Ano Lectivo 2003/2004

MODULROB

Desenvolvimento Componentes Modulares de Controlo

DEPARTAMENTO ENGENHARIA MECÂNICA

UNIVERSIDADE DE AVEIRO

Trabalho Executado por:

David Manuel Costa Gameiro Nº. 20281

Filipe Mostardinha Carvalho Nº. 22085

1 INTRODUÇÃO 1-4

2 OBJECTIVOS 2-4

3 SOLUÇÕES EXISTENTES E PROPOSTA 3-4

3.1 ACTUADORES 3-5

3.2 SENSORES 3-5

3.3 CONTROLO 3-5

3.4 PROPOSTA ALTERNATIVA 3-6

4 COMPONENTES FÍSICOS 4-6

4.1 ACTUADORES 4-6

4.2 SENSORES E PERCEPÇÃO 4-8

4.2.1 POSIÇÃO DAS JUNTAS 4-8

4.2.2 O INCLINÓMETRO 4-9

4.2.3 O GIROSCÓPIO 4-11

4.2.3.1 Tipos de giroscópios existentes 4-12

4.2.3.2 Integração num robot humanóide 4-14

4.2.3.3 Soluções encontradas e escolha do giroscópio 4-15

4.2.3.4 Descrição do circuito de amplificação 4-15

4.2.4 SENSORES TÁCTEIS / FORÇA 4-16

Especificações e disposição dos sensores 4-16

4.2.4.1 Sensores existentes 4-17

4.2.4.2 Solução escolhida 4-18

4.3 MICROCONTROLADORES 4-20

4.3.1 GERAÇÃO DO CÓDIGO INTEL (HEX) 4-21

4.3.1.1 A compilação 4-21

4.3.2 A PROGRAMAÇÃO DE UM PIC 4-25

4.3.2.1 Programação através da porta paralela 4-25

4.3.2.2 Programação via RS232 4-28

5 SISTEMA DE COMUNICAÇÃO 5-30

5.1 HIERARQUIA DO CONTROLO DISTRIBUÍDO 5-30

5.2 SISTEMA MESTRE – ESCRAVO 5-32

5.2.1 I2C 5-32

5.2.2 CAN 5-33

5.3 SISTEMA CONTROLO PRINCIPAL – MESTRE 5-34

6 PROGRAMAS DE CONTROLO 6-34

6.1 CONFIGURAÇÃO DO PIC 6-35

6.1.1 CONFIGURAÇÃO DOS I/O 6-35

6.1.2 CONFIGURAÇÃO DA USART 6-36

6.1.3 CONFIGURAÇÃO DO CAN 6-36

6.1.3.1 Definição do Baudrate 6-37

6.1.3.2 Definição de máscaras e Filtros 6-39

6.1.4 CONFIGURAÇÃO DA ADC 6-41

6.1.5 CONFIGURAÇÃO DO PWM 6-43

6.1.6 INTERRUPTS 6-44

6.2 ESCRAVO 6-46

6.2.1 CONTROLO SERVO 6-46

6.2.2 ACTUALIZAÇÃO POSIÇÃO 6-48

6.3 MESTRE 6-48

6.3.1 CONTROLO PRINCIPAL (RS232) 6-48

6.3.2 ESCRAVOS (CAN) 6-50

7 CONCLUSÃO E RESULTADOS 7-51

8 BIBLIOGRAFIA 8-53

ANEXO A – ESQUEMA PROGRAMADOR TAIT 8-55

ANEXO B – PLACA DE CONTROLO 8-56

ANEXO C – CIRCUITOS DOS SENSORES 8-57

1 Introdução

Um dos sonhos do ser humano sempre foi a criação de um sistema para o servir e auxiliar principalmente para realizar operações consideradas enfadonhas e cansativas. Uma vez que o ser humano apresenta características funcionais que tendem para a perfeição, então uma plataforma humanóide deverá seguir o mesmo caminho.

Este sistema deverá garantir a versatilidade e a capacidade física de um ser humano, reproduzindo certos movimentos e acções.

Actualmente desenrola-se uma tentativa para que no ano 2050 exista uma equipa de futebol de robôs humanóides que consiga jogar contra uma equipa de seres humanos.

2 Objectivos

- Identificar os componentes físicos para as implementações práticas requeridas;
- Implementar e conceber os componentes electromecânicos numa filosofia de modularidade;
- Definir as soluções conceptuais de controlo local e global do sistema;
- Desenvolver os métodos e suportes de comunicação entre os módulos e a unidade central;
- Testar e avaliar a viabilidade das soluções implementadas usando, sistemas elementares de controlo, como uma primeira abordagem.

3 Soluções Existentes e Proposta

As soluções existentes estão espalhadas por diversos países. O Japão apresenta actualmente o maior número de soluções, todas elas com um nível de performance elevado. Na Europa, não se verifica tento empenho como na Ásia. A Suécia (MURPHY), a Itália (ISAAC) e a Alemanha (Brainstormers, NimbRo) apresentam no entanto plataformas mas com sucesso limitado. A Rússia (ARNE) também apresenta uma solução ambiciosa mas que ainda está numa fase de desenvolvimento.

3.1 Actuadores

Visto que uma das partes mais importantes deste projecto é os actuadores (motores), começou-se por investigar o equipamento utilizado por outras equipas com projectos semelhantes.

A escolha incide na maior parte das vezes em servomotores ou motores DC (actuadores eléctricos) mas também existem soluções com outras formas de energia nomeadamente o ar comprimido. Do que se conseguiu observar, os humanóides de actuadores mecânicos são os que tem mais sucesso. Os servomotores apresentam a vantagem de ter um bom factor binário/peso e alguns apresentam-se com dimensões reduzidas e leves. A inclusão de um sistema de controlo também é uma vantagem.

3.2 Sensores

O número de sensores é muitas vezes indicador do grau de complexidade do robô. Seria possível classificar os robots pelos seus graus de complexidade olhando apenas pela sensorização. Os sensores menos utilidade são aparentemente os sensores tácteis (Sony, Honda, Pino, RoboErectus), seguidos da visão, dos sensores inerciais e dos sensores das juntas.

É muito difícil obter informações acerca dos sensores utilizados nas soluções, mas os acelerómetros da Analog Device, os giroscópios da Gyration e da Murata tal como alguns CCDs são algumas vezes citados.

3.3 Controlo

A maior parte baseia-se em controlos centralizados, composto por uma pequena PC104 (realizará todos os cálculos e acções a tomar) e microcontrolador (controlo motores).

Poucas plataformas apresentam soluções de controlo distribuídas (ARNE, MURPHY), recorrendo a protocolos de comunicação em rede.

3.4 Proposta Alternativa

A solução proposta consiste num sistema de controlo distribuído, recorrendo a sistemas modulares de controlo, isto porque:

- Garantem sistemas mais fiáveis (funcionam de forma independentemente, facilidade em detectar anomalias);
- Sistemas de controlo mais simples;
- Fácil actualização;

No entanto, trazem como inconveniente alguma complexidade nas comunicações.

O problema a resolver passaria pela concepção do controlo local e global para o sistema, alem da sua interligação, mas mantendo o ideal de modularidade.

4 Componentes físicos

Os componentes físicos são uma das partes mais delicadas deste projecto, visto que a sua selecção influencia o bom funcionamento de todo o sistema.

4.1 Actuadores

Visto que uma das partes mais importantes deste projecto ser os actuadores (motores), houve a necessidade de escolher o mais apropriado e do controlar utilizando as ferramentas disponíveis (PIC). Fez-se inicialmente uma pesquisa sobre os diferentes tipos de motores que se poderia utilizar: servomotores, motores DC e motores passo a passo. Desta forma, e após alguma investigação relativamente ao equipamento utilizado por outras equipas com projectos semelhantes ao nosso, pôde-se verificar que todos eles tinham uma certa tendência para utilizar servomotores normalmente aplicados em modelismo ou projectos de robótica de pequena escala (Futaba, Hitec) .Trazem também a vantagem de terem um baixo preço e facilidade da aquisição.

O controlo de servomotores é conseguido através de uma onda de pulsos modulados (PWM – Pulse Width Modulation). Seguindo a informação fornecida pelos fabricantes, este PWM funciona a uma frequência de 50Hz (20 ms), onde todas as posições do servo estão

caracterizadas no intervalo de 4% – 10%. A figura 1 ilustra as posições extremas do servomotor com o sinal de entrada correspondente:

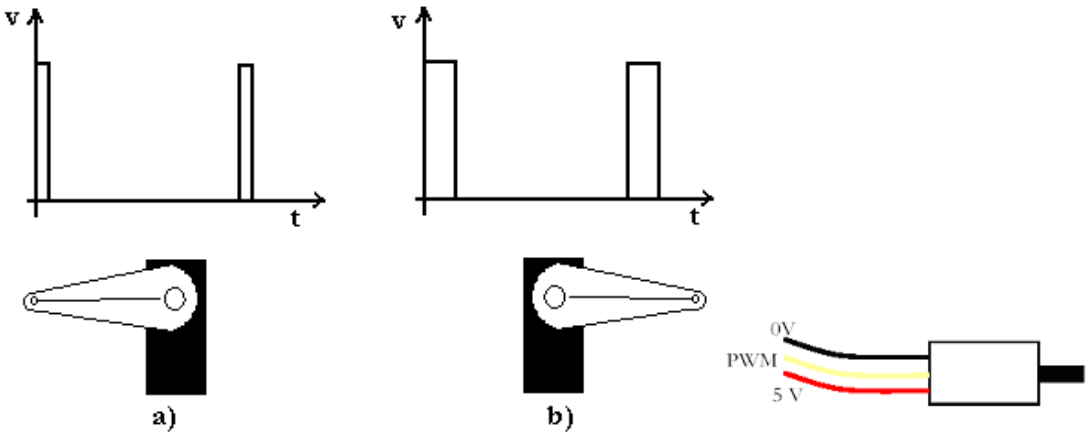


Figura 1 - Posicionamento do servo nos seus extremos

a)duty cycle 4% ➔ 0º b) duty cycle 10% ➔ 180º

O exemplo acima ilustra o caso de um servo "normal" mas existe uma outra gama de sensores ditos "digitais" que trabalham a uma frequência 6 vezes superior. As suas vantagens são uma maior rapidez na resposta, um binário superior relativamente a um servomotor "normal" de mesma categoria e uma maior resolução. Mas este tipo de "servo" também traz a desvantagem de ter um alto consumo de corrente. Além disso, não existe servomotores digitais com o binário que nos era pretendido de 20kg.cm, e foi por esta razão que se adquiriram motores convencionais. A Figura 2 mostra as diferenças entre as placas controladoras dos dois tipos de servomotores.

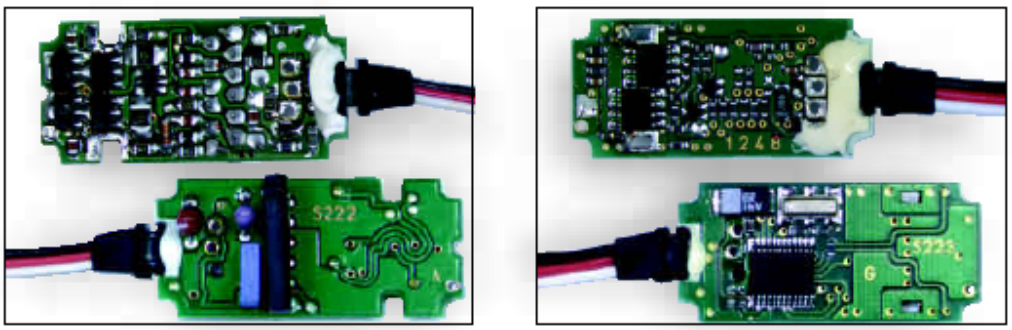


Figura 2 - Foto da placa controladora dos servomotores

convencionais (a esquerda) e digitais (a direita)

O servomotor escolhido é o 815BB+ da Hitec. Este possui um binário de 19,8kg.cm para um peso de 152gr. Este "servo" é utilizado principalmente em mastros de barcos.

Figura 3 - Hitec 805BB+

A Figura 3 representa um servomotor de mesma dimensão que o 815BB+ mas tem uma amplitude máxima de 90º.

4.2 Sensores e Percepção

Tal como o humano, um robô deve ter a percepção do seu maio envolvente de forma a agir adequadamente. Os sensores têm a missão de informar como se situa o robô no espaço e no tempo. É necessário saber a posição das juntas do robô e qual é o movimento que este tinha naquele momento de forma a conseguir antecipar futuros movimentos.

Tendo em conta as informações que desejamos obter, foram encontradas várias soluções tecnológicas. A grande preocupação foi de encontrar sensores de preço razoável mas cujas dimensões e medições se adequassem a um robô humanoide de pequeno porte.

4.2.1 Posição das juntas

Os primeiros sensores analisados foram os das juntas. Para conhecer a posição das juntas, tendo em conta que todas elas são rotacionais, é necessário dispor de um sensor rotativo. Existem três grandes tipos de sensores que se podem ser usados para medição angular das juntas: os potenciómetros, os codificadores e os resolver.

Os resolver são os mais dispendiosos. Os codificadores são principalmente eficazes para múltiplas rotações enquanto que as nossas juntas não irão ultrapassar 180º. O recurso aos potenciómetros impõe-se no então pela questão do custo e da comodidade. Numa primeira fase, pensou-se em utilizar um potenciómetro que se iria acoplar a junta mas verificou-se que era possível utilizar o potenciómetro que se situa dentro do servomotor, ganhando assim espaço. O sinal que sai destes potenciómetros internos varia entre 0,85V e 1,75V. A Figura 4 mostra uma fotografia exterior de um servomotor e um esquema interno de funcionamento acrescentado de um fio (verde) que sai directamente do potenciómetro.

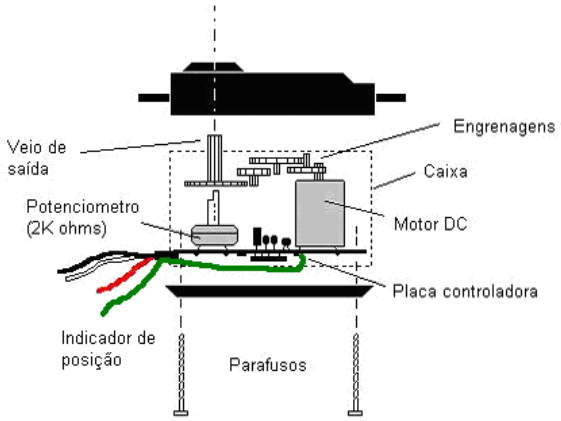


Figura 4 - Esquema interno de um servomotor

4.2.2 O inclinómetro

O inclinómetro é, na realidade, um acelerómetro que mede de acelerações lineares. O objectivo do inclinómetro é indicar a posição angular do tronco do robô com o solo. Utilizou-se o facto de existir a gravidade para "transformar" o acelerómetro em inclinómetro. De forma a ter bons resultados, o acelerómetro deve ter uma gama de trabalho da ordem da aceleração gravidade (1g).

