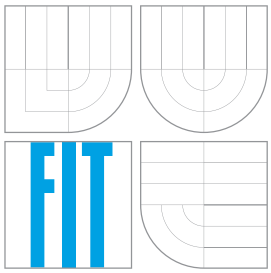
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

# IMPLEMENTACE OSAZOVACÍHO AUTOMATU V PROGRAMOVACÍM JAZYCE SIEMENS S7.

IMPLEMENTATION OF A PICK AND PLACE MACHINE IN SIEMENS S7 PROGRAMMING LANGUAGE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAKUB MAREČEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. GEERT VANDECASTEELE, KHBO

Ing. PAVEL BARTOŠ, BUT
SUPERVISOR

OOSTENDE 2011

# Abstrakt

Tato práce se zabývá návrhem metody, jak řídit SIMATIC S5 zařízení pomocí o generaci novějšího SIMATIC S7 PLC prostřednictvím PROFIBUS sběrnice, doplněného o dotykový panel SIMATIC TP177b DP/PN. Pro komunikaci s S5 zařízením byl zvolen modul IM 308-C DP. Zařízení je zcela řízeno pomocí SIMATIC S7 PLC a dotykového panelu. Jako programovací jazyk byl vybrán STL. Aplikace pro dotykový panel byla vytvořena v prostředí SIMATIC Wincc flexible 2005. Hlavním přínosem této práce je uvedení a popsání metody řízení zařízení pomocí novějšího PLC. Další směr práce je diskutován v závěrečné kapitole.



# Abstract

This work deals with a method of how to control a SIMATIC S5 machine by a newer SIMATIC S7 PLC and a SIMATIC TP177b DP/PN touch panel via PROFIBUS. For communication with the S5 machine an interface module IM 308-C DP was chosen. The control of the machine is completely realized by a newer SIMATIC S7 PLC and a touch panel. As a programing language was chosen Statement List (STL). The application for the touch panel was created in SIMATIC Wincc flexible 2005 environment. The main objective of this work is to propose and describe a method of controlling the machine by a newer PLC. The future work is discussed at the end.



# Klíčová slova

PLC, SIMATIC TP177b DP/PN, PROFIBUS DP, IM 308-C DP, STEP 7, STL programovací jazyk, GRAFCET

# Keywords

PLC, SIMATIC TP177b DP/PN, PROFIBUS DP, IM 308-C DP, STEP 7, STL programing language, GRAFCET


# Citations

Jakub Mareček: Implementation of a pick and place machine in Siemens S7 programming language, Bachelor's thesis, Oostende, KHBO Faculty of Engineering Technology, 2011

# Implementation of a pick and place machine in Siemens S7 programming language

## Declaration

I declare that this thesis is my own work that was coached by my advisors Ing. Geert Vandecasteele and Ing. Pavel Bartoš. I declare that I mentioned all literature sources, which were used.

.....
Jakub Mareček
May 16, 2011

## Acknowledgements

I would like to thank Erasmus program and coordinators, KHBO and BUT for the opportunity to study abroad. Many Thanks to my thesis advisor Ing. Geert Vandecasteele for guiding me throughout this work, for his support, willingness and priceless advices. Also I would like to thank Ing. Pavel Bartoš for his support and willingness. Furthermore, I would like to thank my parents for the opportunity and finance support to study abroad. Also to other Erasmus and Belgian students for a great time in Belgium.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The theme for this work is to implement a pick and place machine, which is controlled by a SIMATIC S5 PLC in Siemens ®STEP 7 programing language and in its environment. This includes, that the machine should be completely controlled by a newer SIMATIC S7 PLC instead of the previous. A part of the project work is also a visualization and a limited control by a SIMATIC TP177b DP/PN touch panel. For connection between the SIMATIC S5 PLC, the SIMATIC S7 PLC and the touch panel it should be used PROFIBUS. The connection allows a communication, by which it is possible to control the machine by the newer SIMATIC S7 PLC and the touch panel. The whole *system* is formed by a pick and place machine, a SIMATIC S5 PLC, a SIMATIC S7 PLC, a SIMATIC TP177b DP/PN touch panel and a PROFIBUS connection.

The aim of this work is to propose and describe a method of how to control a SIMATIC S5 machine by a newer SIMATIC S7 PLC using PROFIBUS. There is also described a design and an implementation of the pick and place machine and an application for the touch panel as well.

The work is structured into following chapters. The main objective of the second chapter is to define and present a programmable logic controller; how it works and how it could be used to control a machine, in general to control a kind of process.

The third chapter is aimed to describe PROFIBUS that allows communication, by which it is possible to control the machine by the newer SIMATIC S7 PLC.

In the fourth chapter, there is mentioned a hardware part of the system, a schema of the machine and other hardware parts used in the project work.

The fifth chapter describes a software part of the work. There could be found a description of STEP 7 environment, programing languages that can be used and a concept of programing of SIMATIC PLCs. There is also introduced GRAFCET as a graphical representation of algorithms of an automated system.

The method how to achieve a communication between the system parts via PROFIBUS and a design how to implement the machine is given in chapter six. There is also mention of a design part for the touch panel application.

The implementation of the system of communication, the program for the pick and place machine and the application for the touch panel is described in the seventh chapter.

The eighth chapter includes the end summary and future work possibilities.

## 1.1 Motivation

The basic motivation for this work is to enable to control the machine by a newer SIMATIC S7 PLC. Afterwards, it allows to use STEP 7 environment, which offers more convenient and easier implementation of the machine than previous.

This work could also serve as an overview of an automated system and give a reader a slight idea of the image of it.

# Chapter 2

# Programmable logic controller

## 2.1 What is a programmable logic controller?

A programmable logic controller (PLC) is in general an embedded system that carries out a program typically used for controlling a system e.g. a press for shaping plastic parts. If we remove plastic covers of a PLC, core hardware could remind us a simple computer formed by a central processor unit that carries out instructions of a program and a memory that stores the program and data as well. The PLC is not a general-purpose workstation as e.g. a desktop computer, but it is flexible enough to be used in many applications for controlling wide range systems. An apt definition of a programmable logic controller (PLC) says in [2], it is: *"A digitally operating system, designed for use in an industrial environment which uses a programmable memory for internal storage of user-oriented instructions for implementing specific functions such as logic, sequencing, timing, counting and arithmetic, to control, through digital or analogue inputs and outputs, various types of machines or processes.*"

Early PLCs were designed to replace relay logic systems. The very first PLCs appeared in about 1969 in the United States as a response to the demands of the automotive industry to develop automated production lines which could keep pace with technical evolution and with new production models. [10]

Previous used logic systems weren't flexible enough and with every change the system had to be rewired. The main advantage of programmable logic controllers against the relay logic systems became their flexibility - a programmable logic controller allows to program the logic while a relay logic system implements the logic by wiring particular logical gates. PLC successfully replaced previous systems.

Since the very first appearance PLCs have become an essential part in many applications which are found in the most industrial sectors i.e. agricultural and food industries, metallurgy, mechanical and automotive engineering. The automotive industry is still one of the largest users of PLCs.

## 2.2 Architecture of a programmable logic controller

Core hardware of a PLC consists of a *processor unit* and a *memory unit*. To these basic parts are added other components which complete a PLC and are necessary for its functionality and for the application itself as well. These parts are internal PLC bus, input and output modules, a power supply unit, communication interface and other special modules.

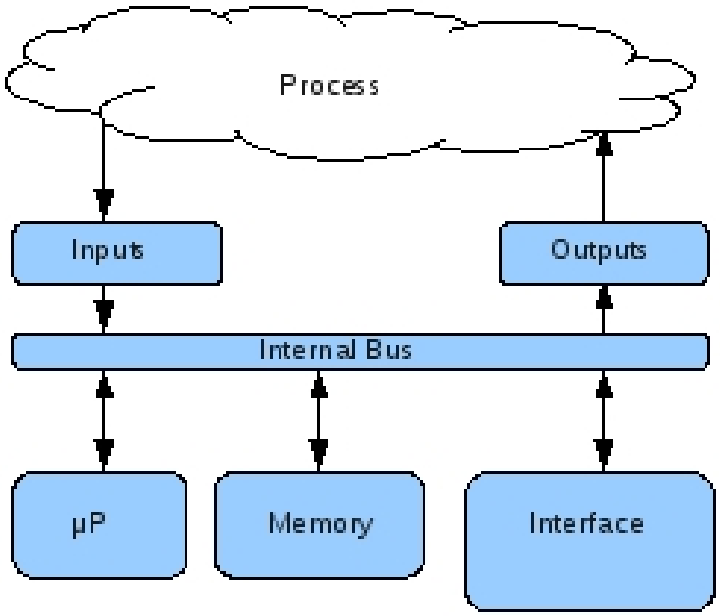



Figure 2.1: Architecture of a programmable logic controller

- The processor unit more precisely a microprocessor represents the "brain" and provides "intelligence" of a PLC. It is used for making decisions on which statements of the user program are executed. It carries out instructions of a program and thus a PLC control a process.

