FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Software Development for ICT and Flash workstation

André Marques

FINAL VERSION

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Paulo José Lopes Machado Portugal

November 22, 2022

Abstract

This document was produced within the scope of the Dissertation curricular unit, of the Masters degree in Electrical and Computer Engineering, in the Automation and Robotics Branch, of the Faculdade de Engenharia da Universidade do Porto.

This document was born from Preh Portugal's need to develop new software for its printed circuit board test lines. This need arises because of Preh Portugal's ambitious plans to implement a new production management system that brings together all processes and equipment on a platform, the Manufacturing Execution System (MES).

The developed software works as an orchestrator and controls the electrical test automation process, interconnecting several individual components in a single application.

The application is currently running on four of the five electronic test lines.

Resumo

Este documento foi realizado no âmbito da unidade curricular de Dissertação, do Mestrado em Engenharia Eletrotécnica e de Computadores, no ramo de Automação e Robótica, da Faculdade de Engenharia da Universidade do Porto.

Este documento nasceu da necessidade da Preh Portugal desenvolver um novo software para as suas linhas de teste de placas de circuito impresso. Esta necessidade surge por causa dos planos ambiciosos da Preh Portugal de implementar um novo sistema de gestão de produção que reúne todos os processos e equipamentos numa só plataforma, o Manufacturing Execution System (MES).

O software desenvolvido funciona como um orquestrador e controla do processo de automatização do teste elétrico, interligando vários componentes individuais numa só aplicação.

Neste momento a aplicação encontra-se a funcionar em quatro das cinco linhas de teste eletrónico.

Agradecimentos

Agradeço em primeiro lugar ao meu orientador da Faculdade de Engenharia da Universidade do Porto, engenheiro Paulo José Lopes Machado Portugal pela disponibilidade, ajuda e conselhos ao longo deste projeto.

À Preh Portugal por ter confiado nas minhas capacidades e oferecer a oportunidade de desenvolver os meus conhecimentos e evoluir como profissional.

Ao engenheiro Armindo Rocha, meu orientador na Preh Portugal, e ao técnico Pedro Pereira pelo excelente acompanhamento e integração na equipa. Assim como, por todo o conhecimento que me transmitiram durante estes seis meses de trabalho.

Por fim, agradeço á minha família, especialmente aos meus pais e irmã, pelo suporte que me deram durante a realização deste projeto, e pela paciência que tiveram para comigo

André Marques

"We cannot solve problems with the same thinking we used to create them."

Albert Einstein (1879 - 1955)

Contents

List of Figures

List of Tables

Abreviaturas e Símbolos

ICT	In Circuit Test
PCB	Printed circuit board
PLC	Programmable logic controller
ILS	In-Line Test System
UUT	Unit Under Test
COM	Component Object Model
vismda	VisualMda
GUI	Graphical user interface
RS232	Recommended standard 232
DTE	Data Transmission Equipment
DCE	Data Communication Equipment
RTS	Ready to Send
CTS	Clear to Send
DTR	Data Terminal Ready
DSR	Data Set Ready
FPT	Flying Probe testing
BPS	Bytes per second
SMT	Surface-mount technology
ILF	In-Line Flash
MS	Microsoft
SQL	Structured Query Language
TNr	Trace Number
PC	Personal Computer
Db	Database
XML	eXtensible Markup Language
AOI	Automatic Optical Inspection
MES	Manufacturing Execution System

Chapter 1

Introduction

On this chapter it is done an introduction to the dissertation developed at Preh Portugal, it consist of a quick introduction to the company and the problem to be solved on this project. This chapter will also cover the motivation and the objectives of the project, and it finishes with a quick overview of the document structure.

1.1 Context

This project was born from Preh's need to create a software that is capable of controlling an In Line Circuit Test (ICT) station. The ICT's main function is to test the electrical components of a printed circuit board (PCBs) and prevent it from continuing the normal production cycle if a problem is found. This application will run on an embedded computer located in the station itself, and shall be written using the C# programming language. The application to be developed will manage the ICT in terms of traceability of the PCB's and will serve as an interface with the programmable logic controller (PLC) that controls the equipment. The application should also communicate with the checksum program that is responsible for testing the electronic components of the PCB.

1.1.1 Preh Portugal

Preh was founded in 1919 by Jakob Preh in Bad Neustadt on Germany. Preh is a multinational company that operates in the automotive industry and is spread across the world, with locations on Germany, Portugal, Sweden, United States of America, China, Romania and Mexico. It partners with major car manufacturers such as BMW, Porsche, Audi, Ford, Volvo, etc.[1]

Preh Portugal facilities, fig.1.1, are located at Trofa, on Rua dos Moinhos da Lagoa. The main products manufactured here are the climate panels, fig.1.2, of cars from the following brands: BMW, Ford, Porsche and Audi.

Figure 1.1: Preh Portugal Facilities [1]

Figure 1.2: Preh Portugal products [1]

1.2 Motivation

The motivation to embrace this project comes from the fact that software development is one of the most important areas when it comes to automate a task or a process. Therefore, this work is relevant because it provides a company with some state-of-the-art technology to improve quality and time efficiency.

1.3 Objectives

The realization of the proposed application can be split into four different steps. The first one consists of preforming a PCB background check, this means that the scanners will read the PCB code and consult a database to verify if it has passed the previous processes. The previous processes consist mainly in the placement of electrical components and their welding, which are not the target in this project. If the PCB passes this verification and has the right work reference, then the program begins its second stage. In this phase, the software sends a signal to the checksum app that starts the PCB test process. After the test is complete (third stage), the software to be developed needs to communicate with the checksum app to get the test report and check if the PCB passed the test. At this time, a new entry shall be registered on the database informing the result.

The final step of the software is to create a functionality that communicates with the Workstation PLC and exchanges information about the condition of the test needles and therefore assist in the maintenance of the station.

The software developed was entitled *"ICT_ILF_Manager"*.

1.4 Document Structure

Besides the introduction, this paper contains four more chapters. In chapter two it will be described the fundamental background technologies to be use during the project. In the chapter three it will be explained the software architecture. In chapter four it will be shown the results obtained ate the end of the project as well as the functional tests used to validate the software. The last chapter is the conclusions and the future work.

Chapter 2

Background

2.1 In-line Circuit Testing

In-Circuit testing, ICT, is a very powerful system when it comes to test PCBs. By using a bed of nails, ICT can test the performance of every single component in a PCB, regardless of the others components connected to them. The analogue components, such as resistors capacitors, inductors, diodes, transistors, and operational amplifiers; are the most commonly tested parts. However, some ICT lines are also able to test digital circuit, but it comes with greater economic cost. ICT plays a big role when it comes to improve product quality because it allows companies to perform high level testing of the circuit boards, preventing malfunctions from occur in the future.[3]

2.1.1 Basic concept of ICT

ICT equipment provides one of the best ways to detect misplaced components and out of range values for the tested components. It can not assure the product functionality, but if it has been well assemble it should work correctly. Since the majority of the faults result from the production process, it usually consists of open/short circuits or wrong placed component.[3]

In-circuit test can be divided into three different elements:

- *In circuit tester*: The circuit tester consists of the set of sensors and drivers that preform the test of the circuit. In average, these systems tend to have more than 1000 of these points.

- *Fixture*: The fixture is the connector between the In circuit tester and the bed of nails, it takes the connections from the set of sensors and routes them to the respective needle on the bed of nails. In Fig.2.1

- *Software*: The software usually consists of a sequence of procedures that instructs the system on the test to be performed, typically it contains the details of what points should be tested, the type of test to run and the pass/fail criteria.

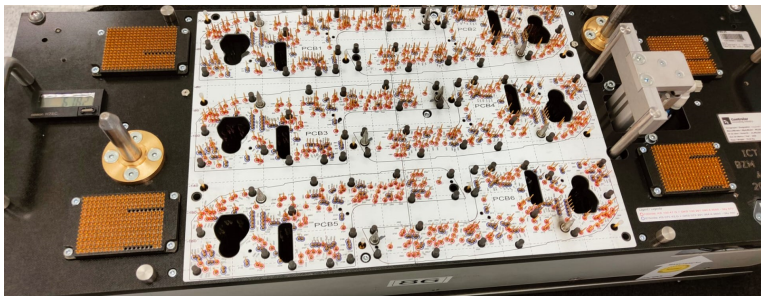


Figure 2.1: Example of a bed of nails

2.1.2 ICT advantages and disadvantages

Like in every other technology, it is important to compare the advantages and disadvantages of the ICT system in order to determine the best form of test application. The In-circuit test advantages are listed below:[3]

- *Easy detection of manufacturing defects*: Like it was said before, most of the problem arise from the production process and those are very easily detected by testing the circuit for open/short circuits or wrong placed components.
- *Easy program generation*: An ICT is easy to program, usually it consists of a spreadsheet environment where every row corresponds to a test step.
- *Results are easy to analyze*: As the system will indicate a flag for every test point where a malfunction is detected and specify the type of error, it makes the problem interpretation and resolution easy.

Despite all the advantages presented, this system also haves some disadvantages that might compromise its use:

- *Fixtures are expensive and difficult to update*
- *Test access is becoming more difficult*: With the advances in electronic technologies the components tend to become smaller and smaller, making the access to the nodes increasingly difficult.
- *Back-driving*: One problem that arises with these tests is that to test some components in integrated circuits, the nodes have to be held at a certain level. This means forcing the output of a digital circuit to an alternative state.

2.1.3 Main ICT Systems

The current electronics manufacturing industry is full of different types of solutions used to test the electric properties of PCB's components. The main types of ICT machines that are available include: Standard ICT machine, Flying probe tester, Manufacturing defect analyser (MDA), and Cableform tester.

2.1.3.1 Flying probe tester

Just as the name implies, flying probe testing (FPT) uses test probes that move on a X/Y grid from test point to test point. These movements are under software control and follow a set of instruction predefined by a programmer. Usually every FPT system comes with its own software to define the PCB format, the test point coordinates, and the set of instructions to be followed. [4]

Advantages of FPT system: [4]

- *There is no need for special fixtures*: In opposite to other test system FPT does not require a "bed of nails" fixture, seeing that the probes move according to software control. Only a simple fixture is needed to hold the panel in place.
- *Easier to adapt to a different hardware*: Because the probes movement is controlled by software, it is simple to change pad positions and type of component by purely changing the test software. This allows to reduce the time needed to pass from the design to the production
- *Reduced test development time*: The software for FPT system can be developed in very little time from the PCB design files.

Despite all the advantages listed before, this system also comes with the disadvantage that it is a slower test system to mass production solutions. [4]

2.2 Analyst ILS In-Line Test System

The test system that already exists in the factory is the Analyst ILS In-Line, more properly the model ILS-1200. There are interesting features of this system which are worth mentioning, namely the main ones:[2]

- Handles boards up to 59cmx38cm with up to 3000 test needles
- test handling is fully automated
- Re-probe for contaminant break-through
- Re-test for failed Unit Under Test(UUT)

The Analyst ILS system can test the entire UUT automatically and without the need to apply power to the PCB components. To achieve that, the system applies techniques such as DC or complex-impedance measurements in conjunction with multipoint guarding. These two methodologies provide enough requirements to find the majority of faults such as shorts, opens and wrong placement of components. Since no power is supplied to the UUT, it is completely safe to test the PCB. The Analyst ILS was designed to perform effective testing for most digital or analog assemblies produced in the current era. The main power-down test capabilities of this machine are the following: [2]

1. Opens/Shorts
2. Resistance/ Capacitance/ Inductance/ Voltage measurements
3. Transistors/ FETs
4. Opto-Isolators/ Relays
5. Diodes/ Zener Diodes/ LEDs
6. Transformers
7. Capacitor polarity

2.2.1 System Software

The system comes with a very complete and easy to use software package that runs on the Windows operating system. Which allows users to find extensive online help. The software package is divided in four major blocks:[2]

1. *Testing Environment:*

   When it comes to the test environment, the system can accommodate a variety of options, being the simplest one a single test, that shows a red indication when the UUT fails or a green one when it passes. In this case, when a UUT fails, the system enters a halt state where the operator has to analyze the faulty UUT and restart the process. However, the system can be configured to preform an automatic re-test of the assemblies or simply move on to the next UUT in line.

   As the test is preformed, it is possible to observe a graphical interface which shows the status of each UUT. At the end of the test the panel can have one of the following states fail/pass/skip, the results are separated by UUT and stored in a report.

