

POLITECNICO DI TORINO

Master's Degree in Electronic Engineering



Master's Degree Thesis

Revamping of the automation and measurement systems of a steel mill and uncertainty evaluation

Supervisors

Prof. Alessio CARULLO

Candidate

Stefano CARROZZA

APRIL 2023

Abstract

The AOD (Argon Oxygen Decarburization) system is the part of a steel mill, found upstream in the manufacturing process, that modifies the steel composition to give it its main characteristics. The AOD is controlled by a PLC (Programmable Logic Controller) system and a weight measuring instrument is used to choose the quantity of material charged on a hopper. The hopper is then transported to the melted steel so that it can dump its content to change the composition. The weight indication given by the scale has to have a maximum admitted error that is under the threshold of 1% of the weight so that the characteristics of the final material do not have appreciable variations and, as a consequence, expensive additions do not have to be made afterwards to reach the target composition. During the thesis activity, a revamping of the system used to control the process has been done to update the older PLC system composed by two obsolete CPU (Central Processing Unit) models with up to date components that have new functionalities. The study of the indication uncertainty has been done by analysing the measuring system while consulting the modules data sheets and by studying the uncertainty propagation through the measuring chain. The PLC update is done by creating a program that can work in the new hardware configuration, which has been implemented by consulting its electrical schemes. The weight uncertainty is evaluated using the probabilistic method and the PLC program is written using the TIA (Totally Integrated automation) portal software. Blocks already used in the current system have been updated to fit in the new program and new functions have been written to handle the movement of the hopper. One of the main outcome of the measuring system investigation was that the most important uncertainty contribution is related to a not proper calibration of the PLC weighing module, the Siwarex, while the contributions related to the loading cells, the junction box and the analog-to-digital converter negligibly affect the weight measurement. For this reason, particular attention has been paid towards the correct implementation of calibration procedures for these types of weighing systems. The PLC program shows that code written in a up to date environment presents several advantages over code written in older systems and this makes possible maintenance interventions easier to be performed; the update of older control systems seems like a good direction to take for future development of other parts of the steel manufacturing process. The weight uncertainty study has found that, after a proper scale calibration, the expanded uncertainty (confidence level 95%) is of about 0.15% of the net weight. A program that handles the revamped system and which can operate the hopper in four different ways has been created and successfully simulated.

Table of Contents

List of Tables	VI
List of Figures	VII
Acronyms	IX
1 Introduction	1
1.1 General introduction	1
1.2 Plant description	2
1.3 Connection network	3
1.4 Objective	5
2 Measuring system	7
2.1 Load cells	8
2.1.1 Strain-gauge and Wheatstone bridge	8
2.1.2 Steel cylinder	9
2.2 Junction box	11
2.2.1 Current junction box	11
2.2.2 Revamped junction box	11
2.2.3 Connection to the weight processing module	11
2.3 Weight processing module Siwarex	12
2.3.1 Current weighing module Siwarex U	13
2.3.2 Revamped weighing module Siwarex WP321	16
3 Uncertainty calculation	21
3.1 Calibration function	21
3.2 Considered uncertainty contribution	23
3.2.1 Load cell contribution	23
3.2.2 Junction box contribution	24
3.2.3 Siwarex contribution	25
3.2.4 Amplifier contribution	25

3.2.5	Analog to digital converter contribution	26
3.2.6	Reading resolution	26
3.3	Gain parameter calculation	26
3.3.1	Gain calculation	30
3.4	Hopper B17 uncertainty	31
3.4.1	Weight calculation	32
3.4.2	Uncertainty propagation	32
3.4.3	Error percentages	39
3.4.4	Revamped propagation	43
4	Electrical scheme	45
4.1	Current scheme	45
4.1.1	Connection network	46
4.1.2	Racks description	47
4.2	Revamped scheme	49
4.2.1	Connection network	51
4.2.2	Racks description	52
4.3	S7 CPU description	55
4.3.1	Process images	56
4.3.2	Confrontation between current and revamped CPU	57
5	Current program	59
5.1	Data insertion	60
5.2	DB17 "DATI.B17"	61
5.3	Other data blocks	64
5.3.1	User defined types	65
5.4	FB17 "gestione_peso_B17"	66
5.5	FB21 "peso_tara_rall_b17"	73
5.5.1	Interface and local variables	74
5.5.2	Segments description	74
5.6	Weight managing function FB224	78
5.6.1	Interface and local variables	79
5.6.2	Code description	81
6	Proposed revamped program	87
6.1	Hardware setup	87
6.2	Old block transfer	90
6.2.1	Data blocks	90
6.2.2	Function blocks	92
6.3	New blocks	93
6.3.1	Data blocks	93

6.3.2	"MOVIMENTO_DB" instance	94
6.3.3	"Blocco_automatico_ausiliario"	95
6.3.4	"Calcolo_rallentamento"	95
6.3.5	"Gestione_automatico"	97
6.3.6	"Movimento"	98
6.3.7	"Presa_input_HMI"	104
6.4	Simulation blocks	105
6.5	Human Machine Interface	106
6.5.1	First page	106
6.5.2	Second page	108
6.5.3	Third page	109
6.6	Simulation results	109
6.6.1	Direct control	110
6.6.2	Local control	112
6.6.3	Manual weighing cycle	114
6.6.4	Automatic weighing cycle	118
6.7	Real implementation	120
7	Conclusion	123
8	Bibliography	129
8.1	Electrical schemes	131
A	Theoretical background	133
A.1	Wheatstone bridge	133
A.1.1	Full bridge	135
A.2	Probabilistic method	136
A.2.1	Verification scale interval	137

List of Tables

2.1	WP321 characteristic table	17
2.2	Technical data table	17
3.1	B17 parameters	23
3.2	Siwatool parameters	30
3.3	Data sheet uncertainty	33
3.4	Perfect calibration D1	35
3.5	Perfect calibration D0	35
3.6	Sensitivity case D1	36
3.7	Sensitivity case D0	36
3.8	Gain case D1	37
3.9	Gain case D0	37
3.10	Siwarex case D1	38
3.11	Siwarex case D0	38
3.12	D1 with sensitivity decay	39
3.13	D1 in perfect conditions	40
3.14	D0 in perfect conditions	41
4.1	CPU confrontation table	57
7.1	Calculation result table	124

List of Figures

1.1	Measuring system scheme for an hopper	3
1.2	Profibus connection	4
1.3	Profinet connection	4
2.1	Measuring system	7
2.2	EXC SIG SENSE connections	12
2.3	Siwarex U components	13
2.4	ADC characteristic	14
3.1	Siwatool configuration	29
3.2	Same indication for different excitation voltages	31
3.3	D1 with sensitivity decay graph	40
3.4	D1 in perfect conditions graph	41
3.5	D0 in perfect conditions graph	42
3.6	Gain error graph	43
4.1	Current network scheme	46
4.2	Main rack scheme	47
4.3	Pulpit scheme	49
4.4	Revamped Profibus network	51
4.5	Revamped Profinet network	51
4.6	Program cycle	56
5.1	Sequence description	78
6.1	Rack 1 scheme and set up	88
6.2	Network and topological view	89
6.3	Index calculation example	96
6.4	Group I activation	100
6.5	Group starting signal	101
6.6	Dumping phase starting signal	101
6.7	Dumping phase activation	102

6.8	HMI page one	107
6.9	HMI page two	108
6.10	HMI page three	109
6.11	Online connection	110
6.12	Starting screen	111
6.13	Direct movement	111
6.14	Direct dumping	112
6.15	Simulation table	112
6.16	Local control starting screen	113
6.17	Local control extractor	113
6.18	Table weight selection	114
6.19	Manual sequence insertion	114
6.20	Exceeding range	115
6.21	Weighing process-active extractor	115
6.22	Weighing process-slow down signal	116
6.23	Weighing process-end of an extraction	117
6.24	Weighing process-dumping	117
6.25	CSM simulated input	118
6.26	Sequence calculation start	119
6.27	Ordered sequence in automatic process	120
6.28	Power control unit	121
A.1	Wheatstone bridge scheme	133
A.2	Wheatstone full bridge	135
A.3	Cylinder scheme	136

Acronyms

ADC

analog to digital converter

AOD

argon oxygen decarburization

BCD

binary-coded decimal

DR

data record

HMI

human machine interface

PDF

probability density function

PGA

programmable gain amplifier

PLC

programmable logic controller

SCL

structured control language

TIA

totally integrated automation

Chapter 1

Introduction

1.1 General introduction

This document is focused on the AOD system of a steel mill and the revamping of the PLC used to control it. The AOD is the argon oxygen decarburization system, which is the part of the manufacturing process that changes the percentage of components in the steel in order to give it different proprieties. The process is divided into several steps that change the steel chemical composition by adding different elements and components in gas and solid form. In order to reach the target composition a precise amount of reagent should be added to the melted steel. Several weight measuring systems are used to achieve this goal and then the final quality of the steel and its characteristics are largely affected by the quality of these measurements. The order of magnitude of the weight of the added material is of hundreds of kilograms up to around 3000 kilograms depending on the wanted target composition, providing an uncertainty lower than a couples of kilograms for the lighter loads and up to 30 kg for the heavier ones so that the relative

uncertainty is lower than 1% is important because correcting possible errors in the steel composition is an expensive process that requires the dilution of the entire compound and, as a consequence, the modification of all the materials percentages. The AOD process, including the weighing of the added material, is automated by a PLC control system. The system during the thesis activity has been going through a revamping and the older components are replaced by newer ones. Currently the weighting system is controlled by a CPU 319-3 PN/DP and it is programmed in step 7, the software used to develop the functions of the series 300 Siemens CPU, while the component that controls the power modules is an older S5 CPU. Because of this difference, the program has to exchange information between the two units and the code is therefore more complicated. In the revamped system the weighting and power modules are all controlled by a CPU 1516-3F and the platform used for the program is TIA portal. Thanks to this improvement the weight control system, the movement control system and the HMI, which are the interfaces that the operator uses to control the manufacturing process in the AOD, can all use the same variables. On the other hand, these updates can upgrade the overall performance of the system because of the new devices better technological parameters and new features and simplify the program structure and accessibility because of the use of the TIA platform.

1.2 Plant description

In the AOD plant, the different materials are collected from a series of silos and transported to the melted steel by three hopper: B17, B18 and A15. The silos are positioned above the corresponding hopper moving track. Each hopper is able

to weight the quantity of collected material thanks to a measuring system 2 (1.1) based on loading cells and dump its content into the melted steel. The movement of all these components is controlled by an operator in the pulpit, the control cabin positioned above the AOD plant. Some chemical components are gases, like argon and oxygen, and are injected directly in the melted steel with some pipes. Other materials that are not stored in the silos can be dumped in the steel with conveyors and are not weighted with the hopper scale.

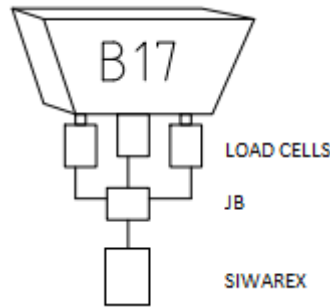


Figure 1.1: Measuring system scheme for an hopper

1.3 Connection network

The PLC control system is composed by several nodes that are connected to each other in two possible ways.

- PROFIBUS DP: It is the protocol used to control several remote I/O slaves, like sensors or actuators, with one central master controller with a bit-serial communication. It stands for process field bus decentralised peripherals. It is used for cyclical data exchange between the peripherals, with variable transmission rate from 9.6 kb/s to 12 Mb/s. The cables are connected through a RS-485 interface and the communication type is half-duplex. The several

peripheral devices are connected in a chain to the controller.

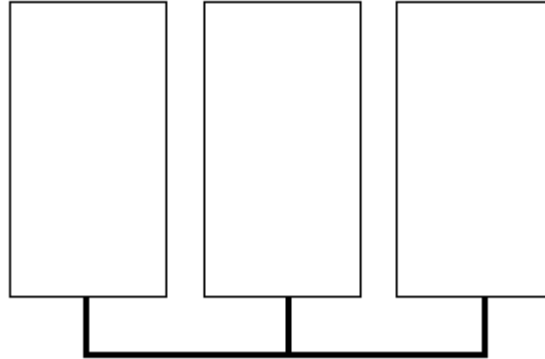


Figure 1.2: Profibus connection

- PROFINET IO: It is a full duplex system used for data communication over industrial ethernet and it is used to connect a IO controller, a PLC CPU, with IO devices, sensors or actuators, which can be composed of several sub devices. This type of connection is used for the transmission of big quantity of data at higher rate, 100 Mb/s, in respect to the Profibus network even beyond the local network. The devices are connected through a central switch to the main controller.

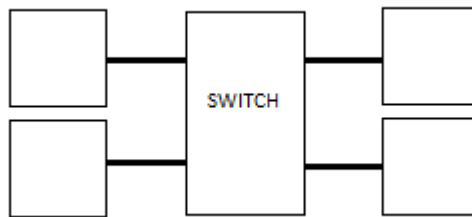


Figure 1.3: Profinet connection

Both these types of communication must be configured in the PLC software (step

7 or TIA portal). Each device connected to the Profibus network must have a Profibus address (usually the CPU has 2) while each device connected to the Profinet network must have an IP address (for example 192.168.148.4).

1.4 Objective

The thesis activity has two primary objectives. The first one is to find out the quality of the weight measure performed by the AOD scales; in particular the focus is put into the hopper B17 which has the task to collect a certain amount of material from a series of silos and then transport it and dump it into the melted steel. The second one is to write a program, using TIA portal, that can control the hopper B17 so that it behaves in a similar way to the current configuration. This is done by transporting old function and data blocks into the new program and by writing new blocks.

The measuring system study is focused on three important blocks: the load cells, the junction box and the Siwarex weight processing module; these are used to measure the weight on the hopper and transfer its value to the PLC. The quality of the measure is evaluated by calculating the uncertainty contribution of all the components and by studying the calibration process.

The program is focused on the automation of the movement part of the hopper that is, in the current configuration, controlled by the older S5 CPU. The primary objective of this part consists in writing new blocks that can move the hopper in accordance with the current control system and create a simulation that can show how it behaves.

Chapter 2

Measuring system

The measuring system is very important because it is used to control how much material is inserted in the melted steel during the AOD process. It is composed by the load cells, the junction box and the Siwarex weight processing module (2.1). The following sections explain how these components work and what is the difference between the current setup and the post revamping setup.

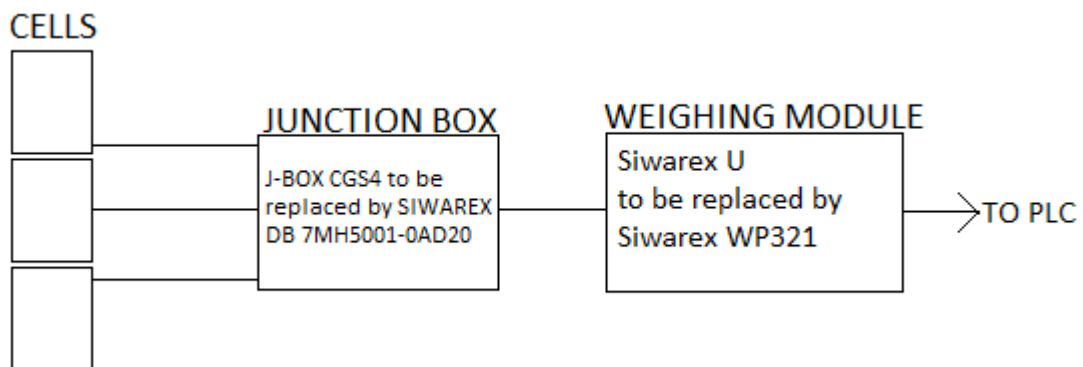


Figure 2.1: Measuring system

2.1 Load cells

A load cell is a device that can measure an applied force by changing its output signal proportionally to the deformation sensed by the cell itself. This means that a load cell can vary its output voltage in function of the weight that is placed upon it. The current system and the revamped one will have the same load cells so all the characteristics of the sensors of the current measuring chain are equal to the revamped one.

2.1.1 Strain-gauge and Wheatstone bridge

There are several types of load cells but the one used in this measuring system are strain-gauge cells. A strain-gauge is an object that can detect mechanical deformation and it is modeled as a variable resistance.

The resistance variation is determined by the gauge factor.

$$K = \frac{\frac{\Delta R}{R}}{\frac{\Delta L}{L}} \quad (2.1)$$

where ΔR is the resistance change with respect to the unloaded resistance R and ΔL is the length change with respect to the nominal length L .

Some uncertainties caused by the zero deviation are still found in this configuration due to little differences in the temperature coefficients; for this reason some compensation resistances are added to the Wheatstone bridge without, however, changing its behaviour.

2.1.2 Steel cylinder

A load cell is composed by a steel cylinder that converts the load deformation in a linear way and with low elastic hysteresis connected to four resistive strain-gauges in a full bridge configuration (A.1.1) so that the deformation can be converted into resistance variation (A.3).

The output voltage is:

$$V_{sig} = \frac{F \cdot V_{exc} \cdot K \cdot 2(1 + \mu)}{E \cdot \pi \cdot D^2} \quad (2.2)$$

where:

- F : Applied mechanical force
- V_{exc} : Excitation voltage
- K : Gauge factor
- μ : Steel Poisson ratio; it indicates how much a material is deformed perpendicularly to a received stress
- E : Elastic modulus; it indicates how much an object will elastically deform when a stress is applied
- D : Diameter of the cylinder

The variables of the cylinder function (2.2) can then be translated into a series of parameters that are provided by the data sheet.

$$V_{sig} = \frac{P \cdot V_{exc} \cdot S}{Range} \quad (2.3)$$

where P is the input weight.

The most important characteristics of a cell are therefore:

- Sensitivity S : its unit of measure is mV/V and it indicates how much the output voltage changes in function of the excitation voltage. The value should be multiplied by the excitation voltage and the input weight range in order to find the maximum possible output of the cell.
- *Range*: it indicates the maximum input weight that the cell can measure and it is expressed in kilograms.
- Number of verification scale interval $\#e$ (A.2.1): it is not used in the previous formula but it is useful in order to find the value of e .
- Value of a verification scale interval e . (A.2.1): it is found by dividing the range with the number $\#e$. It is not used in the previous formula but it is useful to find the reading resolution.
- Excitation voltage V_{exc} .

With these elements it is possible to calculate the cell voltage output given the input weight or find the input weight given the output voltage. It is also possible to conclude that a scale connected to this cell would have a reading resolution of e kg; this is due to the fact that, even if the weighing module can display intermediate values, only the multiples of e are significant because they have been approved during the initial verification phase (A.2.1).

The measuring system under study uses three nominally equal load cells positioned so that each one of them measure a third of the total load.

2.2 Junction box

Given the fact that in the measuring system the weight on the hopper is taken by multiple load cells, the signal must be transferred to the weighing module by a device that can translate three input voltages into one.

2.2.1 Current junction box

The current junction box, J-BOX CGS4, is simply a device that puts in parallel the three cells signals and, as a consequence, creates an output that behaves like a single cell. Sensitivity and excitation voltage are the same as the single cell case, because the cells are all equal to each other, but the range is triplicated. In case of a fault it is possible to find out which cell caused it only by controlling every cell one by one.

2.2.2 Revamped junction box

The revamped system will use a more sophisticated digital junction box, SIWAREX DB 7MH5001-0AD20, that is able to monitor the state of the individual load cells. Thanks to this new feature, the diagnostic control is far quicker because an eventual fault can be immediately examined without the need of checking each cell. The diagnostic can be accessed through the Siwarex WP321 software (section 2.3.2), where different types of signals can be seen in the error report data record.

2.2.3 Connection to the weight processing module

In order to have a weight indication the device is connected to a scale or, in this case, to a weight processing module whose input connector can be seen in figure 2.2.

The connections are the EXC+ and EXC- for the excitation voltage (used as input in the Wheatstone bridge (A.1)) and the SIG+ and SIG- for the signal voltage (the output voltage of the Wheatstone bridge (A.1)). Another pair of output signals, SENSE+ and SENSE-, are positioned in the same spot as the excitation connection. With the sense terminal the Siwarex module obtain a feedback of the true value of the excitation voltage at the Wheatstone bridge. With this information the real excitation voltage can be modified so that its value matches the nominal one and the weight measure becomes, as a consequence, more precise.