Na pesquisa efectuada, foram encontradas vários fabricantes de acelerómetro nomeadamente:

- Analog Device: ADXL103 / ADXL203 (±1,7g, 1 e 2 eixos)

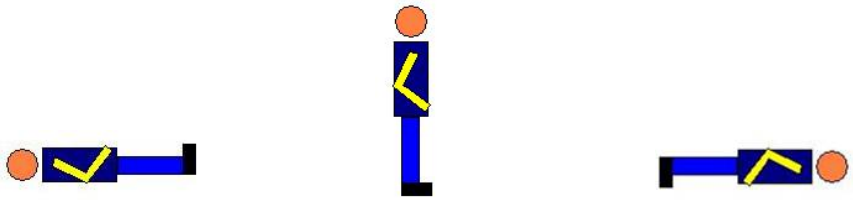
  ADXL202 (±2g, 2 eixos)

  ADXL150 / ADXL250 (±5g até ±50g, 1 e 2 eixos)

	ADXL311 (±2g, 2 eixos) …
- Crossbow:	HF series (±10 to ±100g, 1 eixo)
- Honeywell / Sensotec	MAT53 (-80g, 1 eixo)
	Q-Flex MultiAxis (dispositivo multi-eixos)
	QA650 (+-30g, 1 eixo)
	QA700 (+-30g, 1 eixo)
	QA750 (+-30g, 1 eixo)
	QA1400 (+-60g, 1 eixo) …
- Phone-Or	Optical accelerometer (±1, ±2, ±5, ±10g, 1 eixo)
- Seika	B1 (±3g, 1 eixo)
	B2 (±10g, 1 eixo)
	B3 (±50g, 1 eixo)

Foi utilizado o chip ADXL202JE da Analog Device por ter uma gama de trabalho de 2g, possuir dois eixos de medição ortogonais e serem oferecidas como amostra. Assim sendo, podemos conhecer a inclinação do robô nos dois eixos. De todos os acelerómetros pesquisados, os da Analog são os mais integrados (5mm×5mm).

A amplitude das saídas do acelerómetro é de 1.65V-2.3V o que corresponde a seguintes posições extremas (a título de exemplo):

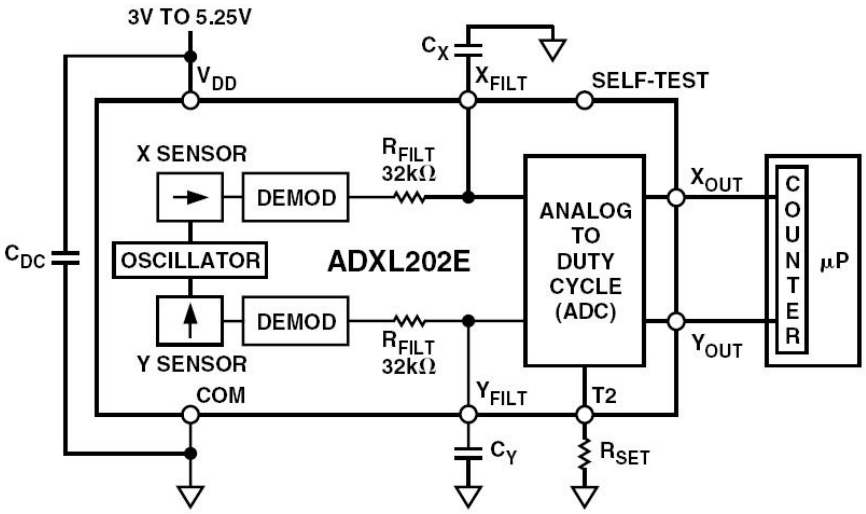


2,3V: robô costas para baixo 2,05V:robô em pé 1,65V: robô de ventre para baixo

Figura 5 - Ilustração da tensão de saída do inclinómetro

Montagem do acelerómetro

O acelerómetro faculta-nos quatro saídas independentes, dois por cada eixo de medição: uma delas é analógica outra digital (PWM) conforme o seguinte esquema interno do acelerómetro:



Xout, Yout:

saídas digitais

Xfilt, Yfilt:

saídas analógicas

Figura 6 - Esquema interno do ADXL202►

O dispositivo fornece sem tratamento, o sinal analógico para a entrada dos micro-controladores. Devido a ruídos proveniente da *breadboard*, as saídas analógicas não tinham sinal de qualidade, decidiu-se então utilizar as saídas PWM e linearizá-las.

4.2.3 O giroscópio

O giroscópio é um dispositivo sensível as velocidades angulares. Existem vários princípios físicos que permitem detectar esta grandeza. Os tipos de "gyros" existentes mais comuns são os mecânicos, os de fibra óptica e os de vibração.

4.2.3.1 Tipos de giroscópios existentes

Giroscópio de massa giratória

Figura 7 - Giroscópio mecânico

É o giroscópio clássico que possui uma massa giratória constante com três eixos móveis. Quando o giro (giroscópio) é inclinado, o efeito giroscópico causa a precessão (movimento ortogonal a direcção de inclinação) no eixo da massa giratória, de onde se sabe o ângulo movido. As limitações mecânicas causam numerosos erros além de serem frágeis. As suas características não se adequam a as da construção de um robô humanoide.

Giroscópio de fibra óptica

O giroscópio de fibra óptica consiste num grande comprimento de fibra enrolada numa bobina. Uma luz laser é enviada nas duas extremidades da fibra usando um divisor de feixe reflecte 50% da luz e transmita 50%. A luz que atravessa a bobina, estática, no sentido dos ponteiros sai no fim com a mesma fase que a luz introduzida em sentido inversa, isto porque ambas viajaram exactamente a mesma distância. Se agora o giroscópio for girado por exemplo no sentido horário, então a luz que viaja na fibra enrolada no sentido horário ira demorar mais tempo a alcançar o fim da fibra porque esta sempre a afastar-se do caminho da luz. A luz introduzida em sentido anti-horário irá demora menos tempo porque a o fim da fibra se aproxima mais rapidamente do início do feixe de luz. Isto introduz uma diferença de fase entre as duas ondas luminosas emergentes que é proporcional a rotação da bobina. Este efeito é chamado efeito de Sagnac (ver Figura 8).

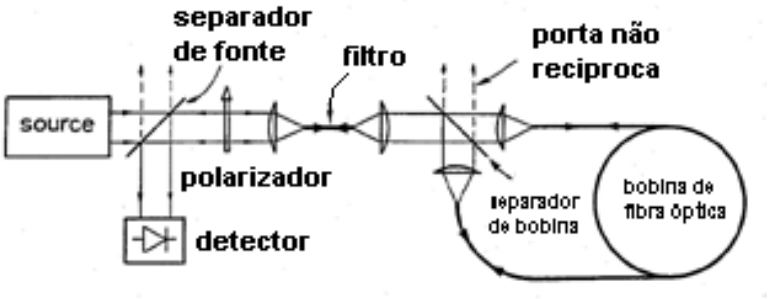


Figura 8 - Princípio de um giroscópio óptico

Um grande problema em relação a este tipo de sensor é a sua fragilidade devido ao anel de fibra óptica.

Giroscópio de vibração

Um elemento vibrante (ressonador), quando girado, é sujeito ao efeito de Coriolis que causa uma vibração secundária ortogonal ao sentido da vibração original. Sentido a esta segunda vibração, a velocidade angular pode ser detectada. Para isso, o efeito piezoeléctrico é frequentemente utilizado, além que os "gyros" vibratório são geralmente chamados "piezo", "ceramic" ou "quartz" giroscópios, embora na verdade nem sempre a vibração e a detecção usar o efeito piezoeléctrico. Este tipo de "gyro" não precisa de manutenção. O inconveniente é, quando é utilizado em vibrações externas, ele não consegue distinguir entre a vibração secundária e a vibração externa. Atenuadores (amortecedores) em volta dele não resolvam o problema porque afectam o movimento rotacional logo fazem como que o servo responda pior. Existem "gyro" CRS da Silicon Sensing System que ultrapassaram este problema.

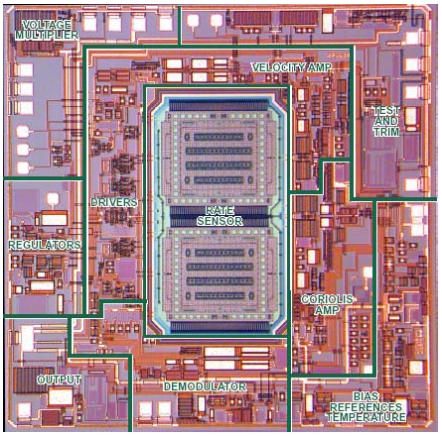


Figura 9 - Fotografia do circuito interno

do ADXRS da Analog Device (7mm×7mm)

4.2.3.2 Integração num robot humanóide

Nem todos os giroscópio existentes são possíveis de ser introduzidos num robô humanoide. Existem muitos factores limitativos que iremos enumerar.

Num primeiro lugar, as dimensões são um factor limitativo. Alguns tipos de sensores apresentados acima não se adequam ao nosso sistema nomeadamente os giroscópios de massa giratória. O giroscópio instalado no Robuter do Laboratório de Automação e Robótica (Gyrostar da KVH) é um bom exemplo de aparelho que, devido ao seu peso e dimensões, não se pode integrar no robô. O tamanho (massa e volume) foi então fundamental durante a pesquisa, sendo o primeiro critério a ser avaliado.

Em segundo lugar vem a tensão de alimentação e o consumo de corrente do dispositivo. De facto, este parâmetro vem antes dos restantes por ser também muito limitativo. O robô foi pensado para trabalhar numa gama de 0 a 6V. Muitas soluções existentes não cumprem esta restrição.

Outro ponto de destaque são as características do giroscópio. Este deve possuir uma gama de medição de ordem semelhante á da grandeza pretendida, deve ter baixo um baixo *drift* e uma boa resolução. Foi estimado a velocidade angular máxima como sendo 360º/s. Além deste valor, indicaria que o robô esteja a rodar demasiado rapidamente para que os sistema tenha tempo de resposta suficiente.

Existem mais factores a ter em conta nomeadamente o preço, a gama de temperatura, a facilidade de integração na placa do circuito (soldadura) e o tipo de dados de saída.

4.2.3.3 Soluções encontradas e escolha do giroscópio

Na nossa pesquisa se destacaram as seguintes marcas de giroscópio:

- Analog Device	ADXRS150
	ADXRS300
- Aptec	ARS-01 & ARS-01S
	ARS-04E & ARS-04R
	ARS-09 & ARS-09S
- Gyration	Micro-gyro 100
- Murata	Gyrostar ENC-03J
	Gyrostar ENC-03M
- NEK TOKIN	MDP-A3U7
- Silicon Sensing System	CRS03
- Systron	AQRS series
	HZ1 series
	QRS11 series

Todos os giroscópios enunciados respeitam a restrição da dimensão mas alguns são demasiados caros (Silicon Sensing System: 348€), outros não se consegue encontrar um revendedor em Portugal (Gyration) etc…

Da lista escolheu-se o Murata ENC-03J. A escolha baseou-se principalmente na disponibilidade comercial. Este "gyro" corresponde muito bem a todos os critérios exigidos. No entanto, a saída do sensor (analógica) é um pouca fraca, por isso é necessário proceder a sua amplificação.

4.2.3.4 Descrição do circuito de amplificação

De forma a amplificar o sinal, utilizou-se um amplificador de instrumentação. Uma pesquisa na Internet mostrou o amplificador de instrumentação AD620 da Analog Device como sendo uma boa escolha. A principal dificuldade na busca foi o facto do amplificador ter uma tensão de alimentação muito baixa (0-5V). O amplificador escolhido consegue ser a

alimentado entre ±2.3V até ±18V. De forma a criar um offset para a tensão de alimentação decidiu-se criar um divisor com díodos e uma resistência como mostra a Figura 10.

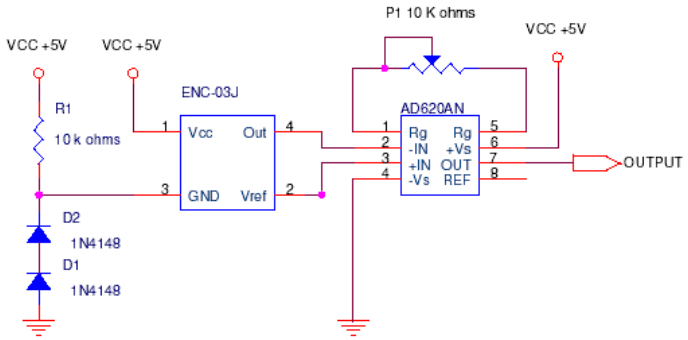


Figura 10 - Esquema de amplificação do giroscópio

A tensão a saída do circuito de amplificação é de 2V é oscila entre 1,9V e 2,2V quando o sensores está a girar.

4.2.4 Sensores tácteis / força

Os sensores tácteis de força, como o nome indica, dão informação da força exercida pelo pavimento em vários pontos do pé do robô. Desta forma consegue-se determinar a verticalidade do centro de massa. Os sensores permitem-nos também ter conhecimentos sobre o pavimento.

Especificações e disposição dos sensores

Tal como os outros sensores, eles devem ser pequenos de forma a serem implementados num pé de área máxima 21cm * 9cm. Devido a forma rectangular das chapas, pensou-se naturalmente dispor um sensor junto a cada vértice, logo são necessários 8 sensores no total dos dois pés como mostra a Figura 11.

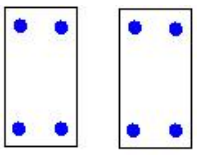


Figura 11 - Disposição dos sensores tácteis

De maneira a determinar uma gama de trabalho dos sensores e recorrendo ao pior cenário, os sensores de força devem no máximo ser sensíveis a força de 80N que corresponderia ao peso de todo o robô na esquina do pé. Este valor só serviu para restringir a busca dos sensores. Quando o robô (de massa aproximada 8kg) está de pé e não se mexe, cada sensor irá medir 10N. Se estiver só num pé, o valor sobe até os 20N. Uma estimativa de uma ordem de grandeza do trabalho dos sensores seria de 40N tendo em conta a não distribuição uniforme do peso do robô nos sensores. A figura a seguir mostra como foi obtido este valor:

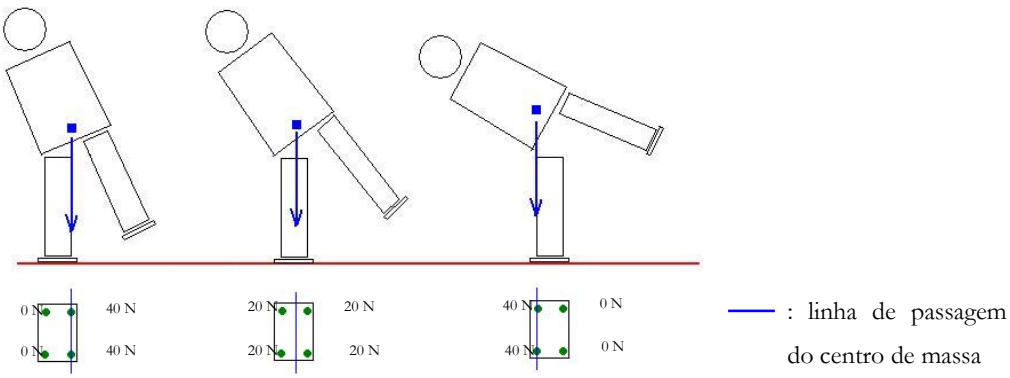


Figura 12 - Diferentes posições do centro de massa em relação ao pé

Na primeira e última ilustração da Figura 12, o robô tem o seu centro de massa em cima da linha azul, ao seja, o robô está numa posição instável e todo o seu peso recai nos dois sensores por qual passa a linha azul.

4.2.4.1 Sensores existentes

Da busca salientaram-se os seguintes dispositivos:

- Honeywell	FS01/03 sensor
	MICROSWITCH Force Sensor
	FSS Low Profile Force Sensors
	FSG & FSL series
- Cooper Instruments	LMP510
- Interface	Model LBM series
- DC Europe	BC300 series

- FlexiForce	A201 (estes sensores não são de pressão mas funcionam como macro-extensores)
- Muse Instruments	M1000 model
- Omegadyne	LCKD compression series
- Sendev	Mini-buttons

Todos eles respeitam o tamanho pretendido. Segundo a informações dos distribuidores, existe uma grande variação de preços desde \$55*2 ≈ 88€ (2 *packs* de 4 dos *FlexiForce* A201 110N) até \$550*8=2400\$ ≈ 1934€ (*Omegadyne* M1000 10lb).

