EPS
Escola Politècnica Superior
UdG

Projecte/Treball Fi de Carrera

Estudi: Enginyeria Tècn. Ind. Electrònica Ind. Pla 2002

Títol: Disseny i construcció d'un braç robot de 5 graus de llibertat governat des de PC via USB.

Document: 1. Memòria

Alumne: Gerard Massaguer Frigolé

Director/Tutor: Antoni Martorano Gomis
Departament: Arquitectura i Tecnologia de Computadors
Àrea: ATC

Convocatòria (mes/any): juliol/2008

ÍNDEX

1 INTRODUCCIÓ

1.1 Antecedents

La humanitat ha construït màquines que imiten parts del cos humà des de fa molts segles. Ja a l'antiga Grècia i Egipte, es van construir estàtues amb braços articulats, que es movien mitjançant politges o hidràulicament.

Però no és fins al principi del segle XX quan apareix el concepte de robot, i fins gairebé la segona meitat d'aquest mateix segle que no sorgeixen els primers robots o autòmats programats per a realitzar tasques.

Aquests primers robots van sorgir gràcies a l'aparició dels circuits integrats, i van ser utilitzats en processos industrials per els fabricants d'automòbils. Aquests primers robots eren manipuladors que podien memoritzar moviments repetitius, assistits per sensors interns, que els ajudaven a fer els moviments amb més precisió.

La segona generació d'aquests robots incorporaven sensors externs, generalment els proporcionaven vista i tacte, que van permetre a aquests autòmats d'interactuar i rebre informació del món exterior.

L'evolució de la Intel·ligència Artificial, ha permès que els nous models de robots puguin rebre molta més informació de més fonts, interpretar-la i actuar en conseqüència.

El braç robot es va crear com a resposta a una necessitat de fabricació d'elements mitjançant la producció en cadena. Aquest tipus de fabricació es basa en que un sol producte és elaborat a partir de petites tasques realitzades independentment unes d'altres i consecutivament. Aquestes tasques són repetitives i moltes requereixen precisió, per la qual cosa es va veure la necessitat de substituir les persones que realitzaven aquestes tasques per màquines. És doncs, en l'àmbit industrial on van sorgir els braços robots, i és aquí on han evolucionat i s'ha expandit. Aquests tipus de braços realitzen doncs feines repetitives, per les quals han estat programats prèviament.

Creiem però, que existeixen altres tipus de tasques, les quals no són repetitives, ni poden ésser programades, que necessiten ser controlades en tot moment per un ésser humà, ja que actualment el nivell d'AI que es pot implementar en els dispositius, no permet una presa de decisions ràpida i correcta en situacions de risc.

Aquestes aplicacions, algunes de les quals fins fa poc dies no es duien a terme, poden ser operacions quirúrgiques, manipulat de substàncies perilloses a distància, desactivat de bombes, realització de tasques en l'àmbit aeroespacial, treballs en ambients agressius, etc.

Són activitats que han de ser realitzades per un ésser humà, però que requereixen precisió, és per això que es creu necessari el disseny d'un prototipus de braç robot estàndard, que permeti a una persona el control total sobre aquest en temps real per a la realització d'una tasca no repetitiva i no programable prèviament.

1.2 Objecte

Pretenem, en el present projecte, dissenyar i construir un braç robot de 5 graus de llibertat, controlat des d'un PC mitjançant un microcontrolador PIC amb comunicació a través d'un bus USB. El robot serà governat des d'un PC a través d'un software de control específic.

1.3 Especificacions i abast

Es dissenyarà i construirà un braç robot de 5 articulacions. Aquestes es trobaran a la base, un braç format per dues articulacions, i un capçal-eina que incorpora una pinça, la qual es pot canviar per qualsevol element segons quin sigui el seu camp d'aplicació. Aquest capçal tindrà 2 graus de llibertat. El moviment de les articulacions es realitzaran mitjançant servomotors. Es dissenyarà també el sistema de control, que serà una placa de control formada per dues fonts commutades que alimentaran els servomotors i un sistema de control i comunicació basat en un microcontrolador PIC. Aquest microcontrolador és el que permetrà la comunicació per tal de poder rebre i executar les ordres donades a través del PC, mitjançant un bus de comunicació USB.

2 BRAÇ ROBOT

2.1 Descripció del braç

El braç robot basarà el seu moviment en servomotors. Aquest tipus de motors permeten tenir un parell molt més elevat que l'aconseguit amb motors de corrent contínua convencionals. Això ens permetrà tenir moviments més ràpids i ens permetrà suportar càrregues majors al braç. A més els servomotors permeten un moviment molt precís que ens permetrà posicionar i desplaçar el braç amb unes distàncies de treball mínimes molt petites.

El braç tindrà 5 graus de llibertat, cada grau de llibertat estarà governat per un servomotor. Les articulacions es trobaran a la base, a l'avantbraç, en el colze, i finalment l'eina, que constarà de 2 graus de llibertat.

Les articulacions de la base, l'avantbraç i del colze, al ser les que han de suportar més pes, estaran governades per servomotors amb un parell més elevat.

L'estructura del braç robot serà de metacrilat de 3 mm de gruix tintat. Aquest material és molt resistent, lleuger i fàcil de mecanitzar en fred. A més permet crear peces de varis costats mitjançant el procés de termoformat, consistent en escalfar la placa de metacrilat i donar-li la forma o l'angle de curvatura desitjats. Això permetrà crear fàcilment els encaixos, suports i forats necessaris per poder-hi acoblar els servomotors i els diferents suports mitjançant cargols. Cada articulació la formaran dues plaques en mig de les quals hi anirà collat el servomotor.

Els servomotors es col·locaran en la posició més adequada gràcies a un joc de suports que incorpora el mateix motor i que permet posicionar-lo en diferents angles. També s'adequarà el moviment de sortida de l'eix del servomotor per a que pugui ésser aprofitat en els seus dos extrems i permetre una millor fixació a l'articulació a moure. La fixació dels servomotors i dels diferents elements del braç es farà amb cargols M3 i amb femelles autoblocants amb fre de nylon per evitar que s'afluixin amb el moviment i les vibracions. A més s'utilitzaran maniguets de goma a cadascun dels 4 punts de fixació de què disposa cada servomotor.

La figura següent ens mostra una visió general de com està format el braç robot.

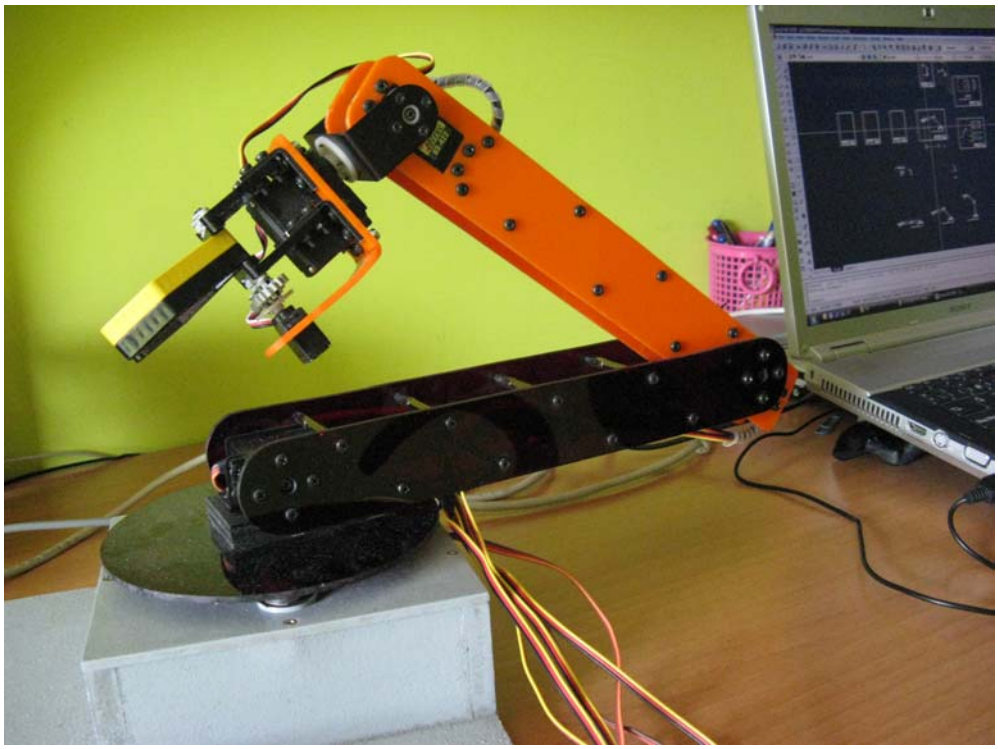

Figura 1. Braç robot

Les parts que formen el nostre braç robot són la base, l'avantbraç, el braç i la pinça. En la figura 2 trobarem senyalitzada cadascuna d'aquestes parts.

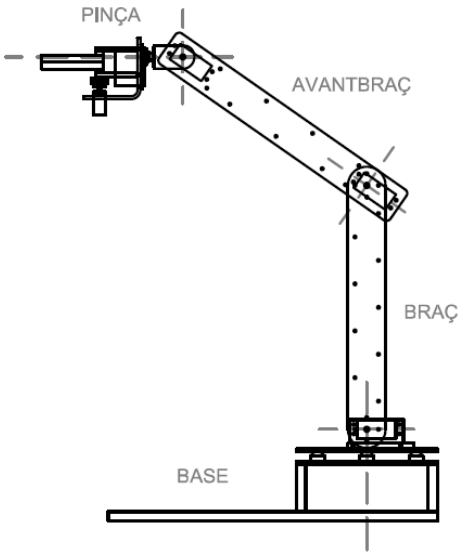

Figura 2. Parts del braç

Les articulacions, cadascuna d'elles formades per un servomotor, han estat anomenades basant-se en el braç humà. Aquestes són la base, l'espatlla, el colze i el canell format per dos servomotors situats en eixos perpendiculars. La següent figura ens mostra aquestes parts.

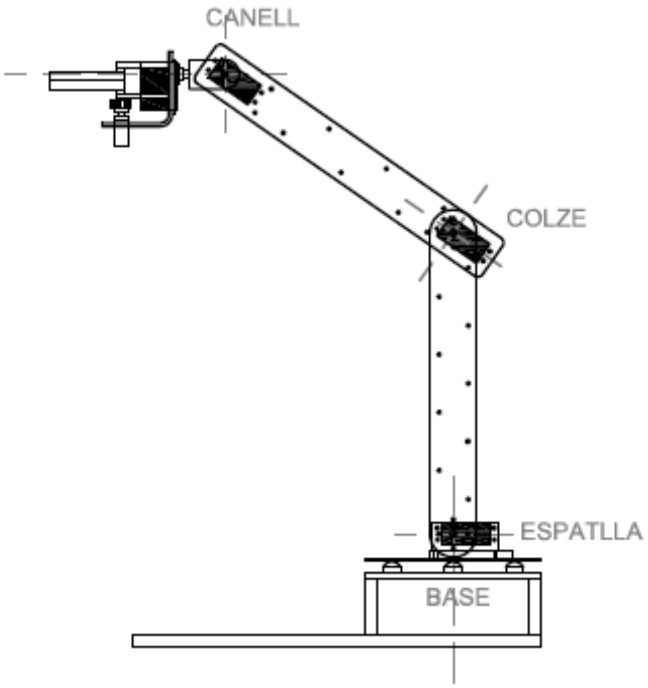

Figura 3. Articulacions

A més la pinça incorpora també un petit servomotor que és el que permetrà el moviment d'obrir i tancar i que no s'ha comptabilitzat amb els graus de llibertat del braç.

2.2 Estructura

2.2.1 BASE

El material de la base i del suport per a la base giratòria serà taula de DM hidròfuga. Tindrà un gruix de 10mm i densitat aproximada 760 kg/m³.

Les parts que formen el suport quadrat per a la base giratòria s'uniran a la base i entre elles mitjançant bisos 2,5x20mm reforçant les testes amb cola blanca. A la part interior del suport es col·locaran 4 angles de 90º per reforçar la unió amb la base.

La tapa del suport serà de DM de 4mm de gruix i disposarà de l'encaix necessari per a col·locar el servomotor de la primera articulació. També disposarà de quatre suports de plàstic que suportaran la base circular giratòria de metacrilat, que en permetran el gir i alhora ajudaran al servomotor de la base a suportar el pes del braç.

La base giratòria es subjecta a la resta de l'estructura de la base a través del servomotor. Aquest disposa d'un accessori consistent en una platina de plàstic circular. Aquest platina es subjecta al servomotor amb un cargol i la subjectarem a la base giratòria mitjançant bisos. Aquesta també disposa d'una zona elevada on es col·loca el suport metàl·lic per al servomotor de l'espatlla i que permet posicionar aquest horitzontalment.

La base rectangular disposarà de quatre potes de goma autoadhesives com a suport, però podrà subjectar-se mitjançant cargols a la superfície de treball, per aconseguir una major estabilitat a l'hora de treballar o podrà ser de dimensions més grans, convertint-se així en la zona de treball.

2.2.2 BRAÇ ROBOT

El material de què està format l'estructura del braç robot és el polimetilmetracrilat. Més conegut comercialment com a metacrilat. En la indústria es subministra en grànuls o làmines.

Competeix en quant a aplicacions amb altres plàstics com són el policarbonat (PC) o poliestirè (PS), però el metacrilat destaca respecte aquests altres plàstics transparents en molts aspectes. Té una transparència del 92%, essent el plàstic més transparent; té una resistència a l'impacte entre 10 i 20 vegades superior al vidre; és resistent a la intempèrie i als rajos ultraviolats, no presenta envelliment apreciable en 10 anys d'exposició exterior; és un excel·lent aïllant tèrmic; té una densitat lleugerament superior a la de l'aigua i la meitat que el vidre; de duresa semblant a l'alumini, és fàcilment ratllable; no produeix gasos tòxics en la combustió i té gran facilitat per a ser mecanitzat o motllejat. No obstant, si el braç robot s'utilitza per al manipulat de substàncies químiques, caldrà tenir en compte els productes que poden atacar aquest material com són: acetat d'etil, acetona, àcid acètic glacial, àcid sulfúric bicromàtic, alcohol amílic, benzol, butanol, diclorometà, triclorometà (cloroform) i toluè.

Per aquestes qualitats és molt utilitzat en la indústria de l'automòbil, il·luminació, cosmètics, espectacles, construcció i òptica entre molts altres.

Distingirem que el metacrilat és el nom més comú per a aquest material quan és subministrat en làmines o planxes.

Es comercialitza en planxes rectangulars d'entre 2 i 120mm de gruix. Existeix en varis graus de resistència, unes 12 qualitats, i en nombrosos colors que obren un món de possibilitats per al seu ús en arquitectura, decoració y enginyeria.

Per al disseny del braç robot utilitzarem planxes de metacrilat d'extrusió de 3 mm de gruix. S'utilitzarà metacrilat translúcid tintat de granate per a les peces de l'avantbraç i la part giratòria de la base. Per a l'estructura del braç i el suport de la pinça utilitzarem metacrilat opac de color taronja.

A la taula 1 tenim algunes de les característiques d'aquest material segons el fabricant.

Propietats físiques		
Densitat	1,2	gr/cm³
Propietats mecàniques		
Resistència a la tracció fins a la deformació	No aplicable	MPa
Resistència a la tracció fins al trencament	83	MPa
Allargament fins al trencament	5	%
Resistència a la flexió	120	MPa
Duresa a la pressió de la bola	185	MPa
Propietats tèrmiques		
Temperatura màx. en utilització	80	ºC
Temperatura de reblaniment VICAT (10N)	116	ºC
Temperatura de reblaniment VICAT (5N)	107	ºC

Taula 1. Característiques del metacrilat d'extrusió

Les peces de metacrilat que formen el braç estaran unides mitjançant separadors metàl·lics hexagonals de 50mm de llargada. Els servomotors estaran subjectats a la peca davantera mitjançant un accessori que transforma l'eix de sortida en una platina circular. La peça davantera tindrà també un forat que permetrà col·locar i extreure el cargol que subjecta la platina a l'eix del servomotor. La peça posterior disposarà d'un forat més petit que la subjectarà a l'extensor de l'eix posterior del servomotor i alhora li permetrà el moviment.

Les peces que formen l'avantbraç tindran un encaix rectangular que permetrà que els dos servomotors sobresurtin. Aquests dos servomotors que estan subjectats al braç, el de l'articulació del colze i un de la del canell, només ho faran a la peça davantera mitjançant els quatre punts de subjecció de què disposa la carcassa del servomotor. La peça davantera tindrà aquest encaix, però només per permetre que l'extensor de l'eix posterior de cada servomotor sobresurti per a poder ser subjectat amb la resta de peces que formen en braç. La peça posterior estarà subjectada amb la davantera mitjançant separadors hexagonals metàl·lics de 20mm.

2.2.3 PINÇA

La pinça estarà formada per un suport també de metacrilat, de les mateixes característiques que les peces que forment l'avantbraç. Tindrà una forma de L vist en alçat i disposarà dels encaixos necessaris per a subjectar un dels servomotors del canell, la pinça i el servomotor d'aquesta.

Els braços robots existents en el mercat, tenen una mà o capçal-eina que disposa de com a màxim 3 graus de llibertat, anomenats yaw, pitch i roll i que en la següent figura veiem a quins eixos pertanyen i quin moviment permeten al capçal.

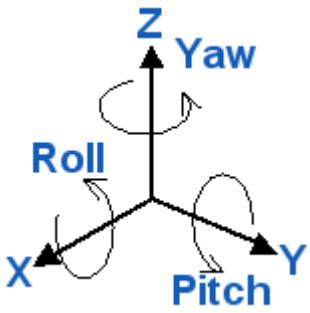

Figura 4. Moviments de la mà d'un braç robot

La pinça disposarà dels moviments pitch i roll. El moviment pitch el farà el servomotor que es troba a l'avantbraç, a través d'una platina metàl·lica en forma de U. Aquesta platina s'unirà al servomotor que es troba al suport de metacrilat de la pinça, i aquest permetrà el moviment roll.