   In an automated in-line system, it's important to keep track of the UUT, so the system is capable to generate automatic reports during each batch of UUTs.[2]

2. *Statistical Process Control:* In order to execute statistical analyzes of the data, this system is capable of generate three different formats of reports. The system can report on all the UUTs, or choose particular samples to analyze.[2]

   - *The Production report:* this report is useful because it allows the user/operator to interpret the data relative to the UUT. This data consists of UUTs failure rate and how many defects were detected. This information is valued because it represents the overall production and can be filtered by: production by shift, production by UUT, and production fail rates.[2]
   - *The Pareto report:* this report lists the type of faults by occurrence and is particularly useful to determine which type of source inputs the greater error. [2]

- *The X-Bar/Sigma report:* On this kind of report the data is graphically displayed and shows the following individual analog measurements: the mean, standard deviation, sigma limits, among others. This information is important because it helps to fine-tune the test program tolerance and control the limits applied to a large range of UUTs.[2]

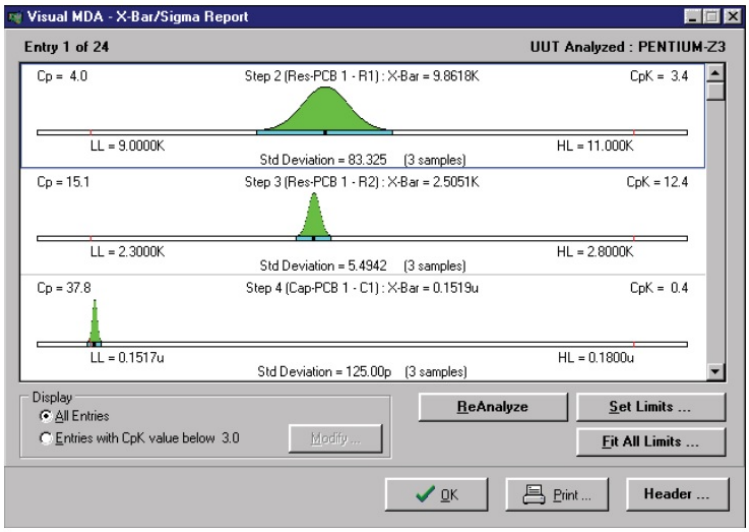


Figure 2.2: Example of an X-Bar/Sigma report[2]

The raw statistics data is saved to the computer disk on the PC in ASCII format, this allows us to write our own custom programs and develop modules to automatically analyze the information. [2]

3. *Test Program Generation:* The system presents the user with all the necessary software (CheckSum) to write test programs and build text fixtures. Despite that, many users fell the need to contract local contractors to help in this process. A spreadsheet environment is used to generate the test programs, in this sheet each row corresponds to one step of the test, however one row may also contain additionally information, for example, in a RES(resistance test) it's necessary to specify two test point names and numbers, a measurement range, a nominal value and the test limits interval.

   The checksum test programming language is full of features, which allows the programmer to include math function, file I/O, operator inputs, and external programs callings. Each test program is limited to 30000 steps, however two or more test programs can be linked together, providing unlimited test programs. In order to facilitate the program generation, the system is prepared to receive CAD data to generate a preliminary test program.

   The input or generation of test programs can be done offline and on any computer, but as there is no simulator, the optimization of the programs must be done at the system station.[2]

4. *Station Configuration:* The station configuration software provides the specification of the hardware present in the system and include self testing of each module. The self test software is important to re-ensure the proper operation of all the system modules, and to calibrate the same according to internal standards. The system can also preform self-calibration against external calibration modules if external traceability is required. [2]

2.3 COM reference library

Component Object Model is a binary-interface standard, object-oriented system used to iterprocess communication. COM objects can be used in a large set of programming languages, such as c++ and C#. [5]

CheckSum offers a COM interface to allow third-party users and developers to develop applications that utilize CheckSum's electrical test hardware and software using their favorite software development environment, such as Microsoft's Visual Studio. The CheckSum software can act as a COM server that can respond to requests from other programs.

The COM interface was introduced with CheckSum's 32-bit (Win XP/Win2000) version of its tester operating system software, generically referred to as VisMDA. The 32-bit version is called visems. The COM server library is named Vismda.

GraphicVismda is a super set of the original AutoVismda COM interface. All of the functions of AutoVismda are supported in GraphicVismda.

The objects within Vismda are organized here by the classification.

- Events – Triggered by something within the vismda environment
- Action – Instructions to the vismda program
- Inquiry – Queries to the vismda program

2.3.0.1 Events

Events are raised by vismda only while a test program is actually executing or has just completed execution. Handling events typically requires both an event handler subroutine as well as a delegate subroutine to avoid cross-program thread issues. Refer to your chosen language's help documentation on event handling. Below, it's possible to observe an event object example:

"OnActivePCB(ByVal Row As Integer, ByVal Col As Integer, ByVal Event As Vismda.TxTestActivePCBEvent, ByVal color As UInteger)"

This event occurs when the active PCB in a Panel Display changes during testing. This event is raised when a PCB test type is encountered in the test program. The event passes the row and column address of the active PCB, the event code and the color of the active PCB. Possible event codes are:

- *teActivePCBSkip* = 0 – PCB was skipped

- *teActivePCBPass* = 1 – PCB passed
- *teActivePCBFail* = 2 – PCB failed
- *teActivePCBTested* = 3 – PCB was tested
- *teActivePCBStop* = 4 – Test stopped
- *teActivePCBSelected* = 5 – PCB selected(testing)

2.3.0.2 Action

Action methods and properties allow the user to modify/initiate test program execution, as well as restrict the ability of an operator/user to interact with the vismad GUI. Except for the StopTest method, all other action methods are intended to operate only prior to or post test program execution. Attempting to invoke them while a test program is actually running may result in undesirable results. Unless otherwise specified, any property listed in this section can also be evaluated.

"ReTest()"

Description: This method is used to perform a Retest. All previous test results, including change to Batch counters, will be discarded. No calling parameters are required by this method.

2.3.0.3 Inquiries

Inquiries allow the programmer to interrogate vismda to determine tester status as well as the pass/fail status of the PCBAs being tested. Some inquires can also be used as action methods to set tester status.

"GetSerialNumberUsed As Integer"

Description: This property is read only and tells the programmer whether or not the current operating environment requires/uses a serial numbers. 0 – Not used, any other value indicates that serial numbers are used.

2.4 RS232 Serial Communication

The RS232 communication protocol was first introduced in the 1960s years, and is today one of the oldest and most popular protocol for serial communications within the industries. The term RS232 stands for *"Recommended standard 232"* and its main application are: computer printers and factory automation devices (such as PLCs). This type of communication is used for small to medium range distances. The maximum cable length recommended for RS232 is 15 meters. Currently, there are more modern protocols, such as RS485, SPI, I2c, CAN, among others.[6]

The serial communication is the process of sending data sequentially from a Data Transmission Equipment (DTE) and a Data Communication Equipment (DCE). In the case of serial communication, the information is sent bit by bit and can be a slow process when compared to parallel

communication. Parallel communication instead of sending one bit at a time, it sends a conveying of multiple binary digits.[7]

- ***Baud rate***: Baud rate is the value that defines the speed of a transmission. It is expressed in bits per second. The most frequent used values are 9600, 11500, 14400, ect...
- ***Stop bits***: Stop bits are used to identify the end of a transmission. Most common values are 1, 1.5, 2 bits.
- ***Parity bit***: The parity bit is the simplest way for error detecting, it is added on the final of the bit sequence. If it has value 0, then it is even, if his value is 1 then the parity is odd.
- ***Handshaking***
  - *No-Handshaking*: If there is no handshaking then the DCE needs to read the data sent by the DTE, before the transmitter sends new data. If the DCE doesn't read the data in time, then the new information will overwrite the old information. The diagram below, Fig.2.3 shows a communication where the receiver fails to read the 4th bit, and it gets overwritten by the 5th bit.

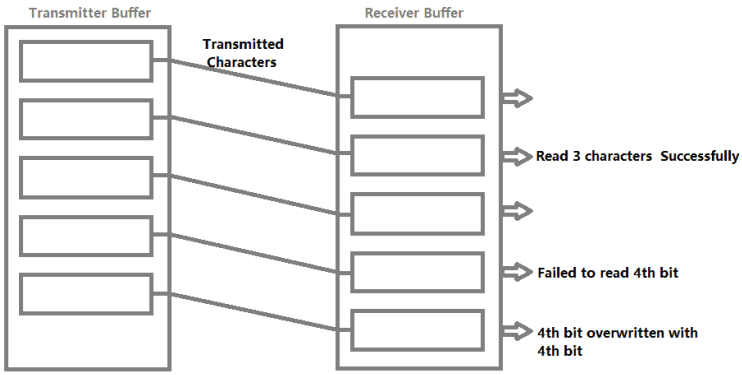


Figure 2.3: Example of No-Handshaking
[6]

  - *Hardware Handshaking*: This type of handshaking allows the receiver to send a signal to the DTC when it is ready to receive the data. The transmitter will send the data to a receiver, which will be loaded into a buffer, and the receiver will send a signal telling the transmitter not to send any more data. this type of protocol uses 4 signal:
    - Ready to Send (RTS)
    - Clear to Send (CTS)
    - Data Terminal Ready (DTR)
    - Data Set Ready (DSR)
  - *Software Handshaking*: The software handshaking uses two characters to start and end a transmission, the X-ON and the X-OFF. In this case, the DCE sends the X-OFF to

stop the DTE from sending more data, and the X-ON to start the transmission. The fig.2.4 represents a communication using software handshaking.

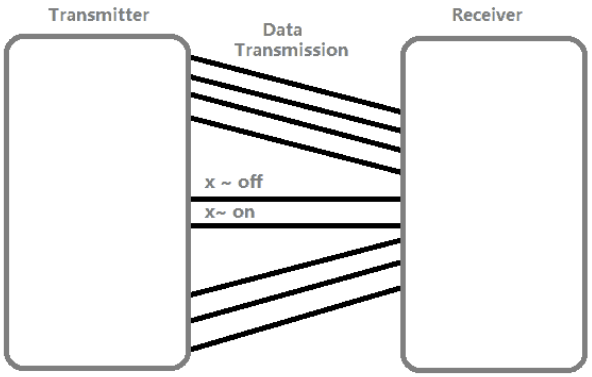


Figure 2.4: Example of Software-Handshaking
[6]

2.5 ICT-ILF Production lines shop floor

In order to understand where the application will run it is necessary to look at all the different processes and stages on the life cycle of the PCBs panel. By looking at the Fig.2.5, it is possible to identify the main stages of a panel life.

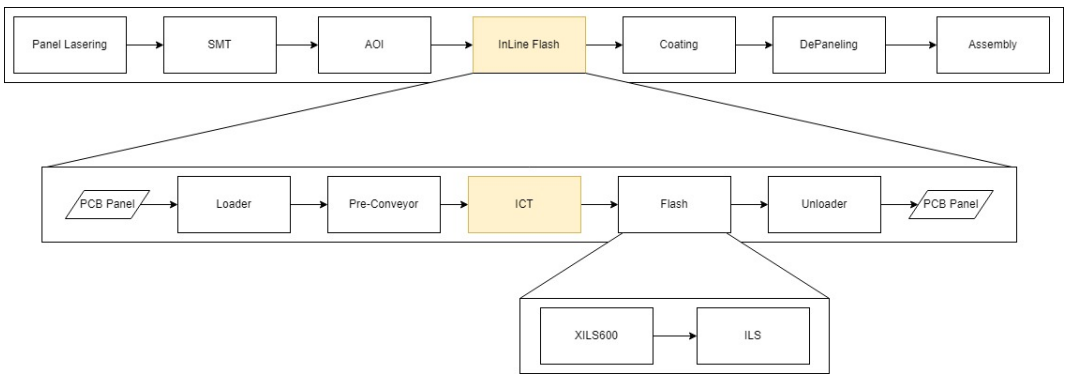


Figure 2.5: Shop Floor Scheme

There are seven main stages the panel most pass to then become the final product. The first consists on the Panel Lasering. At this stage there is only a empty panel and it is where the electrical connections and lines will be printed into the board. At this moment each panel will receive a unique Serial Number that is going to identify the panel until the end of the product life.

The next stage is called *SMT* and is where the electronic components will be pick and placed on the Panel. After the component is placed it will be soldered into the panel. The *AOI* process corresponds to a visual inspection where it is ensured that all the components are placed in the right spot and the solder is well done.

The *InLine Flash* stage is the most relevant to this document because it is where the ICT-ILF Manager will run. This process is composed of five devices:

- *Loader*: The first device is the loader module, fig.2.6, here the operator will place the Panel magazines that will enter the *InLine Flash* process. On the fig.2.7 it is shown an example of a magazine.

Figure 2.6: Loader module

Figure 2.7: Example of a magazine

- *Pre-Conveyor*: This module is a simple conveyor that will carry the Panel until the entry of the ICT module. At this module, fig.2.8, there is installed a scanner that is responsible of reading the Panel Trace Number.

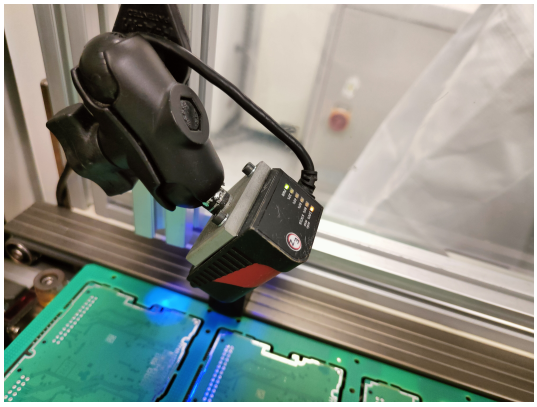


Figure 2.8: Pre-Conveyor module

- *ICT*: This is the most important module because it is where the developed application will be running. This module, presented at fig.2.9, is responsible for the electronic test of the PCBs components. In this module exists a PLC (fig.2.10) that is responsible for controlling the transport of the panel from the *Loader* to the *Flash* module.

Figure 2.9: ICT module

Figure 2.10: PLC on the ICT module

- *Flash*: This module is responsible for programming all the micro controllers on the panel. The most recent lines only have the *XILS600* module (fig.2.11), however the older lines still have a *ILS* module (fig.2.12), because some of the older products still need to be flash in one of those modules.

Figure 2.11: *XILS600* module

Figure 2.12: ILS module

- *Unloader*: This module receives the PCB panels and stores them into magazines. It is the opposite of the *Loader* module.

After the panel goes trough the InLine Flash process it goes to the next stage which is the *Coating* process. In this process it will be applied a protective varnish to every PCB component, with the exception of those that achieve a high working temperature.

