# Asociación Cultural Española de Aficionados a los Mecanos

ACEAM

RELOJ LOCO. Juan Álvarez López

Boletín nº 36

Diciembre 2020

# Editorial

En este número del Boletín le rendimos merecido homenaje a nuestro socio y querido amigo ***Juan Álvarez López***, uno de los socios fundadores de ACEAM, fallecido recientemente y le dedicamos la portada de este Boletín a uno de sus modelos más conocidos, el "Reloj Loco", que se exhibió por primera vez en la Exposición celebrada en el Corte Inglés de Callao-Madrid en el año 2008 y posteriormente en la del Casino Militar de Madrid, en donde está tomada la foto de la portada, y por último en la expo de Tres Cantos (Madrid).

En la foto que aquí insertamos, tomada en la Expo de Las Matas (Madrid), está Juan Álvarez López con los modelos que más recientemente construyó.

Como estamos a final de año aprovecho para desearos que, a pesar del COVID-19, tengamos una Feliz Navidad y que en el Nuevo Año recuperemos la normalidad.

*Antonio Valero Aicua*
*Presidente de ACEAM*

# Contenido

Página

# Dos poliedros explosionados

*Esteban Orozco Vallejo*

## Introducción

### La afición de construir poliedros

En el boletín de ACEAM nº 16 de diciembre de 2010 hay un artículo mío "**Construcción de poliedros con Meccano**", que incluye una introducción general y hace referencia a mi método de construcción de poliedros.

Pertenezco al colectivo de aficionados al Meccano, pero también al de construcción de poliedros, colectivo bastante extendido, aunque probablemente menos numeroso que el del Meccano. Sobre todo si se incluyen en éste último no solo los sistemas de construcción metálicos sino otros de construcción por ejemplo en plástico y con base electrónica o informática.

La representación tradicional de los poliedros es en forma "solidus" (es decir con todos los polígonos de las caras completos), pero también hay desde el Renacimiento la forma "vacuus", en la que los polígonos de las caras se reducen a tiras planas que conforman las aristas. Esto me dió la idea de construir con tiras de Meccano y unirlas mediante un ángulo diedro (ángulo entre los planos de las caras del poliedro) obtenido a partir de los angulares 12 ó 12 c mediante un sencillo sistema o aparato construido con Meccano. Cada dos tiras unidas por un angular sugieren entre ellas una arista virtual del poliedro.

La construcción tiene tres fases:

1. Prefabricación de los polígonos de las caras.
2. Cálculo por trigonometría esférica de los ángulos diedros.
3. Doblado de angulares 12 ó 12 c a los valores obtenidos de los ángulos diedros mediante el sencillo aparato indicado en el boletín de ACEAM nº 16.
4. Unión de las caras por los angulares de los diedros.

Un triángulo esférico se define por tres puntos A, B y C en la superficie de una esfera. El lado o arco AB se define por el ángulo AOB siendo O el centro de la esfera y se nota como c, ya que es opuesto al vértice C. El diedro A, por ejemplo es el ángulo de los planos OAB y OAC y corresponde en el poliedro al ángulo de las dos caras que se encuentran en una arista.

En el cálculo por trigonometría esférica se utiliza en general la fórmula cosA = (cosa – cosb. cosc)/senb.senc y a veces lo que se llama figura de vértice, es decir cortar los lados que concurren en un vértice por un plano perpendicular a la recta que une centro y vértice y operar sobre la pirámide resultante. Hay que tener en cuenta también que en un poliedro de aristas iguales se pueden utilizar esferas que con centro en un vértice pasen por otros varios vértices.

Los aficionados a construir poliedros los construyen en su mayoría en forma “solidus”, es decir con su cara completa y en cartulina que se pega por bordes añadidos, siguiendo a veces el “desarrollo” del poliedro. También, menos, con láminas metálicas, chapa recortable, láminas de plástico o de madera, que se sueldan o unen por sus lados. E incluso con alambrón o alambre que se suelda.

Hay también un buen sistema con nudos y varillas de plástico : el sistema norteamericano ZOME, distinto a los métodos anteriores y que también utilizo. Este sistema ZOME está presente, como material didáctico, en muchos colegios y universidades norteamericanos y europeos. Consta de unos nodos y un conjunto de varillas distintas basadas en la proporción áurea y en la raíz de 2. Es de estructura débil, pero tiene varias ventajas y permite construir rápidamente un cierto número de poliedros.
Ahora bien en realidad es un esqueleto y no se expresan las caras.

El método de construcción en cartón o cartulina es barato pero poco resistente y los métodos de construcción en chapa metálica, de madera o plástico son de difícil ejecución. La construcción en Meccano es cara, pero la ejecución es asequible y el resultado es muy resistente y de bella apariencia. Evidentemente es difícil de construir en Meccano determinados tipos de poliedros, sobre todo los no completamente regulares y las intersecciones o maclas de ellos.

No tenemos noticia de que alguien haya hecho poliedros con Meccano según nuestro método, por lo menos de una forma continua o sistemática.

La construcción de un poliedro con Meccano es en general tediosa por lo repetitiva y a veces lleva un considerable tiempo, pero el resultado es de una gran belleza. Evidentemente es un método caro, por utilizar en muchos casos cientos de tiras y angulares y miles de tornillos.

## Definiciones sobre algunos poliedros

Por ser de interés en nuestro caso vamos a referirnos a los siguientes tipos de poliedros: duales, toroidales y explosionados. Suponemos que el lector sabe lo que es un poliedro convexo, un poliedro regular e incluso un poliedro arquimediano. Un poliedro es convexo si la prolongación de sus caras no corta al poliedro. Los poliedros regulares son los que tienen un único tipo de polígono regular como cara y un único tipo de vértices. Los poliedros arquimedianos son los que tienen caras polígonos regulares de varios tipos y vértices también de varios tipos.

Hablemos ahora de los poliedros duales. Los planos perpendiculares a las líneas que unen el centro del primer poliedro con sus vértices conforman el segundo poliedro dual del primero. Los poliedros duales tienen el mismo número de aristas y el número de caras del primero es igual al de vértices del segundo y el número de vértices del primero igual al de caras del segundo.

Los poliedros toroidales son poliedros traspasados por “túneles” y a veces con espacios interiores vacíos. Por tanto estos poliedros, como se dice en la Topología son de género 1 o superiores (La Topología es la rama de las Matemáticas que estudia las formas). Los hay que tienen todas sus caras polígonos regulares y por tanto son construibles con Meccano.

Vamos a describir ahora el proceso de explosionado de un poliedro. En un poliedro convexo en el que los polígonos de todas las caras tienen lados de la misma longitud a, explosionar consiste en separar perpendicularmente las caras del poliedro inicial o núcleo hasta que su separación sea igual a la arista a. Este explosionado puede ser total (todas las caras) o parcial (no todas las caras, se excluyen uno o varios conjuntos de caras que tienen un polígono determinado). Véase un esquema del proceso de explosionado en la figura 1.


*Figura 1*

Muy similares en sus propiedades a los poliedros con caras polígonos regulares son los poliedros con rombos, que tienen las cuatro aristas iguales aunque los ángulos son sólo iguales dos a dos. En todos los poliedros de este artículo la arista se forma con la tira Meccano de 5 agujeros, que en general da lugar a poliedros de entre 30 a 70 cm de diámetro.

Nosotros vamos a hacer tres modelos en los que se incluye un poliedro interior en otro poliedro exterior. Esto es evidentemente posible sólo con modelos de poliedros tipo "vacuus" y en la práctica sólo con Meccano puede hacerse esta construcción de modo razonable.

# Poliedro explosionado de 122 caras

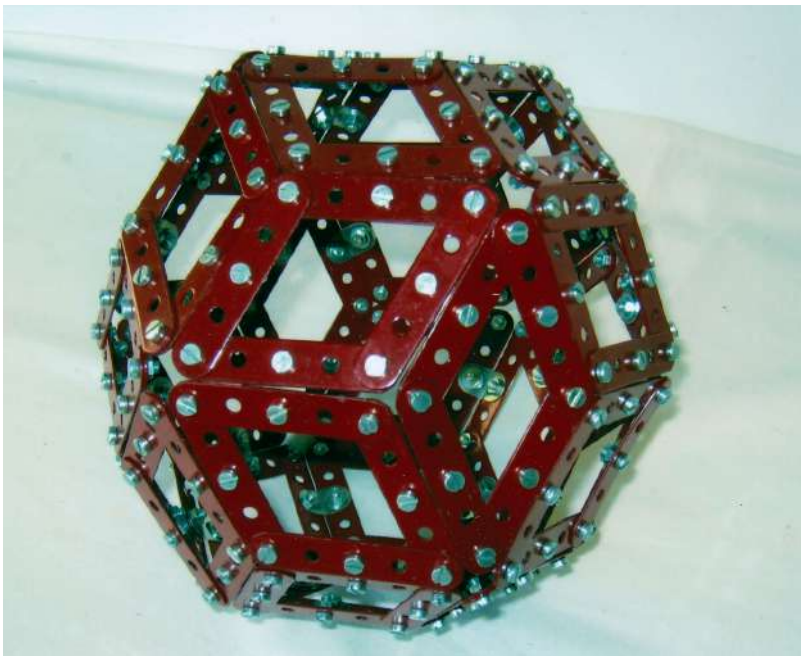
A partir de un poliedro inicial o núcleo, que puede ser el triacontaedro rómbico o el icosidodecaedro, tras un proceso de explosionado se llega al que se denomina "poliedro de 122 caras". Que el poliedro explosionado sea el mismo a partir del triacontaedro rómbico o del icosidodecaedro es debido a que ambos poliedros son duales y todos los poliedros duales tienen el mismo explosionado. No hay muchas citas en libros o artículos sobre el proceso de explosionado. Citaremos solamente el libro "Quadrivium", de Editorial Librero, páginas 178 y 179.

A continuación presentamos las fotos de cada uno de estos dos poliedros iniciales y sus tablas de características. Los rombos del triacontaedro rómbico son áureos : la diagonal menor $d_1$ es el segmento áureo de la diagonal mayor $d_2$ , siendo $d_1 = d_2 (V5-1)/2 = 0{,}618d_2$ También un dibujo general del poliedro de 122 caras y la tabla de sus características.

En este proceso que estudiamos en el poliedro de 122 caras el explosionado es total, es decir explosionan todas las caras del poliedro inicial.

A partir de ahora llamamos 122-1 al poliedro proveniente del triacontaedro rómbico y 122-2 al proveniente del icosidodecaedro.

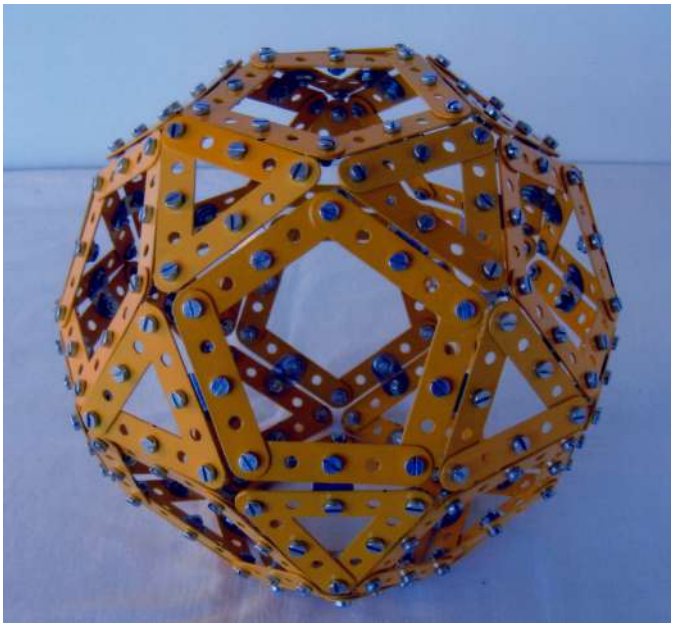
Evidentemente los diedros del poliedro 122-1 y del poliedro 122-2 son los mismos, pero comenzaremos el cálculo de los ángulos diedros del 122 con el poliedro inicial 122-1, es decir estudiando el explosionado del triacontaedro rómbico.



Triacontaedro rómbico . Número de

| Caras | Vértices | Aristas | Diedros |
|---|---|---|---|
| 30 rombos áureos (diagonales $d_1 = \Phi\, d_2$) | 32 12 de orden 5 20 de orden 3 | 60 | $D_{r,r} = 144°$ |
| Total 30 | Total 32 | Total 60 | Total 60 |

*Figura 2. Triacontaedro Rómbico*



| Icosidodecaedro . Número de | | | |
|---|---|---|---|
| Caras | Vértices | Aristas | Diedros |
| Pentágonos 12<br>Triángulos 20 | 30<br>V = 5,3,5,3 | 60 | $D_{5,3} = 142° 37'$ |
| Total 32 | 30 | Total 60 | Total 60 |

*Figura 3. Icosidodecaedro*


*Figura 4. Poliedro de 122 caras*

# Poliedro explosionado 122-1

Para el cálculo de los ángulos diedros a partir del triacontaedro rómbico estudiamos primero la explosión de un vértice de orden 5 a través de una figura de vértice, obteniendo el diedro D4,5 = 156°, 50 (es decir el ángulo diedro entre las caras cuadrados y pentágonos). Después la explosión de un vértice de orden 3, también a través de una figura de vértice, obteniendo D3,4 = 166°,51 (entre triángulos y cuadrados) y el D4,R = 165°,31 (entre los cuadrados y los rombos). Mediante el dibujo de un plano perpendicular a una arista del triacontaedro rómbico o directamente con tiras sobre el modelo vemos la altura a la que hay que colocar el cuadrado entre la abertura de dos planos explosionados contiguos.

El polígono explosionado de un polígono rojo del triacontaedro rómbico convendría que fuera también rojo en el 122-1, pero no tengo suficientes tiras de 5 agujeros rojas y los construyo con tiras de 5 agujeros azules.

Construyo primero el poliedro 122-1 con los 30 rombos áureos azules que son la proyección de los 30 rombos áureos rojos del triacontaedro rómbico. El resto de los lados (triángulos, cuadrados y hexágonos) en ZP o plateado de zinc.

Este poliedro 122-1 se ha construido en dos partes, que denominamos hemisferio norte y hemisferio sur. En el hemisferio sur establecemos la línea de sutura con angulares atornillados en un lado con tornillos distintos de latón, para reconocer después esta línea de sutura. Por ser prácticamente idénticas sólo adjuntamos fotos de los hemisferios norte y sur y métodos de unión de ellos por la línea de sutura no aquí sino en el próximo caso de la explosión de un icosidodecaedro como núcleo o poliedro inicial.

Antes de cerrar los dos hemisferios se pone dentro el poliedro inicial con varillas ya preparadas con collares para sujetar el poliedro inicial interior y el poliedro explosionado exterior. En total disponemos ocho varillas. De ellas cuatro varillas no atraviesan el conjunto de los dos poliedros interior y exterior sino que comienzan con piezas 62 o 62 B (cigüeñas) atornilladas en el exterior del poliedro interior y son luego sujetas al poliedro exterior.

En la figura 5 puede verse una foto del poliedro explosionado 122-1 teniendo en su interior el poliedro inicial : el triacontaedro rómbico rojo.

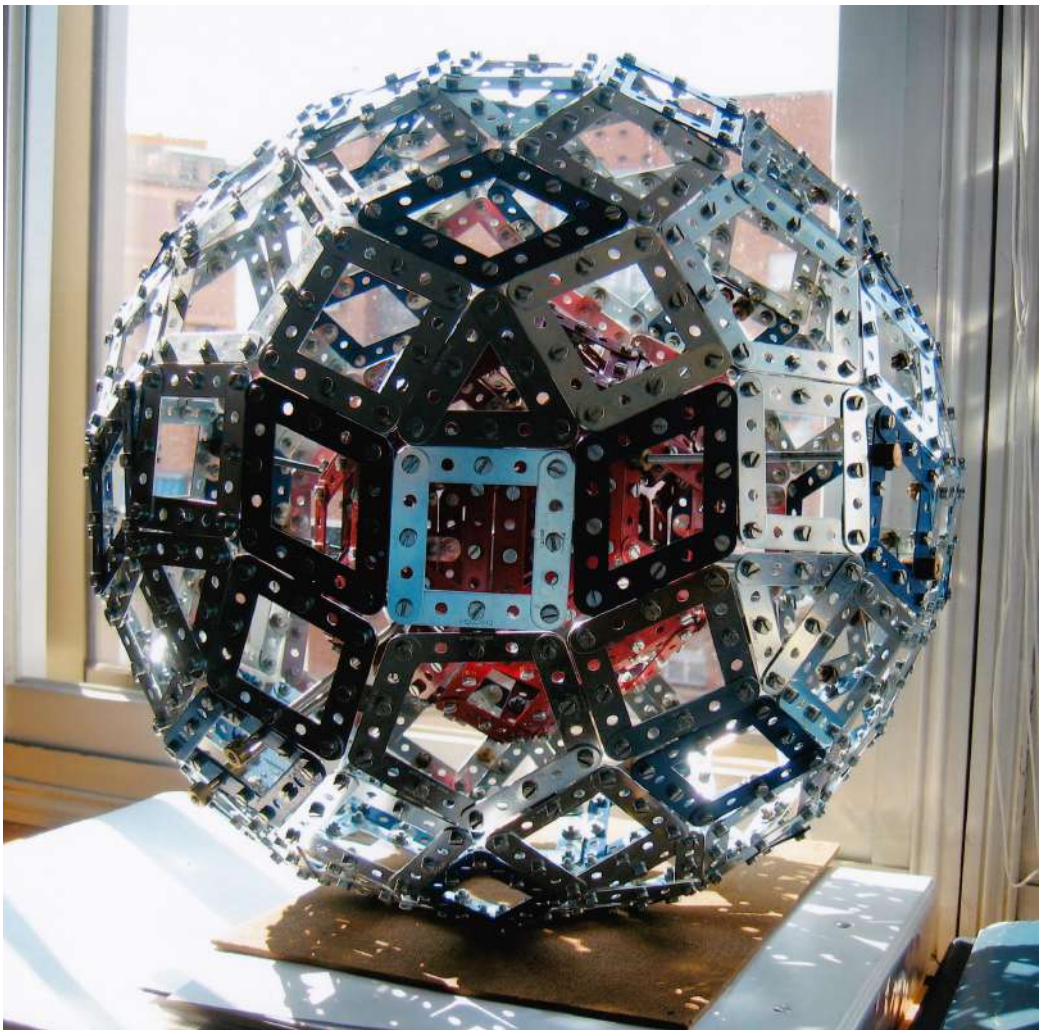


*Figura 5*

# Poliedro explosionado 122-2

Como hemos dicho denominamos poliedro 122-2 al proveniente de la explosión de un núcleo o poliedro inicial icosidodecaedro. Ahora es distinto el coloreado de las caras del 122-2. La proyección de los polígonos amarillos (triángulos y pentágonos) aquí no es azul sino también amarillo, como es más lógico. El resto de polígonos (cuadrados y rombos) va en ZP : plateado de zinc. Hacemos una tabla sobre la proyección de las caras:

| | 122 | Icosidodecaedro | Triacontaedro R. |
|---|---|---|---|
| Triángulos | 20 | 20 | - |
| Cuadrados | 60 | - | - |
| Pentágonos | 12 | 12 | - |
| Rombos | 30 | - | 30 |
| Los 60 cuadrados son sólo de unión de aberturas del explosionado 122 | | | |

Análogamente a lo indicado en relación al poliedro 122-1 este poliedro 122-2 se ha construido en dos hemisferios, preparado una línea de sutura, introducido entre los dos hemisferios el poliedro inicial con varillas preparadas y efectuado el cierre final. Aquí sí adjuntamos una foto de los dos hemisferios (figura 6) y otra dela colocación del núcleo (figura 7). Finalmente otra (figura 8) nos muestra el poliedro 122-2 exterior con su poliedro inicial interior : el icosidodecaedro amarillo.