  The unit mainly works in a cyclic processing mode (described below). It also supports time-controlled processing for processes requiring signals at constant intervals and interrupt-driven processing if the reaction on some process signals must be fast. [3]

- The internal bus of a PLC is used for communication with all the PLC units (a microprocessor unit, a memory unit, input and outputs units and other interfaces).

  Transferred data is used for addressing, data as information and data for controlling.

- The memory unit provides a space for persistence storing the user program and a storage for data stored from inputs and for outputs modules.

- The input and output modules, are besides the user program, the thing that makes a PLC very universal. These modules are used for monitoring and controlling a process. There are two main groups of the modules based on the type of information - digital and analogue.

  The input modules provide a current state of a process. According to these values a PLC makes a decision by the user program and thus create new outputs.

  The outputs modules provide a current action on a process.

- The power supply unit is used for the conversion of the main AC voltage (230V) to the low DC voltage (24V). It provides a power supply for the PLC components including the inputs and outputs and other interface modules.

- The communication interface is used for transfer data on communication networks from or to other PLCs. Most common is PROFIBUS and PROFINET.
- The special modules are used in various applications that measure a location, there are modules for regulation and Fuzzy logic, modules for diagnostic and many more.

## 2.3 Concept of a cycle

A program or *user program* in a PLC is executed *cyclically*. As the project work is based on Siemens ®SIMATIC PLCs, these following steps of execution describe their solution [1]. Information used in this section are from [1].

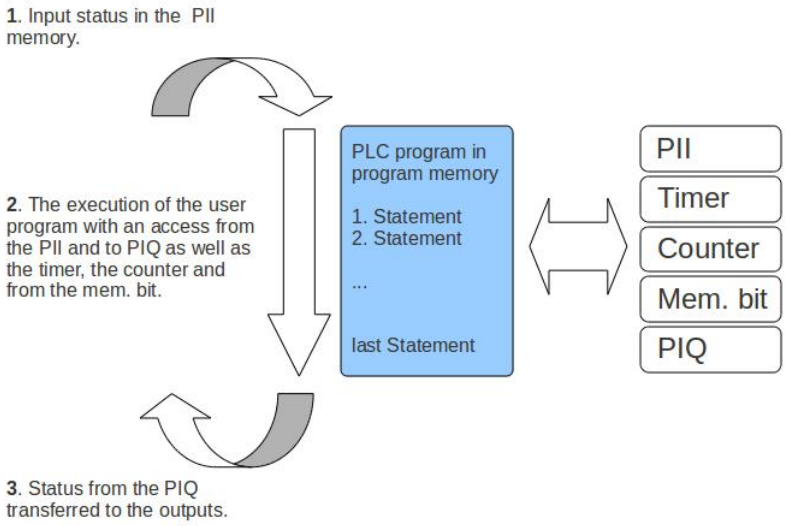



Figure 2.2: Cyclically execution of SIMATIC PLCs

1. When the PLC is switched on, the processor unit questions if the individual inputs have been transmitted or not. If yes, the actual input values (status of the inputs) are stored in the process-image input table (PII).
2. Then the processor unit carries out the user program. The required input information can be already accessed. While proceeding the user program, the new outputs are formed. The results of the outputs are written into a process-image output table (PIQ). Also other storage areas for counters, timers and memory bits can be accessed during the processing of the program.
3. In the third step, the status from the PIQ will transfer to the outputs. After this step, it continues to operate, as seen in point 1 if the PLC hasn't been switched off.

[1] Principles of the cyclically execution are similar to PLCs that are produced by other manufactures.

## 2.4 PLC communication with input and output signals

The inputs and outputs modules more precisely their signals are used for monitoring and controlling a process. A system of the communication between the SIMATIC PLCs and the inputs and outputs modules is described in [1].

The determination of a certain input or output in the program is referred to as addressing. The inputs and outputs of the PLC are mostly defined in groups of eight on digital input or digital output devices. This eight unit is called a *byte*. Every such group receives a number as a byte address.

Every input or output byte is divided into eight individual *bits*. Via the bits it can respond. These bits are numbered from bit 0 to bit 7. Thus one receives a bit address.

Below, there are mentioned following examples of the input and output addressing.

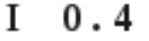



Figure 2.3: Example of an input address

*I* specifies an input, 0 the byte address and 4 the bit address. The byte address and bit address are always separated with a point.


Figure 2.4: Example of an output address

*Q* specifies an output, 5 the byte address and 7 the bit address.

To input and output signals are attached sensors and actors that are used for monitoring and controlling the particular process.

# Chapter 3

# PROFIBUS

## 3.1 Introduction

PROFIBUS (Process Field Bus) is a standardized open field bus system communication. Information used in this chapter can be found in [4], information about PROFIBUS DP in [5].

PROFIBUS is not just one communication system, but a variety of protocols built on the same field-bus technology. Therefore users can combine varieties of PROFIBUS protocols with their own software and other requirements, resulting in a unique application profile. PROFIBUS was formed in 1987 and nowadays is often used in many applications in Europe.

### 3.1.1 Types of PROFIBUS

PROFIBUS has advanced through a few revisions. In some cases, advances have led to a new type of PROFIBUS. In other cases, new revisions mean different versions of the same type of PROFIBUS.

There are three different versions of PROFIBUS:

- PROFIBUS FMS (Fieldbus Message Specification) was designed to communicate between a programmable logic controllers and PCs, sending complex information between them. Unfortunately the FMS technology was not as flexible as needed.

  This protocol was not appropriate for less complex messages or communication on a wider, more complicated network.

- PROFIBUS DP (Decentralized Peripherals) is used in the overwhelming majority of PROFIBUS application profiles nowadays. Application profiles allow users to combine their requirements for a specific solution.

- PROFIBUS PA (Process Automation) standardizes the process of transmitting measured data. PROFIBUS PA was designed specifically for use in hazardous environments.

## 3.2 PROFIBUS DP

In my project I worked with the SIMATIC PLCs that support PROFIBUS DP communication and also the touch panel supports PROFIBUS DP communication. Therefore this

section describes more deeply PROFIBUS DP topology and the communication itself.

### 3.2.1 OSI layer model

The design of the technology modules with PROFIBUS is oriented toward the OSI layer model (Open Systems Interconnection Reference Model). Here, the communication process between two nodes is distributed over seven "layers".

PROFIBUS uses layers 1, 2 and 7:

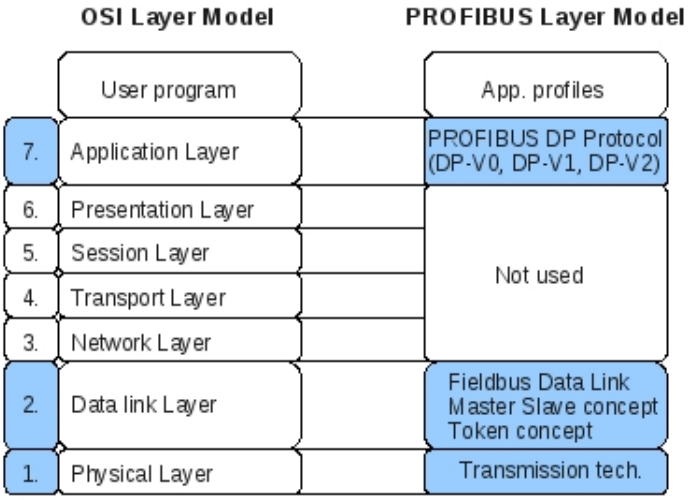



Figure 3.1: References between the OSI model and PROFIBUS

- Layer 1 defines the physical transmission. With PROFIBUS, there are copper-wire versions (RS485 and MBP) and optical and wireless transmission.
- Layer 2 defines the description of the bus access method, including data security. With PROFIBUS, this is the master-slave method in conjunction with the token method.
- Layer 7 forms the interface to the application and thus represents the link between the application and communication. With PROFIBUS, the communication protocol PROFIBUS DP is used here.
- The actual application process lies above layer 7 and it is not part of the OSI model.

### 3.2.2 Communication

PROFIBUS devices communicate using the PROFIBUS DP (Decentralized Periphery) communication protocol, which is the same for all applications and which allows cyclical and acyclical communication and specifies rules for this. The core of the communication process is the master-slave method, where a master (active communication nodes: PLC, PC or control system) cyclically prompt the connected slaves (passive communication nodes: field devices, I/Os, drives) to exchange data.

The polled slave answers the master with a response message. The request message contains the output data, e.g. set point speed of a drive, and the associated response

message contains the input data, e.g. the latest measured value from a sensor. A bus cycle comes to an end once all connected slaves have been polled in order.

In addition to this cyclical communication for the fast exchange of input and output data between the master and slaves at regular intervals, need based data can also be transmitted using PROFIBUS, e.g. device setting data. A master has the initiative, accessing the data of a slave in read or write mode acyclically. There can be more than one master in a PROFIBUS system. In such a case, the access authorization passes from the active master to the next master (token-passing principle).