The penultimate process is the *Depaneling*, here the individual PCBs will be separated from the main panel (fig.2.13).

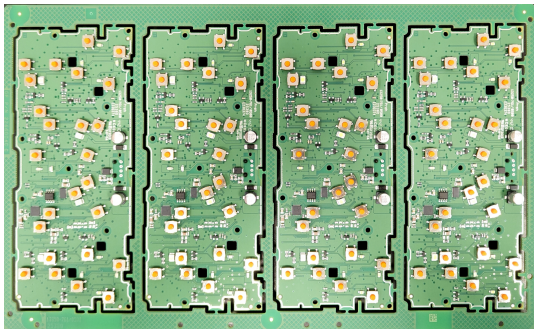


Figure 2.13: Example of a panel of PCBs

The last stage is the *Assembly* process, and it is where the single PCB will be Incorporated on the plastic pieces to form the final product.

2.6 OVERALL EQUIPMENT EFFECTIVENESS

The overall Equipment Effectiveness (OEE) is the metric use by Preh for measuring manufacturing productivity. The OEE can be divided in three main factors:

- Availability: This indicator takes into account the time that a station is available for production during a Planned production. To achieve a 100% score the process needs to be always running during the planned production. This factor can be affected by unplanned stops in production, such as equipment failures and material shortages. [8]
- Performance: This indicator takes into account the cycle times, slow cycle times means less performance. To achive a 100% score the process needs to be running as fast as possible during the production. [8]
- Quality: this indicator is the coefficient of Good parts over the total Parts. It takes into account the parts that have defects and need rework. To achieve a 100% score the process needs to only produce good parts. [8]

The result of the OEE is calculated by multiplying the three factors by each other as it can be seen on fig.2.14. The OEE is very used to identify production problems on the different lines, and it provides a fair metric to compare the productivity of the production lines.

Figure 2.14: OEE formula

Chapter 3

Architecture

3.1 Traceability

Traceability refers to the recording of data through means of data-matrix code or other tracking media, of all movement of a product and steps within the production process. The main goals of the traceability are:

- Record the various stages of a product.
- Validate each stage by previous specifications.
- Create a sequence for materials and processes.
- Create rules for assembly different elements of a part.

3.1.0.1 Infrastructure

The following elements are directly related to the traceability process: Industrial network (covering all productive area), File server, Database server (MS SQL server), and Software applications.

- *Industrial network*: This network is accessible in all productive area. Every assembly line is connected to it providing full access to applications (file server) and the database server.
- *File server*: The file server stores the applications required for the operation of traceability as well all applications of analysis and intervention in the tracking system.
- *Database server*: The database server is a Microsoft SQL server 2008. All production machines based on PC (and some based on PLC), exchange information with the SQL server. The databases are created specifically for the assembly lines and contain all objects necessary for implementing the traceability process. Each project gets its own database.
- *Software applications*:There are two levels of software applications: 1- SQL server level: To access the database, either query previously stored data, or storing information, several stored procedures, functions and views were created that can be used by any application, as

long as they comply with the requirements. 2- User level: Several applications were created, where it is possible to check the history of parts, change their status, download reports, and calculate statistics.

3.1.0.2 Tracing/tracking media

As it can be seen in Fig.3.1 the tracing media utilized by Preh consists of a code with 14 numeric digits, divided int two fields: Group number (first 4 digits) and a Serial number (last 10 digits). Together these two fields define a Trace Number (TNr).

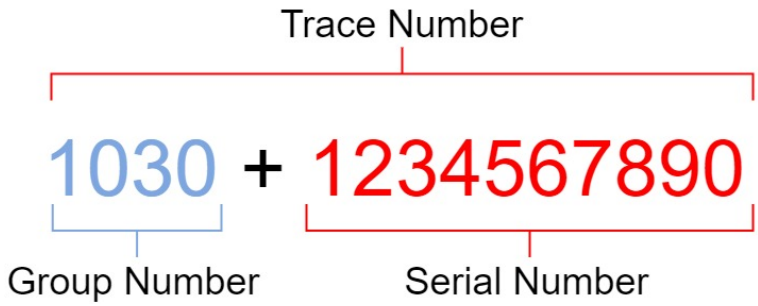


Figure 3.1: Trace Number definition

The group number is also divided into two field, the first two digits define which side of the PCB panel should be facing up when it enters the production line. They follow the structure presented in table 3.1. This information is always true to every project it can only be applied to the assembly lines, for the ICT and Flash process these information is stored on a special table that will be discussed further on this document.

Table 3.1: Trace Number side definition

New Projects:	
10	Top
11	Bottom
Old Projects:	
01	Klima TT project PCB 1 version TP
02	Klima TT project PCB 2 version GB
03	Klima TT project PCB 3 version LV

The last two digits of the group number are used to identify the Preh plant site, as shown in table 3.2

Table 3.2: Preh Plant Sites identification

Preh Plant Site:	
10	Preh Germany
30	Preh Portugal
44	Preh Romania
52	Preh Car Connect
70	Preh Mexico
88	Preh China

3.1.1 Database Structure

At Preh every product is treated as a new project and has its own database. This database is standard and contains several tables, stored procedures, views and functions that altogether make a full functional system. The database structure is presented at fig.3.2. A product can have various versions with different characteristics such as different physical components or different software. To identify these differences every version as a unique reference.

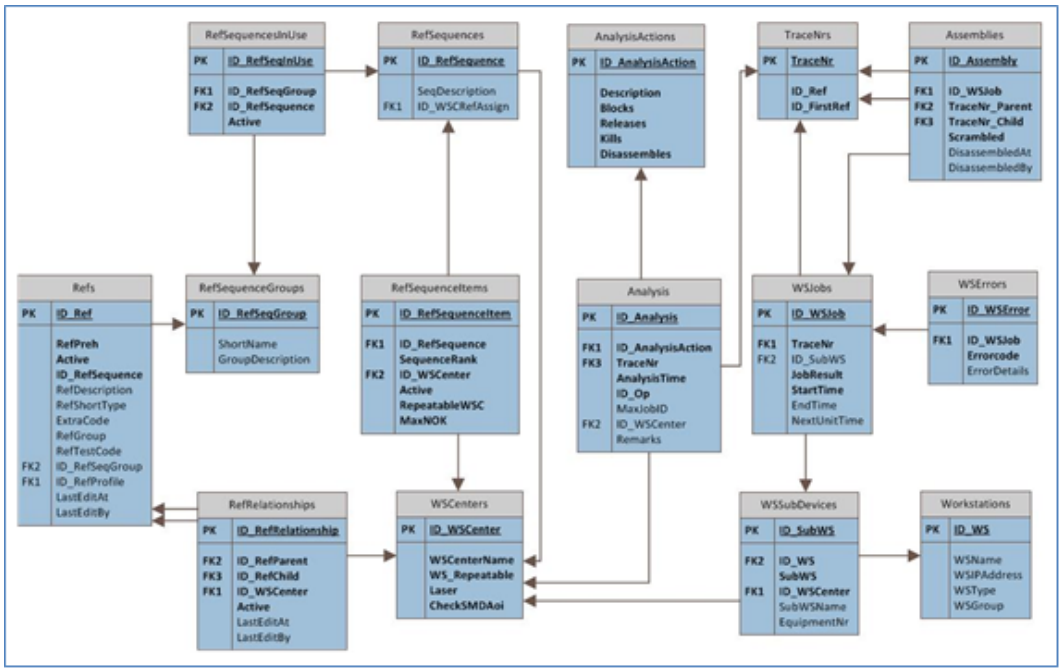


Figure 3.2: Database Structure

- *Refs*: Contains a list with all the references involved in the project.
- *RefSequences*: Each record represents a process sequence rule that a trace number with a specific reference must follow.
- *RefSequenceItems*: Describes the devices that belong to each sequence.

- *RefSequenceGroups*: It is possible for a reference to have more than one sequence. This table describes the groups of sequences that are associated to the references.
- *RefSequencesInUse*: Details the sequences that are associated to each sequence group. It is possible to activate or deactivate sequences within a group.
- *Workstations*: A simple list of all devices involved in the process.
- *WSCenters*: Each workstation center can represent more than one device as long as they perform the same task.
- *WSSubDevices*: Lists all the devices involved in the process.
- *TraceNrs*: The first step of traceability is the association of a reference to a trace number. This table lists all the trace numbers and their reference associations.
- *WSJobs*: A Job is a process step. This table lists the jobs of all the trace numbers, including the device where it took place.
- *Assemblies*: Lists the relationships between trace numbers.

3.1.2 Traceability commands

To assist in software development Preh implemented some store procedures where all the logic resides on the SQL side. In fig.3.3 is represented the order in which these stored procedures need to be executed.

- *AssignRef*: Assigns a reference to a trace number. Usually this command is executed only once at the beginning of the process sequence.
- *CheckTraceNr*: Checks if the trace number can perform an operation in a specific device.
- *CheckAssembly*: Checks if a trace number can be associated with another trace number on a specific device. These store procedure is only used on the Assembly stage.
- *JobStart*: Starts a task or process in a device. By default, the result of an open job is always 1 (NOK).
- *SaveAssembly*: Associates two trace numbers on a specific device according to the configuration rules. These store procedure is only used on the Assembly stage.
- *JobEnd*: Terminates a process or task in a device. The result can be 0 (OK) or 2 to 254 (NOK).
- *UpgradeRef*: When a component has already a reference assigned, it can be upgraded to another one depending on configuration.

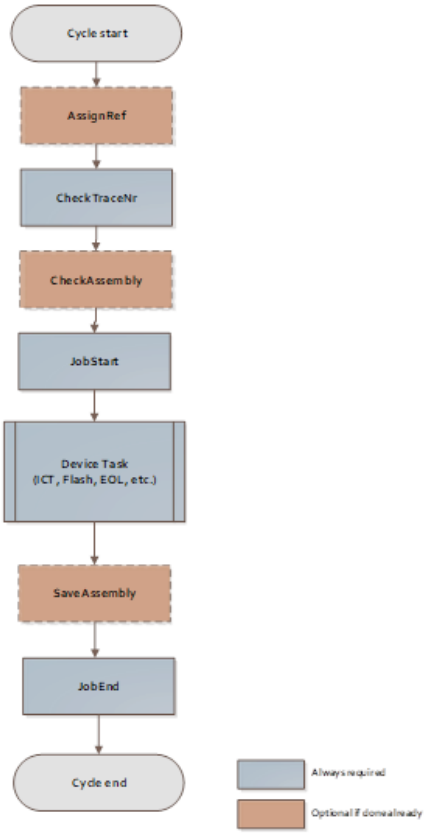


Figure 3.3: Store Procedures Order

3.2 Application block diagram

At fig.3.4 is represented the communications diagram of the application. The ICT_ILF_Manager will be running on the ICT Module embedded PC. It will need to communicate with the ICT module PLC via RS232. The ICT PLC is responsible for making the transport of the panel from the *Loader* module to the ICT module.

The ICT_ILF_Manager also uses a RS232 communication protocol to communicate with the scanner. The scanner is responsible for reading the Panel Trace Number and transmit it to the PC.

There are two possible production lines configurations, on the first the ICT module is directly connected to the Flash module, in this case there is the need to use another RS232 serial communication to transmit the Trace Number that was read before to the Flash module PC.

On the second configuration there is no need to have this communication because there is a Middle Conveyor connecting the ICT module to the Flash module. The Middle Conveyor has a scanner that communicates directly with the Flash Module PC.

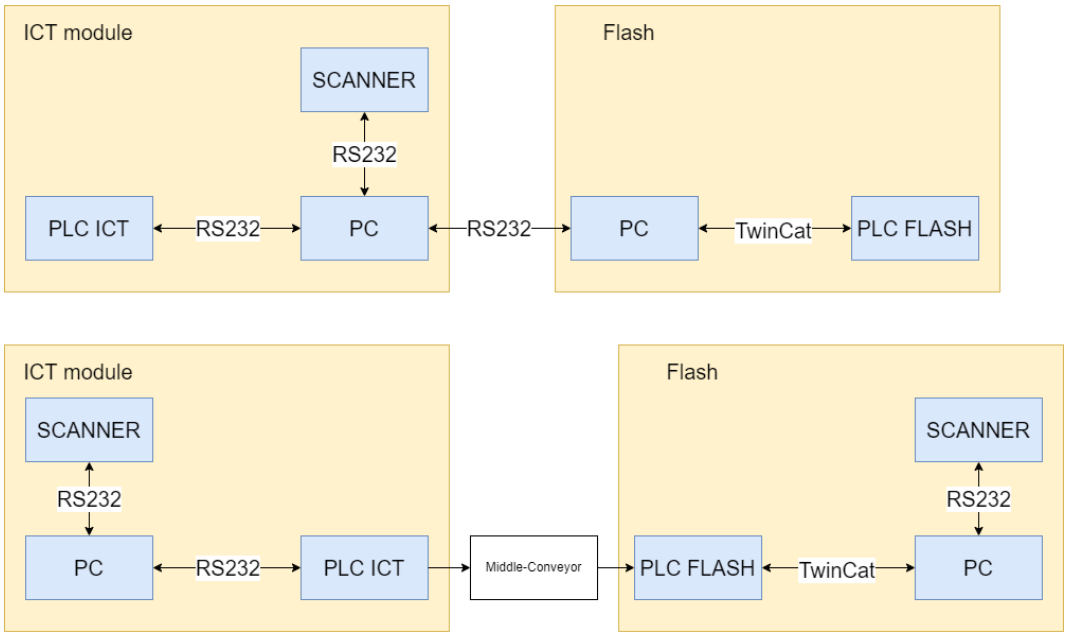


Figure 3.4: ICT_ILF_Manage Communications diagram

The PLC logic program was already implemented at Preh and it follows the protocol at fig.3.5. The RS232 protocol on all the established connections have the following parameters:

- *BaudRate*: 9600 bps;
- *Parity*: None;
- *StopBits*: One;
- *Handshake*: XOnXoff;
- *DataBits*: 8;

The first message that the PLC sends to the PC is the bed of nails identifier, and then it goes to the Initial State. When a Panel arrives at the Pre-Conveyor the PLC transits to a new state, and sends the 0x01 message to the PC.

