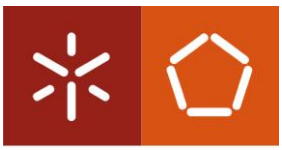
André Manuel Alves Costa

Desenvolvimento de um Sistema de Monitorização Para uma Célula de Assemblagem Automática de Centralinas



Universidade do Minho
Outubro de 2008

Desenvolvimento de um Sistema de Monitorização Para uma Célula de Assemblagem Automática de Centralinas

André Manuel Alves Costa

Tese submetida na Universidade do Minho
para a obtenção do grau de
Mestre em
Engenharia Electrónica Industrial E Computadores

Universidade do Minho
Outubro de 2008

Trabalho realizado sob a orientação do:
Prof. Jaime Francisco Cruz Fonseca

Á minha família e
minha namorada Cláudia Marlene

Agradecimentos

Ao Professor Jaime Fonseca, meu orientador nesta dissertação, pelo seu incentivo, compreensão e exigência demonstrada ao longo de todo o trabalho.

Ao amigo João Vilaça, pelo seu apoio e orientação ao longo deste trabalho, contribuindo muito para o bom andamento do trabalho.

Ao meu colega de laboratório João Borges, por toda ajuda e disponibilidade demonstrada.

Aos meus colegas de curso pela força e apoio nas horas mais difíceis.

Á empresa Silgal pela oportunidade dada.

A todo o pessoal que trabalha na Silgal pelo seu espírito de entre e ajuda e companheirismo.

Ao Eng. José Zózimo, Eng. Nuno e Eng. Gonçalves pela troca de informação, e sugestões dadas.

Á minha namorada pela paciência, compreensão e amor que me dedicou ao longo deste projecto.

Aos meus pais José e Carminda pelo apoio, incentivo e esforço concedido para que completa-se com sucesso toda a minha formação académica.

Á minha querida avó que não teve oportunidade de presenciar, presencialmente, a finalização da minha formação académica.

Ao meu irmão Nuno pela paciência e força que me concedeu.

Aos familiares e amigos não mencionados.

A Todos o meu Muito Obrigado!

Resumo

A utilização de soluções cada vez mais autónomas, faz com que uma empresa se torne mais industrializada e mais atractiva em termos de mercado. A utilização destas soluções envolve o uso da automação e deve ser acompanhada de sistemas de monitorização juntamente com sistemas de armazenamento de histórico.

Nesta dissertação, é descrito todas as fases de estudo e desenvolvimento de uma célula de assemblagem automática de centralinas para automóveis topo de gama requisitada pela empresa "DENSO" sediada em Espanha. São apresentados todos os requisitos necessários para o desenvolvimento da célula, que vão desde a fase de estudo e escolha de todos os componentes do sistema até á sua construção.

É apresentado neste documento, o desenvolvimento de um sistema que permite monitorizar todo o processo de assemblagem efectuado pela célula. Este sistema é desenvolvido em Microsoft Visual Studio 2008 sendo a sua implementação realizada em *Visual C++*.

Como funcionalidades deste sistema temos a visualização de mensagens em tempo real que ilustram o processo de assemblagem, interface para programação de novos programas de assemblagem, recepção, tratamento e processamento de tramas via porta série enviadas por um conjunto de dispositivos ligados ao sistema e armazenamento de toda a informação relativa à assemblagem de cada centralina.

Em relação ao armazenamento de todo o histórico do processo de assemblagem e reprogramação da célula são apresentadas duas soluções implementadas: a primeira passa pela utilização de ficheiros quanto, segunda resulta de uma actualização da célula e utiliza uma base de dados.

Posteriormente, foi desenvolvida uma página *Web* a qual permite a visualização de todo o conjunto de centralinas assembladas, impressão de relatórios de assemblagem em pdf, criação, edição e eliminação de programas utilizados na reprogramação da célula.

Palavras-Chave: Automação, Sistemas SCADA, Comunicação RS-232, Sistemas RFID, Base de Dados, Mysql, Php, Páginas Web, PLC's.

Abstract

The use of solutions based in automation systems, makes a company more modern and more competitive in the market.

In this thesis, is described all the steps of the development of a cell of automatic assembly for expansive cars. This work cell it was requested by the company "DENSO" based in Spain. The requirements for the development of the work cell are presented.

The development system is in Microsoft Visual Studio 2008 and has an implementation based on classes. The main features of this system are: change messages in real time to monitoring the assembly process; interface to change the assembly programs; reception and processing of frames sent by serial port by several devices; storage the information about the assembly process of each unit.

The report of the assembly process is implemented in database. A web page it was developed to consult the information in the database from a remote station.

Índice

Resumo v

Abstract vii

Lista de Tabelas xii

Lista de Figuras xiii

CAPÍTULO 1 1

Identificação do Problema 1

1.1. Introdução 1

1.2. Problema Proposto 4

1.3. Motivação do trabalho 5

1.4. Organização da Tese 6

CAPÍTULO 2 7

Célula de Assemblagem 7

2.1. Requisitos da célula a construir 7

2.2. Modo de operação da célula 11

2.3. Arquitectura da Célula de trabalho 12

2.3.1. Sistemas mecânicos 13

2.3.2. Sistemas electrónicos 17

CAPÍTULO 3 27

Aplicação desenvolvida 27

3.1. Funcionalidades 27

3.2. Interface desenvolvido 27

3.3. Comunicação com os dispositivos ligados por protocolo série 34

3.3.1. Comunicação entre o Autómato e o PC (porta série nº 1) 35

3.3.2. Comunicação entre o PC e o leitor de ID (porta série nº2) 40

3.3.3. Comunicação entre o PC e o leitor de Tag (porta série nº3) 40

3.3.4. Comunicação entre a Aparafusadora e o PC (porta série nº4) 40

3.3.5. Comunicação entre o PC e o Autómato (porta série nº5) 41

3.3.6. Comunicação entre o PC e o Robô 1 (porta série nº6) 43

3.3.7. Comunicação entre o PC e o Robô 2 (porta série nº7) 43

3.3.8. Comunicação entre o PC e a impressora (porta série nº8) 43

3.4. Algoritmos implementados 44

3.4.1. Recepção e tratamento de tramas 44

3.4.2. Algoritmo usado na activação/desactivação e leitura/escrita de zonas de memória no autómato 46

3.4.3. Algoritmo usado na implementação da rotina *timer* 47

3.4.4. Algoritmo geral da rotina "gestão de programas" 58

3.4.5. Edição Programa/Robô 1 ou Robô 2 61

3.4.6. Edição Programa/Robô 2 62

3.4.7. Criação de Novo Programa 63

3.4.8. Mudança de Programa 66

3.5. Estrutura de Ficheiros 69

3.5.1. Estrutura do ficheiro txt de configuração inicial 69

3.5.2. Estrutura do Ficheiro txt do Programa 70

3.5.3. Estrutura do ficheiro da centralina 71

CAPÍTULO 4 75

Actualização da Célula 75

4.1 Novas funcionalidades 75

4.2. Base de dados 75

4.2.1. *Software* usado 75

4.2.2. Estrutura da base de dados desenvolvida 76

4.2.3 Descrição das tabelas usadas na base de dados 77

4.3 Página *Web* 81

4.3.1 *Software* usado 81

4.3.2. Linguagens de programação usadas 82

4.3.3. Estruturação da página *Web* 83

4.4. Rotinas de interacção entre a aplicação desenvolvida e a base de dados 91

4.4.1 Rotina que estabelece a comunicação entre a aplicação e a base de dados 91

4.4.2 Rotina usada no envio de dados para a base de dados 92

4.4.3. Rotina usada no carregamento de dados armazenados na base de dados 93

4.4.4. Rotina usada na actualização dos dados armazenados na base de dados 94

CAPÍTULO 5 95

Conclusões e Sugestões para Trabalhos Futuros 95

5.1. Conclusões 95

5.2. Sugestões para Trabalhos Futuros 97

REFERÊNCIAS 99

Anexos 101

[Anexo A] Biblioteca da Moxa 101

[Anexo B] Porta RS232 105

[Anexo C] Fluxogramas 108

Lista de Tabelas

Tabela 1.1 – Valores da capacidade do equipamento requisitados pelo cliente..... 11

Tabela 3.1 – Tabela que mostra e descreve todo o conjunto de tramas enviadas do autómato para aplicação que por sua vez faz desencadear as respectivas mensagens no mostrador de mensagens da posição 1. 36

Tabela 3.2 – Tabela que mostra e descreve todo o conjunto de tramas enviadas do autómato para aplicação durante a criação, mudança ou edição de programas..... 37

Tabela 3.3 – Tabela que mostra e descreve todo o conjunto de tramas enviadas do autómato para aplicação a qual faz desencadear as respectivas mensagens nos mostradores de estado de peça de cada posição..... 38

Tabela 3.4 – Tabela que mostra e descreve todo o conjunto de tramas enviadas do autómato para aplicação e que servem de controlo para a mesma. 39

Tabela 3.5 – Tabela que mostra e descreve todo o conjunto de tramas enviadas da aparafusadora para aplicação e que são responsáveis pela actualização de algumas das mensagens do mostrador de estado de peça localizado na posição 3. 41

Tabela 3.6- Tabela que mostra as *flags* activadas/desactivadas no autómato durante o processo de assemblagem e que são enviadas pela porta série nº5. 42

Lista de Figuras

Figura 1.1- Automação de uma linha de montagem automóvel. 1

Figura 1.2 - Diagrama simplificado de um sistema de controlo automático. 3

Figura 1.3 - Arquitectura de rede simplificada para um sistema automatizado. 3

Figura 1.4 - Célula de assemblagem de centralinas proposta pelo cliente. 4

Figura 2.1 – Formato do tabuleiro requisitado pelo cliente. 8

Figura 2.2 – Estrutura da célula requisitada pelo cliente. 9

Figura 2.3 - Constituintes da centralina. 12

Figura 2.4 - Perspectiva das diferentes posições de assemblagem da centralina. 12

Figura 2.5 - Célula de assemblagem desenvolvida. 13

Figura 2.6 - Interligação entre os vários sistemas mecânicos que fazem parte da célula. 14

Figura 2.7 - Robô 1. 15

Figura 2.8 - Colocação da tampa em cima da base pelo robô 1. 15

Figura 2.9 - Robô 2. 15

Figura 2.10 - Sequência de aparafusamento para a centralina usada, em que P1, P2, P3, e P4 representam respectivamente parafuso 1, parafuso 2, parafuso 3 e parafuso 4. 16

Figura 2.11 - Depósitos de parafusos. 16

Figura 2.12 – Sistema pneumático. 17

Figura 2.13 – Interligação entre os vários sistemas electrónicos e o computador de secretária. 17

Figura 2.14 - Sistema de visão responsável pelo processamento das regiões de gel localizadas na tampa da centralina. 18

Figura 2.15 - Tampa da centralina com as respectivas zonas de gel e posição programadas, estas zonas variam com a gama de centralina a assemblar. 19

Figura 2.16 - Modo de funcionamento do sistema de análise de gel. 20

Figura 2.17 - Sistema ZFV responsável pela análise do cordão de silicone. 21

Figura 2.18 - Base da centralina e zona do cordão de silicone inspeccionado pelo sensor de silicone..... 21

Figura 2.19 - Constituintes de um sistema RFID. 22

Figura 2.20 – Sistemas RFID utilizados na célula à direita o Omron V600 e à esquerda o Omron V700. 23

Figura 2.21 - Sensor indutivo detector de parafusos. 23

Figura 2.22 - Sensor indutivo detector de rotação de mesa. 24

Figura 2.23 - Conjunto aparafusadora controlador. 24

Figura 2.24 - Placa multiplicadora de portas série usada na ligação dos vários dispositivos ao computador via RS-232. 25

Figura 3.1 – Vista geral do ecrã de monitorização de todo o processo de assemblagem da célula. 28

Figura 3.2 – Esquema de ligação entre os vários dispositivos do sistema e o computador. 29

Figura 3.3 – Relação entre cada uma das áreas de visualização do interface e os dispositivos ligados ao computador. 30

Figura 3.4 – Mostrador de estado de aparafusamento de um conjunto de quatro parafusos, aparafusados pelo robô 2. 31

Figura 3.5 – Mostrador do tempo do cordão de silicone da peça. 33

Figura 3.6 – Esta mensagem surge no interface sempre que uma centralina é concluída e durante o seu processo de assemblagem ocorreram erros. 33

Figura 3.7 – Formato da etiqueta impressa na célula de assemblagem, podemos ver que os erros sucedidos neste caso específico foram: regiões de gel não cumpriram com os requisitos, peça foi forçada e o robô 2 efectuou um mau aparafusamento do parafuso 3..... 34

Figura 3.8 - Algoritmo usado na implementação da rotina de recepção de tramas enviadas pelos vários dispositivos ligados ao computador e recebidas na aplicação. 44

Figura 3.9 - Função responsável pela abertura e configuração da porta série. Neste caso em específico temos a configuração e activação da interrupção por eventos na porta série número 2. 45

Figura 3.11 – Esquema de ligação entre os vários dispositivos que têm interferência na gestão e visualização do *timer*. 47

Figura 3.12 – Amostragem do *timer* no interface após a recepção da trama de activação enviada pelo autómato. 49

Figura 3.13 – Activação de um segundo *timer*, neste momento existem dois *timers* activos mas só o *timer* 1 é visualizado no interface. 50

Figura 3.14 – Autómato envia á aplicação a trama a indicar que a peça deve ser retirada, a aplicação, por sua vez, apaga o *timer* actualmente em visualização e mostra o *timer* associado á nova peça que vai dar entrada na célula de trabalho. 51

Figura 3.15 – Autómato envia à aplicação a trama a indicar que as peças vão continuar o processo de assemblagem, a aplicação por sua vez continua a amostragem do respectivo *timer* mostrando também o estado da peça como “OK”. 51

Figura 3.16 – Autómato envia à aplicação a trama a indicar que a mesa vai iniciar a rotação, a aplicação continua a execução do *timer* que estava em visualização mas apaga-o do interface mostrando agora no interface o *timer* da próxima centralina que vai dar entrada na célula de trabalho. 51

Figura 3.17 – A mesa rodou correctamente e a aplicação analisou o *timer* da peça que se encontra na posição 2 verificando que cumpre os limites estabelecidos desta forma é dada a ordem para o robô 1 iniciar o trabalho e o *timer* associado á peça é eliminado. 52

Figura 3.18 – A mesa rodou correctamente e a aplicação procedeu à análise do *timer* da peça que se encontra na posição 2 verificando que não cumpre os limites estabelecidos logo o robô 1 é impedido de trabalhar e a peça é considerada estragada. 52

Figura 3.19 – Implementação da função responsável pela selecção e inicialização de um novo *timer* associando-o á nova peça. 53

Figura 3.20 – Implementação da função responsável pela contagem dos *timer* activos, e amostragem no interface do *timer* da peça que se encontra na posição 1. 55

Figura 3.21 – Conjunto de instruções executadas após a recepção da trama indicadora que a mesa vai iniciar a rotação. 56

Figura 3.22 – Conjunto de instruções executadas após a recepção da trama indicadora que a mesa terminou a rotação. 57

Figura 3.23 – Conjunto de instruções executadas após a recepção da trama indicadora que a centralina se encontra estragada e não vai avançar no processo. 58

Figura 3.24 – Algoritmo geral usado na implementação da rotina “Gestão de Programas”. 59

Figura 3.25 – Esquema que mostra a interligação e tramas trocadas entre os vários dispositivos intervenientes na gestão de programas. 60

Figura 3.26 – Ecrã apresentado na consola do autómato com a célula a funcionar em modo manual..... 60

Figura 3.27 – Neste ecrã é possível introduzir a *password* de acesso ao menu de gestão de programas. 61

Figura 3.28 – Ecrã de gestão de programas apresentado na consola do autómato, após a validação da *password.* 61

Figura 3.29 – Visualização de todo o processo de edição de programas no robô *pick and place* e no robô de aparafusamento. 62

Figura 3.30 – Edição de programa no robô de aparafusamento (robô 2). 63

Figura 3.31 - Sequência de ecrãs mostrados durante a criação de um novo programa de assemblagem..... 66

Figura 3.32 - Sequência de ecrãs mostrados durante a mudança de programa na célula. 68

Figura 3.33 – Ecrã da consola do autómato no qual é possível visualizar o estado de processo de assemblagem em modo automático, permitindo também a mudança de programa de assemblagem da mesma. 68

Figura 3.34 – Formato do ficheiro txt de configuração inicial da célula. 69

Figura 3.35 – Ficheiro txt do programa número 1. 70

Figura 3.36 – Formato do nome do ficheiro gerado após a conclusão da centralina. 72

Figura 3.37 – Ficheiro gerado na conclusão de uma centralina. 72

Figura 4.1- Ambiente gráfico do *software* usado na construção da base de dados (*phpmyadmin).* 76

Figura 4.2 – Estrutura da base de dados implementada. 76

Figura 4.3 – Implementação da base de dados “Gestão Célula”. 77

Figura 4.4 – Implementação e estrutura da tabela “Centralinas”. 77

Figura 4.5 – Implementação da tabela “Programas”. 79

Figura 4.6 – Implementação da tabela “Utilizador”..... 80

Figura 4.7 – Visualização da comunicação entre a aplicação, base de dados e página *Web*. 81

Figura 4.8 – *Software* usado no desenvolvimento da página *Web* (php *Designer* 2005). 82

Figura 4.9 – Algoritmo usado na construção de toda a estrutura da página *Web*. 84

Figura 4.10 – Interface da página "*Home*". 84

Figura 4.11 – Página que mostra todo o historial de centralinas e suas características de assemblagem..... 85

Figura 4.12 – Página de procura de centralina por número de série..... 86

Figura 4.13 – Após a selecção do ID serão mostradas todas as centralinas com o ID pretendido ou com as mesmas iniciais, neste exemplo o ID seleccionado na Figura 4.12 foi "Renault"..... 86

Figura 4.14 – Página apresentada ao utilizador após a selecção da procura de centralinas por intervalo de datas..... 87

Figura 4.15 – Página que mostra as centralinas assembladas durante o intervalo de datas seleccionado..... 87

Figura 4.16 – Relatório em pdf em que pode ser consultada toda a informação relativa às centralinas assembladas durante um período de tempo inserido pelo utilizador. 88

Figura 4.17 – Página em que é possível visualizar todo o relatório de cada ano de assemblagem da célula, para o período de tempo seleccionado. 89

Figura 4.18 – Relatório anual em pdf de centralinas assembladas durante um período de tempo inserido pelo utilizador. 89

Figura 4.19 – Página em que se pode consultar toda a lista de programas existentes para programação da célula. 90

Figura 4.20 – Página apresentada quando o utilizador solicita a criação, eliminação ou modificação de algum programa. 90

Figura 4.21 – Página que possibilita ao utilizador registado a edição de um programa existente... 90

Figura 4.22 – Página apresentada quando o utilizador está registado e pretende a eliminação de um programa existente. 91

Figura 4.23 – Página apresentada quando o utilizador está registado e pretende a criação de um novo programa de assemblagem. 91

Figura 4.24 – Função responsável por estabelecer a comunicação entre a aplicação e a base de dados. 92

Figura 4.25 – Função usada pela aplicação na inserção da informação em uma tabela específica da base de dados..... 93

Figura 4.26 – Função usada pela aplicação na consulta dos dados na base de dados. 93

Figura 4.27 – Função usada pela aplicação para a edição dos dados alocados na base de dados... 94

Figura 7.1 – Porta série. 105

Figura 7.2 – Constituição da trama para envio da letra “m” via RS-232. 106

Figura 7.3 – Formato de envio de uma trama via porta série..... 107

Figura 7.4 – Algoritmo usado na implementação da rotina que gere o timer do cordão de silicone. 109

Figura 7.5 – Rotina responsável pela verificação dos timers activos e activação de novos timers. 109

Figura 7.6 – Algoritmo usado na implementação da rotina “Mesa vai Iniciar Rotação”. 110

Figura 7.7 – Algoritmo usado na implementação da rotina “Mesa Terminou Rotação”. 111

Figura 7.8 – Algoritmo usado na implementação da rotina “editar programa no robô 1”. 112

Figura 7.9 – Algoritmo usado na implementação da rotina “editar programa no robô 2”. 112

Figura 7.10 – Algoritmo usado na implementação da rotina “criar novo programa” para a célula. 113

Figura 7.11 – Algoritmo usado na implementação da rotina “Mudar Programa” da célula. 114

CAPÍTULO 1

Identificação do Problema

1.1. Introdução

A reestruturação rumo a uma economia realmente moderna, que mostre uma produção industrial sustentada, competitiva, é uma tarefa diária, apoiada no uso eficiente da informação e do conhecimento.

A exposição à competição do mercado, assim como a existência de empresas com maior experiência na adopção de soluções tecnológicas inovadoras e na implantação de práticas administrativas diferenciadas, oxigenou a teia industrial e deu ímpeto rumo à modernização das empresas. Foi neste contexto que a automação tomou conta do panorama industrial moderno (Figura 1.1).

Figura 1.1- Automação de uma linha de montagem automóvel.

A palavra automação está directamente ligada ao controlo automático, ou seja acções que não dependem da intervenção humana. Este conceito é discutível pois a "mão do homem" será sempre necessária, pois sem ela não seria possível a construção e implementação dos processos automáticos. Contudo não é objectivo deste trabalho este tipo de abordagem filosófica, ou sociológica.

Historicamente, o surgimento da automação está ligado com a mecanização, sendo muito antigo, remontando da época de 3500 e 3200 a.C. [1], com a utilização da roda. O objectivo era sempre o mesmo, o de simplificar o trabalho do homem, de forma a substituir o esforço braçal por outros meios e mecanismos, libertando o tempo disponível para outros afazeres, valorizando o tempo útil para as actividades do intelecto, das artes, lazer ou simplesmente entretenimento.

O avanço da automação está ligado, em grande parte, ao avanço da microelectrónica que se deu nos últimos anos. Os PLC's (Controlador Lógico Programável) surgiram na década de 60 [1] e substituíram os painéis de controlo baseados em relés. Com esta substituição diminuiu o consumo de energia, a manutenção ficou mais facilitada assim como a modificação de comandos e as complicadas alterações.

Neste momento os PLC´s podem ser encontrados numa vasta gama de processos: empacotamento, engarrafamento, enlatamento, transporte e manuseio de materiais, turbinas, geração de energia, em sistemas de controlo de ar condicionado, sistemas de segurança, montagem automatizada, linhas de pintura e sistemas de tratamento de água, existentes em indústrias de alimentos, bebidas, química, têxtil, plásticos, papel e celulose, farmacêutica, metalúrgica, etc..

Com a evolução da tecnologia e com o objectivo de tornar um sistema autónomo ainda mais eficaz foram introduzidos programas de computadores. Estes programas, revelaram-se de crucial importância na estrutura de gestão das empresas, facto pelo qual deixaram de ser vistos como meras ferramentas operacionais, ou de engenharia, e passaram a ser vistos como uma importante fonte de informação.

A completa automatização de um sistema envolve o estudo dos quatro elementos representados na Figura 1.2, seja o sistema de pequeno, médio ou grande porte. Estes últimos podem atingir uma complexidade e tamanho tais que, o seu controlo deve ser dividido em níveis, onde a comunicação e "hierarquia" dos elementos é similar a uma estrutura organizacional do tipo funcional. A Figura 1.3 ilustra de forma simplificada este tipo de organização. Em que no primeiro nível encontram-se os sensores e actuadores, no segundo os sistemas de aquisição de dados que é normalmente baseado em PLC ou em

placas colocadas nos barramentos de um computador pessoal e no terceiro nível temos os sistemas SCADA que permitem a monitorização e a supervisão de todo o processo.

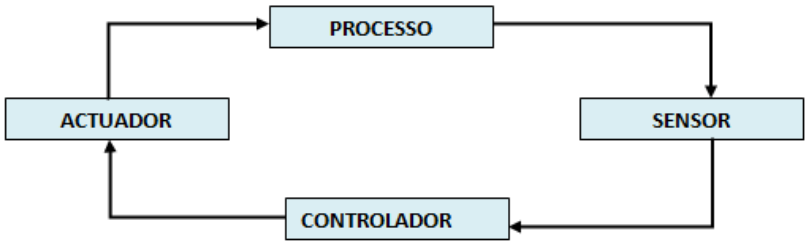


Figura 1.2 - Diagrama simplificado de um sistema de controlo automático.

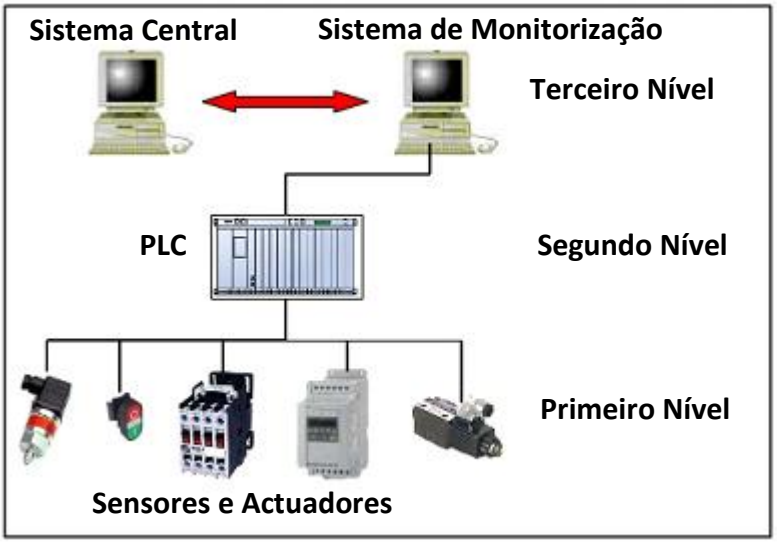


Figura 1.3 - Arquitectura de rede simplificada para um sistema automatizado.

Com as tecnologias de informação actuais é actualmente possível o acesso e partilha de dados entre a área de produção e a área de supervisão e controlo a partir de diferentes locais dentro das instalações das empresas. Desta forma, com o uso de um *browser*, é possível controlar em tempo real uma máquina localizada em qualquer parte do mundo, bastando introduzir o seu *URL browser* sem que haja necessidade de se deslocar.

1.2. Problema Proposto

Tendo em consideração o que foi exposto anteriormente foi celebrado um protocolo entre a empresa Silgal e a Universidade do Minho para o desenvolvimento de uma célula de trabalho, protótipo, para a assemblagem automática de peças à prova de água (centralinas) para carros de topo de gama, tais como o Jaguar V8, Opel NCL, Jaguar X150, Volvo SI6.

Esta célula destina-se à empresa Denso sediada em Barcelona que produz este tipo de peças. A célula para além da estrutura mecânica é composta por dois robôs em que um deles tem como função fazer o *pick and place* da tampa e o outro, o aparafusamento da centralina (Figura 1.4).

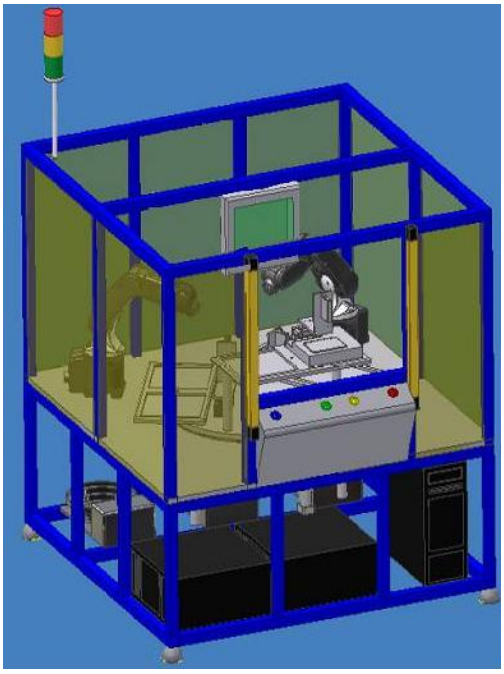


Figura 1.4 - Célula de assemblagem de centralinas proposta pelo cliente.

1.3. Motivação do trabalho

O desenvolvimento desta célula terá as seguintes vantagens em relação ao processo de assemblagem que actualmente se encontra implementado:

- Redução do cansaço visual do operador pois na actual máquina a vigia da colocação das zonas de gel e cordão de silicone em locais específicos da centralina, é realizada manualmente, com esta nova célula o controlo dessas zonas será realizado de forma automática, logo teremos uma maior precisão no controlo de qualidade;
- Redução do esforço físico do operador;
- Melhor rapidez de resposta ao atendimento da produção;
- Restabelecimento mais rápido do sistema produtivo;
- Redução da área de trabalho;
- Maior controlo e gestão de toda a produção;
- Garantia de uma competição mais voraz num mercado cada vez mais globalizado;
- Redução dos custos de produção;
- Aumento da qualidade do produto final;

Como principal desvantagem desta célula temos:

- O seu elevado custo de construção;

1.4. Organização da Tese

O Capítulo 2 apresenta todos os requisitos exigidos pelo cliente para a construção da célula de assemblagem (secção 2.1). Posteriormente, é apresentado o modo de operação a implementar na célula (secção 2.2). Por fim é apresentada a célula de assemblagem desenvolvida juntamente com todos os seus sistemas constituintes: mecânicos e electrónicos (secção 2.3).

