POLITECNICO DI MILANO

Dipartimento di Ingegneria Elettrica

Relatore: Prof. Stefano Maggi
Tutor aziendale: Ing Andrea Battù

Tesi di laurea di:
Marco Berizzi Matr. 853155

Anno accademico 2016/2017

SUPERVISION AND AUTOMATION OF THE ELECTRICAL DISTRIBUTION SYSTEM IN A MEDIUM SIZE INDUSTRIAL PLANT

Schneider Electric

Index

Introduction

This document describes how the supervision of the electrical distribution system inside an industrial plant of medium size (2MW) can be realized using Scada software to communicate with protection devices and PLC to implement automatism.
The document show how protocol IEC61850 can be used both for the connection to the protection devices for the supervision but also to realize some automatism or part of them.

Considering a plant with the need of high reliability for what concern electrical power feeding, the document describes which are the typical automatisms involved:

- ATS (Automatic Transfer System) to switch power supply line
- Load Shedding to reduce black out risk
- Emergency generator management
- Load sharing for emergency generators

For these automatisms, the capabilities offered by IEC61850 has been used as much as possible (for example goose messages for ATS).

This document will be subdivided in two parts:

- The first one will be a theoretical approach concerning two different communication protocols: Modbus and IEC61850.

- The second one will be the description of the supervision and automation system of the electrical distribution inside a medium size industrial plant using the protocols previously described.

A focus has been done on the concept of equipment simulation as a useful mean to test supervision and automatism before real installation on the plant. Simulation of live equipment such as diesel generator is not a simple task and requires deep knowledge of the generator behaviour.

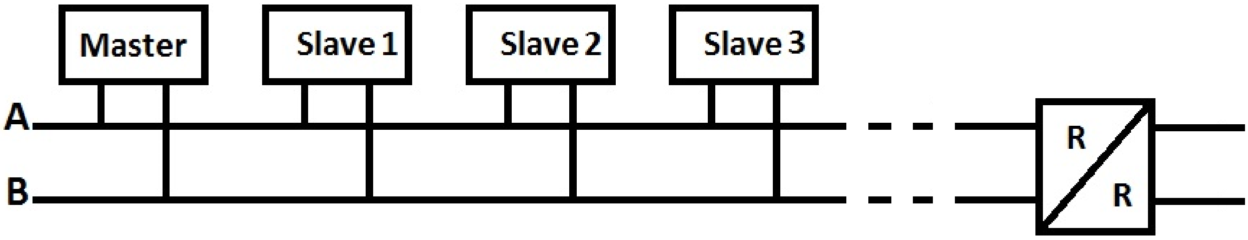
A brief description of hardware architecture of the Ethernet LAN has been also introduced with some hints on media redundancy and their management protocols.

PART 1

INDUSTRIAL PROTOCOLS

1.1) MODBUS

Modbus is a serial communications protocol introduced by Modicon (took over by Schneider Electric) in 1979.
It was thought to be used only as standard for PLCs (programmable logic controllers), but thanks to its robustness and its very short messages, it is used nowadays to connect many of industrial electronic devices.



The physical connection is realized by the standard RS485 half-duplex in which all devices are connected together by two wires A and B (in-out configuration).
This condition implies that only one device at a time can send messages, for this reason a “line moderator” has to be present and unique in this system.
This role is assumed by the master while the other devices are named slaves.
In order to distinguish one component of the system from another one, a slave address is given at each of them, this is a number from 1 to 247 (the physical order on the bus does not matter).

The message is composite by series of bits (logical level of potential difference between the two wires): If the potential of A is lower than that of B the signal is considered 1 otherwise 0.
The rated voltage is 5V and the minimum allowable differential voltage is 200mV.
To avoid a high voltage-drop due to the wires resistance 32 devices at most can be connected on the line, to increase this number a repeater can be placed.

There are two kinds of Modbus: Modbus ASCII and Modbus RTU.
The former is less used then the latter in industrial uses, it is based on a direct conversion of ASCII characters in binary codes, this implies a high number of bits.
For this reason, Modbus RTU is the most common used protocol.
His message is composite by:3.5 chars time of empty message to make the slaves able to recognize the start of the message, 8 bits for the slave address (1÷247), 8 bits for the function code (1÷16 reported in the table on the right), 16 bits for the first register, 16 bits for the number of register (1÷125), and 16 bits reserved for the CRC (Cyclical Redundancy Check).

	Description	Code
BITS	Read Coils	0x01
	Read Discrete	0x02
	Write Single Coil	0x05
	Write Multiple Coil	0x0F
16 bits regs	Read Input Registers	0x04
	Read Holding Registers	0x03
	Write Single Holding Reg	0x06
	Write Multiple Holding Regs	0x10
	R/W Multiple Regs	0x17
	Mask Write Regs	0x16
	Read FIFO Queue	0x18
File	Read File Record	0x14
	Write File Record	0x15
Diagnostic	Read Exception Status	0x07
	Diagnostic	0x08
	Get COM Event Counter	0x0B
	Get COM Event Log	0x0C
	Report Server ID	0x11
	Read Device Identification	0x2B

The CRC is a way to control if the message sent is perfectly equal to that received, it is obtained by cyclically comparing a byte of the message with the previous one.

Every bit of each byte is sent at the end of the other, therefore in order to read the exact message the slave has to be synchronous with the master; it means that the sending frequency has to be equal to the reading one.
Bit rate can range from a minimum of 2400bit/s to a maximum of 115000bit/s.9600buad and 19200buad are the most used bit rate in industrial application.

An example without CRC of Modbus RTU message could be (decoded in decimal):

7 3 50 10

This means that the master wants to read 10 registers starting from the 50th from the slave 7.

The answer will be the repetition of the first two parts (his slave address and the function requested by the master) and the value stored in the registers 50÷59.

1.2) ETHERNET TCP/IP

The name Ethernet derives from the combination of the Greek word aether (to burn/shine) and network.
It is a complex networking technology, invented in 1973-1975 by Xerox, that defines wiring and signalling standards for the physical layer and the data link layer of the ISO/OSI architecture (see picture below).

APPLICATION LAYER	**HTTP, SMTP, SNMP, FTP, Modbus, IEC 61850, DNP…**
PRESENTATION LAYER	**XDR, ASN.1, SMB, AFP, NCP…**
SESSION LAYER	**ASAP, TLS, SSL, ISO 8327 / CCITT X.225, RPC, NetBIOS, ASP…**
TRANSPORT LAYER	**TCP, UDP, RTP, SCTP, SPX, ATP…**
NETWORK LAYER	**IP, ICMP, IGMP, IPX, OSPF, RIP,ARP…**
DATA LINK LAYER	**Ethernet, Token ring, STP, Frame relay, ISDN, ATM, 802.11 WiFi…**
PHYSICAL LAYER	**10BASE-T, 100BASE-T, 1000BASE-T, various 802.11 physical layers**

Old Ethernet II is still used together with the new IEEE 802.3 standard.

IP provides a protocol for the subdivision in different networks while TCP defines rules for the division of data in packets.
Together Ethernet + IP + TCP forms the well-known standard Ethernet TCP/IP used at world level for web communication.

Originally Ethernet used CO-Axial cable that presented these problems:

- Not reliable for large extended networks
- Damage of a single connector, could make the whole Ethernet segment unusable.
- Prone to electrical discontinuity (signal reflections)
- Difficult to debug
- Information is transmitted across a common medium to all, therefore a single node can eavesdrop on all traffic.
- Use of a single cable means that the bandwidth is shared, so that network traffic can slow to a crawl when, for example, the network and nodes restart after a power failure.

Nowadays there are three common standards: 10BASE-T, 100BASE-TX: Fast Ethernet, 1000BASE-T: Gigabit Ethernet; the main difference is the transmission rates respectively: 10Mbits/s, 100Mbits/s, 1000Mbits/s.
All of these three standards use the same connectors (RJ-45) and cable type (Category 5

or 6: four pairs of twisted cables).

1.2.1) MAC Address

In serial Modbus to distinguish a slave from another one the slave address is used; whit Ethernet this role is took in charge by an unique identifier called MAC address (Media Access Control Address).
It is composite by six bytes (48 bits), the first two identifies the manufactorer and the last four the device itself.
The first byte of every existing MAC distinguishes unicast and multicast messages: when a destination MAC starts with an even number only the device with exactly that MAC address will receive the message; instead when the first byte of the destination MAC address is an odd number, all terminals connected together to the local area network will receive the message.

1.2.2) IP Address version 4 (IPv4)

IPv4 is made up of four bytes (32 bits).
It is assigned by the network administrator (unlike MAC, it is exchangeable) and it is the Internet address.
The value of each byte is between 0 and 255.
The first part identifies the network at which this device belongs, named Net.ID, and the rest is used to distinguish different components of the same network, called Host.ID.

Routers are the devices used to forward messages in the right network, they don’t know the host id, but only the net id, to act in this way, a subnet mask is used.

1.2.3) ARP (Address Resolution Protocol)

Ethernet only knows MAC addresses; for this reason a protocol that converts IP addresses in MAC ones is needed; this task is realized by ARP.
It works according this principle:

1. First the station sends a request in broadcast mode at MAC level asking who has a specific IP address.
2. The station concerned answers with its MAC address.
3. The association IP/MAC is stored in a cache ARP
4. Every 30s/2min this cache is erased if there is no communication.

In a network, the IP of a device has to be unique, ARP allows to prevent the double address IP: in fact, as soon as a new device is installed in a network it sends a request ARP with its own address (normally this request must stay without answer).

1.2.4) TCP

Just as every IP device has an IP address, each application inside each terminal(host) on the LAN has a port number; this is needed to allow distinguish different application inside the same host TCP port allows to a single Ethernet stack to handle data for many different applications at the same time.
Some protocols, the most important ones, are assigned to a well-known TCP port number

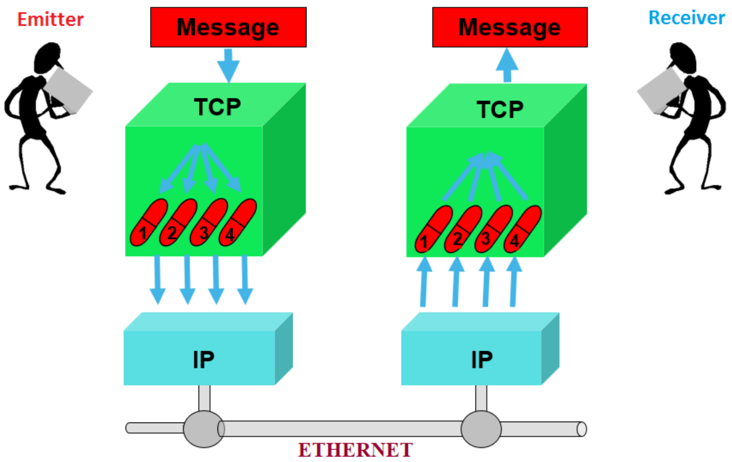
like FTP port 21, HTTP port 80, Modbus port 502, IEC 61850 port 102; other protocols use TCP port number > 2000 which can be freely used. The TCP port must be known by the receiver.

Besides introducing TCP ports to distinguish different applications on the same host, TCP (Transmission Control Protocol) allows a reliable transport of data from a device to another one by:

- Transporting data along a socket using sequence numbers to record the amount of data sent.
- Implementing a send system where each packet of data is acknowledged.
- Implementing an adaptive retry system where a lost packet can be resent without having to abort the entire transmission.
- Provides re-ordering of the IP datagrams at the destination based on sequence numbers - this is needed as IP datagrams can arrive in an order different from the order they were sent.
- Providing flow control that prevents a receiving device from being overloaded with data.
- Providing a checksum on the header and data sent.

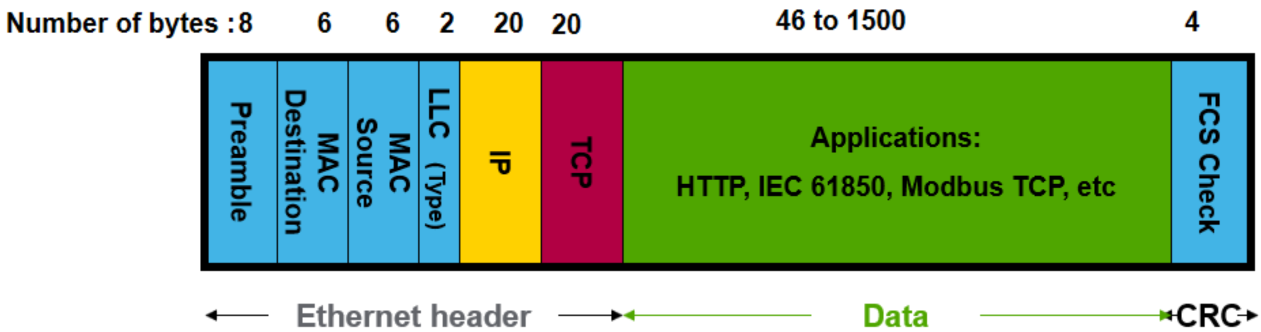
To guarantee a correct communication between two device a TCP socket connection must be established before data can be sent and closed after all data is sent.

