SoftPLC vs. PLC

55576B Matias Harjula

AS-116.3150
Automaation tietotekniikan seminaari
Syksy 2007

1 Summary

This work is meant to give an overview over the field of software based logic controller (SoftPLC), and to some extent compare them to traditional PLCs. Since the PLCs can be considered "old" technology not much effort is put into the inner workings of these and they are used more as a reference. SoftPLC:s should be able to perform the same tasks in the same ways as ordinary PLCs with the exceptions stated later.

Chapter 2 gives a brief explanation of the technology, while chapter 3 deals with the definitions used later on.

Chapter 4 explains the background to the development of SoftPLCs. The development of the PLC from a programmable relay to a complex device capable of high speed process and motion control and communications with other devices. The need for connecting the automation systems to enterprise wide networks and why SoftPLC systems are suitable in the context.

Chapter 5 deals with how the systems are implemented while chapter 6 and 7 explains different aspects of reliability and real time operations. Some insight is given on how real-time extensions for non-RT operating systems work, and how Ethernet can be adapted for use with RT-systems.

Chapter 8 lists some of the interfaces used for software interfacing between the logic controller and other software systems.

Chapter 9 gives some insight into a few of the systems on the market.

Chapter 10 a short comparison between the most important properties of traditional PLCs and SoftPLC systems.

Chapter 11 Conclusions.

References have sometimes been omitted when dealing with traditional PLC:s since these can be considered proved technology with known properties (reliability, speed etc.).

Index

2 Introduction

During the last ten years integration between different systems have become more and more important. PLCs have been connected to computers and data have been collected for data acquisition systems through various exotic interfaces.

With the arrival of the software based PLCs running on ordinary PCs many of the problems have been solved through the use of standard interfaces between software systems while HMIs can be easily customized and a single computer is enough for all operations on a machine or production line. This has been made possible with the rapid development of field bus technologies and is necessary to comply with the need for data from factory- and enterprise wide systems.

In a different field, SoftPLC is a great possibility when it comes to stand-alone machinery and especially moving machinery. One computer with integrated display may be enough for work orders and information, operation of the machine, diagnostics, calculations, navigation and so on.

Some problems can easily be seen. The normal PC operating systems are not made for real time operation and extensions have to be made. The reliability of personal computers may not convince in critical environments and operations.

3 Definitions

The following definitions are used:

PLC: Programmable Logic Controller. Originally a form of programmable relay with a dedicated microprocessor. Nowadays most systems have options for communication, analogue i/o and advanced digital control.

SoftPLC: Software based logic, running on a computer with a multi-purpose operating system

SCADA: Supervisory control and data acquisition. A system that collects data from the automation system into the company network and allows for control from a higher level.

MES: Manufacturing execution system. A system for measuring and controlling factory operations. In this context it acts as a link between the Enterprise Resource Planning system (ERP) and the automation system.

4 Background

In the 1980's the control systems used in process, manufacturing and machine automation systems were DCS (digital control system), PLC (programmable logic controller) and various embedded customized systems. Since then the role of the DCS systems has largely been taken with PLCs with expansion cards for process and motion control.

The PLCs are based on a microprocessor running a dedicated program (much like a micro controller) stored in non-volatile memory. The program cycle is deterministic and the output response timing is thus predictable.

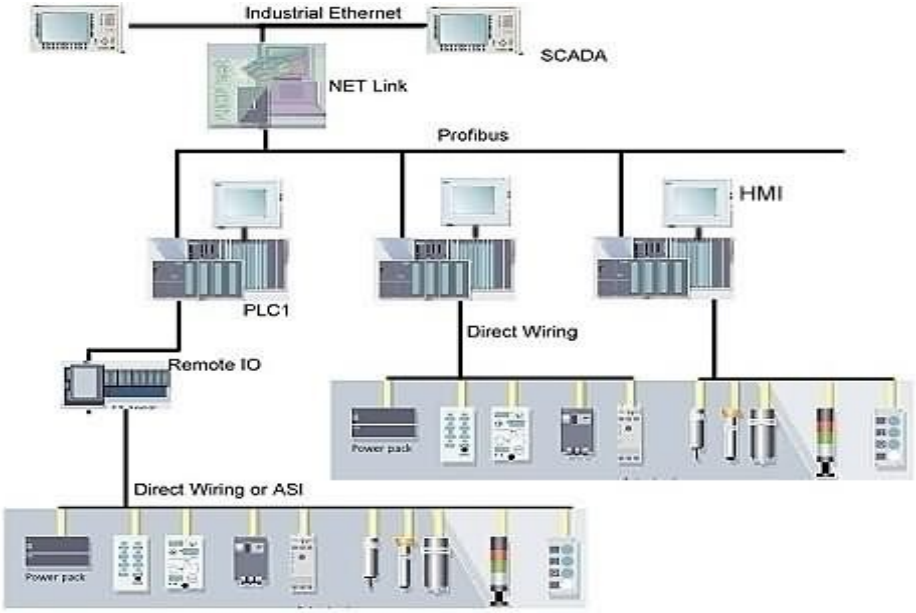
Because of the relative simplicity of a PLC (in comparison to a PC for example) PLCs are known for their great reliability.