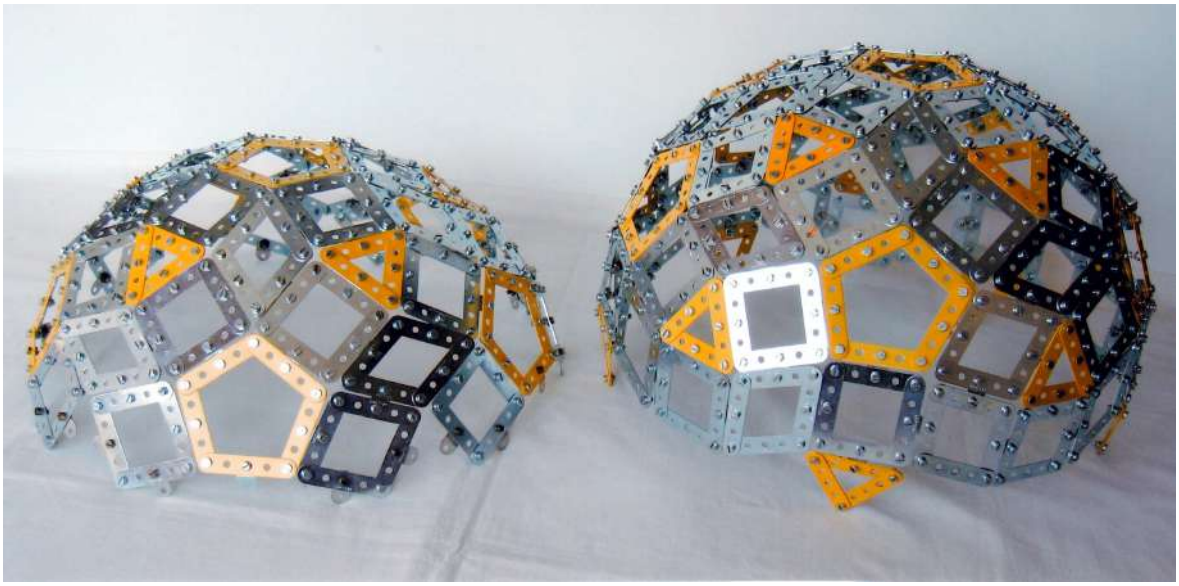


*Figura 6*

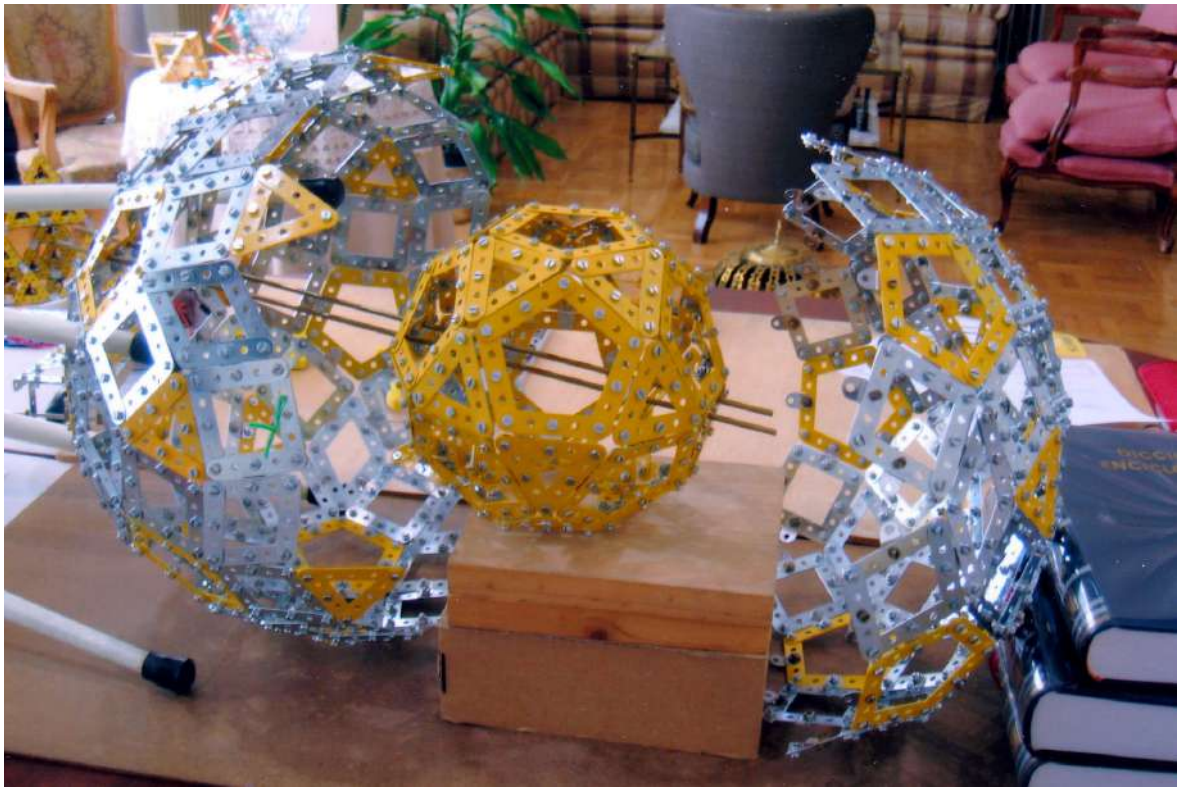


*Figura 7*

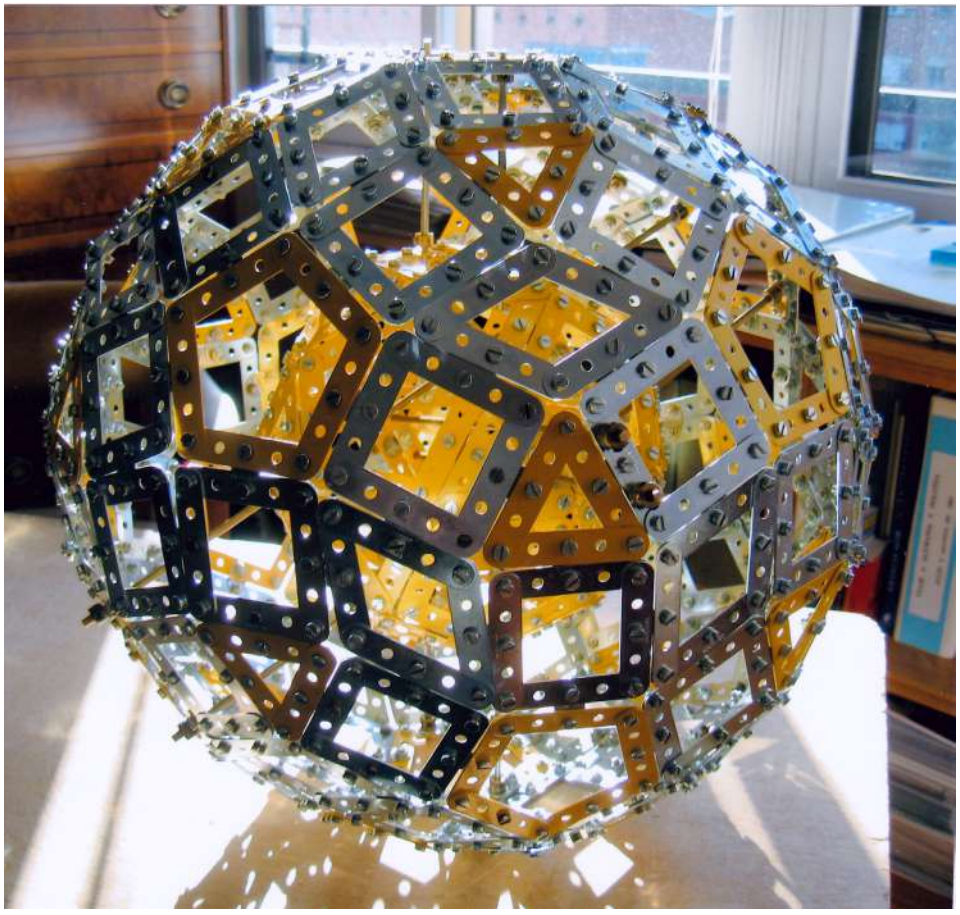
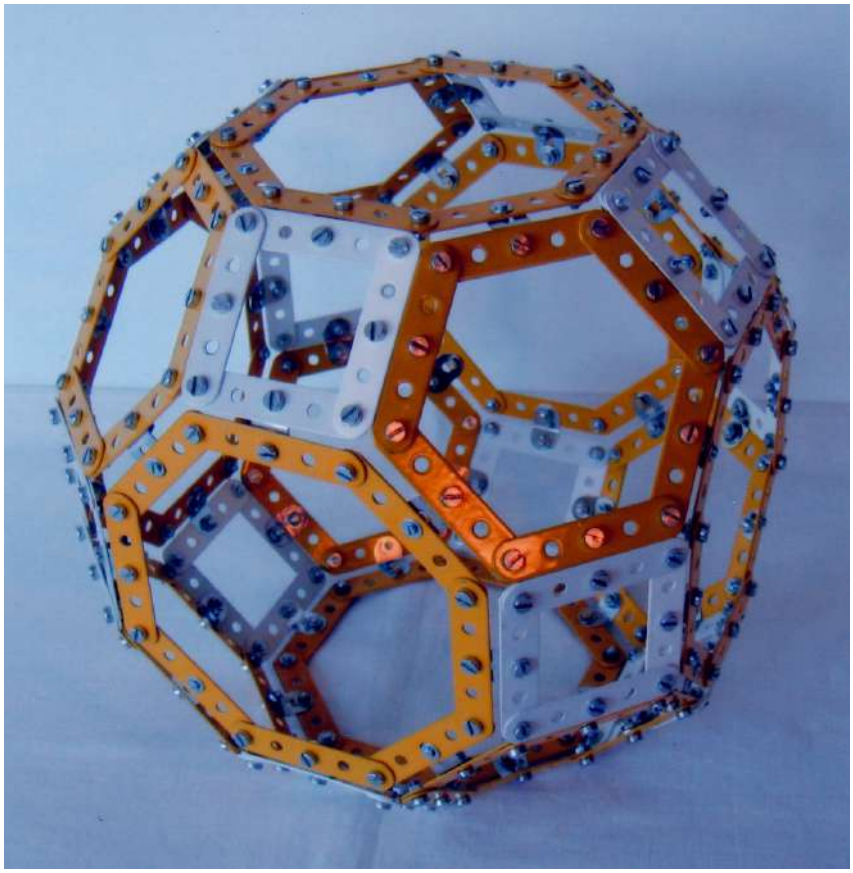


*Figura 8*

# Poliedro bitco toroidal

En este artículo, como su nombre indica, tratamos del explosionado de dos poliedros, uno de explosionado total y otro de explosionado parcial. Tras estudiar el primero : el poliedro de 122 caras, explosionado total, igual, de dos poliedros iniciales (duales entre sí), el triacontaedro rómbico y el icosidodecaedro, vamos a partir ahora como poliedro inicial del tronco-cubo-octaedro, que es el poliedro arquimediano nº 6. Adjuntamos foto (figura 9) y tabla de características del tronco-cubo-octaedro, que resumimos en adelante por su acrónimo TCO. Al poliedro explosionado total del TCO le llamamos por su forma BI-TCO, es decir poliedro BITCO, que es por supuesto convexo.



great rhomb-cub-octahedron

Poliedro arquimediano nº 6 . Tronco-cubo-octaedro

| Caras | Vértices | Aristas | Ángulos diedros |
|---|---|---|---|
| Cuadrados 12<br>Hexágonos 8<br>Octógonos 6 | Orden 3<br>V(4-6-8) | | D(4-6) = 144°,44<br>D(4-8) = 135°<br>D(6-8) = 125°,16 |
| Total 26 | Total 48 | Total 72 | Total 72 |

*Figura 9*

Pero nosotros vamos a estudiar un explosionado parcial del TCO, explosionando los cuadrados y los hexágonos y no los octógonos, poliedro al que vamos a llamar BITCO toroidal.

En Internet, en “Virtual Reality Polyhedra” hay una figura “Toroidal Polyhedra” que en principio no entendemos pero que da la impresión de que está compuesta por dos casquetes norte y sur engarzados entre sí y derivados del poliedro TCO. Luego descubrimos y comprobamos que es un explosionado parcial del TCO. Pero los huecos no explosionados están indicados y no construidos.

Volvemos a nuestro BITCO toroidal. El espacio de los cuadrados no incluidos en la explosión es la base cuadrada de un cuenco azul cuyo perímetro exterior es un octógono no regular. El cuenco azul es el túnel que horada el volumen de tipo corona esférica situado entre el TCO y el casco del BITCO. Véase la figura 10.

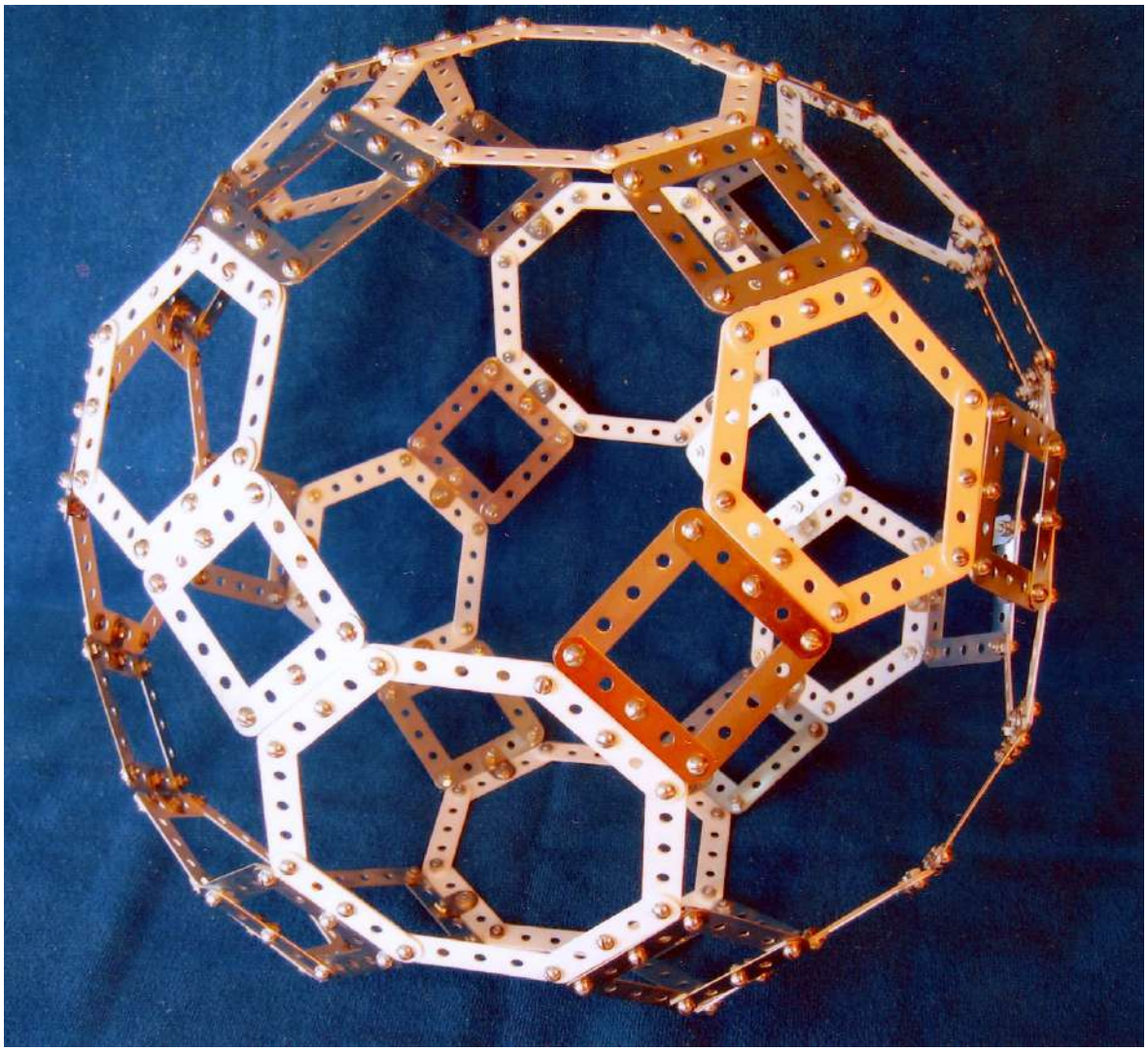


*Figura 10*

En la figura 11 se indica el proceso de colocar los cuencos azules sin base en los cuadrados del TCO no explosionados. Se ve también que se han colocadas cuatro tiras para obtener la posición del cuadrado de relleno para tapar la abertura provocada por la explosión.

En la figura 12 se ve el poliedro BITCO toroidal.

