

# POLITECNICO DI TORINO

MASTER's Degree in ELECTRONIC ENGINEERING



MASTER's Degree Thesis

## Surge $1.2/50\mu s$ Test Automation for DIN-Rail Residual Current CB

Supervisors

Prof. MASSIMO RUO ROCH

Ing. JACOPO FEDERICO SASSO

Candidate

**ROSALIA CICERO**

**JULY 2022/2023**



## Abstract

The thesis project entitled "Surge 1.2/50  $\mu$ s test automation for DIN-Rail Residual Current CB", carried out at the ABB S.p.A company based in Vittuone (MI), starting from the study of residual current circuit breakers was the protagonist of the development of the automation of a test carried out daily within the laboratory that deals with the electromagnetic compatibility verification of products. The task of the surge test is to verify that the devices are sufficiently immune to such disturbances by meeting the functional and safety requirements of the relevant product standards. Residual current circuit breakers during the test may trip as a result of applied over-voltages. The operator must then reset the part and mark the test conditions under which the trip occurred. The purpose of this thesis was to eliminate the presence of an operator during surge testing on differential circuit breakers din-rail, replacing his action with an ABB motor drive and appropriate control logic. In this regard, it is possible to use graphical programming environment to work on such as Labview. As mentioned earlier for the automation of the test, it is necessary to replace the current operator action with a motor control, for which the choice could be the new MOD device under development in the company, which is coupled with the COMM WIFI communication module, also under development. To monitor the progress of the above test, one must include measurement instrumentation such as an oscilloscope, which can analyze voltages and currents related to each surge pulse. After addressing this aspect, since each instrument used has to be networked, it is necessary to introduce the serial communication of the machine used for carrying out the Surge test. COM ports are therefore employed, managed in such a way that they can be used simultaneously by more than one software, since it is necessary that both Labview and the Surge machine's own software can have read and write access to them. To remotely drive the MOD device through the COMM Wifi communication module the Modbus TCP protocol must be validated and implemented. Going through all these steps, it is necessary to create a report file in which are listed the test parameters for each case history, the voltage and current values for each surge pulse and any voltage and current waveforms in cases where a trip or over-current occurs. It is important for the program to consider two possible types of tests: the normal sequence and the random angle sequence; these in fact employ different test structures that impact the data handling by the Labview program created. Finally, it is necessary to create a user interface, through which the latter can manage and monitor the status of the test through visual and audible signals, entering test parameters of numeric or string type, and being able to intervene by starting or stopping the test through appropriate buttons.



*“La più bella e profonda emozione che possiamo provare è il senso del mistero; sta qui il seme di ogni arte, di ogni vera scienza.”*  
*Albert Einstein*

# Table of Contents

<b>List of Tables</b>	v
<b>List of Figures</b>	VI
<b>Acronyms</b>	x
<b>1 INTRODUCTION</b>	<b>1</b>
1.1 Voltage Surge Test Immunity . . . . .	1
1.2 The circuit breaker . . . . .	7
1.2.1 Reference standards for differential devices . . . . .	9
1.2.2 Working Principle . . . . .	10
1.2.3 Operation Method . . . . .	11
1.3 MOD & COMM: Motor operating device and Communication module	16
1.4 Machine used to perform the surge test: AMETEK CTS . . . . .	19
1.4.1 Software Iec.control . . . . .	21
<b>2 COMMUNICATIONS IMPLEMENTATION</b>	<b>23</b>
2.0.1 Ametek cts machine management . . . . .	23
2.0.2 MOD-COMM communication management . . . . .	34
2.0.3 Oscilloscope communication . . . . .	38
2.0.4 Normal sequence test procedure . . . . .	41
2.0.5 Random angle test procedure . . . . .	48
2.0.6 Report generation . . . . .	50
<b>3 MAIN PROGRAM DEVELOPMENT</b>	<b>60</b>
3.0.1 Development . . . . .	60
3.0.2 User Interface . . . . .	66
<b>4 CONCLUSION AND POSSIBLE IMPROVEMENTS</b>	<b>68</b>
<b>A Derivation of components of surge generators</b>	<b>72</b>

<b>B MODBUS TPC PROTOCOL</b>	76
B.0.1 Function Code Categories . . . . .	78
B.0.2 Modbus Components Architecture Model . . . . .	87
<b>C Remote Commands for Surge</b>	89
<b>Bibliography</b>	92

# List of Tables

1.1 TEST LEVELS . . . . .	5
---------------------------	---



# List of Figures

1.1	Scheme of CWG[1]	2
1.2	CWG 1.2/50 $\mu$ s Voltage Surge waveform[1]	2
1.3	CWG 8/20 $\mu$ s Current Waveform[1]	2
1.4	Single phase line-to-line coupling/decoupling network[2]	3
1.5	Single phase line-to-earth coupling/decoupling network[2]	3
1.6	Three phase line-to-earth coupling/decoupling network[2]	4
1.7	Three phase line-to-line coupling/decoupling network[2]	4
1.8	Example of test setup for surge immunity test [2]	6
1.9	[3]	7
1.10	Danger curves current time[4]	8
1.11	Wiring diagram of a single phase residual current circuit breaker [5]	10
1.12	Classification type G,S,M[5]	13
1.13	Classification of residual current circuit breakers [6]	14
1.14	Connection for Mod [7]	16
1.15	Led Mod operations [7]	17
1.16	Connection for Mod [7]	17
1.17	Representation switch-mod-com [7]	18
1.18	MOD-COMM connection [8]	18
1.19	AMETEK CTS COUPLING NX7 front side[9]	19
1.20	AMETEK CTS COUPLING NX7 back side[9]	19
1.21	Delta supply neutral point [10]	20
1.22	Device setup menu.	21
1.23	Pulse window.	22
2.1	Labview VI VSPD.	24
2.2	Labview block diagram of VSPD VI.	26
2.3	Labview subVI AMETEK COMMUNICATION.	26
2.4	Labview subVI AMETEK COMMUNICATION BLOCK DIAGRAM.	27
2.5	Visa configure serial port VI.	27
2.6	Select.	28
2.7	VISA READ function.	28

2.8	VISA READ function and the Property node to pass it the bytes at port. . . . .	28
2.9	Read ucs machine status and notification. . . . .	29
2.10	Search of the value of the angles in the case of random angles. . . .	30
2.11	Search of the value of the voltage. . . . .	31
2.12	Search of the value of the coupling. . . . .	31
2.13	Search of the value of the pulse duration. . . . .	32
2.14	Search of the value of the polarity. . . . .	32
2.15	Search of the value of the pulse. . . . .	33
2.16	Write To AMETEK NX7 subVI . . . . .	33
2.17	Block diagram Write To AMETEK CTS subVI. . . . .	34
2.18	Visa Write function. . . . .	34
2.19	Modbus communication subVI. . . . .	35
2.20	Modbus communication subVI block diagram. . . . .	35
2.21	TCP master creation and communication check. . . . .	36
2.22	Register reading and writing block. . . . .	37
2.23	Reading Holding register function . . . . .	38
2.24	Write Multiple Holding registers function . . . . .	38
2.25	Oscilloscope communication block . . . . .	39
2.26	Initialize function . . . . .	39
2.27	Capture waveforms subVI. . . . .	40
2.28	Block diagram of the capture waveform.vi in case of normal sequence mode. . . . .	41
2.29	Trigger acquisition. . . . .	42
2.30	Read Waveform (single).vi. . . . .	42
2.31	Close.vi. . . . .	43
2.32	For loop for the five pulse on the differential switch. . . . .	43
2.33	Minimum and maximum current and voltage evaluation. . . . .	44
2.34	Fetch waveform function. . . . .	45
2.35	Waveform Min and MAX function. . . . .	45
2.36	Positive result. . . . .	45
2.37	Negative result. . . . .	46
2.38	Waveform capture. . . . .	46
2.39	First part of waveform capture. . . . .	47
2.40	Second part of waveform capture. . . . .	48
2.41	Block diagram of the capture waveform.vi in case of random angles mode. . . . .	49
2.42	Report generation subVI. . . . .	50
2.43	Block diagram of the report generation vi. . . . .	51
2.44	Elements extraction to be included in the table. . . . .	52
2.45	First segment flat sequence structure. . . . .	53

2.46	Create Report.vi. . . . .	53
2.47	Set Report Header Text.vi. . . . .	54
2.48	Append Report Text.vi. . . . .	54
2.49	Second segment flat sequence structure. . . . .	55
2.50	Word easy table vi. . . . .	55
2.51	Append image to report vi. . . . .	56
2.52	Save report to File vi. . . . .	57
2.53	Dispose report vi. . . . .	57
2.54	Second segment in case of random sequence. . . . .	58
3.1	Main Program block diagram. . . . .	60
3.2	First block of main program block diagram. . . . .	61
3.3	Second block of main program block diagram. . . . .	62
3.4	Search and replace string function. . . . .	62
3.5	Main case structure of the block diagram, true condition. . . . .	63
3.6	Sub-case structure to command the machine, true condition. . . . .	63
3.7	Sub-case structure to command the machine, false condition. . . . .	64
3.8	Sub-case structure to user notification, true condition. . . . .	64
3.9	Main case structure of the block diagram, false condition. . . . .	65
3.10	Stop conditions. . . . .	66
3.11	Reading of global variables to monitoring the test. . . . .	66
3.12	User interface. . . . .	67
A.1	1.2/50 $\mu$ s combination wave surge generator . . . . .	72
B.1	Client/server structure [12] . . . . .	76
B.2	MODBUS ADU structure[12] . . . . .	77
B.3	MODBUS TCP/IP ADU structure [6] . . . . .	77
B.4	Modbus data model with separated block[13] . . . . .	78
B.5	Modbus data model with only 1 block[11] . . . . .	78
B.6	Function code definition[11] . . . . .	79
B.7	Modbus Components Architecture Model[8] . . . . .	87



# Acronyms

**ADU**

Application Data Unit

**CB**

Circuit Braker

**CWG**

Combination Wave Generator

**CDN**

Coupling Decoupling Network

**EUT**

Equipment under test

**HV**

High Voltage

**LV**

Low Voltage

**MBAP**

Modbus Application Protocol

**MEI**

Modbus encapsulated interface

**PDU**

Protocol Data Unit

**RCBOs**

Residual Current Circuit Breaker with Over-current Protection

**RCCBs**

Residual Current Circuit Breaker

**RCD**

Residual Current Protective Device

**SSID**

Service Set Identifier

# Chapter 1

## INTRODUCTION

The thesis work carried out in collaboration with ABB S.p.A was focused on the automation of the Surge 1.2/50 $\mu$  test on DIN-Rail switches that is carried out within the company's EMC laboratory, in which tests for electromagnetic compatibility are precisely carried out. This chapter will give some information regarding the context in which the thesis project was carried out, giving some information with respect to the test itself and the devices and machinery involved .

### 1.1 Voltage Surge Test Immunity

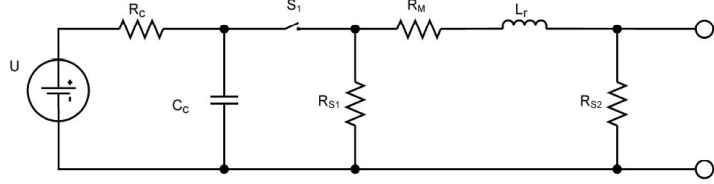
Transient over voltages on an AC power system are caused by events such as: equipment failures, lightning discharges but also by load or capacitor bank switching, and, having devices that are resistant to voltage spikes of this type can be favorable from an application point of view. Usually these transient surges are of short duration, on the order of micro seconds to a few milliseconds, with generically impulsive wave-forms with increasing wavefront is usually between 0.5/10  $\mu$ s.

These tests are regulated by standards, which for Surge are :

- US standard: ANSI/IEEE C62.41-1991, IEEE recommended practice on surge voltages in low-voltage ac power circuits;
- International standard: IEC 61000-4-5, 2017-08 Electromagnetic Compatibility, Testing and measurement techniques–Surge immunity test.

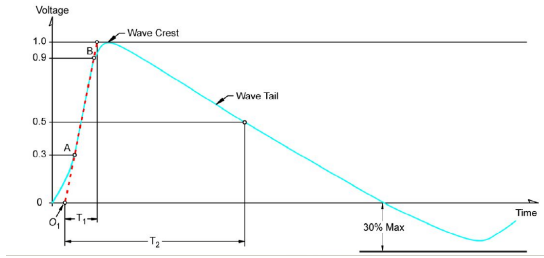
The test equipment has a surge generator and a coupling/decoupling network; combining the wave generators results in a specified open-circuit voltage waveform and a current waveform specified short-circuit. Coupling and decoupling networks are used, therefore, to couple an over voltage generator to the equipment under test and makes sure that dangerous voltages are not fed back into the ac supply circuit. As can be seen in the circuit shown in Figure 1.1 [1], it is present U voltage

generator, which charges capacitor  $C_c$  through resistor  $R_c$ , once charged, switch  $S_1$  causes power to be supplied to the network consisting of resistors  $R_{s1}$ ,  $R_{s2}$ ,  $R_m$  and to the inductance  $L_r$ , which has the task of shaping the wave.

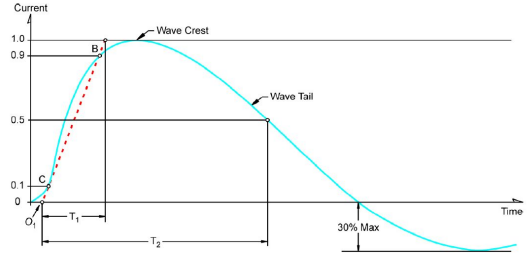


**Figure 1.1:** Scheme of CWG[1]

The components are chosen to meet the open-circuit voltage and short-circuit current values. Specifically, the CWG 1.2/50 us voltage surge waveform and the CWG 8/20 us Current waveform are of the type:



**Figure 1.2:** CWG 1.2/50 $\mu$ s Voltage Surge waveform[1]



**Figure 1.3:** CWG 8/20 $\mu$ s Current Waveform[1]

in which in Figure 1.2:

$$T_1 = 1.67T = 1.2\mu s \pm 30\% \quad (1.1)$$

$$T_2 = 50\mu s \pm 20\% \quad (1.2)$$

$$T = T_b - T_a \quad (1.3)$$

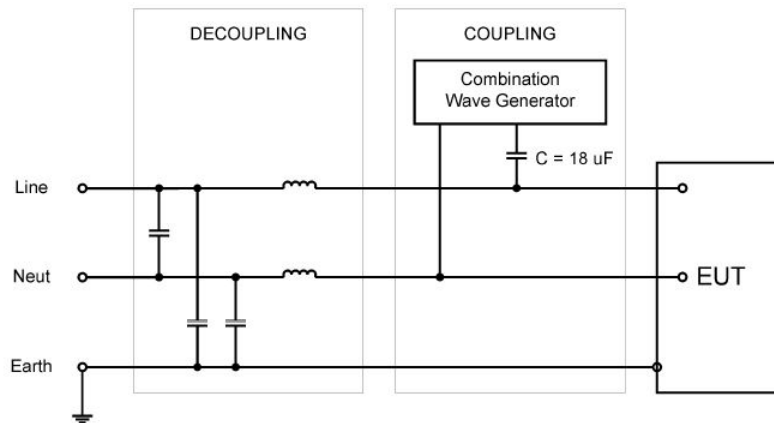
in which in Figure 1.3:

$$T_1 = 1.65T = 8\mu s \pm 30\% \quad (1.4)$$

$$T_2 = 20\mu s \pm 20\% \quad (1.5)$$

$$T = T_b - T_c \quad (1.6)$$

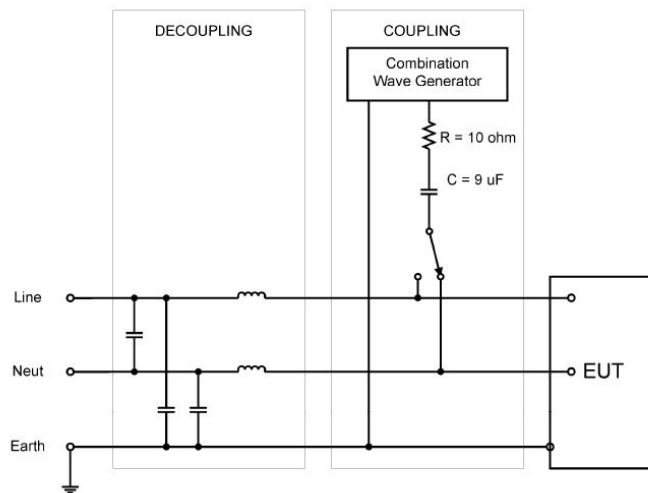




**Figure 1.4:** Single phase line-to-line coupling/decoupling network[2]

As mentioned above, the coupling and decoupling circuit is used to couple the CWG voltage surge with the product under test and simultaneously decouple it from the system.

The inductance, generally of  $1.5mH$ , must be large enough to have the minimum effect on the output of the CWG, but at the same time small enough to allow the product under test to perform normal operation. An example of a coupling/decoupling network has been shown in Figure 1.4, but other possible configurations are shown below.



**Figure 1.5:** Single phase line-to-earth coupling/decoupling network[2]

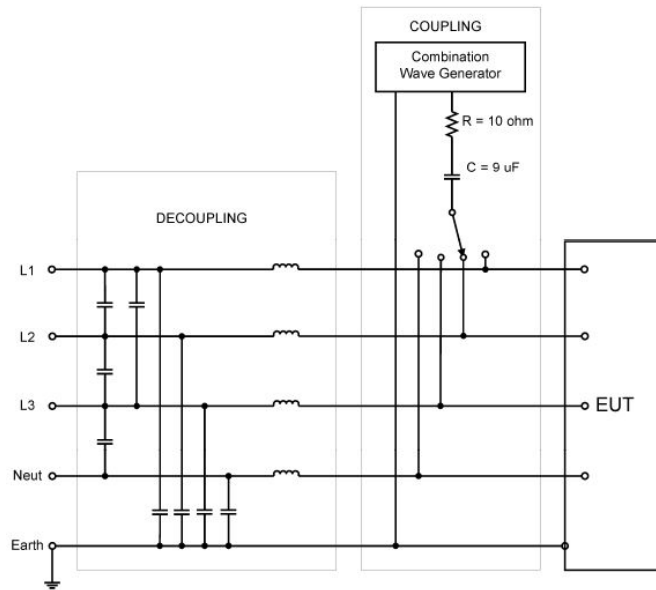


Figure 1.6: Three phase line-to-earth coupling/decoupling network[2]

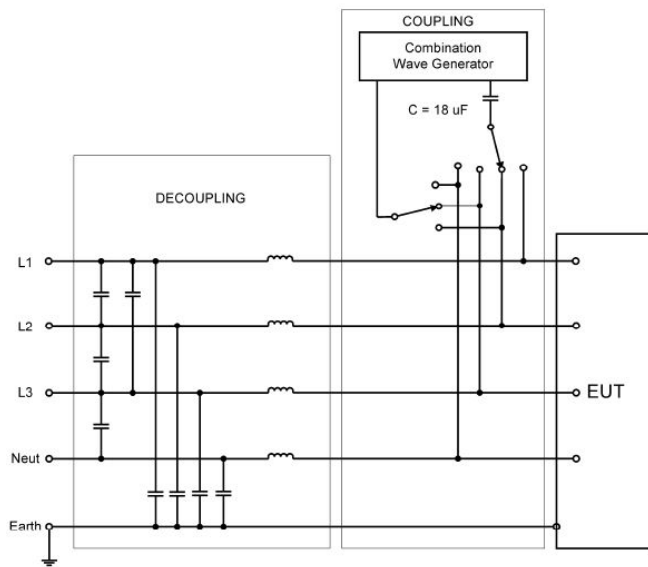


Figure 1.7: Three phase line-to-line coupling/decoupling network[2]

Analysis of the test procedure:

1. Verification of the test instrumentation:

- The combination wave generator;
- The CDN;
- The interconnection cables of the test equipment;
- Surge impulse present at the output terminal of the CDN should be checked without an EUT connected to the system;

2. The establishment of the laboratory reference conditions: tests shall not be performed if the relative humidity is so high as to cause condensation on the EUT or the test equipment and if the electromagnetic conditions of the laboratory do not guarantee the correct operation of the EUT;

3. The confirmation of the correct operation of the EUT;

4. The execution of the test: when testing 3-phase systems, the synchronization of phase angles shall be taken from the same line under test and in particular, the phase taken into account is always the R phase, while the angle is calculated by knowing the phase shift; instead when testing line-to-ground, the lines are tested individually in sequence. As for the angle, the test conducted for line to line or line to ground is the same.

The test plan specify the test setup:

- test level: Table 1.1

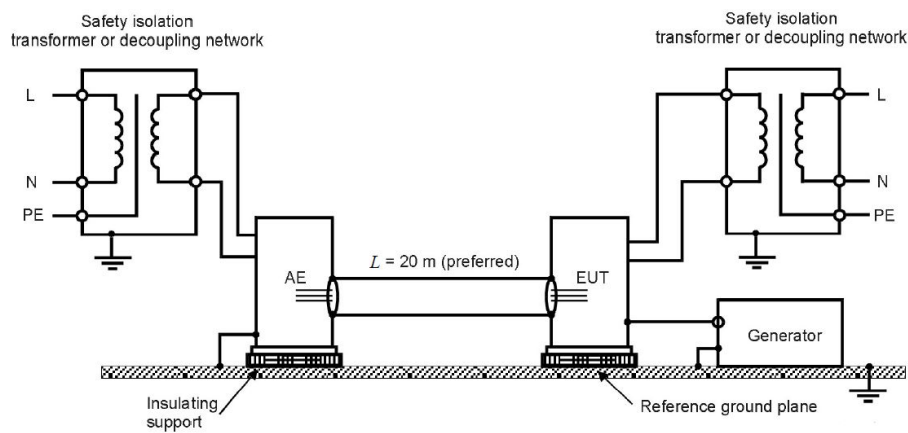
	Open-circuit test voltage [KV]:	
Level	Line-to-line	line-to-ground
1	—	0.5
2	0.5	1
3	1	2
4	2	4
$x^a$	special	special

**Table 1.1: TEST LEVELS**

- number of surge impulses unless otherwise specified by the relevant standard:
  - (a) for d.c. power ports and interconnection lines five positive and five negative surge impulses;

- (b) for a.c. power ports five positive and five negative impulses each at  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$  and at  $270^\circ$ ;
- time between successive impulses: 1 min or less;
  - representative operating conditions of the EUT;
  - EUT ports to be tested.
5. The evaluation of the test results: the test results shall be classified in terms of the loss of function or degradation of performance of the equipment under test.

An example of a test setup is shown below in the Figure 1.8.



**Figure 1.8:** Example of test setup for surge immunity test [2]

The IEC 61000-4-5 standard provides circuit schematics of surge generators, but gives no information with respect to components; how to derive them is shown in the appendix A.

## 1.2 The circuit breaker

This thesis project focused on the automation of Surge test applied to DIN-rail circuit breakers. The earth residual current RCCB is an amperometric protective device that trips, interrupting the power supply, when the circuits and electrical appliances supplied by it, present an earth leakage current. In addition to this, some breakers include also an overcurrent protection (RCBO) that helps in protecting the load circuit from overload and short circuits.

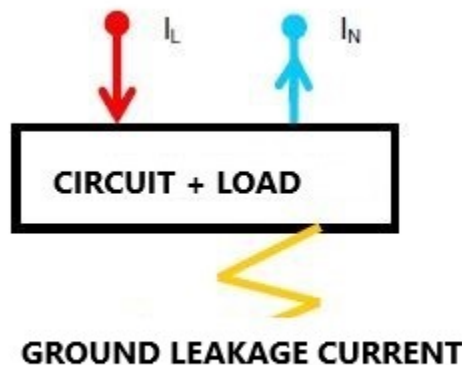
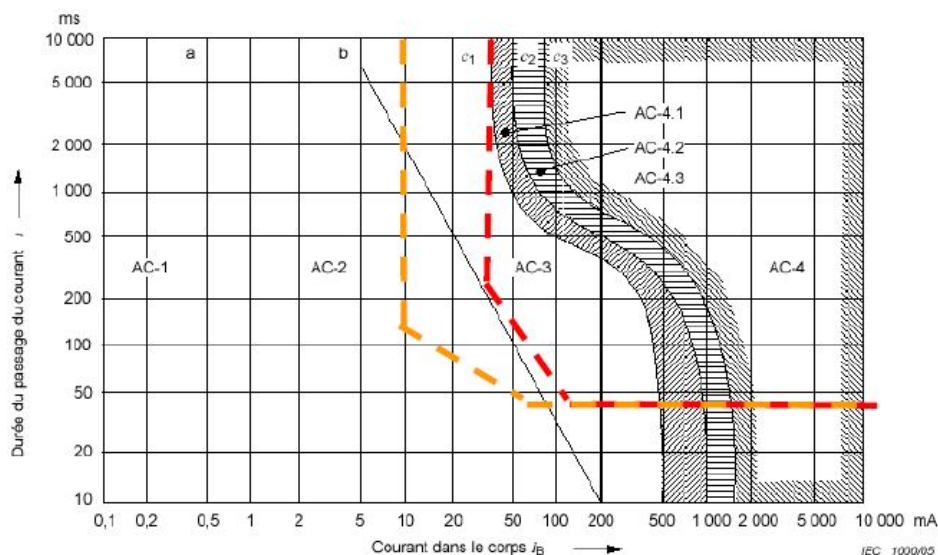


Figure 1.9: [3]

The action of the switch consists of: considering a single-phase line, from the representation in Figure 1.9, it can be seen that, by Kirchoff's first principle, the currents in the phase conductor and the neutral conductor are  $I_L = I_N$ . Should there be a current to earth, given by any possible cause, a leakage current would be created, given precisely by the difference between the two currents:  $I_d = I_L - I_N$ . What should be cautioned, however, is that such a current can flow to earth either through the protective conductor of the PE grounding system or through any other pathway that constitutes an earth, such as could be the body of the person. If a three-phase line is considered, the differential current is expressed by the vector sum of the currents present in the active conductors of the line. Thus, for both single-phase and three-phase lines, the action of the differential switch is to perform the vector sum of the currents present on the active conductors, and for this difference to be zero, it allows power to be supplied by the user, which is immediately interrupted if the predetermined threshold value is exceeded. The main purpose of the residual current circuit breaker is to protect people from indirect (contact with a non-insulated metal mass which, due to a fault, is at a dangerous voltage) or direct (accidental contact with a conductor normally having a dangerous

voltage) contact with live parts, and in cases where it meets certain requirements, it can also be used as a disconnecter, i.e., a device capable of completely de-energizing an electrical installation in its entirety or a section of it. Protection against indirect contacts is provided by adequate grounding of masses, appropriately associated with the earth leakage circuit breaker. For direct contacts, on the other hand, grounding is completely unnecessary, in fact the purpose of the grounding system is to drain to the ground, through the protective conductor (PE), any fault currents that may be passing through the metal masses while ideally maintaining their potential zero so that, in the event of contact with them, any current that may be passing through the person is zero or at least very small. The earthing system, on the other hand, can do nothing for fault currents that should discharge to earth through a person, flowing directly from the phase conductor to the body, without passing through interposed metal masses. Additional protection therefore is achieved by using high-sensitivity  $30mA$  RCCB; this is also useful for indirect contacts in case the ground connection is broken. In Figure 1.10 it is possible to



**Figure 1.10:** Danger curves current time[4]

observe the current time danger curves (curves c and b) taking into consideration electrocutions occurring through small contact surfaces with hand-to-foot path. On the x-axis is the current  $I_b$  passing through the human body and on the ordinates instead the duration of the passage. The red dashed line indicates the inverse tripping time limit characteristic, which is required by the standards for a  $30mA$  residual current circuit breaker.