In recent years the need to interconnect PLC with other systems has become one of the primary development directions. It started with the need to get the PLCs to work with computer based HMI systems and evolved further into interoperability with factory- and enterprise wide reporting systems. The HMI:s were in the beginning dedicated units communicating with the controller over a serial connection, but the development in personal computers have made PC-based HMIs popular. When the data is brought to the PC it can also be transferred to other systems over for example Ethernet with TCP/IP. Where PLCs usually are closed systems which can be programmed only with the manufacturers proprietary tools a relative open architecture was preferred on the PC-side making it easy to use standard components and protocols like OLE for Process Control (OPC) for interaction between software systems.

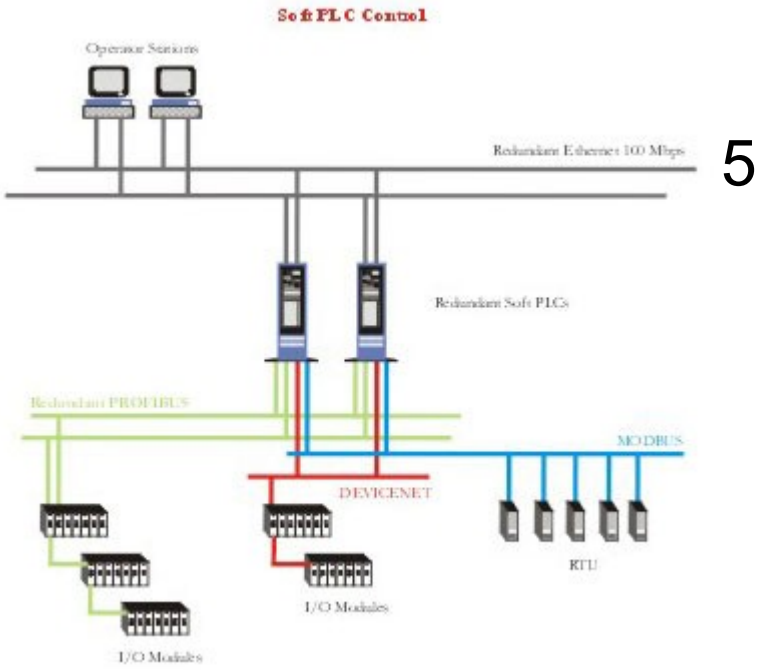
On the other hand, field buses have revolutionized the floor-side automation, the PLC controls the bus and sensors and actuators are controlled with commands on the bus instead of hard-wired analogue or digital signals.

The programming languages in the IEC 61131-3 have reduced some of the problems associated with proprietary programming languages.

The PC:s have since become indefinitely more powerful and rugged versions have been developed. The field buses have made it possible for a PC to act as a controller even when away from the process. In machinery, and especially moving machinery applications where there is need for both control and data acquisition the natural step was to make the PC act as a software based logic controller – SoftPLC. In process and manufacturing environments where the PC acted as a bridge between the automation system and SCADA systems the logical step has been to move more functionality to the PC and thus make interfacing easier. [1]



Example of traditional PLC system topology. The NET link handles the interfacing to the Ethernet where SCADA servers collect data for use in the enterprise systems. All communication interfaces have to be well planned, and all PLC:s have to be individually programmed (I/O, control, HMI, SCADA interaction). Picture from http://www.scantime.co.uk/html/plc.html



Example of SoftPLC system. The automation system is controlled directly from the SoftPLC over remote units connected by thefield buses. HMIs can be controlled by the SoftPLCs, created on the PC running the PLC or ran on the factory Ethernet. Connection to enterprise systems via the SoftPLC's network interface. The controls in the picture are doubled for fail-safe operation. Picture from http://www.protocolindia.com/html/Solution%20Soft%20PLC.htm

Implementations

There are two main paths that have been taken by the manufacturers and users. The first is replacing a traditional PLC with a PC with an interface card for the field bus used in the automation system. The PC controls the system over the field bus by sending messages to sub-controllers, I/O expanders, sensors and actuators.