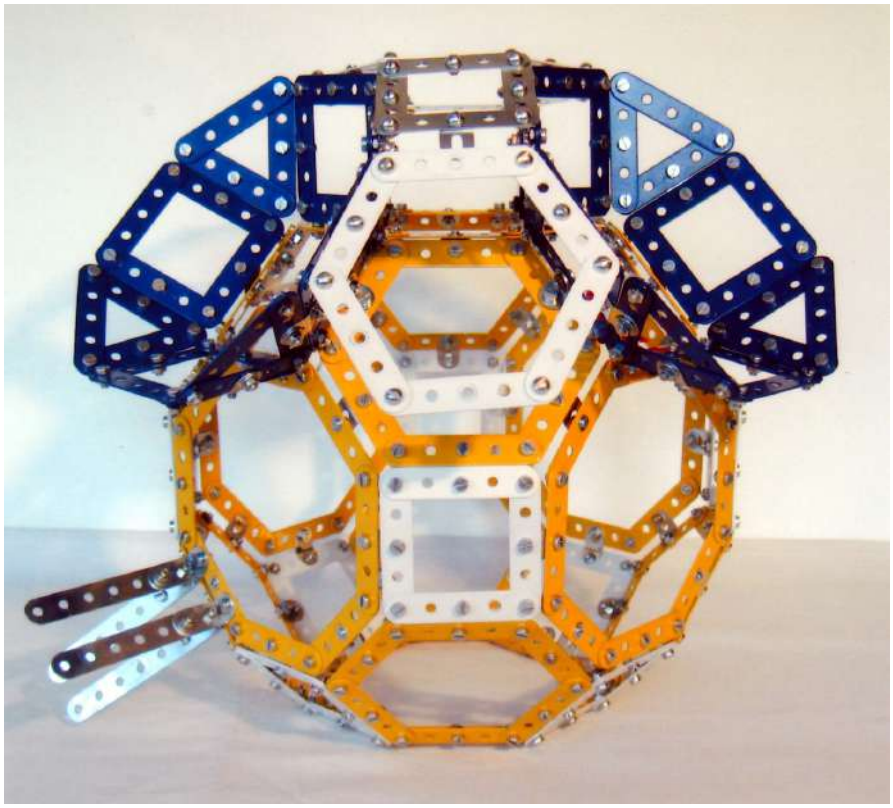


*Figura 11*

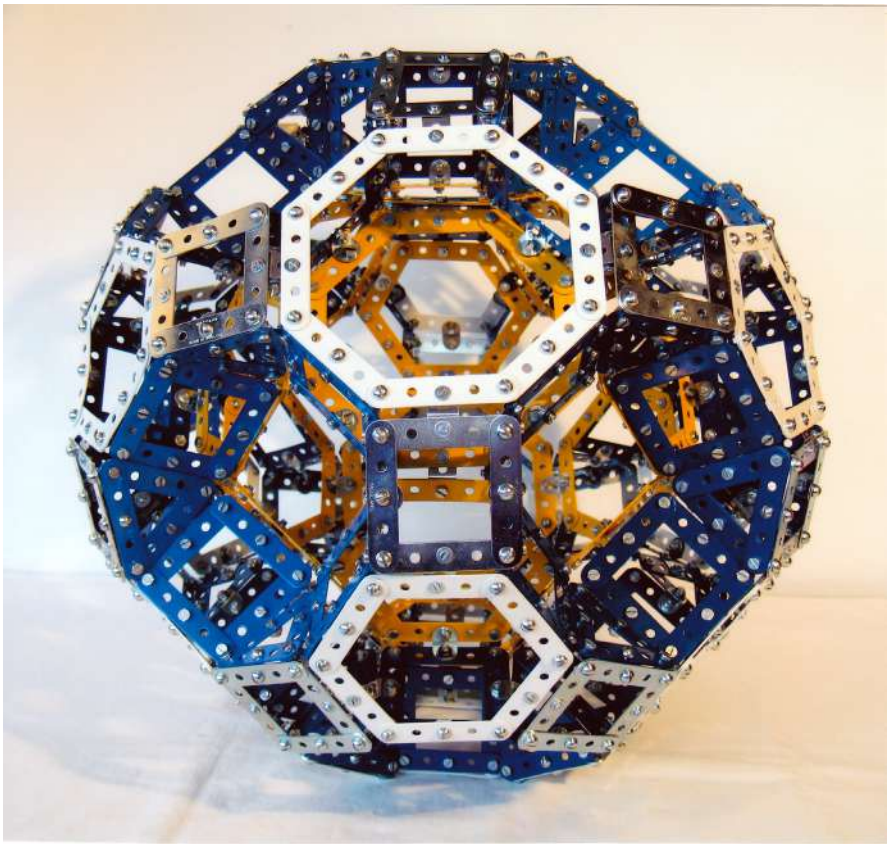
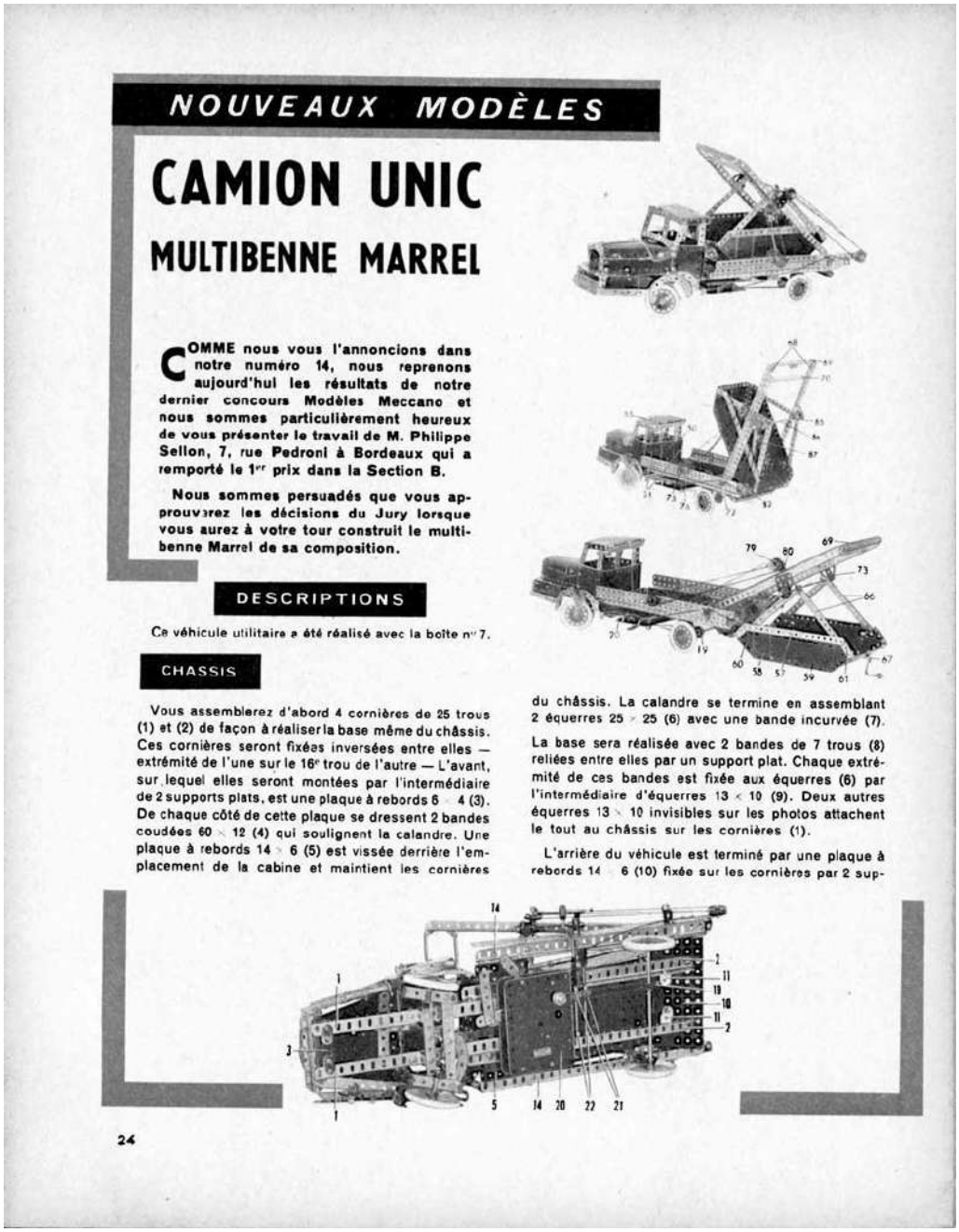


*Figura 12*

# Camión portacontenedores de escombros

*Antonio Valero Aicua*

Los camiones porta contenedores de escombros no se han prodigado en los manuales de Meccano, ni en las creaciones de los aficionados, tal vez porque son relativamente recientes estos camiones, mientras que si se han prodigado los camiones volquete, grúa, cisterna, etc.



NOUVEAUX MODÈLES

CAMION UNIC

MULTIBENNE MARREL

COMME nous vous l'annoncions dans notre numéro 14, nous reprenons aujourd'hui les résultats de notre dernier concours Modèles Meccano et nous sommes particulièrement heureux de vous présenter le travail de M. Philippe Sellon, 7, rue Pedroni à Bordeaux qui a remporté le 1er prix dans la Section B.

Nous sommes persuadés que vous approuverez les décisions du Jury lorsque vous aurez à votre tour construit le multibenne Marrel de sa composition.

DESCRIPTIONS

Ce véhicule utilitaire a été réalisé avec la boîte n° 7.

CHASSIS

Vous assemblerez d'abord 4 cornières de 25 trous (1) et (2) de façon à réaliser la base même du châssis. Ces cornières seront fixées inversées entre elles — extrémité de l'une sur le 16e trou de l'autre — L'avant, sur lequel elles seront montées par l'intermédiaire de 2 supports plats, est une plaque à rebords 6 × 4 (3). De chaque côté de cette plaque se dressent 2 bandes coudées 60 × 12 (4) qui soulignent la calandre. Une plaque à rebords 14 × 6 (5) est vissée derrière l'emplacement de la cabine et maintient les cornières du châssis. La calandre se termine en assemblant 2 équerres 25 × 25 (6) avec une bande incurvée (7).

La base sera réalisée avec 2 bandes de 7 trous (8) reliées entre elles par un support plat. Chaque extrémité de ces bandes est fixée aux équerres (6) par l'intermédiaire d'équerres 13 × 10 (9). Deux autres équerres 13 × 10 invisibles sur les photos attachent le tout au châssis sur les cornières (1).

L'arrière du véhicule est terminé par une plaque à rebords 14 × 6 (10) fixée sur les cornières par 2 sup-

24

*Foto 1*

A pesar de ello, en un Meccano Magazine, versión francesa de enero de 1959, se publicó un modelo de camión porta contenedores, que tenía un sistema de poleas y cuerdas para elevar y bajar el contenedor. En la imagen nº 1 se muestra una página de ese Meccano Magazine con ese modelo de camión porta contenedores de escombros.

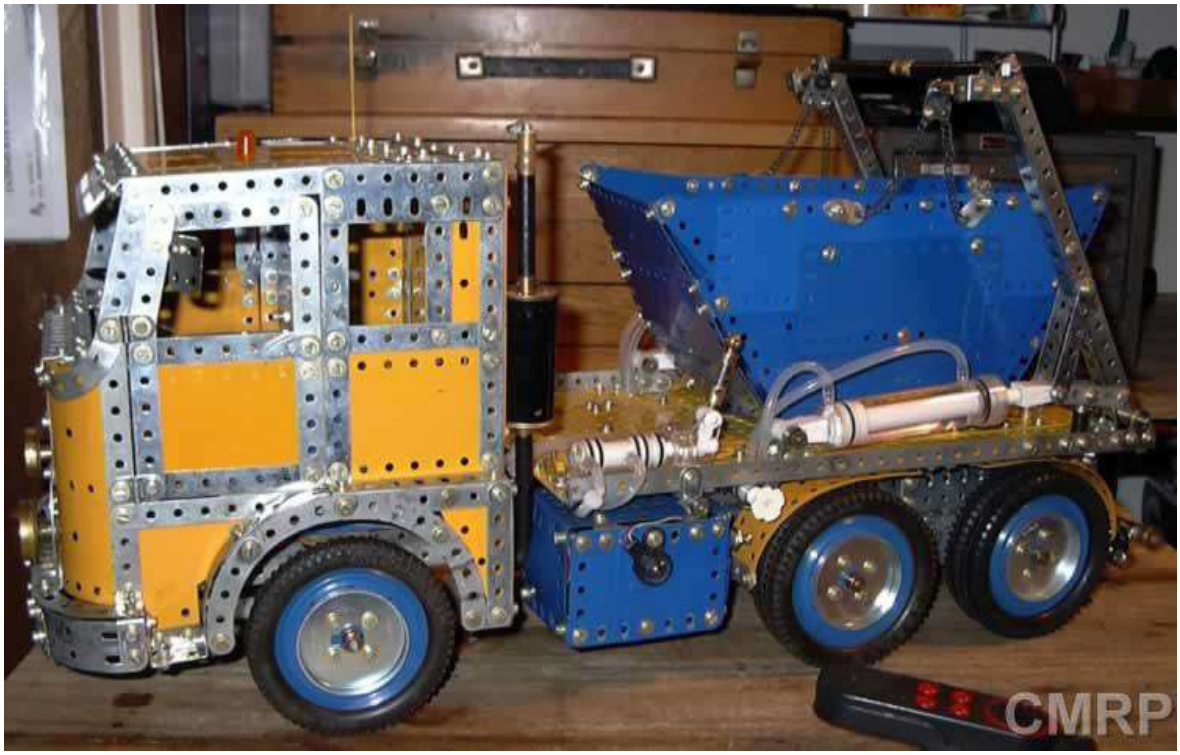



*Foto 2*

En Internet he encontrado otro camión porta contenedores de escombros que tiene la peculiaridad que lleva un sistema hidráulico real, no Meccano, para mover los brazos que suben y bajan el contenedor mediante un pequeño compresor que actúa los cilindros hidráulicos de doble efecto, que se muestra en la foto 2.

Pero el modelo que me ha servido como base para construir mi camión porta contenedores de escombros ha sido el contenido en el ModelPlan 107, diseñado por el aficionado británico Tony James, publicado por MW Order. Foto 3.

ModelPlan 107
MECCANO SKIP LORRY
by Tony James
A 10 Set Model

*Foto 3*

En mi modelo he introducido numerosas modificaciones y mejoras sobre el de Tony James, entre ellas es la total modificación del sistema de movimiento de los brazos que suben y bajan el contenedor. En el mío he imitado los cilindros hidráulicos con dos cilindros unidos (pieza especial semejante a los cortos de Meccano) de 9cm de largo y 15mm de diámetro, a un lado y otro del camión, equivalentes a los de los cilindros hidráulicos reales. Por su parte en el del ModelPlan se han construido con viguetas angulares formado un tubo cuadrado.

Por el interior de esos cilindros pasan sendas varillas roscadas, unidas a cada brazo y por el otro lado a piñones de 25 dientes. En el modelo de Tony James por el interior van ejes arrastrados por cuerdas.
Los cilindros del falso mecanismo hidráulico se aprecian en las fotos 4 y 5.

*Foto 4*

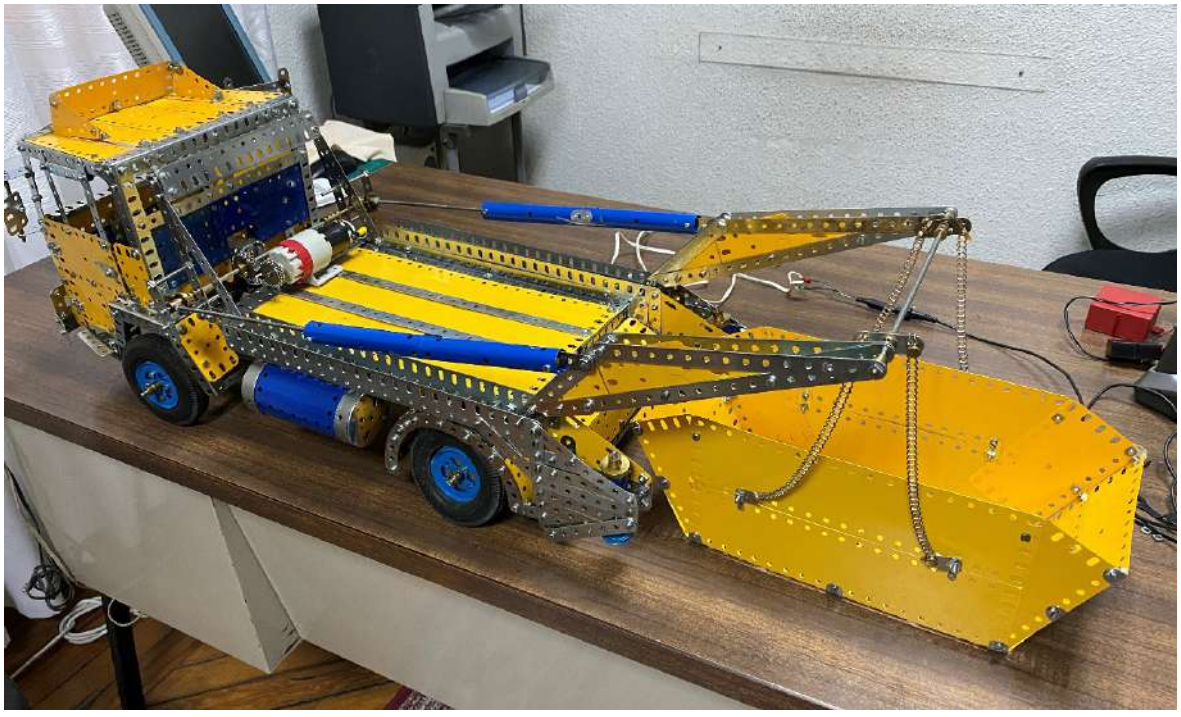


*Foto 5*

El camión porta contenedores es un camión plataforma que en su extremo posterior cuenta con dos brazos móviles, uno a cada lado de la plataforma, del que cuelgan cadenas para engancharlas a los pivotes del contenedor, con el fin de mover el contenedor de la

plataforma al suelo y viceversa. También puede voltearlo para descargar su contenido. Ello se consigue con unas pestañas que están en la parte trasera que se suben cuando se quiere volcar la carga del contenedor. Esas pestañas sujetan el contenedor y lo ponen vertical para que descargue el contenido.

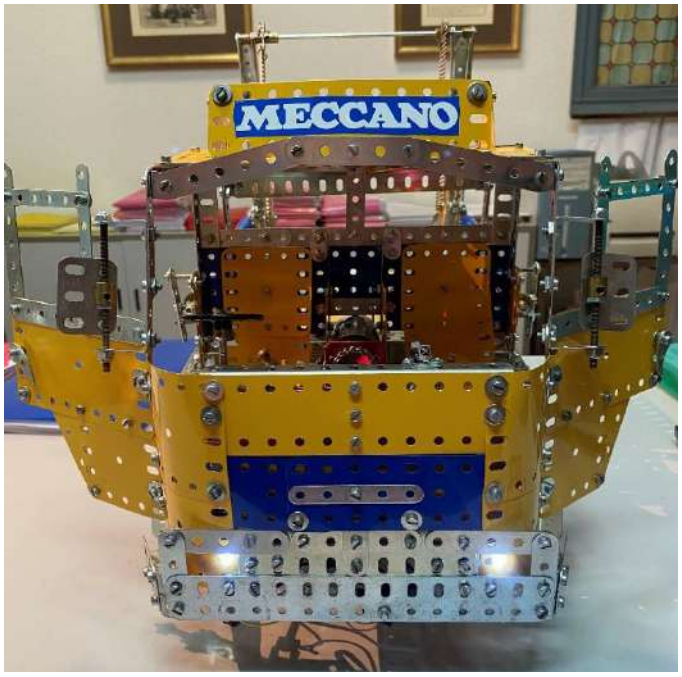



Foto 6

La foto 6 muestra la parte delantera de la cabina del camión, con las puertas abiertas.

El camión cuenta con dos faros, provistos de bombillitas led. Dentro de la cabina, además del volante, y la palanca del cambio de marchas están dos baterías de Litio, de 3,6 voltios cada una.
También se halla el circuito electrónico que controla todas las funciones del modelo, compuesto por una placa Arduino Nano, dos módulos puente H y un receptor de infrarrojos.
Todo ello se aprecia en la foto 7.

