Capítulo 4

Electrónica de control

4.1 Definición de la red de control

La red de control forma la base del tercer nivel de la torre Bot[1] y en relación a sus antecesores, es una de las principales innovaciones en este robot. Al analizar los robots localizados en Internet se vio que la mayoría de ellos y sobre todo cuando eran complicados, utilizaban una red de microcontroladores para mover los servos. Uno de los motivos principales de esto es por la complejidad que lleva realizar el control de los servos sin utilizar algún tipo de ayuda electrónica como son los temporizadores y las interrupciones. Otro motivo es, que aun disponiendo de algún recurso del tipo anterior, no siempre un microcontrolador los tiene en un número suficiente. Por último, aún disponiendo de la cantidad, siempre existiría la limitación de no poder ampliar el sistema y, por otro lado, en caso de necesitar menos se estarían desaprovechando recursos.

Una forma genérica de resolver todos los problemas anteriores es fabricar un módulo que pueda mover un número determinado de servos, exprimiendo lo más posible sus recursos internos y dotar a dicho módulo de la posibilidad de comunicarse con otros, es decir de unirlos entre sí, de manera que para controlar el doble de servos tan sólo haya que poner otro módulo al lado y así sucesivamente.

Para mover todos los motores del robot (al menos ocho) se necesita una electrónica capaz de generar todas estas señales de PWM. En el capítulo anterior se ha visto que para generar ésta utilizando la CT6811 se necesitan dos temporizadores: con uno de ellos se calcula el periodo de la señal (T) y con el otro el ancho del pulso (Ton), valor este último proporcional a la posición deseada del servo (ver figura 4.1). La CT6811 es una tarjeta

[1] En el apartado 1.1 se describe la torre Bot con detalle.

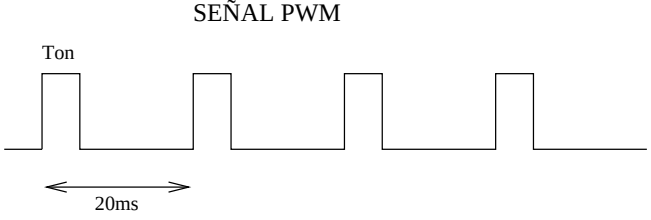

Figura 4.1: Señal de control de un servomecanismo.

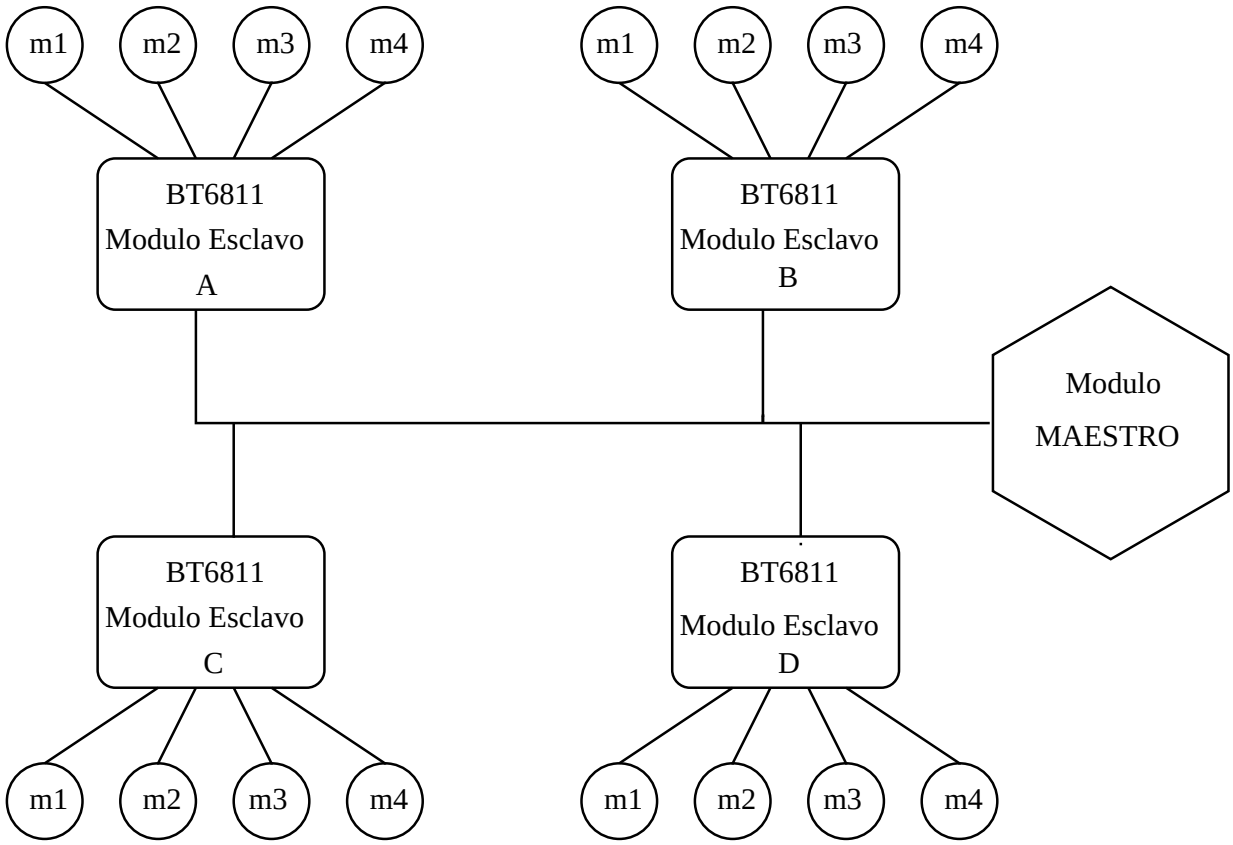

Figura 4.2: Esquema de la red de control.

electrónica que incorpora el microcontrolador 68hc11 de Motorola. Este dispone de cinco timers, por lo que teóricamente se podrá extender la aplicación al control de cuatro servos. Se utilizarán cuatro timers para modular la anchura de cada uno y otro para el periodo.

Al inicio del proyecto todavía no se tenía claro el número de motores necesarios para mover bien el robot. En un principio se pensó en utilizar ocho, pero siempre estaba la posibilidad de tener que utilizar doce. Esto, junto a la característica de que un micro 68hc11 puede mover cuatro servos hizo que se optara por diseñar un módulo electrónico que pudiera controlar cuatro servos y comunicarse con el exterior para conectarse a otros. De esta manera, la red de control seguiría siendo válida en caso de tener que acoplar cuatro motores más. Incluso en el caso futuro de que se incluyera una cabeza y rabo móviles, la red seguiría siendo válida una vez más.

Por otro lado, para lograr que todos los motores se muevan en sintonía permitiendo así que el robot avance, retroceda y gire, se necesita un programa de control unificado, que sepa en todo momento a qué posición mandar cada uno de los ejes de los motores. Este programa podría residir en uno de los módulos anteriores pero se perdería la homogeneidad entre ellos. Lo más interesante es añadir un módulo nuevo que sea el cerebro central y dejar los otros tal cual. A partir de ahora se denominará módulo maestro al que contiene el programa principal del robot y módulos esclavos a los encargados de recibir las posiciones enviadas por el módulo maestro y comunicárselas a los motores. Un diagrama de la red de control se representa en la figura 4.2.