### 3.2.3 Device classes

PROFIBUS devices are divided into three classes based on their functions:

**PROFIBUS DP master (class 1)**

A PROFIBUS DP master of class 1 (DPM1) is a master which uses cyclical communication to exchange process data with its associated slaves. Devices of this type are often integrated in a memory of a programmable logical controller or an automation station of the process control system.

**PROFIBUS DP master (class 2)**

A PROFIBUS DP master of class (DPM2) was originally defined as a master used as a tool in the context of PROFIBUS system commissioning. Devices of this type are usually part of an engineering station used for device configuration. A DPM2 do not need to be permanently connected to the bus system.

**PROFIBUS slave**

A PROFIBUS DP slave is a passive communication node which reacts to requests from the master by sending a response message. Devices in this class are usually field devices (remote I/O, drive, valve, transducer, analyzer) which acquire process variables or play a part in the process by means of manipulated variables. A differentiation is made between compact and modular slave devices.

A modular device comprises a head station containing the fieldbus interface and a number of slots into which various modules can be inserted. By combining different modules, modular slaves can be adapted flexibly to respond to prevailing requirements with regard to input and output data.

Compact devices have a fixed set of input and output data comparable to a modular device with precisely one permanently installed module.

The majority of slave devices in process automation are modular devices on which, rather than being physically present, the individual modules simply exist in the device software (virtual modules). These virtual modules (and, therefore, access to the associated input and output data) are activated or deactivated when cyclic communication is established.

### 3.2.4 Cyclic communication

When the configuration is loaded on the class 1 master with the help of the configuration tool, then the master establishes cyclical communication with the associated slave devices

(MS0 channel). During this power-up phase, the slave adopts a two-stage approach to checking the configuration data received from the master.

At first, the parameters set in the configuration (e.g. master address, watchdog time and ID number) are transferred to the slave (parameterization) and checked (configuration). The ID number is unique for each device type and is assigned by PI. Cyclical communication can only take place if the ID number from the configuration tallies with the ID number is saved in the slave. Next, the information about the configured modules is transferred to the slave and checked.

Cyclical communication can only be established if the modules which are physically present with those set in the configuration or if the device can adapt to the configuration received.

Successful establishment of communication is then verified via the requested diagnostics data. the slave reports invalid parameter or configuration data to the master through corresponding errors in the PROFIBUS standard diagnostics. If the parameter and configuration data is valid, then the master will initiate cyclical communication with the slave device.

# Chapter 4

# Hardware

## 4.1 Composition of the system

The system, as was defined in the Chapter 1, consists of following hardware parts that are described in these sections. It is formed by the pick and place machine, SIMATIC S5 PLC, SIMATIC S7 PLC, SIMATIC TP177b DP/PN touch panel and PROFIBUS connection.

## 4.2 Pick and place machine

The machine is used for grouping of coil cores for an anti-lock braking system (ABS). More precisely, it distributes coil covers onto coil cores and the coil cores are grouping together in two rows of four each. After this, the process is repeated for another set of coil cores.

The system is pneumatic. The machine works in either an automatic mode or in a manual mode. An operator of the machine chooses between the modes.

The machine is originally controlled by a SIMATIC S5 PLC.

### 4.2.1 Schema

The schema of the machine is displayed on Figure 4.1 below. The machine consists of the followings parts:

- The stock is basically used for storing and supplying the coil covers to the process. One more function is positioning the coil cover into a position when the bottom part of the cover is down. The stock is made by the upper part and lower part.
- The upper stock is used for supplying the coil covers to the process. The bottom contains a vibrating surface that is used for movement of the coils covers. The coil covers fall into the lower stock.
- The lower stock is used for storing and positioning the coil covers to the right position. It is spiral shaped and it also contains a vibrating surface for transporting the covers to the conveyor band.
- The conveyor band is used for transporting the covers from the stock to the deck.
- The deck is mainly used for queuing the covers. The covers are arranged in rows of four pieces. On the deck there also operate other components (the pressing pad, the

blocking stick and the lift table) that are used for right positioning that fits the coil cores.

- The pressing pad, the blocking stick and the lift table are used for positioning the covers that fits the coil cores.
- The gripper 1 and 2 are used for transporting the covers right on the coil coils. The coils are stored in a box.

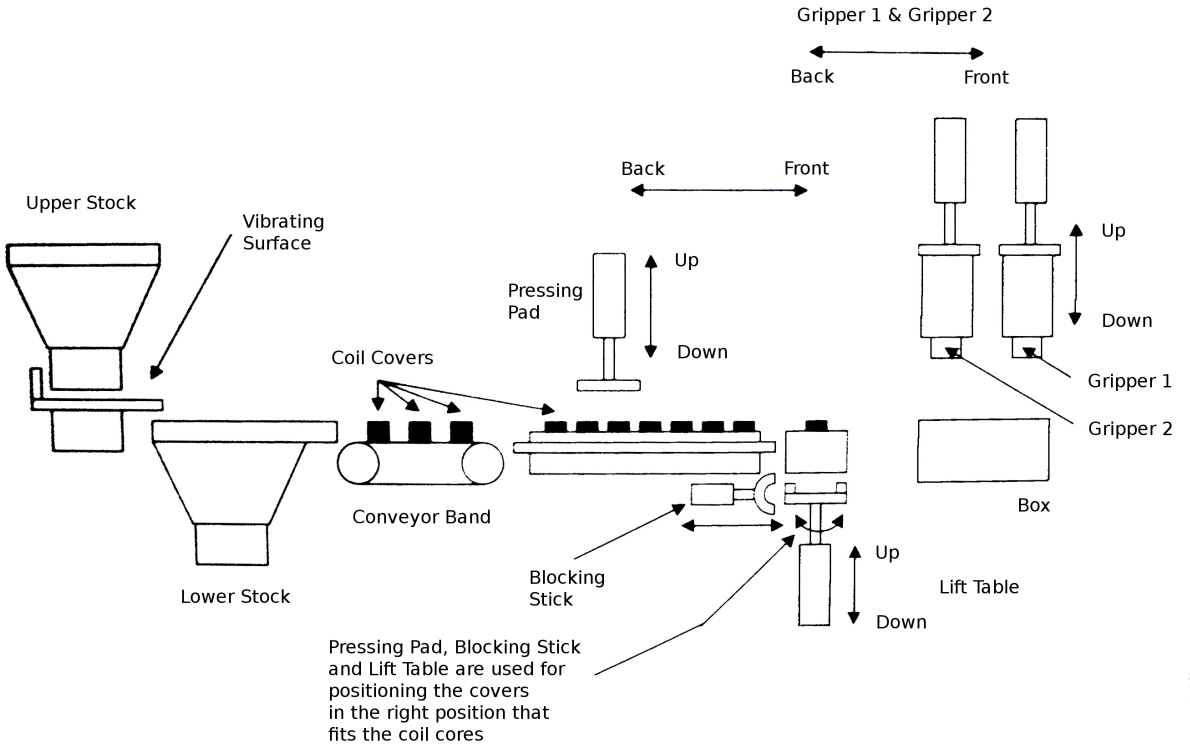



Figure 4.1: Schema of the pick and place machine

### 4.2.2 Operation modes

The pick and place machine operates in either an automatic mode or in a manual mode. In the case of emergency there is also an emergency mode.

**Automatic mode**

By pressing a start button the machine starts operation. A switch has to be also in a position for the automatic mode. The coil covers are supplied from the stock. Once the covers are arranged in the row of four in the right position and the box that stores the coil cores is in the right position as well, then the covers are put onto the coil cores. At first, the first row is put by the gripper 1. That completes one cycle of the process. Then the machine continues again for the gripper 2. After that, the cycle of the procedure is completely. This mode of the machine works in an infinitive loop.

**Manual mode**

Before the manual mode can start, the switch has to be in a position for the manual mode. This mode is completely controlled by an operator who, by pressing a step button, controls the process.

The result of the mode is exactly the same as in the automatic mode.

**Emergency mode**

By pressing an emergency button this immediately invokes the emergency mode. It could also be invoked by opening any of the doors that cover the machine.

All components are immediately set into the safe position. After releasing the emergency button or closing the doors, the machine is set into the default starting position.

## 4.3 SIMATIC S5 PLC

The SIMATIC S5 PLC consists of following parts:

- CPU: 115U 942
- Power supply: M24-20, input 230 V AC, output 24 V DC, 20 A
- Input modules: 2x Digital Input 32x24 VDC
- Output modules: 1x Digital Output 32x24 VDC

There is no interface for communication via PROFIBUS or PROFINET.

### 4.3.1 Controlling of the pick and place machine

The SIMATIC S5 PLC inputs and outputs modules are assigned to sensors and actors of the machine. Inputs represent the status of the machine e.g. a sensor that indicates if gripper 1 and gripper 2 are in the upper position. Outputs are used for actions e.g. a movement of gripper 1 to the lower position.

Inputs start at address I 0.0 and outputs start at address Q 40.0.