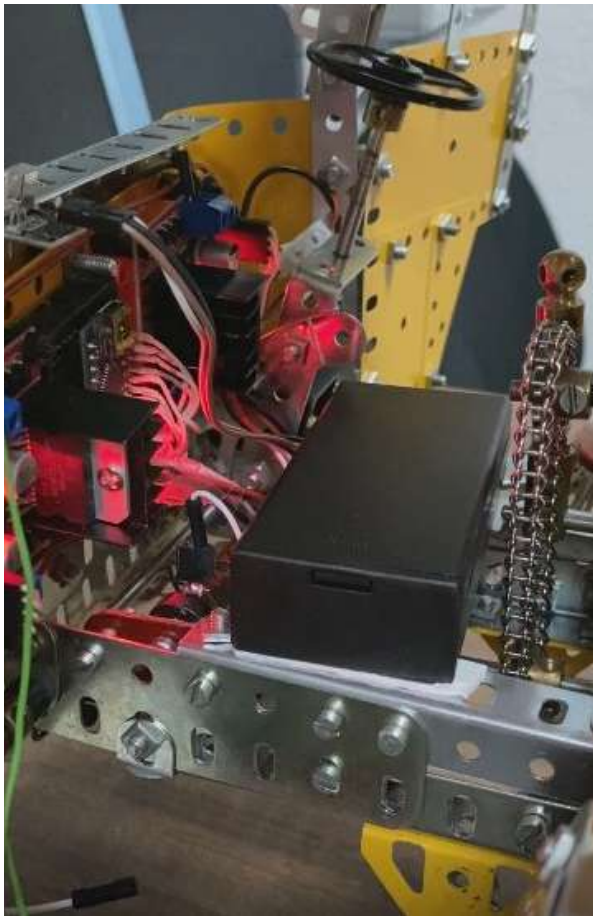


Foto 7

Foto 8

Interior de la cabina con los asientos y las placas que cubren el motor.

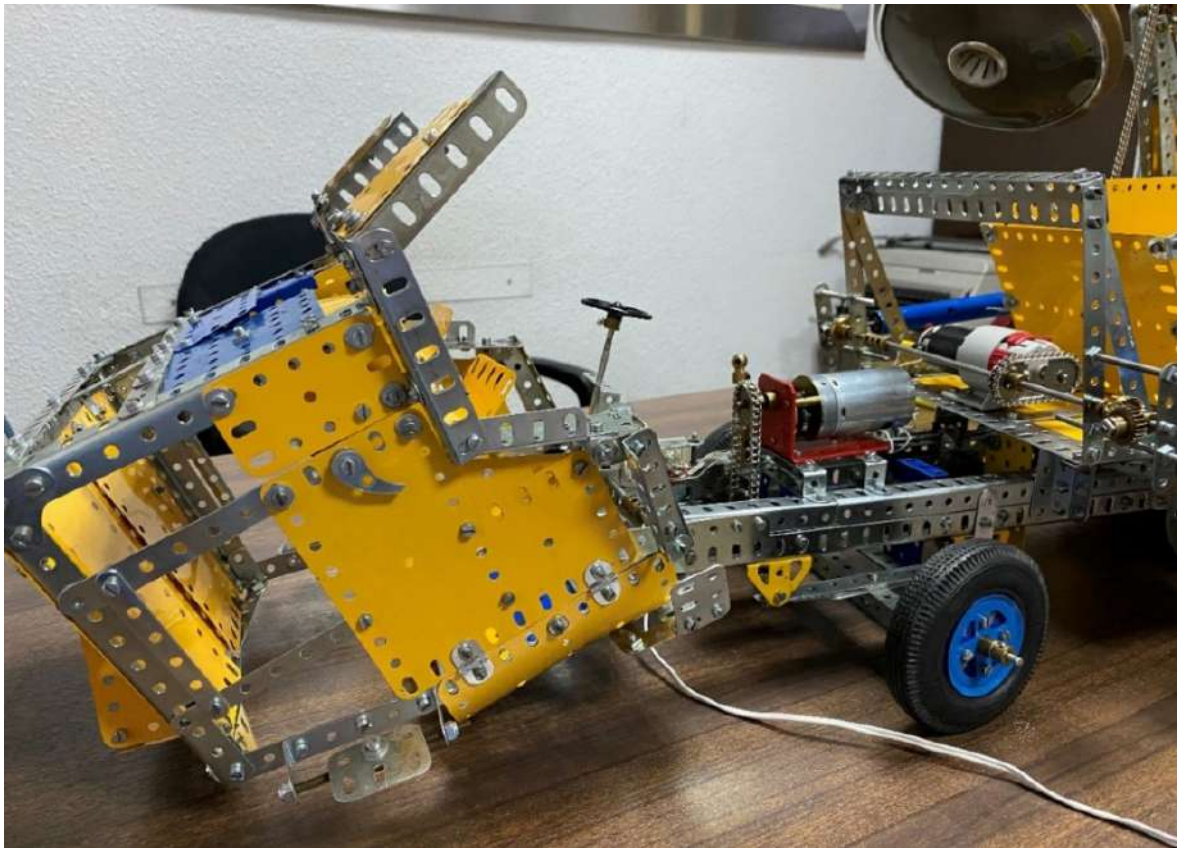


*Foto 9*

La foto 9 corresponde a la cabina que es abatible, dejando al descubierto el potente motor eléctrico con reductoras Hércules (compatible con Meccano), que mediante las ruedas de cadena transmiten a la caja de cambio y al diferencial de las ruedas traseras con fuerza suficiente.

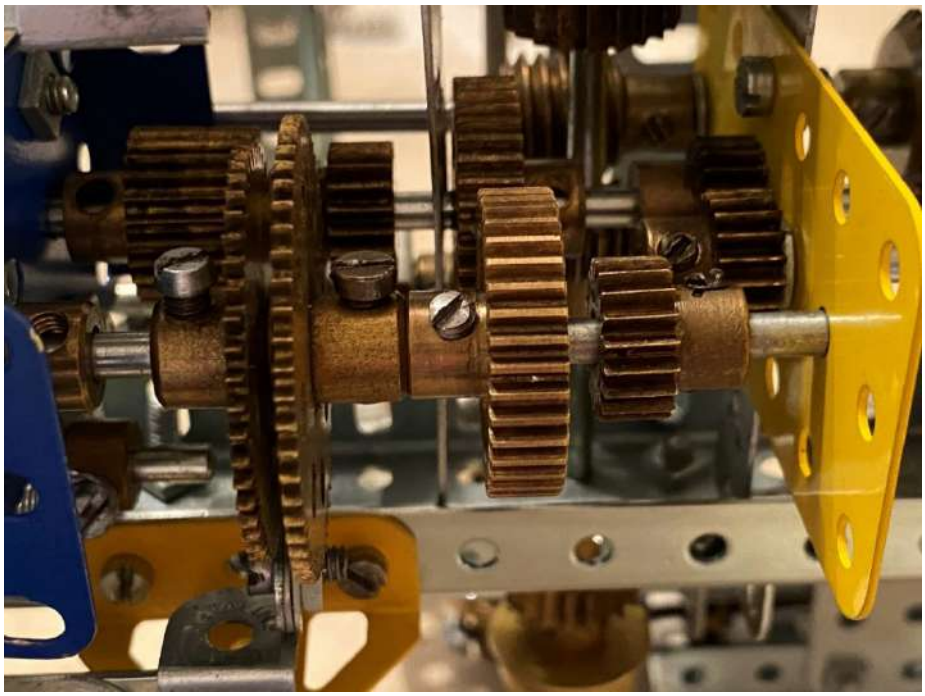


*Foto 10*

La caja de velocidades tiene dos marchas adelante, una atrás y punto muerto, que se actúan desde la palanca que esta dentro de la cabina. Foto 10.

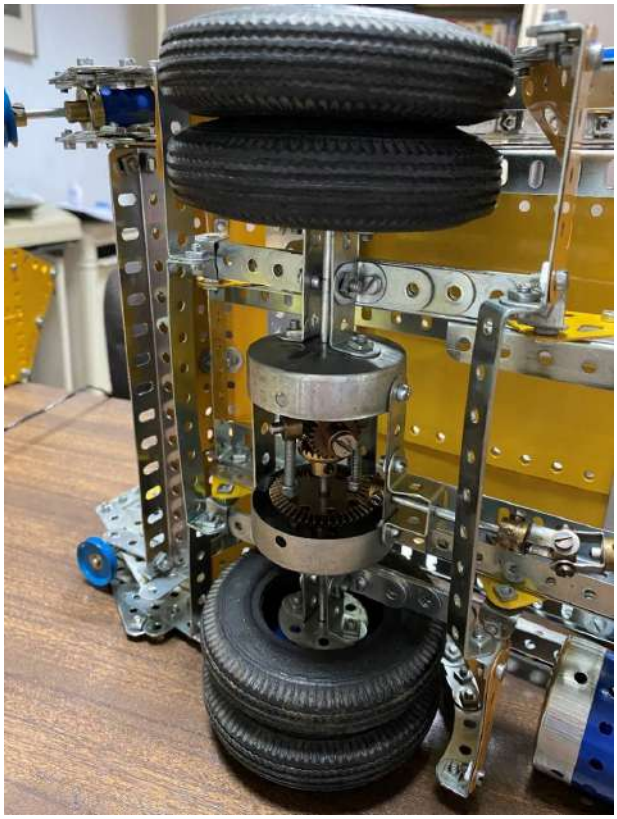


Foto 11

El eje trasero está provisto de un diferencial estándar y lleva dos ruedas a cada lado. Foto 11.

Los neumáticos no son Meccano, son neumáticos publicitarios FIRESTONE, iguales a los del eje delantero de la dirección.

El mecanismo de la dirección es de sinfín y piñón, que mueve la palanca que va a las ruedas delanteras.
Foto 12.

El mecanismo para subir o bajar los brazos utiliza un motor eléctrico de seis velocidades Richard, compatible con MECCANO, que mediante ruedas de cadena lo transmite a unas ruedas catalinas que engranan con los piñones sujetos a las varillas roscadas, al girar éstas roscan o desenroscan los cubos

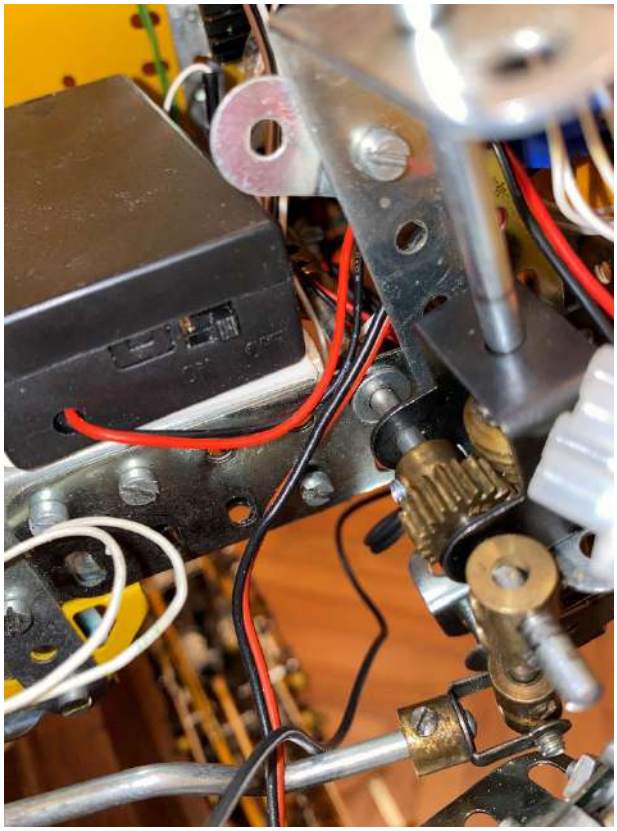


Foto 12

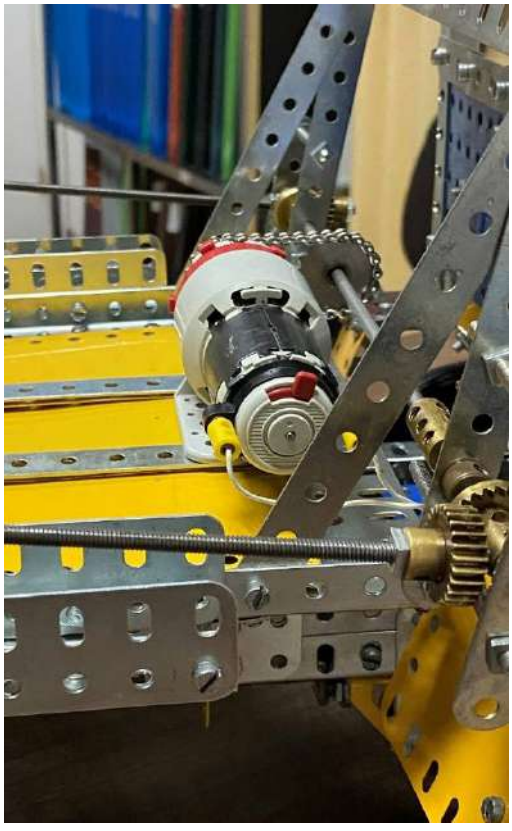


Foto 13

roscados que están en el interior de los cilindros, que hacen subir o bajar los brazos como puede verse en la foto foto 13.

El sketch de Arduino y los diagramas del cableado para hacer funcionar al camión están realizados por nuestro socio **Jesús Alonso**. Como quiera que el código es bastante largo, a cualquier lector que le pueda interesar se lo remitiría, tanto el código como los diagramas, por correo electrónico.

Con el mando de infrarrojos (foto 14), se consigue realizar las siguientes funciones: Mover el camión marcha adelante y marcha atrás y punto muerto, así como acelerar y reducir la velocidad frenándolo. También hace girar las ruedas delanteras a derecha e izquierda y enciende y apaga los faros. Además sube el contenedor a la plataforma del camión y lo baja al suelo.


*Foto 14*

Puede verse en funcionamiento este camión en un vídeo de YouTube, en el siguiente enlace: https://youtu.be/RBdpdM3XNjE

# Motores paso-a-paso

*Jesús Alonso Rodríguez*

*Hace 9 años (boletín nº18 de diciembre de 2011) veíamos una forma sencilla de utilizar motores paso-a-paso en nuestros montajes de Meccano. En aquella ocasión, se trataba de hacerlo sin utilizar electrónica. Esta vez, vamos a entrar más a fondo en este tipo de motores y veremos cómo podemos manejarlos desde un Arduino.*

Vamos a empezar por conocer qué es un motor paso-a-paso y cómo funciona. En la Figura 1, vemos un par de bobinas, con sus núcleos, conectadas en serie y con un imán entre ellas que puede girar sobre su centro. Las líneas discontinuas representan los núcleos de las bobinas, los bucles curvos representan las espiras del bobinado y el rectángulo es el imán que puede girar sobre el punto gordo que hay en su centro y que representa el eje de giro.


*Figura 1*

Si hacemos circular una corriente continua por las dos bobinas, aparecerá un campo magnético transversal al imán que obligará a este a girar 90 grados para alinearse con ese campo. En qué sentido gire el imán dependerá de la polaridad de la corriente; si hacemos que B1 sea positivo y B2 negativo, el imán girará en un sentido y si hacemos que B2 sea positivo y B1 negativo, girará en sentido contrario. A las dos bobinas y sus núcleos vamos a llamarlas “estator” y al imán que gira, “rotor”. Ahora ya tenemos un motor con estator de dos polos y rotor también de dos polos. Una vez alineado el imán, si cambiamos el sentido de la corriente, el imán volverá a girar 180 grados para alinearse de nuevo, pero no sabemos en qué sentido girará; puede hacerlo en cualquiera de los dos sentidos y cada vez en uno diferente, así que este primer motor no nos va a resultar muy práctico. Vamos a añadir un par de bobinas más.

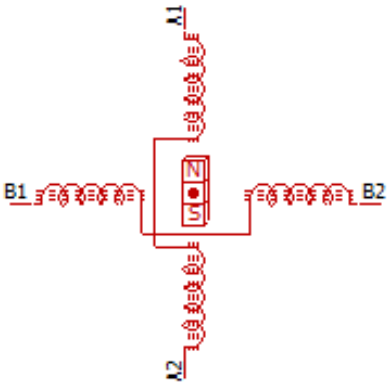



*Figura 2*

En la Figura 2, vemos  dos bobinas en horizontal y dos en vertical. Si vamos aplicando corriente a uno u otro de los bobinados y en uno y otro sentido, el rotor (el imán) girará

90 grados de cada vez y, además, ya podemos determinar en qué sentido girará. Esto ya empieza a ser un motor paso-a-paso práctico. A cada cambio de bobinado y de polaridad lo llamamos “paso”. Ahora podemos decir que tenemos un motor paso-a-paso con estator de cuatro polos y rotor de dos. También podemos decir que tenemos un motor paso-a-paso de 90 grados por paso, o de 4 pasos por vuelta. Por cierto, el nombre técnico de este tipo de motores es: “Motor de reluctancia variable”. En la Figura 3, tenemos un motor paso-a-paso desmontado para ver qué pinta tiene realmente.

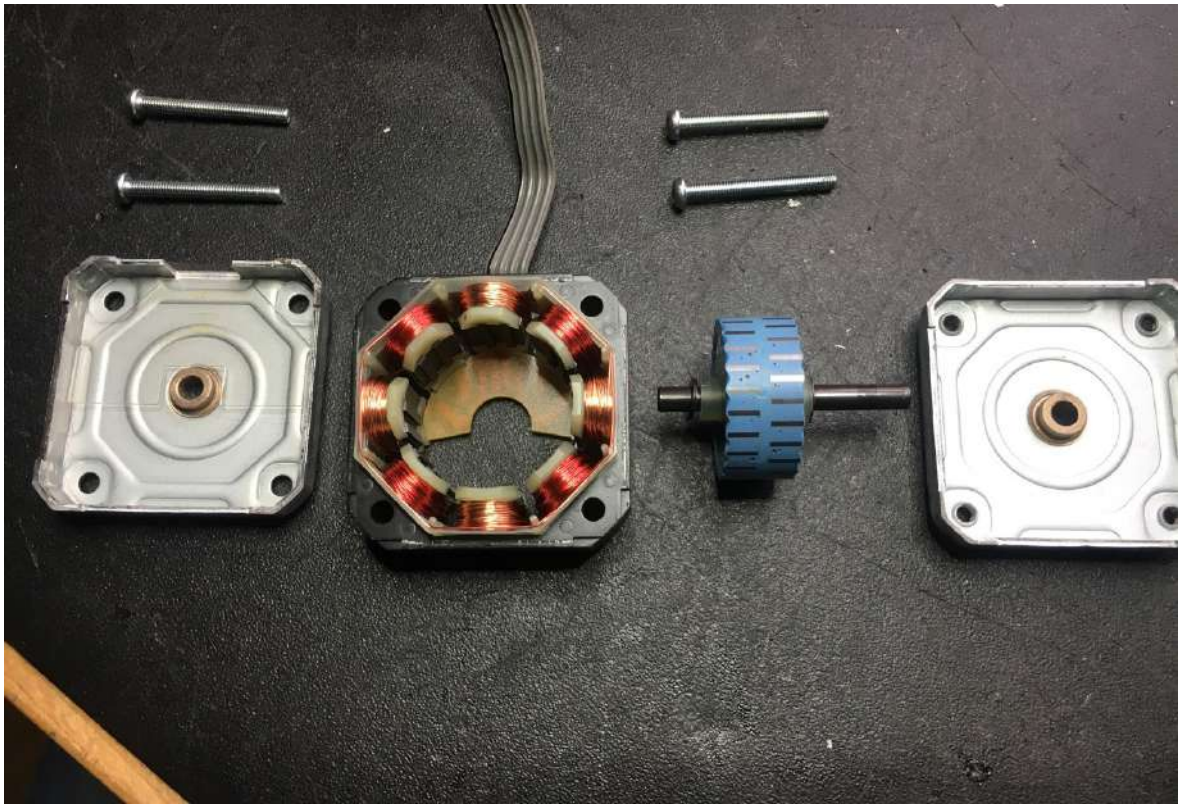


*Figura 3*

Cómo se ve en la foto, tanto el estator como el rotor tienen muchos polos. Es así porque se pretende que el rotor se mueva muy pocos grados con cada paso. Lo habitual en motores paso-a-paso es que sean de 7,5 grados por paso (48 pasos por vuelta) o de 1,8 grados por paso (200 pasos por vuelta). Seguro que a alguien se le ha ocurrido que, si hacemos circular corriente por ambos bobinados, el imán girará sólo 45 grados, en vez de 90. Pues sí, esto se conoce como “avance en medio paso” o en inglés: “micro-stepping” o “half-stepping” y se usa a veces para mejorar la precisión, por ejemplo, en impresoras, escáneres, etc. Nosotros no lo vamos a utilizar, al menos en este artículo; de momento, si queremos pasos más cortos, pondremos una reductora a la salida del motor.