Todavía quedan algunas cuestiones por plantear para tener una red operativa. La

primera es qué tipo de comunicación se va a establecer entre los módulos y qué volumen de información viajará a través de ella. Respondiendo a la primera pregunta se establecerá una comunicación serie síncrona (SPI) y el volumen de datos será proporcional al número de módulos que existan. Las tramas de control se explicarán en el capítulo siguiente, el del software, aunque ahora se comentarán algunas características. Por ejemplo, si se ha decido utilizar la comunicación serie en lugar de una paralelo es por que el 68hc11 está preparado para soportar las primeras ofreciendo mecanismos automáticos de gestión. Resumiendo, su utilización simplifica el software. Se ha utilizado la comunicación síncrona (SPI) en lugar de la asíncrona (SCI) porque se ha reservado ésta última para conectar el robot al PC y permitir depurar las secuencias de movimiento desde éste.

También hay que decidir si se quiere que la información viaje en ambos sentidos, del maestro a los esclavos y viceversa. La ventaja de esto es permitir que los módulos esclavos, los que manejan los servos, puedan también leer algún tipo de información del entorno mediante sensores y transmitirla al módulo maestro para que la procese. De esta manera se puede dotar al robot de una cierta inteligencia distribuida que hace que aunque uno de los módulos falle, existan otras tomas de adquisición de datos que suplanten a la anterior. Todo esto se detallará más concretamente en el capítulo del software, aunque ahora se adelanta la necesidad de añadir un cable para hacer esto. Es decir, además del cable serie se necesitará alguna línea más, que sirva para establecer un protocolo entre los diferentes módulos, de manera que si alguno de ellos quiere transmitir información pueda mandarle una señal al maestro, que será el que escuche.

Para responder a la segunda pregunta que se planteaba al principio, se debería analizar el tráfico de datos en la red y la viabilidad de la misma. Como ya se ha dicho, más adelante se expondrá con detalle el protocolo de la red, ahora se adelantan las siguientes características. El módulo maestro mandará tramas de cuatro bytes por la red, todos los módulos esclavos leerán las tramas y en función del primer byte sabrán si son para ellos o no. En caso de serlo el módulo interpretará el resto de la trama y en caso contrario dejará la interpretación y esperará a recibir la siguiente trama. En el primer prototipo realizado no se van a usar las comunicaciones en sentido opuesto, por lo que no hará falta utilizar más señales de protocolo.

Considerando lo anterior y teniendo en cuenta que la velocidad máxima del bus (SPI) es de 1 Mbit por segundo[2], que la velocidad de los servos es de 0.23seg/60º y que una trama tiene 4 bytes, se pueden sacar las siguientes conclusiones:

1. Suponiendo que el robot tiene 12 motores (caso peor) y que se envía una trama cada vez que se quiera indicar a un motor cualquiera la posición donde se debe situar, se gastarán 12 tramas, es decir 48 bytes, en reposicionar todos los servos del robot.

2. Esos 48 bytes son 384 bits y sabiendo que la red tiene una velocidad de 1 Mbit/seg, se obtiene que en 0.384mseg se habrá enviado toda la información para reposicionar todos los motores.

3. Los motores tardan 0.23seg en girar 60º por lo que en 0.384mseg girarán 0.1º.

[2]La velocidad del SPI cuando el 68hc11 está configurado como maestro es de 1 Mbit/seg, en cambio cuando está configurado como esclavo es de 2 Mbit/seg. En esta red como el flujo de información va del maestro a los esclavos, predominará la velocidad de éste, es decir 1 Mbit/seg.

Una de las características que se apreciaron al hacer las pruebas con los servos era que había un incremento mínimo en la anchura de la señal Ton que permitía apreciar con la vista un desplazamiento del eje. Esto hace que con un byte (255 valores distintos) se pueda codificar el ancho de la señal Ton. Si el rango de giro es de 180º y tenemos 255 valores distintos para indicar la posición la resolución del motor será de 0.7º. Lo que se quiere hacer ver con estos cálculos es que realmente la red no va a estar cargada ya que los motores tardarán más tiempo en llegar a su posición que lo que necesita la red para indicar una nueva posición a todos los motores del robot. Cada variación mínima de posición implica un desplazamiento de 0.7º lo que permite la llegada de al menos siete tramas. En el ejemplo siguiente se aclara la idea anterior: Supóngase que se quiere hacer un giro de 60º en el eje del motor, lo primero que se hará será indicar la nueva posición al motor, para ello se utilizará la red, pero en girar los 60º el servo tardará 0.23seg, intervalo de tiempo durante el cuál no se le deberá indicar a ese motor ningún otro valor de posición. En caso de hacerlo se moverá hasta alcanzar este último objetivo sin pasar por el primero. Vuelve a ocurrir un efecto que favorece a la red, el software tendrá que intercalar pausas para permitir que los motores lleguen a su posición final. Durante estas pausas todos los módulos podrán estar realizando otras funciones.

El lector puede considerar que codificar la posición de un motor con un byte no es suficiente y que para obtener más precisión es necesario hacerlo con dos bytes. Esto no debe plantear ningún problema, la trama se incrementará en una unidad, es decir estará formada por cinco bytes. Ahora en lugar de usar 48 bytes para reposicionar todo el robot se usarán 60, que son 480 bits a una velocidad de 1Mbit/seg, es decir un tiempo de reposicionamiento de 0.48mseg. En ese tiempo un motor puede girar 0.125º, lo que significa que si se quiere ir moviendo un motor paso a paso con incrementos de ángulo menores de 0.125 la red no lo soportará en tiempo real, es decir, el motor llegará antes a su posición que la trama que le indica la nueva posición. Ahora tendrá que quedarse parado mientras que antes se tenía que retardar la trama. Se tiene por tanto otra limitación, pero ésta no debe preocupar al lector, es todavía menos problemática que la anterior que obligaba a insertar pausas. La razón se encuentra en que la BT6811 indica posiciones finales y es el servomecanismo el que se encarga de girar el eje pasando por todas las intermedias. La única razón para que ocurra lo anterior, dar pasos con incrementos de ángulos mínimos, es que se quiera mover el robot a cámara lenta, y por lo tanto se obligue al servomecanismo a parar en cada incremento de ángulo. Pero para que se produzca el efecto visual de cámara lenta además hay que intercalar pausas y ya no habrá problemas con la red, porque entre movimiento y movimiento hay que esperar un tiempo mayor al necesario para transmitir las tramas por el SPI.

También queda por considerar otro efecto: cuando se entrene el robot con el PC, la comunicación entre ambos se hará mediante una conexión serie asíncrona (SCI) a una velocidad de 9600 baudios. Esta velocidad es menor que la del bus síncrono (SPI), por lo que, en este caso, el cuello de botella lo impondrá la conexión PC-Robot. Aunque se sature la comunicación por el SCI, la del SPI estará totalmente descargada.