## 4.4 SIMATIC S7 PLC

The SIMATIC S7 PLC (SIMATIC S7-300, 315-2EG10-0AB0 station) consists of the folowing parts:

- CPU: 315-2PN/DP
- Power suply: PS307, input 230 V AC, output 24 V DC

There is also available PROFIBUS DP interface and PROFINET interface for a communication.

## 4.5 SIMATIC TP177b DP/PN

SIMATIC TP177b DP/PN is a 5.7-inch touch screen panel. It allows an operator to control and monitor machines and plants.

Interfaces for communication with Siemens ®SIMATIC S7 PLC e.g. via MPI, PROFIBUS DP, PROFINET are on-board and available to use.


Figure 4.2: Touch panel SIMATIC TP177b DP/PN. FOTO: Siemens AG.

## 4.6 Communication bus

As a communication bus it is used PROFIBUS DP as assigned. This allows communication with SIMATIC S5 PLC and SIMATIC S7 PLC and finally controlling the machine by SIMATIC S7 PLC and the touch panel.

# Chapter 5

# Software

This chapter is aimed to describe a software part used in the system as a software package and programming languages that could be used for programming the machine in SIMATIC S7 environment. There is also mention of GRAFCET; a graphical representation of algorithms of automated systems, which are used to represent a functionality of the pick and place machine.

## 5.1 Combination of hardware and software

According to the definition of a program logic controller in Chapter 2, a PLC is flexible to use in many applications. The flexibility is mainly given by programming a set of instructions, as it is called a user program. The same basic controllers can be used in many applications for controlling a wide range systems mainly by just changing the program of the PLC. The software part is thus very important.

## 5.2 STEP 7

STEP 7 is a basic software package that provides configuring and programming SIMATIC PLCs. Information used to describe STEP 7 can be found in [6]. The STEP 7 Standard package includes a series of applications (tools) within the software:

- SIMATIC Manager
- Symbol Editor
- NETPRO Communication Configuration
- Hardware-Configuration
- Programming Languages
- Blocks for structuring the program
- Hardware Diagnostics

These tools and applications are used through the process of setting up and managing a project, configuring and assigning parameters to hardware, network setting and configuration, creating programs and programming, simulating and debugging, downloading

a program to a PLC and for testing the automation system. Primarily used application is an environment SIMATIC Manager used for configuring, programming and simulating PLCs.

The STEP 7 Standard package can be extended by optional software packages. These packages provide higher-level programming languages, another technology oriented software and HMI software used for operator control and monitoring. In the project, it uses just HMI software for monitoring and visualization the process.

## 5.3 Programming Languages

With STEP 7 it is possible to create S7 programs in standard languages Ladder Logic (LAD), Function Block Diagram (FBD) or Statement List (STL). Before writing a program, the author needs to decide which language is suitable to use for the particular application.

Following examples below represent, how to implement AND function in LAD, FBD and STL language. Inputs are located at addresses 0.0 and 0.1 and an output is located at address 0.1.

Information used to describe the programming languages was found in [6].

- Ladder Logic Programming Language (LAD) is a graphic representation of the STEP 7 programming language. The syntax for the instructions are similar to a relay ladder logic diagram that enables easy tracking of the signal flow between power rails as it passes through various contacts, complex elements, and output coils.

  The program consists of individual LAD elements arranged in a series that is parallel to one another. Programming of a current path, or rung, begins on the left power rail. [9]

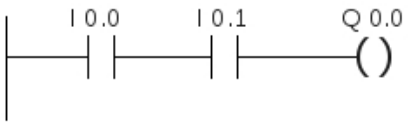



Figure 5.1: Implementation of AND function in LAD

- Function Block Diagram Programming Language (FBD) is also a graphic representation of the STEP 7 programming language and uses the logic boxes familiar from Boolean algebra to represent the logic. Complex functions (for example, math functions) can be represented directly in conjunction with the logic boxes.

  While programming in FBD it is also easy to track the signal flow and the program in FBD can be transferred to STL language as well.

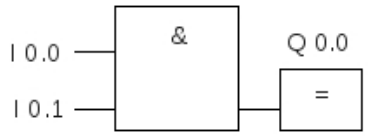



Figure 5.2: Implementation of AND function in FBS

- Statement List Programming Language (STL) is a textual representation of the STEP 7 programming language, similar to some kind of an assembly language. Information to describe STL language was found in [8]. An STL program consists of statements. A *statement* is the smallest autonomous unit of a user program. It represents a work specification for the CPU.

  To make programming easier and more transparent, STL includes some higher-level language constructions e.g. structured data access and block parameters.

```
A     I     0.0
A     I     0.1
=     Q     0.0
```

Figure 5.3: Implementation of AND function in STL

## 5.4 Blocks for structuring a user program

With STEP 7 it is possible to structure the user program into *blocks*. This means it is possible to divide the entire program into smaller units. As a block we can imagine a single unit of instructions (statements) that can be called as a function or procedure, it could also be a unit that stores some data; there is the main block which is called every cycle and many other blocks that are used for system purposes.

This concept of structuring into blocks has important advantages such as larger programs are easier to understand, an individual section can be standardized, we can use procedures for repeat calling, functions as an abstraction of a problem and any modification of the entier program is therefore easier.

### 5.4.1 Block types

There are several types of blocks that are mentioned below. The blocks of the same type are distinguished by numbers. Information used in this subsection can be found in [6].

**Organization blocks (OBs)**

OBs represent the interface between the operating system of a PLC and the user program. By programming OBs the CPU behavior is determined. They also determine the sequence (start events) by which individual program sections are executed. An OB call can

interrupt the execution of another OB. Which OB is allowed to interrupt another OB depends on its priority. Higher priority OBs can interrupt lower priority OBs. The lowest priority has Organization Block for Cyclic Program Processing (OB1) .

They are used for cyclic controlling, start-up behavior of the PLC, interrupt-driven program executing, error handling and other system purposes.

The most important OBs are:

- Start-up Organization Blocks OB100, OB101 and OB102 are used for initialisation. Every time the status of the CPU changes from the STOP status to RUN status the start-up organization blocks are called. After execution, the OB1 is called. The priority level (class) is 27 (one of the highest).
- Organization Block for Cyclic Program Processing The OB1 is an organization block that starts the user program. The operating system calls OB1 cyclically. The priority level (class) of the OB1 is 1 (the lowest).
- Hardware Interrupt Organization Blocks The OB40 to OB47 are used for a hardware interrupt. When the interrupt is detected, the device driver calls the right OB (from OB40 to OB47) if it has a higher priority level (class) then the block that is currently executing. Otherwise the interrupt is ignored. The priority level (class) of OB40 is 16 and it increase until OB47 that has 23.

**Functions (FCs)**

It is possible to program the whole user program into the OB1, but according to the advantages mentioned above and "good programming practices" it is proper to structure the program. Functions are mainly used for this reason. A function is a logic block "without memory". Temporary variables belonging to the FC are saved in the local data stack. This data is then lost when the FC has been executed. To save data permanently, functions can also use shared data blocks. Therefore, we can say a function is a procedure in higher programming languages if it is called regularly or a unit for structuring the program.

**Function blocks (FBs)**

Function blocks are another way to structure the program. The difference between a function and a function block is that the function block has a memory. A function block is a block "with memory". It is assigned a data block as its memory (instance data block). The parameters that are transferred to the FB and the static variables are saved in the instance DB. Temporary variables are saved in the local data stack. Data saved in the instance DB are not lost when execution of the FB is complete. Data saved in the local data stack are, however, lost when execution of the FB is completed.

**Data blocks (DBs)**

Data blocks are used for storaging some data such as bits vectors used as samples, alphanumeric values and for storing the actual parameters and the static data of the FB . Therefore we distinguish Instance Data Blocks that are related to a FB and Shared Data Blocks that are used for storing user data.

**System Functions (SFCs)**

The user, besides his programmed blocks, can also use preprogrammed functions (SFCs) and blocks (SFBs) that are provided. A system function is a preprogrammed function that is integrated on the S7 CPU. You can call the SFC in your program. SFCs are part of the operating system and are not loaded as part of the program. Like FCs, SFCs are blocks "without memory". S7 CPUs provide SFCs for the following functions:

- Copying and block functions
- Checking the program
- Handling the clock and run-time meters
- Transferring data sets
- Addressing modules
- Distributed I/O
- Global data communication
- and others

**System function blocks (SFBs)**

A system function block (SFB) is a function block integrated on the CPU. SFBs are part of the operating system and are not loaded as part of the program. Like FBs, SFBs are blocks "with memory" so the user has to create instance data blocks as well. S7 CPUs provide the following SFBs:

- for communication via configured connections
- for integrated special functions

### 5.4.2 Block call

The main block that starts the user program is OB1. From the OB1 other blocks that form and contain the user program is usually called. The following figure shows an example of a block call. Calling block calls the called block. After the call, the execution of the program comes to the called block and instructions there are completely executed. After the called block is ended, the execution comes back to the calling block right under the call instruction and the execution keeps continuing in the calling block.