Long message can't be sent at once (it will saturate the net), for this reason TCP is so important: it will divide a single message in shorter ones, sent them, reorder them at the receiver, rebuild in this way the original message



1.2.5) Modbus TCP/IP

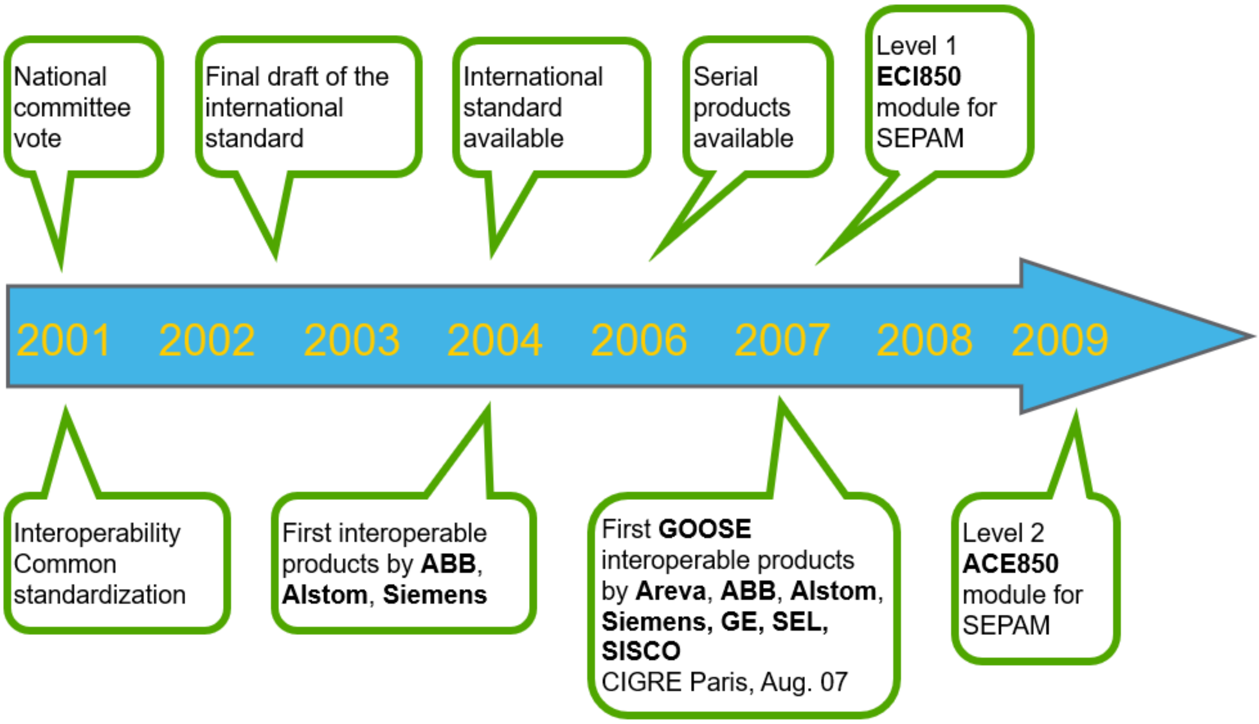
All protocols based on Ethernet TCP/IP are encapsulated in the application section of the freame.



Modbus TCP/IP is the encapsulation of Modbus RTU serial frame inside an Ethernet TCP/IP frame. The original Modbus RTU message has been changed a little bit, mainly due to the elimination of the Modbus CRC

1.3) IEC 61850

IEC 61850 is a structure headed by IEC TC 57 that defines functionalities and description language of devices, with a common communication system architecture inside a substation and a standardized method to access data.

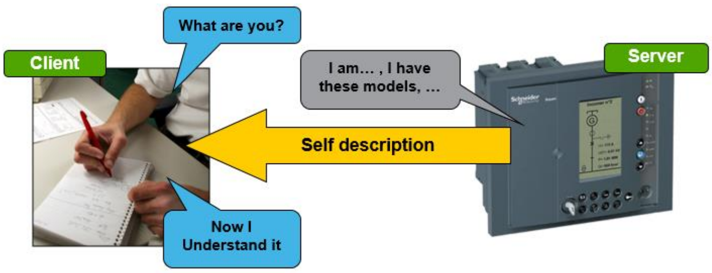


The main improvements that this protocol has brought are: it is self-descripted, time stamped information and memorisation of data when communication is down, fast interlocking and tripping over Ethernet LAN and waveforms transmitted in real-time.

This means that every single event saw by a device will be read by a client with the exact time (stored in the device) at which it happened and it can be send delayed in time, without losing any information.

1.3.1) Self-Description

All 61850 devices called IED (intelligent electronic device) have a standardized description language: Substation Configuration Description Language (SCL) based on XML (Extensible Mark Up Language).



The description defines:

1. The device in the system, that carries his name, his address and the subnet to which it belongs.
2. The data available for each device, based on the library of ICD files and the device type.

3. Transmission behaviour, distinguished in reports, events, periodic, on update.

Thanks to Self-description the commissioning time is reduces a lot (up to 75%) and configuration errors are eliminated due to the fact that they are already ready to be used. This thing is possible only thanks to a sequence of more and more detailed queries:

- Get Logical Device Directory

  Gets a list of the Logical Nodes within the device
- Get Logical Node Directory

  Gets a list of the object references for each class in the LN
- Get DataSet Directory

  Gets a list of the DataSets within the device
- Get Data Directory

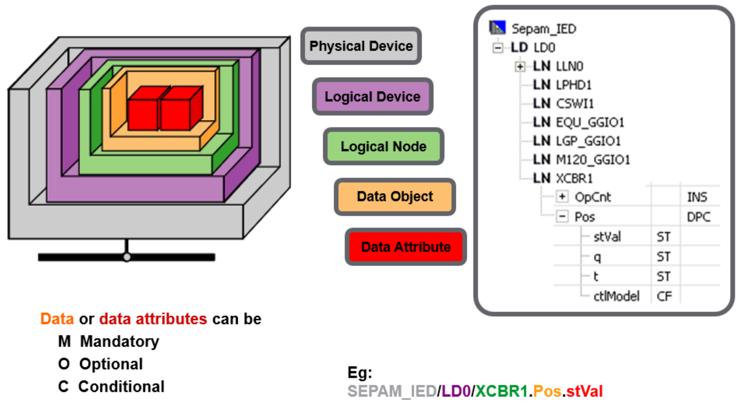
  Gets a list of the data attributes for the objects in the LN
- Get Data Definition

  Gets a list of the definitions of the data attributes for the objects in the LN

1.3.2) Structure of IEC61850

One of the best improvement carried by IEC61850 is a standard to define functionalities of devices: it consists of a normalized hierarchical view of data and functions that are no more store in registers as for Modbus.
IEC 61850-7-4 defines for any Physical Device:

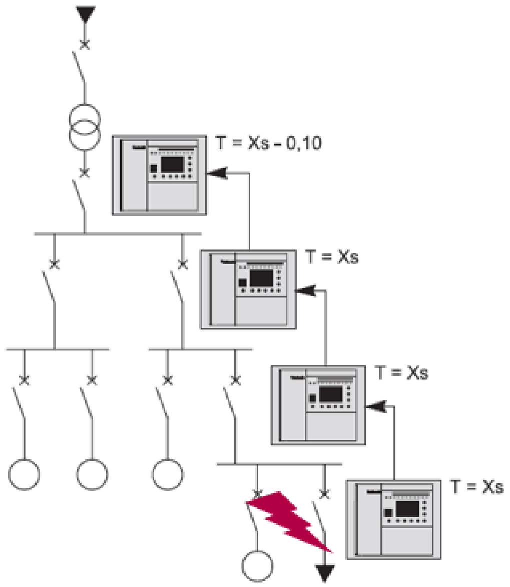
- 13 Logical Device subdivided into 91 different Logical Nodes (LN)
- Every Logical Node contains one or more Data Objects which conform to a common data class specification
- Every Data Object contains one or more elements of Data Attributes
- Data Attributes conform to a functional constraint that specifies what services can be used to access the data



1.3.3) Dataset

Dataset is a group of data objects used for reporting and logging purposes.
It is stored on both the client and the server; in this way only data values need to be transmitted.

1.3.4) GOOSE Messages



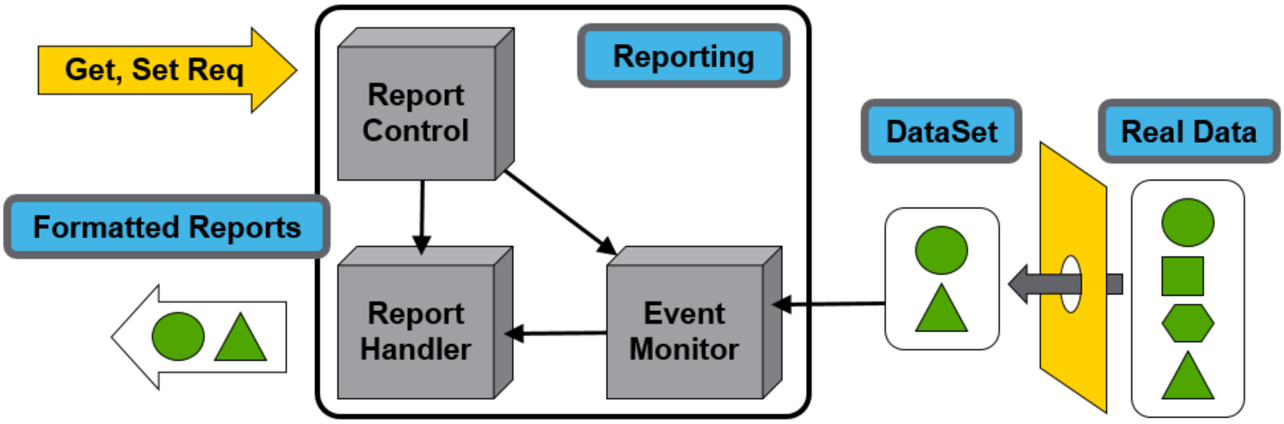
IEC 61850 can handle logic discrimination: when a fault appears, the protection unit blocks the upstream ones from tripping.
This feature is available thanks to multicast messages called GOOSE.
This kind of communication is also used to handle inter-tripping, interlocking, load shedding or whenever devices need to share data.

GOOSE stands for Generic Object-Oriented Substation Event and it is a way to exchange real-time data between devices in an automated electrical substation.
On conventional installations, all GOOSE functions are still often provided via direct digital dedicated wiring, which implies a lot of cables.
GOOSE transmissions are based on broadcasting data: publishers, also called producers, send GOOSE messages on multicast Mac addresses; subscribers, also called receivers, just listen to what is going on in the network, to pick what they need.

Every change of status of a variable present in the dataset assigned to a GOOSE message will cause the message to be sent with the new value.
Due to the fact GOOSE is a broadcast message, subscribers will not send back reception confirmation.
To avoid losing of information the message is sent several times in the first instants and then less and less frequently, till the max delay set.

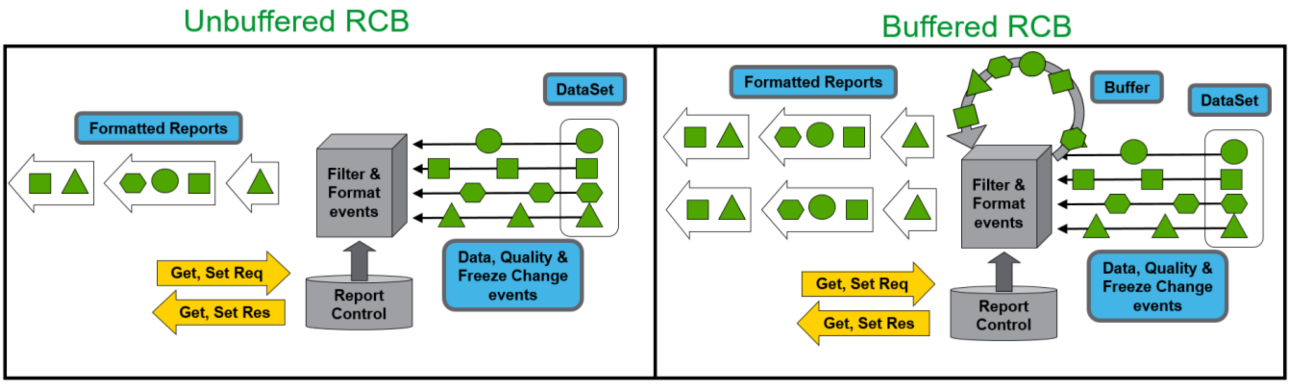
1.3.5) Reporting

Reporting allows to send a dataset in case of some events.
Every report is owned exclusively by a single client.
There are several conditions to send reports: the change of the value of the data in the data set, the change of the quality attribute, periodically.



The Event Monitor controls the real data referenced in the dataset and informs the Report handler when an event occurs.
The Report Handler decides when and how send the report to the subscribed client.

There are two kinds of Reports; buffered and unbuffered.
As the name suggests the difference is in the fact that with unbuffered reports, on event, the report is sent immediately and there is no check of the receiving of this message by the client; otherwise with buffered ones the message is either sent immediately or buffered in case of bad connections to be re send when connection becomes OK.
Moreover, reporting allows Sequence-of-Events (SoE), restoring all information lost during a lack of communication as soon as the “dead” device is restored.



PART 2

SUPERVISION AND CONTROL OF AN ELECTRIC DISTRIBUTION SYSTEM

2.1) INTRODUCTION

The industrial protocols described in part 1 can be used to realize the supervision and the automation of the electrical distribution system of an industrial plant. To be able to describe most of the automation that can be normally found, a plant with need of high availability of the electrical power supply has been considered.
Therefore, the plant will have two separate medium voltage supply lines and emergency diesel generators as well.

The plant will have a low voltage distribution system as well, with 20 different lines protected by circuit breakers with communicating electronic trip unit.
The twenty final lines are fed, in normal condition, half by one transformer and the rest part by the other one.