O Capítulo 3 apresenta o desenvolvimento da aplicação desenvolvida para monitorização de todo o processo de assemblagem ocorrido na célula. Sendo apresentado todas as suas funcionalidades (secção 3.1), e onde pode ser visualizado o interface desenvolvido (secção 3.2) bem como todo o protocolo de comunicação entre o *software* desenvolvido e os sistemas a ele ligado (secção 3.3). Podemos também consultar todo o conjunto de algoritmos desenvolvidos e consequentes implementações das rotinas responsáveis pela recepção e tratamento das tramas bem como funções responsáveis pela realização de toda a reprogramação da célula (secção 3.4). Por fim é apresentado o sistema escolhido para a realização da programação inicial da célula e o sistema que gere toda a realização do histórico de assemblagem (secção 3.5).

O Capítulo 4 apresenta um conjunto de actualizações realizadas a implementar no desenvolvimento de uma futura célula, e são apresentadas todas as novas funcionalidades acrescidas ao processo de assemblagem, decorrentes do processo de actualização realizado (secção 4.1). Entre as quais se realça a implementação de uma base de dados (secção 4.2) e de uma página *Web* (secção 4.3) que permitem respectivamente o armazenamento e visualização de todo o histórico de assemblagem da célula permitindo também a impressão de relatórios em pdf. É também possível analisar todo o conjunto de rotinas implementadas responsáveis pela comunicação, envio e consulta de dados entre a aplicação e a base de dados (secção 4.4).

O Capítulo 5 apresenta as principais conclusões obtidas com a realização deste trabalho (secção 5.1), para além de sugestões para trabalhos futuros (secção 5.2).

CAPÍTULO 2

Célula de Assemblagem

2.1. Requisitos da célula a construir

Como qualquer máquina em fase de desenvolvimento foi-nos fornecido uma série de especificações requisitadas pela empresa compradora (Denso). Das quais devemos realçar as seguintes condições e aspectos de fabrico:

- Construção de uma estrutura metálica em aço onde assentam todas as partes do sistema;
- Utilização de um tabuleiro por produto a trabalhar, a Figura 2.1 apresenta um possível tabuleiro;
- Área de trabalho assente sobre uma mesa rotativa;
- Utilização de um robô de *pick and place* (referência da Denso);
- Utilização de um sistema de pressão para fixação da tampa;
- Utilização de um robô de aparafusamento (referência da Denso);
- Desenvolvimento de uma aplicação de controlo, registo e monitorização de todo o estado do processo em tempo real de modo manter o operador informado sobre o estado do processo de assemblagem;
- *Software* deve ter a capacidade de impressão de etiquetas com o conjunto dos erros ocorridos durante o processo de assemblagem de cada centralina;
- Desenvolvimento de um sistema de mensagens informativas, em tempo real, do decorrer de todo o processo de assemblagem;
- Desenvolvimento de um sistema de monitorização e visualização dos tempos do cordão de silicone associados ás peças constituintes da centralina;

- Utilização de um sistema automático de inspecção visual de zonas de gel, posição de peças no tabuleiro e estado do cordão de silicone;

- Utilização de sistema de leitura por RFID do número de série da centralina *(Serial Number*), e da referência do tabuleiro encontrado na área de trabalho *(Part Number*);

- Utilização de uma consola táctil funcional a qual deve possibilitar a comutação dos modos de funcionamento da célula de modo manual para automático e vice-versa, permitindo também a realização de toda a gestão de programas da célula;

- Utilização de um *Negara Switch* (interruptor de arranque de mão) que permite o arranque inicial da célula;

- Capacidade de a célula iniciar o tratamento de peças cinco minutos após a ligação da célula à alimentação;

- Capacidade de programação em linha. Desenvolvimento da célula de modo a que o seu processo de reprogramação possa ser realizado em menos de duas horas;

- Utilização de um sistema de segurança de acordo com as normas europeias;

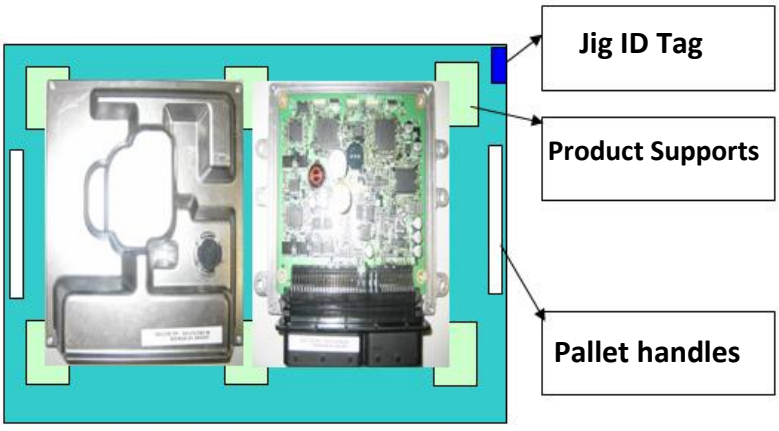


Figura 2.1 – Formato do tabuleiro requisitado pelo cliente.

- O operador enquanto elemento controlador e administrador da célula, tem como tarefa a colocação/retirada de produtos/peças desta forma a sua área de trabalho deve ser fixada a 900 milímetros do chão;

- As dimensões aproximadas e aceitáveis para o equipamento são de (1300W x 1500H x 1500D), como mostra a Figura 2.2;

- O *hardware* deve ser disposto de modo a fornecer acesso para a manutenção e limpeza diária pela parte da frente ou pela parte traseira da célula;

- Os painéis eléctricos devem ser actuados por partes externas e devem ser facilmente acessíveis pela frente ou pela parte traseira da máquina;

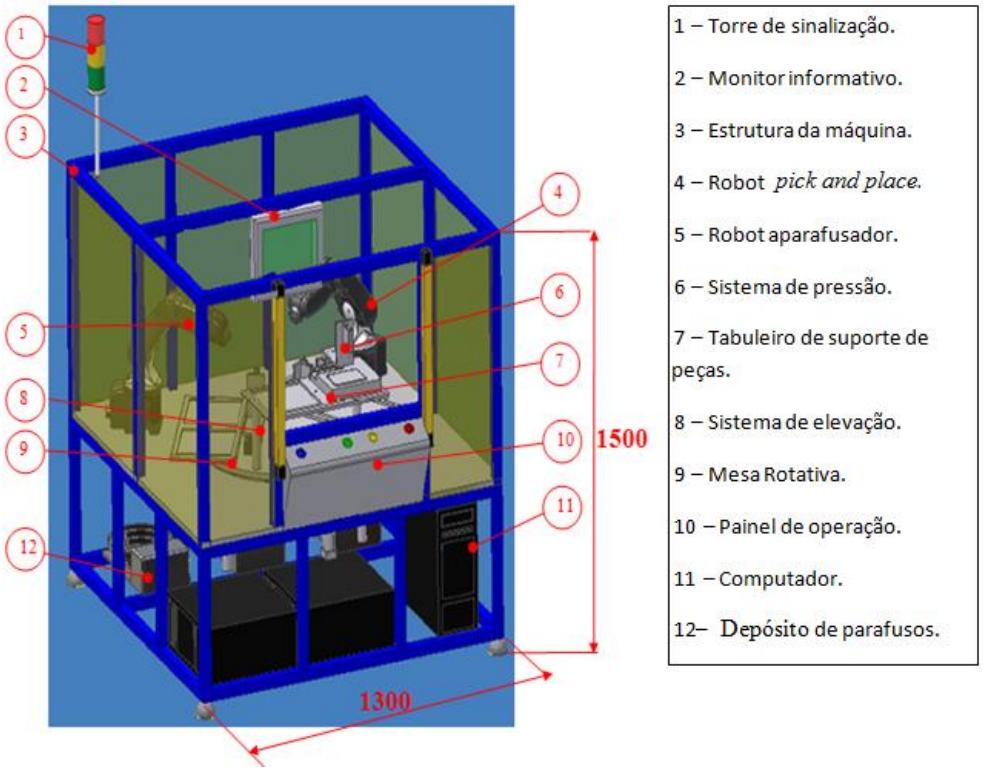


Figura 2.2 – Estrutura da célula requisitada pelo cliente.

Foram também definidos pelo cliente os tempos para uma boa cadência de produção os quais foram subdivididos da seguinte forma: tempo de ciclo, tempo de máquina, tempo de colocação, tempo disponível, percentagem de operação e quantidade processada.

Seguidamente é apresentado mais pormenorizadamente o significado de cada um dos tempos mencionados:

- **Tempo de ciclo (CT)** → é definido como o tempo de entrada das peças até à sua saída, ou seja, é o tempo necessário para a assemblagem completa de uma centralina;

- **Tempo de máquina (MT)** → é definido como a quantidade de tempo necessário para que a célula inicialize, transfira, processe, faça o retorno, e identifique a conclusão do processo;

- **Tempo de colocação (HT)** → é definido como o tempo necessário para carregar e descarregar uma peça da célula;

- **Tempo disponível (AT)** → é definido como o tempo ainda disponível para o processamento dentro de um determinado período de tempo.

- **Percentagem de operação (OR)** → é definida como a percentagem de tempo em que o equipamento deve estar operacional durante o tempo disponível. Usar 85% no cálculo.

- **Quantidade processada (PQ)** → é definida como o número de peças que estão a ser processadas dentro de um tempo de ciclo definido.

Tendo em conta os tempos definidos anteriormente é possível calcular a capacidade do equipamento que é dada por:

$$EC = \left(\frac{AT \times OR}{CT} \times PQ \right)$$

Tempos	Valores definidos
CT	25 Segundos por parte ou menos
HT	5 Segundos por parte ou menos
MT	20 Segundos por parte ou menos
AT	26712 Segundos por deslocamento 26712 Segundos por turnos
OR	85%
PQ	1
EC	908 Partes por deslocamento

Tabela 1.1 – Valores da capacidade do equipamento requisitados pelo cliente.

2.2. Modo de operação da célula

A centralina pode ser dividida em três partes: tampa, PCB e base (Figura 2.3). Sobre a base encontra-se a placa PCB. A assemblagem da centralina consiste na colocação da tampa em cima da base que contém o PCB e posterior aparafusamento.

O processo de assemblagem deverá passar pelas seguintes etapas:

- Colocação dos constituintes da centralina na área de trabalho (Pos.1);
- Verificação de todos os requisitos de avanço (Pos.1);
- Colocação da tampa em cima da base realizada pelo robô *pick and place* (Pos.2);

- Realização do aparafusamento da centralina pelo robô 2 (Pos.3);
- Por fim, retirar o produto assemblado (Pos.1).

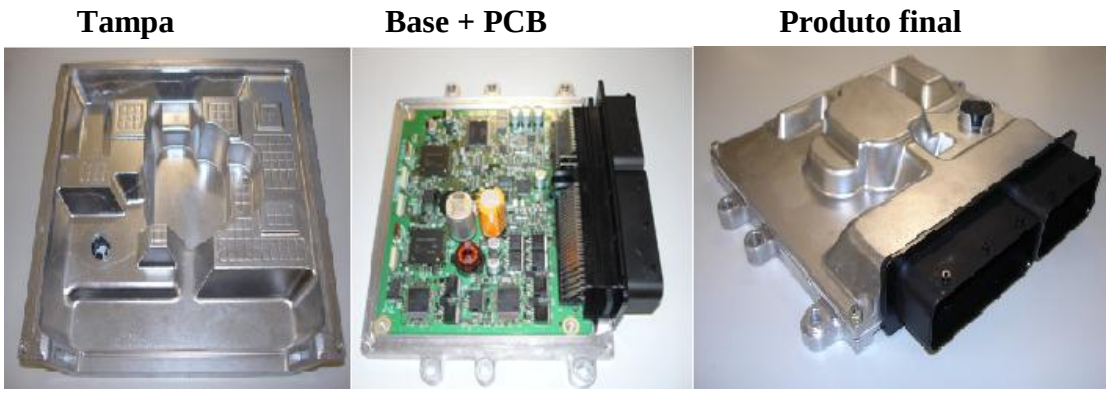


Figura 2.3 - Constituintes da centralina.

Na Figura 2.4 é mostrada uma perspectiva das diferentes posições de assemblagem da centralina.

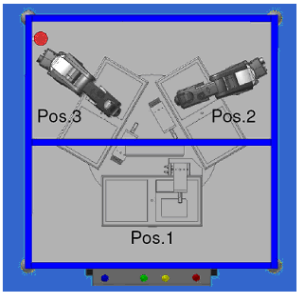


Figura 2.4 - Perspectiva das diferentes posições de assemblagem da centralina.

2.3. Arquitectura da Célula de trabalho

Tendo em consideração os requisitos apresentados pelo cliente, foi construída uma célula de trabalho que apresenta um conjunto de soluções mecânicas e electrónicas que permitem a obtenção de um produto final funcional (Figura 2.5). A célula está dividida em duas partes bem diferenciadas: parte estrutural e sistemas mecânicos. A parte estrutural está por sua vez dividida em dois subconjuntos, estrutura inferior e estrutura superior.

A estrutura inferior é formada por uma estrutura metálica em aço que suporta os mecanismos de accionamento e deslocamento das paletes (tabuleiros) que transportam os elementos a assemblar. Além do descrito, contém também os sistemas pneumáticos e electrónicos de controlo da célula.

A estrutura superior da célula é concebida em perfil de alumínio e tem como finalidade impedir o acesso das pessoas ou corpos estranhos ao normal funcionamento da célula. Inclui uma consola do operador e um software que realiza a monitorização de todo o processo de assemblagem através de um monitor ligado ao computador pessoal.

É por esta parte da célula que é realizada a colocação/retirada das peças.

Figura 2.5 - Célula de assemblagem desenvolvida.

2.3.1. Sistemas mecânicos

Para o desempenho das funções exigidas foram escolhidos os seguintes sistemas mecânicos: mesa rotativa pneumática e respectivas posições de trabalho, sistema de elevação dos tabuleiros, sistema de sucção a ar comprimido da tampa, sistema de aparafusamento, sistema de alimentação de parafusos, sistema de bloqueio/desbloqueio "*clamping*". A interligação dos sistemas mecânicos é mostrada na Figura2.6.

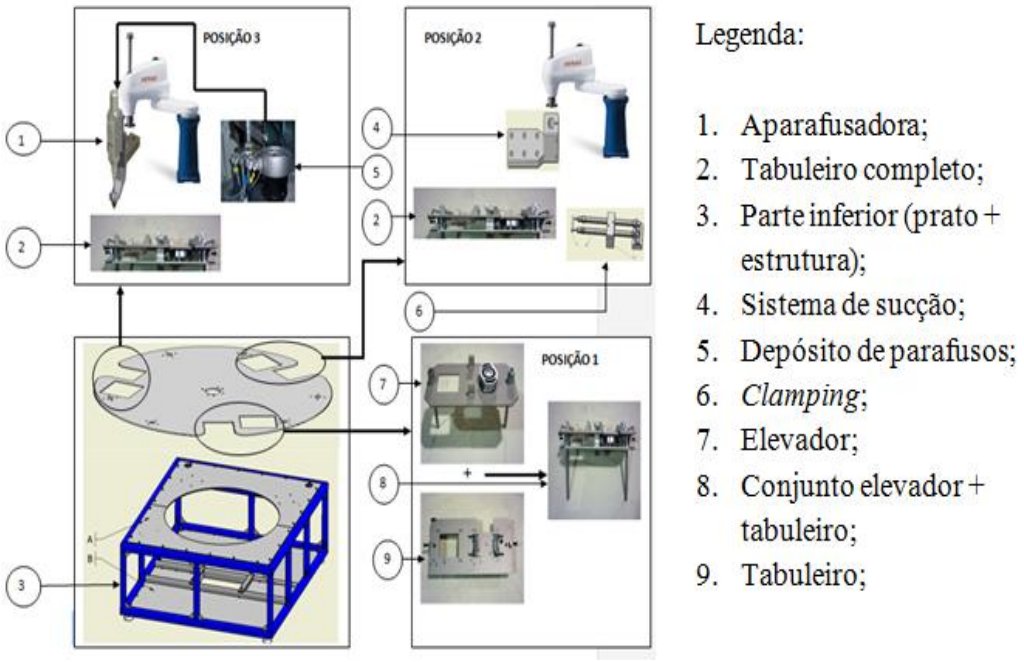


Figura 2.6 - Interligação entre os vários sistemas mecânicos que fazem parte da célula.

Na figura 2.6 é possível observar a utilização de dois braços robóticos, estes braços têm como finalidade levar a cabo as mais variadas tarefas, para as quais são programados. Para realizar as tarefas atribuídas, movem componentes, objectos, ferramentas e outros dispositivos especiais por meio de movimentos e pontos pré-programados.

Os braços robóticos foram identificados como robô 1 e o robô 2, o primeiro, ilustrado na Figura 2.7, está localizado na posição 2 e tem como função a colocação da tampa em cima da base da centralina usando um sistema de sucção a ar comprimido (Figura 2.8). Após a colocação correcta da tampa em cima da base o robô 1 acciona um *clamping* que é um sistema de pressão auxiliar que permite uma melhor fixação das peças.

No fim da colocação da tampa o autómato é informado do sucesso ou não dessa operação continuando o processamento do seu algoritmo de acordo com o resultado recepcionado.

Figura 2.7 - Robô 1.

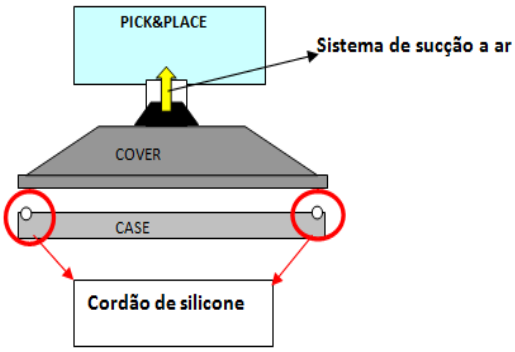


Figura 2.8 - Colocação da tampa em cima da base pelo robô 1.

O segundo braço robótico, o robô 2 ilustrado na Figura 2.9, está localizado na posição 3 é responsável pelo aparafusamento da centralina, usando para isso um sensor que informa o controlador da aparafusadora a ele acoplado que pode inicializar o aparafusamento.

A sequência de aparafusamento da centralina usada pode ser visualizada na Figura 2.10.

Figura 2.9 - Robô 2.

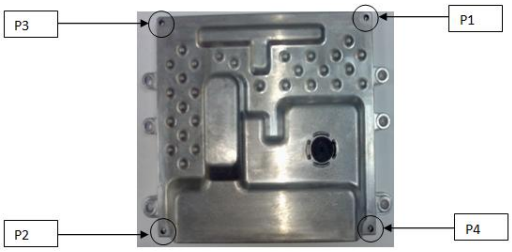


Figura 2.10 - Sequência de aparafusamento para a centralina usada, em que P1, P2, P3, e P4 representam respectivamente parafuso 1, parafuso 2, parafuso 3 e parafuso 4.

O robô 2 encontra-se interligado aos alimentadores automáticos de parafusos ilustrados na Figura 2.11. Estes alimentadores são giratórios e vibratórios, fazendo com que através de um sistema de rosca os parafusos sejam deslocados para a superfície entrando num canal, esperando a sua sucção por parte do robô 2.

Figura 2.11 - Depósitos de parafusos.

Analisando a Figura 2.6 podemos verificar que o sistema dispõe de um elevador localizado na posição 1, responsável pela elevação do tabuleiro para a posterior colocação dos elementos constituintes da centralina, por parte do operador. Após a ordem de descida coloca os elementos na área de trabalho para proceder à assemblagem da centralina.

Para a movimentação das partes mecânicas constituintes do sistema foi utilizado um sistema pneumático, ilustrado na Figura 2.12 o qual é formado por um conjunto de válvulas que quando solicitadas fazem a distribuição de ar comprimido para um conjunto de dispositivos ligados ao sistema, entre eles a mesa rotativa, o elevador, o sistema de sucção

da tampa da centralina acoplado ao robô 1 e o sistema de sucção dos parafusos. Toda a pressão de ar é fornecida por um compressor.

Figura 2.12 – Sistema pneumático.

2.3.2. Sistemas electrónicos

A constituição e interligação de todos os sistemas electrónicos são visualizadas na Figura 2.13.

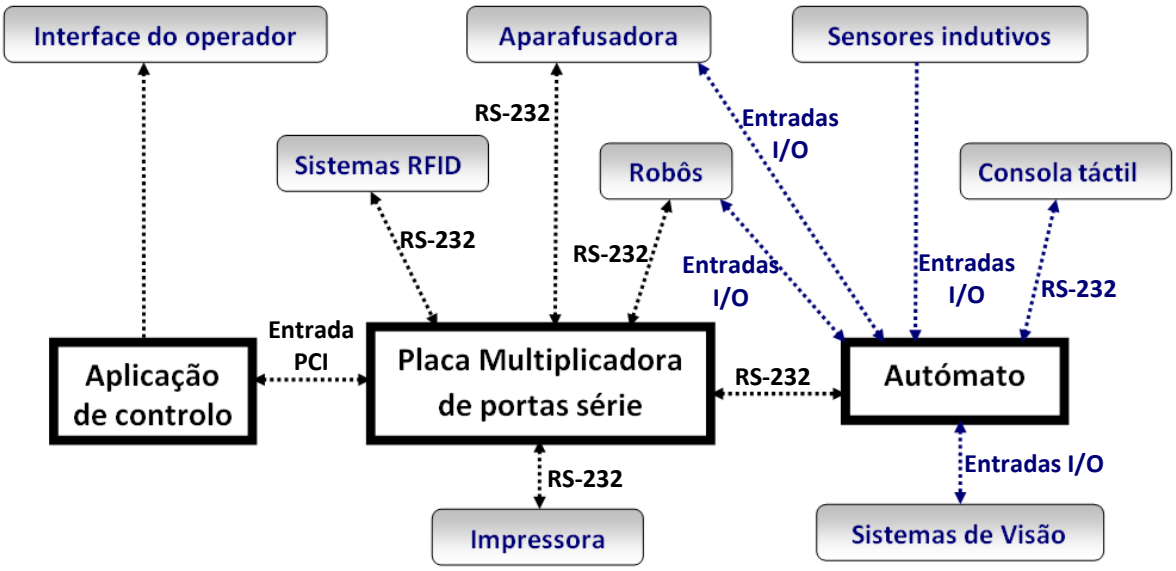


Figura 2.13 – Interligação entre os vários sistemas electrónicos e o computador de secretária.

Analisando a Figura 2.13 podemos identificar e descrever cada um dos sistemas usados.

Sistemas de visão, realizam a aquisição e processamento de imagem de acordo com a sua programação e transmitem o resultado do processamento ao autómato através das entradas I/O.

Depois do estudo realizado foram adoptados sistemas de visão diferentes para análise das regiões de gel e do cordão de silicone. Pois no caso da inspecção das regiões de gel é necessário garantir a correcta colocação de cada uma delas, quanto ao cordão de silicone é apenas necessário a inspecção de um ponto do cordão, pois garantindo a existência desse ponto é garantido pela Denso que o cordão é colocado de forma correcta pela máquina auxiliar.

Desta forma o sistema de visão adoptado é constituído por uma câmara munida de um controlador da marca Omron mais propriamente o modelo ZFX para a análise das regiões de gel. Para a análise do cordão de silicone foi escolhido o sensor de visão da Omron gama ZFV.

O **sistema ZFX**, mostrado na Figura 2.14, dispõe de um controlador e de uma câmara. Como já foi dito anteriormente é da marca Omron e é responsável pela análise das regiões de gel localizadas na tampa da centralina.

Figura 2.14 - Sistema de visão responsável pelo processamento das regiões de gel localizadas na tampa da centralina.

As principais características deste equipamento são:

- Ecrã táctil que fornece uma visualização em tempo real das imagens adquiridas;
- Permite a configuração de várias regiões de inspecção por imagem;
- Possui 32 bancos de memória o que possibilitam a criação de 32 programas diferentes;
- Disponibilização de filtros de imagem.

Para que este controlador responde-se às características específicas do nosso sistema foi necessária a sua prévia programação. Assim sendo foram definidas as zonas de inspecção a analisar e o protocolo de comunicação com o autómato.

Perante as inspecções permitidas e descritas no seu *datasheet* escolhemos o reconhecimento por padrões pois é o que melhor respondia ao nosso problema.

Assim sendo foram definidas e programadas as regiões de gel a processar as quais podem ser visualizadas na Figura 2.15. Para a definição destas zonas foi usado o banco 0.

Posteriormente foi realizada a programação do banco 1 no qual ficou programado o item da posição, o qual tem como finalidade garantir que a tampa da centralina se encontra bem distribuída no tabuleiro encontrado na área de trabalho.

É de notar que estas regiões de gel e posição da tampa variam com a gama de centralina a assemblar.

Figura 2.15 - Tampa da centralina com as respectivas zonas de gel e posição programadas, estas zonas variam com a gama de centralina a assemblar.

Depois de efectuada a programação da posição e regiões de gel definiu-se um *AND* entre os dois bancos onde as programações estavam alocadas de modo a que a saída "OK" (tampa cumpre requisitos) ocorra apenas quando as duas inspecções estão dentros dos requisitos definidos. Basta que uma das verificações não cumpa os requisitos para que o resultado da saída seja negativo "NG" (tampa não cumpre requisitos). Podemos visualizar o principio de verificação do controlador expresso na Figura 2.16.

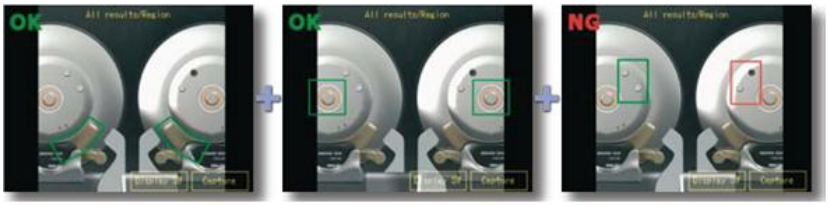


Figura 2.16 - Modo de funcionamento do sistema de análise de gel.

Foram também aplicados filtros de cor e um filtro extra á saida da iluminação com a finalidade de diminuir os refelexos causados pela luz emitida pela iluminação acoplada à câmara sobre a própria peça. Pois como o gel é de cor branca poderia ser confundido com os reflexos da luz natural, podendo causar interferências na análise das regiões por parte do controlador. A iluminação extra acoplada à câmara é formada por uma cintura de led´s de cor vermelha.

A interacção entre o controlador e o autómato segue a seguinte sequência: aquisição da imagem pelo controlador a qual por sua vez é controlada pelo autómato através do envio de um sinal *trigger*, que não é mais que um sinal de tensão. Após o envio do *trigger* a imagem é adquirida e é realizado o processamento das regiões de gel e verificada a correcta colocação da tampa da centralina na respectiva área do tabuleiro. No final de todo o processamento por parte do controlador o mesmo envia ao autómato o resultado da inspecção.

O **sistema ZFV** é responsável pela análise do cordão de silicone e encontra-se ilustrado na Figura 2.17.

Como características específicas deste sistema temos:

- Processamento de imagem em formato de sensor;
- Ecrã táctil que fornece uma visualização em tempo real das imagens adquiridas;
- Possibilidade de configuração de uma rotina de inspecção específica;
- Filtragem automática de cores;

Figura 2.17 - Sistema ZFV responsável pela análise do cordão de silicone.

De modo a responder às necessidades do sistema foram definidas e efectuadas as programações do controlador, entre as quais foram definidas as regiões do cordão de silicone a analisar e foi definido o protocolo de comunicação entre o controlador e o autómato.

No caso da análise do cordão de silicone foi utilizada a inspecção por padrão pois é a que melhor se adaptou às nossas necessidades, pois como já foi referido no caso do cordão de silicone é apenas necessário que o sensor detecte uma parte do cordão a fim de garantir a sua existência total (Figura 2.18).

Todo o controlo de aquisição de imagem é realizado pelo autómato através de um *trigger* enviado ao controlador de análise de silicone o qual por sua vez realiza a aquisição da imagem através do sensor que tem acoplado. Posteriormente é realizado o processamento e tratamento da imagem detectando a presença ou não da área específica do cordão de silicone, no fim o autómato é informado do resultado da operação ("OK" ou "NG").