Y ahora es el momento de ir a la pila de boletines antiguos, buscar el número 18 y releerse el artículo de hace 9 años (si alguien no tiene el boletín, que me lo pida por correo electrónico o por Whatsapp y se lo mando en formato pdf). A partir de aquí, vamos a darle ya a la electrónica, pero antes veremos un par de esquemas de conexionado de las bobinas en motores paso-a-paso de cinco y de seis cables.

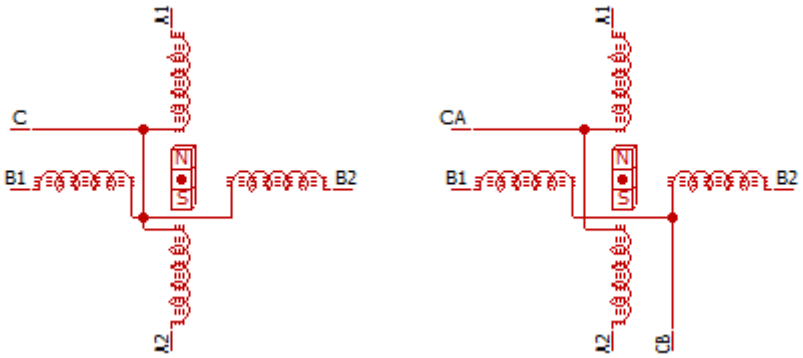



*Figura 4*

En la Figura 4, vemos a la izquierda un motor de cinco cables y a la derecha, uno de seis. En ambos casos, los terminales marcados como “C” o “CA” y “CB” corresponden al punto central de las dos bobinas que forman cada bobinado; en casi todos los motores, estos cables suelen ser de color rojo. Esta forma de conectarlo nos va a permitir utilizar medio bobinado de cada vez, utilizando los terminales “C” (común) o utilizar los bobinados completos, dejando sin conectar los terminales “C”. En el artículo del boletín nº 18 sólo podía utilizar motores de este tipo porque no podía invertir el sentido de la corriente; ahora disponemos de puentes en “H” (ver artículo del boletín de junio) que nos permiten hacer circular la corriente en cualquier sentido, por lo que ya podemos utilizar motores paso-a-paso de cuatro cables. Al utilizar cada bobinado completo, podremos obtener mayor par y trabajar con un voltaje más alto.

En la fotografía de la Figura 3, se ven 8 bobinas en el estator; en realidad, se trata de dos bobinados, ya que las bobinas van conectadas en serie cuatro a cuatro, de forma alternada; de hecho, el esmalte del hilo de cobre es de color rojo en cuatro de ellas y de color cobre en las otras cuatro. Esta fotografía corresponde a un motor de cuatro cables; en uno de 5 o de 6, las bobinas irían colocadas de forma diferente.

Vemos, por tanto, que existen dos formas de gobernar un motor paso-a-paso: invirtiendo el sentido de la corriente o no invirtiéndolo; en este último caso, el motor tendrá que tener tomas centrales y, por tanto, 5 o 6 cables. Si usamos un motor de 4 cables, no tendremos más remedio que realizar inversiones de corriente en los bobinados, pero también podemos utilizar esta técnica en motores de 5 o 6 cables, simplemente, no utilizando los cables comunes (normalmente, de color rojo).

Vamos a empezar por lo más sencillo. Quienes hayan adquirido el kit de iniciación de Arduino, se habrán encontrado con un motor paso-a-paso, junto con su tarjeta controladora, similar al que se muestra en la Figura 5.

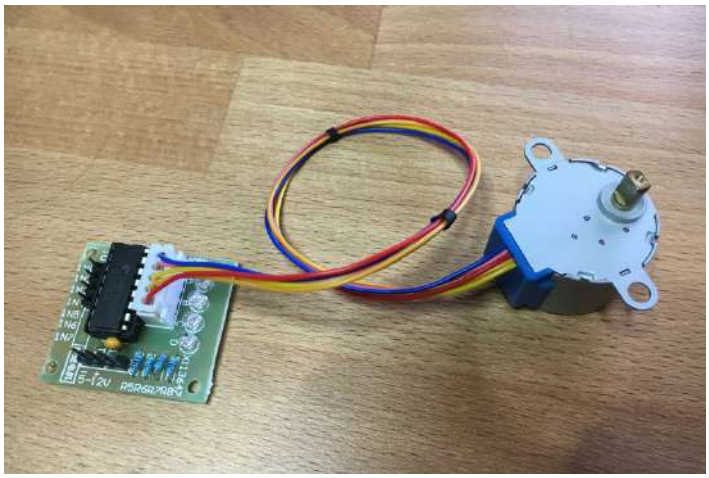


*Figura 5*

No es un motor muy bueno, pero se puede comprar por Internet, en tiendas chinas, por un par de euros con tarjeta controladora y todo. A diferencia de lo que ocurre con la mayoría de motores paso-a-paso, si intentamos girar el eje de este motor con la mano, estando sin conectar, veremos que va muy duro; la razón es que se trata de un motor de pocos polos y poco par al que han acoplado una reductora para que los pasos sean más cortos y presente un mayor par. Se trata de un motor de 5 cables, por lo que no necesitamos puentes en "H" para manejarlo; de hecho, la tarjeta controladora que trae se basa en un circuito integrado ULN2003 que no usa puentes en "H" sino que contiene, en su interior, 7 transistores "darlington", de los que utilizaremos 4. Podemos conectarlo a un Arduino de la siguiente forma:

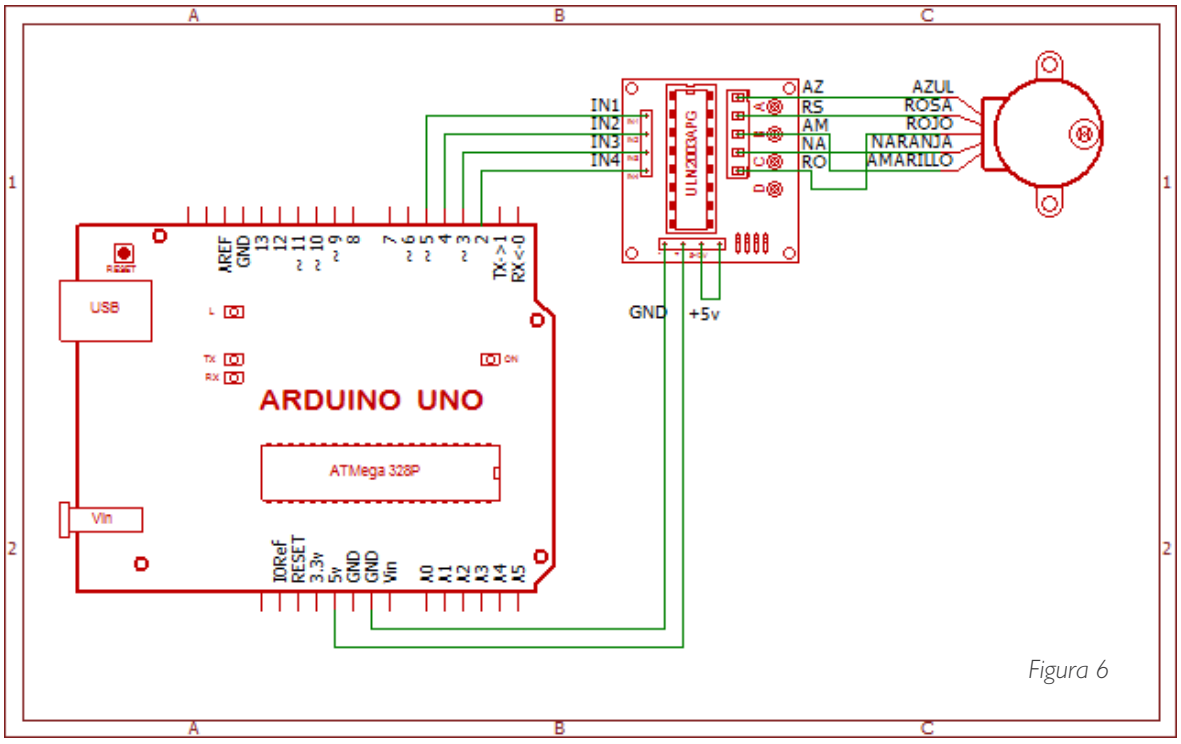



*Figura 6*

Como es un motorcillo que funciona a 5 voltios y consume muy poca corriente, lo alimentamos directamente de la salida "5v" del Arduino. Hemos conectado D5 a IN1, D4 a IN2, D3 a IN3 y D2 a IN4. En realidad, podríamos haber utilizado cualesquiera otras salidas digitales, ya que al escribir el programa definiremos por qué salidas manejamos el motor.

Ahora, sólo tendremos que ir enviando impulsos en secuencia a través de estas salidas y el motor avanzará un paso por cada impulso que enviemos. Cómo estamos utilizando el cable común (el de color rojo), sólo usaremos medio bobinado, es decir una bobina, para cada paso y no necesitaremos invertir el sentido de la corriente.

Luego veremos algo de programación, pero primero vamos a ver otros motores paso-a-paso más "clásicos" y cómo conectarlos, ya que la programación será igual en todos los casos. En la fotografía de la Figura 7, vemos varios motores paso-a-paso de distintos tipos. Unos son de cuatro cables y otros de cinco. Lo habitual es que los cables que no son rojos sean: Negro, Marrón, Naranja y Amarillo. El negro y el marrón son los extremos de uno de los bobinados y el naranja y el amarillo los extremos del otro. La secuencia: "Negro-Naranja-Marrón-Amarillo" hará girar el motor en un sentido y "Amarillo-Marrón-Naranja-Negro" le hará girar en sentido contrario. En los que tienen los cables agrupados en una cinta, ya suelen venir colocados de manera que los impulsos se apliquen en el orden en que

están en la cinta, en uno u otro sentido. Cuando decimos “aplicar un impulso”, nos referimos a poner ese cable a positivo mientras que los demás permanecen a negativo.

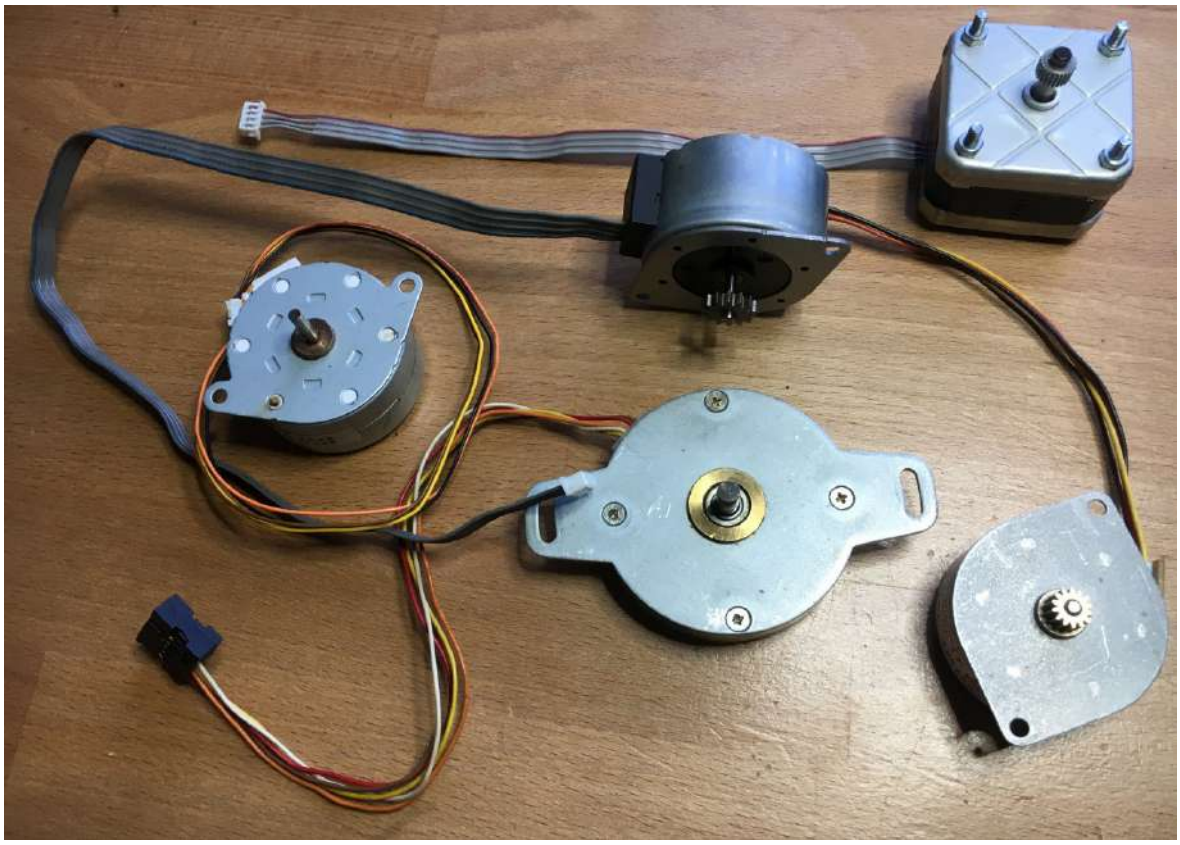


*Figura 7*

Como os habéis releído el artículo del boletín de hace 9 años, habréis visto cómo localizar los cables cuando los colores no sean los indicados, así que ya no hace falta que lo repita aquí. En lo sucesivo, consideraré que trabajamos con un motor que tiene los cables de los colores normalizados. Vamos a ver cómo conectamos al Arduino un motor de cuatro cables. Aquí ya necesitaremos un doble puente en “H”, así que vamos a utilizar el módulo que usamos en el artículo del boletín de junio y que vemos en la foto de la Figura 8.

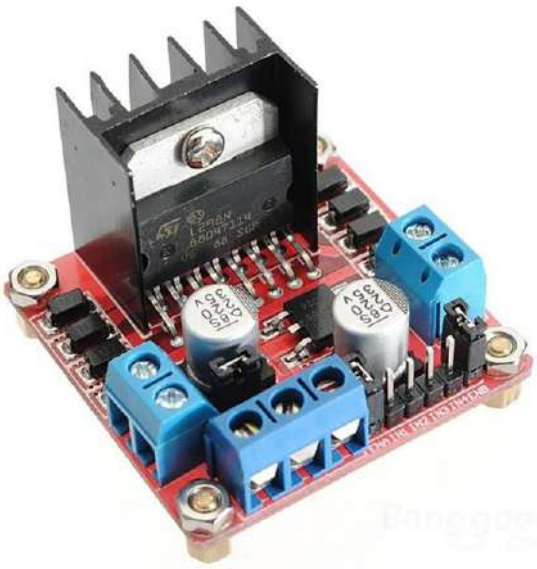


*Figura 8*

Utilizando este módulo, el esquema de conexión podría ser el siguiente:

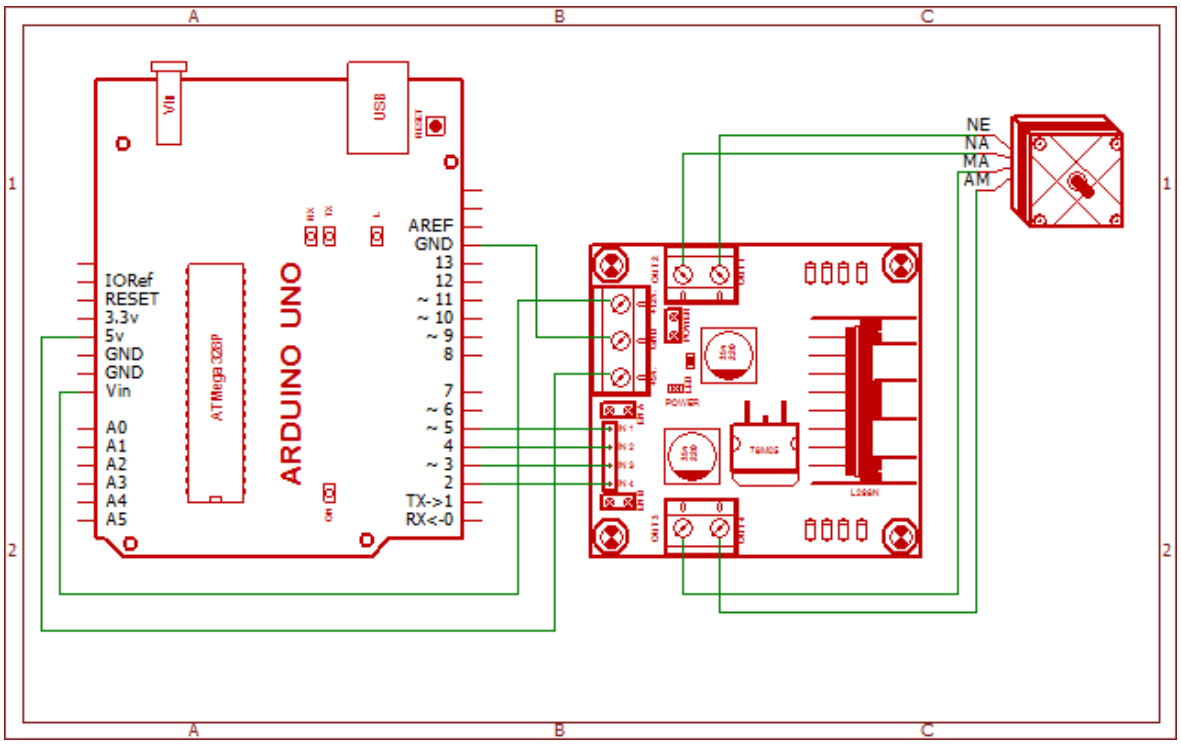



*Figura 9*

Igual que antes, hemos conectado D5 a IN1, D4 a IN2, D3 a IN3 y D2 a IN4. Las salidas del módulo van: OUT1 a Negro, OUT2 a Naranja, OUT3 a Marrón y OUT4 a Amarillo. Suponemos que el motor funciona a más de 5 voltios, así que lo alimentamos desde Vin.