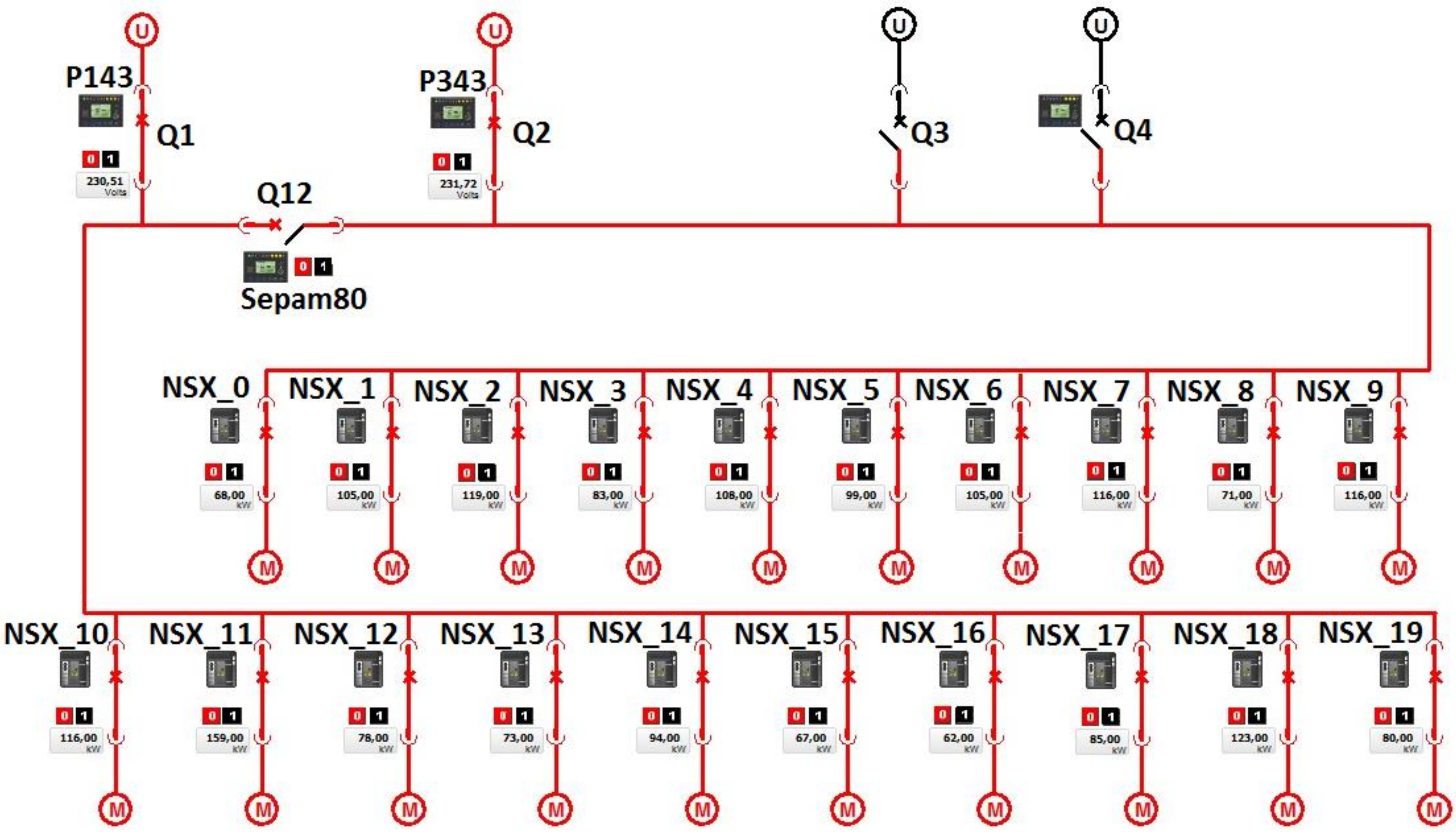
Whenever possible IEC61850 has been used, both for supervision and for automation, to show how this recently introduced protocol can be used in an industrial plant and not only in substations.

The focus is mainly on the continuity of service; in fact, two emergency groups have been designed in this realization.

The automatism elaborated in this work in order to reduce the interruption of the power supply as much as possible, are:

- Automatic Transfer System (ATS)
- Automatic starting of emergency groups
- Load Shedding
- Load Sharing

In the follow, a scheme of the power distribution of the plant is reported.



P143
Q1
230,51 Volts
P343
Q2
231,72 Volts
Q3
Q4
Q12
Sepam80
NSX_0
68,00 kW
NSX_1
105,00 kW
NSX_2
119,00 kW
NSX_3
83,00 kW
NSX_4
108,00 kW
NSX_5
99,00 kW
NSX_6
105,00 kW
NSX_7
116,00 kW
NSX_8
71,00 kW
NSX_9
116,00 kW
NSX_10
116,00 kW
NSX_11
159,00 kW
NSX_12
78,00 kW
NSX_13
73,00 kW
NSX_14
94,00 kW
NSX_15
67,00 kW
NSX_16
62,00 kW
NSX_17
85,00 kW
NSX_18
123,00 kW
NSX_19
80,00 kW

2.1.1) Automatic Transfer System

In normal working condition,—— the general switches Q1 and Q2 are closed, instead the bus-bar one Q12 is open; as showed in Figure1.

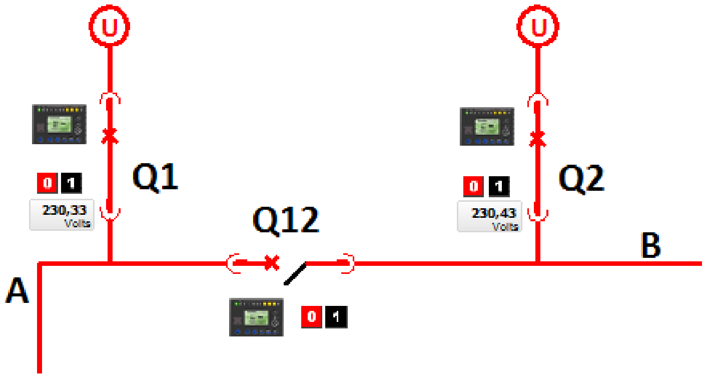


Figure 1

In this configuration, the load is splitted between the two transformers of the two lines, which are two independent system.

Only with this starting point in case of lack of voltage in one of the two lines the automatic transfer system (ATS) can start, obviously only if it is activated.

When the protection relay of one of the two lines see a lack of voltage, after a delay pre-set, the protection 27 will intervene and send the open command at its own breaker.

Without the intervention of ATS all the bar A or B is powered off, instead with ATS as soon as one of the two breakers opens the relay which commands Q12 orders it to close itself. In this final configuration, according to ATS, all the loads connected to both the bars A and B are powered throw only one transformer, as showed in Figure2.

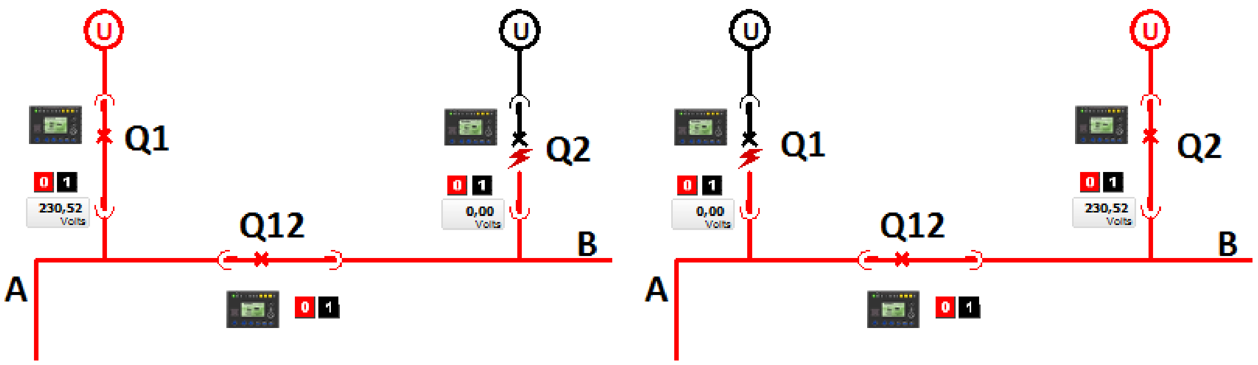


Figure 2

Reset to initial condition

The initial condition can be restored by sending to the system a reset command.
The invers ATS can start only if there is voltage in both the lines (no protection 27 on in both of the line relays), if only one between Q1 and Q2 is open and if Q12 is closed.

In this starting condition after the command of reset and with the ATS activated, the relay which commands the open breaker will sent the close command to it, bringing the system in the temporary condition of “short parallel”; in fact, all the three switches are closed now, as showed in Figure3.

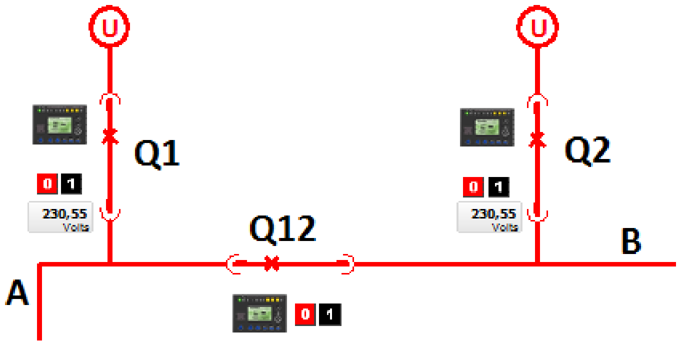


Figure 3

After a short time, to avoid to put permanently the two transformers in parallel, the relay connected to the bus-bar breaker will send the open command to Q12 restoring the original configuration in which the load is splitted in two separate systems A and B.

This procedure allows to reset the system keeping the loads always fed.

Manual Operations

When the ATS is disabled, it is possible to manually “move” the load from both the lines to only one and restore the initial condition too.

The starting point, that allows this procedure, is the same of that described in direct operation of ATS (Figure 1): both lines switches Q1 and Q2 closed, bus-bar Q12 open and presence of voltage in both lines.

In this condition, it is possible to send one of the two manual commands BAR_A or BAR_B that allows the sequence to start.

The relay of the bus-bar breaker will send to it the close command, bringing the system in the “short parallel” configuration already described in the invers operation of ATS (Figure 2); then the relay of the breaker of the line that has to be powered off (Q1 if the command is BAR_B, Q2 if it is BAR_A) send the open command to it; the system is now as showed in Figure4.

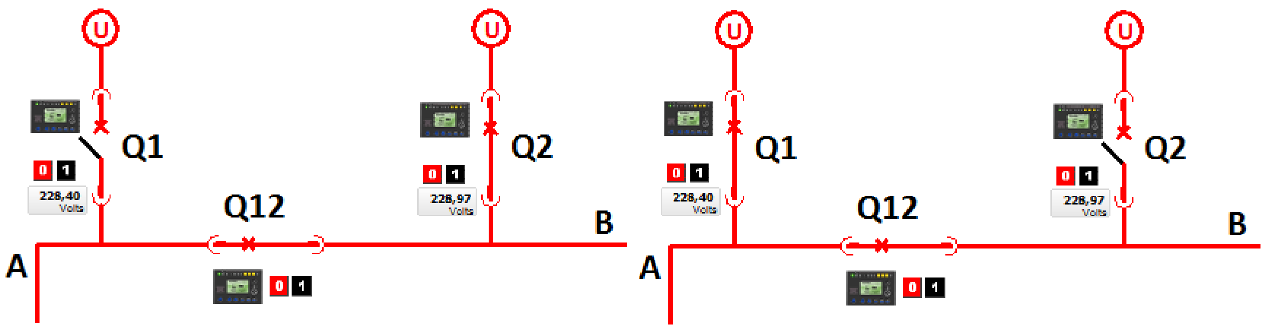


Figure 4

Starting from one of these two configuration, it is possible to restore the starting condition by a manual command of reset (Reset_Man); the procedure will be exactly equal to the one described in “invers operation” of ATS.

If the system is in one of the two cases showed in Figure 4, to transfer the system to the other line a reset procedure has to be done before send the command BAR_A or BAR_B. No direct transfer between the two lines is allowed.

For example, if the loads are all fed by the second line (Figure 4.a) to manually move the loads to the first line (Figure 4.b), it is necessary to reset the system through the reset command (bringing the system in the “normal” condition Figure 1) first and then it is possible to give the command BAR_A.

2.1.2) Emergency Generation

To increase the availability of the system two groups of emergency generation are been planned.
Each of these two groups is realized by a synchronous 4 poles generator coupled with a diesel motor.
In normal condition, the two motors are off and the breakers are open (Figure5).

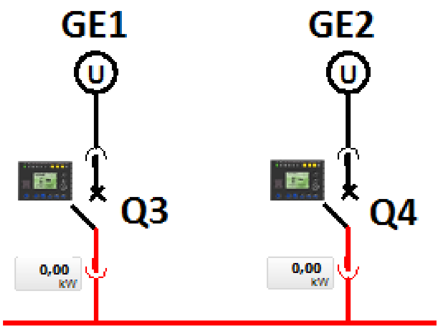


Figure 5

As soon as the ATS has performed its procedure, a command of start will be automatically sent to the first generator (GE1).
It will reach the nominal speed of 1500rpm (that corresponds to the nominal frequency equal to 50Hz); once the speed transient is over the generator control system will send the state of “ready” (Figure 6), which will be read by a synchronizer.

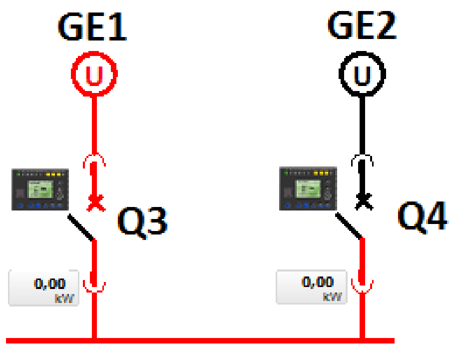


Figure 6

The synchronizer will command the motor to set his speed in such a way that the frequency of the synchronous generator is greater than that of the grid by 0.1Hz (if the grid has a frequency of 50Hz it correspond to a speed of 1503rpm).