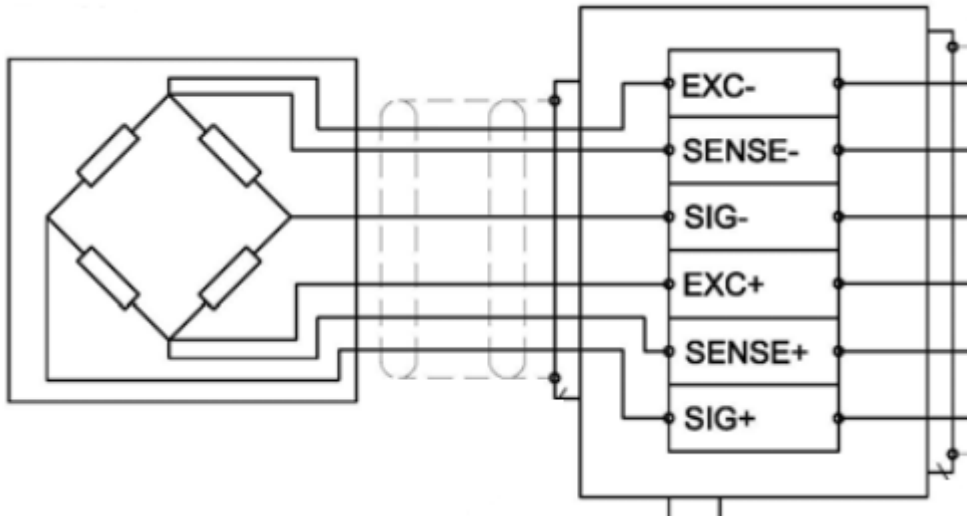


Figure 2.2: EXC SIG SENSE connections

2.3 Weight processing module Siwarex

The Siwarex module is the component of the measurement system that translates the voltage value of the junction box into a digital weight indication used by the PLC. The module is composed by a low pass filter, used for filtering out some noise in the signal, a programmable amplifier, used to adjust the signal amplitude, and

an analog to digital converter, used to translate the voltage value into a digital code. The digital code is eventually converted into a weight indication that is used by the PLC. The gain of the amplifier is actually not known because the manufacturer datasheets do not clearly state its value. To fill this gap, the amplifier gain has been evaluated through the calibration procedure described in section 3.3. On the one hand, this makes it possible to find a gain that can be used in the calibration function but, on the other hand, this creates a gain that depends on the calibration data. The consequence of this choice is that possible errors in the calibration process are reflected on the gain. The calculated gain will therefore vary each time that the calibration data changes, so every time that a new calibration takes place, even if the actual amplifiers gains remain probably the same. The current setup uses a Siwarex U module while the revamped version uses a Siwarex WP321 module.

2.3.1 Current weighing module Siwarex U

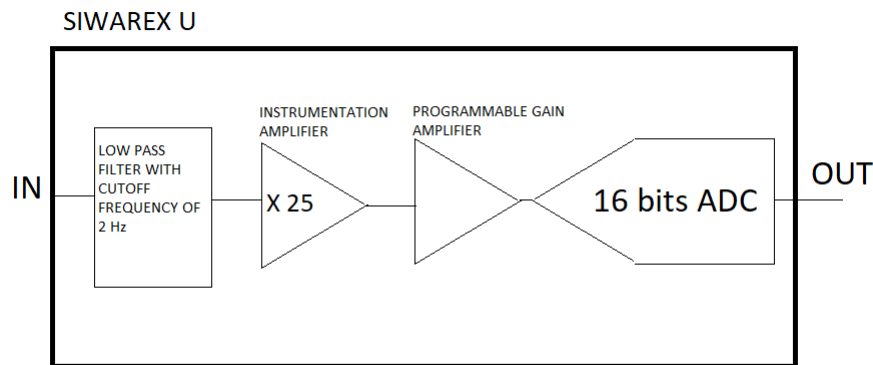


Figure 2.3: Siwarex U components

This module (2.3) can be directly connected to a PC through an RS-232 interface. It has a fourth order low pass filter with a cutoff frequency of 2 Hz. The amplifier is composed by a 25 gain instrumentation amplifier followed by a programmable gain amplifier. The ADC is based on a delta-sigma modulator and has a resolution of 16 bits. The converter operates in unipolar mode but the characteristic is translated to the left so that 4% of it is in the negative field. This means that an input voltage of 0 mV would translate into an output digital code equals to 2427 as it can be seen in the 2.4 figure.

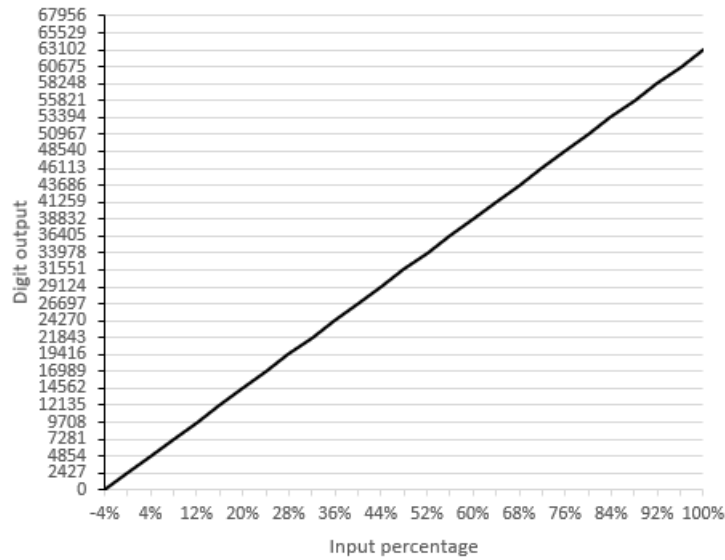


Figure 2.4: ADC characteristic

In order to define the gain of the PGA the calibration must be done. To calibrate this instrument a software called Siwatool is used. It is possible to do a theoretical calibration if the output code corresponding to a certain weight is already known but, in this case, the calibration would have a lower quality. A better method, that is actually used, is to calibrate the instrument using a sample weight and a dynamometer by following this procedure:

1. Connect the Siwarex U module to the PC using the RS-232 interface and start the Siwatool software.
2. Select the wanted hopper, in the studied case the B17 one, and start communication.
3. From the pulpit commands translate the hopper on the spot with the hook and then secure the working area; that is making sure that the hopper is unable to move.
4. Press the "receive" button so that the software can fetch the old calibration data of the scale.
5. Make sure that the hopper is empty and then press the button "valid zero". Now the module has registered the current output digit as the code where the weight indication is 0 kg. This mean that the offset has been found.
6. Make sure that the indication is 0 kg and press the button "transmit" to send the data to the hopper.
7. Connect the dynamometer to the hopper and wait until the zero is set.
8. Connect the whole system to the hook on the floor.
9. Apply a force that is approximately 70% of the scale maximum range.
10. Read the value shown by the dynamometer.
11. Press again the "receive" button.
12. On the Siwatool software, enter the seen value plus 12 kg for the tare of the dynamometer.

13. Press again the "transmit" button.
14. Press on the "valid calibration point" button. Now the system has built the calibration function and the indicated weight must be the same that was inserted.
15. End the PC connection and disconnect the dynamometer.
16. After the hopper is activated again, press the "tare" button on pulpit console so that the Siwarex module and the pulpit have the same weight indication.

At this point a new procedure starts in order to find out the quality of the calibration. From the pulpit a command is given to fill the hopper with a certain quantity of a material; this is poured into a container that is then transported to the scale of the scrap park. If the weight indication of the scrap park scale is the same that was selected in the AOD pulpit, excluding the container tare, then the calibration is finally completed. The scrap park scale can be used for this purpose because it is periodically checked by external staff and it has certificates that proves its traceability and reliability.

2.3.2 Revamped weighing module Siwarex WP321

This module can be directly connected to a PC through an RS-485 interface. Its low pass filter, which has a cutoff frequency of 2 Hz, can be set to be of the second order or the fourth order. Unfortunately there is no precise data on the amplifier or the ADC but some of it can be extrapolated from the Siwarex data sheet.

Table 2.1: WP321 characteristic table

	Default	Minimum	Maximum
Calibration weight 0	0	1	9999999
Calibration weight 1	100	1	9999999
Calibration weight 2	0	1	9999999
Digits for D0	0	-3999999	+3999999
Digits for D1	2000000	-3999999	+3999999
Digits for D2	0	-3999999	+3999999

Looking at the possible output digit value in the 2.1 table, it is evident that the ADC operates in bipolar mode and that it has at least 23 bits because it can have a digit value that goes from -3999999 to +3999999. The maximum possible value of the calibration weight is well above to the range of the cells system.

The revamped module has an overall better performance compared to the current device; the technical data table (Table 2.2) shows some comparisons.

Table 2.2: Technical data table

	Siwarex U	Siwarex WP321
Supply voltage [V]	24	24
Maximum consumed current [mA]	220	100
Typical consumed power [W]	4.8	2
Minimum input voltage for e [μ V]	1.5	0.5
Update rate [Hz]	50	100/120
Internal resolution [bits]	16	>23
Weights representation	-32768 : +32767	-2000000 : +2000000
Maximum uncertainty	0.05% FS	0.05% FS

An important parameter that is shown here, and that will be used in the uncertainty calculation chapter (3), is that the maximum uncertainty provided by the data sheet is equal for the two devices.

Besides all of this, the WP321 device has more features than its predecessor like the diagnostic feature, that is used thanks to the new digital junction box (2.2.2),

and the self-adjusting feature that enable the scale to self adjust for slow zero drift. This last feature enable the scale to be less susceptible to calibration decay (but it does not substitute a periodical calibration of the instrument).

The Siwatool software can access directly to module data records. A data record is where important information is stored and where useful parameters can be set. Two very important data records are the DR3, which contains the calibration parameters, and the DR10, which contains the load cells parameters. Both of these are used during the calibration process of the instrument.

Also in this case it is possible to perform a theoretical calibration but, as in the other device, the results are less precise. Sample weight calibration with this device can be different from the other one because it is possible to make the calibration function using two sample weights, as seen in the 2.1 table. However, in the context of the AOD hopper, a calibration with just one sample weight is preferable because the characteristic has a mostly linear behaviour. In order to calibrate the instrument some other parameters have to be added. These parameters are, on the DR3, the unit of weight, in this case kilograms, the maximum range of the scale expressed in unit of weight, which must be less or equal to the maximum range of the cells system, and the numeric step, which is the value of the minimum weight that the module can reliably indicate. The numeric step is calculated by dividing the singular cell range by 3000 because the cell is a C3 model. The DR10 parameters are the number of load cells used, the sensitivity of each one of them and the range in unit of weight of every cell.

As will be explained in the 3 chapter, the calibration process is very important to guarantee a good quality of the weight indication and, for this reason, an effective

calibration procedure must be implemented also for this device. A possible procedure would likely follow similar steps to the current device: data reception of the scale, zero weight selection, external weight added (probably with a dynamometer like the current case), tare weight selection, pulpit indication modification and final check with an external weight.

Chapter 3

Uncertainty calculation

The calculation of the uncertainty propagation of the measuring system described in chapter 2 is performed by using the probabilistic method (A.2). The random variables regarding the cells and weighing modules uncertainty contribution are assumed uncorrelated to each other (because there is no reason to believe otherwise) and all of them are taken from the respective device manuals.

3.1 Calibration function

The goal now is to find the relation between the input weight and the ADC output code.

The input output relationship of a load cell is derived by the equation 2.3:

$$P = \frac{V_{cell} \cdot Range_{cell}}{S_{cell} \cdot V_{exc}} \quad (3.1)$$

where P is the input weight, $Range_{cell}$ is the load cell maximum range, S_{cell} is the cell sensitivity, V_{exc} is the excitation voltage and V_{cell} is the cell output voltage.

If the parameters of the cells junction box system are inserted instead of only the cell one then the relationship between weight input and the junction box voltage output is found.

$$P = \frac{V_{JB} \cdot Range_{JB}}{S_{JB} \cdot V_{exc}} \quad (3.2)$$

where V_{exc} is the same in both cases, S_{JB} has the same value as before and $Range_{JB}$ is equal to the cell one multiplied by the number of cells.

The input output relationship of the amplifier is:

$$V_{JB} = \frac{V_{ADC}}{G} \quad (3.3)$$

where G is the gain and V_{ADC} is the output voltage that enters the ADC.

The input output relationship of the current ADC is:

$$V_{ADC} = (DIGIT - 2427) \cdot \frac{V_{ref}}{2^n} \quad (3.4)$$

where the output code $DIGIT$ is corrected by 2427 because of the Siwarex U calibration characteristic (2.3.1), n is the number of bits and V_{ref} is the ADC reference voltage.

The complete calibration function is therefore the following:

$$P = \frac{\frac{(DIGIT - 2427) \cdot \frac{V_{ref}}{2^n}}{G} \cdot Range_{JB}}{S_{JB} \cdot V_{exc}} \quad (3.5)$$

The found calibration function is valid for the current configuration and not for the revamped one because the revamped ADC operates in bipolar mode and does

not need a correction value of the output digit.

In the hopper B17 case, using the current configuration, the used parameters are shown in table 3.1:

Table 3.1: B17 parameters

$Range_{cell}$	2000	kg
S	2	mV/V
V_{exc}	10.3	V
number of cells	3	
$Range_{JB}$	6000	kg
#e	3000	
e	2	kg
n	16	bits
V_{ref}	2500	mV

It is important to notice that the gain G is not in the table because it is unknown as explained in the 2.3 section; the procedure on how to find it is shown in the paragraph 3.3.

3.2 Considered uncertainty contribution

The following contributions are provided by the respective device manuals in a deterministic way and, as consequence, they must be translated to a uniform distribution (A.2).

3.2.1 Load cell contribution

- Combined error E_c : It indicates the hysteresis and linearity errors. The hysteresis error is the output variation for the same input weight depending on whether the value is reached by increasing or decreasing the input. The

linearity error is the maximum variation from a straight line that the output characteristic has.

- Variability (repeatability) E_v : It indicates the maximum variation of output voltage when the input weight and the environmental factors are the same.
- Creep E_{creep} : It indicates the maximum variation of the output while the cell is under prolonged stress.
- Sensitivity E_s : It is the cell sensitivity uncertainty.

The sensitivity and excitation voltage uncertainty contribution (the last of which is not indicated because it is not present on the data sheet) should not be considered if the calibration has taken place; this because the calibration characteristics of the instrument are a function of sensitivity and excitation voltage. Once the instrument is calibrated the output weight indication is not influenced by their uncertainties if their values remain constant; this can be proven by looking at the gain calculation procedure (3.3). Also for this reason a periodical check on the instrument calibration is very important (2.3.1).

3.2.2 Junction box contribution

Given the fact that there are three load cells, which output voltages are averaged together, the uncertainty contribution should be changed to reflect that.

$$\begin{aligned}
 V_{JB} &= \frac{V_{cell1} + V_{cell2} + V_{cell3}}{3} = \frac{(V_{exc} \frac{P/3}{Range_{cell}})(S_{cell1} + S_{cell2} + S_{cell3})}{3} = \quad (3.6) \\
 &= V_{exc} \cdot S_{JB} \cdot \frac{P}{Range_{JB}}
 \end{aligned}$$

From the previous formula the following uncertainty propagation can be derived:

$$u_r(cell) = \sqrt{u_r^2(c) + u_r^2(v) + u_r^2(creep)} \quad (3.7)$$

$$u_r(JB) = \sqrt{3 \cdot \left(\frac{1}{3}u_r(cell)\right)^2}$$

$$u(S_{JB}) = \sqrt{3 \cdot \left(\frac{1}{3}u(S_{cell})\right)^2} \quad (3.8)$$

The junction box itself does not add any other uncertainty.

3.2.3 Siwarex contribution

The weight processing module uncertainty contribution can be computed in two ways. The first one is to use the data sheet information which states the uncertainty of the whole Siwarex module (last element of the table 2.2). This is probably the best choice that can be made because it takes into consideration all the possible Siwarex uncertainty contributions. Nevertheless it is interesting to proceed in another way, looking at the uncertainty propagation in the amplifier and in the ADC, because their singular contribution can be seen. This can only be done in the Siwarex U case because there is no data on the revamped module amplifier and ADC.

3.2.4 Amplifier contribution

The gain is found by mathematical means (3.3). If the calibration of the system is assumed to be perfect then the gain uncertainty is derived by the mathematical calculation uncertainty. No other contribution of this stage is listed on the data

sheet.

3.2.5 Analog to digital converter contribution

The ADC adds the following uncertainty contributions:

- Input noise E_{in}
- Quantisation error E_Q
- Offset error E_o
- Non differential linearity E_{ld}
- Linearity error E_l
- Gain error E_g

The reference voltage uncertainty is not provided in the data sheet and as a consequence is not taken into account. This is one of the reasons why taking the whole Siwarex contribution returns a more accurate result.

3.2.6 Reading resolution

The final contribution to add is the uncertainty due to reading resolution. As seen in the load cells paragraph (2.1) the scale reading resolution is given by the value of a verification scale interval e (A.2.1) of the load cells system.

3.3 Gain parameter calculation

As seen in the table 3.1, the gain G is the only parameter whose value is missing in the calibration function because it is determined every time that a new calibration

takes place. This is due to the fact that the gain present in the calibration formula is not the actual amplifier gain, as explained in the 2.3 section, because it is found by using the calibration data.

The following example shows how the gain is influenced by the calibration data: Two cell systems called system A and system B are used. System A has a range of 1000 kg and system B has a range of 500 kg; both have a sensitivity of 2 mV/V. The used ADC has, for simplicity sake, an output that goes from 0 to 1000 digits. If the systems are loaded with half their maximum range, 500 kg and 250 kg, they will both provide the same V_{JB} at 10.3 mV (as explained in the 3.1 section). Because they have the same V_{JB} and the same actual amplifier gain, their ADC digital output is the same at 500 digits but their weight indications are one the half of the other because of the $\frac{1}{2}$ factor given by the difference in ranges in the calibration function. The calculated gains are the same in both systems and they reflect the actual amplifier gain.

$$V_{JB} = 10.3 \rightarrow G_{amp} \cdot 10.3 \text{ it is the value in input at the ADC}$$

$$\text{so } DIGIT = 500 \rightarrow A : 500kg \ B : 250kg$$

Now if the conditions of the system A changes and, for the same 500 kg in input, the V_{JB} is slightly higher than in the previous case. The actual gain amplifier is the same as before so the ADC digital output value is slightly bigger in the system A to respect to the system B. The weight indications are, however, still correct at 500 kg and 250 kg because the calibration process has assigned the higher digit value to the correct weight indication thanks the difference in calibration function.

$$Va_{JB} = 10.506 \quad Vb_{JB} = 10.3 \rightarrow G_{amp} \cdot 10.506 \text{ and } G_{amp} \cdot 10.3$$

are the value in input at the ADC

$$\text{so } DIGITa = 510 \quad DIGITb = 500 \rightarrow A : 500kg \ B : 250kg$$

because the ADC input is $G_{amp} \cdot 10.3 \cdot G_{calibration}$ with $G_{calibration} = 1.02$
and $510/1.02 = 500$

This difference in digital value assignment is reflected in the proposed calibration function as a gain variation because of how its value is calculated; this means that the factor given by parameters uncertainties seen in the above formula is united with the gain in the studied calibration formula $G = G_{amplifier} \cdot G_{calibration}$. Where $G_{calibration}$ is due to the parameters differences from the nominal value in the real calibration formula.

The gain calculation procedure is useful because it makes possible to define a gain value for the calibration function even if the actual gain of the amplifier is not known. This method makes the calibration formula dependant on the data used in the calibration process; this means that the uncertainty calculation should only take into account the parameters that can vary after a calibration has taken place.

The procedure used to calculate the gain is the following:

1. The information of the Siwatool calibration page must be consulted:

Figure 3.1: Siwatool configuration

The important data provided by this image are the output digit D0, which is the output code when the scale has to indicate 0 kg, the output digit D1, which is the output code when the scale has to indicate the calibration weight, and the value of the sample calibration weight.

2. The difference between the digits must be calculated. The result, called Delta, is the number of digits that indicates the calibration weight. This step can be done because there is a linear relationship between the weight and the digital code.
3. Finally, G must be calculated using the inverse formula of the calibration function by inserting the sample calibration weight in place of P and Delta

in place of DIGIT (the 2427 correction it is not needed because Delta is a difference of codes).

$$G = \frac{Delta \cdot V_q \cdot Range_{JB}}{P \cdot S_{JB} \cdot V_{exc}} \quad (3.9)$$

Where V_q is the quantisation voltage of the ADC and its value is $V_q = \frac{V_{ref}}{2^{16}}$.

3.3.1 Gain calculation

Using the information of the image 3.1 the values of the parameters are in the 3.2 table:

Table 3.2: Siwatool parameters

D0	26602
D1	53953
Delta	27351
Sample weight	2702 kg

So the formula becomes:

$$G = \frac{27351 \cdot \frac{2500}{2^{16}} \cdot 6000}{2702 \cdot 2 \cdot 10.3} = 112.4687771 = 112.468 \pm 0.001 \quad (3.10)$$

Again, the found gain is not the actual amplifier one because it takes into consideration the calibration data.

This way of calibration, as stated before, compensates possible variations on sensitivity and excitation voltage. If, as an example, the value of V_{exc} was $V_{exc}=10$ V instead of $V_{exc}=10.3$ V then the maximum junction box output voltage (derived from figure 3.2) would be 20 mV instead of 20.6 mV and all the voltages in respect to the same weights would be scaled accordingly. This means that the output digits D0 and D1 would be scaled down and their difference (Delta) would also be

smaller but the sample weight would still be the same as before. This means that even if the codes are different the actual weight indication would be the same after a calibration of the system. In fact, using the calibration function, these results shown in figure 3.2 can be achieved:

$$\begin{aligned}
 2702kg &\rightarrow 27351 \text{ digit} \\
 27351 \text{ digit} &\rightarrow V_{JB} = 9.276mV \\
 \frac{9.276}{10.3} &= \frac{X}{10} \rightarrow X = V_{scaled} = 9.006mV \\
 9.006mV &\rightarrow 26554 \text{ scaled digit} \\
 26554 \text{ scaled digit} &\rightarrow 2702kg
 \end{aligned}$$

Figure 3.2: Same indication for different excitation voltages

where the scaled formula is changed by a factor of 1.03 because of the difference in excitation voltages.

This means that a scale correctly calibrated in different conditions provide the same weight indication.

3.4 Hopper B17 uncertainty

The weight and uncertainty propagation can now be calculated by using the probabilistic approach (A.2) and the just calculated gain $G=112.468$. The calculations are done in the calibration weight cases $D0=26602$ and $D1=53953$ so that the tare weight can be calculated and the uncertainty propagation can be seen in a realistic case.