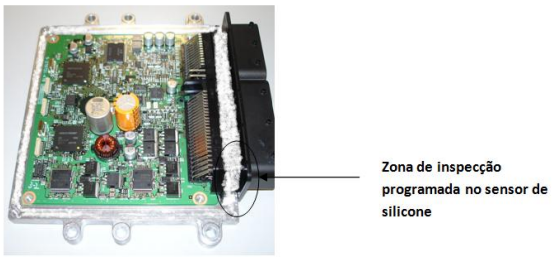


Figura 2.18 - Base da centralina e zona do cordão de silicone inspeccionado pelo sensor de silicone.

Sistemas RFID, usados na leitura da referência do tabuleiro em actual utilização na célula e na leitura do número de série de cada centralina, foram usados dois sistemas que

utilizam a tecnologia RFID – *Radio Frequency Identification* como o nome indica funciona por Radiofrequência.

Os sistemas de RFID que se encontram actualmente no mercado têm uma arquitectura bastante simples, que se baseia em quatro componentes básicos – o ***tag***, a ***antena***, o **leitor** (*Reader*) e um módulo de ***middleware***, sistemas mostrados na Figura 2.19.

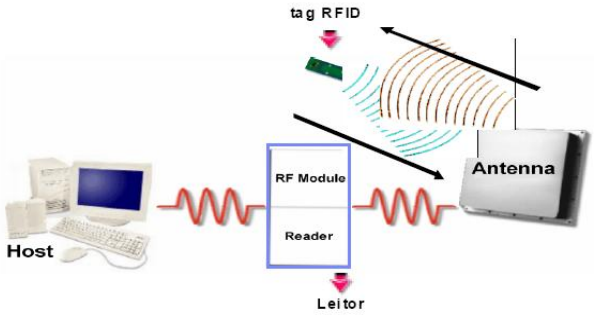


Figura 2.19 - Constituintes de um sistema RFID.

As vantagens deste tipo de sistemas relativamente aos sistemas de identificação baseados em código de barras podem ser resumidas aos seguintes itens [2]:

- Não estão limitados à leitura por contacto;
- Os *tags* são lidos a alta velocidade;
- Os *tags* podem ser usados em ambientes industriais e podem ser reutilizáveis;
- Os *tags* podem ser lidos através de sujidade, tinta, vapor, lama, plástico e madeira;
- Os *tags* passivos têm um tempo de vida virtualmente infinito;
- Os *tags* suportam uma maior quantidade de dados e podem ter funcionalidades de leitura/escrita.

No nosso caso foram utilizados dois sistemas da Omron (Figura 2.20), o Omron V700 e Omron V600 responsáveis respectivamente pela leitura do número de série que está associado a cada centralina e pela leitura do *Part Number*, ou seja, a identificação dos tabuleiros que estão na área de trabalho da célula.

Figura 2.20 – Sistemas RFID utilizados na célula à direita o Omron V600 e à esquerda o Omron V700.

Sensores indutivos, são emissores de sinal que detectam, sem contacto directo, elementos metálicos que atravessam o seu campo magnético convertendo-o num sinal eléctrico inteligível.

As principais características que distinguem este tipo de sensores são:

- Grande durabilidade pois não necessitam de contactos mecânicos para a detecção;
- Excelente precisão e garantia da supervisão;
- Altas frequências de comutação até 3000Hz;
- Indicação de sinal através de Leds;
- Insensível a trepidações;
- Podem ser utilizados em ambientes industriais agressivos;
- Versatilidade de formatos e formas de fixação;

No nosso caso, foram utilizados tendo como finalidade a detecção da correcta rotação da mesa rotativa e para a contagem dos parafusos aparafusados pelo robô 2.

O sensor de parafusos, ilustrado na Figura 2.21, indica através de um sinal luminoso a passagem de um parafuso e envia um sinal eléctrico o qual informa o robô 2 que por sua vez realiza aparafusamento e contagem dos parafusos.

Figura 2.21 - Sensor indutivo detector de parafusos.

Quanto ao sensor de detecção de rotação de mesa (Figura 2.22) foram usados dois, encontrando-se espaçados de 60 graus um do outro tendo como finalidade a indicação da correcta rotação da mesa. A mesa rotativa dispõe de 3 saliências espaçadas de 120 graus, a rotação da mesa é controlada pelo autómato tendo como condição de paragem a resposta dos sensores indutivos.

Figura 2.22 - Sensor indutivo detector de rotação de mesa.

Aparafusadora, responsável pelo aparafusamento da centralina na posição 2. Foi usada uma aparafusadora da marca Atlas Copco, ilustrada na Figura 2.23 a qual dispõe de um controlador o qual possibilita a programação de todas as variáveis necessárias ao aparafusamento entre as quais o ângulo e o torque. Este controlador quando programado, fornece a informação de todos os valores de torque e ângulo ocorridos durante a fase de aparafusamento bem como a ocorrência e descrição de algum erro que possa ocorrer.

Figura 2.23 - Conjunto aparafusadora controlador.

Placa Multiplicadora Rs-232, placa utilizada tendo como objectivo o aumento do número de portas série disponibilizadas pelo computador. Foi utilizada a placa da marca Moxa modelo Cp-118 U, representada na Figura 2.24.

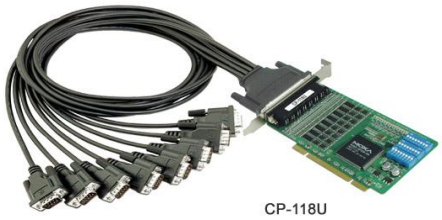


Figura 2.24 - Placa multiplicadora de portas série usada na ligação dos vários dispositivos ao computador via RS-232.

As principais características desta placa e as quais influenciaram a sua escolha são:

- Placa largamente usada na indústria em sistemas de automação, e por essa razão uma placa com *software* e *hardware* bem testado;
- Disponibilização de uma ligação PCI para ligação ao computador;
- Disponibilização de oito portas séries como saídas servindo as mesmas de interligação entre o computador e os sistemas a ele ligados.
- Disponibilização de uma biblioteca formada por um conjunto de funções usadas no controlo de recepção/emissão de dados através das diversas portas série que dispõe. No nosso caso era essencial a existência de uma placa que disponibiliza-se o uso de interrupções por eventos ou *threads*, ou seja, funções que se encontram em constante "escuta" da ocorrência de eventos e que na ocorrência dos mesmos executam as funções previamente definidas.

CAPÍTULO 3

Aplicação desenvolvida

3.1. Funcionalidades

O *software* desenvolvido tem como função realizar a gestão e monitorização de todo o processo de assemblagem da centralina.

Este *software* apresenta as seguintes características principais:

- Um interface de monitorização e controlo de todo o processo de assemblagem;
- Programação dinâmica da célula;
- Controlo, selecção e visualização do *timer* do cordão de silicone;
- Interacção com os vários dispositivos do sistema;
- Facilidade de reprogramação da célula;
- Armazenamento e possibilidade de consulta de todo o historial de assemblagem das centralinas e suas características;
- Impressão de etiquetas, em caso de ocorrência de algum erro durante o processo de assemblagem de cada centralina.

3.2. Interface desenvolvido

O interface de visualização do operador desenvolvido encontra-se dividido em 3 partes representando, respectivamente, cada uma das posições de trabalho, como ilustra a Figura 3.1. Este interface foi desenvolvido usando o *Microsoft Visual Studio 2008* sendo a sua implementação baseada em janelas de diálogo.

Legenda:

1-Área de trabalho do robô 2;

2-Área de trabalho do robô 1;

3-Posição que faz de interface entre operador e a máquina;

Figura 3.1 – Vista geral do ecrã de monitorização de todo o processo de assemblagem da célula.

Os dispositivos ligados a cada uma das portas RS-232 disponíveis no computador podem ser visualizados na Figura 3.2. Como se pode observar pela figura foram utilizadas oito portas série todas elas da placa Moxa (referida anteriormente na secção 2.3.2) colocada no barramento PCI do computador pessoal.

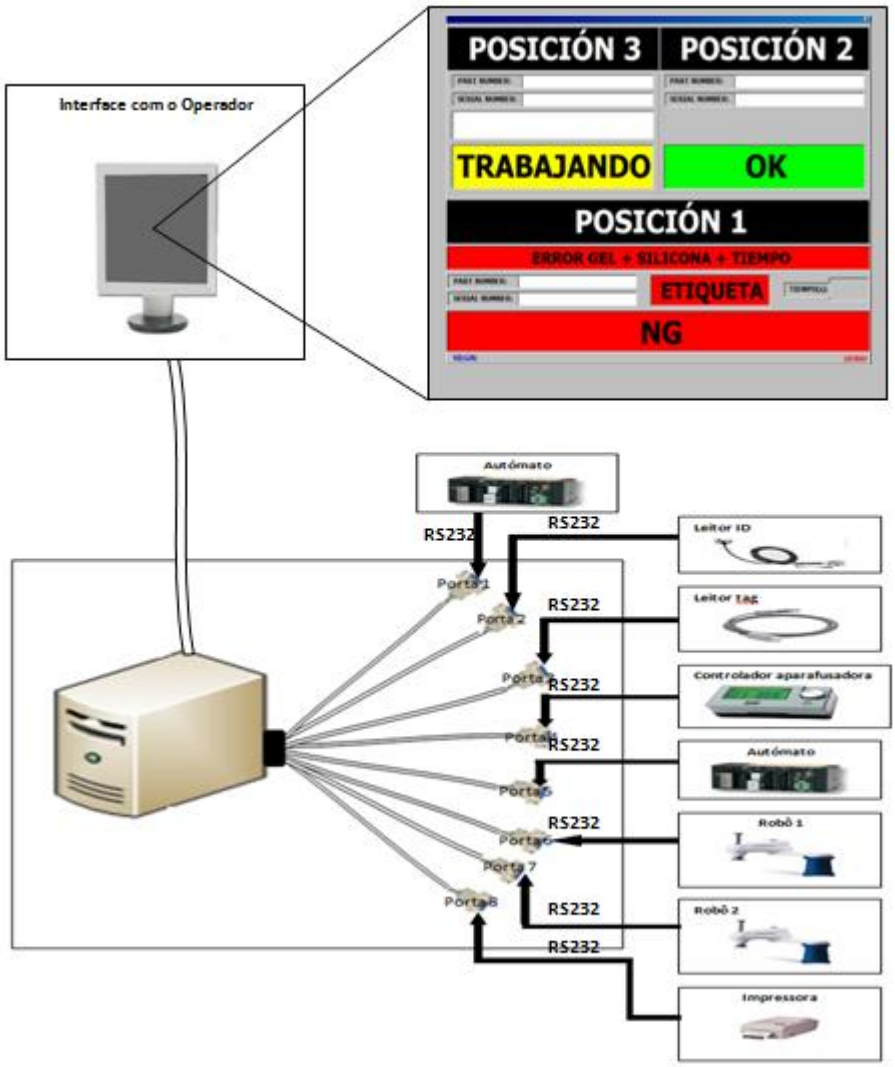


Figura 3.2 – Esquema de ligação entre os vários dispositivos do sistema e o computador.

A Figura 3.3 mostra a relação entre cada uma das áreas de visualização do interface e os dispositivos físicos que fornecem a informação para a actualização dessas mesmas áreas.

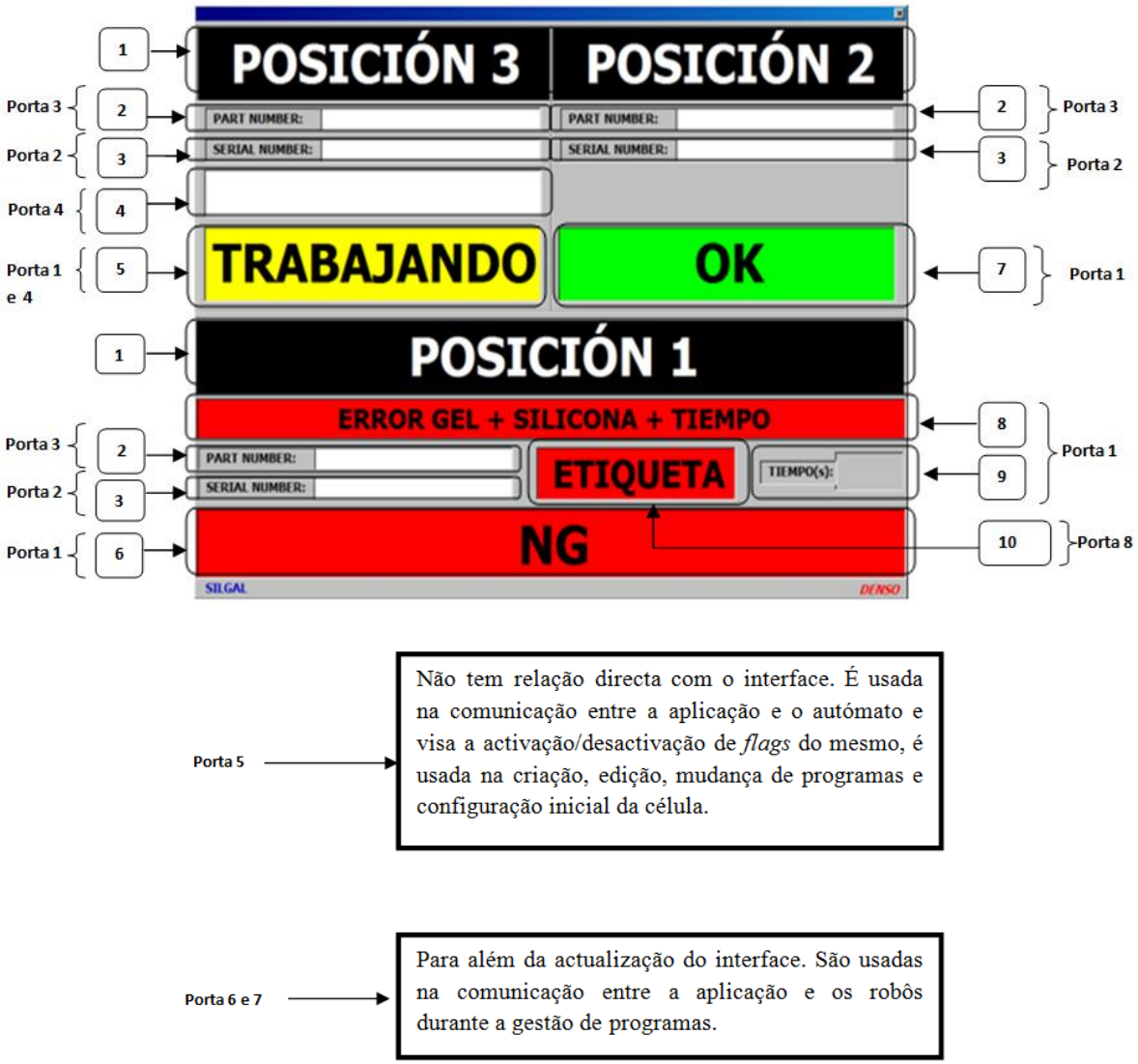


Figura 3.3 – Relação entre cada uma das áreas de visualização do interface e os dispositivos ligados ao computador.

Analisando a Figura 3.3 vamos de seguida realizar a identificação e descrição de cada uma das áreas de visualização constituintes do interface:

1. **Identificador de posição**

   Apresenta uma mensagem estática e meramente informativa**;**

2. Part Number

Representa a *tag* ou modelo da centralina para a qual o tabuleiro encontrado na respectiva área de trabalho é compatível, como podemos observar pela Figura 3.3 esta área é actualizada pela porta série número 3;

3. *Serial Number*

Representa o número de série da centralina (ID), que está a ser trabalhada na respectiva posição, como podemos observar pela Figura 3.3 esta área é actualizada pela porta série número 2.

4. Mostrador de estado de aparafusamento

Em cada parafuso colocado na centralina pelo robô 2 é mostrado neste ecrã o número do parafuso, o seu torque, o ângulo, e se ocorreu algum erro no decorrer do processo de aparafusamento. Todos estes dados são referentes à centralina que está a ser trabalhada na posição três. Como se pode observar pela Figura 3.3 esta área é actualizada pela porta série número 4.

Como exemplo, apresentamos na Figura 3.4 o resultado do aparafusamento, de uma centralina constituída por quatro parafusos, efectuado pelo robô 2.

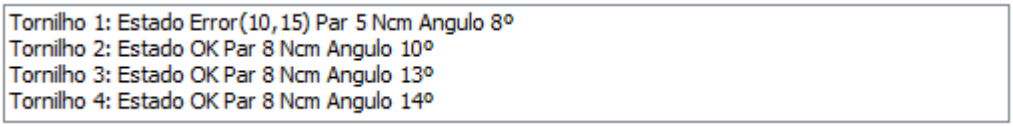


Figura 3.4 – Mostrador de estado de aparafusamento de um conjunto de quatro parafusos, aparafusados pelo robô 2.

5. Mostrador do estado da peça na Posição 3

Este mostrador é actualizado via porta série número 1 e 4. Nesta janela é possível monitorizar todo o processo de aparafusamento da centralina e visualizar o seu estado de conclusão "OK" ou "NG".

6. Mostrador do estado da peça na Posição 2

Nesta área é visualizado um conjunto de mensagens que caracterizam todo o processo de trabalho da centralina pelo robô 1. Como possíveis mensagens visualizadas nesta área temos "OK", "TRABAJANDO" e "NG", todas elas desencadeadas pelo autómato (porta série nº1) como podemos observar pela Figura 3.3. Em itens seguintes entraremos em mais detalhes sobre o significado das mensagens apresentadas.

7. Mostrador do estado da peça na Posição 1

Nesta área é visualizado um conjunto de mensagens que caracterizam o estado das peças localizadas na posição 1, estas mensagens resultam do teste de validação efectuado ao conjunto de requisitos necessários para que as peças avancem no processo de assemblagem. Como mensagens possíveis temos "OK", "TRABAJANDO" e "NG", todo o significado destas mensagens é explicado com mais detalhe em itens seguintes. Como podemos observar pela Figura 3.3 esta área é actualizada pela porta série número 1.

8. Mostrador de mensagens da Posição 1

Nesta área é visualizada toda a descrição dos erros ocorridos durante a fase de validação das peças constituintes da centralina. Para além deste conjunto de mensagens de erro podemos ver também todas as mensagens de gestão de programas bem como mensagens indicadoras do modo de funcionamento da célula de trabalho. Todas estas mensagens são desencadeadas via porta série nº1, e encontram-se descritas em mais detalhe em itens seguintes.

9. Mostrador de tempo de cordão de silicone

Nesta área é visualizada a contagem do tempo do cordão de silicone.

Este tempo ou *timer* é activado através de um relé ligado á máquina auxiliar de colocação de silicone, que ao disparar, activa uma zona de memória no autómato, o qual envia de imediato uma trama via porta série número 1, a

aplicação faz a sua leitura e desencadeia a contagem e amostragem decrescente do *timer*, que pode ser visualizado no mostrador da Figura 3.5.

Este timer é mostrado desde a entrada da peça na célula até à sua chegada à posição 2, neste instante este *timer* é apagado e é mostrado o *timer* da próxima peça a entrar na posição 1;

Figura 3.5 – Mostrador do tempo do cordão de silicone da peça.

10. Mostrador de mensagens da impressora

Nesta área é visualizada a mensagem ilustrada na Figura 3.6, que surge na conclusão da centralina e no caso da ocorrência de algum erro durante o processo de assemblagem da mesma. Após a visualização desta mensagem a impressora realiza a impressão de uma etiqueta com as características da centralina e a descrição dos erros ocorridos durante o seu processo de assemblagem.

Figura 3.6 – Esta mensagem surge no interface sempre que uma centralina é concluída e durante o seu processo de assemblagem ocorreram erros.

Os campos constituintes desta etiqueta são:

- **ID** → Número de série da centralina (Id);
- ***Serial*** → Código do tabuleiro (*Tag*);
- **Data** → Data da conclusão da assemblagem da centralina;
- **Hora** → Hora da conclusão da assemblagem da centralina;
- ***Error*** → Tipos de erros ocorridos durante a execução da centralina;
    - **Tiempo** → Tempo limite do cordão de silicone colocado na peça foi ultrapassado;
    - **Gel** → Regiões de gel não cumprem as especificações exigidas;

- **Silicone** → Cordão de silicone não está de acordo com as especificações exigidas;
- **Forçada** → As peças foram forçadas a avançar no processo pelo operador;
- **Robô *pick and place*** → Ocorreu um erro na colocação da tampa da centralina;
- **Robô de aparafusamento** → Ocorreu um erro de aparafusamento;
- **T1, T2, T3, T4** → Erro de aparafusamento no parafuso 1, 2, 3 ou 4 respectivamente;

Na Figura 3.7 é apresentado um exemplo de uma etiqueta impressa na célula.

ID: 07072500159027972093900200000000
Serial: JAGUARV8
Data: 4-03-2008
Hora: 20:06:14

Error: Gel, Forçada, Robô 2, T3

Figura 3.7 – Formato da etiqueta impressa na célula de assemblagem, podemos ver que os erros sucedidos neste caso específico foram: regiões de gel não cumpriram com os requisitos, peça foi forçada e o robô 2 efectuou um mau aparafusamento do parafuso 3.

Na ocorrência de um erro durante a execução da centralina para além da impressão da etiqueta podemos consultar o ficheiro da centralina que é gravado numa pasta específica no computador pessoal o qual possui a mesma hora e data de assemblagem impressas na etiqueta.

3.3. Comunicação com os dispositivos ligados por protocolo série

A seguir são apresentados os protocolos de comunicação usados entre a aplicação e os dispositivos a ela ligados. Em Anexo B é apresentada uma descrição mais pormenorizada das características de uma comunicação série e seus parâmetros.

3.3.1. Comunicação entre o Autómato e o PC (porta série nº 1)

Porta usada na comunicação entre o autómato e a aplicação implementada.
As configurações desta comunicação são 9600,par,8 *bits*,2 *stop bits*.

Como áreas do interface actualizáveis através desta porta temos o mostrador de estado de peça da posição 1, o da posição 2, o da posição 3, o mostrador de mensagens da posição 1 e o mostrador do tempo do cordão de silicone.

Através desta porta são enviadas a partir do autómato um conjunto de tramas mostradas nas tabelas 3.1, 3.2, 3.3 e 3.4 as quais são recebidas no computador e tratadas pela aplicação desenvolvida desencadeando a função correspondente, a qual por sua vez faz desencadear a actualização do interface.

Tramas responsáveis por desencadear as mensagens apresentadas no mostrador de mensagens da posição 1	Mensagens	Descrição da trama
A0h | 03h	**Error Silicona + Tiempo**	Trama indicadora que silicone e tempo de cordão de silicone não cumprem com os requisitos exigidos;
A0h | 04h	**Error Gel + Tiempo**	Trama indicadora que gel e tempo de cordão de silicone não cumprem com os requisitos exigidos;
A0h | 05h	**Error Gel + Tiempo + Silicona**	Trama indicadora que silicone, gel e tempo de cordão de silicone não cumprem os requisitos exigidos;
A0h | 06h	**Error Tiempo**	Trama indicadora que silicone e gel cumprem requisitos exigidos mas o tempo limite para o cordão de silicone foi ultrapassado ("NG");
A0h | 07h	**Quitar Producto/ Cambiar Produto**	Trama indicadora de que a barreira se encontra aberta, desta forma o operador deve colocar uma nova centralina, e retirar a assemblada ou em caso de a célula estar a realizar um fim de turno o operador deve retirar a centralina assemblada e iniciar a rotação da mesa.
A0h | 17h	**Mesa Girando**	Trama indicadora de que a mesa vai iniciar a sua rotação;
A0h | 18h	**Errol gel**	Trama indicadora de que o cordão de silicone cumpre os requisitos e tempo de cordão de silicone está dentro dos limites aceitáveis;
A0h | 19h	**Error silicona**	Trama indicadora de que as regiões de gel e tempo do cordão de silicone cumprem com os requisitos exigidos;
A0h | 2Ah	**Error Gel + Silicona**	Trama indicadora de que o cordão de silicone e regiões de gel não cumprem requisitos exigidos mas o tempo de cordão de silicone está dentro dos limites aceitáveis;
A0h | 2Bh	**Espera Producto Proceso Anterior**	Trama indicadora que a peça na posição 1 foi validada com sucesso cumprindo todos os requisitos e espera agora que o robô 2 conclua o aparafusamento da centralina para que se possa avançar no processo;

Tabela 3.1 – Tabela que mostra e descreve todo o conjunto de tramas enviadas do autómato para aplicação que por sua vez faz desencadear as respectivas mensagens no mostrador de mensagens da posição 1.

Tramas de gestão de programas responsáveis pela amostragem das mensagens apresentadas no mostrador de mensagens da posição 1	Mensagens	Descrição da trama
A0h | 30h	**Procesando Fin De Turno**	Trama indicadora que se vai processar ao esvaziamento de peças da célula, esta operação é realizável com a célula a trabalhar em modo manual ou automático;
A0h | 31h	**Fin De Turno**	Trama indicadora que o fim de turno terminou, encontrando-se a célula sem peças nas respectivas posições de trabalho;
A0h | 32h	**(Mostra lista programas)**	Trama que solicita à aplicação a abertura e amostragem da lista de programas existentes, esta operação é realizável com a célula a trabalhar em modo manual ou automático;
A0h | 33h	**Criando Nuevo Programa**	Trama indicadora que se vai proceder à criação de um novo programa de assemblagem de centralinas, esta operação é apenas realizável com a célula a trabalhar em modo manual;
A0h | 34h	**Editar Programa End**	Trama indicadora da finalização da edição de programa no robô 1, esta operação é apenas realizável com a célula a trabalhar em modo manual;
A0h | 36h	**Editar Programa End**	Trama indicadora da finalização da edição de programa do robô 2, esta operação é apenas realizável com a célula a trabalhar em modo manual;
A0h | 37h	**Cambiar Programa End**	Trama indicadora da conclusão da mudança do programa em execução. Esta operação pode ser realizada quer em modo automático realizando primeiramente um fim de turno ou então com a célula em modo manual;

Tabela 3.2 – Tabela que mostra e descreve todo o conjunto de tramas enviadas do autómato para aplicação durante a criação, mudança ou edição de programas.

Tramas responsáveis pela amostragem das mensagens de estado da peça	Mensagem	Descrição
A0h | 2Bh	**OK**	Esta mensagem aparece ao operador quando todas as condições de verificação testadas na posição 1 foram validadas com sucesso, e as peças estão prontas para continuar o seu processo de assemblagem;
Qualquer trama de erro apresentada no mostrador de mensagens	**NG**	Esta mensagem aparece ao operador quando alguma das condições de validação (regiões de gel, tempo, cordão de silicone e posição) falha neste caso a peça não deve avançar no processo de assemblagem;
A0h | 15h	**TRABAJANDO**	Indica que o robô de *pick and place* está a realizar a colocação da tampa da centralina sobre a base da mesma;
A0h | 16h	**TRABAJANDO**	Indica que o robô 2 está a executar o aparafusamento da centralina;
A0h | 20h	**NG**	A ocorrência desta mensagem indica que robô 1 não trabalhou ou a peça não foi concluída com sucesso, desta forma a peça não será trabalhada pelo robô 2, sendo o robô 2 desactivado pelo autómato na próxima rotação de mesa;
A0h | 0Bh	**OK**	A ocorrência desta mensagem indica que o robô 1 terminou o trabalho com sucesso e a peça está pronta para avançar para a posição 3;
A0h | 0Dh	**MODO AUTOMÁTICO**	Esta mensagem é mostrada ao operador na comutação da célula de modo manual para o modo automático;
A0h | 0Eh	**MODO MANUAL**	Esta mensagem é mostrada ao operador na comutação da célula para o modo manual;

Tabela 3.3 – Tabela que mostra e descreve todo o conjunto de tramas enviadas do autómato para aplicação a qual faz desencadear as respectivas mensagens nos mostradores de estado de peça de cada posição.

Outras tramas enviadas pelo autómato para a aplicação via porta série número 1	Mensagem	Descrição
A0h | 08h	**(Barreira fechada)**	Trama indicadora de barreira de protecção fechada;
A0h | 09h	**(*Start* premido)**	Trama indicadora de botão *start* premido;
A0h | 11h	**(*Timer* activado)**	Trama indicadora de que a colocação do cordão de silicone foi efectuada com sucesso, e que o tempo da peça deve arrancar de imediato;
A0h | 12h	**(Barreira interrompida)**	Trama indicadora de barreira de protecção interrompida;
A0h | 13h	**(Tempo ok)**	Trama indicadora de que o tempo de cordão de silicone não foi ultrapassado e respeita os requisitos exigidos;
A0h | 14h	**(Peça Forçada)**	Trama indicadora de que as peças foram forçadas pelo operador a avançar no processo mesmo não cumprindo todos os requisitos exigidos;
A0h | 0Ah	**(Mesa terminou rotação)**	Trama desencadeada quando ocorre a rotação completa da mesa;
A0h | 2Ch	**(Centralina estragada)**	Trama indicadora de que as peças não foram forçadas pelo operador, ou seja, as peças não cumprem os requisitos exigidos e que por essa razão foram retiradas da célula (peças estragadas);
A0h | 2Eh	**(Inicio teste torque)**	Trama indicadora de inicialização de teste de torque da aparafusadora, ou seja, irá ocorrer a calibração da aparafusadora pelo robô 2.