La pinça estarà formada per dos dits de plàstic en forma de L a 45º. Estaran subjectats al suport per 4 punts, dos de superiors i dos d'inferiors i cadascun disposarà d'un eix de gir que s'allargarà per la part inferior dels dits i on es subjectaran dos engranatges idèntics. Un d'aquests engranatges és el que estarà unit al servomotor que permetrà el moviment d'obrir i tancar de la pinça.

Els dits que forment la pinça tindran adherida a la part interior espuma antilliscant, que permetrà al servomotor fer més força per a subjectar els objectes sense que aquest pateixi, i alhora evitarà que els objectes caiguin de la pinça.

A la figura 5 tenim una vista general de la pinça.

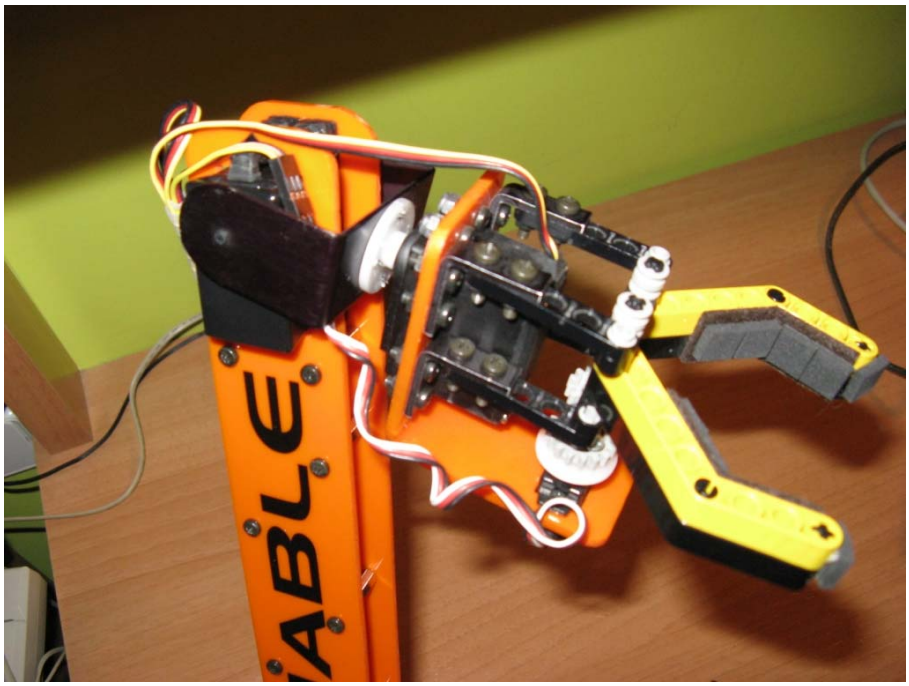

Figura 5. Pinça

2.3 Servomotors

2.3.1 DESCRIPCIÓ

El servomotor és un petit però potent dispositiu que disposa en el seu interior d'un petit motor de corrent contínua amb un reductor de velocitat i multiplicador de força, també disposa d'un petit circuit que governa el sistema. El recorregut de l'eix de sortida és de 180º en la majoria d'ells, però pot ser fàcilment modificat per a tenir un recorregut lliure de 360º i actuar així com un motor convencional.

Figura 6. Parts d'un servomotor

El control de posició l'efectua el servo internament mitjançant un potenciòmetre que va connectat mecànicament a l'eix de sortida i controla un PWM intern, que mitjançant un sistema de regulació modifica la posició de l'eix de sortida fins que aquesta coincideix amb la posició indicada. En aquesta posició el motor del servo deixa de consumir corrent i tan sols circula un petit corrent fins al circuit intern. Si forcem el servo (movent l'eix de sortida amb la mà) en aquest moment el regulador de posició intern ho detecta i envia el corrent necessari al motor per a corregir la posició.

Quan un servomotor ha de suportar una càrrega superior a les seves prestacions aquest simplement no pot moure la càrrega, intenta assolir la posició i al no aconseguir-la, el circuit que controla el servo demana més intensitat a la alimentació fins a assolir el parell màxim, el que ens proporciona seguretat en el dispositiu, ja que només augmenta la intensitat que consumeix fins que assoleix el parell màxim, un cop en aquest punt com que internament no pot aconseguir un parell major, no necessita tampoc un corrent major.

Per a controlar un servo s'ha d'aplicar un pol, de durada i freqüència específics, PWM. Tots els servos disposen de tres cables: dos per a alimentació Vcc i Gnd (vermell i negre respectivament) i altre cable (groc, blanc o taronja) per a aplicar el tren de polsos de control que indiquen al servo la posició que ha d'assolir.

En el diagrama de blocs de funcionament del servomotor de la figura 7 es pot veure el cicle i el bucle de realimentació que actua comparant la consigna de control amb la informació de posició del servo. La diferència la rep el driver que actua en conseqüència sobre el motor.

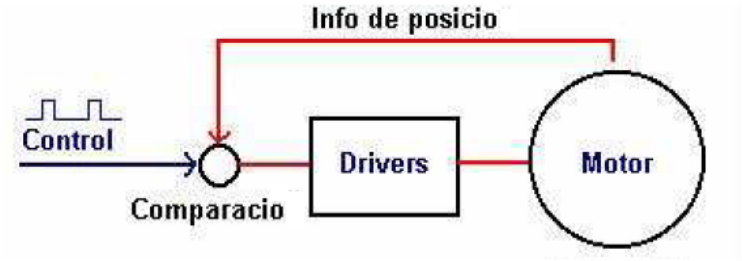

Figura 7. Diagrama de blocs

La figura 8 mostra els mecanismes integrats en el servomotor. El conjunt d'engranatges que efectua una reducció de la velocitat a l'eix i en conseqüència una augment del parell, que és el es busca en aquests tipus de motors.

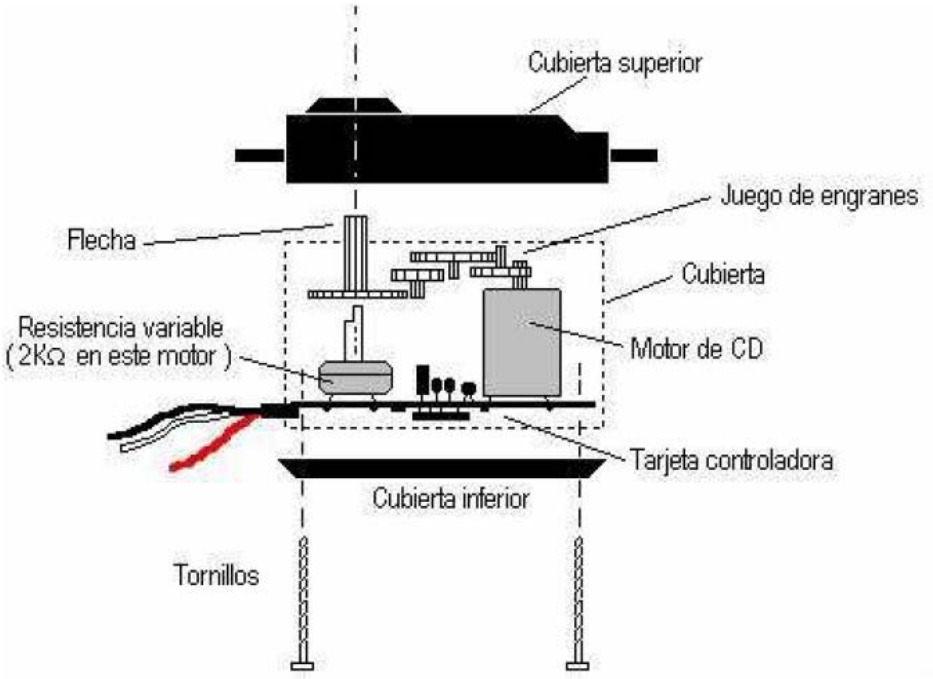

Figura 8. Especejament del servomotor

El control per amplada de polsos PWM, consisteix en enviar un tren de polsos, cadascun dels quals dura 20 milisegons. En aquest temps el senyal que s'envia està un temps en estat alt i la resta fins arribar als 20ms està en estat baix. La durada d'aquest temps en estat alt és el que determina la posició que el circuit de control detecta que s'ha de posicionar el servomotor. Mentre el PWM s'apliqui a un servomotor aquest mantindrà la posició que rep, quan deixi de rebre'l el servomotor restarà en repòs i no farà força.

En la taula 2 estan indicats els valors de control per amplada de polsos de diverses marques que comercialitzen servomotors:

Fabricant	Duració del pols [ms]			Freq. [Hz]
	Mínima (0º)	Neutral (90º)	Màxima (180º)	
Futaba	0.9	1.5	2.1	50
Hitech	0.9	1.5	2.1	50
Graupner/Jr	0.8	1.5	2.2	50
Multiplex	1.05	1.6	2.15	40
Robbe	0.65	1.3	1.95	50
Simprop	1.2	1.7	2.2	50

Taula 2. Posició a partir de la duració del pols

A la figura 9 es mostra la disposició del servo segons el nivell de modulació per amplada de pols que tingui a la consigna.

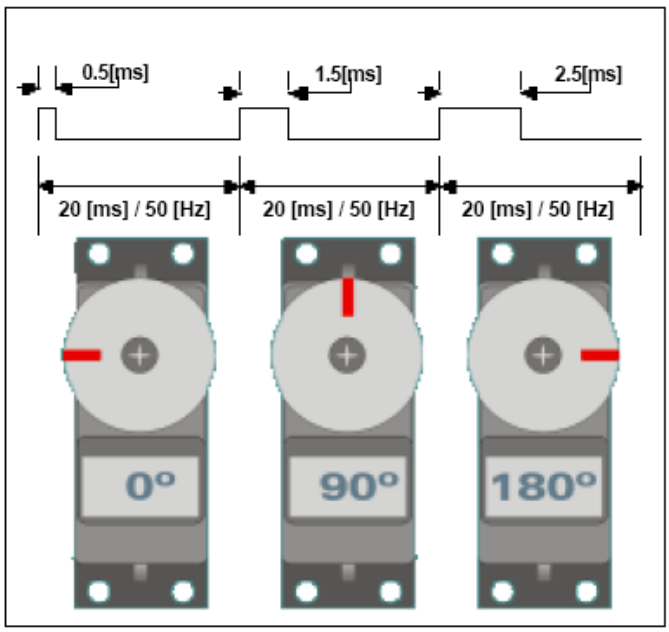

Figura 9. Posició del servo a partir de la duració del pols

2.3.2 SERVOMOTORS UTILITZATS

En el present projecte, s'han utilitzat 3 models diferents de servomotors, depenent de la posició en què es troben. A la taula 3 tenim una relació dels servomotors utilitzats i l'articulació a la qual pertanyen.

Articulació	Servomotor
Base	DY-S0213
Espatlla	DY-S0213
Colze	DY-S0213
Canell (Pitch)	HS-422
Canell (Roll)	HS-422
Pinça	DY-S0205

Taula 3. Servomotors utilitzats

Tots els servomotors venen acompanyats d'un conjunt d'accessoris per a l'eix de sortida que ens permet realitzar múltiples aplicacions amb ells. Per al projecte s'han utilitzat les platines circulars per a unir-los a l'estructura del braç i els maniguets antivibratoris també per a subjectar-los mitjançant cargols. La figura 10 ens mostra alguns dels accessoris que incorporen.

Figura 10. Accessoris inclosos amb els servomotors

El servomotor DY-S0213 de la casa Dong Yang Servo Power Model Co., Ltd. és un servomotor de gran potència i mida estàndard, gràcies a que disposa d'engranatges i eix de sortida metàl·lics i 2 coixinets de boles. Aquestes característiques li proporcionen gran velocitat i un parell molt elevat. A la taula 4 tenim reflectides les seves característiques.

Alimentació	4,8 – 7,2 V
Parell	14 kg×cm
Velocitat	0,16s/60º
Dimensions	40,8×21,7×41,3 mm
Pes	55 g
Engranatge	metàl·lic
Connexions	Vermell – Positiu Negre – Massa Taronja – Senyal

Taula 4. Característiques DY-S0213

S'ha escollit aquest model ja que és un dels que ofereix una relació de prestacions i preu més competents del mercat. La característica que més ens interessava era el parell que pot proporcionar. Existeixen altres models al mercat amb més parell, però a uns preus molt elevats o són difícils d'aconseguir al país.

La figura 11 ens mostra una imatge d'aquest servomotor amb els engranatges metàl.lics.

Figura 11. Servomotor DY-S0213

Els servomotors utilitzats en el canell són de la casa Hitec RCD el model HS-422, és un servomotor de mides estàndards i característiques també estàndards. És un dels servomotors més utilitzats en aquest tipus d'aplicacions i en aeromodelisme i radiocontrol.

La figura 12 es mostren les especificacions del servo HS422 utilitzat en el braç robot.

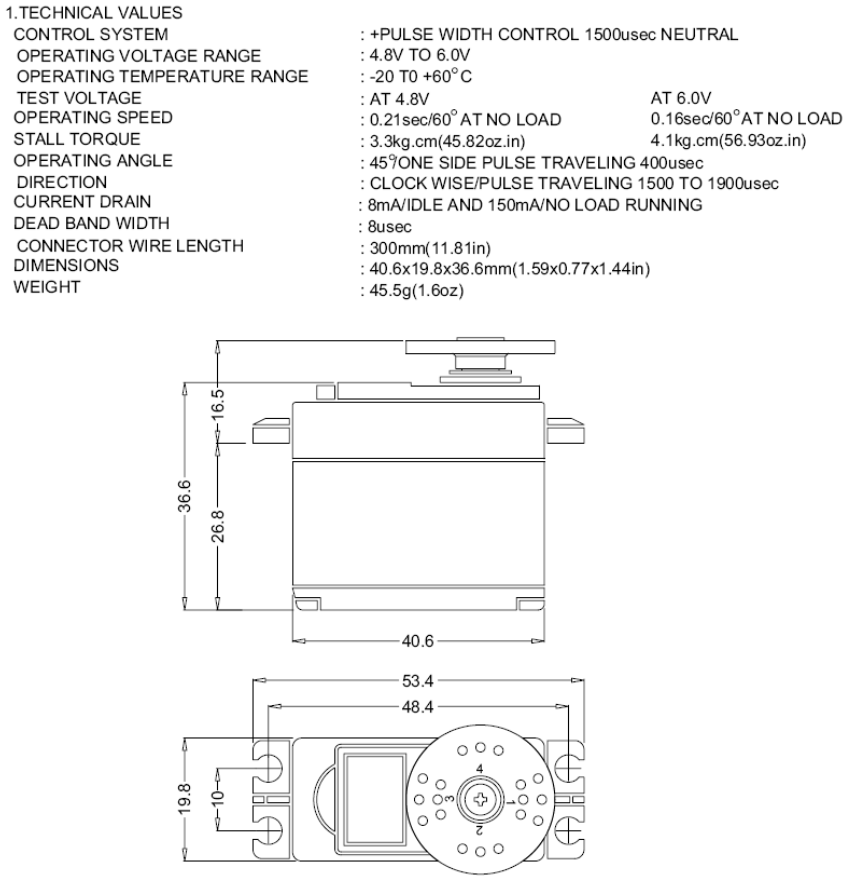

Figura 12. Especificacions servomotor HITEC HS422

Aquest servomotor té un parell inferior, 4,1kg×cm si li apliquem una alimentació de 6 volts. Tampoc disposa d'eix i engranatges de sortida metàl·lics, però com que s'utilitzen en les articulacions del canell i només han de suportar el pes de la pinça i de l'objecte a moure, compleixen els requisits que necessitem per a les especificacions del braç robot.

El servomotor utilitzat en la pinça també és de la casa DY-S0205. La taula següent ens mostra les seves principals característiques.

Alimentació	4,8 – 6 V
Parell	1,3 kg×cm
Velocitat	0,12s/60º
Dimensions	22,8×11,6×20,6 mm
Pes	8 g
Engranatge	plàstic
Connexions	Vermell – Positiu Negre – Massa Blanc – Senyal

Taula 5. Característiques DY-S0205

Aquest servomotor és de dimensions molt reduïdes i de molt poc pes, el que el fa ideal per a manejar l'obertura i tancament de la pinça, ja que té un parell suficient per agafar objectes i ens evita augmentar la mida de la pinça, o posar més pes a l'extrem del braç, cosa que ens evita que els servomotors que n'han de suportar tot el pes, pateixin més.

Tots els servomotors tenen els mateixos rangs de treball per al PWM. Segons característiques del fabricant, són 0,9ms i 2,1ms per a les posicions extremes de 0º i 180º respectivament i 1,5ms per a la posició central de 90º.

Els servomotors tenen un marge de treball de 180º, en la realitat aquest marge, queda ampliat uns quants graus més a banda i banda d'aquest marge.

3 CIRCUIT DE CONTROL

3.1 Descripció general del circuit

El circuit de control del braç robot i comunicació amb el PC tindrà dues parts diferenciades i separades. Una part és l'adequació de l'energia i l'altre és el mòdul de control i comunicació.

La part d'adequació de l'energia serà l'encarregada de variar els nivells de tensió i intensitat d'entrada, a uns valors que puguin ésser utilitzats pels elements que precisen alimentació, en aquest cas els servomotors del braç i els sensors de posició.

Aquests nivells necessaris seran 6 volts i 7 volts de corrent contínua.

La part de control i comunicació estarà basada en un microcontrolador PIC18F2550 que interpretarà les ordres donades des del PC, generarà els polsos necessaris per a crear els PWM que controlaran les posicions dels servomotors i llegirà el senyal dels optoacobladors instal·lats al braç o externs a aquest per a detectar posicions. La comunicació del microcontrolador amb el PC es farà amb un bus USB amb el qual el microcontrolador està preparat per a treballar.

La figura 13 ens mostra una visió general de la placa de control.