At this moment the ICT_ILF_Manager replies with the code 0x21, which indicates it will trigger the scanner to read the Panel Trace Number. At this stage stage the PLC enters a state where it will wait for the PC to reply with a good code, bad code, or ignore code message. If the PC sends the code 0x23 it means the panel cannot be tested by the ICT module and the PLC proceeds to expel the panel to the *Unloader* module. If the PC replies with 0x25 it means tho panel is ready to be tested and the PLC carries it to the ICT module test position, entering the state *"Test Position"*. If the PC response is 0x22 it means the panel will skip the ICT module, but it will stop at the Flash module to be programmed.

Once the PLC is at the state *"Test Position"* it will wait for the ICT_ILF_Manager to send the 0x24 signal, which indicates the electric test will start. The PLC will then wait for the test result.

Once the electric test fails, the ICT_ILF_Manager will send one of these three messages:

- *0x26*: If the panel needs to do a ReTest. Every time panel fails the electric test it is defined by Preh ICT Protocol that it needs to do one Retest before it is considered a defective PCB.

- *0x34*: If the panel passes the electric test. Upon receiving this message the PLC will carry the Panel to the the Flash module, or to the Middle Conveyor.

- *0x2c*: If the Panel fails the electric test and has already been Retested. After receiving this message the PLC will cast out the Panel to the *Unloader* module.

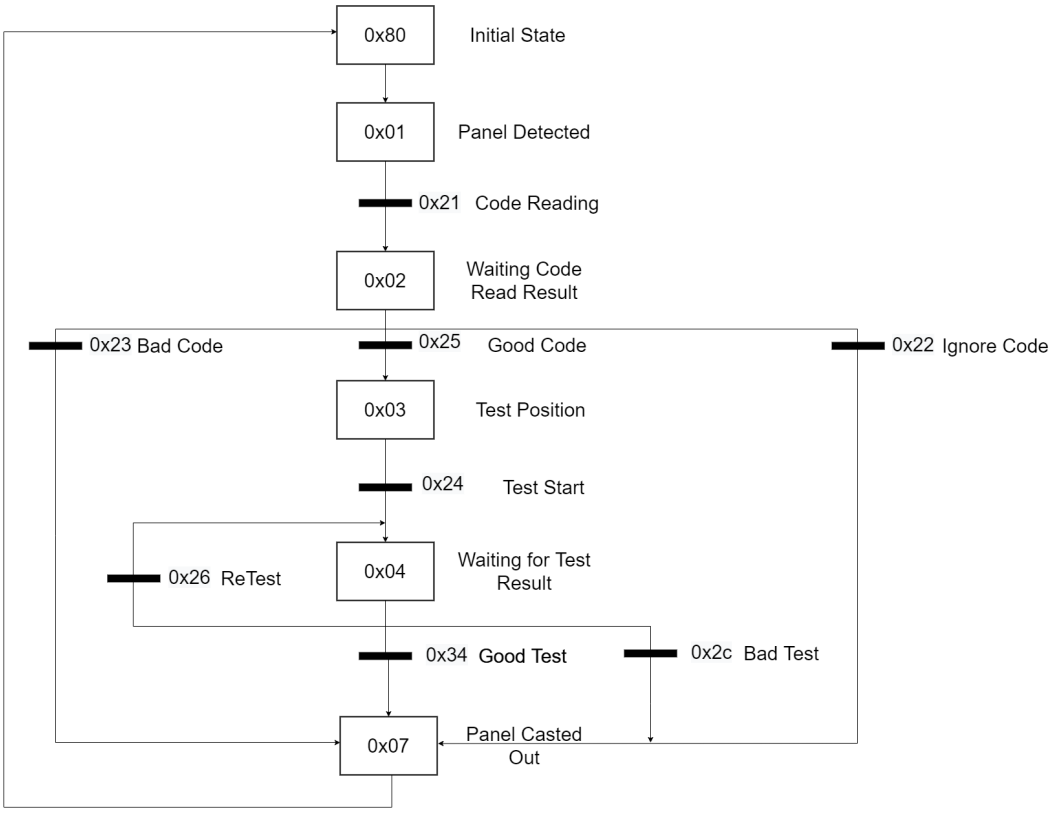


Figure 3.5: ICT module PLC Protocol

As it was mention before of the main requirements of this project is to make a application that does not depend on the database internal structure. The solution to this problem was to develop small external application with the purpose of interact with the Db. In the Fig.3.6 is possible to observe the interactions between the main program (ICT_ILF_Manager) and the smaller ones.

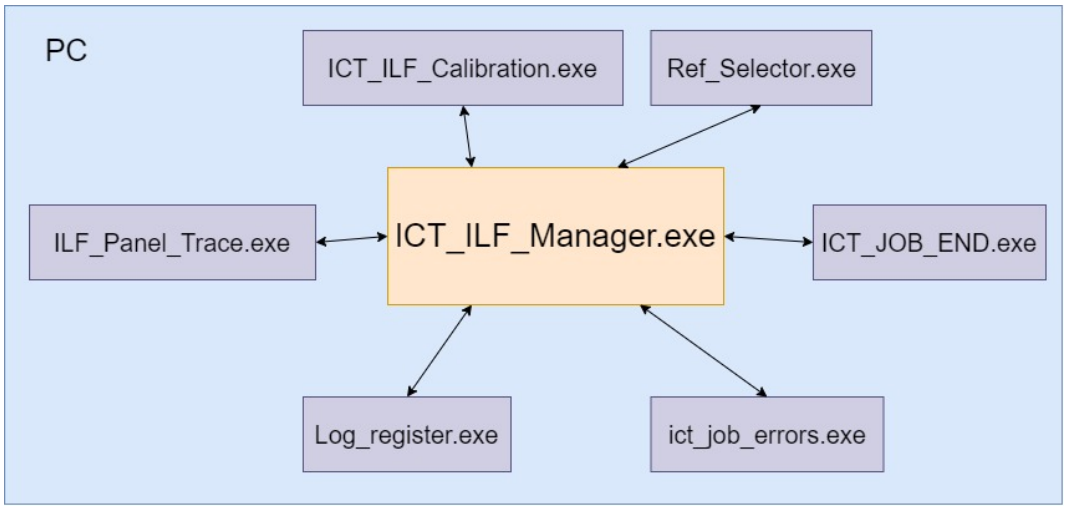


Figure 3.6: ICT_ILF_Manage Interaction with external Apps

- *ICT ILF Calibration.exe*: All the workstations at Preh need to be calibrated at least once every six months. This executable is used to query the Db about the workstation calibration status. It receives the workstation ID as an input. This code already existed on Preh it just needed to be upgraded to write the result to a xml file;

- *Ref Selector.exe*: This program is used to choose the project and the references that will be tested in the InLine Flash process;

- *ICT_JOB_END.exe*: This program is used to execute the *JobEnd* stored procedure, It receives as an input the project name, the *JobID* and the result of the *Job*. After receiving these parameters it connects to the project database and closes the *Job* by assigning the corresponding result;

- *ict_job_errors.exe*: This executable is only used if a panel fails the electric test and has the function of writing the components that failed the test and the measured values to the project database. It receives as an input the project name, the *JobID*, the component name, the measured value, and the high and low limits the value should be in;

- *Log_register.exe*: This executable has the purpose of writing to the database all the changes a user makes in the ICT-ILF Manager. It receives only two parameters, the *UserId* and a description of the operation done by the user. This program is called every time a new user logs IN or logs OUT, and every time a user changes any of the ICT-ILF configurations;

- *ILF_Panel_Trace.exe*: This software is very important for the proper functioning of the ICT-ILF Manager because it checks if the Panel has passed through all the process needed so that it can enter the ICT module. It also does a big part of the traceability of the panel, as it will be explained further on this document. This code already existed on Preh it just needed to be upgraded to add a new feature.

3.2.1 ILF-Panel-Trace

As it as mentioned before this program already existed at Preh but there was the needed to implement a new feature. The new features consists on checking which side of the Panel should be facing up when entering the ICT module. In Fig.3.7 it is shown a black box model of the program.

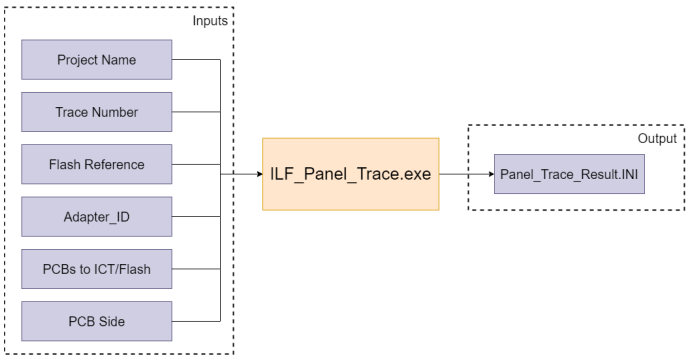


Figure 3.7: ILF-Panel-Trace block diagram

By observing the image it is possible to identify the parameters that need to be passed to the ILF_Panel_Trace:

- *Project Name*: indicates the project name so that the program connects to the corresponding database.
- *Trace Number*: indicates the TNr of the panel to be analysed;
- *Flash Reference*: indicates the reference to be applied to the panel;
- *Adapter ID*: identifies the workstation that the panel will enter;
- *PCBs to ICT/Flash*: identifies which individual PCBs needs to be tested or flashed;
- *PCB Side*: contains information about the panel side that is facing up at the moment: T for Top and B for Bottom;

After receiving the input parameters the ILF_Panel_Trace executes the procedure described in the flowchart at fig.3.8. The first step it does is checking if the panel has passed the previous process, the AOI inspection. If it has not passed it displays a Popup message to inform the operator and generates the result file. In case the panel has passed the AOI then it will be executed the *AssignRef* store procedure. This procedure will assign the Flash reference to the panel.

After it is assign it will execute the *CheckTnr* store procedure to verify if the panel as all the necessary conditions to enter the ICT or the Flash module. If all the previous conditions were successfully executed then the ILF_Panel_Trace will execute the *JobStart* procedure where it will assign one *JobID* to every individual PCB on the Panel.

Finally the ILF_Panel_Trace will verify if the current side of the panel is correct, if it is then a result file will be generated and the program will end. If the side is not correct then a message will popup informing the operator to flip the panel. On fig.3.9 there is an example of the results file.

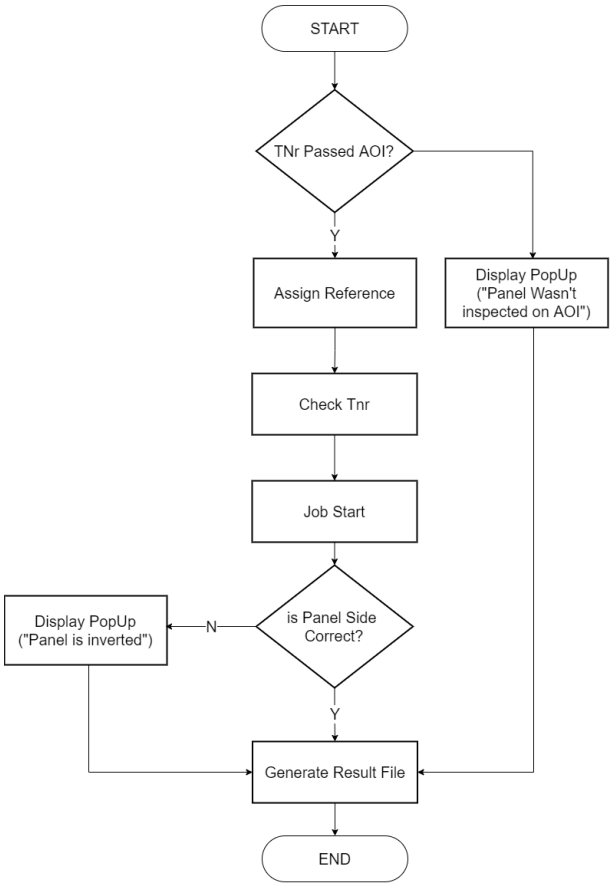


Figure 3.8: ILF-Panel-Trace Flowchart

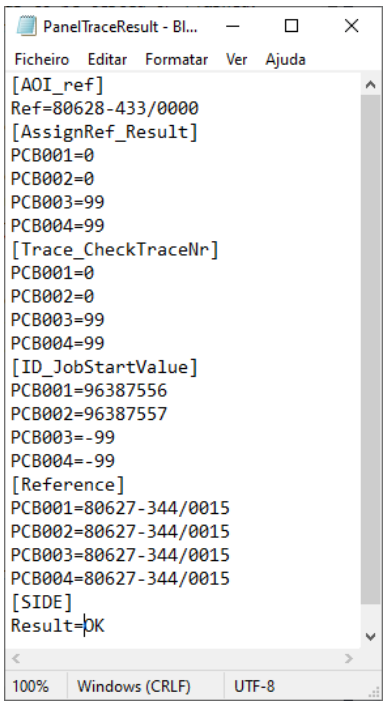


Figure 3.9: ILF-Panel-Trace Result File

3.2.2 ICT_ILF_Manager

In order to make a faster and more efficient windows application, the ICT_ILF_Manager was divides into three main Threads. The first Thread is dedicated to the graphical interface and it is always running in loop to ensure that the graphic content is always being updated. The most important method of these thread is the Start Button click, which is presented at the fig.3.10.