Tabela 3.4 – Tabela que mostra e descreve todo o conjunto de tramas enviadas do autómato para aplicação e que servem de controlo para a mesma.

3.3.2. Comunicação entre o PC e o leitor de ID (porta série nº2)

Porta que serve de comunicação entre o leitor de ID's e a aplicação implementada.

É através desta porta que é enviado o número de série ("*Serial Number"*), mostrado no interface, de cada centralina que está a ser trabalhada na respectiva posição.

As configurações desta comunicação são 9600,par,8 *bits*,1 *stop bits*.

3.3.3. Comunicação entre o PC e o leitor de Tag (porta série nº3)

Porta de comunicação entre o leitor de TAG's e a aplicação implementada.

É através desta porta que é enviado o código do tabuleiro ("*Part Number*"), mostrado no interface, é este código que indica o modelo de centralina para o qual o tabuleiro encontrado na área de trabalho é compatível.

As configurações desta comunicação são 9600,par,7 *bits*,2 *stop bits.*

3.3.4. Comunicação entre a Aparafusadora e o PC (porta série nº4)

Porta de comunicação entre a aparafusadora e a aplicação implementada.

É através desta porta que são enviados, actualizados e mostrados no interface todos os dados relativos ao resultado de cada aparafusamento. Esta informação é enviada pelo controlador da aparafusadora para a aplicação.

As configurações desta comunicação são 9600,par,7 *bits*,2 *stop bits*.

As tramas enviadas através desta porta contêm toda a informação relevante e relativa ao processo de aparafusamento, ou seja, número do parafuso, valores de torque, ângulo resultantes do aparafusamento de cada parafuso, e o estado do aparafusamento ("OK" ou "NG"), todos estes dados são referentes ao aparafusamento de cada parafuso.

É através desta porta que são desencadeadas algumas das mensagens que indicam o resultado do processo de aparafusamento da centralina na posição 3 como é mostrado e descrito na tabela 3.5.

Tramas enviadas da aparafusadora para a aplicação durante o aparafusamento	Mensagem	Descrição
Conjunto de tramas indicadoras que o aparafusamento total foi concluído com sucesso	**OK**	Mensagem que ocorre na conclusão do aparafusamento efectuado pelo robô 2 e que indica o seu sucesso estando neste momento a centralina está pronta para ser retirada;
Conjunto de tramas indicadoras que o aparafusamento total não foi concluído com sucesso	**NG**	Mensagem que indica que o robô 2 não efectuou o aparafusamento ou este não foi efectuado com sucesso.

Tabela 3.5 – Tabela que mostra e descreve todo o conjunto de tramas enviadas da aparafusadora para aplicação e que são responsáveis pela actualização de algumas das mensagens do mostrador de estado de peça localizado na posição 3.

3.3.5. Comunicação entre o PC e o Autómato (porta série nº5)

Porta de comunicação entre a aplicação implementada e o autómato.

Através desta porta são enviadas, a partir da aplicação, tramas de activação/desactivação de *flags* no autómato e tramas de escrita/leitura de determinadas zonas de memória. O processo de envio destas tramas ocorre durante uma gestão de programas e na configuração inicial da célula de trabalho. Esta porta não tem relação directa com o interface de visualização do operador.

As configurações desta comunicação são 9600,par,7 *bits*,2 *stop bits*.

Na tabela 3.6 podemos visualizar o tipo de *flags* activadas/desactivadas pela aplicação no autómato;

***Flags* activadas/desactivadas pela aplicação no autómato via porta RS-232 Nº5**	**Descrição da *Flag***
Tag	Avisa o autómato de que a "Tag" lida corresponde ou não ao tipo de programa de assemblagem em actual execução na célula;
Inicialização concluída	*Flag* indicadora de que todo processo de configuração inicial da célula, se encontra concluído e que o autómato pode inicializar o processo automático;
Robô 2	*Flag* que indica que ocorreu um erro durante a fase de aparafusamento;
Tempo máximo	*Flag* indicadora que foi atingido o tempo máximo do cordão de silicone existente na peça;
Robô 1 não trabalha	A activação desta *flag* impede o robô 1 de trabalhar e ocorre quando o tempo máximo do cordão de silicone é ultrapassado e a peça não se encontra nas devidas condições para continuar o seu processo de finalização;
Setup	*Flag* que informa o autómato do sucesso ou não da operação de mudança, edição e criação de programas;

Tabela 3.6- Tabela que mostra as *flags* activadas/desactivadas no autómato durante o processo de assemblagem e que são enviadas pela porta série nº5.

A seguir é apresentada toda a informação que a aplicação solicita ao autómato no decorrer da gestão de programas, consultando zonas de memória específicas e usando a porta série número 5:

- **Na criação de um novo programa** é efectuada a leitura da *Tag* e do Id correspondentes ao programa, do número de programa a carregar no controlador de aparafusamento e no controlador de análise das regiões de gel, o número de depósito de parafusos a seleccionar na célula, o código de parafusos

a inserir nos depósitos em caso de falta, tempos de cordão de silicone (máximo e critico) e por fim o número de pontos de aparafusamento a realizar pelo robô 2;

- **Na edição ou mudança de programa** para um já existente é realizada a leitura do número do programa a editar ou a mudar;

Durante a configuração inicial da célula de assemblagem esta porta é usada para carregar todos os dados de configuração inicial para o autómato. Estes dados estão armazenados num ficheiro (ficheiro de configuração inicial mostrado na secção 3.5.1) localizado numa pasta específica do computador pessoal.

3.3.6. Comunicação entre o PC e o Robô 1 (porta série nº6)

Porta que serve de comunicação entre a aplicação implementada e o robô 1.

É através desta porta que são editados os pontos de *pick and place* e carregados novos programas no robô 1.

As configurações desta comunicação são 38400, nenhuma, 8 *bits*, 1 *stop bits*.

3.3.7. Comunicação entre o PC e o Robô 2 (porta série nº7)

Porta que serve de comunicação entre a aplicação implementada e o robô 2.

É através desta porta que são editados os pontos de aparafusamento e carregados novos programas no robô 2.

As configurações desta comunicação são 38400,nenhuma,8 *bits*,1 *stop bits*.

3.3.8. Comunicação entre o PC e a impressora (porta série nº8)

Porta que serve de comunicação entre a aplicação implementada e a impressora.

Esta porta é usada pela aplicação para o envio das tramas que contêm os erros ocorridos durante o processo de assemblagem de cada centralina. Estas tramas são recebidas e impressas pela impressora.

As configurações desta comunicação são 9600,nenhuma,8 *bits*,1 *stop bits*.

3.4. Algoritmos implementados

Nesta secção é realizado o levantamento de todos os algoritmos implementados e que contribuíram para a solução final.

Como complementação de alguns dos algoritmos são apresentadas as funções que os implementam.

3.4.1. Recepção e tratamento de tramas

A comunicação entre os vários dispositivos e a aplicação implementada é efectuada usando a biblioteca de rotinas disponibilizada pela placa da Moxa, que se encontra descrita em detalhe no Anexo A. Na Figura 3.8 é ilustrado o algoritmo usado na recepção e tratamento de tramas pela aplicação desenvolvida. É de realçar que a recepção de tramas é realizada através de rotinas que utilizam como princípio de funcionamento o recurso á "interrupção por eventos", ou seja, rotinas que se encontram em constante "escuta" da ocorrência de eventos nas portas série e que na ocorrência dos mesmos fazem desencadear a acção pré-definida.

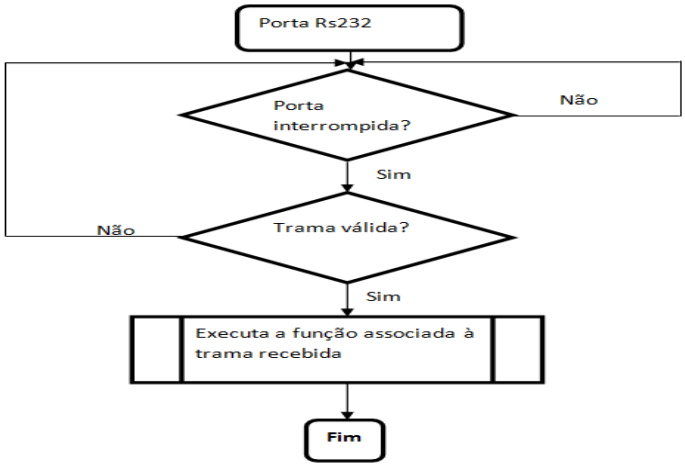


Figura 3.8 - Algoritmo usado na implementação da rotina de recepção de tramas enviadas pelos vários dispositivos ligados ao computador e recebidas na aplicação.

Através da análise da Figura 3.8 verifica-se que a após a recepção de uma trama desencadeada por algum dos dispositivos ligados ao computador, a aplicação faz a sua leitura e reconhecimento, caso a trama seja válida a aplicação desencadeia de imediato a execução da tarefa correspondente.

As funções utilizadas na implementação deste algoritmo encontram-se descritas nas Figuras 3.9 e 3.10.

Abertura e Configuração das portas série

```
1.   //Abertura da porta
2.   ret=sio_open(2);

3.   if(ret==SIO_OK)
4.   {
5.           //Configuração dos parâmetros da porta série
6.           ret=sio_ioctl(2,B9600,P_EVEN|BIT_8|STOP_2);

7.           if(ret==SIO_OK)
8.             {
9.                   //Activar a espera por "eventos" na porta
10.                  ret=sio_term_irq(2,CB_PORT1,char(0x0F));

11.                  if(ret!=SIO_OK)
12.                    {
13.                          s.Format(_T("Error: Port1"));
14.                          AfxMessageBox(s);
15.                    }
16.                  else
17.                          sio_flush(2,2);
18.            }
19.  }
```

Figura 3.9 - Função responsável pela abertura e configuração da porta série. Neste caso em específico temos a configuração e activação da interrupção por eventos na porta série número 2.

Recepção e Processamento das tramas

```
1.   void CALLBACK CB_PORT1(int port)
2.   {
3.       int len;
4.       char buf[4];

5.       / /Leitura de dados da porta
6.       len=sio_read(port,buf,3);
7.                    .
8.                    .
9.                    .
10.  }
```

Figura 3.10 - Função responsável pela leitura e processamento da trama recebida, na interrupção da porta série número 2.

As rotinas das Figuras 3.9 e 3.10 fazem parte da biblioteca da moxa e foram adaptadas de modo a melhor satisfazer as nossas necessidades. No Anexo A temos a descrição detalhada de cada uma das rotinas utilizadas desta biblioteca.

3.4.2. Algoritmo usado na activação/desactivação e leitura/escrita de zonas de memória no autómato

Na comunicação com o autómato via porta série nº 5 é usado o protocolo *hostlink* (Omron). Este protocolo já tinha sido implementado anteriormente em outros projectos tendo sido construída uma biblioteca de funções (dll).

Das funções implementadas foram utilizadas as seguintes:

Funções de leitura:

- **Char* AR_area_read(int node,int begin, int num_word)**

  Devolve a trama necessária para efectuar uma leitura do estado dos relés auxiliares (AR).

  Depois de calculada a trama é efectuado o seu envio via porta série e utilizando as funções da biblioteca da moxa.

  Como argumentos da função temos: Número do nó (0 no caso de existir apenas um PLC), nº da zona onde se inicia a leitura, número de posições a ler a partir da posição inicial.

  Exemplo: Char* a= AR_area_read(0,0,1); O valor devolvido pela função deverá ser "@00RJ00000001??*" esta trama está agora pronta a ser enviada para o PLC.

- **Char* DM_area_read(int node,int begin, int num_word)**

  Devolve a trama necessária para uma leitura do estado da memória de dados (DM).

  Os campos da função são idênticos aos explicados anteriormente.

  Exemplo: Char* a= DM_area_read(0,0,1); O valor devolvido pela função deverá ser "@00RD00000001??*" esta trama está agora pronta a ser enviada para o PLC.

Funções de escrita:

- **Char* AR_area_write(int node,int begin_word, int data_begin)**

  Esta função calcula a trama necessária para uma escrita nos relés auxiliares.

  Exemplo**:** Char* a= IR_area_write(0,10,1); O valor devolvido pela função é "@00AR00100001??*" esta trama está agora pronta a ser enviada para o PLC.

  Como argumentos da função temos: Número do nó (0 no caso de existir apenas um PLC); nº da zona onde se inicia a escrita; valor a escrever (só é possível escrever para uma posição de memória de cada vez).

- **Char* DM_area_write(int node, int begin_word, int data_begin)**

  Idêntica à anterior, adaptada para as áreas de memória de dados.

3.4.3. Algoritmo usado na implementação da rotina *timer*

O esquema de ligação entre os vários dispositivos usados na gestão e visualização do *timer* é mostrado na Figura 3.11.

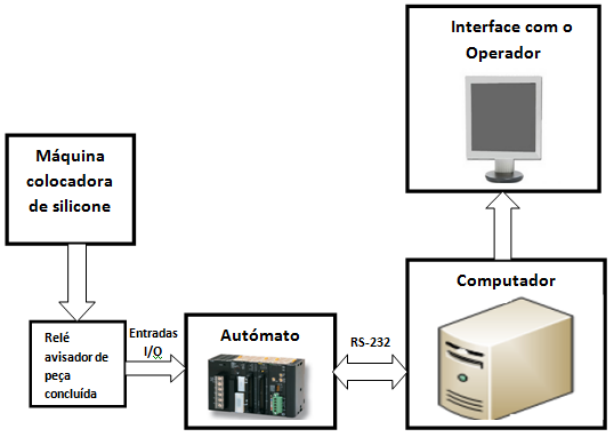


Figura 3.11 – Esquema de ligação entre os vários dispositivos que têm interferência na gestão e visualização do *timer*.

As tramas enviadas pelo autómato e que interferem na gestão do *timer* do cordão de silicone controlado pela aplicação, são:

- **"A0h | 10h"**, "**A0h | 11h"**, "**A0h | 17h", "A0h | 0Ah", "A0h" | 2Ch";**

Flags activadas/desactivadas pela aplicação no autómato via porta série nº5 no decorrer do *timer*:

- **Tempo máximo** → *Flag* activada quando o tempo máximo do cordão de silicone existente na peça é ultrapassado;

- **Robô1 não trabalha** → *Flag* activada se o tempo máximo do cordão de silicone é ultrapassado e a peça não se encontra nas devidas condições para continuar o processo de finalização. Esta *flag* quando activada impede o robô 1 de trabalhar;

Nesta aplicação podemos ter até três timers activos em simultâneo. Este número de *timers* são suficientes tendo em conta a cadência das peças fornecidas pela máquina auxiliar de colocação do cordão de silicone e das regiões de gel.

Como complementação dos algoritmos da rotina *timer* vamos, de seguida mostrar de forma gráfica a execução da mesma.

As acções ocorridas quando a máquina auxiliar termina a colocação do cordão de silicone são as seguintes:

- O relé que está directamente ligado à máquina auxiliar envia um sinal para activação de um *timer* no autómato;

- O autómato informa o computador que um *timer* foi inicializado;

- Por sua vez a aplicação desencadeia a interrupção correspondente, a qual é responsável pela realização do reconhecimento da trama e pela amostragem do valor máximo do *timer* no interface do operador, começando a sua contagem decrescente. O valor máximo deste *timer* é configurado e lido do ficheiro do programa (descrito na secção 3.5.2 deste capítulo) actualmente em execução na célula. Neste caso particular o valor do *timer* é definido com o valor de 200 s.

É ilustrado na Figura 3.12 o valor do *timer* associado à peça que vai iniciar o seu processo de assemblagem.

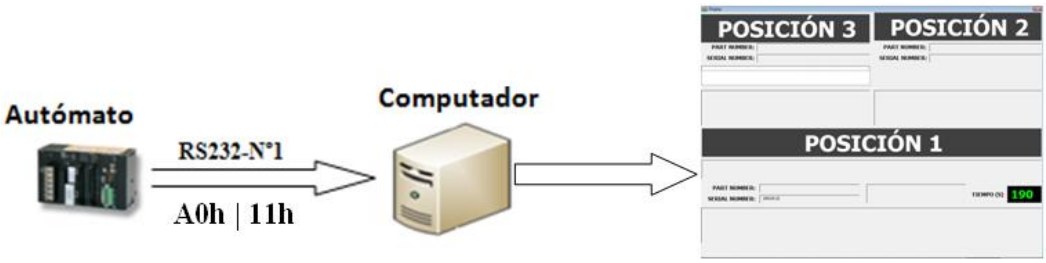


Figura 3.12 – Amostragem do *timer* no interface após a recepção da trama de activação enviada pelo autómato.

A fase seguinte do processo é a entrada dos elementos constituintes da centralina (tampa + base) na área de assemblagem desencadeando-se as seguintes acções:

- O *timer* da peça localizada na posição 1 continua a sua amostragem e contagem decrescente;
- São analisados todos os requisitos necessários para o avanço das peças no processo;

Caso surja outro sinal de *timer*, como o anterior, então é desencadeado um segundo *timer* que se vai associar à segunda peça a dar entrada na célula. É de realçar que este *timer* é apenas activado internamente e não é mostrado no monitor do computador enquanto a primeira peça não avançar para a posição 2.

Supondo um exemplo particular em que foi disparado um segundo *timer* espaçado de 10 segundos do primeiro, como se pode observar pela Figura 3.13, o *timer* ainda em visualização é o *timer* 1 da primeira peça. Mas internamente o valor do *timer* 2 já se encontra a ser decrementado, tendo o valor de 198 segundos, para este exemplo específico.

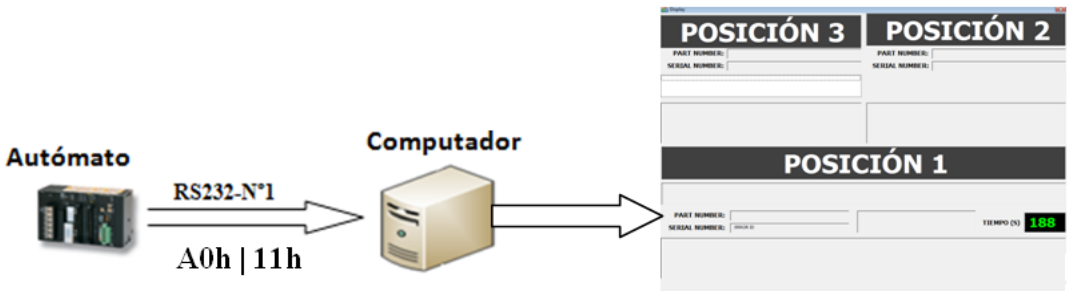


Figura 3.13 – Activação de um segundo *timer*, neste momento existem dois *timers* activos mas só o *timer* 1 é visualizado no interface.

Depois da análise e validação dos requisitos de avanço no processo: regiões de gel, cordão de silicone, tempo de silicone e a correcta posição das peças no tabuleiro, podem surgir três tipos de tramas enviadas à aplicação pelo autómato:

- Trama que informa que as peças constituintes da centralina foram forçadas a avançar no processo, pelo operador;
- Trama que informa que as peças constituintes da centralina estão prontas para avançar no processo;
- Trama que informa que as peças constituintes da centralina devem ser retiradas pois não cumprem os requisitos mínimos exigidos (peças estragadas).

De acordo com a trama recepcionada pela aplicação, a mesma pode desencadear uma das seguintes acções:

- Se a trama recebida indica que as peças não cumprem os requisitos mínimos (estragadas), então será mostrado o interface da Figura 3.14, no qual é possível verificar que o *timer* 1 foi eliminado e o *timer* agora em visualização é o *timer* 2, referente á nova peça que vai dar entrada na célula.
- Caso as peças cumpram todos os requisitos para avançar no processo de assemblagem, ou então tenham sido forçadas a avançar no processo pelo operador, o interface mostrado será o ilustrado na Figura 3.15 no qual podemos verificar que o *timer* 1 continua a sua contagem decrescente.

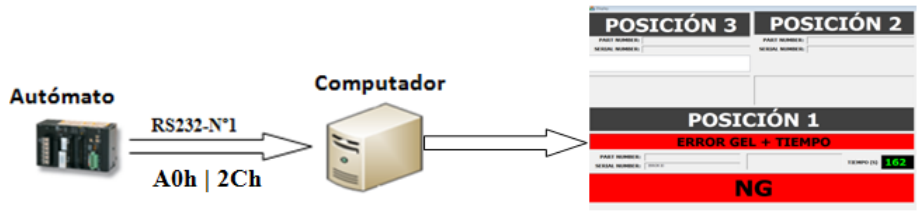


Figura 3.14 – Autómato envia á aplicação a trama a indicar que a peça deve ser retirada, a aplicação, por sua vez, apaga o *timer* actualmente em visualização e mostra o *timer* associado á nova peça que vai dar entrada na célula de trabalho.

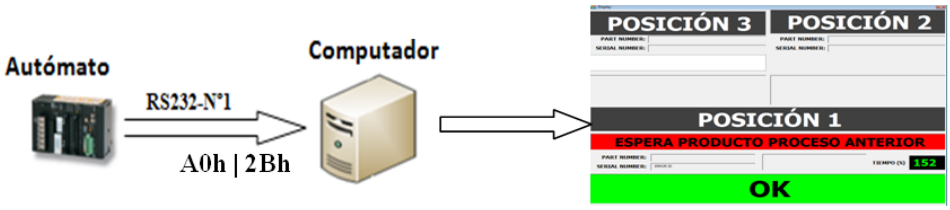


Figura 3.15 – Autómato envia à aplicação a trama a indicar que as peças vão continuar o processo de assemblagem, a aplicação por sua vez continua a amostragem do respectivo *timer* mostrando também o estado da peça como "OK".

No caso em que as peças são forçada ou cumprem todos os requisitos de avanço então o autómato, faz a mesa rodar enviando a trama respectiva para a aplicação ("A0h | 17h"), a qual responde mostrando o interface da Figura 3.16, no qual se pode observar que o *timer* agora visualizado é o *timer 2*, mas internamente o *timer 1* mantém a sua contagem pois será necessário verificar se o mesmo não se esgota até à entrada das peças na posição 2. O *timer 1* tem agora o valor de 151 segundos para este exemplo particular.

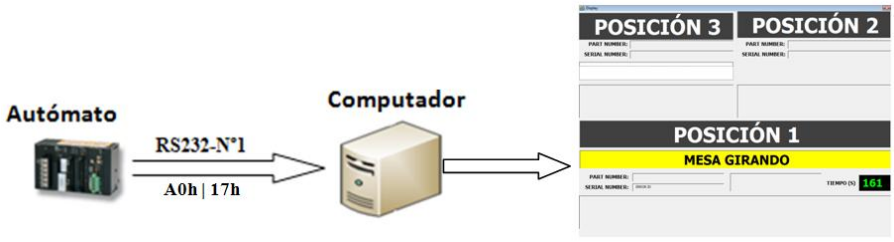


Figura 3.16 – Autómato envia à aplicação a trama a indicar que a mesa vai iniciar a rotação, a aplicação continua a execução do *timer* que estava em visualização mas apaga-o do interface mostrando agora no interface o *timer* da próxima centralina que vai dar entrada na célula de trabalho.

Após a rotação completa da mesa indicada pelo sensor indutivo e pela chegada da trama "A0 h | 0A h" à aplicação, são executadas as seguintes acções:

- A aplicação avalia o *timer* da peça encontrada na posição 2;
- Caso cumpra os requisitos exigidos então o autómato segue o seu algoritmo, mandando o robô 1 trabalhar (condição por defeito), Figura 3.17;
- Caso o limite de tempo do cordão de silicone estabelecido tenha sido ultrapassado é activada uma *flag* no autómato que cancela o trabalho do robô 1 (*flag* robô 1 não trabalha) e é declarado o estado das peças como "NG", Figura 3.18;

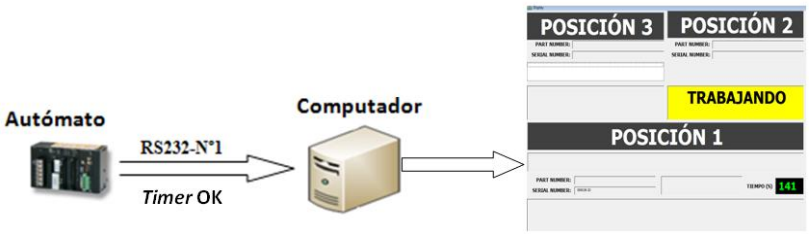


Figura 3.17 – A mesa rodou correctamente e a aplicação analisou o *timer* da peça que se encontra na posição 2 verificando que cumpre os limites estabelecidos desta forma é dada a ordem para o robô 1 iniciar o trabalho e o *timer* associado á peça é eliminado.

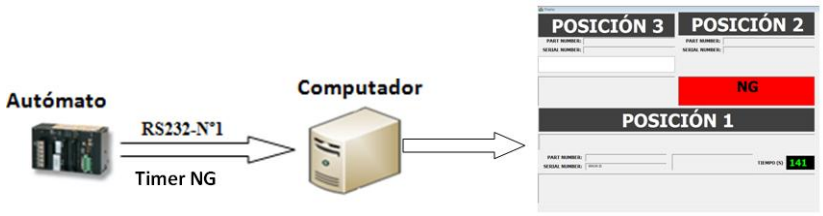


Figura 3.18 – A mesa rodou correctamente e a aplicação procedeu à análise do *timer* da peça que se encontra na posição 2 verificando que não cumpre os limites estabelecidos logo o robô 1 é impedido de trabalhar e a peça é considerada estragada.

Depois de visualizada toda a execução gráfica associada à rotina do *timer*, vamos de seguida apresentar a sua implementação e análise.

São duas as funções implementadas que fazem a selecção, contagem e amostragem do tempo de cordão de silicone. Estas funções são executadas sempre que existe o envio por parte do autómato da trama de activação do *timer* ("A0h |11h"). A primeira função a entrar em execução é a função selecciona *timer*, após a sua finalização é executada a função que realiza a contagem e amostragem do *timer*. Estas funções são descritas a seguir de forma mais pormenorizada.

Função "selecciona e inicializa novo *timer*"

Após a recepção de um novo *timer* esta função faz a verificação dos *timers* activos, e activa o novo *timer* associando-o à nova peça. A implementação desta função é apresentada na Figura 3.19.

```
1.    // Verificação e activação do timer

2.    if(!m_bt3 && m_bt2 && m_bt1)
3.    {
4.          m_bt3=true;
5.          m_tt3=(float)rd;
6.    }

7.      if(!m_bt3 && !m_bt2 && m_bt1)
8.    {
9.          m_bt2=true;
10.         m_tt2=(float)rd;
11.   }

12.   if(!m_bt1 && !m_bt2 && !m_bt3)
13.   {
14.         m_tt1=(float)rd;
15.         m_bt1=true;
16.   }

17.   // Se ainda nenhum timer está a ser mostrado no
18.   //interface então será mostrado o timer 1

19.   if(m_ttshow==-1)
20.         m_ttshow=1;
```

Figura 3.19 – Implementação da função responsável pela selecção e inicialização de um novo *timer* associando-o á nova peça.

Observando a Figura 3.19 vai-se proceder de seguida á análise da função "selecciona e inicializa novo *timer*":

- **Linha de código número 1 até à linha número 16** → Verificação dos *timers* activos até ao momento e inicialização de um novo *timer*.

- **m_bt1, m_bt2 e m_bt3** → são respectivamente as variáveis de activação do t*imer* 1, *timer* 2, *timer* 3.
- **Rd** → é o valor máximo do *timer* referenciado no ficheiro do programa explicado neste capítulo na secção 3.5.2.
- **m_ttshow** → define o *timer* a visualizar no interface, a visualização está dependente do seu valor quando igual a:
  1- Visualização do *timer* 1;
  2- Visualização *do timer* 2;
  3- Visualização *do timer* 3;

Função "decrementa e mostra *timer*"

Função responsável pela contagem decrescente e amostragem do *timer* no interface esta função pode ser visualizada na Figura 3.20.

```
1.    if(m_bt1) //Verifica se timer 1 está activo
2.    {
3.                m_tt1-=0.5;  //Decrementa timer 1

4.                if(m_tt1<0)
5.                m_tt1=0;
6.    }

7.    if(m_bt2)
8.    {
9.                m_tt2-=0.5;

10.               if(m_tt2<0)
11.               m_tt2=0;
12.   }

13.   if(m_bt3)
14.   {
15.               m_tt3-=0.5;

16.               if(m_tt3<0)
17.               m_tt3=0;
18.   }

19.   switch(m_ttshow) //Escolhe timer a mostrar no interface
20.   {
21.               case 1:´
22.                     m_ttshow_t=m_tt1;
23.                     break;

24.               case 2:
25.                    m_ttshow_t=m_tt2;
26.                    break;

27.               case 3:
28.                   m_ttshow_t=m_tt3;
29.                   break;
30.   }
```

```
31.    if(m_ttshow>0)
32.    {
33.           //Verifica se o valor do timer em visualização é menor ou igual a zer

34.                if(m_ttshow_t<=0)
35.                 {
36.                     m_msg_pos1_temp.put_ForeColor(0x0000ff);
37.
38.                     if(m_ttshow_t==0)
39.                       {
40.                              m_timexeded=true;

41.                              if(!m_flagtempo)
42.                                {
43.                                   m_plc->PLC_Flag_tempmax();
44.                                   m_ttshow=-1;
45.                                }

46.                                else
47.                              m_plc->PLC_Flag_braco1trabalha();
48.                       }
49.                 }

50.                sss.Format(_T("%.0f"),m_ttshow_t);
51.                m_msg_pos1_temp.SetWindowText(sss);
52.    }
```

Figura 3.20 – Implementação da função responsável pela contagem dos *timer* activos, e amostragem no interface do *timer* da peça que se encontra na posição 1.