Todas estas reflexiones inducen a dar por buena la viabilidad de realizar un control a través de una red distribuida, y por lo tanto se continuará con el diseño del módulo esclavo, llamado a partir de ahora BT6811. Por último, el lector puede estar preguntándose cuál es el porcentaje de utilización de la CPU del 68hc11, si le dará tiempo a implementar las cuatro señales de PWM, y si se podrán estar realizando otras tareas en paralelo. Todas

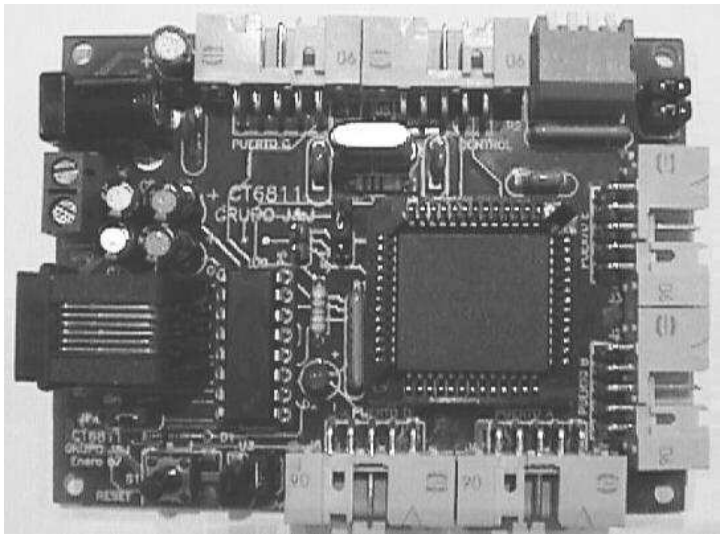

Figura 4.3: Foto de la tarjeta electrónica CT6811.

estas preguntas se responden en el capítulo del software, pero se adelanta que son todas afirmativas. El microcontrolador 68hc11 está la mayor parte del tiempo desocupado.

4.2 Módulo maestro: CT6811

El módulo maestro de la red será el que ejecute el programa principal del robot enviando las tramas de control a los módulos esclavos, que a su vez serán los que controlen los servomotores. En lugar de diseñar un nuevo hardware para tal función se ha optado por utilizar la tarjeta CT6811 de Microbótica (figura 4.3). Dicha tarjeta se utiliza para desarrollar aplicaciones hardware y software basadas en el microcontrolador MC68HC11 de Motorola. Pero, no sólo sirve para el desarrollo de *prototipos*, sino que también está pensada para funcionar en *sistemas profesionales* . Se trata de una tarjeta muy versátil que ha sido desarrollada por integrantes de Microbótica y cuyos esquemas están disponible en su web.

Entre las características principales de la CT6811 se destaca su posibilidad de utilización en distintos modos, la existencia de conectores para acceder a todos los pines del microcontrolador y su tamaño reducido. Respecto a la primera característica hay que mencionar que todos los modos se han utilizado en alguna de las fases del proyecto:

1. El modo **entrenador**: Al desarrollar el software se iban probando los algoritmos en la misma CT6811. De esa forma se verificaba el correcto funcionamiento.

2. El modo **autónomo**: Una vez terminado el programa principal se deja grabado en la tarjeta y a partir de ahí no es necesario mandar dicho programa desde el PC cada vez que se quisiera poner en marcha el robot.

3. El modo **periférico inteligente**: Este modo se ha utilizado para buscar las secuencias de movimiento del robot. Al ser bastante complicado calcularlas a ojo, se ha desarrollado un programa en el PC que permite generarlas, grabarlas y más tarde

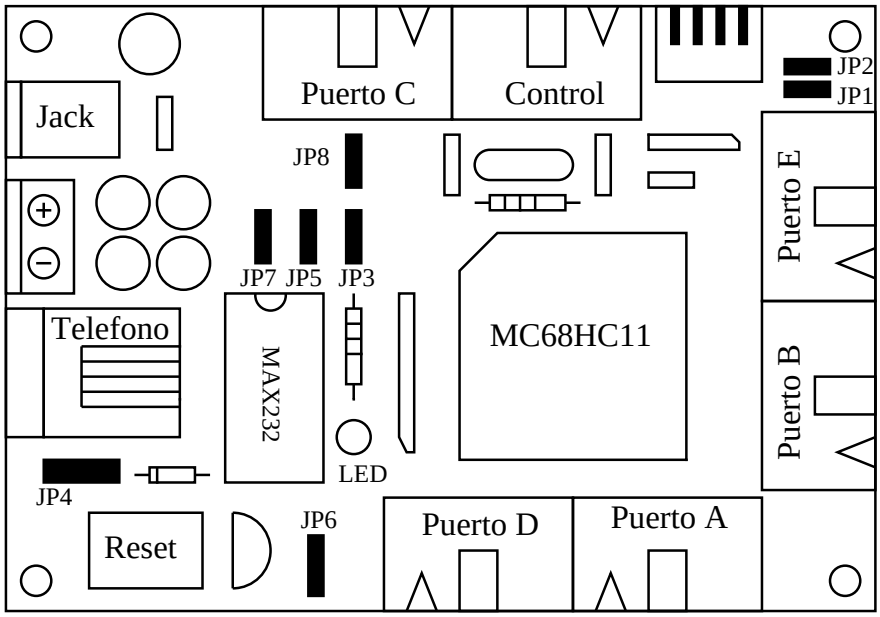

Figura 4.4: Componentes de la CT6811.

reproducirlas. Dicho programa se comunica permanentemente con la CT6811 para que ésta actúe de puente conectando el PC a la red de control, permitiendo así que las tramas las controle el PC en lugar del programa maestro de la CT6811.

En relación a los buses de salida hay que recordar que la red de control se basa en el bus SPI del 68hc11. Por lo tanto para su realización se tendrán que sacar al exterior los pines MISO, MOSI, SCK y SS, que se encuentran en el conector llamado Puerto D. Además de los pines del Puerto D, hay muchos otros que permitirán ampliar la red, añadir sensores, o conectar un PC

Por último se ha mencionado el tamaño porque hay otras tarjetas electrónicas de mayor tamaño que hacen bastante difícil su integración en la estructura mecánica del robot. Las reducidas dimensiones se deben sobre todo a no utilizar un módulo extra con memoria externa, optando en lugar de esto por emplear un modelo de microcontrolador con mayor memoria interna del tipo EEPROM, como puede ser el MC68HC811E2 o el MC68HC711E9.