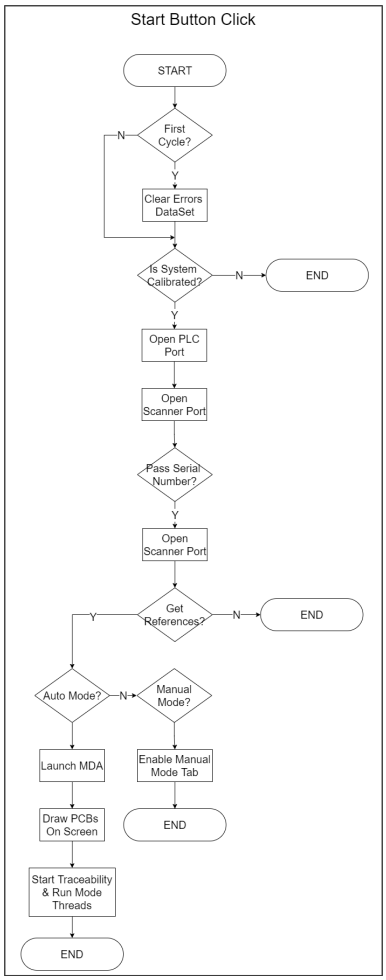


Figure 3.10: Start Click Flowchart

The second thread is dedicated to do the traceability of the panel to be tested, the flowchart of this Thread corresponds to the fig.**??**. On this thread the scanner is always being triggered until it reads a valid TNr. After reading the TNr it does the operation *Check TNr for ICT* which executes the ILF_Panel_Trace executable for the ICT module. If this validation fails then it will execute the ILF_Panel_Trace for the XILS600 and the ILS modules, this is done because even though the *Check TNr for ICT* failed the panel may still need to be Flashed.

This Thread uses a concept of four flags:

- *Good TNr Flag*: is set if the panel is good to enter the ICT module.

- *Bad TNr Flag*: if is set it means the panel can not enter any of the ICT or Flash modules and needs to be cast out to the *Unloader* module.
- *inverted panel*: this flags indicates if the panel is inverted.
- *Flash Only Flag*: this flag indicates the panel needs to skip the ICT module and enter the Flash.

After the panel leaves the Pre-Conveyor it is done a full reset of these flags, so that the next panel can enter. This traceability Thread is important because it allows the ICT_ILF_Manager to do the panel traceability of a panel while the previous panel is being tested on the ICT module.

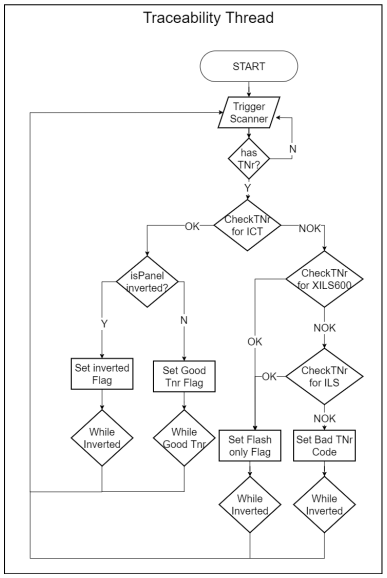


Figure 3.11: Traceability Thread Flowchart

The third, and last Thread corresponds to the automatic run mode, the flowchart of this thread is presented at fig.3.12. These Thread is responsible for communicating with the PLC and the Checksum Electric Test app. The first thing this thread does is verify if exists a panel on the Pre-Conveyor module. If it does, it will trigger the scanner in order to verify if the TNr is equal to the one that the traceability Thread validated.

After these verification's the thread has a different behaviour accordingly to the traceability flag that is activated:

- If the *good Tnr flag* is activated the ICT_ILF_Manager will send a command to the PLC to bring the panel to the test position. When the panel arrives the ICT_ILF_Manager will give an order for the Checksum app to perform the electric test. After the test is concluded the ICT_ILF_Manager will get the result and send it to the PLC. Finally, the *JobEnd* stored procedure is executed to write the test result on the project database.
- If the *bad Tnr flag* is activated the ICT_ILF_Manager will send a command to the PLC to cast out the panel.

- If the *inverted flag* is activated the operator will get a message pop up to flip the panel. After the operator acknowledge the action the program will execute a new traceability to check if everything is OK.
- If the *Flash Only flag* is activated the CT_ILF_Manager will send a command to the PLC to ignore the panel and send it to the next station.

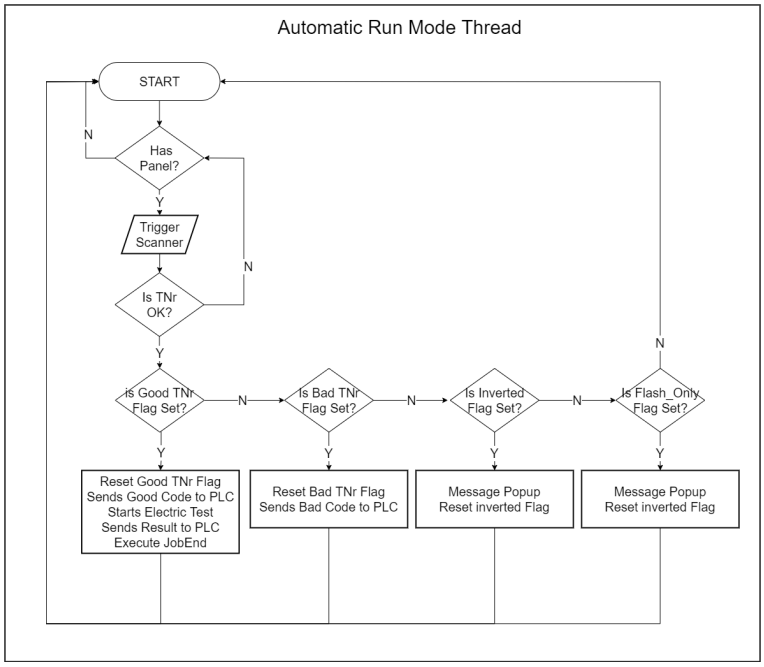


Figure 3.12: Automatic Run Mode Thread Flowchart

Chapter 4

Results

4.1 Aplication ICT-ILF Manger

4.1.1 Authentication

The authentication is an important factor to identify the user and verify the access to the application functionalities. The user identification uses only two fields, the operator control number, which consists of the company employee number, and a password. The Fig.4.1 illustrates the user authentication interface.

Figure 4.1: Login Window

After a successful authentication the user has access to the application functionalities depending on its authorization level. There are three different levels of access:

- *Administrator*;
- *Technician*;
- *Basic*;

The Administrator is the highest level of access and can use all of the functionalities and change all of the configurations. The Technician also as access to all of the functionalities but cannot change the configurations. The Basic user is the lowest level and can only start the application.

4.1.1.1 Main Windows

After the authentication procedure the ICT-ILF Manager opens its main window as it is shown in fig.4.2.

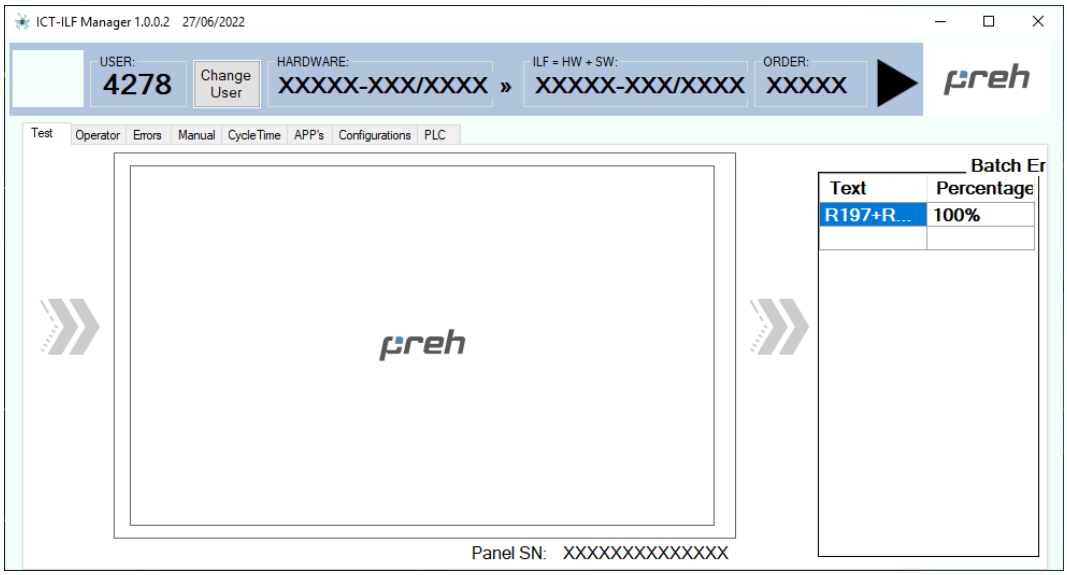


Figure 4.2: Main/Start Window

On this window the user can find the find the following information: the user control number, the Hardware reference, the ILF Reference, the production order and a list of the Batch errors. Most of these fields will be blank or undefined because the user still needs to start the production process by clicking on the start button.

4.1.2 Start process

After cliking on the start button the user needs to manually reset the PLC on the ICT machine. This procedure is mandatory because the PLC only transmits the bed of nails ID once on the startup. The message at fig.4.3 will popup to ensure the operator preforms the reset.

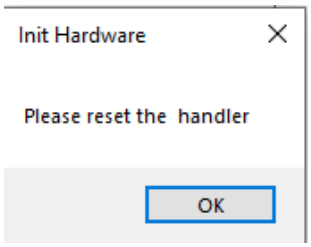


Figure 4.3: Reset PLC message

Once the reset is done the ICT-ILF Manger will generate the ref_sel_conf.xml configuration file, fig.4.4 which contain all the necessary information to launch the Ref_Selector program. Every element of the xml file contains important information about different aspects.

ref_sel_conf - Bloco de notas

Ficheiro Editar Formatar Ver Ajuda

```
<?xml version="1.0" encoding="utf-8"?>
<config>
  <LINE>
    <SPEC_FILES>C:\CheckSum\specfile\TestStand</SPEC_FILES>
    <HW_PROGRAM>YES</HW_PROGRAM>
    <ICT_CONTROL>YES</ICT_CONTROL>
    <MANUAL_HW>NO</MANUAL_HW>
        <ID>4</ID>
  </LINE>
  <HW_PROGRAM>
    <EXE_PATH>C:\CONTROLAR\GET_SAP_REF\GET_SW_FromSAP.exe</EXE_PATH>
    <RESULT_FILE>C:\CONTROLAR\GET_SAP_REF\SW_From_SAP.txt</RESULT_FILE>
    <FILTER>8062</FILTER>
  </HW_PROGRAM>
  <ICT_CONTROL>
    <EXE_NAME>ictseqval_cli.exe</EXE_NAME>
    <EXE_PATH>C:\CONTROLAR\ICTSequenceValidator\</EXE_PATH>
    <BAT_FILE>C:\CONTROLAR\ICTSequenceValidator\my_exp.bat</BAT_FILE>
    <RESULT_FILE>C:\CONTROLAR\ICTSequenceValidator\result.ini</RESULT_FILE>
  </ICT_CONTROL>
</config>
```

Ln 21, Col 10 100% Windows (CRLF) UTF-8 com BOM

Figure 4.4: Reference Selector configurations file

- *SPEC_FILES*: Contains the path to the folder where the ICT electric test sequences are stored.

- *HW_PROGRAM*: is a flag that indicates if the Ref_selctor should use the SAP software to relate the Flash reference with the ICT reference.

- *ICT_CONTROL*: indicates if the local ICT electric test sequence file needs to be compared with a control sequence file stored on the company network.

- *MANUAL_HW*: indicates if the association between ict reference and flash reference will be done manually by the user or automatically using the "*HW_Program*".
- *ID*: identifies the line where the ICT-ILF Manager is running.
- *HW_Program*: this node contains information about the "*HW_Program*" such as the path to the executable, the path to the result file and the filter to be applied.
- *ICT_CONTROL*: this node contains information about the "*ICT_CONTROL*" such as the path to the executable and the path to the result file.

After the *"ref_sel_conf.xml"* is generated the *"Ref_Selector.exe"* will launch, fig.4.5. In this window the user will need to select the operation mode, either *"Auto"* or *"Manual"*, the Project and the flash reference that will be produced. The Project combo box will be filled automatically with the projects that are compatible with the current bed of nails.

