MASTER's THESIS – Renewable Energy Management

TH Köln (University of Applied Sciences)
ITT- Institute for Technology and Resources Management in the Tropics and Subtropics
and
SMS-Group, EASA

Optimization and development of Mould Level Control with MATLAB/Simulink and Comparison of energy consumption between the electric and hydraulic drive unit

Biswanath Pramanik

2018

ITT
Institute for Technology and Resources Management in the Tropics and Subtropics

Technology
Arts Sciences
TH Köln

Renewable Energy Management

TH Köln (University of Applied Sciences)
ITT - Institute for Technology and Resources Management
in the Tropics and Subtropics

SMS-Group, EASA

Optimization and development of Mould Level Control with MATLAB/Simulink and Comparison of energy consumption between the electric and hydraulic drive unit

Thesis to Obtain the Degree of

MASTER OF SCIENCE
Renewable Energy Management
DEGREE AWARDED BY COLOGNE UNIVERSITY OF APPLIED SCIENCES

PRESENTS:
STUDENT'S NAME
Biswanath Pramanik
SUPERVISOR OF THESIS AIT
Prof. Dr. Rainer Scheuring
SUPERVISOR OF THESIS ITT
Prof. Dr. Johannes Hamhaber

SUPERVISOR IN SMS-Group, EASA:
Mr. Dmitry Konstantinov and Mr. Kristian Stieglitz

DATE OF SUBMISSION
(15.01.2019)

presented by

Biswanath Pramanik Student no.:11118001 Email: biswanath.pramanik@smail.th-koeln.de

Declaration in lieu of oath

By

"Full Name of Student"

This is to confirm my Master's Thesis was independently composed/authored by myself, using solely the referred sources and support.

I additionally assert that this Thesis has not been part of another examination process.

Place and Date *Signature*

1 Acknowledgement

I would like to thank Prof. Dr. Rainer Scheuring and Prof. Dr. Johannes Hamhaber for their guidance and personal engagement who made this thesis possible, in the company SMS-Group. I also thank all employees of SMS-Group for their time and constant support.

2 Abstract

This project is focused on the generation of hardware independent code for PLCs and the comparison for energy consumption patterns of hydraulic and electric drive unit. This works is dedicated to MLC (mould level control) in a continuous casting machine, which is used to cast steel slabs continuously.

The code generation is done with the help of the PLC coder which is present in the software Simulink. The programming is done entirely in MATLAB. The application of the generated code is tested on the Siemens S7-1500 PLC. For executing the code and the development of the HMI (human machine interface) Siemens software TIA Portal V15 has been used. Moreover, for further analysis of signals and testing the code, a PDA or process data acquisition system, IBA system is used.

For energy analysis also the IBA system is used.

3 Contents

1 Acknowledgement 4
2 Abstract 5
3 Contents 6
4 Introduction 8
5 Fundamentals 8
5.1 Continuous Casting Machine 8
5.2 Mould Level Control 10
5.2.1 Structure of MLC 10
5.2.2 Control Structure of MLC 11
5.2.3 Function Description 13
6 S7-1500 PLC 15
7 Simulink 20
7.1 Solvers 20
7.2 Fixed step and Variable step solver 21
7.3 Continuous and Discrete solvers 21
7.4 Algebraic Loops 22
7.5 Simulating discrete systems 22
7.6 Simulink PLC Coder 23
7.7 Steps for code generation 24
7.8 Matrix Data Types 25
8 Programming in Simulink for PLC 25
8.1 Algebraic Loops 31
8.2 Creating a Library 31
8.3 Programming 31
9 TIA Portal 34
9.1 Hardware configuration 34
9.2 Programming in TIA Portal 37
9.2.1 Organization blocks (OBs) 37
9.2.2 Rules for Cyclic Interrupts 38
9.2.3 Functions (FC) 39
9.2.4 Function blocks (FB) 39
9.2.5 Instances 40

9.2.6 Instance Data Blocks 40
9.2.7 Multi-instances 40
9.2.8 Global data blocks (DB) 40
9.3 Transferring the Simulink generated code to the PLC 41
9.4 Sample time generator 44
9.5 6.3.2 Memory allocation 45
9.6 Organisation of the blocks in TIA Portal 48
10 HMI with TIA Portal 50
11 PDA System and Testing 54
11.1 Traces 54
11.2 IBA System 55
12 Test results 57
13 Energy analysis 62
13.1 Electric drive unit 62
13.2 Hydraulic drive unit 65
13.3 Noise generation 67
13.3.1 Harmonics 68
13.3.2 Effects of harmonics 69
13.3.3 Control of harmonics 70
13.3.4 Methods to control harmonics 71
13.3.5 Noise in the Hydraulic MLC 72
13.3.6 Noise in the Electric MLC 72
13.4 Comparison of the electric drive MLC and the Hydraulic drive MLC 74
14 Conclusion 75
15 List of codes 76
16 List of Figures 77
17 References 79

4 Introduction

The continuously developing next-generation technology helps to increase the market share and to keep a company ahead of competition. Few aspects which are very important in the development process, which are higher quality, higher productivity and reduced costs.
For over a decade the Power PMAC motion controller was used to provide high performance, flexible and efficient machine controller for the mould level control. But currently there is a need to upgrade the system to a better and faster motion controller. This migration of hardware brings along the tedious work of developing the entire program in the new programming environment and the new programming language, relevant to the new hardware. This process consumes a lot of time. The motive is to use Simulink and avail the feature of the PLC coder which can generate codes for various PLCs for a program which was originally coded in MATLAB. But the task is not simple. Every system brings along its own complications and challenges in programming. The main challenge here is to develop a program without compromising any functionality of the previous program, while taking care that it adapts to the new hardware entirely.

5 Fundamentals

5.1 Continuous Casting Machine

In steel plants the liquid steel must be casted in the required form, size and weight. In old casting technology, liquid steel was casted in the shape of blocks, called block casting. Block casting is still in use only for specific applications, but the modern technology is continuous casting (Stahlinstitut VDEh, 2007). The figure below shows a Continuous Casting Machine (CCM).

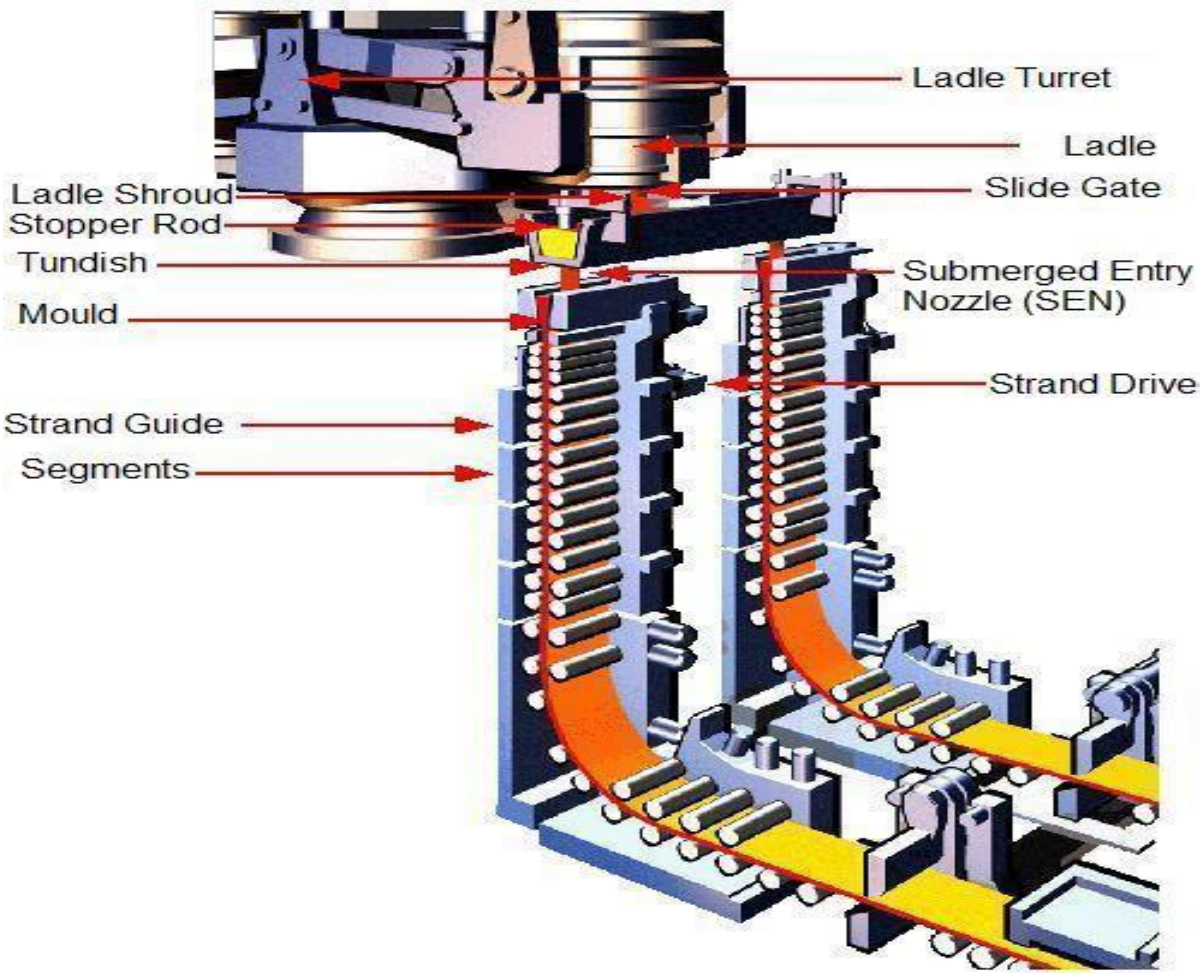


Figure 5.1.1 : CCM overview

During the casting the fluid steel from ladle runs through the ladle shroud into the tundish. The slide gate controls the flow. From the tundish the steel flows into the mould through the Submerged Entry Nozzle (SEN). The flow of steel into the mould is controlled by the stopper rod. The stopper rod is normally controlled by a hydraulic cylinder or an electric motor. It is very important to have a very precise and fast position control of the stopper rod to keep the mould level at the desired value.

When the mould is cooled, the solid shell of the strand is formed in the mould. The core of the strand is still liquid. While the strand moves through the segments, it is cooled by the sprayers with water and finally the core crystallizes as well. The strand is guided by the rolls. These strand guides are controlled by the strand drives. When the strand solidifies completely, the solid strand is cut by the shears or by the flying cutting machine (gas-jet cutting) to the rectangle pieces - slabs. The slabs are the semi-finished products, which are normally used for the hot rolling mills.

CCM is a complex machine, but it has the following advantages (Stahlinstitut VDEh, 2007):

• Infinite strand

• Smooth and fast crystallization

• Possibility for automation

One of the most important parts of the process is automation. The typical technological problems in CCM, which influence the automation process, are (Stahlinstitut VDEh, 2007):

• Clogging of the SEN

• Sticking of the material to the mould

• Shell break and the strand break

• Bulging of the strand between the rolls

• Oscillation marks on the strand surface

• Inclusions of other materials

Some of the automation processes of CCM are listed below:

• Control of the steel level in the tundish

• Mould Level Control (MLC)

• Remote Adjustable Control (RAM)

• Hydraulic Mould Oscillation (HMO)

• Hydraulic Segment Adjustment (HSA)

5.2 Mould Level Control

The quality of the slabs as well as the casting process is directly dependent on the functional accuracy of the MLC. By controlling the steel level in the mould, the following problems can be avoided:

• Strand shell break and the strand break

• Mixing the steel with the casting powder. The powder covers the surface of the steel in the mould to protect it from combining with the oxygen present in the ambient air.

• Overflow of the steel from the mould

There are many factors, which can influence the MLC. In other words, the following values can be either the parameters for control or the disturbances:

• Change in the temperature of the steel, the mould or the strand

• The chemical composition of the steel

• Weight of the steel in the tundish

• Casting speed

• The mechanical parts of the whole CCM

• Other automation processes (HMO, control of the steel level in the tundish)

• Handling of the casting powder

5.2.1 Structure of MLC

The structure of the MLC is shown in the figure below. The reference to the mould level is set in the work station by from the Human Machine Interface (HMI). This reference value is sent to the TCS,

which is a PLC S7-417. This PLC communicates with the S7-1500 PLC over Profibus. The motion controller or the S7-1500 PLC gets the reference value via Profibus. The actual value of the mould level is measured with the Berthold radioactive sensor and with the VUHZ eddy current sensor. The Berthold sensor has a bigger measurement range, but it has a static measurement error and the noise in the signal is bigger than in the VUHZ signal. The Berthold sensor is preferred for most of the plants.

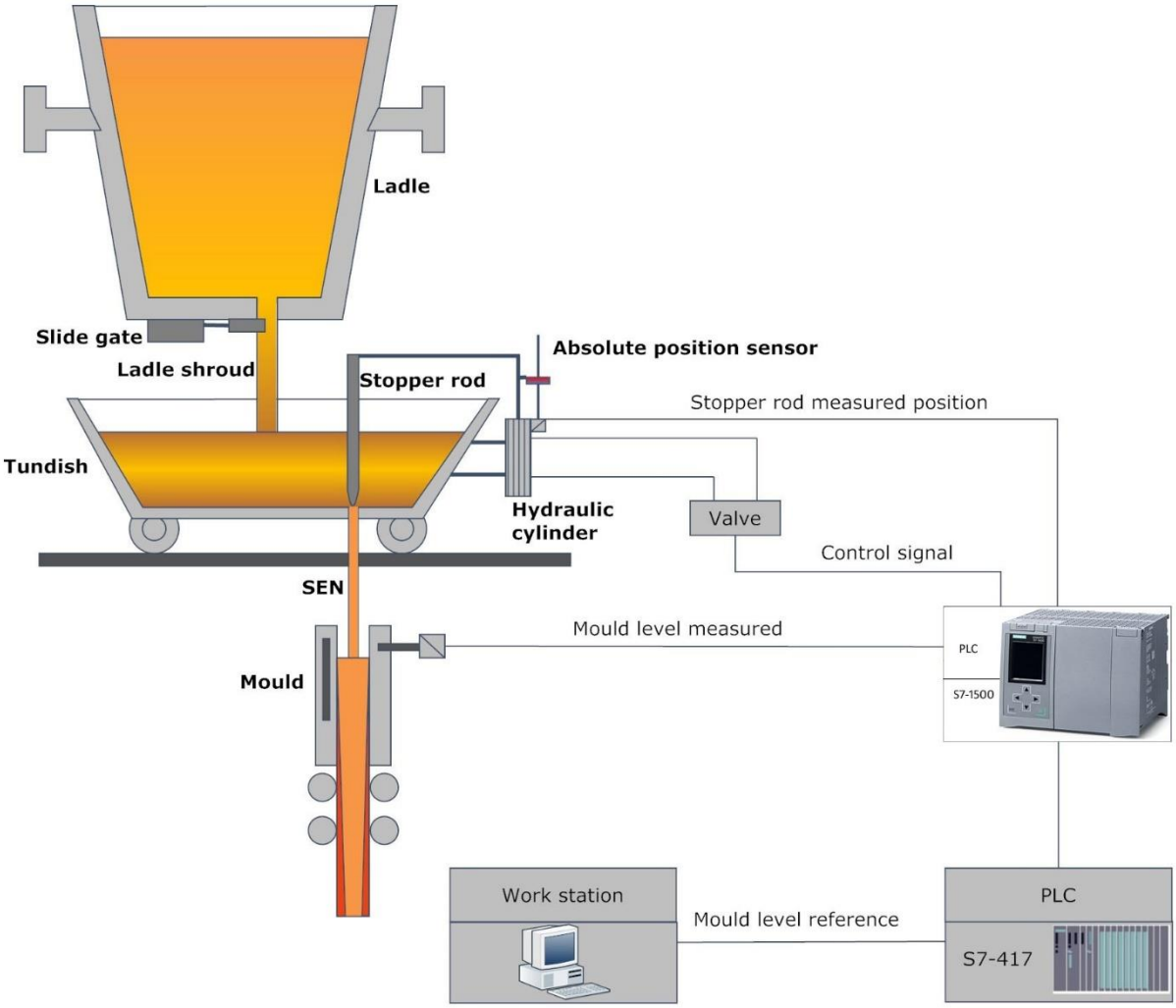


Figure 5.2.1: Structure of MLC

The S7-1500 PLC calculates the control error for the mould level controller and executes the control algorithm. The result of the calculations is the reference position for the stopper rod. The feedback position is given by a Synchronous Serial Interface (SSI) absolute position sensor, which is mounted on the hydraulic cylinder. The second control algorithm is executed for the stopper rod position. The result is sent to the proportional valve (in case of the hydraulic unit), which can control the stopper rod movements by means of the hydraulic cylinder.

5.2.2 Control Structure of MLC

The figure below illustrates demonstrates the control structure for the MLC application. Every plant is unique, and they have their own settings. That is why the medium numerical values are given.

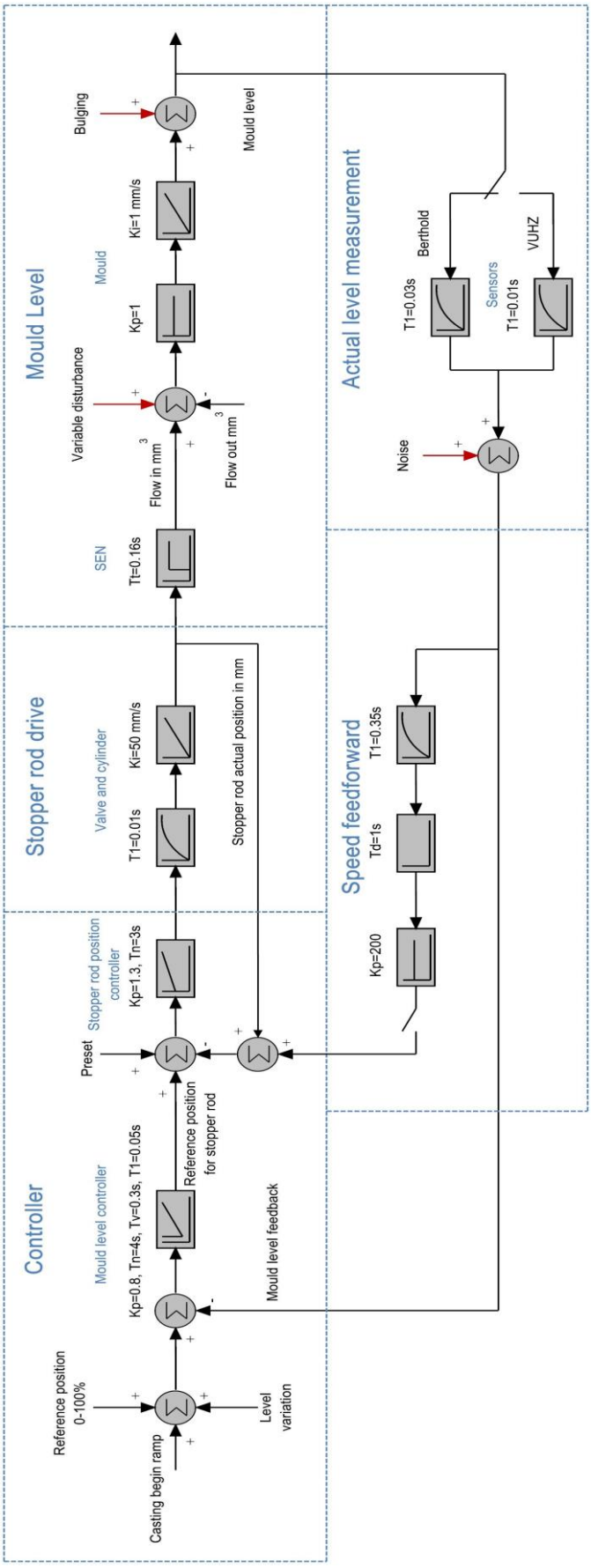


Figure 5.2.2: Example of the control structure for the MLC application

The structure can be described as the cascade controller with the internal stopper rod loop and an external mould level loop. The Controller and the Speed feedforward blocks are implemented in the motion controller. The *Controller* part includes the PID-T1 (Proportional-Integral-Derivative with the first order filter) controller for the mould level and the PI (Proportional-Integral) controller for the stopper rod.

The *Speed feedforward* block is a feedforward part of the mould level controller, which is the compensation of the bulging influence on the mould level.

The *Stopper rod drive* is the representation of the proportional valve with the cylinder. The valve is an aperiodic block, which controls the oil flow. The hydraulic cylinder is an integrator of the oil flow.

The *Mould level* block is a simple integrator of the steel flow in with a small delay. There are also influences of the flow out, disturbances and the bulging.

The *Actual level measurement* represents the sensors, which can be used for the mould level measurements.

5.2.3 Function Description

The MLC application includes following functions (Vadim Treivous, 2007):

• **Mould level control**. The initialization of the controller gains, the closing of the control loop and the movement to a new reference mould level are executed here.

• **Stopper rod control**. The initialization of the controller gains, the closing of the control loop, the movement to a new reference stopper rod position and the stopper rod position calibration are executed here.

• **Preset function**. The predefined position of the stopper rod is calculated in this function in a case of the casting speed change. The calculations are based on the theoretical research and the input data for the calculation are the geometry of the mould, the casting speed, the stopper rod characteristic and the liquid core reduction data. The stopper rod characteristic shows the dependency of the open area in the tundish on the rod stroke.

• **Adaptive control for the different steel grades**. The controller gains for the mould level controller change in dependency on the steel grade.

• **Controlled D-part of the mould level controller**. The derivative part of the mould level controller can be turned on/off.

• **Controller gain adaptation**. The function changes the proportional gain of the mould level controller which is dependent on the width of the strand.

• **Clogging detection**. The function detects if the SEN is blocked. If the stopper rod is moving up and the casting speed is constant, then the clogging is detected.

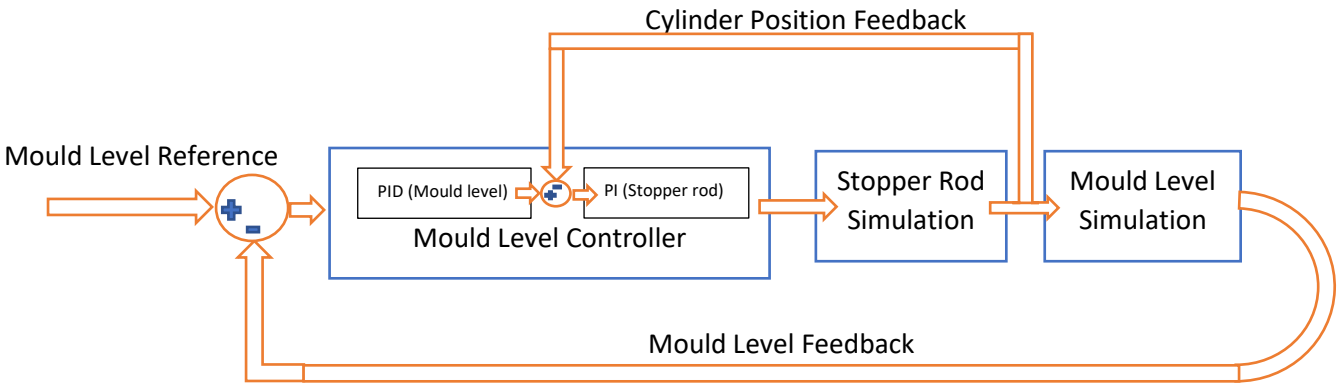
• **Statistic function**. The function calculates the statistical data for the control error of the mould level. It helps to evaluate the quality of the control.

• **Oscillation detection**. The detection of the mould level oscillation is realised in this function. If there is an oscillation, then the proportional gain of the mould level control will be decreased for the certain time in order to get rid of the oscillation.

• **Simulation**. It is a special program, which simulates the processes of the real system. So, the functionality of the whole software can be tested and the controller can be tuned with the help of the simulation.

• **Speed feedforward**. Compensates the mould level oscillations.

Brief representation of the structure is in the diagram below.



6 S7-1500 PLC

S7-1500 PLC (programmable logic controller) is the latest generation PLC from Siemens. It is a successor of the previous ranges of PLCs which are S7-300 and S7-400 PLCs, which are still predicted to prevail in the industries for the coming years. S7-1500 is currently one of the fastest PLCs available in the market. It has the latest technology features such as Profinet port as standard, Integrated Security, Integrated Technology including Motion Control and Profidrive, Integrated System Diagnostics.

The TIA Portal (Totally Integrated Automation Portal) is the latest automation software from Siemens. It is a software with one engineering environment and one software project for all automation tasks. The previous generation of softwares (Step7, Wincc Flexible, Wincc explorer etc.) which had to be used separately for different tasks has been integrated into one software. It is designed for maximum engineering efficiency and user-friendliness.

When the S7-1500 is used in integration with the TIA Portal all the new features can be availed such as security, shared data management, a uniform operating concept and centralized services. The SIMATIC S7-1500 automation system supports all the available conventional communication standards. Technology CPUs are available for functions such as extended Motion Control. The SIMATIC S7-1500 CPUs are also available as fail-safe controllers. Diagnostic functions across all components make troubleshooting simple. Changes to the parameters can be done fast and with great ease with the display which is integrated. Security functions which are also integrated, help against manipulation and know-how theft and provide added security mechanisms for the configuration of secure networks.

All the SIMATIC S7-1500 CPUs offer integrated Motion Control functions, which interests us for this project.

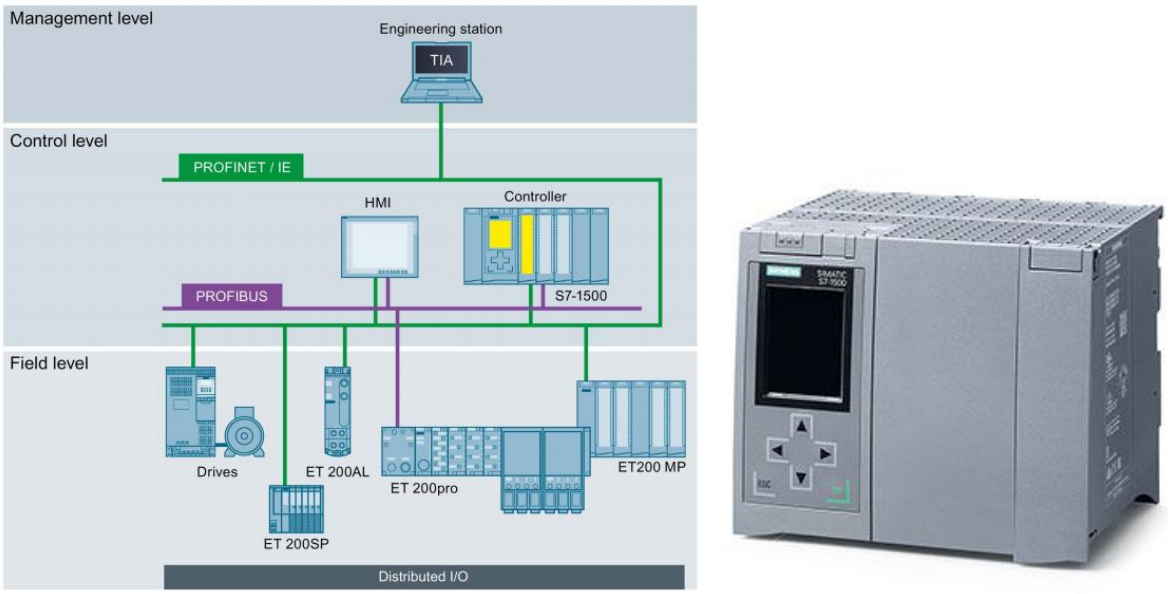


Figure 5.2.1: SIMATIC S7-1500 at management, control and field level

SIMATIC S7-1500 controllers has the capability to be scaled in terms of processing speed and configuration limits. They also provide networking facilities via different communications standards. Safety Integrated, Motion Control, and other technology functions can be applied to all plant sizes. It also comes with a display which was absent in previous generation of PLCs which allows control of various features such as change the IP address of the CPU directly and quick access to diagnostics alarms.

Other features include Communication via Ethernet/PROFINET, Communication via PROFIBUS, HMI communication, Communication via OPC UA, Web server, technology functions, system diagnostics, and protection functions integrated (Simatic S7 1500/ET 200 MP automation system in a nutshell, 2016).

For this project I am using 1516T-3 PN/DP CPU in integration with TIA Portal.

It weighs 1978 grams and has a dimension of width 175 mm, height 147 mm and depth 129 mm. It has adjustable cycle time and all access protection.