As soon as the two voltages, measured one at the bas-bar and one at the generator side, are in phase, the synchronizer will send the close command to the relay which command the switch Q3; at this point the set of the generator power will be equal to the difference between the actual power required by the loads and eighty percent of the rated power of the transformer. (Figure 7)

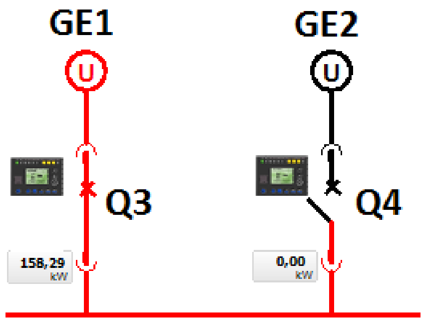


Figure 7

When the ATS has been reset, automatically the system will send an open command to the switch Q3 and then it will stop the diesel motor.

Starting from the condition showed in Figure 4, if a lack of voltage in the second line will happen, the emergency generator enters in the islanding mode.
In this configuration, the only aim of the emergency generator controller is to keep the frequency as stable as possible

When the generator is feeding the loads stand alone, it is possible to reclose the switches Q1 and Q2 by the command Reset_Off_Grid.
Only if at least one of the two lines voltage is present (no 27-protection trip present), the reclosing procedure can start.

A synchronizer brings the diesel motor speed at a value that allows a difference of frequency (f(grid)-f(ge)) equal to 0.1Hz and then when the two voltages are in phase, it orders the reclose of the switches on lines with presence of voltage.

If both of the switches are closed the inverse ATS starts: Q12 will be open and then reset the ATS, so Q3 will open too; instead if only one of the two switches is closed, then the cogeneration is restored.

2.1.3) Load Shedding

If the loads require a power that is too high for the correct function of the system a part will be eliminated by opening the switches which control the twenty lines following the rules of Load Shedding (LS).

The causes that make the load shedding starts are:

- Intervention of ATS

  After ATS has brought all loads under a single line, an amount of power equal to the difference between the actual power required by the loads and the rated power of the transformer, which is dimensioned to supply only half of the total loads, has to be detached.

- Lowering of frequency (Generators stand-alone)

  Mainly during the passage from cogeneration to the only alimentation of emergency generators, following an intervention of the protection 27 on both the two lines, the amount of power delivered by the GEs has to grow immediately in order to make up the lack of the main line.
  This “jump” of power implies a drastic reduction of frequency; the GE can keep frequency inside acceptable values till this difference of power is equal to half his rated one.
  The difference between the actual “jump” and half of the rated power of the GEs has to be detached.

- Generators Overload

  Generators cannot stand with a power higher than his rated one; after a pre-set time the difference between the actual power delivered by the generator and his rated one has to be detached.

Thanks to the measure of the power requested by every line, a very accurate LS can be done.

Rules that LS follows are:

1. Priority

   Every Load has its own priority number (from 1 to 5), which identifies the importance of that line; higher is the number less important is its continuity of service and more easily will be detached.

2. Number of load

   Analysing loads having the same priority, LS will detach number of load as less as possible.

   Following this approach, it is better to detach one load which is absorbing 100kW than two of 50kW

3. Power

   The aim of LS is: detach load till reach a difference of power that is the lower possible.

2.1.4) Load Sharing

Without reattaching any load after the sequence of load shedding due to the ATS, the second group of emergency generation is totally useless.
Since it is possible to reclose manually the switches of the 20 lines after their automatic opening, only one group isn't capable to sustain the power to keep the absorption of the grid less than 80% the rated power of the transformer and much less it is capable to supply the entire power loads when it is in islanding mode.
For this reason, when the power delivered by the first emergency group is greater than 75% his rated ones, an automatic procedure will start the second synchronous machine.

The way in which this emergency group is attached to the bus-bar is exactly the same of the first one (described in "emergency generator").
The unique difference is that the synchronizer is going to see the difference of frequency between the two generators and not between GE2-grid; in this way, independently of the fact that the first generator was or wasn't in islanding mode the correct frequency, and as consequence the correct speed, is always obtain.
After this the two lines are in parallel. (Figure 1)

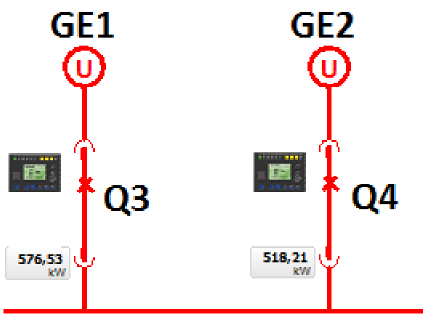


Figure 8

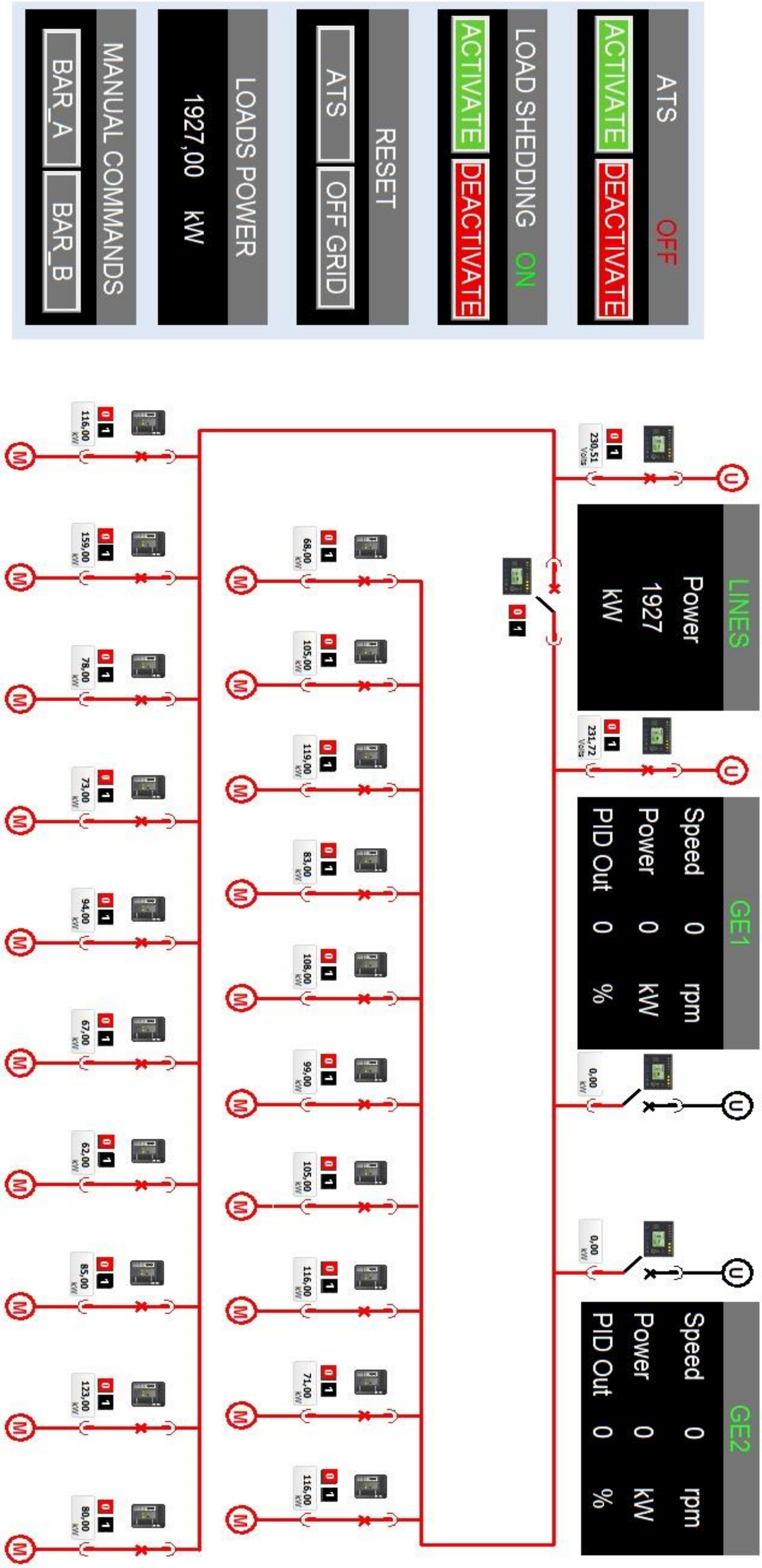
Without any cross-check system, the two generators will deliver different powers which means overload one of the two, that obviously is an incorrect function mode.
The load sharing aim is exactly this, keep the generation equally for the two lines.
To act like this, the overall power that the two groups have to deliver is computed and then splitted equally as set of power of both of them.
In this way after a transient, the power delivered by the second generator is the same as that produced by the first one in every situation. (Figure 2)

Figure 9

Starting from the condition of Figure 2, if the power delivered by the first emergency generator decreases under 20% of its rated power, after a pre-set delay, the relay which commands the switch Q4 sends to the open command it.

2.1.5) PSE

The status of the entire system of electrical distribution is showed in a graphic page of Power Scada Expert.



In this page, it is present

the status of:

- Q1: first line general switch (open/closed/tripped)
- Q2: second line general switch (open/closed/tripped)
- Q12: bus-bar switch (open/closed/tripped)
- Q3: first emergency line switch (open/closed/tripped)
- Q4: second emergency line switch (open/closed/tripped)
- The 20 load switches (open/closed/tripped)
- ATS (on/off)
- Load Shedding (on/off)

Command:

- Open of each switch visualized on the page (“0”)
- Close of each switch visualized on the page (“1”)
- Activate ATS (“Activate” in ATS section)
- Deactivate ATS (“Deactivate” in ATS section)
- Activate LS (“Activate” in LS section)
- Deactivate LS (“Deactivate” in LS section)
- Reset ATS (“ATS” in reset section)
- Reset Islanding mode (“OFF GRID” in reset section)
- Transfer load on bar A (“BAR_A” in manual commands, visible only when this command is available)
- Transfer load on bar B (“BAR_B” in manual commands, visible only when this command is available)
- Transfer load on both bars (“MANUAL RESET” in manual commands, visible only when this command is available)

Measures of:

- Power required by loads (“LOADS POWER”)
- Power delivered through the transformers (“LINES POWER”)
- Voltage level of line 1 (Indication of relay of line switch 1)
- Voltage level of line 2 (Indication of relay of line switch 2)
- Power delivered by GE1 (Indication of relay of emergency line switch 1)
- Power delivered by GE2 (Indication of relay of emergency line switch 2)
- Rotational speed of synchronous generator 1 (“Speed” in GE1 section)
- Rotational speed of synchronous generator 2 (“Speed” in GE2 section)
- Power of diesel motor of emergency group 1 (“Power” in GE1 section)

- Power of diesel motor of emergency group 2 (“Power” in GE2 section)
- Out indication of the PID controller sent as impulses to the diesel motor 1 (“PID Out” in GE1 section)
- Out indication of the PID controller sent as impulses to the diesel motor 2 (“PID Out” in GE2 section)
- Power required by a single load (Indication of relay of every load line switch)

2.1.6) Human Machine Interface

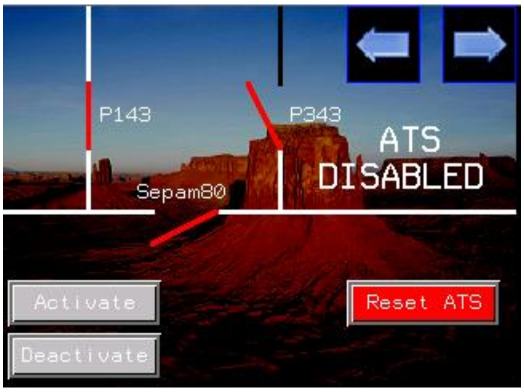
Some of the variables present in the PSE are also available in an external device: Magelis HMISTU655.
It is a touch screen display HMI (human machine interface).

The aim to use a HMI is to have some useful information and some commands displaced among the place of installation of the system.

Three graphics pages have been programmed for this device:

- ATS
- Emergency Generation
- Load shedding and Powers

ATS (Page 1)



In this page, the management of the ATS is available:

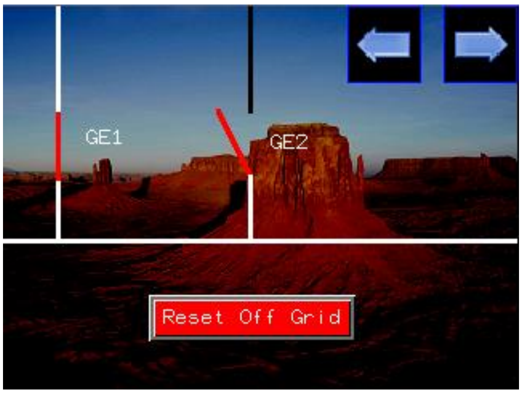
- Status of Q1 (open/closed)
- Status of Q2 (open/closed)
- Lack of voltage line 1 (line black/line white)
- Lack of voltage line 2 (line black/line white)
- Status of Q12 (open/closed)
- Status of ATS (activate/disabled)
- Command Activate ATS
- Command Deactivate ATS
- Command Reset ATS
- Command transfer load on bar A (visible only when this command is available)
- Command transfer load on bar B (visible only when this command is available)
- Command reset load on both bars (visible only when this command is available)

Emergency Generation (Page 2)

In this page, the management of the Emergency Generation is available:

- Status of Q3 (open/closed)
- Status of Q4 (open/closed)

- Generator 1 ready (line white/line black)
- Generator 2 ready (line white/line black)
- Command reset islanding mode
- Written “GE in islanding mode” when this condition is true



Load shedding (Page 3 upper part)

In this half page, the management of the Load shedding is available:

- Command Activate LS
- Command Deactivate LS
- Status of LS (activate/disabled)

Powers (Page 3 lower part)

In this half page, the display of the powers is available:

- Power request by loads
- Power delivered by transformers
- Power delivered by the first emergency group
- Power delivered by the second emergency group

2.2) Realization description

In this chapter it is described how the automation functions have been implemented.