3.4.1 Weight calculation

The parameters used in this equation are the one described in the table 3.1. The calculated values are rounded to a nano volt. The results are the gross weights of the D0 (3.12) and D1 (3.11) cases.

$$V_{ADC} = (53953 - 2427) \cdot \frac{2500}{2^{16}} = 1965.560913mV \quad (3.11)$$

$$V_{JB} = \frac{V_{ADC}}{112.468} = 17.476623mV$$

$$P = V_{JB} \cdot \frac{6000}{20.6} = 5090.27kg \rightarrow 5090kg$$

$$V_{ADC} = (26602 - 2427) \cdot \frac{2500}{2^{16}} = 922.203064mV \quad (3.12)$$

$$V_{JB} = \frac{V_{ADC}}{112.468} = 8.199692mV$$

$$P = V_{JB} \cdot \frac{6000}{20.6} = 2388.26kg \rightarrow 2388kg$$

With the result of the D0 case the tare weight can be found: TARE=2388 kg. By knowing this the net weight can easily be found by subtracting the gross weight to the tare: NET=5090-2388=2702 kg which is the same value as the sample weight used for the calibration.

3.4.2 Uncertainty propagation

A list of the used uncertainties contributions provided with the deterministic approach by the data sheet and translated into a uniform distribution is shown in the table 3.3:

Table 3.3: Data sheet uncertainty

Instrument	Contribution	Deterministic value		Probabilistic value (approximated)	
LOAD CELLS 3.2.1	E_c	0.025	%	0.0144	%
	E_v	0.01	%	0.0005	%
	E_{creep}	0.05	%	0.0288	%
	E_s	0.1	%	0.0577	%
ADC 3.2.5	E_{in}	353	nV	204	nV
	E_Q	0.5	LSB	0.28867	LSB
	E_o	2	LSB	1.15470	LSB
	E_{ld}	0.5	LSB	0.28867	LSB
	E_l	0.003	% FS	0.0017	% FS
	E_g	31	ppm	17.89785	ppm
AMPLIFIER	G.ERROR	0.001		0.000577	
SIWAREX		0.05	% FS	0.0288	% FS

From this table the sensitivity and junction box contributions can be found using the formula 3.7 and 3.8: $u_r(cell) = 0.0327\% \rightarrow u_r(JB) = 0.0188\%$ and $u(s) = 0.000666 \text{ mV/V}$.

By combining together the ADC contributions (row 6 to 10) the following uncertainty is found: $u(ADCout) = 2.04066 \text{ LSB}$. This is translated to input uncertainty and combined with the input noise E_{in} to find $u(V_{ADC}) = 0.077845 \text{ mV}$.

At this point the uncertainty propagation can be found by deriving the equations of the probabilistic method (A.2) using the calibration function seen in the 3 section.

The propagation chain is:

$$u(amp) = \sqrt{\left(\frac{V_{ADC}}{G^2}\right)^2 \cdot \left(\frac{G.ERROR}{\sqrt{3}}\right)^2 + \left(\frac{1}{G} \cdot u(ADC)\right)^2} \quad (3.13)$$

$$u(cells) = \sqrt{u(amp)^2 \cdot \left(\frac{Range_{JB}}{S \cdot Vexc}\right)^2 + (u_r(JB) \cdot V_{JB} \cdot \frac{Range_{JB}}{S \cdot Vexc})^2 + \left(\frac{V_{JB} \cdot Range_{JB}}{S^2 \cdot Vexc} \cdot u(s)\right)^2} \quad (3.14)$$

where the last contribution is due to sensitivity uncertainty and the gain G is not the actual amplifier gain but it is the one calculated in the 3.3 section because of the reasons explained in 2.3.

Finally the reading resolution contribution can be added; this is found with the value of a verification scale interval e (A.2.1): $E_{res} = \frac{e}{2} \rightarrow \div \sqrt{3} \rightarrow u(res)$.

With this the final calculation can be done:

$$u(weight) = \sqrt{u(cells)^2 + \left(\frac{E_{res}}{\sqrt{3}}\right)^2} \quad (3.15)$$

Now, using the equations mentioned above and the B17 parameters of the 3.1 table, different uncertainty cases can be studied. These cases are the perfect calibration case, the sensitivity case, the gain case and the Siwarex case.

- Perfect calibration case:

Here a perfect calibration of the device is supposed; this means that the amplifier gain uncertainty is kept at a minimum (only due to mathematical calculation) and the sensitivity contribution is eliminated because it is supposed that, after a possible calibration, its value does not change. With these suppositions it is found that:

Table 3.4: Perfect calibration D1

$u(V_{ADC})$	0.077845 mV
$u(amp)$	0.000698 mV
$u(cells)$	0.9781 kg
$u(weight)$	1.1357 kg
$\frac{u(weight)}{NET} \cdot 100$	0.042%
$\frac{u(weight)}{GROSS} \cdot 100$	0.022%

Table 3.5: Perfect calibration D0

$u(V_{ADC})$	0.077845 mV
$u(amp)$	0.000693 mV
$u(cells)$	0.4922 kg
$u(weight)$	0.7587 kg
$\frac{u(weight)}{GROSS} \cdot 100$	0.032%

- Sensitivity case:

Here a case where, after a good calibration so that the gain uncertainty can be considered small like in the previous case, the sensitivity changes after the scale is already calibrated. These two variables are not correlated to each other, even if the sensitivity (and excitation voltage) appears in the gain calculation equation, because the parameters used in the calibration process are fixed and different to the one that are affected by uncertainty. This case is not particularly realistic because a sensitivity drift (and also a excitation voltage drift) after a calibration would be compensated by a periodical calibration. It is nevertheless interesting to look at this situation as an hypothetical intermediate case. With these suppositions it is found that:

Table 3.6: Sensitivity case D1

$u(V_{ADC})$	0.077845 mV
$u(amp)$	0.000698 mV
$u(cells)$	1.9571 kg
$u(weight)$	2.0405 kg
$\frac{u(weight)}{NET} \cdot 100$	0.075%
$\frac{u(weight)}{GROSS} \cdot 100$	0.040%

Table 3.7: Sensitivity case D0

$u(V_{ADC})$	0.077845 mV
$u(amp)$	0.000693 mV
$u(cells)$	0.9353 kg
$u(weight)$	1.0991 kg
$\frac{u(weight)}{GROSS} \cdot 100$	0.046%

- Gain case:

Here a case where the gain uncertainty is put to G.ERROR=1. This type of error can happen if the calibration is not done correctly, for example because the procedure (2.3.1) is not followed. This possibility is compensated by the fact that the calibration procedure is always followed; this means that an uncertainty of this kind is realistically never verified. With these suppositions it is found that:

Table 3.8: Gain case D1

$u(V_{ADC})$	0.077845 mV
$u(amp)$	0.089718 mV
$u(cells)$	26.2038 kg
$u(weight)$	26.2102 kg
$\frac{u(weight)}{NET} \cdot 100$	0.970%
$\frac{u(weight)}{GROSS} \cdot 100$	0.515%

Table 3.9: Gain case D0

$u(V_{ADC})$	0.077845 mV
$u(amp)$	0.042099 mV
$u(cells)$	12.2958 kg
$u(weight)$	12.3093 kg
$\frac{u(weight)}{GROSS} \cdot 100$	0.515%

- Siwarex case:

Here a case where, instead of calculating the propagation of the ADC and the amplifier, the uncertainty of the entire Siwarex module is used. This means that the value of $u(amp) = \frac{0.05\% \cdot 20.6}{\sqrt{3}}$. This is probably the most realistic case because it takes into consideration the maximum possible contribution of the Siwarex module. With these suppositions it is found that:

Table 3.10: Siwarex case D1

$u(amp)$	0.005946 mV
$u(cells)$	1.9788 kg
$u(weight)$	2.0613 kg
$\frac{u(weight)}{\frac{NET}{GROSS}} \cdot 100$	0.076%
$\frac{u(weight)}{GROSS} \cdot 100$	0.040%

Table 3.11: Siwarex case D0

$u(amp)$	0.005946 mV
$u(cells)$	1.7893 kg
$u(weight)$	1.8804 kg
$\frac{u(weight)}{GROSS} \cdot 100$	0.078%

Some observation can be made:

1. The weight uncertainty is in every case, except the gain one, into acceptable ranges because it always has a value of a couple of kilograms. These values are acceptable because the scale is used for big input weights, like the one shown in the previous examples, and the relative uncertainty compared to the net weight is always less than 0.076%. This means that, in regular working conditions, the system is able to provide a reliable weight indication.

2. The only exception to the previous point is in the gain case where the weight uncertainty is way bigger even if the gain uncertainty is increased by only one point. This means that a gain uncertainty, due to a bad calibration, is something that must be avoided at all costs because it is the biggest contributor to uncertainty.

3. The uncertainty of the D1 indication is always bigger than the D0 one; this is due to the fact that some uncertainty contributions are relative to the actual value of a variable and, as a consequence, when the values are bigger because of a bigger output code, the uncertainty is bigger. The relative uncertainty, on the other hand, becomes smaller with an input weight increase; this happens because the uncertainty increase is not as big as the weight variation that provoke it. This means that bigger weight values have higher absolute uncertainty but lower relative uncertainty.

3.4.3 Error percentages

To see which contribution is the most relevant one all the uncertainties are compared to each other; these are the results:

- D1 with sensitivity uncertainty:

Table 3.12: D1 with sensitivity decay

uncertainty	percentage
ADC	0.98
GAIN	0.02
SENSITIVITY	69.00
CELLS	21.99
READING RESOLUTION	8.01

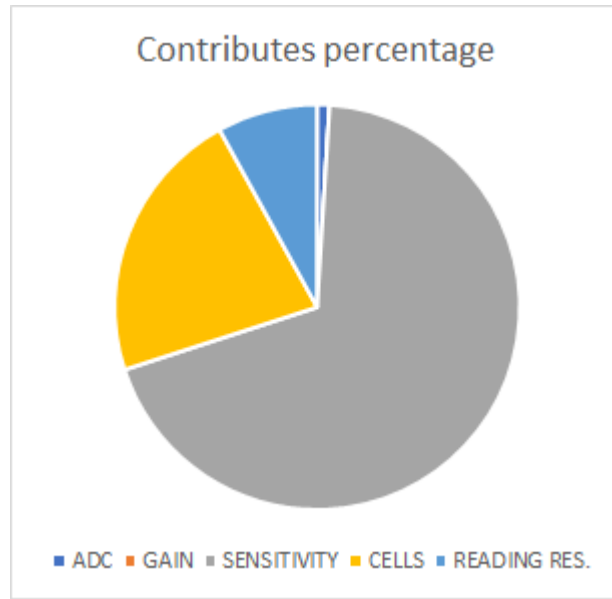


Figure 3.3: D1 with sensitivity decay graph

When sensitivity is taken into account then its contribution is the biggest of all (a similar consideration could be said about the excitation voltage but its uncertainty is not available). This means that this type of uncertainty could become very relevant if the sensitivity is allowed to decay after the calibration has already taken place. This is another reason why a periodical calibration is important.

- D1 in perfect conditions:

Table 3.13: D1 in perfect conditions

uncertainty	percentage
ADC	3.15
GAIN	0.05
CELLS	70.97
READING RESOLUTION	25.83

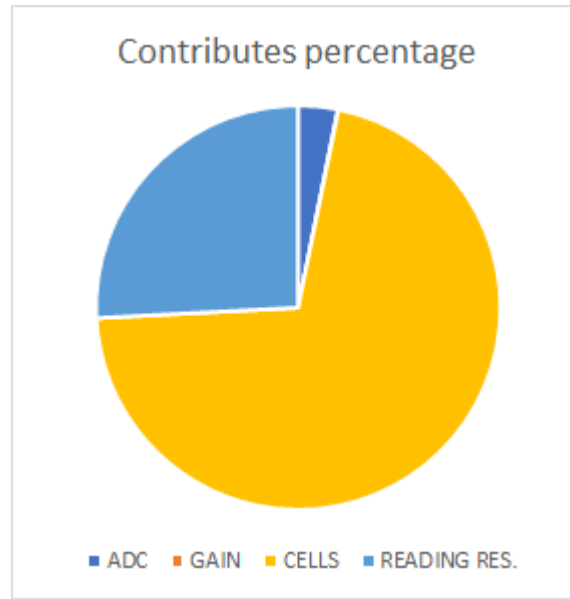


Figure 3.4: D1 in perfect conditions graph

In an optimal case, where neither the sensitivity nor the excitation voltage have decayed, the biggest contribution is brought by the cells, at 71%, and the reading resolution also contributes to about 26%. This means that, in an ideal case, the biggest uncertainty contributor is the cells system.

- D0 in perfect conditions:

Table 3.14: D0 in perfect conditions

uncertainty	percentage
ADC	7.06
GAIN	0.03
CELLS	35.01
READING RESOLUTION	57.90

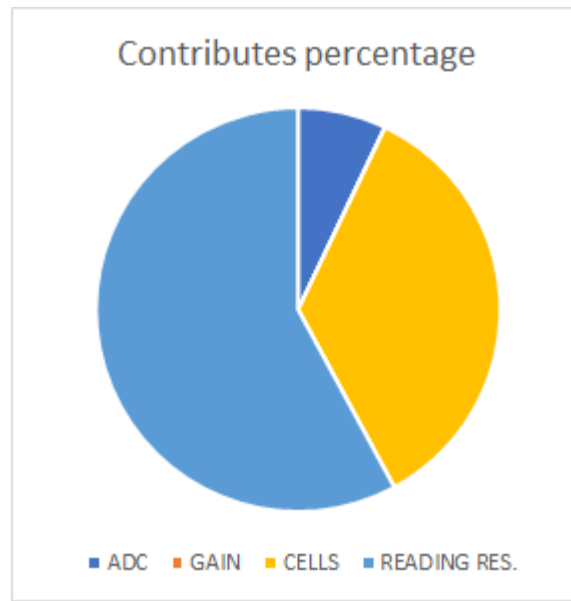


Figure 3.5: D0 in perfect conditions graph

In an ideal case, where only the tare weight is used, the biggest contribution is brought by the reading resolution. This is due to the fact that the reading resolution is fixed for every weight, because it depends on the verification scale interval value e (A.2.1), while the cells contribution is dependant on the input weight. The ADC contribution percentage is also increased for the same reason. This means that when the weight is small, the uncertainty is mainly given by the reading resolution and not by the propagation in the measuring system.

- Gain uncertainty:

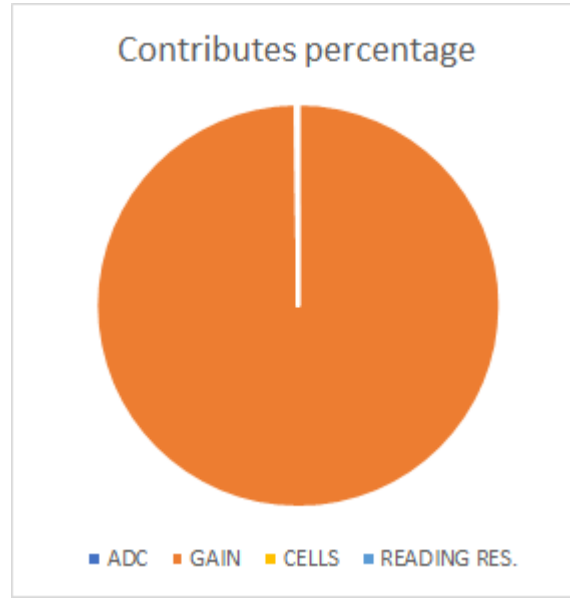


Figure 3.6: Gain error graph

In this case the gain uncertainty, which is the calculated gain in section 3.3 and not the actual amplifier gain, is increased by just one unit, a value used to simulate a possible error in the calibration procedure, and the result is that its contribution in the final uncertainty is more than 99%. This type of uncertainty demonstrates that a good calibration of the weighing module is very important to guarantee an acceptable weight indication.

3.4.4 Revamped propagation

An analysis on the revamped system can be performed by following similar steps to the current system. The problem is that several information about the amplifier and the analog to digital converter are missing. Still some assumptions can be made:

- The revamped and current systems have a similar calibration function because

the cells are the same in the two cases, the difference in junction boxes does not affect the function and the weight processing modules behave in a similar way, even if the revamped version is not detailed.

- The uncertainty propagation concerning the cells and the junction box system is the same because the junction box does not affect it, or at least its data sheet does not provide any information about it, and the cells are the same as before so their uncertainty contribution does not change.
- The uncertainty contribution given by the weight processing module is probably inferior in the revamped module compared to the other for two reasons: on the one hand the ADC has more bits than its counterpart, even if the exact number is not known, and as a consequence the ADC uncertainty contribution is expected to be lower because of the lower value of the quantisation error. On the other hand, as it can be seen in the 2.2 table, the module uncertainty is 0.05% FS in both devices but the expected full scale is less in the revamped module compared to the current one because the supplied excitation voltage is inferior.

From these points a conclusion can be made: the weight indication uncertainty in the revamped system is equal to or even less than the current system uncertainty when the measuring system has been correctly calibrated.

Chapter 4

Electrical scheme

In order to understand what changes the revamping process will bring, it is important to see how the PLC control system is structured in the electrical cabin. Both the current setup, which is actually used to control the AOD process, and the revamped setup, which will be used after the revamping procedure is completed, are analysed in this chapter.

4.1 Current scheme

The current setup can be divided into two main electrical cabins. The first one is the main AOD cabin and it is controlled by a CPU S7-319-3 PN/DP which handles the system control signals, including the ones concerning the weighing system, and it is connected to the pulpit and all the peripheral controllers through a Profibus connection or a Profinet connection (1.3). The second one takes care of the power modules, those that control the power supply of the system and the movement of the hoppers, and it is controlled by a CPU S5-943-7UB11. When the

working area must be secured, like during the weighing system calibration process (2.3.1), the modules that are sectioned are in this area. The fact that this section is controlled by an older S5 CPU is a huge drawback because it has worse technical parameters compared to an S7 system and, as it can be seen in the 5.3 paragraph, the integration between the two systems is complicated. The main CPU interacts with the other by sending it control signals so that the power section can activate the machinery accordingly; on the other direction, the S5 CPU transmits back a series of state signals that the main CPU can use to control the process.

This main system is connected to a series of other peripherals that are important to control the AOD process flow but are not a main focus in this document because they have nothing to do with the B17 hopper measuring system or movement control. An important connection still is the one to the AOD pulpit because it is where the operator can control the process through a HMI.

4.1.1 Connection network

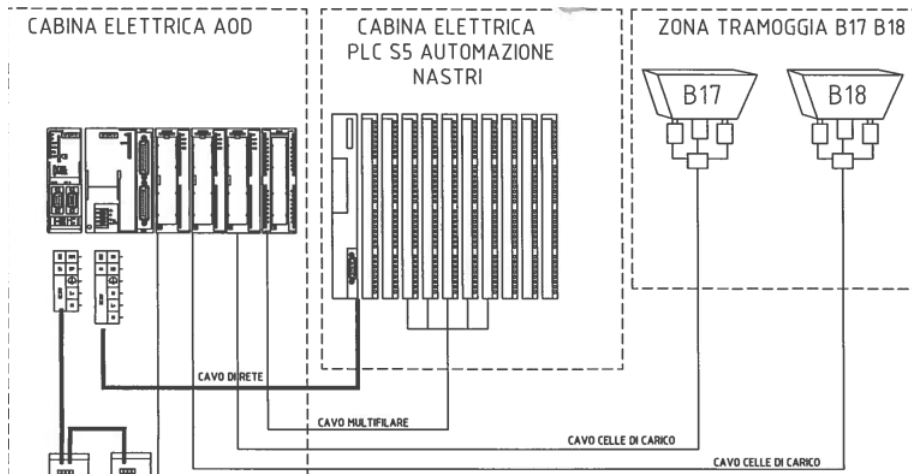


Figure 4.1: Current network scheme

The scheme 4.1 shows that the main CPU and the power one are both connected to the Profinet network through an ethernet cable. The two sections are also connected to each other with a point to point connection so that the data can easily be shared. The hoppers are connected directly to the Siwarex u module through the cell cables (2.2.3). The other peripherals, including the pulpit one, are all connected through a Profibus network.

4.1.2 Racks description

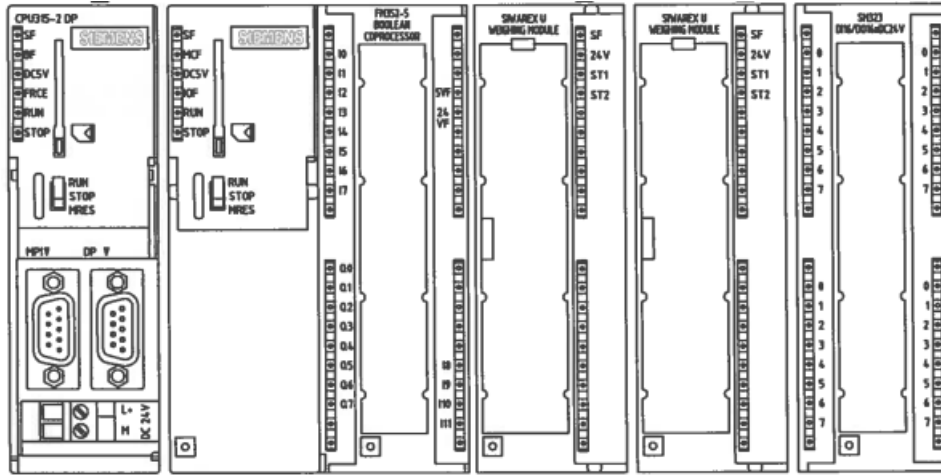


Figure 4.2: Main rack scheme

The main rack seen in figure 4.2 is composed by these modules:

- CPU 319-3 PN/DP: It is the module that is directly connected to the Profibus network; a more detailed description of this module is found at 4.3.
- Communication processor CP 343-1GX30-0XE0: It is used to connect the modules to the ethernet network through Profinet (1.3). The CPU already has a Profinet port but this module is inserted so that a network with more devices can be made.