The main features are listed below:

Article number	6ES7516-3TN00-0AB0
General information	
Product type designation	CPU 1516T-3 PN/DP
HW functional status	FS05
Firmware version	V2.5
Engineering with	
• STEP 7 TIA Portal configurable/integrated as of version	V15 (FW V2.5)
Configuration control	
via dataset	Yes
Display	
Screen diagonal [cm]	6.1 cm
Control elements	
Number of keys	6
Mode selector switch	1
Supply voltage	
Type of supply voltage	24 V DC
permissible range, lower limit (DC)	19.2 V
permissible range, upper limit (DC)	28.8 V
Reverse polarity protection	Yes
Mains buffering	
• Mains/voltage failure stored energy time	5 ms
• Repeat rate, min.	1/s
Input current	
Current consumption (rated value)	1.2 A
Current consumption, max.	1.55 A
Inrush current, max.	2.4 A; Rated value
I^2t	0.02 A^2·s
Power	
Infeed power to the backplane bus	12 W
Power consumption from the backplane bus (balanced)	30 W
Power loss	
Power loss, typ.	24 W
Memory	
Number of slots for SIMATIC memory card	1
SIMATIC memory card required	Yes

Article number	6ES7516-3TN00-0AB0
Work memory	
• integrated (for program)	1.5 Mbyte
• integrated (for data)	5 Mbyte
Load memory	
• Plug-in (SIMATIC Memory Card), max.	32 Gbyte
Backup	
• maintenance-free	Yes
CPU processing times	
for bit operations, typ.	10 ns
for word operations, typ.	12 ns
for fixed point arithmetic, typ.	16 ns
for floating point arithmetic, typ.	64 ns
CPU-blocks	
Number of elements (total)	6 000; Blocks (OB, FB, FC, DB) and UDTs
DB	
• Number range	1 ... 60 999; subdivided into: number range that can be used by the user: 1 ... 59 999, and number range of DBs created via SFC 86: 60 000 ... 60 999
• Size, max.	5 Mbyte; For non-optimized block accesses, the max. size of the DB is 64 KB
FB	
• Number range	0 ... 65 535
• Size, max.	1 Mbyte
FC	
• Number range	0 ... 65 535
• Size, max.	1 Mbyte
OB	
• Size, max.	1 Mbyte
• Number of free cycle OBs	100
• Number of time alarm OBs	20
• Number of delay alarm OBs	20
• Number of cyclic interrupt OBs	20; With minimum OB 3x cycle of 250 µs
• Number of process alarm OBs	50
• Number of DPV1 alarm OBs	3
• Number of isochronous mode OBs	2
• Number of technology synchronous alarm OBs	2
• Number of startup OBs	100

Article number	6ES7516-3TN00-0AB0
• Number of asynchronous error OBs	4
• Number of synchronous error OBs	2
• Number of diagnostic alarm OBs	1
Nesting depth	
• per priority class	24
Counters, timers and their retentivity	
S7 counter	
• Number	2 048
Retentivity	
– adjustable	Yes
IEC counter	
• Number	Any (only limited by the main memory)
Retentivity	
– adjustable	Yes
S7 times	
• Number	2 048
Retentivity	
– adjustable	Yes
IEC timer	
• Number	Any (only limited by the main memory)
Retentivity	
– adjustable	Yes
Data areas and their retentivity	
Retentive data area (incl. timers, counters, flags), max.	512 kbyte; In total; available retentive memory for bit memories, timers, counters, DBs, and technology data (axes): 472 KB
Extended retentive data area (incl. timers, counters, flags), max.	5 Mbyte; When using PS 60W 24/48/60V DC HF
Flag	
• Number, max.	16 kbyte
• Number of clock memories	8; 8 clock memory bits, grouped into one clock memory byte
Data blocks	
• Retentivity adjustable	Yes
• Retentivity preset	No
Local data	
• per priority class, max.	64 kbyte; max. 16 KB per block

Article number	6ES7516-3TN00-0AB0
Address area	
Number of IO modules	8 192; max. number of modules / submodules
I/O address area	
• Inputs	32 kbyte; All inputs are in the process image
• Outputs	32 kbyte; All outputs are in the process image
per integrated IO subsystem	
– Inputs (volume)	8 kbyte
– Outputs (volume)	8 kbyte
per CM/CP	
– Inputs (volume)	8 kbyte
– Outputs (volume)	8 kbyte
Subprocess images	
• Number of subprocess images, max.	32
Hardware configuration	
Number of distributed IO systems	64; A distributed I/O system is characterized not only by the integration of distributed I/O via PROFINET or PROFIBUS communication modules, but also by the connection of I/O via AS-i master modules or links (e.g. IE/PB-Link)
Number of DP masters	
• integrated	1
• Via CM	8; A maximum of 8 CMs/CPs (PROFIBUS, PROFINET, Ethernet) can be inserted in total
Number of IO Controllers	
• integrated	2
• Via CM	8; A maximum of 8 CMs/CPs (PROFIBUS, PROFINET, Ethernet) can be inserted in total
Rack	
• Modules per rack, max.	32; CPU + 31 modules
• Number of lines, max.	1
PtP CM	
• Number of PtP CMs	the number of connectable PtP CMs is only limited by the number of available slots
Time of day	
Clock	
• Type	Hardware clock
• Backup time	6 wk; At 40 °C ambient temperature, typically
• Deviation per day, max.	10 s; Typ.: 2 s
Operating hours counter	
• Number	16

Article number	6ES7516-3TN00-0AB0
Clock synchronization	
• supported	Yes
• to DP, master	Yes
• in AS, master	Yes
• in AS, slave	Yes
• on Ethernet via NTP	Yes
Interfaces	
Number of PROFINET interfaces	2
Number of PROFIBUS interfaces	1
1. Interface	
Interface types	
• Number of ports	2
• integrated switch	Yes
• RJ 45 (Ethernet)	Yes; X1
Functionality	
• IP protocol	Yes; IPv4
• PROFINET IO Controller	Yes
• PROFINET IO Device	Yes
• SIMATIC communication	Yes
• Open IE communication	Yes
• Web server	Yes
• Media redundancy	Yes; MRP Automanager according to IEC 62439-2 Edition 2.0
PROFINET IO Controller	
Services	
– PG/OP communication	Yes
– S7 routing	Yes
– Isochronous mode	Yes
– Open IE communication	Yes
– IRT	Yes
– MRP	Yes; As MRP redundancy manager and/or MRP client; max. number of devices in the ring: 50
– MRPD	Yes; Requirement: IRT
– PROFIenergy	Yes
– Prioritized startup	Yes; Max. 32 PROFINET devices
– Number of connectable IO Devices, max.	256; In total, up to 1 000 distributed I/O devices can be connected via AS-i, PROFIBUS or PROFINET

Article number	6ES7516-3TN00-0AB0
2. Interface	
Interface types	
• Number of ports	1
• integrated switch	No
• RJ 45 (Ethernet)	Yes; X2
Functionality	
• IP protocol	Yes; IPv4
• PROFINET IO Controller	Yes
• PROFINET IO Device	Yes
• SIMATIC communication	Yes
• Open IE communication	Yes
• Web server	Yes
• Media redundancy	No
PROFINET IO Controller	
Services	
– PG/OP communication	Yes
– S7 routing	Yes
– Isochronous mode	No
– Open IE communication	Yes
– IRT	No
– MRP	No
– PROFIenergy	Yes
– Prioritized startup	No
– Number of connectable IO Devices, max.	32; In total, up to 1 000 distributed I/O devices can be connected via AS-i, PROFIBUS or PROFINET
– Number of connectable IO Devices for RT, max.	32
– of which in line, max.	32
– Number of IO Devices that can be simultaneously activated/deactivated, max.	8; in total across all interfaces
– Number of IO Devices per tool, max.	8
– Updating times	The minimum value of the update time also depends on communication share set for PROFINET IO, on the number of IO devices, and on the quantity of configured user data
Update time for RT	
– for send cycle of 1 ms	1 ms to 512 ms

Article number	6ES7516-3TN00-0AB0
– Of which IO devices with IRT, max.	64
– Number of connectable IO Devices for RT, max.	256
– of which in line, max.	256
– Number of IO Devices that can be simultaneously activated/deactivated, max.	8; in total across all interfaces
– Number of IO Devices per tool, max.	8
– Updating times	The minimum value of the update time also depends on communication share set for PROFINET IO, on the number of IO devices, and on the quantity of configured user data
Update time for IRT	
– for send cycle of 250 µs	250 µs to 4 ms; Note: In the case of IRT with isochronous mode, the minimum update time of 500 µs of the isochronous OB is decisive
– for send cycle of 500 µs	500 µs to 8 ms
– for send cycle of 1 ms	1 ms to 16 ms
– for send cycle of 2 ms	2 ms to 32 ms
– for send cycle of 4 ms	4 ms to 64 ms
– With IRT and parameterization of "odd" send cycles	Update time = set "odd" send clock (any multiple of 125 µs: 375 µs, 625 µs ... 3 875 µs)
Update time for RT	
– for send cycle of 250 µs	250 µs to 128 ms
– for send cycle of 500 µs	500 µs to 256 ms
– for send cycle of 1 ms	1 ms to 512 ms
– for send cycle of 2 ms	2 ms to 512 ms
– for send cycle of 4 ms	4 ms to 512 ms
PROFINET IO Device	
Services	
– PG/OP communication	Yes
– S7 routing	Yes
– Isochronous mode	No
– Open IE communication	Yes
– IRT	Yes
– MRP	Yes
– MRPD	Yes; Requirement: IRT
– PROFIenergy	Yes
– Shared device	Yes
– Number of IO Controllers with shared device, max.	4
– Asset management record	Yes; Per user program

Article number	6ES7516-3TN00-0AB0
PROFINET IO Device	
Services	
– PG/OP communication	Yes
– S7 routing	Yes
– Isochronous mode	No
– Open IE communication	Yes
– IRT	No
– MRP	No
– MRPD	No
– PROFIenergy	Yes
– Prioritized startup	No
– Shared device	Yes
– Number of IO Controllers with shared device, max.	4
– Asset management record	Yes; Per user program
3. Interface	
Interface types	
• Number of ports	1
• RS 485	Yes; X3
Functionality	
• PROFIBUS DP master	Yes
• PROFIBUS DP slave	No
• SIMATIC communication	Yes
Interface types	
RJ 45 (Ethernet)	
• 100 Mbps	Yes
• Autonegotiation	Yes
• Autocrossing	Yes
• Industrial Ethernet status LED	Yes
RS 485	
• Transmission rate, max.	12 Mbit/s
Protocols	
Number of connections	
• Number of connections, max.	256; via integrated interfaces of the CPU and connected CPs / CMs
• Number of connections reserved for ES/HMI/web	10

Article number	6ES7516-3TN00-0AB0
• Number of connections via integrated interfaces	128
• Number of S7 routing paths	16
SIMATIC communication	
• S7 communication, as server	Yes
• S7 communication, as client	Yes
• User data per job, max.	See online help (S7 communication, user data size)
Open IE communication	
• TCP/IP	Yes
– Data length, max.	64 kbyte
– several passive connections per port, supported	Yes
• ISO-on-TCP (RFC1006)	Yes
– Data length, max.	64 kbyte
• UDP	Yes
– Data length, max.	2 kbyte; 1 472 bytes for UDP broadcast
– UDP multicast	Yes; Max. 5 multicast circuits
• DHCP	No
• SNMP	Yes
• DCP	Yes
• LLDP	Yes
Web server	
• HTTP	Yes; Standard and user pages
• HTTPS	Yes; Standard and user pages
PROFIBUS DP master	
• Number of connections, max.	48; for the integrated PROFIBUS DP interface
Services	
– PG/OP communication	Yes
– S7 routing	Yes
– Data record routing	Yes
– Isochronous mode	Yes
– Equidistance	Yes
– Number of DP slaves	125; In total, up to 1 000 distributed I/O devices can be connected via AS-i, PROFIBUS or PROFINET
– Activation/deactivation of DP slaves	Yes

Article number	6ES7516-3TN00-0AB0
OPC UA	
• Runtime license required	Yes
• OPC UA Server	Yes; Data access (read, write, subscribe), method call, custom address space
– Application authentication	Yes
– Security policies	Available security policies: None, Basic128Rsa15, Basic256Rsa15, Basic256Sha256
– User authentication	"anonymous" or by user name & password
Further protocols	
• MODBUS	Yes; MODBUS TCP
Media redundancy	
• Switchover time on line break, typ.	200 ms; For MRP, bumpless for MRPD
• Number of stations in the ring, max.	50
Isochronous mode	
Isochronous operation (application synchronized up to terminal)	Yes; With minimum OB 6x cycle of 375 µs
Equidistance	Yes
S7 message functions	
Number of login stations for message functions, max.	32
Program alarms	Yes
Number of configurable program alarms	10 000
Number of simultaneously active program alarms	
• Number of program alarms	600
• Number of alarms for system diagnostics	200
• Number of alarms for motion technology objects	160
Test commissioning functions	
Joint commission (Team Engineering)	Yes; Parallel online access possible for up to 8 engineering systems
Status block	Yes; Up to 8 simultaneously (in total across all ES clients)
Single step	No
Number of breakpoints	8
Status/control	
• Status/control variable	Yes
• Variables	Inputs/outputs, memory bits, DBs, distributed I/Os, timers, counters
• Number of variables, max.	
– of which status variables, max.	200; per job
– of which control variables, max.	200; per job

Article number	6ES7516-3TN00-0AB0
Forcing	
• Forcing, variables	Peripheral inputs/outputs
• Number of variables, max.	200
Diagnostic buffer	
• present	Yes
• Number of entries, max.	3 200
– of which powerfail-proof	500
Traces	
• Number of configurable Traces	4; Up to 512 KB of data per trace are possible
Interrupts/diagnostics/status information	
Diagnostics indication LED	
• RUN/STOP LED	Yes
• ERROR LED	Yes
• MAINT LED	Yes
• Connection display LINK TX/RX	Yes
Supported technology objects	
Motion Control	Yes; Note: The number of technology objects affects the cycle time of the PLC program; selection guide via the TIA Selection Tool or SIZER
• Number of available Motion Control resources for technology objects (except cam disks)	6 400
• Required Motion Control resources	
– per speed-controlled axis	40
– per positioning axis	80
– per synchronous axis	160
– per external encoder	80
– per output cam	20
– per cam track	160
– per probe	40
• Number of available Extended Motion Control resources for technology objects	192
• Required Extended Motion Control resources	
– for each cam	2
– for each set of kinematics	30

Article number	6ES7516-3TN00-0AB0
• Positioning axis	
– Number of positioning axes at motion control cycle of 4 ms (typical value)	55
– Number of positioning axes at motion control cycle of 8 ms (typical value)	80
Controller	
• PID_Compact	Yes; Universal PID controller with integrated optimization
• PID_3Step	Yes; PID controller with integrated optimization for valves
• PID-Temp	Yes; PID controller with integrated optimization for temperature
Counting and measuring	
• High-speed counter	Yes
Standards, approvals, certificates	
Suitable for safety functions	No
Ambient conditions	
Ambient temperature during operation	
• horizontal installation, min.	0 °C
• horizontal installation, max.	60 °C; Display: 50 °C, at an operating temperature of typically 50 °C, the display is switched off
• vertical installation, min.	0 °C
• vertical installation, max.	40 °C; Display: 40 °C, at an operating temperature of typically 40 °C, the display is switched off
Ambient temperature during storage/ transportation	
• min.	-40 °C
• max.	70 °C
Configuration	
Programming	
Programming language	
– LAD	Yes
– FBD	Yes
– STL	Yes
– SCL	Yes
– GRAPH	Yes
Know-how protection	
• User program protection/password protection	Yes
• Copy protection	Yes
• Block protection	Yes

(Simatic S7 1500 CPU 1516T-3 PN/DP, 2017)

7 Simulink

Simulink is a software from MathWorks. It is used for modelling, simulating, and analysing dynamic systems. It supports systems which are linear and nonlinear modelling, which can be done in continuous time, sampled time, or as a hybrid. Systems can also be multirate, which means it can have different parts that are sampled or updated at different rates. In other words, it is a graphical user interface for MATLAB but it has lots of additional features for analysing and visualising a system.

It can also be called interactive simulation. Which means parameters can be changes while the simulation is executing and analysed how it behaves. It has access to all the tools and functions of MATLAB. An existing MATLAB model can be easily incorporated in the Simulink model. Simulink offers a wide range of standard blocks with different functions, but in addition to that it allows the possibility to create custom blocks which can be used in a same way in combination with other standard blocks. MATLAB and Simulink are both integrated into each other, so both can run simultaneously, and results can be analysed or changed in both the environments at any point.

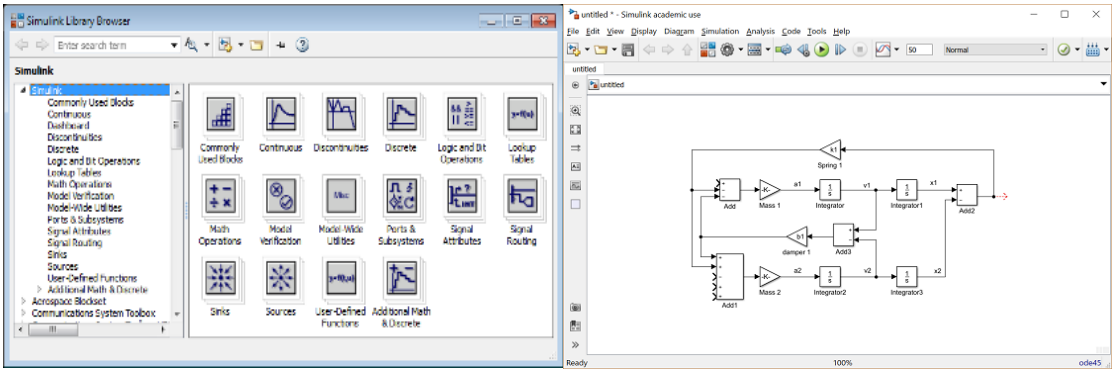


Figure 5.2.1: Simulink library View and example model view

Simulink has a block diagram semantics consisting of blocks and lines. These lines represent signals which are derived from engineering areas such as feedback control theory and signal processing.

7.1 Solvers

A dynamic system is simulated by computing its states at successive time steps over a specified time span, using information provided by the model. The process of computing the successive states of a system from its model is known as solving the model. Sometimes one way of solving a model is not the best way for all systems. For that reason, it has various programs known as solvers. Each solver has a particular approach to solve a model. The “Configuration Parameters” dialog box can be used to choose various solvers most suitable for a particular model.

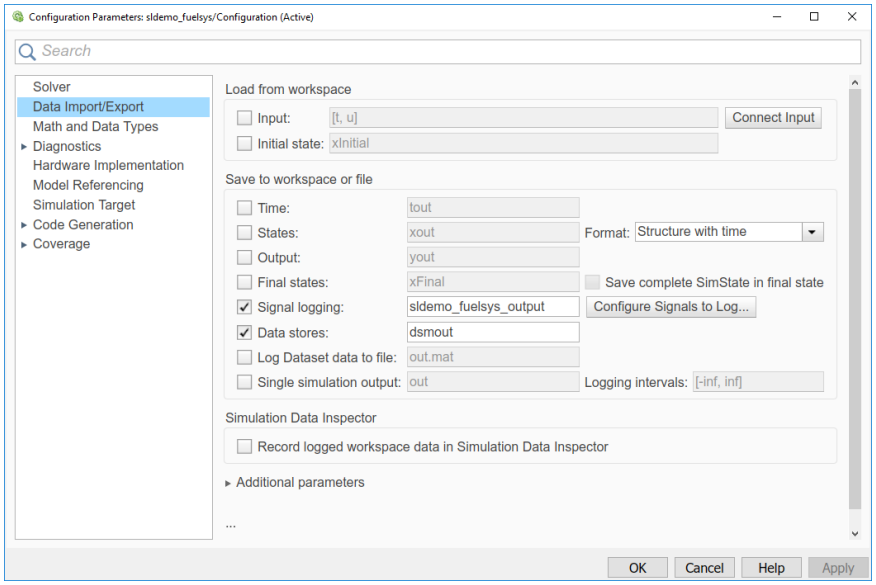


Figure 7.1.1: Configuration Parameters dialog box

7.2 Fixed step and Variable step solver

Fixed step solvers solve the model at fixed time intervals. This is called step size. On the other hand, a variable step solver optimises the step size. It reduces the step size when the states are changing very fast. And it increases the step size when the states are changing slowly. It allows it to avoid unnecessary steps. A variable step solver is more accurate and increases the computational efficiency. For this project I am using only fixed-step solver which is the exact representation of the computation in a PLC.

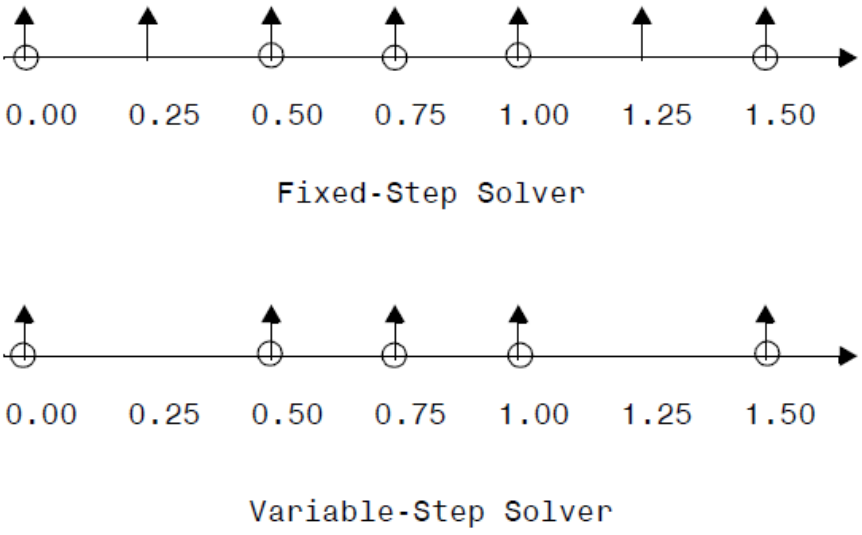


Figure 7.2.1: Example of fixed step and variable step

7.3 Continuous and Discrete solvers

Continuous solvers use numerical integration to compute a model's continuous states at the current time step from the states at previous time steps and the state derivatives. Continuous solvers rely on the model's blocks to compute the values of the model's discrete states at each time step. Mathematicians have developed a wide variety of numerical integration techniques for solving the

ordinary differential equations (ODEs) that represent the continuous states of dynamic systems. Simulink provides an extensive set of fixed-step and variable-step continuous solvers, each implementing a specific ODE solution method. Whereas Discrete time solvers are used to solve discrete models.

A continuous solver can be used to solve a model containing both continuous and discrete states but not a discrete solver.

7.4 Algebraic Loops

In a block the output is calculated from inputs. In certain cases, when some inputs are a result of outputs the result cannot be calculated. These are called algebraic loops.

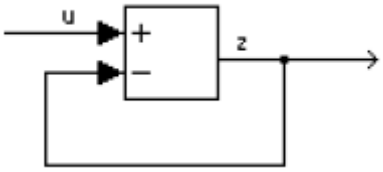


Figure 7.4.1: Example of an algebraic loop

Automatic algebraic loop elimination can be enabled by selecting the “minimise algebraic loop occurrences” parameter on the block parameters dialog box.

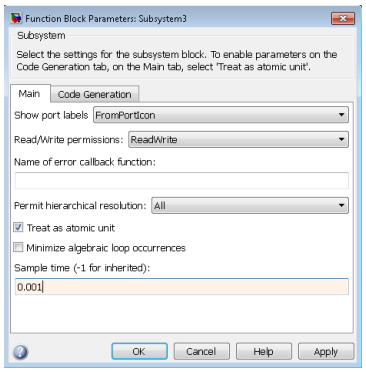


Figure 7.4.2: Minimise algebraic loop occurrences

But it has its limitation to eliminate algebraic loops. The best practice to eliminate algebraic loops is to add a delay block to the feedback.

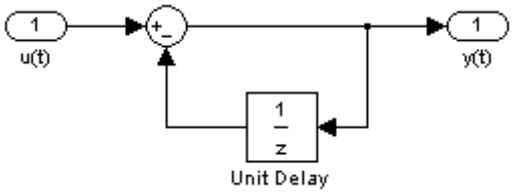


Figure 7.4.3: Example of algebraic loop elimination

7.5 Simulating discrete systems

Simulink can simulate discrete systems including components which can operate at different sample times and hybrid systems, which contains both continuous and discrete systems.

Sample time block parameters

Sample time can be specified for a block in block parameters.

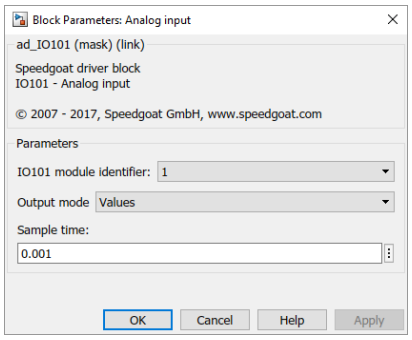


Figure 7.5.1: Sample time in block parameters

Sample time inheritance

Most blocks can inherit their sample time from the blocks connected to their inputs.

(Matlab&Simulink, 2018)

7.6 Simulink PLC Coder

Simulink PLC coder is designed with a motive to generate a code which is hardware independent. It allows generating IEC 61131-3 Structured Text code from the Simulink models. IEC is a standard that defines the structured text language for which the Simulink PLC Code software generates code. With this Model-Based Design approach can be utilized and transferred it to the programmable automation controller and programmable logic controller environment. This will be a great advantage to save time when it comes to the model to coding phase. Changes in model design and algorithms can be implemented in no time. It is most useful for control and algorithm design and test engineers in PLC manufacturing, Machine manufacturing, System integration. But to avail this tool knowledge of MATLAB, Simulink, PLCs and Structured Text Language is required.

The Simulink PLC coder is compatible with following IDE (integrated development environment).

- 3S-Smart Software Solutions CoDeSys
- B&R Automation Studio
- Beckhoff TwinCAT
- KW-Software MULTIPROG
- OMRON Sysmac Studio Version
- Phoenix Contact PC WORX
- Rexroth IndraWorks version 13V12 IDE
- Rockwell Automation RSLogix 5000 Series
- Siemens SIMATIC STEP 7
- Siemens TIA Portal
- Generic
- PLCopen XML

For this project I am using TIA Portal V15 which is a Siemens IDE. One must be careful while choosing the versions as all the versions might not be compatible.

7.7 Steps for code generation

1. It is important to make solver settings by using the Configuration Parameters dialog box. The Type must be Fixed step and the Solver must be discrete. Because PLCs process digital data. And care must be taken that the model is also discrete. Continuous states will not generate a code.
2. The part of the code for which the code generation is desired must be encapsulated in a subsystem. A subsystem generally holds several blocks it has inputs and outputs acting as a single block.

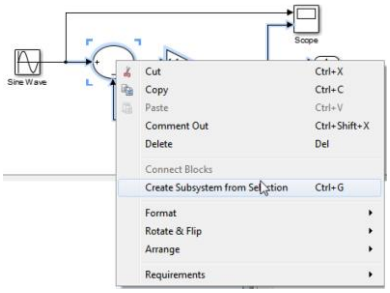


Figure 7.7.1: Creating a subsystem

3. The subsystem must be treated as an Atomic subsystem. An Atomic subsystem executes its elements separately with its own execution order. It can be done by going to block parameters.

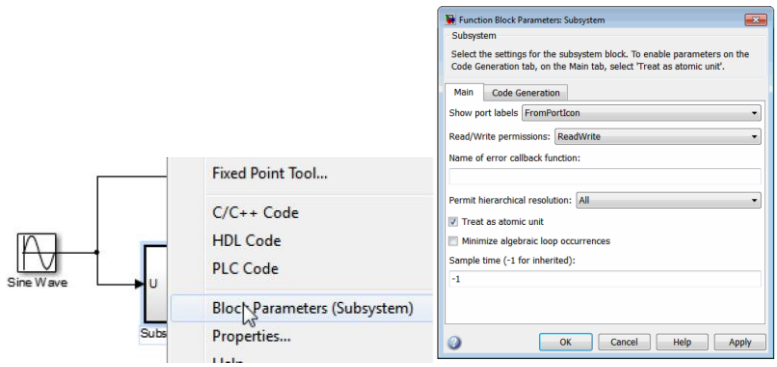


Figure 7.7.2: Creating an atomic subsystem

4. The system compatibility for structured text code generation must be checked. The coder verifies the criteria and displays the information.

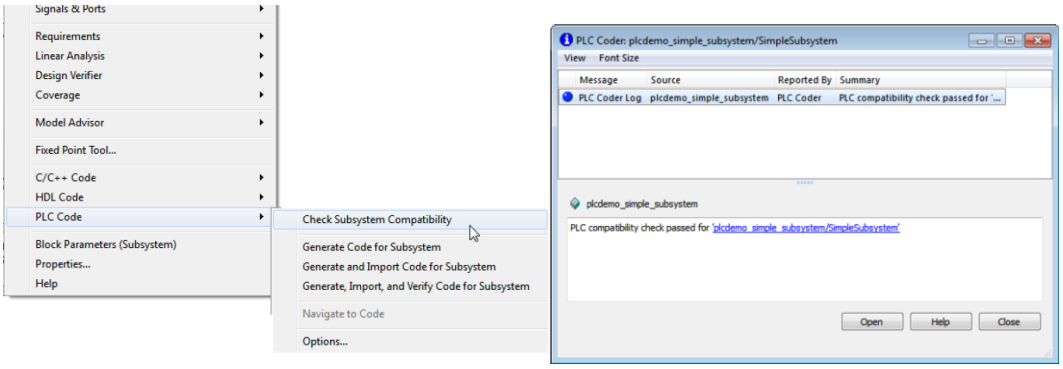


Figure 7.7.3: Checking compatibility for code generation

5. Then the target IDE must be chosen.

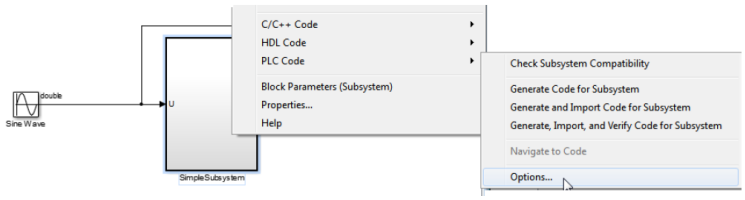


Figure 7.7.4: Choosing target IDE

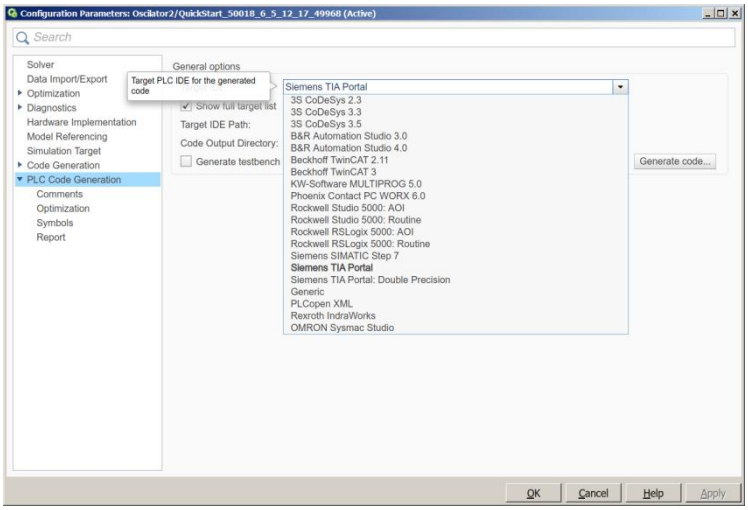


Figure 7.7.5: Choosing target IDE

6. After the IDE is chosen, code can be generated by choosing the **PLC Code** > **Generate Code for Subsystem**. After the code is generated, the relevant file can be found in the selected directory, which can be examined.

In the case of Siemens PLC, it generates an SCL code. SCL (Structured Control Language) is a high-level textual programming language which is based on PASCAL. It is also based on a standard for PLCs (programmable logic controllers).

7.8 Matrix Data Types

Care must be taken while programming with array data types. It converts matrix data types to single-dimensional vectors.

(Matlab&Simulink, 2014)

8 Programming in Simulink for PLC

Every PLC has a scan cycle. Scan cycle is the time in which the PLC scans all the inputs, executes the program and creates all the outputs related to the program. Scan cycle should be very small for fast processing. Usually it is in milliseconds.