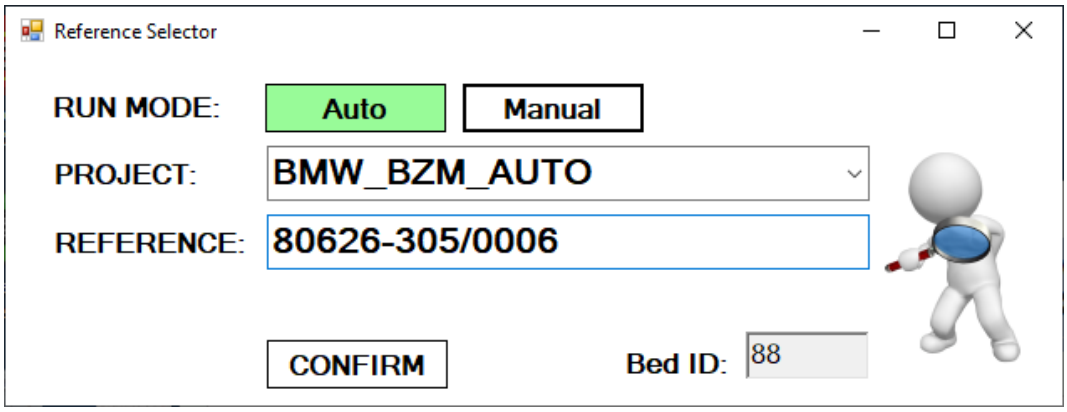


Figure 4.5: Reference Selector Window

Right after the user fill all the parameters the *"Ref_Selector"* will generate a results file, *"result_ref.xml"*, so that the *"ICT-ILF Manager"* can get the product information.

The results file, fig.4.6, contains the following information:

- *flash_ref*: indicates the Flash reference selected by the operator.
- *ict_ref*: indicates the ICT reference associated with the selected Flash reference.
- *ict_validator*: indicates if the ICT sequence file passed the "*ICT_CONTROL*" validation
- *project*: contains the selected project
- *ICTadapter*: every ICT equipment has a unique ID associated to the references, this information is needed to execute the *"Check trace number"* store procedure.
- *pcbs*: indicates which pcbs in the panel needs to be tested by the ICT, an one means it has to tested and a zero means it does not.

- *side*: indicates the side that needs to be facing up when the panel enters the ICT module, B means bottom and T means Top.
- *smd_panel_ref*: contains the panel reference, this information is only used to get the panel picture that will be displayed on the main window.
- *run_mode*: indicates the operation mode.
- *XILS600adapter*: contains the *XILS600* unique id, if the production line does not have a *XILS600* module them a zero will be passed.
- *FLASHadapter*: contains the Flash module unique identifier.
- *FLASHpcbs*: indicates which PCBs need to be flashed, this information is important because in some projects there are *"couples"* of PCBs, this is, two individual PCBs that will be assembled on the same final piece and only one has a microcontroller that requires programming.
- *BONEID*: contains the bed of nails unique identifier mounted on the Flash or *XILS600*.
- *RUN_HW*: indicates if the "*HW_Program*" was successfully executed.

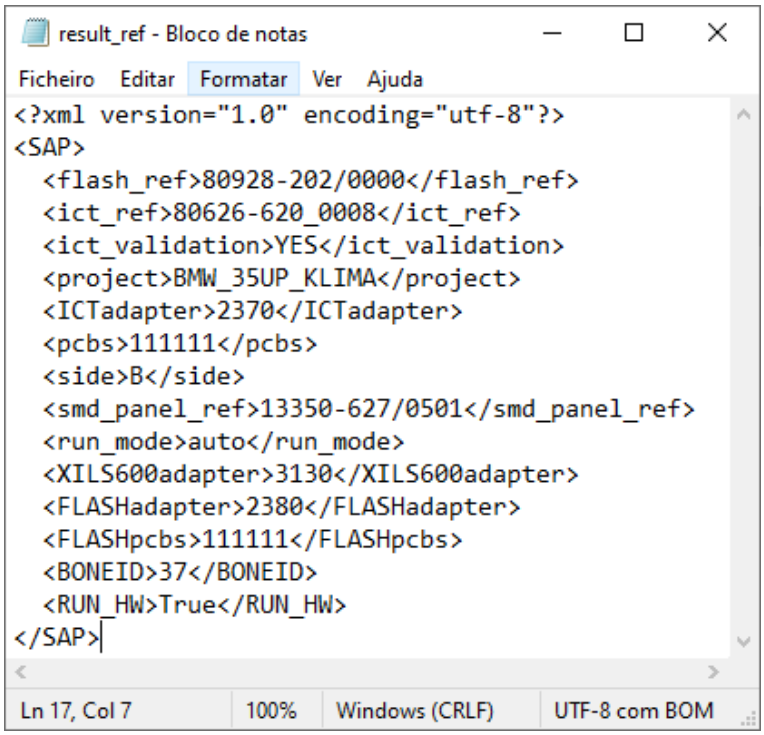


Figure 4.6: Reference Selector results file

Once the ICT-ILF Manager loads the *"result_ref.xml"* file, the user will be redirected for the main window again, but this time all the information will be loaded on the screen, as can be seen

on fig.4.7. This time a panel picture will appear on the screen with all the single PCBs properly identified.

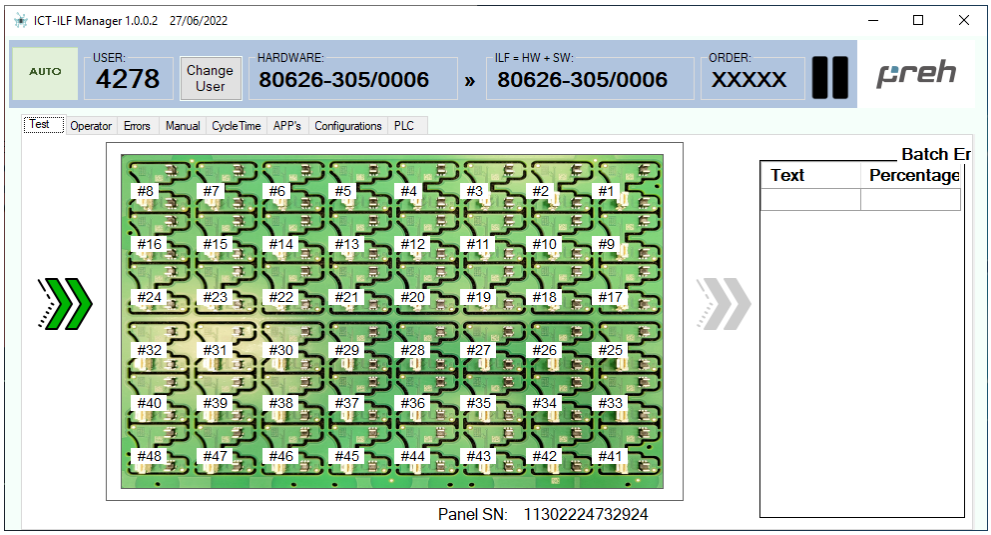


Figure 4.7: Main Window after running the Ref_Selector program

At this moment the ICT-ILF Manager is ready to start testing in series, all the user needs to do is load the panel magazines into the loader module.

While the line is working the interface changes accordingly to the test stages. This is, when a panel is going to enter the ICT module the left arrow on the screen turns green to indicate the panel entrance, and when the panel arrives at the test position the arrow turns grey again. The same happens to the right arrow when the panel is leaving the ICT module. While the ICT is performing the eletric test the boxes that identify the individual PCBs turn yellow to indicate that the PCB is being tested or black if the PCB is being skipped, fig.4.8. At the end of the test the PCBs that passed will turn green, and the PCBs that failed will turn red.

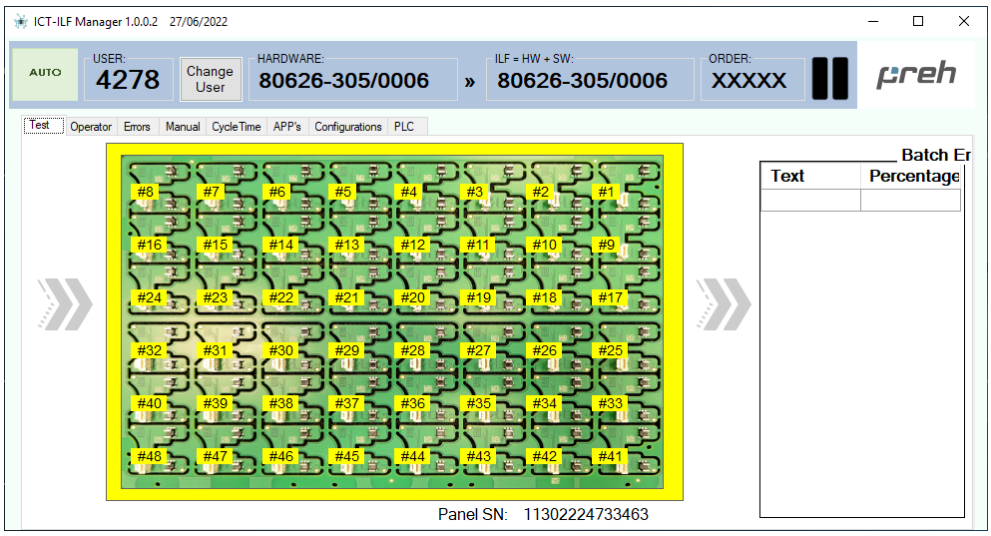


Figure 4.8: Main Window while testing Panel

4.1.3 Operator Tab

On this tab, the operator can consult which panels have failed the ICT test, as it is shown on fig.4.9. This functionality is one of the most important ones, because it is where the user can see when and why a PCB or Panel have not passed the electric test. The operator can see the following information:

- Trace Number of the panel that failed;
- Cavity/PCB that failed;
- Date and hour of the occurrence;
- By clicking on the failed occurrence a popup will load with details about which components have failed. This information contains the value measured and the range of values it should be in.

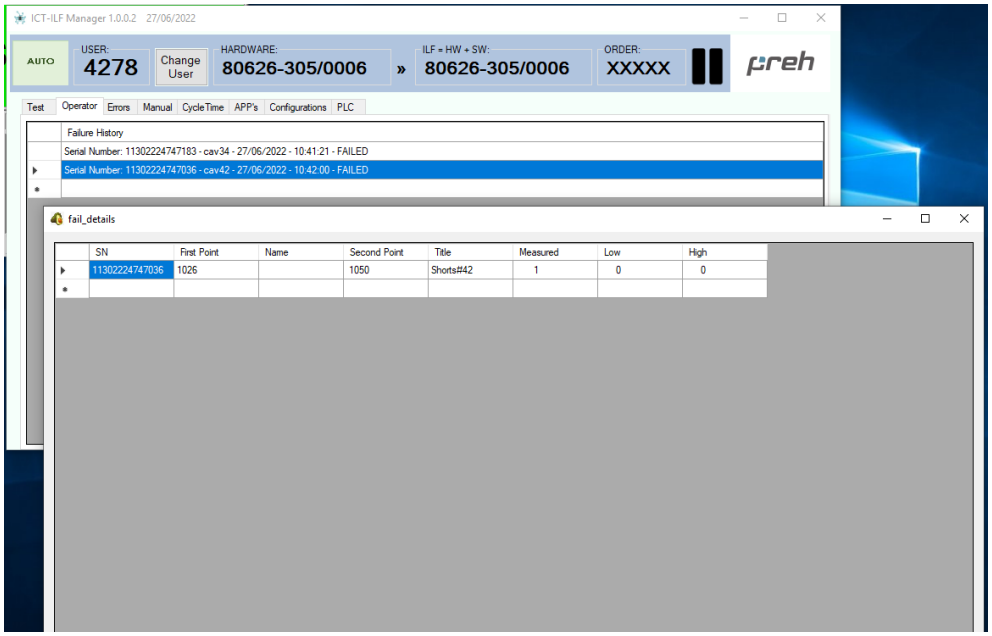


Figure 4.9: Operator Tab and Error details popup

This tab is very important feature because with the old testing software the operators had to go to a isolated workstation called *"analysis station"* and insert the Panel trace number to get the error details. Now, they can see it faster and save some time to do other tasks.

4.1.4 Errors Tab

The Errors window shows all the ICT errors detected since the program is runnig. It shows the following information:

- Trace Nember of the panel;

- Frist test point;
- Second test point;
- Name and Title, usually it is the same, but depends on what is inserted on the ict sequence file, it identifies what type o compenet has failed (resistor, capacitor, etc...);
- The measured value;
- Lowest and highest value to pass the test;

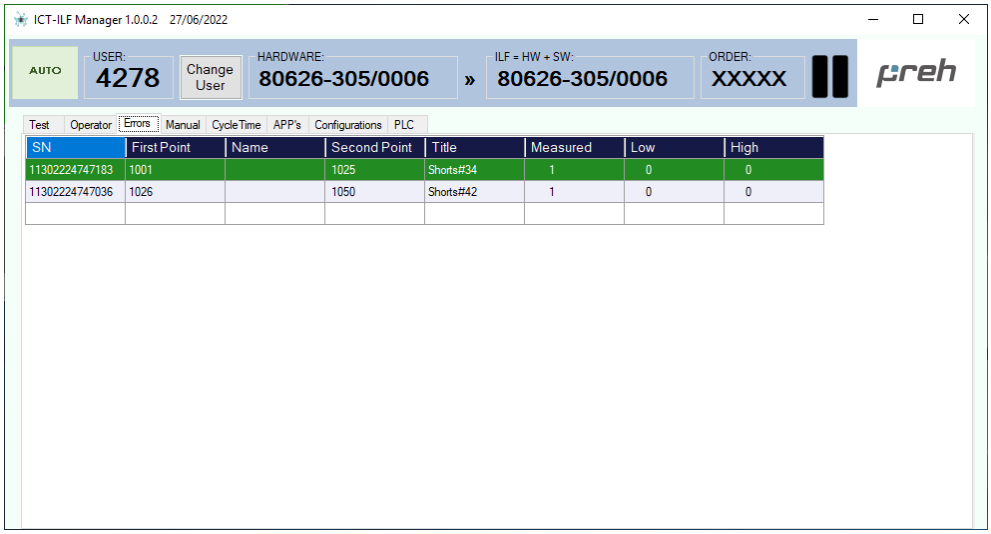


Figure 4.10: Errors Window

4.1.5 Manual Mode

The manual mode is only enabled when the user selects the manual operation mode at the application start up, it was built with the purpose of testing a panel without using traceability. This is important for when a new project enters production and it is necessary to preform various tests on a panel to validate the ict sequence test. This mode is also used when the operators need to calibrate the bed of nails, or change some of needles. The main reason to change the needles is because with time needles tend to get crooked and it generates bad contacts.