4.2.4.2 Solução escolhida

Após análise das soluções existentes, entendeu-se que era possível criar sensores de força específicos para o robô usando extensómetros, como aqueles utilizados em ensaio mecânicos. Aproveita-se o facto de uma chapa metálica se deformar proporcionalmente a força aplicada quando está no regime elástico. Ao medir a deformação temos então uma informação da força exercida. A Figura 13 a seguir mostra a solução encontrada para o pé:

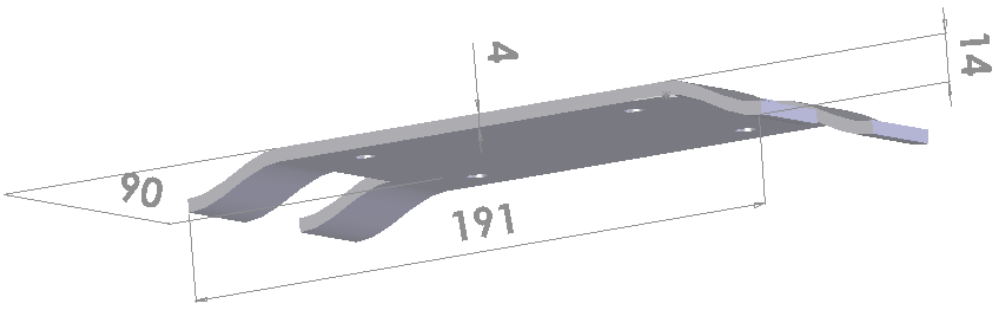


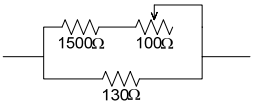
Figura 13 – Modelização de uma solução para o pé

Um extensómetro consiste basicamente numa resistência que aumenta de valor quando está sujeita a alongamento. Esta variação depende do tipo de extensómetro aplicado. O extensómetro utilizado para testes tem um *gauge* factor de 2,02 e aguenta uma extensão máxima de 50000 μ-strains (50mm/m → 5% de deformação). Foi realizado um estudo de

modo a saber qual é a deformação da chapa no lugar onde está colado o extensómetro para espessuras de chapa de aço diferentes. Este estudo foi realizado através do software CosmosWorks da SolidWorks. Obtivemos que a deformação local da chapa para um esforço de 50N era de 0,017% para uma chapa de 3mm, de 0,049% para uma chapa de 2mm e 1,06% para uma chapa de 1mm.

O resultados mostram que acima de 1mm, não ocorre o perigo de extensão máxima por baixo do extensómetro, no entanto não se pode escolher a chapa de 1mm. Durante os ensaios com uma chapa de 1,5mm, observou-se uma deformação após múltiplas utilizações. O facto da chapa na estender mais do que é permitido no sitio onde está colado o extensómetro não prevê a situação de deformação no resto da chapa. Portanto é necessário ter uma chapa superior a 1,5mm. O outro caso limitativo da espessura deve-se ao facto da deformação da chapa debaixo do extensómetro ser tão baixo que o sinal de saída do amplificador será demasiado fraco, diminuindo a resolução. Como não se deseja utilizar mais do que um amplificador por extensómetro logo devera-se respeitar uma espessura máxima. Como demonstrado abaixo, está praticamente a ser utilizado no seu limite, portanto a espessura não poderá exceder os 2mm.

De forma a obter uma diferença de potencial mensurável, é necessário montar uma ponte de Wheatstone microajustável que permite medir uma diferença de potencial entre o estado de repouso e o estado de tracção. Poder ter o maior rigor nas medições da ponte, decidiu-se apostar por uma montagem de resistências (130Ω e 1500Ω) e potenciómetros multivoltas em paralelo (100Ω). A amplitude da resistência é então:



$$\begin{cases} \left(\frac{1}{130}+\frac{1}{1500}\right)^{-1} = 119{,}632\Omega \\ \left(\frac{1}{130}+\frac{1}{1600}\right)^{-1} = 120{,}231\Omega \end{cases}$$

Amplificou-se o sinal via um amplificador de instrumentação (ver próxima figura). Seguindo o procedimento acima, obtemos então um sinal que têm uma amplitude de 0,4V bem quantificável pelo PIC.

Durante as experiências, apercebeu-se que havia flutuações ao nível do sinal de mesma grandeza que o próprio sinal. Estas flutuações se deve ao suporte de montagem do circuito, por isso decidiu-se montar o circuito numa placa de *wrapping*.

A Figura 14 abaixo representa o esquema efectuado para o sensor do pé.

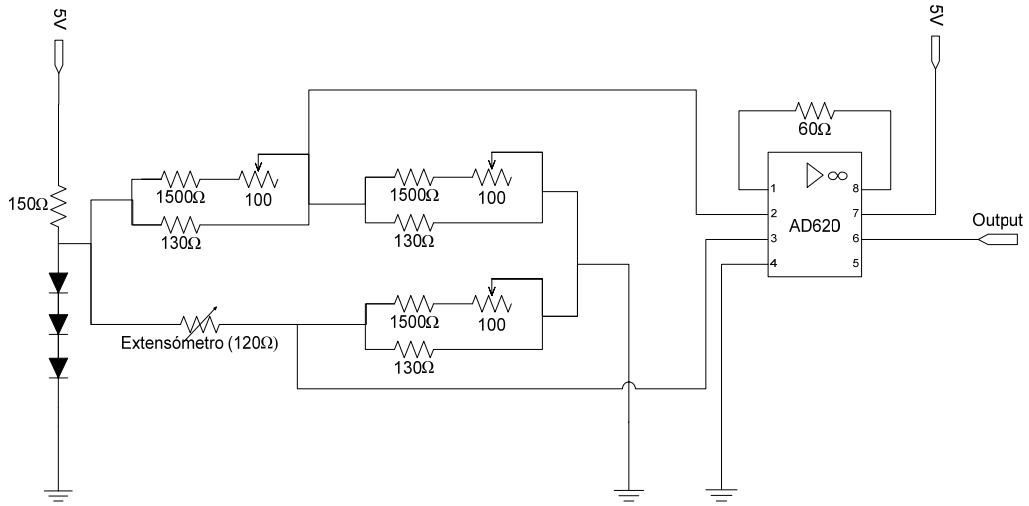


Figura 14 - Esquema eléctrico do sensor do pé

Como se vê, para a amplificação do circuito utilizou-se uma resistência de 60 ohms, o que corresponde a uma amplificação do sinal de:

$$R_G = \frac{49{,}4k\Omega}{G-1} \Rightarrow G = \frac{49{,}4k\Omega}{R_G} + 1 = \frac{49{,}4k\Omega}{60} + 1 \approx 824{,}3$$

O limite do amplificador é 1000 vezes o sinal de entrada.

4.3 Microcontroladores

O controlo apenas poderia ser realizado por unidades de processamento de pequena dimensão, ao qual se poderia anexar componentes. Os componentes a anexar seriam ADC, suportes comunicação (USART, I2C, CAN, etc), entre outros.

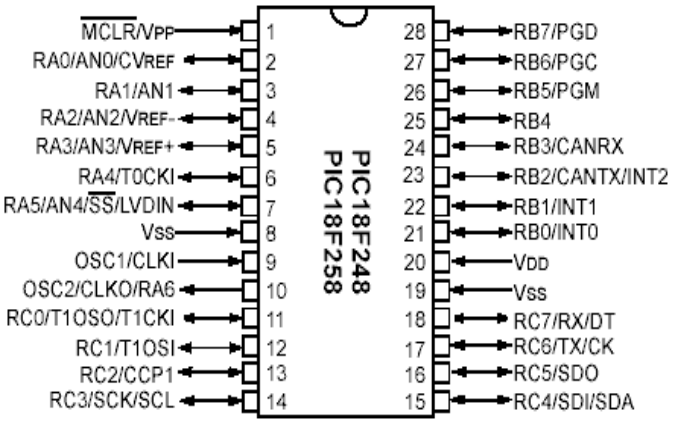
A solução passou por recorrer a microcontroladores os quais incluem módulos de comunicação, geração PWM, ADCs, etc. Recorrendo a projectos de anos transactos como algum suporte verificou-se o recurso a microcontroladores (PIC) da Microchip®[12], com informação e documentação sobre tais integrados (manuais, exemplos de aplicação, etc.). De salientar ainda a enorme gama de microcontroladores que a Microchip®[12] possui, permitindo assim uma fácil escolha do microcontrolador a usar.

O microcontrolador a usar teria de possuir uma boa capacidade de armazenamento de dados além de uma boa velocidade de relógio, alem de diversos protocolos de comunicação. Consultando soluções produzidas pela Microchip®[12] conclui-se que o PIC18F258 seria a melhor solução, o qual apresenta a seguinte as seguintes características:

Características fundamentais	PIC18F258
Frequência de operação	DC-40MHz
Memória Flash	32K
Memória dados	1536 Bytes
Interrupts	17
Portas (I/O)	Portos A, B, C
Timers	4
PWM	1
Comunicação série	MSSP, USART, CAN
Conversores A/D de 10 bit	5 Pinos

Figura 15 – Principais características do PIC18F258 e diagrama dos pinos

Uma das características mais significativas deste PIC consiste num novo módulo de comunicação (CAN) e na possibilidade de atribuir prioridades aos interrupts além de poder atingir velocidade de relógio até 40 MHz.



4.3.1 Geração do código intel (Hex)

O código intel é uma listagem dos códigos que o processador do microcontrolador entende. O ficheiro com o código intel é realizado através do compilador MPLABC18 com base no ambiente MPLAB IDE 6.40 da Microchip®[12].

4.3.1.1 A compilação

O início de um novo projecto realiza-se através da selecção do sub menu *New* em *Project*. De seguida deve-se indicar o nome a dar ao projecto e a localização para esse projecto.

Na janela Workspace adiciona-se os diversos ficheiros a compilar, source ou header files, além da localização dos ficheiros linkers os quais se encontram no directório do compilador MPLABC18 na subdirectoria **lkr**.

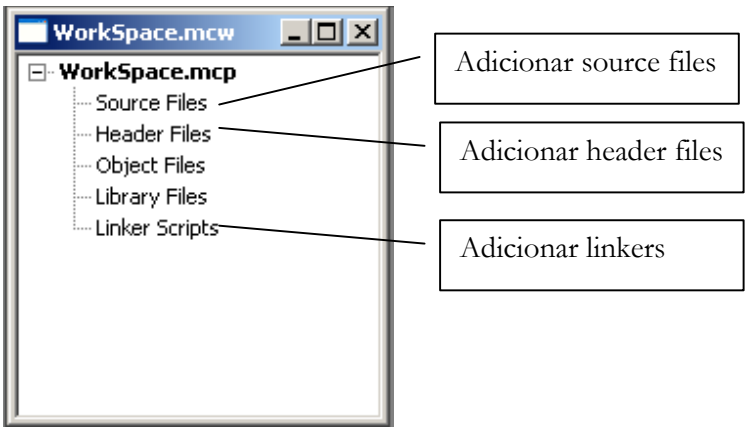


Figura 16 - WorkSpace do

1) Selecção do compilador através do menu *Select language ToolSuite*

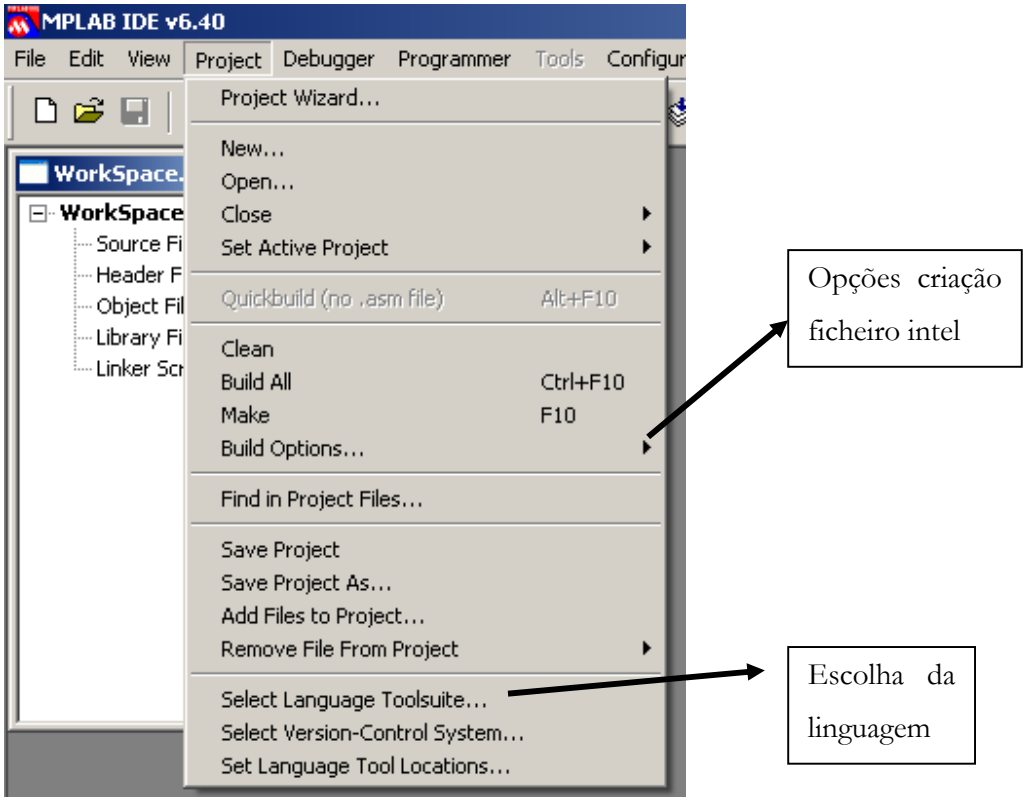


Figura 17 – Menu de selecção da linguagem

2) Indicar qual a ferramenta (compilador) a usar Microchip C18

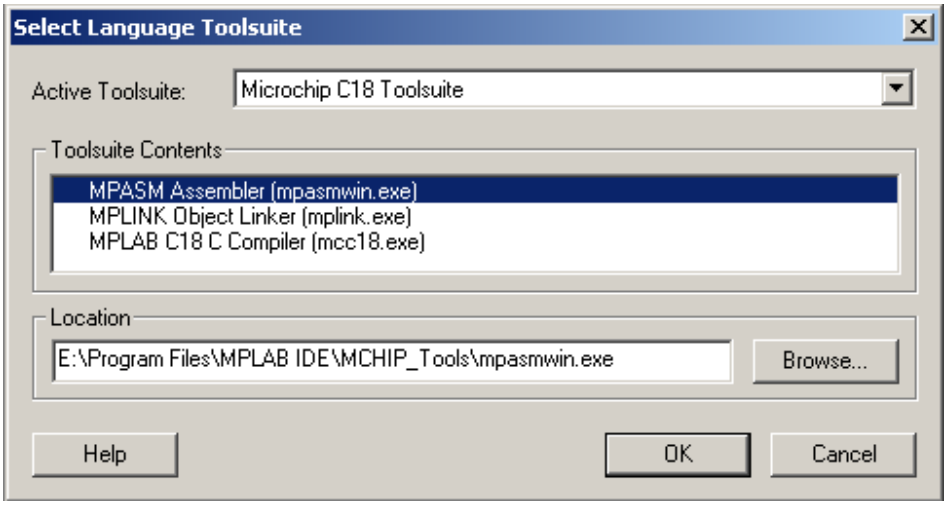


Figura 18 - Escolha da linguagem

3) Definir a opções de criação do ficheiro intel

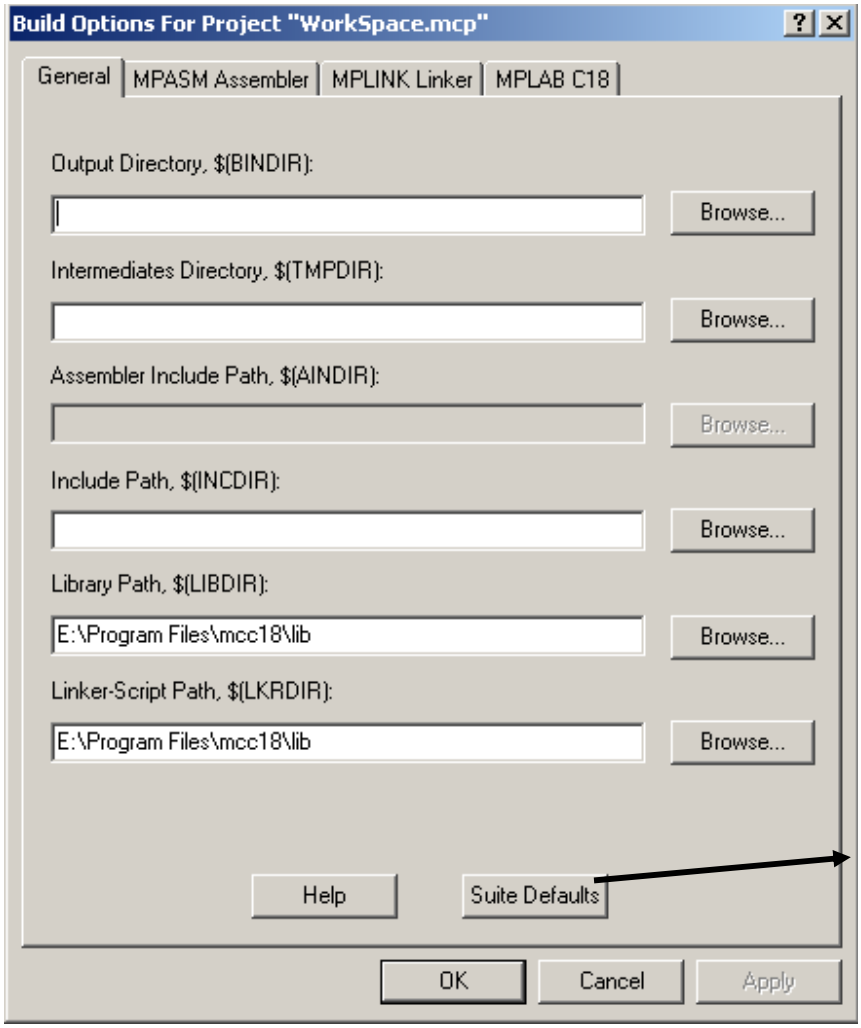


Figura 19 - Localização de livrarias

4.3.2 A programação de um PIC

Existem três formas de transferir um programa para o PIC:

- Através de um dispositivo especifico – programador (linha série/ paralela);
- Através da linha série;
- Através do bus CAN, ou outro;

4.3.2.1 Programação através da porta paralela

Um dos métodos mais utilizados na programação de PIC's, é através da utilização de um programador específico para programar este tipo de microcontroladores. O qual foi desenhado inicialmente para programar o PIC16F84 por um senhor chamado BOB Blick [11], mas que funciona com a maior parte dos microcontroladores da Microchip®[12].

O esquema electrónico deste circuito encontra-se em anexo (Anexo A).

Existem outras soluções para realizar a programação do PIC, tal como Schaer+ (Figura 20) ou o dispositivo MPLAB ICE2000 da Microchip®[12] o qual permite escrever e testar online o código em Assembler ou C, tanto o software como os esquemas electrónicos são de difícil acesso.

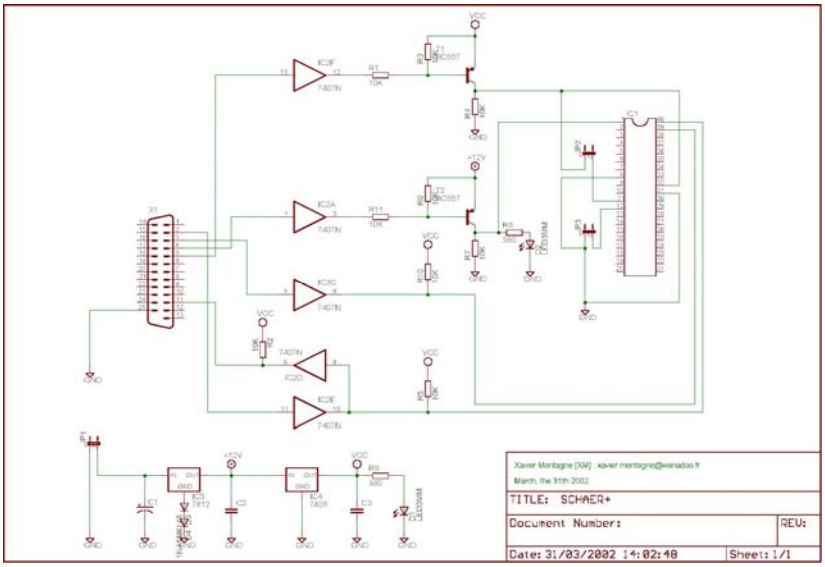


Figura 20 - Esquema programador Schaer+

Na programação do PIC através da porta paralela usou-se o software IC-PROG [14] o qual foi desenvolvido para funcionar debaixo da plataforma Windows® 2000. Para o correcto funcionamento do programa deve-se realizar os seguintes passos:

1) Colocar o ficheiro IcProg.sys no mesmo directório do ficheiro executável Icprog.exe
2) Se executar o programa na plataforma Windows® XP definir a sua compatibilidade (Figura 21)
3) Executar o programa IC-PROG[14]entrar no menu *Settings* e seleccionar o sub menu *Options* e activar a opção Windows® XP na pasta *Misc* (Figura 22)

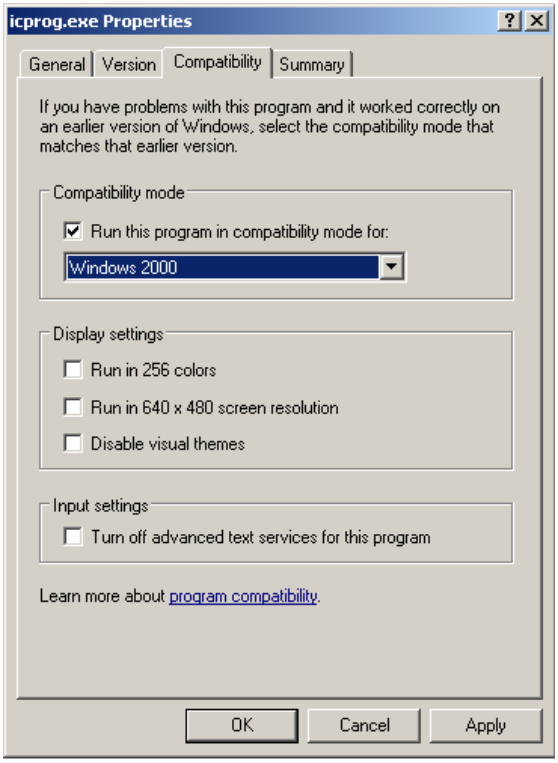


Figura 21 - Compatibilidade ICProg[14] ⇔ XP

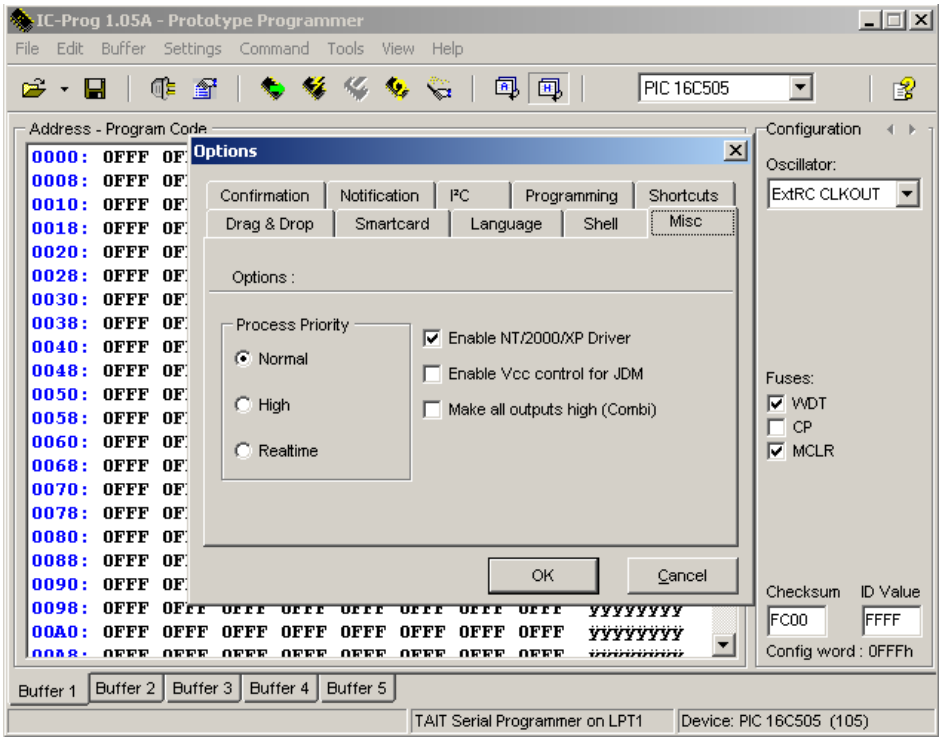


Figura 22 – Configuração IC-Prog[14]

Depois de efectuar os passos anteriores, o programa reiniciará efectuar-se-á a configuração do hardware para a placa a utilizar.

Para realizar a configuração clicar no menu *Settings* e seleccionar o sub menu *Hardware*

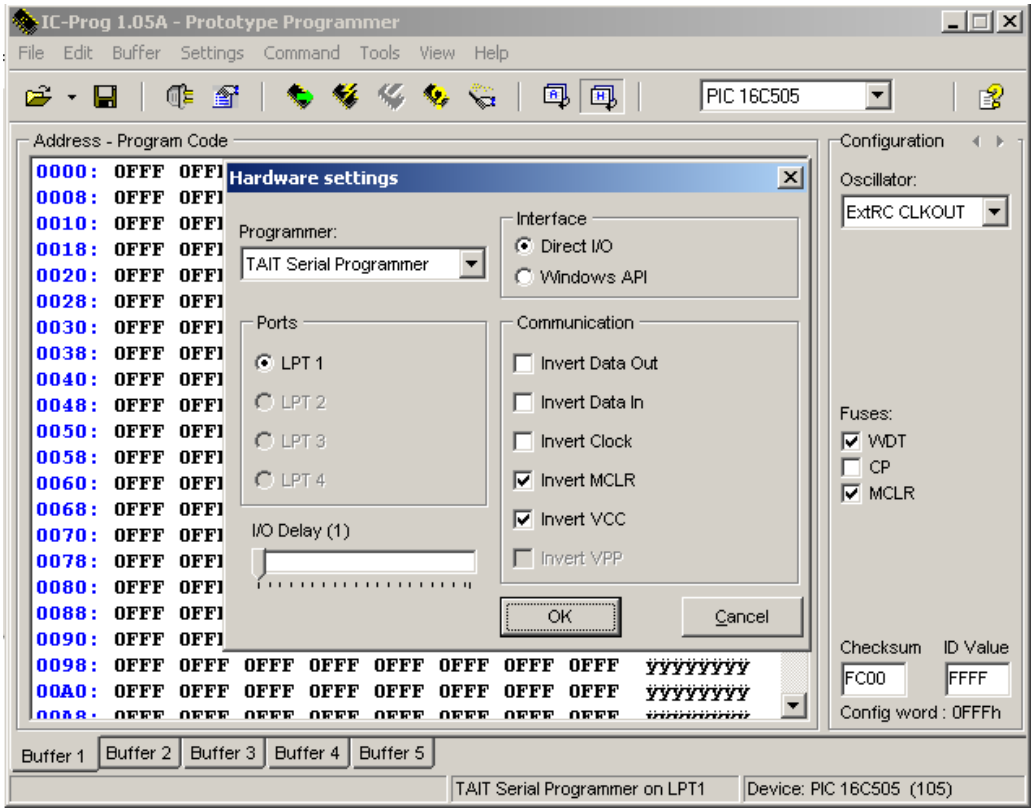


Figura 23 - Configuração do Hardware

Seleccionar o programador TAIT – Serial Programmer na ComboBox Programmer, activar Invert MLCR e Invert Vcc. O I/O Delay deverá ter o valor de 7 para o processador de Pentium® IV a 1.7GHz.

O problema do uso deste programador é o facto da necessidade de colocar o PIC nesta placa, tornando-se um fardo numa fase de desenvolvimento do código.

4.3.2.2 Programação via RS232

Para resolver a lacuna da programação via porta paralela, aparece a programação pela porta série sem necessidade de hardware extra. Para que isso seja possível é previamente programado no PIC um pequeno programa que controla a comunicação (**bootloader**) com o PC e que transfere a informação recebida para a memória de programa do PIC. A programação do **bootloader** é efectuada com ajuda do programador anterior.

Bootloader

O bootloader é um pequeno programa colocado no início da memória (flash, Figura 24) do PIC que irá controlar as comunicações do PIC. O *bootloader* interage com um software a

correr no PC – Tiny Bootloader [15]o qual controla o envio do ficheiro em formato Intel® (HEX).

Figura 24 – Localização bootloader na memória

O bootloader é executado sempre que o PIC é inicializado ficando um certo tempo à espera da chegada de informação pela linha série. Chegando dados pela porta série realiza a sua transferência para a zona de memória livre (código utilizador), caso contrário havendo um código válido em memória dá-se a sua execução.

Na programação do PIC 18F258 recorreu-se ao bootloader desenvolvido pelo senhor Claudiu Chiculita[15], o qual proporciona o código fonte e uma aplicação para o descarregamento do ficheiro hex. O código fonte do bootloader pode ser adaptado a diversos PIC's com diferentes velocidades de clock, bastando alterar as primeiras três linhas de código (Figura 25).

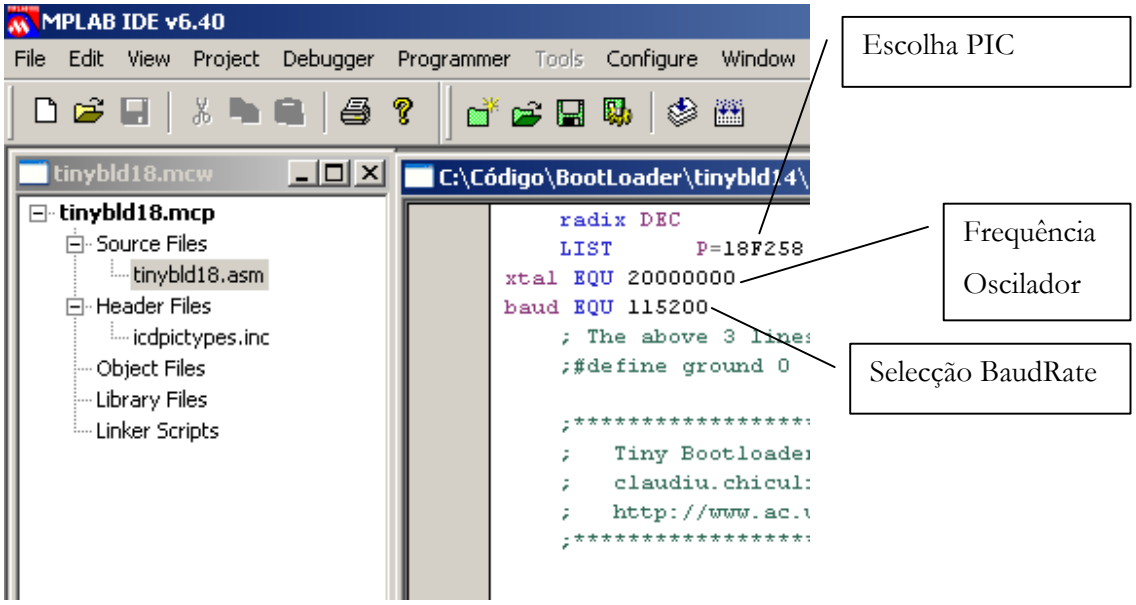


Figura 25 - Ilustração adaptação do Tiny bootloader [15]

Compilar o código com o MPLABC18[12] ou HT-PIC18[13] em ambiente MPLAB IDE[12] e realizar o seu download através do Tiny Bootloader (Figura 26).