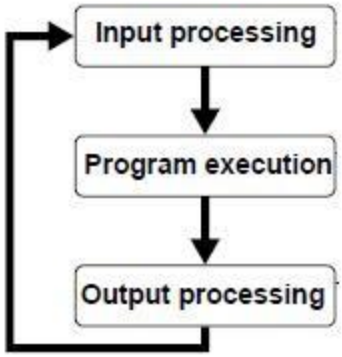


Figure 7.8.1: Scan cycle

PLCs have a provision of running different parts of the program with different scan cycle time. Running the most critical programs with a faster scan time has advantages of precision. For example, a PID control logic. But there are parts of the program which do not require a fast scan time. For example, a program which scans a manual switch input to turn on a light. Running this program under a fast scan cycle is unnecessary as the switching on and off operation will be very less frequent. Even if it is frequent, it is not possible that the manual operations can be done in milliseconds. It will only mean that the PLC will have to execute the program a greater number of times than necessary. It may require more RAM usage and cause the PLC to slow down. So, it is wise to use the Computational power of the PLC where required.

But this solution brings few problems in programming the PLC. Consider the example of a ramp.

A ramp decides the time in which a value in its initial state will reach its final. In other words, it has a slope from state A to B. In discrete programming it does it by incrementing the value of State A every scan cycle by a stated value or step size. In the figure below, it has a sample time of 1 second and a step size of 1 unit.

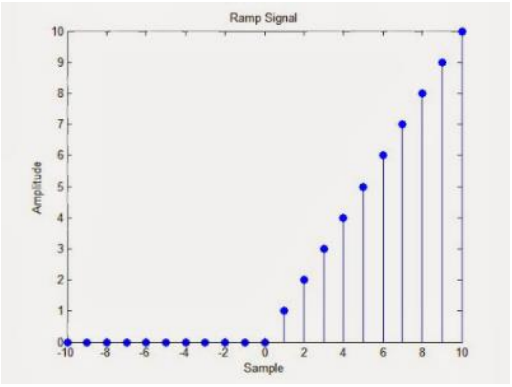


Figure 7.8.2: Discrete steps for ramp

But if the same ramp program is running with a different scan cycle the slope of the ramp will vary. To correct this, the step size must be changed.

It is not wise to reprogram or adjust the same program for different scan cycles. One solution it to engineer sample time or utilize the sample time value to calculate the step size. Step size = Original step size x sample time.

This is just one example. Same way a sine wave, a triangle wave or a saw tooth should have the same frequency and amplitude. Any time dependent program should behave in the exact same way in every scan cycle. There are several solutions by the usage of sample time.

These examples were discussed to explain one problem in code generation using Simulink. Simulink provides many functions blocks, which are Simulink standard blocks. These standard blocks do not calculate their values based on the sample time. It is possible to define at which scan cycle these blocks will run but they have their own calculation methods. Which means if a code is generated using these standard blocks, they will function differently in different scan cycle times in a PLC.

So, the solution is to create our own blocks with our own program. Simulink has a provision for that. They are called User defined blocks. There are several ways to create a custom block.

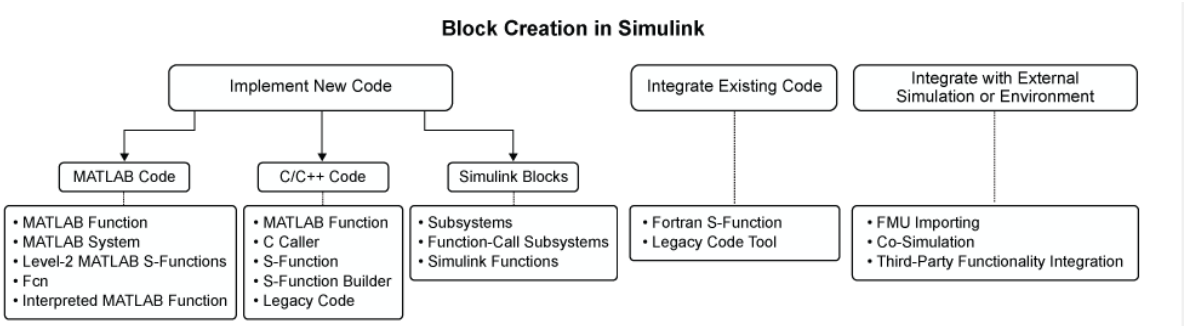
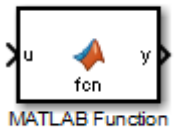


Figure 7.8.3: Block creation

(MathWorks, 2018)

For this project I am using MATLAB Function blocks.



While programming these blocks care must be taken to make sure each variable defined as outputs has some value assigned to it at every point during the execution of the program or else it will generate an error.

Every function used inside the MATLAB Function should also be available in SCL. For example, Rand (Random number function used to generate a random number within a range) or the Sin (sinus function) they are unavailable in SCL. If these functions are used Simulink will fail to generate a code. Most of the basic functions like the mathematical operators and logical operators are common in every programming language. Special functions like noise generator, ramp or sinus must be recreated.

Below are some of the codes for the functions I created. In each case Ts is the sample time.

Code 1: Ramp Function

```
function Output = Ramp(OldOut,Input,SpeedUp,SpeedDown,Ts,Start)

if Start
    if Input > OldOut
        Step = SpeedUp*Ts;
        if (Input-OldOut) > Step
            Output= OldOut + Step;
        else
            Output= Input;
```

```
        end
    else
     if Input < OldOut
        Step = SpeedDown*Ts;
        if (OldOut- Input) > Step
            Output = OldOut - Step;
        else
            Output = Input;
        end
     else
         Output = Input;
     end
    end
else
    Output = Input;
End
```

Code 2: Sine Wave Generator

```
function [ActPhase, Output] = Sine(OldPhase, Amp, Freq, Ts, StartPhase, Enable, Hold)
if Enable
    if Hold
        ActPhase = OldPhase;
        Output = Amp * sin(ActPhase);
    else
        ActPhase = (2* pi* Freq *Ts)+ OldPhase + StartPhase;
        if ActPhase > (2 * pi)
            ActPhase = mod(ActPhase,2*pi);
        end
        Output = Amp * sin(ActPhase);
    end
else
    Output = 0;
    ActPhase = 0;
end
```

Code 3: Saw tooth Generator

```
function [ActPhase, Output] = SawTooth(OldPhase, Amp, Freq, Ts, StartPhase, Enable, Hold)
if Enable
    if Hold
        ActPhase = OldPhase;
        Output = Amp/(2*pi) * ActPhase;
    else
        ActPhase = (2* pi* Freq *Ts)+ OldPhase + StartPhase;
        if ActPhase > (2 * pi)
            ActPhase = mod(ActPhase,2*pi);
        end
        Output = Amp/(2*pi) * ActPhase;
    end
else
    Output = 0;
    ActPhase = 0;
End
```

Code 4: Triangle Generator

```
function [ActPhase, Output] = Triangle(OldPhase, Amp, Freq, Ts, StartPhase, Enable, Hold)
if Enable
    if Hold
        ActPhase = OldPhase;
         if ActPhase < 0.5*pi
            Output = (2*Amp/pi)*ActPhase;
        else
            if ActPhase < 1.5*pi
                Output = Amp + (Amp/pi)*(pi - 2 *ActPhase);
            else
                Output = -Amp + (Amp/pi)*(2 *ActPhase - 3*pi);
            end
        end
    else
        ActPhase = (2* pi* Freq *Ts)+ OldPhase + StartPhase;
        if ActPhase > (2 * pi)
            ActPhase = mod(ActPhase,2*pi);
        end
        if ActPhase < 0.5*pi
            Output = (2*Amp/pi)*ActPhase;
        else
            if ActPhase < 1.5*pi
                Output = Amp + (Amp/pi)*(pi - 2 *ActPhase);
            else
                Output = -Amp + (Amp/pi)*(2 *ActPhase - 3*pi);
            end
```

```
        end

    end
else
    Output = 0;
    ActPhase = 0;
end
```

Code 5: Filter

First Order filter

```
function Output = Filter(OldOutput, Input, Ts, T1)

Output = (1/(Ts + T1))*((Ts*Input) + (T1*OldOutput));
```

Code 6: Discrete Derivative

The algorithm used is:

The discrete derivative block computes an optionally scaled discrete time derivative as follows:

Y(tn) = (Ku(tn)/Ts) - (Ku(tn-1)/Ts)

Where:

u(tn) and y(tn) are the block's input and output at the current time step, respectively

u(tn-1) is the block input at the previous time step.

K is a scaling factor

Ts is the simulation's discrete step size, which must be fixed

(MathWorks, 2018)

```
function [Output, InputDelay] = DiscreteDerivative(OldInput, Input, Tv, Ts)
InputDelay = Input;
Output = (Tv/Ts)*(Input - OldInput);
```

Code 7: Discrete Integrator

Algorithm used is Backward Euler method: y(n) = y(n-1) + K*[t(n) - t(n-1)]*u(n)

For a given step n > 0 with simulation time t(n), output y(n), input u(n)

K is the gain

(MathWorks, 2018)

```
function [Output, UpperLimitReached, LowerLimitReached]  = DiscreteIntegrator(OldOut, Input, Tn, Ts,
UpperLimit, LowerLimit, Start)
if Start
if Tn > 0
    Out = OldOut + (Ts/Tn)*Input;
    if Out > UpperLimit
        Output = UpperLimit;
        UpperLimitReached = true;
        LowerLimitReached = false;
    elseif Out < LowerLimit
        Output = LowerLimit;
        UpperLimitReached = false;
        LowerLimitReached = true;
    else
    Output = Out;
    UpperLimitReached = false;
    LowerLimitReached = false;
    end
else
    Out = OldOut;
    Output = Out;
    UpperLimitReached = false;
    LowerLimitReached = false;
```

```
end
else
Output = 0;
UpperLimitReached = false;
LowerLimitReached = false;
end
```

Code 8: Draft Disturbance

```
function DraftDist = DraftDisturbance(OldDraftDist, DraftSlope, Ts, Lim, Enable, Hold)

if Enable
    if Hold
        DraftDist = OldDraftDist;
    else
        DraftDist = OldDraftDist + DraftSlope*Ts;
        if DraftDist > Lim
            DraftDist = Lim;
        end
    end
else
    DraftDist = 0;
end
```

Code 9: Noise Generator

The algorithm used to generate random noise is

x_0 = given, $x_{n+1} = P_1 x_n + P_2 \pmod N$ $n = 0,1,2,...$ (*)

Mod N means that the equation containing the expression on the right is divided by N, and then replaced with the remainder

x_0 , N, P, and P_2 can be varied to get a wide range and density of noise. I have used P1 = 269, P2 = 71, N = 1000

(The University of Utah, 2011).

```
function [XOut , Out]  =Rand(XOld, P1, P2, N)

X = XOld*P1 + P2;

XOut = mod(X, N);

Out = XOut/ N;
```

8.1 Algebraic Loops

When coding for loops the problem of algebraic loop error occurs. So, the functions must be adapted accordingly. Few examples are displayed below.

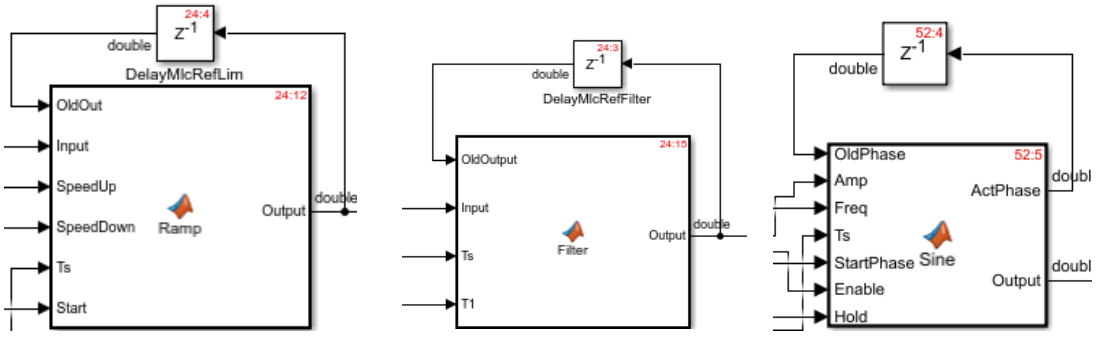


Figure 8.1.1: Solving algebraic loops

8.2 Creating a Library

Once all the functions are created, it is convenient to group them into a library from where they can be reused. The library can be locked, and changes can be made only when unlocked. If changes are made in the library blocks, the changes will be reflected in the blocks which were used from the library. This is of great advantage when there is a requirement to make changes. The changes will be reflected in every block used.

Code 10: Creating a Library

```
function blkStruct = slblocks
        % This function specifies that the library should appear
        % in the Library Browser and be cached in the browser repository

        Browser.Library = 'Control';
        % 'Control' is the name of the library

        Browser.Name = 'Control';
        % 'Control' is the library name that appears in the Library Browser

        blkStruct.Browser = Browser;
```

8.3 Programming

After the library blocks are created. They can now be dragged, dropped and connected to create further functions. Library blocks are not sufficient to program everything. So the user defined mat functions can be further used to create the rest of the functions, such as mathematical functions and logical operators.

Figure below is the complete view the entire program.

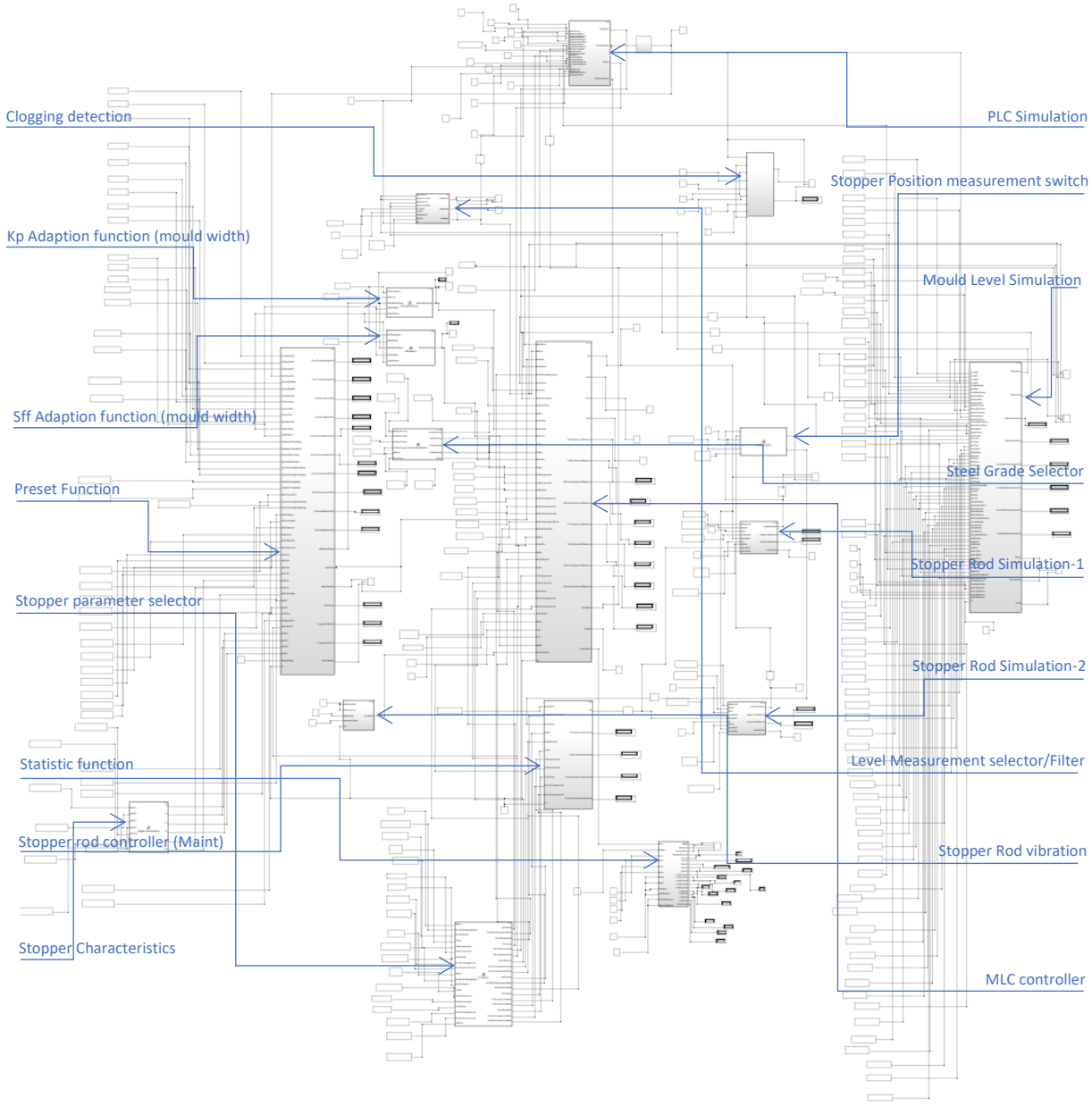


Figure 8.3.1: Complete view of the entire Simulink program

It is the exact representation of the program which is running in the PLC. In case of Simulink it is a bit different in terms of execution order, parameter assignment and connection of data.

Each block takes its parameter from a constant block with a variable assigned to it. The datatype of the variables also must be assigned because the generated code for the PLC will have the same variables with same data types.

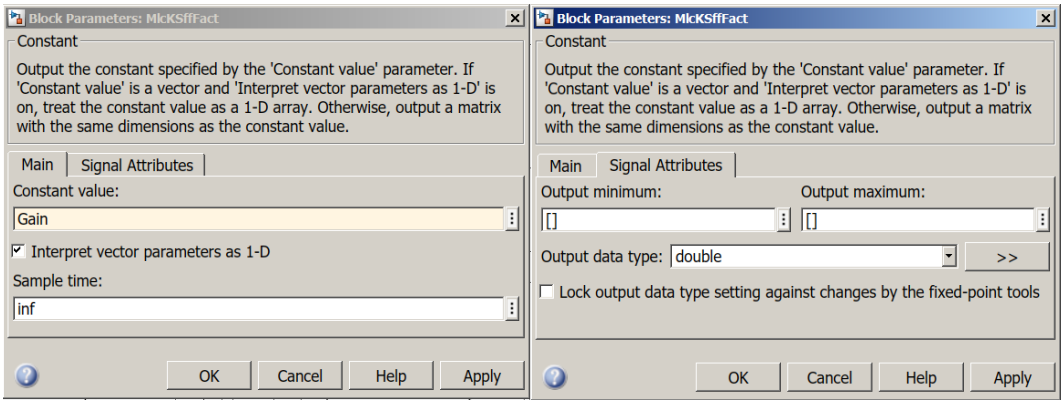


Figure 8.3.2: Assigning parameter and data type

The value to the variable is assigned in the command window. It can be done my creating an (.m) file or mat file. On running the file all the variables will contain the assigned values.

Few additional functions are created, which are not a part of the main program, such as the PLC function created for simulation purpose to emulate the role of the external S7-400 PLC and its signals coming to the S7-1500 PLC.

Steel Grade Selector

The purpose of this block is to assign values on selection of a steel grade. Originally this block was supposed to take 2D array or 10X5 matrix, as input and the output was supposed to be 5 values which make the gain parameters.

But there are few limitations to the code generator, when dealing with arrays. It cannot generate a code for 2D arrays moreover, there is a difference in array index, In MATLAB array starts from 1 and in TIA Portal it starts from 0. It creates a lot of confusion in code generation which fails to function properly.

Therefore 5 row matrices were used as input and the index assignment was made adaptable.

Code 11: Steel Grade Selector

```
function [KpSteelGroup, TnSteelGroup, TvSteelGroup, T1SteelGroup, KdOn] =
SteelGradeSelector(KpSteelGroupIn, TnSteelGroupIn, TvSteelGroupIn, T1SteelGroupIn, KdOnIn, SteelGroup )

if SteelGroup >=1 && SteelGroup <= 10
   Index = SteelGroup;
else
     Index = 1;
end

    KpSteelGroup = KpSteelGroupIn(Index);
    TnSteelGroup = TnSteelGroupIn(Index);
    TvSteelGroup = TvSteelGroupIn(Index);
    T1SteelGroup = T1SteelGroupIn(Index);
    KdOn = KdOnIn(Index);
```

Stopper Characteristics is a similar block. Similar solution was applied to it also.

Every function when generated a code will be converted into Function block in Tia portal.

9 TIA Portal

The TIA Portal (Totally Integrated Automation Portal) is the latest automation software from Siemens. It is a single software containing one engineering environment and it allows to create only one software project for all automation tasks. The previous generation of softwares which had to be used separately for different tasks has been integrated into one software. It has been designed in such a way that it can provide maximum user-friendliness and engineering efficiency. Every software package has been integrated in an engineering framework such as from hardware configuration and programming to visualization.

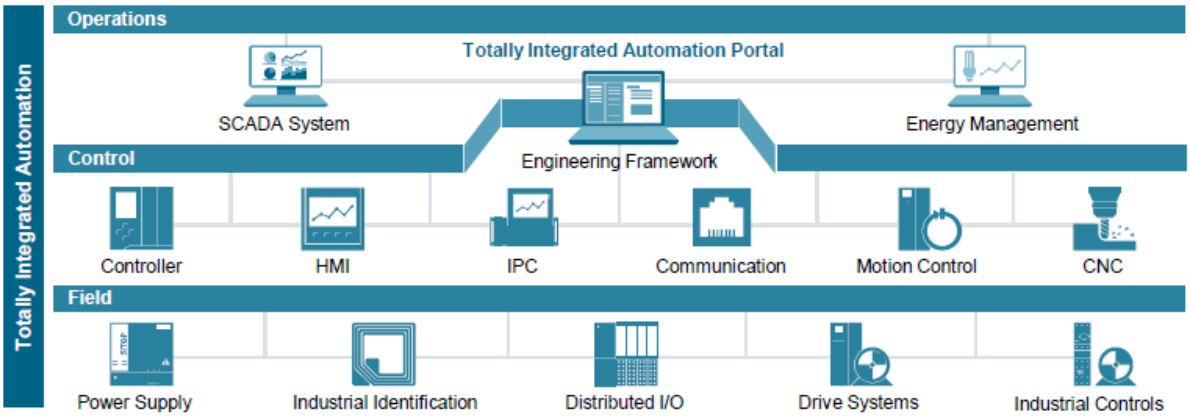


Figure 8.3.1: Totally integrated automation overview

(TIA Portal V15 news, 2018)

There are tons of features which are irrelevant to this project. So, I will not discuss each of them. I will only discuss General Hardware configuration and basic programming guidelines.

9.1 Hardware configuration

On the start-up page it allows to select various hardware and devices. On the device and network tab we can choose any PLC hardware, HMI hardware or we can create a PC station for SCADA or other data exchange purposes. It gives a list of all the compatible versions of hardware to choose from.

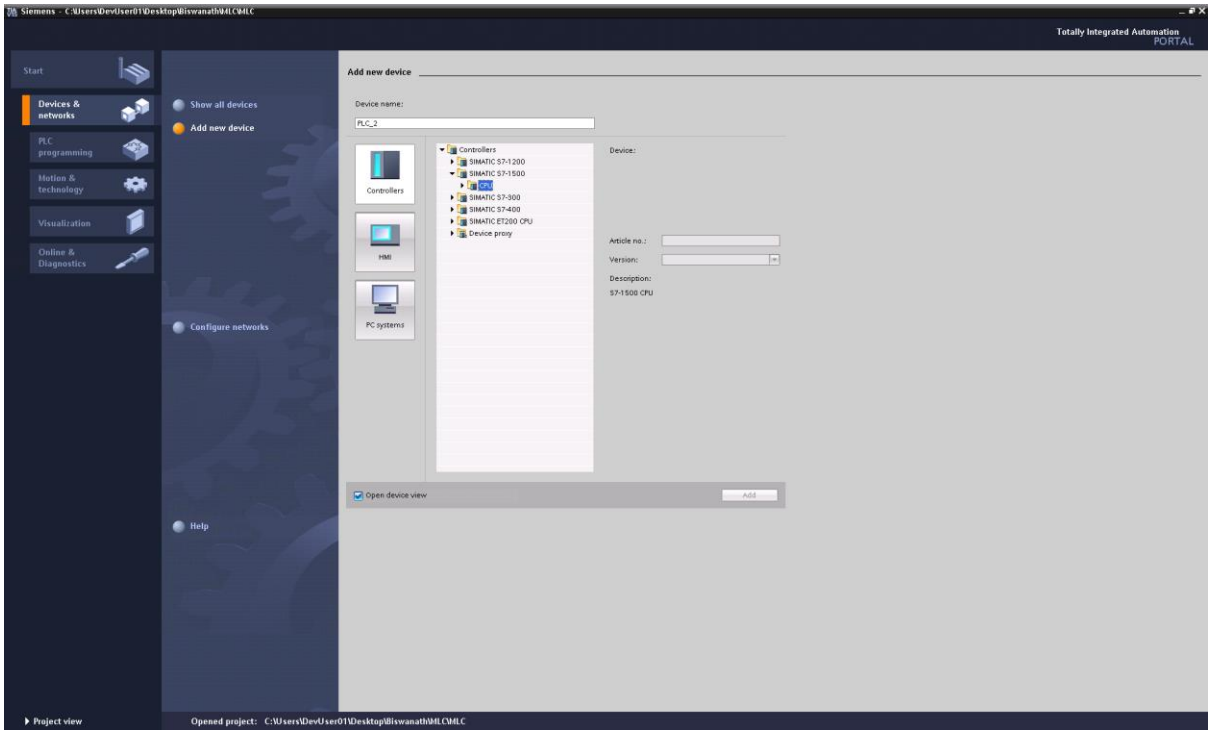


Figure 9.1.1: Selection of hardware

It has a hardware detection feature which can be used if the exact model of the hardware in not known.

Different hardware like extension modules for RIO (remote input output) or I/O modules can be added to the project by dragging and dropping from the hardware catalogue. After that, the network configuration can be defined. This can be displayed in the network view.

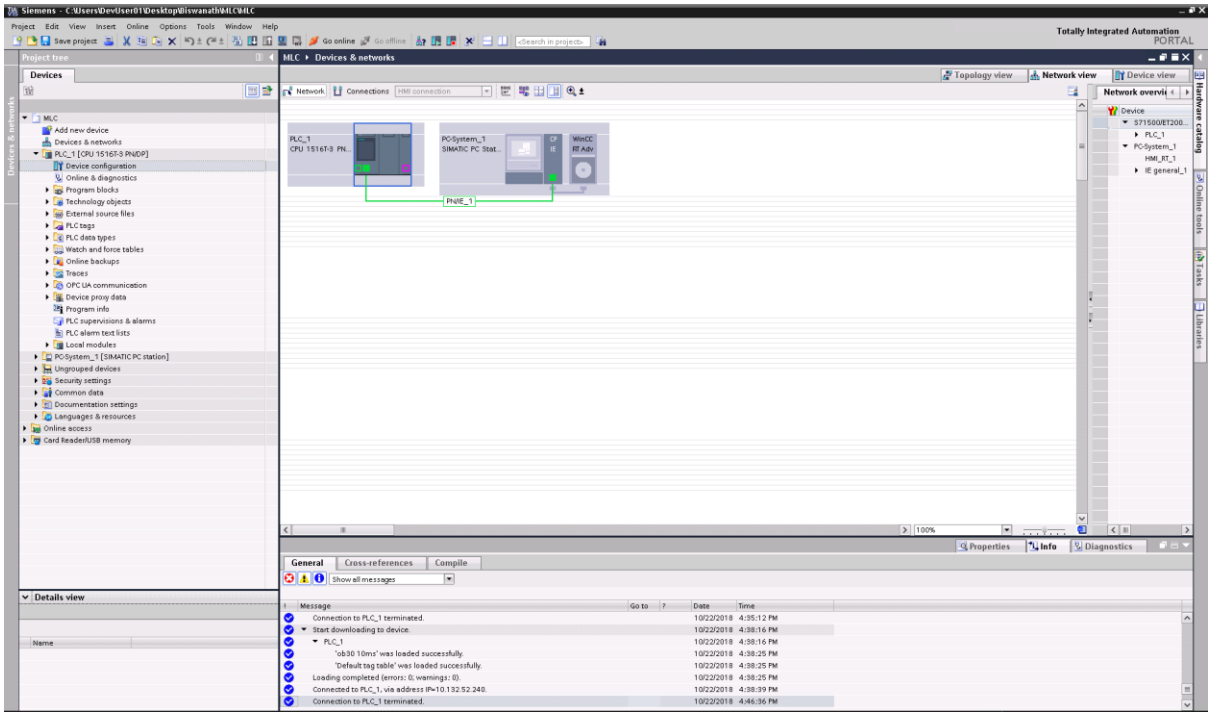


Figure 9.1.2: Network view

Under the properties tab, the entire device configuration, such as change of IP address, data exchange and other settings can be done.

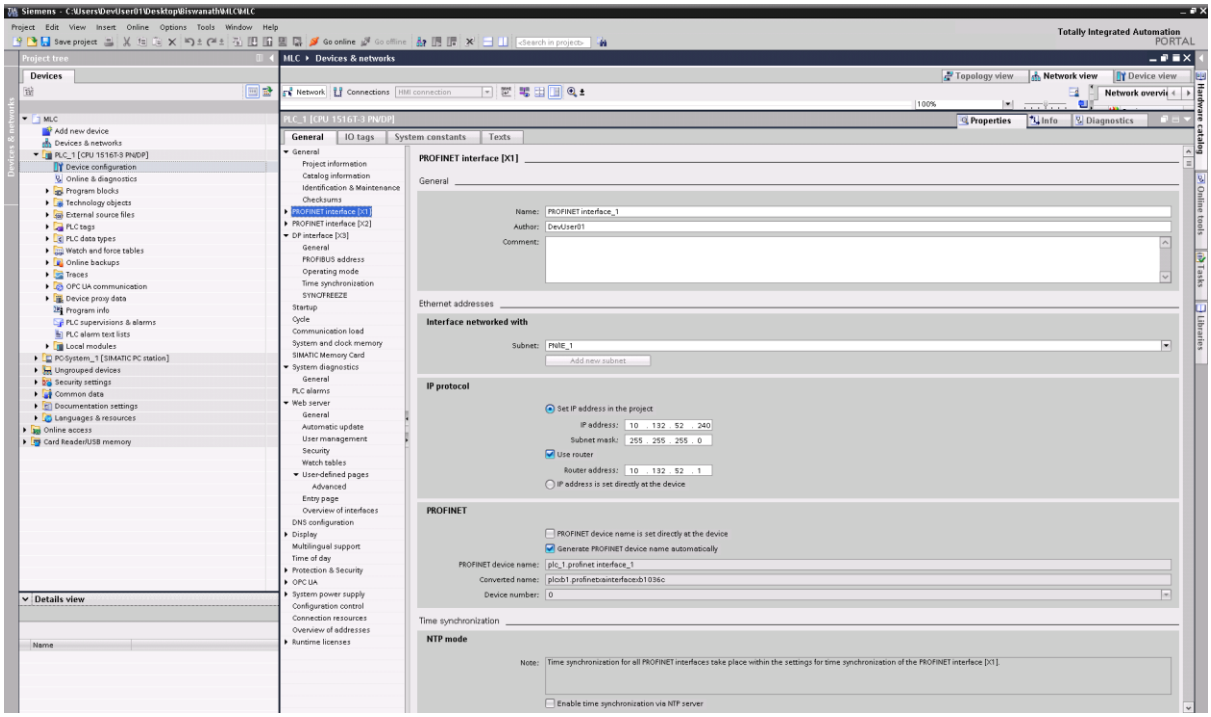


Figure 9.1.3: Properties tab

After the entire configuration is done it can be compiled and tested for errors. After that it must be downloaded into the hardware.