4.2.1 Configuración final de la CT6811

La CT6811 en este proyecto estará configurada como tarjeta autónoma, es decir al introducir la alimentación ejecutará un programa grabado en la EEPROM interna, que será el que controle al resto de módulos. Originalmente la tarjeta viene con el modelo de microcontrolador 68hc11A1 que tiene 512 bytes de EEPROM situados en la posición $B600. Esto implica que para hacer lo anterior hay que arrancar la placa en BootStrap y tener el jumper JP5 conectado (ver figura 4.4), con lo que las comunicaciones serie asíncronas se pierden, y con ello la posibilidad de conectar el PC. Por eso se ha optado por sustituir este micro por otro modelo, en concreto el 68hc811E2, que tiene 2 Kbytes de EEPROM situadas desde la dirección $F800 hasta la $FFFF. Ahora se podrá arrancar la tarjeta en SingleChip, es decir con el switch 2 arriba y los otros abajo, ya que se utilizará el vector de interrupción de reset

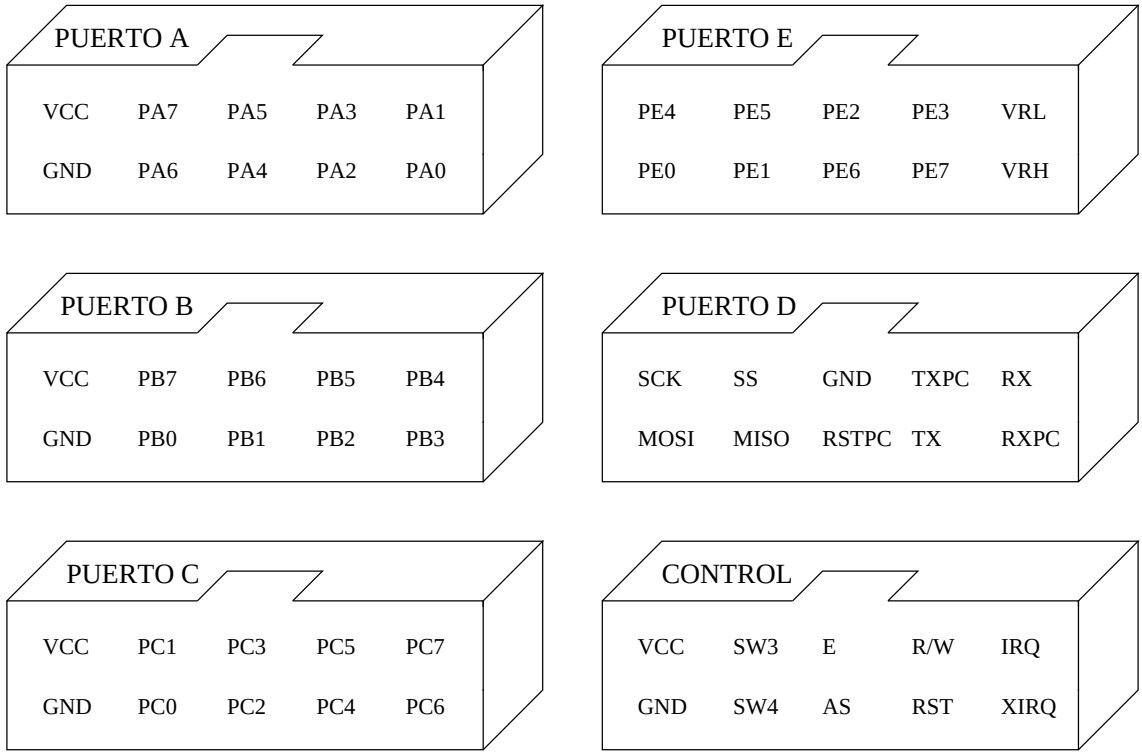

Figura 4.5: Buses de salida de la CT6811.

situado en la dirección $FFFE para indicar al microcontrolador el comienzo del programa a utilizar. Además el jumper JP5 estará quitado y las comunicaciones serie asíncronas en funcionamiento, por lo tanto el PC podrá comunicarse con la CT6811. En resumen, al cambiar el modelo de micro se consigue más memoria y conservar en modo autónomo las comunicaciones con el PC.

Después de esta pequeña aclaración se procede a enumerar la situación de cada jumper y switch, junto con una breve descripción del motivo. Más sobre información la tarjeta se puede encontrar en el manual de usuario de la CT6811 que se encuentra en el apéndice B del proyecto.

1. El conector de teléfono se usará para conectar el PC al robot. La comunicación que se establecerá entre ambos será del tipo serie asíncrona (SCI). El puerto serie del PC cumple la norma RS-232 que no es compatible con la TTL del microcontrolador, por lo que la CT6811 incluye un circuito integrado (MAX232) que actúa de interfaz entre ambas.

2. El Puerto D se usará para conectar la CT6811 a la red de control. En dicho puerto hay conexiones con los dispositivos: SCI del micro (TTL), SCI del PC (RS232) y con el SPI. Será este último el que se utilice para acceder a la red, por lo que de todos los pines del puerto, sólo se utilizarán MISO, MOSI, SS y SCK (ver figura 4.5). El significado de cada uno se explicará al final del capítulo.

3. El jumper JP6 estará quitado para inhabilitar el LVI, circuito que produce un *reset* en el microcontrolador cuando la tensión de alimentación cae por debajo de los 4.5

voltios. En este tipo de proyectos en los que hay varios motores es bastante frecuente que estos induzcan ruido en la línea de alimentación de la electrónica y produzcan con ello caídas de tensión por debajo del valor anterior. Por lo tanto, para evitar los *reset* esporádicos se quitará dicho jumper. Experiencia que se ha probado y ha dado buenos resultados.

4. El jumper JP5 estará también quitado. Como se comentó antes, para arrancar el programa no será necesario utilizar el modo *Bootstrap* junto con JP5. Bastará con configurar la CT6811 en *Single Chip* y disponer evidentemente de un microcontrolador modelo E2. Si no se hubiera optado por esta solución, al colocar dicho jumper, las comunicaciones SCI se perderían y con ellas la posibilidad de conectar el PC

5. Para configurar la tarjeta en *Single Chip* hay que poner el switch 2 arriba y el resto abajo.

6. La alimentación será de 6 voltios y se utilizará la clema de conexión situada entre el conector de teléfono y el jack de alimentación. La polaridad es la siguiente: Por Vcc se introducirá la tensión positiva y por GND la negativa.

7. EL jumper JP3 estará puesto, de manera que actuando el bit PA6 del puerto A del microcontrolador se encenderá o apagará el LED de la CT6811. Servirá de ayuda para establecer si el programa maestro está o no funcionando.

8. El jumper JP7 estará situado en la posición OFF para anular la función del reset software. Esto se hace ya que el robot en modo autónomo no estará conectado al PC, estando todas las líneas de conexión entre ambos al aire. De todas ellas, hay una que por motivos de ruido puede hacer un *reset* en el microcontrolador. Para evitar este efecto no deseado, se colocará el jumper JP7 en la posición OFF.

9. Los jumpers JP1 y JP2 estarán puestos para que automáticamente se establezcan los niveles de referencia del conversor analógico digital en VRH=Vcc y VRL=GND.

10. El jumper JP8 sólo se quitará cuando se quiera programar un microcontrolador de la familia E9. Dado que no es el caso en este proyecto siempre estará puesto.