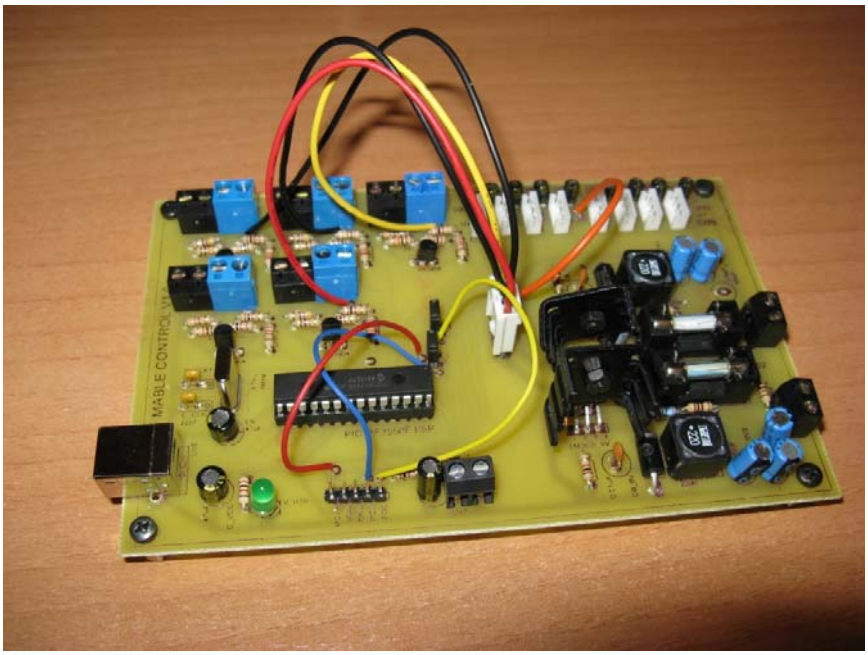

Figura 13. Placa de control

3.2 Adequació de l'energia

La placa de control disposarà de 2 fonts d'alimentació de 6 i 7 volts, ambdues de 3 ampers.

Es farà una primera adequació de la tensió de la xarxa a un nivell de continua. Aquesta adequació la farà una font externa que ens convertirà el nivell de 230 volts alterns de la xarxa, a 15 volts de contínua i 4,5 ampers. Les fonts incorporades a la placa de control adequaran aquests nivells als 6 i 7 volts i 3 amperes abans esmentats.

Les fonts d'alimentació utilitzades en la placa de control per a l'alimentació dels servomotors són fonts commutades, de tipus reductor o step-down. Les fonts commutades presenten una sèrie d'avantatges respecte les fonts lineals que es representen a la taula 6.

PARÀMETRE	LINEAL	COMMUTADA
Potència de sortida	<50W	<2000W
Rendiment	30-55%	60-90%
Arrissat	0.5-20 mVpp	10-100 mVpp
Regulació		
Línia	0.02-0.1%	0.05-1%
Càrrega	0.05-0.1%	0.1-1%
Temps		
Manteniment	2ms	8-40ms
Establiment	50µs	300µs

Taula 6. Comparativa fonts alimentació

A més, les fonts commutades presenten l'avantatge que no basen la seva transformació de l'energia en un transformador, el que permet una reducció important de l'espai i també dels components, per a la placa de control.

3.2.1 DISSENY

Les dues fonts dissenyades són de 6 i 7 volts de sortida i ambdues de 3 ampers. S'ha optat per un adaptador de tensió extern a la placa de control, que ens permetrà prèviament adaptar la tensió de la xarxa a 15 volts en contínua i 4,5 ampers. Això ens ha permès simplificar el circuit de les fonts commutades internes i també obtenir un nivell de seguretat contra contactes indirectes a les persones o als components, ja que s'ha deixat la part de transformació de la tensió alterna de la xarxa a contínua, externa a la placa de control.

Les fonts s'han dissenyat a través del software online Webench Online Power Supply Design Tools de la casa National Semiconductors. Aquest programa ens permet introduir els paràmetres elèctrics de la font d'alimentació desitjada i ens dissenya una font amb els seus components externs i en recomana alguns en concret.

Un cop introduïts els paràmetres, el software ens permet escollir entre una àmplia gamma de circuits integrats que ens permetran assolir les característiques desitjades. S'ha escollit l'integrat LM2676 de la mateixa casa. Aquest circuit es troba en dos tipus d'encapsulats, hem escollit l'encapsulat tipus TA07B. Existeixen diferents versions ja programades per a 3.3 V, 5V, 12 V i una versió ajustable, que ve programada de sèrie a 5V, però que permet, variant pocs components externs, obtenir marges de tensió i intensitat de sortida diferents dels abans esmenats. La següent figura mostra un esquema típic de connexionat per a aquest tipus de circuits.

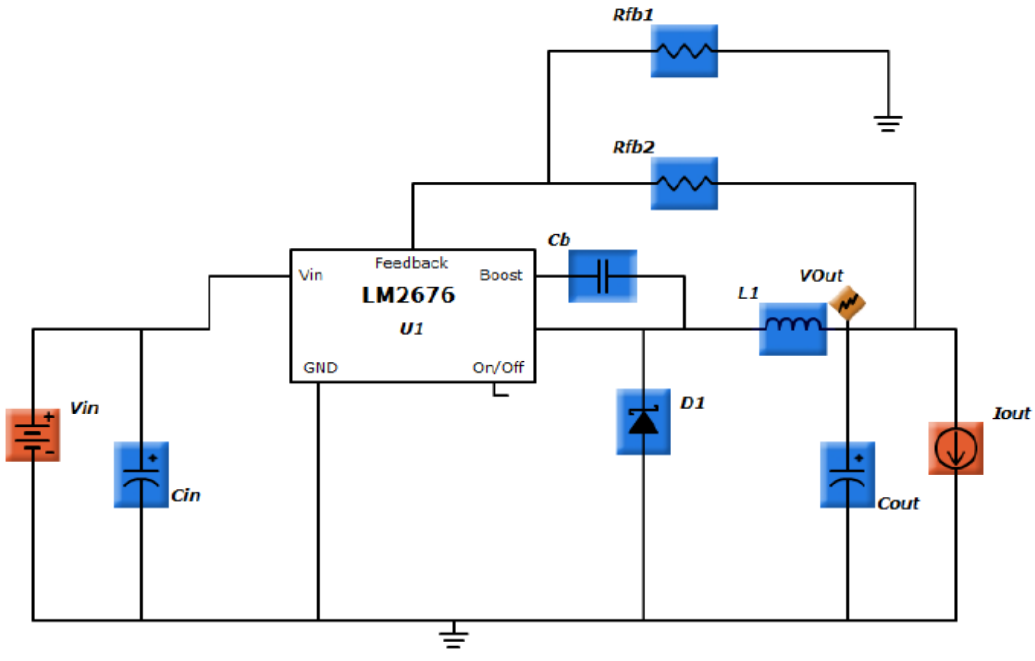

Figura 14. Circuit bàsic LM2676

La relació de components recomanats per el fabricant per a la font de 7V 3A és la indicada en la taula 7.

Denominació	Component	Valor
Cb	Condensador ceràmic	0.01µF
Cin	Condensador electrolític	10µF, 0.004Ω
Cout	Condensador electrolític	68µF, 0.0015Ω (2 unitats)
D1	Diode Schottky	3A, 0.45V
IC	Circuit integrat	LM2676T-ADJ
L1	Inductància de potència	22µH, 0.0233Ω
Rfb1	Resistència	1000Ω
Rfb2	Resistència	4870Ω

Taula 7. Components font 7V, 3A

La relació de components recomanats per el fabricant per a la font de 6V 3A és la indicada en la taula 8.

Denominació	Component	Valor
Cb	Condensador ceràmic	0.01µF
Cin	Condensador electrolític	10µF, 0.004Ω
Cout	Condensador electrolític	68µF, 0.0015Ω (3 unitats)
D1	Diode Schottky	3A, 0.45V
IC	Circuit integrat	LM2676T-ADJ
L1	Inductància de potència	22µH, 0.0233Ω
Rfb1	Resistència	1000Ω
Rfb2	Resistència	4020Ω

Taula 8. Components font 6V, 3A

3.2.2 REALITZACIÓ

S'han escollit el màxim de components dels models i marques recomanats per el programa. No s'ha optat per alguns d'ells perquè al tractar-se de components SMD ens dificultava el muntatge del circuit. Les resistències Rfb2 d'ambdues fonts no són de valors normalitzats, per tant, s'ha optat per a deixar una previsió a la placa pel circuit de cada font, de 3 resistències que es podran connectar en sèrie. Aquestes resistències, podran ser 3 o menys, de valors normalitzats, que sumats i prèviament comprovats donin el valor recomanat o el màxim aproximat possible. Igualment per als condensadors de la sortida Cout, que el fabricant ens dona la opció amb diferents valors de condensadors i en diferents quantitats, s'ha optat per deixar una previsió a la sortida de cada font commutada per a la connexió de fins a 3 condensadors en paral·lel.

El fabricant ens garanteix per a la font de 7 volts una eficiència del 92%, i un cop realitzada en el circuit de control s'ha comprovat que la sortida compleix amb el valor de tensió desitjat de 7V±1%.

Per a la font de 6 volts en garanteix una eficiència del 91%, i un cop realitzada en el circuit de control s'ha comprovat que la sortida compleix amb el valor de tensió desitjat de 6V±1,33%.

Com a mesura de seguretat, la sortida de cada font, disposa d'un fusible de 3 ampers per si es dona el cas que els servomotors demanessin més intensitat que la màxima proporcionada per la font. Això evitarà que es destrueixin o malmetin components, que hi hagi sobreescalfaments dins la caixa que conté el circuit de control. També ens permetrà protegir el braç robot i els servomotors enfront a sobrecàrregues que pugui patir.

Cada font alimentarà 4 connectors de 3 pins, els que ens permetrà tenir fins a 8 servomotors, quatre a 7 volts i quatre més a 6 volts, que podran ser controlats des del circuit. La font de 6 volts també alimentarà els circuits d'alimentació i adequació del senyal dels sensors òptics.

3.3 Mòdul de control i comunicació

3.3.1 MICROCONTROLADOR

Amb l'expansió creixent del bus USB en els ordinadors també han sorgit nous dispositius que funcionen amb aquest bus, és el cas dels nous microcontroladors de la casa Microchip de les sèries PIC18F2**5* i PIC18F4**5* en microcontroladors de 8 bits, i d'altres models també en 16 i 32 bits. Aquests dispositius permeten una connexió directa amb el bus USB 2.0, actual en mercat i que incorporen tots els ordinadors de nova fabricació.

La progressiva desaparició del port sèrie dels ordinadors, especialment en ordinadors portàtils, enfront l'USB ha propiciat l'aparició d'aquesta nova gamma de microcontroladors.

Per al control dels servomotors, dels sensors òptics i la comunicació via USB amb el PC de la placa de control del braç robot s'ha escollit el microcontrolador PIC18F2550 de la casa Microchip. Algunes de les seves característiques són: 8 bits, fins a tres interrupcions externes, fins a 2 mòduls Capture/Compare i PWM amb una resolució de fina a 10 bits i que es poden incrementar en un d'addicional, dos comparadors analògics duals amb multiplexat intern, possibilitat de dos rellotges externs i algunes més les trobem a la taula 9.

Features	**PIC18F2550**
Operating Frequency	DC – 48 MHz
Program Memory (Bytes)	32768
Program Memory (Instructions)	16384
Data Memory (Bytes)	2048
Data EEPROM Memory (Bytes)	256
Interrupt Sources	19
I/O Ports	Ports A, B, C, (E)
Timers	4
Capture/Compare/PWM Modules	2
Serial Communications	MSSP, Enhanced USART
Universal Serial Bus (USB) Module	1
Streaming Parallel Port (SPP)	No
10-Bit Analog-to-Digital Module	10 Input Channels
Comparators	2
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR (optional), WDT
Programmable Low-Voltage Detect	Yes
Programmable Brown-out Reset	Yes
Instruction Set	75 Instructions; 83 with Extended Instruction Set enabled

Taula 9. Característiques PIC18F2550

Disposa de 3 ports d'entrades i/o sortides, PORTA, PORTB i PORTC. El PORTA disposa de 6 pins que es poden configurar com a entrades/sortides digitals ó com a entrades analògiques, aquest últim mode només és possible en 5 dels seus pins. El PORTB consta de 8 pins que també es poden configurar com a entrades/sortides digitals i 5 d'ells també com a entrades digitals. El PORTC és de 7 pins, 5 dels quals es poden configurar com a entrades/sortides digitals i la resta com a entrades digitals.

A la figura 15 podem veure el diagrama de pins amb l'ús de cadascun, per a l'encapsulat de 28 pins PDIP utilitzat en la placa de control.

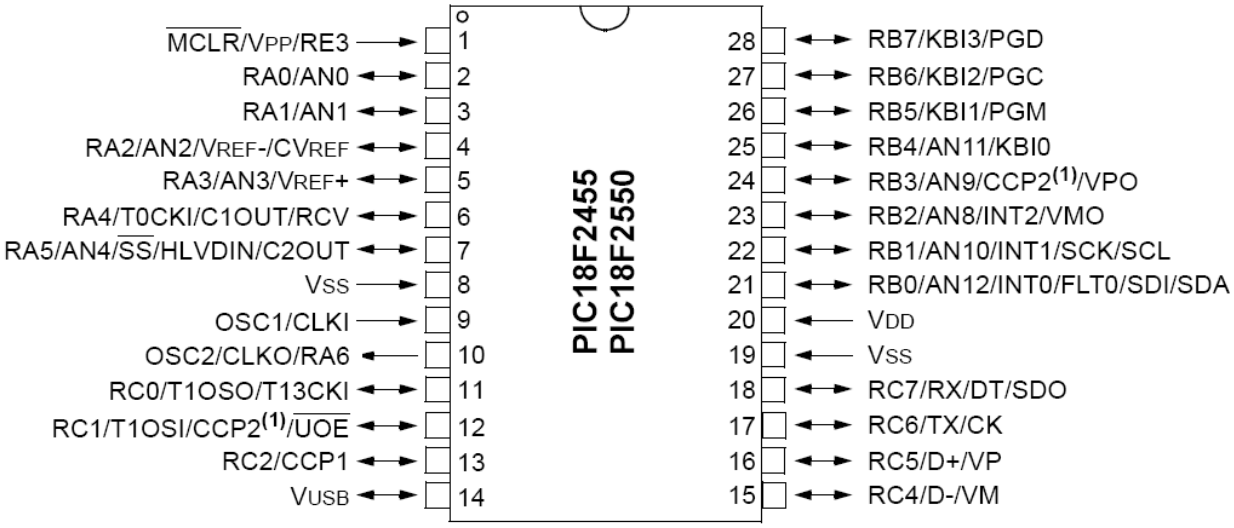

Figura 15. Diagrama de pins

El PIC18F2550 disposa ja de dues sortides de PWM, la RC2 i la RB3. Aquestes no són suficients per a controlar 8 servomotors, pel que utilitzarem les 8 sortides del port B com a sortides digitals, a través de les quals s'emetran els 8 PWM necessaris per a controlar els servomotors.

El PWM serà programat internament utilitzant dos Timers, els que ens permetrà establir la freqüència dels polsos, 50Hz, i l'altre ens permetrà variar el temps del pols en estat alt, entre els valors mínims i màxims que necessiten els servomotors, que són entre 0,9 i 2,1 ms. Com que el període entre polsos és de 20 ms i la durada màxima en estat alt d'un pols pot ser 2,1 ms això ens permet, enviar fins a 9 polsos consecutius com a màxim cada període, tal i com mostra la figura 16.

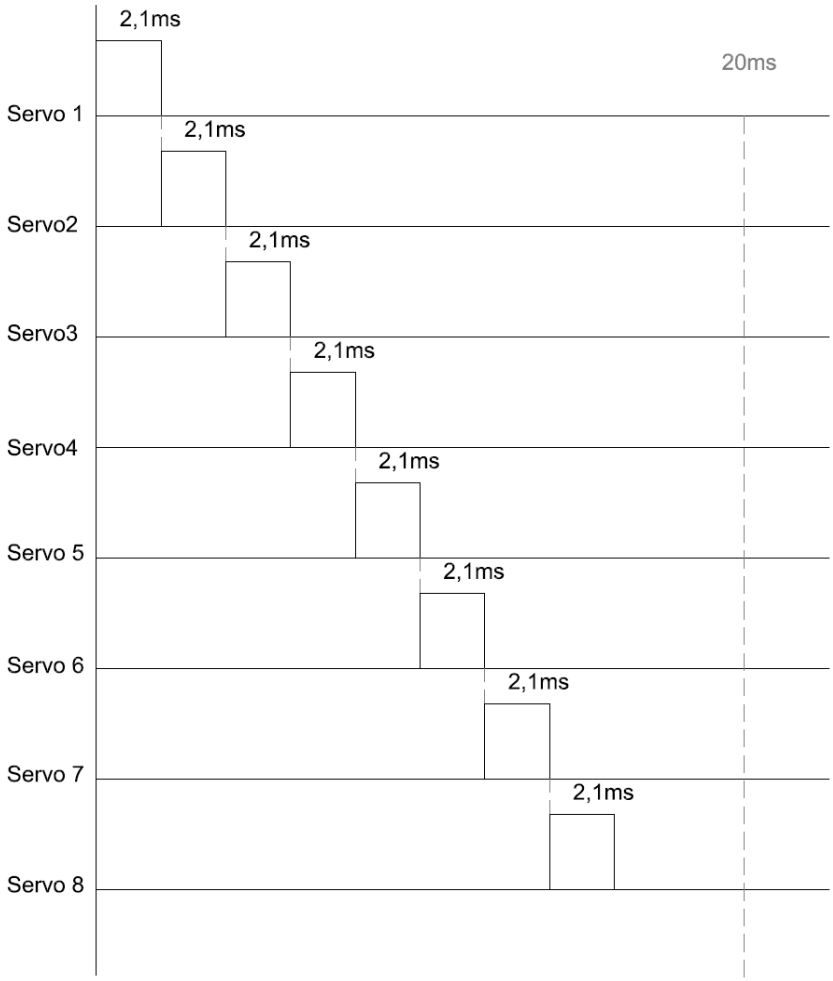

Figura 16. PWM implementats

El sensors òptics es connectaran als pins RA0, RA1, RA2, RA3 i RA5 del PORTA que són els que permeten tenir entrades tan digitals com analògiques, el que servirà per a configurar els sensors també com a tals. El nivell de tensió dels sensors estarà adequat a un valor possible per a les entrades del microcontrolador ja que estan alimentats a 6V, aquest valor serà 4,125V.