We can also identify 4 different zones:

1. zone AC-1: no current perception;
2. zone AC-2: perception of current without any hazardous effects;
3. zone AC-3: generally reversible pathological effects;
4. zone AC-4: even severe pathological effects with the possibility of irreversible damage, such as ventricular fibrillation that is not spontaneously reversible, cardiac arrest cardiac arrest etc.

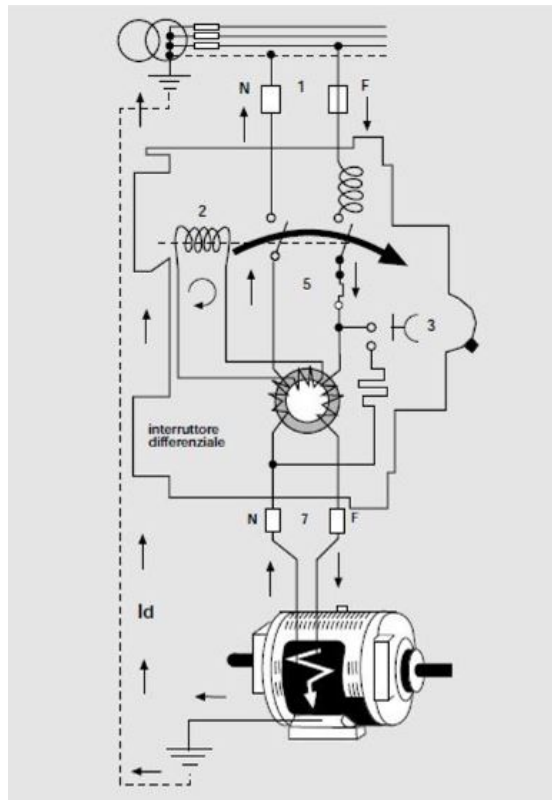
Thus, from this graph it is possible to deduce why the  $30mA$  residual current circuit breaker is considered safe to provide the additional protection, which in the case of direct contact does not prevent the possible electrocution, but limits it so that there is no irreversible damage. Residual current breakers with different sensitivity could be used, such as  $10mA$  or  $6mA$  as can be seen in the above image, where the yellow curve represents the tripping limit for the  $10mA$  differential, which would go to avoid the risks of the AC-3 zone.

However, it is clear that the need for continuity of service requires that such sensitive differentials shall be used in a limited way, so a choice is made due to safety and service requirements.

It should be pointed out, however, that for currents greater than  $500mA$  it is not possible to establish a short electrocution duration such that it can be considered safe, so in that case they may not be protected by the residual current circuit breaker. The action of the residual current circuit breaker fails in two cases, as it is possible to observe in Figure 1.10 in high duration zones where the current exceeds  $10mA$ , and in high current zones with currents greater than  $500mA$ , and it also fails when there is a "direct bipolar contact," that is a simultaneous contact with two active conductors: phase and neutral or phase and phase; in this case it is ineffective since there is no leakage current to earth.

### 1.2.1 Reference standards for differential devices

The reference standards for residual current circuit breakers for household and similar use, up to 400 V and 125 A, are IEC EN 61008-1 and IEC EN 61009-1. The former for circuit breakers so-called "pure" earth leakage circuit breakers (RCCBs), i.e., which perform only protection against differential currents; the second for so-called magneto-thermal earth leakage circuit breakers (RCBOs), i.e., the circuit breakers that also incorporate the function of over-current protection. Both of these standards should be supplemented with IEC EN 61543, which deals with the electromagnetic compatibility.



**Figure 1.11:** Wiring diagram of a single phase residual current circuit breaker [5]

### 1.2.2 Working Principle

Components:

1. Line terminals
2. Polarized release
3. Test and control button
4. Electromagnetic release
5. Thermal release
6. Differential transformer
7. Load terminals

The residual current sensor is a differential transformer consisting of a toroidal core made of magnetic material, a primary winding, physically consisting of the



assembly of several coilings made by winding line conductors onto the core with the same number of turns and same winding direction, and a secondary coiling . When no leakage is present, the vector sum of the currents in the conductors is zero and thus their respective contributions to the magnetic flux cancel each other out. If, on the other hand, there is leakage to earth a magnetic flux equal to the vector sum of the currents passing over the conductors is generated, which induces a voltage on the secondary winding. The primary current therefore will be the leakage current, which above a certain threshold causes the earth leakage circuit breaker to open. The trip actuator is a so-called demagnetization-polarized relay; it is precisely this type of actuator that is used because it does not require an auxiliary power supply, as it needs a very small amount of power that is drawn from the energy of the dispersion fault. Other residual current circuit breakers, however, which use for their operation the energy drawn from the power grid, usually have a common moving-core electromagnet as the actuator. The actuator and sensor generally are connected by a coupling circuit, which may be more or less complex depending on requirements.

### 1.2.3 Operation Method

1. Voltage Independent Circuit Breakers (VI): in order to set the trip mechanics in motion, they use only the energy due to the fault provided by the differential sensor itself; this means that they trip at the moment when the differential fault current  $I_d$  reaches the expected trip conditions, regardless of the actual presence of the line voltage at the input terminals.
2. Voltage Dependent Circuit Breakers (VD): they use the power grid itself on which they are mounted as their power source, so in the event of a lack of the correct mains voltage they will not function. They are also called "electronic RCD" because the interface circuit between sensor and actuator is an electronic circuit that amplifies the signal coming from the sensor and therefore requires an independent power supply. Consequently, the actuator usually consists of a moving-core electromagnet in place of the demagnetization-polarized relay.

The reliability in the system of the protection obtained through VD circuit breakers, is less than that obtained through VIs; this is because in the case where there is an interruption of the neutral upstream of the circuit breaker, the power supply is lost and therefore the VD is no longer able to intervene but having a phase present it is still possible for a ground fault to occur. This episode in single-phase systems does not occur frequently since, with the neutral interrupted, no equipment powered by the circuit breaker would be able to operate and the user might notice the fault but, despite this, the potential risk of indirect contact without any protection would remain. The risk, on the other hand, increases in three-phase circuit breakers fed

by two conductors, since despite having an interrupted phase the appliances fed by the circuit breaker could still operate.

From the functional point of view, we can distinguish:

- Magneto-thermal circuit breakers: combine the functions of the residual current circuit breaker with those of the circuit breaker, i.e., those of over-current protection;
- Pure residual current circuit breakers: they are without thermal-magnetic protection and they trip only in the event of a fault with earth leakage;
- Differential circuit breakers: these are not true circuit breakers being devoid of the trip mechanism, contacts and input terminals, but are used in conjunction with a circuit breaker.

**Classification of differential circuit breakers according to tripping sensitivity:** differential circuit breakers are classified according to their tripping sensitivity, by means of the rated differential tripping current,  $I_{\Delta n}$ , which is the differential current at which the circuit breaker must compulsorily trip within a predetermined time; the standards also introduce a lower tripping threshold, "rated differential non-tripping current" equal to  $0.5I_{\Delta n}$ , which represents the maximum differential current at which the breaker must not trip even if applied for an indefinitely long time. Switches are classified as follows:

- high sensitivity: for values of  $I_{\Delta n}$  equal to  $6mA$ ,  $10mA$ ,  $30mA$ ;
- low sensitivity: for values of  $I_{\Delta n}$  equal to  $100mA$ ,  $300mA$ ,  $500mA$ ,  $1A$ ,  $2A$  (not suitable for additional protection).

The most commonly used type is the  $30mA$  type, because it performs both additional protection and protection from indirect contact and fire at the same time; circuit breakers at  $10mA$  give higher additional protection but are not used often because they are prone to unexpected trips and do not provide continuity of service.

Residual current circuit breakers are also classified according to the tripping time, which in general is the time interval between the instant of instantaneous application of fault current and the instant of contact opening. In addition, the standards also define the minimum non-tripping time, which is the maximum time interval between the instant of instantaneous application of the fault current and the instant of its sudden termination, such that the breaker still remains closed. On the basis of these parameters of identify switches of type G "generic" , type S "selective".

On all TWO types G and S, high immunity is achieved against disturbances of an impulse that could cause the switch to trip untimely.

	$I_{\Delta n}$	$2 \times I_{\Delta n}$	$5 \times I_{\Delta n}$	5 A, 10 A... 200 A, 500 A	
Generic (not delayed)	0,3 s	0,15 s	0,04 s	0,04 s	minimum non-intervention duration
Type G	0,3 s	0,15 s	0,04 s	0,04 s	minimum non-intervention duration
	0,01 s	0,01 s	0,01 s	0,01 s	minimum non-intervention duration
Type S (only $I_{\Delta n} > 30$ mA)	0,5 s	0,2 s	0,15 s	0,15 s	minimum non-intervention duration
	0,13 s	0,06 s	0,05 s	0,04 s	minimum non-intervention duration
Type M (only $I_{\Delta n} > 30$ mA e $I_n > 32$ A)	5 s	1 s	-	0,5 s	minimum non-intervention duration
	0,2 s	0,2 s	-	0,2 s	minimum non-intervention duration

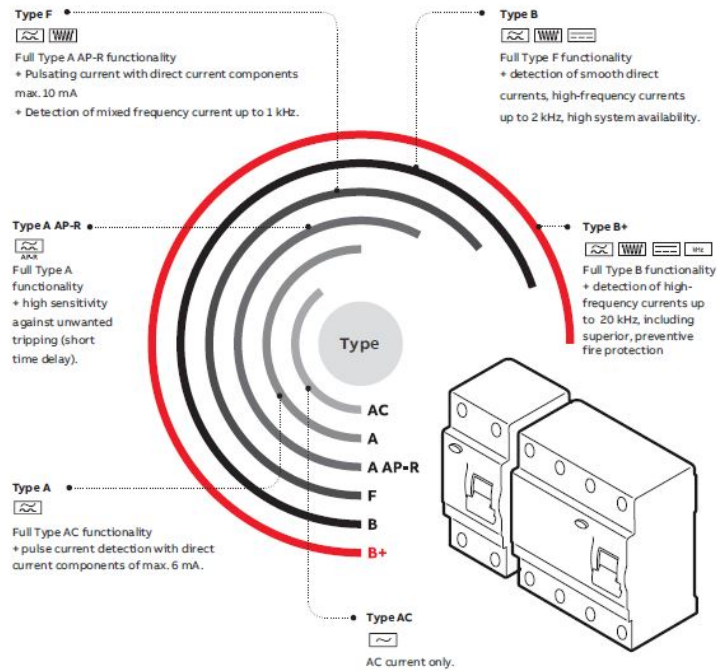
Figure 1.12: Classification type G,S,M[5]

**Classification of residual current circuit breakers according to the waveform of fault current  $I_f$ :**

all circuit breakers covered by the current standards are expected to be installed on sinusoidal alternating-voltage networks at a frequency of 50/60Hz; what happens, however, is that the presence of electrical equipment using nonlinear semiconductor electronic circuits connected to the power supply may result in the leakage to earth of differential currents that are not of the sinusoidal alternating form, a fact that leads to two problems. The first problem relates to the fact that differential currents may not be detected by common-type circuit breakers and thus may not trip; in addition, the presence of non-alternating differential current may alter the behavior of the differential in the event of a superimposed alternating differential current.

Classification [6]:

- Type AC RCD: operation is guaranteed only for suddenly applied or slowly increasing sinusoidal alternating currents; they are suitable for purely resistive loads or for circuits as phase adjusters.
- Type A RCD: includes the case of type A and also guarantees tripping in the case of unidirectional pulsating differential currents with or without phase angle control and pulse current detection, with DC components of maximum 6 mA. They are usually used for single-phase rectifiers, bridge rectifiers, and phase bias rectifiers.
- Type A AP-R RCD: operation is guaranteed for AC case and also has high sensitivity against unwanted tripping;
- Type F RCD: guaranteed tripping as for the A AP-R switch and in addition for pulsed current with DC components maximum 10 mA and mixed-frequency



**Figure 1.13:** Classification of residual current circuit breakers [6]

current sensing up to 1 kHz. These are used to ensure protection even in the presence of inverters.

- Type B RCD: which trip is guaranteed for case F and in addition detection of smooth direct currents, high-frequency currents up to 2 kHz, high system availability. these are also used in the presence of three-phase or two-phase rectifiers.
- Type B+ RCD: includes type B operation and also detection of high-frequency currents up to 20 kHz, including superior, preventive fire protection.

A key characteristic is the robustness to unwanted tripping; by this is meant any opening of the circuit breaker in the absence of an actual ground fault. The causes of unwanted tripping can be of different kinds, whether of permanent or transient type, currents flow to earth by means of normalized impedance of capacitive type, present between the line conductors and the earth, which precisely because of their capacitive nature favor the dispersion to earth of currents, whose harmonic content is shifted toward frequencies greater than that of the grid. The main causes of unwanted tripping are disturbances of an impulsive nature due to over-voltages of lightning origin, or from switching devices and the presence of reactive loads.

Standards today provide standardized tests to verify robustness to this type of intemperate tripping, such as the SURGE test.

### 1.3 MOD&COM: Motor operating device and Communication module

The RCD switch, in the specific case, is coupled with a "motor operating device", commonly called MOD, which is part of the Motor Drivers family, which can be high (110-240  $V_{AC}$ ) or low (24-48  $V_{AC/DC}$ ) voltage, depending on the model. It operates at a nominal frequency of 50 – 60 $Hz$ , with a maximum power consumption of less than 20 $W$ , which when idle becomes less than 0.4 $W$ . The rated impulse withstand voltage (1.2/50) is 4 $kV$  for the HV and 800 $V$  for the LV, while the dielectric test voltage is 2.5 $kV$ , which by the current standard becomes 2 $kV$ , at a height of 2m and an operating time of less than 1s.

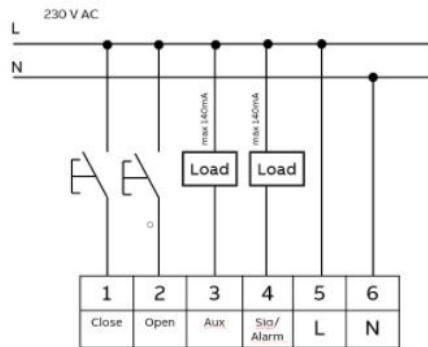


Figure 1.14: Connection for Mod [7]

**Operation:** The MOD gives the possibility to have manual or remote opening and closing of the coupled device. When the coupled device opens due to a fault, the open or close command is delayed for a time of 8 seconds, at which time the coupled current breaker, which trips due to an over-current, for example, is given a chance to cool down the bi-metal and return to rest. Led operations are shown in Figure 1.15:

MOD operations during a mains voltage drop:

Regarding the connectivity, communication accessories can be with wired connectivity, thus with RS485 protocol, or Wireless.

Through these communication accessories it must be possible to read the status of the device and also give basic commands, remotely. The communication accessory is coupled from the MOD through a 4-pin connector that communicates via RS485 protocol, and is powered by the MOD itself. In the specific thesis case, the communication accessory is wireless and communicates via MODBUS TCP protocol, over a 2.4 $GHz$  local area network with the 802.11b/g/n standard. This device called COMM, is equipped with bi-color LED (red/green) indicating the status of the device, and also has a button for initial configuration or reset to factory settings.






MOD		
	OFF	Device not powered
	Blinking Green	Device Powered but remote commands NOT activated
	Fix Green	Device Powered and remote commands activated
	Blinking Red	-
	Fix Red	MPD has trip for a fault

Figure 1.15: Led Mod operations [7]



















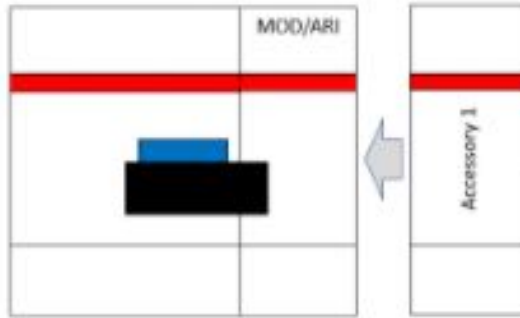
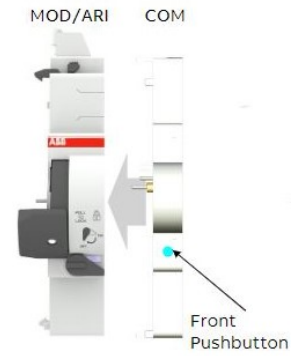
MPD status	MOD status	MOD status definitions	LED before drop	What's happen during the drop off (ARI turns OFF)	What's happen at the mains restoring	LED after drop
Steady Closed (ON)	Active	The device is ready to work		MPD stays ON	It recovers its status: MPD closed and MOD activated	
Steady Closed (ON)	No Active	The remote commands are not active		MPD stays ON	It recovers its status: MPD closed and MOD not active	
Steady Open (OFF)	Active	The device is ready to work		MPD stays OFF	It recovers its status: MPD open and MOD activated	
Steady Open (OFF)	No Active	The remote commands are not active		MPD stays OFF	It recovers its status: MPD open and MOD not active	
Steady Open (OFF)	Alarm	The status after a fault trip of MPD		MPD stays OFF	It recovers its status: MPD open and MOD in Alarm state	
Steady Open (OFF)	Dead time	The time needed to activate the remote commands after a trip		MPD stays OFF	It recovers its status: MPD open and MOD in Alarm state; dead time is reset	
Closing	Remote closing	The closing of an MPD from remote		The handle could stop in middle position	Open manoeuvre then stay open	
Opening	MPD Tripping	The fault trip of an MPD		MPD most probably goes in OFF	Open manoeuvre then MOD goes in Alarm state (no dead time)	
Opening	Remote opening	The opening of an MPD from remote		MPD most probably goes in OFF	Open manoeuvre then it recovers its status: MPD open and MOD activated	

Figure 1.16: Connection for Mod [7]

On first connection or after a reset, the wireless device provides its own wireless network with the specific SSID and has a specific dedicated web page where you can go to set the settings but also change its modes of operation. In fact, the COMM-WIFI works in "access point" mode for which it generates its own network as mentioned before, or in "station" mode for which it instead connects to the local network and is assigned a specific static or dynamic IP address, depending on the user's choice. In this specific case, the device was used in station mode



**Figure 1.17:** Representation switch-mod-com [7]



**Figure 1.18:** MOD-COMM connection [8]

and connected to the laboratory network. Information on the MODBUS TCP/IP communication protocol is given in Appendix B.



## 1.4 Machine used to perform the surge test: AMETEK CTS

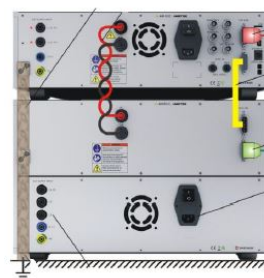
The machine with which the surge test is carried out is the AMETEK CTS compact NX7, used together with coupling and decoupling NX. The compact NX7 is a compact generator that simulates the effects of electromagnetic interference conducted for immunity testing, according to standards; in particular, it contains the modules of Burst, Surge, Ring wave, Power fail.

The machine can work locally or remotely via the iec.control software with which communicates via serial communication. On the front of the machine there are apart from stat, stop and back buttons, the BNC outputs for measuring generator current pulse, generator voltage pulse and generator trigger. On the back instead, important is the external synchronization output.

The machine can have single-phase internal coupling or three-phase external coupling; in the specific case it is used with external coupling.



**Figure 1.19:** AMETEK CTS COUPLING NX7 front side[9]



**Figure 1.20:** AMETEK CTS COUPLING NX7 back side[9]

In the front of the NX coupling network we can see at the bottom right the outputs to the EUT, as the surge pulses are coupled to the power lines. At the back of the machine there are the Ground, both for the generator and the coupled network, connected to the ground plane, the HVS surge cable, SLC 500 Sys.Link cable, the machine power supply, and the EUT supply inputs. In this specific case the coupling used is the NX7 bsr-3-690-32 coupling, with EUT AC voltage  $[V]=3 \times 690$ , EUT current ranges  $[A]=32$ .

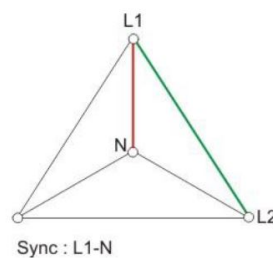
The machine internally is divided into two parts: the decoupling and filtering part, which is necessary to decouple the low-impedance mains power supply from the setup under test and also to protect the equipment connected to the power

supply, and then there is the coupling part that injects the transients to the lines of a power system, AC power and DC power. According to IEC 61000-4-5 the required couplings are: line to line (L1-L2, L1-L3, L2-L3, L1-N, L2-N, L3-N) or lines to earth (L1-PE, L2-PE, L3-PE, N-PE).

Data for this type of coupling network:

1. Impulse voltage Surge: 7.0 kV 3.5 kA  $\pm$  10
2. Regarding the EUT:
  - (a) 3 phase: L1, L2, L3, N, PE DC: L1/DC+, N/DC-;
  - (b) EUT supply voltage: 3 x 690 V AC  $\pm$  10%, 1.000 V DC;
  - (c) Frequency: 50 Hz, 60 Hz;
  - (d) DC current capacity is same as for AC with separate DC input plugs.
3. Regarding the Output Plug:
  - (a) AC lines: 32 A 6 mm safety banana plug ;
  - (b) DC lines: 32 A 6 mm safety banana plug ;
  - (c) Data lines Surge 4 mm safety banana plug, double insulated;
  - (d) DC Input (rear): 200 A; KBT10BV-AX/M, 10 mm socket with bayonet locking;

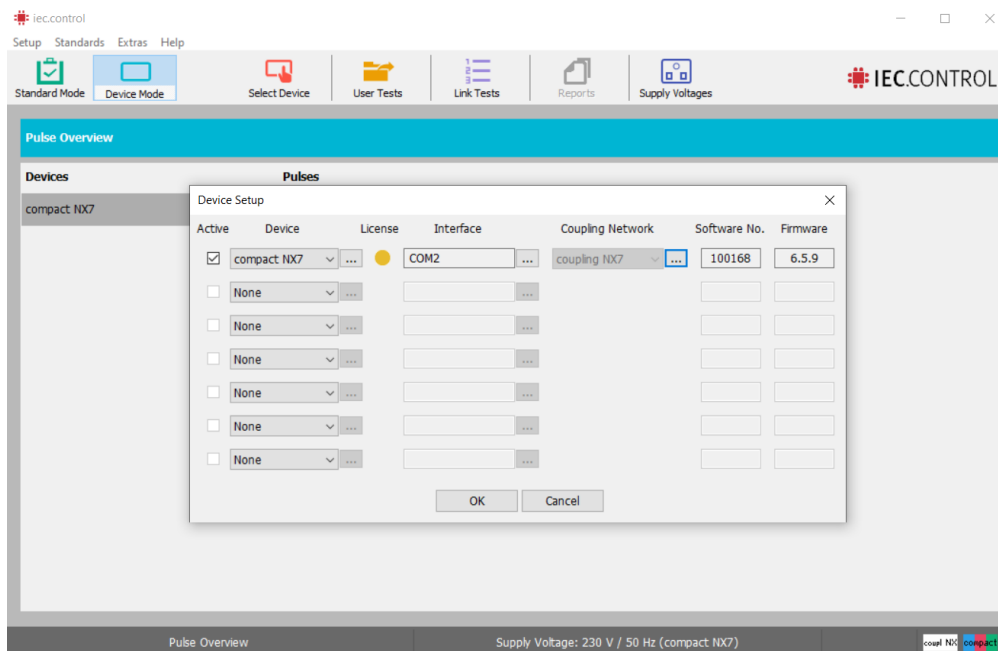
As for the synchronization signal to activate the Surge, this is generated in three-phase coupling, taken between L1-N; if the neutral is missing, as in the case of a delta supply, then an artificial neutral point is given by an artificial high impedance network. For each possible coupling the generator calculates the correct phase angle.