The block from which is able to call other blocks can be OB, FB or FC. The called block must be one of FB, FC, SFB or SFC. For example, from an OB isn't possible to call another OB. The block call is made by instructions. In STL programming language we can use either an unconditional block call or a conditional block call. The parameter of the instruction is the particular called block.

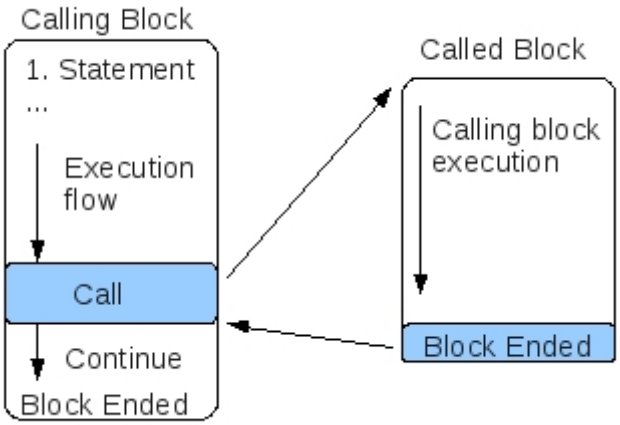



Figure 5.4: Example of calling a block

## 5.5 GRAFCET

Information from [10] was used in this section. The GRAFCET was originally a methodology intended for the design and graphical representation of the command algorithms of an automated system. With its rigorous correspondence with the logic to be programmed have led certain manufacturers to transform GRAFCET into a programming language.

The GRAFCET language can be used in either a graphical or a literal representation. The machine is described by the graphical form. The graphical form uses elementary diagrams which are put together to express the meaning - step, associated command, transition, symbol of convergence or divergence, with the possibility of expressing logic conditions.

The graphical representation of GRAFCET is made up of a set of stages, arcs, transitions, labels and receptivities.

**The stage**

The stage is a situation of the system in which all or part of the control unit is not changed with respect to the input-outputs of the automated system. Conventionally a stage is representation by a square numbered in its upper part. The initialization stage is marked by a double outline.

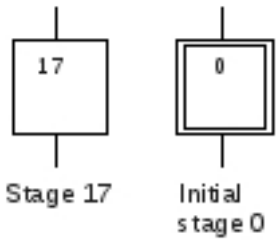



Figure 5.5: Example of a stage and an initial stage

**Actions associated with a stage**

With each stage there can be associated actions on the system. These actions are specified in a rectangle, the label, situated at the right of the stage symbol.

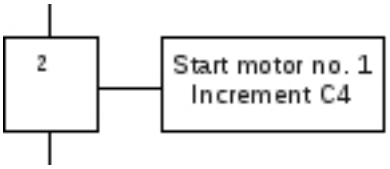



Figure 5.6: Example of an action

**Transition and receptivity**

A transition is a "barrier" necessary in separating two successive stages. The receptivity associated with a transition is a Boolean function expressing the logic condition which allows the clearance of the transition (necessary condition).

The satisfaction of a receptivity is still not a sufficient condition for the freeing of a transition: it is also necessary for the previous stage to be activated.

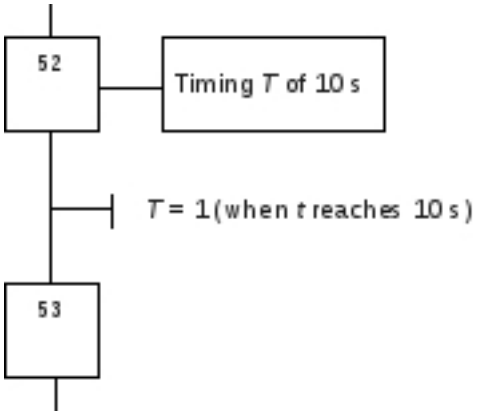



Figure 5.7: Example of a receptivity associated with a transition

**ARCS**

Arcs are a directed straight line segment linking to a transition, or a transition to a stage, but never two transitions or two stages together.

By convention the vertical direction from top to bottom is not marked with arrows (implicit direction).

# Chapter 6

# Design of the system

## 6.1 Description of the system

The system consists of the three basic parts. The first part is the pick and place machine, second is SIMATIC STEP 5 PLC which inputs and outputs modules are assigned to the sensors and actors of the machine so it originally controls the machine and the third part is SIMATIC STEP 7 PLC that should completely control the machine instead of STEP 5 PLC. The part of the system is also SIMATIC TP177b DP/PN touch panel that is used for a visualization and controlling the machine as well.

The system is illustrated on Figure 6.1 below. The design of the whole system can be divided into software and hardware part.

The hardware part describes a design of the communication via PROFIBUS DP that allows a communication with SIMATIC S5 PLC and SIMATIC S7 PLC and finally controlling the machine by SIMATIC S7 PLC.

The software part consists of a design of a program for SIMATIC S7 PLC that controls the machine and an application for the touch panel.

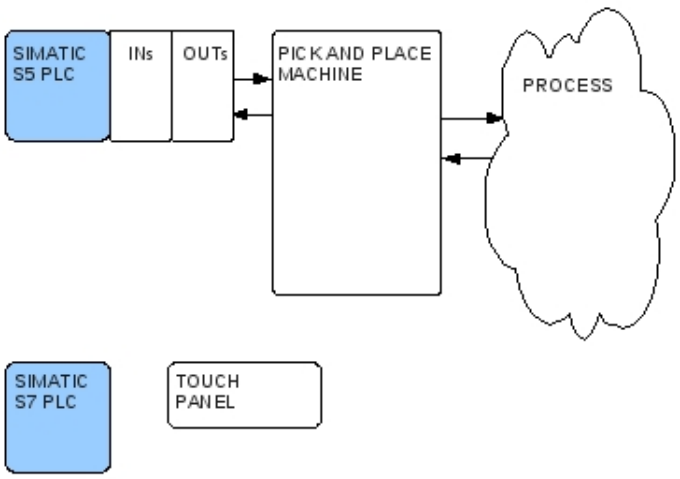



Figure 6.1: Graphic description of the system

## 6.2 Design of the system communication

The concept of this system is that both PLCs and the touch panel are connected via PROFIBUS DP and that allows a communication between them and finally provides a control of the machine by SIMATIC S7 PLC.

The problem of this part is to find a way how to connect both PLCs and the touch panel via PROFIBUS DP.

### 6.2.1 PROFIBUS DP connection for SIMATIC S5 PLC

The CPU 115U 942 of the SIMATIC S5 PLC is not available with an integrated PROFIBUS DP interface; therefore, there is a need to find an interface module which allows a communication via PROFIBUS DP.

I chose an interface module IM 308-C DP for the CPU 115U 942 of SIMATIC S5 PLC.

**Interface module IM 308-C DP**

Information used in this paragraphs are from [7]. Module IM 308-C DP is a PROFIBUS DP master and/or slave module for SIMATIC S5-115U/H up to S5-155U/H. It provides up to 122 passive nodes. The IM 308-C enables connection between the distributed I/O stations to the S5-115U, S5-135U and S5-155U programmable controllers via the PROFIBUS-DP bus.

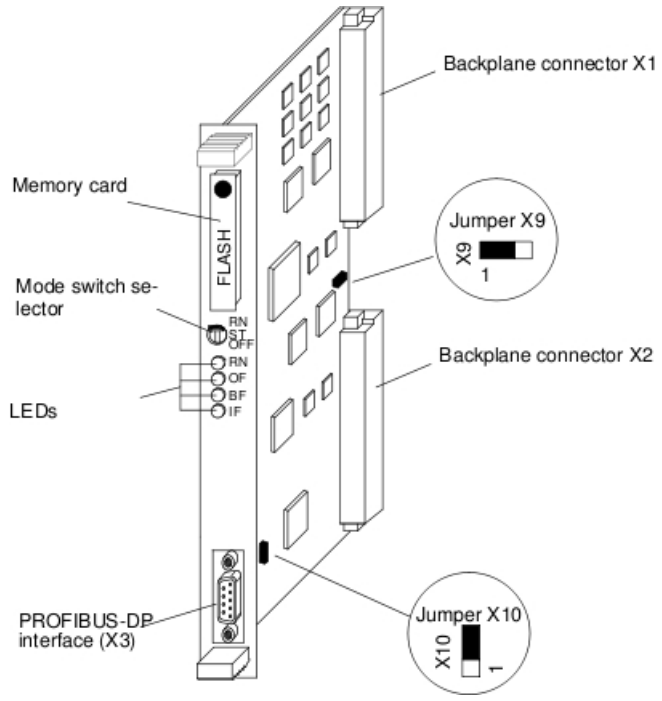



Figure 6.2: Interface Module IM 308-C DP. Figure reprinted from [7].

More information about its functionality and setting can be found in [7].

### 6.2.2 PROFIBUS DP connection for SIMATIC S7 PLC

The CPU 315-2PN/DP of SIMATIC S7 PLC is a CPU that is made available with an integrated PROFIBUS DP interface.

