

SIGNIFICANT IMPLEMENTATION CHANGES

This section describes the most relevant implementation modifications introduced in the UNICOS-CPC framework due to constraints and capabilities of the S7-1500 controllers.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEPHA114

Code Structure Re-engineering

As described in previous chapters, the increased memory resources provided in the S7-1500 controllers (in general the internal memory has been increased by factor 10 compared to S7-300 controllers) eliminates all the limitations concerning the number of data blocks that can be declared in a PLC application. The object instances can now be declared with their own instance DB, instead of the encapsulation provided by the multi-instance data block. However, in this way the amount of code of a UNICOS-CPC application is significantly increased. As a result, we have detected some limitations for large applications on the total amount of integrated memory for program. In order to address this problem and reduce the amount of code, the communication blocks that were re-implemented had to be addressed by means of an index, instead of being passed as parameters to the function. This reduced the amount of code by a factor 3 for the most common objects.

On the other hand, code reduction provokes the creation of additional dependencies between many of the programming blocks. As an example, the communication block structure is a UDT that is common to any UNICOS-CPC application and could be passed as a parameter to the object in its FC. Since those UDTs are common to all applications, they are included in the object library and compiled without dependencies, meaning they are always compiled before the instance calls. Some instances use those data structures, therefore the correct compilation order has to be ensured. That was also the case for the FCs, that only needed the UDTs to be compiled in advance. Due to the change in the communication block calls (that now use the block in the code itself, instead of as a parameter), the FC blocks need to be compiled after the instances call. That required a reorganization of the code, that now includes a compilation of a subset of instances to ensure compilation consistency. This reorganization of code is shown in Fig. 10 for a Digital Input (DI) object.

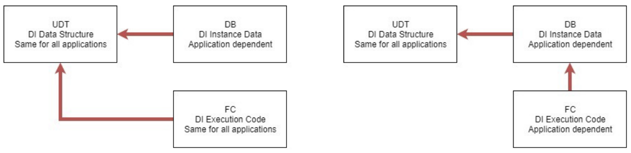


Figure 10: Dependencies before and after the memory code reduction.

Periphery Errors

A new feature introduced in the implementation of UNICOS-CPC for S7-1500 controllers consists in a new design of the mechanism for the detection of errors in the different hardware IO channels. The errors in the physical inputs and outputs (IOErrors) are now handled inside the instance DB of the object itself. To do so, at the PLC CPU start-up phase, before the user program is executed cyclically, a new mechanism detects the physical position of every input and output object that needs to be considered (module and rack or station) and write it to the object itself. Then, when an error occurs in a particular module, rack or station, only the objects that have the same physical position are considered, and the error is set or removed accordingly. This reduces significantly the process of identifying physically the input/outputs when an error occurs, as their location is previously defined.

CONCLUSIONS

The new Siemens S7-1500 controllers and their development tool Step 7 in TIA Portal offer new possibilities that can be very useful for the development of UNICOS-CPC applications; including not only for the code generation of the framework itself, but also for the logic developed by the users of the framework in their applications. However, the lack of a data buffering mechanism to ensure data transmission over a communication failure, seems to be a big disadvantage for the implementation of new communication drivers that could make the PLC code much more independent from the communication used. In addition, both S7+ and OPC UA drivers lack from a mechanism that allows triggering data transmission from the PLC to the SCADA in the user program and therefore, the management of buffered data in the PLC, a necessary requirement to transmit critical data (events) deterministically.

The TSPP implementation in the S7-1500 family is the only mechanism that supports events buffering during a disconnection that we have seen to date. We will therefore develop solutions to increase performance, for example by changing the BSEND function call and move it to a cyclic interrupt or to a time-delayed interrupt, reducing the PLC cycle time while maintaining as much as possible the communication performance.

OPC UA is already a successful example in industry communication standards due to its flexibility, reliability and openness. However, the implementation of OPC UA in the S7-1500 controllers for UNICOS-CPC is far from being a feasible solution for PLC-SCADA communication.

REFERENCES

[1] Ph. Gayet *et al.*, "UNICOS a framework to build industry like Control systems: Principles and methodology", in *Proc. ICALEPCS'05*, Geneva, Switzerland, 2005, paper WE2.2-6I.

[2] J. Ortola Vidal, UNICOS-CPC 6, PLC architecture on Siemens S7. `http://unicos.web.cern.ch/unicos-cpc-documentation`

[3] J. Ortola Vidal, E. Blanco, M. Boccioli, and T. Nunes da Rocha, "An event driven communication protocol for process control: Performance evaluation and redundant capabilities", in *Proc. ICALEPCS'13*, San Francisco, USA, paper MOPPC024, pp. 111-114.