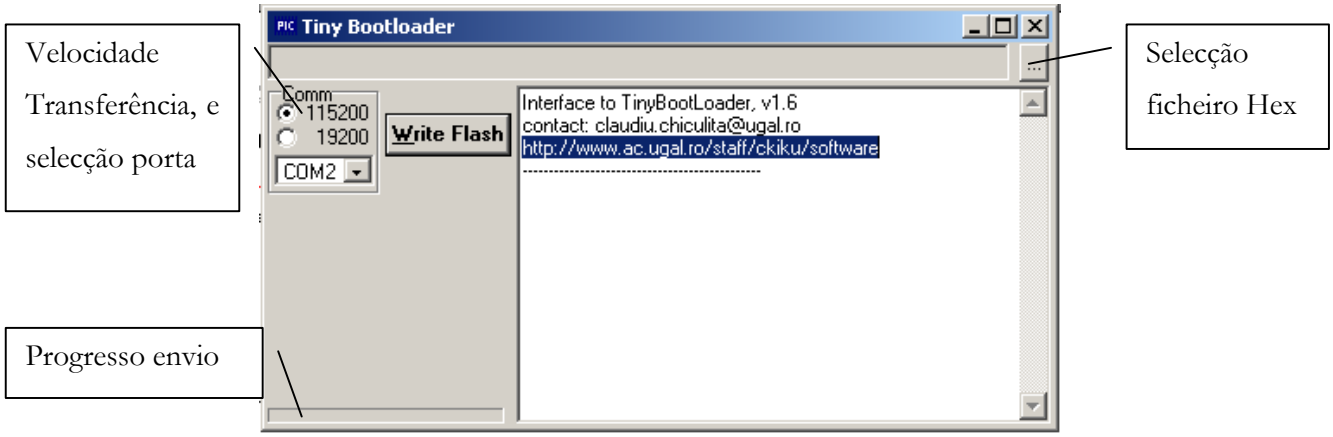


Figura 26 – Aparência do Software download ficheiro hex

5 Sistema de Comunicação

O sistema de comunicação tornou-se um requisito a ter em conta, como tal a selecção dos suportes de comunicação, para as diversas zonas de controlo.

5.1 Hierarquia do Controlo Distribuído

O controlo global é composto por três partes: o Controlador Principal, Mestre e os Escravos, como exemplificado na Figura 27.

A função do controlador Principal consiste em analisar os dados recebidos (valor dos diversos sensores) e usando um algoritmo apropriado calcular o valor a impor às diversas juntas. Este algoritmo deverá obedecer a regras dinâmicas e estáticas do sistema mecânico, componente a ser executada noutro projecto.

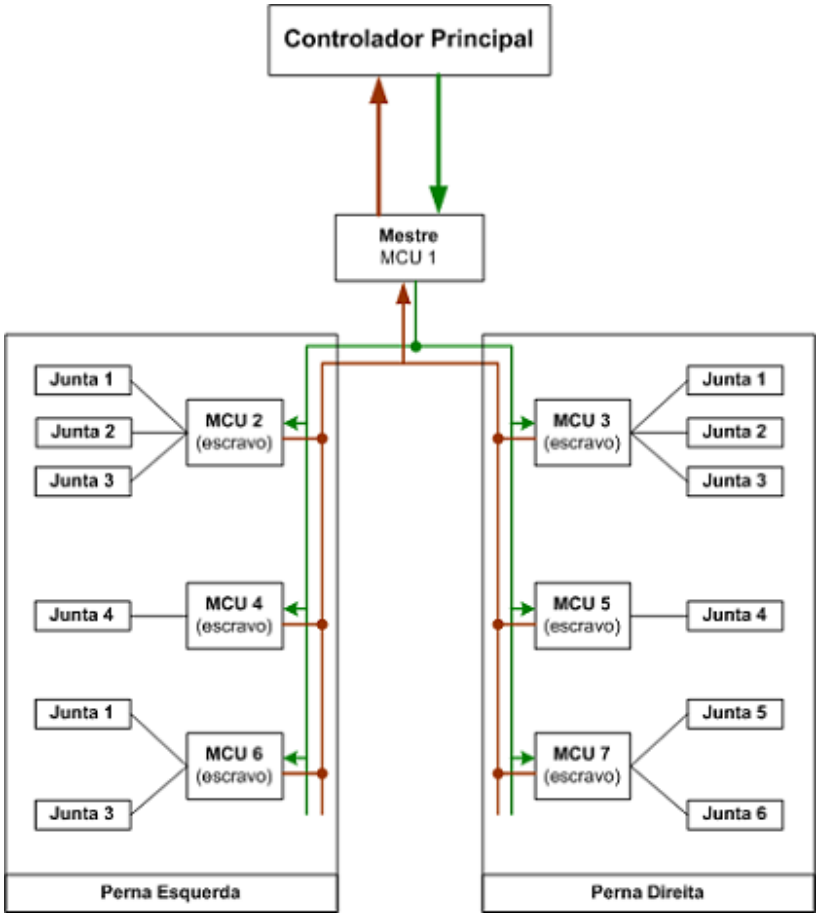


Figura 27 – Controlo distribuído

Olhando para a Figura 27, visualiza-se uma analogia entre este controlador e o cérebro humano.

O Mestre não terá necessidade de efectuar cálculos relativamente à configuração física do robô, o seu papel consistirá apenas em transmitir dados recebidos dos Escravos ao Controlador Principal, como do Controlador Principal aos diversos Escravos (encarrega-se igualmente da leitura dos sensores inerciais).

O sistema Mestre terá como objectivos:

- Montar uma estrutura de dados, composta pelas posições a impor a cada junta e pelos valores dos diversos sensores dessas mesmas juntas;
- Comunicar com os Escravos, transmitir novas posições e receber valor dos sensores;
- Comunicar com o Controlador Principal, receber novas posições e transmiti-las, tal como atender a pedidos de valores (sensores).

- Obtenção do valor dos sensores inerciais.

O Escravo (controlador local) será responsável pelo controlo dos actuadores e obtenção dos valores dos sensores de cada articulação. Este sistema será responsável por uma das zonas mais sensíveis do todo o controlo, devido à necessidade de uma grande eficiência na geração da onda de controlo dos servos motores.

Com base nas notas anteriores, verifica-se que o controlador Principal terá o papel de realizar cálculos, ou seja, aquele que pensa e reage. O Mestre apenas tem o papel de transmitir ao Escravo as ordens.

5.2 Sistema Mestre – Escravo

Esta é uma das etapas fundamentais deste projecto, pois o sistema modular só se torna realmente útil se for possível interligar os vários módulos. Para isto acontecer tem de se recorrer a formas de comunicação Multiposto (barramento).

O sistema de comunicação Mestre – Escravo deve obedecer a alguns princípios, entre as quais:

- Deve ser possível ter um código comum para todos os módulos;
- Admitir a comunicação entre módulos;
- Endereçamento dos vários módulos[1];
- Velocidade de comunicação (alta);
- Comprimento máximo da mensagem de dados[2].

As soluções existentes por defeito no microcontrolador seleccionado é o: I2C (**I**nter **I**ntegrated **C**ircuit) e CAN (**C**ontroller **A**rea **N**etwork).

5.2.1 I2C

O bus I2C é um protocolo virado, inicialmente para dispositivos situados pertos uns dos outros, mais precisamente no interior dos aparelhos electrónicos, neste estado o I2C é líder.

[1] Forma de diferenciar os diversos microcontroladores na rede.

[2] Comprimento máximo suficiente para enviar toda a informação de uma só vez. Sobrando assim mais tempo para outras comunicações ou outros reenvios

Ele interliga com facilidade, só recorrendo a 2 fios, os vários componentes do aparelho, não existindo um comprimento máximo de mensagem, possuindo uma velocidade máxima de transmissão de 3.4Mbits. O protocolo baseia-se no endereço e os masters só podem comunicar com um slave de cada vez.

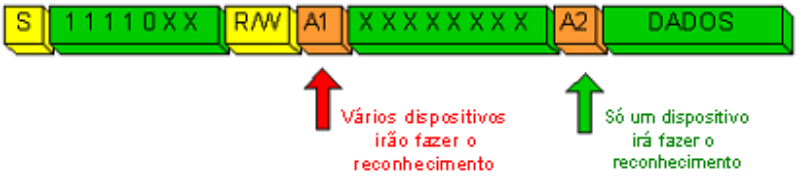


Figura 28 - Formato Standard Mensagem I2C

5.2.2 CAN

O CAN é um protocolo que nasceu no meio automóvel pela Bosch. É baseado na mensagem e não no endereço o que significa que as mensagens não são transmitidas entre nós com base no seu endereço. Na própria mensagem está incluída a prioridade e os dados a serem transmitidos, a uma velocidade máxima de 1Mbits. A mensagem é escutada por todos os nós na rede, cabe a cada nó realizar a análise da mensagem deixando-a ou não passar para futuro processamento. A mensagem pode ser enviada para um nó ou diversos dependendo da forma como a rede foi desenvolvida.

Outro benefício do protocolo baseado na mensagem é o facto de podermos adicionar outros nós à rede, sem haver necessidade de reprogramar todos os nós existentes. O novo nó adicionado com base na mensagem recebida decide se deve ou não processar a sua informação.

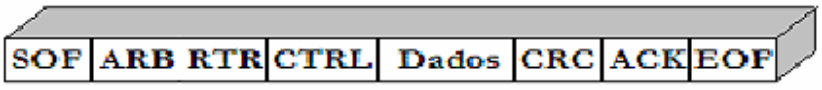


Figura 29 – Formato Standard Mensagem CAN

O comprimento da mensagem é fixo, sendo possivel enviar 8 bytes de informação no campo Dados (Figura 29).

A escolha recaiu sobre o CAN principalmente devido ao facto de este ser um protocolo baseado na mensagem. Desta forma, futuramente a construção da rede pode ser modificada para que os controladores locais possam ouvir a informação uns dos outros tomando decisões (poder local).

5.3 Sistema Controlo Principal – Mestre

A comunicação entre o Controlo Principal (CP) – Mestre poderia ser realizado através de vários suportes tais como:

- USART;
- USB;
- IrDA;
- BlueTooth.

O suporte USB é ideal para transmitir grandes quantidades de dados a grande velocidade, mas a sua implementação é muito difícil. Os outros dois (IrDA e Bluetooth) são tecnologias wireless e como tal não são protocolos tão fiáveis para este tipo sistemas de controlo como os de ligação física.

A interligação CP – Mestre é realizada recorrendo a uma ligação RS232 devido à sua fácil utilização e implementação. Além de permitir velocidades de transmissão aceitáveis e com pequena susceptibilidade a erros, desde que a distância entre extremos seja pequena.

6 Programas de Controlo

Tanto no PIC Mestre como no Escravo, o programa base de controlo é semelhante, consistindo em:

- Configuração dos TRIS, ADC, geração de PWM e *interrupts*;
- Configuração módulo CAN;
- Ciclo infinito com a leitura dos sensores.

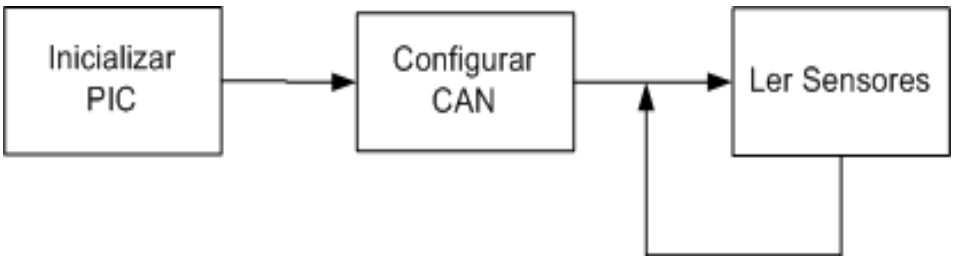


Figura 30 - Algoritmo Base do PIC

O algoritmo da Figura 30 funciona normalmente até ser interrompido por um *interrupt*, podendo este possuir uma prioridade alta ou baixa (como explicado posteriormente).

6.1 Configuração do PIC

Para que exista um eficiente funcionamento por parte das placas de controlo (Mestre/Escravo) deve-se proceder a correcta configuração dos pinos do PIC. Para que o compilador faça a modificação dos correctos registos deve-se incluir a livraria correcta.

6.1.1 Configuração dos I/O

Por análise do datasheet verifica-se que determinados pinos podem exercer mais do que uma função daí a necessidade uma configuração acertada. Primeiro deve-se indicar se determinado pino é uma entrada ou saída e apenas depois indicar qual a sua função.

A configuração dos pinos é realizada através dos registos TRISA, TRISB e TRISC os quais correspondem aos PORTA, PORTB e PORTC.

De seguida um exemplo duma configuração possível a colocar nos pinos

```
/*-------------------------------------------------------+
|                  Definicoes dos TRIS do PIC            |
+-------------------------------------------------------*/
//Porto A  <3:7> saídas excepto <0:2>
TRISA = 0b11111111;
//Porto B <2> entradas CanRx, <0:1><3:7> saídas
TRISB = 0b00001010;
//Porto C <0:6> saídas, RC7 entrada (RX USART)
TRISC = 0b10000000;
```

Figura 31 – Exemplo de configuração TRIS

6.1.2 Configuração da USART

A comunicação série RS232 (USART) é realizada a 9600 baud, com um 1 stop bit e 8 bits de dados, em modo assíncrono a alta velocidade (BRG = 1), para alterar a velocidade de transmissão de dados basta alterar valor de SPBRG o qual é dado pela seguinte equação

$SPBRG = \frac{Frequencia\,Oscilador/baud}{16} - 1$, em que a frequência do oscilador vem em MHz e o valor de baud encontra-se dentro dos valores da seguinte tabela.

BAUD RATE (Kbps)	20 MHz		SPBRG valor (decimal)
	KBAUD	% ERRO	
0.3	NA	-	-
1.2	NA	-	-
2.4	NA	-	-
9.6	9.62	+0.16	129
19.2	19.23	+0.16	64
76.8	78.13	+1.73	15
96	96.15	+0.16	12
300	312.50	+4.17	3
500	416.67	-16.67	2
HIGH	1250	-	0
LOW	4.88	-	255

Figura 32 - Tabela a relação entre SPBRG e o baud rate

No programa de controlo toda esta operação é realizada através da função SetUsart, a qual tem como parâmetros de entrada a Frequência do oscilador (MHz) e o baud rate a colocar na USART.

6.1.3 Configuração do CAN

O CAN é protocolo série que funciona em rede como tal é caracterizado pelo seu baudrate. Como referido anteriormente é um protocolo baseado na mensagem. Todos os nós escutam a mensagem enviada pelos outros cabe a cada um decidir se precisa ou não dessa informação. Realiza-se esta operação através de mascaras e filtros, como será exemplificado posteriormente.

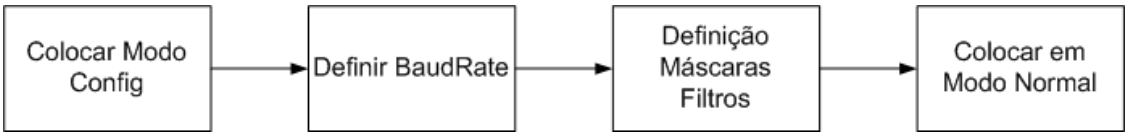


Figura 33 - Sequencia na configuração do CAN

6.1.3.1 Definição do Baudrate

Para definir o baud rate do CAN convêm entender como funciona a montagem de um bit, só assim poderemos fazer a sua definição.