The program is usually built on either a specialized real time operating system (RTOS) or on a standard system with a real time kernel or real time extensions. Examples of RTOSs are VxWorks and QNX. [1] In the last years a lot of attention has been directed at Windows CE .Net which possess real time capabilities and at the same time large portions of the Windows API making for example HMI programming easier. The Windows CE systems are developed for embedding into OEM products which gives configurability over appearance and functions.[2]

Operating systems with real time add-ons include RTLinux and extensions for Windows XP such as Ardence RTX or HyperKernel by Nematron for Windows 2000.

The hardware can be an ordinary computer or a rugged industrial PC. The industrial PC often has the display and sometimes even the keyboard integrated in the same unit as the CPU, making it a compact package.

The PC is connected to the field bus with an interface card (usually PCI or derivatives) made for the appropriate field bus. The computer is connected to the factory or company network via the standard Ethernet interface.

The automation system is connected to SCADA and MES systems via software interfaces on the PC. Standard Windows interfaces (e.g. OPC) can easily be used, which is one of the SoftPLC:s main advantages over traditional PLC:s. On a traditional PLC all variables that are to be recorded or viewed on the SCADA or MES systems have to be defined separately, as well as enabling listening for the control messages from higher systems. The interfacing computer then also has to be configured for the same messages and the appropriate interfaces has to be set up.

The other way around is the embedded version. This differs from a traditional PLC mostly in that runs a real operating system instead of just the dedicated program. The embedded systems can have I/O points on their own or they can communicate over field buses. The operating systems allows other programs to run simultaneously, for example HMIs if the computer is attached to a screen, web or other servers for upstream data acquisition. The computers are often embedded in the same case as the monitor, making the whole system compact and durable.

The embedded version has some distinct advantages for example in machinery and especially moving machinery applications. One is that if an embedded computer is needed for other operations such as reporting, work orders or maintenance scheduling some cost can be cut when leaving out the PLC while programming becomes easier with better interfaces between the automation system and the software.

Another use of the embedded version is to use it for enhancing a product by including an embedded computer and allowing the product to control external devices, for example a robot could control the rest of the robot cell through the embedded computer. This is also possible with traditional PLC:s but SoftPLC:s allow for increased flexibility and interoperability with SCADA systems, something not necessarily facilitated on the robot's own control hardware and software.

[9]

6 Real-time properties

The real time requirements for automation systems vary, but there are almost always a need for deterministic response times. The maximum response time requirement can be somewhere between a few to hundreds of microseconds. More interesting than the shortest or average response time is usually the worst case scenario. Especially in motion or process control a too long delay can have disastrous consequences.

The popular operating systems from the Microsoft Windows family have the desired user interfaces and interfaces to other software systems, but lacks deterministic timing behaviour by design. This is usually solved by using a real time extension or kernel. The idea is that the real time kernel takes control over the inputs, outputs and interrupts. The Windows CE .Net operating system has built-in real time properties.

The system is now divided into two parts, one deterministic (the real time kernel) and one non-deterministic (Windows XP/2k/NT) running on two virtual machines. The deterministic side has direct access to hardware and a predictable priority scheduling system. It listens to the interrupts and those that are not configured to be important to the control system are ignored. [3]

The real time properties of modern SoftPLC systems are good enough for use with motion controllers such as NC- or servo controllers, and can timing wise be used for most automation tasks. [Siemens]

6.1 Real time Ethernet

The PC that are used as a host for the SoftPLC are very well adapted for use with TCP/IP over Ethernet. The problem is however that Ethernet is not suited for timing critical tasks. More traffic on the network decreases the response times and since TCP/IP is built for very large environments it contains a lot of overhead when used in a normal automation system.

The problem is basically that every host can broadcast at any time, and if routing algorithms are not optimized flooding may occur, prohibiting all other messages from being delivered.

Different methods have been used to solve the problems. Technologies based on the IEEE 1588 relies on synchronization between the clocks on the network and different methods for timing the messages sent.