Após a chegada de um novo *timer* a aplicação desencadeia a função "selecciona *timer*" e só após a conclusão da execução da mesma é executada a função "decrementa e mostra *timer*", mostrada na Figura 3.20 e analisada seguidamente:

- **Linha de código número 1 até à linha número 18** → é verificado quais os *timers* já inicializados e é realizada a sua contagem decrescente;

- **Linha de código número 19 à linha número 30** → é verificado qual o valor da variável que controla a amostragem do *timer* (m_ttshow), a qual pode ter como valores o 1, o 2 e 3, sendo respectivamente o *timer* 1, o 2 e o 3;

- **Linha de código número 31 à linha número 49** → é verificado se o valor do *timer* (m_ttshow_t) ultrapassou os limites aceitáveis em caso afirmativo é activada no autómato a *flag* de tempo máximo e a *flag* que impede o robô 1 de trabalhar;

- **Linha de código número 50 e 51** → são responsáveis pela amostragem no interface do *timer* da peça que se encontra na posição 1.

Função "Mesa vai Iniciar Rotação"

Esta função, mostrada na Figura 3.21 entra em execução após a recepção da trama "A0h | 17h" (mesa vai iniciar rotação).

```
1.  m_peca[3]=m_peca[2];
2.  m_peca[2]=m_peca[1];

3.  if(m_bt2)
4.  m_ttshow=2;
```

Figura 3.21 – Conjunto de instruções executadas após a recepção da trama indicadora que a mesa vai iniciar a rotação.

Observando a Figura 3.21 vamos proceder á sua análise:

- **Linhas de código número 1 e 2** → é realizada uma rotação nas estruturas das peças, pois as peças que se encontravam na posição 1 passam para a posição 2, e as da posição 2 passam para a posição 3, desta forma é necessário que os dados das respectivas peças acompanhem a sua rotação no processo.

- **Linhas de código número 3 e 4** → é verificado se o *timer* 2 está activo, em caso afirmativo será mostrado no interface, quanto ao *timer* 1 continuará a sua contagem internamente até à chegada à posição 2;

Função "Mesa Terminou Rotação"

Após a completa da rotação da mesa é executado na aplicação o conjunto de instruções visualizado na Figura 3.22.

```
1.      if(m_bt1 || m_bt2 || m_bt3)
2.      {
3.              m_bt1=m_bt2;
4.              m_bt2=m_bt3;
5.              m_bt3=false;
6.              m_tt1=m_tt2;
7.              m_tt2=m_tt3;
8.              m_tt3=rd;
9.              m_flagtempo=false;
10.             m_ttshow=1;
11.     }
```

Figura 3.22 – Conjunto de instruções executadas após a recepção da trama indicadora que a mesa terminou a rotação.

Observando a Figura 3.22 vamos proceder á análise da função "Mesa Terminou Rotação":

- **Linha de código número 1 à linha número 11** → é verificado se algum dos *timers* existentes se encontra activo, em caso afirmativo é realizada uma rotação de *timers*, ficando o *timer* 1 com o valor do *timer* 2, ou seja, com o valor do *timer* que se encontra na posição 1, e o *timer* 2 fica com o valor de *timer* 3 por sua vez o *timer* 3 ficará com o valor definido no ficheiro do programa em actual execução e explicado neste capítulo na secção 3.5.2. Na última linha é definido que o *timer* a mostrar no interface é o *timer* 1, que é o antigo *timer* 2.

Função "Centralina Estragada"

Após a validação de todos os requisitos na posição 1 as peças podem conter falhas de colocação do cordão de silicone, regiões de gel, posição e tempo de cordão de silicone desta forma não devem avançar no processo, embora o operador as possa forçar a avançar caso entenda que a falha existente não prejudique o funcionamento futuro da centralina.

Caso o operador defina que as peças constituintes da centralina não devem avançar, então serão retiradas e consideradas estragadas neste caso o autómato avisa a aplicação enviando a trama "A0h | 2Ch", por sua vez a mesma executa o conjunto de instruções mostradas na Figura 3.23.

```
1.    //imprimir etiqueta de erros de assemblagem

2.    m_msg_pos1_etiq.SetWindowText(_T("ETIQUETA"));
3.    m_etiqueta=true;
4.    Imprimir_erro(m_peca[0],ho, data);

5.    m_peca[0].s_gel=0;
6.    m_peca[0].s_ID=_T("");
7.    m_peca[0].s_pec_f=0;
8.    m_peca[0].s_selic=0;
9.    m_peca[0].s_tempo=0;

10.   if(m_bt1 || m_bt2 || m_bt3)
11.   {
12.           m_bt1=m_bt2;
13.           m_bt2=m_bt3;
14.           m_bt3=false;
15.           m_tt1=m_tt2;
16.           m_tt2=m_tt3;
17.           m_tt3=rd;
18.           m_flagtempo=false;

19.           if(m_bt1 || m_bt2 || m_bt3)
20.                   m_ttshow=1;

21.           else
22.           {
23.                   m_ttshow=-1;
24.                   m_msg_pos1_temp.SetWindowText(_T(""));
25.           }
26.   }
```

Figura 3.23 – Conjunto de instruções executadas após a recepção da trama indicadora que a centralina se encontra estragada e não vai avançar no processo.

Como podemos observar pela Figura 3.23 a execução da função "Centralina Estragada" é idêntica á função "Mesa Terminou Rotação", tendo simplesmente em acréscimo a impressão de uma etiqueta de erro a colocar na centralina com a informação do tipo de erro ocorrido e que não permitiu o avanço das peças constituintes da centralina no processo de assemblagem.

Os algoritmos em fluxograma usados para a implementação de todo o processo de gestão de *timers* podem ser consultados no Anexo C.

3.4.4. Algoritmo geral da rotina "gestão de programas"

Considera-se como gestão de programas a edição, criação de novos programas para a célula e a mudança do programa de assemblagem na célula. Exceptuando a mudança de

programa que pode ser realizada com a célula em modo manual ou em modo automático todas as outras operações apenas são realizáveis em modo manual.

O algoritmo simplificado usado na construção da rotina "Gestão Programas" é visualizado na Figura 3.24. Todas as rotinas visualizadas neste algoritmo podem ser consultadas em mais detalhe no Anexo C.

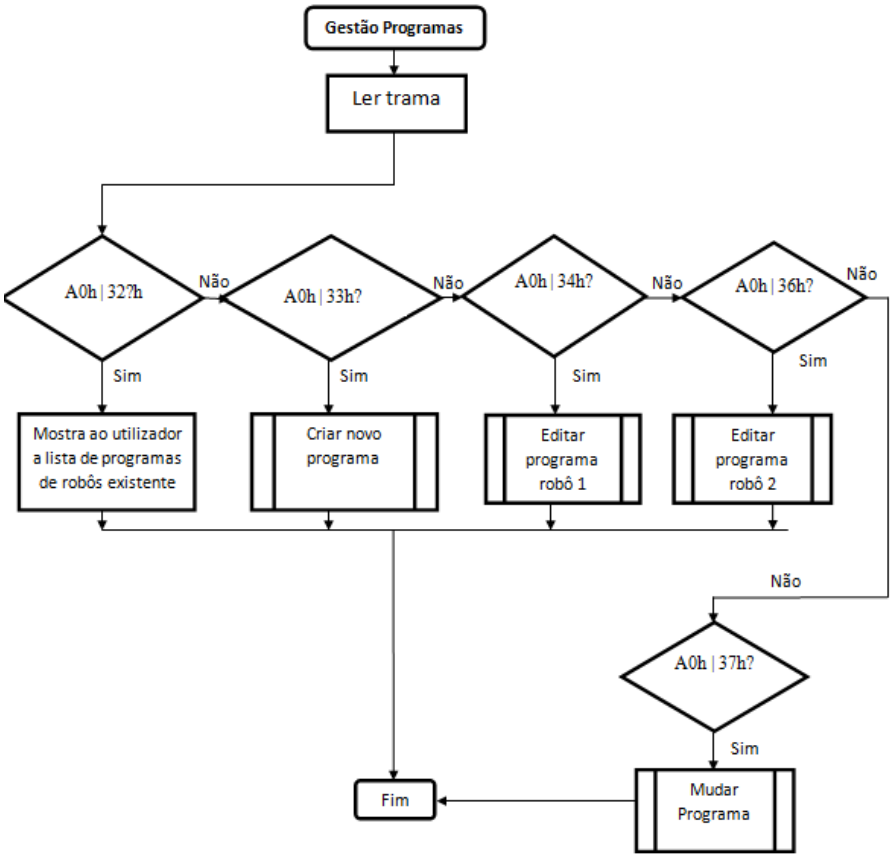


Figura 3.24 – Algoritmo geral usado na implementação da rotina "Gestão de Programas".

Como complementação do algoritmo mostrado na Figura 3.24 vamos de seguida apresentar execução das suas funções.

A configuração de programas é realizada entre a consola do autómato → autómato → aplicação → robôs (Figura 3.25).

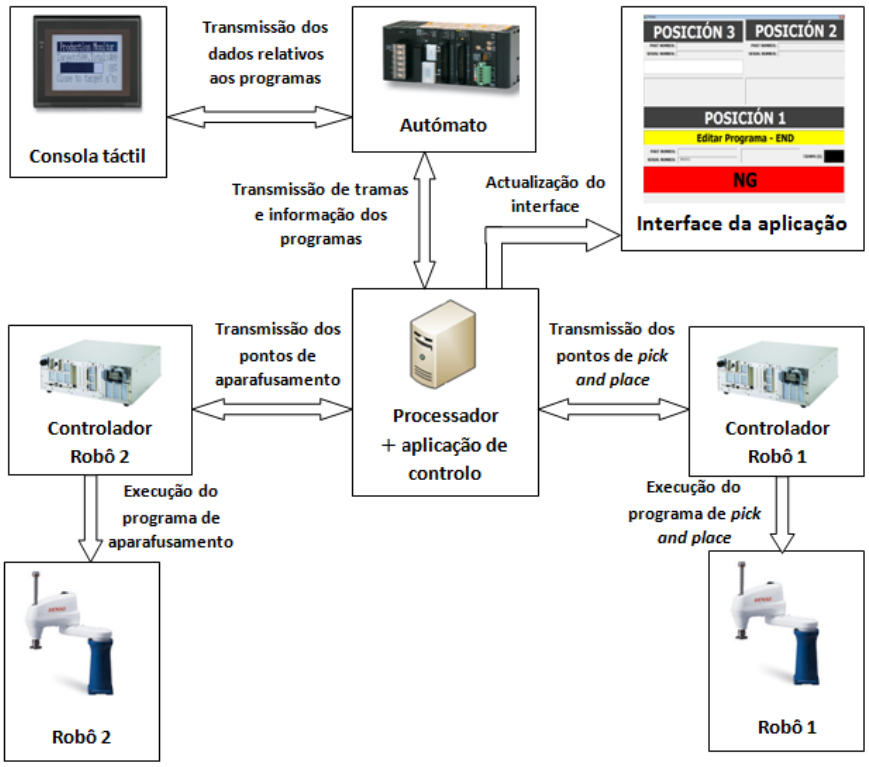


Figura 3.25 – Esquema que mostra a interligação e tramas trocadas entre os vários dispositivos intervenientes na gestão de programas.

Para a inicialização de toda a gestão de programas é necessário primeiramente a introdução de uma *password* na consola do autómato a qual restringe o acesso a estes menus por parte do operador comum. Para a introdução da *password* utiliza-se o ecrã visualizado na Figura 3.27, nesta fase a célula tem de estar obrigatoriamente em modo manual (Figura 3.26).

É de notar que a *password* é previamente definida no ficheiro de configuração inicial descrito neste capítulo (secção 3.5.1). Esta *password* é carregada para o autómato durante a configuração inicial da célula.

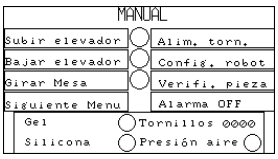


Figura 3.26 – Ecrã apresentado na consola do autómato com a célula a funcionar em modo manual.

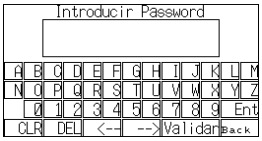


Figura 3.27 – Neste ecrã é possível introduzir a *password* de acesso ao menu de gestão de programas.

Após a validação da *password*, é apresentado o ecrã 'Gestión Programas', ilustrado na Figura 3.28.

Figura 3.28 – Ecrã de gestão de programas apresentado na consola do autómato, após a validação da *password.*

O ecrã da Figura 3.28 possibilita as seguintes operações:

1. **'Nuevo Programa'** → Permite ao operador a criação de um novo programa de configuração para toda a célula (Figura 3.31);
2. **'Editar Programa'** → Permite ao operador a edição dos programas a executar pelo robô 1 (Figura 3.29) ou pelo robô 2 (Figura 3.30);
3. **'Cambiar Programa'** → Permite ao operador a mudança do programa a executar na célula (Figura 3.32);
4. **'Menu Anterior'** → Permite ao operador retroceder para o ecrã 'Manual' (Figura 3.26).

3.4.5. Edição Programa/Robô 1 ou Robô 2

Como complemento, é mostrado na Figura 3.29 toda a interacção entre a consola do autómato e o interface da aplicação durante a edição de programa no robô 1.

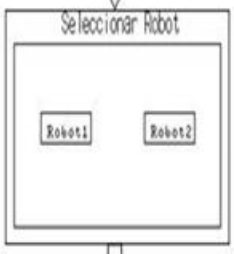
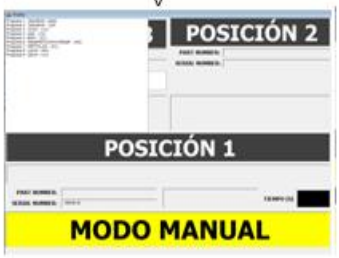
Ecrãs Autómato /Aplicação desenvolvida	Descrição
	Este é o ecrã geral de gestão de programas apresentado na consola do autómato e que permite a criação, edição e mudança de programas na célula de assemblagem. Depois de premido o botão "**Editar Programa**" surge o interface seguinte;
	No ecrã da consola do autómato solicita-se ao operador a escolha do robô a editar. A identificação de todos os programas disponíveis no computador e prontos a ser editados são mostrados no canto superior esquerdo do monitor do computador, como se pode visualizar na figura ao lado.
	No ecrã da consola (ecrã mais à esquerda) solicita-se ao operador que seleccione o programa a editar. Este ecrã é mostrado independentemente da opção seleccionada no ecrã anterior (robô 1 ou robô 2). Caso o robô seleccionado anteriormente seja o robô 1 então o processo de edição está concluído, sendo enviado para a aplicação a trama "**A0h 34h**", a aplicação por sua vez procede à leitura dos novos pontos *de pick and place* recorrendo ao robô 1. Por fim os pontos são alterados no ficheiro do programa editado.

Figura 3.29 – Visualização de todo o processo de edição de programas no robô *pick and place* e no robô de aparafusamento.

3.4.6. Edição Programa/Robô 2

Caso a opção seleccionada no ecrã "Seleccionar Robot" seja o Robô 2 serão visualizados para além dos ecrãs mostrados anteriormente (Figura 3.29) os ecrãs da Figura 3.30.

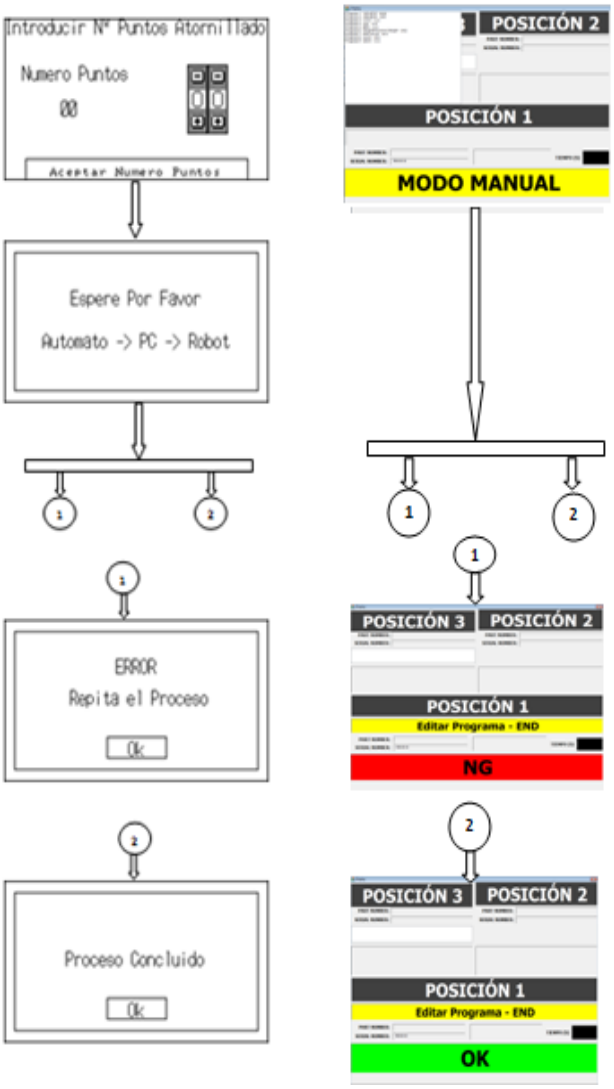
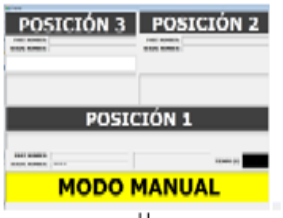
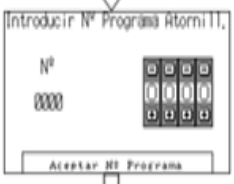
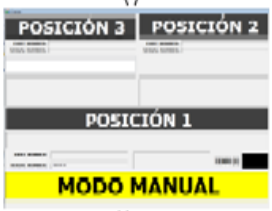
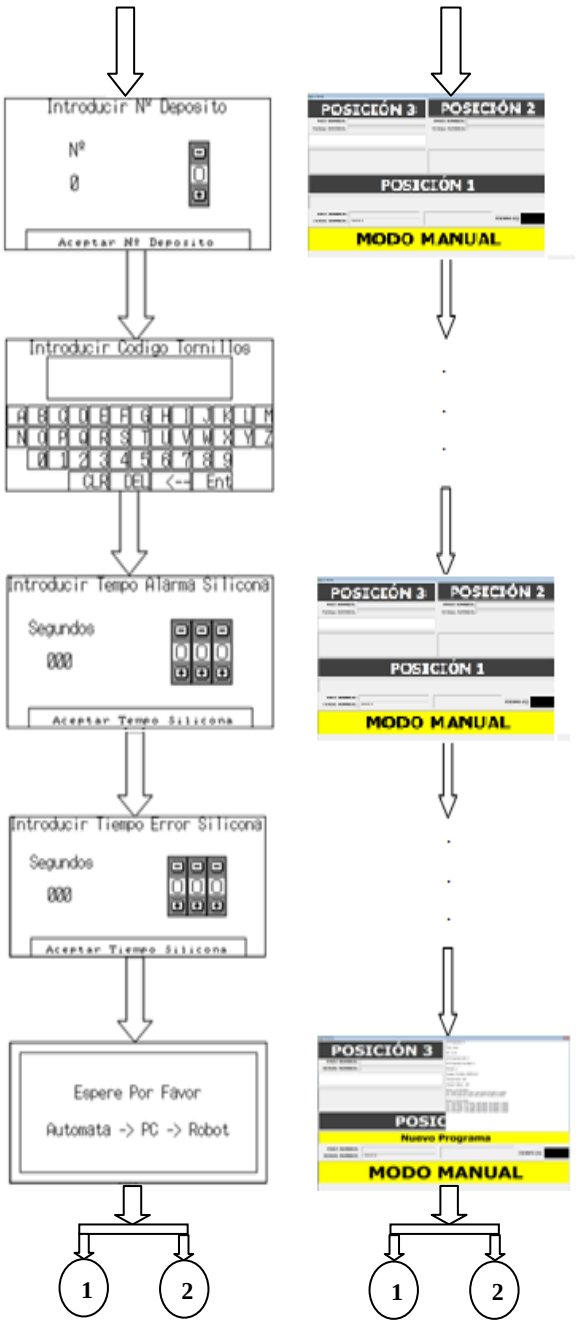
Ecrãs Autómato /Aplicação desenvolvida	Descrição
	Na consola do autómato é visualizado o número de pontos de aparafusamento do programa seleccionado anteriormente os quais podem ser editados. Quanto ao interface apresentado no computador não sofre alteração pois neste estágio não é realizada nenhuma comunicação entre o autómato e a aplicação. Depois da introdução do Id do programa a editar no caso do robô 1 ou o número de pontos de aparafusamento no caso do robô 2 é visualizada a janela do lado esquerdo a qual indica que a aplicação está a ler do autómato todos os dados editados. No final da edição a aplicação guarda todos os dados no ficheiro e por fim informa o autómato do sucesso ou não da operação. Se a edição de programa não se processar de acordo com o pretendido a aplicação desactiva *flag setup* (=0) via porta série número 5 e são visualizados os seguintes ecrãs: do lado esquerdo (ecrã do autómato), do lado direito (ecrã do monitor); Caso a edição de programa se processar com sucesso a aplicação activa *flag setup* (=1) via porta série número 5 e são visualizados os seguintes ecrãs: do lado esquerdo (ecrã autómato), do lado direito (ecrã do monitor);

Figura 3.30 – Edição de programa no robô de aparafusamento (robô 2).

3.4.7. Criação de Novo Programa

Como complemento, é apresentado na Figura 3.31 toda a interacção entre a consola do autómato e o interface da aplicação no decorrer de uma criação de um novo programa de assemblagem.

Ecrãs Autómato /Aplicação desenvolvida	Descrição
	Depois de premido o botão "**Nuevo Programa**", a execução de todo o processo de criação de novo programa é o seguinte;
	Solicita-se ao operador através da consola do autómato (ecrã apresentado mais à esquerda) a introdução da Tag, ou seja o código do tabuleiro a associar a este novo programa. No ecrã do monitor do computador é visualizada a mensagem a informativa de que a célula está em funcionamento manual;
	Nesta fase, no ecrã da consola, solicita-se ao operador a introdução dos primeiros dígitos descritos no ID da centralina que vai ser trabalhada pela célula com este novo programa. Estes dígitos servem para que seja garantido que a centralina a trabalhar é realmente a correcta e corresponde aos moldes do tabuleiro inserido na célula.
 	No ecrã da consola, solicita-se ao operador a introdução do número de programa de aparafusamento a carregar no controlador da aparafusadora e a ser associado a este novo programa. O ecrã apresentado no monitor é o mesmo que no processo anterior, pois ainda não foi realizada qualquer comunicação entre o autómato e a aplicação. Nesta fase o autómato limita-se a guardar os dados introduzidos na consola em posições de memória específicas;
	No ecrã da consola é solicitado ao operador a introdução do número de programa a carregar no controlador de análise das regiões de gel o qual vai ser associado a este novo programa.



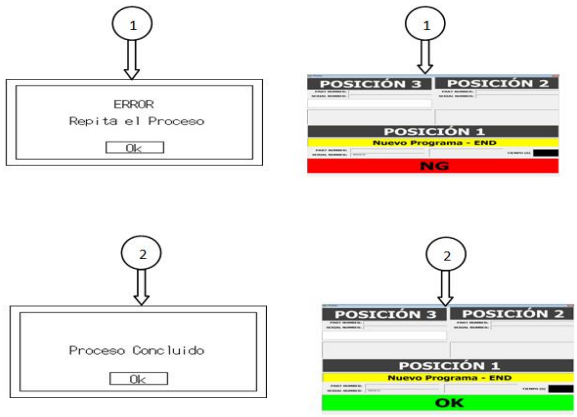
Neste ecrã é solicitado ao operador a introdução do número do depósito de parafusos a activar na execução deste novo programa de assemblagem, ou seja, é este o depósito a que o robô 2 recorre para retirar os parafusos para o aparafusamento da centralina;

Neste ecrã solicita-se ao operador a introdução da referência dos parafusos a inserir no depósito em caso de falta.

Neste ecrã é solicitado que operador realize a introdução do tempo de cordão de silicone para o qual o alarme deve ser accionado, avisando o operador que o tempo do cordão de silicone se encontra a atingir o limite;

Neste ecrã solicita-se ao operador a introdução do tempo máximo para o cordão do silicone, o qual quando ultrapassado a peça deve ser retirada da célula considerando a mesma como estragada.

Nesta fase o autómato comunica com o computador, avisando-o da chegada de um novo programa, por sua vez a aplicação consulta as posições de memória onde se encontram os dados introduzidos nos ecrãs anteriores e procede à sua leitura via porta série número 5 por fim todos os dados lidos são guardados num ficheiro específico, depois pode-se visualizar no canto direito do interface todos os dados do programa criado, por fim o autómato é avisado do sucesso ou não da operação.

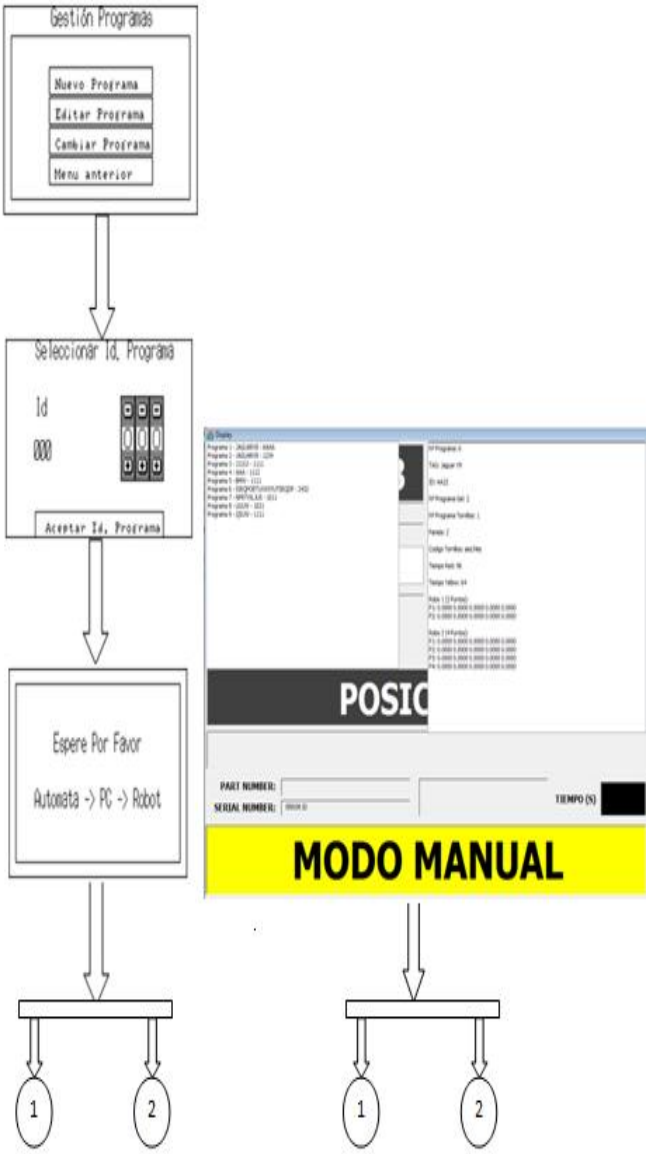
<table>
<tr>
<td> 
</td>
<td>Se a criação de um novo programa de assemblagem não se processar de acordo com o pretendido a aplicação desactiva flag setup (=0) via porta série número 5 e são visualizados os seguintes ecrãs: do lado esquerdo (ecrã do autómato), do lado direito (ecrã do monitor);

Se a criação de um novo programa de assemblagem se processar com sucesso a aplicação activa flag setup (=1) via porta série número 5 e são visualizados os seguintes ecrãs: do lado esquerdo (ecrã autómato), do lado direito (ecrã do monitor);</td>
</tr>
</table>

Figura 3.31 - Sequência de ecrãs mostrados durante a criação de um novo programa de assemblagem.