- Signal module analog output: It is used to set the reference voltage of the silos extractors so that their vibrating speed and, as a consequence, the speed at which a material is dropped into the hopper can be controlled.
- Siwarex u modules: They are the weight processing modules described in the measuring system chapter 2.3.1. The first one has two input channels and it is used to acquire data from the B17, the studied scale, and the B18 hoppers. The second one as a single input channel and it is used to acquire data from the A15 hopper.
- Signal module digital input/output: The input bits are used to acquire state signals from the S5 PLC, to enable some emergency bits and to acquire some state signals from the lime controller, a peripheral that is not studied in this document. The output bits are all sent to the S5 PLC as control signals used for the hoppers. The connections between the main rack and the power rack that can be seen in the 4.1 figure are connected to this module.

The power controlling modules are inserted in a different electrical cabinet. The system is divided into three racks. The first one is composed by the S5 CPU, which is the controller of this section, and different digital input modules. The presence of two interface modules enables this rack to be connected to the other two racks. The second is composed by digital input and digital output modules while the third is composed by digital input/output modules. The signals are either used to exchange information with the main PLC (connected to the signal module digital input/output), to check the status of the machinery, when received as inputs by sensors, or to control and activate the machinery, when they are sent to actuators. As an example, some input bits are switched on when the hopper is in a particular

position or when the quantity of material inside a silo reaches a low level while some output bits are used to switch on the extraction of material from a silo or to activate the forward or backward movement of a hopper.

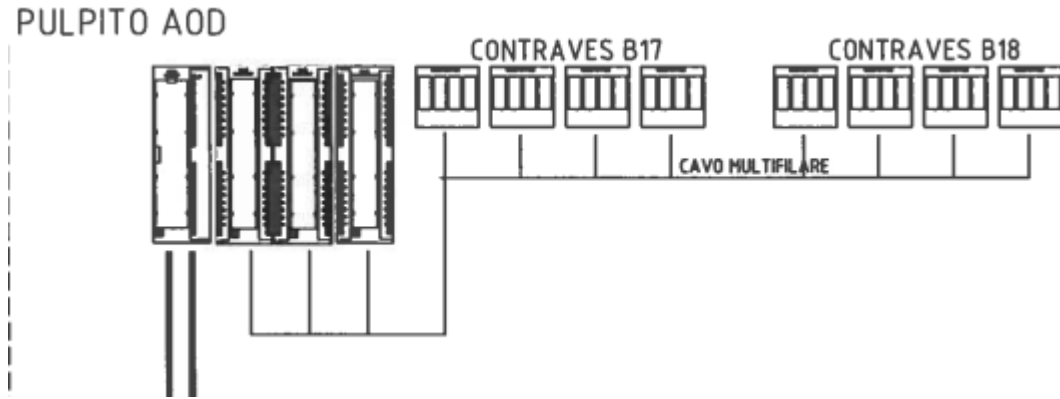


Figure 4.3: Pulpit scheme

The AOD pulpit (4.3) is controlled by an ET200M controller, a module that is used to connect the main CPU with peripheral input/output modules, and it is connected to the Profibus network (1.3). This rack is composed by both digital and analog input and output modules which are used to check external data and signals or control other sections that are not of interest in this study. An important input in this section is the "contraves" data which is used in the control of the hopper (5).

4.2 Revamped scheme

The revamped system is divided into several cabins, some of them contain the power cabinet and the power distribution cabinet while 6 other columns contain the PLC cabinet which is used to control the entire system. These other peripheral cabinet are connected through ET200 I/O controllers. Like in the current setup,

the different sections are connected to each other through Profibus and Profinet networks (1.3).

The first part of the scheme shows the power distribution cabinets; these contains the modules used to interact with the actuators and the sensors and also how the power is distributed to the entire system. Both of these are controlled by the PLC and are not studied in great detail. The last part of the scheme shows several peripheral modules which are used to control different part of the AOD process, only the peripherals relevant to the B17 hopper are studied. The central part of the scheme shows the PLC cabinet which is the one that contains the control modules of the system; most of the proposed user program input and output come from this section of the system (6). It must be noted that, as opposed to the current system, this one only have one CPU that controls the entire system (4.3.2).

4.2.1 Connection network

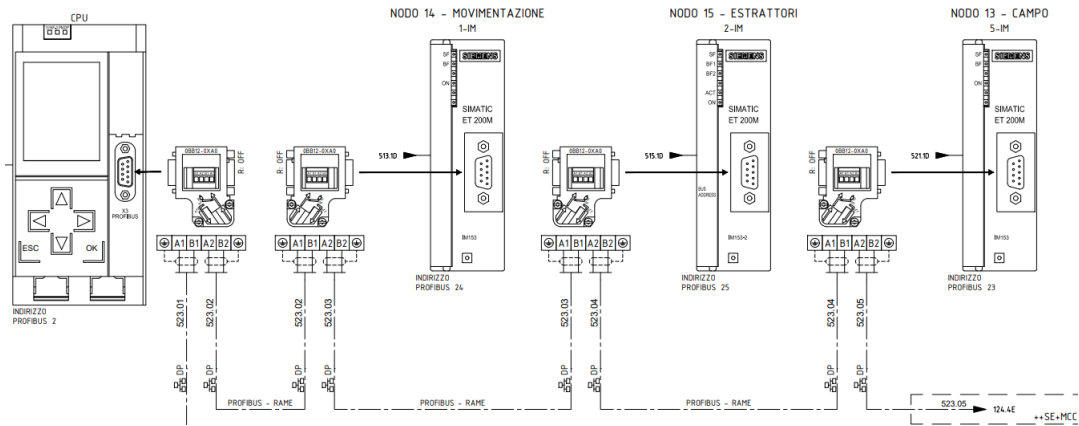


Figure 4.4: Revamped Profibus network

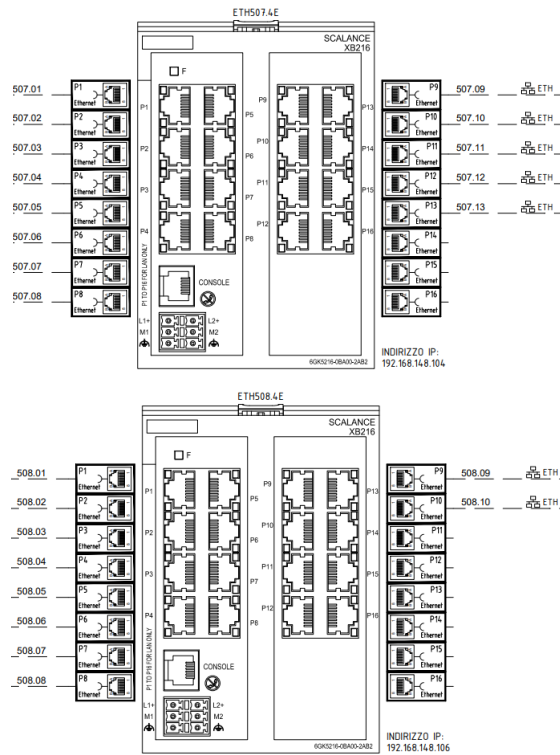


Figure 4.5: Revamped Profinet network

The CPU is linked through a Profibus connection (4.4) in chain with the rack 1, 2, 5 peripherals and also to other, non studied, peripherals, including the hoppers encoders and the remote controller for the silos.

The CPU is linked through a Profinet connection (4.5) to two switches that connect the entire ethernet network. The network is formed by all the local control panels, including the B17 one, the main operator panel that is found on the AOD pulpit, the PLC power supplier and to the rack 3 and 4 peripherals. It must be noted that, in this configuration, a communication processor is not needed because of the CPU superior technical parameters compared to the 319-3 ones (4.3.2). For this reason the Profinet cables are connected directly to the 1516F CPU. An easier to read network representation can be seen in the 6.2 figure.

In contrast to the current system network configuration, this one as a lot of peripherals linked through a Profinet connection. As it can be seen in the 1.3 paragraph a Profinet network can share a bigger quantity of information in less time so the revamped system is more performing also for this reason.

4.2.2 Racks description

The CPU, which is located into the first column, controls five peripherals contained in as many columns; each one of them has a specific purpose.

1. Node 14 - movement: connected to the main CPU with a ET200M peripheral controller through a Profibus connection. The rack contains digital input modules, digital output modules and analog input modules. The digital inputs all come from the power cabinet and power distribution cabinet; these are signals of power suppliers, actuators, protection switches and general switches of other columns. The analog input have different motors currents. The digital

outputs are all reserved. These signals are all used to control the power supply and power distribution of the system.

2. Node 15 - extractors: connected to the main CPU with a ET200M peripheral controller through a Profibus connection. The rack contains digital input modules, digital output modules, analog input modules and analog output modules. The digital inputs receive status signals and feedback signals of the extractors and hoppers motors. The digital outputs are used to control the status of silos and hoppers extractors through a vibration board; also they activate the silo from which material have to be extracted. The analog inputs have different extractors motor current and the vibration boards current. The analog output contains the vibration boards set point which is used to control the speed at which a material is extracted from a silo.
3. Node 16 - safe: connected to the main CPU with a ET200SP peripheral controller through a Profinet connection. This type of controller enables the use of fail safe modules which increase the system reliability. The rack contains fail safe digital input modules and fail safe digital output modules. The inputs receive some sensors status, different enabling feedback signals and different emergency buttons signals. The outputs control different enabling signals for motors, actuators and general switches. To increase reliability, several signals have redundant connections. The main purpose of this rack is to collect all the safety related signals so that the machinery can be safely interacted with and work in a reliable way.
4. Node 11 - emergencies gates: connected to the main CPU with a ET200SP peripheral controller through a Profinet connection. As said before this type

of module enable the use of fail safe modules. The rack contains fail safe digital input modules, fail safe digital output modules and the Siwarex WP321 (2.3.2), the weight measuring modules explained in the 2 chapter. The digital inputs are received from emergency buttons and other safety related signals; to increase reliability many of these signals have redundant connections. The outputs are all signals that control the access to the gates. The signals with which this rack interacts are all safety related except the weightings data taken from the A16, B17 and B18 hoppers which are fundamental for the AOD weighing process.

5. Node 13 - field: connected to the main CPU with a ET200M peripheral controller through a Profibus connection. The rack contains digital input modules, digital output modules and analog input modules. The digital inputs are signals coming from the peripheral controllers like the local control panels and the local power supply switch. The digital outputs are enabling gates signals and movement signals. The analog inputs are temperature data of other cabins and other machinery. This rack main purpose is to exchange signals with the field peripheral controllers.
6. Local control panel "B17-SILI_B" (100): it is not one of the main racks and it is used to manage the local control panel of the B17 hopper. It is composed of two HMI button panels that control the left right movement, the extractors vibration and the hopper vibration. It also has a ET200M peripheral control system and some digital input modules, used for local control mode, and a digital output module, used to activate some signalling lights.

4.3 S7 CPU description

This section contains a description on how a generic S7 CPU works and some technical data of the CPU studied in this project.

The memory area is usually divided into three parts:

1. Load memory: It is a non volatile memory and it is stored in a removable micro memory card. It contains all the code blocks, data and system configurations including the hardware, network connections and units parameters. A code block that is not actually used in the user program it is still saved in this memory. For the CPU to work the external micro memory card with the load memory must be inserted.
2. Work memory: It is a volatile memory integrated into the CPU and cannot be extended. It contains the code and data block that are actually used during the user program run time and it is useful to the program elaboration. In more modern CPU this memory is split into code work memory and data work memory.
3. System memory: It is integrated in the CPU and cannot be expanded. It stores the process images (4.3.1), the merker, counters, timers and the local data (like temporary variables). This memory has both a volatile area and a non volatile area where specific operands could be stored. As an example, usually merker operands are reset to '0' if the CPU is switched off and then switched back on but if the merker is saved in a retentive area then its value is maintained. The data blocks are usually saved in the retentive area of this memory but they could also be saved in the volatile area if the programmer chooses so.

When the program is elaborated only the system and work memory are used. The retentive memories are deleted only with a memory reset command or a reset to factory settings.

4.3.1 Process images

If in the user program an operand is addressed, in an input or output area (E, A), then the actual state of the signal is not interrogated but the image process, which is divided into input and output image process, is accessed. Using this method is a great advantage because it creates a coherent version of a signal that can be used during the entire cyclical elaboration of the program; this means that if a state changes during the program elaboration the process image would remain the same until the next image update. Furthermore accessing the process image is quicker than accessing directly the input/output unit because it is stored in the CPU system memory. The direct access to a unit could still be done by using the command "P" in the user program; this could be wanted if the accessed input/output is not stored in the process image because its dimension is not big enough.

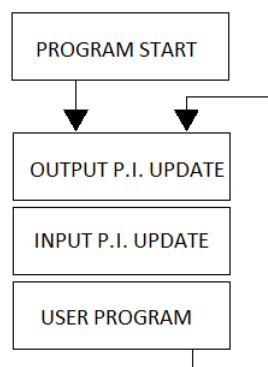


Figure 4.6: Program cycle

The output image process and the input image process are updated at the beginning of the cycle as shown in figure 4.6, after this the user program is executed using the data stored in the process images.

4.3.2 Confrontation between current and revamped CPU

Table 4.1: CPU confrontation table

	319-3 PN/DP	1516F-3 PN/DP
Supply voltage [V]	24	24
Load memory	8 MB	32 GB
Work memory [MB]	1.4	6
Bit op. processing time [ns]	10	10
Word op. processing time [ns]	20	12
Fixed p. op. processing time [ns]	20	16
Floating p. op. processing time [ns]	40	64
Total number of blocks (DB,FC,FB)	4096	8000
Maximum size of blocks [kB]	64	512
I/O process image dimension [kB]	2	32
Maximum module per rack	8	32
Typical power loss [W]	3.5	7
Fail safe mode	NO	YES

The 4.1 table shows some of the most relevant technical parameters. The bigger memories permit the revamped CPU to handle more modules; this is why it does not need any other CPU to handle the power modules section. The processing time of the operations is always faster in the revamped CPU except for the floating point operations which, however, are not used as much as the other types so the overall processing time of a program cycle is usually shorter. In order to have a reliable setup a CPU must be able to handle faults in a safe way; this means that, in case of fault, the system must have enough resources to go into a functional state. The 1516F CPU has a fail safe mode which means that it can handle demanding

safety standards. Thanks to this mode, standard program and fail safe program can coexist in a single CPU so that the fail safe data can be processed by the standard user program thus increasing the system reliability. Thanks to the display, which is not present in the 319 CPU, some network configurations can be modified on the spot in the revamped CPU and fault creating error can be directly analysed on site reducing downtime. Because of the CPU different performances the 1516F has a typical power consumption higher than the 319.

Chapter 5

Current program

The weighing system is managed in the main CPU program cycle by a series of function and data blocks. While only the B17 hopper blocks are analysed, the other hoppers are programmed in a similar way. The program cannot directly handle the movement of the hopper because of the division between the S7 CPU and the S5 CPU (4.1); this disadvantage is overcome in the revamped program because the different parts are managed by the same CPU (4.2). The code is written using the German mnemonic (E A for input and output).

The types of block seen in this chapter are:

- Data block: it is a type of block that can contain different types of simple data, like boolean or integer values, or structures, like arrays. The information is saved in the non volatile part of the memory (4.3) so it is retained even after a stop. A data block could be global, so that every part of the program can access its information, or an instance of a function block, so that only the chosen block can use its data like local variables.

- **Function:** it is a type of block that contains a part of the user program. It can use all the global data but it does not have any instance data block. It can be called by other blocks to create a program structured in a hierarchical way.
- **Function block:** it is a similar type to the one just described, the difference is that it can have an instance data block; this means that it can have local static data that is retained.
- **Organisation block:** it is a type of that organises the program; it is usually used to handle exceptions and interruptions. The most important is the OB1 "main program", which contains the user program that is executed in the main cycle.

The used blocks are described in the AWL (instruction list), KOP (ladder logic) or SCL (structured control language) languages.

5.1 Data insertion

The information needed to perform a sequence of weightings can be entered in two ways. The data of the actual steel composition and the target steel composition could be directly analysed by the operator and the sequence of weightings to be done can be manually inserted for every addition through an interface in the pulpit. Another way is for the steel composition data to be analysed by a software called CSM which is able to generate a series of weighing sequences so that the target composition can be reached.

This means that the system can operate in manual mode or automatic mode. The manual data is directly stored in the "DB_SequenzaPesateB17" data block

while automatic data is stored in "NrTipoCodiceMateriale" and only afterwards it is transferred to the "DB_SequenzaPesateB17" DB in the third segment of the FB17 (5.4). The bits that signal which mode is being used are stored in the "SEND_COMANDI_HMI_S5" data block.

5.2 DB17 "DATI.B17"

The DB17 is the global data block that contains the most important information about the B17 hopper.

- "peso_siwarex", INTEGER: it is used to store the weight value taken by the Siwarex U module.
- "valore_contraves_1_dec", INTEGER: it is the target value set for every charge (1, 2, 3, 4) in integer format.
- "valore_contraves_1_bcd", WORD: it is the target value set for every charge (1, 2, 3, 4) in BCD format.
- "valore_peso_impostato", INTEGER: it corresponds to the "contraves" value of the charge in progress.
- "prima_pesata" (seconda, terza, quarta), INTEGER: it is used to memorise the "pesata_in_corso" variable when the corresponding charge (1, 2, 3, 4) is active.
- "silo_1_pesata", INTEGER: it contains the number of the silo (from 1 to 12) corresponding to the charge (1, 2, 3, 4).

- "soglia_tara", INTEGER: it contains the constant value 15 and it is used in the tare function (5.5).
- "peso_calcolato", INTEGER: it is used to store the net weight value on the hopper.
- "peso_tara", INTEGER: it is used to store the value of the tare.
- "peso_start_materiale", INTEGER: it is the net weight of the hopper at the start of a new charge; it is used to eliminate the other charges contribution.
- "pesata_in_corso", INTEGER: found thanks to the "peso_start_materiale" variable, it contains only the weighing in progress charge.
- "somma_contraves", INTEGER: it is the sum of the "contraves" values.
- "somma_pesate", INTEGER: it is the sum of the weightings "prima/seconda/terza/quarta_pesata".
- "puls_tara", BOOL: not used. Some variables present in this data block are not used in the actual program. This is probably due to the fact that these variables were used in older version of the program but now they no longer serve their purpose.
- "abilitazione_contraves", BOOL: not used.
- "liv.minimo", BOOL: it indicates if the weight is under the minimum value.
- "liv.massimo", BOOL: it indicates if the weight is under the maximum value.
- "peso_raggiunto", BOOL: it indicates if the target weight on the hopper has been reached in the weighing in progress.

-
- "rallentamento", BOOL: it indicates if the slow down weight on the hopper has been reached in the weighing in progress. When this bit is set the speed at which material is dropped is modified.
 - "superamento_portata", BOOL: it indicates if "somma_contraves" is over the hopper range.
 - "one_shot_up_marcia", BOOL: not used because the tare function (5.5) already uses a static variable.
 - "appoggio_one_shot_up_marcia", BOOL: not used for the same reason as the above.
 - "peso_stabile", BOOL: not used.
 - "pesata_in_corso", BOOL: it is set when the silos extractor is activated.
 - "rif_velocità_x_vibra", INTEGER: it contains the reference voltage for the silos extractor; depending on its value the speed at which material drops is changed.
 - "rall_pesata_in_corso", INTEGER: it contains the slow down weight depending on the material of the weighing in progress.
 - "silo_pesata_in_corso", INTEGER: it indicates the number of the silo used in the weighing in progress charge; its value correspond to one of the "silo _1 _pesata" (1, 2, 3, 4) variable.

5.3 Other data blocks

Besides the DB17 other data block are used in the program. The program is divided into a S7 part and a S5 part because of the PLC structure of the system (4) and for this reason a way of communicating between the two must be implemented. Some of the data blocks, the one that contain "S5" in their name, are used to store information that must be sent to or received from the S5 section of the program.

DB221 - "NrTipoCodiceMateriale": it contains some important information about the materials; some of them are constant, like the name and the slow down weight, while other are set by the CSM software (5.1), like the proposed quantity of a material.

DB300 - "DB_SequenzaPesateB17": it is the fundamental data block that is manipulated by the FB224 (5.6). It is used to store important information about the weighing in progress.

DB215 - "RECEIVE_STATI_S5_HMI": it receives information from the S5 section that can be controlled by the HMI.

DB151 - "RECEIVE_S5": it receives information from the S5 section about the status of the machinery, like if it is in movement or if the extractors are running.

DB150 - "SEND_S5": send to the S5 section some information about the weighing process.

DB200 - "SEND_COMANDI_HMI_S5": it contains some controls that can be activated through the HMI.