 (The download icon on TIA Portal).

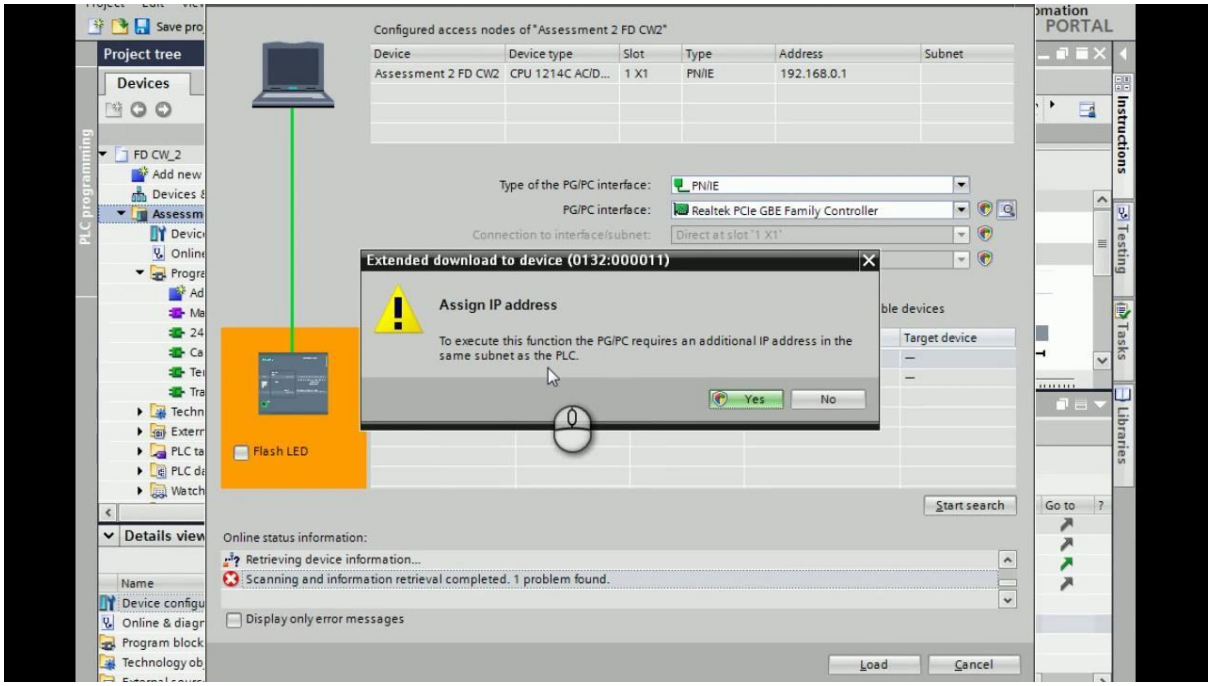


Figure 9.1.4: Downloading

(TIA Portal SIMATIC Creating the project and hardware, 2013)

9.2 Programming in TIA Portal

The common programming languages in PLC are

- LAD (Ladder Logic)
- FBD (Function Block Diagram)
- STL (Statement List)
- SCL (Structured Control Language)
- Graph (SFC (Sequential Function Chart) or CFC (Continuous Function Chart)

All the above-mentioned languages are compatible with S7-1500 PLC.

When we talk about programming in PLC we come across few Terms called blocks.

- Organization blocks (OBs)
- Function blocks (FBs)
- Functions (FCs)
- Data blocks (DBs).

The above mentioned are a part of the user program.

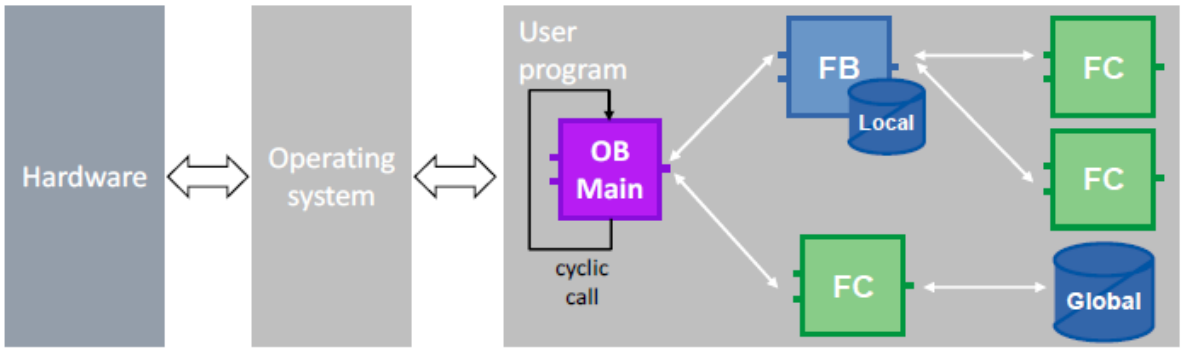


Figure 9.2.1: Operating system and user program

The user program keeps executing in a cyclic loop.

9.2.1 Organization blocks (OBs)

The Main OB is present in the project by default. It is the OB which is executed first. A new OB can be added by clicking on the Add new block icon.

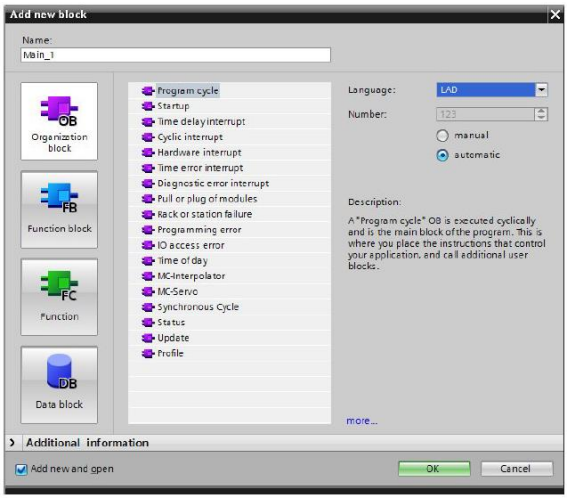


Figure 9.2.2: choosing OBs

OBs are the interface between the operating system and the user program. They control processes like, Start-up behaviour of the controller, Cyclic program processing, Interrupt controlled program processing, error handling.

There are several types of OBs as can be seen in the figure above. For example

Startup OB: You can specify the conditions for starting up your CPU (initialization values for RUN, startup values for I/O modules) by writing your program for the startup in the Organization blocks OB100 for restart (warm restart), OB101 for hot restart, or OB102 for cold restart.

Time Delay interrupt OB: OBs with which delayed execution of parts of your user program can be performed are called time delay interrupt OBs. Triggering of the time delay interrupts happen when the delay time specified in SFC32 SRT_DINT has expired.

Hardware interrupt OB: OBs that react to signals from the external modules such as signal modules (SMs), communications processors (CPs), function modules (FMs) are called Hardware interrupt OBs. The OBs can be configured to the signals from digital or the analog modules. Which means the decision can be made to start the OB based on the signals.

Cyclic Interrupt OB: OBs that interrupt the cyclic processing of a program at certain intervals are called cyclic interrupt OBs. Cyclic interrupts are initiated after certain intervals. The interval starts at the time when the CPU changes from STOP to RUN mode.

9.2.2 Rules for Cyclic Interrupts

When the interval is specified, it must be made sure that there is sufficient time between the start events of the individual cyclic interrupts for processing the cyclic interrupts themselves. If parameters are assigned to deselect cyclic interrupt OBs, they cannot be started. The PLC will identify a programming error and will switch to STOP mode.

The interval in the cyclic interrupts parameter block using TIA Portal. There is a default interval for each OB and a priority class which can be changed.

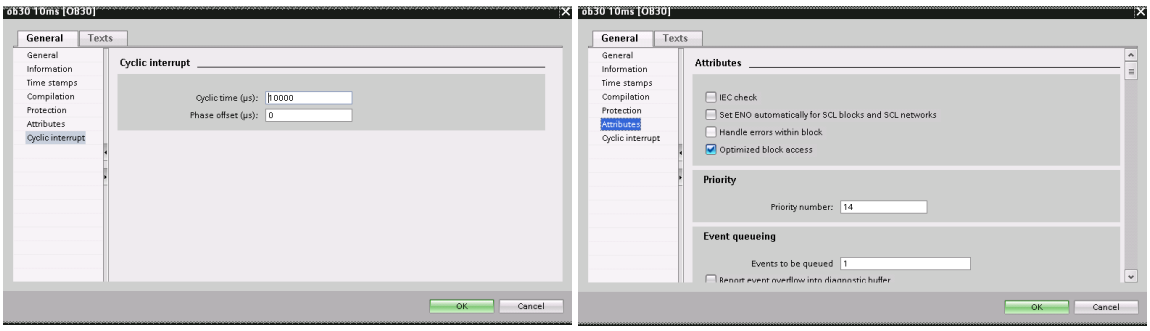


Figure 9.2.3: Assigning parameters to cyclic interrupt OB

For this project I am using OB1 and OB30 cyclic interrupt OB.

9.2.3 Functions (FC)

FCs are blocks which do not have cyclic data storages. They cannot save the values of block parameters until the next call and they must be provided with actual parameters when called.

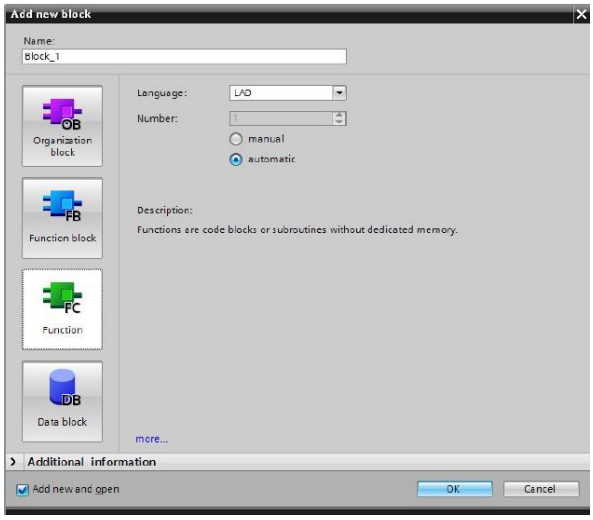
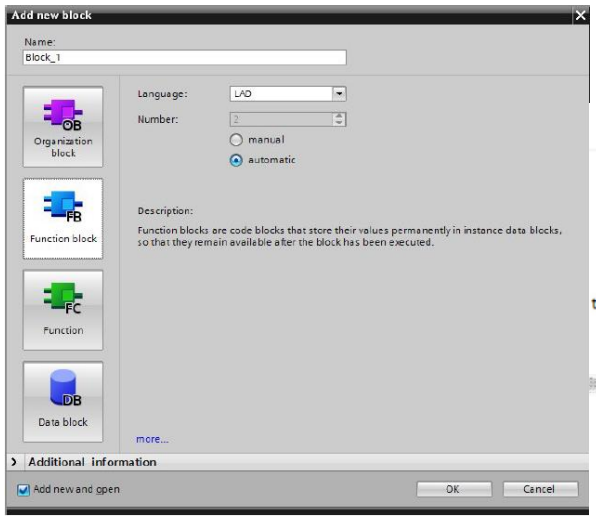


Figure 9.2.4: Adding a new FC

9.2.4 Function blocks (FB)



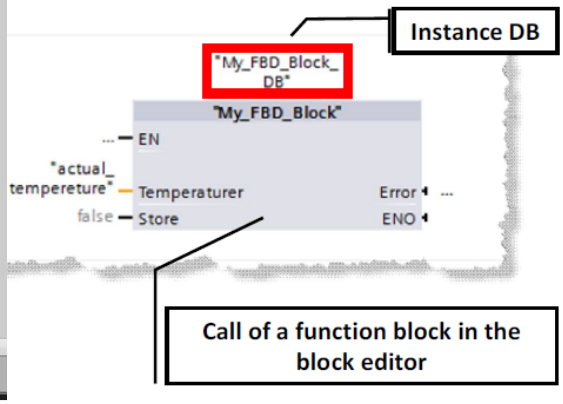


Figure 9.2.5: Adding a new FB

FBs are blocks with cyclic data storage, in which values are permanently stored. The cyclic data storage happens in an instance DB. The values are passed from cycle to cycle as they have static tags. A FB can be called multiple times in different parts of the program. This makes it possible to create a reusable code and apply the same piece of code in several locations.

9.2.5 Instances

The call of a function block is called instance. The data with which the instance is working is saved in an instance DB.

9.2.6 Instance Data Blocks

Instance DBs need not be created manually. They are generated automatically when an FB is called. The structure of the instance is defined in the related FB and all the changes can be done inside the FB. Instance DBs are only assigned to an FB. Instance DBs are always created according to the specifications in the FB interface that's why they cannot be changed in the instance DB.

9.2.7 Multi-instances

Function blocks can store their data in the instance DB of another FB if they are called inside another FB. This is called multi-instances. Therefore, a function block always saves their data in the instance data block of the higher-level FB. There is no change in the functionality of the blocks with multi-instances.

9.2.8 Global data blocks (DB)

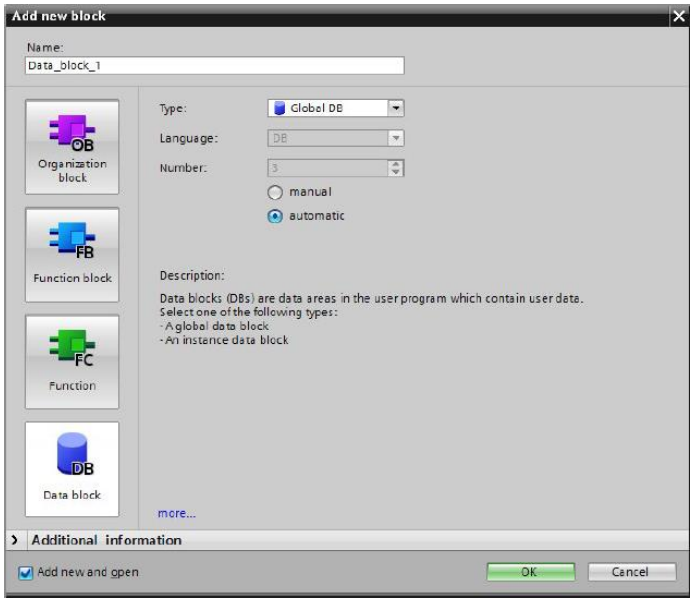


Figure 9.2.6: Adding a DB

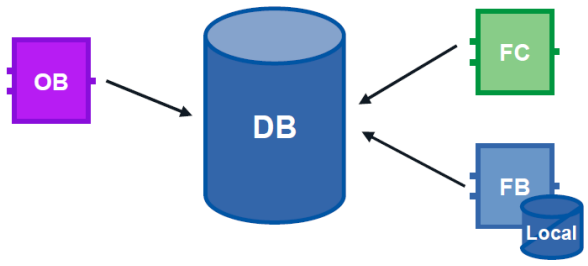


Figure 9.2.7: Global DB as central data memory

Variable data is located in data blocks that are available to the entire user program.

(Programming Guideline for S7-1200/S7-1500, 2014)

9.3 Transferring the Simulink generated code to the PLC

The file which Simulink generates is a (.SCL) extension. The SCL can be viewed easily by using notepad.

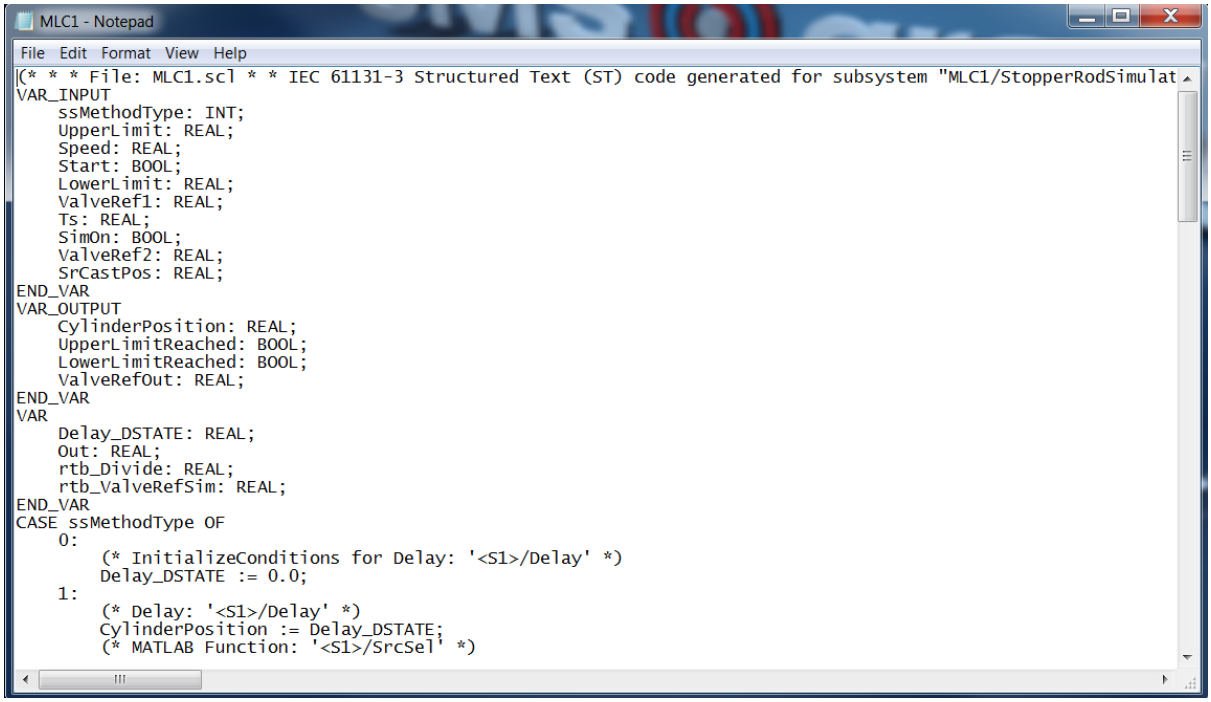


Figure 9.3.1: SCL Code

To transfer it to TIA Portal we have to use the feature add external file.

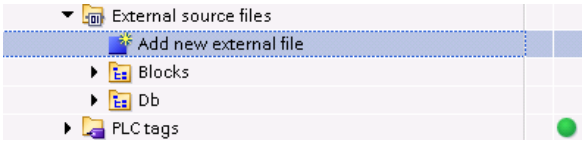


Figure 9.3.2: Adding an external file

On clicking the **add new external file** icon it will ask for a path to the file.

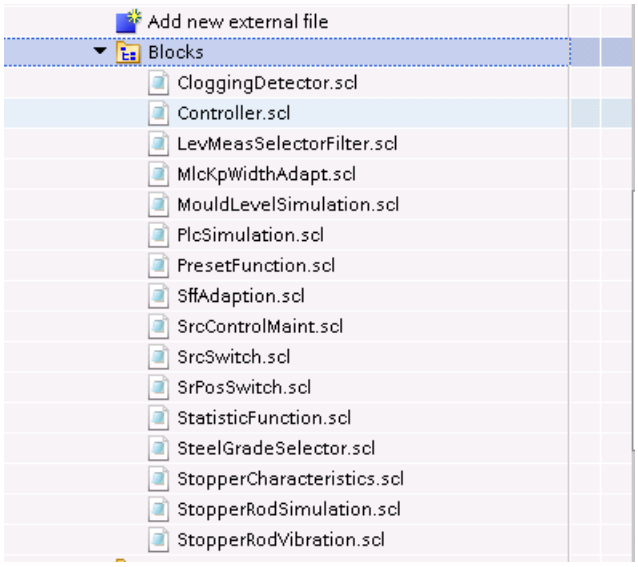


Figure 9.3.3: External SCL files

Same way files can also be exported. In the figure below, I am exporting a Data Block. It will become a (.DB) file and can also be viewed in Notepad.

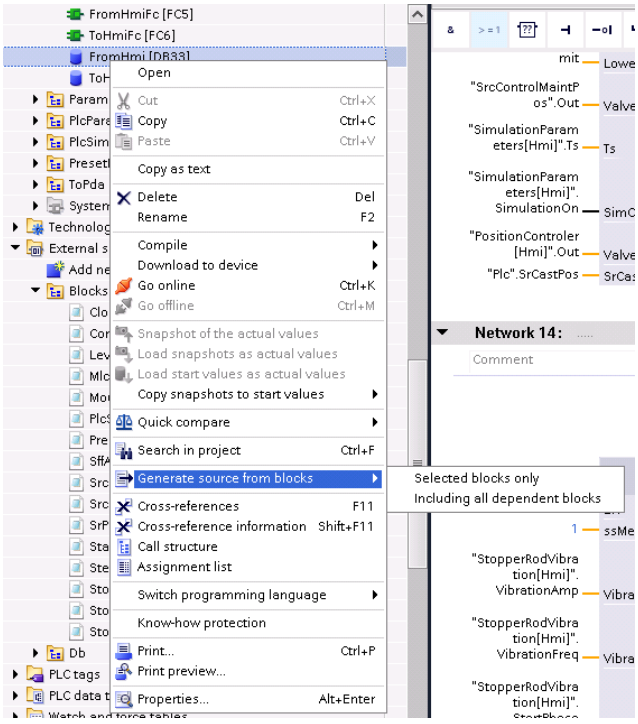


Figure 9.3.4: Exporting files

Figure 9.3.5: External DB file

When the files are added it can be viewed in TIA portal. Now (FBs) or function blocks can be generated from the source. As shown in figure below.

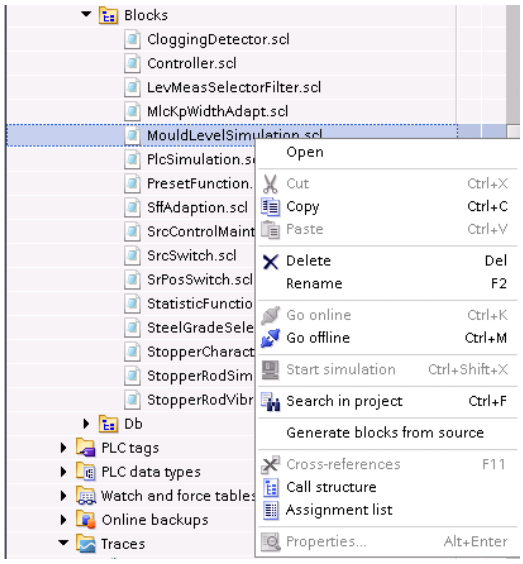


Figure 9.3.6: Generating blocks from source

The generated function blocks can be viewed under Program blocks.

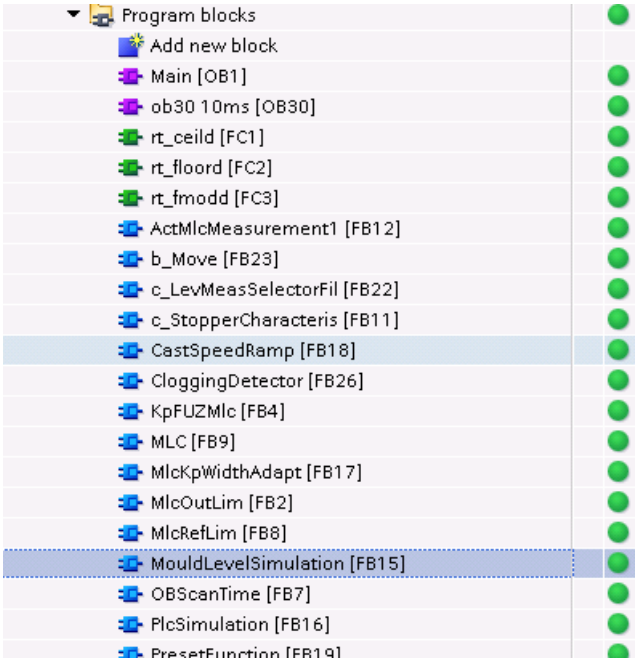


Figure 9.3.7: Generated function blocks

Once the function blocks are generated it can be dragged to the OB. On dropping the FB to the OB it will generate an instance DB as shown in the figure below. Then all the relevant parameters can be

connected to the inputs. The function blocks generated will have exactly the same inputs as the Subsystems in Simulink from where the code is generated. The inputs will also have the same data types as in simulink.

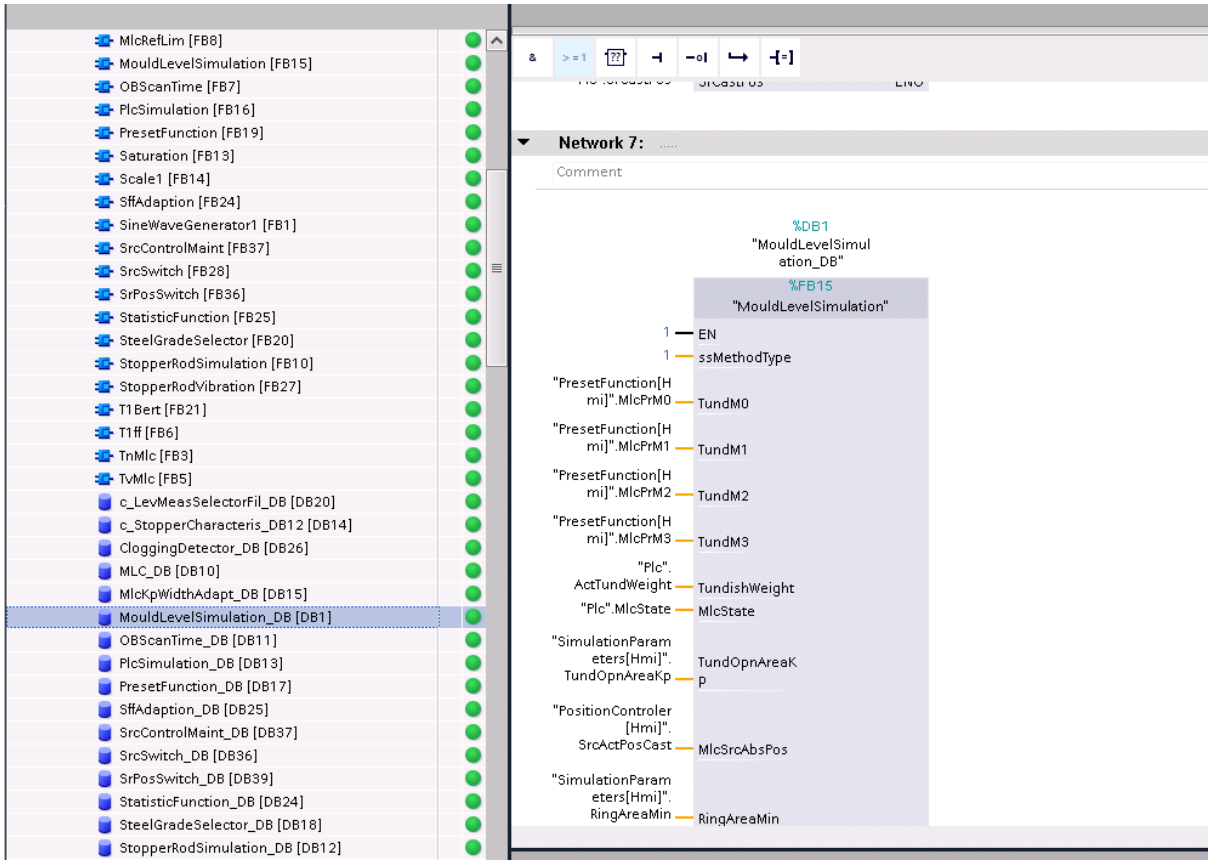
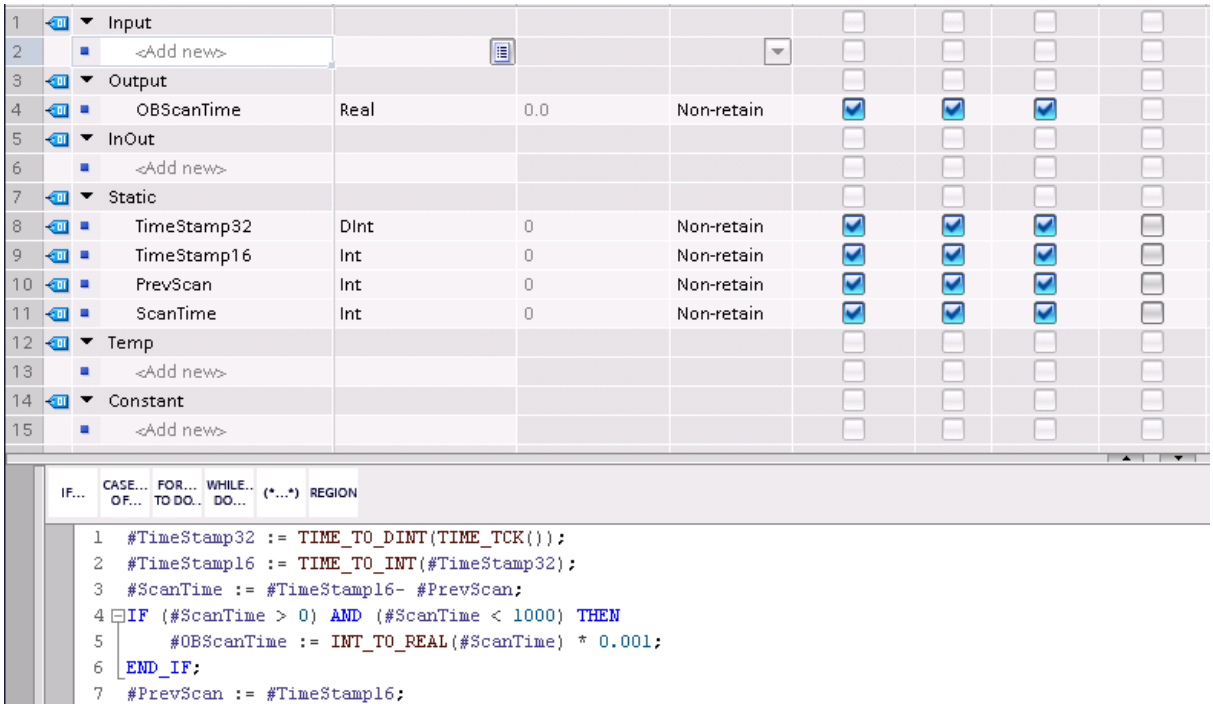


Figure 9.3.8: Calling a function block automatically generates instance DBs

9.4 Sample time generator

To feed the sample time to every sample time dependent block, I am using a function block, which generates the sample time value of the OB in which it is placed. If the sample time of the OB is changed the value will adapt to it.