3.4.8. Mudança de Programa

Como complemento, é apresentado na Figura 3.32 toda a interacção entre a consola do autómato e o interface da aplicação no decorrer da operação mudança do programa de assemblagem.

Ecrãs Autómato /Aplicação desenvolvida	Descrição
	Depois de premido o botão "**Cambiar Programa**", a sequência de ecrãs apresentada será a seguinte. É visualizado no canto esquerdo do interface a listagem de todos os programas e no canto direito é mostrado todas as características do programa em actual execução. Quanto à consola apresenta ao operador o programa em actual execução na célula e solicita a sua mudança para um dos programas existentes na célula; Depois da selecção na consola do novo programa a carregar na célula, é enviada para a aplicação a trama indicativa da ocorrência da mudança de programa **"A0h | 37h"** por sua vez é desencadeada a função correspondente a qual faz todo o carregamento das características do programa seleccionado consultando para isso o ficheiro do programa correspondente, alterando também o programa de inicialização localizado no ficheiro de configuração inicial. Na conclusão de todo o processo o autómato é informado do resultado da operação.

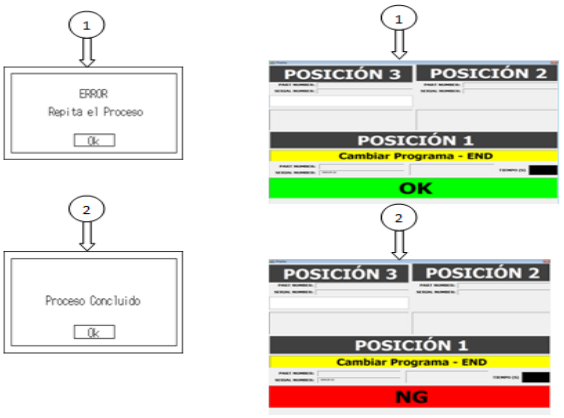
<table>
<tr>
<td rowspan="2"> 
</td>
<td>Se a operação mudança de programa de assemblagem não se processar de acordo com o pretendido a aplicação desactiva flag setup (=0) via porta série número 5 e são visualizados os seguintes ecrãs: do lado esquerdo (ecrã do autómato), do lado direito (ecrã do monitor);</td>
</tr>
<tr>
<td>Caso a operação mudança de programa se processar com sucesso a aplicação activa flag setup (=1) via porta série número 5 e são visualizados os seguintes ecrãs: do lado esquerdo (ecrã autómato), do lado direito (ecrã do monitor);</td>
</tr>
</table>

Figura 3.32 - Sequência de ecrãs mostrados durante a mudança de programa na célula.

Nota: Em modo automático é também possível toda a operação de mudança de programa seleccionando o botão "Camb. Prod" no ecrã mostrado na Figura 3.33:

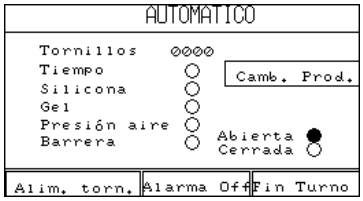


Figura 3.33 – Ecrã da consola do autómato no qual é possível visualizar o estado de processo de assemblagem em modo automático, permitindo também a mudança de programa de assemblagem da mesma.

Após a selecção do botão "Camb. Prod" a célula efectuará um fim de turno, ou seja, dá-se a realização do esvaziamento das peças existentes na sua área de trabalho. Após o mesmo é necessária a realização de todo o procedimento descrito na Figura 3.32.

3.5. Estrutura de Ficheiros

Usou-se um sistema de ficheiros para gestão, configuração e programação da célula de assemblagem. Assim sendo, nesta secção, vamos de forma detalhada explicar cada uma das estruturas dos ficheiros usados.

3.5.1. Estrutura do ficheiro txt de configuração inicial

Na estrutura deste ficheiro (Figura 3.34) é colocada toda a informação necessária para o funcionamento da célula. Esta informação é carregada no autómato na inicialização da célula de assemblagem e só após a validação de todo o processo de inicialização é possível a colocação da célula em funcionamento automático, activando para isso a *flag* "inicialização conluida" no autómato. É de realçar que este ficheiro se encontra localizado no computador em uma directoria específica.

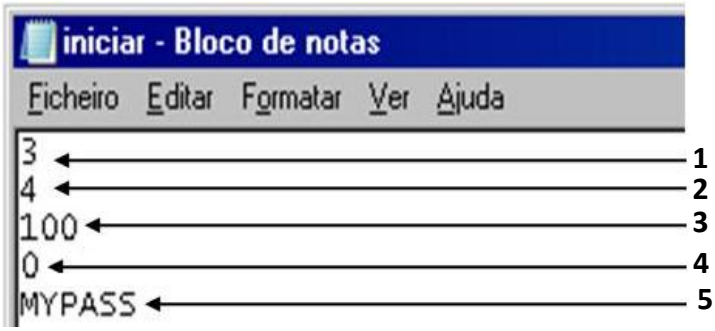


Figura 3.34 – Formato do ficheiro txt de configuração inicial da célula.

Observando a Figura 3.34 processa-se de seguida à identificação e descrição de cada um dos seus respectivos itens:

1- Número do programa a ser utilizado pela célula (descrito no ficheiro do programa secção 3.5.2). É através deste campo que é seleccionado o ficheiro de programa a ser carregado e executado pela célula;

2- Limite mínimo de parafusos é o número mínimo de parafusos a partir do qual o operador terá obrigatoriamente de colocar os parafusos no respectivo depósito.

3- Limite de parafusos para acender a luz amarela neste item é colocado o número de parafusos para o qual a luz de alarme de cor amarela, é activada;

4- Modo de arranque *Start* ou *Negara switch* neste item escolhe-se o modo de arranque da célula o qual pode ser realizado através do botão *Start* ou utilizando o interruptor *Negara Switch*, quando a '0' é seleccionado o modo *Start* quando a '1' é seleccionado o modo *Negara Switch*;

5- *Password* neste item define-se a *password* de acesso á janela de gestão de programas acedida na consola táctil do autómato;

3.5.2. Estrutura do Ficheiro txt do Programa

Este ficheiro contém toda a informação relativa à configuração da célula como um todo, ou seja contém todos os dados a carregar na célula para configurar a mesma para a assemblagem de uma nova gama de centralinas. É de realçar que este ficheiro se encontra localizado no computador em uma directoria específica.

O programa a carregar e executar na célula está dependente do item colocado na linha número 1 do ficheiro de configuração inicial descrito na secção anterior.

Como exemplo vamos apresentar a estrutura do ficheiro do programa número 1 o qual se encontra ilustrado na Figura 3.35.

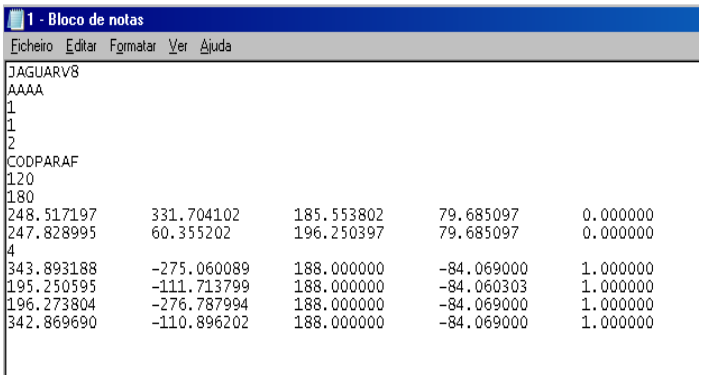


Figura 3.35 – Ficheiro txt do programa número 1.

Observando a Figura 3.35 procede-se de seguida á análise de cada um dos seus respectivos itens:

- **Linha 1** → Código do tabuleiro, Tag;
- **Linha 2** → Identificação dos quatro dígitos de controlo do ID (identificador da centralina);
- **Linha 3** → Número do programa a carregar no controlador de análise das regiões de Gel;
- **Linha 4** → Número do programa a carregar na aparafusadora;
- **Linha 5** → Número do depósito de parafusos a seleccionar;
- **Linha 6** → Referência dos parafusos a inserir no depósito de parafusos;
- **Linha 7** → Tempo crítico de cordão de silicone, ou seja, tempo para o qual se procede à activação da luz de aviso de cor amarela;
- **Linha 8** → Tempo de cordão de silicone para o qual surge a activação da luz de aviso de cor vermelha, ou seja, o tempo esgotou-se;
- **Linha 9 e 10** → Coordenadas dos pontos do robô de *pick and place*, em que a primeira coordenada realiza a sucção da tampa e a segunda efectua a colocação desta sobre a base da centralina;
- **Linha 11** → Identificação do número de pontos de aparafusamento a realizar pelo robô 2;
- **Linha 12, 13, 14, 15** → Coordenadas de aparafusamento de cada parafuso, correspondendo cada linha a um parafuso.

3.5.3. Estrutura do ficheiro da centralina

Ficheiro gerado na conclusão da centralina o qual é responsável pelo registo de toda a informação relativa ao processo de assemblagem. Este ficheiro fica alocado no computador e é direccionado para uma directoria específica. O formato do nome do ficheiro criado após a conclusão da centralina é ilustrado na Figura 3.36.

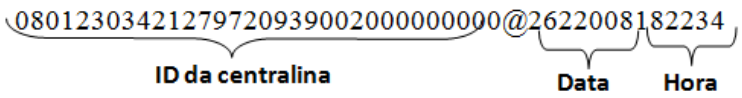


Figura 3.36 – Formato do nome do ficheiro gerado após a conclusão da centralina.

Na Figura 3.37 apresenta-se um exemplo de um ficheiro gerado pela aplicação na conclusão de uma centralina.

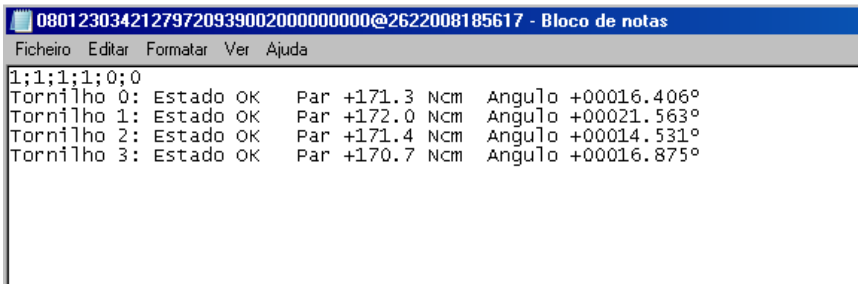


Figura 3.37 – Ficheiro gerado na conclusão de uma centralina.

Observando a identificação do ficheiro da Figura 3.37 verifica-se que a centralina assemblada tem como ID o "080123034212797209390020000000000" e a conclusão do seu processo de assemblagem deu-se no dia 26/02/2008 (formato dia/mês/ano) às 18:56:17.

Observando a informação de assemblagem retida no ficheiro pode-se verificar que a primeira linha é formada por um conjunto de 0 e 1 que representam respectivamente "OK" e "NG". Seguidamente apresenta-se a descrição de cada um desses dígitos representados:

- **Primeiro dígito** representação do estado do tempo de cordão de silicone ("OK" ou "NG");
- **Segundo dígito** representação do estado das regiões de gel ("OK" ou "NG");
- **Terceiro dígito** representação do estado do cordão de silicone ("OK" ou "NG");
- **Quarta dígito** representação se a peça foi ou não forçada no decorrer do processo ("OK" ou "NG");
- **Quinto dígito** representação do resultado da operação efectuada pelo robô de *pick and place* ("OK" ou "NG");

- **Sexto dígito** representação do resultado da operação efectuada pelo robô de aparafusamento (“OK” ou “NG”);

Nas linhas seguintes, ou seja, linha 2 à linha 5 apresenta-se toda a informação resultante do aparafusamento. Na qual é possível observar o número do parafuso, o seu torque e ângulo e se o aparafusamento foi ou não bem sucedido.

CAPÍTULO 4

Actualização da Célula

4.1 Novas funcionalidades

Com a finalidade de melhorar o desempenho global da célula de assemblagem foram propostos os seguintes melhoramentos:

- Desenvolvimento de uma base de dados que permita a substituição do sistema em ficheiros;
- Construção de uma página *Web* para visualização em tempo real de todos os dados de assemblagem da célula;

Todas estas novas funcionalidades têm como vantagens em relação ao sistema actualmente usado (ficheiros):

- Configuração da célula em qualquer ponto da fábrica bastando para isso um computador ligado em rede;
- Consulta restritiva da informação acerca das centralinas por datas de assemblagem e impressão de relatório das mesmas;
- Possibilidade de consulta e actualização dos programas a executar na célula em modo online;
- Consulta em tempo real de todo o histórico de centralinas assembladas na célula;
- Impressão de relatório anual acerca do trabalho executado pela célula;

4.2. Base de dados

4.2.1. *Software* usado

Para a implementação da base de dados foi usada uma ferramenta de nome "xampp", que contém um conjunto de aplicações necessárias para a construção e

simulação de páginas *Web* interligadas com base de dados. Entre as quais o *phpmyadmin* (Figura 4.1).

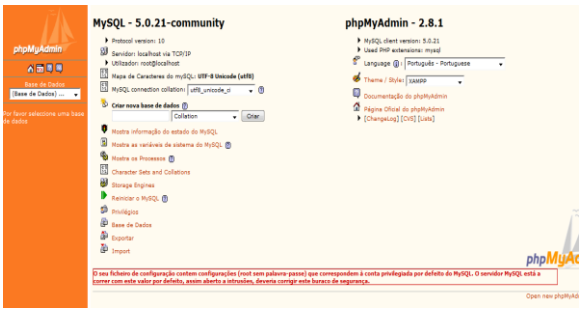


Figura 4.1- Ambiente gráfico do *software* usado na construção da base de dados (*phpmyadmin).*

4.2.2. Estrutura da base de dados desenvolvida

Depois de estudada toda a informação a armazenar na base de dados e a informação a ser visualizada na página *Web* chegou-se á conclusão que a arquitectura da base de dados que melhor respondia às nossas necessidades é a ilustrada na Figura 4.2.

Como se pode observar pela Figura 4.2 a base de dados é formada pelas tabelas "Centralinas", "Programas" e "Utilizadores".

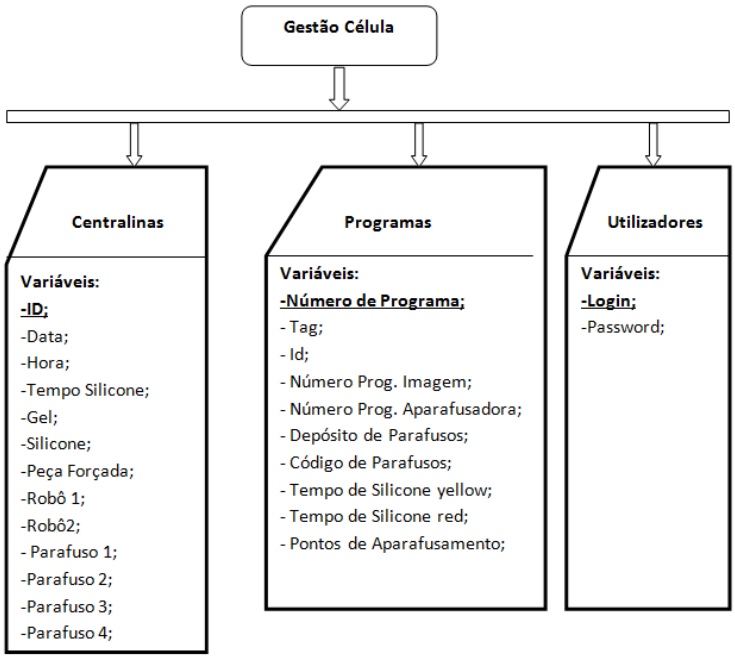


Figura 4.2 – Estrutura da base de dados implementada.

A implementação da base dados "Gestão Célula" encontra-se ilustrada na Figura 4.3.

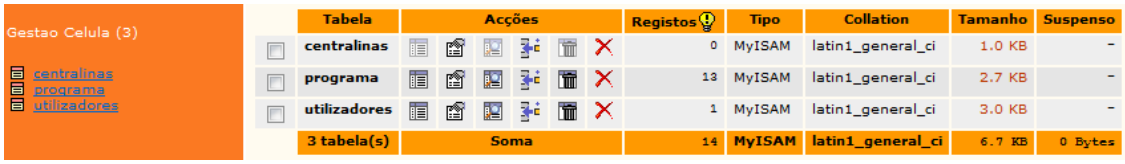


Figura 4.3 – Implementação da base de dados "Gestão Célula".

4.2.3 Descrição das tabelas usadas na base de dados

Tabela "Centralinas" responsável pelo armazenamento de todos os dados de cada centralina assemblada na célula.

Nesta tabela são armazenados todos os dados relativos ao processo completo de assemblagem da centralina. Estes dados são enviados pela aplicação quando se concluí o processo de assemblagem.

A estrutura e implementação desta tabela encontra-se ilustrada na Figura 4.4.

Figura 4.4 – Implementação e estrutura da tabela "Centralinas".

Observando a Figura 4.4 procede-se de seguida à identificação e descrição de cada uma das entidades apresentadas na tabela:

- **Id** → Identificação do número de série da centralina que completa a sua assemblagem, este item tem a chave primária ou identificadora da tabela "Centralinas", ou seja, não pode ocorrer o registo nesta tabela de duas centralinas com o mesmo Id;

- **Data** → Data de conclusão da assemblagem da centralina;
- **Hora** → Hora de conclusão da assemblagem da centralina;
- **Tempo de silicone**

- **"OK"** → A centralina foi finalizada cumprindo os requisitos exigidos para o tempo de cordão silicone;
- **"NG"** - O tempo limite de cordão de silicone foi ultrapassado no decorrer do processo de assemblagem;

- **Gel**
  - **"OK"** → Regiões de gel cumprem as condições exigidas;
  - "**NG"** → Regiões de gel não cumprem as condições exigidas;

- **Silicone**
  - **"OK"** → Cordão de silicone cumpre as condições exigidas;
  - **"NG"** → Cordão de silicone não cumpre as condições exigidas;

- **Peça forçada**
  - **"OK"** → Apesar das peças constituintes da centralina não cumprirem os requisitos exigidos na posição 1 foram forçadas pelo operador para avançar no processo;
  - **"NG"** → As peças constituintes da centralina não foram forçadas a avançar no processo o que pode ocorrer por duas razões ou avançaram de forma automática por ordem do autómato, ou então a centralina foi considerada como estragada;
- **Robôi**
  - "OK" → Robôi trabalhou de acordo com o pretendido;
  - "NG" → Robôi não trabalhou correctamente;

- **Aparafusamento** → Representação de toda a informação relativa ao processo de aparafusamento de cada um dos parafusos realizado pelo robô 2. A informação visualizada neste item refere-se ao número do parafuso, torque, ângulo e se o aparafusamento do parafuso foi ou não bem sucedido. É de notar que o número de parafusos a aparafusar pelo robô 2 varia com o modelo de centralina.

Tabela "Programas", é responsável pelo armazenamento de todos os programas actualmente existentes e prontos a carregar na célula. Cada programa destina-se a trabalhar uma gama específica de centralinas.

A estrutura e implementação desta tabela encontra-se ilustrada na Figura 4.5.

	Campo	Tipo	Collation	Atributos	Nulo	Defeito	Extra	Acções
☐	NumPrograma	int(200)			Não			
☐	Tag	varchar(256)	latin1_general_ci		Não			
☐	Id	varchar(256)	latin1_general_ci		Não			
☐	NumProgImag	int(50)			Não			
☐	NumProgApara	int(50)			Não			
☐	Panela	int(50)			Não			
☐	CodParafusos	varchar(256)	latin1_general_ci		Não			
☐	TempoYelow	int(50)			Não			
☐	TempoRed	int(50)			Não			

Figura 4.5 – Implementação da tabela "Programas".

Observando a Figura 4.5 procede-se de seguida à identificação e descrição de cada uma das entidades apresentadas na tabela:

- **Número do Programa** → Identificação do número do programa a carregar na célula, a selecção do número de programa a ser carregado na célula é realizada no ficheiro de configuração inicial. Este item encontra-se registado com a chave primária ou identificadora da tabela, ou seja, não poderá existir o registo de dois programas com o mesmo número;

- **Tag** → Identificação do código do tabuleiro;

- **Id** → Identificação dos quatro primeiros digitos do número de série da centralina, estes digitos são iguais no mesmo modelo, e servem para controlar se as centralinas que entram na área de trabalho da célula são ou não compatíveis com o tabuleiro e programa que está em actual execução;

- **Número Programa de Imagem** → Identificação do programa a seleccionar e carregar no controlodador de processamento de imagem das regiões de gel, pois estas zonas podem variar com modelo de centralina;

- **Número Programa de Aparafusadora** → Identificação do programa a seleccionar e carregar no controlodador da aparafusadora;

- **Depósito de parafusos** → Identificação do número do depósito de parafusos (1 ou 2) a seleccionar na célula, será a este depósito que o robô 2 efectuará a sucção dos parafusos;

- **Código parafusos** → Identificação do código de parafusos a inserir, em caso de necessidade, no depósito de parafusos activo;

- **Tempo *yellow*** → Identificação do tempo de cordão de silicone para o qual deve surgir um aviso ao operador, indicador que o tempo do cordão de silicone se encontra a terminar;

- **Tempo *red*** → Identificação do tempo de cordão de silicone para o qual o silicone deixa de cumprir os requisitos exigidos, e a peça deve ser retirada da célula.

- **Pontos de Aparafusamento** → Identificação do número de pontos de aparafusamento a realizar pelo robô 2.

Tabela "Utilizador", contém as informações dos utilizadores que podem aceder à área restrita da página *Web*. Esta área permite a criação, eliminação e modificação de programas.

A estrutura e implementação desta tabela é ilustrada na Figura 4.6.

	Campo	Tipo	Collation	Atributos	Nulo	Defeito	Extra	Acções
☐	login	varchar(256)	latin1_general_ci		Não			
☐	password	varchar(256)	latin1_general_ci		Não			

Figura 4.6 – Implementação da tabela "Utilizador".

Observando a Figura 4.6 procede-se de seguida à identificação e descrição de cada uma das entidades da tabela apresentada:

- ***Login*** → Identificação do *Login* do utilizador, este item tem a chave primária ou identificadora da tabela;

- ***Password*** → Identificação da *Password* de acesso do utilizador a toda a área restrita da página *Web;*

4.3 Página *Web*

O desenvolvimento da página *Web* realiza-se tendo como linguagem de programação predominante o php e usa-se o mysql como comunicador com a base de dados. Ou seja, não existe uma comunicação directa com a aplicação desenvolvida, a mesma apenas se limita a consultar os dados armazenados na base de dados, como se pode comprovar pela Figura 4.7. Pode-se verificar que a base de dados serve de intercomunicador entre a página *Web* e a aplicação desenvolvida e vice - versa.

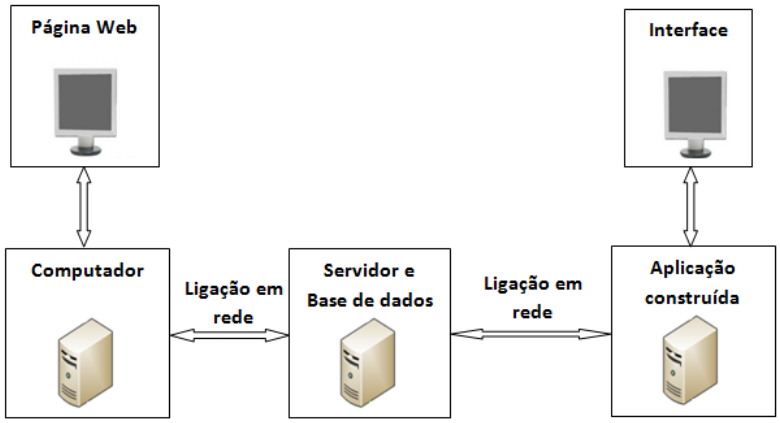


Figura 4.7 – Visualização da comunicação entre a aplicação, base de dados e página *Web*.

4.3.1 *Software* usado

PHP Designer 2005 foi o *Software* usado para o desenvolvimento da página *Web* utilizando como linguagens de programação o php e o *HTML*. Pode-se visualizar na Figura 4.8 o ambiente gráfico do *software*.

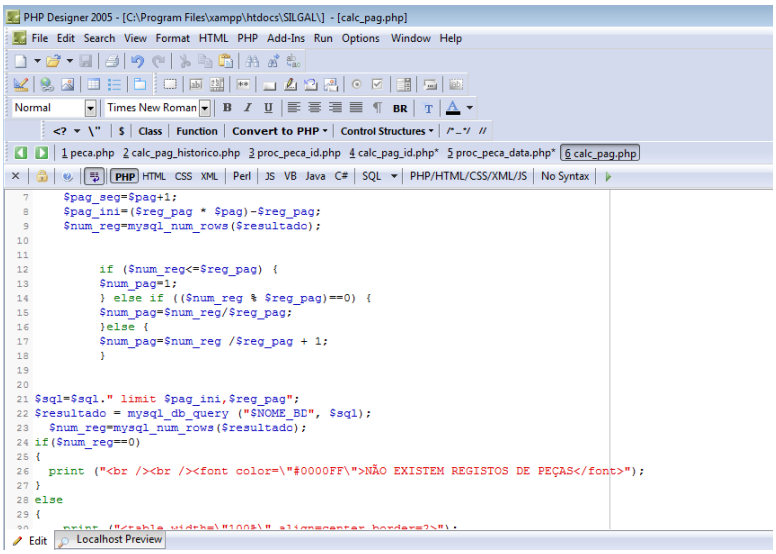


Figura 4.8 – *Software* usado no desenvolvimento da página *Web* (php *Designer* 2005).

4.3.2. Linguagens de programação usadas

PHP (*Hypertext Preprocessor)* é uma linguagem de programação muito utilizada para criar conteúdo dinâmico na *World Wide Web*.

Como características principais [5] [6] deste tipo de linguagem de programação são apresentadas as seguintes:

- Velocidade e robustez;
- Estruturada e orientada a objectos;
- Portabilidade - independência da plataforma – escreve-se uma vez, e pode ser executada em diferentes plataformas;
- Sintaxe similar à Linguagem C/C++ e o PERL;

MySQL é um sistema usado na criação de bases de dados, que utiliza a linguagem SQL (*Structured Query Language* - Linguagem de Consulta Estruturada) como interface. É actualmente uma das bases de dados mais populares, com mais de 10 milhões de instalações pelo mundo. Entre os utilizadores de bases de dados em MySQL pode-se destacar a: NASA, HP, Nokia, Sony, Alcatel, Cisco Systems entre outros.

Como características principais [7] [9] [11] que distingue este tipo de linguagem de programação temos:

- Portabilidade (suporta praticamente qualquer plataforma actual);

- Compatibilidade (existem drivers ODBC, JDBC e .NET e módulos de interface para diversas linguagens de programação, como Delphi, Java, C/C++, Python, Perl, PHP. ASP e Ruby)
- Excelente desempenho e estabilidade;
- Pouco exigente quanto a recursos de *hardware*;
- Facilidade de uso;
- É um *Software* Livre;
- Suporte a vários tipos de tabelas (como MyISAM, InnoDB e Maria), cada um deles com um fim específico;
- Aceita controlo transaccional;
- Aceita *Triggers*;
- Aceita *Stored Procedures* e *Functions*;
- Replicação facilmente configurável;

4.3.3. Estruturação da página *Web*

Definidas todas as funcionalidades a que a página *Web* deve responder, implementou-se o algoritmo ilustrado na Figura 4.9. É possível observar que a página *Web* é formada por uma página principal *(Home)* a qual entra em execução na primeira visita. Dentro da página *Home* temos um conjunto de *links* os quais permitem a visualização de todo o conteúdo requisitado.