**Figure 1.21:** Delta supply neutral point [10]

### 1.4.1 Software Iec.control

As mentioned earlier the machine can be controlled remotely through the Iec.control software, which communicates via USB OR IEEE488 interface. When the software is started, it is possible to set from the "Device Setup Menu", the devices with which you want to run the test, if networked and thus visible through the serial interface; then it is possible to choose the standard to be followed and the type of test to be performed, all tests executable by the machine can be managed through the software. Also from the "Device Setup menu", in the "Coupling Network" section it is possible to set watchdog events, so that you can choose how to continue if communication is lost; in particular, it is possible to choose whether to stop the test or to continue it [11].



**Figure 1.22:** Device setup menu.

From the "Supply voltage Setup menu" it is possible to set the parameters of the Variac transformer (if connected); in particular the voltage range is chosen, which can be High: 230V/50Hz or Low: 115V/50Hz; it is also possible to set the maximum voltage of the Variac, the time required from 0% to 100% and if necessary also the Variac control speed correction value.

If instead a Tapped Transformer is used, it is also possible to set the type of transformer whether manual or automatic.

When all these settings have been made it is possible to choose the type of test and pulse to be applied on the EUT.

In the specific case it is chosen to do the  $1.2/50\mu s$  surge test for which you have a "Pulse window" where you can start the test by choosing the preferred parameters for the test:

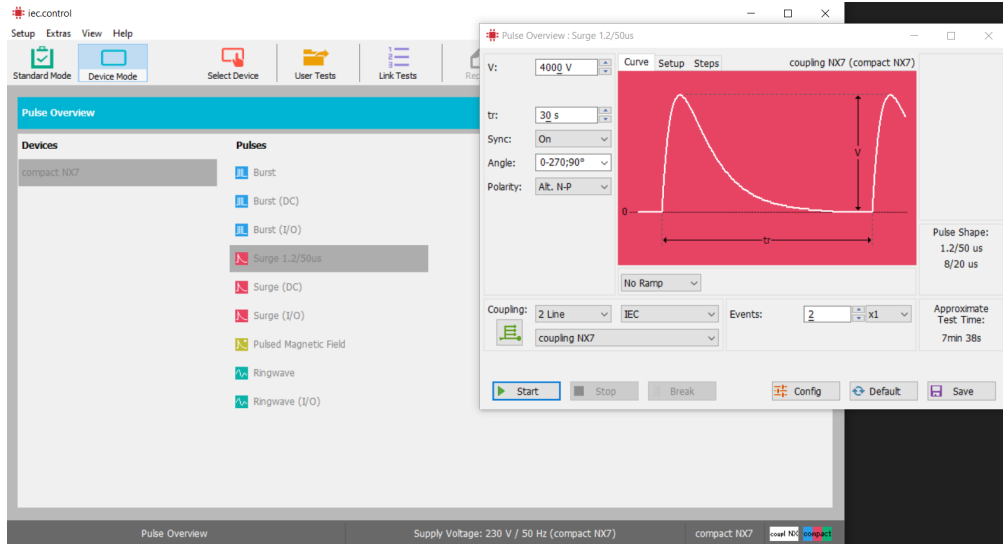


Figure 1.23: Pulse window.

You set the voltage to be applied, and the duration of the pulse, the synchronization whether active or not, the sequence of angles to be applied, which can be normal or random angles, the polarity only positive or only negative or alternating, the coupling and also there the correct coupling network is entered; finally the number of pulses needed for the test is set. From this screen, using the start button, it is possible to start the test, suspend it and stop it and also it is possible to view a log file in which all the communication between the machine and the software is shown.

In the appendix c it is possible to find all the commands that are sent and the responses received by the machine for remote control.

## Chapter 2

# COMMUNICATIONS IMPLEMENTATION

This chapter contains the basic body of this thesis work. Having seen the context in which the work was carried out, the reference standards and the operation of the devices, let us now see what devices were used and needed to be managed for this thesis work and how communication with them was implemented. The purpose of the project was to create a Labview program capable of managing and commanding the entire test, thus replacing the user who up to that point had been placed constantly to follow the test and rearm the part, whenever it was tripped, verifying that no irreversible damage had occurred, and also going to record at each surge pulse applied on the EUT what the voltage and current values were, so that we would then have a final overall view of the test progress.

To carry out the entire Surge Test, apart from the devices already seen in the introductory chapter, such as AMETEK CTS machine and the MOD associated with the communication module, it was also necessary to use an oscilloscope capable of measuring the voltage and current output from the machine.

The entire implementation of the labview program for the management and communication of each device is shown below.

### **2.0.1 Ametek cts machine management**

As seen in the introductory chapter, to remotely control the AMETEK CTS machine it is necessary to use the iec.control software, therefore not being able to do without it, what was deemed necessary for communication with the machine was access to the communication between the machine and the software.

As we know, it takes place serially, in particular on COM ports, so the first step of the project focused on the management of these ports.

A com port can be managed by only one software at a time, so having to have access via Labview to the communication that is managed by another software, it was necessary to create a structure capable of creating virtual com ports on which to redirect the data traffic, exchanged between the machine and the Iec.control and also able to send commands to the machine. This first part of the work was not as simple as one might think. Initially it was thought to use open source solutions, but from the first moment they showed inefficiency; subsequently, having a USB output from the machine, a hardware type solution was also thought of, therefore a physical USB splitter, which however was unusable as the solutions on the market in most cases provide for data transmission only on one of the two arms of the splitter, while the other is used for the power supply. Therefore a solution of this type would have been useless and moreover inefficient, since the structure of a splitter does not foresee that the two new ports created respectively see what the other port is sending.

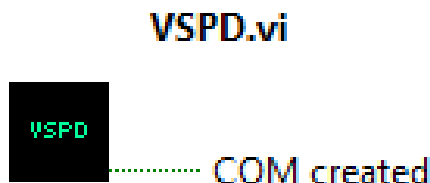
What was thought, to be the best choice, was the creation of a "complex COM port bundles" through the use of the VIRTUAL SERIAL PORT DRIVER PRO software. The above structure enables the transmission of serial data in and out of any number of real or virtual serial interfaces. With this advanced function, data received from one of the "In" side ports can be redirected to all ports of the "OUT" side and vice versa. So the following structure was created:

```

in com4 real
in com6 virtual
out com1 virtual
out com2 virtual
    
```

The machine communicates through the com1 virtual port with the Iec.control, Labview is able to read the data sent by the machine to the software on the virtual com2 port, while the data sent from the software to the machine are read on the virtual com6 port.

What has been seen was realized using the SDK version of the above program, which can be integrated on labview; for this purpose, a special VI was created, for the creation of such a structure, called "VSPD", shown in Figure 2.1.



**Figure 2.1:** Labview VI VSPD.

This VI inside, as observable in the Figure 2.2, contains:

- The Connect to Local Service function which establishes connection with VSPD Pro Service on the local machine for bundle creation and ports management;
- the Add Bundle function which creates a new ports bundle, to which the bundle name is given as input, which must always be of the type "Complex bundle (1)" and which returns on the output "function return" value 1 or 0 respectively whether the structure is created correctly or not;
- the Bundle Set Baud Rate Emulation function which enables/disables baudrate emulation for virtual ports in the specified bundle; input to this function is given in name of the reference bundle and also the value of the baud rate you want to set. This function returns in output value 1 or 0 respectively if the setting was successful or not.
- the Bundle add port function is repeated in parallel 4 times in order to add the necessary ports to the structure.

This function requires as inputs the name of the referenced bundle, the name of the port to be added, the side in which it is to be added, i.e., inbound or outbound, and finally the type of port, whether real, virtual, shared or switcher. Instead, as output it returns 1 or 0 and depending on whether the ports are created correctly or not.

In the specific case, first the real COM4 input com port is created, then the virtual COM6 input port, then the virtual COM2 output port, and finally the virtual COM1 output port.

- After the creation of COM6 and COM1 port, two functions "Bundle Set Main Port For Side" are used to set the main port for each side. In our case the main ports will be COM4 in input and COM2 in output.
- Once the COM6 port is set as the main one, the function " Get Bundle State" is then used which gets the current state of a port bundle.

Having done this, if everything has been done correctly a Boolean variable is set as true.

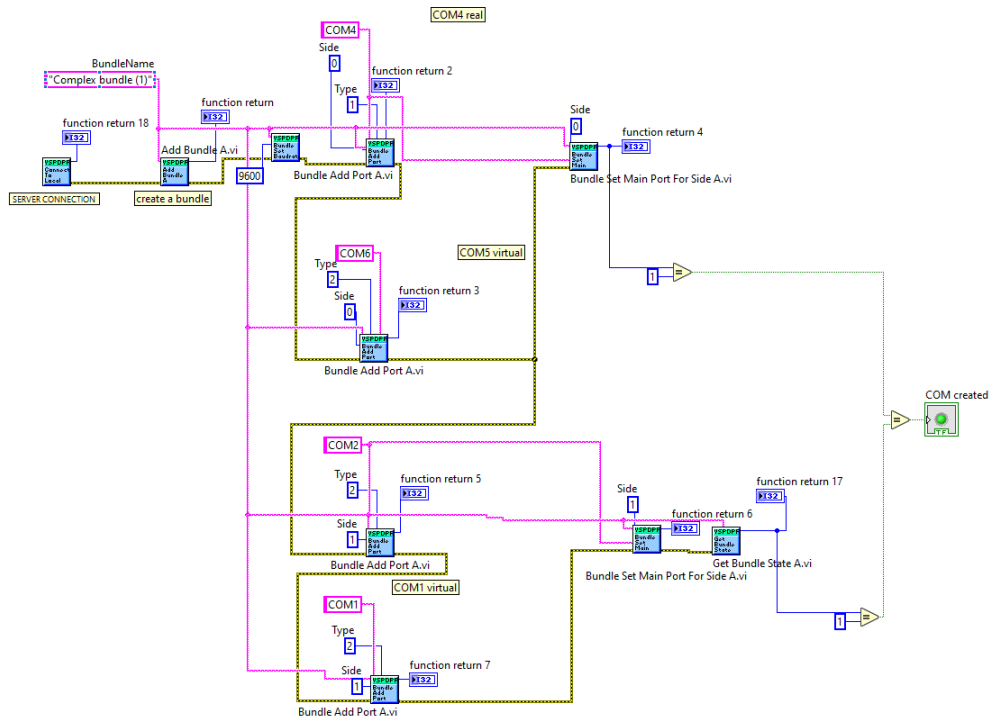


Figure 2.2: Labview block diagram of VSPD VI.

Once this structure was created, it was possible to proceed with the creation of a Labview subVI, called "AMETEK COMMUNICATION" shown in the Figure 2.3, which carries out all the reading part on the ports.

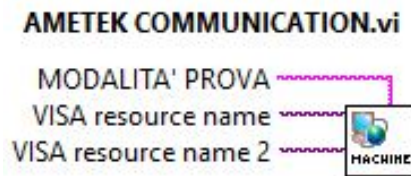
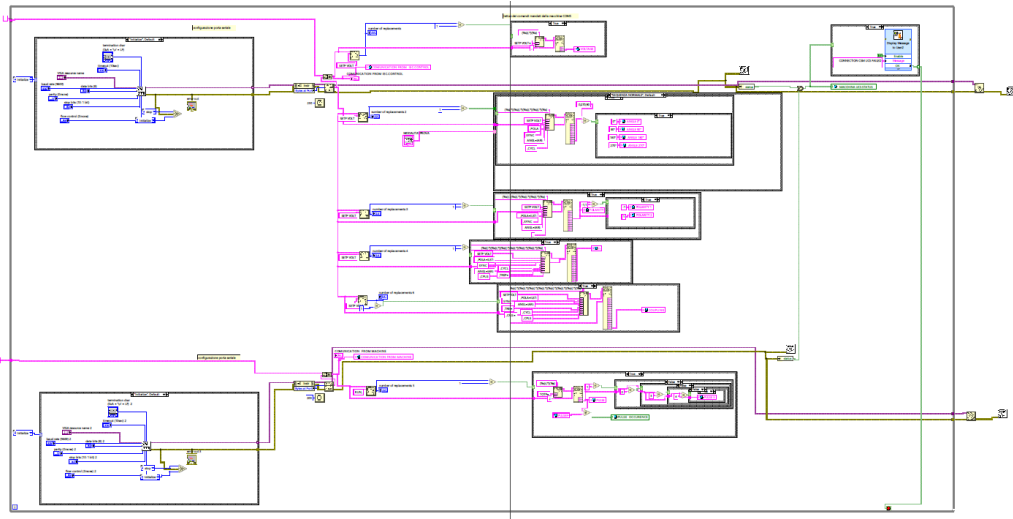


Figure 2.3: Labview subVI AMETEK COMMUNICATION.

The block diagram of the subVI is shown below:

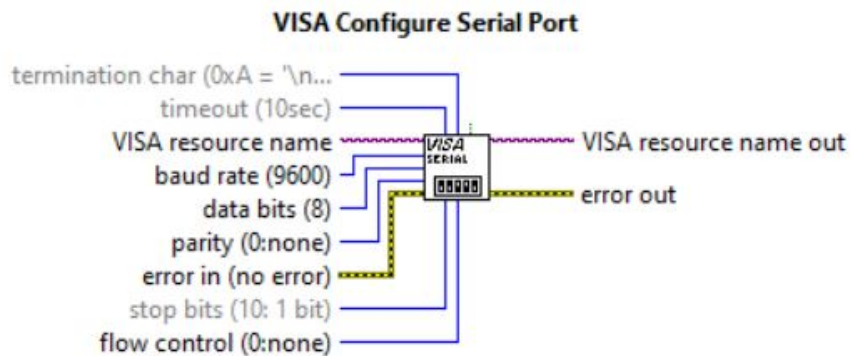




**Figure 2.4:** Labview subVI AMETEK COMMUNICATION BLOCK DIAGRAM.

Since the graph is very large and this figure is not clear, it is preferred to analyze the implemented schematic in several blocks.

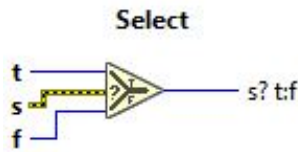
In the first block, the configuration of the serial port was implemented using the VISA configure serial port driver function, in which the baud rate, data bits, party bit, stop bit, flow control bit, timeout and termination character settings are set.



**Figure 2.5:** Visa configure serial port VI.

Furthermore, through a case structure composed of the INITIALIZE and STOP cases, and through the use of "SELECT" which outputs the values of t or f, depending on the work of s, if s=true we will have output t, if s= false on output we will have f.

So in our case what happens is that, if the configuration of the port is successful, the error terminal connected to "s" of the Select will be false and it will output



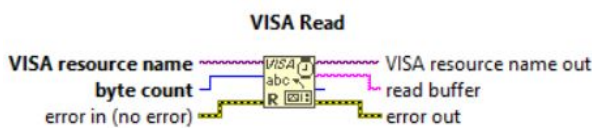
**Figure 2.6:** Select.

"initialize", and therefore continuing to initialize the port; if instead the error terminal is true, so something in the configuration has gone wrong, then Select will go to stop and the user will be notified via a pop-up that the connection with the AMETEK CTS machine has failed.

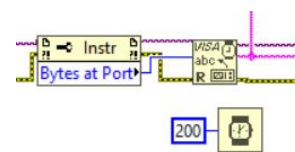
So this block is also used to monitor the status of the connection with the machine during the test, as it is placed inside a while loop, which then continues to repeat this operation until the connection with the machine fails.

This block is used twice for both ports used by Labview to communicate with the machine. To read the communication on the ports, the VISA read function is used, which, specifying the port from which it must read, reads the specified number of bytes, in our case it is passed through the Property Node set to "bytes at port", which constantly returns the number of bytes present on the port, since the number varies during communication. At the output of the VISA read function it is present the "read buffer" terminal on which there are the read data. Then communication is shown on this terminal.

Furthermore, a  $200ms$  wait is inserted, in order to clearly read the communication on the read buffer and the latter is inserted in a "concatenate string" together with an empty string initialized outside the while loop, so that the strings are shown in sequence, as is possible to observe in figure 2.8.



**Figure 2.7:** VISA READ function.



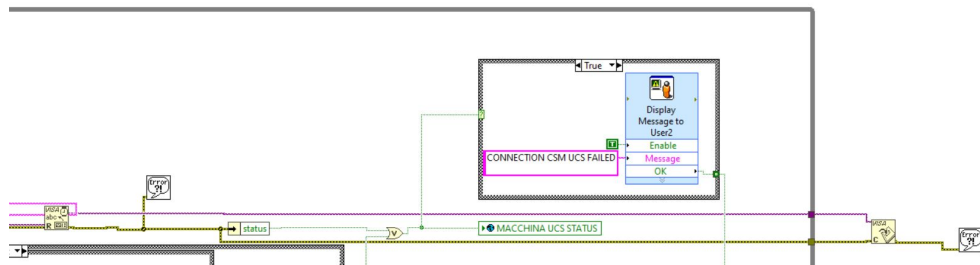
**Figure 2.8:** VISA READ function and the Property node to pass it the bytes at port.

Inside the while loop an "Unbundle by name" set to state has also been added, which, having as input the error terminal in output from the Visa read, returns true or false output value. If there is an error while reading, the status will be set as true and will be notified through a pop-up and the while loop will be stopped.

In addition, the status output is sent as a write to a Boolean global variable called "UCS STATUS MACHINE" which will later be used in other parts of the code and which in particular if true will interrupt the entire program execution, Figure 2.9.

Outside the while loop, once interrupted, the VISA close function is then used to terminate the Visa session opened at the beginning.

What was discussed has been done for both com ports and the two states are then put in or logic function and both connected to the "machine ucs status" variable, since it is necessary that the test is interrupted if one or both ports has an error communication.



**Figure 2.9:** Read ucs machine status and notification.

Now what is important to understand is what information needs to be extracted from the communication read.

What needs to extract are parts of the communication which are then used for control or which are used as variables in other parts of the program. In particular, from the reading on com5, which is therefore the one relating to the information sent from the machine to Iec.control, it is necessary to extract the set voltage value at which the surge test is performed, the angle, the polarity, the pulse duration and the coupling.

Regarding the angle, the sequence of angles entered can be "NORMAL", from 0° to 270° with 90° steps, or "RANDOM", from -198° to 162° with 36° steps. The polarity, on the other hand, can be positive or negative.

To intercept these values, knowing the structure of the strings from the machine manual, it was possible to implement the following mechanism: the program through the "Search and Replace String" function searches, within the strings relating to the communication of the COM5 port, the string which is the beginning of the string that contains the information; if it is found, the mentioned function gives the value 1 on the terminal "number of replacement", and it enables the

case structure made up of true and false cases. If true, then through the "Format Into String" function, the piece of string located between the piece of string that precedes the value sought and the next is searched for within the sentence and this is shown on the "resulting string" terminal, which is sent on the "regular expression" terminal of the "Match regular expression" function, which on the other "input string" terminal has the communication strings. This function searches for a regular expression in the input string starting at the entered offset. If the function finds a match, it splits the string into three sub-strings and any number of sub-matches, and displays the value sought.

In the specific case of the angles it is known that the string in which is contained the information is of the type:

*"SETPVOLT = 4000, POLA = 1, SYNC = ON, ANGL = IARI : 0 : 270 : 90, CYCL = 2, TREP = 15, CPLG = L1 - N, CPLS = IEC, CPLT = SURGE - IEC - 3PH, EUT1 = NOTF, TRIF = OFF"*

so the "Search and Replace string" function searches for "SETP VOLT", when it is found a case structure is sets to "true", and there the "Format Into string function" starts looking in the function among the SETP VOLT, POLA, SYNC, ANGL=IARI, CYCL offsets and the "Match regular expression" function will find the searched value as the eighth sub-match.

If the value is found, another case structure on "true" is enabled, within which the values of the angles that will affect the test are assigned to global variables, then used in other parts of the program. Since the test can be carried out in normal sequence or in random angles, as previously said, then the user is expected to set the chosen mode before starting the program, and if the normal sequence is selected, the angles found will be  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$ ,  $270^\circ$ , while in case of random angles it will be  $198^\circ$ ,  $234^\circ$ ,  $270^\circ$ ,  $306^\circ$ ,  $342^\circ$ ,  $18^\circ$ ,  $54^\circ$ ,  $90^\circ$ ,  $126^\circ$ ,  $162^\circ$ .

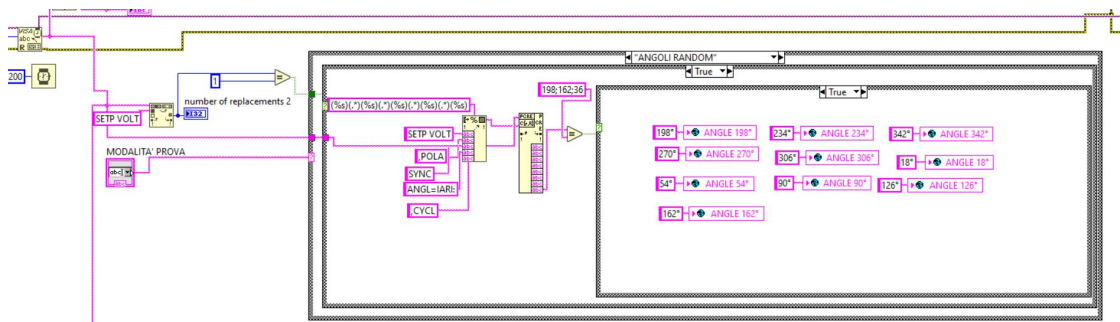
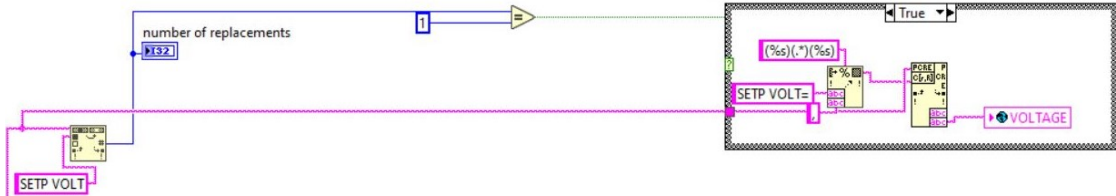


Figure 2.10: Search of the value of the angles in the case of random angles.

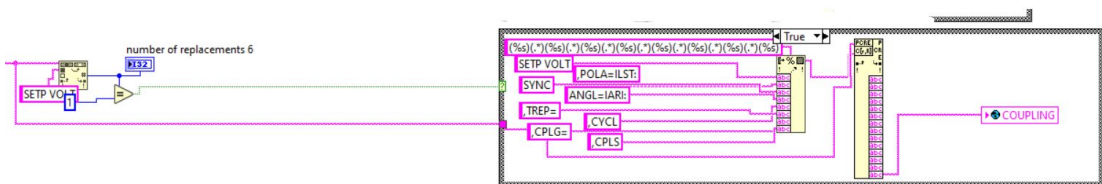
In the case of the voltage, in the string like the previous, the "Search and Replace string" function searches for "SETP VOLT", when it is found a case structure

is sets to "true", and there the "Format Into string" function starts looking in the function among the "SETP VOLT" and "," offsets and the "Match regular expression" function will find the searched value as the second sub-match and it will write this string value inside a "VOLTAGE" global variable.



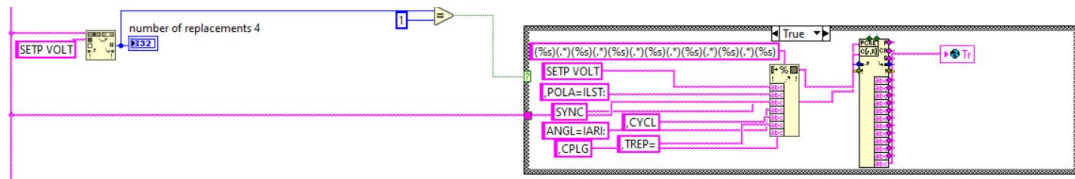
**Figure 2.11:** Search of the value of the voltage.

In the case of the coupling, in the string previously viewed, the "Search and Replace string" function searches for "SETP VOLT", when it is found a case structure is sets to "true", where the "Format Into string" function starts looking in the function among the SETP VOLT, POLA, SYNC, ANGL=IARI, CYCL, CPLG=, CPLS offsets and the "Match regular expression" function will find the searched value as the fourteenth sub-match.



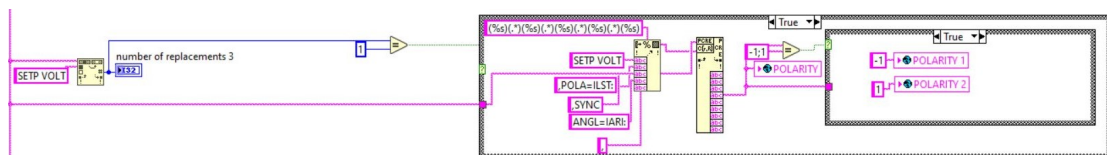
**Figure 2.12:** Search of the value of the coupling.