5.3.1 User defined types

The information in these data blocks is organised in structures which are created with the use of user defined type. Because of this organisation the data of more complex data blocks can be accessed more easily. The user defined data that are important to the weighing process are used in "DB__SequenzaPesateB17" and "Nr-TipoCodiceMateriale". These structures are exploited in the FB224 (5.6) function to manage the weighing sequence.

- "DatiSilo" is used to store the data relative to a silo; it contains constant information like the number of the silo, the type of the material, the slow down weight and also the data relative to the automatic weighing process like the CSM variables (5.1).
- "CaricaTramoggia" is used to store the data of a single sequence; in the DB300 case it is used for the weighing in progress data.

The "Dati" part of the structure stores for every charge the number of the selected silo in "NSiloSelezione" and the corresponding weight to be charged is stored in "KgSelezione".

Other parts of the structure, namely "Indice", "Destinazione" and "Stato", are used to control the automatic weighing cycle. In particular "Stato" indicates the status of a sequence:

- 0: To be done
- 1: Forced
- 2: Disabled
- 3: In execution
- 4: Done

This structure is similar to the "Sequenza" data type which is used in other data structures; in the part of the system under study they serve the same purpose and their description can be seen as the same.

- "CaricheTramoggia" is used to store the data of several sequences; in the DB300 it is used for the hopper series of sequences data.

The "Sequenza" part is an array of 40 "Sequenza" data type (the variable name is equal to that of the data type but the two should not be confused).

The "CaricaModifica", "CaricaAttuale" and "CaricaManuale" structures are all of "Sequenza" type and are used to store the data of a sequence to modify the sequences array, the charge in progress data (not to be confused with the weighing in progress data which is stored in the "CaricaTramoggia" type variable) and the data of the sequence of the manual mode.

The "Stati" part of the structure is used to store a series of indexes and a series of status bits used in the FB224 (5.6) block.

The "Comandi" part of the structure is used to store a series of control bits used in the FB224 (5.6) block.

5.4 FB17 "gestione__peso__B17"

This function block is the main one that handles the weighing process of the B17 hopper. Even if it is a function block it does not need an instance data block because local static variables are not used. It is directly called by the "main program" OB1. It is composed by 14 segments. It is written both in KOP and AWL languages. The segments description is the following:

1. It is used to save the weight of the Siwarex module in the corresponding DB17 variable. The value is accessed directly from the port and not from the process image (4.3.1), in fact the input port is written with a P. This is done because the process image is not big enough to contain the accessed area.

```
L PEW 274
```

```
T "DATI_B17".peso_siwarex
```

2. The calculated weight is saved in the S5 data block so that it can be transmitted to the S5 section of the program.

```
L DATI_B17.peso_calcolato
```

```
T SEND_S5.PESO_REALE_B17
```

3. Some data saved in the DB221 by the CSM software (5.1) are transferred in the corresponding spot of the DB300. The transferred information is the quantity of proposed material, called "totale_CSM", the percentage between proposed and set, called "tolleranza_CSM", the quantity of material of the charge in progress, called "totale_scaricato", and the priority of the silo. The segment is not very efficient because every one of the described transfers is performed with a single MOVE operation for every single silo; this produce a total of 48 operations.
4. The FB224 is called here (5.6). Most of the boolean inputs are linked to signals received from the S5 CPU and are saved in the "Receive_S5" data block except "FinePesate", which is a one shot merker, and "PesoMinimo", which is linked

to the "Dati_B17" data block. The integer inputs are either linked to the "Dati_B17" data block or are directly inserted as constants. The in/out parameters are either status bits stored in the "SEND_COMANDI_HMI_S5" data block or are more complex structures saved in the "DB_SequenzaPesateB17" data block.

5. The quantity of material of the charge in progress is located in the DB300. This section transfers the selection to the "contraves" variables of the DB17 in integer and BCD format. This is done for the all the charges (1, 2, 3, 4).

```
L DB300.GestioneCariche.CaricaAttuale.Dati:KgSelezione[1]
T DATI_B17.valore_contraves_1_dec
ITB
T DATI_B17:valore_contraves_1_bcd
```

6. The just saved "contraves" values are summed in the corresponding DB17 spot.

```
L DATI_B17.valore_contraves_1_dec
L DATI_B17.valore_contraves_2_dec
+I
L DATI_B17.valore_contraves_3_dec
+I
L DATI_B17.valore_contraves_4_dec
+I
T DATI_B17.somma_contraves
```


7. The just found sum is compared to the hopper range, which is represented by the constant 2950, and if the value is bigger then the "superamento_portata" bit in the DB17 is set.
8. The data of the weightings is summed together in the DB17.

```

L DATI_B17.prima_pesata
L DATI_B17.seconda_pesata
+I
L DATI_B17.terza_pesata
+I
L DATI_B17.quarta_pesata
+I
T DATI_B17.somma_pesate

```

9. This segment is divided into two parts. In the first part a timer is set when a signal is received from the DB151 that indicates the start of a new weighing process. If a timer is able to switch on, after 3.2 seconds, then the corresponding second part is activated otherwise no operation is performed. This is done for the group I, II, III, IV using the timers from 1 to 4.

```

U RECEIVE_S5.I_GRUPPO_CONTR_B17_AUTO
L S5T#3S200MS
SE T 1      //timer set
U T 1      //timer interrogation
SPB M001    //jump to the corresponding second part

```

The second part does several things: it saves the silo of the weighing in progress, located in the DB300, to the DB17 silos variables and it saves the variable "valore_impostato" of the DB17 by subtracting 5 to the "contraves" value of the charge in progress. Finally, the timer T8, which is set up in the tare function FB21 (5.5), is interrogated. If its value is set then the program skips to the end of the segment, otherwise the weighing in progress is saved into the DB17 and the DB300.

```
U T 8          //timer interrogation
SPB M006       //jump to end
L DATI_B17.pesata_in_corso
T DATI_B17.prima_pesata
T DB300.PesateAttuali.Dati.KgSelezione[1]
SPA M006       //jump to end
```

The timer is used so that the weight can still be saved and updated even after the silo extractor has stopped vibrating. Every charge (1, 2, 3, 4) has its own part in this segment and, because of the fact that the program is written using jumps, only one charge handling is performed at a time so that different charges do not overload one another.

10. Some signaling outputs, whose values are stored in the process image (4.3.1), are updated using the DB17 data; these are the minimum weight indication, the maximum weight indication, the target weight reached and the slow down weight reached.

-
11. This segment is empty.
 12. In the first part of this segment a one shot merker, which is set only for one cycle, called "OS_DOWN_B17_MARC" is activated when the hopper goes into a specific position, signaled by two bits in the "RECEIVE_STATI_S5_HMI" data block. The merker is also activated when the bit called "B17_IN_MARCIA" from the "RECEIVE_S5" data block have a negative front, signaling that the hopper has been shut down.
 "OS_DOWN_B17_MARC" is immediately after interrogated, if its status is '0' then no operation is done, if its status is '1' then the weightings relative data and the "peso_raggiunto" signal in the "DATI_B17" data block are reset.
 13. The FB21 which is the tare calculation function is called here (5.5) along with its instance data block DB21. A variable is assigned to every interface parameter. The input variables are used in read mode in the function without being modified.
 - "Peso_siwarex" is linked to the Siwarex weight measure stored in the "DATI_B17" data block.
 - "Soglia_tara" is linked to the 15 constant value saved in "DATI_B17".
 - "valore_contraves_dec" is linked to the "valore_impostato" variable of the "DATI_B17" data block which is written in the ninth segment of the FB17 block.
 - "Set_rall" is linked to slow down weight of the charge in progress material stored in "DATI_B17".rall_pesata_in_corso. This variable is different for every material and its value is located in the "NrTipoCodiceMateriale"

data block. The access to the wanted value is complicated because the data block must be addressed through a pointer variable. This means that another function is used to calculate the address, fetch the slow down value and save it to the "rall_pesata_in_corso" variable of the "DATI_B17" data block.

- "start_vibro" is linked to a signal received from the S5 CPU and it indicates if an extractor is vibrating.
- "pulsante_tara" is linked to a signal that is sent to the S5 CPU. It comes from a HMI signal and it indicates if the tare button has been pushed.
- "abil_contraves" is not used and therefore is not linked to anything.
- "Tramoggia_in_movimento" is linked to a signal received from the S5 CPU and it indicates if the hopper is moving.

The output variables are written in the function and read in other parts of the program.

- "liv_min" is linked to the corresponding "DATI_B17" variable. It indicates if the lower limit of the scale is reached.
- "liv_max" is linked to the corresponding "DATI_B17" variable. It indicates if the upper limit of the scale is reached.
- "Peso_raggiunto" is linked to the corresponding "DATI_B17" variable. It indicates if the wanted weight of the charge in progress has been reached.
- "Rallentamento" is linked to the corresponding "DATI_B17" variable. It indicates if the slow down weight, inserted as one of the function input, has been reached in the charge in progress.

- "Superamento_portata_tram" is not used and therefore is not linked to anything.
- "PesoStabile" is not used and therefore is not linked to anything.

The in/out variables come from other parts of the program and are modified inside the function.

- "Peso_calcolato_tramoggia" is linked to the corresponding "DATI_B17" variable. It stores the net weight on the hopper.
- "Peso_tara" is linked to the corresponding "DATI_B17" variable. It stores the tare weight of the hopper.
- "Peso_start_materiale" is linked to the corresponding "DATI_B17" variable. It stores the net weight on the hopper at the start of a new charge.
- "Pesata_in_corso" is linked to the corresponding "DATI_B17" variable. It indicates the net weight of only the charge in progress.

14. The "RECEIVE_S5".I_GRUPPO_CONTR_B17_AUTO (I, II, III, IV) is interrogated and depending on which one is activated the silo number data of the charge in progress stored in the "DB_SequenzaPesateB17" data block is saved in the "NSiloSelezioneAttualeB17" variable of the "SEND_S5" data block.

5.5 FB21 "peso_tara_rall_b17"

This function block is used to calculate several important data relative to the weighing in progress and it sets and resets some of the boolean values used to control the process. Thanks to its parametric structure the block can be used

in different parts of the program; as a matter of fact it is used in every hopper managing block. In the B17 case this block is called in the fourth segment (5.4). Because of the block use of static variables, like the ones used to support calculations, an instance data block that can store them must be called along with the function.

5.5.1 Interface and local variables

In order to be utilised in different functions this block has different interface variables.

INPUT: "Peso_siwarex", "Soglia_tara", "valore_contraves_dec", "Set_rall",
"start_vibro", "pulsante_tara", "abilContraves", "Tramoggia_in_movimento"

OUTPUT: "liv_min", "liv_max", "Peso_raggiunto", "Rallentamento",
"Superamento_portata_tram", "PesoStabile"

IN/OUT: "Peso_calcolato_tramoggia", "Peso_tara", "Peso_start_materiale",
"Pesata_in_corso"

Because of the fact that this is a function block local static variables can also be used. The local variables, as well as the interface parameters, can be distinguished from the global values because they are accessed with a `#` sign.

5.5.2 Segments description

The block is composed by eight segments.

1. A one shot up signal, which is a local static variable called "o_s_pulstara", is created for the "pulsante_tara" input; this means that the variable is active only for one cycle after a positive front on the input. The one shot variable

can be at '1' only for one cycle after the normally closed contact connected to the supporting variable switches it off.

2. A one shot up variable called "o_s_up_marcia_vibro" is created for the "start_vibro" input using the same technique as the first segment.
3. The first half of this segment is used to define the tare value. The tare is calculated every time that a series of conditions are met as

```
L #peso_siwarex
T #peso_tara
```

the tare weight is directly taken from the Siwarex value. This operation is done only when:

- the calculated net weight on the hopper is inside the tare threshold of 15 (saved in the "soglia_tara" variable).
- the one shot up signal of the tare button is on or the one shot up signal of the extractor vibrator is on.

This means that the tare is recalculated when the tare button is pressed but also when the vibration of the silos extractor is turned on. This means that the tare value could potentially change every time that a new sequence of weightings is started. This type of approach has both advantages and disadvantages. An advantage is that if the weight calculated by the measuring system (2) fluctuates for some reason, for example because the hopper is not completely empty at the start of a new charging cycle or because the

environmental factor have changed so much that the weight indication for the same value has changed, then the weighing process can still be performed with an acceptable degree of uncertainty because the calibration function of the Siwarex module is translated without being modified and the net quantity of material in the hopper can still be calculated. A disadvantage is that this mechanism, for long periods of time, could lead to a situation where the weight indicated by the Siwarex module, and therefore coherent with the calibration function, is significantly different from the weight measure used by the system. A case like this implies that the calibration function of the instrument is no longer valid and that the weight used in the system is probably not a reliable indication of the real value. This eventuality is prevented by a periodical instrument calibration (2.3.1).

The second half of the segment is always performed and it is used to find the net weight on the hopper by subtracting the tare to the gross weight:

```
L #peso_siwarex
L #peso_tara
-I
T #peso_calcolato_tramoggia
```

4. At the start of a new charge, when the "o_s_up_marcia_vibro" signal is activated, the "rallentamento" and "peso_raggiunto" bit are reset and the starting net weight on the hopper is saved in the "peso_start_materiale" variable.

The second part of the segment is used to set the "peso_raggiunto" bit. At

first the weight of the charge in progress is saved in a static variable called "appoggio_sottrazione" by subtracting the charge starting weight with the net weight; subsequently the "peso_raggiunto" bit is set when the calculated weight is higher than the target weight in "valore_contraves_dec" except if the "o_s_up_marcia_vibro" is activated to avoid conflict with the preceding charge.

The third part of the segment is used to set the "rallentamento" bit. At first the weight threshold which activates the bit is calculated and saved in a static variable called "appoggio_rall" by subtracting the slow down weight saved in "set_rall" to the target weight in "valore_contrave_dec"; then the "rallentamento" bit is set if the weight of the current charge is higher than the calculated value.

The final part of the segment is used to save the weight of the charge in progress to the "pesata_in_corso" variable. To do this the timer "T8", which is also interrogated in the ninth segment of the FB17 block (5.4), is initiated as a delay of the negative front of the "start_vibro" input. For as long as the timer is on, so for 3 s after the extractor has stopped vibrating, the "appoggio_sottrazione" variable is saved in the "pesata_in_corso" output. The timer purpose is to keep saving the weight on the hopper even after the end of the extraction so that an accurate measurement can be made.

5. It reset the "peso_raggiunto" bit when the hopper is moving. This is useful because when the hopper is in movement no weighting is in progress and the bit that indicates that the target weight has been reached is no longer of use.
6. This segment calculates when the range of the scale has been surpassed but

this is not useful because the same operation is already performed in the seventh segment of the FB17 block (5.4).

7. The net value of the hopper is compared to the range of the scale, which is 2950 in the B17 case, and if it is bigger than the "liv_max" bit is set.
8. The net value of the hopper is compared to the minimum value of the scale, which is set to be 3, and if it is smaller the "liv_min" bit is set.

5.6 Weight managing function FB224

This function is called in the fourth segment of the FB17 block (5.4) and it is written in SCL. The weightings are organised in a structure stored in the "DB_SequenzaPesateB17" data block. A single weighting process is composed of a series of up to 10 sequences. A sequence is a series of up to 4 charges where a charge is the weighing of a single material. The process is therefore composed of a series of castings of charges of different materials into the melted steel (5.1).

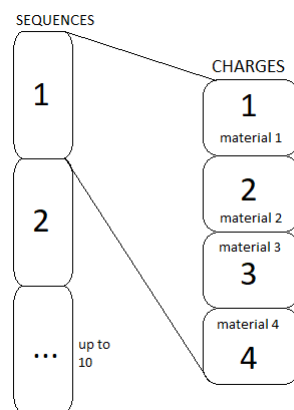


Figure 5.1: Sequence description

The overall purpose of the block is to organise this series of sequences into a specific order and to transfer the data of the charge in progress to the rest of the program.

Like the tare calculation function, this block has a parametric structure so that it can be called in different part of the program. This function uses temporary variables, which are a type of local variable that is not saved after the block execution is concluded, and static variables, which need an instance data block to be stored and their value is saved even after the block execution is concluded. The temporary variables are mainly used to control the indexes of cycles.

5.6.1 Interface and local variables

Like in the other parametric function these variables can be accessed with the # sign. The interface parameters are divided into input and in/out.

INPUT: BOOL

- "FinePesate": it indicates when the weighing process has ended.
- "PesoMinimo": it indicates if the scale weight is below the minimum limit.
- "CicloAttivo": it indicates the status of the automatic weighing cycle.
- "Scaricamento": it indicates if the material in the hopper is being dumped.
- "NavettaMovimento": it indicates if the hopper is moving.
- "EstrattoriSiliMarcia": it indicates if the silos extractors are vibrating.
- "NavettaPosizioneScarico": it indicates if the hopper is positioned in the dump area.

INTEGER

- "PesoTramoggia": it is the net weight on the hopper.
- "LimitePesata": it is the limit of weight that can be loaded on the hopper in one time.
- "NSequenze": it is the number of sequences and its value must be lower than 40.
- "NCariche": it is the number of charges in a sequence and its value must be lower than 5.
- "IndicePesataAttuale": it is the index of the sequence in progress.
- "ValorePesataAttuale": it is the weight value of the sequence in progress.

IN/OUT: BOOL

- "AutomaticoNavetta": it indicates if the hopper is in automatic mode and it is not on at the same time as the next parameter.
- "ManualeNavetta": it indicates if the hopper is in manual mode and it is not on at the same time as the previous parameter.
- "StartCicloCarico": it is used to control the start of a new automatic cycle and it is closely linked to the "CicloAttivo" input parameter.

USER DEFINED DATA (described in the UDT section (5.3.1))

- "DatiSilo": it is an array of 12 elements and it contains the data of every silo.
- "PesateAttuali": it contains the information about the weighing in progress and its type is "CaricaTramoggia".

- "Tramoggia": it contains the information about the sequences and its type is "CaricheTramoggia"; it is the variable that is most interacted with in the function program because it contains the command bits used to control the function flow and the sequences description used for the automatic weighing process.

The static variables are used for different purposes. Some of them are indexes and some of them are used to create one shot up signals like in the first segment of the tare function (5.5). The other statics are used together with the "Comandi" signals of the "Tramoggia" input; the inputs data is transferred to the static variables so that they can control the program flow for only one cycle, e.g:

```
ResetSp := Tramoggia.Comandi.Hmi.ResetSetPoint;  
Tramoggia.Comandi.Hmi.ResetSetPoint := 0;
```

In the example the static variable "ResetSp" that is used in the block code is set to the value selected by the input and then the signal resets to "0" so that the same operation is not done in two consecutive cycles unless a command is given from outside the function block.

The temporary variables are indexes or values used in FOR cycles.

5.6.2 Code description

The code can be divided into these sections:

- Enabling sequence calculation. The sequence calculation part of the code is enabled if the "Stato" bit of every sequence is not 3 or 4. If the status is 3 or 4 it means that the weighing is already in execution or it has already been done; this means that the sequence has already been calculated and

there is no need to enable again its calculation. In this section the control bit "Tramoggia.Stati.Bit.SequenzaSelezionata" used to see if a sequence has been selected for an automatic weighing cycle is written based on the "SelSequenza" static variable. The total amount of material that must be cast in the to be calculated sequence is also found here and saved in the temporary variable "TotaleAux".

- Sequence calculation. This part is done only if the sequence calculation is enabled.

At first every element of the "Tramoggia.Sequenza" array is reset with the use of a for cycle then a multi layered cycle begins.

The outer layer consists of a for cycle that scans the silos priority from 1 to 12 with a temporary variable index.

In the inner layer, a for cycle is tasked with finding the silo number that corresponds to the selected priority; if the found silo CSM total weight is not "0" the program continues with the silo number saved in the temporary variable "Silo", otherwise the outer cycle starts over with a new priority. After the silo number has been found, the program continues with a repeat until cycle whose objective is to distribute the specific material CSM weight into a series of weighing sequences. The end-of-cycle condition is reached if all the proposed weight has been distributed or if the limit of sequences has been reached; in this last case an error flag is turned on. Inside the cycle, the weight is distributed so that the "Tramoggia.Sequenza" array is filled in order charge by charge and sequence by sequence. Every sequence can carry a maximum of a weight limit, inserted as an input, so if a single material exceeds the capacity then the "Residuo" variable is updated and the cycle starts over with

the residual weight to be distributed. After the inner cycles, the error flag is interrogated; if it is on then the outer cycle is exited because there is no more space in the sequences, otherwise the for cycle starts over with a new priority. In summary, the purpose of this part of the program is to organise the materials proposed by the CSM software so that they are cast in the melted steel ordered by priority. The maximum number of cast is given by the maximum number of sequences, in this case 10, and each sequence can have up to 4 charges of different materials. This order is saved in the "Tramoggia.Sequenza" array which is an in/out parameter.

- Sequence manipulation. In this part the sequence, which should have already been calculated, can go through different transformations.

At first the index of the sequence to be modified is fetched from the "Tramoggia.Stati" input parameter and then saved in a static variable; the total weight of the modified charges is also saved in a static variable.

A series of conditions, which includes the static variables derived from the "Comandi" input, is then interrogated to see which type of modification must be done to the sequences. The modification to be done is saved in the "Tramoggia.CaricaModifica" parameter. The possible manipulations are:

- Reset: the "Tramoggia.CaricaModifica" parameter is reset.
- Modify: the sequence addressed by the index is substituted with the one saved in "Tramoggia.CaricaModifica".
- Insert: the "Tramoggia.CaricaModifica" sequence is inserted at the index position; the sequences that follow are shifted forward by one place.
- Cancel: the sequence addressed by the index is removed; the sequences

that follow are shifted back by one place so that there is no hole in the series.