The first part treats the implementation of:

- Sepam80 logic used to treat
  - ATS
- PLC M580 used to treat
  - Load Shedding
  - Emergency generation
  - Simulation

And then the second one describes the connections:

- Physical connection
- Protocol communication

As concern the hardware architecture several devices have been used.

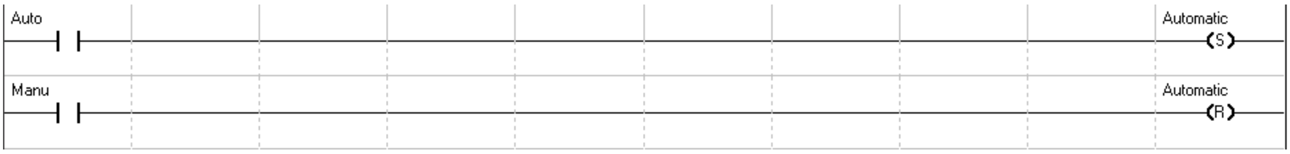
To demonstrate the good interaction between different devices ensured by the use of IEC 61850 protocol; the relays dedicate to protects the main lines (commanding breakers Q1, Q2, Q12) are from two different series: Micom P143 on line1 P343 on line2 and Sepam80 acting on the bus-bar breaker.

Instead, each of the 20 line breakers is controlled by a relay NSX.

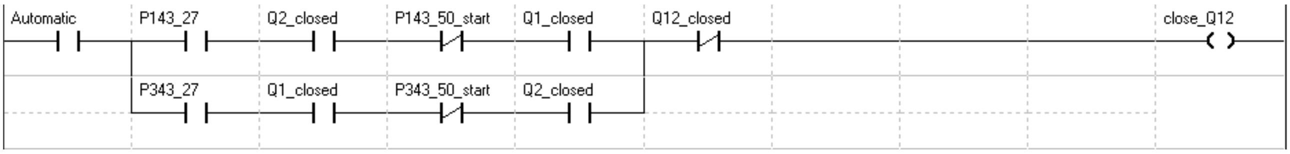
2.2.1) Automatic Transfer System

Sepam80, the relay which controls the bus-bar switch, will control the ATS process described previously.

External inputs, from PSE and HMI, decide if the system is in automatic or manual mode; this information is stored in a memory



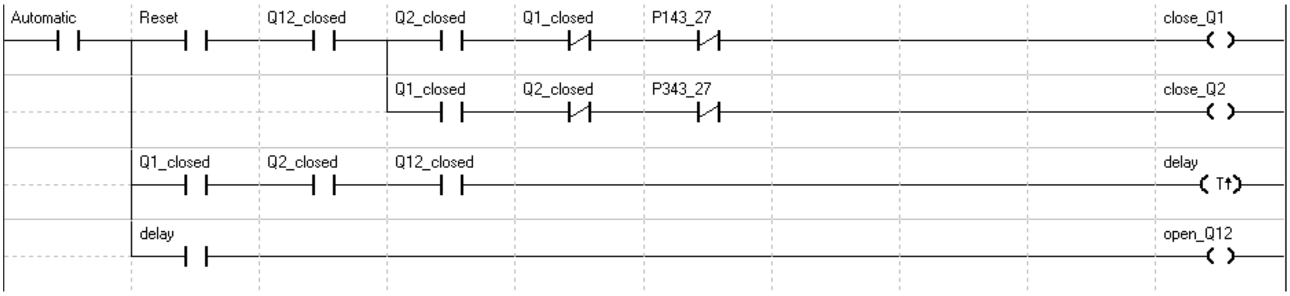
In case of automatic mode, the opening of the lines switch is done by the intervention of the line relays in consequence of a lack of voltage; the only operation that Sepam80 has to do is verify this condition and close his switch.



The initial condition that allows to close the bus-bar switch is no presence of overcurrent in the line in which the 27 protection is going to intervene, both of lines switches closed and obviously Q12 open.

Reset to initial condition

Instead the reset process is controlled in this way:

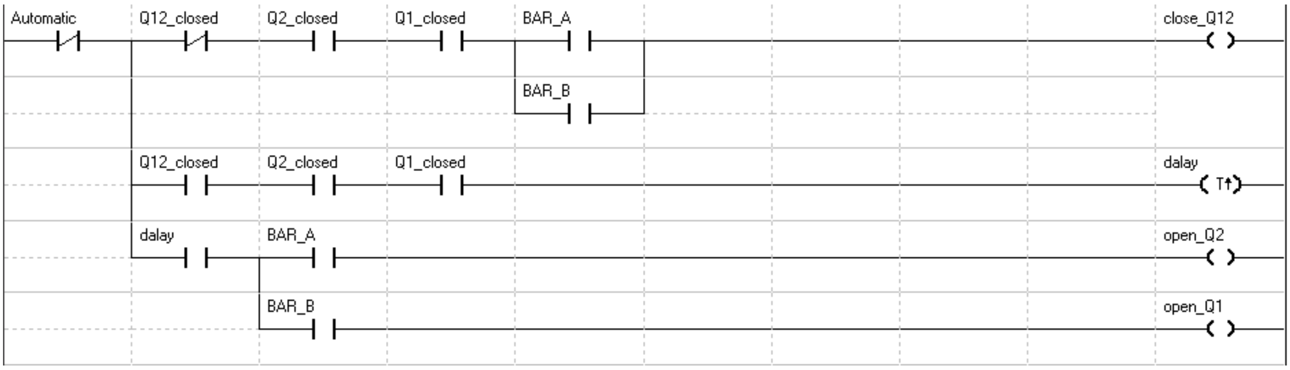


If the mode is automatic, the bus-bar switch is closed, the reset command from PSE or display is send, only one between Q1 and Q2 is open and without lack of voltage; Sepam80 will sent to the relay which command the open line switch the close command, as soon as all three switches are closed, after a pre-set delay (it could be set to 0s to avoid parallel between the two transformers) it will open Q12.

Manual Operations

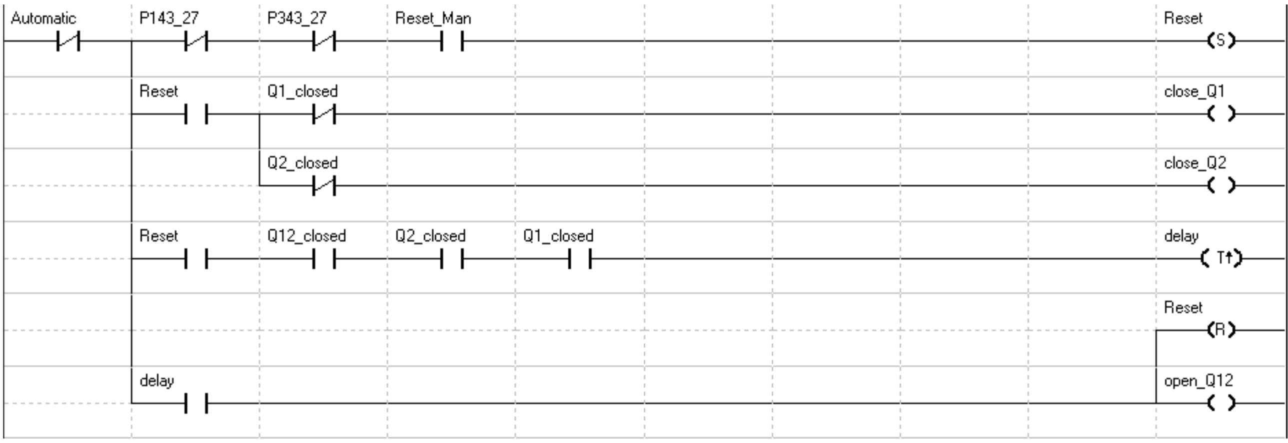
As regards the manual operations, the command BAR_A, BAR_B and Reset_Man are visible on the display and on PSE only when their command is allowed, and it is reset when the transfer process is finished; for this reason, it wasn't foreseen a block logic to prevent incorrect action on the system.

As concern the direct manual operation (command BAR_A and BAR_B) the logic implemented is:



Being in the starting condition both of line switches closed no check on presence of lack of voltage or overcurrent is required, the first line describes the immediately close of Q12 after a transfer command.
When the system is manual mode and all three switches are closed after a pre-set delay (it could be set to 0s to avoid parallel between the two transformers) a command of open is sent to the relay of the line that has to be powered off (Q1 if the command is BAR_B, Q2 if it is BAR_A).

Instead the reset command is realized following this approach:



In this case the check of presence of voltage is important, because one of the two line switches is open in this starting condition.
If this procedure is allowed the logical condition "1" is stored in a memory called "reset", with his contact closed the reset procedure will sent the close command to the relay corresponding to the opened line switch.
When Q1, Q2 and Q12 are closed after a delay (it could be set to 0s to avoid parallel between the two transformers), Sepam80 will open the bus-bar switch and set to logical "0" the memory corresponding to the reset sequence.

2.2.2) Load Shedding

This feature is controlled by the PLC M580.
In order to follow the rules of load shedding described before, a table with all the loads priorities and powers reordered is required.

The load power is read from every of the twenty line switches and stored in a table in which at this value is assigned two parameters: the line priority and number.

To obtain the final table two consecutive steps are done: in the first the reordering criteria is only based on the priority of every line, instead in the second, starting from the previous one, without changing the priority order, according to the power the final table is obtained.

First table

```
(*table with reordered priorities*)
tot:=0;
prio_max:=5;
k:=prio_max;
x:=0;
(*priorities scan*)
while k>0 do
        potenza[k]:=0.0;
        priorita[k]:=0;
        j:=0;
(*switches scan*)
        while j<20 do
(*lowest addresses assigned to highest priority switches*)
                if NSX[j].priorita=k then
                        Tabella[x].int_chiuso:=NSX[j].InterrChiuso;
                        Tabella[x].priorita:=k;
                        Tabella[x].Numero:=j;
                        Tabella[x].potenza:=NSX[j].P;
                        Tabella[x].inserito:=0;
                        priorita[k]:=priorita[k]+1;
                        x:=x+1;
                        potenza[k]:=potenza[k]+NSX[j].P;
                end_if;
                j:=j+1;
        end_while;
        tot:=tot+priorita[k];
        k:=k-1;

end_while;
```

With two “for” loops the first passage is done:
Knowing that the priority is a number between 1 and 5 (5 is the first to be detached); five times all switches are scanned, every loop an indicator starting from 5 is decreased.
If the priority of the actual switch is the same as the indicator, it is stored in this provisional table.
In an array of five elements the number of switches with the same priority is stored.

Second table

```
(*table with reordered priorities and powers (from highest to lowest) *)
P_tot:=0.0;
u:=0;
while u<20 do
h:=prio_max;
(*priorities scan*)
        while h>0 do
                l:=0;
                partenza:=u;
(*reordering of all switches with same priority*)
                while l<priorita[h] do
                        pos:=partenza;
                        max:=-10.0;
(*switches with same priority scan*)
                        while pos<(partenza+priorita[h]) do
(*reasearch of max power switch not assigned yet *)
                                if Tabella[pos].inserito=0 and Tabella[pos].potenza>max then
                                        max:=Tabella[pos].potenza;
                                        inserisci:=pos;
                                end_if;
                                pos:=pos+1;
                        end_while;
(*assignment of max power switch*)
                        Tabella[inserisci].inserito:=1;
                        Finale[u].Numero:=Tabella[inserisci].Numero;
                        Finale[u].potenza:=Tabella[inserisci].potenza;
                        Finale[u].priorita:=Tabella[inserisci].priorita;
                        Finale[u].int_chiuso:=Tabella[inserisci].int_chiuso;
                        u:=u+1;
                        l:=l+1;
                        P_tot:=P_tot+Tabella[inserisci].potenza;
                end_while;
                h:=h-1;
        end_while;
(*check of presence of value in the first table*)
                if tot=0 then u:=20; end_if;
end_while;
```

Starting from “table one” and thanks to the knowledge of the number of switch with same probability, the reordering according power is easily done:
Until all switches are stored in the final table a first for loop is launched, in this one a second for loop scans all the priority numbers.
Entering in this state we are analysing all switches with same priority, they are all read a number of times equal to their number thanks to a third and a fourth for loop; every time the load with highest power not yet stored is write in the final table.

After this “reordering passages” a table with priority (from 5 to 1) and power (from the highest to the lowest) is obtained.