Other solutions aim at reducing the overhead and improve on the routing of the messages. As an example Beckhoff's EtherCAT where messages are routed to the devices based on their MAC-addresses while TCP/IP and UDP/IP overhead is reduced to a minimum. [4]

7 Reliability

The main obstacle for the success of the SoftPLC systems are their reliability. The traditional PLC is usually regarded as an extremely reliable device [No citation needed], while the PC:s are considered unreliable [No citation needed].

The problems with personal computers can largely be contributed to the wealth of features in the operating systems and the many hardware configurations possible.

The manufacturers of SoftPLC systems have different options to reduce the risks of malfunction. Embedded systems can be limited in hardware options to ensure that the system is running on a system with tested components, and if a more limited operating system than the desktop versions of Microsoft Windows is used the complexity can be reduced and higher reliability can be achieved. [3]

On PC:s running SoftPLC:s the real-time jitter originating from the hardware can be measured using diagnostics tools to ensure compatibility between the components.[6]

Since most of the operating systems used on larger SoftPLC systems do not have native real time support there is always a risk of the OS locking up the computer. To prevent this "blue screen of death" phenomenon the real time extension runs the OS as a thread on a virtual machine. The logic control thread runs on and tries to safely bring the system down. [6 / 7 / 3]

8 Interfaces

8.1 Programming

The programming tools for SoftPLC controllers usually implement the languages in the standard IEC 61131-3. While this was suppose to guarantee portable code,i.e. a program made on one system should be portable to competing systems with only a small effort in reality the proprietary extensions and differences in the way I/O definitions are made have proved to be enough to prevent portability. This seems to be the case with ordinary PLCs as well.

8.2 Interfaces

The PLC programs can usually be configured to communicate with the world in a number of different ways. They of course communicate with the target system through the field bus, or in case of a I/O card on the computer directly with the actuators and sensors. If the field bus is of a master-slave type the controller can assume the master role.

The software can also communicate with other software systems in different ways. The PC can run a web server for data acquisition or as a HMI interface, various other servers (ftp, sash...) or connect to other programs via the interfaces provided by the operating system. The most used "protocol" is OLE for Process Control. OPC was developed to act as a driver between computer software and automation hardware. The protocol consists of different specifications for different uses, for example data acquisition, alarms and commands.

Different networking protocols are also used for the data exchange, as well as Microsoft's interface controls (dll, ocx).

9 Systems on the market

The systems on the market can be divided into two types. One is replacing the traditional PLC in factory automation and is mostly chosen because of the need for close integration between different software system. Most suppliers of larger PLCs have some kind of SoftPLC solution.

The other type is the embedded systems, often used in in independent machinery, and using no HMI at all or the computer display as HMI. Reasons for using this is usually in the case where the display is used as HMI that cost is reduced where compared to having separate components and better usability where everything is controlled from one system. There are also OEM products intended for integration into other products to provide more functionality.

9.1 Siemens

The Siemens product line of SoftPLC systems goes under the name of WinAC. The system consists of software and cards for interfacing the PC to the field bus. The system is available with Profibus DP and Profibus DP/MPI cards and Ethernet support is included as standard.

The system is run on an ordinary PC with Windows (XP) and real-time extensions. The system can be programmed with the standard SIMATIC programming tool Step 7 and IEC 61131-3 languages (with some extensions).

Modules are available for motion control

An Open Development Kit is available for programming and configuration of interfaces to other hardware and software systems. [5] [6]

9.2 Beckhoff

Where Siemens provide almost anything in the range of PLCs Beckhoff is focused on software based automation. Their product family is called TwinCAT, and consists of different control products, including modules for motion control.

The systems run on Windows (XP, embedded, CE) and have standard interfaces for connecting to other systems (for example OPC, OCX, DLL)

The field bus options include Lightbus, PROFIBUS DP, CANopen, RS232, RS485, Ethernet TCP/IP (EtherCAT). EtherCat is Beckhoff's own real time implementation of the Ethernet bus.[7]

9.3 SoftPLC (company)

SoftPLC is specialized in embedded systems. In their product line they have a range of solutions. From small CPUs to panel-PCs with field bus cards. One of the most interesting solutions is the Tealware rack mounted range, where the CPU and the I/O modules are modules mounted in a backplane like a traditional PLC. They also sell just their controller software for user implementations of hardware.