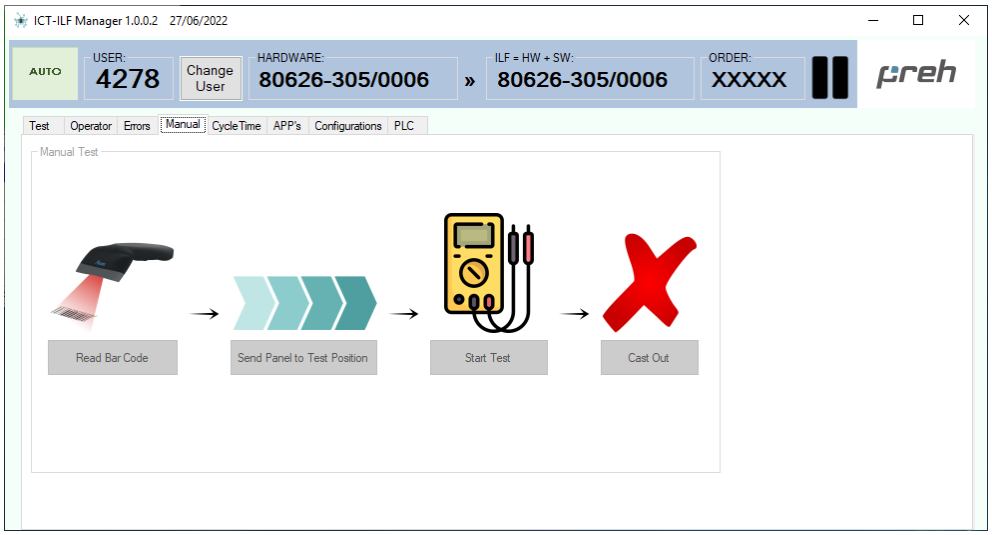


Figure 4.11: Manual Mode Window

As it is shown on fig.4.11, on this window there are four buttons:

- *Read Bar Code*: Reads the trace number of a panel and shows it on the screen;
- *Send panel to test position*: As the name indicates it brings the panel from the pre-conveyor to the ICT module;
- *Start Test*: Starts the electric test;
- *Cast Out*: Expels the panel to the unloader module;

4.1.6 Cycle Time Window

On this window, the user can consult some statistics about the line performance, as it can be seen on fig.4.12 this window contains:

- Panel and Individual PCBs stats: In these group boxes is displayed the number of panels/PCBs that passed the ICT test, the number of panels/PCBs that failed, and the total. This information helps to determine if the product is failing too much and something needs to be done, or if it is all OK.
- Average Process Time: This indicator measures the time, in seconds, a panel takes trough the process from the moment it enters the ICT until the moment it leaves. This factor helps to determine how much time the product needs to be properly tested on the ICT-ILF Line.
- Average Cycle Time: This indicator measures how much time, in seconds, two consecutive panels take to enter the ICT module. This is, when a panel enters the ICT module a timer is started and only when the next panel arrives at the ICT module the timer ends. This indicator is important because it helps identifying some production problems. These are main reasons for a high cycle time:

1. Flash is slow: if the average process time of the flash module is higher than the ICT average process time then the average cycle time of the ICT will be higher. This is because a panel that has already been tested will have to wait for the previous panel to leave the flash module.

2. Operators fault: if the operator is distracted and does not add a new magazine to the ICT line when the previous magazine is empty.

3. Slow AOI process: If the panels inspection, on the AOI process, takes longer than the ICT Process time then the ICT station will have to wait for the arrival of the panels. And by consequnce the cycle average time will increase.

- OEEs group: This indicator is not being used at the moment and was only implemented thinking on the future. When the MES is implemented it will be possible for the ICT station to retrieve the OEE information from the company system.

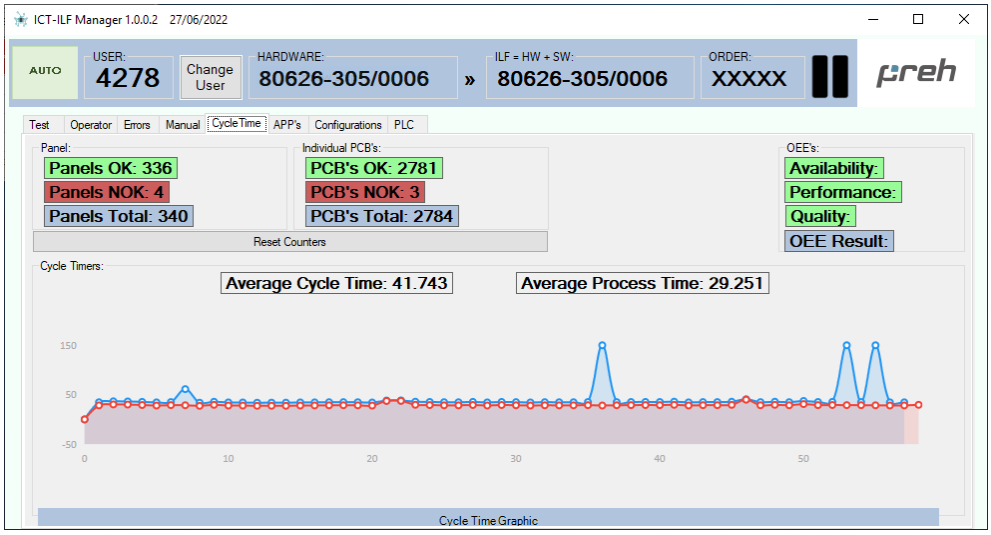


Figure 4.12: Cycle Time Window

4.1.6.1 Apps Window

In fig.4.13, it is visible the external apps window. This window is used only to launch some external applications that are used by the operators. This windows go out of this document specter but is a important feature because it makes the operators work more fluid and centralized.

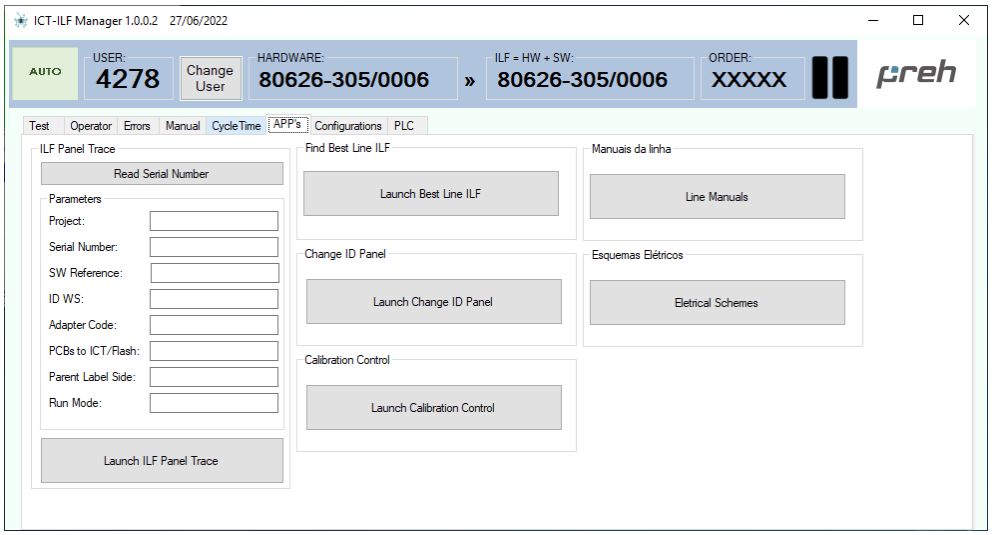


Figure 4.13: External Apps Window

4.1.7 Configurations Tab

The configurations window, in fig.4.14, can only be accessed by users with the Technician or Administrator security level. However, the Technician level can only change the Line Configuration group and the Administrator can change all the settings.

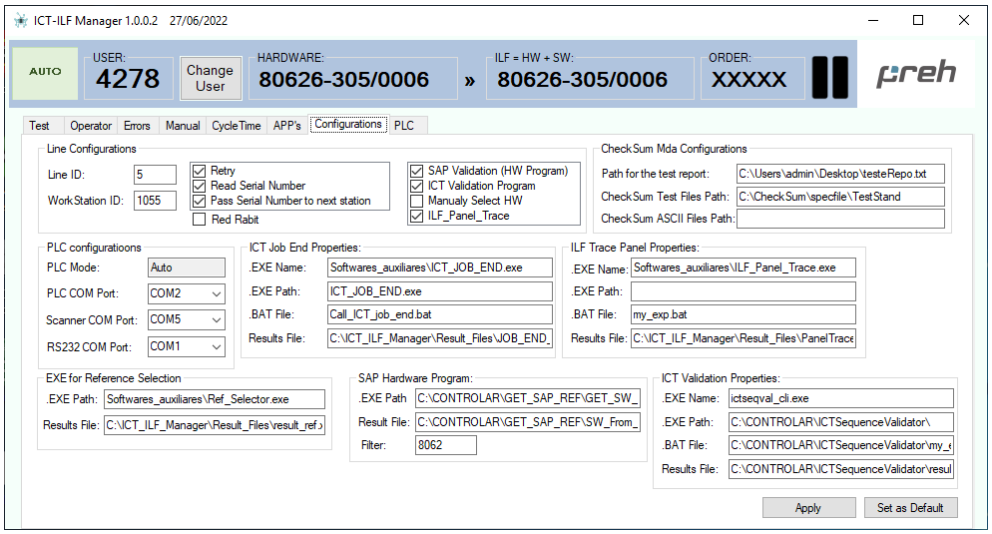


Figure 4.14: Configuration Window

- Line Configurations group: On this group the user can change line and the workstation ID, and can also enable/disable the validations that the ICL-ILF Manager must do. These configurations need to access by the Technician security level because different projects have different validations.
- PLC Configuration group: On this group the user can change all the communication ports used by the software. The RS232 COM port refers to the serial port used by the Flash

module to receive the Trace number from the ICT module. This port is only used for the lines where does not exist a middle conveyor between the ICT and the Flash modules.

- All the other group are used to set the configurations of the support software, such as executable path and result files path.

All the configurations are loaded from a local XML called *"Line_Config.xml"*, at the start of the ICT_ILF_Manager. The appendix B show an example of that file.

4.2 Tests and software validation

In order to validate the ICT-ILF Manager was elaborated a list of tests. This list had the purpose of trying to find bugs and test how the application would react to special conditions. In the table 4.1 is described the list of tests performed. This list was elaborated with my Preh advisor, engineer Armindo Rocha, and with the technician of the ICT station, Pedro Pereira.

Table 4.1: List of tests table

Nº	Description	Result
1	Fail reading Panel Trace number	OK
2	Fail ICT Panel test	OK
3	Panel without AOI verification	OK
4	Inverted Panel	OK
5	Test transmission of the TNr to Flash module	OK
6	Panel with different ICT reference	OK
7	Panel outside the production sequence	OK
8	Change panel after the traceability was done	OK
9	Verify if software does the panel retest	OK
10	Panel only to Flash	OK
11	Simulate a panel that fails on all the PCBs electric test	OK
12	Simulate a panel where not all the PCBs need to be Flashed	OK
13	Simulate a panel with a high number of PCBs	OK
14	Disable the hardware control program	OK
15	Disable the ICT Validator program	OK
16	Test user level access	OK
17	Verify application logs on the Db	OK
18	Simulate a Flash reference with an old index	OK
19	Change Reference and User	OK
20	Verify machine in emergency mode	OK
21	Test changes on the configurations tab	OK
22	Redo all the test on every production lines	OK
23	Test ICT-ILF Manager when ICT station needs to be calibrated	OK
24	Test if panel is not well placed on test position	OK
25	Interrupt cycle when ICT is testing	OK
26	Test user interface graphical changes	OK
27	Test the TNr screen indication	OK

4.3 Conclusion

On this chapter all windows and corresponding features of the ICT-ILF Manager were described. It was also shown how the validation of the software was done. For a deeper analysis of the correct procedures to navigate the application consult the appendix A.

Chapter 5

Conclusions and future work

5.1 Conclusions

With this document, it is possible to conclude that the work done in this project is of great importance to the company (preh), given that it replaced some old and outdated software with a new well-structured application capable of being built on top of, i.e, if in the future there is the need to create and add more features to the software it is easier to develop new modules compatible with this new application. This project comes exactly from the Preh need of a software capable of being upgraded at any time. This necessity comes from the fact that the company will upgrade their plant floor software to a new manufacturing execution systems (MES). And in the future may appear the necessity to add new features to the software.

The ICT_ILF_Manger was installed in four out of five InLine Flash stations. The software is installed on the Preh network and the different lines can all run the same executable. This as done so that when there is a new version of the software the developer only needs to alter the network folder, instead of having to manually replace the software in each line it is implemented.

The software has three different levels of access, it is up to the Administrator to assign the necessary permissions to each user. Each user is to blame for any damage caused by changes in the software configurations.