In the case of the pulse duration, in the string previously viewed, the "Search and Replace string" function searches for "SETP VOLT", when it is found a case structure is sets to "true", and there the "Format Into string" function starts looking in the function among the SETP VOLT, POLA, SYNC, ANGL=IARI, CYCL, TREP, CPLG=, CPLS offsets and the "Match regular expression" function will find the searched value as the twelfth sub-match.



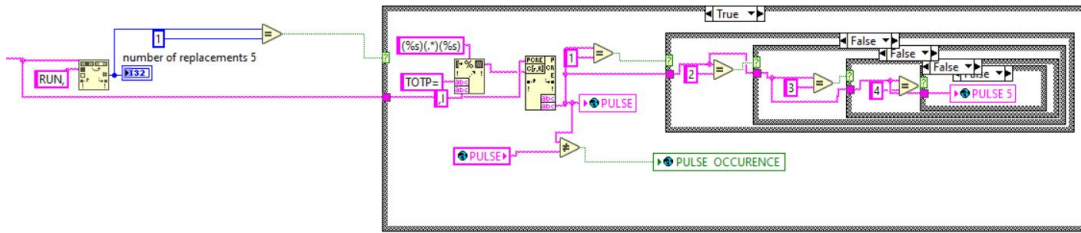
**Figure 2.13:** Search of the value of the pulse duration.

In the case of the polarity, in the string previously viewed, the "Search and Replace string" function searches for "SETP VOLT", when it is found a case structure is set to "true", and there the "Format Into string" function starts looking in the function among the SETP VOLT, POLA, SYNC, ANGL=IARI offsets and the "Match regular expression" function will find the searched value as the fourth sub-match and it will write this string value inside a "POLARITY" global variable. If the found value is equal to -1:1, then the global variable POLARITY 1 will be associated to the value -1, while the global variable POLARITY 2 to the value 1.



**Figure 2.14:** Search of the value of the polarity.

As for com2 port communication, the current pulse value is needed; according to the working principle described above the Search and Replace string function searches for "RUN,", when it is found a case structure is set to "true", and there the "Format Into string" function starts looking in the function among the "TOTP=" and "," offsets and the "Match regular expression" function will find the searched value as the second sub-match and it will write this string value inside a "PULSE" global variable. If the found value is equal to 1, then the global variable PULSE 1 will be associated to 1, instead if the found value is equal to 2, then the global variable PULSE 2 will be associated to 2; if the found value is equal to 3, then the global variable PULSE 3 will be associated to 3, instead if the found value is equal to 4, then the global variable PULSE 4 will be associated to 4 or if the found value is equal to 5, then the global variable PULSE 5 will be associated to 5, since the surge test usually is done with a number of pulse equal to 5.

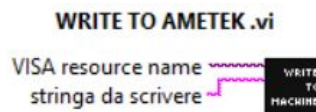


**Figure 2.15:** Search of the value of the pulse.

As previously mentioned, all what was described is carried out within a while loop which is interrupted only in the event whereby the communication with the machine fails on one of the two ports or on both.

As can be seen in the Figure 2.3, the output from this subVI does not have any terminal, but the user must enter the mode in which the test is carried out, thus choosing between normal or random sequence mode, and the names of the com ports on the which ones to communicate.

In another section of the program it was then necessary to send commands to the machine, directly from the Labview program created for test management and not through the Iec.com software; in this regard, another subVI was created, dedicated only to writing and it is called "WRITE TO AMETEK" , Figure 2.16-2.17.



**Figure 2.16:** Write To AMETEK NX7 subVI

Within this VI, as previously seen, communication with the machine is established through the "VISA configure serial port" function, setting the correct port COM settings, and then through the "VISA write" function, Figure 2.18, it is sent to the machine the desired command.



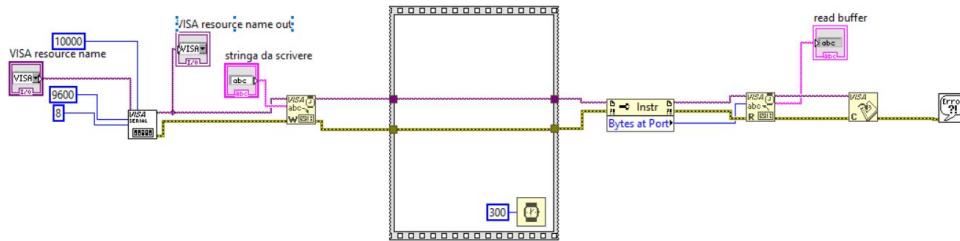


Figure 2.17: Block diagram Write To AMETEK CTS subVI.

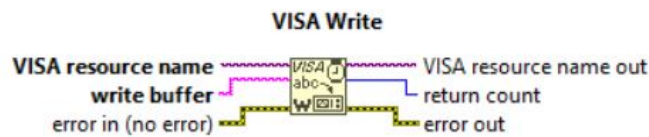


Figure 2.18: Visa Write function.

By setting a wait of 300ms and then using the "VISA read" function with input "bytes at port", it was possible to verify that the command was written correctly. Subsequently, the communication session is closed with the "VISA close" function.

## 2.0.2 MOD-COMM communication management

In the introductory chapter about the EUT, which in our case is a DIN-RAIL RCD, it was said that it communicates by means of the COMM communication module through the MODBUS TCP/IP protocol.

One of the purposes of the thesis project was to replace the operator who during the surge test was placed in front of the RCD, ready to manually reset the sample in case of tripping, checking that it had not had permanent damage; for this purpose in this chapter will be shown as the Modbus communication management has been implemented both in reading and in writing, as it has been deemed necessary to constantly read the contents of the register in which the trip event is signaled and, in this case, the reset is commanded by writing on the appropriate register, checking that the sample is not damaged and that therefore the test can be continued without problems or otherwise by interrupting the test.

The subVI created was called "Modbus COMMUNICATION" as shown in the Figure 2.19, in which the IP address of the communication module, the port on which it communicates and a Boolean variable used to stop the test are present in input; therefore, if the stop test terminal is "true" this subVI is broken.

Regarding the IP address, this is assigned by the user who will connect the



communication module to his network, through the dedicated web page and this can be both static and dynamic: a static IP address is recommended so that this setting does not have to be changed every time the device is used. The port on which it communicates has been set to 502 and should not change.

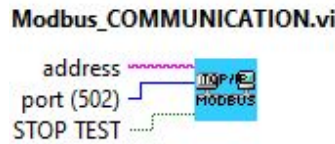


Figure 2.19: Modbus communication subVI.

The block diagram shown in the Figure 2.20 is analyzed piece by piece, being very large and unclear.

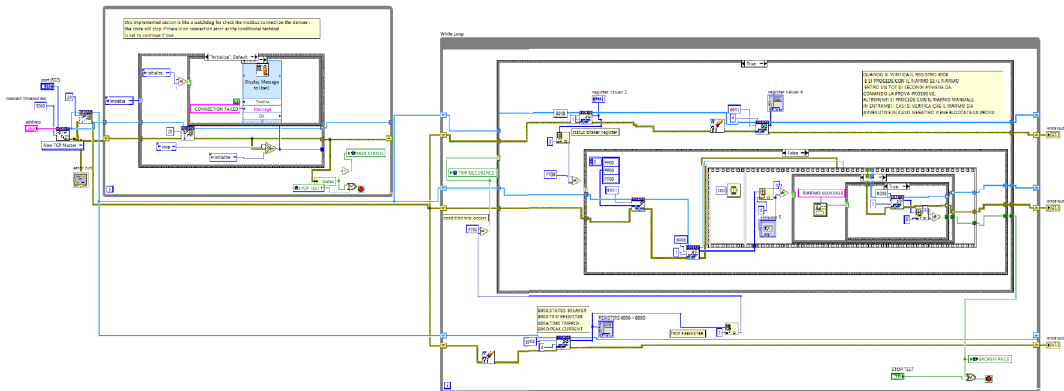


Figure 2.20: Modbus communication subVI block diagram.

As can be seen in the Figure 2.21, the "Create TCP Master" function of the Modbus API library is used to establish the connection with the device, which Creates a standard TCP master using an IP transmission data unit. This instance looks for a slave at the address and port specified and the created TCP instance is input to the "Set Unit ID VI" of the same library, which provides a simple function for defining the unit ID, which is the target of Modbus master read and write operations. The unit ID in the specific case is 20.

Once this has been done, another block has been implemented in order to monitor the connection. This consists of a while loop, within which a case structure is present and which case selector is a control made up of two cases, "initialize" and "stop". When the program is started, the control is on the "initialize" case, for which there is continuous initialization of the port through the "set unit ID " function

seen above, to verify that there are no connection errors. The error terminal at the output of the function is connected to a selector, like the one previously used to check the connection with the machine, so if there is an error, the Selector will output the stop case which will go as a control signal to another case structure, for which a pop-up will be shown to notify the user of a communication error. If instead there are no errors, the Selector will output the case "initialize" and the initialization will continue cyclically.

The error terminal is also linked to an "Unbound by name" set to status, which then shows the status of the connection, which is linked in writing to a boolean variable called "MOD STATUS". So in the event of an error, or if the test is stopped through the "STOP TEST" variable in the while loop, it is stopped.

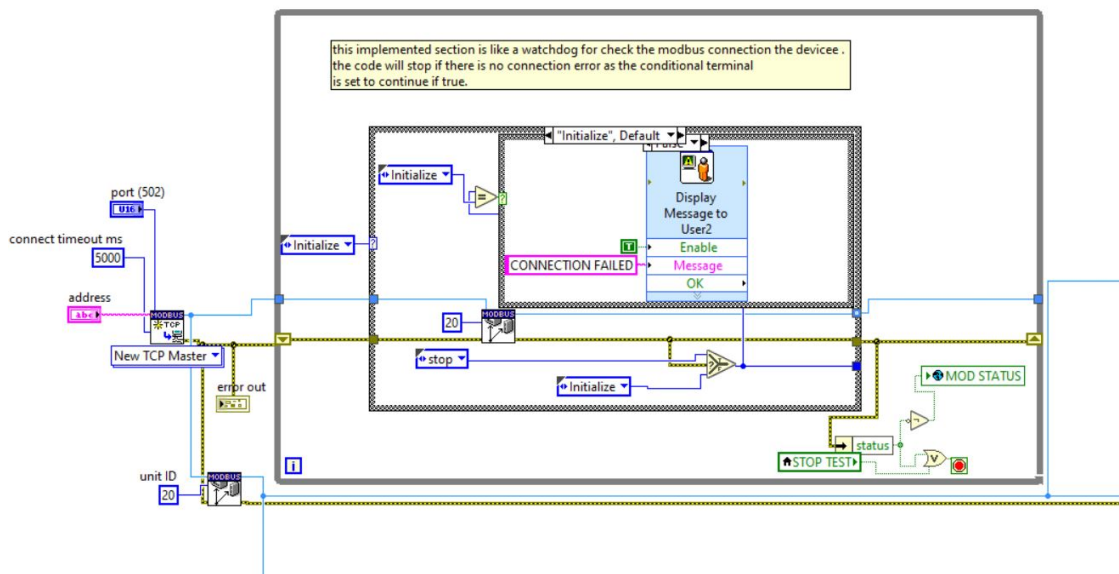
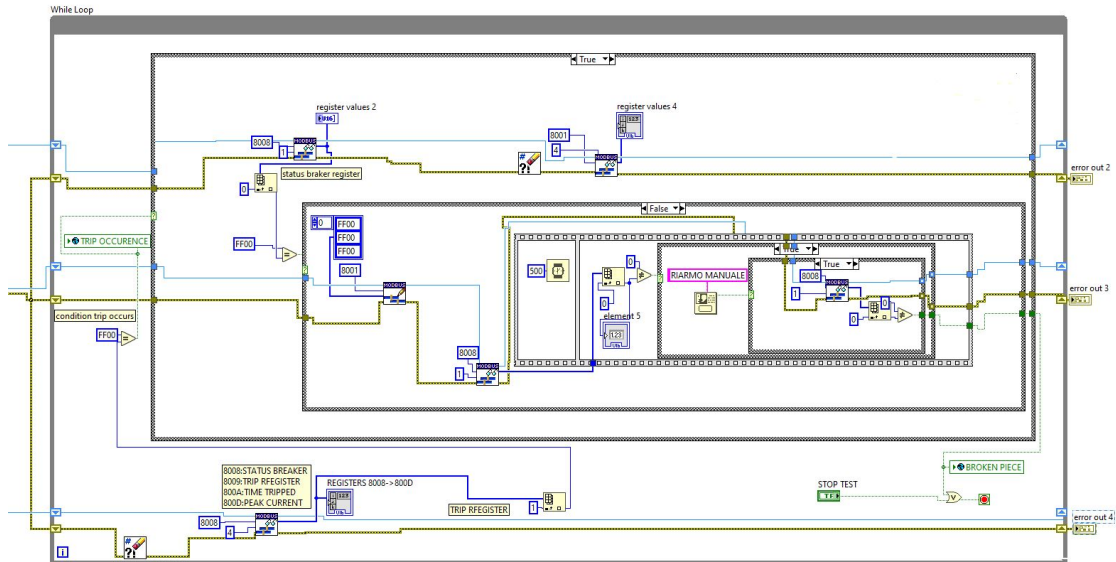


Figure 2.21: TCP master creation and communication check.

The remaining block of this subVI is the one that handles writing and reading to registers, Figure 2.22. Within a while loop, as a first step, the reading of four registers, knowing the Modbus map of the MOD, was implemented through the "Read Holding Registers" function of the Modbus API library, Figure 2.23:

- 8008:STATUS BREAKER
- 8009:TRIP REGISTER
- 800A:TIME TRIPPED
- 800D:PEAK CURRENT.

Register 8009 is particularly interesting because, if a trip occurs, the register will be written with the value 0xFF00, while if it does not occur we will have 0x0000.



**Figure 2.22:** Register reading and writing block.

Therefore, within this while loop, the value of this register is constantly read, and when the trip condition occurs, a case structure is set to the "true", within which, as a first step, the register 8008 “Status Breaker” is read and it will have the value 0xFF00 if the switch is closed, while 0x0000 if it is open. If it occurs that it is open, within a case structure, via the "Write Multiple Holding Register" function of the Modbus API library, Figure 2.24, the following registers will be written:

- 8001: Command breaker
- 8002: Enable/Disable remote input control
- 8003: Enable/Disable opening/closing command via communication.

In the specific case they are written with the value 0xFF00, thus commanding circuit-breaker closing, enabling remote control and enabling remote opening or closing.

Immediately afterwards, register 8008 is read again using the "Read Holding Register" function, to see if the circuit-breaker has actually been closed. Consequently to this, through a flat sequence structure made up of two blocks, which consists of sub-diagrams that are executed in sequence, it waits 500ms, and after that the content of the previously read register 8008 is taken: if this is different

from 0, therefore it has not received the closing command correctly and the user is asked, through a pop-up, to proceed with the manual reset.

Once the reset has been carried out, the value on the "Status Breaker" register will be read again; if this time the reset has taken place correctly the test continues without problems, therefore the whole cycle will be carried out again inside the while loop, while if the value on the register is still different from 0, it means that the piece has been damaged and therefore the test must be stopped.

This is signaled via the boolean variable "BROKEN PIECE" and the loop is stopped. The while loop is also stopped if the "STOP TEST" variable is true.

If, on the other hand, the reading of register 8008, before manual reset, shows that the closing command has been received correctly, and that therefore the "status breaker" value is 0x0000, the test will continue normally and the boolean variable "BROKEN PIECE" will be set to false".

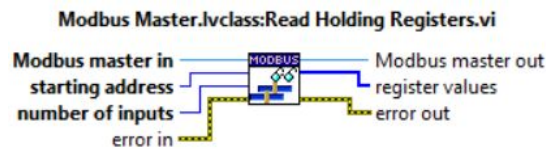


Figure 2.23: Reading Holding register function

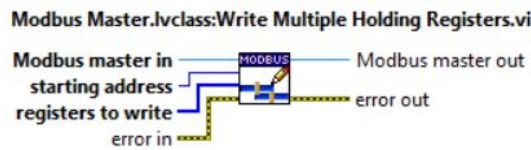


Figure 2.24: Write Multiple Holding registers function

### 2.0.3 Oscilloscope communication

Another device used was the oscilloscope, in order to measure voltage and current output from the machine and therefore the values applied to the circuit breaker during the test and also the trigger output signal from the machine.

Within a subVI called "REPORT GENERATION", which will be analyzed later, a section has been dedicated to communication with the oscilloscope, Figure 2.25.

For the oscilloscope used, a library supported by Labview was already foreseen, so it was not necessary to use other functions, except those written for the appropriate instrument.

The instrument was networked via an Ethernet cable, it was associated with an IP address and then through the "Initialize" function of the Sigilent SDS 1000 2000

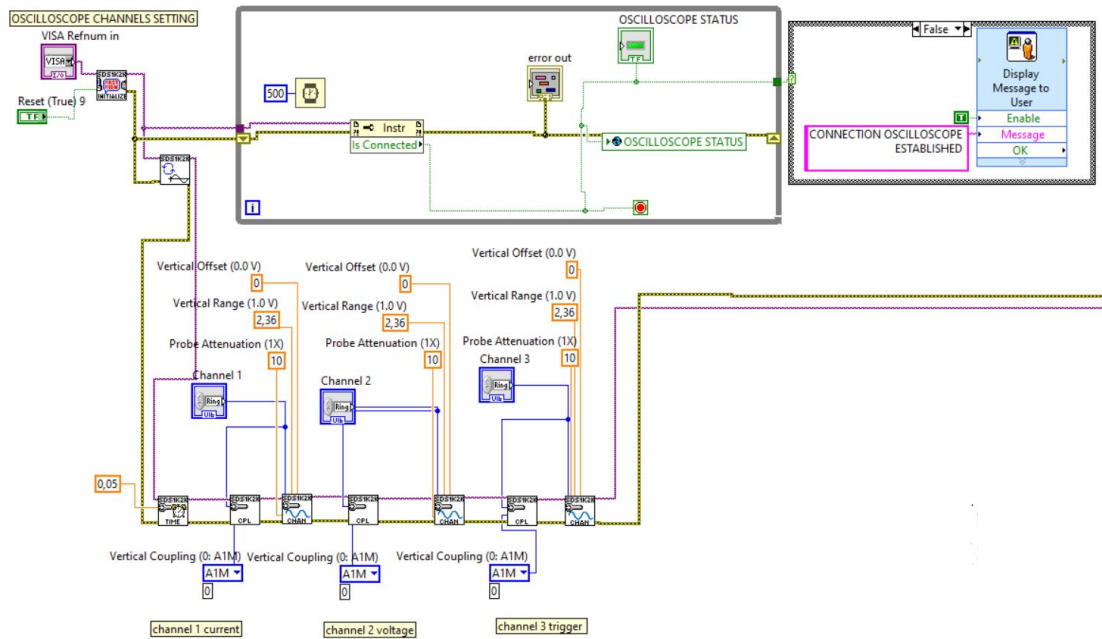


Figure 2.25: Oscilloscope communication block

series library, communication was established, Figure 2.26.

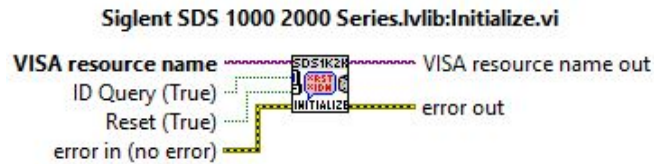


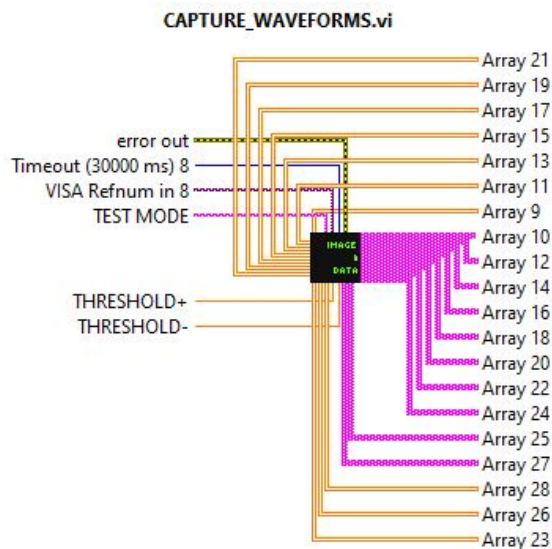
Figure 2.26: Initialize function

Subsequently it was necessary to set the various channels used, for which the "Configure Continuous Acquisition" function of the same library was used, so that the instrument runs continuously. After that the time-base and the settings on the horizontal axis were set through the appropriate "Configure time-base" function, then for each channel used the "Configure coupling" and "Configure channel" functions were used, through which the vertical coupling, vertical offset, vertical range and probe attenuation are set. There are three channels used, so this was repeated three times.

Furthermore, to verify that the communication is always active, a while loop has been inserted in which a property node, set to "is connected", outputs a Boolean variable connected both to the global variable "OSCILLOSCOPE STATUS" and to

the condition of stop of the while loop; so in case the connection fails the while loop will be stopped and it will be notified through the Boolean variable and a pop-up containing the message “CONNECTION OSCILLOSCOPE FAILED”.

In this thesis work it was required that, in addition to managing the test, the program created was able to generate a report file containing the voltage and current values recorded on the circuit-breaker when the surge impulse is given, and if check for an over-current or a trip the voltage, current and trigger waveform are also reported. To do this within the aforementioned subVI, another subVI called “CAPTURE WAVEFORMS.vi” was used, Figure 2.27, within which the waveforms on the three channels are read and, when the required conditions are met, the waveform is captured and then saved and inserted in the report.



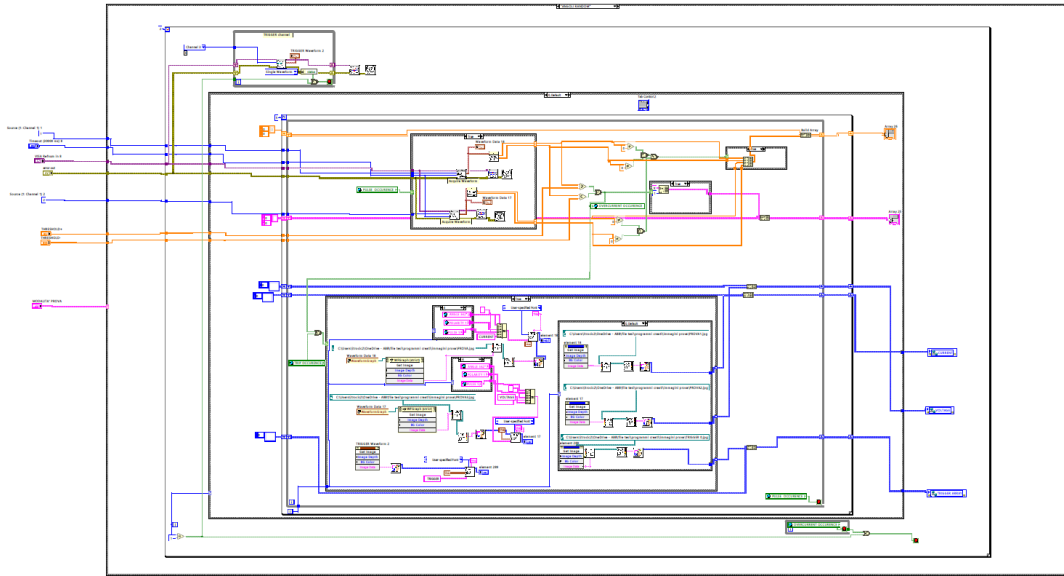
**Figure 2.27:** Capture waveforms subVI.

The block diagram of the subVI is reported in Figure 2.28 and then parsed.

As can be seen, the schematic is very large and unclear to see, so each block will be analyzed.

The subVI consists of a case structure which contains the two solutions relating to the test procedure:

- normal sequence: it foresees a number of impulses equal to five but eighth repetitions to perform will be present for each coupling, since the repetition of the angles will be from 0° to 270° first with negative phase and then with positive phase;
- random angles: it provides for a number of impulses equal to five but two repetitions to perform will be present for each coupling, since the angle will



**Figure 2.28:** Block diagram of the capture waveform.vi in case of normal sequence mode.

be varied with each impulse, so there will be 5 negative phase impulses with angles from  $-198^\circ$  to  $-342^\circ$ , and 5 positive phase with angles from  $18^\circ$  to  $162^\circ$ .

So when the user chooses the type of test to start, this will also be set as a control for this case structure.

#### 2.0.4 Normal sequence test procedure

Inside the normal sequence case it is found a for loop with index equal to eight, therefore eight repetitions, inside which it is found:

- A **while loop** for which the trigger signal is constantly read through the "Read Waveform (single).vi" function of the aforementioned library, Figure 2.28, having as input the oscilloscope address, the channel on which the user sets the trigger signal and the error coming from the oscilloscope communication block.

At the output terminal of this VI labeled TRIGGER WAVEFORM, the continuously acquired waveform is displayed, Figure 2.29.

If there should be an error in reading this signal, the while loop is blocked since the error output terminal is connected to the "unbundle by name" set to status, which will give a false Boolean output; the while loop is also stopped when the eighth iterations of the for loop finish.



Outside the while loop, the "Close.vi" function of the same library is present, which terminates the communication session when the while loop is terminated, Figure 2.31.

Having a while loop it is important that the oscilloscope address and the error are sent to a shift register in order to have no errors in reading the signal.