Ahora vamos a empezar a escribir ya un poquito de código:
Para hacer las cosas bien desde el principio, empezamos por definir las conexiones como constantes:

```
#define PASO1 5
#define PASO2 4
#define PASO3 3
#define PASO4 2
```

También vamos a definir otra constante que utilizaremos para controlar el tiempo, en milisegundos, que dura cada paso. Como ejemplo, he puesto 20 milisegundos:

```
#define TPASO 20
```

Cuanto más alto sea este valor, más lento se moverá el motor y al revés, cuanto más bajo sea, más rápido se moverá. El valor mínimo, sin embargo, habrá que determinarlo experimentalmente para cada motor, ya que si bajamos mucho este valor, puede que el motor no se mueva. Si queremos que el programa pueda variar la velocidad del motor, podemos meter el valor de TPASO en una variable en lugar de en una constante; a mayor valor de TPASO, más despacio se moverá.

Ahora vamos a declarar dos variables globales, así que lo hacemos antes de cualquier función:

```
byte pasos[4]={PASO1,PASO2,PASO3,PASO4};
int paso=0;
```

La variable “pasos[4]” es un vector que contiene las salidas por donde tenemos que ir metiendo los impulsos para cada paso. La variable “paso” es el índice para irnos moviendo sobre este vector.
Y ya llegamos a la función de inicialización:

```
void setup() {
  for(int i=0;i<4;i++){
    pinMode(pasos[i],OUTPUT);
    digitalWrite(pasos[i],LOW);
  }
  // ...
}
```

De momento, sólo hemos inicializado las salidas, luego haremos alguna cosita más.
A partir de aquí, ya estamos listos para funcionar. Crearemos una función a la que llamaremos cada vez que queramos que el motor avance un determinado número de pasos en uno u otro sentido:

```
void pasoMotor(unsigned int nPasos,boolean sentido){
  // Avanza o retrocede nPasos
  // segun sentido sea true o false.
  while(nPasos>0){
   digitalWrite(pasos[paso],LOW);
   if(sentido){
      paso++;
      if(paso>3){paso=0;}
    }else{
      paso--;
      if(paso<0){paso=3;}
    }
    digitalWrite(pasos[paso],HIGH);
    delay(TPASO);
   nPasos--;
  }
}
```

Llamaremos a la función pasándole dos parámetro: el número de pasos que queremos que avance (“nPasos”) y en qué sentido (“sentido”). He definido el primer parámetro como un “unsigned integer”, es decir 16 bits, por lo que puede contener cualquier número entero positivo entre “0” y “65535”. Si la llamamos pasándole “0”, retornará sin hacer nada; si la llamamos pasándole “65535”, el motor se tirará 22 minutos dando vueltas (con TPASO = 20 milisegundos). El segundo (“sentido”) es de tipo “boolean”, así que sólo puede tener dos valores: “true” y “false”; si es true, el motor avanzará; si es false, retrocederá.

Si hemos puesto el valor de TPASO en una variable para poder alterar la velocidad del motor, como argumento de la sentencia “delay” pondremos la variable, claro, no la constante.

Y ya tenemos todo lo necesario para manejar nuestro montaje de MECCANO con un motor

paso-a-paso; copiamos todo este código en nuestro programa y, desde “void loop()” vamos llamando a la función cada vez que queramos mover el motor.

A partir de la primera vez que llamemos a la función, el motor estará recibiendo siempre corriente, aunque esté parado. La mayoría de motores paso-a-paso (o sea, todos menos el del kit de Arduino) quedan libres cuando no reciben corriente y no se mantienen en la posición en la que se los dejó. Si estamos usando el motor para mover, por ejemplo, el brazo de un robot o la pluma de una grúa, haría muy mal efecto que, al terminar de moverse, se cayera por su peso porque el motor ha dejado de sujetarlo. No obstante, puede haber aplicaciones en las que queramos que el motor quede libre después de moverse; en ese caso, después de cada movimiento, podemos hacer:

```
digitalWrite(pasos[paso],LOW);
```

¿Y ya hemos terminado? Pues no. Por desgracia, la vida nunca es tan fácil.

Imaginemos que hemos utilizado el motor paso-a-paso para mover el carro de una grúa, o el brazo de un robot. Cuando encendemos nuestro montaje, el carro, el brazo o lo que sea puede estar en cualquier posición, donde se haya quedado cuando lo apagamos y no tenemos forma de saber en qué posición está. Si queremos que nuestro sistema tenga siempre controlada la posición del brazo (o de lo-que-sea), que para eso hemos puesto un paso-a-paso y no un motor normal, tendremos que llevarlo primero a un lugar conocido que tomaremos como referencia. Normalmente, ese lugar se coloca en uno de los extremos del recorrido; digamos, al final del sentido en el que el motor retrocede.

¿Os acordáis de las impresoras antiguas que, cuando las encendías, movían un poco el cabezal hacia la derecha y luego lo llevaban totalmente a la izquierda? Pues ahora ya os estaréis imaginando por qué hacían eso. Resulta que el cabezal de impresión se movía con un motor paso-a-paso.

Este punto de referencia se suele llamar “posición home” pero como estamos en España, yo lo voy a llamar “casita”. Cuando arrancamos nuestro montaje, lo primero que tenemos que hacer es llevar lo-que-sea que mueve el paso-a-paso a casita. ¿Cómo sabemos que está en casita? Pues porque ponemos un fin de carrera, un fotoacoplador o un potenciómetro que nos da un valor diferente según donde esté. Tenemos que crear una función que, cuando la llamemos, nos devolverá un valor boolean, que será “true” si lo-que-sea está en casita y “false” si no está. Supongamos que lo-que-sea que hayamos puesto, cuando se activa, nos hace baja la entrada digital 8. He llamado a la función, “casita”:

```
boolean casita(){
  if(digitalRead(8)==LOW){return true;}
  else{return false;}
}
```

O supongamos que es algo analógico que tenemos que leer por una de las entradas analógicas y, cuando está activado, nos da un valor menor que un determinado UMBRAL (que habremos definido como constante). Otra versión de “casita”:

```
boolean casita(){
  if(analogRead(A0)<UMBRAL){return true;}
  else{return false;}
}
```

Y ahora, creamos otra función que nos manda lo-que-sea a casita:

```
void aCasita(){
  pasoMotor(10,true);
  while(!casita()){
    pasoMotor(1,false);
  }
}
```

¿Por qué empezamos por hacer “pasoMotor(10,true)”? Pues porque puede que estemos ya en casita o más allá y nunca llegaríamos, pero también, porque no conocemos la posición angular relativa del motor con respecto al vector “pasos[4]”, y esta primera llamada nos lo sincroniza. Ahora, podemos reescribir nuestro “void setup()” para que inicialice el sistema correctamente:

```
void setup() {
  for(int i=0;i<4;i++){
    pinMode(pasos[i],OUTPUT);
    digitalWrite(pasos[i],LOW);
  }
 aCasita();
 // ...
}
```

A partir de aquí, podemos guardar en una variable la posición del motor, incrementándola cada vez que avancemos y decrementándola cada vez que retrocedamos; de esta forma, siempre sabremos dónde está lo-que-sea que estemos moviendo con el motor.

¿Y con esto hemos terminado? Pues no del todo. En realidad, no hace falta utilizar un Arduino para manejar un motor paso-a-paso. Existen circuitos que se denominan “secuenciadores” y que nos hacen el trabajo que aquí hace la función “pasoMotor”. En la Figura 10, vemos la foto de uno de estos circuitos, el más usado entre aficionados, que se denomina “EasyDriver”. Está pensado para manejar motores de cuatro cables. Los dos bobinados se conectan en los contactos de arriba a la izquierda, marcados “A” y “B” y se maneja a través de los contactos de abajo a la derecha, marcados “GND”, “STEP” y “DIR”. Cada vez que recibe un impulso por “STEP”, hace avanzar o retroceder el motor un paso; si “DIR” es alto, avanza; si “DIR” es bajo, retrocede. Este circuito puede manejar muchos tipos de motores, limita la corriente que llega a las bobinas e, incluso, puede hace “half stepping”.

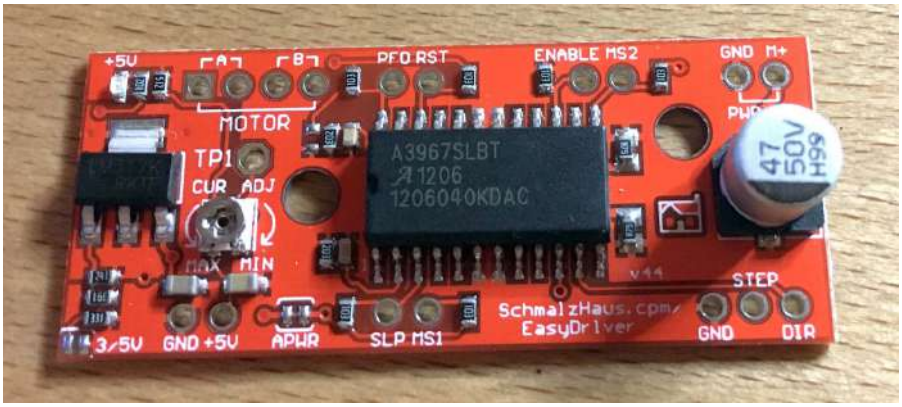



Figura 10

Y ahora sí, ya hemos terminado. Como siempre, si alguien tiene alguna pregunta, estaré encantado de responder a través del chat de Whatsapp o en la dirección de correo: **jalonsovivienda@gmail.com**

# Utilizando servos en Meccano

*Jesús Alonso Rodríguez*

*Hemos estado viendo cómo utilizar motores bipolares y paso-a-paso en nuestros montajes de Meccano. Ahora vamos a ver cómo podemos utilizar servos y controlarlos tanto desde una radio convencional de telemando como desde un Arduino*

A estas alturas, casi todos los aficionados al Meccano saben lo que es un servo pero, por si acaso, vamos a explicar un poquito no sólo qué es un servo, sino y sobre todo, cómo funciona. Llamamos "servo" a los pequeños motorcitos con reductora que se utilizan en aviones, coches, helicópteros y barcos de radiocontrol. En la Figura 1 vemos algunos servos, concretamente, dos del tamaño más frecuente y un microservo.

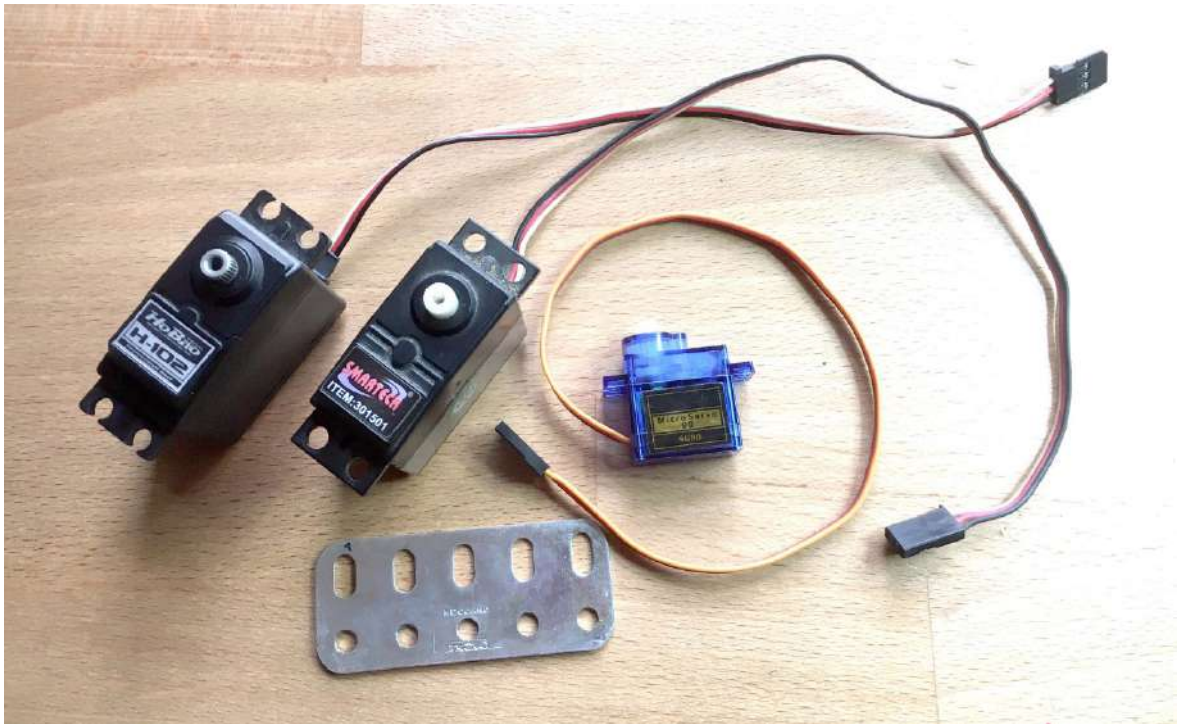



*Figura 1*

Se denominan "servos" porque funcionan en "bucle cerrado" (ver artículo sobre codificadores rotativos del boletín nº 35 de junio de 2020, página 17). Básicamente, constan de un pequeño motor acoplado a una reductora; a la salida de la reductora, su última rueda actúa sobre un pequeño potenciómetro que informa al sistema de control sobre su posición angular. El eje de salida puede girar únicamente 180 grados, pero la gran ventaja que tienen es la precisión con la que pueden posicionarse dentro de esos escasos grados de giro. Esto los hace muy adecuados para controlar los alerones de un avión, la dirección de un coche o el timón de un barco y por eso se utilizan mucho en radiocontrol, pero también se están utilizando ahora para mover pequeños brazos robóticos. Yo utilicé un microservo en el trailer que expuse en Badajoz, para mover los limpiaparabrisas. También veremos aquí cómo modificar un servo para que pueda girar varias vueltas y, en vez de controlar con precisión su posición angular, controlaremos su velocidad de giro. En las Figuras 2, 3 y 4 vemos un servo desarmado.


Figura 2

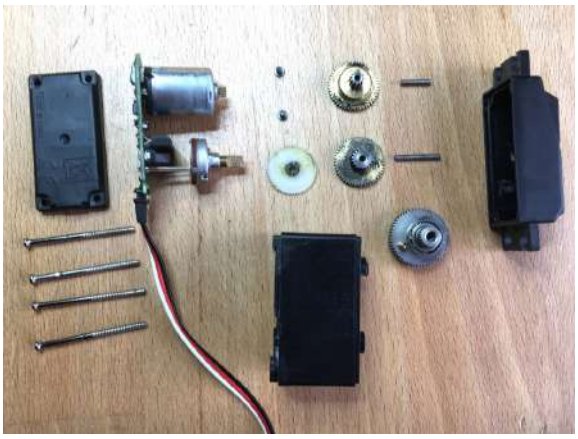


Figura 3

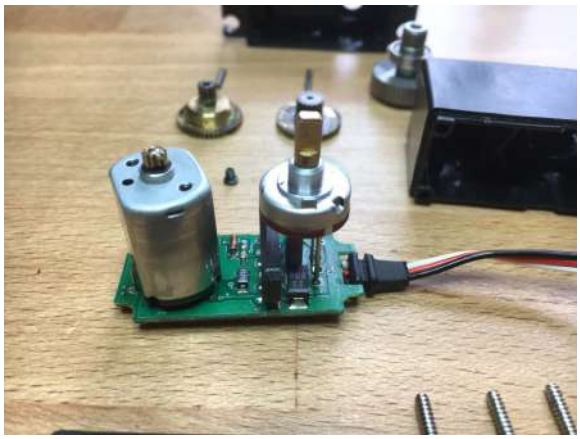


Figura 4

Figura 5

Vemos el pequeño motor y el potenciómetro montados sobre la tarjeta de control, que va en la parte de abajo. La salida de un servo es a través de un eje corto y estriado muy poco adecuado para usarlo en Meccano, pero sobre ese eje se pueden acoplar multitud de cigüeñas y ruedas que son fáciles de unir a una cigüeña o una rueda con buje de Meccano. En la Figura 5, vemos algunos de estos accesorios.

Existen muchos tipos de servos, más grandes, más pequeños, con engranajes metálicos o de plástico, de alta potencia, etc. Se pueden adquirir fácilmente en tiendas de modelismo o, a través de Internet, en tiendas chinas.

## Funcionamiento de un servo

Prácticamente, todos los servos de modelismo funcionan a 5 voltios. En la Figura 2, vemos el conector que utilizan que también está normalizado. El cable negro (en algunos es marrón) se conecta al negativo de la alimentación; el cable rojo (que siempre va en el centro del conector) se conecta al positivo de la alimentación. Finalmente, por el cable blanco (en algunos puede ser naranja o amarillo) se introduce la señal de control. Este cable va siempre en un extremo del conector y lo habitual es que este extremo tenga una pequeña pestaña para evitar que se conecte al revés.

La señal de control consiste en una cadena de impulsos de duración variable. La duración de los impulsos controlará la posición angular del servo. El estándar es entre 1000 y 2000 microsegundos, de forma que impulsos de 1000 microsegundos colocarán la cigüeña de salida del servo totalmente desplazada a la izquierda (0 grados), mientras que 2000 microsegundos la colocarán totalmente desplazada a la derecha (180 grados). Lógicamente, impulsos de 1500 microsegundos la colocarán en el centro (90 grados). Este es el estándar, luego cada fabricante hace lo que le da la real gana y nos encontraremos servos en los que el rango inferior varía entre 700 y 1000 microsegundos y el superior entre 2000 y 2300. Por eso, las radios de modelismo tienen un "trimer" junto a cada "joystick" para ajustar el movimiento. Cuando utilicemos los servos desde Arduino, también tendremos una forma de adaptarnos a las características particulares de cada servo.

Este sistema se conoce como "Telemando Digital Proporcional" o simplemente "TDP". Es curioso que se llame "digital" cuando se trata de un sistema totalmente analógico. En la actualidad, existen sistemas de telemando realmente digitales en los que lo que se envía son códigos digitales, aunque estos códigos se decodifican en el receptor y, al final, se utilizan servos de este mismo tipo.

## Servos de giro continuo

Existen en el mercado servos de giro continuo, donde la duración de los impulsos controla la velocidad y el sentido de giro. En este caso, impulsos de 1500 microsegundos corresponderán a motor parado, 1000 microsegundos a máxima velocidad en sentido antihorario y 2000 microsegundos a máxima velocidad en sentido horario. Lamentablemente, estos servos son difíciles de encontrar y con frecuencia, más caros (se venden menos), así que vamos a ver cómo podemos modificar un servo normal para convertirlo en uno de giro continuo.

Lo primero que hay que tener en cuenta es que la modificación que hagamos será irreversible o difícilmente reversible, así que deberemos contar con que, una vez que modifiquemos un servo y lo convirtamos en uno de giro continuo, probablemente se quede así para siempre.

Para convertir un servo normal en uno de giro continuo, sólo tenemos que hacer dos cosas: Eliminar el tope de giro para la rueda de salida (la última de la reductora) y desacoplar esta rueda del potenciómetro, dejando este en el centro de su recorrido. La forma de llevar a cabo estas modificaciones será diferente en cada servo de cada fabricante. Normalmente, será más fácil en un servo con engranajes de plástico, ya que bastará con cortar el resalte que tiene la última rueda para hacer tope. En otros casos, tal vez tengamos que cortar los topes de giro en la tapa del servo, porque el resalte o pestaña de la rueda sea metálico y más difícil de cortar. Para desacoplar el potenciómetro, suele ser suficiente con desencajarlo de la tapa pero, en algunos casos como el que se muestra desarmado en las fotos, será necesario desoldarlo y acortarle las patas para que quede más abajo, junto a la tarjeta. Es muy importante dejar el potenciómetro en el centro de su recorrido, ya que esto determinará el punto de parada del motor (a qué duración de impulsos se para) y la simetría de velocidades en ambos sentidos. Si se prefiere, se puede quitar el potenciómetro y sustituirlo por dos resistencias de 2K7 (2.700 Ohmios) ¼ de vatio, que soldaremos en los puntos de conexión del potenciómetro, cada una de ellas entre uno de los extremos y el central.