Os *links* apresentados na página principal são os seguintes:

- Historial de centralinas fabricadas;
- Procurar centralina por número de série (ID);
- Procurar centralinas por data de fabrico;
- Imprimir relatório de célula;
- Gestão de programas;

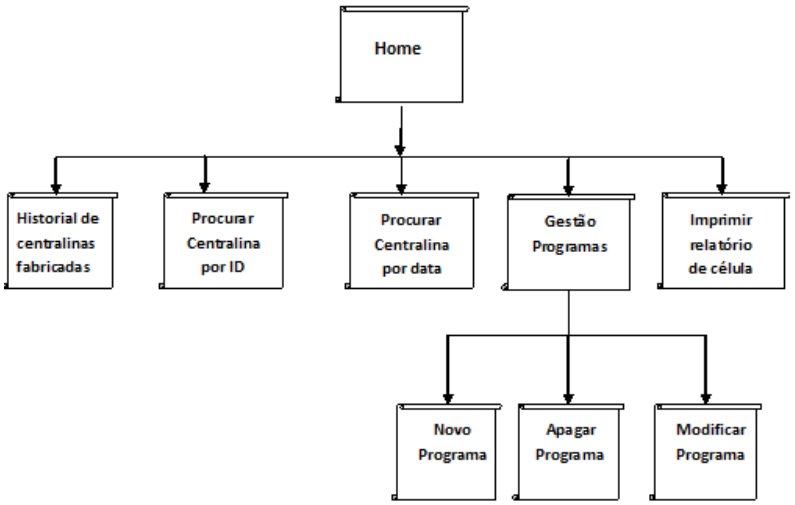


Figura 4.9 – Algoritmo usado na construção de toda a estrutura da página *Web*.

Procede-se de seguida a uma descrição de cada um dos itens apresentados na Figura 4.9.

Home, página apresentada no primeiro acesso a qual contém um pequeno manual de instruções referente ao modo como se pode navegar na página o que facilita a navegação para novos operadores que possam ingressar na empresa. O interface desta página pode ser visualizado na Figura 4.10.

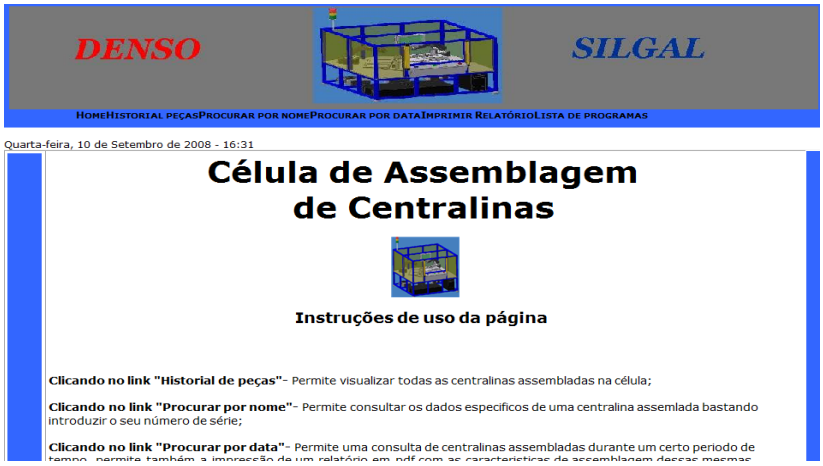


Figura 4.10 – Interface da página "*Home*".

Historial de centralinas fabricadas, nesta página em particular é possível a visualização de todas as centralinas assembladas e seus respectivos dados de fabrico, ou seja, nesta página é visualizado todo o historial de assemblagem da célula.

Na Figura 4.11 é ilustrado todo o conteúdo e ambiente gráfico desta página no qual é apresentado todo o conjunto de características de assemblagem da centralina entre as quais temos:

- Número de série;
- Data de assemblagem;
- Hora na qual terminou o processo de assemblagem;
- Conjunto de erros ocorridos durante a assemblagem entre os quais:
  - Tempo de cordão de silicone ("OK" ou "NG");
  - Regiões de Gel ("OK" ou "NG");
  - Cordão de Silicone ("OK" ou "NG");
  - Peça Forçada ("OK" ou "NG");
  - Robô 1 ("OK" ou "NG");
  - Robô 2 ("OK" ou "NG");
- Registo de todos os dados de aparafusamento.

Todos estes dados são obtidos através de comandos *sql* enviados desde a página desenvolvida até á base de dados. A tabela da base de dados consultada é a tabela "Centralinas".

Figura 4.11 – Página que mostra todo o historial de centralinas e suas características de assemblagem.

Procurar centralina por número de série, esta página, ilustrada na Figura 4.12, possibilita a visualização das centralinas assembladas tendo em conta o seu número de série. Pode-se observar pela Figura 4.12 que inicialmente é solicitado, ao operador, o "ID" da centralina a consultar, o qual pode ser directamente introduzido através da janela apresentada mais á direita a qual apresenta todos os números de série existentes na base de dados.

Após a introdução do número de série é realizada uma consulta á base de dados mais propriamente à tabela "Centralinas" da qual são retirados e mostrados todos os dados respeitantes á centralina seleccionada como se pode observar na Figura 4.13.

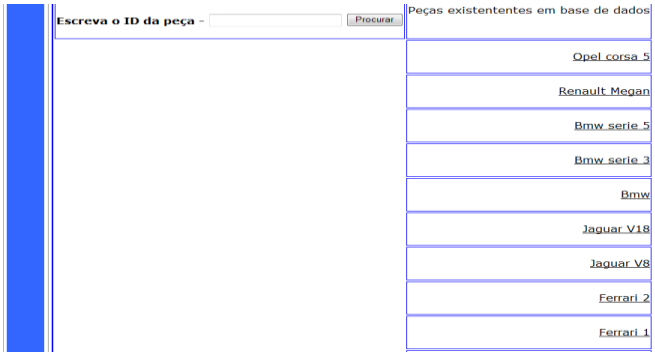
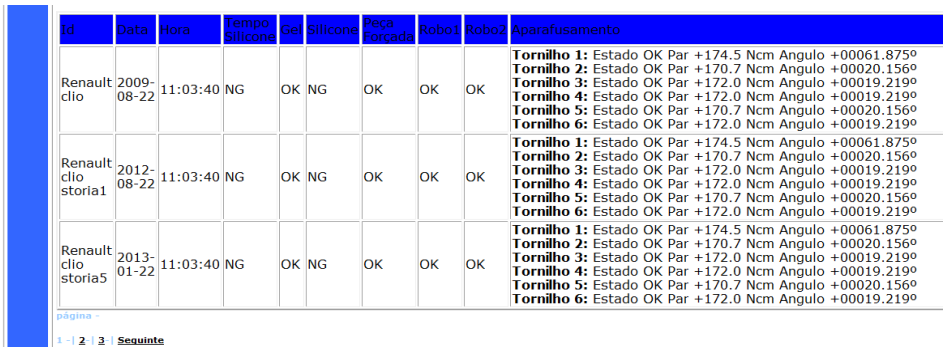


Figura 4.12 – Página de procura de centralina por número de série.



Id	Data	Hora	Tempo Silicone	Gel	Silicone	Peça Forçada	Robo1	Robo2	Aparafusamento
Renault clio	2009-08-22	11:03:40	NG	OK	NG	OK	OK	OK	**Tornilho 1:** Estado OK Par +174.5 Ncm Angulo +00061.875º **Tornilho 2:** Estado OK Par +170.7 Ncm Angulo +00020.156º **Tornilho 3:** Estado OK Par +172.0 Ncm Angulo +00019.219º **Tornilho 4:** Estado OK Par +172.0 Ncm Angulo +00019.219º **Tornilho 5:** Estado OK Par +170.7 Ncm Angulo +00020.156º **Tornilho 6:** Estado OK Par +172.0 Ncm Angulo +00019.219º
Renault clio storia1	2012-08-22	11:03:40	NG	OK	NG	OK	OK	OK	**Tornilho 1:** Estado OK Par +174.5 Ncm Angulo +00061.875º **Tornilho 2:** Estado OK Par +170.7 Ncm Angulo +00020.156º **Tornilho 3:** Estado OK Par +172.0 Ncm Angulo +00019.219º **Tornilho 4:** Estado OK Par +172.0 Ncm Angulo +00019.219º **Tornilho 5:** Estado OK Par +170.7 Ncm Angulo +00020.156º **Tornilho 6:** Estado OK Par +172.0 Ncm Angulo +00019.219º
Renault clio storia5	2013-01-22	11:03:40	NG	OK	NG	OK	OK	OK	**Tornilho 1:** Estado OK Par +174.5 Ncm Angulo +00061.875º **Tornilho 2:** Estado OK Par +170.7 Ncm Angulo +00020.156º **Tornilho 3:** Estado OK Par +172.0 Ncm Angulo +00019.219º **Tornilho 4:** Estado OK Par +172.0 Ncm Angulo +00019.219º **Tornilho 5:** Estado OK Par +170.7 Ncm Angulo +00020.156º **Tornilho 6:** Estado OK Par +172.0 Ncm Angulo +00019.219º

Figura 4.13 – Após a selecção do ID serão mostradas todas as centralinas com o ID pretendido ou com as mesmas iniciais, neste exemplo o ID seleccionado na Figura 4.12 foi "Renault".

Procurar centralina por data, esta página possibilita ao utilizador a procura de um conjunto de centralinas assembladas num determinado período de tempo. O interface visualizado após a selecção do *link* "Procurar por Data" encontra-se ilustrado na Figura 4.14.

Após a selecção do intervalo de datas procede-se à consulta da tabela "Centralinas" da base de dados a qual por sua vez responde enviando todos os dados que cumprem os critérios seleccionados. Posteriormente é mostrada a página com os respectivos dados os quais se encontram ilustrados na Figura 4.15.

Nesta página temos ainda a possibilidade de imprimir todos os dados, resultantes da procura restritiva, para uma folha em pdf a qual pode ser observada na Figura 4.16.

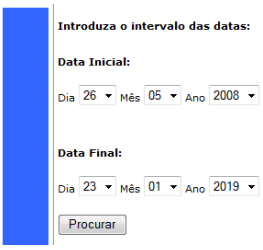


Figura 4.14 – Página apresentada ao utilizador após a selecção da procura de centralinas por intervalo de datas.

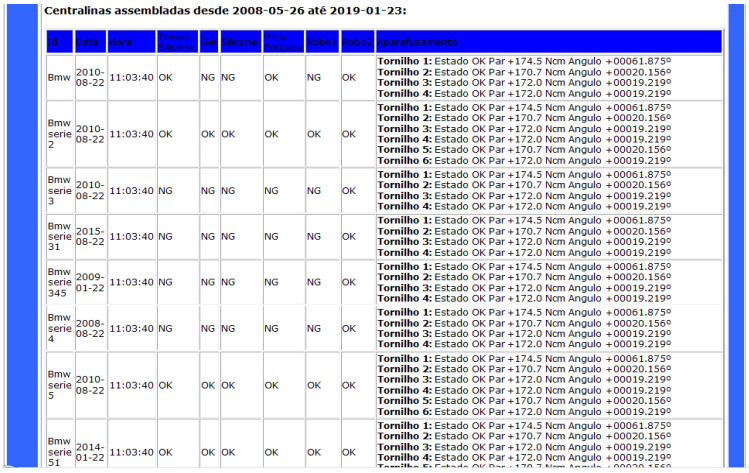
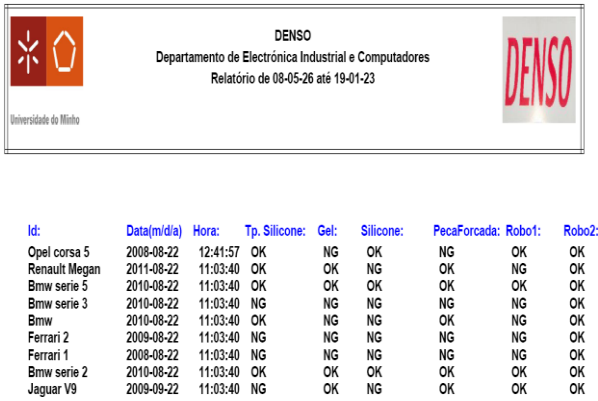


Figura 4.15 – Página que mostra as centralinas assembladas durante o intervalo de datas seleccionado.



Id:	Data(m/d/a)	Hora:	Tp. Silicone:	Gel:	Silicone:	PecaForcada:	Robo1:	Robo2:
Opel corsa 5	2008-08-22	12:41:57	OK	NG	OK	NG	OK	OK
Renault Megan	2011-08-22	11:03:40	OK	OK	NG	OK	NG	OK
Bmw serie 5	2010-08-22	11:03:40	OK	OK	OK	OK	OK	OK
Bmw serie 3	2010-08-22	11:03:40	NG	NG	NG	NG	NG	OK
Bmw	2010-08-22	11:03:40	OK	NG	NG	OK	NG	OK
Ferrari 2	2009-08-22	11:03:40	NG	NG	NG	NG	NG	OK
Ferrari 1	2008-08-22	11:03:40	NG	NG	NG	NG	NG	OK
Bmw serie 2	2010-08-22	11:03:40	OK	OK	OK	OK	OK	OK
Jaguar V9	2009-09-22	11:03:40	NG	OK	NG	OK	OK	OK

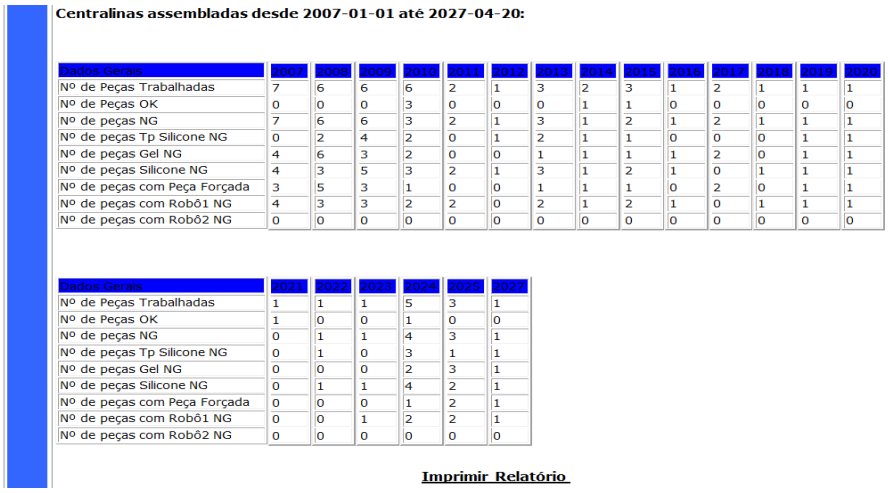
Figura 4.16 – Relatório em pdf em que pode ser consultada toda a informação relativa às centralinas assembladas durante um período de tempo inserido pelo utilizador.

Imprimir relatório de célula, após a selecção desta página é visualizado o interface da Figura 4.17 no qual se pode consultar todo o historial de assemblagem de centralinas relativamente a cada ano de trabalho.

O relatório é anual e disponibiliza a seguinte informação:

- Número de centralinas que deram entrada na célula;
- Número de centralinas assembladas sem qualquer erro;
- Número de centralinas assembladas com erro;
- Identificação dos erros ocorridos:
    - Número de erros de tempo de cordão silicone ultrapassado;
    - Número de testes para os quais as regiões de gel não cumpriram com os requisitos mínimos exigidos;
    - Número de vezes em que o tempo do cordão de Silicone não cumpriu com os requisitos mínimos exigidos;
    - Número de vezes em que a peça foi forçada pelo operador a avançar no processo;
    - Número de vezes em que o robô 1 não concluiu trabalho com sucesso;
    - Número de vezes em que o robô 2 não concluiu trabalho com sucesso;

Caso o utilizador assim o pretenda pode ainda ser realizada a impressão de toda a informação em uma página em pdf ficando a mesma com o formato ilustrado na Figura 4.18.



Centralinas assembladas desde 2007-01-01 até 2027-04-20:

Dados Gerais	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
Nº de Peças Trabalhadas	7	6	6	6	2	1	3	2	3	1	2	1	1	1
Nº de Peças OK	0	0	0	3	0	0	0	1	1	0	0	0	0	0
Nº de peças NG	7	6	6	3	2	1	3	1	2	1	2	1	1	1
Nº de peças Tp Silicone NG	0	2	4	2	0	1	2	1	1	0	0	0	1	1
Nº de peças Gel NG	4	6	3	2	0	0	1	1	1	1	2	0	1	1
Nº de peças Silicone NG	4	3	5	3	2	1	3	1	2	1	0	1	1	1
Nº de peças com Peça Forçada	3	5	3	1	0	0	1	1	1	0	2	0	1	1
Nº de peças com Robô1 NG	4	3	3	2	2	0	2	1	2	1	0	1	1	1
Nº de peças com Robô2 NG	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Dados Gerais	2021	2022	2023	2024	2025	2027
Nº de Peças Trabalhadas	1	1	1	5	3	1
Nº de Peças OK	1	0	0	1	0	0
Nº de peças NG	0	1	1	4	3	1
Nº de peças Tp Silicone NG	0	1	0	3	1	1
Nº de peças Gel NG	0	0	0	2	3	1
Nº de peças Silicone NG	0	1	1	4	2	1
Nº de peças com Peça Forçada	0	0	0	1	2	1
Nº de peças com Robô1 NG	0	0	1	2	2	1
Nº de peças com Robô2 NG	0	0	0	0	0	0

Imprimir Relatório

Figura 4.17 – Página em que é possível visualizar todo o relatório de cada ano de assemblagem da célula, para o período de tempo seleccionado.

DENSO
Departamento de Electrónica Industrial e Computadores
Relatório de 01-01-2007 até 20-04-2027

Universidade do Minho

RELATÓRIO DE MÁQUINA

Ano	2007
Numero Peças Trabalhadas	7
Peças OK	0
Peças NG:	7
Numero de Erros de Tempo Silicone	0
Numero de erros de Gel:	4
Numero de erros de Silicone:	4
Numero de Peças Forçadas	3
Numero de Erros Robo 1	4
Numero de Erros Robo 2	0
Ano	2008
Numero Peças Trabalhadas	9
Peças OK	0

Figura 4.18 – Relatório anual em pdf de centralinas assembladas durante um período de tempo inserido pelo utilizador.

Gestão Programas, nesta área, mostrada na Figura 4.19, é possível a consulta de todos os programas actualmente existentes na célula de assemblagem. Cada um destes programas pode ser carregado e executado na célula sendo para isso necessário a

introdução do seu número no ficheiro de configuração inicial mais propriamente na linha número 1 como foi descrito anteriormente.

Caso o utilizador pretenda a alteração, eliminação de algum programa existente ou então a criação de um novo programa deverá seleccionar no submenu existente na parte inferior da página a operação desejada. Após a selecção é apresentada uma zona reservada na qual será solicitado o *login* (Figura 4.20).

Caso o utilizador se encontre registado tem acesso aos interfaces mostrados nas Figuras 4.21, 4.22, 4.23 podendo respectivamente proceder a toda a gestão de programas.

Figura 4.19 – Página em que se pode consultar toda a lista de programas existentes para programação da célula.

Figura 4.20 – Página apresentada quando o utilizador solicita a criação, eliminação ou modificação de algum programa.

Figura 4.21 – Página que possibilita ao utilizador registado a edição de um programa existente.

Figura 4.22 – Página apresentada quando o utilizador está registado e pretende a eliminação de um programa existente.

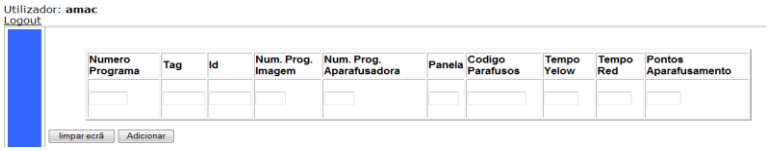


Figura 4.23 – Página apresentada quando o utilizador está registado e pretende a criação de um novo programa de assemblagem.

4.4. Rotinas de interacção entre a aplicação desenvolvida e a base de dados

Nesta secção é apresentado todo o conjunto de rotinas responsáveis pela comunicação e interacção entre a aplicação e a base de dados.

4.4.1 Rotina que estabelece a comunicação entre a aplicação e a base de dados

Esta função, ilustrada na Figura 4.24, é responsável pela realização da configuração de toda a comunicação entre a aplicação e a base de dados.

```
1.  bool CDisplayApp::OpenConnection()
2.  {
3.      CString sHost= _T("localhost");
4.      CString sUser= _T("root");
5.      CString sPassword= _T("");
6.      CString sDatabase= _T("Gestao Celula");

7.    if(!mysql_real_connect(m_pLinkDb,sHost,sUser,sPassword,
      sDatabaseChar, 3306, NULL,0))
8.     {
9.           CString strText;
10.          strText.Format(_T("Error: '%s'"),mysql_error(m_pLinkDb));
11.          AfxMessageBox(strText);
12.          return false;
13.    }

14.    return true;
15. }
```

Figura 4.24 – Função responsável por estabelecer a comunicação entre a aplicação e a base de dados.

Procede-se de seguida à análise da Figura 4.24:

- **Linha de código número 1** → Declaração da função;
- **Linha de código número 3** → Seleccção do tipo de servidor;
- **Linha de código número 4 e 5** → Seleccão do utilizador e da *password* de acesso á base de dados;
- **Linha de código número 6** → Nome da base de dados a consultar;
- **Linha de código número 7 á linha 15** → É testado se a comunicação com a base de dados foi concluida com sucesso, caso contrário é mostrada a mensagem no monitor com o possivél erro de comunicação ocorrido.

4.4.2 Rotina usada no envio de dados para a base de dados

Esta rotina, ilustrada na Figura 4.25, é responsável pelo envio dos dados a armazenar em uma tabela especifica da base de dados. No nosso caso esta função é usada para o armazenamento de todas as centralinas assembladas e suas respectivas caracteristicas de assemblagem, bem como no envio de um novo programa de assemblagem para a base de dados.

```
sql.Format(_T("INSERT INTO "Tabela_De_Inserção" \ VALUES
('%s')"),nome_das_variavéis );
```

Figura 4.25 – Função usada pela aplicação na inserção da informação em uma tabela específica da base de dados.

Na Figura 4.25 mostra-se o formato da instrução responsável pela inserção de dados numa determinada tabela da base de dados.

4.4.3. Rotina usada no carregamento de dados armazenados na base de dados

Esta função, ilustrada na Figura 4.26, responsabiliza-se pelo carregamento dos dados em uma tabela da base de dados, os dados podem ser carregados na sua totalidade (não usando a condição "where") ou pode-se simplesmente restringir o seu carregamento definindo a condição "where". No nosso caso esta função é usada na mudança e edição de um novo programa.

```
1.   sql.Format("SELECT * FROM "tabela" WHERE "condição");

2.   if(mysql_real_query(theApp.GetConnection(),sql,sql.GetLength()) == 0 )
3.   {
4.      MYSQL_RES* res;

5.      if ((res = mysql_store_result(theApp.GetConnection())))
6.       {
7.          MYSQL_ROW Row;

8.           if((Row = mysql_fetch_row(res)))
9.           {
10.               Var_Entidade= Row[i];//valor da coluna prendendida
11.               UpdateData(FALSE);
12.           }

13.           mysql_free_result(res);
14.      }
15.  }

16.  else
17.  {
18.         CString strText;
19.         strText.Format("Error: '%s'",mysql_error(theApp.GetConnection()));
20.         AfxMessageBox(strText);
21.  }
```

Figura 4.26 – Função usada pela aplicação na consulta dos dados na base de dados.

Procede-se de seguida à análise da Figura 4.26:

- **Linha de código número 1** → Instrução responsável pela consulta de uma tabela especifica alocada na base de dados;

- **Linha de código número 2** →Verifica-se se a comunicação entre a aplicação e a base de dados foi estabelecida com sucesso;
- **Linha de código número 3 á linha número 15** →Caso a comunicação entre a aplicação a base de dados se encontra estabelecida então realiza-se um varrimento por parte da aplicação á tabela da base de dados seleccionada e são retirados todos os dados tendo em conta a condição estabelecida.
- **Linha de código número 16 á linha número 21** → Caso a comunicação entre a base de dados e aplicação não se tenha sido estabelecido com sucesso, então será visualizado no monitor o possível erro de comunicação ocorrido.

4.4.4. Rotina usada na actualização dos dados armazenados na base de dados

Esta rotina, ilustrada na Figura 4.27, é responsável pela actualização dos dados alocados na base de dados. No nosso caso esta função é usada no decorrer da edição do programa no robô 1 e no robô 2. Ou seja, após a conclusão do envio por parte do autómato de todos os dados editados, a aplicação faz desencadear esta rotina a qual realiza a edição dos dados na tabela "Programas" da base de dados.

```
sql.Format("UPDATE "tabela_a_editar" SET "dados_a_editar" WHERE "condição" ");
```

Figura 4.27 – Função usada pela aplicação para a edição dos dados alocados na base de dados.

Na Figura 4.27 é descrito o formato da instrução a enviar á base de dados para realizar a actualização da informação contida numa tabela em específico.

CAPÍTULO 5

Conclusões e Sugestões para Trabalhos Futuros

5.1. Conclusões

Pode-se concluir deste trabalho que os objectivos foram atingidos com sucesso pois a aplicação projectada e implementada correspondeu aos requisitos exigidos pelo cliente.

Devido à grande complexidade de todo o projecto, o desenvolvimento da aplicação de controlo e monitorização do processo de assemblagem foi dividido em diferentes fases de modo a que solução final corresponde-se sem grandes desvios ao que era pretendido. Assim sendo numa primeira fase foi realizado todo o estudo dos sistemas a interligar ao computador. Posteriormente foram definidos todos os protocolos de comunicação entre o computador e cada um dos dispositivos criando-se um conjunto de tabelas de tramas a serem enviadas pela porta série ao computador, através dos vários sistemas.

Terminada a fase de estudo, passou-se ao desenvolvimento dos algoritmos e suas implementações. Para teste de algumas das implementações realizadas usaram-se softwares próprios de monitorização das portas série de modo a perceber se a trama enviada/recebida era ou não a esperada.

Como principais implementações desenvolvidas temos:

Implementação de um sistema responsável pela recepção e processamento das tramas enviadas pelos diferentes dispositivos ligados por porta série.

Implementação de um sistema de actualização do interface em tempo real com um conjunto de mensagens indicativas do estado de todo o processo de assemblagem realizado na célula.

Outra das grandes prioridades e grandes problemas da célula de trabalho foi a implementação de todo o controlo e gestão do tempo de cada peça finalizada pela máquina responsável pela colocação do silicone nas centralinas, pois estas não poderiam ser assembladas se o tempo de alarme do silicone fosse ultrapassado. Para resolver este

problema foi implementado um sistema complexo que tem como responsabilidade o de sincronizar os vários grafcets que controlam os tempos das peças no autómato e todo processo de controlo automático desenvolvido na aplicação. Assim sendo foi implementado um conjunto de três timers na aplicação os quais são activados através de um conjunto especifico de tramas conhecidas e enviadas pelo autómato. Cada timer é associado a uma centralina sendo o seu valor decisivo no decorrer do processo de assemblagem. Nesta fase deve-se realçar as constantes alterações à forma de controlo do timer exigidas pelo cliente e a realização de todos estes testes em ambiente de simulação uma vez que não nos foi possível a utilização da máquina auxiliar de colocação de silicone.

Outros dos processos de grande importância implementados nesta aplicação foi a activação/desactivação e escrita/leitura de tramas em posições específicas de memória do autómato, estes processos são responsáveis pela configuração inicial da célula entrando também em execução durante a gestão dos programas de assemblagem.

Outra das funcionalidades requisitadas pelo cliente foi o desenvolvimento de um sistema capaz de realizar todo o histórico de centralinas fabricadas o qual tem como finalidade o de fornecer todas as características de fabrico da centralina. Como resposta às necessidades do cliente desenvolveu-se uma estrutura de dados que é inicializada após entrada da centralina na célula. Esta estrutura realiza todo o acompanhamento da centralina ao longo de todo o processo registando todas as características e possíveis erros de assemblagem. No final do processo é gerado um ficheiro com o conjunto de informação armazenada na estrutura.

Outro dos requisitos da célula de trabalho seria criar uma forma de gerir os seus programas de forma a operar sobre vários modelos de centralinas, esta gestão é realizada usando a consola do autómato para toda a gestão de programas e o armazenamento dos mesmos é realizado no computador através de uma estrutura de ficheiros. Para este problema foi necessário criar um conjunto de rotinas capazes de realizar a gestão completa dos programas e criar um sincronismo com o autómato de forma a que os robôs possam operar com os programas correctos.

Por fim foram implementadas actualizações a instalar numa possível célula a requisitar pelo cliente de entre as actualizações temos a utilização de uma base de dados a qual tem como função realizar todo o histórico e armazenamento das características de assemblagem de cada centralina bem como todo o armazenamento do

programas de assemblagem a carregar na célula. Esta nova funcionalidade substitui o sistema de ficheiros utilizado permitindo assim uma melhor organização e consulta do histórico. Aliado ao desenvolvimento da base de dados foi desenvolvido uma página *Web* com a finalidade de visualização online de todo o histórico bem como impressão de relatórios de todo o processo de assemblagem desenvolvido pela célula. Outra das grandes potencialidades desta página é a possibilidade de criação de novos programas permitindo desta forma a programação de toda a célula online e em qualquer ponto da empresa.