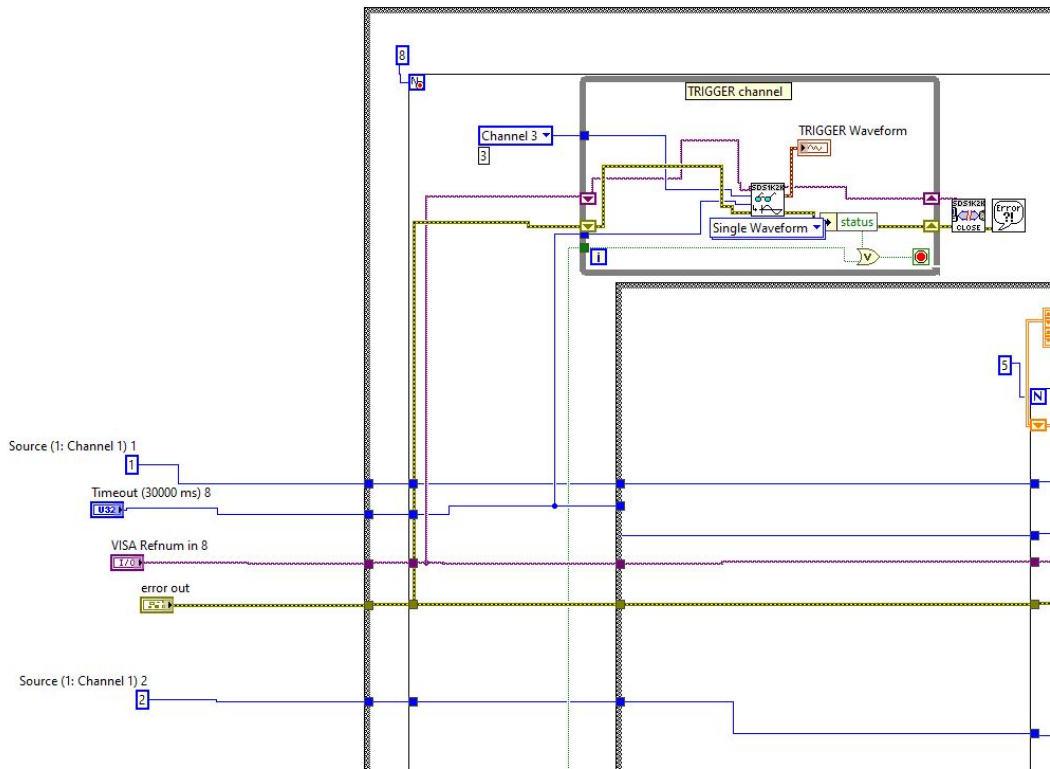


Figure 2.29: Trigger acquisition.

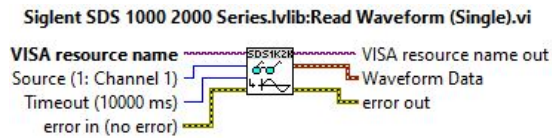


Figure 2.30: Read Waveform (single).vi.



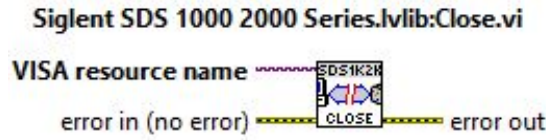


Figure 2.31: Close.vi.

- **Case structure** made up of eight cases, in which selector is the index of the interactions of the for loop in which it is contained. Inside each case it is found a for loop with an index equal to five, i.e. the five impulses which by law must be administered to the EUT, Figure 2.32 .

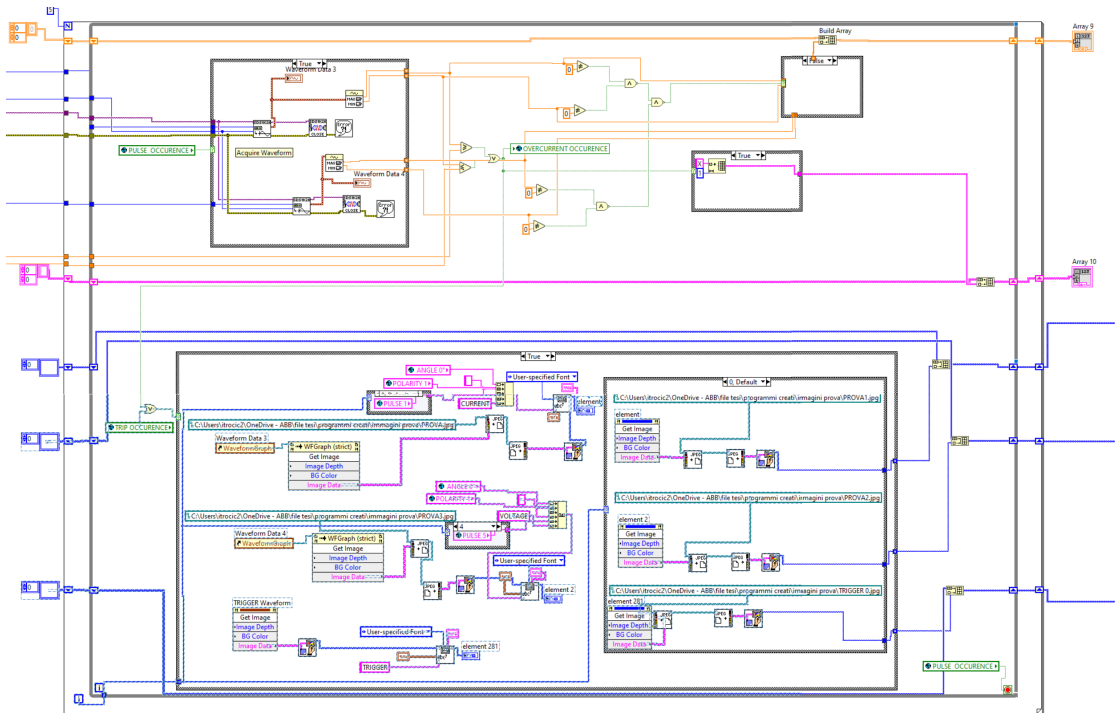


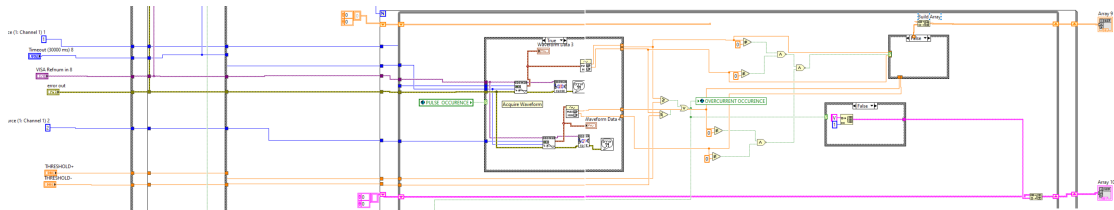
Figure 2.32: For loop for the five pulse on the differential switch.

All eight cases contain the same content, what changes is the angle and the polarity combination :

- Case 0: contains by five pulses having negative polarity and angle equal to  $0^\circ$ ;
- Case 1: contains by five pulses having negative polarity and angle equal to  $90^\circ$ ;

- Case 2: contains by five pulses having negative polarity and angle equal to  $180^\circ$ ;
- Case 3: contains by five pulses having negative polarity and angle equal to  $270^\circ$ ;
- Case 4: contains by five pulses having positive polarity and angle equal to  $0^\circ$ ;
- Case 5: contains by five pulses having positive polarity and angle equal to  $90^\circ$ ;
- Case 6: contains by five pulses having positive polarity and angle equal to  $180^\circ$ ;
- Case 7: contains by five pulses having positive polarity and angle equal to  $270^\circ$ .

Inside the for loop there is a while loop in which there is a case structure for which, if the selector, which is the global variable "pulse occurrence", is true therefore the surge impulse is given by the machine, then the current and voltage signals on the channels are read and their maximum and minimum values are taken and the waveform is displayed, Figure 2.33.



**Figure 2.33:** Minimum and maximum current and voltage evaluation.

This is done through the "Fetch waveform" function of the aforementioned library, Figure 2.34, and through the "Waveform Min and Max" function of Labview, Figure 2.35. Once it is done, the session is immediately closed with the "Close.vi" function.

Minimum and maximum values for both voltage and current have been taken into consideration since the polarity can be both positive and negative. Considering that it is important to save these values and then go and insert them in the final report file, then it was thought that externally to the case structure, these are inserted inside an array, if they are different from zero; this array will then be released at the end of the five iterations, so at the end of the eighth cases, there will be eighth arrays containing minimum and maximum voltage and current values. However, what is of fundamental importance is to

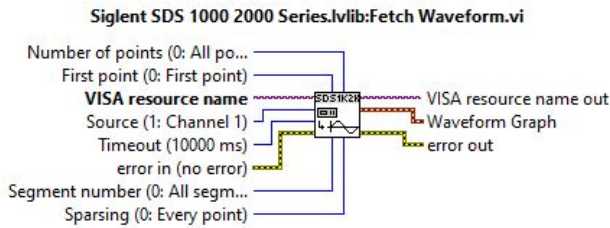


Figure 2.34: Fetch waveform function.

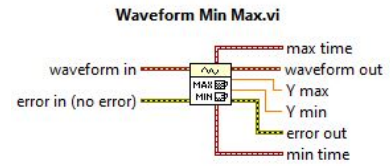


Figure 2.35: Waveform Min and MAX function.

determine if an over-voltage occurs, for which it is necessary to capture the waveform, stop the test to prevent the piece from being further damaged and finally mark the event in the table.

Therefore to do this, before starting the program, the user is asked to enter current threshold values for which the over-current is determined by comparing it to the value obtained from reading the signal; therefore the over-current event occurs either if the maximum current value is higher than the threshold or if the minimum current value is lower than the threshold. This is written in a global Boolean variable called "over-current occurrence".

The latter is used as a selector for two further case structures:

1. The first case structure composed of two cases for which, if the variable is false, it means that no over-current has occurred and therefore it will be marked in the positive result report "V", Figure 2.36, instead if the variable is true, then the over-current has occurred and will be marked as failed "X", Figure 2.37. This is done by inserting the outcome into an array released at the end of the five iterations. At the end of the eighth cases, eighth arrays containing the results to then be included in the report will be present;

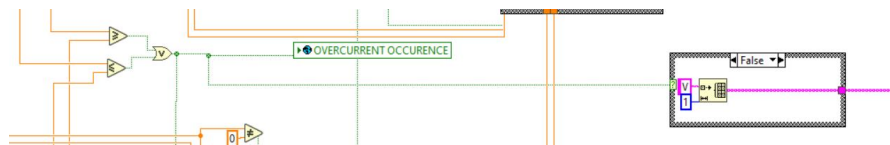


Figure 2.36: Positive result.

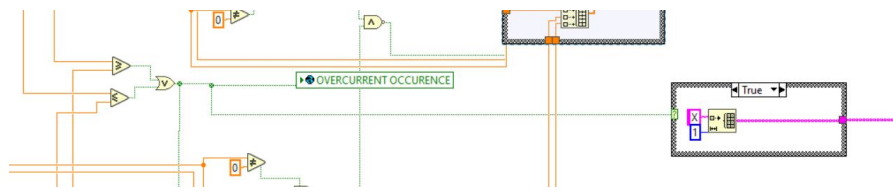


Figure 2.37: Negative result.

2. The second case structure is the one that takes care of capturing the waveforms in the event that either over-currents or a trip condition occur, Figure 2.38. Therefore the selector of the case structure will be the Boolean output of a logic or having as inputs the variables over-current and trip occurrence, if these occur.

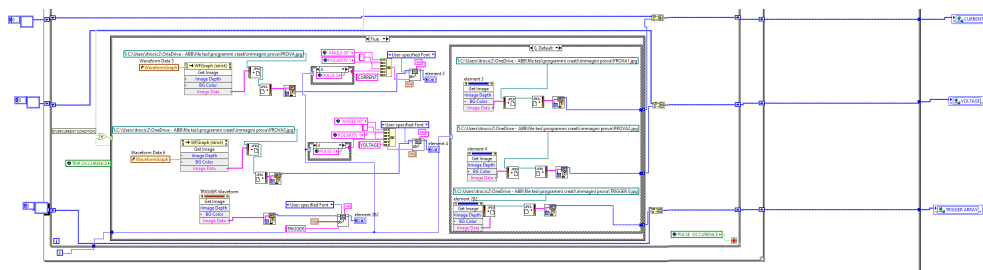


Figure 2.38: Waveform capture.

Within this case structure are located:

- As is possible to observe in Figure 2.39, three blocks composed of an "invoke node", which is in charge of invoking a method on a reference. In the specific case the method chosen is the "get image" to obtain the images of the waveform on the three channels, and this has the "data image" in output.

Once that is done, the image must be saved and it is also necessary to associate the test parameters at the time it was generated, for which the "Write JPEG file" function is used, to which the data of the image output from the invoke node and the path in which to save the image are given as input.

The output terminal of this function gives the path that is input to the "Read JPEG file" function, which generates the image data to input to the "Draw flattened bitmapx.vi" function in charge of Draws at 1-,4 -,8-bit pixmap or a 24-bit RGB pixmap into a picture.

At the output of this the new image it is obtained to which the test parameters are associated through the "Draw text into a rect.vi"

function: angle, polarity, current or voltage depending on the waveform considered and the impulse which varies because this part is repeated with the for loop with index five, equivalent to the five pulses. In the case of the trigger, however, only the "trigger" label is entered. At the end of this last function the new image is obtained.

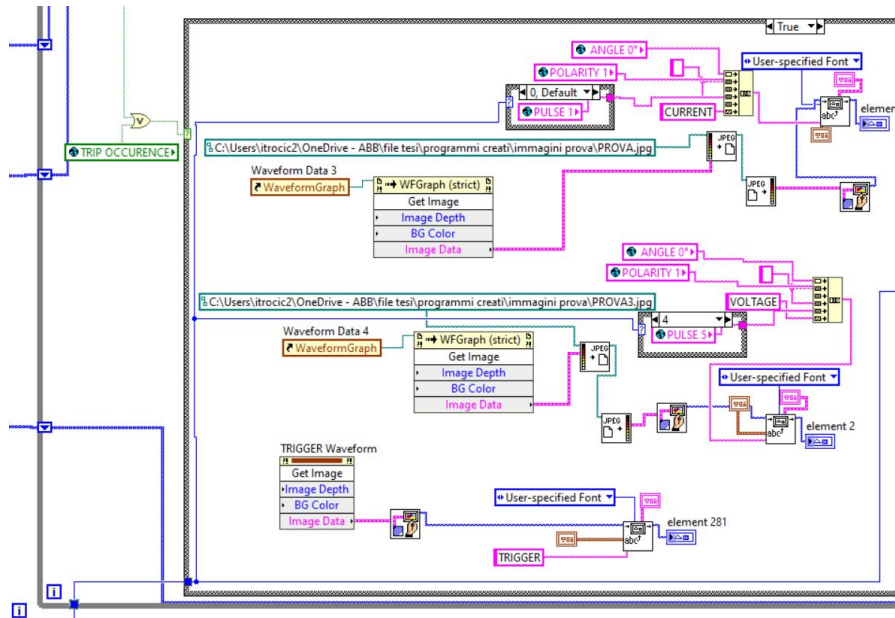


Figure 2.39: First part of waveform capture.

- It is also present another case structure which selector is the index of the for cycle with five iterations; every time the index is increased, the new image is saved with the mechanism implemented before, and then saved within an array released at the end of the eighth iterations of the upstream for loop.

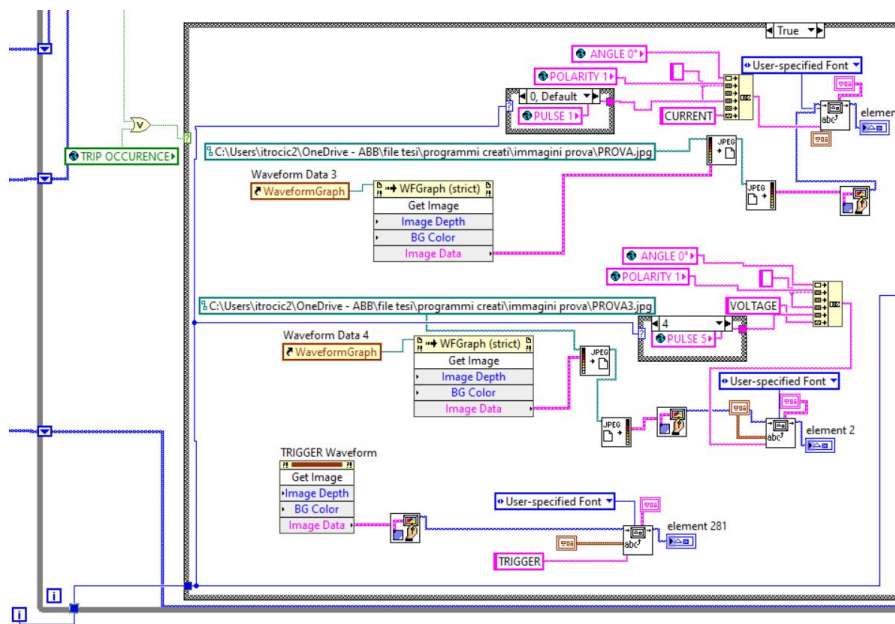


Figure 2.40: Second part of waveform capture.

As can be seen in the schematics shown, the test results, the minimum and maximum voltage and current values and the generated images are sent inside shift registers and then released in an array at the end of the cycles, in such a way to memorize and transfer at the end of the cycles in which they have been entered.

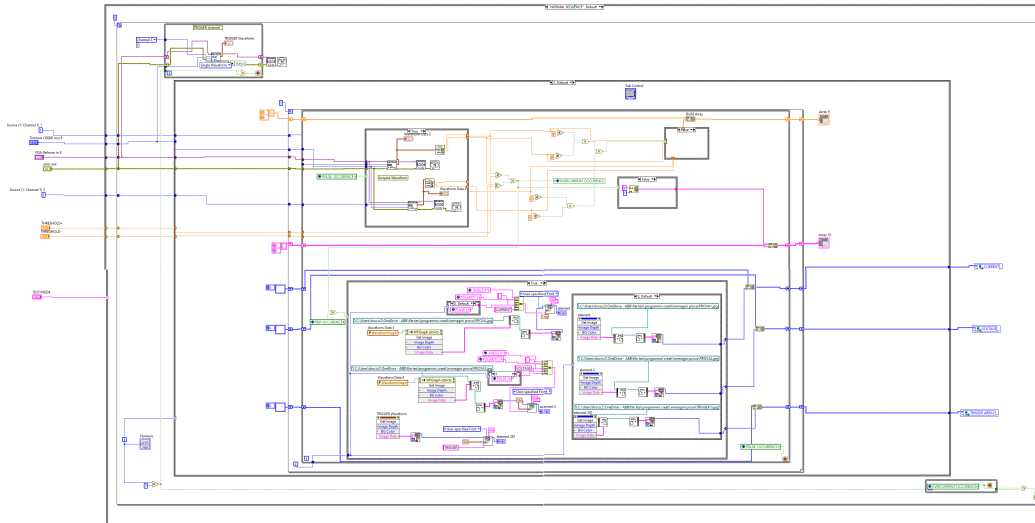
Furthermore, the eighth iteration for loop is also created with the "conditional terminal", meaning that while the for loop usually ends once the number of iterations for which it was created has finished, this time instead a further stop condition has been added, Figure 2.28.

In fact, it can be seen that inside the for loop there is a stop condition, like the one found in the while loop, and in this specific case it has been associated with the global Boolean variable "over-current occurrence" but also at the end of the iteration cycles; therefore, once the eighth cycles have finished or if the over-current condition occurs, the cycle is terminated. The global variable "over-current occurrence" in read mode is inserted in a while loop so that it is read constantly.

### 2.0.5 Random angle test procedure

If the test is performed in random angles mode, the process will be slightly different as there will no longer be the need to perform a for loop with eight iterations but only two, as in this mode there will be for each coupling, two cycles of five pulses, one with positive polarity and one with negative polarity, and each of the five

pulses will be applied at a different angle.



**Figure 2.41:** Block diagram of the capture waveform.vi in case of random angles mode.

There is therefore a for loop with index equal to two, Figure 2.41, also in this case with the "conditional terminal" associated with the over-current conditions and the end of the iterations, within which are found:

- A while loop for reading the trigger signal, exactly the same as the one used in normal sequencing mode;
- For each of the two cycles a Case Structure composed of two cases is presents, in which the selector is the index of the interactions of the for cycle in which it is contained.

Inside each case there is a for loop with an index equal to five, i.e. the five pulses which by law must be applied to the EUT. Both cases contain the same content, what changes is the angle inside each of the five pulses and the polarity that is affecting the test at that moment.

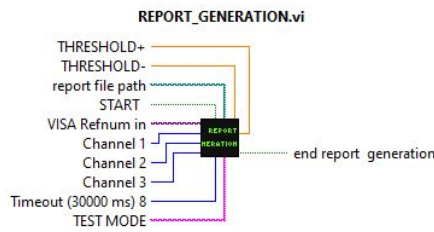
So we will have:

- case 0: negative polarity
  - \* Pulse 0: angle 198°
  - \* Pulse 1: angle 234°
  - \* Pulse 2: angle 270°
  - \* Pulse 3: angle 306°

- \* Pulse 4: angle 342°
- case 1: positive polarity
  - \* Pulse 0: angle 18°
  - \* Pulse 1: angle 54°
  - \* Pulse 2: angle 90°
  - \* Pulse 3: angle 126°
  - \* Pulse 4: angle 162°

## 2.0.6 Report generation

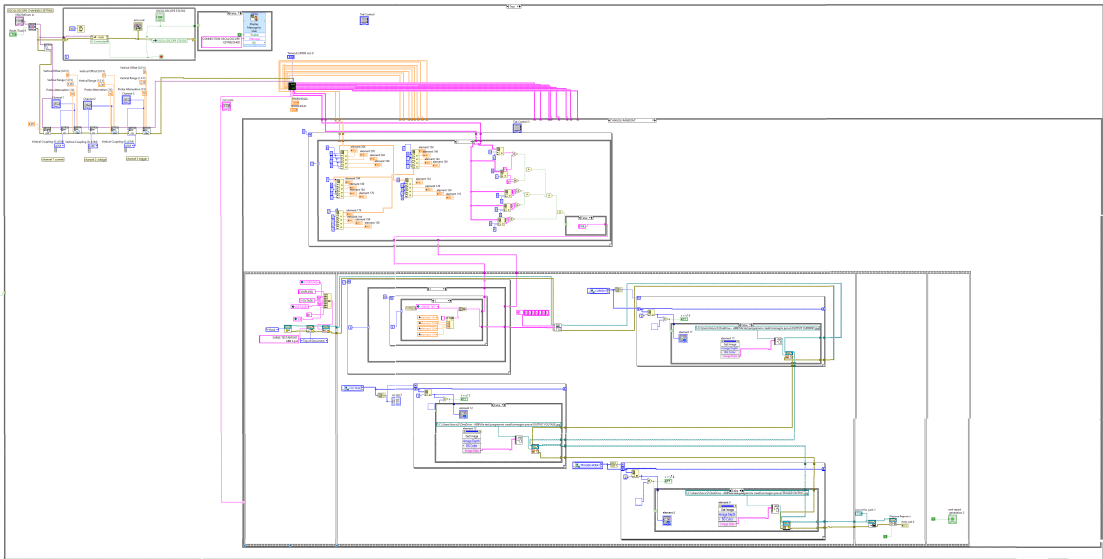
As previously mentioned, the section discussed above is part of another subVI called "REPORT GENERATION", Figure 2.42, which deals with the creation and compilation of the final test report, within which all the information relating to the test, necessary to get a overall view of how the test actually went is entered.



**Figure 2.42:** Report generation subVI.

In particular, the final test report will contain: a table within which all the voltage and current values applied to the RCCB are inserted to the release of the surge impulses, the images relating to the wave-forms of voltage and current in the cases in which over-current and trip conditions occur, and also the outcome of the test if positive or negative.





**Figure 2.43:** Block diagram of the report generation vi.

The block at the top, containing the part of the connection with the oscilloscope and the subVI "Capture Wave-forms" has already been analyzed; from now it is explained how the data obtained from the communication with the oscilloscope is entered within the report.

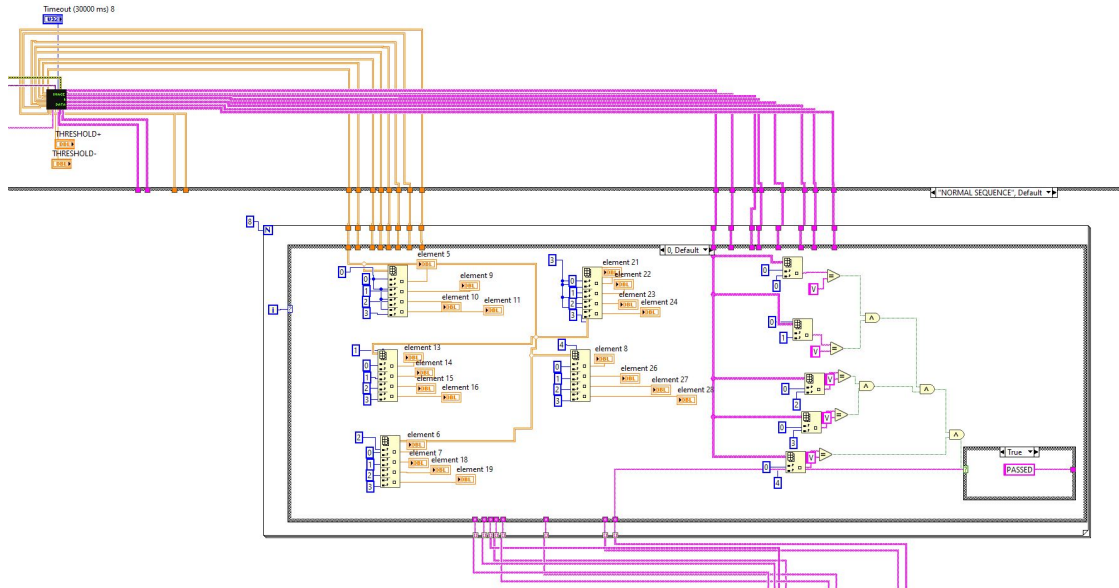
Also in this case, depending on the test mode chosen by the user, the creation of the report will vary:

- Normal sequence mode: inside a case structure, in the normal sequence case, there is a for cycle with an index equal to eight, and which index is the selector of another case structure. At each iteration the elements of the eight arms created in the previous part of the VI will be taken, containing the maximum and minimum values of current and voltage affecting the current iteration, through the "Index array" function to which, giving input the array of interest and the index of the elements of interest, this brings out the desired element out.