Con esta información y una radio TDP de radiocontrol, ya podemos telecomandar montajes de Meccano, pero todavía podemos hacer algo más: si manejamos los servos desde un Arduino, tendremos un control preciso tanto de posicionamiento como de velocidad, sin tener que utilizar puentes en "H" ni ningún otro tipo de circuitería; solamente el Arduino y los servos.

# Conexión de servos al arduino

Cómo tendremos que poner algunos esquemas eléctricos, el símbolo con el que representaremos un servo es el que vemos en la Figura 6:

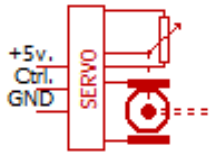
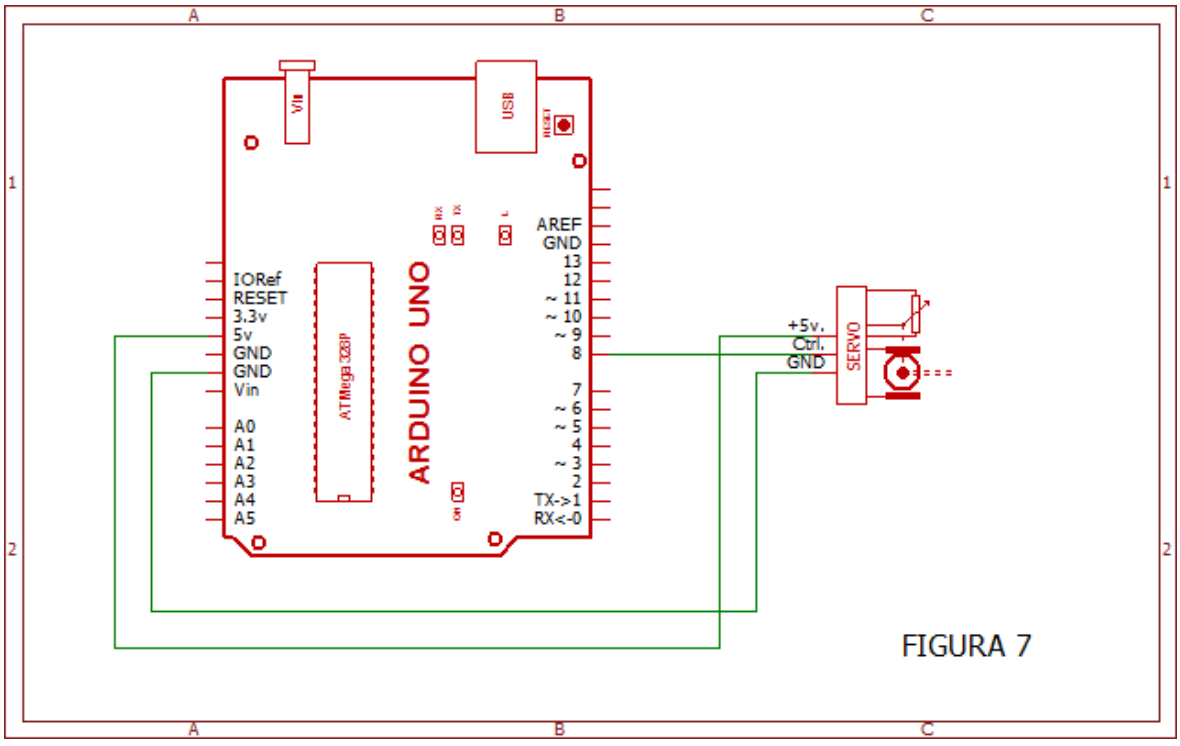



*Figura 6*

La pata marcada "+5v" corresponde al cable rojo, la marcada "Ctrl" corresponde al cable blanco, amarillo o naranja y la marcada "GND" corresponde al cable negro o marrón. Veamos ahora cómo conectamos esto a un Arduino:




Cómo vemos en la Figura 7, más fácil no puede ser: conectamos el cable rojo a la salida "5v" del Arduino, el cable negro o marrón a la salida "GND" y el cable blanco, amarillo o naranja a cualquier pin digital del Arduino. Literalmente, cualquier pin vale, pero procuraremos evitar los pines 0 y 1 ya que se utilizan, internamente, para comunicarse con el ordenador; no es que no podamos usarlos, sí podemos pero, por ejemplo, no podríamos utilizar el terminal serie para depurar el programa. El pin 13 también conviene evitarlo porque lleva conectado internamente el diodo led "L" de la placa y nos podría introducir efectos no deseados.

Podemos conectar hasta 12 servos simultáneamente a un Arduino Uno o a un Arduino Nano, y 48 servos simultáneamente a un Arduino Mega. Sin embargo, cuando conectemos

más de uno o dos servos, conviene no utilizar la salida “5v” del Arduino, sino alimentarlos aparte, ya que, en determinados casos, pueden consumir más miliamperios que los que puede suministrar esta salida.

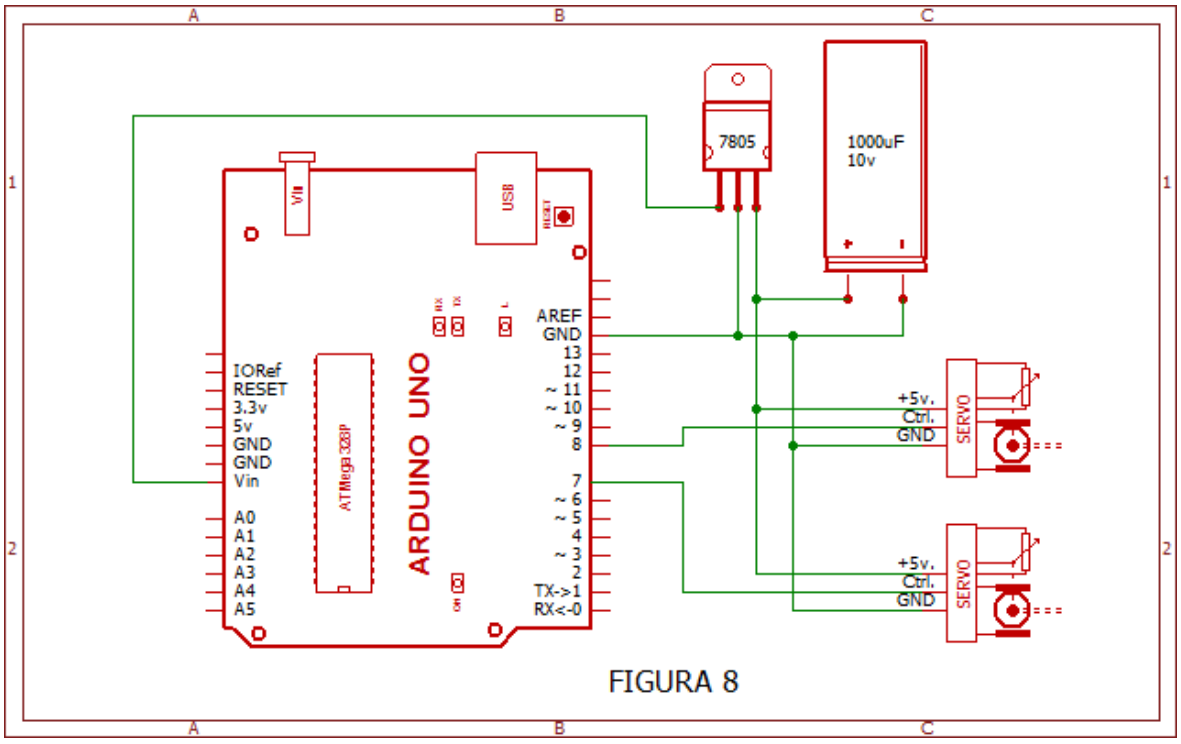



FIGURA 8

En la Figura 8, vemos una posible forma de alimentar los servos. El circuito integrado 7805 es un regulador de tensión que nos reduce el voltaje al que alimentemos el Arduino (normalmente, entre 9 y 12 voltios) a los 5 voltios que necesitan los servos. El condensador de mil microfaradios sirve para absorber los picos de corriente que puedan provocar los servos en el momento de moverse. Conviene ponerle un disipador de calor al 7805 (se atornilla a la aleta con un tornillo M3). También, si se van a poner más de 5 servos, conviene poner más circuitos 7805; aproximadamente, uno por cada 4 o 5 servos.

# Programación

Y ahora, vamos a ver cómo hacemos que esto funcione. Vamos a utilizar la biblioteca “Servo.h” que viene incluida en el propio entorno de desarrollo integrado (IDE) de Arduino. Supondremos que tenemos dos servos conectados, como se ve en la Figura 8; uno de ellos al pin 7 y otro al pin 8. No es necesario que configuremos con “pinMode()” estas conexiones como salidas, ya que la propia biblioteca se encarga de ello.

Antes de nada, es importante saber que, cuando utilicemos la biblioteca “Servo.h” en Arduino Uno y Nano, no podremos utilizar los pines 9 y 10 para generar señal PWM, es decir, no podemos hacer “analogWrite()” en estos pines. Igualmente, en Arduino Mega, no podremos utilizar los pines 11 y 12 para lo mismo, si utilizamos más de 12 servos.
Antes de “void setup()”, incluiremos la biblioteca en nuestro programa:

```
#include <Servo.h>
```

Luego, crearemos dos objetos, uno para cada uno de los servos:

```
Servo servo1;
Servo servo2;
```

Ya dentro de “void setup()”, asignamos cada servo a su pin:

```
servo1.attach(7);
servo2.attach(8);
```

En realidad, no es necesario que esta asignación se produzca dentro de “void setup()”. Podemos asignar y desasignar un servo a un pin tantas veces como queramos. Cuando un servo está asignado a un pin (se ha ejecutado un “attach()”), el motor del servo está trabajando para mantenerlo en la posición en la que se encuentre y, por tanto, consumiendo algo de corriente. Cuando no necesitemos que un determinado servo esté funcionando, es mejor desasignarlo del pin; por ejemplo:

```
servo2.detach()
```

Si desasignamos todos los servos, podríamos hacer un “analogWrite” sobre los pines 9 y 10, pero hay que tener en cuenta que la señal PWM dejaría de enviarse en el momento en que volviéramos a hacer un “attach()” y asignar cualquier servo a un pin. El motivo es que la biblioteca “Servo.h” y la generación de señal PWM por estos pines utilizan el mismo timer interno del procesador y por eso son incompatibles entre sí.

Cuando asignamos un servo a un pin, por defecto, la biblioteca considera que los valores límite de duración de impulsos para ese servo son 544 microsegundos de mínimo y 2400 microsegundos de máximo. Si conocemos los valores límite del servo que estamos asignando, podemos indicárselos al hacer el “attach()”; por ejemplo:

```
servo1.attach(7,1000,2000);
servo2.attach(8,700,2300);
```

Y ahora, ya podemos mover los servos; para ello, escribiremos en el servo correspondiente el ángulo, en grados, al que queremos que se posicione; por ejemplo, vamos a poner el “servo1” en el centro:

```
servo1.write(90);
```

Aquí hemos indicado la posición en grados (90 grados es el centro del recorrido) pero también podemos indicarlo en microsegundos:

```
servo1.writeMicroseconds(1500);
```

Finalmente, tenemos dos funciones más; una nos permite saber si un determinado servo está asignado a un pin:

```
boolean asignado=servo2.attached();
```

Como se ve, nos devuelve un valor booleano que será “true” si está asignado y “false” si no lo está. La otra función nos permite leer la posición de un servo:

```
byte angulo=servo1.read();
```

Nos devolverá un valor entre 0 y 180 que nos indica la posición del servo en grados.
Todo lo que hemos hablado sobre posiciones y grados es igualmente válido para velocidad y sentido de giro cuando estemos usando un servo de giro continuo. En este caso, un ángulo de 90 corresponderá a motor parado; un ángulo de 0 corresponderá a velocidad máxima en sentido antihorario y un ángulo de 180 corresponderá a velocidad máxima en sentido horario.

La utilización de servos en nuestros montajes, aunque pueda encarecer un poco (son algo más caros que los motores convencionales), nos abre un montón de posibilidades al tiempo que nos simplifica enormemente la capa electrónica de nuestros proyectos. Como siempre, las preguntas al chat de Whatsapp o al correo: **jalonsovivienda@gmail.com**

# Trituradora de árboles gigantes

*José Enríquez Victoria*

En las selvas difíciles se necesitan máquinas especiales como esta trituradora de árboles gigantes.

Cualquier máquina o dispositivo que permita a los agricultores y selvicultores limpiar los matorrales o arbustos de forma rápida y económica para fines productivos fue realmente bienvenida en su día, a pesar de lo elevado de su inversión inicial siendo rentabilizada por su utilización en cooperativas. Tal máquina es la trituradora de árboles desarrollada por la Empresa Tourneau, siendo ésta una versión más pequeña de la serie G de trituradora de árboles construida para la limpieza de la jungla a gran escala.

Estas máquinas desempeñaron un papel importante en la producción mundial de alimentos; millones de acres no son productivos debido al enorme gasto de tiempo y dinero necesarios para limpiar de matorrales selvas sin valor.

Las estadísticas de la ONU revelaron que la población mundial aumentaba en ese momento a un ritmo que superaba los 125.000 habitantes por día y lo que esto significaba en términos de alimentación.

Resultaba irónico que las naciones más ricas enviaran comida y ropa a personas que en realidad vivían en un suelo mucho más rico y que podría ser muy productivo si se limpiara y drenara.

La necesidad de recursos, cada vez mayor, hizo que se desarrollaran estas trituradoras de árboles. Mediante su uso, se pudieron preparar rápida y económicamente millones de acres nuevos para cultivos alimentarios, pastos, embalses y sembrar árboles nuevos, así como para comunicaciones.

La importancia es notoria en áreas como África, Sudamérica, Asia y Australia donde un solo proyecto de desmonte puede equivaler a 100.000 acres o más.
Por la magnitud de estos trabajos se construyeron las primeras trituradoras de árboles (Serie G) para hacer frente a las selvas más duras, como las de Liberia, Perú y Venezuela, donde estas máquinas desarrollaron gran actividad.

## Características técnicas

Esta gigantesca máquina (Foto nº 1) de 74 pies de largo, 22 de ancho y 19 de alto (1 pie= 0,3048 metros) usa su enorme peso mientras trabaja. En la imagen apreciamos su gran tamaño mediante la comparación con una persona.

Esta gran máquina requiere un solo hombre para operar, subido en una cabina muy por encima de los rodillos, utilizando controles sencillos de tipo pulsador, para avanzar, girar y retroceder.

Cuando se trabaja en un terreno accidentado con madera pesada o en terrenos arenosos, los grandes motores de engranajes eléctricos dan a la Trituradora de Arboles una excelente prestación.

Este monstruo de 150 toneladas de peso con sus enormes rodillos de 20 pies, obviamente requiere una enorme potencia, esto se consigue con un motor diésel de 600 h.p. para que accione un generador eléctrico que alimenta a los motores previstos para generar movimiento en las zonas necesarias.

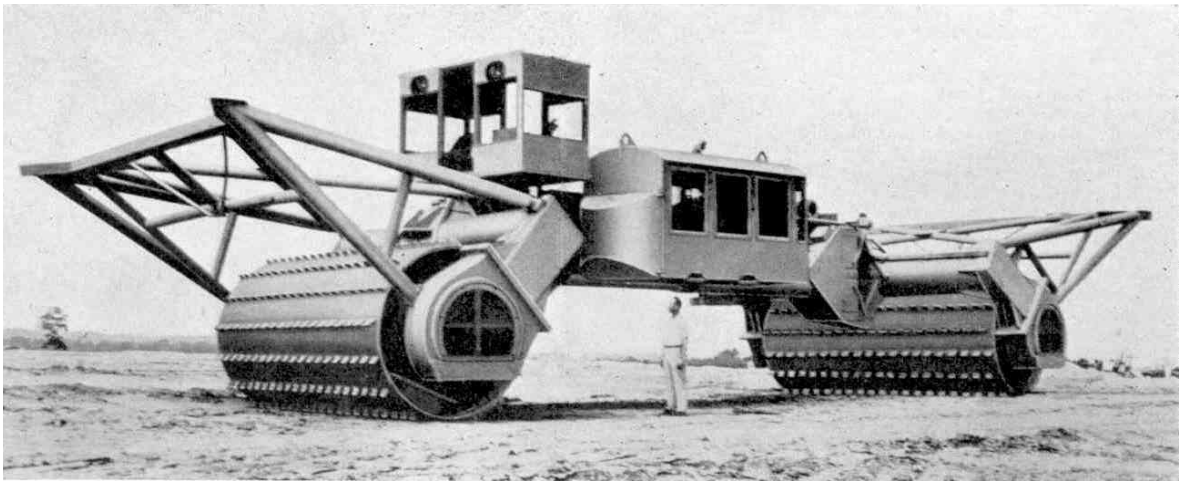


*Foto nº 1. Trituradora de árboles gigantes.*

Pasemos ahora a contar el modelo de Meccano.

Los movimientos de esta máquina en un principio son bastantes sencillos. Los dos grandes rodillos tienen que moverse hacia adelante y hacia atrás; el trasero es más complejo, puesto que lleva la dirección de la máquina.
En la foto nº 2 se tiene una idea del conjunto de esta máquina trituradora de árboles.