1	Input							
2	<Add new>							
3	Output							
4	OBScanTime	Real	0.0	Non-retain	☑	☑	☑	☐
5	InOut							
6	<Add new>							
7	Static							
8	TimeStamp32	DInt	0	Non-retain	☑	☑	☑	☐
9	TimeStamp16	Int	0	Non-retain	☑	☑	☑	☐
10	PrevScan	Int	0	Non-retain	☑	☑	☑	☐
11	ScanTime	Int	0	Non-retain	☑	☑	☑	☐
12	Temp							
13	<Add new>							
14	Constant							
15	<Add new>							

IF... CASE... OF... FOR... TO DO.. WHILE.. DO... (*...*) REGION

```
1  #TimeStamp32 := TIME_TO_DINT(TIME_TCK());
2  #TimeStamp16 := TIME_TO_INT(#TimeStamp32);
3  #ScanTime := #TimeStamp16- #PrevScan;
4  IF (#ScanTime > 0) AND (#ScanTime < 1000) THEN
5      #OBScanTime := INT_TO_REAL(#ScanTime) * 0.001;
6  END_IF;
7  #PrevScan := #TimeStamp16;
```

Figure 9.4.1: Sample time generator

9.5 6.3.2 Memory allocation

All the inputs and outputs will have Non-retain type memory (it will be erased on PLC restart). And all the delay states will be allocated to the static memory on the FB. Therefore, if there is a situation of limited memory usage, one must be careful about the delay state usage.

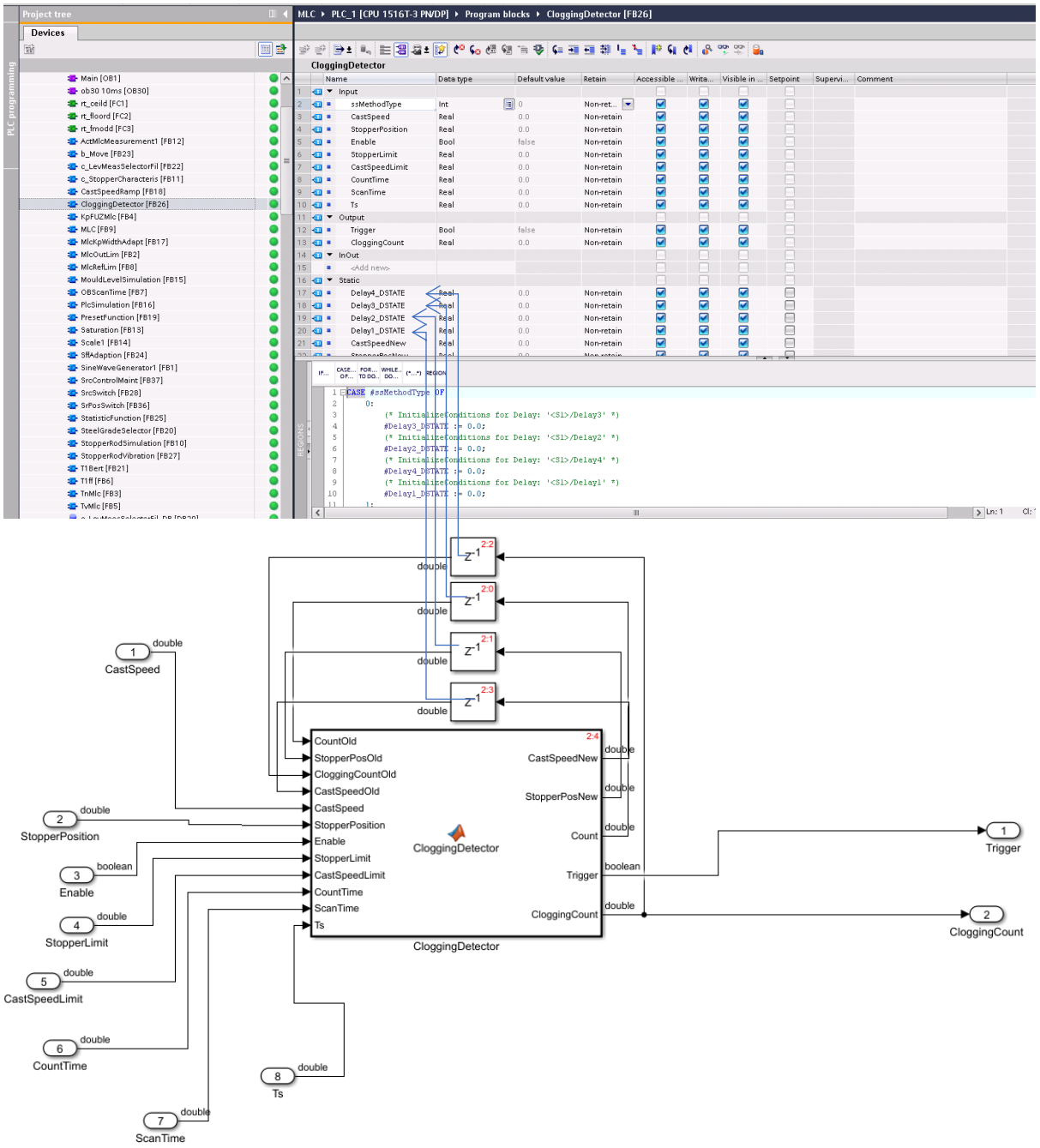


Figure 9.5.1: Static Dbs

There will be two additional inputs on the FB. Two more than what has been programmed on Simulink. They are auto generated. One is EN, which enables or disables the block. They both are Boolean data type. When the EN is true it will enable the block, which means the PLC will execute the program inside the block during it scan cycle. If it is false it will skip the block during execution.

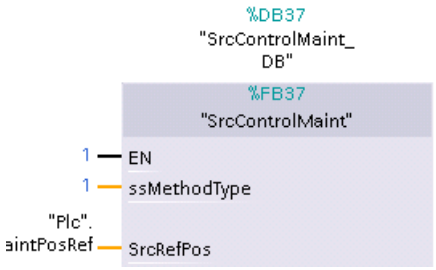


Figure 9.5.2: Enable port

The second input is SSMethodType. If it is true it will retain the delay states or else, it will make the delay states to zero. It can be helpful if there is a requirement to reinitialize the values during the execution of the program.

```
CASE ssMethodType OF
    0:
        (* InitializeConditions for Delay: '<S1>/Delay' *)
        Delay_DSTATE := 0.0;
    1:
        (* Delay: '<S1>/Delay' *)
        CylinderPosition := Delay_DSTATE;
        (* MATLAB Function: '<S1>/SrcSel' *)
        (* MATLAB Function 'StopperRodSimulation/SrcSel': '<S4>:1' *)
        (* '<S4>:1:3' if CastPos ~= 2 *)
        IF SrCastPos <> 2.0 THEN
            (* '<S4>:1:4' Ref = Ref1; *)
            ValveRefOut := ValveRef1;
        ELSE
```

Figure 9.5.3: SSMethdType case

When programming we are bound to make mistakes. If it is required to update a block, it is unnecessary to do the whole process of dragging, generating instance DB and then connecting the parameters. Update block call option can be used to update the block. The new block will have all the changes and all the input parameters of the unchanged inputs will remain connected.

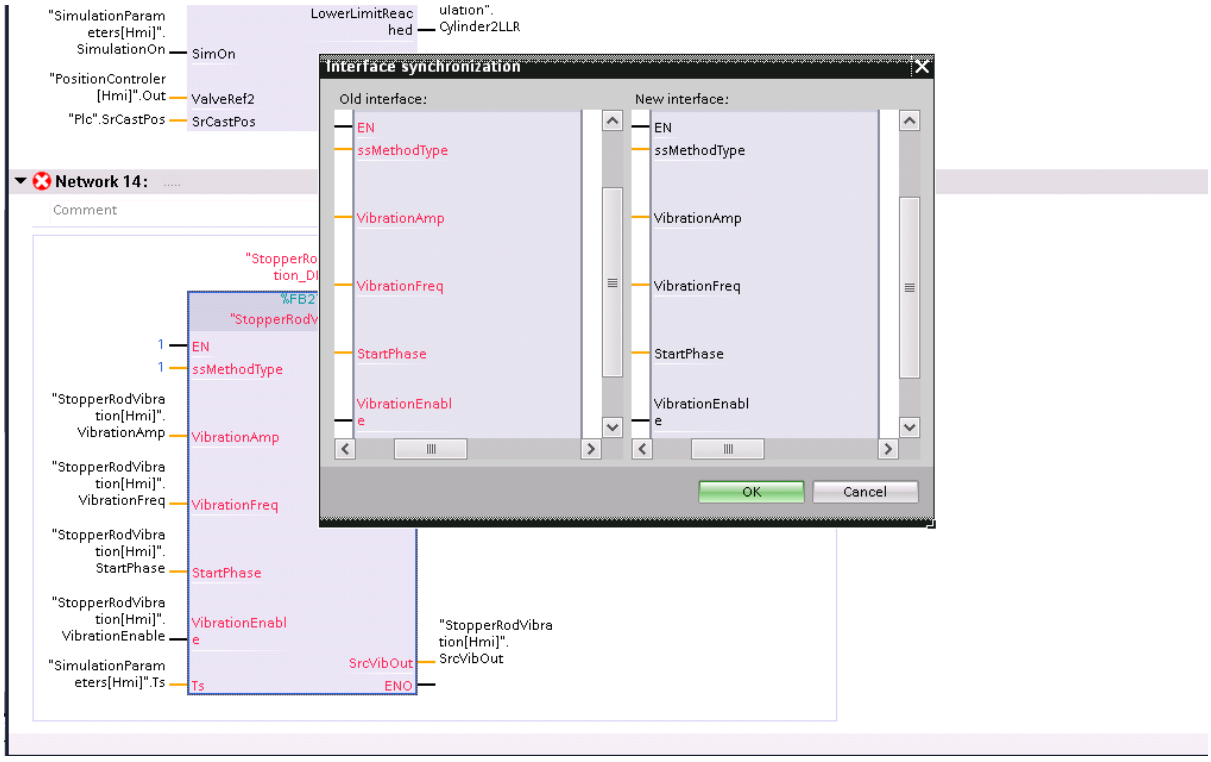


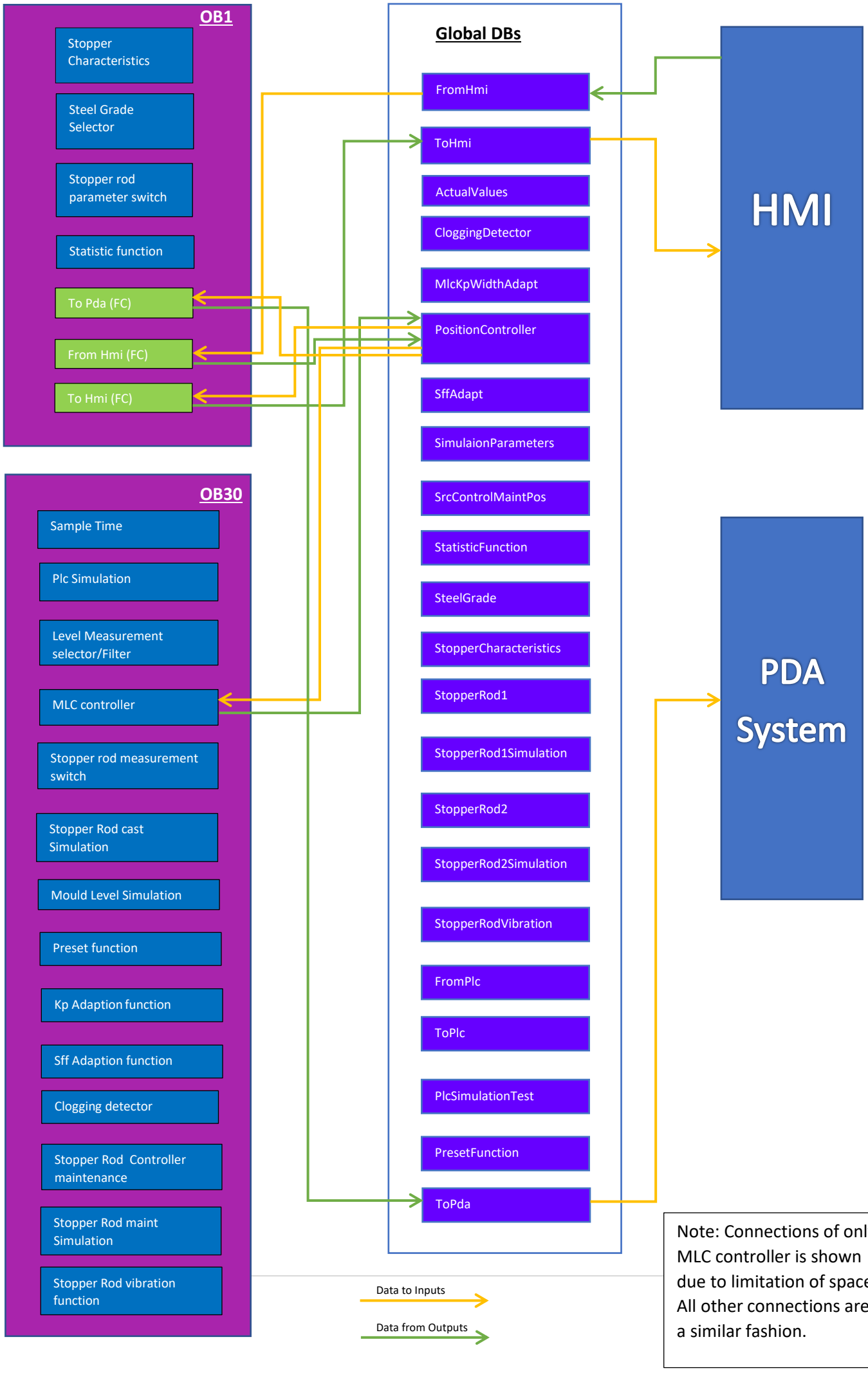
Figure 9.5.4: Updating block call

9.6 Organisation of the blocks in TIA Portal

There are same numbers of FBs as the number of subsystems in Simulink. I have used 2 OBs, OB1 which is the main OB and OB30 which is the cyclic OB. Scan cycle was 10 milliseconds. I have several DBs for data exchange between the blocks and the HMI.

Organising the blocks in a manner is very important. It should match the execution order of the program in Simulink or else the program will not work as expected.

In Simulink the execution order can be customised but in TIA Portal it always executes the program from top to bottom. I have kept few functions in OB1.These function does not require fast scan cycle to work precisely. This will save the work memory of the PLC and therefore there will be less stress on the PLC.


OB1
Stopper Characteristics
Steel Grade Selector
Stopper rod parameter switch
Statistic function
To Pda (FC)
From Hmi (FC)
To Hmi (FC)
OB30
Sample Time
Plc Simulation
Level Measurement selector/Filter
MLC controller
Stopper rod measurement switch
Stopper Rod cast Simulation
Mould Level Simulation
Preset function
Kp Adaption function
Sff Adaption function
Clogging detector
Stopper Rod Controller maintenance
Stopper Rod maint Simulation
Stopper Rod vibration function
Global DBs
FromHmi
ToHmi
ActualValues
CloggingDetector
MlcKpWidthAdapt
PositionController
SffAdapt
SimulaionParameters
SrcControlMaintPos
StatisticFunction
SteelGrade
StopperCharacteristics
StopperRod1
StopperRod1Simulation
StopperRod2
StopperRod2Simulation
StopperRodVibration
FromPlc
ToPlc
PlcSimulationTest
PresetFunction
ToPda
HMI
PDA System
Data to Inputs
Data from Outputs
Note: Connections of only MLC controller is shown due to limitation of space. All other connections are in a similar fashion.

10 HMI with TIA Portal

Lot of HMI hardwares can be configured using the TIA Portal. It gives a list of all the Siemens make HMI hardwares, as can be seen in the figure below.

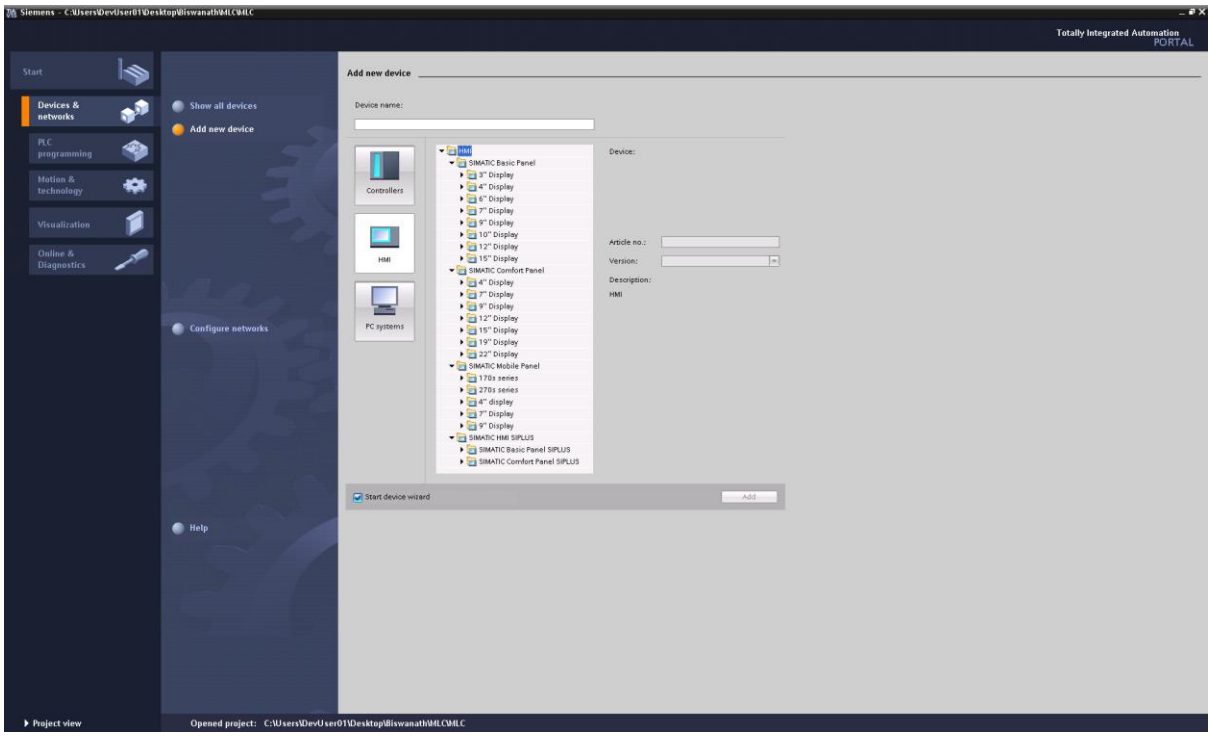


Figure 9.6.1: Choosing HMI hardware

But for my project I did not use a separate HMI. I created a SCADA station.

For that a PC station must be created first.

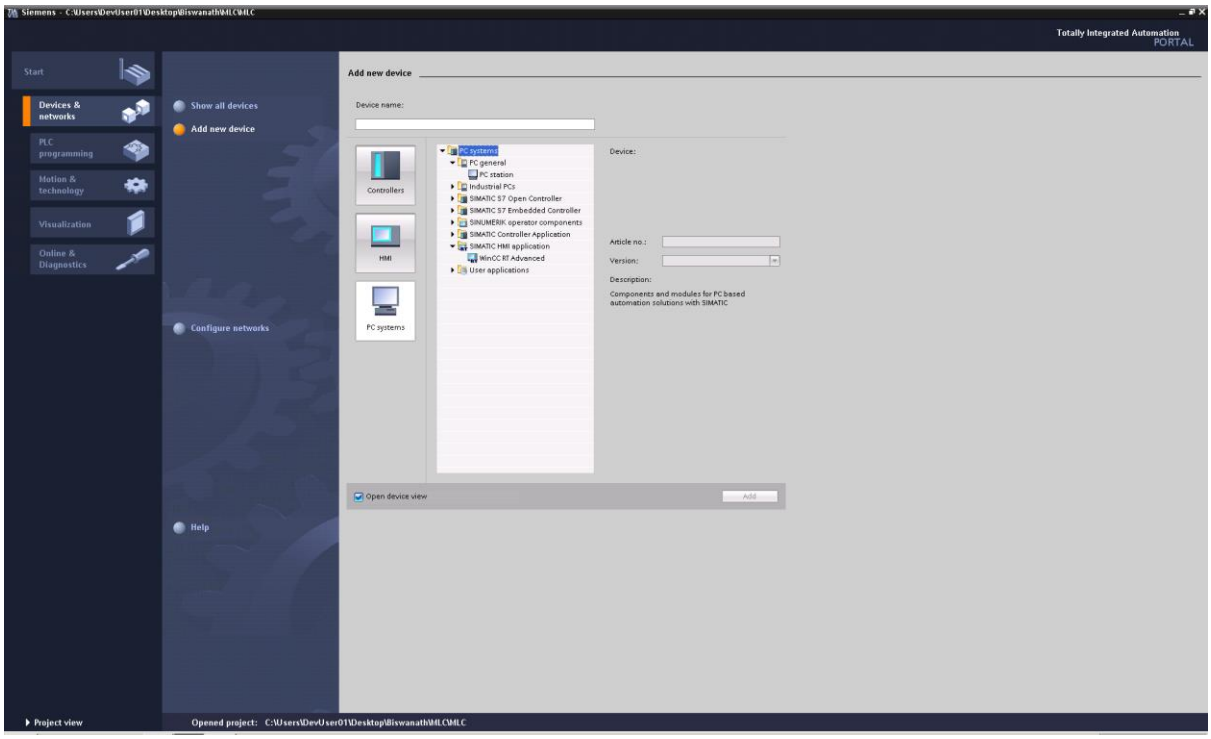


Figure 9.6.2: Choosing a PC station

After the PC station is created. I added IE general and WinCC RT Adv from the hardware catalogue and assigned the IP address.

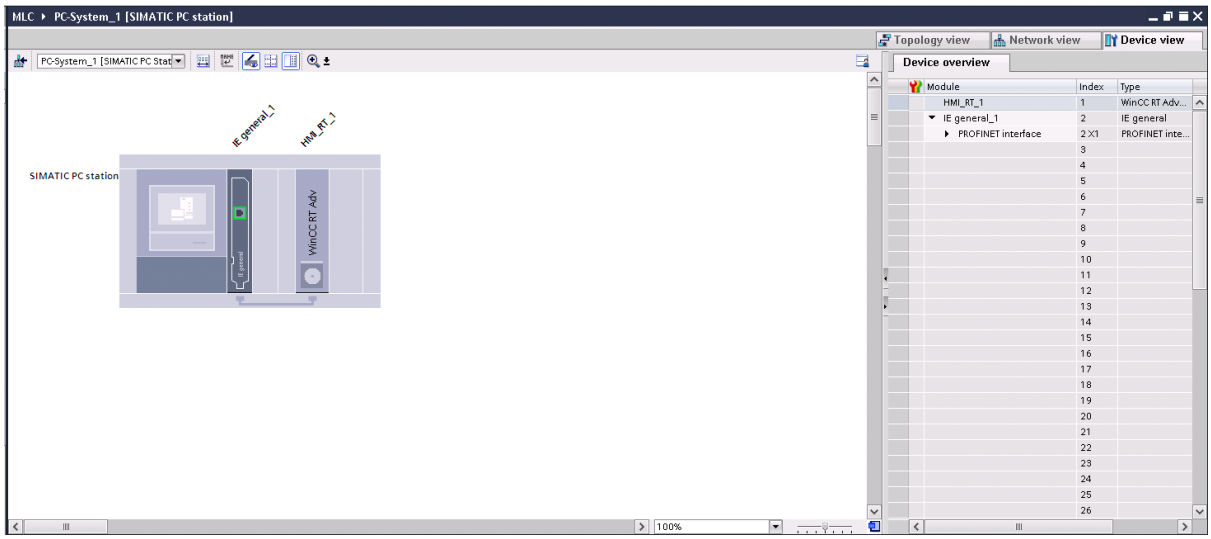


Figure 9.6.3: WnCC RT Advanced

Then it should be connected to the PLC.

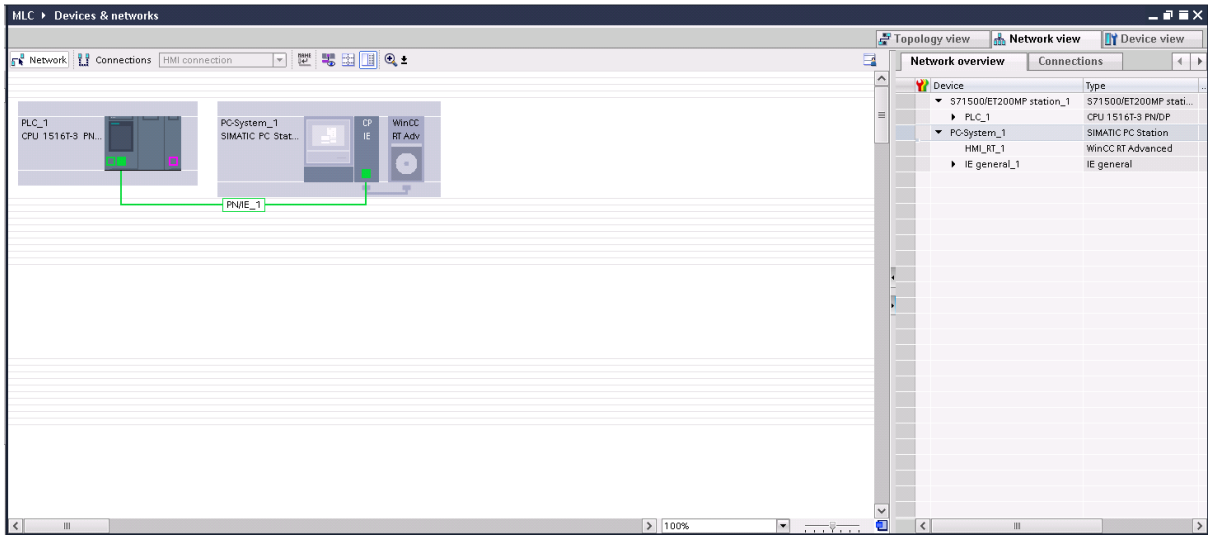


Figure 9.6.4: Network view, HMI connected to PLC

Then the screens, templates, Pop-up screens, slide in screens can be added.

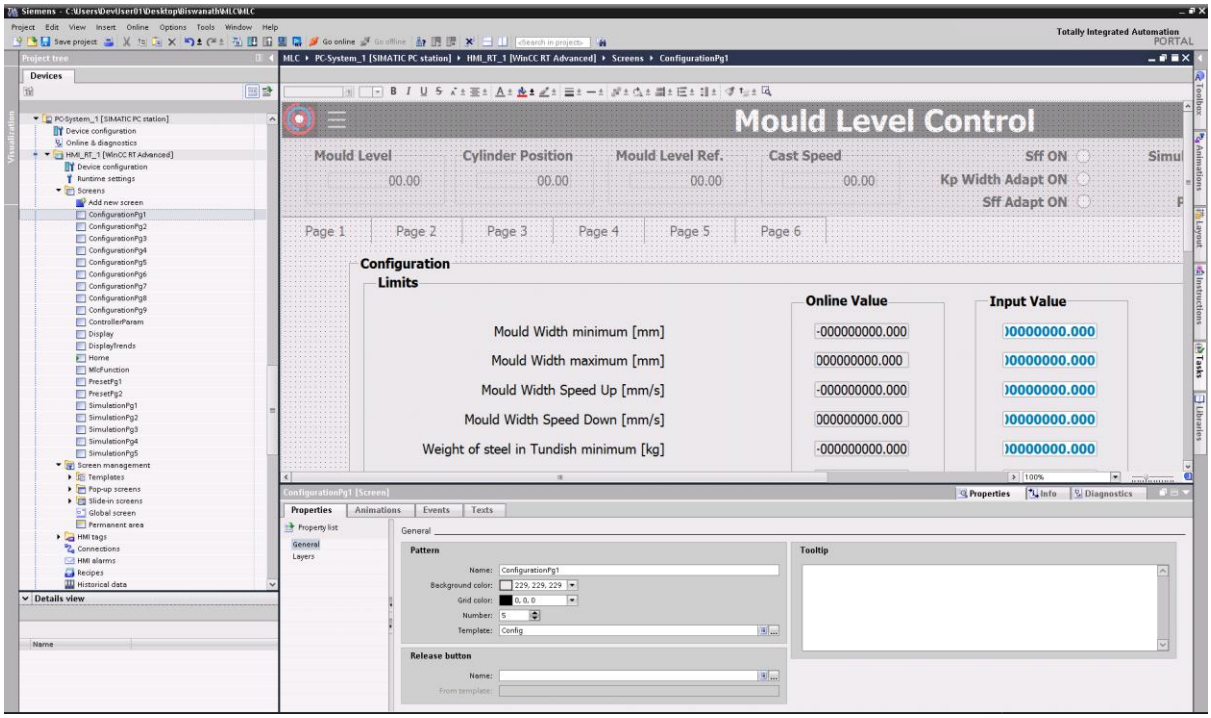


Figure 9.6.5: Screens

To the screens, Buttons, input fields, Output fields can be added and linked to the tags.

Appearance such as colour, layout and styles can be configured in the properties tab.