- Disable: the status of the indexed sequence is changed to 2, disabled, or is changed back to 0, to be done, if it already was 2.
- Force: the status of the indexed sequence is changed to 1, forced, or is changed back to 0, to be done, if it already was 1.
- Controls on the automatic weighing cycle. This part of the code is used to define how the charge in progress is managed and it is composed by several parts.
 - Reset setpoint: if the static variable derived from the "Comandi" input is on then the charge in progress and the manual charge parameters are reset.
 - Indexes search. A for cycle is used to look for the first forced sequence, status 1, and the first to do sequence, status 0. These two are then used to find the in progress index "IndiceCaricaAttuale".
 - Start cycle control: if the "StartCiclo" static variable is on then the automatic weighing process is started and the start cycle control bit is enabled.
 - Start cycle conditions: the conditions for the start of the automatic weighing cycle, including the above mentioned control bit, are interrogated. If all the conditions are met then the static variable "StartCicloSequenza", which is used to control the automatic weighing process, is set.
 - Stop cycle control: if the "StopCiclo" static variable is on then the automatic weighing process is stopped and the cycle control bits are reset.

- Parameters reset: if the hopper is in the discharge position and the hopper extractor has finished vibrating then the charge in progress, the weighing in progress and the manual charge parameters are reset.
- Charge in progress and weighing in progress update. This part is used to transfer the elements of the sequences into the charge in progress parameter "Tramoggia.CaricaAttuale" so that it can be manipulated by the rest of the user program.

If the automatic cycle is not enabled then "Tramoggia.CaricaAttuale" is taken from the "Tramoggia.CaricaManuale" parameter.

If the automatic cycle is enabled then the charge in progress is selected by looking at the control bits set in the previous part of the code. The first available sequence, indicated by the "IndiceCaricaAttuale" index, is selected and its status on the sequences series is changed to 3, in execution. When the in progress weighing process is finished the status of the sequence is updated to 4, done, and the following weighing cycle begins. This cycle continues until all the sequences in the "Tramoggia.Sequenza" array are executed or until the "StopCiclo" bit is switched on.

Finally, the data stored in "Tramoggia.CaricaAttuale" is transferred to the "PesateAttuali" parameter.

This means that the data present on the in/out current charge and current weighting parameters, which are used by the weighing managing blocks of the user program, either comes from the manual charge selected by the operator or from an element of the sequences array created by the CSM (5.1) software.

Chapter 6

Proposed revamped program

Because of the new hardware configuration described in the 4.2 paragraph the user program has to be modified. The new program is written in TIA portal. This program is written using the English mnemonic instead of the German one so the symbols used for input output are I Q instead of E A; also the contacts are coded using A instead of U. A simulation that is capable of showing some results is also needed so that the part of the program that is proposed can be analysed on its own, without it being inserted in a real system.

6.1 Hardware setup

The first thing to do is setting up the hardware configuration based on the electrical schemes. In particular the right type of modules are inserted and they are associated with the corresponding input/output area (6.1).

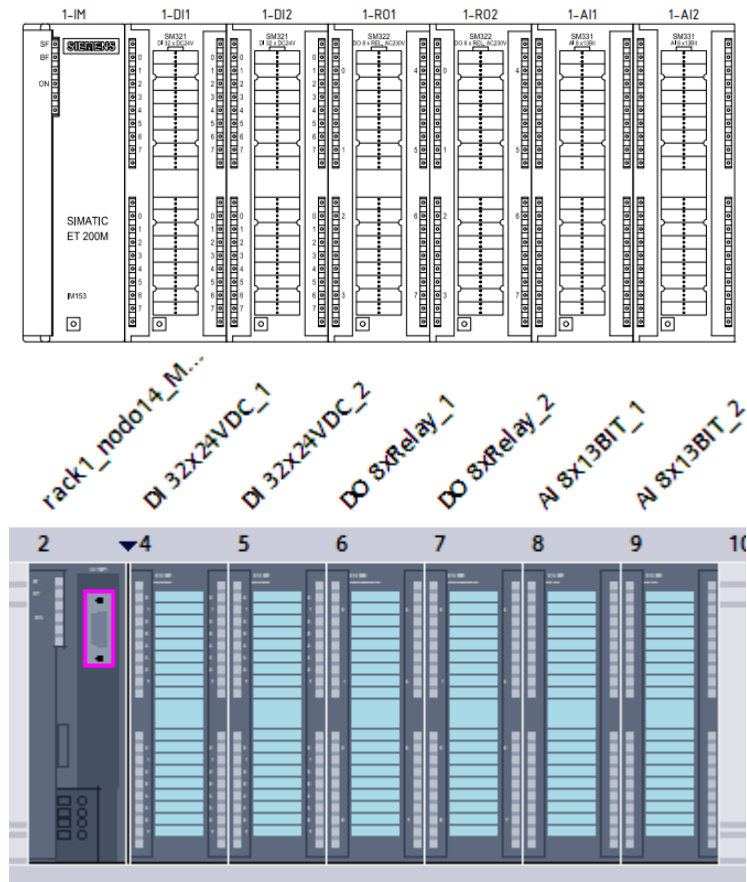


Figure 6.1: Rack 1 scheme and set up

Instead of adding all the possible modules, only the PLC cabinets racks and the B17 relative peripherals are inserted along with the HMI associated with the operator controls.

The symbol table is also updated to include the entered input/output signals.

The next thing to do is creating the Profinet and Profibus network like in figure 6.2 by following the images 4.4 and 4.5. An IP or Profibus address should also be assigned to every controller.

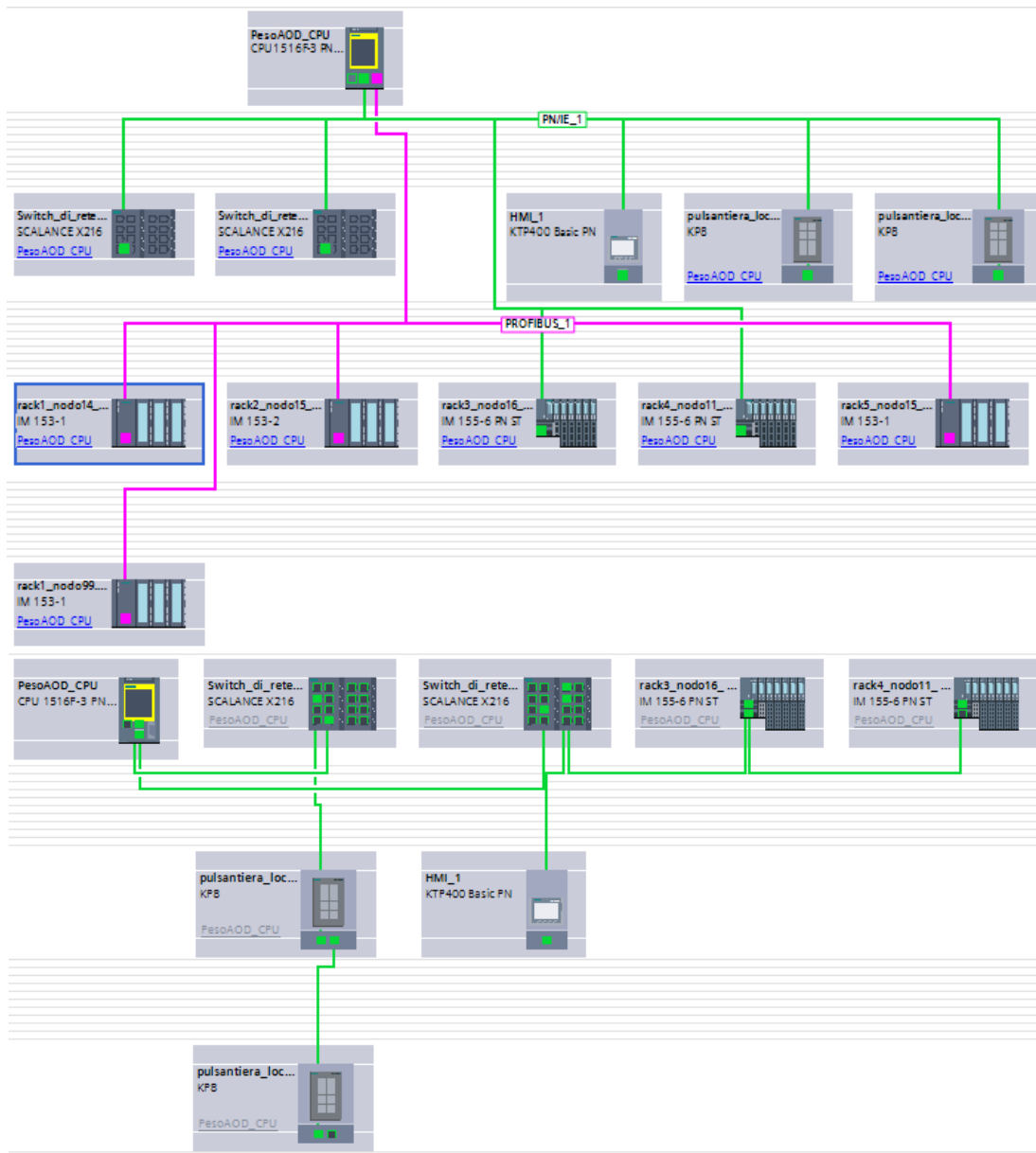


Figure 6.2: Network and topological view

It should be noted that the HMI called "pulsantiera locale B17" are used as local control panel. The program treats its buttons as inputs and two available input areas, I0 and I2, are selected to be these inputs. An example of access to the local variable is I0.0.

6.2 Old block transfer

The new program objective is to perform the same operations as the currently used user program; for this reason some blocks described in the 5 chapter are modified and translated so that they can be the foundation for the revamped program.

6.2.1 Data blocks

To maintain a similar data structure the data blocks are transferred to the revamped program but only the part actually relevant to the weight handling.

- DB17 "Dati_B17": this is still the main data block that stores all the weight relevant hopper information. The unused variables are removed to save space and the "peso_minimo" and "peso_massimo" integer variables are added and they respectively store the constant 3 and 2950; these are used in the program instead of 3 and 2950.
- DB221 "NrTipoCodiceMateriale": this data block is still used but only the hopper relevant parts are transferred. The priority of the material is also added here because it greatly facilitates the data transfer for the automatic weighting mode. The stored variables are the name, slow down weight, CSM data, priority, regular vibration reference and slower vibration reference for each silo.

- DB215 "RECEIVE_STATI_S5_HMI" becomes "RECEIVE_STATI_HMI": it is used to connect the movement blocks with the automatic weighing managing blocks. This behaviour mirrors the one of the old program where the block is used to connect the S5 CPU with the S7 CPU. The variables "posizione _b17 _su _b19" and "fc _stop _b17 _silo _b2" are not used because they are substituted by the simulation variable "pos0" stored in the new simulation data block "DATI_PER_LA_SIMULAZIONE".
- DB151 "RECEIVE_S5_HMI" becomes "RECEIVE_MOVIMENTAZIONE": it is used to connect the movement blocks with the weight managing blocks. This behaviour mirrors the one of the old program where the block is used to connect the S5 CPU with the S7 CPU.
- DB150 "SEND_S5" becomes "SEND_MOVIMENTAZIONE": this data block is transferred to maintain symmetry but it is not actually used because the weight data can be directly accessed from "Dati_B17" even by the movement handling functions because the data no longer needs to be transferred to the S5 CPU.
- DB200 "SEND_COMANDI_HMI_S5" becomes "SEND_COMANDI_HMI": the transferred variables still serve the same purpose as in the old program. A variable for the HMI button that start an automatic weighing cycle is added here.
- DB300 "DB_SequenzaPesateB17": this data block is still fundamental to the user program because it stores the data used in the FB224 function (5.6). In order to transfer this block several user defined type are made to create a structure compatible with the older functions (5.3.1).

6.2.2 Function blocks

Instead of creating weight handling functions from scratch, older blocks are transferred so that the weighting process can be managed in the same way. This operation is not done for the movement functions because they are coded in a step 5 environment, a way older software that has almost no compatibility with TIA portal.

- FB17 "Gestione_peso_B17": The second segment is no longer necessary because "DATI_B17.peso_calcolato" can be used everywhere in the program. The third segment is substituted with the new blocks "blocco _automatico _ausiliario" (6.3.3), which does the same thing as the old segment, and "presa_input_HMI" (6.3.7) that takes the input from the HMI. The tenth segment is empty because the outputs are placed in the new block "movimento" (6.3.6). The twelfth segment now turns on the "OS _DOWN _B17 _MARC" when the simulation bit "pos0" signals that the hopper is in the dump position; also it resets the "silo_pesata_in_corso" when "OS _DOWN _MARC" activates and the "peso_raggiunto" reset is moved to the "peso_tara_rall_b17" block. The fourteenth segment is removed because it is easier to fetch the current silo number directly from the "DATI_B17" data block.
- FB21 "peso_tara_rall_b17": The unused interface parameter are eliminated; the unused segments are also removed. The fifth segment is modified so that "rallentamento" and "peso_raggiunto" are reset together when the hopper is moving and when the system shuts down. The numbers in the segment 7 and 8 are substituted with the "peso_min" and "peso_max" variables of "DATI_B17".

- FB224 "Gestione_pesate": This block is not modified but some interface parameters are modified. The new parameters have the same function as the old ones. This block is used for the sequence calculation and the control of the automatic weighing cycle; the sequence manipulation part is not used.

6.3 New blocks

Many new data blocks and function blocks are inserted in the program to handle the movement part.

6.3.1 Data blocks

The new function blocks all have instance data block. Some other data blocks are also added.

- "DB_controllo_della_movimentazione" is used to store different control bits used for the movement simulation and the HMI.
 - "destra" active when the right movement command is enabled.
 - "sinistra" active when the left movement command is enabled.
 - "pulsante_di_inizio" connected to the HMI button used to start the manual weighing cycle.
 - "pronti_a_iniziare" used in the "MOVIMENTAZIONE" function block as a control signal.
 - "scarica_della_tramoggia" used to handle the hopper dumping.
 - "comanda_posizionamento" HMI signal that is activated to enable the direct movement commands.

- "comanda_destra" HMI signal used to directly move the hopper to the right.
 - "comanda_sinistra" HMI signal used to directly move the hopper to the left.
 - "comanda_vibro_tramoggia" HMI signal used to directly activate the hopper's vibration.
- "estrattori" contains 12 bits used to control the silos extractor in the "MOVIMENTO" function block at the segment 12, 13, 16.
 - "input_dati_HMI" is used to store the data inserted from the HMI (6.5.2); its content is handled in the "Presa_input_HMI" function block.
 - "DATI_PER_LA_SIMULAZIONE" contains some simulation data used by the "FUNZIONE_DI_SIMULAZIONE_MOVIMENTO_TRAMOGGIA" function, the "pos" bits and the "posizione_tramoggia" integer variable.

6.3.2 "MOVIMENTO_DB" instance

The "MOVIMENTO" function block has several static variables.

- "partenza_primo_gruppo" (primo, secondo, terzo, quarto) are signals used to control the start of a new weight charge.
- "partenza_scarico" has the same function as the previous signals but for the dumping phase.
- "caso_di_scarico" and "appoggio_scarico" are used to signal the dumping phase.

- "void1" (1, 2, 3, 4) are used to signal if in the sequence in progress the numbered charge is empty.
- internal signal, used to command the direct movement and extractor.

6.3.3 "Blocco_automatico_ausiliario"

This function is used instead of the third segment of the FB17. The advantage is that the great number of KOP instructions is reduced to a simpler AWL block that uses the LOOP command. The wanted variables ("Totale_CSM", "tolleranza_CSM", "totale_scaricato" and "priority") are fetched from the DB221 and transferred to the DB300 through an array access, where the address is given by the index of the loop.

```
cycle: T #index
L "NrTipoCodiceMateriale".siloB[#index]...
T "Db_sequenza_pesate".DatiSilo[#index]...
...
L #index
LOOP cycle
```

6.3.4 "Calcolo_rallentamento"

This function is used to fetch the slow down weight and the vibration speed reference from the DB221. In the other program this function is used to calculate the pointer variable used to access the DB221 array, making different calculations to find the right result, while here the array is addressed through an index, exploiting the fact that TIA portal can access them in this simpler way. The first segment is used to

find the addressing index by looking at the current silo number; if the value is out of bounds the default value 1 is used (6.3).

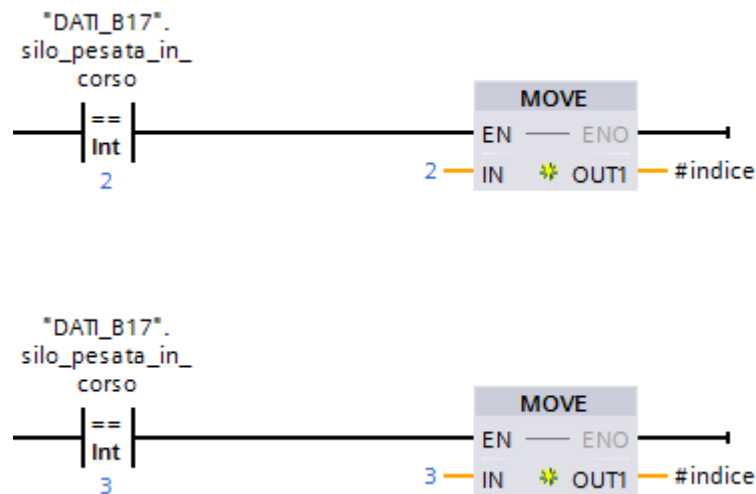


Figure 6.3: Index calculation example

In the second segment the slow down rate value is accessed thanks to the index and it is transferred to the "peso_tara_rall" variable of the "DATI_B17" data block with a MOVE command.

In the last segment the variable that store the extractor vibration speed reference in the "DATI_B17" data block is set according to three cases: if the DATI_B17."rallentamento" bit is set then the slower reference is chosen, if it is not set then the regular reference is chosen and if the hopper is controlled by the local control panel, signaled by the input "selettore __remoto/locale", then a default reference is used.

6.3.5 "Gestione__automatico"

This block is used to handle the automatic weighing process that is controlled by the FB224 "gestione_pesate". The other program uses a series of signals that come from the S5 CPU while here the global data can be used. This block is directly called by the OB1 main program block. This block is composed by 10 segments.

1. the segment is used to simulate the data arriving from the CSM. The choice is made through a button in the HMI that select two possible set of data. In a possible real implementation, this segment should be substituted with one that takes the actual CSM data.
2. the static variable "os_down_scaricamento", which is active at the negative front of the hopper material dump signaled by "DB __controllo __della __movimentazione". scarica __della __tramoggia, is calculated.
3. used to simulate the first choice of the CSM data; it loads some plausible values to the DB221. In a possible real implementation it should be removed.
4. used to simulate the second choice of the CSM data; it loads some plausible values to the DB221. In a possible real implementation it should be removed.
5. the sequence selection command is activated when a negative front is detected in the calculation sequence command, in this way the FB224 (5.6) can handle the sequence after it has already been calculated.
6. it indicates the start of a new cycle with the signal "StartCiclo" stored in the "Comandi" structure of the DB300; this is activated if the corresponding HMI button is pressed or when the previous cycle is ended (and the "os_down_scaricamento" bit is activated).

7. when the "start_ciclo_auto_b17" command is activated by the FB224 (5.6) then the bit "CicloAutoCaricTramB17" is set.
8. the stop cycle command, which activates the stop cycle control part of the FB224 (5.6), is activated. This happens when the "OS_MARC_DOWN" turns on, which in an automatic cycle means that the material dump position has been reached and a cycle has ended, or when all the automatic castings are ended.
9. here the bit for the manual case and the bit for the automatic case, respectively "SEND_COMANDI_HMI".MAN_CARIC_TRAMOGG_B17 and "SEND_COMANDI_HMI".AUTO_CARIC_TRAMOGG_B17, are set and reset when the manual start button is pressed. This is done here because the opposite, where manual is reset and automatic is set, is already handled inside the FB224 (5.6).
10. the positive front of "fine_colata_auto" is transferred to the merker OS_FINE_COLATA which is used in the FB224 as an input parameter.

6.3.6 "Movimento"

This block is used to control the movement of the hopper which can be shifted in four different mode.

- manual mode, if the weight sequence is set up by the operator and the "pulsante_di_inizio" HMI variable has been activated.
- automatic mode, if the weight sequence is set up by the CSM software and the "calcolaSequenza" command has been activated.

- direct mode, if the movement is directly controlled by a HMI command when the "selettore_remoto/locale" input is switched off and the "comanda_posizionamento" HMI variable is switched on.
- local mode, if the "selettore_remoto/locale" input is switched on the movement is controlled by the local control panel.

The weighing process is divided into up to four groups, which represent the four possible charges of a sequence, and then it is followed by a dumping phase that handles the material casting.

This implementation is expected to work in the simulated environment, some modification should be made if this block worked in a real implementation. This function is called directly by the OB1 main program block.

The block has several static variables (6.3.2) and it is formed by seventeen segments.