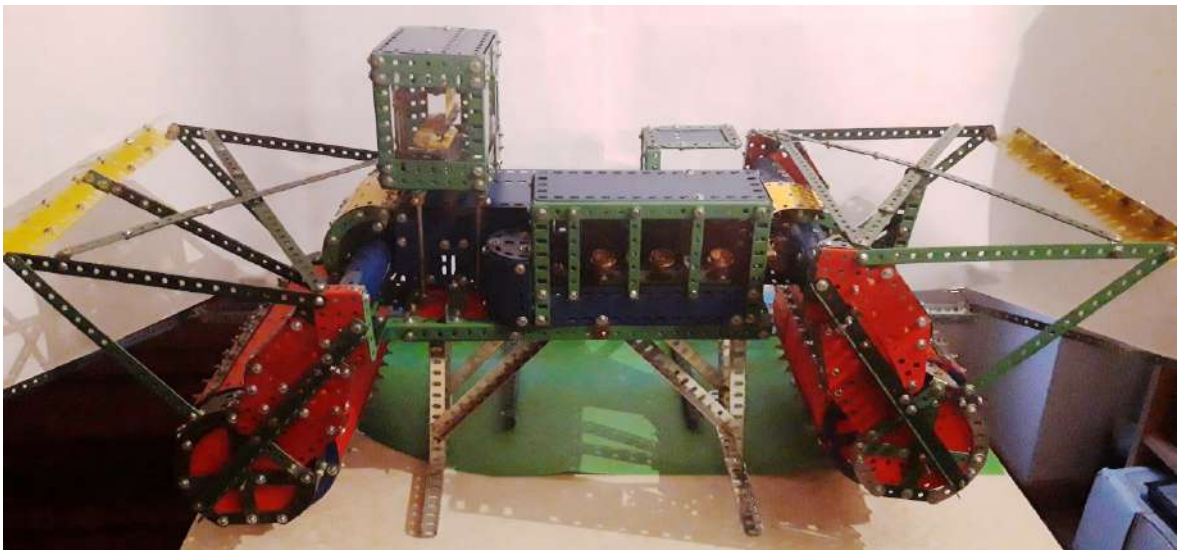


*Foto nº 2. Vista general*

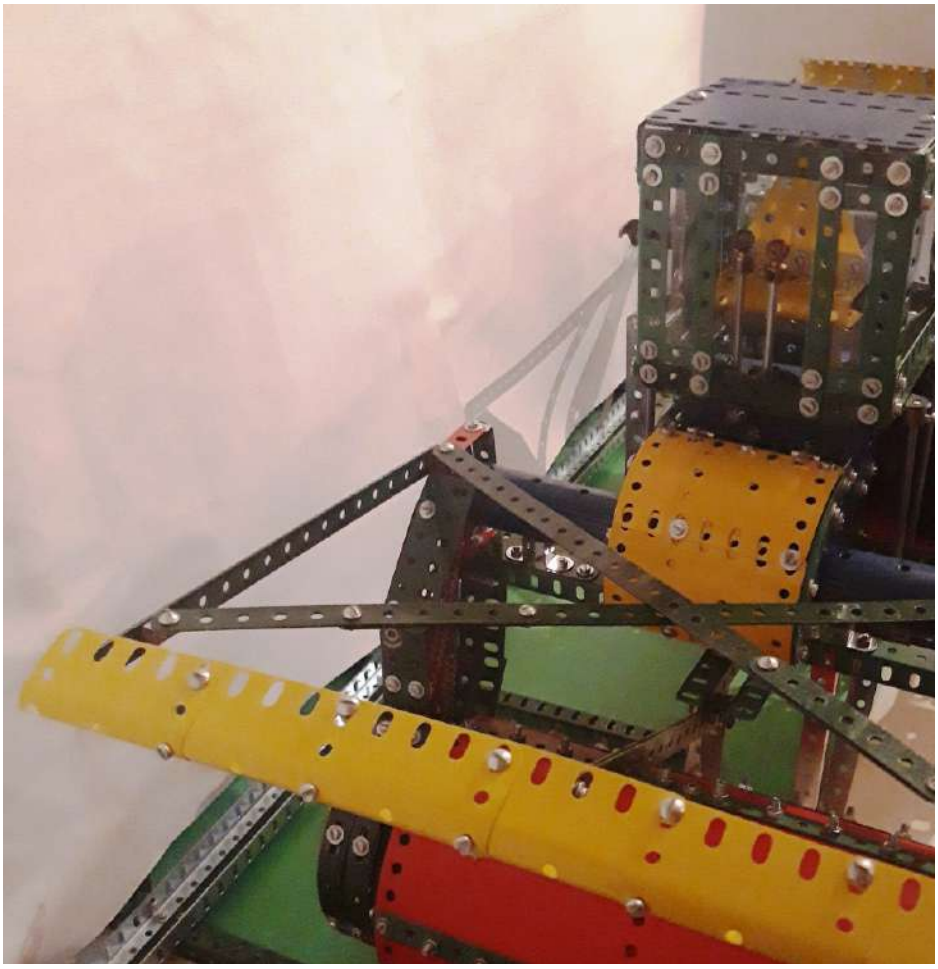


*Foto nº 3. Tambor delantero y Cabina*

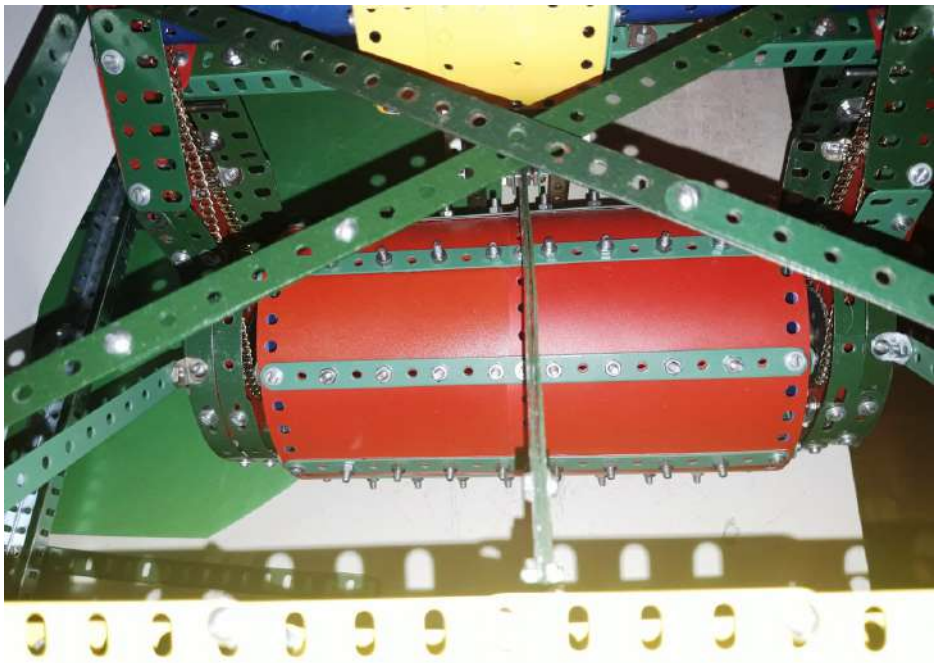


*Foto nº 4. Detalle del tambor*

El tambor delantero que se aprecia en la foto nº 4 lleva una plancha curvada, que se supone que serviría para tronchar los árboles y las púas para triturarlos. He figurado las púas con tornillos de 9,5mm en ambos tambores. En total he utilizado entre los dos tambores 96 tornillos. Los tambores están hechos con plástico, para quitar peso al modelo y no estropear las piezas. He realizado la cabina con una serie de mandos para accionar los tambores.

He procurado que todo el sistema de engranajes no se vea para dar mayor realismo al modelo.

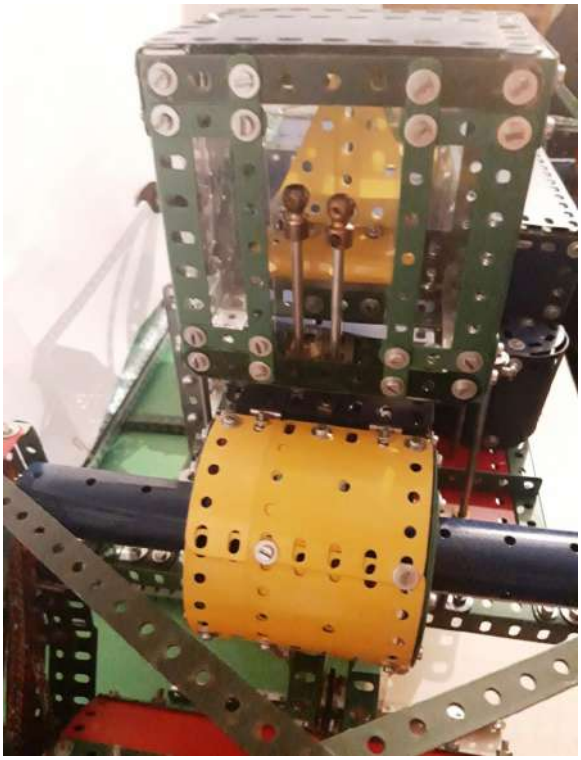


*Foto nº 5.*
*Detalle de la cabina*

*Foto nº 6.*
*Engranajes del tambor delantero*

Desde el cuadro central que después veremos, por medio de cadenas y sus correspondientes ruedas he transmitido el movimiento a los tambores a través de una rueda de cadena de 50mm. Los ejes van insertados por dentro de dos cilindros de 115mm (Pieza 163 C) y en los extremos de estos cilindros pequeñas ruedas de cadena, que a su vez engranarían a otra rueda de cadena, esta de 38mm. y a través del eje que se ve en la foto haría mover el tambor.

*Foto nº 7 Cabina de los motores*

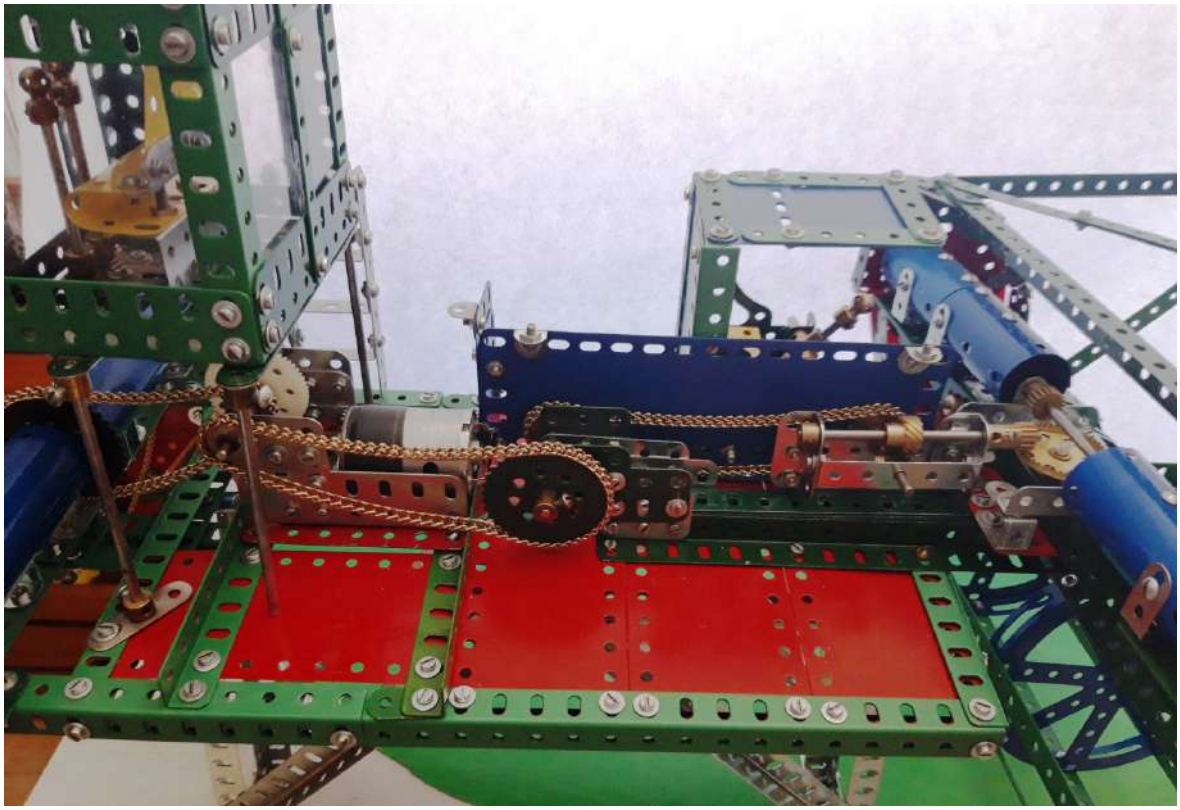


*Foto nº 8 Transmisión y tambor de dirección*

En esta cabina estarían los mandos de los motores. La transmisión a los dos tambores se ve claramente en la foto nº 8.

## Descripción del tambor de dirección

La transmisión desde el motor re realiza mediante cadenas, dos piñones helicoidales para cambiar el sentido de los ejes, dos piñones y una rueda catalina para poder girar el tambor y que este siga en movimiento, al fondo se ve una palanca en una cabina que tirando hacia adelante y hacia detrás empuja el tambor. En la foto nº 9 se ve el tambor sin las planchas que lo recubren.

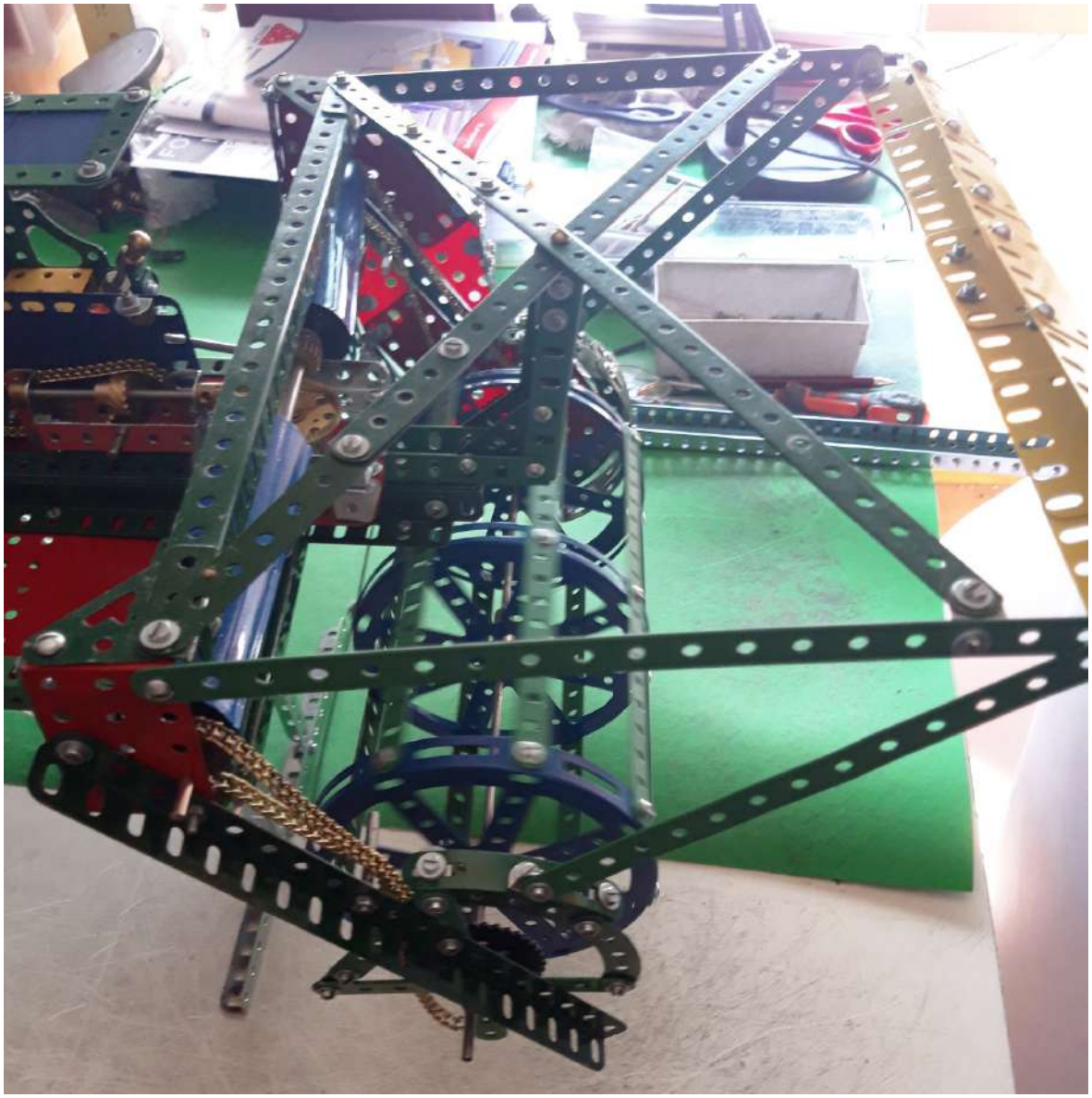
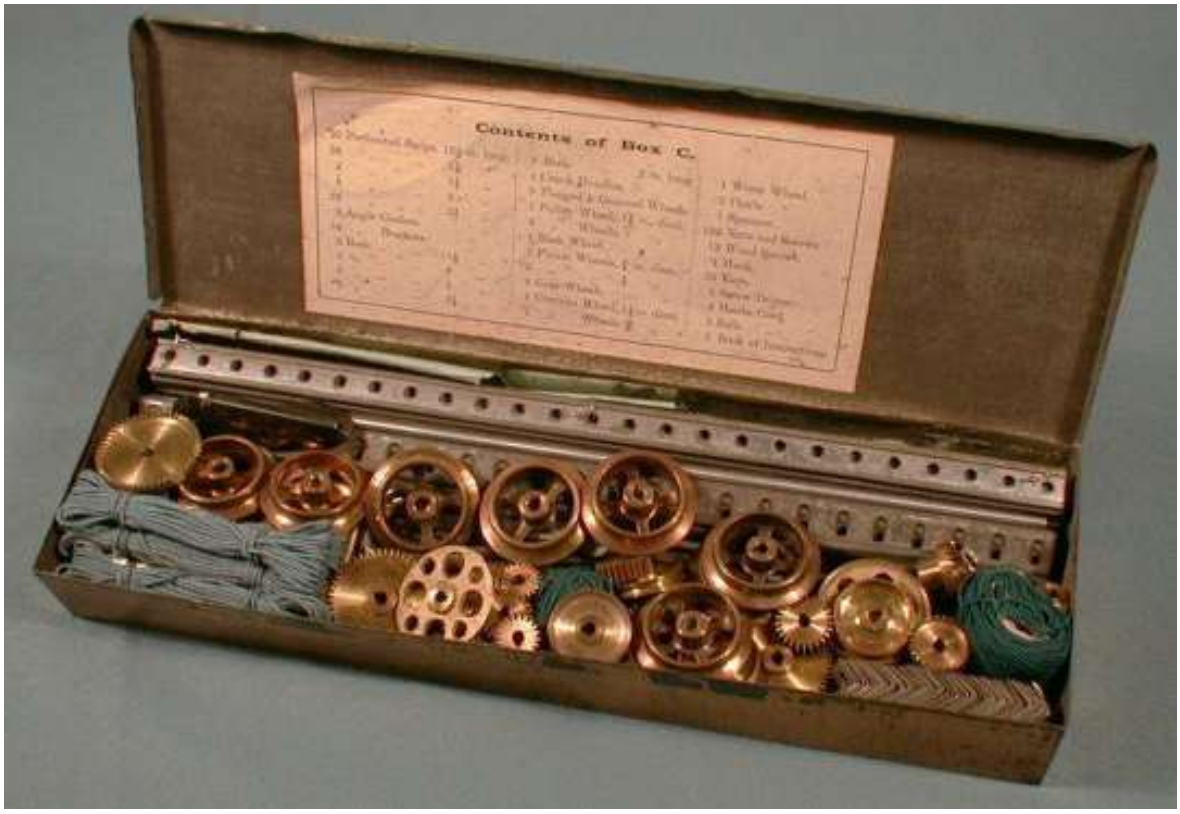


*Foto nº 9 Tambor sin recubrimiento*

Sobre una tabla o superficie dura las púas no dejan avanzar el modelo, pero sobre serrín o tierra si funciona.
De esta forma concluyo la reseña de esta singular máquina.


*Equipo de un mecano de 1906, en caja de hojalata, cuando todavía no se utilizaba la marca MECCANO*
*Colección de Antonio Valero*


Felicitación Navideña realizada por Esteban Orozco y su esposa Pilar