11. El jumper JP4 estará situado en la posición RST, de forma que al pulsar el pulsador de la CT6811 se produzca un *reset* en el microcontrolador. Su otra posición no se suele utilizar salvo raras excepciones.

4.3 BT6811: Módulo esclavo

El módulo esclavo (ver figura 4.6) se va a encargar de escuchar la información que viaja por la red de control y de capturar las tramas que vayan dirigidas a él. Éstas las enviará el módulo maestro y contendrán información de la posición y estado de los servomecanismos. Además la BT6811 generará constantemente las señales de PWM necesarias para mover los cuatro servos asociados a ella.

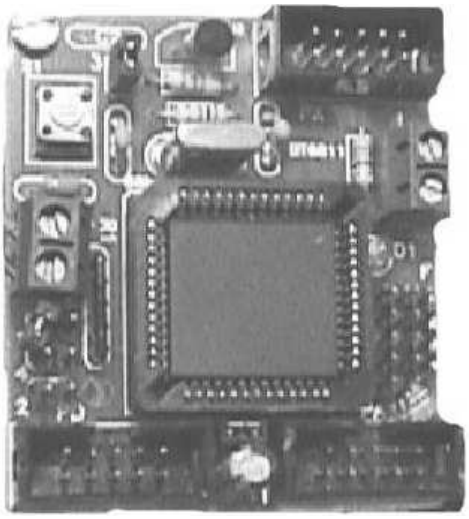

Figura 4.6: Foto del módulo esclavo o BT6811.

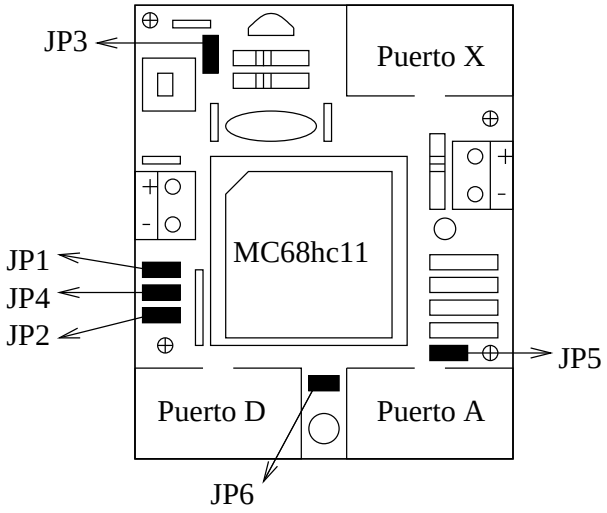

Figura 4.7: Componentes de la BT6811.

Para la realización del módulo se va a partir del conocimiento adquirido en el desarrollo de la CT6811[3], sobre todo para facilitar el diseño hardware y la programación posterior, factores determinantes en la puesta a punto del módulo. Realmente esta tarjeta es una simplificación de la CT6811, partiendo de su esquemático y eliminando todo aquellos elementos que no eran necesarios se ha llegado a una tarjeta con las siguientes características:

1. El módulo esclavo va a recibir toda la información de la red de control, por lo que no será necesario disponer del adaptador del PC. Se eliminan así dos componentes que ocupaban bastante espacio: el conector telefónico y el MAX232 junto con su electrónica asociada.

[3]La CT6811 es una tarjeta desarrollada por el autor y otros compañeros en el año 1996. Actualmente se encuentra disponible como hardware abierto en la WEB de Microbótica (www.microbotica.es). Esto significa que tanto su esquemático como manuales están publicados para todo el mundo.

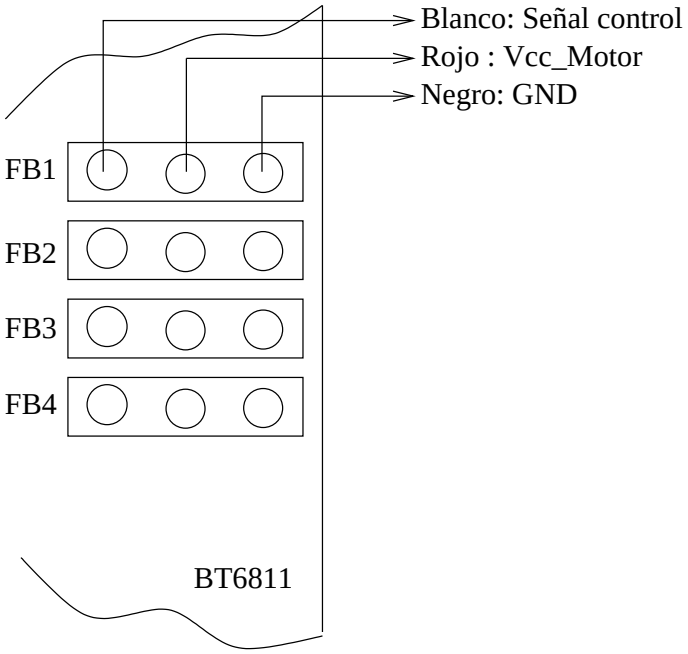

Figura 4.8: Colocación del cable de los servomecanismos en la BT6811.

2. En la BT6811 se han eliminado los switches de control y se han reconfigurado los jumpers. Ahora el *JP1* selecciona el modo de funcionamiento del microcontrolador, el *JP2* se colocará para obligar al modo *bootstrap* a ejecutar el programa de la EEPROM, el *JP3* activa o desactiva el LVI, el *JP4* permite sacar por el puerto D la alimentación de la tarjeta, el *JP5* permite que los motores se alimenten con la misma tensión de la electrónica y el *JP6* activa o desactiva la señal de control *SS*.

3. Se ha dejado el conector del Puerto A para poder seguir utilizando los capturadores y comparadores del microcontrolador, además viene muy bien para conectar otros módulos existentes en el mercado[4].

4. Se han incluido unos pines para poder conectar directamente los servomotores a la BT6811 sin necesidad de ninguna conexión adicional. La conexión se realiza mediante un cable triple que trae el servomecanismo de fabrica y cuyo sentido es el mostrado en la figura 4.8. La señal de control de los servos se conecta a través de los pines con el bus B del microcontrolador, y en concreto; el bit PB0 controla la salida FB1, el bit PB1 la salida FB2, el bit PB2 la salida FB3 y por último PB3 la salida FB4.

5. Para reducir el tamaño de la BT6811 se han eliminado los conectores de expansión que menos se usaban en la CT6811. Los puertos C y E del micro son muy útiles para conectar periféricos, debido sobre todo a que el primero es bidireccional y el segundo permite entradas analógicas. Por eso se ha decidido conservarlos pero utilizando un

[4]Microbótica S.L. dispone además de una tarjeta periférica que se conecta por medio del Puerto A a la CT6811 y a la BT6811. Esta tarjeta incorpora un driver de potencia para manejar dos motores de CC y unos circuitos de polarización para leer sensores. Se puede encontrar más información de la misma en la web de Microbótica [10].