### 6.2.3 PROFIBUS DP connection for SIMATIC TP177b DP/PN touch panel

The SIMATIC TP177b DP/PN touch panel is made available with an integrated PROFIBUS DP interface as well.

## 6.3 Design of the application for the pick and place machine

The specification of the pick and place machine can be found in Section 4.2. Its functionality as the automatic, manual and emergency mode is described by GRAFCET schema.

In this section there are terms used from Chapter 5. There is a describtion of the GRAFCET schema of the machine and also a concept of the block structure of the program.

### 6.3.1 GRAFCET schema

The GRAFCET schema of the pick and place machine has two initial stages M10.0 shared with the automatic and manual mode and M12.0 for the emergency mode.

The automatic or manual mode starts either when a start button is pressed or a step button is pressed. After the start, there have to be proceed out a several actions that turn on the upper stock, the vibrating surface, the lower stock, the conveyor band and open the main valve. Then the schema continues with other conditions and after that with another stage. Once the machine ends the first cycle of moving the covers onto the coil coils, it continues again for the other gripper.

The emergency mode could be invoked immediately by pressing an emergency button or if any of the doors that cover the machine are opened. Once the emergency mode completes all stages, the emergency button is unpressed and all doors are closed, the initial stages are set as default.

### 6.3.2 Block design

The concept of the program for the pick and place machine consists of following block structure. There is used OB1 object and two functions objects. I chose functions instead of functions blocks, because there is no need of static memory for remembering any variables, when there is work just with inputs and outputs. There is also use of OB100 for initializing the two initial stages that is called immediately after the run mode of the PLC.

OB1 is cyclically called and from it, the FC1 and FC2 are called. At first, FC1 is called which represents the logic conditions and after that FC2 is called which represents the actions.

**OB100**

The OB100 is used for initialization. Every time the status of the CPU changes from the STOP status to RUN status the start-up organization block OB100 is called. The block initialize the two initial stages M10.0 and M12.0.

**OB1**

After execution the 0B100 the OB1 is cyclically called. OB1 starts the program that drives the pick and place machine. In this block two functions FC1 at first and after that FC2 are called. After STOP or SHUT DOWN the PLC ends its work.

**FC1**

FC1 contains the logic conditions. After satisfaction, the condition and with the active previous stage, it is possible to deactivate previous stage and activate the next stage.

**FC2**

FC2 contains the actions. There the instructions are carried out under the active stage.

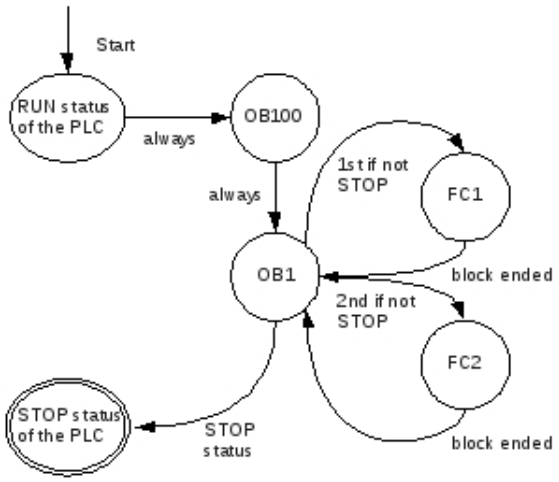



Figure 6.3: Graphic description of the block design

## 6.4 Design of the application for the touch panel

SIMATIC TP177b DP/PN touch panel should be used for the control and visualization of the pick and place machine. The application should be created with SIMATIC Wincc flexible 2005 environment.

The application should allow to view the entire process schema, it should also visualize the status of the machine and there should be some acting components for limited controlling of the machine.

# Chapter 7

# Implementation of the system

In this chapter, the whole system implementation is described. At first, the implementation of the communication, then the program for the pick and place machine and the application for the touch panel.

## 7.1 Implementation of the system communication

In this section, there is used the previous design of the communication discussed in Section 6.2 and also information about PROFIBUS DP presented in Chapter 3.

Once we have the required hardware parts, it is possible to start implementing the communication. The concept of the system communication works with two master stations and one slave station.

The masters stations is the SIMATIC S7 PLC and the SIMATIC TP177b DP/PN touch panel.

The slave station is the interface module IM 308-C DP.

### 7.1.1 PROFIBUS DP setting

There was used the following setting of the communication using PROFIBUS DP:

- Profile: DP
- Transmission rate: 1.5 Mbps
- Highest station address (HSA): 126
- Default bus parameters and theirs cyclic distribution

There was used the following setting of the devices:

- CPU 315-2PN/DP (SIMATIC S7 PLC): master (class 1), address 24
- SIMATIC TP177b DP/PN (touch panel): master (class 1), address 31
- IM 308-C DP (communication interface of the SIMATIC S5 PLC): slave, address 16

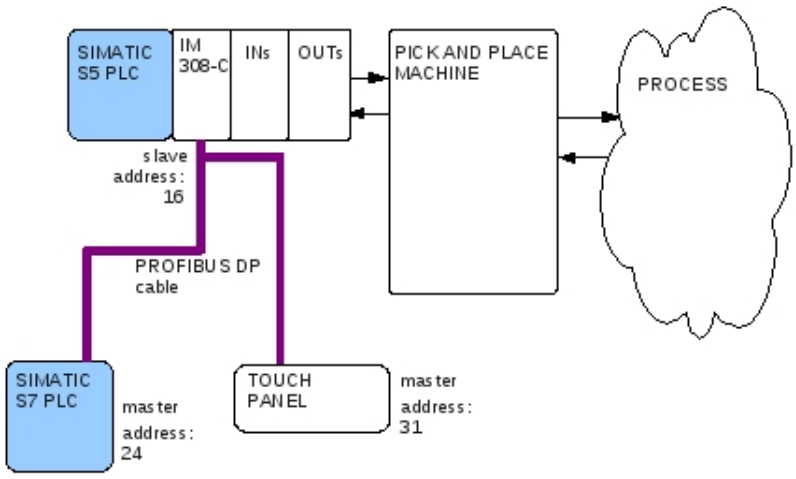



Figure 7.1: Graphic description of the system communication

### 7.1.2 IM 308-C DP setting

The module IM 308-C DP is set as a slave with address 16. It is also set to contain three digital inputs and three digital outputs sections, which are used to exchange data between the machine inputs, outputs and the SIMATCI S7 PLC program, as explained below.

Table 7.1: Digital inputs and digital outputs sections in the module IM 308-C DP

| Designation | Input Address | Output Address |
|---|---|---|
| 16bytes DI/constcy 1 byte | 50 .. 65 | |
| 16bytes DI/constcy 1 byte | 66 .. 81 | |
| 16bytes DI/constcy 1 byte | 82 .. 97 | |
| 16bytes DO/constcy 1 byte | | 50 .. 65 |
| 16bytes DO/constcy 1 byte | | 66 .. 81 |
| 16bytes DO/constcy 1 byte | | 82 .. 97 |

The digital inputs from address I 50.0 represent the real digital inputs used in the machine that start from I 0.0. The same is valid for the digital outputs from address Q 50.0 and in the machine Q 40.0. We need to represent the inputs and outputs like that, because it is not possible to change the real outputs directly nor a direct access to the real inputs.

The program in SIMATIC S7 PLC works with these inputs and outputs addresses; therefore, there is a need to translate the inputs and outputs addresses into to the real addresses. The translation is made by a short program, which is ran by SIMATIC S5 PLC as explained below.

### 7.1.3 Communication between the SIMATIC S7 PLC and the machine

The communication between the SIMATIC S7 PLC and the machine is proceeded indirectly through the module IM 308-C DP, more precisely through its inputs and outputs addresses via PROFIBUS DP. The communication is in detail described below.

The SIMATIC S7 PLC runs a program that controls the machine.

The SIMATIC S5 PLC runs a short program that reads inputs from the machine (from address 0.0) and writes them as outputs (from address 50.0) by using the module IM 308-C DP. There are available to the SIMATIC S7 PLC. The program (of the SIMATIC S5 PLC) also reads inputs from the module IM 308-C (from address 50.0), which were changed by the SIMATIC S7 PLC program and writes them as outputs to the machine (from address 40.0).

## 7.2 Implementation of the pick and place machine

### 7.2.1 Programing language

As a programing language for the program, I chose Statement List (STL), because it is based on a kind of assembly language and because of my computational science background. STL program is hard for maintenance; therefore, the previous block design was very important.

### 7.2.2 Concept of the program

The implementation implements the GRAFCET schema of the machine and uses the block structure, as it was designed and described in Section 6.3.

The principle of the stages is to do the particular actions. The conditions check if the particular actions have been executed. Once the actions are carried out, the conditions that check them are set to true and the program can continue with the next stage.

The stages were implemented as memory bits e.g. the initial stage M10.0 is implemented as the first bit (bit 0) in memory address M10.

The transitions were implemented as logical functions.

### 7.2.3 Important parts

While implementing the GRAFCET shcema of the machine; there had to be solved following problems, how to implement a timer and a possitive edge detection.