LINE	PRIORITY	POWER
7	5	116.0
9	5	116.0
8	5	71.0
16	5	62.0
6	4	105.0
5	4	99.0
15	4	67.0
18	3	123.0
10	3	116.0
4	3	108.0
14	3	94.0
2	2	119.0
1	2	105.0
17	2	85.0
3	2	83.0
11	1	159.0
19	1	80.0
12	1	78.0
13	1	73.0
0	1	68.0

Load Shedding

As soon as a load shedding is required, using the final table obtained before, the three rules followed by the LS are realized in this way:

```
(*priorities scan*)
        z:=prio_max;
        posizione:=0;
        while z>0 and p_stacca>0.0 do
(*check: is the power to be detached higher then the total power of this priority group?*)
                if potenza[z]<p_stacca then
(*detachment of all loads with this priority*)
                        zi:=posizione;
                        while zi<posizione+priorita[z] do
                                if Finale[zi].int_chiuso then
                                        set(Finale[zi].apri);
                                        set(Finale[zi].stacca);
                                        p_stacca:=p_stacca-Finale[zi].potenza;
                                        p_staccata:=p_staccata+Finale[zi].potenza;
                                end_if;
                        zi:=zi+1;
                        end_while;

                else
(*STEP 1: detachment loads with power lower then the total one*)
                        zi:=posizione;
                        while zi<posizione+priorita[z] do
                                if Finale[zi].potenza<p_stacca and Finale[zi].int_chiuso then
                                        set(Finale[zi].apri);
                                        set(Finale[zi].stacca);
                                        p_stacca:=p_stacca-Finale[zi].potenza;
                                        p_staccata:=p_staccata+Finale[zi].potenza;
                                end_if;
                                zi:=zi+1;
                        end_while;
(*STEP 2: detachment of the load with the lowest power, higher then the detachment one*)
                        zi:=posizione;
                        min_mag:=99999999999999999999999999999999999999.9;
                        aprire:=100;
                        while zi<posizione+priorita[z] do
                                if Finale[zi].potenza>p_stacca and Finale[zi].potenza<min_mag and not Finale[zi].apri  then
                                        aprire:=zi;
                                        min_mag:=Finale[zi].potenza;
                                end_if;
                                zi:=zi+1;
                        end_while;

                        if aprire<100 then
                                set(Finale[aprire].apri);
                                set(Finale[aprire].stacca);
                                p_stacca:=p_stacca-Finale[aprire].potenza;
                                p_staccata:=p_staccata+Finale[aprire].potenza;
                        end_if;
(*STEP 3: if it is possible, reclosing of switches of "small" power loads opened at step 1*)
                        zi:=posizione;
                        while zi<posizione+priorita[z] do
                                if (p_stacca+Finale[zi].potenza)<0.0 and Finale[zi].apri then
                                        reset(Finale[zi].apri);
                                        reset(Finale[zi].stacca);
                                        p_stacca:=p_stacca+Finale[zi].potenza;
                                        p_staccata:=p_staccata-Finale[zi].potenza;
                                end_if;
                                zi:=zi+1;
                        end_while;

                end_if;
                posizione:=posizione+priorita[z];
                z:=z-1;

(*loads detached memorization*)
                istac:=0;
                while istac<20 do
                finale_mem[0][istac].apri:=Finale[istac].apri;
                istac:=istac+1;
                end_while;
        end_while;
end_if;
```

All priority, starting from the maximum one, are scanned.
An initial check, to verify if the total power of the current group of loads is lower than the total one that has to be reduced, is done.
If it is true, all loads with this priority are detached, else it means that the last switches that will be set to be open is in this group.
To select which are the “final” switches that has to be opened, three passages are done.

Every time one switch has set to be opened the power absorbed by the load is subtracted to the set point prefixed, otherwise added if it is demarked from the open list.
In the first one all loads with power lower than the residual one, are marked to be detached.
In the second one the switch of the load with the lowest power, higher than the residual one, is open.
After this, the power detached is higher than the set point, in the last passage a check, if it is possible to demark a switch set to be open, is done.

An example, in which only a group is underlined, is done:

Power to detach: 170kW		Detached?		
Line	Power [kW]	STEP1	STEP2	STEP3
5	150	YES	YES	YES
9	100	NO	NO	NO
15	25	NO	YES	YES
13	5	YES	YES	NO
0	2	YES	YES	NO
Residual Power [kW]		+13	-12	-5

2.2.3) Emergency Generation

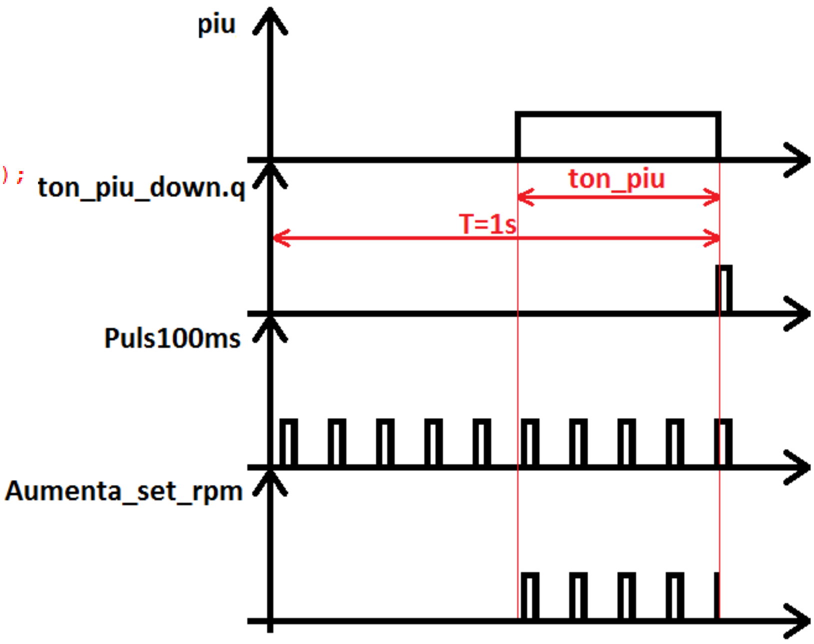
As for the load shedding, the management of the emergency groups is programmed in the PLC M580.
The unique commands that PLC will sent to the motor connected to the synchronous generator are two digital output corresponding to two impulses called “aumenta_set_rpm” and “diminuisci_set_rpm”.
When the logic which control the diesel motor receives one of these two bits, it will change in his internal PID his set of speed.
With “aumenta_set_rpm” it will increase his set point, the opposite with “diminuisci_set_rpm”.
To realize this impulses two couples of timer has been used

```
ton_piu_up (IN := not ton_piu_down.q, PT := t#1s - real_to_time(ton_piu));
ton_piu_down (IN := ton_piu_up.q, PT := real_to_time(ton_piu));
piu:=ton_piu_up.q;

ton_meno_up (IN := not ton_meno_down.q, PT := t#1s - real_to_time(ton_meno));
ton_meno_down (IN := ton_meno_up.q, PT := real_to_time(ton_meno));
meno:=ton_meno_up.q;
```

The analysis will be done only for the first of the two couples, the idea is the same for the other one.
Thanks to this two timers a square wave with period of 1s and duty cycle depending on the parameter “ton_piu” is generated and stored in the variable called “piu”.

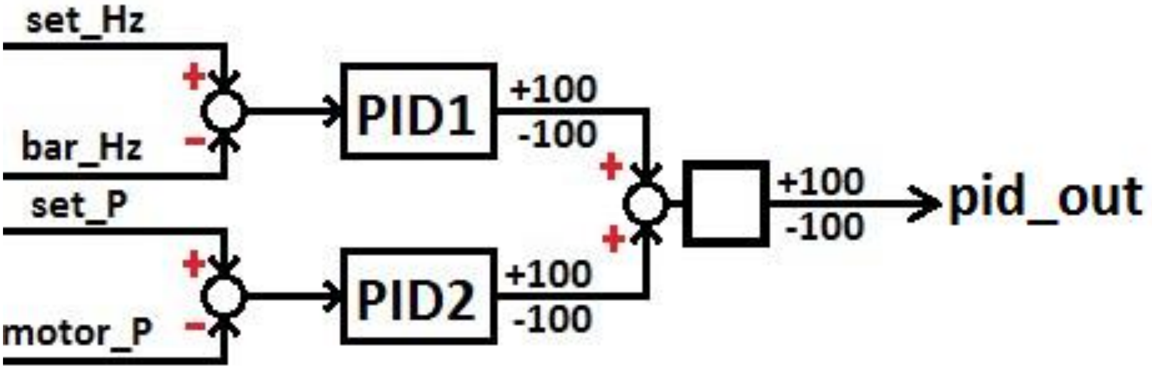
To convert this wave in a series of impulses, an auxiliary variable is needed.

```
Puls_base:=%S5;
Puls100ms := RE (IN := Puls_base);
```



The system bit S5, which is a variable that assume the logical value “1” for 50ms and has a period equal to 100ms, will be stored in an ebool called puls_base.
An ebool variable is composite by two bits, in the first one there is the value assumed last PLC loop and in the second one the current value.
Using the function RE which set to “1” the variable “Puls100ms” every time the two bits of Puls_base are different, an impulse with period 100ms is obtained.
Doing the logical AND between the two variables “piu” and “Puls100ms” our impulses based on the variable “ton_piu” is stored in “aumenta_set_rpm”.

The value of “ton_piu” and “ton_meno” depends on the output of two PIDs which control frequency of the system and power of the diesel motor.
Since these two PIDs are going to control an impulsive output in which the balance is set to 0 only the proportional part will be activate.



The scaled sum of the output of the two PIDs is named “pid_out” if this is a positive number “ton_piu” will be higher than 0 if it is negative, “ton_meno” will. These two variables will time type measured in milliseconds, to make them ranging from 0ms to 1000ms “pid_out” will be multiplicated by a coefficient of 10.

```
if pid_out>0.0 then
        ton_piu:=pid_out*10.0;
        ton_meno:=0.0;
else if pid_out<0.0 then
        ton_meno:=-pid_out*10.0;
        ton_piu:=0.0;
else
        ton_meno:=0.0;
        ton_piu:=0.0;
end_if;
end_if;
```

Considering the different wanted behaviours of the emergency generators described previously, different condition of the two PIDs will be treated.
The following table will show these states of the system, reminding that during the parallel with the grid the generation is thought to permit to the transformer to work at 80% of its rated power.

			Power		Frequency	
Mode	Q3	second group	Activate?	Set [kW]	Activate?	Set [Hz]
	open		NO	-	NO	-
Islanding	closed	off	YES	P_load	YES	50
Islanding	closed	on	YES	P_load/2	YES	50
Parallel	closed	off	YES	P_load - 800	NO	-
Parallel	closed	on	YES	(P_load - 800)/2	NO	-

As concern the second emergency group, same approach has been treated; obviously only the third and the last row of the table have to be considered in this case.

2.2.4) Simulation

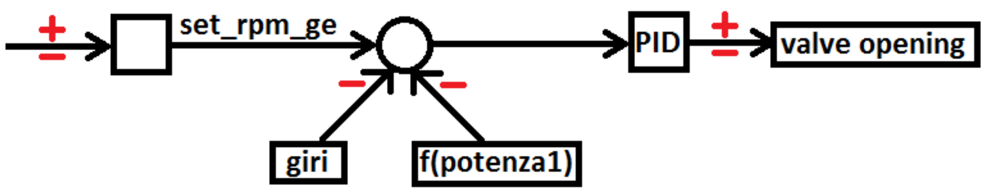
An important point during the realization of one project is the simulation of the system work.
In fact, it is impossible to have all the devices ready to be tested before the commissioning of the project and some parts of them cannot be verified at the end of the work too (e.g. test at 100% the LS is unthinkable, lots of loads have to be supplied and then powered off).
For example, during the realization of this project only the three, ATS involved, relays were physically available, all other component during the test phase were just simulated.
Of course, without any tests no project can be considered “finished”, some sections of the PLC program are dedicated to this part.
After this preliminary part and after finding out everything is working, this simulative part will be deleted and the real variable will be connected.

To clarify the upper described idea, the section regarding the emergency group generator simulation will be analysed.

The digital output “aumenta_set_rpm” and “diminuisci_set_rpm” aren’t, in this phase, real physic outputs but just internal bit memory, when these are equal to 1 the simulation will change the target speed of the motor called “set_rpm_ge”.

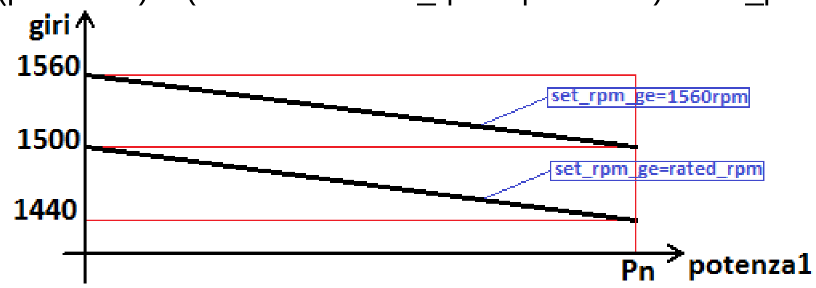
```
if aumenta_set_rpm then
set_rpm_ge:=set_rpm_ge+0.1;
end_if;

if diminuisci_set_rpm then
set_rpm_ge:=set_rpm_ge-0.1;
end_if;
```

Now, two behaviours have to be simulated: the grid in parallel mode and the islanding one.
As concern the first one, it is important to consider that in this case the emergency group isn’t capable to change the frequency, since it is imposed by the grid and it will fit with this. No frequency measures have been done to get the exact value of frequency at which the synchronous machine will work, for this reason, a rated value of 50Hz it is set in this configuration, which corresponds to a speed of 1500rpm, imposed in the variable “giri”.
The internal regulation of the real motor follows the block diagram here reported



In which f(potenza1) is a function depending on the power generated by the diesel motor and simulated by the variable “potenza1”.
This function is realized with the purpose to make the graphic “giri/potenza1” a straight line with angular coefficient depending on a coefficient called “statismo”:

f(potenza1) = (statismo * rated_rpm * potenza1)/rated_power



To emulate this behaviour the invers formula obtained starting from the block diagram has been used to compute every 100ms the power produced by the diesel motor.

```
if int_ge_chiuso and not isola then
        giri:=1500.0;
        if puls100ms then
        (* Pmot= (set_rpm - rpm_attuali) * Pn / (s * rpm_nominali) *)
        potenza1:=(set_rpm_ge-giri)*pn/(statismo*1500.0);
        potenza_imp:=potenza1;
        end_if;
end_if;
```

“potenza_imp” is the generated power by the synchronous generator which is, losses excluded, equal to the produced one.

When the emergency group is producing alone (or in parallel with another GE) all the power required by the load, another factor has to be considerate.
When the power required by loads changes, due to the detachment or insertion of one load or due to a transitory behaviour of the loads, the generated one has to change immediately too, keeping the difference between power in and out always equal to 0.
The real phenomena that happens is that, for example, after an increase of power required, this is immediately provided by the synchronous machine that compensate the difference between the loads power and the diesel power, slowing down his rotor and therefore decreases the frequency of the system; same and opposite if the power required is less than before.
Then the PLC PID seeing a difference frequency from the set point will change the value of “set_rpm_ge” using the physical digital output “aumenta_set_rpm”/“diminuisci_set_rpm”, without changing the power, reports the rated frequency of 50Hz.
To simulate this thing the final configuration, without PID control, has to be thought.
The full motor speed (rpm_regime) following the same block diagram will be:

rpm_regime = set_rpm_ge - ((statismo* rated_rpm * potenza_imp) / rated_power)

Compute this value, every 100ms, the actual speed “giri” will get closer and closer to the final one “rpm_regime” with steps always smaller.
The size of the steps depends on a coefficient called “coef_transitorio”. Power will follow the same trend as it did in the previous case.