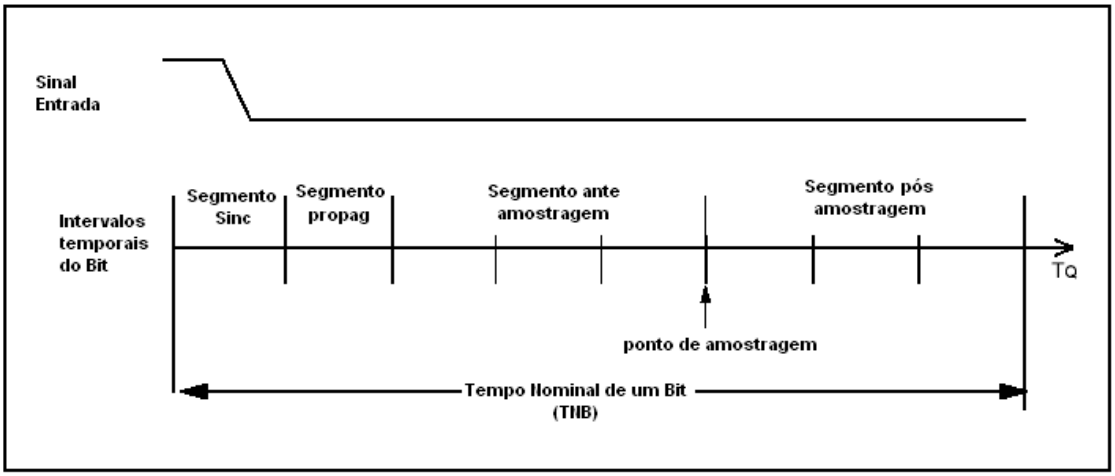


Figura 34 - Tempo Nominal de um bit

Normalmente o bit é formado por 4 partes;

- Segmento de sincronização, segmento responsável pela sincronização dos diversos nós.
- Segmento de propagação, segmento antes da estabilização do sinal.
- Segmento antes do ponto de amostragem
- Segmento após o ponto de amostragem

O ponto de amostragem é ponto ideal de leitura do bit. Todos os segmentos dependem de T_Q (tempo de quanta) o qual é ditado pela frequência do oscilador através da seguinte relação,

$$T_Q(\mu s) = \left(2(BRP+1)\right) \Big/ F_{osc}(MHz)$$

em que BRP pode tomar valores entre 0:63, este registo é apenas um factor de escala que permite que outros nós da rede possam funcionar com diferentes frequências de oscilação.

O tempo nominal de um bit é obtido com base na soma dos vários segmentos.

$$TNB = T_Q\left(SegSinc + Seg\,\mathrm{Pr}\,op + SegAm1 + SegAm2\right)$$

O segmento ante e pós ponto de amostragem podem tomar valores no intervalo [0, 7], tal como o segmento de propagação.

T_Q=0.01μs; para um oscilador de 20MHz com um BRP = 0;

Trazendo os seguintes valores para as seguintes fases;

SegSinc = 1; SegProp = 2; SegAm1 = 3; SegAm2 = 4 ⇒ 1Mbit

No entanto a montagem do bit no módulo CAN do PIC 18F258 necessita de outra fase denominada por fase de salto a qual se pode localizar na fase ante ou na pós ponto de amostragem (Figura 35, Figura 36) tendo efeitos diferentes no bit. Esta fase aparece devido a pequenas instabilidades nos cristais.

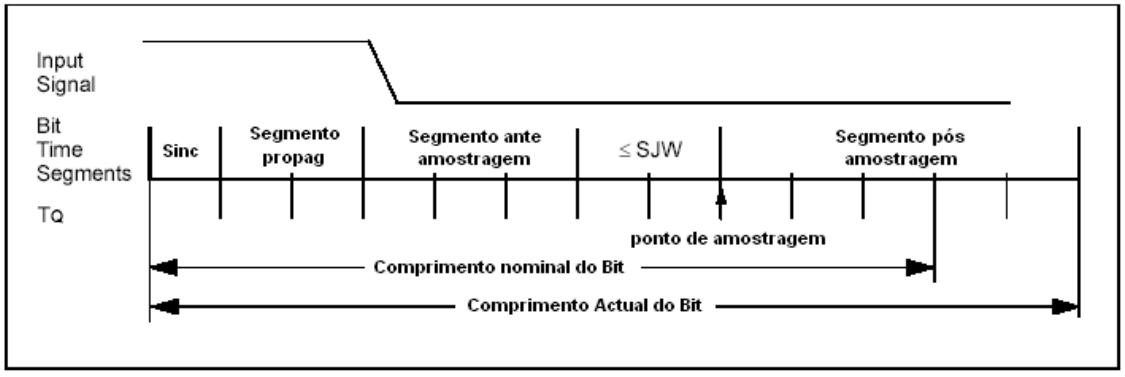


Figura 35 - Aumento do comprimento do bit

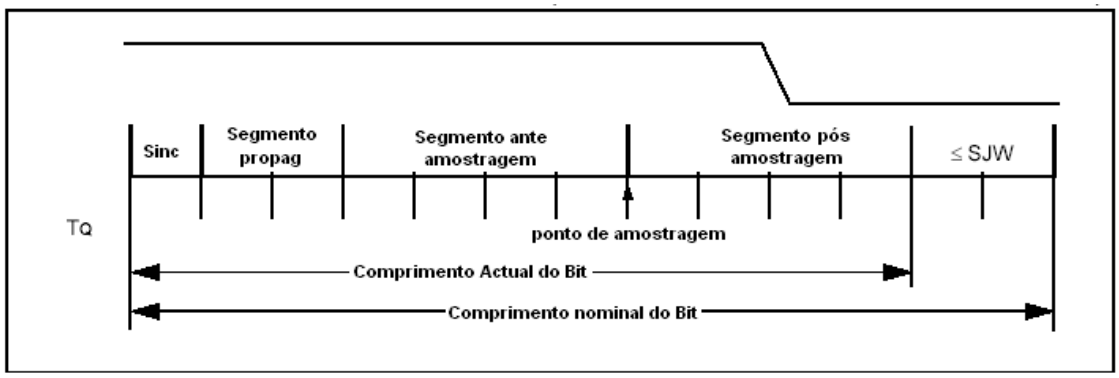


Figura 36 - Diminuição do comprimento do bit

O valor atribuído a este segmento depende de algumas regras. Como o oscilador em todos os nós funciona à mesma frequência e apresentam boa qualidade (não cerâmicos) o valor a dar ao registo SJW deverá ser 0.

Para realizar a definição do baudrate do CAN deverá ser modificadas as macros existentes na header file InitCan.h

```
//Macros relativas ao CAN, todos os valores sao obtidos tendo em conta um
//um baud rate de 1 Mbps
#define FASE_SALTO 1          //    Reajuste do comprimento do bit
#define FASE_PROP 2           //    inicio propagação bit
#define FASE_SEG1 3           //    Fase ante amostragem
#define FASE_SEG2 4           //    Fase após amostragem
#define BRP 1           //  Escala entre o oscilador e o baud
```

Estes são os valores desejados a colocar m cada registo do CAN. A rotina criada para definir o baudrate encarrega-se de adaptar os valores a cada registo, consistindo apenas em retirar uma unidade a todos estes valores.

6.1.3.2 Definição de máscaras e Filtros

As máscaras e filtros são os registos responsáveis por aceitar ou não as mensagens escritas no bus.

A mensagem é formada por um identificador de 11 bits o qual indica quem deve aceitar a mensagem. O módulo CAN possui um "assemblador" de mensagens, o qual monta todas as

mensagens escritas na rede, mas apenas deixa passar para os buffers de recepção as mensagens que cumprirem os filtros e as máscaras.

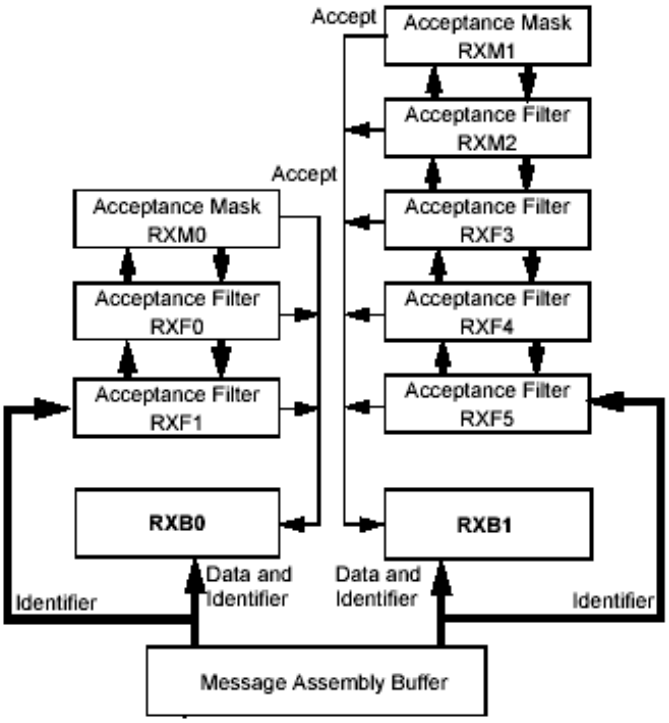


Figura 37 - Diagrama de filtragem do identificador

As máscaras definem quais os bits a serem usados pelos filtros. Sendo só a mensagem aceite se cumprirem pelo menos um filtro.

Bit Mascara	Bit filtro	Identificador	bit de aceitação ou rejeição
0	x	x	aceita-se
1	0	0	aceita-se
1	1	0	rejeita-se
1	0	1	rejeita-se
1	1	1	aceita-se

Exemplo: imagine-se o envio de uma mensagem com o identificador igual a 0b01011, e que teríamos um nó com a seguinte mascara 0b11110 e o filtro 0b11010, a mensagem não era aceite porque o bit 4 era diferente ao identificador.

Neste trabalho as máscaras possuem o valor 1 em todas as suas posições, como exemplificado no seguinte excerto do código, pois requer-se que sejam verificados todos os bits do identificador com o filtro, sendo só assim a mensagem aceite.

```
//Mascara autorizando a que o filtro verifique todos os bits do identificador
#define MASCARA 0x7FF
```

Os filtros terão o nome dado a cada microcontrolador na rede.

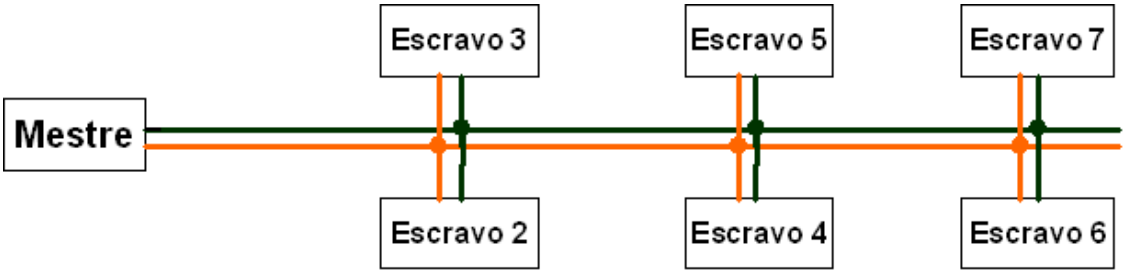


Figura 38 - Identificação de cada nó na rede

O PIC Mestre terá como identificador o número 1 e todos os outros PIC's os valores indicados na Figura 38.

6.1.4 Configuração da ADC

A correcta configuração da ADC é um elemento importante, para uma correcta obtenção do valor do potenciómetro dos servomotores e dos restantes sensores (força e inerciais).

Para se configurar a ADC é necessário configurar os valores de referência a que esta vai trabalhar. Estes poderão ser os valores de tensão de funcionamento do PIC, ou outros externos. Como estes valores ainda não estão definidos deixa-se em aberto a sua definição, prevê-se a sua localização no intervalo 0.85 V – 3.3 V.

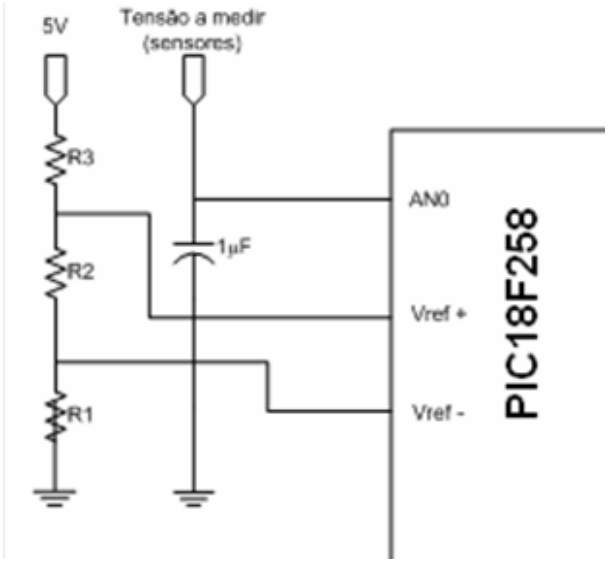


Figura 39 - Configuração da ADC

Por análise da Figura 39 verifica-se a existência duma ligação ao pino contendo V_{ref+} (RA3) e outra a V_{ref-} (RA2). Sendo o valor das tensões de referência dadas pelos seguintes valores

$$V_{ref+} = \frac{5(R1+R2)}{R1+R2+R3} \text{ e } V_{ref-} = \frac{5R1}{R1+R2+R3}$$

através do divisor resistivo da Figura 39.

Como a ADC do PIC possui uma resolução de 10bits, consegue-se mudar um bit da leitura apenas com a variação de tensão dada por $V_{ref+} - V_{ref-} / 2^{10}$.

A configuração da ADC faz-se com base nos registos ADCON0 e ADCON1.

Exemplo da configuração imposta neste trabalho:

```
ADCON0 = 0b10000001; //Velocidade conversão Fosc/32 com canal 0 de conversão e
                     //activa a conversão
ADCON1 = 0b01001111; //Vref+ RA3 Vref- RA2 e justificado à esquerda
```

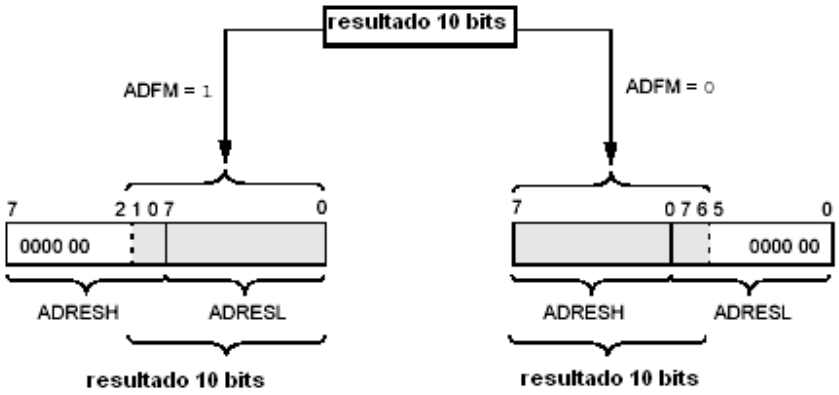


Figura 40 - Formação da leitura de 10 bits

6.1.5 Configuração do PWM

Para assegurar o controlo do servomotor ou o envio das mensagens para os diversos escravos recorreu-se ao módulo CCP1 contido no PIC. Este módulo funciona tanto em Capture, Compare ou PWM. Neste trabalho será utilizado como PWM. O módulo necessita do registo Timer2 para que possa funcionar nesta funcionalidade.