Del PORTC utilitzarem els pins RC4 i RC5 com a receptors de les dades enviades pel bus USB a través dels pins D- i D+. Utilitzarem RC0 i RC1 per activar els LEDs indicadors de detecció i de recepció de dades respectivament.

L'oscil·lador del microprocessador s'ha fet seguint l'esquema de la figura 17.

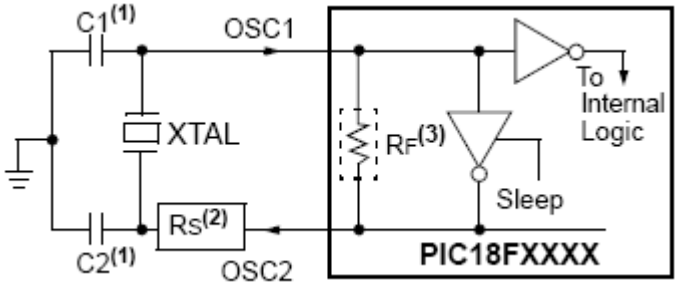

Figura 17. Oscil·lador

Per el qual s'ha escollit un cristall de 8 MHz i dos condensadors de 22pF seguint les recomanacions del fabricant, i que és suficient per a poder tenir uns temps de Timer suficientment petits com per poder fer funcionar correctament els PWM.

El microcontrolador s'encarregarà del control dels 8 servomotors, llegirà les entrades dels sensors i rebrà les dades del programa Mable Control des de l'ordinador que li donaran la informació necessària per a posicionar els servomotors.

L'alimentació serà a 5 V de tensió contínua i es farà a través del bus USB amb els pins VCC i GND d'aquest, que es connectaran a les potes 20 i 19 respectivament, del microcontrolador.

El circuit incorpora un polsador de reset extern, per a poder reiniciar el sistema en cas de fallada.

A més es disposarà d'un LED indicador per informar de que el bus USB alimenta el microcontrolador. Aquest també controlarà dos LEDS més indicatius. El LED vermell s'encendrà quan el microcontrolador sigui detectat correctament per l'ordinador, i el LED verd intermitent, que la transmissió de dades és constant.

3.3.2 SENSORS DE POSICIÓ

El microcontrolador també rebrà el senyal digital proporcionat per 5 sensors que es podran col·locar en diferents parts del braç, de la base o externs, depenent de la utilització final del braç i del lloc de treball d'aquest, i que ens permetran establir criteris de seguretat, calibració o recepció de dades externes.

Aquests sensors es col·locaran segons el lloc o l'ús a què es destini el braç robot i permetran evitar al braç rangs de treball perillosos o interactuar amb altres processos exteriors al braç robot, actuant com a sensors. Estaran connectats al circuit de control mitjançant cablejat adient i regletes per a placa. La placa estarà serigrafiada amb l'ús de cadascun dels 4 pins de què disposa l'optoacoblador, per evitar confusions en el connexionat.

Aquests sensors són òptics i estan basats en optoacobladors ranurats com els que es mostren a la figura 18.

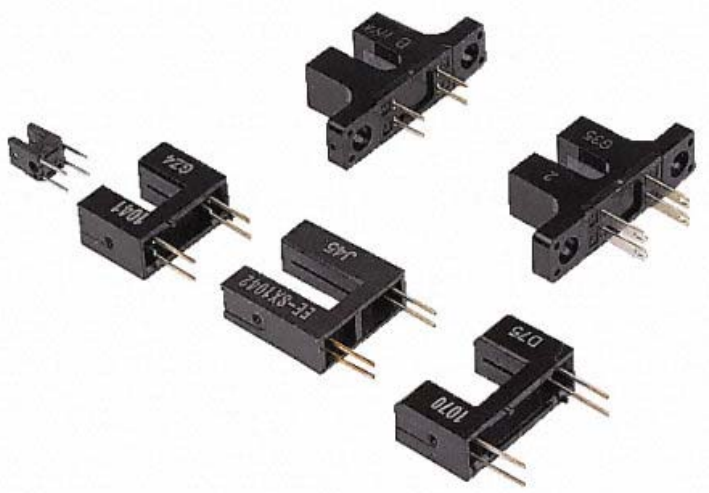

Figura 18. Optoacobladors ranurats

Un optoacoblador és un dispositiu d'emissió i recepció de llum que funciona com un transistor el qual la base s'excita mitjançant llum infraroja. La llum és emesa per un diode LED que satura un component optoelectrònic en forma de fototransistor. Es combina en un sol dispositiu semiconductor, fotoemissor i fotoreceptor, la connexió entre els quals només és òptica.

Com que l'emissor i el receptor estan separats físicament, això ens permet utilitzar-los com a dispositius analògics que detectarien la variacions de llum al passar objectes translúcids entremig de l'encapsulat, o com a dispositius digitals tot-o-res que permeten detectar quan el feix de llum infraroja és tallat per un objecte opac. És aquest últim cas que serà utilitzat com a sensor en el circuit del present projecte.

L'optoacoblador utilitzat és de la casa OSRAM el model SFH 9315. L'amplitud de l'ona lluminosa és de 950 nm i està equipat amb filtre de llum de dia, per a poder treballar en qualsevol zona i hora del dia. El circuit que alimentarà l'optoacoblador serà el que es mostra a la figura 19.

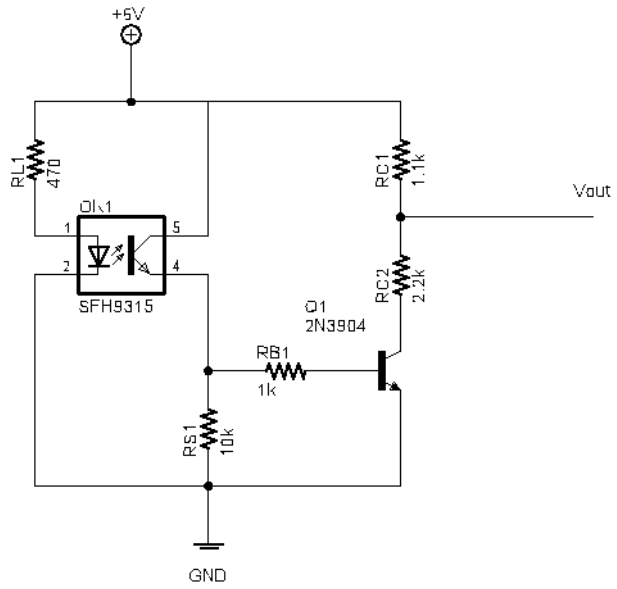

Figura 19. Circuit optoacoblador

Per alimentar el circuit utilitzarem la font d'alimentació de 6 volts. El diode LED necessita una resistència ja que és un dispositiu que funciona per intensitat. El senyal rebut pel receptor actuarà sobre la base d'un transistor que ens permetrà transformar el senyal en tot-o-res per a poder ser rebut pel microcontrolador.

Les entrades del microcontrolador no permeten valors de tensió més elevats que l'alimentació del microcontrolador, com que el circuit l'alimentem a 6 volts, es posa un divisor de tensió que adequarà el senyal, que es pot simplificar amb la figura 20.

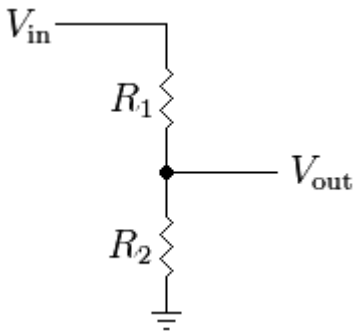

Figura 20. Divisor de tensió

Per trobar les resistències i el valor de sortida de tensió s'utilitza la fórmula representada en l'equació 1.

$$\text{Vout} = \text{Vin} \times \frac{\text{R}_2}{\text{R}_1 + \text{R}_2} \qquad \text{(Eq. 1)}$$

Utilitzant per a les resistències R_1 i R_2 els valors comercials de 1kΩ i 2,2kΩ s'obté el valor de tensió a la sortida de l'equació 2.

$$\text{Vout} = 6 \times \frac{2{,}2\text{k}}{1\text{k} + 2{,}2\text{k}} = 4{,}125\text{V} \qquad \text{(Eq. 2)}$$

Aquest valor de tensió si que es pot aplicar com a entrada al microcontrolador. Mentre el sensor de l'optoacoblador rebi llum donarà un valor a l'entrada corresponent del microcontrolador de 4,125 volts, quan el feix de llum es talli, aquest valor passarà a ser 0 volts.

3.4 Hardware

El circuit de control anirà muntat dins una caixa de plàstic amb tapa superior d'alumini. La placa del circuit estarà subjectada a la caixa mitjançant quatre separadors metàl·lics hexagonals de 10mm de llargada.

En la tapa davantera d'alumini es col·locaran els elements de senyalització i de control que seran: interruptor on-off que permetrà l'entrada d'energia de la font externa al circuit, LED indicador que ens mostrarà quan el circuit està alimentat i polsador de reset que permetrà resetejar el microcontrolador en cas d'error.

La caixa també disposarà en un lateral d'un connector d'alimentació femella de 2,5 mm. En l'altre lateral tindrà un forat quadrat que permetrà al connector USB tipus B sortir a l'exterior de la caixa, i així permetre la connexió del cable fins al PC. Per la part posterior entraran els cables dels servomotors i dels sensors del braç robot.

Els diferents LEDs indicadors de que disposa la part de control i comunicació seran interns, ja que no es precisen durant el control i només seran visibles per quan es programi si es retira la tapa davantera.

La figura 21 ens mostra l'acabat final del circuit de control dins la caixa.

Figura 21. Caixa del circuit de control

4 PROGRAMACIÓ

4.1 El bus USB

El Bus Sèrie Universal ó USB és un port que serveix per a connectar perifèrics a l'ordinador. Va ser creat l'any 1996 per set empreses: IBM, Intel, Norhtern Telecom, Compaq, Microsoft, Digital Equipment Corporation i NEC.

Va sorgir com la necessitat d'unificar tots els connectors creant-ne un de més senzill i de majors prestacions. Així va néixer l'USB, amb una velocitat de 12Mbits/s, i en la seva evolució i actual versió USB 2.0 a una velocitat de 480Mbits/s, enfront la velocitat d'entre 600Kbits/s i 1,5Mbits/s d'un port paral·lel, o els 112Kbits/s d'un port sèrie. Actualment es troba en fase experimental la versió 3.0 que permetria una transmissió de dades de fins a 4,8Gbit/s.

Treballa com una interfície plug&play entre el PC i certs dispositius com poden ser, teclats, ratolins, escàners, càmeres, impressores, mòdems, plaques de so, etc. El sistema plug&play ens permet la connexió d'un dispositiu informàtic a l'ordinador sense la necessitat d'instal·lar cap hardware o software específic. El dispositiu és detectat automàticament i no és necessari que el sistema s'apagui o busqui nou hardware. Es permet la connexió de fins a 127 perifèrics USB.

És un bus de 4 fils els quals 2 es destinen per a la transmissió de dades, i els altres dos permeten al dispositiu perifèric que es connecta a l'ordinador rebre alimentació d'aquest a través de l'USB, ja que subministra 5 volts de contínua. A la taula 10 tenim els diferents fils que formen el bus.

Pin	Nom	Color de cable	Descripció
1	VCC	Vermell	+5V
2	D-	Blanc	D-
3	D+	Verd	D+
4	GND	Negre	Massa

Taula 10. Cablejat USB

Les dades de l'USB són transmeses per un cable parell trenat, amb una impedància de 90Ω±15% anomenats D+ i D-. Aquests, col·lectivament utilitzen senyalització diferencial en half dúplex per combatre els efectes del soroll electromagnètic en enllaços llargs. D+ i D- normalment operen en conjunt i no són connexions simplex.

Els nivells de transmissió del senyal varien de 0 a 0,3V per baixos (zeros) i de 2,8 a 3,6V per a alts (uns) en les versions 1.0 i 1.1 del bus, i en ±400mV en Alta Velocitat (USB 2.0). En les primeres versions, el coure dels cables no estan connectats a massa, però en el mode d'alta velocitat es té una terminació de 45Ω a massa o un diferencial de 90Ω per acoblar la impedància del cable.

És un bus basat en el pas d'un testimoni, semblant a altres busos com els de les xarxes locals en anell amb pas de testimoni i les xarxes FDDI. El controlador USB distribueix testimonis per el bus. El dispositiu, la direcció del qual coincideixi amb la que porta el testimoni respon acceptant o enviant dades al controlador. Aquest també gestiona la distribució d'energia als perifèrics que ho requereixin.

Utilitza una tipologia d'estrelles apilades que permeten el funcionament simultani de 127 dispositius. La figura 22 ens mostra l'estructura de capes del bus USB.

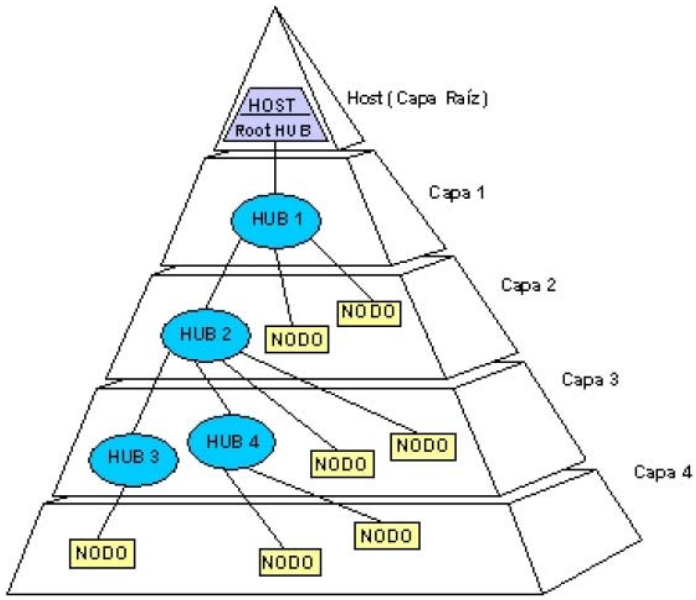

Figura 22. Estructura de capes del bus USB

En l'arrel o vèrtex de les capes, hi ha el controlador amfitrió o host que controla tot el tràfic que circula per el bus. Aquesta tipologia permet a molts dispositius connectar-se a un únic bus lògic sense que els dispositius que es troben més avall en la piràmide sofreixin retard. A diferència d'altres arquitectures, l'USB no és un bus d'emmagatzematge i enviament, de manera que no es produeix retard en l'enviament d'un paquet de dades cap a les capes inferiors. Com a detall sorprenent és que cada port utilitza una única sol·licitud d'interrupció independentment dels perifèrics que tingui connectats, per això no hi ha risc de conflictes entre una quantitat de dispositius que d'altra manera no podrien ser connectats per falta de recursos. De la mateixa manera tampoc utilitzen assignació de memòria.

El software client s'executa en el host i correspon a un dispositiu USB; es subministra amb el sistema operatiu o amb el dispositiu USB. El software del sistema USB, és el que suporta USB en un determinat sistema operatiu i es subministra amb el sistema operatiu independentment dels dispositius USB o del software client.

L'estàndard USB especifica toleràncies per impedància i especificacions mecàniques relativament baixes per els seus connectors, intentant minimitzar la incompatibilitat entre els connectors fabricats per diferents companyies. Una meta a la que s'ha aconseguit arribar. L'estàndard USB, a diferència d'altres estàndards, també defineix mides per l'àrea al voltant del connector d'un dispositiu, per evitar el bloqueig d'un port adjacent per el dispositiu en qüestió.

Les especificacions USB 1.0, 1.1 i 2.0 defineixen dos tipus de connectors per connectar els dispositius al servidor: A i B. No obstant, altres fabricants de perifèrics més petits han desenvolupat els seus mitjans de connexió petits, i han aparegut al mercat gran varietat d'ells. La longitud màxim dels connectors és de 5 metres. A la figura 23 tenim els tipus de connectors més estesos i els seus pins de connexió.

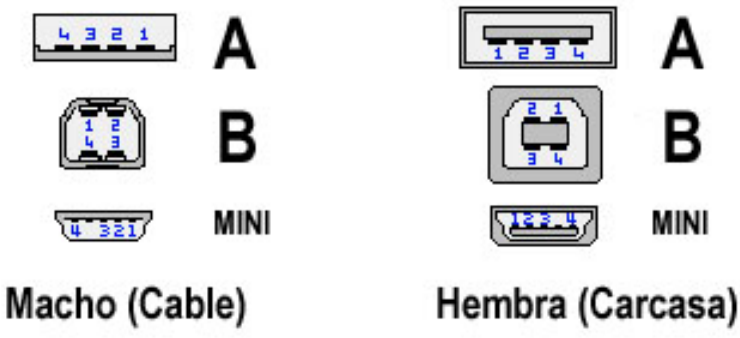

Figura 23. Connectors USB

La ràpida proliferació i adaptació del port USB als nous ordinadors ha fet que cada vegada desapareguin més els ports sèrie i paral·lels dels ordinadors, i la majoria de portàtils ja no incorporen el port sèrie, mentre que els ports USB van guanyant terreny per la seva versatilitat i funcionalitat plug&play. Davant d'aquesta situació, la tradicional programació i comunicació amb microcontroladors a través de port sèrie es veu complicada i ens obliga moltes vegades a utilitzar adaptadors o dispositius estranys per adaptar les connexions USB a port sèrie i viceversa.

Per això, juntament a el sorgiment de programadors per microcontroladors via USB, en el present projecte s'ha utilitzat la nova sèrie de microcontroladors PIC que permeten una comunicació directa via USB amb qualsevol ordinador. Permeten així desenvolupar un dispositiu que permet ésser programat i controlat total i únicament a través del bus USB.

4.2 Programació del PIC18F2550

Per a la programació del PIC18F2550 primerament s'ha creat el codi que permetrà el control dels dispositius i la comunicació amb l'ordinador amb el programa MikroBASIC versió 4.0.0.3 de la casa MikroElektronika.