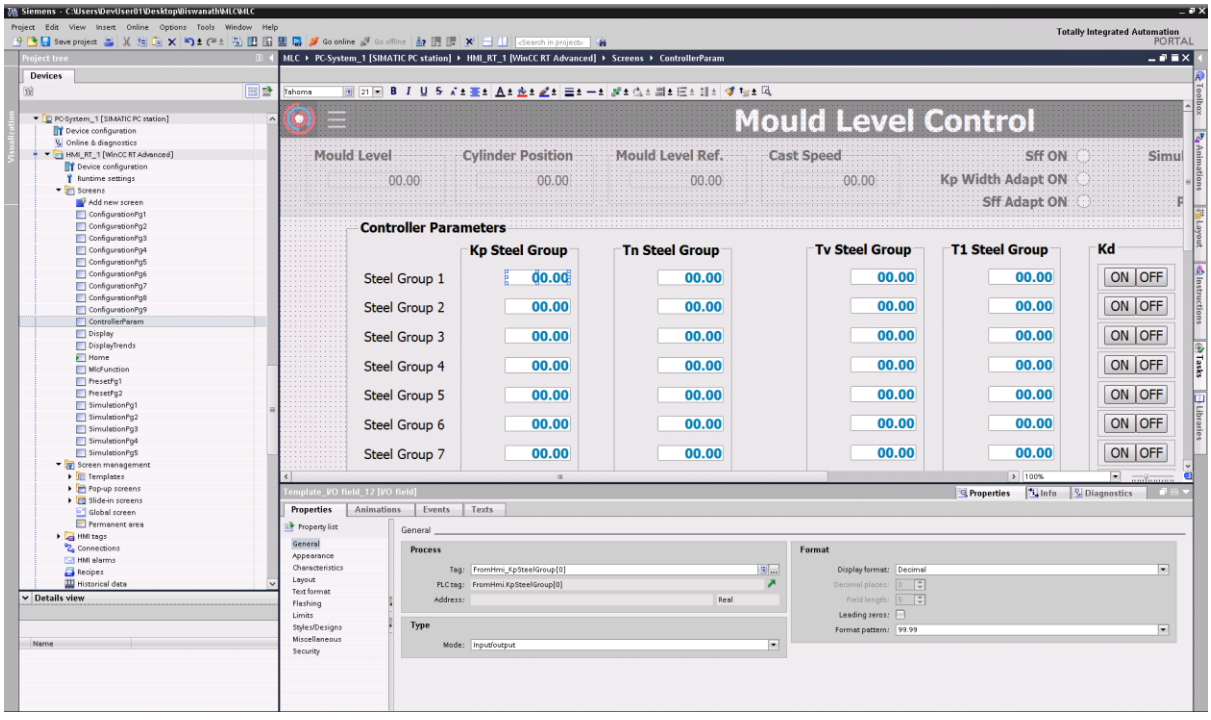


Figure 9.6.6: Screens

Animation and Events can also be configured.

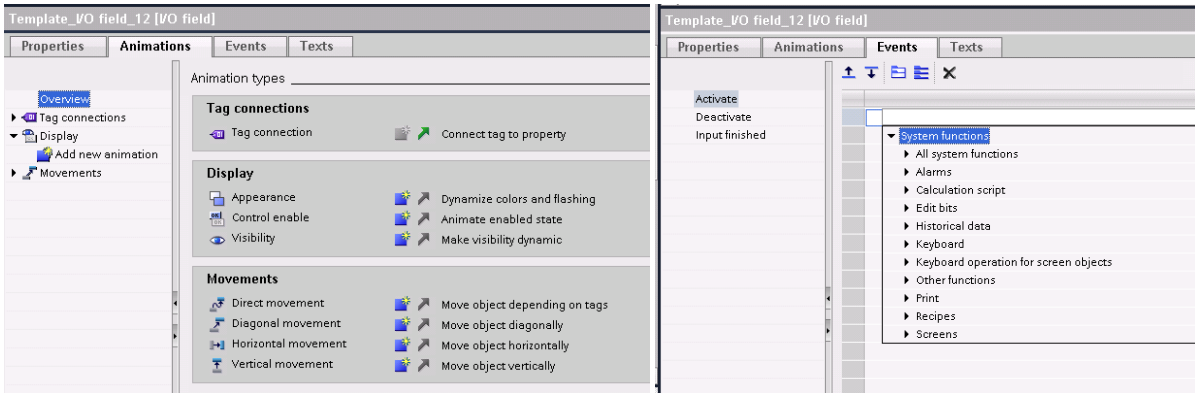


Figure 9.6.7: Animation and events tab

In total I have 15 screens under 6 sections. Which are Display, Controller Parameters, Preset, Mlc Function, Configuration, Simulation. Two example screens are given below.

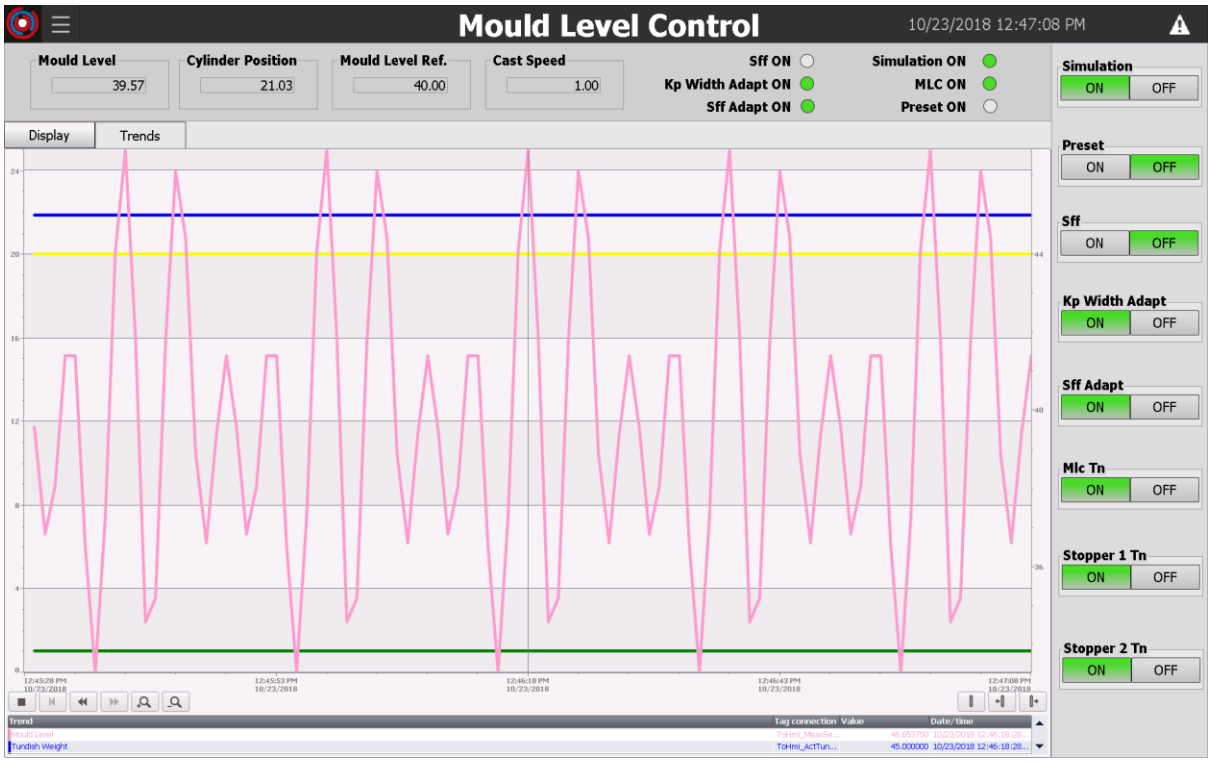


Figure 9.6.8: Trend view of HMI

Figure 9.6.9: Controller parameters

11 PDA System and Testing

There are various ways to visualize how the program is running and how the inputs and outputs are responding, during the execution.

Can be done with the HMI trends, but the data acquisition is very slow in case of HMI trends so it is not a very good choice for precise analysis.

11.1 Traces

There is a new feature in TIA Portal which was absent in the previous versions called traces. Where, each value at every scan cycle can be viewed without loss of any data.

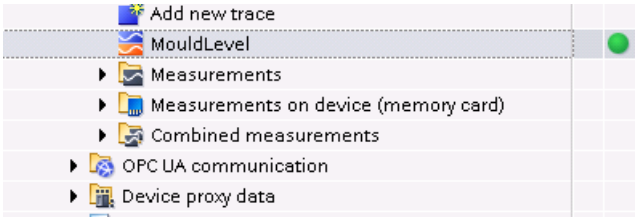


Figure 11.1.1: Adding a new trace

A trace can be easily configured. with the add new trace option. Signals can be added by dragging a DB or any tag to the configutation tab in the traces page. Colours and combination of the signals can be configured in the trace view.

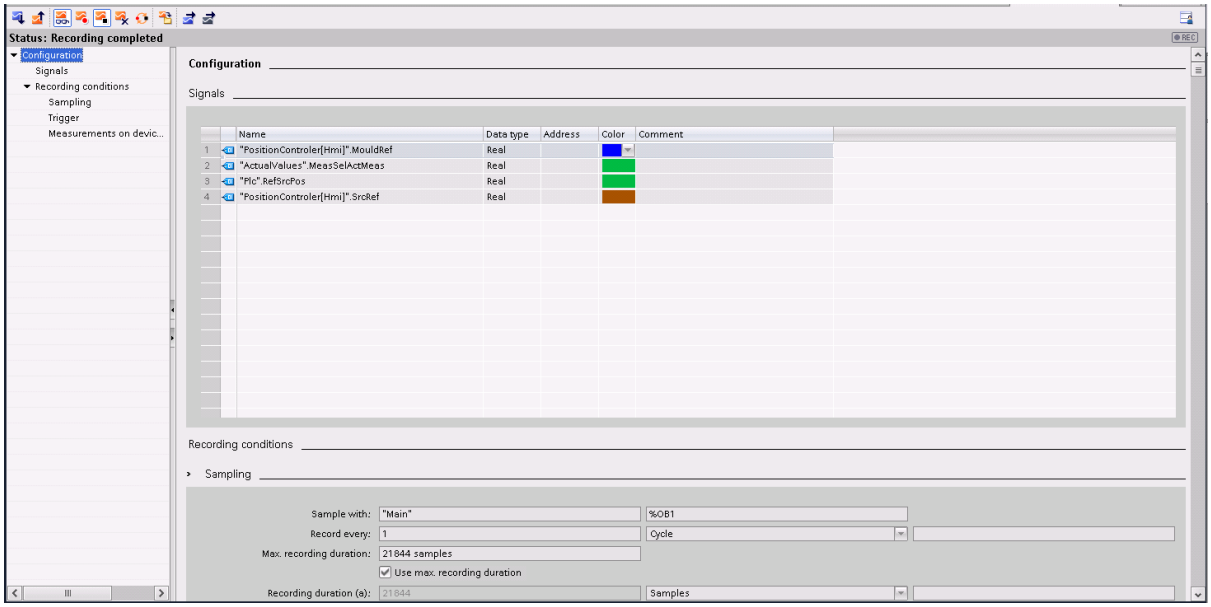


Figure 11.1.2: configuring traces

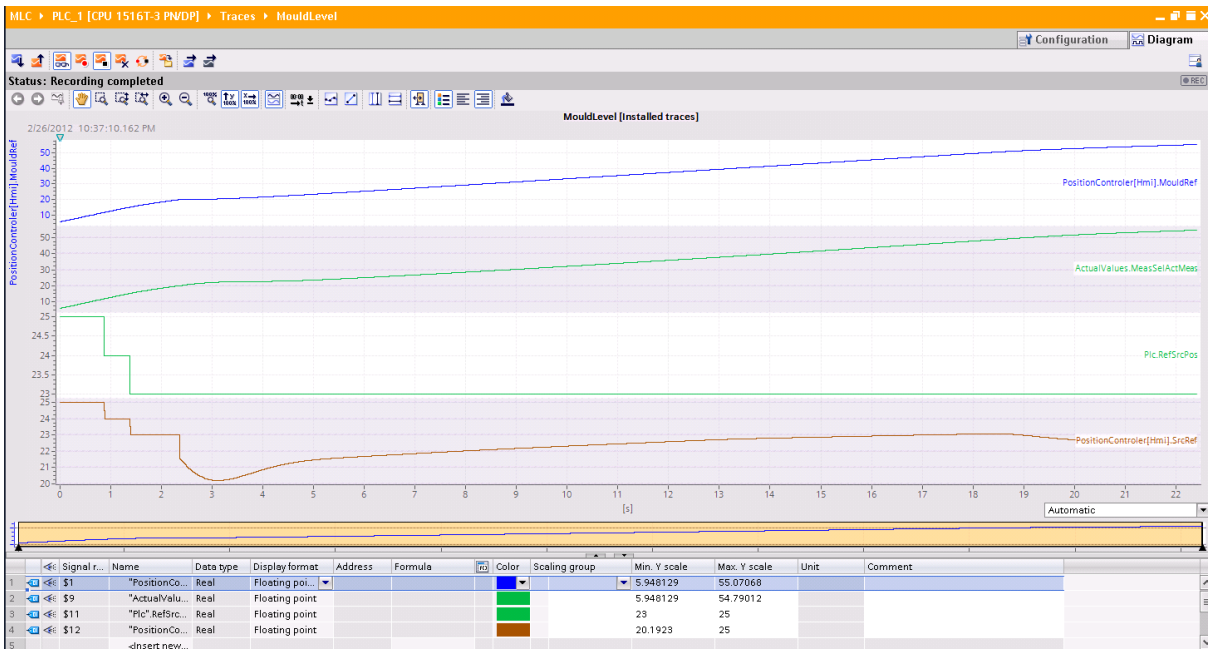


Figure 11.1.3: Trace of Mould level control

It works very precisely but there is one disadvantage associated to it, that it can only record up to a maximum 21844 samples. It has an option of automatically repeat recording. But the previous data is lost. Usually in an actual system, process data accusation software is used. In this, IBA Module is used for PDA System.

11.2 IBA System

There is a dedicated hardware on which the software runs. They are used, as they are robust and have various interfaces. They have measurement of cycle time in (µs) microseconds.

In my case, I did not use the hardware, but I used the same software for analysing and testing the program.

It takes only few steps to setup the software. The TIA project must be saved and the whole project must be copied into the hardware. After that an Iba client must be created.

It can be done by using the I/O manager in the software. IP address has to be assigned. The connection can tested in the connection tab. The connection type should be TCP/IP S7-1X00.

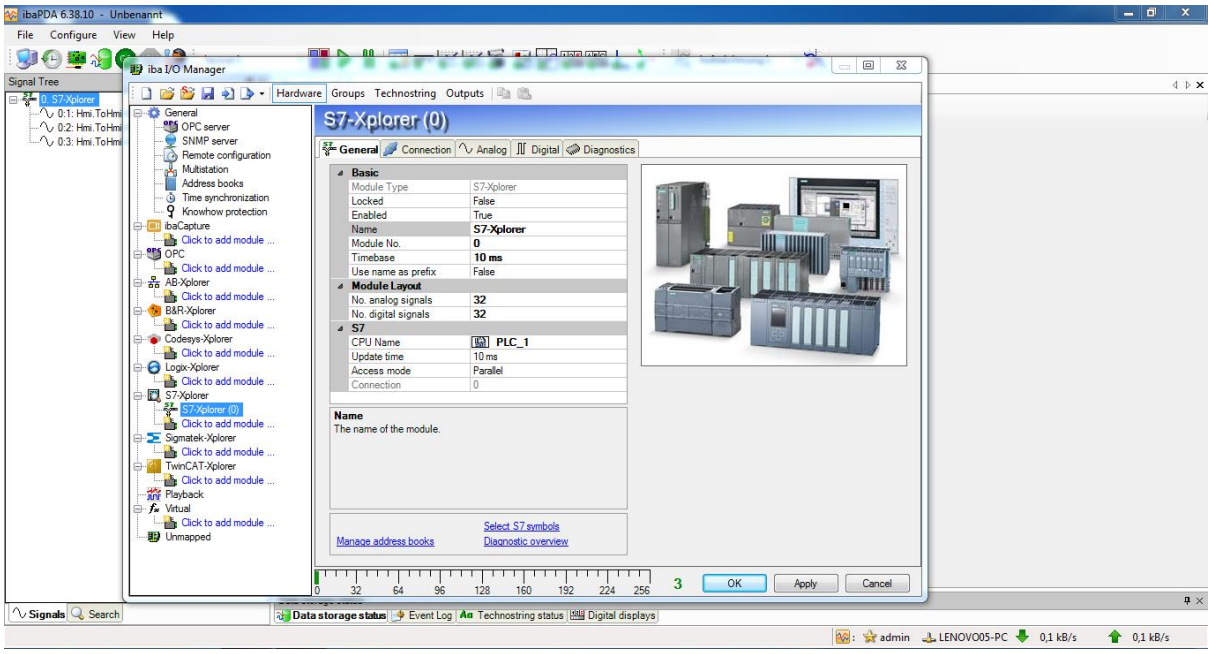


Figure 11.2.1: Configuration tab

A new address book can be created by giving a path to the project file. One must be careful if the version is not compatible the file may not be visible.

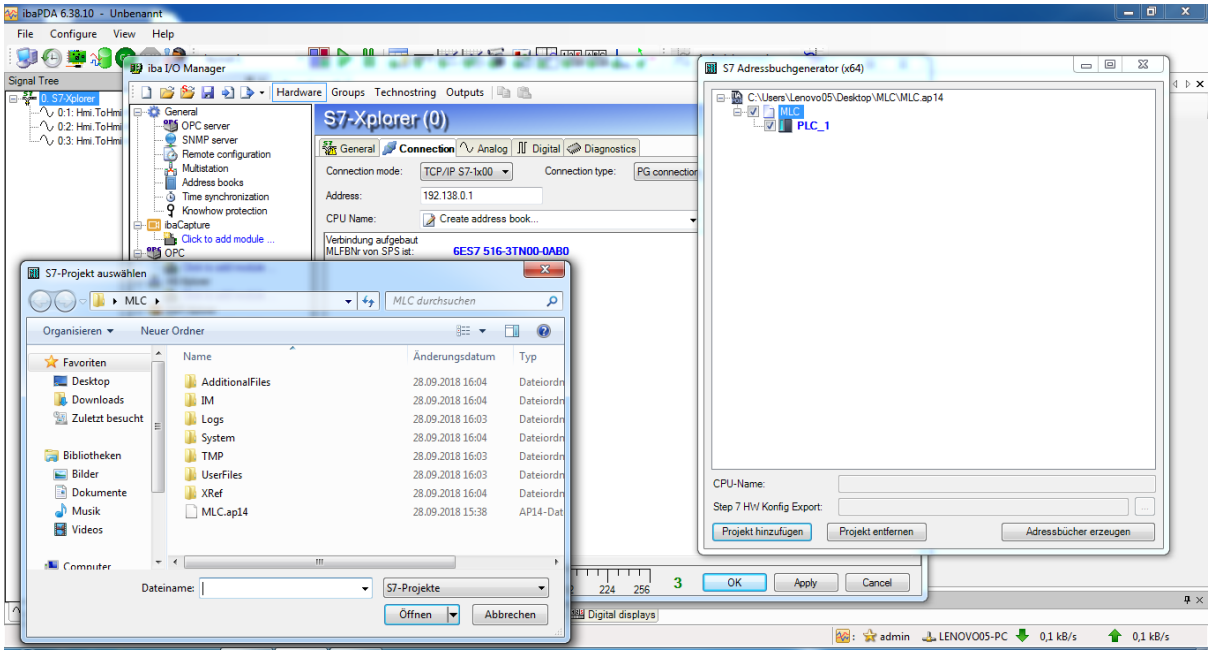


Figure 11.2.2: Creating address book

In the analog or digital tab signals according to their type can be added which is desired for analysis.

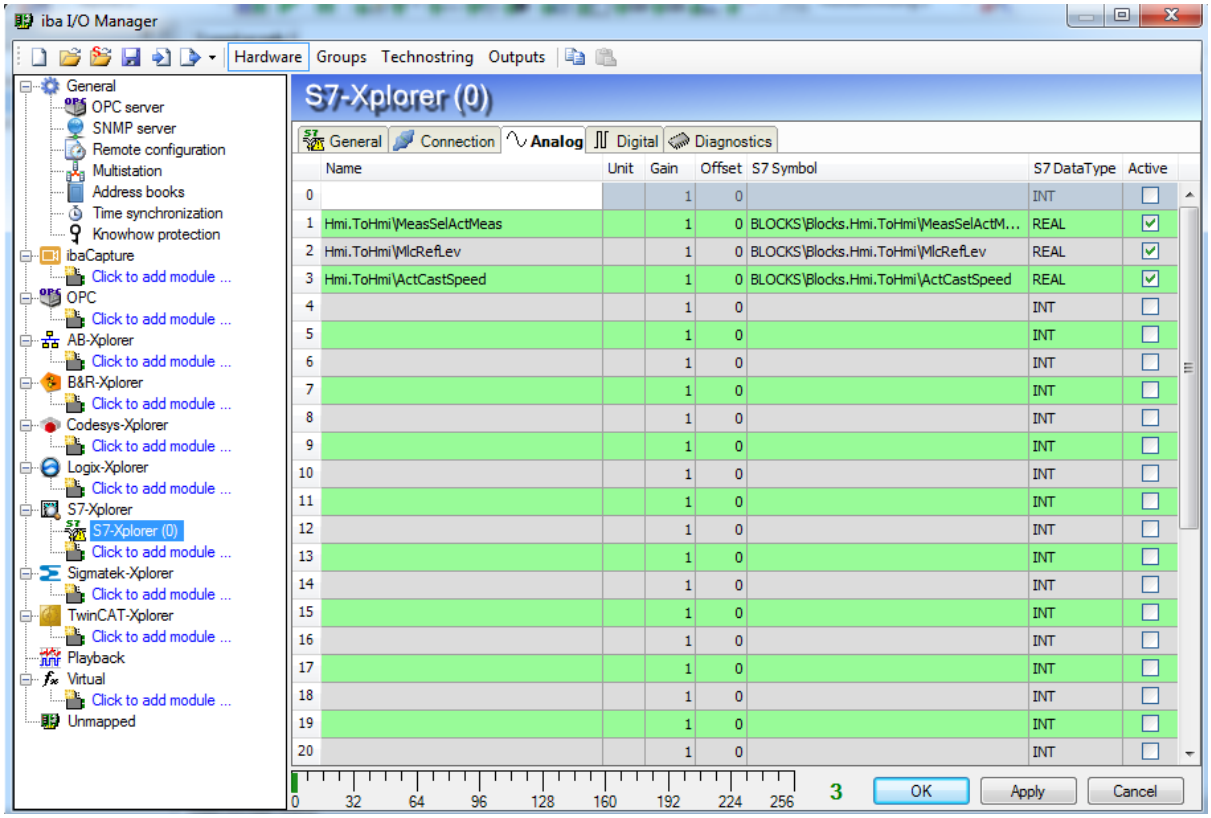


Figure 11.2.3: Adding signals

After the setup is complete signals can be dragged and dropped for analysis.

12 Test results

To test the functions I used realistic parameters, which were actually used in the field. I referred to some of the actual data from running plants.

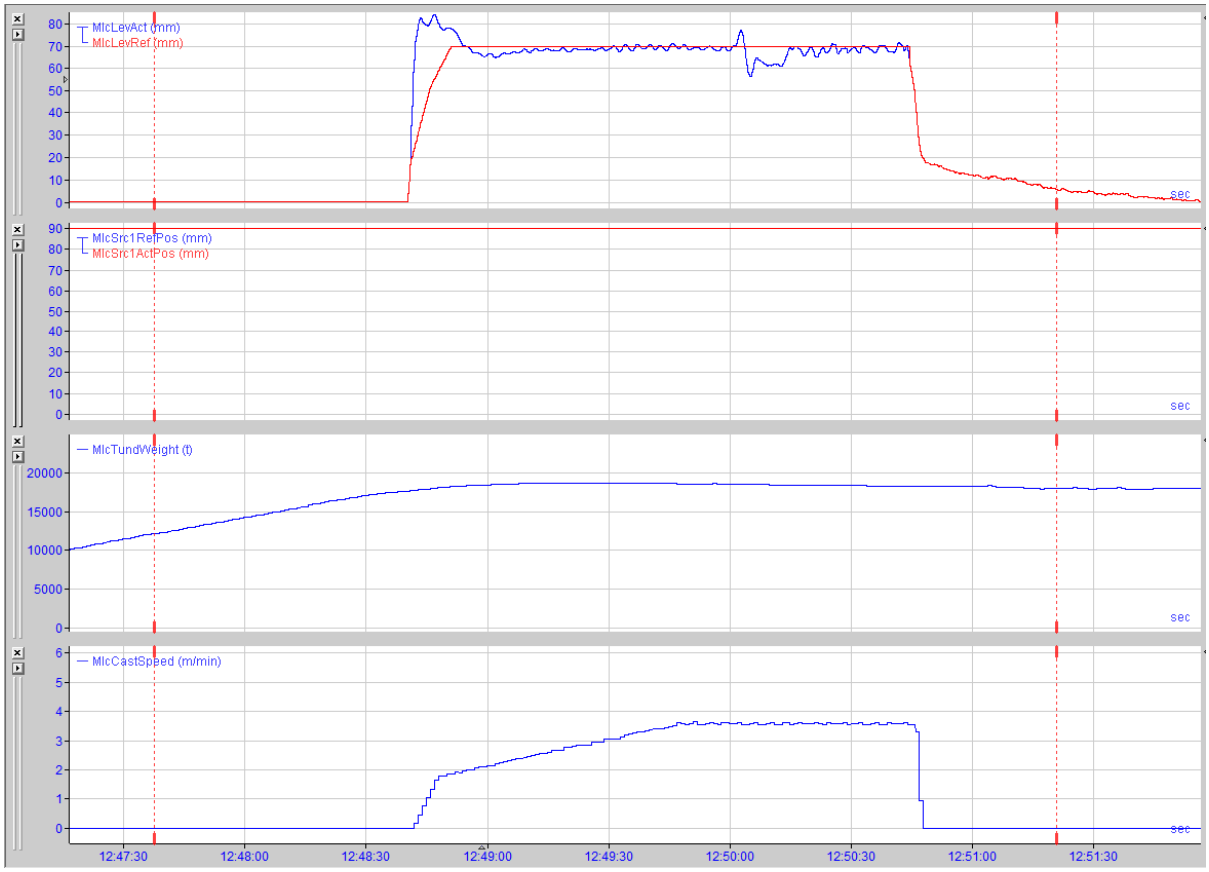


Figure 11.2.1: Example of real data showing Cast speed, mould level ref and act, tundish weight and stopper reference and actual position

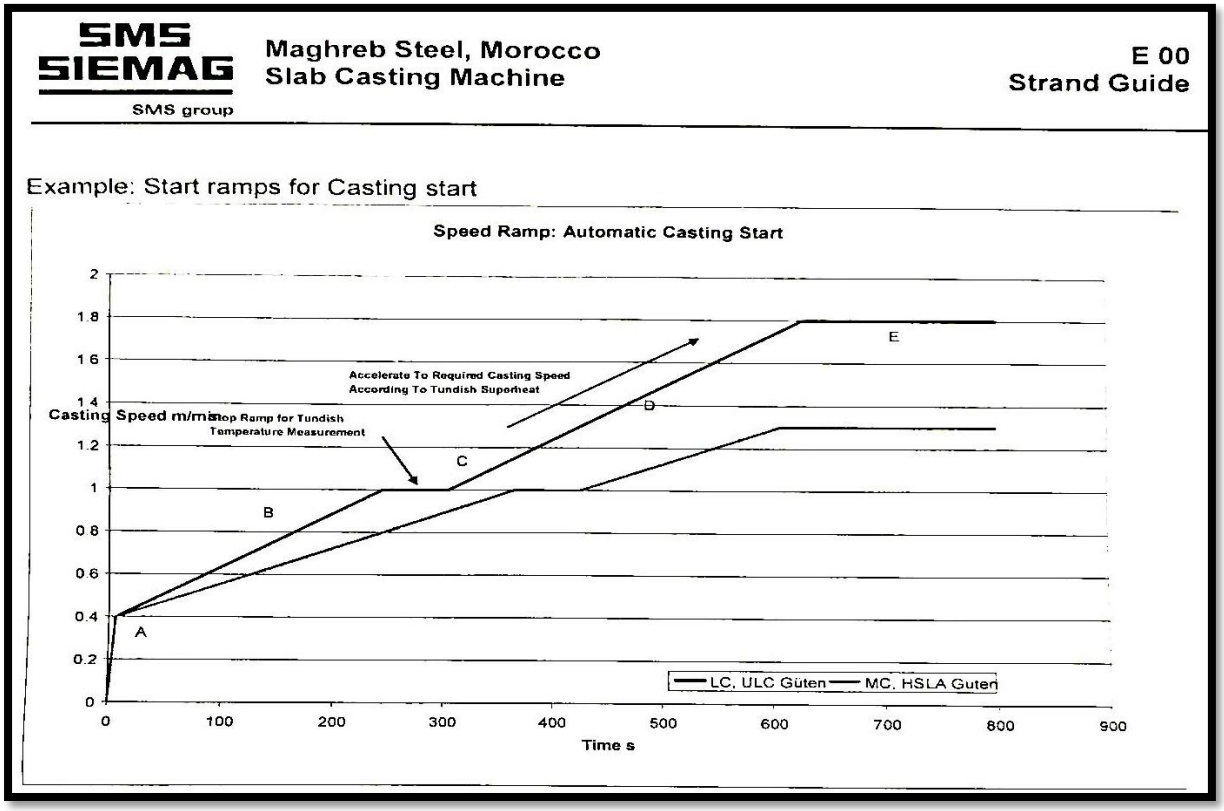


Figure 11.2.2: Example of cast start Casting speed ramp

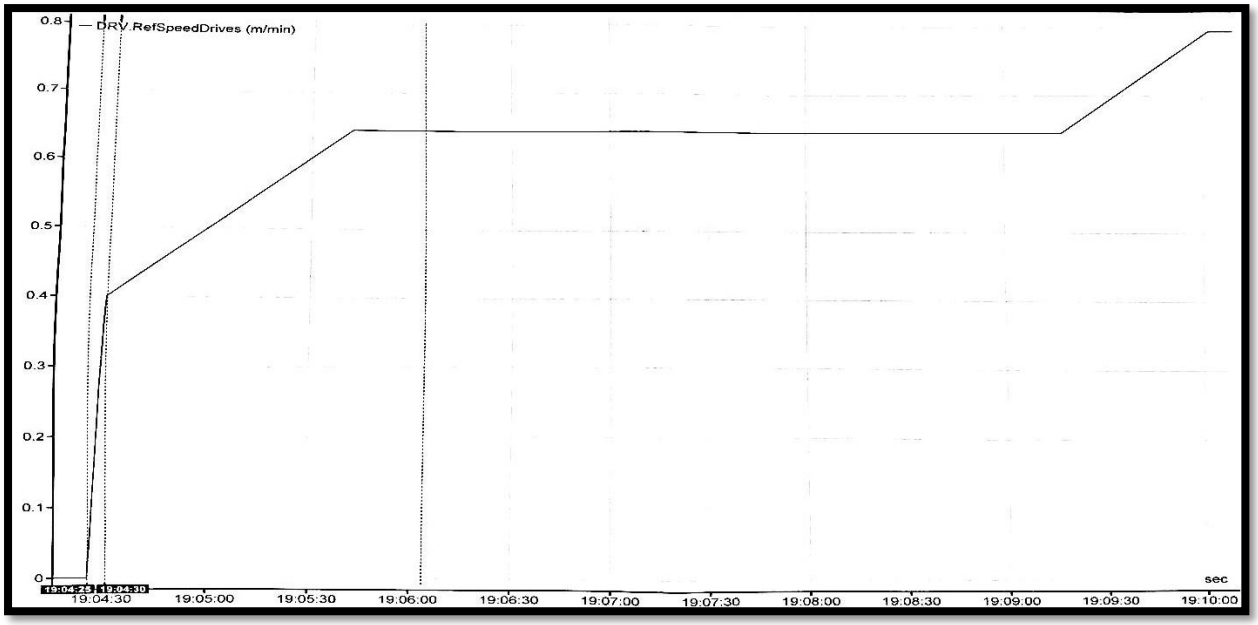


Figure 11.2.3: Example of cast start Casting speed ramp

It is not possible to include test results for all the cast start situations. I have included results for one cast start situation and only for the MLC controller due to limitations of the size of the report.