Escrevendo valores diferentes no registo *PR2*, alteramos o valor da frequência do PWM. O valor a colocar em *PR2* pode ser calculado através da seguinte fórmula

$$\mathrm{PR2} = \frac{\mathrm{F_{osc}}}{4 \times \mathrm{PWM_{Freq}} \times \mathrm{TMR2_{prescaler}}} - 1.$$

O valor do duty-cycle é escrito no registo CCPR1L e calculado através da seguinte fórmula

$$\mathrm{Duty\ Cycle} = \frac{\mathrm{F_{osc}}}{8 \times \mathrm{PWM_{Freq}} \times \mathrm{TMR2_{prescaler}}}.$$

Como neste trabalho o valor do duty-cycle é praticamente irrelevante, todos os cálculos são realizados pelo PIC. Na rotina InitPic a qual recebe como parâmetros de entrada a Frequência do oscilador (MHz), a frequência do PWM (KHz) a gerar e o baud rate da USART.

```
/*-------------------------------------------------------------------------------------------+
|                                Definicoes do TIMER2 e Gerador PWM                         |
+-------------------------------------------------------------------------------------------*/

//Calculo de PR2, para gerar o pwm, pag 130 datasheet
ValPR2 = (byte)(FreqOsc*1E3/(4.0*FreqAct*PresScalerTimer2))-1;
//valor a contar para 50% de duty cycle
Duty50 = (byte)(FreqOsc*1E3/(2.0*FreqAct*PresScalerTimer2));
//Parar a compilacao devido ao valor obtido para o PR2
#if ((ValPR2<0) || (ValPR2>256) || (Duty50<0 || Duty50>256))
      #error "valor de PR2 nao e aceitavel."
#endif

PR2 = ValPR2;


switch (PresScalerTimer2)
{
      case 1:{ T2CON = 0b00000100; break;}
      case 4:{ T2CON = 0b00000101;break;}
      case 16:{ T2CON = 0b00000110;break;}
}

//ocupar apenas a posicao dos bits menos significativos do pwm
CCP1CON = (Duty50 & 0x03) << 4;
//bits + significativos do dutycycle
CCPR1L = Duty50 >> 2;
```

6.1.6 Interrupts

Como indicado anteriormente este PIC permite dar prioridades diferentes aos *interrupts* existentes no PIC. Consegue-se realizar estas prioridades através dos registos IPR do PIC.

```
/*-------------------------------------------------------------------------------------+
|                                  Definicoes dos Interrupts                          |
+-------------------------------------------------------------------------------------*/

// Global interrupt enable bit (enables all unmasked interrupts)

//ligar o interrupt dos rising edges dos osciladores externos

INTCON2bits.INTEDG0 = true; //interrupt na ascens„o da onda 50Hz

INTCON2bits.INTEDG1 = true; //interrupt na ascens„o da onda


//Prioridades do Interrupts
INTCON3bits.INT1IP = true; //alta prioridade no interrupt
IPR1bits.RCIP = false;
IPR3bits.RXB0IP = false;
IPR3bits.RXB1IP = false;

RCONbits.IPEN = true; //Activa interrupts priorit rios

INTCONbits.GIEH = true;    //Interrupts prioritarios activos
INTCONbits.GIEL = true;    //Interrupts menos prioritarios activos

//Activa o interrupt, para o caso que chegue algo pela porta serie
PIE1bits.RCIE = true;

INTCONbits.INT0IE = true; //oscilador de 50Hz ligado a RB0

INTCON3bits.INT1IE = true; //oscilador da onda de resolucao ligada
```

Definindo *interrupts* prioritários é necessário localizar as rotinas de serviço aos *interrupts* em zonas especificas da memória.

```
// Rotina de Interrupt mais prioritaria
#pragma code InterruptVectorhigh = 0x08
void InterruptVectorHigh (void)
{
    ServoIsr(); //jump to interrupt routine
}

// Rotina de Interrupt de menor prioridade
#pragma code InterruptVectorlow = 0x18
void InterruptVectorlow(void)
{
    ComIsr(); //Salta para a rotina menos importante
}
```

6.2 Escravo

O controlador local (escravo) possui o algoritmo de controlo com mais responsabilidade comparativamente ao Mestre, visto que ele tem de controlar os actuadores que suportam toda a estrutura do robô. Devido à sua importância, dedicou-se um tempo significativo do projecto para o seu desenvolvimento.

Este algoritmo é divido e duas partes principais: controlo do servo e actualização posição.

6.2.1 Controlo Servo

Este PIC é responsável pelo controlo local em cada junta através da geração do PWM de controlo de cada servo motor. O algoritmo de controlo dos servos ocorre perante um *interrupt* de alta prioridade, garantindo a execução do controlo dos servos em primeiro lugar.

Como referido anteriormente, o controlo dos servos passa por gerar um PWM de frequência de 50Hz com um duty cycle entre 4% – 10%.

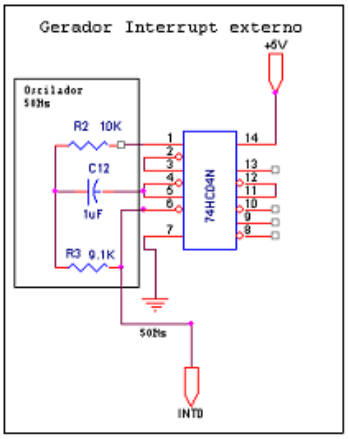


Figura 41 - Oscilador externo 50Hz

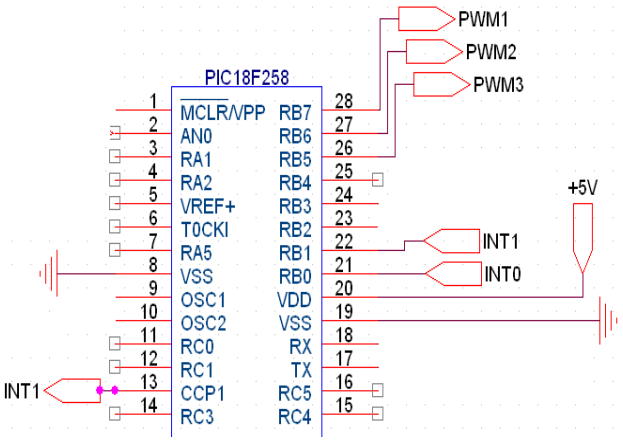


Figura 42 - Entrada das ondas de controlo

A solução passou por colocar um oscilador externo com uma frequência fixa (50 Hz), o qual irá estar conectado ao pino RB0. A cada passagem de 0 para 5V (Figura 41), o oscilador gera um *interrupt* que activa um oscilador de frequência fixa (80KHz). Esta frequência define a resolução do PWM (incremento mínimo).

Cada vez que existe um *interrupt* devido ao pino RB1 (Figura 42), incrementa-se o contador até que o número de contagens equivale ao tempo pretendido para esse PWM. A relação entre a contagem, o tempo do duty cycle do PWM e a frequência de resolução é: $Contagens = DutyCycle * Freq\,\mathrm{Re}\,s$

Duty(%)	Contagens	
4	64	MinPwm
10	160	MaxPwm

O valor de pwm1, pwm2 e pwm3 é dado por:

$$MinPwm + \frac{MaxPwm - MinPwm}{180} * Angulo$$

O controlo obedece ao seguinte algoritmo:

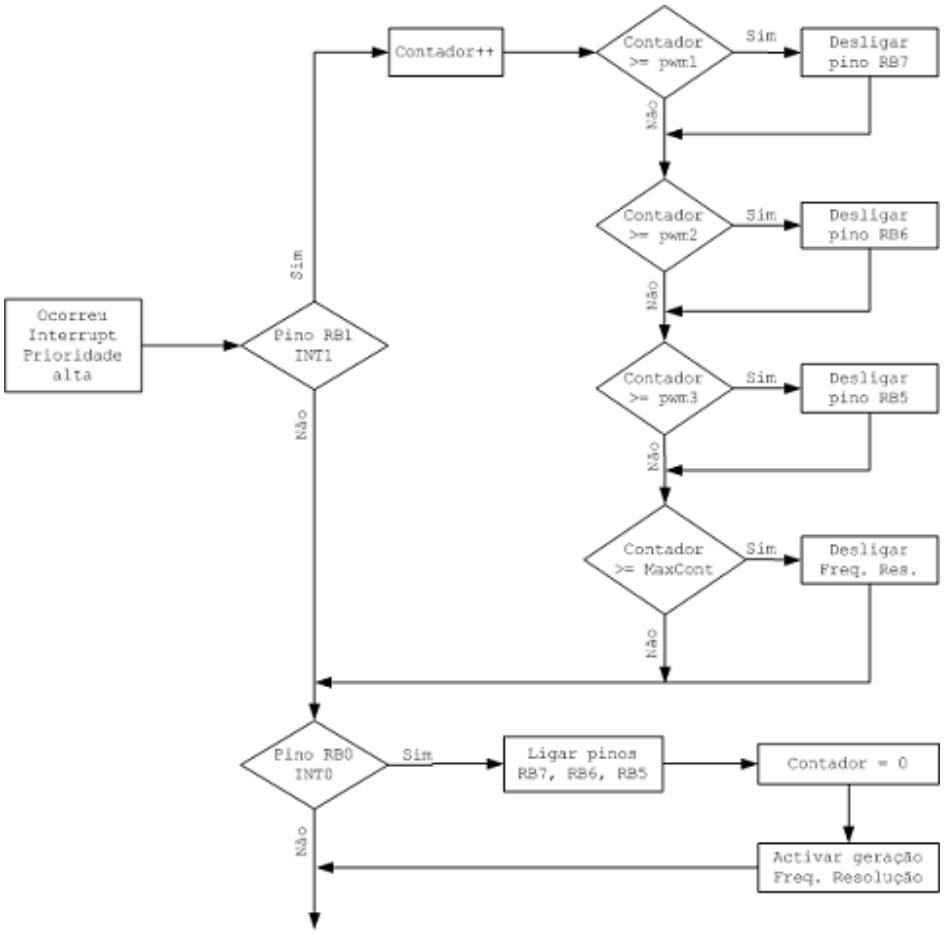


Figura 43 - Controlo Servo

6.2.2 Actualização Posição

A actualização da posição é realizada através de um *interrupt* (baixa prioridade) associado à chegada de uma mensagem pelo CAN. Novas posições dos servos (pwm1, pwm2 e pwm3) são adquiridas.

A mensagem enviada a cada Escravo através do CAN consiste em três bytes, simbolizando cada byte a nova posição de cada servo (junta).

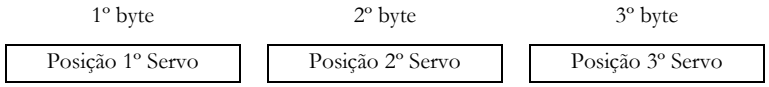


Figura 44 - Mensagem enviada CAN (Mestre)

Seguidamente é enviada uma mensagem com todos os valores dos sensores existentes.

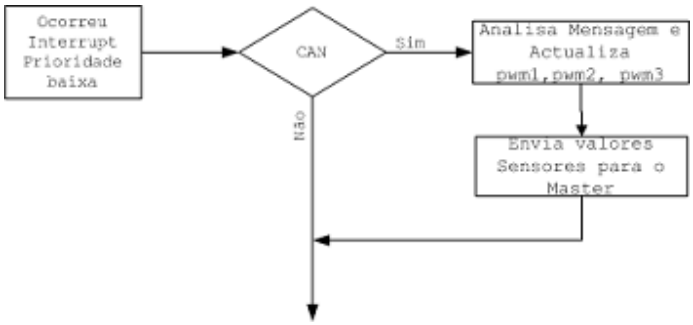


Figura 45 - Recepção e Envio CAN

6.3 Mestre

Como explicado anteriormente, este controlador apenas irá ser uma simples ponte entre os escravos e o controlador principal, tendo ao seu cargo a leitura dos sensores inerciais (acelerómetros e giroscópios).

O programa de controlo do Mestre diferencia-se pelas rotinas de *interrupt*. As rotinas existentes têm prioridades diferentes para o Controlo Principal e para os Escravos.

6.3.1 Controlo Principal (RS232)

O programa do Mestre recebe mensagens enviadas do Controlo Principal através duma comunicação RS232 que têm um comprimento fixo de 2 bytes.

O formato mínimo encontrado para a recepção de novas posições dos servos e pedidos de valores sensoriais.

1º byte							
a	**b**	**c**	**d**	**e**	**f**	**g**	**h**

2º byte							
x	x	x	x	x	x	x	x

O primeiro byte divide-se em três partes:

Tipo de mensagem				
Bits			Descrição	
A	**b**	**c**	Tipo	2º byte
0	0	1	Escrita	[0-180º], Nova posição
				0xFF, HomePosition todas as Juntas
0	1	0	Leitura	Sensores toda a junta
				0xFF, Sensores todo membro
0	1	1		Posição actual Servo
1	0	0		Valor sensores força pé

Membro		
d	0	Perna
	1	Braço
Lado		
e	0	Esquerdo
	1	Direito

Juntas			
Bits			Descrição
f	g	h	
0	0	0	Master
0	0	1	Anca (S1)
0	1	0	Anca (S2)
0	1	1	Anca (S3)
1	0	0	Joelho
1	0	1	Tornozelo (S1)
1	1	0	Tornozelo (S2)

Tabela 1 - Mensagens a enviar para o Mestre

O segundo contém o valor numérico relativo ao primeiro byte.

Exemplo:

Imagine-se a necessidade de actualizar a posição do tornozelo esquerdo par 45º então a mensagem a enviar seria:

1º byte							
0	0	1	0	0	1	0	1

2º byte							
0	0	1	0	1	1	0	1

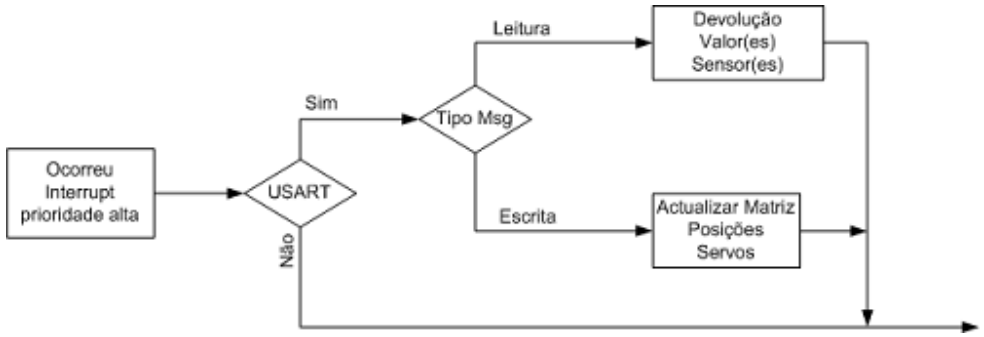
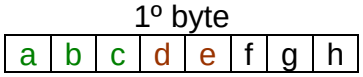


Figura 46 - Fluxo análise da mensagem recebida Controlo Principal

Visto existir pedidos de valores ao Master, houve necessidade de um formato para a mensagem. Esta mensagem consiste basicamente no primeiro byte, a qual informa quantos bytes serão precedidos.



Tipo de mensagem			
Bits			Descrição
a	**b**	**c**	
0	0	1	Segue-se um byte
1	0	0	segue-se quatro bytes

Tabela 2 - Mensagem enviada

Na Tabela 2 está explicado o significado dos primeiros três bits, sendo o significado dos restantes análogo aos bits da Tabela 1.

6.3.2 Escravos (CAN)

Enquanto a mensagem recebida (proveniente do Escravo) consiste num comprimento de 8 bytes, com o seguinte formato.