All PLCs from SoftPLC are running on Linux OS.[8]

9.4 CoDeSys

CoDeSys by 3S Smart Software Solutions are approaching the SoftPLC business strictly from a software point of view. Their products include a programming environment according to IEC 61131-3 and a runtime part available for a variety of hardware architectures. They also have a full suite of tools for the OEM integrator and end user. The maker of for example an industrial robot

can with CoDeSys include some PLC functions as well as some interfaces to data acquisition systems with little effort and using the same hardware that controls the motion. The robot cell can work as an independent unit where the robot controller also handles the external logic and control of the cell and deals with the SCADA system. [9]

10 SoftPLC compared to the traditional PLC

Integration:

SoftPLC	PLC
- integrates very well with other software systems, either on the same host computer using standard interfaces such as OPC or on the Ethernet network	- Interfaces exist for communicating with computer systems. Must be configured on the controller. Interfaces are rapidly improving.

Reliability:

- Reliability questionable when compared to traditional PLCs. The operating system and PC hardware are critical factors.	- Extremely good. Programming errors or hardware failures most critical.

Programming:

- IEC 61131-3 (with extensions) - Computer programmer can interface to automation system	- IEC 61131-3 (with extensions) - PLC and computer programming needed for interfacing

Price:

-From low to high. Applications needing a computer or HMI can benefit from a shared solution	-from very low to high. HMIs or interfaces to larger systems will usually increase price.

11 Conclusions

Around the turn of the century it seemed like the SoftPLC technology was heading towards a bright future. The ability to run on cheap PC parts, standardized programming languages and great interoperability with enterprise systems should have been enough for it to replace the old technology with dedicated PLCs. The biggest problem for SoftPLC solutions is the reliability. The traditional PLC has a good reputation and in a production environment breaks are usually costly, even when not dangerous. The fact that standard components can be used instead of proprietary has the drawback that the parts are not tested for compability.

Some factories have switched to computer based automation, and the systems are capable of real time operation, both for logic and motion and process control. While the lack of multi-purpose RTOS can be seen as a weakness the extensions available for Windows and Linux systems seems to be enough for most systems.

The increased integration of production systems (automation-MES-ERP) will probably either make the SoftPLC solutions more popular or force the old PLCs to be more like software solutions. Especially where the amounts of data to be collected are big the SoftPLCs will probably find their places.

The situation with embedded solutions is completely different. Here the SoftPLC can replace a lot of electronics and at the same time development cost. Moving machinery (harvesters, tractors...) can have just one on-board computer that handles everything, is easily upgradeable and relatively cheap. Even though the systems are not directly compatible the standard programming languages ensures that if the computer goes out of production, it can quite painlessly be replaced by a new one with similar specifications.

Small stand-alone units where no HMIs are needed and critical systems will probably continue to be based on dedicated units for a foreseeable future.

One interesting point is that a shift towards software based automation will transform the automation system suppliers into software makers (as discussed in [1]). The example with CoDeSys shows one possible concept where hey don't supply hardware at all and instead help their customers write drivers for the platform chosen.

12 References

[1] Koskinen, Kari 2000. PC Automaatiosovelluksissa – nykytila ja tulevaisuuden näkymät (referenced on 1.11 2007) Available at: http://www.tkk.fi/Yksikot/Auttieto/pcautom.htm

[2] Dicklin, Ronald 2003. Windows CE .NET Combines Real-Time Control, HMI and Networking, RTC Magazine July 2003. (referenced on 1.11 2007). Available at: http://www.rtcmagazine.com/home/article.php?id=100190&pg=1

[3]Christensen, Peter 2005.Design Considerations for the Development of Real-Time Windows Systems, RTC Magazine June 2005. (referenced on 1.11 2007). Available at: http://www.rtcmagazine.com/home/article.php?id=100360

[4] Montague, Jim 2003. Ethernet Hits Real-Time...Really, Control Engineering , December 1, 2003. (referenced on 1.11 2007). Available at: http://www.controleng.com/article/CA339683.html

[5]Prosessori 06/2005 Tuoteuutiset: Tehokasta PC-automaatiota

[6]Siemens corporate website [referenced on 3.11 2007]. http://www.siemens.com/winac

[7] Beckhoff corporate website [referenced on 4.11 2007] http://www.beckhoff.com/english/twincat/default.htm

[8] SoftPLC corporate website [referenced on 3.11 2007]. http://www.softplc.com/products.php

[9] 3S Corporate web site [referenced on 4.11 2007] http://www.3s-software.com/index.shtml?en_Products

[10] OPC Foundation. About OPC, what is OPC? [referenced on 3.11 2007]. http://www.opcfoundation.org/Default.aspx/01_about/01_whatis.asp