The full source code of this project can be found here: https://drive.google.com/drive/folders/1XPXVBHRTlNHMxo-nXWxizPFLRParmf5Z?usp=sharing

5.2 Future Work

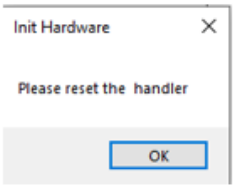
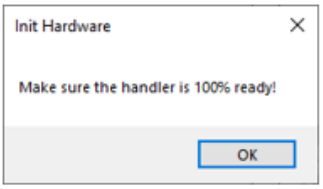
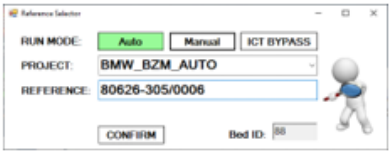
The software was successfully developed to achieve all the impose requirements. Despite these success the software should continue to improve in order to adapt to the company needs. A new feature that could be interesting is the implementation of a MQTT broker that sends the production information to a cloud server. This way it would b possible to have real time data about every production line on a single cloud.

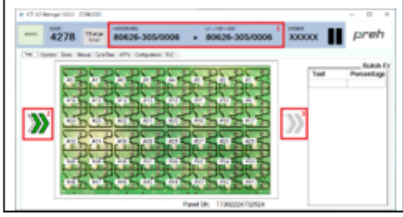
Appendix A

Procedure Manual

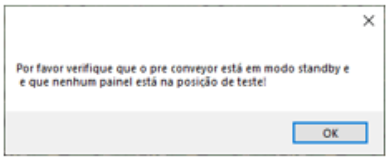
The Procedure Manual is in Portuguese because it was made for the Preh Portugal employees with the purpose of helping them to work with the application.

Procedimento para iniciar o ICT_ILF_Manager em modo Automático.

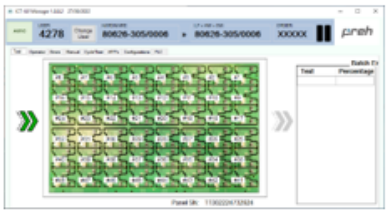
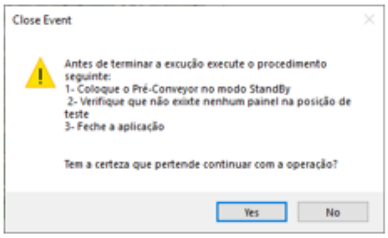
Passo 1: 	1- Efetuar o Login com o número de controlo e respetiva password.
Passo 2: 	2- O operador deverá verificar o seu Numero de Controlo na "GroupBox" destacada na imagem 3- Pressionar o botão de Start para iniciar a produção.
Passo 3: 	4- Clicar no OK e proceder ao reset do PLC 4.1- Garantir uma transação de "security" OFF para ON 4.2- Garantir uma transação do modo Manual para o modo AUTO
Passo 4: 	5- Esperar que a luz verde do ICT acenda 6- Clicar no Botão OK
Passo 5: 	7- confirmar o Id da cama de agulhas detetada pela aplicação. 8- Proceder á escolha do Projeto e da Referência que será produzida, assim como o modo de funcionamento a adotar. 9- Selecionar o botão "CONFIRM" para confirmar a escolha.

Passo 6: 	10- Verificar se as referências de Hardware e ILF estão corretas. **Notas:** A partir deste momento o modo de teste automático é ativado e marca o inicio da produção. Neste menu é também visível duas setas que indicam se o painel a ser testado está a entrar ou a sair do ICT

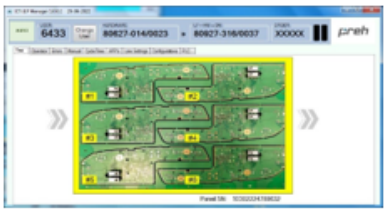
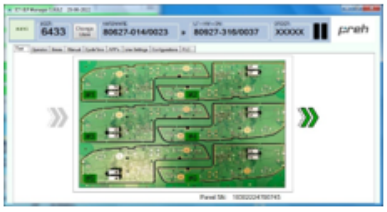
Procedimento para mudança de referência

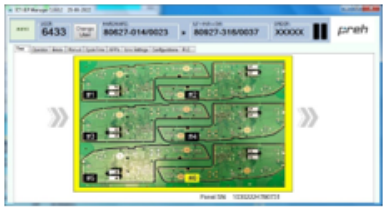
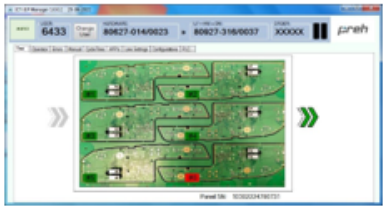
Passo 1: 	1- Clicar no botão de Pausa **Notas:** Deve colocar o Pre-Convevor no modo standBy e verificar que não existe nenhum painel na posição de teste
Passo 2: 	2- Confirmar que o procedimento anterior foi efetuado corretamente. 3- Clicar no botão "OK"
Passo3: 	4- Efetuar o mesmo procedimento utilizado para iniciar o ICT_ILF_Manager em modo Automático a partir do passo 2.

Procedimento para encerrar aplicação:

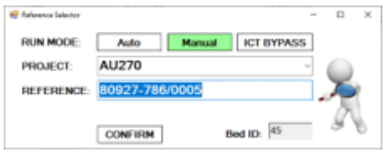
Passo 1: 	1- Colocar o Pré-Conveyor em modo StandBy 2- Verificar que não existe nenhum, painel na posição de teste 3- Fechar a aplicação
Passo 2: 	4- Confirmar que o passo anterior foi efetuado corretamente 5- Clicar no botão "YES" se pretender fechar a aplicação, ou clicar no botão "NO" se o passo anterior não foi executado corretamente. Notas: Se selecionar o botão "NO" a aplicação irá continuar com o funcionamento normal.

Estados do Painel na situação de teste

Testing State: 	A cor amarela significa que o painel está a ser sujeito ao teste elétrico.
Good Test 	A cor verde identifica os PCBs que passaram no teste elétrico

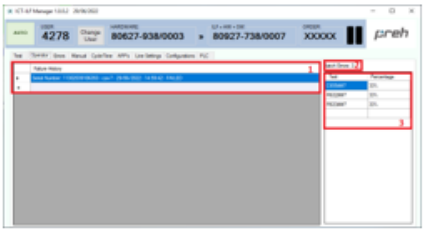
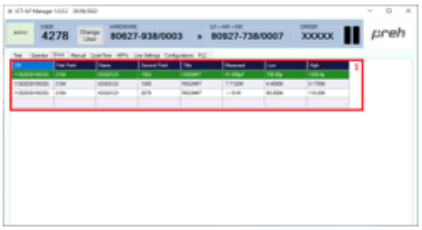
Skip State: 	A cor preta representa os PCBs que não foram sujeitados ao teste elétrico (deram Skip).
Failed Test: 	A cor vermelha representa os PCBs que falharam no teste elétrico.

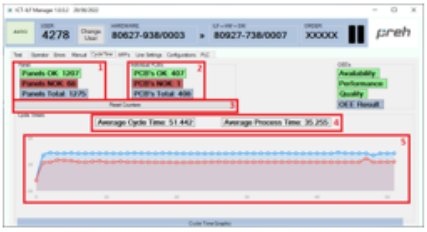
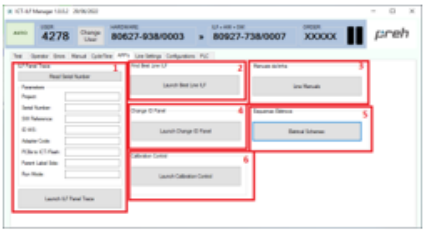
Procedimento Para Inicializar a aplicação em modo manual

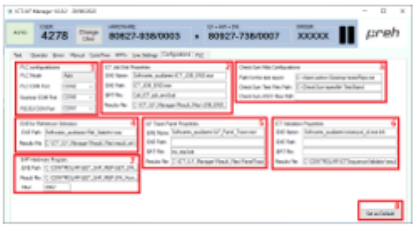
Passo 1: 	1- Pressionar o botão de Start para iniciar a produção.
Passo 2: 	2- No campo "RUN MODE" escolher a opção "Manual" 3- Nos campos "PROJECT" e "REFERENCE" escolher a base de dados e a referência pretendidas 4- Clicar no botão de confirmação ("CONFIRM") **Notas:** Antes de proceder á seleção da referência deve sempre verificar se o ID da cama de agulhas (campo "Bed ID") está correto!
Passo 3: 	5- No ICT_ILF_Manager selecionar a Tab "Manual" 6- Colocar o painel a ser testado no Pré-Convevor 7- Clicar no botão "Read Bar Code" para proceder á leitura do Trace Number do painel Notas: Apenas utilizadores com nível de acesso igual ou superior a 2 conseguem aceder ao modo Manual
Passo 4: 	8- Confirmar o Trace Number e clicar no botão com o icon "correto" Notas: Caso não seja o painel correto deve clicar no botão referente ao código errado e repetir o passo anterior para o painel certo.

Passo 5: 	9- Clicar no botão "Send Panel To Test Position" para transporter o painel até á posição de teste Notas: O operador deverá verificar que o transporte do painel foi efetuado com sucesso antes de continuar para o próximo passo. Se o painel não chegar á posição de teste, este deve ser retirado do modulo ICT e a aplicação deve ser reiniciada.
Passo 6: 	10- Clicar no botão "Start Test" para iniciar o teste eletrico dos PCBs 11- Caso seja necessário é possível efetuar um ReTeste clicando o botão "RETest" Notas: A aplicação checksum abre e seleciona a sequencia de Hardware automaticamente após clicar no botão para iniciar o teste.
Passo 7: 	12- Clicar no botão "Cast Out" para expulsar a peça 13- Repetir o processo para os restantes painéis a partir do Passo 3. Notas: Após expulsar o painel este será transportado para a magazine das "PCBs NOK".

Apresentação e descrição dos menus e funcionalidades da Aplicação:

Tab "Test": 	1- Indicação do modo de funcionamento 2- Indicação do NC do usuário & botão para mudar de utilizador 3- Indicação das referências de Hardware e da ILF (HW+SW) 4- Indicação do numero da ordem de encomenda 5- Fotografia do painel e indicação dos PCBs individuais 6- Seta indicadora de entrada de painel 7- Seta indicadora de saída de painel 8- Label com Serial Number do painel
Tab "Operador": 	1- Histórico de falhas de PCBs individuais 2- Indicação do numero de erros elétricos acumulado 3- Erros elétricos acumulados ordenados por frequência
Tab "Errors": 	1- Listagem de todos os erros ocorridos durante a execução da aplicação

Tab "Cycle Time": 	1- Indicação do numero de painéis "OK" e "NOK" 2- Indicação do numero de PCBs individuais "OK" e "NOK" 3- Botão para fazer "reset" aos contadores 4- Indicação do tempo médio de ciclo e de processo 5- Gráfico com os últimos 100 registos temporais
Tab "APPs": 	1- Parâmetros para execução singular do software ILF Panel Trace + botão para preenchimento automático dos parâmetros + botão para lançar ILF_PANEL_TRACE.EXE 2- Botão para executar a aplicação "Find Best Line ILF" 3- Hiperligação para a pasta de manuais da linha 4- Botão para lançar exe externo "Change ID Panel" 5- Hiperligação para a pasta dos esquemas elétricos do projeto 6- Botão para lançar a aplicação de calibração da estação no modo visual
Tab "Line Settings": 	1- Indicação do ID da linha e da "workstation" 2- 3- 4- Botão para aplicar as alterações efetuadas

Tab "Configurations"-	
	1- Configurações relativas ao PLC 2- Configurações relativas ao EXE ICT_Job_End 3- Configurações relativas ao CheckSum MDA 4- Configurações relativas ao EXE de escolha de referências 5- Configurações relativas ao ILF_Panel_Trace 6- Configurações do programa de validação de sequencias ICT (ictseqval_cli.exe) 7- Configurações relativas ao EXE que vai buscar as referencias ao SAP. 8- Botão para criar um novo ficheiro xml com as novas alterações.

Revision overview

Revision number	Revision date	Change	Created	Released
00		André Marques	André Marques	André Marques

Appendix B

Line_Config.xml

Figure B.1: Line Config_File.xml Configurations

References

[1] Preh group. Accessed on 17.05.2022. URL: https://www.preh.com/en/company/history.

[2] checksum LLC. Checksum analyst-ils in-line test system. pages 3–8, 2014. URL: https://checksum.com/PDFs/analyst_ils.pdf.

[3] Ict, in circuit test tutorial. Accessed on 15.02.2022. URL: https://www.electronics-notes.com/articles/test-methods/automatic-automated-test-ate/ict-in-circuit-test-what-is-primer.php.

[4] How flying probe testing works for pcb assembly. Accessed on 22.06.2022. URL: https://www.protoexpress.com/blog/how-flying-probe-testing-works-for-pcb-assembly/.

[5] Component object model (com). Accessed on 16.02.2022. URL: https://docs.microsoft.com/en-us/windows/win32/com/component-object-model--com--portal.

[6] Rs232 serial communication protocol. Accessed on 14.02.2022. URL: https://circuitdigest.com/article/rs232-serial-communication-protocol-basics-specifications.

[7] Rs232 serial communication protocol. Accessed on 14.02.2022. URL: https://www.electricaltechnology.org/2020/05/rs232-serial-communication-protocol.html.

[8] Overall equipment effectiveness. Accessed on 15.06.2022. URL: https://www.oee.com/.