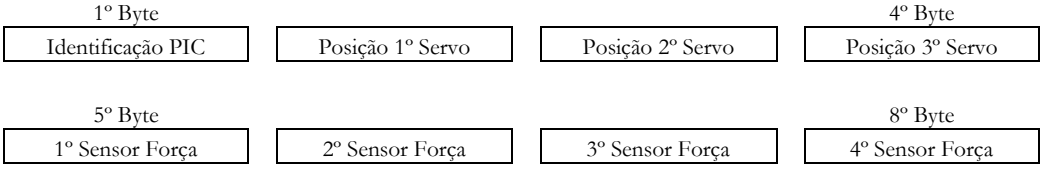


Figura 47 - Mensagem recebida CAN (Escravo)

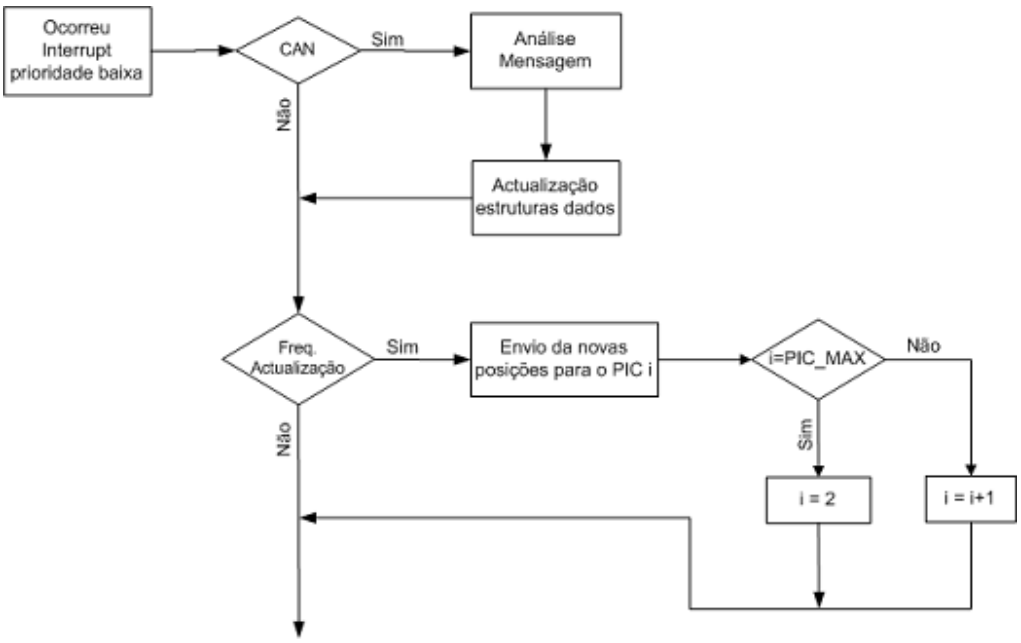


Figura 48 - Fluxo de análise mensagem recebida

7 Conclusão e Resultados

No trabalho desenvolvido conseguiu-se implementar um sistema modular, tanto do ponto de vista das comunicações como a nível sensorial e do controlo.

O suporte de comunicação CAN entre os diversos módulos foi implementado com sucesso. Conseguiu-se modificar as posições dos servomotores e também adquirir valores dos sensores. No entanto, as potencialidades do CAN não estão todas a ser utilizadas, estando a sua aplicação mais próximas do protocolo I2C, ou seja, existe apenas um nó "master" que comunica com um "slave" de cada vez. Também falta testar as placas finais de controlo onde estarão colocados os multiplexers.

Conseguisse implementar todos os sensores, mas falta produzir as placas dos pés com os sensores de força. A construção das placas para os pés requer precisão de maneira a que as quatro extremidades possam tocar no chão ao mesmo tempo. Uma análise deve ser igualmente feita ao nível das tensões de referencia. Estas são diferentes nas diferentes partes do corpo do robô e por isso, para o mesmo valor angular de uma junta no joelho e no tornozelo por exemplo, o valor da diferença de potencial irá ser diferente.

Este projecto, visto a sua amplitude e a sua complexidade, lançou-nos um verdadeiro desafio. Este desafio ainda está longe de estar acabado mas este trabalho também pretendeu dar algumas bases e alguns conhecimentos para os futuros alunos que, esperamos nós, irão continuar esta aventura, a procura do servo perfeito para um homem, uma criação a sua imagem. As dificuldades foram muitas mas as alegrias foram maiores.

8 Bibliografia

Livros

[1] PIC Microcontroller Project Book, John Iovine, MacGraw-Hill.

[2] PIC microcontrollers for beginners,too!, Nebojsa Matic, livro online, http://www.mikroelektronika.co.yu/english/product/books/PICbook/picbook.htm

[3] Linguagem C, Luís Damas, FCA

[4] K.Pazul, Controller Area Network (CAN) Basics, Application Note AN713,Microchip;

[5] M.Stanczyk, Smart Sensor CAN Node using the MCP2510 and PIC16F876, Application Note AN212;

[6] TJA1041, CAN High-Speed Transceiver, Phillips Semiconductors;

[7] Apontamentos Teóricos Informática Industrial, J.P.Santos, Universidade de Aveiro;

[8] Apontamentos Interface e periféricos, Barramentos e interfaces de Comunicação Série, J.A.Fonseca, Universidade de Aveiro;

[9] Apontamentos Instrumentação I, Extensómetros, Gustavo da Silva, Escola Superior de Tecnologia de Setúbal.

[10] Sensores e actuadores, J.R. Azinheira, Instituto Superior Técnico.

URls

[11] URL: Homepage do programador de PIC's, http://www.bobblick.com/techref/projects/projects.html

[12] URL: Homepage microchip® com muita informação e códigos exemplos, http://www.microchip.com

[13] URL: Homepage da Hi – Tech Software (compilador de C para PIC), http://www.htsoft.com/

[14] URL: Homepage do software IC-Prog, contém informações sobre todos os programadores suportados e respectivos esquemas, http://www.ic-prog.net/index1.htm

[15] URL: Homepage do senhor Claudiu Chiculita autor do tinybootloader, http://www.ac.ugal.ro/staff/ckiku/software

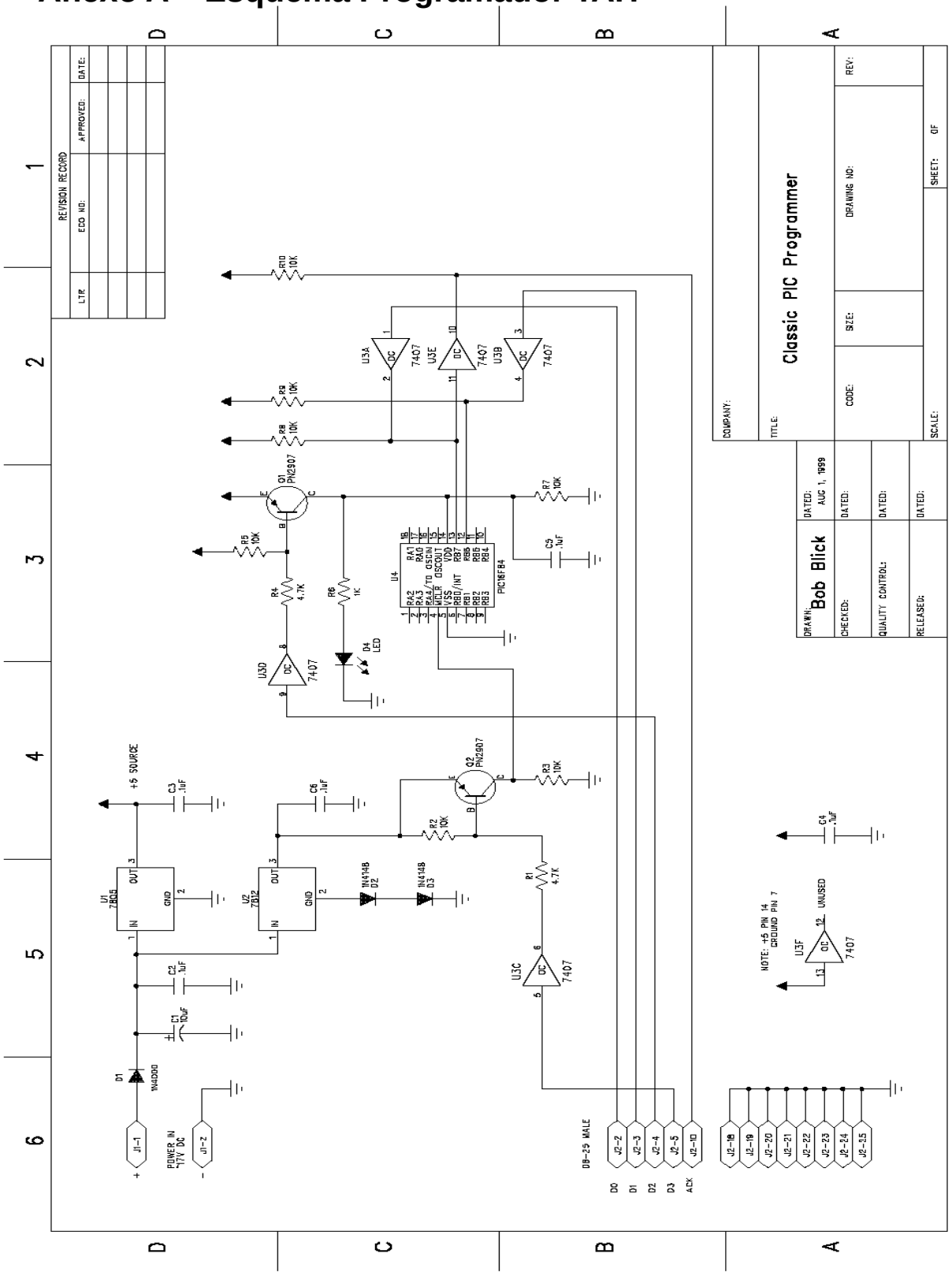
[16] URL: Homepage pagina servos Futaba

[17] URL: Homepage pagina servos Hitec

[18] Pagina com informação útil sobre servomotores, http://www.seattlerobotics.org/guide/servohack.html

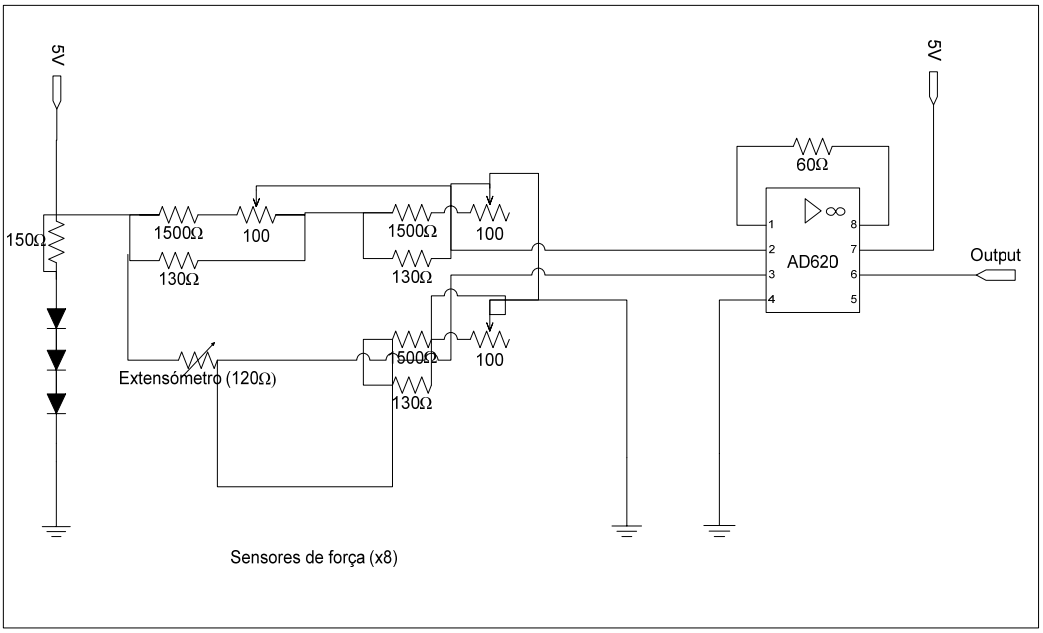
[19] Página da GlogalSpec, "Acceleration and Vibration Sensing", autor desconhecido.

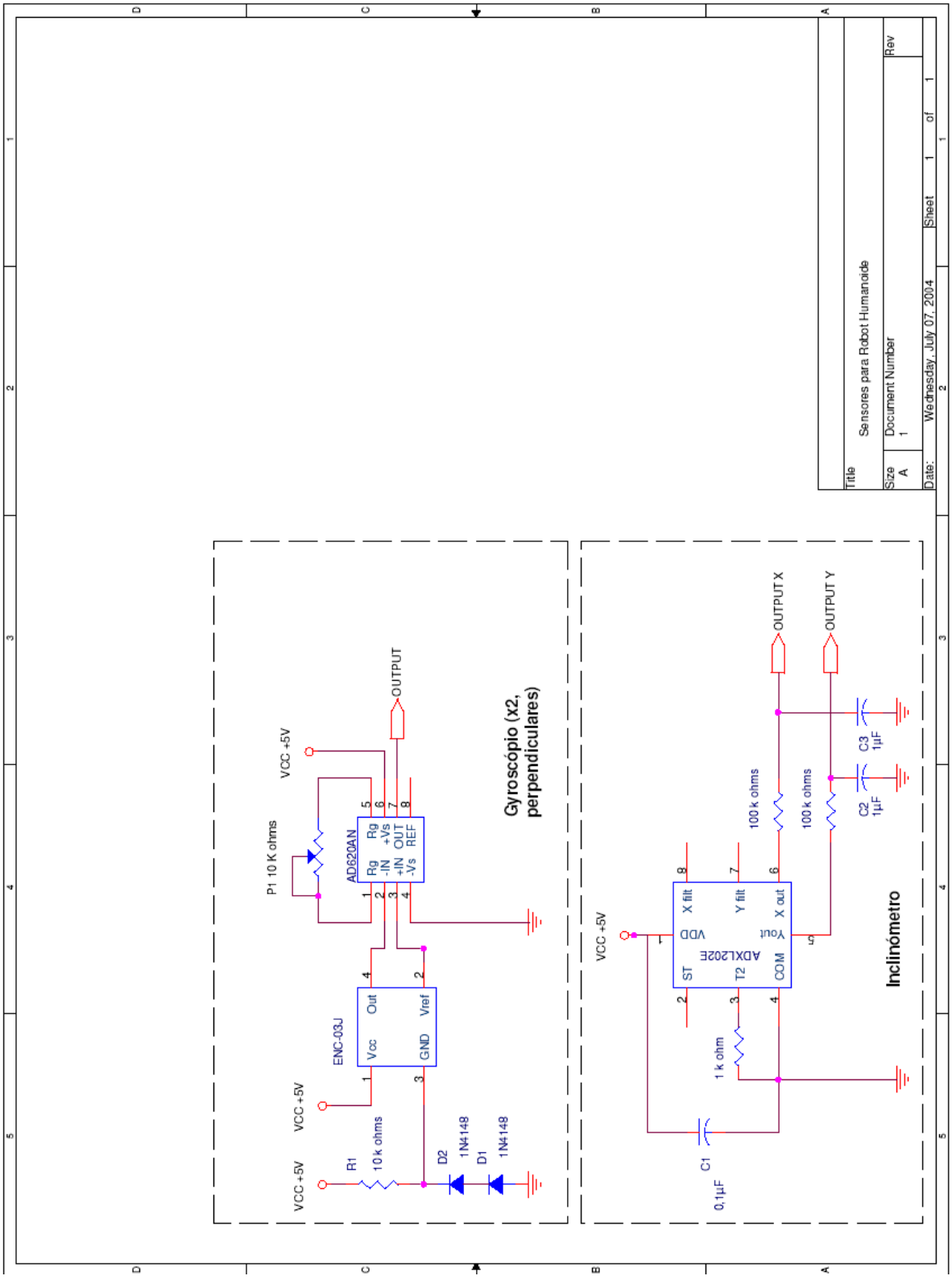
Anexo A – Esquema Programador TAIT



Anexo B – Placa de Controlo

Anexo C – Circuitos dos Sensores




Title
Sensores para Robot Humanoide
Size
A
Document Number
1
Rev
Date:
Wednesday, July 07, 2004
Sheet
1
of
1
Gyroscópio (x2, perpendiculares)
VCC +5V
P1 10 K ohms
AD620AN
Rg
-IN
+IN
-Vs
+Vs
OUT
REF
OUTPUT
ENC-03J
Vcc
Out
GND
Vref
R1
10 k ohms
D2
1N4148
D1
1N4148
Inclinómetro
ADXL202E
ST
X filt
Y filt
VDD
X out
Y out
COM
T2
100 k ohms
1 k ohm
C1
0,1µF
C2
1µF
C3
1µF
OUTPUT X
OUTPUT Y