This will be repeated five times within one case, since there are five impulses for each coupling. The same will be done for the outcome of the test.

The array 9,11,13,15,17,19,21,23 are those containing minimum and maximum voltage and current values, while the array 10,12,14,16,8,20,24 are those containing the results test; if the results of the test, in the five case pulses, are all "V" then the test will be marked as "passed", otherwise if at least one of the elements is "X", then the test is considered "failed ". To make this type of evaluation, the elements extracted from the array are equalized to a string

containing the value "V" through the "Equal?" function which output is a True or False Boolean value, Figure 2.44.

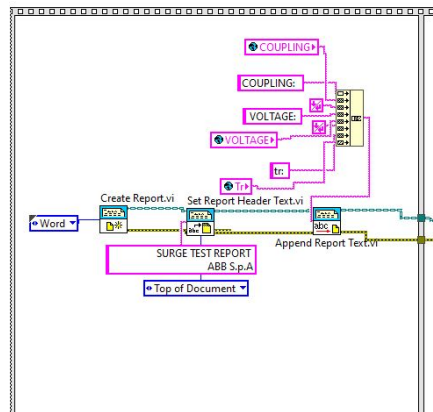


**Figure 2.44:** Elements extraction to be included in the table.

The results of this operation carried out on all the elements of the array relating to the case considered are put in a logical and, which output will enable a small case structure consisting of two cases: in the case "True" the test is marked as "passed" and a string containing the "passed" outcome is considered , if instead the outcome is "False" then the other case of the structure will contain the "Failed" string.

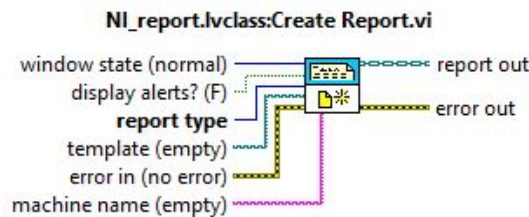
Subsequently there is a flat sequence structure, consisting of four segments:

- First segment within which the report is created as a Word document:



**Figure 2.45:** First segment flat sequence structure.

there is the "Create Report VI." of LabView, Figure 2.45, to which the type of document desired is in input, therefore in this case Word; as output there are the terminals that returns the reference of the report created and the error terminal, which are in inputs to the "Set report header Text.VI" function, Figure 2.47, through which the header "Surge test report ABB S.p.a " is set at the top of the document .



**Figure 2.46:** Create Report.vi.

Once again, the reference of the report and the error are present in output, and this time are set in input to the "Append Report Text.VI" Figure 2.48, by means of which some test information such as Coupling, Voltage and duration of Surge's impulse is entered inside the report . This information is taken by reading the global variables coupling, voltage and Tr and inserting them within a "concatenate string".

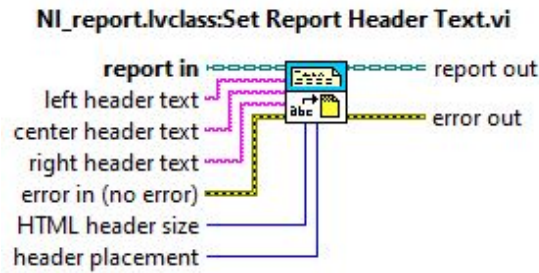


Figure 2.47: Set Report Header Text.vi.

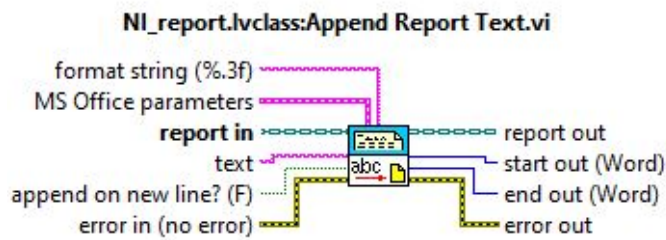


Figure 2.48: Append Report Text.vi.

- The second segment, Figure 2.49, on the other hand, contains the generation of the table within which voltage and current values are inserted, the outcome of the test and its completion. The table is created through "Word Easy Table.VI", Figure 2.50, to which the reference of the report created and also the error exiting the "Append Report Text.vi" are given in input. In addition, the names of the columns are set to the entrance terminal called "Column Headers" that will be:

1. ANGLE
2. POLARITY
3. I PULSE
4. II PULSE
5. III PULSE
6. IV PULSE
7. V PULSE
8. RESULT

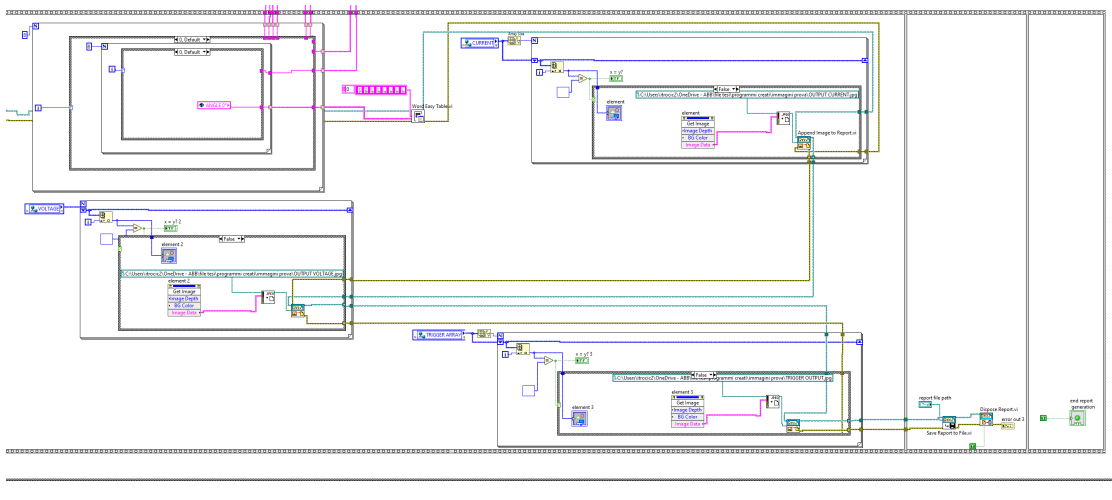


Figure 2.49: Second segment flat sequence structure.

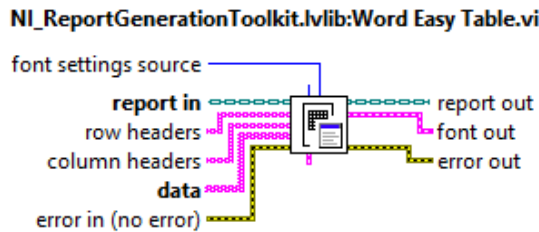


Figure 2.50: Word easy table vi.

The filling of the columns takes place through the "data" terminal. The input data is processed through a for loop with index equal to eight, within which it is present a case structure whose selector is the aforementioned index. Inside each of these eight cases contains a further for cycle with index equal to 8 in which it will be in the selector of another internal case structure which cases will contain:

- \* Case 0: angle
- \* Case 1: polarity
- \* Case 2: minimum and maximum value of voltage and current of the first impulse, these are entered by creating the local variables of the elements generated within the previously analyzed for loop, and entered within a "format into string" necessary because for enter this data in the table must be in string format;
- \* Case 3: minimum and maximum voltage and current value of the second pulse;

- \* Case 4: minimum and maximum voltage and current value of the third pulse;
- \* Case 5: minimum and maximum voltage and current value of the fourth pulse;
- \* Case 6: minimum and maximum voltage and current value of the fifth pulse;
- \* Case 7: outcome of the test which is the string selected by the case structure present in the previous for cycle.

In doing so, the table will contain all the test data relating to the five pulses that are given for all eight couplings.

Having done this, to insert the images relating to impulses in which an over-current or a trip occurred, three for loops were inserted with an index equal to the size of the "CURRENT", "VOLTAGE" and "TRIGGER" arrays obtained within the subVI "CAPTURE WAVE-FORMS", for which global variables were created and are now used in read mode.

Within these for loops through the "index array" function, to which the index of the for loop is given as the index of the element, all the elements of the array that contain an image, and therefore not a null element, will be saved again through the function "write JPEG file" previously seen and then inserted into the word file via the "append image to report.vi" to which the path of the image to be considered is given, Figure 2.51. If there are any null elements in the array, they will be discarded.

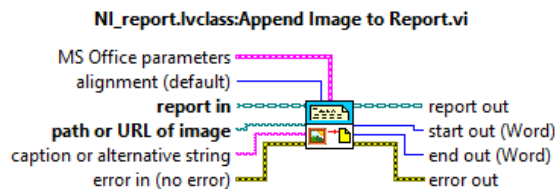


Figure 2.51: Append image to report vi.

- The third segment contains the part in which the report is saved through the "save report to file.vi", Figure 2.52, in which is entered the path where you want to save the file and then the "dispose report.vi", Figure 2.53, necessary to close the report and release its interface, which saves memory.

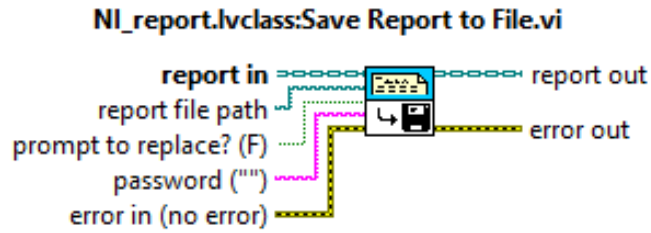


Figure 2.52: Save report to File vi.

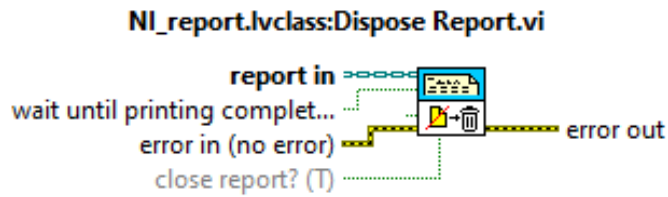
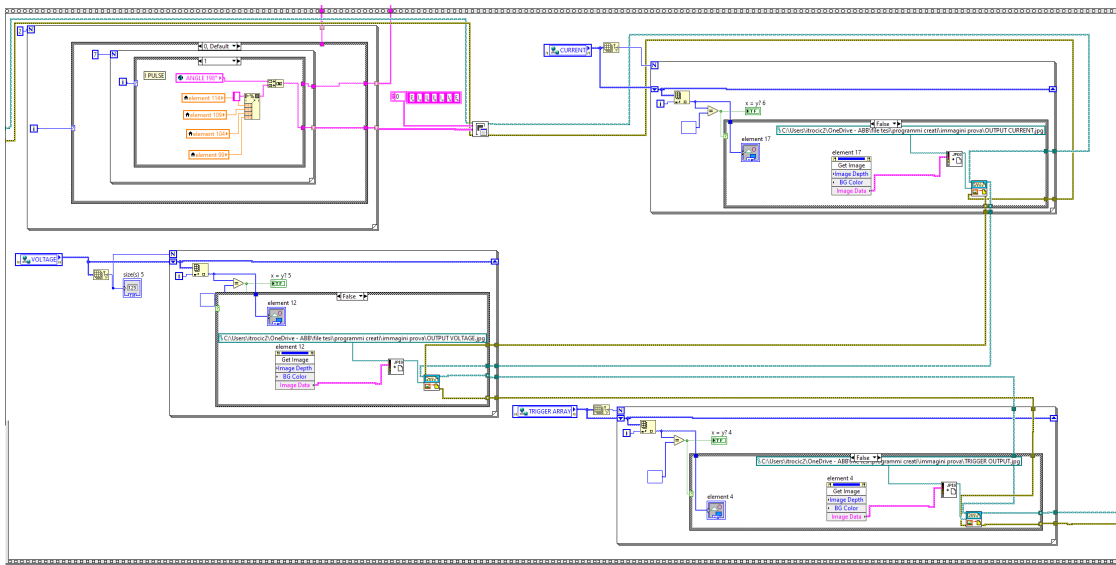


Figure 2.53: Dispose report vi.

- the fourth segment contains a Boolean variable set to true, which is used to notify the end of report generation, which also means the end of the program execution.
- Random corners mode:
 

in this case the structure is very similar to the one just seen for the normal sequence case, what changes are the number of iterations of the first for loop in which the elements are extracted from the arrays containing minimum and maximum voltage and current values and also the outcome of the test in the two couplings, which will be equal to two whereas before it was equal to eight. This will also have repercussions in the second segment of the flat sequence structure, Figure 2.54, which in turn will contain a for loop with index equal to two, in which the elements to be inserted in the columns of the table will be selected.



**Figure 2.54:** Second segment in case of random sequence.

The index of this for loop is also the selector of a case structure within which there is another for loop with index equal to seven which is in turn a selector of a further case structure.

The seven cases of this structure will contain in the first iteration:

- Case 0: negative polarity
- Case 1: minimum and maximum value of voltage and current of the first impulse, these are entered by creating the local variables of the elements generated within the previously analyzed for loop, and entered within a "format into string" necessary because to enter this data in the table must be in string format; these will be concatenated to the value of the angle relative to the impulse under consideration, therefore for the first impulse angle  $198^\circ$ .
- Case 2: minimum and maximum voltage and current value of the second pulse; linked to the value of the angle relative to the impulse under consideration,  $234^\circ$ .
- Case 3: minimum and maximum voltage and current value of the third pulse; linked to the value of the angle relative to the impulse under consideration  $270^\circ$ .
- Case 4: minimum and maximum voltage and current value of the fourth pulse; linked to the value of the angle relative to the impulse under consideration  $306^\circ$ .



- Case 5: minimum and maximum voltage and current value of the fifth pulse; linked to the value of the angle relative to the impulse under consideration 342°.
- Case 6: result of the test which is the string selected by the case structure present in the previous for loop

while in the second iteration:

- Case 0: positive polarity
- Case 1: minimum and maximum value of voltage and current of the first impulse, these are entered by creating the local variables of the elements generated within the previously analyzed for loop, and entered within a "format into string" necessary because for enter this data in the table must be in string format; these will be concatenated to the value of the angle relative to the impulse under consideration, therefore for the first impulse angle 18°.
- Case 2: minimum and maximum voltage and current value of the second pulse; linked to the value of the angle relative to the impulse under consideration, 54°.
- Case 3: minimum and maximum voltage and current value of the third pulse; linked to the value of the angle relative to the impulse under consideration 90°.
- Case 4: minimum and maximum voltage and current value of the fourth pulse; linked to the value of the angle relative to the impulse under consideration 126°.
- Case 5: minimum and maximum voltage and current value of the fifth pulse; linked to the value of the angle relative to the impulse under consideration 162°.
- Case 6: result of the test which is the string selected by the case structure present in the previous for loop

The rest of the structure will remain the same.

# Chapter 3

# MAIN PROGRAM DEVELOPMENT

## 3.0.1 Development

In this chapter, it will be shown how the VIs analyzed in the previous chapter were used to create the main program that takes care of the entire test management.

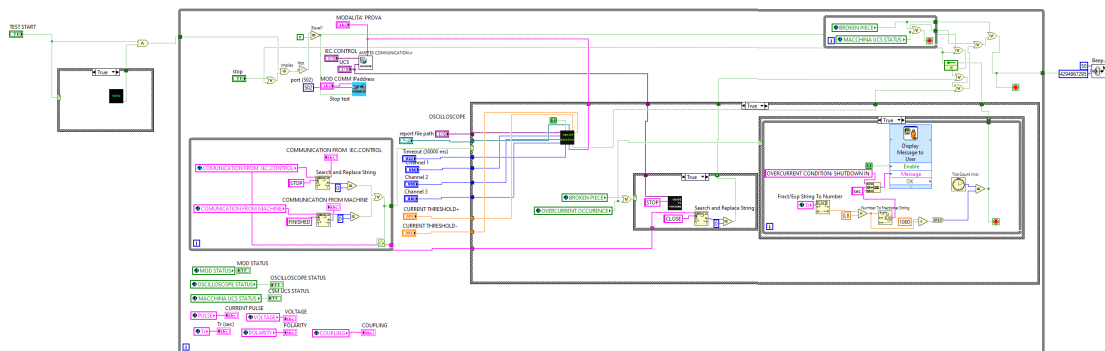


Figure 3.1: Main Program block diagram.

As can be observed in the Figure 3.1, the entire block diagram is carried out within a while loop, for which stop conditions will be shown in the course of the description and by a small case structure which has as its selector the start signal sent by the user ; when start is given, the true case of the structure is selected, for which the creation of the com ports that are needed for communication is performed through the execution of the VSPD VI. Once these ports are correctly created, the Boolean variable set to true and put in "logical or" with the start signal, start the structure within the while loop. On the other hand, if the start signal is not given

by the user, the case structure will be on the "false" case for which no com ports are created and therefore the program cannot be executed.

Inside is the first block, Figure 3.2, in the upper left corner, which contains the Boolean start and stop variables of the test that will be selected by the user to start or if necessary to terminate the test before the end of the internal program cycle. These two are also the inputs of the logical function "implies", which negates the first input and performs the logical or with the second input; in this case it will negate the start signal and perform the logical or with the stop signal. The negated output of this function, which coincides with the condition for which only the start command is active, enables communication with the MOD-COMM module.

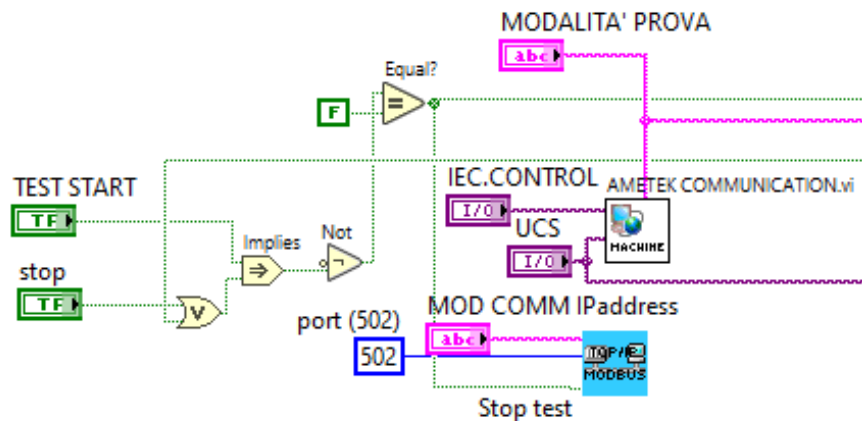


Figure 3.2: First block of main program block diagram.

As shown in the Figures 3.2, this Boolean variable will be linked to the stop test input in the "modbus communication" subVI, since in case it is true it will enable communication, while if false it will go to stop communication with the mod and stop the test in general .

In addition, the user will be asked to enter the IP address of the MOD-COMM block to communicate with.

At the same time, communication with the machine is also started through the subVI "AMETEK COMMUNICATION.vi" seen in the Figure 2.1, for which the user is asked to enter the COM ports related to communication from the Iec.control software and the machine part; the user is also asked to select the mode of the test between "normal sequence" and "random corners."

In another while loop internal to the previous one, Figure 3.3, the communication received from the Iec.control software and the cts machine is constantly read via the global variables "communication from Iec.control" and "communication from

machines" , which are written within the subVI "AMETEK COMMUNICATION".

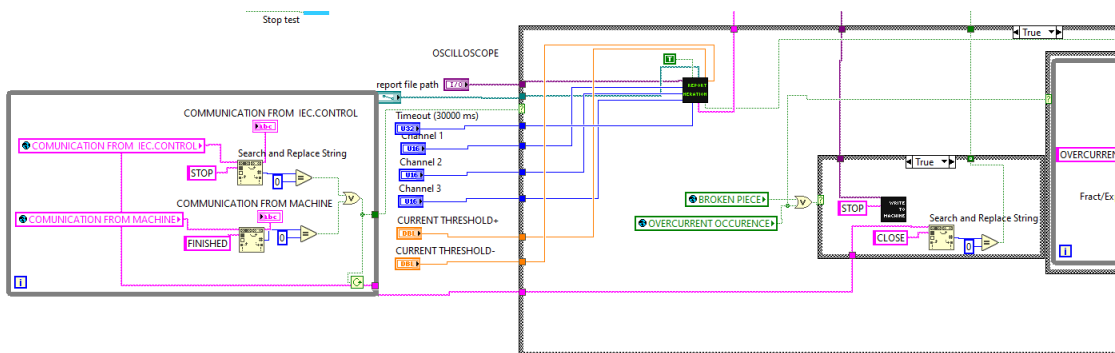


Figure 3.3: Second block of main program block diagram.

By reading the communication from the software through the "Search and Replace String" function, Figure 3.4, the string "STOP" is searched, while from the communication from the machine the string "FINISHED" is searched; if either or both of these should be present the function will output the value 1, which means that the test is ended or has been interrupted, so it will go to interrupt the communication with the oscilloscope and in case the test has not been ended but rather interrupted, it will be sent to the machine in command to close the communication, which we will see later.

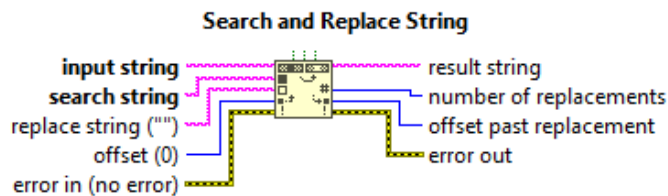


Figure 3.4: Search and replace string function.

The outcome of the search for the two conditions is sent as input to a logic or, the output of which is the selector of a case structure, internal to the main while loop, which enables or disables communication with the oscilloscope and creation of the report.

If the switch is "true," the two conditions have not occurred, and communication with the surge generator is active, then the "report generation" subVI seen in Figure 2.40 will be enabled. For this, the user is asked to enter the address of the oscilloscope, the path on which they want to go to save the report that will be generated, the three channels of the oscilloscope on which to go to set current, voltage and trigger, the timeout, that is the specified acquisition period, which by

default is set to 30ms but could be varied and finally the positive and negative current threshold values, values that vary depending on the differential switch considered. Also linked to this subVI is the selected test mode, since as seen in the section, the user's choice goes to enable different sections of the subVI, and thus affects program execution.

Enabling subVI is achieved by a Boolean variable set as true on the terminal.

As the output terminal of the subVI from the "end report generation", Figure 2.40, we get a Boolean variable which, if true, indicates that the report has been terminated and thus represents one of the conditions indicating the termination of the process for which program execution can be stopped; if it is false, the program continues its normal execution.

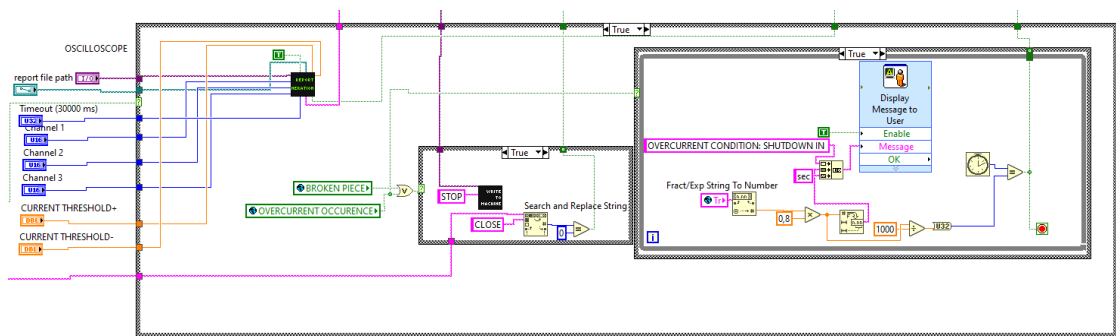


Figure 3.5: Main case structure of the block diagram, true condition.

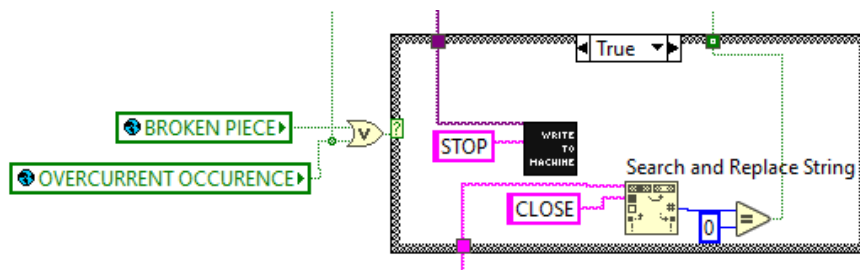
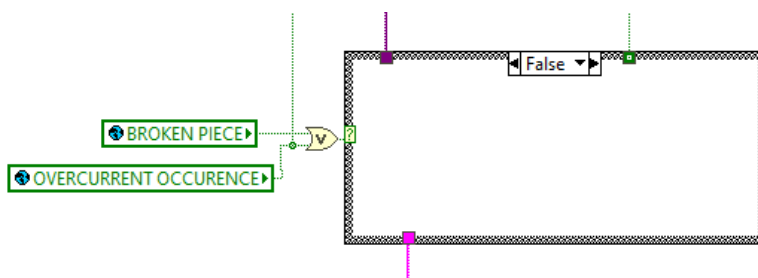


Figure 3.6: Sub-case structure to command the machine, true condition.

