Asociación Cultural Española de Aficionados a los Mecanos

ACEAM

CARRETILLA ELEVADORA. Antonio Valero Aicua

Boletín nº 35

Junio 2020

Editorial

Elaboramos este nuevo boletín durante el Estado de Alarma por la COVID-19, confinados en nuestras casas para evitar que se extienda la pandemia. Muchos habremos aprovechado este confinamiento para dedicar más tiempo al Meccano. Esperemos que todos nuestros socios y sus familias se encuentren bien.

La fábrica Meccano no ha lazado ninguna novedad interesante, solo se ha puesto a la venta una nueva caja para construir 25 modelos (solo uno cada vez), para coches de Rallye, según aparece en su página web www.meccano.com.

La imagen corresponde a la portada de manual, que se puede descargar completo en la indicada página web.

Con este boletín viene un pin con el logo de ACEAM, un obsequio para cada socio.

Antonio Valero Aicua
Presidente de ACEAM

Contenido

Página

Meccano por capas

Jesús Alonso Rodríguez

Una nueva forma de acometer la ingeniería de proyectos complejos en MECCANO

A medida que ha ido pasando el tiempo, los proyectos de Meccano se han ido complicando cada vez más. Al principio, se trataba de combinar unas cuantas tiras, quizá alguna vigueta, quizá alguna placa, en el mejor de los casos, algunas ruedas y algunas piezas especiales. Con el tiempo, los aficionados hemos querido ir más allá: empezamos colocando un motor para que nuestro proyecto "se moviera sólo"; algunos conjuntos de engranajes permitieron, posteriormente, que ese motor pudiera aplicarse a diferentes movimientos. Algo más tarde, empezamos a poner más de un motor y ya se empezó a complicar la parte eléctrica: tableros de interruptores, mandos remotos por cable, finales de carrera, etc. Con el tiempo, algunos empezamos a descubrir las posibilidades que la electrónica aportaba a nuestros proyectos y empezamos a incluir circuitos electrónicos más o menos complejos, tales como reguladores de velocidad para motores, secuenciadores de luces e incluso algún circuito lógico para automatizar el funcionamiento. Y en esto, llegó la robótica y descubrimos la versatilidad de Meccano para construir robots; pero los robots requieren cierta "inteligencia" y empezamos a aplicar las soluciones de control electrónico de otros fabricantes, como Lego, Fischertechnik, etc. O empezamos a desarrollar nuestras propias soluciones de control basadas en la plataforma Arduino.

En el momento actual, un proyecto de Meccano puede llegar a ser tan complejo como algunos proyectos industriales. El aficionado al Meccano se enfrenta, hoy en día, a proyectos que requieren desde el diseño de una estructura sólida y eficaz, a la vez que estéticamente atractiva, hasta el desarrollo de un software específico que permita que su proyecto funcione, pasando por la aplicación de motores y mecanismos adecuadamente dimensionados o por el diseño y construcción de un sistema electrónico de considerable complejidad. Tareas todas estas que, en el mundo industrial, suelen ser llevadas a cabo por equipos multidisciplinares, compuestos por varios profesionales especializados en diferentes campos tecnológicos. Así, el desarrollo de un proyecto complejo de Meccano se ha convertido en un auténtico desafío intelectual para el aficionado medio que tiene que "hacérselo todo" por su cuenta y, con frecuencia, con escasa o nula ayuda exterior. Cómo en cualquier proyecto industrial complejo, la solución pasa por dividir la totalidad del proyecto en un número de proyectos de menor

entidad y, por tanto, más fáciles de acometer que, aun estando relacionados entre sí, puedan ser acometidos por separado.

En el entorno informático, es muy frecuente separar los sistemas complejos en capas ("layers" en inglés, pronunciado: "leiers") que puedan ser analizadas, diseñadas e incluso, construidas de manera independiente, aunque sin perder de vista la interrelación que deben tener para poder integrarse en un mismo proyecto. Además de la ventaja de poder centrar la atención en una sola capa en cada momento, aparece también la ventaja de la reusabilidad, es decir, de la posibilidad de reutilizar, en un determinado proyecto, una capa que se diseñó para un proyecto anterior; incluso aunque haya que realizar algunos cambios de menor entidad. En este contexto, entendemos como "capa" una parte de un proyecto que, aunque deba tener relación con el resto, tiene una entidad por sí misma y, dados unos requisitos de interoperabilidad, puede ser desarrollada de manera independiente. Hay que indicar que una capa no tiene, necesariamente, una entidad física, sino que es, más bien, un concepto lógico. Vamos a ver un ejemplo que aclarará esta idea.