1. the first group is started. This can happen when the starting button is pressed, in a manual case, or with a new automatic cycle. The signal cannot switch on if the static variable "void1" is on or if the hopper is controlled by the local control panel. The variable "pronti_a_iniziare" must be on.
2. this segment controls the right and left movement signals which are used in the simulation function (6.4). The right signal is turned on during a weighing cycle (manual or automatic) and the current silo is further to the right than the position of the hopper or if the internal right command (segment 15) is switched on. Right cannot be activated during the dumping phase. The left signal is turned on during a weighing cycle (manual or automatic) and the current silo is further to the left than the position of the hopper or if you are in the dumping phase. Left is also turned on if the internal left command

(segment 15) is switched on. Neither of the two direction can be activated if the hopper position is already at the corresponding limit.

3. "RECEIVE_MOVIMENTAZIONE".TRAMOGGIA_B17_IN_MOVIM, the bit that signals the hopper movement, is switched on if right or left are set.
4. the signal used to control the first charge of the sequence, "RECEIVE_MOVIMENTAZIONE". I_GRUPPO_CONTR_B17_AUTO, is activated when the current phase is the first group (set in the segment 1). The signal is switched off when the starting signal of the second group is activated or when the starting signal of the dumping phase is activated (6.4).

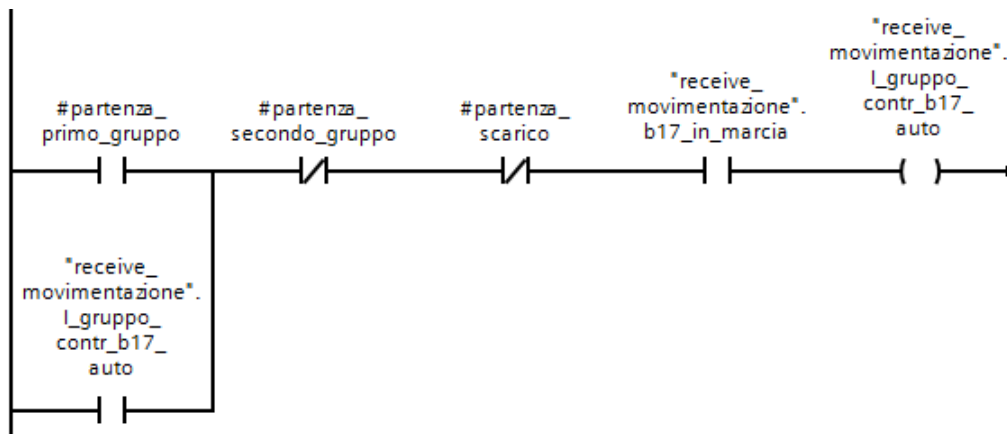


Figure 6.4: Group I activation

5. it is like the previous segment but for the second group.
6. it is like the previous segment but for the third group.
7. it is like the previous segment but for the fourth group. The signal here can only be switched off by the starting of the dumping phase because there is no fifth group to start.

8. activation of the group starting signals (6.5), from 2 to 4, and the dumping phase starting signal (6.6). These signals are one shot because they are active for only one program cycle. Their activation is started by the positive front of the "peso_raggiunto" variable and it is controlled by the "void" static variables and the current active group.

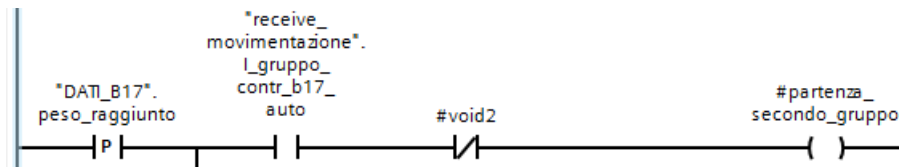


Figure 6.5: Group starting signal

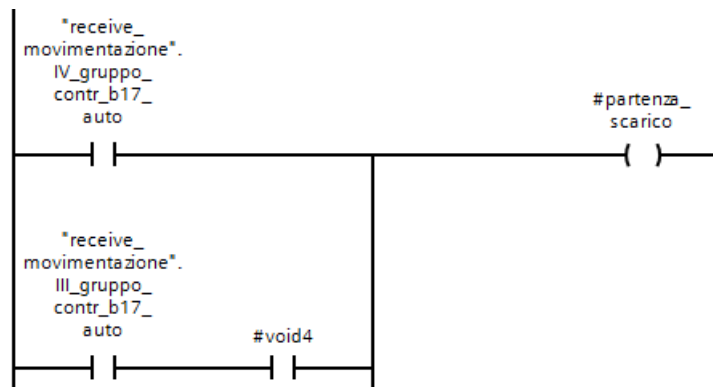


Figure 6.6: Dumping phase starting signal

- The dumping start signal is activated if a current group "void" signal is on or at the end of the fourth group. The "partenza_gruppo" signal instead starts if the previous "gruppo_contr_b17_auto" is on and the corresponding "void" signal is off.

9. after 3 seconds that the starting dump signal is activated, a timer used to maintain consistency with the timing of the charges, the dumping phase

variable "caso __di __scarico" is switched on. The signal is reset when the "pronti __a __iniziare" variable turns on because the weighing cycle (manual or automatic) ends (6.7).

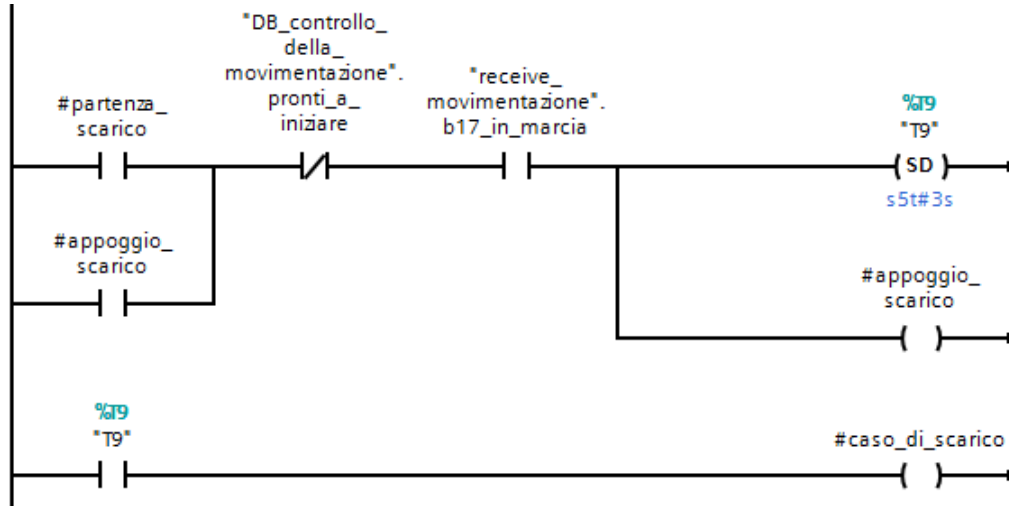


Figure 6.7: Dumping phase activation

10. activation of the dumping command called "scarica_della_tramoggia" stored in the "DB_controllo_della_movimentazione" data block. It can be turned on during a dumping phase if the hopper is at the dumping position "pos0". The signal can also be activated by the internal vibration signal (segment 15).
11. the "pronti_a_iniziare" signal is handled here. The signal is important for controlling other segments in the block (segment 1, 2, 9, 10, 12, 15). It is turned on if the remote controls are enabled, none of the groups are active (because a weighing cycle is active) nor the dumping phase is. The signal can be activated during a dumping phase only in this condition: the net calculated weight is lower than the minimum weight "DATI_B17".peso_minimo for more than 3 seconds; this indicates that a weighing cycle is ended. The timer here

- is inserted to allow the hopper to completely empty during a casting similarly to how the timer "T8" is used in the tare function (5.5).
12. the signals that activate each silo extractor stored in the "estrattori" data block are activated here. In the remote control case, during a weighing cycle, an extractor is activated only if the hopper is positioned under the current silo and it stays active until the target weight is reached. In the local control case the extractor can be activated using the local control panel.
 13. if any of the previous segment signals is on then the "RECEIVE _MOVIMENTAZIONE". `marcia _estrattori _sili _b` is turned on.
 14. the "void" static variables are handled here; if a "contraves" value is "0" in the "DATI_ B17" data block then the corresponding bit is set. These signals are used in the segment 8 to manage the group order.
 15. the direct commands, for remote and local mode, are handled here. Both the control mode activate the internal command static variable (used in the segment 2 and 10). The right, left and vibration remote commands can be activated by the HMI if the direct control bit is set and if the "pronti _a _iniziare" variable is on; the local commands are activated by the local control panel if the local mode is enabled.
 16. here the signals are connected to the respective outputs. `"b17 _in _marcia"`, `"tramoggia _b17 _in _mov"`, `"scarica _della _tramoggia"`, all the extractors signals and the extractors vibration speed reference. While this segment is not fundamental in this implementation, because a simulation is used, it would be very important in a real set up to control the actuators of the system.

17. The "fine_colata" signal is activated when the current charge index becomes "0" (this happens only at the end and at the start of an automatic weighing cycle) and if the status of the first sequence is different than "0" (this happens only if the weighing cycle has already started). The signal is turned on if both previous conditions are verified for more than 1 second to avoid a signal activation between two automatic weighing cycles.

6.3.7 "Presa_input_HMI"

The block is called in the third segment of the FB17 to set up the manual weighing variables; its parametric output called "caricaControllo" is connected to the "CaricaManuale" structure of the DB300. The block is divided in two segments. The first part controls the validity of the data put in the HMI table "input _dati _HMI" by looking that the selected weight is greater than "0" and that the silo number is between 1 and 12.

The second segment of the function is used to order the valid data into the output parameter structure and fill the unused space with zeros so that no empty spot is left in the array. This process is done by a series of conditional jumps controlled by the static variables set up in the first segment.

```
A #valido1
JC v11
...
//case where 1 is valid and must be put in the first position
v11: L "Input_Dati_HMI".FatiHMI.KgSelezione[1]
T #caricaControllo.KgSelezione[1]
L "Input_Dati_HMI".FatiHMI.NSiloSelezione[1]
```

```
T #caricaControllo.NSiloSelezione[1]
A #valido2
JC v22
A #valido3
JC v23
...
//case where 3 is valid and must be put in the second position
v23: L "Input_Dati_HMI".FatiHMI.KgSelezione[3]
T #caricaControllo.KgSelezione[2]
L "Input_Dati_HMI".FatiHMI.NSiloSelezione[3]
T #caricaControllo.NSiloSelezione[2]
...
//case where the fourth position must be filled with 0
void4: L 0
T #caricaControllo.KgSelezione[4]
T #caricaControllo.NSiloSelezione[4]
...
```

6.4 Simulation blocks

In order to see how the program works a simulation is needed because the system still is not implemented in a real setup. The FC called "FUNZIONE _DI _SIMULAZIONE _MOVIMENTO _TRAMOGGIA" is used to simulate the position of the hopper and the sensors that signals it. In the first segment a series of position variables called "pos" stored in the "DATI _PER _LA _SIMULAZIONE" data

block are used to simulate the hopper sensors position; when the position of the hopper corresponds to one of the possible values then the signal turns on. In a real implementation these signals should be provided as input by sensors. In the second segment a counter whose limits are 0 and 12 is used to simulate the hopper movement and its value is stored in the "posizione _tramoggia" variable of the simulation data block. The counter is updated using the right and left signals, while in a real implementation these two should be used to command an actual motor. In the third segment a timer is used to delay the right and left signals; this is done to avoid an instant movement after the command is given. In a real implementation this segment should be removed because it is just used for simulation purposes.

6.5 Human Machine Interface

TIA portal gives the ability to design the HMI using the variables already present in the user program; this means that a data block variable can be directly linked to an HMI command. This is a great advantage compared to the other configuration where the HMI set up is more complicated. The used HMI is connected to the Profinet network (6.1). The HMI is also used to simulate the CSM software inputs (5.1). The local control panel signals are handled by the program as inputs (area I0 and I2) and are not therefore present in this section.

6.5.1 First page

The first page shown in figure 6.8 contains different information. On the left a series of light show the current hopper position, where 0 is the dumping position and the other are the silos from b1 to b12. The "movimento" light turns on when

the hopper is moving. The "scaricamento" light turns on when the hopper extractor is vibrating. The "estrattore e rall" lights turn on when the target weight is reached and when the slow down weight is reached. "PESO NETTO" indicates the net weight on the hopper. The "pronto a iniziare" light switch on when the "pronto_a_iniziare" variable of the simulation data block is on, this is useful to see if a weighing cycle is in progress or not. The button "INIZIO" can be pressed to start a manual weighing cycle. The "ATTIVAZIONE" switch can be used to simulate the switch on and off of the system, when it is off no operation can be made. On the bottom, inside a green rectangle, the direct control buttons are located. The switch "CONTROLLO DIRETTO" can enable or disable these controls. The arrow buttons can move the hopper to the left and to the right while the "VIBRAZIONE" button can be used to activate the hopper extractor. It should be noted that when these controls are used the "movimento" and "scaricamento" lights above are switched on.

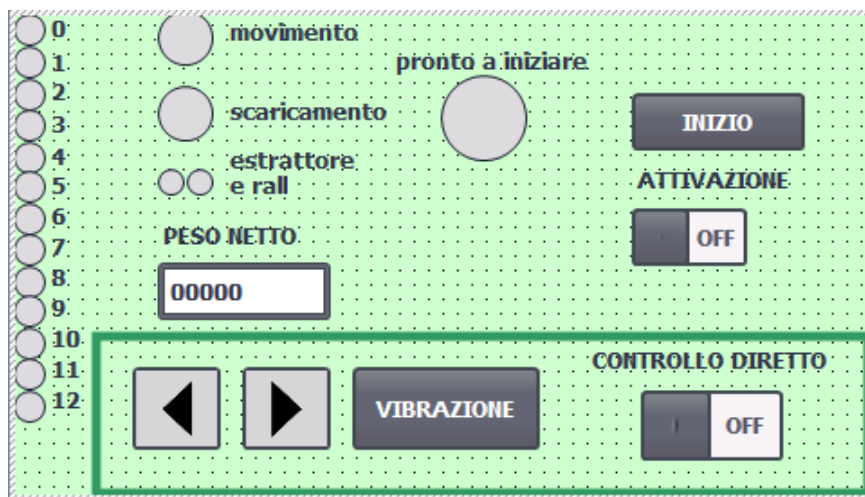


Figure 6.8: HMI page one

6.5.2 Second page

The second page of the HMI shown in figure 6.9 is used to insert the manual weighing data. On the left an input table can be filled in with weight selection and silo selection, these simulate the operator materials choices. On the right a table with the same structure shows the current sequence; this is the ordered left data in a manual weighing case or the current selected sequence in an automatic weighing case. A light on the bottom shows if the inserted sequence weight is bigger than the maximum range of the hopper.

The HMI page two interface is displayed on a blue dotted background. It features two tables and a status indicator.

Selezione sequenza:

0000	00
0000	00
0000	00
0000	00

Sequenza ordinata:

0000	00
0000	00
0000	00
0000	00

SUPERAMENTO PORTATA

A grey circle indicator is shown next to the text "SUPERAMENTO PORTATA".

Figure 6.9: HMI page two

6.5.3 Third page

The third page shown in figure 6.10 is used to simulate the CSM software. The buttons "scelta uno" and "scelta due" are used to simulate a plausible weight selection (seen in the first segment of the "gestione __automatico" block 6.3.5) and the chosen data are shown in the left table in silo order. The "SELEZIONE SEQUENZA" is used to calculate the sequence in the FB224 (5.6) block and the "INIZIO CICLO" button is used to start an automatic weighing process.

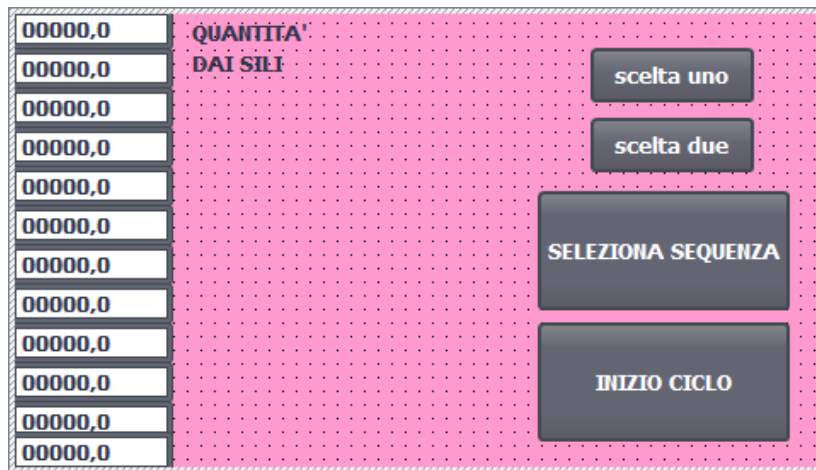


Figure 6.10: HMI page three

6.6 Simulation results

To see how the system works a simulation is done. The software used is S7-PLCSIM, capable of simulating an hardware setup so that the user program can be used online. In order to be connected, the program should be linked to the PLCSIM S/-1200/S7-1500 interface through the 192.168.148.4 port as shown in figure 6.11.

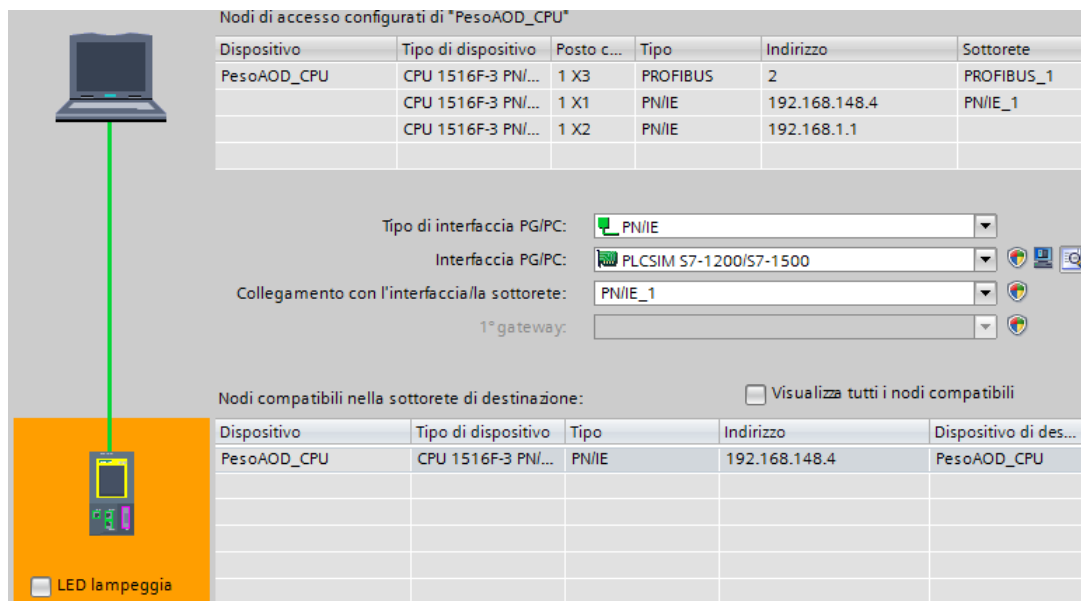


Figure 6.11: Online connection

The online program can be interacted with by the HMI screen so that a series of results can be shown. Of course, the system is not really connected to a machinery that can dump real material so the weight on the hopper must be selected by the simulator through a table where the "pesatura __tramoggia __b17" input variable (IW3016) can be modified.

6.6.1 Direct control

This type of simulation shows how the hopper can be directly controlled by the user. To see the results only the first page of the HMI is needed. The first thing to do is to activate the system by switching on the "ATTIVAZIONE" switch, if everything is correct then the "pronto a iniziare" light should turn green (6.12).



Figure 6.12: Starting screen

To turn on the direct control mode the "CONTROLLO DIRETTO" switch should be turned on. Now, if the arrows are pressed, the "movimento" light turn green and the position is shown to change on the left lights (6.13).



Figure 6.13: Direct movement

In the same way, the "VIBRAZIONE" button can be pressed and the "scaricamento" light turns on (6.14).



Figure 6.14: Direct dumping

6.6.2 Local control

This type of simulation shows how the hopper can be controlled through the local control panel. Given the fact that the local control are treated as inputs the simulation table must be used to simulate their change (6.15).

"selettore_remot...	%I3.0	Bool	FALSE
"tramoggia_b17...	%I2.7	Bool	FALSE
"tramoggia_b17...	%I2.6	Bool	FALSE
"marcia_estrattor..	%I0.0	Bool	FALSE
"marcia_estrattor..	%I0.1	Bool	FALSE

Figure 6.15: Simulation table

After switching on the system, the local control mode can be activated with the "selettore _remoto/locale" (I3.0). In the HMI screen the "pronto a iniziare" light should turn red (6.16).



Figure 6.16: Local control starting screen

Now the hopper can be moved by activating the "tramoggia _b17 _avanti" or the "tramoggia _b17 _indietro" inputs (I2.7 I2.8) and the HMI "movimento" light should turn on like the previous case. The same thing can be said about the "marcia _estrattore _b17" that activates the dumping of the hopper and turns on the "scaricamento" light on the HMI like in the previous case. Finally, a "marcia _estrattore _bX" (X from 1 to 12) can be turned on to activate the corresponding silo's extractor; this is signaled in the HMI by the "estrattore" light turning green (6.17).



Figure 6.17: Local control extractor

6.6.3 Manual weighing cycle

From the 3.4 section the tare has been calculated to be 2388 kg so a starting Siwarex input value can be set to be 2388 (6.18).

"pesatura_tramo...	%IW3016	DEC	2388
--------------------	---------	-----	------

Figure 6.18: Table weight selection

After the system has been turned on, the wanted sequence can be selected on the second page of the HMI (6.19).