At the same time, within the same case structure, the global Boolean variables "broken piece" and "over-current occurrence" are read constantly and sent as input to a logical or, the output of which is the selector of an additional case structure: if either or both variables are true then the selected case will be the "true" for which the stop condition is sent to the cts machine , via the "write to machine" subVI, seen in the figure, to which the com port related to the communication coming from the machine and the "stop" command is given as input, Figure 3.6.

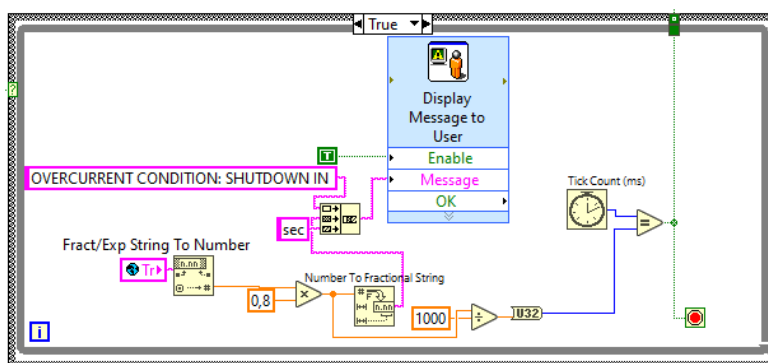
At the same time, the communication coming from the iec.control software is read to verify that the command has been received, so what is done is to search the string "close"; if present the output of the "Search and Replace String" function will have value 1 so this is an additional condition to stop the program execution, but if it is not present the output will be 0 and the program will continue execution.

In the case where both the Boolean global variables "broken piece" and "over-current occurrence" are false, then the "false" case of the case structure is enabled, for which no command is sent to the machine.



**Figure 3.7:** Sub-case structure to command the machine, false condition.

The global variable "over-current occurrence" is also selector of an additional case structure, Figure 3.5-3.6: in case it is true the user is notified that this condition has occurred and that in a time interval equal to 80% of the time value set as "Tr" the test will be stopped. Then, after that time interval has elapsed, the while loop is stopped and the condition is again sent as a stop condition for the entire program.



**Figure 3.8:** Sub-case structure to user notification, true condition.

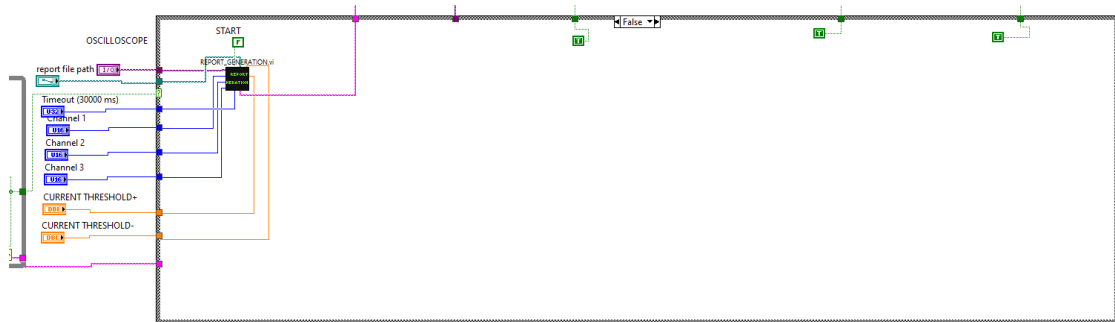
To determine the time interval, the function "Fract/Exp String To Number" was used, which is in charge of reading the numerical value from the string contained in the Boolean variable "Tr" written in the subVI " AMETEK COMMUNICATION"

and returns this value as a number; this is then multiplied by 0.8 and divided by 1000, as a "tick count" will be used, which returns the value in milliseconds, in order to evaluate the elapsed time and stop the test when the desired value is reached.

Then, the moment the counter has reached the required value the while loop, which is contained within this specific case structure for notification of the "over-current" condition, is stopped and the Boolean stop condition is once again sent to terminate execution.

In the case where instead the global Boolean variable "over-current occurrence" is false then the program continues execution.

If the "Search and Replace String" function verifies one of its "stop " or "finished" conditions on communications with the software and the machine, then the selector will go to false, and in this case the "report generation" subVI will be sent a Boolean variable set as false on the "Start" terminal, which will therefore not enable the subVI, and all stop conditions coming from this case structure will be set as true.



**Figure 3.9:** Main case structure of the block diagram, false condition.

Additional stop conditions are given to the global Boolean variables "BROKEN PIECE" and "MACHINE UCS STATUS" which are read constantly within an additional while loop, contained within the main one, and linked to a logical or in which if one and both variables are true it stops the execution of the while loop and sends out the condition that interrupts program execution, Figure 3.10 .

All of the mentioned stop conditions are concatenated with each other through logical ORs, so that if even one of them is encountered then the program and test execution are immediately aborted.

In addition, within the main while loop, in order to have continuous monitoring of the test progress, the following global variables are also read continuously:

- MOD STATUS
- OSCILLOSCOPE STATUS
- MACHINE UCS STATUS

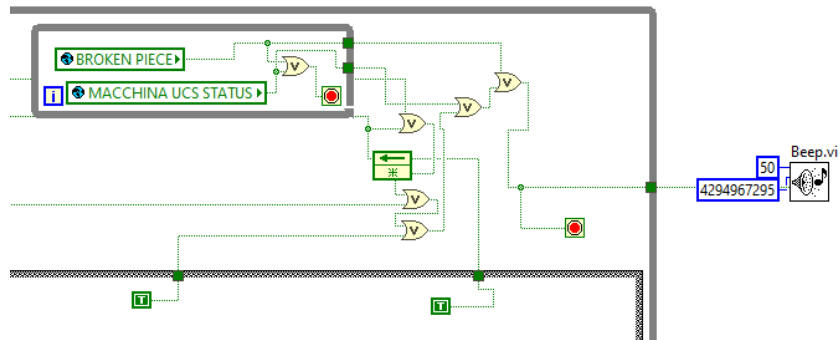


Figure 3.10: Stop conditions.

- PULSE
- VOLTAGE
- Tr
- POLARITY
- COUPLING.

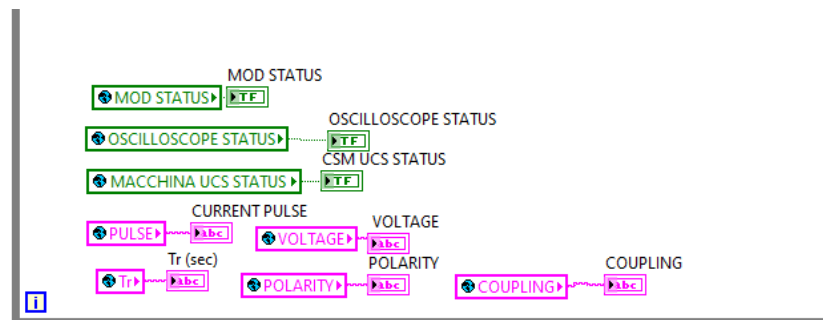


Figure 3.11: Reading of global variables to monitoring the test.

### 3.0.2 User Interface

This main VI, through its front panel, also forms the user interface, as can be seen in the Figure 3.12.

The Boolean variables START and STOP were inserted as two large buttons so that the user is immediately clear what to press to start or stop the test; also for



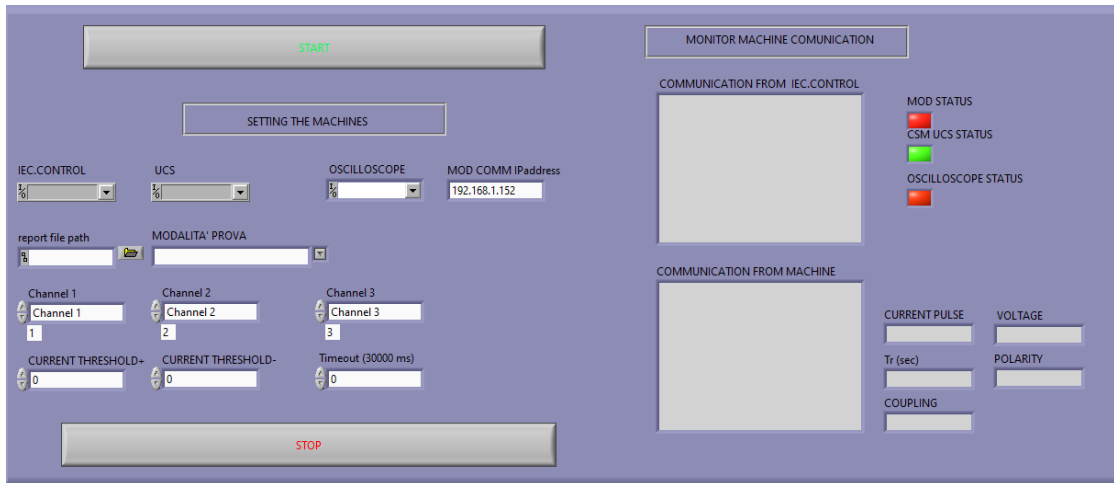


Figure 3.12: User interface.

communication with the oscilloscope, a drop-down menu was used in which the devices in the network that can be connected to are shown; the user to be sure of the device to be considered can check it through the NI MAX application, in which section "Device and interfaces" , under "network devices" shows the list of devices.

As for the IP address related to the communication module, the user will be able to enter it manually. As far as the test mode is concerned, a drop-down menu has once again been included in which there are the two possible cases to be selected, while for the choice of the path in which to save the report file, the possibility has been given to navigate within the folders and search for the desired destination. The oscilloscope channels are set by default but the user has the option of changing them via "text rings"; the same is used to set the threshold current values and timeout, but the values can also be written manually by the user.

The right side of the interface contains all the monitoring of communication with the devices, so you can see two boxes named " COMMUNICATION FROM IEC.CONTROL" and "COMMUNICATION FROM MACHINE" which show all the communication strings exchanged and then laterally we have the mod, machine and oscilloscope status indicators. Also present are the current values of pulse, voltage, Tr, polarity and coupling. This is done to get an instant view of the progress of the test and any problems in communication with the devices.

When program execution is finished, an audible signal is emitted.

## Chapter 4

# CONCLUSION AND POSSIBLE IMPROVEMENTS

The main purpose of this thesis work was to automate the Surge test process, carried out on differential circuit breakers by replacing the presence of an operator with a motor control, designed by the same company, implementing through appropriate control logic.

In this regard, Labview was chosen as the graphical programming environment to work on.

The project was carried out following well-defined steps; first, the choice of the HW/SW components to be used was carried forward: the AMETEK CTS machine, already present in the laboratory, was chosen as the instrument for carrying out the test of Surge 1.2/50  $\mu$ s, a compact multi-function generator that simulates the effects of electromagnetic interference, which provides several communication interfaces such as USB type A and B , Opto link and Ethernet, a factor of interest since it gave several possibilities for having the machine in the laboratory network.

As mentioned earlier for the automation of the test, it was necessary to replace the current operator action with a motor control, for which the choice fell on the new MOD device under development in the company which provides for coupling with the COMM WIFI communication module, also under development.

In addition, it was necessary to use an oscilloscope that during the test would measure current and voltage output from the machine, then administered to the switch, so that voltage and current measurements could be made with each surge pulse applied to the switch. A 4-channel oscilloscope with 200 MHz bandwidth already supported by national instrument was chosen for this purpose, which allowed for easy integration on Labview since the library was already supported.

Such an oscilloscope was then networked via Ethernet cable and in doing so was given an IP address, through which communication was established for the acquisition of readings made during the test; it was provided that for each test case, as soon as there is a machine trigger and a surge pulse is released from the machine, voltage and current measurements are made by the oscilloscope and are saved in case it has either a trip condition or the current on the switch exceeds the threshold set by the user.

After addressing this aspect, the communication part of the machine was developed; in this regard, communication via com port was chosen, for which it was necessary to find a solution that would allow multiple software to access the physical port simultaneously. Such a solution was found in the use of software that allows the creation of virtual ports to which traffic is redirected.

At this point, these created virtual ports were accessed by two different pieces of software: Labview and the machine management program; the latter specifically serves precisely to correctly set up the machine for carrying out the test. As far as Labview is concerned, it was used in read by making sure that all the data traffic exchanged between the machine and its software could be read correctly, but also in write since when necessary the created program must be able to send commands not passing through the main software of the machine. The structure given on the machine's command list was used for such commands.

The Modbus map of commands to be given to the Wifi communication module of the motor control was then validated, whereby it was ascertained that by sending certain paradigms with the structure provided by the Modbus protocol, indeed the MOD associated with the differential switch actually executes the desired commands. This communication was then implemented on Labview through the appropriate Modbus library, such that when the trip condition occurs the reset occurs and if the reset occurs correctly then the test is continued, otherwise it is stopped.

Having gone through all these steps, the process of creating a report file was then created in which the test parameters for each case history, the voltage and current values for each surge pulse, and any voltage and current waveforms in cases where a trip or over-current occurs. The report created was generated as a word file, for ease of management and so that it was easily editable compared to a PDF file for example. All these parts carry with them possible conditions for stopping the program, so it has been predicted that the program will be stopped in the following cases:

- If the user gives the stop command
- If communication with the machine fails
- If as a result of a trip the reset is not performed correctly

- In which an over-current occurs and after 80% of the pause time between two different pulses the operator does not take action
- At the end of the test during normal operation

The program created within it contains two case histories, normal sequence execution or execution by random angles; one of these two test modes is chosen by the user before startup and involves different data and image collection and related tabulation: in the case of normal execution for each coupling (L-N, L-PE, N-PE) five pulses are released for each angle starting from 0° arriving at 270°, with 90° increment, repeating it for both positive and negative polarity; in the case of random angles instead for each coupling, considering positive and negative polarity, for each of the 5 pulses the angle will be incremented by 36°, starting from -198° to arrive at +162°. In the first case forty measurements will be made for each coupling, while in the second case only ten.

Finally, the user interface was made, in which the buttons from which the user can command start and stop were included; a section in which the device settings are given for:

- Three drop-down menus from which to select the two com ports related to machine and software communication and the oscilloscope address;
- A string control, in which to manually enter the IP address of the motor control communication module;
- A string & path from which to select the path in which to save the report;
- A drop-down menu from which the user can select the mode of the test by choosing between normal sequence and random angles;
- Six text rings in which the oscilloscope channels to be used, the positive and negative threshold currents, and the timeout representing the acquisition period are selected.

There is also a section where program and test status can be monitored; in fact, two displays appear showing all the communication strings exchanged between the machine and the software, three LEDs indicating the status of active or failed communication with the Ametek CTS machine, oscilloscope and motor control. In addition, the current pulse, applied voltage, by pulse duration, current polarity, and current coupling are also shown; this will allow the user to know the status of the test and devices used clearly and take action if necessary. At the end of the test, for any of the case histories seen above, the program emits an acoustic signal.

Compared with what has been done so far, it is certainly possible to make some improvements to the created program, which will now be explained.

At this time the program can handle surge tests involving only five pulses per case, in case a different or larger number of pulses were set from the machine's Iec.control software, the program would no longer be able to record the information from them.

In this regard, a possible improvement to the program could involve handling any number of pulses, the numerical value of which would be entered manually by the user, and which would then affect the for cycles that were analyzed within the Capture wave-forms subVI. This change would also imply a varying number of arrays created, which would affect the reading portion of the array elements that was addressed within Report generation subVI.

As seen earlier, reading at this time is done by making explicit the reference of the element considered within the array, if we were to introduce the change in question, surely the read handling implemented in this way would no longer work well, as we would have a variable number of arrays and therefore the reference of all arrays would not be known a priori.

Such a solution was not considered at this stage of the project, as the work was based on what is the current scenario of the tests conducted, and such a change would have entailed an overall revisiting of the program at a late stage of the thesis project.

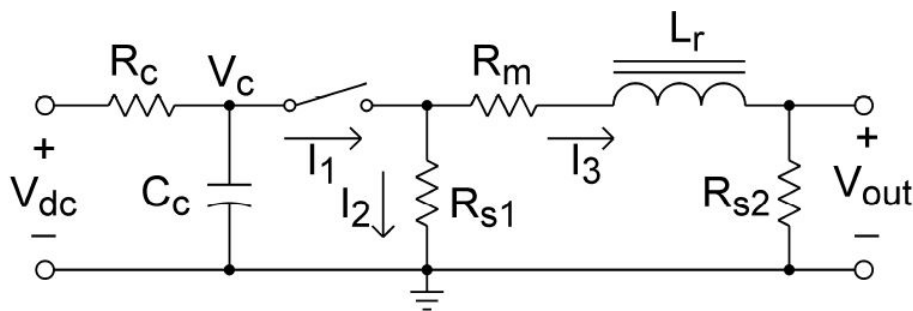
Also, in this first version of the project, the report was generated on a word file, for ease of management through Labview, and because a file that could still be edited once saved was needed. In the future, a report generated on Excel could also be thought of, for which the management of the data tables created would have to be reviewed, which is not easy, or a PDF file in which, however, would not be editable once created, except by going to convert it when needed.

# Appendix A

## Derivation of components of surge generators

This chapter reviews the circuit components of surge generators not mentioned in the standard, in the specific case of  $1.2/50 \mu\text{s}$  voltage generator [1].

Consider the following generator:



**Figure A.1:**  $1.2/50 \mu\text{s}$  combination wave surge generator

The considered generator generates a surge that has:

1. an open-circuit voltage front time of  $1.2 \mu\text{s}$ ;
2. an open-circuit voltage duration of  $50 \mu\text{s}$ ;
3. short-circuit current front time of  $8 \mu\text{s}$ ;
4. a short-circuit current duration of  $20 \mu\text{s}$ ;

The ratio of the peak open-circuit output voltage to the peak short-circuit current defines the effective output impedance, which for such a generator is  $2 \Omega$ . It is

possible to derive the Laplace equations for the open-circuit voltage considering that at  $t=0$  the switch is closed, the capacitance  $C_c$  is charged  $V_{dc}$ , and the initial inductor current is 0;  $R_c$  considered negligible.

$$I_1(s) = C_c(-sVc(s) + Vdc) \quad (A.1)$$

$$I_2(s) = \frac{V_c(s)}{R_{s1}} \quad (A.2)$$

$$I_3(s) = \frac{V_c(s) - L_r s I_3(s)}{R_m + R_{s2}} \implies I_3(s) = \frac{V_c(s)}{R_m + R_{s2} + l_r s} \quad (A.3)$$

KCL equation:

$$I_1(s) + I_2(s) + I_3(s) = 0 \quad (A.4)$$

$$C_c(-sVc(s) + Vdc) - \left(\frac{V_c(s)}{R_{s1}}\right) - \left(\frac{V_c(s)}{R_m + R_{s2} + l_r s}\right) = 0 \quad (A.5)$$

from which derives:

$$V_c(s) = \frac{V_{dc} R_{s1} C_c (R_m + R_{s2} + l_r s)}{s^2 R_{s1} L_r C_c + [C_c R_{s1} (R_{s2} + R_m) + L_r] s + R_{s1} + R_{s2} + R_m} \quad (A.6)$$

Equation for open-circuit output voltage :

$$V_{OC}(s) = I_3(s) R_{s2} \quad (A.7)$$

hence:

$$V_{OC}(s) = \frac{V_{dc} R_{s1} R_{s2} C_c (R_m + R_{s2} + l_r s)}{(R_m + R_{s2} + l_r s) [s^2 R_{s1} L_r C_c + [C_c R_{s1} (R_{s2} + R_m) + L_r] s + R_{s1} + R_{s2} + R_m]} \quad (A.8)$$

from which we derive the equation in the time domain of  $V_{out}$ :

$$V_{OC} = \frac{2V_{dc} R_{s1} R_{s2} C_c \sinh\left[\frac{\sqrt{R_{s1}^2 (R_{s2} + R_m)^2 C_c^2 - [2R_{s1} (R_{s2} + R_m) + 4R_{s1}^2] L_r C_c + L_r^2}}{2R_{s1} L_r C_c} t\right]}{\sqrt{R_{s1}^2 (R_{s2} + R_m)^2 C_c^2 - [2R_{s1} (R_{s2} + R_m) + 4R_{s1}^2] L_r C_c + L_r^2}} e^{\frac{(C_c R_{s1} (R_{s2} + R_m) + L_r)}{(s R_{s1} L_r C_c)} t}} \quad (A.9)$$

$$V_{OC} = V_{dc} \tau_1 \left(\frac{R_{s2}}{L_r}\right) (1 - e^{\frac{-t}{\tau_1}}) e^{\frac{-t}{\tau_2}} \quad (A.10)$$

where:

$$\tau_1 = \frac{R_{s1} L_r C_c}{\sqrt{R_{s1}^2 (R_{s2} + R_m)^2 C_c^2 - [2R_{s1} (R_{s2} + R_m) + 4R_{s1}^2] L_r C_c + L_r^2}} \quad (A.11)$$

$$\tau_2 = \frac{2R_{s1}L_rC_c}{[C_cR_{s1}(R_{s2} + R_m) + L_r] - \sqrt{R_{s1}^2(R_{s2} + R_m)^2C_c^2 - [2R_{s1}(R_{s2} + R_m) + 4R_{s1}^2]L_rC_c + L_r^2}} \quad (A.12)$$

for 1.2/50  $\mu$ s waveform:

$$V_{oc} = AV_p(1 - e^{\frac{-t}{\tau_1}})e^{\frac{-t}{\tau_2}} \quad (A.13)$$

with  $A=1.037$ ,  $\tau_1 = 0.4074 * 10^{-6}$ ,  $\tau_2 = 68.22 * 10^{-6}$

If it is desired to determine when the peak open circuit voltage occurs :

$$\frac{d}{dt}[V_{dc}\tau_1\frac{R_{s2}}{L_r}(1 - e^{\frac{-t}{\tau_1}})e^{\frac{-t}{\tau_2}}] = \quad (A.14)$$

$$= V_{dc}\frac{R_{s2}}{L_r}[e^{\frac{-t}{\tau_1}}e^{\frac{-t}{\tau_2}} - \tau_1[(\frac{1 - e^{\frac{-t}{\tau_1}}}{\tau_1})e^{\frac{-t}{\tau_2}}]] = 0 \quad (A.15)$$

from which derives the peak of the open circuit voltage and when it occurs occurs:

$$t_{ocp} = \tau_1 \ln\left(\frac{\tau_1 + \tau_2}{\tau_1}\right) \quad (A.16)$$

$$V_p = V_{dc}\frac{R_{s2}}{L_r}\tau_2\left(\frac{\tau_1}{\tau_1 + \tau_2}\right)^{\frac{\tau_1 + \tau_2}{\tau_2}} \quad (A.17)$$

It is also possible to derive the equation for the short-circuit current in the Laplace domain:

$$I_{sc} = \frac{V_{dc}R_{s1}C_c}{[s^2R_{s1}L_rC_c + (R_{s1} + R_mC_c + L_r)s + R_{s1} + R_m]} \quad (A.18)$$

that in the time domain is:

$$I_{sc} = \frac{V_{dc}e^{\frac{-t}{\tau_{sc}}}}{L_r\omega_{sc}} \sin(\omega_{sc}t) \quad (A.19)$$

considering

$$\tau_{sc} = \frac{2R_{s1}L_rC_c}{R_{s1}R_mC_m + L_r} \quad (A.20)$$

$$\omega_{sc} = \frac{\sqrt{(4R_{s1} + 2R_m)R_{s1}L_rC_c - R_{s1}^2R_m^2C_c^2 - L_r^2}}{2R_{s1}L_rC_c} \quad (A.21)$$

Also in this case it is possible to evaluate the peak of current and when it occurs:

$$\frac{d}{dt} \left( \frac{V_{dc}e^{\frac{-t}{\tau_{sc}}}}{L_r\omega_{sc}} \right) \sin(\omega_{sc}t) = \quad (A.22)$$



$$= \frac{V_{dc}\sqrt{1 + \tau_{sc}^2\omega_{sc}^2}}{L_r\tau_{sc}\omega_{sc}} e^{\frac{-t}{\tau_{sc}}} \cos(\omega_{sc}t + \text{atan}(\frac{1}{\tau_{sc}\omega_{sc}})) = 0 \implies \quad (\text{A.23})$$

$$\implies t_{scp} = \frac{\text{atan}(\tau_{sc}\omega_{sc})}{\omega_{sc}} \quad (\text{A.24})$$

$$\implies I_p = \frac{V_{dc}\tau_{sc}e^{\frac{-\text{atan}(\tau_{sc}\omega_{sc})}{\tau_{sc}\omega_{sc}}}}{L_r\sqrt{1 + \tau_{sc}^2\omega_{sc}^2}} \quad (\text{A.25})$$

the first negative peak is:

$$\implies I_{sc} = \frac{-V_{dc}\tau_{sc}e^{\frac{-\text{atan}(\tau_{sc}\omega_{sc}) + \pi}{\tau_{sc}\omega_{sc}}}}{L_r\sqrt{1 + \tau_{sc}^2\omega_{sc}^2}} \quad (\text{A.26})$$

and it occurs at

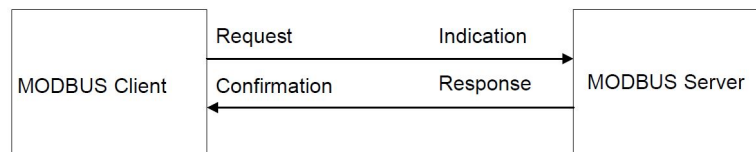
$$\implies t_{scp} = \frac{\text{atan}(\tau_{sc}\omega_{sc}) + \pi}{\omega_{sc}} \quad (\text{A.27})$$

The unknown variables are:  $C_c, L_r, R_{s1}, R_{s2}, R_m, V_{dc}$ ; at this point to reach the goal what it is needed is the definition of the functions which defines the voltage and current waveform in terms of the components and capacitor voltage, set the target values for the parameters and find a solve block to find the component value and voltage capacitor which minimize the error. After that it is possible to proceed computing the parameters through the chosen solve block and at the end compare the values found with the target values.