És un compilador en llenguatge BASIC per microcontroladors PIC molt estès últimament. És un entorn que suporta molts models de microcontroladors i disposa d'un enorme llistat de llibreries dividides en comunicacions: RS-232, RS-485, I2C, PS/2, USB, interfícies LCD, etc.

El fabricant posa a disposició una versió gratuïta que ens limita l'arxiu .hex a una mida màxima de 2kb.

Un cop el codi està creat, el MikroBASIC el compila i crea un arxiu hexadecimal, que hem anomenat MABLE_CONTROL_V1.0.hex. Aquest arxiu serà el que carregarem al microcontrolador.

Tot i que el MikroBASIC permet ser enllaçat directament amb software de programació per PICs com ara el WinPIC800, aquest però només compatible amb un programador via USB, el GTP-USB.

El programador utilitzat en aquest projecte és el MPLAB ICD2 de la casa Sure Electronics. És compatible amb el software de programació MPLAB IDE de la casa Microchip, que permet ser descarregat gratuïtament.

MPLAB IDE és un software que ens permet crear programes per a microcontroladors PIC i programar-los. També permet importar arxius en format .hex. Aquest serà el nostre cas ja que a l'utilitzar un compilador diferent per la seva facilitat d'ús, haurem de crear un fitxer hexadecimal que podrà ser importat a l'MPLAB IDE, versió 8.10 utilitzada en el present projecte. Juntament amb el fitxer .hex importarem la llibreria USBdsc.bas que també s'ha de carregar al microcontrolador de la placa de control per tal de què aquesta sigui reconeguda com a dispositiu USB per l'ordinador.

El programador utilitzat compatible amb el software de Microchip, ens permet programar els disposistius de dues formes: directament sobre el circuit prèviament deixant una previsió de pins de connexió per al cable connector, o a través d'un sòcol ZIF que permet muntar dispositius de fins a 44 potes.

La placa de control disposa d'un sòcol per poder extreure el PIC18F2550 i programar-lo en el sòcol ZIF, aquesta és la opció escollida per a la programació del microcontrolador en aquest projecte. S'ha deixat una previsió per realitzar una programació directa a la placa de control a través d'un array de pins. La taula 11 ens mostra quines són les potes demanades per aquest programador per a programar el PIC.

PIN	Número
MCLR/VPP	1
VDD	20
VSS	19
ICSPDAT/PGD	28
ICSPCLK/PGC	27

Taula 11. Pins programació externa

Els pins 27 i 28 són utilitzats com a sortides per al control dels servomotors. Per tal d'evitar interferències en la programació i per no haver de desconnectar els servomotors, s'ha previst en aquestes pistes jumpers de connexió que podran ser extrets, aïllant així la programació del servomotor.

4.3 Software de control pel PC

4.3.1 FUNCIONAMENT

El programa dissenyat per a un possible control del braç robot des d'un ordinador ha estat dissenyat amb Visual Basic 6.0, i s'anomena Mable Control.

S'ha dissenya de manera que puguin esser explotades totes les possibilitats de la placa de control. Permetrà el control de fins a 8 servomotors i de 5 entrades que poden esser analògiques o digitals per a recepció d'informació externa, que són els màxims per als que està preparada la placa.

La figura 24 ens mostra la pantalla principal del programa Mable Control.

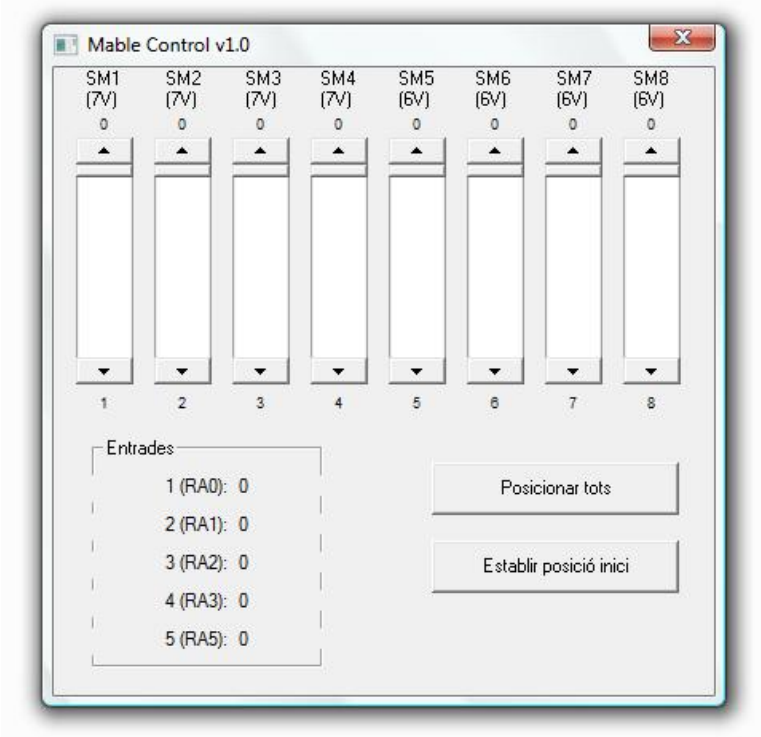

Figura 24. Pantalla Principal

El funcionament del programa és molt senzill. Consta d'una pantalla principal on trobem vuit barres desplaçables o sliders, que permeten el control de cadascun dels servomotors. Sobre cada barra hi ha indicat la numeració de servomotor i si pertany al bloc de 7 volts o de 6 volts. Cadascuna d'aquestes barres ens permet variar la posició del servomotor en 256 divisions possibles, essent 127 la posició intermitja, 0 un dels extrems i 255 l'altre.

Les barres poden ésser desplaçades arrastrant l'slider central, el que ens permetrà efectuar moviments ràpids del braç, també podem desplaçar la barra amb les fletxes col·locades a l'extrem superior i inferior, que desplaçarà cada cop la barra només una de les 256 possibles posicions i permetrà moviments més precisos, i finalment, també es poden desplaçar prement l'espai que queda entre l'slider central i un dels extrems de la barra, el que provocarà moviments suaus, lents i continus.

A la part inferior esquerra trobem la part que llegeix el valor de les 5 possibles entrades de què disposa la placa.

Les barres ens permeten moure cadascun dels servomotors per separat, però el programa també incorpora un botó anomenat Posicionar tots, que ens permet variar les posicions de tots els servomotors a la vegada.

Finalment incorpora un botó anomenat Establir posició inici, que permet gravar la posició actual al microcontrolador per així, en posteriors usos del braç robot, aquest inicialitzarà cadascun dels servomotors en la posició abans indicada.

El programa també ens avisa quan la placa no és detectada correctament, mitjançant la finestra d'error que ens mostra la figura 25.

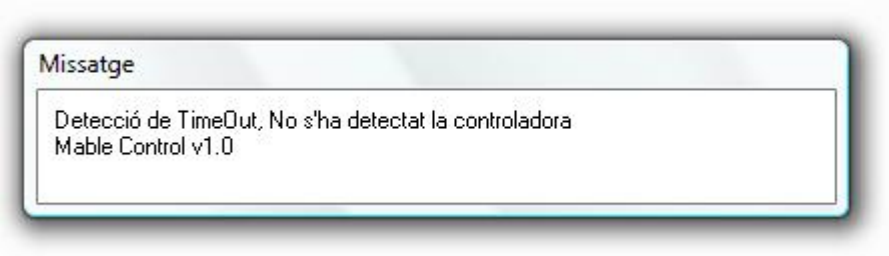

Figura 25. Finestra d'error

4.3.2 CONSIDERACIONS

Per a què el programa pugui enviar les dades a través d'USB, la casa Intel proporciona uns mòduls i llibreries per a crear una comunicació amb el programa Microsoft Visual Basic.

Serà necessària la instal·lació d'una llibreria de tipus API a la carpeta Windows de l'ordinador si és que aquest no la té. Aquesta llibreria és la apigid32.dll, que es pot descarregar gratuïtament de la pàgina web d'Intel.

Per al programa creat amb Visual Basic 6.0 caldrà carregar dos mòduls, també proporcionats per Intel, que s'hauran d'implementar i que permetran la comunicació amb el dispositiu USB, són l'AccessIODevice, utilitzat en totes les crides a l'USB, i l'APIcalls utilitzat per accedir a les llibreries Win32 API.

Com que s'han utilitzat les llibreries d'Intel, la placa de control i el software només funcionaran amb ordinadors amb microprocessador Intel, o que siguin compatibles.

El software ha estat provat i funciona correctament amb les versions de Windows XP, i Windows Vista.

Si la placa no és correctament detectada el botó de reset incorporat ens permet resetejar el microcontrolador i que l'ordinador el torni a detectar.

Caldrà desconnectar tots els dispositius USB que necessitin controlador de l'ordinador i que no siguin la placa de control abans d'executar el programa Mable Control, ja que altres dispositius fan que el programa no detecti la placa de control.

5 RESUM DEL PRESSUPOST

El projecte de disseny i construcció d'un braç robot de 5 graus de llibertat i de la placa de control i comunicació amb el PC via USB, i del disseny i implementació del software de control té un cost de dos mil nou-cents noranta-dos euros amb vint-i-set cèntims, sense IVA.

6 CONCLUSIONS

Amb el descrit anteriorment es pretén haver explicat les característiques del projecte, així com el seu abast i funcions, demostrant així que s'han complert els objectius inicials de dissenyar i construir un braç robot de 5 graus de llibertat, constituït a partir de servomotors; i de dissenyar i construir la placa el control d'aquest braç i de comunicació amb el PC via USB, juntament amb el software que permetrà el control del braç per part de l'operari des del PC. El projecte seria ampliable per al control de robots de fins a 8 graus de llibertat i la possibilitat de connectar fins a 5 sensors òptics tal i com s'ha previst en el disseny i construcció del circuit de control.

Gerard Massaguer Frigolé
Enginyer Tècnic Industrial esp. Electrònica Industrial

Banyoles, 20 de juny de 2008

7 RELACIÓ DE DOCUMENTS

El projecte està format per els documents: memòria, plànols, plec de condicions, estat d'amidaments i pressupost.

8 BIBLIOGRAFIA

CADSOFT COMPUTER GmbH. Llibreries Eagle Layout Editor. Alemanya. (ftp://85.10.203.38/pub/userfiles/libraries/, 22 d'abril de 2008)

CENTRO DE FORMACIÓN DEL PROFESORADO E INNOVACIÓN EDUCATIVA VALLADOLID II. Servomotores. Espanya. (http://cfievalladolid2.net/tecno/cyr_01/robotica/sistema/motores_servo.htm, 25 d'octubre de 2007)

DONG YANG SERVO POWER MODEL CO. LTD. DYS0213 datasheet. Xina. (http://doc.diytrade.com/docdvr/499822/5199508/1203236582.pdf, 2 de novembre de 2007)

FERNÁNDEZ-CARO, J. Connectar PIC18F2550 via USB. (http://www.hobbypic.com/index.php?option=com_content&task=view&id=14&Itemid=32, 12 de gener de 2008)

FERNÁNDEZ-CARO, J. Driver WinUSB. (http://www.hobbypic.com/index.php?option=com_content&task=view&id=31&Itemid=27, 12 de gener de 2008)

FOROS DE ELECTRONICA. Implementar PWM en un PIC. (http://www.forosdeelectronica.com/about3565.html, 3 de Març de 2008)

GARCÍA-CUERVO. Programació amb PICC. (http://picmania.garcia-cuervo.com/PICC.htm, 25 de novembre de 2007)

HITEC. HS422 datasheet. USA. (http://www.hitecrcd.com/Servos/hs422.htm, 2 de novembre de 2007)

INTEL CORPORATION. USB Design by Example. USA (http://www.intel.com/intelpress/usb/examples/VBOverview.htm, 12 de gener de 2008)

LAFEBRE, G. Conectar un pic al PC por USB. Ecuador. (http://www.4shared.com/file/3028918/bd65dcc9/conectando_un_pic_al_pc_con_el_usb.html 24 d'octubre de 2007)

LAFEBRE, G. Ejemplos PIC-PC USB. (http://www.freewebs.com/glafebre/tp2550.htm, 24 d'octubre de 2007)

LAFEBRE, G. Robot Eleazar. Ecuador. (http://www.freewebs.com/glafebre/eleazar.htm, 25 de novembre de 2007)

MICROCHIP TECHNOLOGY INC. Paquet de desenvolupament per a aplicacions amb microcontrolador PIC. MPLAB IDE v8.10. USA 2008. (http://ww1.microchip.com/downloads/en/DeviceDoc/39632D.pdf)

MICROCHIP TECHNOLOGY INC. PIC18F2550 datasheet. USA. (http://ww1.microchip.com/downloads/en/DeviceDoc/39632D.pdf, 15 d'octubre de 2007)

NATIONAL SEMICONDUCTOR CORPORATION. LM2676 datasheet. USA. (http://cache.national.com/ds/LM/LM2676.pdf, 2 d'abril de 2008)

NATIONAL SEMICONDUCTOR CORPORATION. Programa Webench per a dissenyar fonts d'alimentació. USA. (http://www.national.com/appinfo/power/webench.html, 2 d'abril de 2008)

OLIVER, A. Apunts i pràctiques de robots i manipuladors industrials. Universitat de Girona. Girona. 2006.

ON SEMICONDUCTOR CORP. 1N5822 datasheet. USA (http://www.onsemi.com/pub_link/Collateral/1N5820-D.PDF, 3 d'abril de 2008)

REIXACH, P., FIGUERAS, A. Pràctiques aplicacions industrials dels microprocessadors. Universitat de Girona. Girona. 2006.

RS AMIDATA. Catàleg 2006/2007. Madrid. 2006.

SALAS, W. Programadores de microcontroladores PIC. (http://wsalas.blogspot.com/, 2 de novembre de 2007)

SALAS, W. USB PIC bootloader Boot25. (http://wsalas.blogspot.com/, 2 de novembre de 2007, 2 de novembre de 2007)

TODOPIC. Sobre el 18F2550 y un bootloader por USB. Argentina. (http://todopic.mforos.com/46840/3412972-sobre-el-18f2550-y-un-bootloader-por-usb/, 3 d'abril de 2008)

WIKIPEDIA. Directiva RoHS. (http://es.wikipedia.org/wiki/Rohs, 17 de maig de 2008)

WIKI-ROBOTICS. Hardware. (http://www.iearobotics.com/wiki/index.php?title=Portada#Hardware, 31 d'octubre de 2007)

WÜRTH ELEKTRONIK GmbH&Co. WE-PD 7447709220 datasheet. (http://www.wuerth-elektronik.de/eisos/pdf/7447709220.pdf, 3 d'abril de 2008)

9 GLOSSARI

PIC: Programmable Interrupt Controller (Controlador d'Interrupcions Programable)

LED: Light Emitting Diode (Diode Emissor de Llum)

USB: Universal Serial Bus (Bus Sèrie Universal)

Plug&Play: Endollar i utilitzar

AI: Artificial Intelligence (Intel·ligència Artificial)

PWM: Pulse Width Modulation (Modulació per Amplada de Pols)

Yaw: Virar

Pitch: Inclinació

Roll: Rodar

A. CÀLCULS I PROGRAMES

A.1. Càlculs mecànics

En aquest apartat es descriuran els càlculs realitzats per a determinar el pes que pot aixecar el braç robot i fins a quina distància és capça d'operar amb ells. Ens servirà per a determinar punts on és millor no arribar, i també amb quina posició és millor, ja que tenim múltiples possibilitats d'arribar a un lloc de diferents maneres.

La fórmula que utilitzarem és la del parell mecànic, tal i com s'expressa en l'equació 3:

$$\tau = F \times d \qquad \text{(Eq. 3)}$$

On:
τ: parell mecànic expressat en N×m ó en g×cm
F: és la força expressada en N ó en g
d: és la distància expressada en m ó en cm

Caldrà tenir en compte d'utilitzar les mateixes unitats quan es realitzin els càlculs. A efectes de comoditat tots els càlculs aquí reflectits han estat realitzats utilitzant com a unitats de parell g×cm.

El següent pas és determinar el parell que ens donen els diferents servomotors. Segons les especificacions dels fabricants els valors trobats són els reflectits a la taula 12:

MODEL	Articulació	PARELL
DYS0213	Base	14,0 kg×cm
DYS0213	Espatlla	14,0 kg×cm
DYS0213	Colze	14,0 kg×cm
HS-422	Canell	4,1 kg×cm
HS-422	Mà	4,1 kg×cm
DYS0205	Pinça	1,3 kg×cm

Taula 12. Parell dels servomotors

També serà necessari realitzar un diagrama esquemàtic del braç robot situant les masses i parells de força amb les seves respectives distàncies fins al punt que sigui objecte del càlcul. Les masses de les diferents parts del braç les trobem a la taula següent:

Part	**Massa (g)**
Servomotor DYS0213	55,0
Servomotor HS-422	45,5
Suport avantbraç	210,0
Suport braç	190,0
Suport pinça, pinça, HS-422, DYS0205	99,5

Taula 13. Masses

A.1.1. CAS MÉS DESFAVORABLE

El primer cas que s'analitzarà serà el més desfavorable. Aquest és amb el braç completament estirat i el motor que ha de vèncer un parell més gran és el que es troba a l'articulació de l'espatlla.

A la figura següent trobem un diagrama esquemàtic del braç robot amb les diferents masses i distàncies entre elles i els parells dels motors. Per als servomotors s'ha considerat el centre de massa en el punt de l'eix de gir, en les estructures de l'avantbraç i del braç el centre de masses és al punt mig de la seva llargada i en la pinça el centre de masses es troba pròxim a l'extrem del servomotor HS-422.