Figure 6.19: Manual sequence insertion

From the image it can be seen that the incorrect inputs are eliminated and the sequence is ordered on the right (6.3.7). It must be noted that if a too big value is inserted the "SUPERAMENTO PORTATA" light turns on (6.20).

Selezione sequenza		Sequenza ordinata	
800	52	5000	5
5000	5	200	8
0	8	0	0
200	8	0	0


 **SUPERAMENTO PORTATA**

Figure 6.20: Exceeding range

Now, to start the weighing cycle, the "INIZIO" button must be pressed; at this point the "pronto a iniziare" light turns red and the hopper moves until the selected silo is reached and the extractor turns on (6.21).

0		movimento		pronto a iniziare	INIZIO
1		scaricamento			
2		estrattore e rall			
3					ATTIVAZIONE
4					ON
5		PESO NETTO			
6		<input type="text" value="0"/>			
7					
8					
9					
10					
11					
12					

CONTROLLO DIRETTO
  **VIBRAZIONE**  **OFF**

Figure 6.21: Weighing process-active extractor

The input value on the simulation table is now modified to simulate the material

charge on the hopper. When the slow down weight is reached ("peso _rall" has been set to 10 in this simulation so the slow down weight is 2478) the "rall" light turns yellow and the speed reference of the extractor is modified; the net weight is also modified (6.22).



Figure 6.22: Weighing process-slow down signal

When the target weight minus 5 is reached the timer part of the program, handled in the FB17 and FB21 (5.4, 5.5), activates so that the target weight is reached on the scale; in the simulation this means that the weight arrives at the wanted value (2488 in this case).

At this point the hopper moves again to the next silo and the same process is repeated until all the charges are completed (6.23).



Figure 6.23: Weighing process-end of an extraction

When the cycle ends, the dumping phase starts and the hopper is moved to the dumping position; here the hopper extractor is turned on and the "scaricamento" light turns green. The weight on the simulation table is reduced to simulate the material dumping (6.24).



Figure 6.24: Weighing process-dumping

When all the material is dumped, and the net weight reaches "0", the weighing cycle is complete and the HMI turns back to the initial status with the "pronti a iniziare" light green (6.12).

6.6.4 Automatic weighing cycle

In order to start an automatic weighing cycle the third page of the HMI must be used; here a choice between "scelta uno" and "scelta due" (explained in "gestione _automatico" (6.3.5)) is made and the simulated sequence is shown on the left (6.25).



Figure 6.25: CSM simulated input

Now the "SELEZIONA SEQUENZA" button is pressed and released so that the sequence can be calculated in the FB224 (5.6), if the sequence is correctly selected the button should turn green (6.26).



Figure 6.26: Sequence calculation start

To start the cycle the "INIZIO CICLO" button should be pressed. The automatic weighing follows the same steps as the manual one: the selected silo is reached, the correct lights turn on when a condition is met ("movimento", "scaricamento", "estrattore e rall" and "PESO NETTO"), the target weight is reached for every charge and when a sequence is completed the hopper is discharged at the dumping position. At this point the next cycle automatically starts without the need to press any button. This continues until the final sequence is reached and the weighing process finally ends.

It must be noted that, in the second page of the HMI, the ordered sequence of the automatic cycle is shown instead of the manual sequence (6.27).

Seleziona sequenza		Sequenza ordinata	
800	52	400	1
100	5	400	2
0	8	400	3
200	8	400	4


 **SUPERAMENTO PORTATA**

Figure 6.27: Ordered sequence in automatic process

When the cycle is ended the first page returns to the starting status (6.12) and to deselect the sequence the "SELEZIONA SEQUENZA" button on the third page must be pressed.

6.7 Real implementation

In order to have a series of results a part of the program is used to manage a simulation (6.4). This choice leads the program to have some simplifications that let it be simulated but make it unable to work in a real set up. The differences are highlighted in this section.

A big problem that this program would have in a real implementation is that several modules are missing in the network. This choice has been made because the excluded modules are not used to control the weighing process, but they are nonetheless fundamental to make a real implementation work. All the alarms are, for example, very important for a real set up but they are not handled in the

proposed program.

Another simplification is the turn on signal called "receive __movimentazione. b17_in __marcia" that is used to simulate the activation of the hopper machinery. While in the program the variable is, for simplicity sake, turned on with the button "ATTIVAZIONE" located in the first page of the HMI (6.5.1), in reality this variable would be controlled by several other inputs which, especially the ones found in the first rack (4.2.2), would in fact disable the hopper's movement. An example of these inputs are the variable "INTERRUTTORE 110 VAC AUSILIARI INVERTER TRASLAZIONE TRAMOGGIA B17" and "INTERRUTTORE ALIMENTATORE 24 VDC ENCODER NASTRO B17" or other power related variables.

Another simplification is the handling of the hopper position. While in the program the position is managed by a counter and different bits stored in the "DATI __PER __LA __SIMULAZIONE" data block, the actual data should come from the power module control unit shown in figure 6.28.

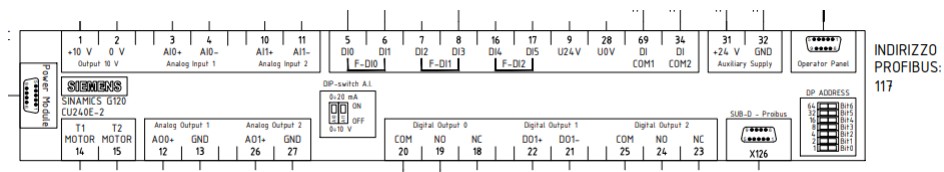


Figure 6.28: Power control unit

This would use the control signals to move accordingly the hopper. The control unit is connected to the Profibus network and can therefore use the same variables that are used by the simulation function. The function block "FUNZIONE __DI __SIMULAZIONE __MOVIMENTO __TRAMOGGIA" should therefore be replaced with a block that handles the hopper's power unit. With the same logic, the position signals should come from the "finecorsa" sensor inputs and other possible positions,

like the second dumping position that is used during the system calibration, should be added and handled accordingly.

Another difference comes from the CSM handling function (first segment of the "Gestione __automatico" (6.3.5) block). The data that in the simulation comes from a selection in the third page of the HMI (6.5.3) should, in a real implementation, come from the actual CSM software. The CSM selections could be inserted as inputs or as a HMI data.

Chapter 7

Conclusion

The first objective of the study is to find out the quality of the performed weight measurement . The measuring system has been analysed in the "measuring system" 2 chapter and the uncertainty of its weight indication has been analysed in the "uncertainty calculation" 3 chapter. The calculation has taken into account the contribution of the load cells, the junction box and the Siwarex weighing module; the amplifier and the analog to digital converter of the last module have also been analysed. The result of these calculation can be seen in the 3.4.2 section where different type of results are found (7.1). The most optimistic case is the perfect calibration one, where the uncertainties due to calibration are eliminated and only the ADC, amplifier, junction box and cells contribution are taken into account. The second observed is a case where the sensitivity of the cells changes after the calibration, something that usually does not happen because of the periodical instrument calibration. The third case is probably the most realistic one because it takes into account all of the Siwarex module possible uncertainty contribution by inserting in the propagation formula the Siwarex data sheet value, 0.05% FS.

This case is also the one that, with a good calibration, have the biggest uncertainty. The last case is the one where the performed calibration is not perfect and as a consequence the calculated gain error increase.

Table 7.1: Calculation result table

CASE	ABSOLUTE [kg]	RELATIVE
Perfect calibration	1.1	0.042%
Sensitivity case	2.0	0.075%
Siwarex case	2.1	0.076%
Gain case	26.2	0.970%

As it can be seen by the 7.1 table, the results are all quite small because the absolute uncertainty is found to be a little more than 2 kg so, even by expanding the uncertainty with a coverage factor of 2 so that the confidence level becomes 95%, the maximum expected error of the weight indication is always less than 5 kg, a value which is acceptable because it let the AOD operate without the need of making too many corrections. The percentage of uncertainty relative to the net weight is in fact really small with a maximum at 0.076% which expanded by 2 becomes 0.152%.

The case different by all the others is when the gain factor of the Siwarex module amplifier is different from the expected value because of a bad calibration. The weight uncertainty increases a lot in this case reaching 26 kg, a value which can create problems to the AOD process. By expanding the uncertainty with a 2 coverage factor the uncertainty becomes more than 52 kg, a value that is not acceptable. Even by looking at the relative uncertainty, which may seem small at 0.970%, a problem arises. In the studied case, where 2702 kg of material are weighed, 26 kg of uncertainty are probably still manageable because the relative uncertainty, as said before, is still small. By using a coverage factor of 2 the

uncertainty rises to 1.94%, a value that would force the steel composition to be diluted and corrected with a loss of resources and time. The situation changes for the worse if the material weighed on the scale is inferior because, as explained at the end of section 3.4.2, a lower load implies an higher relative uncertainty. In a case with a weight of 500 kg, for example, and by expanding the uncertainty with a coverage factor of 2 to have a 95% confidence level, the error percentage could reach up to 5% of the wanted weight.

From these data we can conclude that, in order to obtain a weight indication that does not negatively affect the AOD process, a good calibration of the weight measuring system is fundamental. Fortunately this can easily be achieved by following the calibration steps (2.3.1). The revamped system also has similar uncertainty value and the revamping process does not seem to greatly impact the measurement uncertainty as it can be seen in the (3.4.4) section. This is due to the fact that the used cells system is the same in both configurations, that the junction box difference does not imply any increased uncertainty and that the uncertainty given by the weighing module is the same for both setups.

The second objective of the study is to write a program that can move the hopper so that it can be used in the revamped AOD process. The proposed program is analysed in the "proposed revamped program" (6) chapter. The program is made by implementing the electrical scheme (4.2), transferring some blocks from the current program to the revamped program, writing new blocks to handle the hopper movement, creating new HMI screen so that the program can be interacted with and creating a simulation. Even if the program is not actually ready to be used in a real implementation (as explained in the 6.7 "real implementation" section) its behaviour can still be observed by looking at the simulation results in the (6.6)

section. The proposed program is able to move the hopper in four different modes: direct control mode, local control panel command mode, manual weighing mode and automatic weighing mode. From the simulation images we can conclude that the proposed program could be implemented to handle the AOD process in the new (4.2) hardware set up. The advantages of this type of program, in addition to the revamped modules parameters themselves, can also be seen in the code itself; the number of variables in data blocks can be greatly reduced because the communication between two different types of CPU is no longer necessary, also some part of code can be greatly simplified (like the third segment of the FB17 (5.4)) thanks to the TIA portal functionalities.

In conclusion both objectives have been reached because the uncertainty of the weight indication has been found and its main contribution has been identified, also a functioning program in TIA portal that can move the B17 hopper in a similar way to how it is currently moved has been created and its characteristics have been analysed thanks to a simulation.

For future prospects, the update of older CPU model to more modern one seems like a good direction to take; weighing systems, like the studied one, do not seem to be greatly influenced by a PLC revamping and the software level advantages combined with possible new modules functionalities can simplify the code and therefore simplify possible maintenance operations. The HMI configuration is also greatly simplified in a system based on the TIA portal software so trying to update older systems can improve also this aspect. In the context of weighing uncertainty, the control of calibration procedure and calibration quality seem to be the most important factor to look after so inserting modules that can better control the calibration data, like the studied Siwarex WP321, looks like a good direction to

take for possible future revamping; updating the load cells, on the other hand, does not seem to be a fundamental step to take because of how the uncertainties propagate through the system. The definition of an effective calibration procedure is also a fundamental step to take in a possible revamping process because of the great influence that calibration quality has on measure indication quality.

Chapter 8

Bibliography

Celle di carico, Accessori per celle di carico, Cassetta di connessione digitale SIWAREX DB. Siemens. 2022.

CS5516 CS5520 16-bit & 20-bit Bridge Transducer A/D Converters. Cirrus logic. 2005.

Dara Trent. Load Cell and Strain Gauge Basics | Load Cell Central. "<https://www.800loadcel.com/load-cell-and-strain-gauge-basics.html>". 2019.

Load Cells & Force Transducers used in Trade Devices-Field & Laboratory Evaluation Manual. "<https://ised-isde.canada.ca/site/measurement-canada/en/laws-and-requirements/load-cells-force-transducers-used-trade-devices-field-laboratory-evaluation-manual#Section2.4>". 2012.

OIML R76-1 International Recommendation Non automatic weighing instruments Part1: Metrological and technical requirements-Tests. International Organisation of Legal Metrology. 2006.

Operating Manual HBM Weighing cells and force transducers with strain gauge measuring systems C1, C3, C3H2. Hottinger Baldwin Messtechnik.

SIEMENS Information and Training Automation and Drives Simatic S7, Programmazione avanzata Corso S7-PRO2. Siemens. 2001.

SIEMENS Scuola Automazione Industriale Simatic S7, Programmazione 1 Corso S7-PRO1. Siemens. 1999.

SIEMENS Simatic, Lista istruzioni (AWL) per S7-300/400 Manuale di riferimento. Siemens. 2010.

SIEMENS Simatic, S7-1500 CPU 1516-3 PN/DP (6ES7516-3AN00-0AB0). Siemens. 2014.

SIEMENS Simatic, S7-300 CPU 31xC e CPU 31x: Dati tecnici Manuale del prodotto. Siemens. 2008.

SIEMENS Simatic net, S7-300 - Industrial Ethernet/PROFINET CP 343-1 Manuale del prodotto. Siemens. 2012.

SIEMENS Sistemi di pesatura Sistema elettronico di pesatura SIWAREX WP321 Istruzioni operative. Siemens. 2019.

SIEMENS Siwarex u (One and Two-Channel Model) Equipment Manual. Siemens. 2005.

SIEMENS Siwarex u Modulo di pesatura universale per SIMATIC S7 300 e ET 200M Guida all'uso. Siemens. 2012.

What is Strain Gauge Load Cell? Working Principle, Construction & Applications. Electrical Workbook. "<https://electricalworkbook.com/strain-gauge-load-cell/>". Jul, 2021.

8.1 Electrical schemes

Lab/Ele. Cogne Acciai Speciali Pesi AOD, Schema quadro CPU. Cogne Acciai Speciali Aosta. Mar, 2009. electrical scheme.

Macciò. COGNE Srl ACCIAI SPECIALI, Forno AOD Sostituzione automat. caricamento leghe, Elenco ingressi-uscite digitali rack principale e ET200. Siemens S.p.A. Dec, 1995. electrical scheme.

L. Zappella. Cogne Acciai Speciali Aosta, Nuovo impianto additivi AA3313 Quadro di potenza. R.Z. s.r.l.. Apr, 2022. electrical scheme.

L. Zappella. Cogne Acciai Speciali Aosta, Nuovo impianto additivi AA3313 Quadro di distribuzione. R.Z. s.r.l.. Apr, 2022. electrical scheme.

L. Zappella. Cogne Acciai Speciali Aosta, Nuovo impianto additivi AA3313 Quadro PLC. R.Z. s.r.l.. Apr, 2022. electrical scheme.

L. Zappella. Cogne Acciai Speciali Aosta, Nuovo impianto additivi AA3313 Remoto livello sili B_F (99). R.Z. s.r.l.. Apr, 2022. electrical scheme.

L. Zappella. Cogne Acciai Speciali Aosta, Nuovo impianto additivi AA3313 Pulsantiera locale B17-Sili_B (100). R.Z. s.r.l.. Apr, 2022. electrical scheme.

Appendix A

Theoretical background

In this chapter there is a list of theoretical notions that are useful to understand the document.

A.1 Wheatstone bridge

The Wheatstone bridge is a circuit used to accurately measure an unknown resistance. It is formed by three known resistance and the unknown one placed in two branches (A.1).

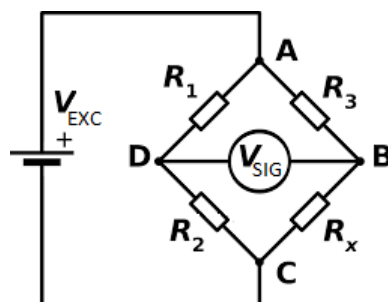


Figure A.1: Wheatstone bridge scheme

Where the A and C points represent the excitation voltage input while the B and D points represent the signal voltage output. By studying the voltage divider of the two branches it is easy to see that the unknown resistance, which value is in this case dependant on the input weight, can be calculated by measuring the output voltage:

$$V_{sig} = V_{exc} \left(\frac{R_2}{R_2 + R_1} - \frac{R_x}{R_x + R_3} \right) \quad (A.1)$$

$$R_x = \frac{-R_3 \cdot \left(\frac{V_{sig}}{V_{exc}} - \frac{R_2}{R_2 + R_1} \right)}{1 + \frac{V_{sig}}{V_{exc}} - \frac{R_2}{R_2 + R_1}} \quad (A.2)$$

This means that if the voltage at the output changes then the unknown variable resistance is changing.

Usually, in order to have an high output sensitivity relative to the resistance variation, the four resistance are selected equal to each other. If the variable resistance is representing a strain-gauge, like in the case under study, then it is better to use resistances that have the same value as the nominal resistance of the strain gauge (A.3).

$$R_1 = R_2 = R_3 = R_x(\text{nominal value}) \quad (A.3)$$

In a case like this the output voltage would be 0 V when the system is unstressed while it would vary when the system is stressed by a force. This type of system is however very dependant on temperature because the length of the strain-gauge and its relative resistance are both susceptible to temperature changes. To compensate this problem, and to accentuate the mechanical distortions, the Wheatstone bridge is formed by four strain-gauge placed transversely to each other in a full bridge configuration (A.2).

A.1.1 Full bridge

As said in the previous section, because resistive strain-gauges are used, the full bridge configuration is used to compensate the temperature effects. By looking at the A.1 equation, an hypothetical variation due to temperature is compensated on the resistance fractions, because the factor simplify at the numerator and denominator.

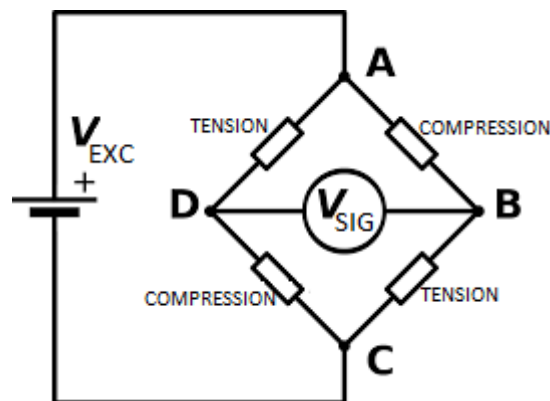


Figure A.2: Wheatstone full bridge

The strain gauges are placed perpendicularly to each other (A.3) so that when a force is applied the voltage output is modified by a significant value.

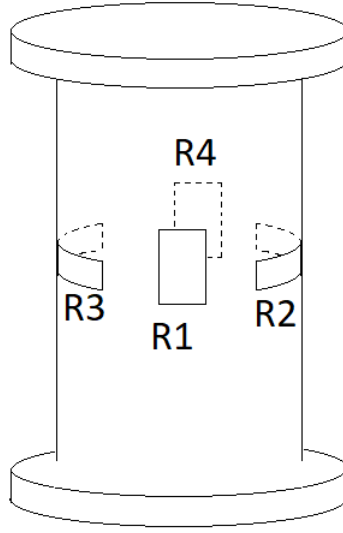


Figure A.3: Cylinder scheme

A.2 Probabilistic method

In order to evaluate the uncertainty of a measurement a calculation must be performed to correctly combine the uncertainty contributions of each component of the measuring system. A simple sum of every contribution, like in the deterministic approach, would be too pessimistic; for this reason a method that return a more realistic result, the probabilistic approach, is used. In this case the measurement is considered a random variable and it is characterised by a probability density function. The goal of this method is to find the standard uncertainty of the measuring system, a value that can be multiplied by a coverage factor, in this case $K=2$, to find an interval that contains the true value of the measurement with a confidence level, in this case 95%. If the uncertainty of a device is given by a data sheet, like in the studied case, then a suitable PDF must be assigned to each given

random variable. In this document the individual components standard uncertainty $u(x)$ is found by translating the data sheet uncertainty information $\delta(x)$, that is provided with the deterministic model, into a uniform distribution.

$$u(x) = \frac{\delta(x)}{\sqrt{3}} \quad (\text{A.4})$$

A function whose variables are random variables ($y = f(x_1, \dots, x_n)$) is a random variable as well and has a standard uncertainty that is derived by its variables standard uncertainty. A measure that is derived by other measure, and is therefore in their function, it is an indirect measure. In order to find out the standard uncertainty of an indirect measure whose N variables are all uncorrelated to each other the following formula must be used:

$$u_c^2(y) = \sum_{i=1}^N \left(\frac{\delta f}{\delta x_i} \right)^2 \cdot u^2(x_i) \quad (\text{A.5})$$

The formula finds the square of the function standard uncertainty u_c by performing a sum of the square of the derivative of the function for each variable multiplied by the square of the variable standard uncertainty.

A.2.1 Verification scale interval

The verification scale intervals e , expressed in unit of mass, indicate the points in which the cell has been verified during the initial test phase, before the sale of the product. A load cell is able to measure intermediate values, because of its strain gauges structure (A.1.1), but during the test phase it has been approved only for the described intervals e and, for this reason, scales are usually set to only

show weights which are multiples of the verification scale interval of the cell. The weighing module under study (2.3.1) is able to show also intermediate values but only the multiples of the interval e should be considered for the reasons previously explained.