Podemos concluir que toda a implementação do controlo da célula de trabalho foi extremamente complexa, necessitando de uma total concentração nas prioridades ao controlo da mesma. No final tornou-se extremamente gratificante pois todos os requisitos foram preenchidos e a operação da célula de trabalho deu-se sem falhas, tornando-se capaz de operar em ambiente industrial 24 horas por dia.

Todo este trabalho, foi de grande importância para conferir e aperfeiçoar alguns dos conteúdos aprendidos e aprender novos conceitos permitindo desta forma valorizar toda a nossa formação académica e adquirir um conhecimento mais profundo da realidade e necessidades do mundo empresarial.

5.2. Sugestões para Trabalhos Futuros

Como trabalho futuro requisitado pelo cliente para a construção de uma nova célula temos:

- Programação automática da célula com base na leitura do código do tabuleiro e da referência da centralina. Isto é, a nova aplicação que gere a célula de trabalho deverá ter a capacidade de carregar os novos programas, nos vários dispositivos do sistema para que os mesmos trabalhem a respectiva centralina. Com esta actualização podemos optimizar o trabalho da célula podendo a mesma assemblar três modelos diferentes de centralinas em simultâneo.

- Implementação de um sistema de *timers* que tenha em conta uma maior cadência de centralinas fornecidas pela máquina auxiliar de colocação de regiões de gel e cordão de silicone;

- Possibilidade da centralina ser assemblada de forma automática em ciclos diferentes de forma a obter a sua conclusão, ou seja, caso a centralina obtenha erros que possam ser resolvidos percorrendo um novo ciclo o *software* no fim do primeiro ciclo deve reagir de acordo com os erros obtidos durante o processo de assemblagem verificando também os tempos do cordão de silicone e só depois dar a ordem para a célula completar a centralina. Esta fase pode ocorrer caso por exemplo o robô 2 efectue um processo de aparafusamento incompleto.

REFERÊNCIAS

[1] Prof. Msc. Marcelo Eurípedes da Silva, "Curso Automação", EEP – Escola de Engenharia de Piracicaba, 05 de Setembro de 2007;

[2] IBM Global services Integrated Technology Newsletter, RFID: melhor, mais rápido, mais eficiente, IBM, 19ª edição – Junho 2005;

[3] Bjarne Stroustrup, "The C++ Programming Language", Addison-Wesley, 1994;

[4] Prof : André Bernardi, Curso de Extensão MFC – Básico, 1999;

[5] Fábio Santos /Alexandre Arroyo, Centro de Computação da Unicamp, PHP Avançado, 2006;

[6] C: Fernandes, J. Marques, Programação com PHP 4, 2ª edição, FCA- Editora de Informática, Lisboa, 2001.

[7] Luis Damas, SQL, 7ª edição, FCA-Editora de informática;

[8] C. J. Date, An Introduction to DATABASE SYSTEM. Addison Wesley, 6ª ed;

[9] Joe Celko, "INSTANT SQL Programming", WROX;

[10] Ana Mendes, Francisco Guimarães e Sérgio Oliveira, Programação de bases de dados com Visual Basic.Net, FCA, 2003:

[11] Paul Dubois, "MySQL", New Riders;

Referências WWW

http://industrial.omron.pt/

http://www.atlascopco.pt/ptus/

http://www.densorobotics.com/

http://www.exkola.com.br/scripts/detalhe.php?servico=carreira&id=224

http://www.php.net

http://www.eq.uc.pt/~jorge/aulas/internet/paginaweb

http://dev.mysql.com/doc/mysql/pt/index.html (manual do mysql em português)

Anexos

[Anexo A] Biblioteca da Moxa

Esta biblioteca dispõe de um conjunto de funções para trabalhar com as portas série disponibilizadas pela placa.

Funções de configuração da porta

Sio _open()

Função responsável pela abertura da porta série permitindo a transferência/recepção de dados.

Sintaxe:

sio_open(port);
port=número da porta COM;

Devolve:

SIO_OK – Porta disponível;
SIO_BADPORT - Número da porta inválida.
SIO_OPENFAIL - Porta não encontrada ou utilizada por outro programa.

Exemplo:

```
int port = 2;
ret = sio_open (port);
if ( ret != SIO_OK ) {/* Open fail  */}
```

Sio_close()

Desabilita a recepção/transmissão de dados pela porta série;

Sintaxe:

sio_close(port);

port=número da porta COM;

Devolve:

SIO_OK – Porta fechada com sucesso;

SIO_BADPORT - A porta série não foi aberta anteriormente.

sio_ioctl()

Função responsável pela configuração dos parâmetros da comunicação entre eles baud rate, *bits* de dados, paridade e *stop bit*;

Sintaxe

sio_ioctl (port, Bbauderate, P_Paridade| BIT_Bits_dados | STOP_Stop_Bits);

Exemplo:

Configuração da porta COM 2 com os seguintes parâmetros:

Baud rate = 38400, *Paridade* = *NONE*, *Bits* de dados = 8, *Stop Bit* = 1

```
ret = sio_ioctl ( 2, B38400, P_NONE | BIT_8 | STOP_1 );
if (ret != SIO_OK) {/* Setting fail  */}
```

Sio_flush ()

Elimina dados do buffer de saída e/ou entrada;

Sintaxe

sio_flush(int port, int func)

Valores de "func"

0 : Apaga os dados do buffer de entrada;

1 : Apaga os dados do buffer de saída;

2 : Apaga os dados do buffer de entrada e saída;

Exemplo:

Eliminar dados do buffer de entrada e de saída.

```
ret = sio_flush ( 2, 2 );
if (ret != SIO_OK) {/* Fail  */}
```

Função de leitura da porta série

Sio_read( )

Efectua a leitura dos dados que dão entrada na porta série.

Sintaxe:

sio_read(int port, char *buf, int len)

port = Número da porta COM a efectuar a leitura;

buf = variavél que armazena os dados recebidos;

len = número de caracteres a ler;

Devolve:

> 0 dados recebidos;

= 0 Dados não recebidos;

Exemplo:

```
int     len;
char    buf[10];
len = sio_read( 2, buf, 10);
if (ch < 0) {/* Fail  */}
```

Função de escrita na porta série

Sio write( )

Escreve um bloco de dados no buffer da porta série pretendida.

Sintaxe:

sio_write(int port, char *buf, int len)

port = Número da Porta COM

buf = Variavél que contém os dados a enviar;

len = número de caracteres da variavél buf a tranasmitir;

Devolve:

>= 0 dados transmitidos;

Exemplo:

```
int     len;
char    buf[10];
len = sio_write( 2, buf, 10);
if (len< 0) {/* Fail  */}
```

Rotinas de interrupção

Para definir uma recepção de dados por interrupção, é necessário definir um caractrer terminador da trama a receber. Após a recepção desse tipo de tramas a porta série desencadeia um rotina específica.

Sintaxe:

sio_term_irq(int port, VOID (CALLBACK *func)(int port), char code)

port = COM port number

func = nome da rotina que queremos que seja desencadeada

code = character terminador da trama

Exemplo:

```
//Declaração da rotina a ser desencadeada

VOID CALLBACK TermIrq(int port)

{//código a ser executado}

.
.
.
ret = sio_term_irq( 2, TermIrq, 'A');
if (ret != SIO_OK) {/* Fail  */}

}
```

[Anexo B] Porta RS232

Como vimos anteriormente a comunicação entre a aplicação e os vários dispositivos a ela ligados é realizada via comunicação porta RS-232 (Figura 7.1), o computador dispõe de 8 portas RS-232 disponibilizadas pela placa multiplicadora de portas séries (placa moxa). Para que a comunicação entre os vários dispositivos e a aplicação ocorra de forma correcta é necessário um protocolo comum entre ambos.

Figura 7.1 – Porta série.

O protocolo de comunicação série RS-232 requer que se especifique quatro parâmetros: a taxa de transmissão da trama (baud rate), o número de *bits* de dados a enviar na trama, a paridade da trama, e o número de *stop bits*.

Uma trama é formada por um *start bit* (primeiro *bit* da trama) seguido por *bits* de dados, depois a indicação da paridade é opcional, e por fim temos um *bit* que indica o fim da trama (*stop bit*). Como exemplo, é mostrado na Figura 7.2 a constituição da trama para o envio da letra "m".

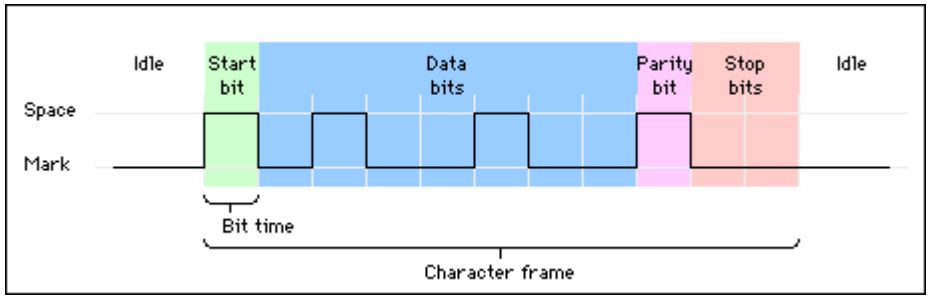


Figura 7.2 – Constituição da trama para envio da letra "m" via RS-232.

Parâmetros usados na comunicação RS-232

Baud rate

É definido como sendo a taxa de transmissão da trama, ou seja, é a quantidade de *bits* por segundo transmitida de um dispositivo para outro. As taxas comuns para uma transmissão são 300, 1200, 2400, 9600, 19200, etc. Tipicamente ambos os dispositivos devem estar configurados com a mesma velocidade, alguns dispositivos, porém, podem ser configurados para auto-detectar a velocidade.

Paridade

A Paridade é um método que consiste na alteração do *byte* a enviar de modo a no fim da transmissão se verificar a precisão e exactidão dos dados. A paridade da trama é normalmente nula (não usada), mas pode ser par ou

ímpar. A paridade nula é simples, os dados não são modificados. Na paridade par, os dados são acomodados de modo a que o número de *bits* a 1 (isto é, a sua contagem num *byte*) seja um número par, isto é conseguido definindo o *bit* de paridade (geralmente os *bits* mais ou menos significativo) como 0 ou 1. Na paridade impar, o número de *bits* a 1 é um número impar. A paridade pode ser usada pelo receptor para detectar a transmissão de erros - se um *byte* foi recebido com o número errado de *bits* 1, então ele deve estar corrompido. Se a paridade estiver correcta então não devem haver erros, ou então há um número de erros par.

Stop Bits

São os *bits* enviados no fim de cada *byte* transmitido com o intuito de permitir que o receptor do sinal se sincronize. Existe uma convenção para a notação se uma configuração de *software* de uma conexão série, esta notação é da forma D/P/S. Sendo que a configuração mais comum é a 8/N/1 que especifica que são transmitidos 8 *bits* de dados, paridade nula e um *stop bit*. O número de *bits* de dados pode ser 7, 8 ou (às vezes) 9. Paridade pode ser nula (N), impar (O) ou par (E), o *bit* de paridade é emprestado aos *bits* de dados, então 8/E/1 significa que um dos oito *bits* de dados é utilizado como *bit* de paridade. Podem ser utilizados na trama 1, 1,5 ou 2 *stop bits* .

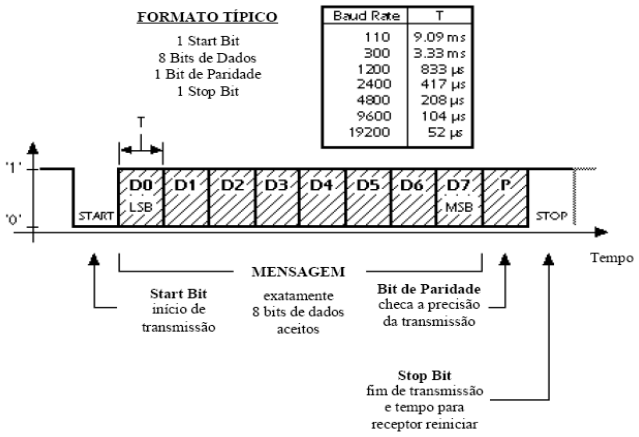


Figura 7.3 – Formato de envio de uma trama via porta série.

[Anexo C] Fluxogramas

Fluxograma rotina "Gestão Timer"

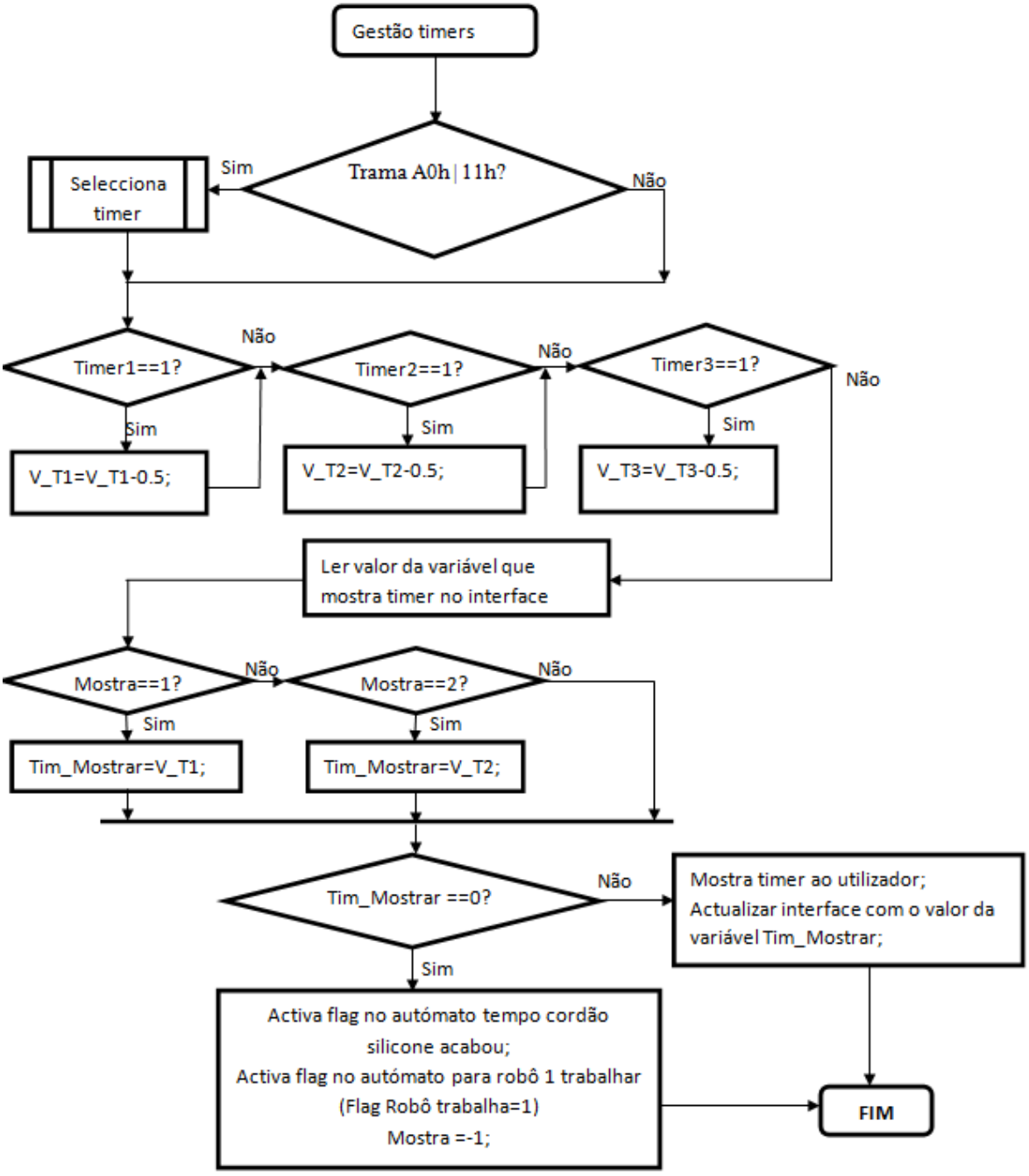


Figura 7.4 – Algoritmo usado na implementação da rotina que gere o timer do cordão de silicone.

Fluxograma da rotina "Selecciona *Timer*"

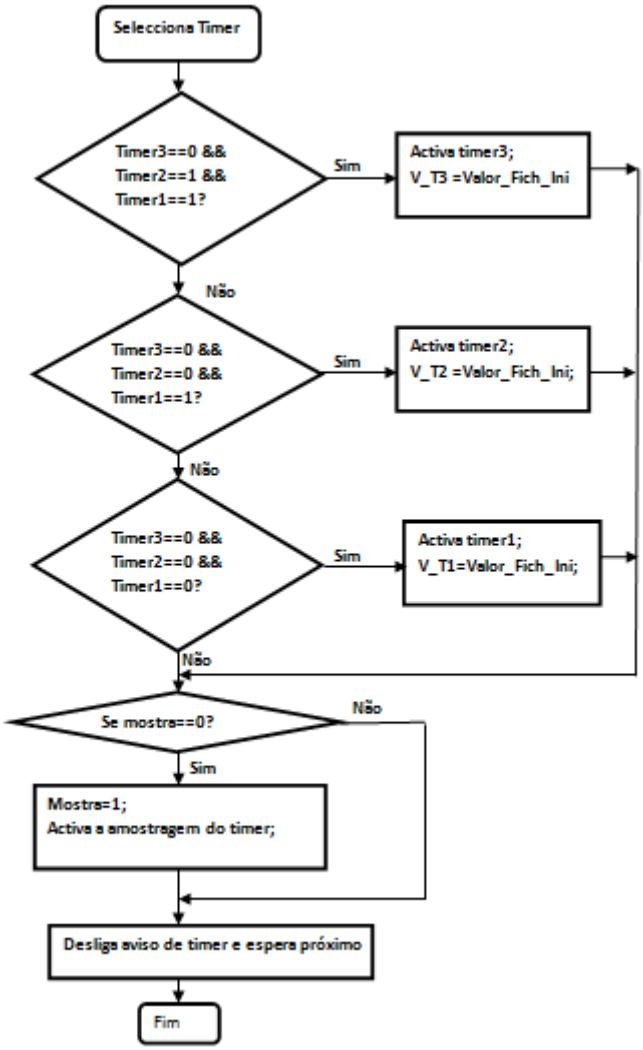


Figura 7.5 – Rotina responsável pela verificação dos timers activos e activação de novos timers.

Fluxograma da rotina “Mesa vai Iniciar Rotação”

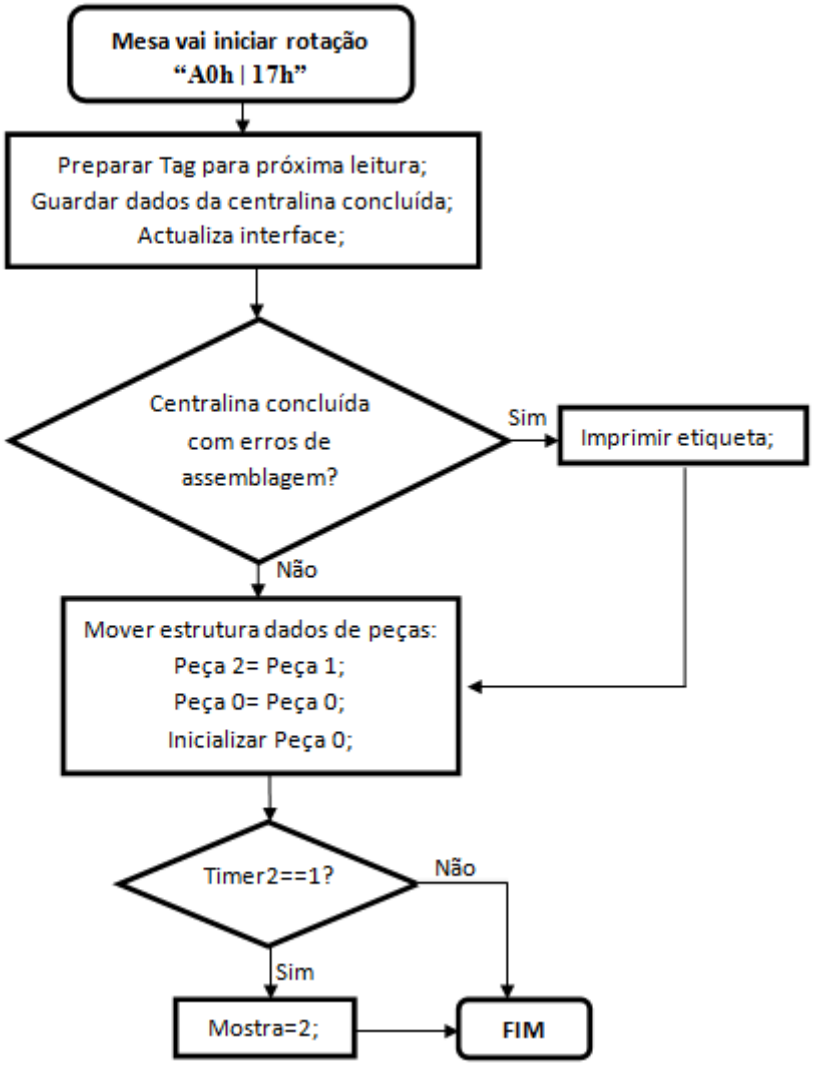


Figura 7.6 – Algoritmo usado na implementação da rotina “Mesa vai Iniciar Rotação”.

Fluxograma da rotina "Mesa terminou a rotação"

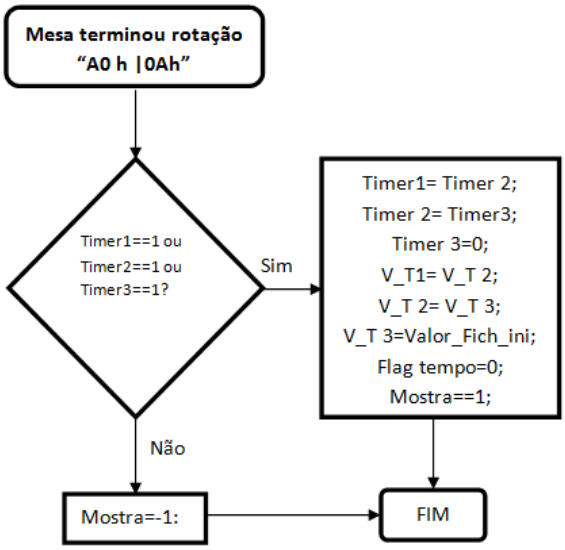


Figura 7.7 – Algoritmo usado na implementação da rotina "Mesa Terminou Rotação".

Fluxograma editar programa robô 1

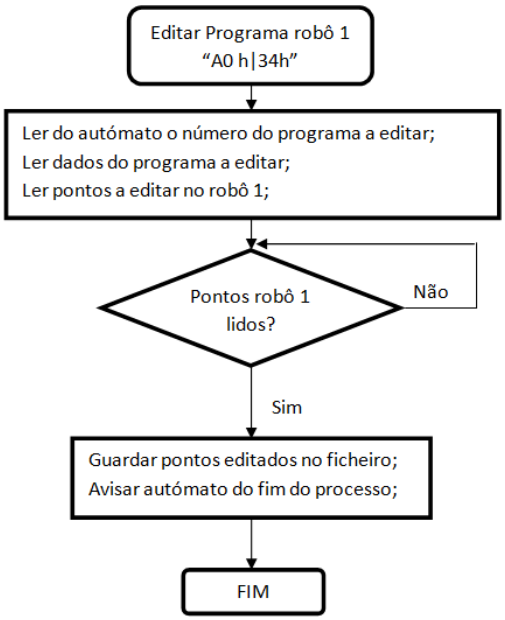


Figura 7.8 – Algoritmo usado na implementação da rotina "editar programa no robô 1".

Fluxograma editar programa robô 2

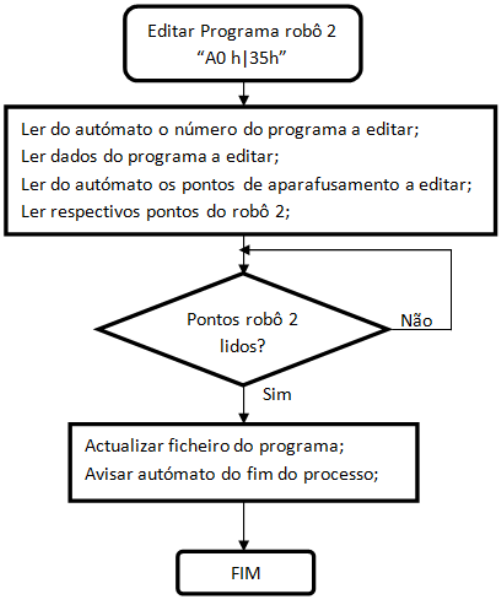


Figura 7.9 – Algoritmo usado na implementação da rotina "editar programa no robô 2".

Fluxograma Criar Novo Programa

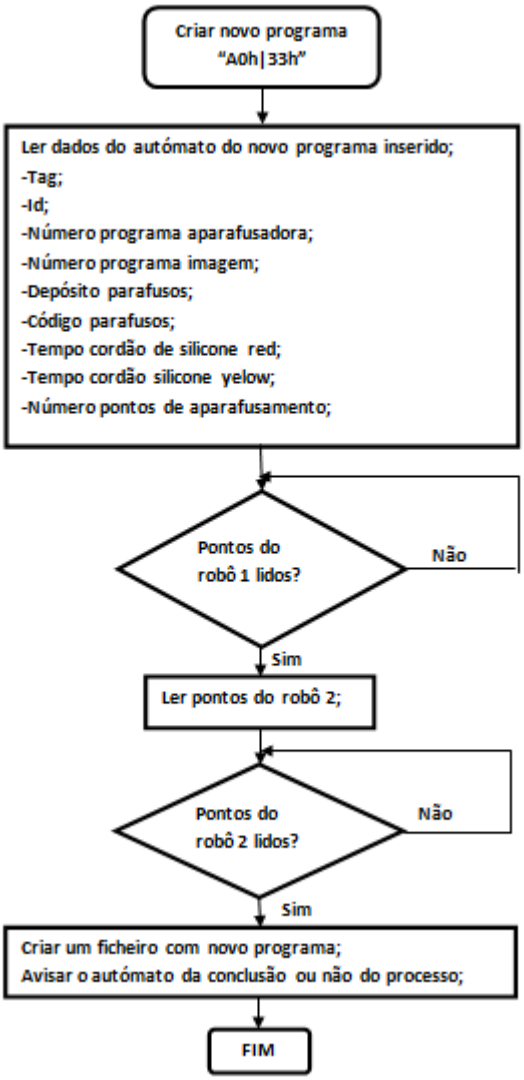


Figura 7.10 – Algoritmo usado na implementação da rotina "criar novo programa" para a célula.

Fluxograma Mudar Programa

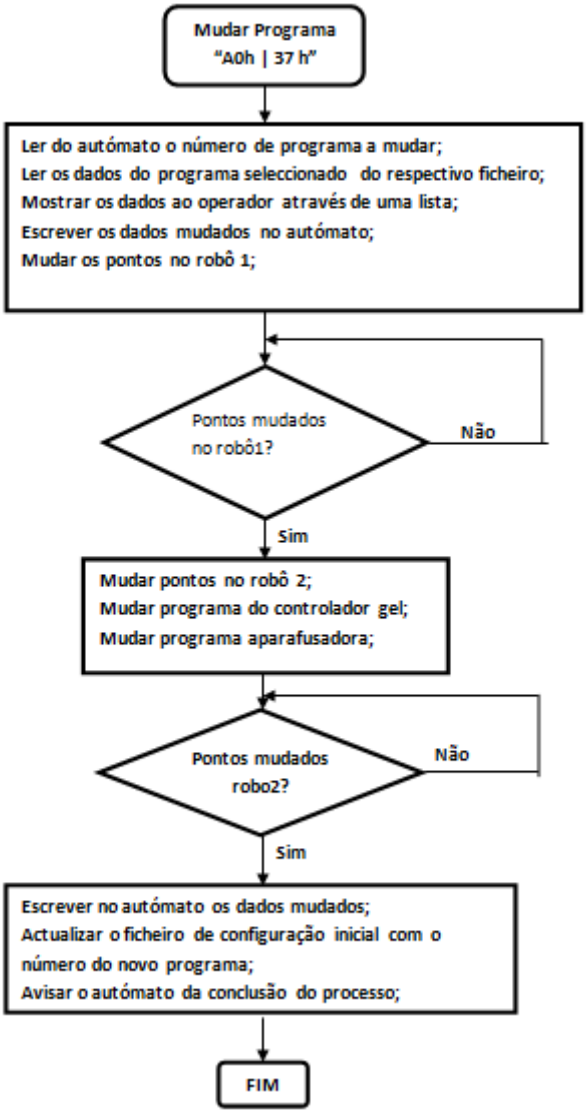


Figura 7.11 – Algoritmo usado na implementação da rotina "Mudar Programa" da célula.