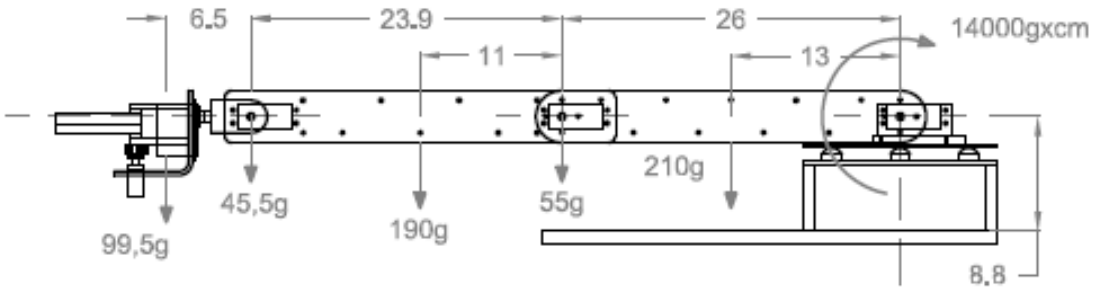

Figura 26. Braç en posició extrema

De l'anterior informació n'extraiem la informació necessària per a calcula el parell màxim que ha de suportar el servomotor de l'espatlla, i que apliquem a la equació 4.

$$\tau = \sum (F \times d) \qquad \text{(Eq. 4)}$$

A l'equació 5 es pot observar el sumatori dels diferents parells provocats per cadascuna de les masses i les distàncies horitzontals respecte el servomotor de l'articulació de l'espatlla.

$$\begin{aligned} \tau = {} & (210g \times 13cm) + (55g \times 26cm) + (190g \times 36{,}9cm) + (45{,}5g \times 49{,}9cm) + \\ & (99{,}5g \times 56{,}4cm) = 2730\,g \times cm + 1430\,g \times cm + 7011\,g \times cm + \\ & 2270{,}45\,g \times cm + 5611{,}8\,g \times cm = 19053{,}25\,g \times cm \end{aligned} \qquad \text{(Eq. 5)}$$

S'observa que el parell a vèncer és més gran que el que pot suportar el servomotor, que és de 14000g×cm, per tant en aquests marges de treball el braç robot no podrà operar. S'ha comprovat però que el motor pot suportar el pes del braç en aquesta posició.

A.1.2. BRAÇ EN POSICIÓ VERTICAL

El segon cas a analitzar és amb el braç en posició vertical a 90º, tal i com mostra la figura 27.

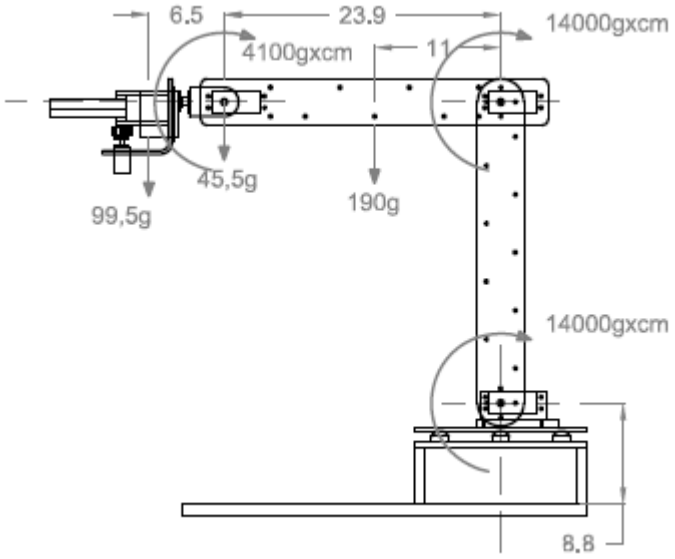

Figura 27. Braç en posició extrema

Començarem calculant el parell que ha de suportar el servomotor de la base segons l'equació 6. Cal tenir en compte, que la massa de l'estructura de l'avantbraç i del servomotor de l'articulació del colze no es tenen en compte, ja que la distància horitzontal d'aquestes respecte el servomotor de l'espatlla és zero.

$$\tau = (190g \times 10{,}9cm) + (45{,}5g \times 23{,}9cm) + (99{,}5g \times 30{,}4cm) = \\ 2071\ g \times cm\ + 1087{,}45\ g \times cm\ + 3024{,}8\ g \times cm\ = 6183{,}25\ \ g \times cm \quad \text{(Eq. 6)}$$

El model de servomotor DYS0213, que és el que es troba en aquesta articulació, pot suportar un parell de fins a 14000 g×cm, per tant encara pot suportar el parell addicional resultant de l'equació 7.

$$\tau = 14000\ g \times cm\ -\ 6183{,}25\ \ g \times cm = 7816{,}75\ g \times cm \quad \text{(Eq. 7)}$$

Aquest parell addicional que encara pot suportar és el que ens permet afegir una massa col·locada a l'extrem de la pinça que seria l'objecte que ens interessa desplaçar amb el braç. Aïllant la massa de la fórmula del parell, tal i com mostra l'equació 8, obtindrem la massa màxima que podrem aguantar i/o desplaçar en aquestes condicions.

$$m = \frac{\tau}{d} = \frac{7816{,}75\ g \times cm}{30{,}4\ cm} = 257{,}13\ g \quad \text{(Eq. 8)}$$

El braç robot en aquestes condicions ens permetrà desplaçar o aguantar un objecte de massa màxima la resultant de l'equació anterior. El servomotor del colze, en aquest cas, i al tenir les mateixes característiques que el de l'espatlla, podrà treballar amb una massa idèntica, ja que les variables de l'equació per al càlcul són exactament les mateixes.

Comprovem ara, si el servomotor del canell podrà suportar una massa més gran o més petita, ja que es troba en una posició diferent i és d'unes característiques també diferents. Els passos a seguir són els mateixos que anteriorment, però amb les condicions d'aquest nou punt de treball tal i com mostra l'equació 9.

$$\tau = 99{,}5g \times 6{,}5\ cm = 646{,}75\ \ g \times cm \quad \text{(Eq. 9)}$$

Aquest és el parell que ha de suportar només de l'estructura i dels servomotors de la pinça. Busquem ara el parell addicional que podrà manejar i la massa amb què podrà treballar amb les equacions 10 i 11 respectivament.

$$\tau = 4100\ g\times cm\ -\ 646{,}75\ \ g\times cm = 3453{,}25\ g\times cm \qquad \text{(Eq. 10)}$$

$$m = \frac{\tau}{d} = \frac{3453{,}25\ g\times cm}{6{,}5\ cm} = 531{,}27\ g \qquad \text{(Eq. 11)}$$

La massa que pot suportar a la pinça és superior, per tant agafarem com a valor límit el més restrictiu que en aquest cas és per la massa màxima que poden suportar els servomotors de l'espatlla i del colze que és de 257,13 g.

A.1.3. AVANBRAÇ EN POSICIÓ VERTICAL

El tercer cas que s'analitzarà serà amb el braç inclinat 55º respecte la base, l'avantbraç en posició totalment vertical i la pinça perpendicularment a aquest.

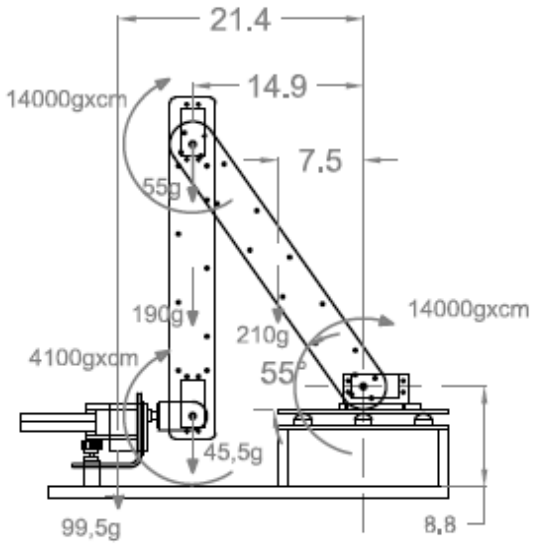

Figura 28. Avantbraç en posició vertical

En aquesta posició cal calcular la distància horitzontal del centre de masses de l'estructura de l'avantbraç fins el servomotor trigonomètricament, ja que el centre continua estan al punt mig de l'avantbraç i aquest ara es troba inclinat 55º.

A l'igual que el cas anterior el primer pas és calcular el parell que ha de suportar el servomotor de la base segons l'equació 12.

$$\begin{aligned}\tau &= (210\,g\times 7{,}5\,cm)+(55\,g\times 14{,}9\,cm)+(190\,g\times 14{,}9\,cm)\ + \\ &(45{,}5\,g\times 14{,}9\,cm)+(99{,}5g\times 21{,}4cm)=1575\ g\times cm\ +819{,}5\,g\times cm\ + \\ &2831\,g\times cm\ +677{,}95\,g\times cm+2129{,}3\,g\times cm=8032{,}75\ g\times cm\end{aligned} \quad \text{(Eq. 12)}$$

El model de servomotor DYS0213, que és el que es troba en aquesta articulació, pot suportar un parell de fins a 14000 g×cm, per tant encara pot suportar el parell addicional resultant de l'equació 13.

$$\tau = 14000\,g\times cm\ -\ 8032{,}75\ g\times cm = 5967{,}25\,g\times cm \quad \text{(Eq. 13)}$$

Altre cop, aquest parell addicional que encara pot suportar és el que ens permet afegir una massa col·locada a l'extrem de la pinça. Aïllant la massa de la fórmula del parell, tal i com mostra l'equació 14, obtindrem la massa màxima que podrem aguantar i/o desplaçar en aquestes condicions.

$$m = \frac{\tau}{d} = \frac{5967{,}25\,g\times cm}{20{,}4\,cm} = 292{,}51\,g \quad \text{(Eq. 14)}$$

Aquesta massa és lleugerament superior que l'obtinguda en el cas anterior. Com més s'acosti la pinça amb l'objecte a la base del braç, més gran serà la massa que podrà desplaçar.

Comprovem ara, si el servomotor del colze podrà suportar una massa més gran o més petita, en l'equació 15. Les masses de l'estructura del braç i del servomotor del canell en aquest cas no influeixen ja que la distància horitzontal respecte el punt actual de treball és zero.

$$\tau = 99{,}5g\times 6{,}5cm = 646{,}75\ g\times cm \quad \text{(Eq. 15)}$$

Comprovem ara quina és la massa que pot aguantar amb les equacions 16 i 17.

$$\tau = 14000\ g \times cm\ -\ 646{,}75\ \ g \times cm = 13353{,}25\ g \times cm \qquad \text{(Eq. 16)}$$

$$m = \frac{\tau}{d} = \frac{13353{,}25\ g \times cm}{6{,}5\ cm} = 2054{,}35\ g \qquad \text{(Eq. 17)}$$

Finalment la massa que podrà aguantar el servomotor del canell és la mateixa que en el cas anterior 531,27. Per tant, el cas més restrictiu torna a ser per al servomotor de l'espatlla.

A.2. Codi programació PIC18F2550

El codi del programa de la placa de control creat amb el programa Mikrobasic 4.0.0.3 i implementat al PIC18F2550 a través del programa MPLAB IDE v8.10 amb l'arxiu MABLE_CONTROL_V1.0.hex en format hexadecimal.

```
program MABLE_CONTROL_V1.0


'Controladora                MABLE                CONTROL                v1.0
'*************************************************************
'per a Mikrobasic 4.0.0.3
'
'Creat per: Gerard Massaguer
'gMaSsa
'
'-HARDWARE:
'    MCU: PIC 18F2550
'    Cristall: 8 Mhz
'    Sortides per a 8 servos al PORTB
'*************************************************************************
*****

include "USBdsc"

dim Tmr_0 as Word absolute $0FD6
dim Tmr_1 as Word absolute $0FCE

const Cbytes as byte = 64

dim userWR_buffer  as byte[Cbytes]
dim userRD_buffer  as byte[Cbytes]
dim Dato           as byte[Cbytes]
dim Pos            as byte[8]

dim Servo          as byte
dim TmpServo       as byte
dim Flg            as Boolean
dim sRef           as byte
dim TopeMax        as byte
dim e              as byte
dim i              as byte
dim UsByte         as byte
dim TmpRead        as byte
dim AnalogI        as word
dim wr_adr         as word
dim aa as byte

'Rutina d'interrupcio
sub procedure interrupt
  If TestBit(PIR2,USBIF) = 1 then
    HID_InterruptProc
  End if
```

```
  'Bandera de la interrucio del timer 0
  If TestBit(INTCON,TMR0IF)=1 then
    Flg = true

    Tmr_1 = -59               'Timer 1
    PIR1.TMR1IF = 0
    T1CON.TMR1ON = 1

    INTCON.TMR0IF = 0         'Timer 0 (on)

    LATB.Servo=1              'On Servo
  End if
end sub


'Rutina d'inicialitzacio
sub procedure Init_Main
      INTCON  =  0                                  'Deshabilita  GIE,  PEIE,
TMR0IE,INT0IE,RBIE
      INTCON2 = 0xF5
      INTCON3 = 0xC0
      RCON.IPEN = 0                 'Deshabilita el nivell de Prioritat de
les interrupcions
      PIE1 = 0
      PIE2 = 0
      PIR1 = 0
      PIR2 = 0

  CMCON = 7
  ADCON1 = %10000000

      'Ports
      TRISA = %11111111           'PORTA son entradas
      TRISB = %00000000           'PORTB sortides als servos
      TRISC = %00000000           'PORTC sortides

  'Reset dels ports
  PORTA=0
  PORTB=0
  PORTC=0

      LATA = 0
      LATB = 0
      LATC = 0
end sub

'Programa principal

main:
  TmpServo=0
  sRef=0
  TopeMax=0

  Init_Main

  HID_Enable(@userRD_buffer, @userWR_buffer)

  Delay_mS(2000)             'Esperem un parell de segons
```

```
 'Reset de les posicions dels servos (llegeix les posicions inicials dels
servos a la eeprom)

  for e=0 to 7
    If Eeprom_Read(e) = 0 then
     Eeprom_Write(e, 127)
    end if
    Pos[e] = Eeprom_Read(e)
  next e

  For e=0 to 64                'Reset de les dades de I/O
    userWR_buffer[e]=0
    userRD_buffer[e]=0
    Dato[e]=0
  next e
  Servo=0


  T0CON =  %01000011           'Establim el Timer 0 (Preescaler 1:16)
  INTCON = %10100100
  Tmr_0 = 103                   '1220 ms

  T1CON = 0                    'Borrem el registre del timer1 T1CON
  Tmr_1 = -59                  '0'236 ms (minim per els servos)
  PIE1 = 0                              'TMR1 i les interrupcions externes
deshabilitades
  PIR1 = 0
  T1CON = %00110000            'Establim la configuracio del timer 1
  INTCON.TMR0IF = 0
  T0CON.TMR0ON=1               'timer 0 en marxa

  LATC.0= 1                    'LED de POWER
  LATC.7= 1                    'Habilitem el RELE
  while true
      UsByte = HID_Read
      i = 0
      while i < UsByte
        TmpRead = userRD_buffer[i]
        inc(i)
      wend

      Dato[TmpServo] = userRD_buffer[TmpServo]
      Inc(TmpServo)

      TopeMax=3
      If (Dato[0]=65) OR (Dato[0]=80) then TopeMax=2 End if
      If (Dato[0]=73) OR (Dato[0]=83) then TopeMax=3 End if
      If Dato[0]=84 then TopeMax=9 End if

      if TmpServo > TopeMax then
        TmpServo=0

        if (Dato[0] = 65) and (Dato[2]=35) then        'Enviem el valor de
les entrades analogiques (comanda "A")
          LATC.1 = 1                                    'Encenem el led de
comanda rebuda
          Dato[0] = 0                                 'Reset de la dada 0
          userRD_buffer[0]=0                              'Reset del buffer
d'entrada
          sRef = Byte(Dato[1])                                'Referencia a
l'entrada que solicitem "dato(1)"
```

```
          AnalogI = Adc_Read(sRef)
          userWR_buffer[0] = byte(AnalogI >> 2)        'Emplenem el buffer
de sortida amb el valor 8 bits analogic
          HID_Write(@userWR_buffer, 1)                'Enviem la dada
        End if

        if (Dato[0] = 80) and (Dato[2]=35) then        'Enviem la posicio
del servo sRef
          LATC.1 = 1                                   'Encenem el led de
comanda rebuda
          Dato[0] = 0                                 'Reset de la dada 0
          userRD_buffer[0]=0                            'Reset del buffer
d'entrada
          sRef = Byte(Dato[1])                         'Referencia al servo
"dato(1)"
          userWR_buffer[0] = Pos[sRef]                  'emplenem el buffer
de sortida amb la posicio del servo
          HID_Write(@userWR_buffer, 1)                'Enviem la dada
        end if

        If (Dato[0]=73) and (Dato[3]=35) then           'Rebem la posicio
d'inici comanda "I"
          LATC.1 = 1                                   'Encenem el led de
comanda rebuda
          Dato[0] = 0                                 'Reset de la dada 0
          userRD_buffer[0]=0                            'Reset del buffer
d'entrada
          sRef = Byte(Dato[1])                        'Referencia al servo
          Pos[sRef] = Byte(Dato[2])                    'Capturem la posicio
que arriba
          Eeprom_Write(sRef,Pos[sRef])                 'Grabem la posicio
d'inici del servo
        end if

        if (Dato[0]=83) and (Dato[3]=35) then         'Rep la posicio d'un
servo comanda "S"
          LATC.1 = 1                                   'Encenem el led de
comanda rebuda
          Dato[0] = 0                                 'Reset de la dada 0
          userRD_buffer[0]=0                            'Reset del buffer
d'entrada
          sRef =Byte(Dato[1])                         'Referencia al servo
          Pos[sRef] = Byte(Dato[2])                    'Capturem la posicio
que arriba
        end if

        if (Dato[0]=84) and (Dato[9]=35) then          'Si rep la comanda
"T" significa tots els servos
          LATC.1 = 1                                   'Encenem el led de
comanda rebuda
          Dato[0] = 0                                 'Reset de la dada 0
          userRD_buffer[0]=0                            'Reset del buffer
d'entrada
          'Posicio dels servos
          For aa=0 to 7
            Pos[aa] = Byte(Dato[aa+1])
          next aa
        end if
      end if
```

```
      If Flg Then                                          'Es detecta la
bandera del Timer 0
        While PIR1.TMR1IF = 0
        Wend
        Tmr_1= -(Pos[Servo] * 1)
        PIR1.TMR1IF = 0                                   'Posem el Timer 2 en
marxa
        While PIR1.TMR1IF = 0
        Wend
        T1CON.TMR1ON = 0                                 'Aturem el Timer 2

        LATB.Servo=0                                     'Servo a Off
        Inc(Servo)
        if Servo > 7 then
          Servo=0
          LATC.1 = 0                                        'Apaguem el led de
comanda rebuda
        end if
        Flg = false
      End If
  wend

  HID_Disable
  'Aquest codi es necessari per carregar les variables de la rutina
d'interrupció HID_InterruptProc.
  'En temps real mai s'executarà
  if false then
      HID_InterruptProc
      wr_adr = @DeviceDescr
      wr_adr = @ConfigDescr
      wr_adr = @InterfaceDescr
      wr_adr = @HID_Descriptor
      wr_adr = @EP1_RXDescr
      wr_adr = @EP1_TXDescr
      wr_adr = @HID_ReportDesc
      wr_adr = @LangIDDescr
      wr_adr = @ManufacturerDescr
      wr_adr = @ProductDescr
      wr_adr = @StrUnknownDescr
  end if
end.
```