**Timer**

The GRAFCET schema of the machine contains a several parts of waiting. This part was implemented by using a timer. I used an on-delay timer.

Here is an example how to use an on-delay timer.

```
A   M10.5    // If M10.5 stage is active.
L   S5T#10s  // Load 10 seconds into accumulator (ACCU 1).
SD  T1       // Start timer T1 as an on-delay timer.

A   T1       // Check the expiration of timer T1.
```

**Possitive edge detection of a step button**

The manual mode is controlled by a step button. When it is pressed, the program should move to the next stage.

Here is an example how to detect a positive edge of an input (a step button). There are also used two auxiliary variables (bit M140.0 and M140.1). FP instruction detects a rising edge when the RLO (satisfaction of a condition or conditions) transitions from "0" to "1" and indicates this by RLO = 1.

```
A       "Step"          // If the step button pressed.
FP      M140.0          // FP detection - transitions from "0" to "1".
=       M140.1          // It is indicated by RLO = 1.  Saved in M140.1.

A       M10.0           // If M10.5 stage is active.
AN      "Switch_AuMa"   // And the manual mode is set.
A       M140.1          // And M140.1 (where is saved the result).

 ...                    // Move to the next stage.

R       M140.1          // Reset M140.1 (for next usage).
```

## 7.3 Implementation of the touch panel

The application for the touch panel was created in SIMATIC Wincc flexible 2005 environment that is an extended application of Siemens ®STEP 7 software package. The created application includes four views on the process and machine.

On the first Figure 7.2 is displayed the overview of the process. This view is displayed after starting up the application. There is located a lamp that indicates if the machine is working or not, there is displayed if the machine is working in the automatic or manual mode, also there are located lamps that by flickering indicate if the emergency button is pressed or if any doors is opened and lamps that are used to display the status of other parts of the machine. By pressing the "Process" button or by pressing the area that indicate if "Machine is working", the second screen is activated.

The second view on the Figure 7.3 displays the entire process. This screen is composed as an overview of all the parts of the machine. There are also lamps that indicate the current operation mode of the machine and emergency status. Every part of the machine is made as a schema symbol that signifies the status of activating or deactivating. If the schema line is displayed with green color, it means that the particular part is active, otherwise it displays with black color. There are also lamps that are supplying sensors of the machine. If the area of the lamp is filled by yellow color, it indicates that the part is located in the particular position by the sensor. The sensors are mostly used to indicate a location of the part or appearance e.g. appearance of the covers in the queue. There is also used "START" and "STEP" button. These buttons are used for controlling the machine. An operator can choose between the buttons located on the machine or buttons used on the touch panel - the result is the same.

The last two screens provide a detail view on the process of positioning the covers and on grippers. Thus, an operator has a clear imagine about which parts are currently using.

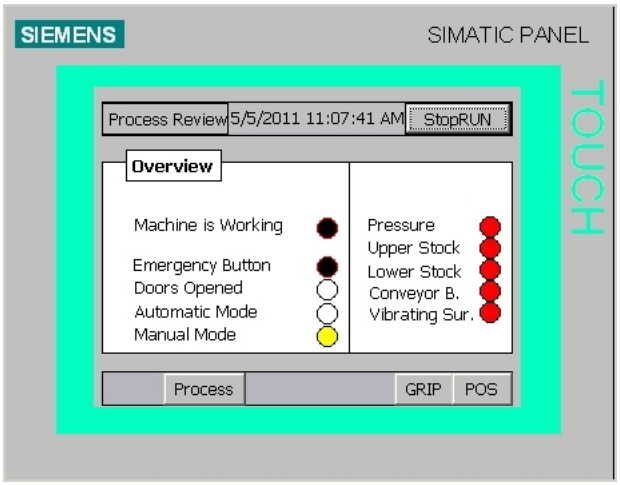



Figure 7.2: Overview screen

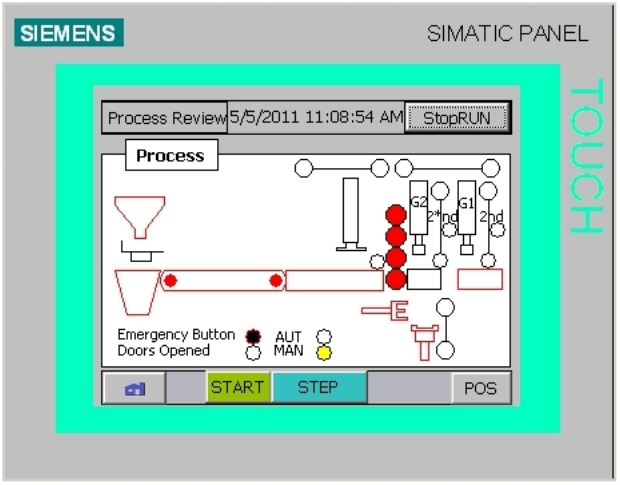



Figure 7.3: Process screen

# Chapter 8

# Conclusion and Future Work

The main object of this bachelor's thesis was to propose and describe a method of how to control a SIMATIC S5 machine by a newer SIMATIC S7 PLC using PROFIBUS. We successfully designed and implemented the system of communication, the program controlling the machine in STEP 7 STL language and the application for the touch panel. All requirements were accomplished.

The project work has already been used by other domestic students who had their laboratory concerning the implementation of the machine. They could use all of advantages that come with STEP 7 environment.

The work could be used as an idea of a design and implementation of the program for the machine. The theoretical part could also be useful for future international students as a description of the machine in English language.

This thesis could also serve as an overview of an automated system and give to a reader a slight imagine about it.

The future work might be rewiring the inputs and outputs of the machine to a newer type of PLC. This method of how to control the machine by a newer PLC provides an incomparable programming comfort, but there is still a need for the use of the old PLC. This solution illustrates the problematic of communication, a student can compare both environments, notify the progress and programing comfort, but it is not suitable for a practical use. If we wanted to implement and use the machine in reality, the ideal solution would be replacement of the old PLC by a newer type and rewire the inputs and outputs.

# Bibliography


[1] Module A3 - 'Startup' PLC Programming with STEP 7. [online] http://www.automation.siemens.com/mcms/sce/en/advanced_training/training_material/download_training_material/a_basics_step7_programming/Documents/a03_startup.pdf, 02/2002. Manual.

[2] *Programmable Controllers – Part 1: General Information.* International Electrotechnical Commission, IEC 1131-1, 1992, (also British Standard BS EN 61131:1994).

[3] SIMATIC S5, S5-135U, CPU 928B - Version - 3UB21, 1996. Programming Guide.

[4] PROFIBUS System Description, Technology and Application. [online] http://www.profibus.com/nc/downloads/downloads/profibus-technology-and-application-system-description/download/9592/, 2008.

[5] PROFIBUS, Comprehensive Protocol Overview. [online] http://www.rtaautomation.com/profibus/, April, 2011.

[6] SIMATIC, Programming with STEP 7, Edition 03/2006. Manual.

[7] SIMATIC S5, ET 200 Distributed I/O System. [online] http://support.automation.siemens.com/WW/llisapi.dll/csfetch/1142470/ET200_e.pdf?func=cslib.csFetch&nodeid=1142140, Edition 04, 1995. Manual.

[8] Hans Berger. *Automating with STEP 7 in STL and SCL.* Publicis Corporate Publishing, 3rd revised edition, 2005. ISBN 3-89578-243-2.

[9] Hans Berger. *Automating with STEP 7 in LAD and FBD.* Publicis Corporate Publishing, 4th edition, 2008. ISBN 978-3-89578-297-8.

[10] Gilles Michel. *Programmable Logic Controllers, Architecture and Application.* John Wiley & Sons, 1990. ISBN 0-471-92463-6.