PUERTO X	Puerto C (bidireccional D)	Puerto E (entada D/A)
PX0	PC0	PE3
PX1	PC1	PE7
PX2	PC2	PE2
PX3	PC3	PE6
PX4	PC4	PE5
PX5	PC5	PE1
PX6	PC6	PE4
PX7	PC7	PE0

Tabla 4.1: Correspondencia de bits en el Puerto X.

solo conector en lugar de dos. Es decir, ahora el puerto C y el puerto E del microcontrolador comparten el mismo conector de salida llamado: Puerto X, cuya disposición y equivalencia de pines se muestra en la tabla 4.1.

6. Se ha eliminado la posibilidad de usar el pulsador de reset como entrada de interrupción y el LED de pruebas ahora se ha conectado al bus B del microcontrolador, en concreto al bit PB4.

En el apéndice A se encuentra el manual de usuario de la BT6811, en él se describe todo lo anterior con mucho más detalle. También se explica que al no tener la BT6811 posibilidad de conectarse directamente al PC, para grabar el microcontrolador habrá que optar por una de las siguientes dos opciones:

- Sacar el microcontrolador del zócalo y grabarlo con un grabador o insertándolo en una CT6811. La ventaja es que es fácil para hacerlo un par de veces, pero para muchas ocasiones no es recomendable.
- Utilizar un módulo exterior que realiza la conversión de niveles. Esta opción parece más compleja pero se puede utilizar como adaptador una CT6811 sin el microcontrolador. Al final no hará falta sacar y meter el micro de la BT6811 y a la larga será más confortable. Esta última opción es la que se utilizado en este proyecto.

De los tres modos de funcionamiento que disponía la CT6811, la BT6811 sólo utiliza uno, en concreto el *modo autónomo*. Para disponer cualquiera de los otros dos, *modo entrenador* y *modo periférico inteligente*, se necesita añadir externamente el adaptador para el PC.

4.3.1 Configuración final de la BT6811.

La BT6811 sólo funciona en modo autónomo, lo que implica que, o bien se utiliza el truco del *bootstrap*, o se utiliza un micro 68hc11E2 en modo *singlechip* [5]. En el robot se ha optado por la segunda opción ya que permite almacenar programas más grandes y además mantiene las comunicaciones serie, aunque éstas no serán utilizadas.

Una vez fijado el modelo de microcontrolador se procede a enumerar la configuración de cada uno de los jumpers:

[5] En el apéndice A se explican con detalle ambos métodos.

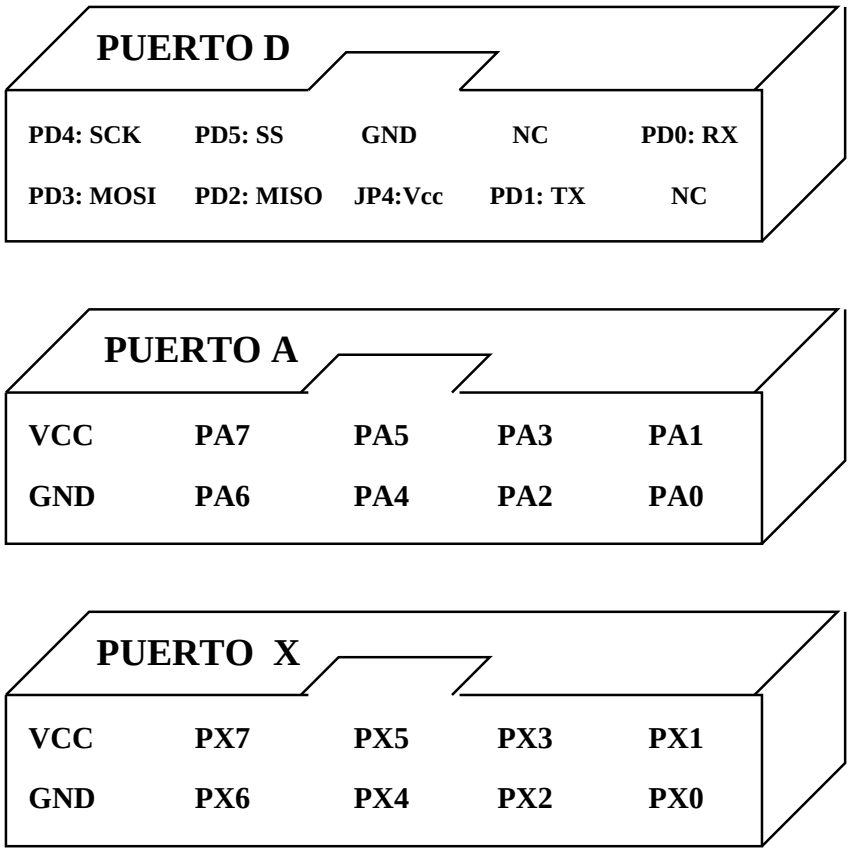

Figura 4.9: Buses de salida de la BT6811.

1. El Puerto D se usará para conectar la BT6811 a la red de control. En dicho puerto hay conexiones con los dispositivos: SCI del micro (TTL), SCI del PC (RS232) y con el SPI. Será este último el que se utilice para acceder a la red, por lo que de todos los pines del puerto, sólo se utilizarán MISO, MOSI, SS y SCK (ver figura 4.9). El significado de cada uno se explicará al final del capítulo

2. El jumper JP1 (#B/S) estará quitado para seleccionar el modo de funcionamiento *singlechip.*

3. El jumper JP2 (Go EEPROM) estará quitado. Este jumper tiene significado cuando el JP1 está conectado, es decir cuando la BT6811 esta configurada en modo *bootstrap*, como no es el caso no se utilizará.

4. El jumper JP3 estará quitado para inhabilitar el LVI, circuito que produce un *reset* en el microcontrolador cuando la tensión de alimentación cae por debajo de los 4.5 voltios. En este tipo de proyectos en los que hay varios motores es bastante frecuente que estos induzcan ruido en la línea de alimentación de la electrónica y produzcan con ello caídas de tensión por debajo del valor anterior. Por lo tanto, para evitar los *reset* aleatorios se quitará dicho jumper.

5. El jumper JP4 (EXP) estará quitado. Su función es permitir sacar por el puerto D la alimentación TTL, pero debido a la existencia de una CT6811 en la red de control tendrá que estar quitado, ya que la CT6811 no puede recibir ninguna señal de alimentación por dicho puerto.

6. El jumper JP5 (Vcc ON) estará quitado. Su utilidad es permitir llevar la alimentación TTL a los motores, de manera que no sea necesario conectar una fuente externa (batería, pila, etc) para poder moverlos. Debido a la cantidad de motores que se van a conectar y a la potencia eléctrica que estos van a consumir es necesario añadir la fuente externa, y por lo tanto no será necesario utilizar la propia de la electrónica (TTL).

7. El jumper JP6 (SS) estará puesto. La naturaleza del protocolo que se va a implementar sobre la red de control hace que sea necesario que este jumper este puesto, de manera que la señal SS (Slave Select) del maestro (CT6811) estará conectada a todas los esclavos (BT6811).

8. La alimentación TTL, es decir la de la electrónica, será de 6 voltios y se utilizará la clema de conexión situada entre el pulsador de reset y el jumper JP1. Por '+' se introducirá la tensión positiva.