Test results include the results of testing the functions in Simulink platform and in TIA Portal platform which is analyzed in the IBA system.

Note: Cast speed is different in both the platforms in the examples below.

MLC controller	Input Parameters
Mould level controller (PID)	The output of the mould level controller is fed to the Stopper rod simulation. And the stopper position is fed to the Mould level simulation. The mould level from the simulation is a feedback to the mould level controller. The gain parameters for the MLC controller comes from the steel grade selector which currently are. KpSteelGroup = 0.6, TnSteelGroup = 3.5s, TvSteelGroup = 0.3s, T1SteelGroup = 0.06s, KdOn = true, KpFUZMlc = 1, MlcTnOn = true .Mould level ref. = 50 mm

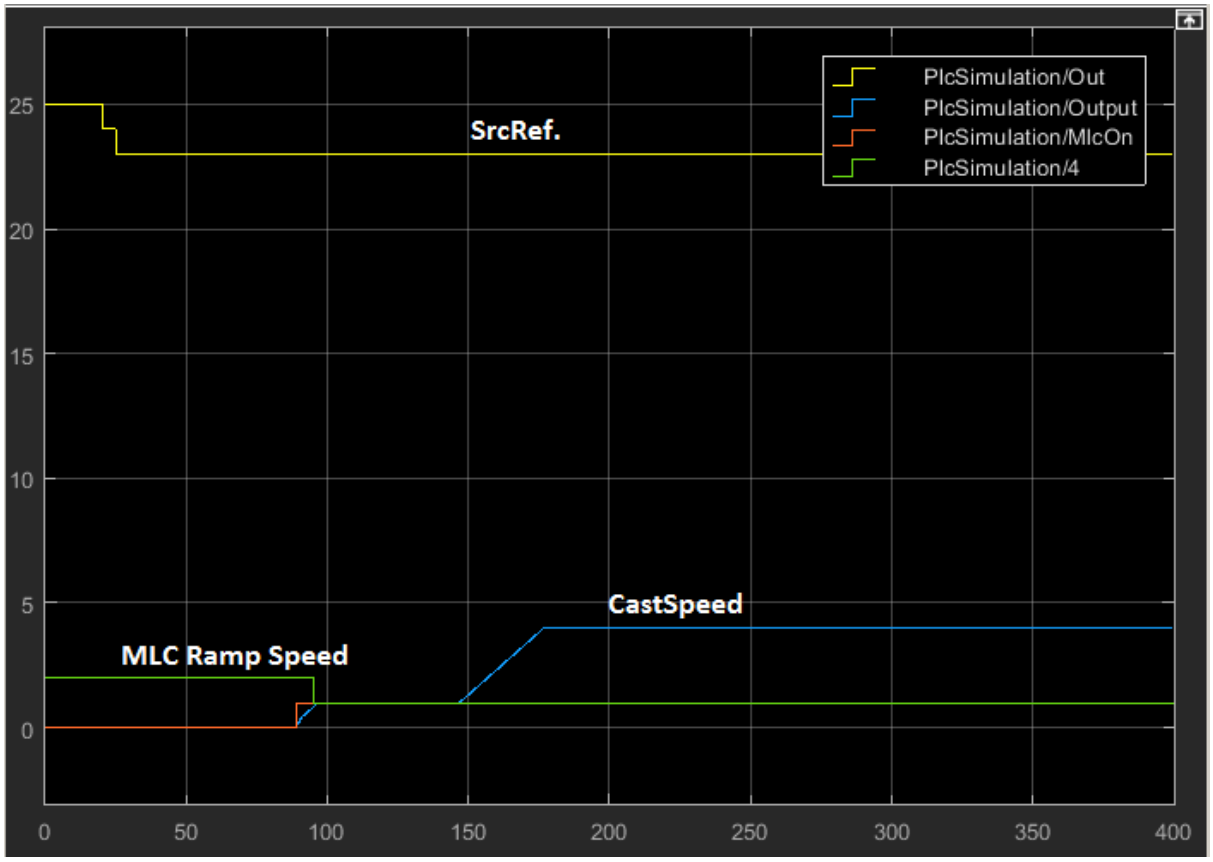


Figure 11.2.4: Simulink scope view of Inputs to mould level controller

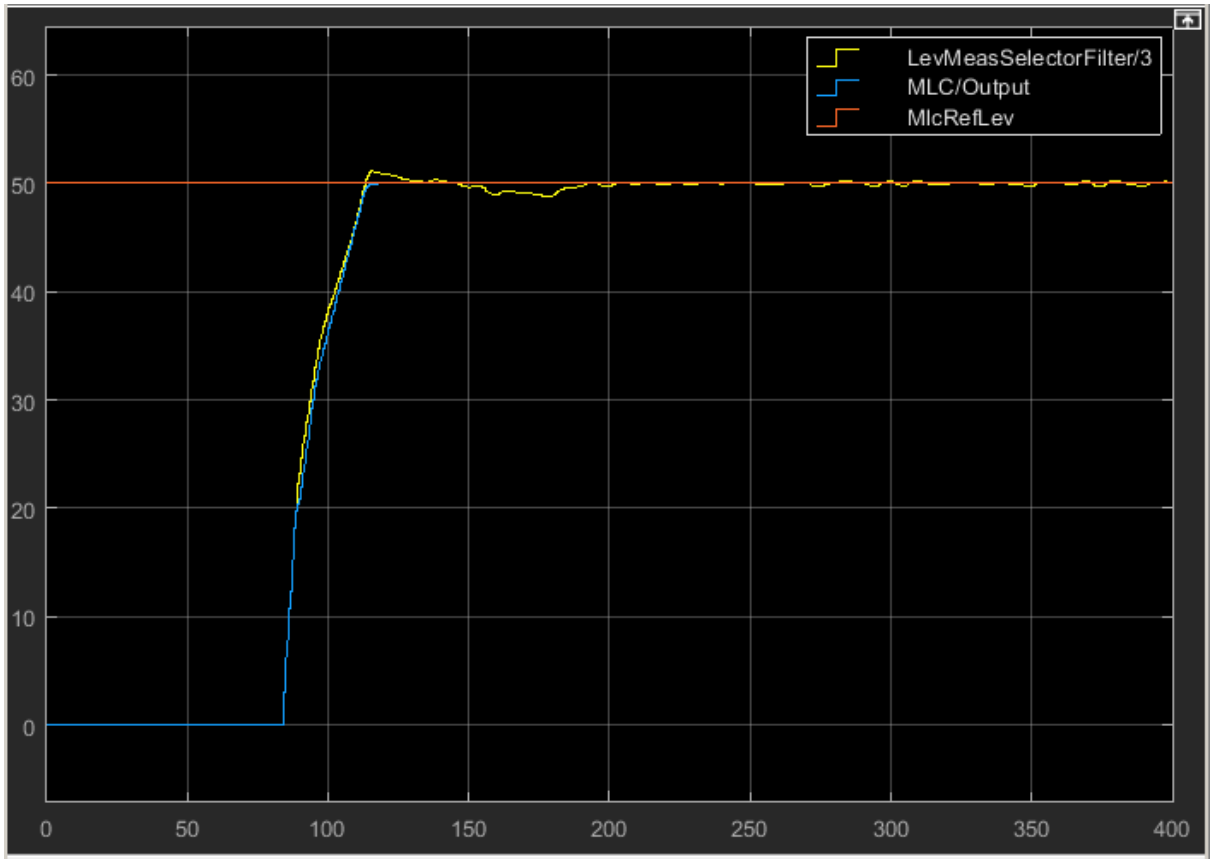


Figure 11.2.5: Simulink scope view of Mould level and reference

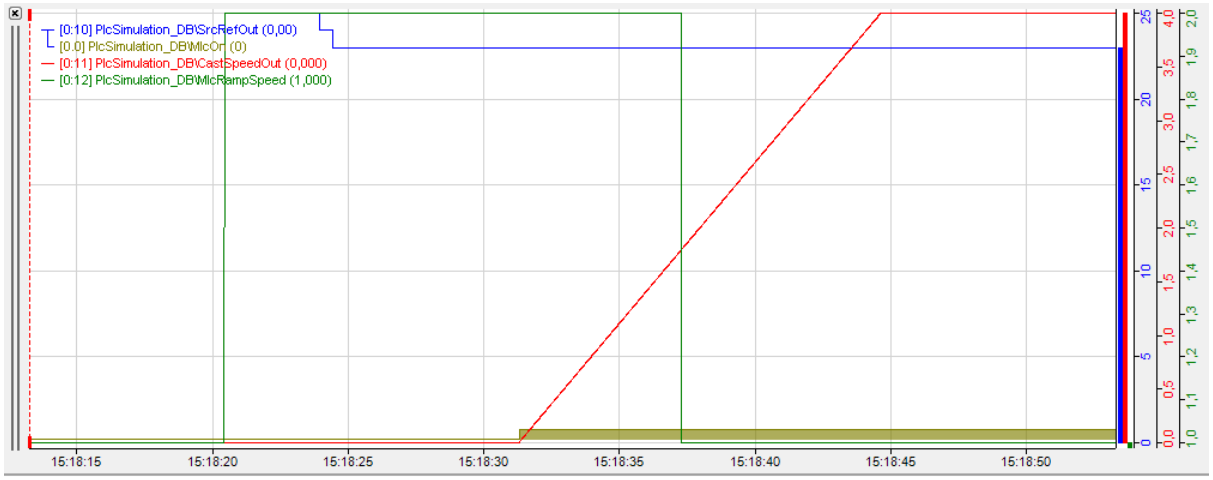


Figure 11.2.6: PLC inputs in IBA PDA

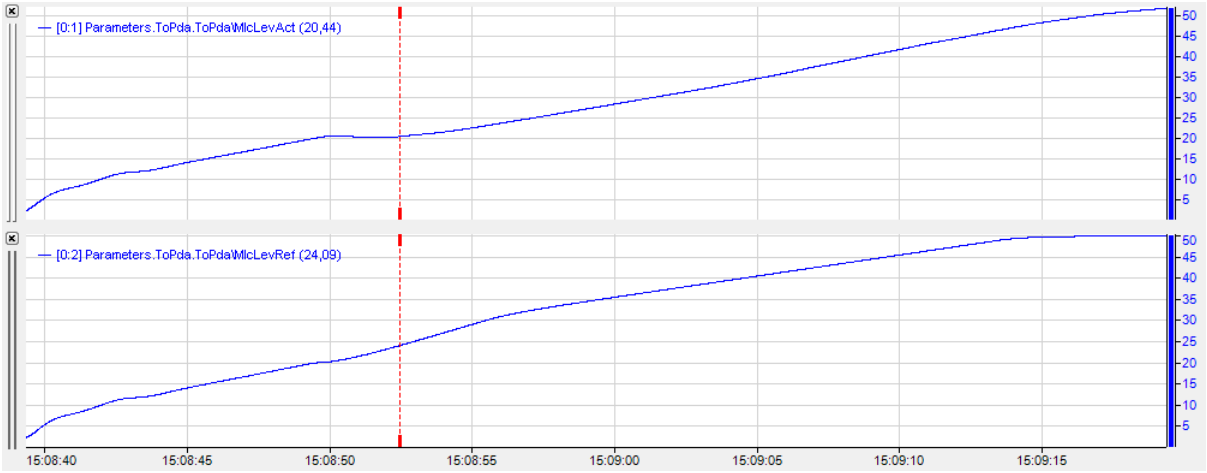


Figure 11.2.7: Mould level and reference

Function's Checklist					
Function	**Status**	**Matlab Programmed**	**Matlab Tested**	**PLC Programmed**	**PLC Tested**
Clogging detection	Done	Done	Done	Done	Done
Statistic function	Done	Done	Done	Done	Done
Preset function	Done	Done	Done	Done	Done
Kp Adaption funktion (mould width)	Done	Done	Done	Done	Done
Sff Adaption funktion (mould width)	Done	Done	Done	Done	Done
Mould Level Simulation	Done	Done	Done	Done	Done
Plc Simulation	Done	Done	Done	Done	Done
Steel Grade Selector	Done	Done	Done	Done	Done
StopperCharacteristics	Done	Done	Done	Done	Done
Stopper Rod Simulation	Done	Done	Done	Done	Done
Mould Level Controller	Done	Done	Done	Done	Done
Stopper Rod Vibration	Done	Done	Done	Done	Done
Level Measurement Selector/Filter	Done	Done	Done	Done	Done

Figure 11.2.8: Checklist of functions

13 Energy analysis

The above-mentioned system was of the hydraulic drive unit. But the stopper rod can be driven by a hydraulic drive or an electric drive. The choice of this is dependent on the plant requirements and the decision of the customer whether a hydraulic system is required, or an electrical system is required.

Both systems have their own advantages and disadvantages.

For this project I will make a comparison of the energy consumption patterns of both the system and make a conclusion which system consumes what amount of energy.

13.1 Electric drive unit

The only difference here is, instead of the hydraulic cylinder. It is driven by an electric motor, which is driven by (VVFD variable voltage variable frequency drives).

The motor used is a Synchronous Servomotors.

A synchronous motor is an AC motor. When it is in steady state, the rotation of the shaft is synchronized with the frequency of the supply current, the rotation period is equal to an integral number of AC cycles.

The motor used here is 6SM 56L-3000 motor from KOLLMORGEN Siedel.

These are brushless DC motors for demanding servo applications. When these motors are used with digital servo-amplifiers, they become suitable for complex positioning tasks such as in industrial robots, machine tools, transfer lines etc. As dynamics and stability is a high requirement the servomotors contain permeant magnets in the rotor. Neodymium-iron-boron magnetic material are made up of rare earth material which are used in these motors which make this possible. The servo-amplifier drives the three-phase winding. These windings are integrated into the stator. As the commutation is performed electronically by the servo-amplifier, therefore the motor does not have any brushes.

The technical details about the motor are listed in the figure below.

Data	Symb	Dim	6SM 45S-3000	6SM 45M-3000	6SM 45L-3000	6SM 56S-3000	6SM 56M-3000	6SM 56L-3000	6SM 71K-3000	6SM 71S-3000	6SM 71M-3000	6SM 100K-3000	6SM 100S-3000	6SM 100M-3000	6SM 100L-3000
Standstill torque	M_0	Nm	**0,85**	**1,7**	**3,2**	**3,8**	**7,0**	**10,0**	**10,5**	**16,5**	**22,0**	**25,0**	**36,0**	**46,0**	**57,0**
Standstill current	I_{0rms}	A	**1,3**	**1,3**	**2,4**	**2,8**	**4,8**	**7,6**	**8,0**	**12,3**	**15,6**	**18,8**	**26,7**	**35,0**	**42,0**
Rated speed	n_n	min^{-1}	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000	3000
Torque constant	K_{Trms}	Nm/A	0,68	1,36	1,36	1,33	1,45	1,32	1,31	1,35	1,41	1,33	1,35	1,32	1,35
Voltage constant	K_E	mVmin	58	116	116	114	124	113	112	115	121	114	116	113	115
Mains supply voltage	U_n	V							400						
Rated torque at n_n	M_n	Nm	0,8	1,6	2,9	3,6	6,4	8,4	9,5	13,4	16,3	19,9	24,6	27,1	28,0
Rated current	I_n	A	1,4	1,3	2,3	2,9	4,7	6,7	7,6	10,5	12,2	15,7	19,2	21,8	22,5
Rated power	P_n	kW	0,25	0,5	0,91	1,13	2,0	2,6	3,0	4,2	5,1	6,2	7,7	8,5	8,8
Peak current	I_{0max}	A	4,7	5,6	9,6	12,4	19,6	28,3	32,0	44,0	51,0	72,0	113,0	150,0	180,0
Motor pole no.	p_{Mot}	-							6						
Resolver pole no.	p_{Res}	-							2						
Winding resistance Ph-Ph	R_{20}	Ω	25,4	34,0	11,9	9,4	4,0	1,8	1,65	0,8	0,57	0,46	0,22	0,16	0,12
Winding inductance Ph-Ph	L	mH	54,0	99,0	47,0	54,0	30,0	15,8	19,6	12,0	9,0	10,5	7,0	5,0	4,0
Insulation class	-	-							F, DIN 57530						
Switch point therm contact	-	°C							45 ±5						
Design	-	-							IM B5 (V1, V3), DIN 42950						
Rotor moment of inertia	J	kgcm²	1,5	2,1	3,4	5,2	10,0	15,0	20,0	31,0	42,0	74,0	108,0	141,0	175,0
Static friction torque	M_R	Nm	0,127	0,131	0,14	0,154	0,18	0,208	0,23	0,28	0,334	0,4	0,49	0,58	0,67
Radial load permitted at shaft end with n_n	F_R	N	370			530			700			1050			
Axial load permitted at shaft end with n_n	F_A	N	120			170			230			350			
Tolerance class flange	-	-							N, DIN42955						
Vibration class	-	-							N, DIN ISO 2373						
Thermal time constant	t_{TH}	min	15	20	20	20	20	20	25	30	35	32	40	41	46
Weight standard	G	kg	4,5	5,5	6,5	6,1	8,0	10,3	11,7	15,8	20,0	26,0	33,0	40,0	49,0
Order number standard	-	-	81681	81684	81752	81682	81683	81753	81679	81754	81680	84855	84856	84876	84874
EMV-RES connector	-	-							12 poles, round						
RES cable, shielded	-	mm^2							4x2x0,25						
Power connection	-	stud			M4					M6				M8	
Motor cable, shielded	-	mm^2			4x1 or 4x1,5					4x2,5		4x4	4x6	4x10	4x16
max. ∅ of motor cable	-	mm			15					20			28		
max. ∅ of braking cable	-	mm			8						12,5				
Holding torque	M_{BR}	Nm		6,5			12			20			60		
Operating voltage	U_{BR}	V=							24 +6/-10%						
electrical power	P_{BR}	W		16			18			22			50		
Moment of inertia	J_{BR}	kgcm²		1,06			3,6			9,5			57,5		
Release delay time	t_{BRH}	ms		10 - 30			30 - 60			20 - 60			70 - 160		
Application delay time	t_{BRL}	ms		5 - 15			10 - 20			10 - 35			30 - 60		
Weight of the brake	G_{BR}	kg		0,6			1,1			1,9			5,4		
Motor cable with brake	-	mm^2			4x1 + 2x0,75 or 4x1,5 + 2x0,75					4x2,5 + 2x1			-		
Separate braking cable	-	mm^2							4x1,5 or 4x2,5						
Order number with -G-	-	-	81870	81869	81871	81868	81867	81866	81865	81864	81863	84699	84857	84877	84875

Figure 13.1.1: Motor details

(Kollmorgen Seidel, 2000)

The Current data that I have used is from the IBA system which made it easier for analysis. In the figure below, I have mould level, stopper position and the current data arranged together.

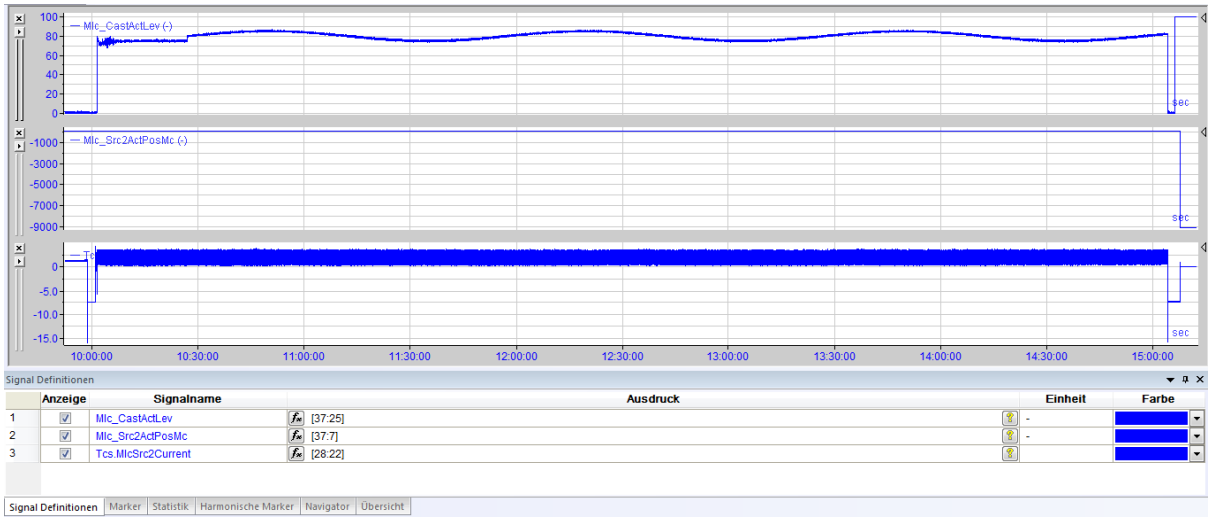


Figure 13.1.2: Continuous time data of Mould level Stopper position and Current drawn

To calculate the energy, current data is very useful.

The figure below shows the current drawn during cast start.

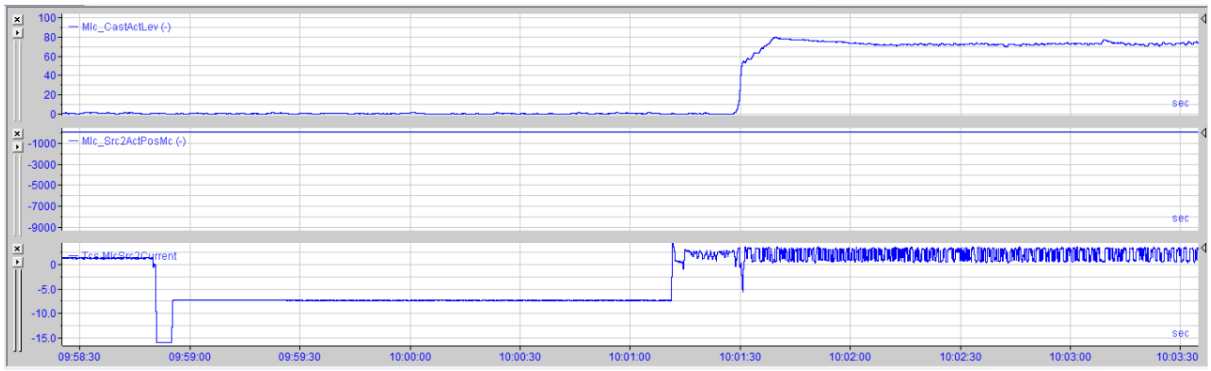


Figure 13.1.3: Data of Mould level, Stopper position and Current drawn during cast start

The figure below shows the current drawn during casting.

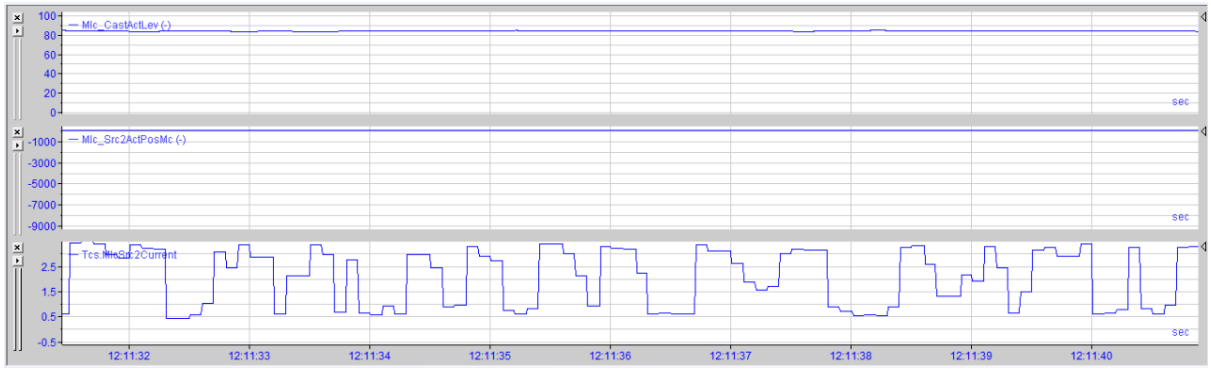
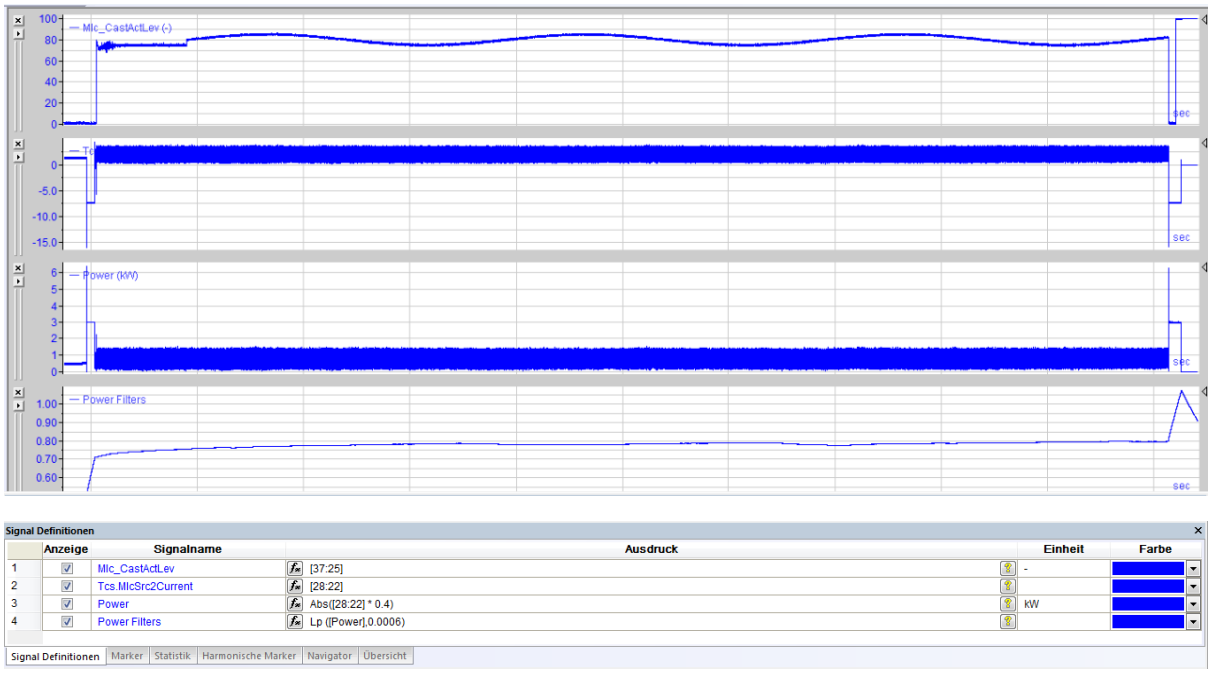


Figure 13.1.4: Data of Mould level, Stopper position and Current drawn during casting

It can be observed that from the data that the cast start duration barely took 15 seconds. And in a modern continuous casting machine casting machine cast start is a onetime process. It only stops

when there is a call for maintenance or there is an emergency break down. It can be observed that there is not much difference in the current drawn during the cast start and the casting period. And during the casting period the current drawn falls in a range.

In the IBA system it is very easy to make calculations using the available data. Therefore, to the current, the rated voltage was multiplied to get the power data. And since the power data obtained is fluctuating and is very noisy it was passed through a filter to get a steady power value.



Signal Definitionen

	Anzeige	Signalname	Ausdruck	Einheit	Farbe
1	☑	Mlc_CastActLev	[37:25]	-	
2	☑	Tcs.MlcSrc2Current	[28:22]		
3	☑	Power	Abs([28:22] * 0.4)	kW	
4	☑	Power Filters	Lp ([Power],0.0006)		

Signal Definitionen | Marker | Statistik | Harmonische Marker | Navigator | Übersicht

Figure 13.1.5: Calculation of power

If we ignore the power during the cast start which is very insignificant compared to the casting time. The power is 0.78 KW.

To calculate the energy consumption, I need time duration.

For comparison purpose I considered time duration of 1 hour. Therefore, the total energy consumed in one hour is 0.78 KW X 1hr which equals 0.78 kWh of energy.

13.2 Hydraulic drive unit

The hydraulic MLC has a cylinder which drives the stopper rod. It has a dedicated hydraulic pump station. A dedicated pump station can provide a dedicated fluid pressure to the cylinder which will not be influenced by other operations. The layout of the pump station can be seen in the figure below.