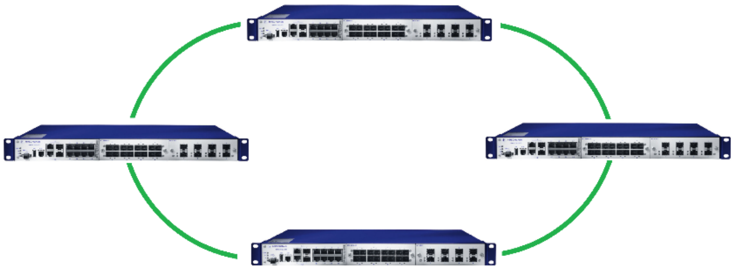
```
if int_ge_chiuso and isola then
        if puls100ms then
        rpm_regime:=set_rpm_ge - (statismo*1500.0*(potenza_imp/pn));
        giri:=giri+ (rpm_regime-giri)/coef_transitorio;
        potenza1:=(set_rpm_ge-giri)*pn/(statismo * 1500.0);
        end_if;
end_if;
```

2.2.5) Physical connection

In order to make all used devices communicating, a physical connection has to be set. The easiest way to realize this is just connecting all devices to a switch.

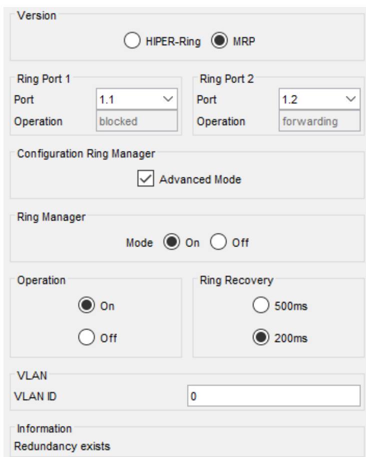
This method is never used for a very simple reason: the total absence of redundancy. In fact, in case of a single failure of a connection, at least one device will be out of communication; moreover, in case of switch failures, all device won’t be able to receive messages.

A first improvement is to use a single ring architecture, in which a series of switches are connected among them forming a circle.



A single fault in the connectors forming the circle does not inhibit the communication. Without doing anything this configuration is not allowed for ethernet protocol; in fact, having more than one path, when a multicast message is sent, it will return to the sending switch, which will resend again it saturating in this way the network.

A lot of protocols are used to control this loop condition (Hyper ring or MRP are used and analysed in this project).



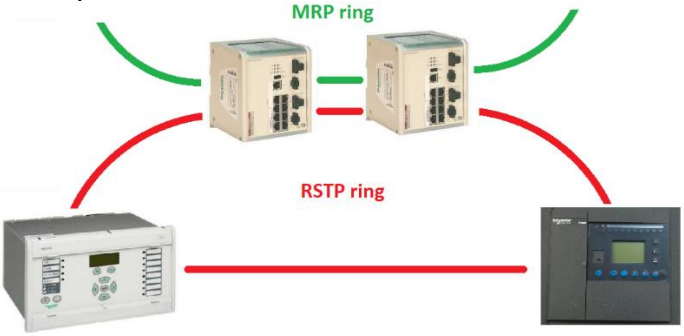
A chosen switch, part of the ring, called “ring master” is in charge to control the ring condition: knowing the two ports belonging to the loop, it will disable one of them restoring the single path useful to make the ethernet protocol working.

To check the condition “first failure on the ring” it continuously sends a control message on the ring; if it receives this message in the other port it means that all connections are good; otherwise there is a lack of communication and it immediately enables the other port restoring the initial condition.

A single ring is not the best solution; in fact a fault outside the ring (connection between a device and a switch) or of a switch causes an interruption of signals.

To avoid this problem too, subrings MRP could be used.

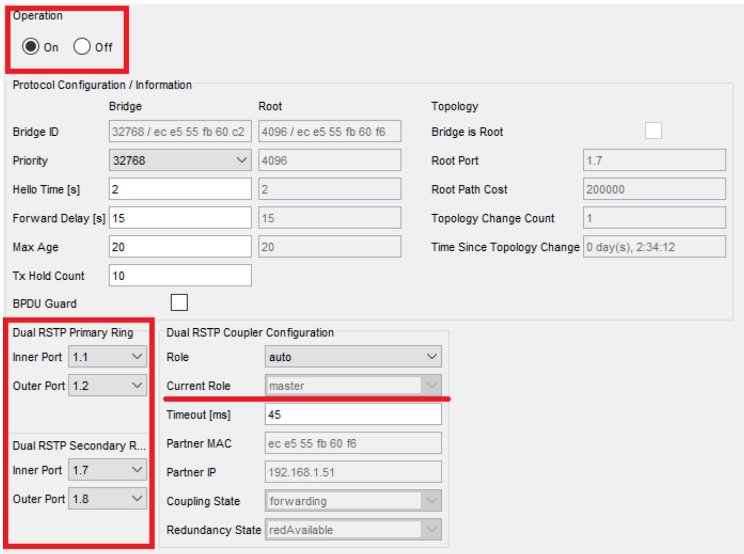
The problem is that devices like Sepam80 don’t support this protocol, but only RSTP. RSTP is a more complex kind of protocol; in fact, it can manage a generic mesh connected system; of course, a loop is a subset of it and so it can be controlled by RSTP too.
In this case a device called “root” will search a single path to reach every device and the connection “in excess” will be opened; the main difference between MRP and RSTP is this one: not only the “commander” of the system enables/disables connection, but all the devices that belong to this process can.



With this configuration, a full redundancy capable to let all devices communicate even after a failure is obtained.

RSTP and MRP cannot coexist in the same architecture, to separate them, particular switches called DRS have to be present in the connection parts of the two rings.
DRS switches act always as a couple: one of them is the master and the other is the slave, when master is working, slave is a passive component as the rest of loop device; in case of a master failure, the slave takes is role.
To act like that, the four ports belonging as pairs to MRP and RSTP has to be initialized and RSTP protocol has to be disabled on the two MRP ports.

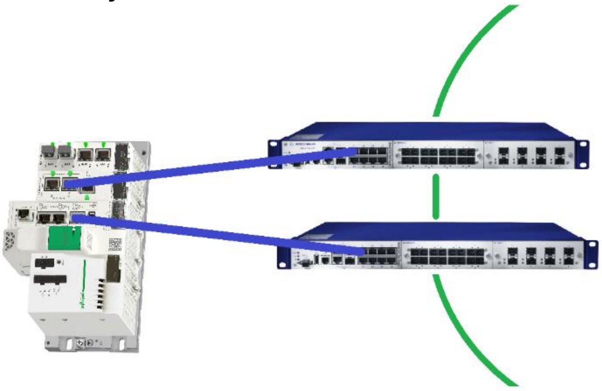
Port	Stp active	Dual RSTP active
1.1		
1.2		
1.3	✓	
1.4	✓	
1.5	✓	
1.6	✓	
1.7	✓	✓
1.8	✓	✓



The PLC M580 in the configuration used in this project doesn't have a double port with the same IP address; for this reason, a different redundancy method has been used.

Two different ports with two different IP addresses are connected to two different switches belonging to the MRP ring.

The Modbus instruction of reading and writing send to the other devices are configured, as default, on the first port; if this communication is interrupt, it automatically repeats the same procedure on the other port.



Having the PC the same problem, the same strategy has been adopted.

As result of being more subject to problems like uncontrolled multicast messages (broadcast storm), which can cause a block of the net; PC has been isolated from the rest of the devices.

The "insulator" takes the name of router.

Remembering that messages cannot go outside a net; for example, if the mask is 255.255.255.0 all devices with a different net id (first three numbers e.g. 192.168.1) will not be received directly.

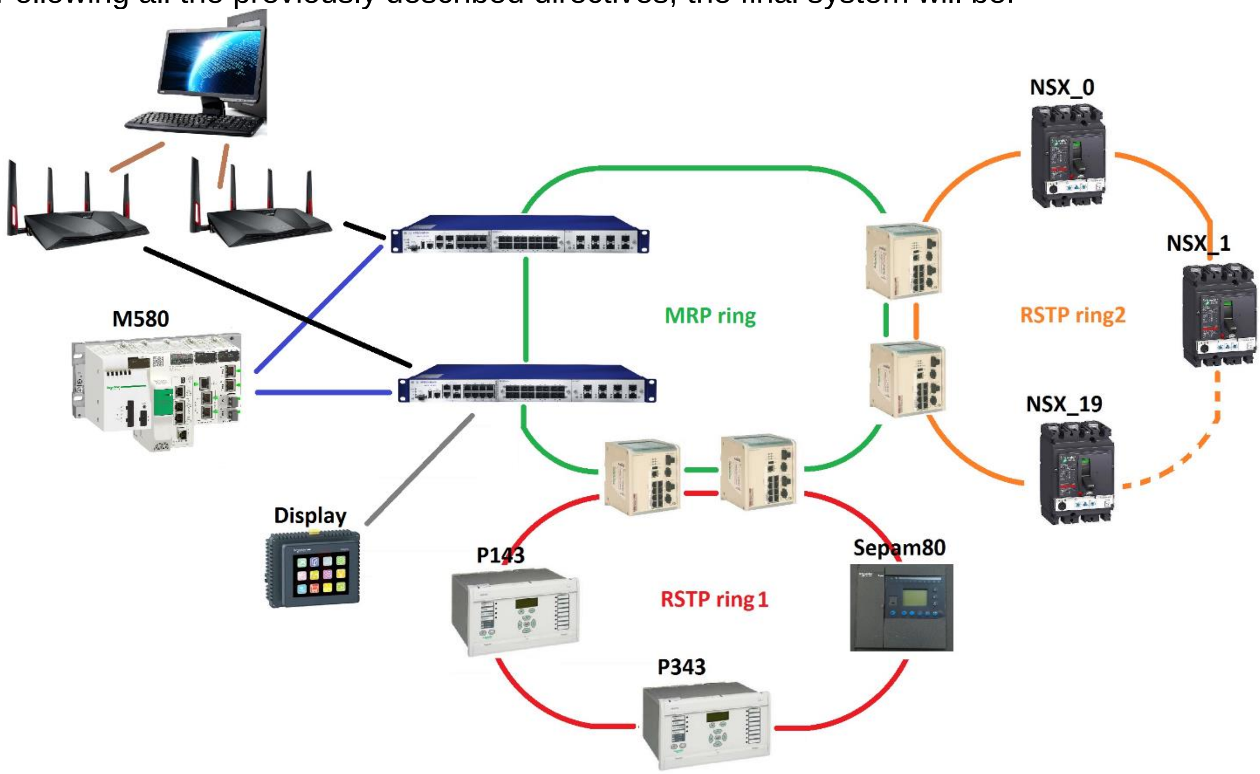
The role of the router is take the massages with a different net id from the net in which is circulating and send to the right sub net.

Since broadcast message has no IP address, it will be filtrated by the router.

So, putting between the system (net id 192.168.1) and the two PC ports (net ids 192.168.2 and 192.168.3) two routers, no broadcast message will be sent from the PC.

Only the display HMI will not be put in a redundancy mode, this is because it is equipped with only one port, making impossible any redundant configuration.

Following all the previously described directives, the final system will be:

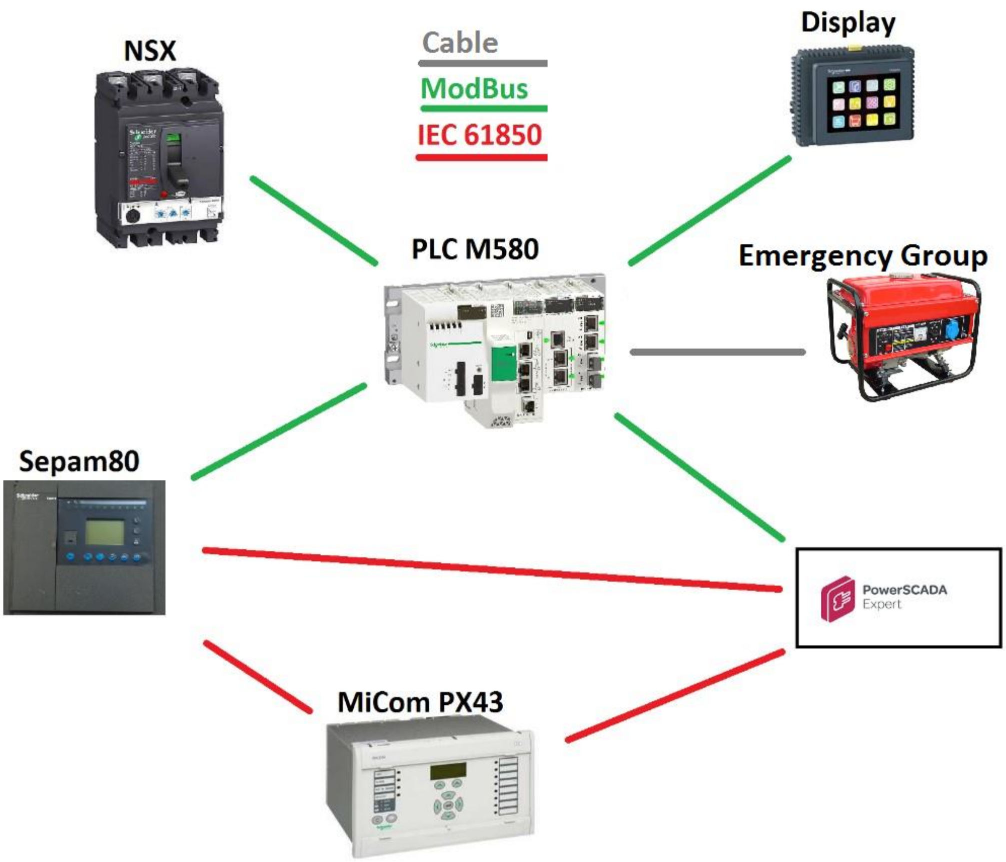


2.2.6) Communication

To follow what has been described previously, the component of our system has to communicate each other information.
First thing to be considered is which protocols can use these devices:

- PLC M580: Modbus
- NSX: Modbus
- Display HMI: Modbus
- Power Scada Expert: Modbus, IEC61850
- Sepam80: Modbus, IEC61850
- MiCom P143: IEC61850
- MiCom P343: IEC61850

Using two different protocols, devices, able to understand only one of them, to communicate with a different language speaking component have to be "bounced" on a third element.
For example, the information related to the states of the two line-switches memorized in MiCom relays are needed to the PLC in order to compute the state of islanding mode of the emergency group.
PLC will read this information through Sepam80 which speaks both the languages.