## Appendix B

# MODBUS TPC PROTOCOL

This chapter shows how the MODBUS TCP/IP protocol is implemented. This type of protocol provides Client/Server communication between devices connected on the same network.

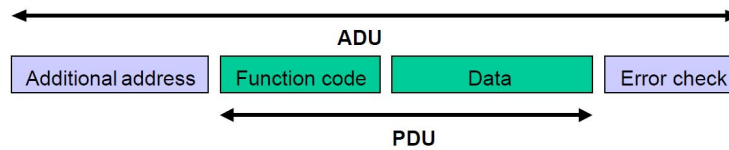


**Figure B.1:** Client/server structure [12]

As can be seen in Figure B.1, communication is based on 4 types of messages:

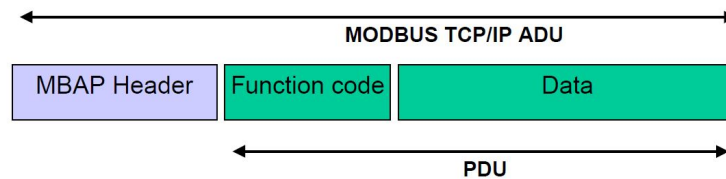
1. MODBUS Request: sent by the client to initiate the transition;
2. MODBUS Confirmation: is the response message that is received from the client;
3. MODBUS Indication: is the request message that is received from the server;
4. MODBUS Response: is the response message that is sent by the server.

The MODBUS protocol defines the PDU, consisting of function code and data; mapping the modbus protocol to a specific network will introduce other fields that give the ADU. In a MODBUS PDU each data item is addressed from 0 to 65535.  $\text{TCP MODBUS ADU} = 253 \text{ bytes} + \text{MBAP (7 bytes)} = 260 \text{ bytes}$ .



**Figure B.2:** MODBUS ADU structure[12]

In the specific case of MODBUS TCP/IP there is an ADU consisting of MBAP HEADER, function code and data.



**Figure B.3:** MODBUS TCP/IP ADU structure [6]

The MBAP HEADER is entered to identify the MODBUS ADU. THE MBAP HEADER contains:

1. 2byte TRANSICTION ID: it represents the identification of a request or response and it is initialized by the client;
2. 2byte PROTOCOL ID: for the modbus protocol it is equal to 0 and it is initialized by the client;
3. 2byte LENGTH: this field represents the number of the following bytes and it is initialized by the client for the request and by the server for the response;
4. 1byte UNIT ID: this field is needed to identify the connected remote slave and it is initialized by the client.

All MODBUS/TCP ADUs are sent via TCP to registered port 502.

To represent address and data modbus uses the "big-endian" representation, whereby, when a quantity is greater than 1 byte, to transmit it the most significant part is sent first and then the remaining part. Mod-bus data model is based on 4 types of data:

1. DISCRETE INPUT: Single bit, read only;
2. COILS: Single bit, read-write;
3. INPUT REGISTERS: 16-bit word, read-only;

4. HOLDING REGISTERS: 16-bit word, read-write;

All the data handled via modbus (bits, registers) must be located in device application memory, but there are many ways to organized the data in the device. Two possible examples are shown in Figures B.4 and B.5 :

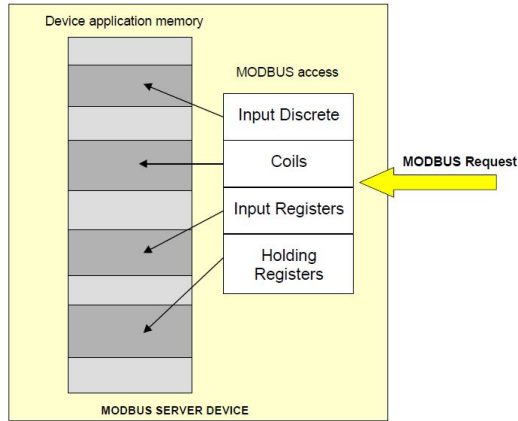


Figure B.4: Modbus data model with separated block[13]

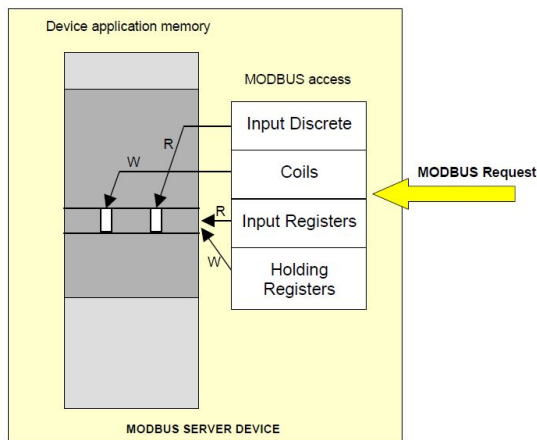


Figure B.5: Modbus data model with only 1 block[11]

### B.0.1 Function Code Categories

There are three main categories:

1. Public Function Codes: these are well defined and unique function codes;

2. User-Defined Function Codes: user can select and implements a function inside the ranges 65-72 and 100-110, the function code is not guaranteed to be unique;
3. Reserved Function Codes: these are used by some companies for legacy products and that are not available for public use.

Public function code definition:

				Function Codes			
				code	Sub code	(hex)	Section
Data Access	Bit access	Physical Discrete Inputs	Read Discrete Inputs	02		02	6.2
		Internal Bits Or Physical coils	Read Coils	01		01	6.1
			Write Single Coil	05		05	6.5
			Write Multiple Coils	15		0F	6.11
	16 bits access	Physical Input Registers	Read Input Register	04		04	6.4
		Internal Registers Or Physical Output Registers	Read Holding Registers	03		03	6.3
			Write Single Register	06		06	6.6
			Write Multiple Registers	16		10	6.12
			Read/Write Multiple Registers	23		17	6.17
			Mask Write Register	22		16	6.16
			Read FIFO queue	24		18	6.18
	File record access		Read File record	20		14	6.14
			Write File record	21		15	6.15
	Diagnostics		Read Exception status	07		07	6.7
			Diagnostic	08	00-18,20	08	6.8
			Get Com event counter	11		0B	6.9
			Get Com Event Log	12		0C	6.10
			Report Server ID	17		11	6.13
			Read device Identification	43	14	2B	6.21
	Other		Encapsulated Interface Transport	43	13,14	2B	6.19
		CANopen General Reference	43	13	2B	6.20	

Figure B.6: Function code definition[11]

The structures are[9]:

- 02 (0x02) Read Discrete Inputs:
 

**REQUEST:**  
 FUNCTION CODE (1Byte):0x02  
 STARTING ADDRESS (2Bytes):0x0000 to 0xFFFF  
 QUANTITY of INPUTS (2Bytes):1 to 2000

**RESPONSE:**  
 FUNCTION CODE (1Byte):0x02  
 BYTE COUNT (1Byte):N\*  
 INPUT STATUS (N\*1Byte)  
 N=Quantity of Input

**ERROR:**

ERROR CODE(1Byte):0x82

- 01 (0x01) Read Coils:

**REQUEST:**

FUNCTION CODE(1Byte):0x01

STARTING ADDRESS(2Bytes):0x0000 to 0xFFFF

QUANTITY of COILS (2Bytes):1 to 2000

**RESPONSE:**

FUNCTION CODE(1Byte):0x01

BYTE COUNT(1Byte):N\*

COIL STATUS(n Byte):n=N o N+1

N=Quantity of Input

**ERROR:**

ERROR CODE(1Byte): function code + 0x80

- 03 (0x03) Read Holding Registers:

**REQUEST:**

FUNCTION CODE (1Byte):0x03

STARTING ADDRESS (2Bytes):0x0000 to 0xFFFF

QUANTITY of REGISTERS (2Bytes):1 to 125

**RESPONSE:**

FUNCTION CODE (1Byte):0x03

BYTE COUNT (1Byte):2xN\* REGISTER VALUE (N\*x2Bytes)

N=Quantity of register

**ERROR:**

ERROR CODE(1Byte):0x82

- 04 (0x04) Read Input Registers:

**REQUEST:**

FUNCTION CODE (1Byte):0x04

STARTING ADDRESS (2Bytes):0x0000 to 0xFFFF

QUANTITY of REGISTERS (2Bytes): 0x0001 to 0x007D

**RESPONSE:**

FUNCTION CODE (1Byte):0x04

BYTE COUNT (1Byte):2xN\*

INPUT REGISTER (N\*x2Bytes)

N=Quantity of register

**ERROR:**

ERROR CODE(1Byte):0x84

- 05 (0x05) Write Single Coil:

**REQUEST:**

FUNCTION CODE(1Byte):0x05

OUTPUT ADDRESS(2Bytes):0x0000 to 0xFFFF

OUTPUT VALUE(2Bytes):0x0000 or 0xFF00

**RESPONSE:**

FUNCTION CODE(1Byte):0x05

OUTPUT ADDRESS(2Bytes):0x0000 or 0xFFFF

OUTPUT VALUE(2Bytes):0x0000 or 0xFF00

**ERROR:**

ERROR CODE(1Byte):0x85

- 06 (0x06) Write Single Register:

**REQUEST:**

FUNCTION CODE(1Byte):0x06

REGISTER ADDRESS(2Bytes):0x0000 to 0xFFFF

REGISTER VALUE(2Bytes):0x0000 or 0xFF00

**RESPONSE:**

FUNCTION CODE(1Byte):0x06

REGISTER ADDRESS(2Bytes):0x0000 or 0xFFFF

REGISTER VALUE(2Bytes):0x0000 or 0xFF00

**ERROR:**

ERROR CODE(1Byte):0x86

- 07 (0x07) Read Exception Status (Serial Line only):

**REQUEST:**

FUNCTION CODE(1Byte):0x07

**RESPONSE:**

FUNCTION CODE(1Byte):0x07

OUTPUT DATA(1Byte):0x00 to 0xFF

**ERROR:**

ERROR CODE(1Byte):0x87

- 08 (0x08) Diagnostics (Serial Line only):

**REQUEST:**

FUNCTION CODE(1Byte):0x08

SUB-FUNCTION(2Bytes)

DATA (Nx2Bytes)

**RESPONSE:**

FUNCTION CODE(1Byte):0x08

SUB-FUNCTION(2Bytes)

DATA (Nx2Bytes)

**ERROR:**

ERROR CODE(1Byte):0x88

- 11 (0x0B) Get Comm Event Counter (Serial Line only):

**REQUEST:**

FUNCTION CODE(1Byte):0x0B

**RESPONSE:**

FUNCTION CODE(1Byte):0x0B

STATUS(2Bytes):0x0000 to 0xFFFF

EVENT COUNT(2Bytes):0x0000 to 0xFFFF

**ERROR:**

ERROR CODE(1Byte):0x8B



- 12 (0x0C) Get Comm Event Log (Serial Line only):

**REQUEST:**

FUNCTION CODE(1Byte):0x0C

**RESPONSE:**

FUNCTION CODE(1Byte):0x0C

BYTE COUNT (1Byte):N\*

STATUS(2Bytes):0x0000 to 0xFFFF

EVENT COUNT(2Bytes):0x0000 to 0xFFFF

MESSAGE COUNT(2Bytes):0x0000 to 0xFFFF

EVENTS(N-6)X1Byte

N=Quantity of events+3x2Bytes

**ERROR:**

ERROR CODE(1Byte):0x8C

- 15 (0x0F) Write Multiple Coils:

**REQUEST:**

FUNCTION CODE(1Byte):0x0F

STARTING ADDRESS (2Bytes):0x0000 to 0xFFFF

QUANTITY of OUTPUTS (2Bytes):0x0001 to 0x07B0

BYTE COUNT (1Byte):N\*

OUTPUT VALUE (N\*X1Byte)

N=Quantity of output

**RESPONSE:**

FUNCTION CODE(1Byte):0x0F

STARTING ADDRESS (2Bytes):0x0000 to 0xFFFF

QUANTITY of OUTPUTS (2Bytes):0x0001 to 0x07B0

**ERROR:**

ERROR CODE(1Byte):0x8F

- 16 (0x10) Write Multiple registers:

**REQUEST:**

FUNCTION CODE (1Byte):0x10

STARTING ADDRESS (2Bytes):0x0000 to 0xFFFF

QUANTITY of REGISTERS (2Bytes): 0X0001 to 0x007B

BYTE COUNT (1Byte):2xN\*  
REGISTER VALUE (N\*x2Bytes)

**RESPONSE:**

FUNCTION CODE (1Byte):0x10  
STARTING ADDRESS (2Bytes):0x0000 to 0xFFFF  
QUANTITY of REGISTERS (2Bytes): 0X0001 to 0x007B

**ERROR:**

ERROR CODE(1Byte):0x90

- 17 (0x11) Report Server ID (Serial Line only):

**REQUEST:**

FUNCTION CODE(1Byte):0x11

**RESPONSE:**

FUNCTION CODE(1Byte):0x11  
BYTE COUNT (1Byte)  
SERVER ID  
STATUS(1Bytes):0x00=OFF to 0xFF=ON  
ADDITIONAL DATA

**ERROR:**

ERROR CODE(1Byte):0x91

- 20 (0x14) Read File Record:

**REQUEST:**

FUNCTION CODE(1Byte):0x14  
BYTE COUNT (1Byte): 0x07 to 0xF5 bytes  
SUB-REQ X REFERENCE TYPE (1Byte): 06  
SUB-REQ X FILE NUMBER(2Byte): 0x0001 to 0xFFFF  
SUB-REQ X RECORD NUMBER(2Byte): 0x0001 to 0x270F  
SUB-REQ X RECORD LENGTH(2Byte):N  
SUB-REQ X+1...

**RESPONSE:**

FUNCTION CODE(1Byte):0x14  
RESP. DATA LENGTH(1Byte):0x07 to 0xF5  
SUB-REQ X FILE RESP.LENGTH(1Byte): 0x07 to 0xF5 bytes

SUB-REQ X REFERENCE TYPE (1Byte): 06  
SUB-REQ X RECORD DATA(Nx2Bytes)  
SUB-REQ X+1...

**ERROR:**

ERROR CODE(1Byte):0x94

- 22 (0x16) Mask Write Register:

**REQUEST:**

FUNCTION CODE (1Byte):0x16  
REFERENCE ADDRESS (2Bytes):0x0000 to 0xFFFF  
AND MASK (2Bytes): 0x0000 to 0xFFFF  
OR MASK (2Bytes): 0x0000 to 0xFFFF

**RESPONSE:**

FUNCTION CODE (1Byte):0x16  
REFERENCE ADDRESS (2Bytes):0x0000 to 0xFFFF  
AND MASK (2Bytes): 0x0000 to 0xFFFF  
OR MASK (2Bytes): 0x0000 to 0xFFFF

**ERROR:**

ERROR CODE(1Byte):0x96

- 23 (0x17) Read/Write Multiple registers:

**REQUEST:**

FUNCTION CODE (1Byte):0x17  
READ STARTING ADDRESS (2Bytes):0x0000 to 0xFFFF  
QUANTITY TO READ (2Bytes): 0x0001 to 0x007D  
WRITE STARTING ADDRESS (2Bytes):0x0000 to 0xFFFF  
QUANTITY TO WRITE (2Bytes): 0x0001 to 0x0079  
WRITE BYTE COUNT (1Byte):2xN\*  
WRITE REGISTER VALUE (N\*x2Bytes)  
\*N=Quantity to Write

**RESPONSE:**

FUNCTION CODE (1Byte):0x17  
BYTE COUNT (1Byte):2xN\*  
READ REGISTERS VALUE (2Bytes)

\*N=Quantity to Read

**ERROR:**

ERROR CODE(1Byte):0x97

- 24 (0x18) Read FIFO Queue:

**REQUEST:**

FUNCTION CODE (1Byte):0x18

FIFO POINTER ADDRESS (2Bytes):0x0000 to 0xFFFF

**RESPONSE:**

FUNCTION CODE (1Byte):0x18

BYTE COUNT (2Byte)

FIFO COUNT (2Byte):  $\leq 31$

FIFO VALUE REGISTERS (2Bytes)

\*N=Fifo count

**ERROR:**

ERROR CODE(1Byte):0x98

- 43 / 14 (0x2B / 0x0E) Read Device Identification:

**REQUEST:**

FUNCTION CODE (1Byte):0x2B

MEI TYPE (1Byte): 0x0E

READ DEVICE ID CODE (1Byte): 01 | 02| 03| 04

OBJECT ID (1Byte): 0x00 to 0xFF

**RESPONSE:**

FUNCTION CODE (1Byte):0x2B

MEI TYPE (1Byte): 0x0E

READ DEVICE ID CODE (1Byte): 01 | 02| 03| 04

CONFORMITY LEVEL (1Byte): 0x01 or 0x02 or 0x03 or 0x81 or 0x82 or 0x83

MORE FOLLOWS (1Byte): 00/FF

OBJECT ID (1Byte): Object ID number

NUMBER OF OBJECTS (1Byte)

LYST OF:

OBJECT ID (1Byte)

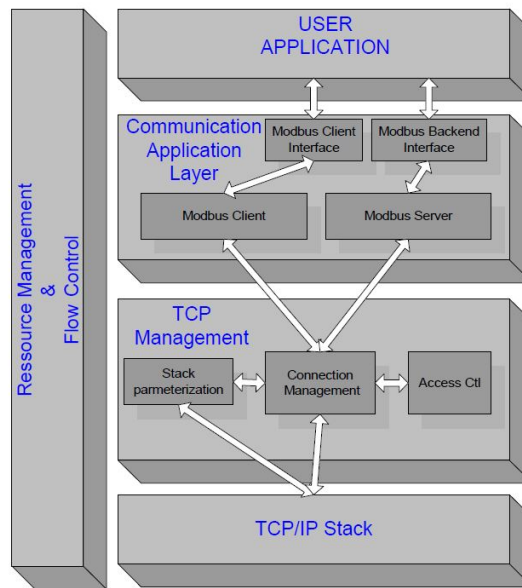
OBJECT LENGTH (1Byte)

OBJECT VALUE (object length): depends on the object ID

**ERROR:**

FUNCTION CODE(1Byte):0xAB: Fc 0x2B+0x80

### B.0.2 Modbus Components Architecture Model



**Figure B.7:** Modbus Components Architecture Model[8]

- Communication Application Layer; there are two interfaces:
  - Client and server MODBUS interface: the client interface allow the user application to control the information exchanged with remote device, while the server interface enable the local read and write local actions;
  - Backend interface: it is the interface in which the application objects are defined;
  - MODBUS client design; client can receive three events:
    1. Demand from the user application to sent a request: the request has to be send using the TCP management component service and the TCP management module can give back an error due TCP connection, for example;

2. Response from the TCP management: client analyzes the response and it sends confirmation to user application;
  3. The expiration of a time out due to a non-response: in this case it is possible send a new retry or a negative confirmation to the user application;
- MODBUS Server unit: it is needed to provide access to map application objects to readable and writable MODBUS objects, in order to have the possibility to set the application object attribute and to have the ability to trigger services on application objects. Some services can be processed directly by the MODBUS SERVER, others need the interaction with the user application and others require the MODBUS backend service.
- TCP Management layer:
    - Connection Management: It is needed for the communication between client and server, which requires a TCP connection management module. The connection management can be done in two ways: the user application manages TCP connection or the connection management is done by this module;
    - Access Control Module: it is a security module needed because sometimes it is possible that it is needed to deny data access to unwanted hosts;
  - TCP/IP Stack layer: it can be parameterized in order to arrange data flow control, address management, and connections to different system constraints;
  - Resource management and Data flow control: the data flow control is provided in all the layers in order to have a balanced in-band and out-band data flow between client and server.

# Appendix C

## Remote Commands for Surge

This chapter lists the commands that can be used remotely for surge testing:

1. \*IDN?: device identification  
response: <model>,<Firmware version>,<Software version>,<available coupling>,<available phenomena>.
2. \*GTL: command to close the communication and send the device to local mode.
3. STAT? : command to request the device's current status  
response:<device status>,<status code> 00 <signal code>,<status code>|<sync freq> |<iteration info>
4. STAT:PEAKS?: the current peak measure and the voltage peak measure values of the last triggered pulse.  
response: TOTP,ITRC,CYCL,IPEA,UPEA
5. STAT:SLAVE?: command to obtain the current state of the coupling device associated.
6. STAT:AAPR?: command to obtain all the auto adaptable parameters.
7. STAT:DEPR?:command to obtain all the auto detectable parameters.
8. CLER <error code>: Acknowledge a critical error.
9. PWMD?: command to requests the current power mode.  
response: AC or DC

10. PWMD:<power mode>: command to set the power mode of the system to AC or DC.
11. PHMD?: command to request the current phase mode.  
response: 1PH or 3PH
12. PHMD:<phase mode>: command to set the phase mode of the power mode of the system.
13. SYNS?: command to request the current source of the synchronization signal set in system.  
response: INT or EXT
14. SYNS:<sync mode>: command to set the synchronization mode of the system.
15. DEFP:<supply path>: command to sets the default power supply path as PF1 (nominal voltage of the power supply) or PF2 (nominal voltage of the power supply and the multipurpose output voltage).
16. VMSN:<integer>: command to set the nominal voltage of power supply, which is an integer between 0 and 300 V.
17. MPOV:<float>: command to set the level of DC 0-10V output voltage, percentage value between 0 and 100.
18. CPLD:<pulse>?: command to request the coupling module.  
format:<pulse>=< device>|<associated device>
19. CPLD:<pulse> <type>,<device>|<ass. device>: command to set the coupling device to use for a given module.
20. CRNO?: command to request if the system has current a current range.  
response: CRNO=<range operation option> where the range operation can be NONE or SEL (selectable range)
21. CRGL?: command to request the current ranges that the system can operate.  
response: CRGL=<list of integers>
22. CRGE?: command to requests the current ranges of the system.
23. CRGE:<integer>: command to set the current range of the system.
24. OPTM:<integer>: command to set the optimization of the delay, it can be 0,1 or -1 respectively: deactivate, activate and deactivate optimization and switch off the connected coupler.



25. LDMD:<pulse> <parameters> : command to set the module and the parameters of the test.
26. SETP: <parameters> : command to set the test's parameters.
27. START: command to start the test.
28. BREAK: command to pause the test.
29. STOP: command to stop the running test.
30. CLOSE: command to close the run window going back to remote windows.
31. CWDS?: command to request the current state of the watchdog.  
response: CWDS=<status>, where status can be ON or OFF
32. CWDS: <activation> : command to set the watchdog on/off.
33. ACPL?: command to requests what automatic couplings devices has detected the system.
34. SCOA:<phenomenon>? <coupling device>: command to requests the available coupling standards.
35. TSTO?: command to request the status of the test on button of the devices.
36. TSTO <state>: command to set ON or OFF teh test on button.

# Bibliography

- [1] Douglas E. Powell and Bryce Hesterman. «Introduction to Voltage Surge Immunity Testing». In: IEEE Power Electronics Society Denver Chapter Meeting (Sept. 2007) (cit. on pp. 1, 2, 72).
- [2] IEC 61000-4-5, Edition 3.1 2017-08 (cit. on pp. 3, 4, 6).
- [3] Claudio Amadori. *l'interruttore differenziale*. 2008 (cit. on p. 7).
- [4] IEC 1000/05 (cit. on p. 8).
- [5] ABB SACE. *Guida all'installazione e al dimensionamento dell'impianto elettrico* (cit. on pp. 10, 13).
- [6] ABB SACE. *ADDENDUM – 2022 1st edition Electrical installation solutions for buildings – Technical details* (cit. on pp. 13, 14, 77).
- [7] ABB S.p.A-Elettrificazione. *Product scope MOD/ARI/ARH range, Etidion 2* (cit. on pp. 16–18).
- [8] ABB S.p.A-Elettrificazione. *MOD WiFiCommunication Module 2022-11-07* (cit. on pp. 18, 87).
- [9] Modbus. *MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b37*, April 26, 2012 (cit. on pp. 19, 79).
- [10] Emtest. *UserManual-compact-NX series-E-V2.34\_SPZ.doc. 2.34\_SPZ /24.11.2021| 25.11.21* (cit. on p. 20).
- [11] Emtest. *RemoteManual-compact-NX-E-V1.21.docx. 1.21/ 02.04.2020* (cit. on pp. 21, 78, 79).
- [12] Modbus-IDA. *MODBUS Messaging on TCP/IP Implementation Guide V1.0b*. April 26, 2012 (cit. on pp. 76, 77).
- [13] Jakub Arm - Zdenek Bradac. <https://www.researchgate.net>. 2016 (cit. on p. 78).