9. La alimentación de los motores será de 6 voltios y se utilizará la clema de conexión situada entre el conector PX y los conectores de los motores. Por '+' se introducirá la tensión positiva y por '-' la negativa.

4.4 Descripción física de la red

Ya se han visto los diferentes módulos que forman la red y alguna de las características de la misma. Pero todavía queda por explicar su realización física. Partiendo de la figura

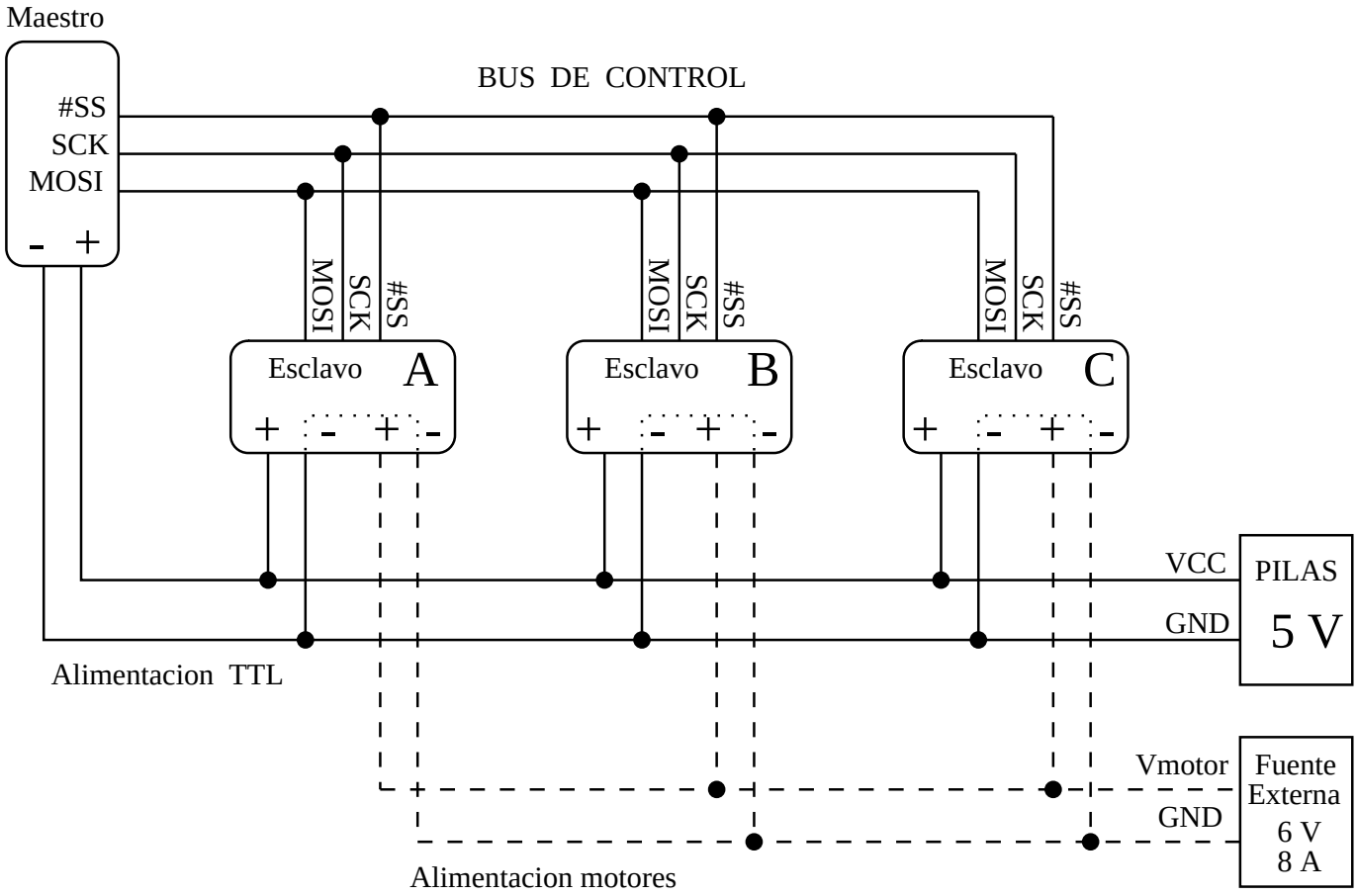

Figura 4.10: Diagrama de conexión de la red de control.

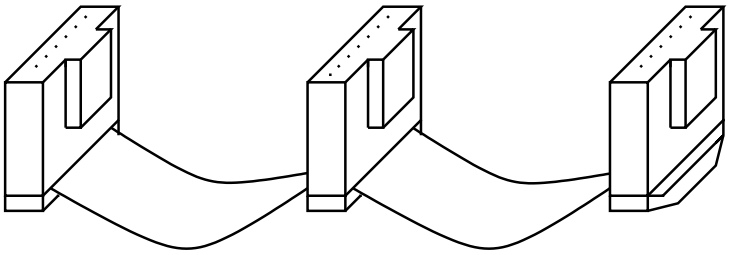

Figura 4.11: Cable tipo bus utilizado para conectar los distintos módulos.

4.10 que representa el diagrama de conexión se estudiarán para empezar los tres grupos de señales de control:

1. El primero, *BUS DE CONTROL*, conecta mediante tres líneas el módulo maestro (CT6811) con los tres módulos esclavos (BT6811). La línea *#SS*, es la encargada de avisar a los esclavos del inicio de una comunicación por parte del maestro. La línea *MOSI* transporta los datos que salen del maestro y llegan a todos los esclavos. Por último la línea *SCK* transporta la señal de reloj o sincronismo.

2. El segundo, *Alimentación TTL*, es el encargado de conectar la fuente de alimentación de 5 voltios con las entradas de tensión de los diferentes módulos. En la CT6811 ésta se introduce por la única clema que hay y en la BT6811 por la clema V_TTL (ver figura A.4).

3. El tercero, *Alimentación motores*, se encarga de conectar la fuente de potencia (5 voltios, 8 amperios) con las entradas de tensión para los servomecanismos, únicamente presentes en los módulos esclavos. La conexión se realizará por la clema indicada como V_Motor (ver figura A.4).

Para conectar las diferentes líneas del *bus de control* se utilizará un cable tipo bus, representado en la figura 4.11 y que une pin a pin los conectores Puerto D de todos los módulos. Al hacer esto, no sólo se estarán conectando las tres señales que interesan (MOSI, #SS y SCK), sino también todas las demás. Esto no plantea ningún problema, al contrario permitirá que la red siga siendo válida en futuras mejoras. Por ejemplo, en esta red la información sólo viaja por la línea MOSI del maestro a los esclavos, pero en el Puerto D existe otra que permite el retorno de la información, es decir de los esclavos al maestro. Aunque en la red preliminar no se ha contemplado ésta por no ser utilizada, realmente está presente, ya que al unir los módulos con el cable de bus se habrá realizado la conexión física.