PLC M580 ↔ NSX

The communication between PLC and relays NSX is made using Modbus protocol.
In particular the are two different kinds of messages: reading of power and state and writing of commands open and close.
As concern the readings, since the maximum number of registers which can be read in only one instruction is 125 and there are 476 spaces between the state of the switch (number 562) and the power absorbed by the load (number 1038), two reading instruction will be send to each of the twenty NSXs.

```
esegui_read_var_1[i]:=Puls100ms and not tabella_gest_readvar_1[i][0].0;
if esegui_read_var_1[i] then
        READ_VAR (ADR := ADDM (IN := 'Ethernet_NOR_1{Ethernet_NSX[i]}1'),
                OBJ := '%MW',
                NUM := 560,
                NB := 5,
                GEST := tabella_gest_readvar_1[i],
                RECP => buffer_ricez_1[i]);
NSX[i].InterrChiuso:=buffer_ricez_1[i][2].0;
end_if;
esegui_read_var_2[i]:=Puls100ms and not tabella_gest_readvar_2[i][0].0;
if esegui_read_var_2[i] then
        READ_VAR (ADR := ADDM (IN := 'Ethernet_NOR_1{Ethernet_NSX[i]}1'),
                OBJ := '%MW',
                NUM := 999,
                NB := 50,
                GEST := tabella_gest_readvar_2[i],
                RECP => buffer_ricez_2[i]);
NSX[i].P:=int to real(buffer ricez 2[i][37]);
end_if;
```

Every 100 milliseconds two READ_VARs will be launched, in this way every 2 seconds all switches position and load power absorption is refreshed.
This time could be reduced till reach the minimum time removing the condition "Puls100ms" and let only "not tabella_gest_readvar_" in charge to control if the same reading instruction is already running.
The problem is that in this way 40 READ_VARs could be working together that is higher than the maximum numbers expected by the PLC.
2 seconds isn't a long time for this kind of application in which loads behaviour is (except for the inrush current) quite stationary.
All data read by PLC will be stored in an array called "buffer_ricez_".

NSX expects as a command of open/close not a single bit set to 1 but a series of registers written with different values.
In particular, to consider valid a command, the values reported in the table on the right have to be sent at the same time.
These numbers are stored in two arrays called "buffer_w_apri" for the open command and "buffer_w_chiudi" for the close one.

Register	Values	
	Open	Close
7999	904	905
8000	10	
8001	4353	
8002	1	
8003	13107	
8004	13107	

Open command

```
WRITE_VAR (ADR := ADDM (IN := 'Ethernet_NOR_1{Etherne_NSX[i]}1'),
           OBJ := '%MW',
           NUM := 7999,
           NB := 6,
           EMIS := buffer_w_apri,
           GEST := tabella_gest_write[i]);
```

Close command

```
WRITE_VAR (ADR := ADDM (IN := 'Ethernet_NOR_1{Etherne_NSX[i]}1'),
           OBJ := '%MW',
           NUM := 7999,
           NB := 6,
           EMIS := buffer_w_chiudi,
           GEST := tabella_gest_write[i]);
```

PLC M580 ↔ Sepam80

The communication between PLCM580 and Sepam80 has to be faster than the previous described one; in fact, this interaction is, for example, used to activate the LS (managed by the PLC) as soon as ATS has intervened (programmed in Sepam80).
For this reason, the Modbus requests aren't send as previously with READ_VAR, WRITE_VAR, but with the IO/scanning mode.

ID unità	Timeout stato (ms)	Frequenza di ripetizione (ms)	Indirizzo RD	Lunghezza RD	Ultimo valore	Indirizzo WR	Lunghezza WR
1	1500	50	3217	1	Mantieni valore	0	0
1	1500	50	0	0	Mantieni valore	3208	1
1	1500	50	3219	1	Mantieni valore	0	0
1	1500	50	0	0	Mantieni valore	3200	1
1	1500	50	0	0	Mantieni valore	3209	1

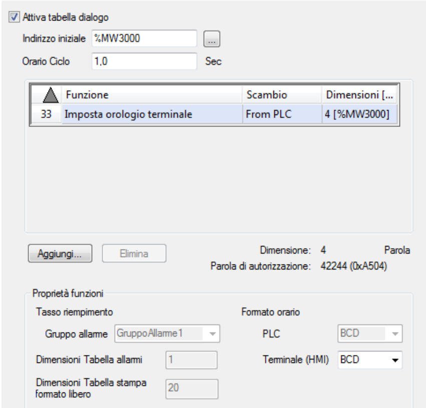
With these method M580 will read his needed values on Sepam80 bits called "remote indications" and will write other bits called "remote controls".
In Sepam80 logic, the association between remote controls and indications with the corresponding variable is done.
For example, to read the state (automatic or manual) of the ATS, Sepam80 will report the value contained in his internal memory to the remote indications TS064

Which is allocated in the register 3219, bit 15.
With the third row of the previous table M580 will read this register and will memorize it in his internal variable "sepam80.inputs.free1"; the first 8 bits in the register sepam80.inputs.free1[0] and the rest in sepam80.inputs.free1[1].
Then the value of the single bit will be copied in the Boolean variable "ats_on".

```
ats_on:=sepam80.inputs.free1[1].7;
```

PLC M580 ↔ Display

The Display need to read and write register to both M580 and Sepam80.
It isn't capable to communicate simultaneously with two different devices, for this reason all variables needed to the HMI are took by the PLC from Sepam80.
In this way only one communication is initialized.
Looking only at system variables both devices could be chosen.
The key point that makes one choose better than the other one is that this display can read the date from a device and only PLC can provide him this information.
Using Vijeo Design (the display software configurator), 4 PLC registers can be chosen to redirect them at the only aim to memorize the current data in BCD format. In this project the register 3000÷3003 have been selected.




```
RRTC_DT (OUT => DataOraPLC);
ArrayDataOraPLC := DT_TO_ARINT (IN := DataOraPLC);
%MW3000:=ArrayDataOraPLC[1];
%MW3001:=ArrayDataOraPLC[2];
%MW3002:=ArrayDataOraPLC[3];
%MW3003:=ArrayDataOraPLC[4];
```

PLC M580 ↔ Emergency Group

The connection between relay switches of the emergency group and the motors themselves, considering that only two digital command will send to each of them; it is realized with a direct cabled connection.
The PLC outputs “aumenta_set_rpm” and “diminuisci_set_rpm” as for the commands open/close Q3 or Q4 are replaced on physical outputs of M580.
Same thing is for the PLC physical inputs in which the state of the switches and the motors will be write.

PLC M580 ↔ Power Scada Expert

The Modbus communication is easily managed by this two device, reporting the same PLC register in which the information required by PSE are stored, the sharing of this information is done.
For example, the instantaneous speed of the first emergency group motor is memorized in a variable called “giri” at the address MW200:

PLC configuration

Nome	Tipo	Indirizzo
giri	REAL	%MW200

PSE configuration

Tag Name	I/O Device	Data Type	Addres
giri	M580	REAL	%MW200

Sepam80 / MiCom PX43 ↔ Power Scada Expert

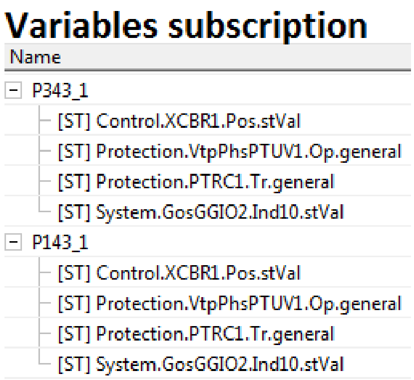
PSE can communicate in IEC61850, so it can import all variables available in relays’ CID to visualize their status, their measures and their time-stamped alarms on its graphical page.
It can use simultaneously different protocol for different devices so it could speak to Sepam80 in Modbus too.
The easier way to find a variable is an important thing that pushes the decision to communicate with Sepam80 in IEC61850 too.

In fact, just knowing the way in which PSE “follows” the standardized paths, without searching any register addresses or any documentation, a variable can be imported in this program.
For example, the current of phase A will be find using this string:

LD0/MMXU1MXA$phsA$cValmagf

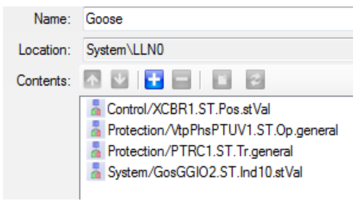
Sepam80 ↔ MiCom PX43

The unique, compatible with this system, protocol supported by the two relays Micom (P143 and P343) is IEC 61850.
The interaction between these devices is guaranteed by GOOSE messages.
The main information exchanged between Sepam80 and the two relays are those that regard the ATS process.
Sepam80, which runs all the ATS actions, will receive as input goose message and to make it available to be used in his logic it chains them with his internal memory called ME_GSE_X, where “X” stands for the hundreds of the identification number of the goose input; for example, in our project all inputs are connected to ME_GSE_5 and so all variable used are G5XX.



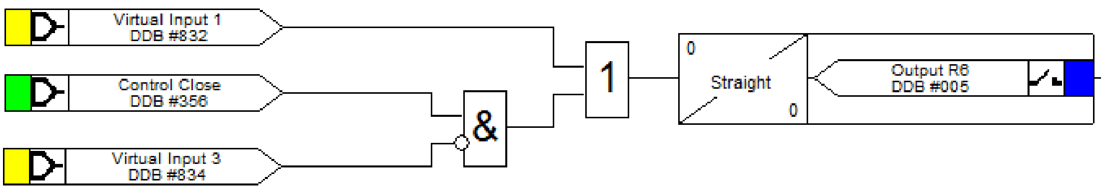
Variables assignment

IED Name	DA	Descr...	G501	G502	G503	G504	G505	G506	G507	G508
P143_1	QR	Quality								
P143_1	QD	Qualit...								
P143_1	[ST] Control.XCBR1.Pos.stVal	Retur...	X							
P143_1	[ST] Protection.VtpPhsPTUV1.Op.general	DDB ...			X					
P143_1	[ST] Protection.PTRC1.Tr.general	DDB ...					X			
P143_1	[ST] System.GosGGIO2.Ind10.stVal	DDB ...							X	
P343_1	QR	Quality								
P343_1	QD	Qualit...								
P343_1	[ST] Control.XCBR1.Pos.stVal	Retur...		X						
P343_1	[ST] Protection.VtpPhsPTUV1.Op.general	DDB ...				X				
P343_1	[ST] Protection.PTRC1.Tr.general	DDB ...						X		
P343_1	[ST] System.GosGGIO2.Ind10.stVal	DDB ...								X



Not all variables can be subscribed by Sepam80, just ones are programmed to be used as goose message.
To do this, a goose dataset has to be created in the sending relays.

To send to relays the open/close/block_close commands the opposite procedure will be done: in Sepam80 will be created a goose dataset and Micom relays will subscribe them and then they assign these to their internal variables (called “virtual input”).
To use this information internal logic of the relay has to be changed.
For example, to connect the closure inhibition (virtual input 3), the manual close (control close) and the external close command (virtual input 1) to the physical relay close command (output R6), this scheme has been used:



Final considerations

The main aim of this project was ensuring a total continuity of service in a medium industrial plant.
The main issue examined is the presence of lack of voltage in one of the two feeding lines, in this case the standard procedure adopted is:

- Open the breaker of the line with the lack of voltage present.
- Instant detach of the “less important” loads to keep the power under the transformer rated one.
- Close the bus-bar breaker
- Send command “start” to the emergency group 1
- Close the emergency group breaker when the synchronism is established
- Reattach of the automatic opened loads
- If necessary, close the second emergency group when it is allowed

In parallel of this main feature there is the behaviour of the emergency generators in case of islanding mode.
In this situation, found that the only aim of the GE is keep constant the value of frequency, the detach of loads is always present, in order to avoid instantaneous change of high power absorption.

The reset of any “emergency” situation has been done automatically but after a command. In fact, only the procedures that allow to keep constant (or reduce) the alimentation of the loads can be considered safe; an automatic restore of the pre-fault situation could be inspected by people and then dangerous.
Moreover, to ensure that every action done in this system cannot bring it in a dangerous situation both for plant that for people, every manual command can be send only if there is the safe condition to stat that process.

An interruption of communication among the devices involved in these applications is a critical point (for example if the bus-bar relay is not connected with the other ones no ATS can starts).
In order to ensure this, a complex physical connection has been done; in which also after a fault of one device, the communication is ensured.

Not the entire project has been reported in this treat, in fact, the purpose wasn’t to create a technical report, but to show in a more practical way the main different communication protocols and almost all the automatisms usually implemented in a low voltage system.