A.3. Codi de la llibreria inclosa USBdsc.bas

Tot seguit inserim el codi del mòdul de llibreria inclosa dins el programa de control del PIC18F2550 i que permet la comunicació via USB del microcontrolador amb el programa en Visual Basic .NET. La llibreria s'anomena USBdsc.bas.

```
module USBdsc
include "HIDconstant"

'*************************************************************************
****
' The number of bytes in each report,
' calculated from Report Size and Report Count in the report descriptor
'*************************************************************************
****

 const HID_INPUT_REPORT_BYTES      = 1
 const HID_OUTPUT_REPORT_BYTES     = 10

 const HID_FEATURE_REPORT_BYTES    = 2

'*************************************************************************
****

' Byte constants

'*************************************************************************
****

 const NUM_ENDPOINTS               = 2
 const   ConfigDescr_wTotalLength          =   USB_CONFIG_DESCRIPTOR_LEN   +
USB_INTERF_DESCRIPTOR_LEN   +   USB_HID_DESCRIPTOR_LEN   +   (NUM_ENDPOINTS   *
USB_ENDP_DESCRIPTOR_LEN)
 const HID_ReportDesc_len          = 47

 const Low_HID_ReportDesc_len      = HID_ReportDesc_len
 const High_HID_ReportDesc_len     = HID_ReportDesc_len >> 8

 const Low_HID_PACKET_SIZE         = HID_PACKET_SIZE
 const High_HID_PACKET_SIZE        = HID_PACKET_SIZE >> 8

'*************************************************************************
****

' Descriptor Tables

'*************************************************************************
****

 const DeviceDescr as byte[USB_DEVICE_DESCRIPTOR_LEN*2] = (
    USB_DEVICE_DESCRIPTOR_LEN, 0,              ' bLength                    -
Length of Device descriptor (always 0x12)
    USB_DEVICE_DESCRIPTOR_TYPE, 0,           ' bDescriptorType        - 1 =
DEVICE descriptor
```

```
    0x00, 0,                                    ' bcdUSB                - USB
revision 2.00 (low byte)
    0x02,  0,                                                           '
(high byte)
    0x00, 0,                                   ' bDeviceClass          - Zero
means each interface operates independently (class code in the interface
descriptor)
    0x00, 0,                                  ' bDeviceSubClass
    0x00, 0,                                  ' bDeviceProtocol
    EP0_PACKET_SIZE, 0,                           ' bMaxPacketSize0       -
maximum size of a data packet for a control transfer over EP0
    0x34, 0,                                      ' idVendor              -
Vendor  ID (low byte)
    0x12,  0,                                                           '
(high byte)
    0x01, 0,                                      ' idProduct             -
Product ID (low byte)
    0x00,  0,                                                           '
(high byte)
    0x01, 0,                                   ' bcdDevice             - ( low
byte)
    0x00, 0,                                  '                          (high
byte)
    0x01, 0,                                       ' iManufacturer         -
String1
    0x02, 0,                                      ' iProduct              -
String2
    0x00, 0,                                     ' iSerialNumber         - (
None )
    0x01, 0                                   ' bNumConfigurations    - 1
)

'************************************************************************
****

 const ConfigDescr as byte[USB_CONFIG_DESCRIPTOR_LEN*2] = (
    USB_CONFIG_DESCRIPTOR_LEN, 0,               ' bLength               -
Length of Configuration descriptor (always 0x09)
    USB_CONFIG_DESCRIPTOR_TYPE, 0,            ' bDescriptorType       - 2 =
CONFIGURATION descriptor
    ConfigDescr_wTotalLength, 0,             ' wTotalLength          - Total
length  of  this  config.  descriptor  plus  the  interface  and  endpoint
descriptors that are part of the configuration.
    0x00, 0,                                   '                          (
high byte)
    0x01, 0,                                      ' bNumInterfaces        -
Number of interfaces
    0x01, 0,                                       ' bConfigurationValue   -
Configuration Value
    0x00, 0,                                      ' iConfiguration        -
String Index for this configuration ( None )
    0xA0, 0,                                      ' bmAttributes          -
attributes - "Bus powered" and "Remote wakeup"
    50, 0                                     ' MaxPower              - bus-
powered draws 50*2 mA from the bus.
)

'************************************************************************
****
```

```
 const InterfaceDescr as byte[USB_INTERF_DESCRIPTOR_LEN*2] = (
    USB_INTERF_DESCRIPTOR_LEN, 0,              ' bLength                    -
Length of Interface descriptor (always 0x09)
    USB_INTERFACE_DESCRIPTOR_TYPE, 0,        ' bDescriptorType        - 4 =
INTERFACE descriptor
    0x00, 0,                                    ' bInterfaceNumber       -
Number of interface, 0 based array
    0x00, 0,                                    ' bAlternateSetting      -
Alternate setting
    NUM_ENDPOINTS, 0,                          ' bNumEndPoints          -
Number of endpoints used in this interface
    0x03, 0,                                    ' bInterfaceClass        -
assigned by the USB
    0x00, 0,                                 ' bInterfaceSubClass    - Not A
boot device
    0x00, 0,                                ' bInterfaceProtocol    - none
    0x00, 0                                 ' iInterface            - Index
to string descriptor that describes this interface ( None )
)

'*************************************************************************
****

 const HID_Descriptor as byte[USB_HID_DESCRIPTOR_LEN*2] = (
    USB_HID_DESCRIPTOR_LEN, 0,                ' bLength                    -
Length of HID descriptor (always 0x09)
    USB_HID_DESCRIPTOR_TYPE, 0,              ' bDescriptorType        - 0x21
= HID descriptor
    0x01, 0,                                    ' HID class release number
(1.01)
    0x01, 0,
    0x00, 0,                                ' Localized country code (none)
    0x01, 0,                                 ' # of HID class descriptor to
follow (1)
    0x22, 0,                                ' Report descriptor type (HID)
    Low_HID_ReportDesc_len, 0,
    High_HID_ReportDesc_len, 0
)

'*************************************************************************
****

 const EP1_RXDescr as byte[USB_ENDP_DESCRIPTOR_LEN*2] = (
    USB_ENDP_DESCRIPTOR_LEN, 0,                ' bLength                    -
length of descriptor (always 0x07)
    USB_ENDPOINT_DESCRIPTOR_TYPE, 0,         ' bDescriptorType        - 5 =
ENDPOINT descriptor
    0x81, 0,                                  ' bEndpointAddress       - In,
EP1
    USB_ENDPOINT_TYPE_INTERRUPT, 0,            ' bmAttributes             -
Endpoint Type - Interrupt
    Low_HID_PACKET_SIZE, 0,                  ' wMaxPacketSize         - max
packet size - low order byte
    High_HID_PACKET_SIZE, 0,                 '                        - max
packet size - high order byte
    1, 0                                       ' bInterval              -
polling interval (1 ms)
)

'*************************************************************************
****
```

```
 const EP1_TXDescr as byte[USB_ENDP_DESCRIPTOR_LEN*2] = (
    USB_ENDP_DESCRIPTOR_LEN, 0,                ' bLength                   -
length of descriptor (always 0x07)
    USB_ENDPOINT_DESCRIPTOR_TYPE, 0,          ' bDescriptorType         - 5 =
ENDPOINT descriptor
    0x01, 0,                                  ' bEndpointAddress       - Out,
EP1
    USB_ENDPOINT_TYPE_INTERRUPT, 0,             ' bmAttributes             -
Endpoint Type - Interrupt
    Low_HID_PACKET_SIZE, 0,                   ' wMaxPacketSize          - max
packet size - low order byte
    High_HID_PACKET_SIZE, 0,                  '                         - max
packet size - high order byte
    1, 0                                        ' bInterval               -
polling interval (1 ms)
)

'*********************************************************************
****

 const HID_ReportDesc as byte[HID_ReportDesc_len*2] = (
    0x06, 0,                                  ' USAGE_PAGE (Vendor Defined)
    0xA0, 0,
    0xFF, 0,
    0x09, 0,                                  ' USAGE ID (Vendor Usage 1)
    0x01, 0,
    0xA1, 0,                                  ' COLLECTION (Application)
    0x01, 0,
'  The Input report
    0x09, 0,                                  ' USAGE ID - Vendor defined
    0x03, 0,
    0x15, 0,                                  '   LOGICAL_MINIMUM (0)
    0x00, 0,
    0x26, 0,                                  '   LOGICAL_MAXIMUM (255)
    0x00, 0,
    0xFF, 0,
    0x75, 0,                                  '   REPORT_SIZE (8)
    0x08, 0,
    0x95, 0,                                  '   REPORT_COUNT (2)
    HID_INPUT_REPORT_BYTES, 0,
    0x81, 0,                                  '   INPUT (Data,Var,Abs)
    0x02, 0,
'  The Output report
    0x09, 0,                                  ' USAGE ID - Vendor defined
    0x04, 0,
    0x15, 0,                                  '   LOGICAL_MINIMUM (0)
    0x00, 0,
    0x26, 0,                                  '   LOGICAL_MAXIMUM (255)
    0x00, 0,
    0xFF, 0,
    0x75, 0,                                  '   REPORT_SIZE (8)
    0x08, 0,
    0x95, 0,                                  '   REPORT_COUNT (2)
    HID_OUTPUT_REPORT_BYTES, 0,
    0x91, 0,                                  '   OUTPUT (Data,Var,Abs)
    0x02, 0,
'  The Feature report
    0x09, 0,                                  ' USAGE ID - Vendor defined
    0x05, 0,
    0x15, 0,                                  '   LOGICAL_MINIMUM (0)
    0x00, 0,
```

```
    0x26, 0,                                    '   LOGICAL_MAXIMUM (255)
    0x00, 0,
    0xFF, 0,
    0x75, 0,                                    '   REPORT_SIZE (8)
    0x08, 0,
    0x95, 0,                                    '   REPORT_COUNT (2)
    HID_FEATURE_REPORT_BYTES, 0,
    0xB1, 0,                                    '   FEATURE (Data,Var,Abs)
    0x02, 0,
'  End Collection
    0xC0, 0                                     ' END_COLLECTION
)

'*************************************************************************
****

 const LangIDDescr as byte[8] = (
    0x04, 0,
    USB_STRING_DESCRIPTOR_TYPE, 0,
    0x09, 0,                                    ' LangID (0x0409) - Low
    0x04, 0                                     '                 - High
)

'*************************************************************************
****

 const ManufacturerDescr as byte[16] = (
    8, 0,
    USB_STRING_DESCRIPTOR_TYPE, 0,
    "g", 0, 0, 0,
    "M", 0, 0, 0,
    "a", 0, 0, 0,
    "S", 0, 0, 0,
    "s", 0, 0, 0,
    "a", 0, 0, 0
)

'*************************************************************************
****
 const ProductDescr as byte[100] = (
    50, 0,
    USB_STRING_DESCRIPTOR_TYPE, 0,
    "M", 0, 0, 0,
    "a", 0, 0, 0,
    "b", 0, 0, 0,
    "l", 0, 0, 0,
    "e", 0, 0, 0,
    " ", 0, 0, 0,
    "C", 0, 0, 0,
    "o", 0, 0, 0,
    "n", 0, 0, 0,
    "t", 0, 0, 0,
    "r", 0, 0, 0,
    "o", 0, 0, 0,
    "l", 0, 0, 0,
    " ", 0, 0, 0,
    "v", 0, 0, 0,
    "1", 0, 0, 0,
    ".", 0, 0, 0,
    "0", 0, 0, 0
)
```

```
'*************************************************************************
****

 const StrUnknownDescr as byte[4] = (
    2, 0,
    USB_STRING_DESCRIPTOR_TYPE, 0
)

'*************************************************************************
****

'*************************************************************************
****

' Initialization Function

'*************************************************************************
****

implements
sub procedure InitUSBdsc
  asm
    _InitUSBdsc:
  end asm
end sub
end.
```

A.4. Programa de control per l'ordinador

A.4.1. FINESTRA PRINCIPAL

El codi de la finestra principal del programa Mable Control realitzat amb Microsoft Visual Bàsic 6.0 per al control del braç robot des de l'ordinador.

```
Option Explicit
Dim a As Integer
Dim CargoRutines As Boolean
Dim Valor1(64) As Byte
Dim Valor2(64) As Byte
Private Const Placa As String = "MABLE_CONTROL_v1.0"

Function DetectarUSB() As Boolean

'Encenem el timer de timeout
TimeOut = False
TimerOut.Enabled = True

'Identifiquem la controladora
While OpenUSBdevice(Placa) = False And TimeOut = False
    DoEvents
Wend

'Si s'habilita el TimeOut pot ser perque no es detecta la placa
If TimeOut Then
    Missatge.Accepto = False
    Missatge.Caption = "Detecció de TimeOut, No s'ha detectat la
controladora Mable Control v1.0"
    Missatge.Temporitzat = True
    FrmMissatge.Show vbModal
Else
    DetectarUSB = True
End If

End Function

Private Sub LlegirPosicions_USB()

'Variable de deteccio de la placa controladora
PlacaDetectada = False

If DetectarUSB Then
    'Llegim la posicio d'inici dels servos
    Valor1(0) = Asc("P")                                    'Capçalera
    Valor1(2) = Asc("#")                                        'Codi fi
d'enviament
    For a = 0 To 7
        Valor1(1) = a                                       'Nº de Servo al
que accedim
        'Escribim la posicio
        Call WriteUSBdevice(AddressFor(Valor1(0)), 10)
        'Retorna la posicio a Valor2
        Call ReadUSBdevice(AddressFor(Valor2(a)), 1)
```

```
        'Grabem el valor de la posicio a la memoria d'intercanvi
        Servo(a).Posicio = Valor2(a)

        'Si pasa aqui es que ha detectat la placa
        PlacaDetectada = True

        'Presentem la posicio en l'etiqueta de posicio de cada servo
        Lval(a).Caption = Trim(Str(Servo(a).Posicio))
    Next a

    'S'ha detectat la placa axi que escanegem les entrades
    TimerAnalog.Enabled = True
End If

End Sub



Private Sub Command1_Click()
'Comprobem que ha detectat la placa
If PlacaDetectada = False Then Exit Sub

'Envio a tots els servos ****************************
Valor1(0) = Asc("T")                'capçalera

For a = 0 To VS1.Count - 1
    Valor1(a + 1) = VS1(a).Value    'Preparacio del buffer
Next a
Valor1(9) = Asc("#") '# = 35        'Codi fi d'enviament

'Escribim les dades al port USB
Call WriteUSBdevice(AddressFor(Valor1(0)), 10)
End Sub

Private Sub Command2_Click()
'Comprobem que ha detectat la placa
If PlacaDetectada = False Then Exit Sub

For a = 0 To VS1.Count - 1
    Valor1(0) = Asc("I")        'Capçalera
    Valor1(1) = a               'Nº de Servo
    Valor1(2) = VS1(a).Value    'Establim la dada de posicio
    Valor1(3) = Asc("#")        'Codi fi d'enviament
    'Escribim les dades al port USB
    Call WriteUSBdevice(AddressFor(Valor1(0)), 10)
Next a
End Sub

Private Sub Form_Load()

Seleccionat = 0

'Timer de timeout
TimerOut.Interval = 1000    '1 segon
TimerOut.Enabled = False
TimeOut = False

'Timer d'ecaneig per les entrades
TimerAnalog.Interval = 500  '500ms
TimerAnalog.Enabled = False
```

```
'Carreguem els topes maxims y minims dels servos
For a = 0 To 7
    Servo(a).Maxim = 255
    Servo(a).Minim = 1
    Servo(a).Inici = 127
    LNum(a).Caption = (a + 1)
Next a

'Llegim les posicions dels servos
Call LlegirPosicions_USB

'Establim els rangs dels cursors
For a = 0 To VS1.Count - 1
    VS1(a).Max = Servo(a).Maxim
    VS1(a).Min = Servo(a).Minim

    If Servo(a).Posicio > 0 Then VS1(a).Value = Servo(a).Posicio
Next a

'Bandera que mostra que es pot entrar als servos
CargoRutines = True
End Sub


Private Sub TimerAnalog_Timer()
'Comprobem que ha detectat la placa
If PlacaDetectada = False Then Exit Sub

'Llegim el valor de les entrades
Valor1(0) = Asc("A")                                    'Capçalera
Valor1(2) = Asc("#")                                        'Codi fi
d'enviament
For a = 0 To 4
    Valor1(1) = a                                          'Nº d'entrades a
les que accedim
    'Escribim la posicio
    Call WriteUSBdevice(AddressFor(Valor1(0)), 10)
    'Retorna la posicio a Valor2
    Call ReadUSBdevice(AddressFor(Valor2(a)), 1)

    'Escribim el valor rebut en l'etiqueta corresponent
    ValAnalog(a).Caption = Valor2(a)
Next a
End Sub

Private Sub TimerOut_Timer()
TimerOut.Enabled = False
TimeOut = True
End Sub

Private Sub VS1_Change(index As Integer)
'Si la bandera esta en fals no entra
If CargoRutines = False Then Exit Sub

'Comprobem que ha detectat la placa
If PlacaDetectada = False Then Exit Sub

Valor1(0) = Asc("S")                                'Capçalera d'enviament
Valor1(1) = index                                    'Nº de Servo al que fa
referencia
```

```
Valor1(2) = VS1(index).Value                          'Establim el valor que
volem al buffer
Valor1(3) = Asc("#")                                 'Codi de fi de trama
'Escribim el valor a l'etiqueta superior de l'slider
Lval(index).Caption = Valor1(2)

'Escribim el valor al port USB
Call WriteUSBdevice(AddressFor(Valor1(0)), 10)

End Sub

Private Sub VS1_Scroll(index As Integer)
Call VS1_Change(index)
End Sub
```