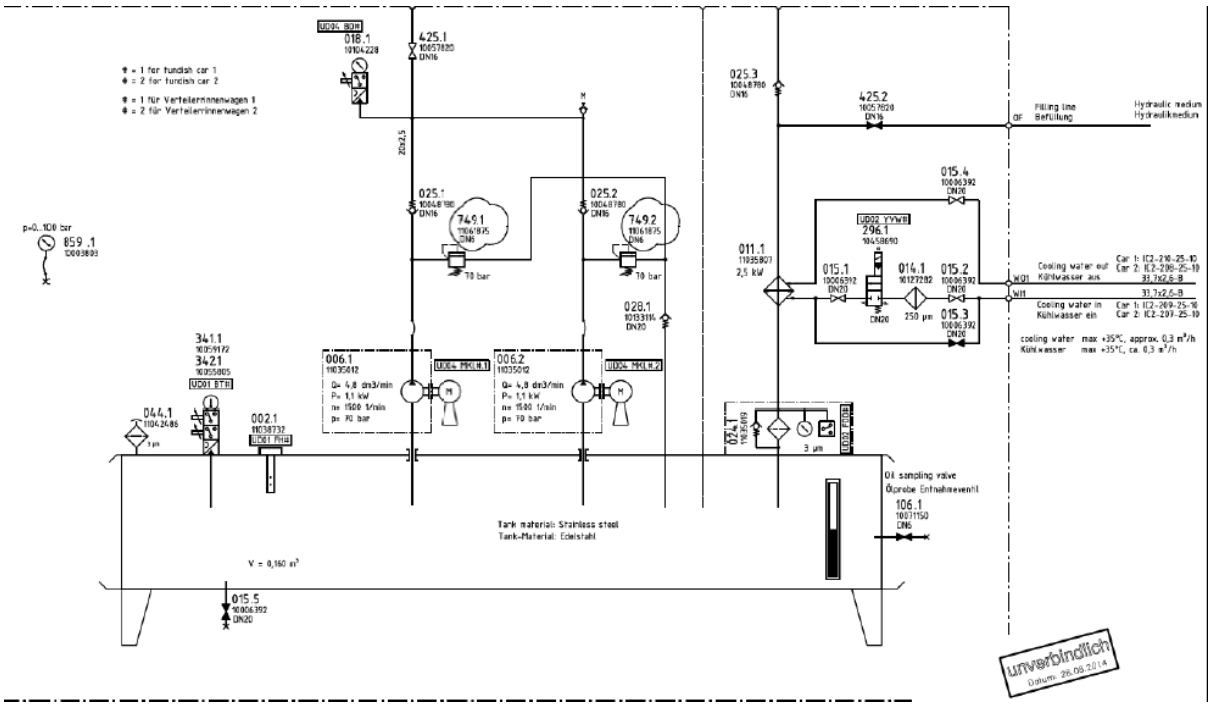


Figure 13.2.1: Hydraulic pump station MLC

It has two dedicated pressure pumps which provide constant oil pressure. The operating pressure of the system is 70 bars. The rated oil flow is 4.8 cubic meters/minute. The motors rated RPM is 1500 rotations/minute.

The power rating of the motor is 1.1 KW. It runs on a 400 V AC supply.

It has two motors because of safety reasons. One is a backup motor because if one fails the other takes over.

Usually the pump station has a circulation pump which circulates the oil through a cooler. But in this case, it is cooled by a water cooler. It can be observed that the return line passes through a cooler. There is an external input for the cold water and an output line for the hot water to leave. The cooling water circulation depends on the customer. It can be a dedicated water pump for water circulation or the cooling water can be coming from a cold-water line which is pumped by a decentralised pump pumping cooling water to the whole plant. This only makes the calculation of energy complex.

The figure below illustrates the current consumption in percentage. It is the current is percentage of the nominal current of the motor.

Nominal current is the full load current rating of the motor. It is the current drawn when the motor is working at full load.

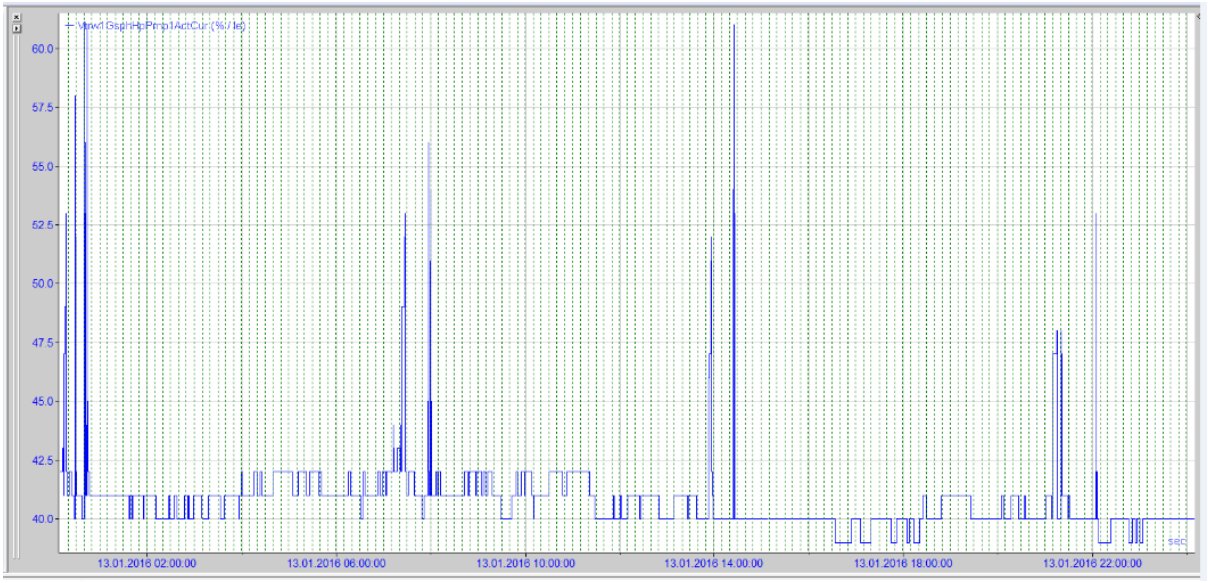


Figure 13.2.2: Current drawn by the motor in percentage of the nominal current of the motor

From the data it can be observed that current drawn is 40% to 42% of the nominal current. For convenience we can consider it 41%.

Since the nominal current is 1.94 amperes. The current drawn is 41% of 1.94 amperes.

Therefore, the current drawn during casting period is 0.7954 amperes.

The energy consumption after one hour of operation would be 0.32 kWh. This is the energy consumption alone by the pressure pump. But the total energy consumed is the sum of energy consumed by the pressure pump and the energy consumed to circulate the cooling water. Since that is entirely costumer dependent, I must make assumptions. If I assume the energy consumed to circulate the cooling water is ¼ of the energy consumed in the pressure pump. Then the total energy consumed equals 0.4 kWh.

13.3 Noise generation

A load which draws current from the supply which is proportional to the applied voltage is called linear load. Loads such as simple resistive loads, incandescent lamps etc. are linear loads.

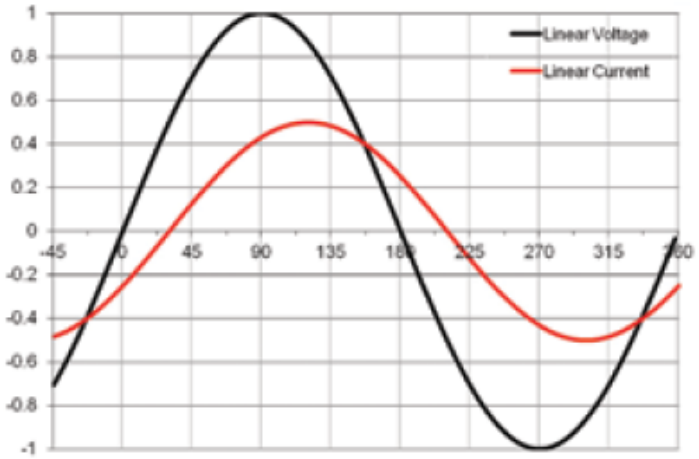


Figure 13.3.1: Voltage and current of a linear load

A load which draws a non-linear current when supplied by a sinusoidal voltage is called a non-linear load. Loads such as variable frequency drives, computers etc. are non-linear loads.

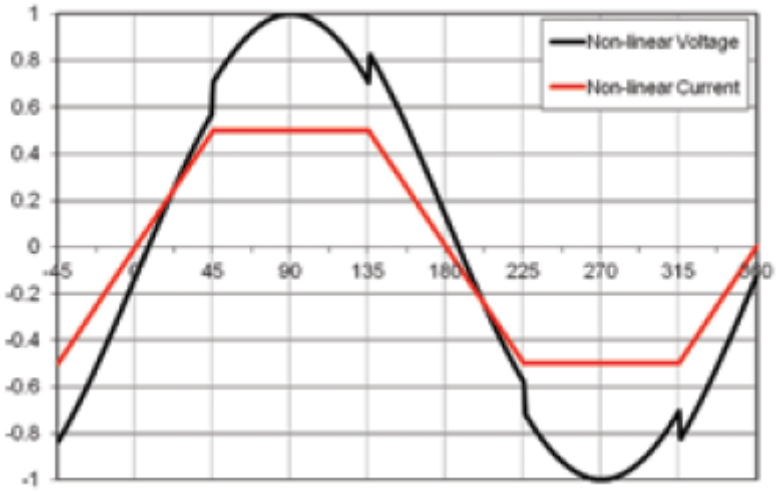


Figure 13.3.2: Voltage and current of a non-linear load

These non-sinusoidal currents contain harmonic currents that interact with the impedance of the power distribution system to create voltage distortion that can affect both the distribution system equipment and the loads connected to it.

13.3.1 Harmonics

According to IEEE 519-1992 a sinusoidal component of a periodic wave or quantity which can be voltage or current which has a frequency that is an integral multiple of the fundamental frequency is called harmonics.

Static power converters are the largest non-linear loads in a power system. They use power semiconductor devices to convert power into various forms. One of the major application of power converters is variable frequency drives used to control a motor. These power converters draw non-linear currents. These non-linear currents distort the supply voltage waveform at the point of common coupling (PCC). The PCC is defined as a point between the system owner or operator and a user. The PCC is usually considered as the point in the power system which is closest point to the user. This is the point from where the system owner or operator can provide service to another user. The figure below shows the power distribution system with the point of common coupling (PCC). It is the single line representation. The source/system voltage (vs) is assumed to be purely sinusoidal and the system/source impedance is represented by an inductance Ls.

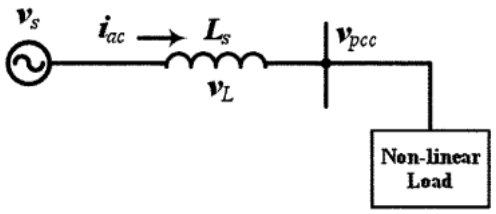


Figure 13.3.3: Single line diagram of power distribution system

The voltage at the PCC, vPCC can be calculated by subtracting the voltage drop (vL) across the system impedance due to the flow of non-linear current iac.

vPCC = (vs-vL) = {vs – Ls(d(iac)/dt}

The non-sinusoidal voltages and currents can be divided into sinusoidal components. The fundamental frequency which is the system frequency is the first harmonics. It can be 50 hertz or 60 hertz depending on the country. The harmonic components of the voltages and currents are integer multiples of the fundamental frequency. For example, on a supply of 60Hz, the 3rd harmonic is 3 times 60Hz = 180Hz, the 5th harmonic is 5 times 60Hz = 300Hz, and so on. When all the harmonic currents are added to the fundamental a waveform known as complex wave is formed.

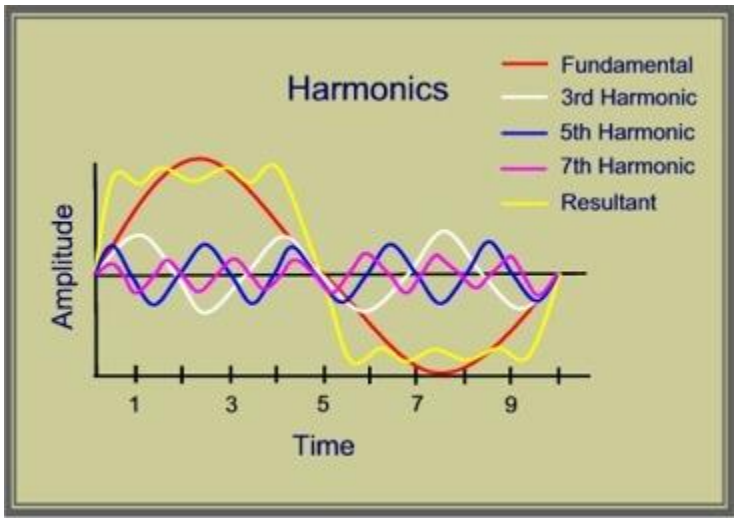


Figure 13.3.4: Figure showing the first, third, fifth harmonic and the combined distorted waveform

Harmonic spectrum and distortion factor: The harmonics produced by the semiconductor converter equipment in steady state condition of operation are called characteristic harmonics of the converter and are expressed as:

h = np ± 1

h = order of harmonics n = an integer 1, 2, 3,…. p = number of pulses per cycle

Total harmonic distortion

According to IEEE 519-1992, the THD or the total harmonic distortion is the index which is used to measure the amount of distortion in the voltage or current waveform. It is calculated as the ratio of the root-mean-square of the harmonic content to the root-mean-square value of the fundamental quantity and expressed as a percent of the fundamental.

Total demand distortion

According to IEEE 519-1992, the TDD or the total demand distortion is the index which is used to measure the total effect of distortion in the current waveform at the PCC. It is the percentage of the maximum demand current at the PCC. It is calculated as the ratio of root mean square of the harmonic content, to the root-mean-square of the maximum demand load current at the PCC. This is expressed as a percentage of maximum demand load current.

13.3.2 Effects of harmonics

- Heating of generators
- Heating of transformers

- Heating of induction motors
- Heating of cables
- Premature tripping of circuit breakers and fuses
- Flickering in the lighting system
- Harmonic currents can interact with the power factor correction capacitors and damage the plant equipment
- Cables carrying harmonics can generate electromagnetic interference to the control cables carrying signals
- Conventional protection equipment relies on conventional measurement techniques or the heating effect of current. In presence of nonlinear leads overloaded cables may be undetected and the cables and busbars may catch fire.
- Conventional electrical meters may malfunction

13.3.3 Control of harmonics

IEEE 519-1992 Guidelines developed some standards for harmonics for the industries. Since harmonic current produced in the industrial workspace reflect through distribution system impedances and generate harmonic voltages on the utility distribution systems. The figure below defines how much harmonic currents an industrial user can inject onto the utility distribution system in the range 120V to 69KV.

Maximum harmonic current distortation in percent of IL Individual harmonic order (odd harmonics)						
I_{SC}/I_L	< 11	11≤h<17	17≤h<23	23≤h<35	35≤h	TDD
<20*	4.0	2.0	1.5	0.6	0.3	5.0
20<50	7.0	3.5	2.5	1.0	0.5	8.0
50<100	10.0	4.5	4.0	1.5	0.7	12.0
100<1000	12.0	5.5	5.0	2.0	1.0	15.0
>1000	15.0	7.0	6.0	2.5	1.4	20.0

*Even harmonics are limited to 25% of the odd harmonic limits above. Current distortions that result in a dc offset, e.g., half-wave converters, are not allowed. * All power generation equipment is limited to these values of current distortion, regardless of actual I_{SC}/I_L. Where: I_{SC} = maximum short-circuit current at PCC. I_L = maximum demand load current (fundamental frequency component) at PCC.*

Figure 13.3.5: Current Distortion Limits for

The figure below defines voltage distortion limits that can be reflected onto the utility distribution system.

Bus voltage at PCC	Individual voltage distortation (%)	Total voltage distortion THD (%)
69 kV and below	3.0	5.0
69.001 kV through 161 kV	1.5	2.5
161.001 kV and above	1.0	1.5
Note: High-voltage systems can have up to 2.0% THD where the cause is an HVDC terminal that will attenuate by the time it is tapped for a user.		

Figure 13.3.6: Voltage Distortion Limits

13.3.4 Methods to control harmonics

The prevalent methods to control harmonics are

- Delta-Delta and Delta-Wye Transformers
- Isolation Transformers
- Use of Reactors (Comprised of Inductors (L))
- Passive Harmonic Filters or Line Harmonic Filters (Comprised of L-C circuit)
- 12-pulse converter front end
- 18-pulse converter front end
- Active filters (electronic power factor correction)
- Active front end (comprised of several AC drive and UPS system companies to offer a low input harmonic footprint)
- Power System Design

All the above mention solutions are effective top to bottom. The one at the bottom is most affective and most expensive and the one at the top is least affective and least expensive.

The figure below illustrates different solutions in relation to the total harmonic distortion.

	Harmonic order (*h*)	5	7	11	13	17	19	23	25	*THD*
Typical values of harmonic current (% of fundamental current) of different types of front end configurations (% I_h / I_1)	6-pulse without line reactor (Stiff source)	80.0%	58.0%	18.0%	10.0%	7.0%	6.0%	5.0%	2.5%	**101.5%**
	6-pulse with 2-3% line reactor	40.0%	15.0%	5.0%	4.0%	4.0%	3.0%	2.0%	2.0%	**43.6%**
	6-pulse with 5% line reactor	32.0%	9.0%	4.0%	3.0%	3.0%	2.0%	1.5%	1.0%	**33.9%**
	6-pulse with line harmonic filter (LHF)	2.5%	2.5%	2.0%	2.0%	1.5%	1.0%	0.5%	0.5%	**4.9%**
	12-pulse	3.7%	1.2%	6.9%	3.2%	0.3%	0.2%	1.4%	1.3%	**8.8%**
	18-pulse	0.6%	0.8%	0.5%	0.4%	3.0%	2.2%	0.5%	0.3%	**3.9%**

NOTE: Relative short circuit ratio of the power system is assumed to be between 20 to 50. For a relative short circuit ratio higher than 50 (strong supply system), the values in table above will be higher.

Figure 13.3.7: Typical values of harmonic currents for different types of front ends

(Siemens - Harmonics in Power Systems, Causes Effects and Control, 2013)

13.3.5 Noise in the Hydraulic MLC

The source of power in this system is the 1.1 kW induction motor. When and induction motor is running at no load the nonlinearity of the motor depends on the saturation of the magnetic circuit. And when it is running at a load the nonlinearity increases and it also depends on the torque of the motor. But the nonlinearity is negligible compared to a variable frequency drive.

(Research gate discussion, 2015)

13.3.6 Noise in the Electric MLC

The variable frequency drive used in the electric MLC is from the maker Kollmorgen. Servostar 610 model has been used in this application.

Some of the technical description in illustrated in the figure below.

The SERVOSTAR 600 family of digital servo amplifiers

Standard version

- 6 current ratings (1.5 A -Europe only-, 3 A , 6 A , 10 A , 14 A, 20 A)
- 3 instrument widths : 70 mm for 1.5A up to 10A rated current
  100 mm for 14A rated current
  120 mm for 20A rated current
- Wide range of rated voltage (3x208V $_{-10\%}$ to 3x480V $^{+10\%}$)
- Overvoltage category III acc. to EN 61800-5-1
- Shield connection directly at the servo amplifier
- 2 analog setpoint inputs
- Integrated CANopen (default 500 kBaud), for integration into CAN bus systems and for setting parameters for several amplifiers via the PC-interface of one amplifier
- Integrated RS232, electrically isolated, integrated pulse-direction interface
- Synchronous servomotors, linear motors and asynchronous motors can be used

Electrical supply

- Directly off grounded 3 phase system,
  230V$_{-10\%}$ ... 480V$^{+10\%}$, 50 Hz,
  208V$_{-10\%}$ ... 480V$^{+10\%}$, 60 Hz
  TN-system or TT-system with grounded neutral point, max. 42,000 rms symmetrical amperes.
  Connection to other mains supply networks only with insulating transformer ⇨ p.46
- B6 rectifier bridge, directly off 3-phase earthed (grounded) supply system, integral power input filter and inrush circuit
- Single-phase supply (e.g. for setup) is possible
- Fusing: (e.g. fusible cutout) provided by the user
- Shielding: All shielding connections directly on the amplifier
- Output stage: IGBT- module with isolated current measurement
- Brake circuit: with dynamic distribution of the brake power between several amplifiers on the same DC bus link circuit. Internal brake resistor as standard, external brake resistors if required
- DC bus link voltage 260 — 900 VDC, can be switched in parallel
- Interference suppression filter for the supply input (to category 3) is integrated
- Interference suppression filter for the 24V aux. supply (to category 3) is integrated

Figure 13.3.9: Technical data of the servostar drives

(Kollmorgen-Servostar 601…620, 2015)

What is interesting for us in the predicting the noise generation is the number of pulses it uses. It can be observed in the marked area above that it uses a B6 rectifier.

Among all the line frequency three phase rectifiers, the most used is the six-pulse full bridge rectifier B6. Since we know that it a six-pulse rectifier we can put it in the six-pulse category of total harmonic distortion. (refer to Figure 13.3.10: Typical values of harmonic currents for different types of front ends explained above)

The noise generation prediction is very superficial in this case, as various measures can be taken to eliminate harmonics.

13.4 Comparison of the electric drive MLC and the Hydraulic drive MLC

Clearly the Hydraulic MLC consumes less energy compared to the Electric MLC.

Electric: 0.78 kWh

Hydraulic: 0.40 kWh

The above calculation is carried out for the time duration of 1 hour. The difference will be bigger for a greater duration of time.

Nevertheless, not only the hydraulic MLC consumes less energy, but it also induces negligible harmonics to the power system, therefore it does not impose any stress to the power system. By using hydraulic MLC, not only the cost of energy consumption can be reduced but also the additional cost to the harmonic mitigating devices, equipment and planning can be avoided.

14 Conclusion

The conclusion can be divided into two parts

1. Hardware independent Code generation: After the successful implementation and code generation of the Mould level control system from Simulink to TIA Portal environment. It can be said that hardware independent code generation is possible. But it also comes with some limitations such as incompatibility of usage of Simulink standard blocks. But it also has solutions to the problems such as usage of custom blocks, which can be used in several ways. Therefore, by smart utilization of the software, problems can be solved, and the desired results can be achieved.
2. Energy consumption comparison of the Electric MLC and the Hydraulic MLC: After carrying out the relevant calculations with the available data. It can be concluded that, not only the hydraulic MLC consumes less energy, but it also induces negligible harmonics to the power system, therefor it does not impose any stress to the power system. By using hydraulic MLC, not only the cost of energy consumption can be reduced but also the additional cost to the harmonic mitigating devices, equipment and planning can be avoided.

15 List of codes

Code 1: Ramp Function 27
Code 2: Sine Wave Generator 28
Code 3: Saw tooth Generator 28
Code 4: Triangle Generator 28
Code 5: Filter 29
Code 6: Discrete Derivative 29
Code 7: Discrete Integrator 29
Code 8: Draft Disturbance 30
Code 9: Noise Generator 30
Code 10: Creating a Library 31
Code 11: Steel Grade Selector 33

16 List of Figures

Figure 5.1.1 : CCM overview 9
Figure 5.2.1: Structure of MLC 11
Figure 5.2.2: Example of the control structure for the MLC application 12
Figure 5.2.1: SIMATIC S7-1500 at management, control and field level 16
Figure 5.2.1: Simulink library View and example model view 20
Figure 7.1.1: Configuration Parameters dialog box 21
Figure 7.2.1: Example of fixed step and variable step 21
Figure 7.4.1: Example of an algebraic loop 22
Figure 7.4.2: Minimise algebraic loop occurrences 22
Figure 7.4.3: Example of algebraic loop elimination 22
Figure 7.5.1: Sample time in block parameters 23
Figure 7.7.1: Creating a subsystem 24
Figure 7.7.2: Creating an atomic subsystem 24
Figure 7.7.3: Checking compatibility for code generation 24
Figure 7.7.4: Choosing target IDE 25
Figure 7.7.5: Choosing target IDE 25
Figure 7.8.1: Scan cycle 26
Figure 7.8.2: Discrete steps for ramp 26
Figure 7.8.3: Block creation 27
Figure 8.1.1: Solving algebraic loops 31
Figure 8.3.1: Complete view of the entire Simulink program 32
Figure 8.3.2: Assigning parameter and data type 33
Figure 8.3.1: Totally integrated automation overview 34
Figure 9.1.1: Selection of hardware 35
Figure 9.1.2: Network view 35
Figure 9.1.3: Properties tab 36
Figure 9.1.4: Downloading 36
Figure 9.2.1: Operating system and user program 37
Figure 9.2.2: choosing OBs 38
Figure 9.2.3: Assigning parameters to cyclic interrupt OB 39
Figure 9.2.4: Adding a new FC 39
Figure 9.2.5: Adding a new FB 39
Figure 9.2.6: Adding a DB 40
Figure 9.2.7: Global DB as central data memory 40
Figure 9.3.1: SCL Code 41
Figure 9.3.2: Adding an external file 41
Figure 9.3.3: External SCL files 42
Figure 9.3.4: Exporting files 42
Figure 9.3.5: External DB file 43
Figure 9.3.6: Generating blocks from source 43
Figure 9.3.7: Generated function blocks 43
Figure 9.3.8: Calling a function block automatically generates instance DBs 44
Figure 9.4.1: Sample time generator 45
Figure 9.5.1: Static Dbs 46
Figure 9.5.2: Enable port 47
Figure 9.5.3: SSMethdType case 47

Figure 9.5.4: Updating block call....47
Figure 9.6.1: Choosing HMI hardware....50
Figure 9.6.2: Choosing a PC station....50
Figure 9.6.3: WnCC RT Advanced....51
Figure 9.6.4: Network view, HMI connected to PLC....51
Figure 9.6.5: Screens....52
Figure 9.6.6: Screens....52
Figure 9.6.7: Animation and events tab....53
Figure 9.6.8: Trend view of HMI....53
Figure 9.6.9: Controller parameters....54
Figure 11.1.1: Adding a new trace....54
Figure 11.1.2: configuring traces....55
Figure 11.1.3: Trace of Mould level control....55
Figure 11.2.1: Configuration tab....56
Figure 11.2.2: Creating address book....56
Figure 11.2.3: Adding signals....57
Figure 11.2.1: Example of real data showing Cast speed, mould level ref and act, tundish weight and stopper reference and actual position....58
Figure 11.2.2: Example of cast start Casting speed ramp....58
Figure 11.2.3: Example of cast start Casting speed ramp....59
Figure 11.2.4: Simulink scope view of Inputs to mould level controller....60
Figure 11.2.5: Simulink scope view of Mould level and reference....61
Figure 11.2.6: PLC inputs in IBA PDA....61
Figure 11.2.7: Mould level and reference....62
Figure 11.2.8: Checklist of functions....62
Figure 13.1.1: Motor details....63
Figure 13.1.2: Continuous time data of Mould level Stopper position and Current drawn....64
Figure 13.1.3: Data of Mould level, Stopper position and Current drawn during cast start....64
Figure 13.1.4: Data of Mould level, Stopper position and Current drawn during casting....64
Figure 13.1.5: Calculation of power....65
Figure 13.2.1: Hydraulic pump station MLC....66
Figure 13.2.2: Current drawn by the motor in percentage of the nominal current of the motor....67
Figure 13.3.1: Voltage and current of a linear load....67
Figure 13.3.2: Voltage and current of a non-linear load....68
Figure 13.3.3: Single line diagram of power distribution system....68
Figure 13.3.4: Figure showing the first, third, fifth harmonic and the combined distorted waveform69
Figure 13.3.5: Current Distortion Limits for....70
Figure 13.3.6: Voltage Distortion Limits....71
Figure 13.3.7: Typical values of harmonic currents for different types of front ends....72
Figure 13.3.8: Servostar 600 series drives....73
Figure 13.3.9: Technical data of the servostar drives....73

17 References

1. (Stahlinstitut VDEh, 2007): Stahlfibel, 2007
2. (Vadim Treivous, 2007): Mould level control: Ist-zustand und Ausblick, 2007
3. (Simatic S7 1500/ET 200 MP automation system in a nutshell, 2016) [Online] Available at: https://support.industry.siemens.com/cs/document/109481357/simatic-s7-1500-et-200mp-automation-system-in-a-nutshell?dl=en&lc=de-ww [Accessed 11.11.18]
4. (Simatic S7 1500 CPU 1516T-3 PN/DP, 2017) [Online] Available at: https://support.industry.siemens.com/cs/attachments/109749072/s71500_cpu1516t_3_pn_dp_manual_en-US_en-US.pdf?download=true [Accessed 11.11.18]
5. (Matlab&Simulink, 2018) Simulink Getting Started Guide [Online] Available at: https://www.mathworks.com/help/pdf_doc/simulink/sl_gs.pdf [Accessed 11.11.18]
6. (Matlab&Simulink, 2014) Simulink PLC Coder[Online] Available at: https://in.mathworks.com/products/sl-plc-coder.html [Accessed 11.11.18]
7. (MathWorks, Block Creation, 2018) [Online] Available at: https://in.mathworks.com/help/simulink/block-creation-1.html [Accessed 11.11.18]
8. (MathWorks, 2018) Discrete Derivative [Online] Available at: https://www.mathworks.com/help/simulink/slref/discretederivative.html [Accessed 11.11.18]
9. (MathWorks, 2018) Discrete Integrator [Online] Available at: https://www.mathworks.com/help/simulink/slref/discretetimeintegrator.html [Accessed 11.11.18]
10. (The University of Utah, 2011) Random Number Generator [Online] Available at: http://www.math.utah.edu/~pa/Random/Random.html [Accessed 11.11.18]
11. (TIA Portal V15 news, 2018) [Online] Available at: www.siemens.com/tia-portal [Accessed 11.11.18]
12. (TIA Portal SIMATIC Creating the project and hardware, 2013) [Online] Available at: https://www.automation.siemens.com/salesmaterial-as/interactive-manuals/getting-started_simatic-s7-1500/documents/EN/config_en.pdf [Accessed 11.11.18]
13. (Programming Guideline for S7-1200/S7-1500, 2014) [Online] Available at: https://support.industry.siemens.com/cs/document/81318674/programming-guidelines-and-programming-styleguide-for-simatic-s7-1200-and-s7-1500?dti=0&lc=en-WW [Accessed 11.11.18]
14. (Kollmorgen Seidel, 2000) Synchronous Servomotors Series 6SM45..100 [Online] Available at: http://www.wiki-kollmorgen.eu/wiki/DanMoBilder/file/pdf_archiv/motoren/6SM-baumuller/2000-05/6SMbm_e.pdf [Accessed 11.11.18]
15. (Siemens - Harmonics in Power Systems, Causes Effects and Control, 2013) [Online] Available at: https://www.industry.usa.siemens.com/drives/us/en/electric-drives/ac-drives/Documents/DRV-WP-drive_harmonics_in_power_systems.pdf [Accessed 11.11.18]
16. (Research gate discussion, 2015) [Online] Available at: https://www.researchgate.net/post/An_induction_motor_is_a_linear_load_or_a_nonlinear_load_Then_how_to_identify_the_non_linear_loads_and_liner_loads_in_electrical_applications [Accessed 11.11.18]
17. Kollmorgen-Servostar 601…620, 2015) [Online] Available at: https://www.kollmorgen.com/sites/default/files/public_downloads/S601%20Installation%20Manual%20EN%20%28REV%2002-2015%29.pdf [Accessed 11.12.18]