Los siguientes buses se implementan con un par de cables cada uno, que se irán pasando de módulo en módulo conectando las clemas entre sí según corresponda. La razón de no utilizar una única alimentación para todo el robot ha sido para ofrecer una mayor inmunidad contra el ruido. Los servomotores consumen una potencia variable en función del esfuerzo mecánico que se les exige, esa variación puede producir caídas de tensión bastante importantes, que podrían hacer que la electrónica se desprogramase.

Si en lugar de utilizar la CT6811 como módulo maestro se utilizase otra tarjeta se podría introducir el *bus de alimentación* TTL por el *bus de control*. El motivo es que los módulos esclavos están preparados para recibir la tensión TTL por el puerto D, pero la CT6811 no.

También hay que decir que el módulo esclavo cortocircuita las líneas GND de los buses de alimentación, de esta forma se establece una única referencia de masa.

En la figura 4.12 se ha representado un croquis de la estructura final del robot con la localización de cada tarjeta electrónica. Además se han numerado los servomotores y se han asociado en tres grupos de cuatro unidades, cada uno de los cuales se ha hecho corresponder con cada una de las tres tarjetas esclavas (BT6811). La correspondencia es sencilla; cada servo lleva asociado un código formado por una letra y un número. La letra identifica la tarjeta esclava y el número indica el número de entrada a la que ha sido conectado. Así por ejemplo el servomotor *SERVO C2* se ha conectado a la entrada 2 de la tarjeta *BT6811 C*. Al final se ha obtenido lo siguiente:

1. La tarjeta esclava *BT6811 A* tiene conectados cuatro servomecanismos, tres de los cuales forman la pata delantera derecha y el cuarto forma la tercera articulación de la pata trasera derecha. Para recordar: El servomotor A1 se conecta a la primera entrada de la BT6811 A (motor 1) y forma la primera articulación de la pata delantera derecha, el servomotor A2 se conecta a la segunda entrada (motor 2) y forma la segunda articulación, el servomotor A3 se conecta a la tercera entrada (motor 3) y forma la tercera articulación. Finalmente el servomotor A4 se conecta a la cuarta entrada (motor 4) y forma la tercera articulación de la pata trasera derecha.

2. La tarjeta esclava *BT6811 B* vuelve a tener cuatro servomecanismos conectados, tres de los cuales forman la pata delantera izquierda y el cuarto forma la segunda articulación de la pata trasera derecha. La asociación es como la expresada antes.

3. Por último la tarjeta esclava BT6811 C tiene tres servomotores que forman la pata trasera izquierda y un cuarto que forma la primera articulación de la pata trasera derecha.

Al final la red está compuesta por tres tarjetas esclavas y una maestra, entre todas ellas controlan los doce motores del robot. Aunque no se ha implementado, se ha considerado ampliar el sistema con cuatro servomecanismos más, dos de ellos se utilizarían para poner al robot una cola móvil y los otros dos para hacer lo mismo con la cabeza. Dado como se ha concebido la red y la estructura, añadir un módulo esclavo para controlar esos cuatro nuevos motores es una tarea inmediata, tan sólo habrá que preocuparse de ampliar el software, cuestión que se aborda en el siguiente capítulo.

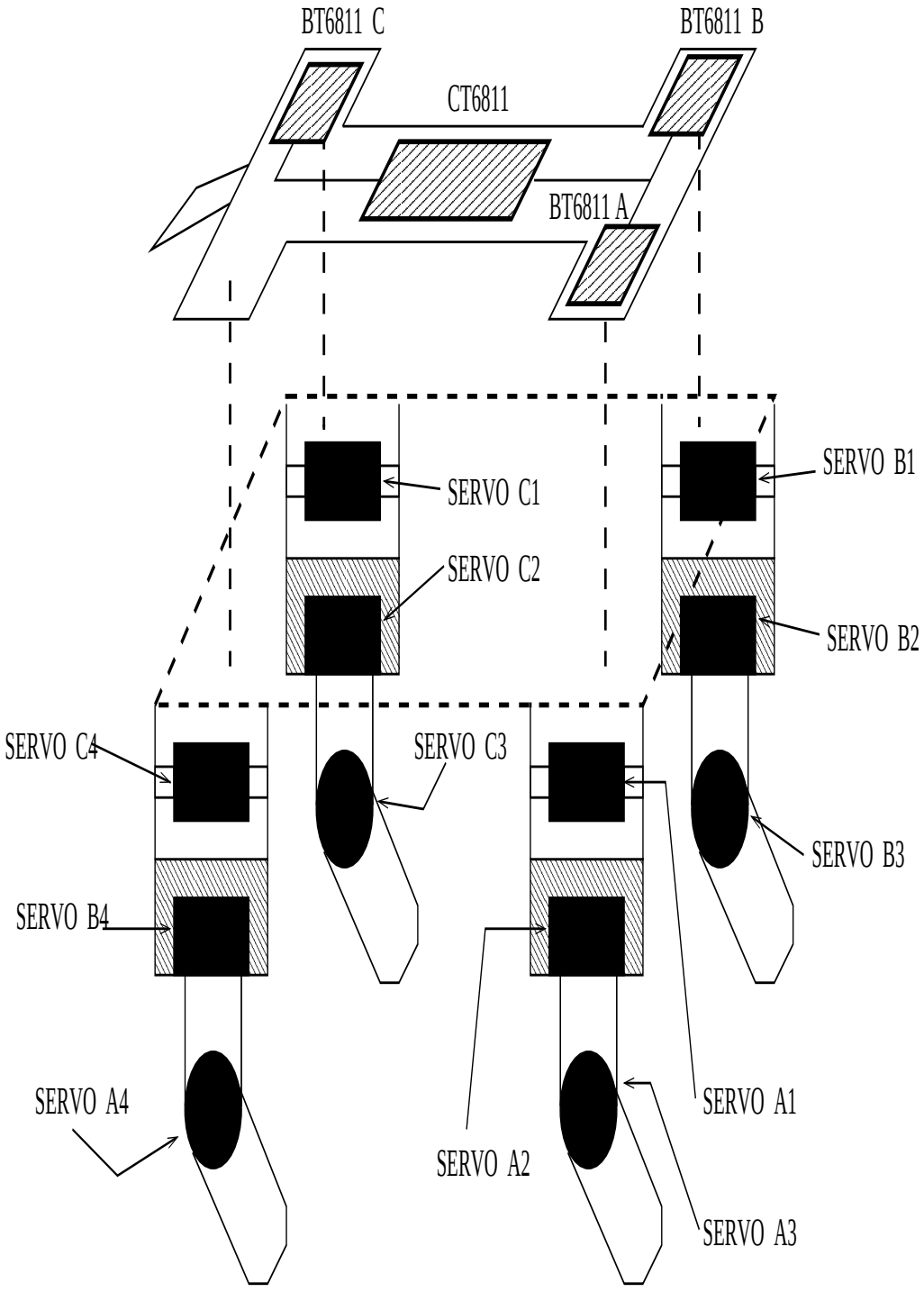

Figura 4.12: Diagrama de conexión de los motores.