A.4.2. FINESTRA DE MISSATGE

El codi de la finestra de missatge d'error per no detecció de la placa del programa Mable Control realitzat amb Microsoft Visual Bàsic 6.0 per al control del braç robot des de l'ordinador.

```
Option Explicit
Dim Mouse_X As Single, Mouse_Y As Single


Private Sub CmdOk_Click()
Missatge.Accepto = True

Unload Me
End Sub

Private Sub CmdCancel_Click()
Missatge.Accepto = False
Unload Me
End Sub

Private Sub Form_Activate()
'Fa el formulari sempre visible
Call SetWindowPos(hWnd, -1, 0, 0, 0, 0, SWP_NOMOVE Or SWP_NOSIZE)
End Sub

Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = 13 Then Call CmdOk_Click
If KeyCode = 27 Then Unload Me
End Sub

Private Sub Form_Load()
Me.Caption = "Missatge"

Missatge.Accepto = False
Label1.Caption = Missatge.Caption

CmdOk.Visible = Not Missatge.Temporitzat
CmdCancel.Visible = Not Missatge.Temporitzat

TimerEnd.Interval = 2000
TimerEnd.Enabled = Missatge.Temporitzat

End Sub

Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As
Single, Y As Single)
Mouse_X = X: Mouse_Y = Y
End Sub
```

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As
Single, Y As Single)
If Button = 1 Then
    'Estableix l'incona d'arrastre
    Me.MousePointer = 5
    If X <> Mouse_X Then Me.Left = Me.Left - (Mouse_X - X)
    If Y <> Mouse_Y Then Me.Top = Me.Top - (Mouse_Y - Y)
End If
End Sub

Private Sub Form_MouseUp(Button As Integer, Shift As Integer, X As Single,
Y As Single)
Me.MousePointer = 0
End Sub

Private Sub Form_Unload(Cancel As Integer)
Set FrmMissatge = Nothing
End Sub

Private Sub Label1_MouseDown(Button As Integer, Shift As Integer, X As
Single, Y As Single)
Call Form_MouseDown(Button, Shift, X, Y)
End Sub

Private Sub Label1_MouseMove(Button As Integer, Shift As Integer, X As
Single, Y As Single)
Call Form_MouseMove(Button, Shift, X, Y)
End Sub

Private Sub Label1_MouseUp(Button As Integer, Shift As Integer, X As
Single, Y As Single)
Call Form_MouseUp(Button, Shift, X, Y)
End Sub

Private Sub TimerEnd_Timer()
Missatge.Accepto = False
Unload Me
End Sub
```

A.4.3. VARIABLES

Codi del mòdul Variables, declarades per el programa Mable Control realitzat amb Microsoft Visual Bàsic 6.0 per al control del braç robot des de l'ordinador, contingut al fitxer Variables.bas.

```
Option Explicit

Declare Function SetWindowPos Lib "user32" (ByVal hWnd As Long, ByVal
hWndInsertAfter As Long, ByVal X As Long, ByVal Y As Long, ByVal cx As
Long, ByVal cy As Long, ByVal wFlags As Long) As Long
Public Declare Function GetPrivateProfileString Lib "kernel32" Alias
"GetPrivateProfileStringA" (ByVal lpApplicationName As String, ByVal
lpKeyName As Any, ByVal lpDefault As String, ByVal lpReturnedString As
String, ByVal nSize As Long, ByVal lpFileName As String) As Long
Public Declare Function WritePrivateProfileString Lib "kernel32" Alias
"WritePrivateProfileStringA" (ByVal lpApplicationName As String, ByVal
lpKeyName As Any, ByVal lpString As Any, ByVal lpFileName As String) As
Long

Global Const SWP_NOMOVE = &H2 'per fer formularis sempre visibles
Global Const SWP_NOSIZE = &H1 'per fer formularis sempre visibles

'Parametres dels enviaments a la placa de control
Global Const Ids As String = ">"

'Indicador de que el port esta obert
Public PlacaDetectada As Boolean

'Matriu de servos
Private Type TipoServo
    Maxim As Integer
    Minim As Integer
    Posicio As Integer
    Inici As Integer   'Posicio d'inici
End Type
Public Servo(7) As TipoServo

Public Seleccionat As Integer 'Servo seleccionat

Private Type TipoMissatge
    Caption As String       'Text presentat
    Accepto As Boolean      'Accepta
    Temporitzat As Boolean   'Indica si si s'autotencara(no presentara
Botons)
End Type
Public Missatge As TipoMissatge

Public TimeOut As Boolean   'bandera de timeOut
```

A.4.4. MÒDUL ACCESIODEVICE

Codi del mòdul AccesIODevice utilitzat en el programa Mable Control i necessari per a la seva comunicació amb el microcontrolador a través del bus USB, contingut al fitxer HIDInterface.bas del programa creat amb Visual Basic 6.0.

```
'   This module is common to all of the Example programs
'   It declares the {Open, Read, Write, Close} calls for the USB device
'       These user-calls are translated into OS system calls
'    This module also contains several support routines used by all of the
examples
'
'   Declare module-wide variables
Private HidHandle As Long
Public Function OpenUSBdevice(NameOfDevice$) As Boolean
' This function searches the system HID tables for NameOfDevice$
' If found then it opens the device and returns TRUE, else it returns FALSE
Dim HidGuid As Guid: Dim Success As Boolean: Dim Openned As Boolean: Dim
Buffer(256) As Byte
Dim DeviceInterfaceData As Device_Interface_Data
Dim FunctionClassDeviceData As Device_Interface_Detail
'
'   First, get the HID class identifier
Call HidD_GetHidGuid(HidGuid.Data(0))
'    Get a handle for the Plug and Play node, request currently active HID
devices
PnPHandle& = SetupDiGetClassDevs(HidGuid.Data(0), 0, 0, &H12)
If (PnPHandle& = -1) Then ErrorExit ("Could not attach to PnP node")
'
HidEntry& = 0: Openned = False
DeviceInterfaceData.cbsize = 28 'Length of data structure in bytes
'   Look through the table of HID devices
Do   While   SetupDiEnumDeviceInterfaces(PnPHandle&,   0,   HidGuid.Data(0),
HidEntry&, DeviceInterfaceData.cbsize)
'   There is a device here, get it's name
    FunctionClassDeviceData.cbsize = 5
    Success          =          SetupDiGetDeviceInterfaceDetail(PnPHandle&,
DeviceInterfaceData.cbsize, _
           FunctionClassDeviceData.cbsize,
UBound(FunctionClassDeviceData.DataPath), BytesReturned&, 0)
    If (Success = 0) Then ErrorExit ("Could not get the name of this HID
device")
' Convert returned C string to Visual Basic String
    hidname$ = "": i& = 0
    Do While FunctionClassDeviceData.DataPath(i&) <> 0
        hidname$  =  hidname$  &  Chr$(FunctionClassDeviceData.DataPath(i&)):
i& = i& + 1: Loop
' Can now open this HID device
    Dim SA As Security_Attributes
    HidHandle& = CreateFile(hidname$, &HC0000000, 3, SA, 3, 0, 0)
    If (HidHandle = -1) Then ErrorExit ("Could not open HID device")
' Is it OUR HID device?
    If       HidD_GetProductString(HidHandle&,       AddressFor(Buffer(0)),
UBound(Buffer)) Then
        DeviceName$ = "": i& = 0
```

```
        Do While Buffer(i&) <> 0: DeviceName$ = DeviceName$ &
Chr$(Buffer(i&)): i& = i& + 2: Loop
        If (StrComp(DeviceName$, NameOfDevice$) = 0) Then
            Openned = True: Exit Do: End If
        End If 'HidD_GetProductString
    Call CloseHandle(HidHandle&) ' Was not OUR HID device
    HidEntry& = HidEntry& + 1 ' Check next entry
    Loop 'SetupDiEnumDeviceInterfaces returns FALSE when there are no more
entries
SetupDiDestroyDeviceInfoList (PnPHandle&)
OpenUSBdevice = Openned
End Function
Public Sub ReadUSBdevice(BufferPtr&, ByteCount&)
' This subroutine "reads" from an openned USB device
' This routine gets an Input Report from the USB device and returns the
data
' NOTE that ReadFile is a BLOCKING system call, ie it will wait for the USB
device to respond
' Do not configure the USB device to "Generate report only on change" since
the program
' will appear to 'hang'
' Use a local buffer so that the ReportID (=0) at ReportBuffer(0) may be
removed
Dim ReportBuffer(256) As Byte
If ByteCount& > 254 Then ErrorExit ("Maximum ByteCount for ReadUSBdevice is
254")
If ByteCount& < 1 Then ErrorExit ("Minimum ByteCount for ReadUSBdevice is
1")
Success = ReadFile(HidHandle&, AddressFor(ReportBuffer(0)), ByteCount& + 1,
BytesReturned&, 0)
If (Success = 0) Then
    ErrorExit ("Could not get an Input Report")
Else
    Call CopyBuffer(AddressFor(ReportBuffer(1)), BufferPtr&, BytesReturned&
- 1)
End If
End Sub
Public Sub WriteUSBdevice(BufferPtr&, ByteCount&)
' This subroutine "writes" to an openned USB device
' Copy the user buffer into a local buffer so that a ReportID (=0) may be
prepended
' The first byte will contain the ReportID (=0)
Dim ReportBuffer(256) As Byte
If ByteCount& > 254 Then ErrorExit ("Maximum ByteCount for WriteUSBdevice
is 254")
Call CopyBuffer(BufferPtr&, AddressFor(ReportBuffer(1)), ByteCount&)
ReportBuffer(0) = 0 ' ReportID
Success = WriteFile(HidHandle&, AddressFor(ReportBuffer(0)), ByteCount& +
1, BytesWritten&, 0)
If (Success = 0) Then ErrorExit ("Could not write an Output Report")
End Sub
Public Sub CloseUSBdevice()
' This subroutine closes the USB device that we have been using
Call CloseHandle(HidHandle&)
End Sub
Public Function ReturnHexByte(Text$) As Byte
' Converts the first two characters of text$ into a byte
Dim Value As Byte
Utext$ = UCase(Text$) ' Convert to uppercase for search
HexString$ = "0123456789ABCDEF" ' Non-Hex characters = 0
Value = 0
```

```
For i& = 0 To 15
    If Mid(Utext$, 1, 1) = Mid(HexString$, i& + 1, 1) Then Value = Value +
(16 * i&)
    If Mid(Utext$, 2, 1) = Mid(HexString$, i& + 1, 1) Then Value = Value +
i&
    Next i&
ReturnHexByte = Value
End Function
Public Function TwoHexCharacters$(Value As Byte)
HexString$ = "0123456789ABCDEF"
TwoHexCharacters$ = Mid(HexString$, Int(Value / 16) + 1, 1) &
Mid(HexString$, Int(Value And &HF) + 1, 1)
End Function
Public Function TwoDecimalCharacters$(Value As Byte)
DecimalString$ = "0123456789"
Tens& = Int(Value / 10): Units& = Value - (10 * Tens&)
TwoDecimalCharacters$ = Mid(DecimalString$, Tens& + 1, 1) &
Mid(DecimalString$, Units& + 1, 1)
End Function
Public Function ThreeDecimalCharacters$(Value As Byte)
h& = Int(Value / 100): t& = Int((Value - (100 * h&)) / 10): u& = Value -
(100 * h&) - (10 * t&)
ThreeDecimalCharacters$ = h& & t& & u&
End Function
Public Sub ErrorExit(Reason$)
ErrorCode = GetLastError()
Call MsgBox(Reason$, vbCritical)
'If ErrorCode <> 0 Then
    End
'End If
End Sub

Public Sub WUSBdevice(BufferPtr&, ByteCount&)
' This subroutine "writes" to an openned USB device
' Copy the user buffer into a local buffer so that a ReportID (=0) may be
prepended
' The first byte will contain the ReportID (=0)
'Dim ReportBuffer(256) As Byte
'If ByteCount& > 254 Then ErrorExit ("Maximum ByteCount for WriteUSBdevice
is 254")
'Call CopyBuffer(BufferPtr&, AddressFor(ReportBuffer(1)), ByteCount&)
'ReportBuffer(0) = 0 ' ReportID
Success = WriteFile(HidHandle&, BufferPtr&, ByteCount& + 1, BytesWritten&,
0)
If (Success = 0) Then ErrorExit ("Could not write an Output Report")
End Sub
```

A.4.5. MÒDUL APICALLS

Codi del mòdul APIcalls utilitzat en el programa Mable Control per accedir a les llibreries WIN32 API de l'ordinador, contingut al fitxer OSInterface.bas del programa creat en Visual Basic 6.0.

```
'   This module is common to all of the Example programs
'    It declares the data types and system calls required to access the
Windows operating system
'
Public Type Security_Attributes: nLength As Long: lpSecurityDescriptor As
Long: bInheritHandle As Long: End Type

Public Type Guid: Data(3) As Long: End Type 'A GUID is 16 bytes long

Public Type Device_Interface_Data
cbsize As Long: InterfaceClassGuid As Guid: Flags As Long: ReservedPtr As
Long: End Type

Public Type Device_Interface_Detail: cbsize As Long: DataPath(256) As Byte:
End Type

Public Type ConfigurationDataType: cookie As Long: cbsize As Long:
RingBuffersize As Long: End Type

Public Type HidD_Attributes
cbsize As Long: VendorID(1) As Byte: ProductID(1) As Byte: VersionNumber(1)
As Byte: Pad(10) As Byte: End Type

'    Declare the functions from Dan Appleman's "Programmers Guide to the
Win32 API"
Declare      Function      AddressFor      Lib      "apigid32.dll"      Alias
"agGetAddressForObject" (PassedByReference As Any) As Long
Declare  Sub  CopyBuffer  Lib  "apigid32.dll"  Alias  "agCopyData"  (ByVal
SourcePtr&, ByVal DestPtr&, ByVal ByteCount&)
'
'   Declare the API calls that I am using
Declare Sub HidD_GetHidGuid Lib "HID.dll" (GuidPtr&)

Declare    Function    SetupDiGetClassDevs    Lib    "setupapi.dll"    Alias
"SetupDiGetClassDevsA" _
(GuidPtr&, ByVal EnumPtr&, ByVal HwndParent&, ByVal Flags&) As Long

Declare  Function  SetupDiDestroyDeviceInfoList  Lib  "setupapi.dll"  (ByVal
DeviceInfoSet&) As Boolean

Declare Function SetupDiEnumDeviceInterfaces Lib "setupapi.dll" _
(ByVal   Handle&,   ByVal   InfoPtr&,   GuidPtr&,   ByVal   MemberIndex&,
InterfaceDataPtr&) As Boolean

Declare Function SetupDiGetDeviceInterfaceDetail Lib "setupapi.dll" Alias
"SetupDiGetDeviceInterfaceDetailA" _
(ByVal    Handle&,    InterfaceDataPtr&,    InterfaceDetailPtr&,    ByVal
DetailLength&, _
ReturnedLengthPtr&, ByVal DevInfoDataPtr&) As Boolean
```

```
Declare Function GetLastError Lib "kernel32" () As Long

Declare Function CreateFile Lib "kernel32" Alias "CreateFileA" _
(ByVal lpFileName$, ByVal dwDesiredAccess&, ByVal dwShareMode&,
lpSecurityAttributes As Security_Attributes, _
 ByVal dwCreationDisposition&, ByVal dwFlagsAndAttributes&, ByVal
hTemplateFile&) As Long

Declare Sub CloseHandle Lib "kernel32" (ByVal HandleToClose As Long)

Declare Function ReadFile Lib "kernel32" _
(ByVal Handle&, ByVal BufferPtr&, ByVal ByteCount&, BytesReturnedPtr&,
ByVal OverlappedPtr&) As Long

Declare Function WriteFile Lib "kernel32" _
(ByVal Handle&, ByVal BufferPtr&, ByVal ByteCount&, BytesReturnedPtr&,
ByVal OverlappedPtr&) As Long

Declare Function DeviceIoControl Lib "kernel32" _
(ByVal hDevice&, ByVal dwIoControlCode&, lpInBuffer&, ByVal nInBufferSize&,
_
 lpOutBuffer&, ByVal nOutBufferSize&, lpBytesReturned&, lpOverlapped&) As
Long

Declare Function HidD_GetPreparsedData Lib "HID.dll" (ByVal Handle&, ByVal
BufferPtr&) As Long
Declare Function HidD_GetAttributes Lib "HID.dll" (ByVal Handle&,
BufferPtr&) As Long
Declare Function HidD_GetManufacturerString Lib "HID.dll" (ByVal Handle&,
ByVal BufferPtr&, ByVal Length&) As Long
Declare Function HidD_GetProductString Lib "HID.dll" (ByVal Handle&, ByVal
BufferPtr&, ByVal Length&) As Long
Declare Function HidD_GetSerialNumberString Lib "HID.dll" (ByVal Handle&,
ByVal BufferPtr&, ByVal Length&) As Long
Declare Function HidD_GetIndexedString Lib "HID.dll" (ByVal Handle&, ByVal
index&, ByVal BufferPtr&, ByVal Length&) As Long
Declare Function HidD_GetConfiguration Lib "HID.dll" (ByVal Handle&, ByVal
BufferPtr&, ByVal Length&) As Long
Declare Function HidD_SetConfiguration Lib "HID.dll" (ByVal Handle&, ByVal
BufferPtr&, ByVal Length&) As Long
Declare Function HidD_GetPhysicalDescriptor Lib "HID.dll" (ByVal Handle&,
ByVal BufferPtr&, ByVal Length&) As Long
```