# Appendix A

# CD Contents

- `README` – software description and basic usage
- `src/` – source codes
- `doc/` – text

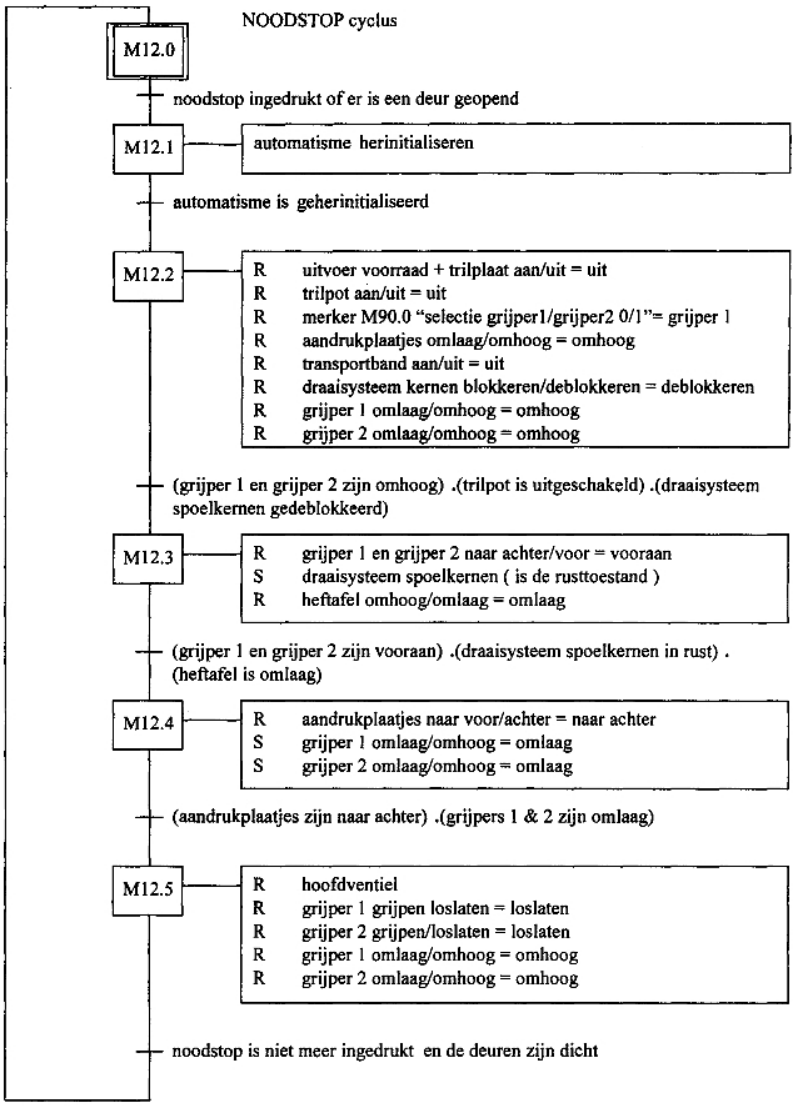
# Appendix B

# List of Abbreviations

| | |
|---|---|
| PLC | programmable logic controller |
| CPU | central processor unit |
| AC | alternating current |
| DC | direct current |
| LAD | ladder logic |
| STL | statement list |
| FBD | function block diagram |
| OB | organisation block |
| FC | function |
| FB | function block |
| DB | data block |
| SFC | system function |
| SFB | system function block |
| FMS | fieldbus message specification |
| DP | decentralized peripherals |
| DP | process automation |
| MPI | multi point interface |

# Appendix C

# GRAFCET Schema of the emergency mode



# Appendix D

# Selection of the mode in SIMATIC STEP S7 PLC

```
// Selection of the mode

//M10.0 is def set in OB100

// M10.1
// - Automatic Mode
 A M 10.0
 A "Switch_AuMa"
 A "Start"
 A "EmButton_UP"
 AN "Door_CO"
// - Manual Mode
 O(
 A M 10.0
 AN "Switch_AuMa"
 A M 140.1
 R M 140.1
 )
 S M 10.1
 R M 10.0

//M12.0 is def set in OB100

// M12.1 - Emergency Mode
 A M 12.0
 A(
 ON "EmButton_UP"
 O "Door_CO"
 )
 S M 12.1
 R M 12.0
```

# Appendix E

# Program in SIMATIC S5 PLC

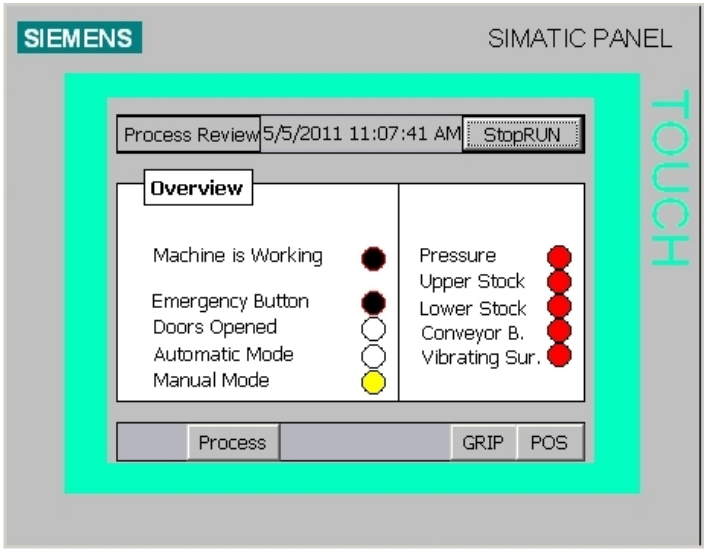
[1]

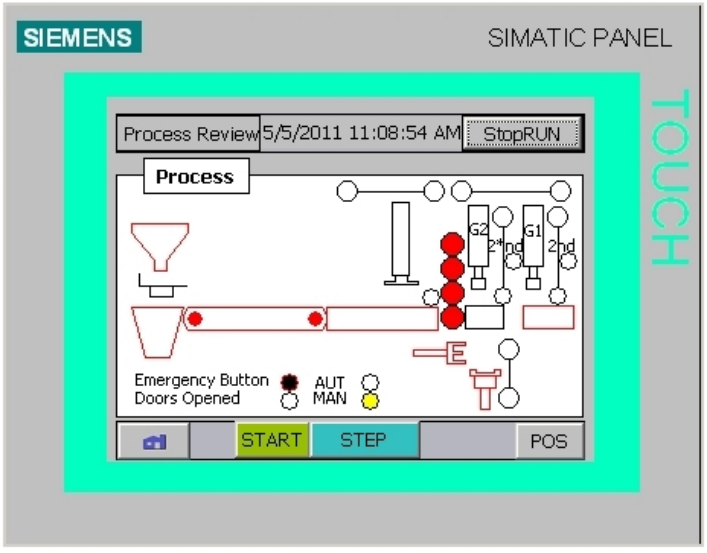
```
L EW 50
T AW 40
L EW 52
T AW 42

L EW 0
T AW 50
L EW 2
T AW 52
L EW 4
T AW 54

BE
```

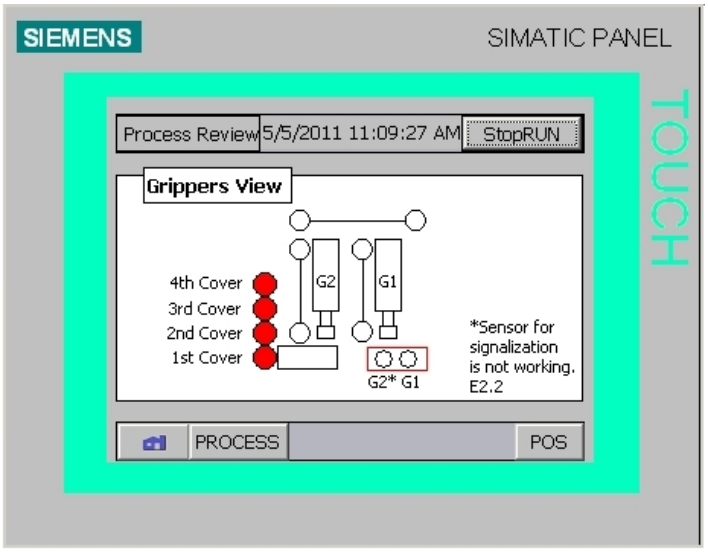
[1]The S5 envrinment was in German language. E (Eingabe) means input (I), A (Ausgabe) means output (Q).

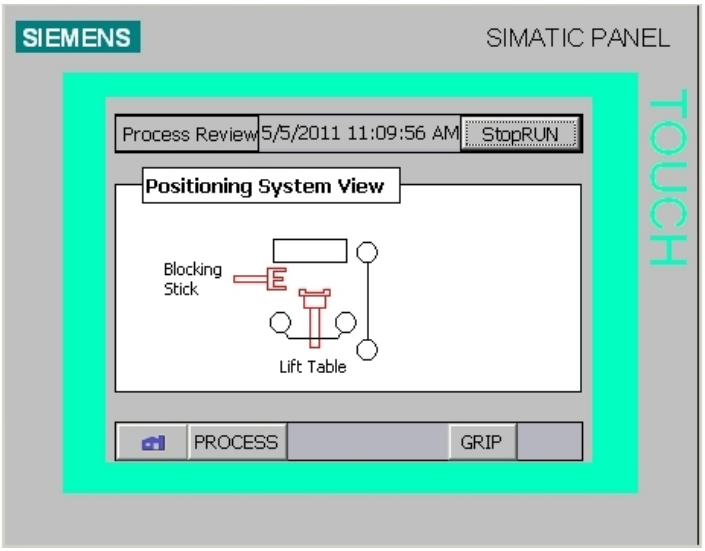
# Appendix F

# Touch panel screens




SIEMENS
SIMATIC PANEL
TOUCH
Process Review
5/5/2011 11:09:27 AM
StopRUN
Grippers View
4th Cover
3rd Cover
2nd Cover
1st Cover
G2
G1
G2* G1
*Sensor for signalization is not working. E2.2
PROCESS
POS


SIEMENS
SIMATIC PANEL
TOUCH
Process Review
5/5/2011 11:09:56 AM
StopRUN
Positioning System View
Blocking Stick
Lift Table
PROCESS
GRIP