Imaginemos que empezamos a construir una maqueta de la Torre Eiffel. Básicamente, se tratará de crear una estructura, inspirada en el modelo real, que mantenga las proporciones y resulte armoniosa a la vista. Podríamos decir que esta parte es la "Capa Estructural". Podemos quedarnos ahí y nuestro proyecto tan sólo tendrá esa capa. Pero ahora imaginemos que queremos ponerle ascensores como la verdadera torre Eiffel; necesitaremos instalar motores con sus reductoras, las cabinas correspondientes que corren por sus guías, los contrapesos, los cables de tracción que eleven las cabinas y dejen bajar a los contrapesos; quizá, unos interruptores de final de carrera para que los motores no se traben al final del recorrido; quizá algunos detectores de paso para permitir paradas en pisos intermedios. Esta sería la "Capa Electromecánica". Probablemente, nos apetezca instalarle unas luces para crear efectos de color, como en la real, y sus correspondientes circuitos de control; y, ya puestos, ¿qué tal poder manejar los ascensores desde un control remoto? ¿Y qué tal que funcionen solos en una secuencia que parezca aleatoria, para reforzar el efecto de realidad? Al final, acabamos montando varios sistemas electrónicos y algún Arduino; esta es la "Capa Electrónica". Finalmente, si hemos puesto un Arduino (o cualquier otro sistema de control), necesitaremos escribir un programa ("firmware") que lo haga funcionar y realizar las acciones que queremos; esta es la "Capa Lógica".

Así que ya hemos visto que un proyecto de Meccano puede tener cuatro capas:

- Capa Estructural.
- Capa Electromecánica.
- Capa Electrónica.
- Capa Lógica.

Por supuesto, no todos los proyectos han de tener todas las capas, pero en los que tengan más de una, cada capa puede ser acometida por separado y, lo que es más importante: las soluciones de diseño adoptadas para cada una de las capas pueden ser reutilizadas en otro proyecto. Cuando hablamos de reutilizar, no nos referimos a reutilizar las piezas, lo que implicaría desmontar el proyecto, sino a reusar las soluciones de diseño encontradas. Por ejemplo: en la capa electromecánica de un chasis

de auto, suele haber un diferencial y, a veces, una caja de cambios. Cuando hagamos otro chasis de auto, podemos ponerle un diferencial o una caja de cambios similar.

Esta idea de la reutilización de soluciones toma mucha relevancia en las capas electrónica y lógica. A fin de facilitar la reusabilidad de las soluciones que desarrollemos, tanto los circuitos electrónicos como los programas deberán estar concebidos de la forma más modular posible. Cuando diseñemos la capa electrónica de un modelo, deberemos pensar cuales son las funciones básicas que queremos que realice (controlar un motor, secuenciar una serie de luces, etc.) y hacer un diseño independiente para cada una de estas funciones; de esta forma, acabaremos teniendo un conjunto de diseños básicos que podremos reutilizar, en otros montajes, como si de piezas de Meccano se tratara. De igual manera, al escribir la capa lógica (el programa), deberemos separar cada función básica (manejar un motor bipolar, manejar un motor paso-a-paso, leer códigos de un receptor de infrarrojos, leer las pulsaciones de un teclado, etc.) en una entidad diferente; en programación, cada una de estas entidades se denomina “Función”. Por ejemplo, podemos tener una función de lectura de códigos desde un receptor de infrarrojos; cada vez que llamemos a la función, nos devolverá el código que ha leído o un cero si no ha leído ninguno. Un programa bien escrito no debe ser nunca una sucesión de comandos puestos uno detrás de otro, sino una colección de funciones que se van llamando, a medida que hacen falta, desde un bucle principal de ejecución. De esta forma, nuestros programas serán como un conjunto de piezas de Meccano ensambladas para constituir un proyecto. Muchas de estas funciones ya están escritas y se pueden descargar de Internet en forma de “bibliotecas” de funciones, (en inglés: “Libraries”, pronunciado: “laibraris”), (algunos dicen “librerías”; la traducción correcta es “bibliotecas”). Tenemos bibliotecas para generar y leer códigos de infrarrojos, manejar servos, manejar pantallas de presentación, etc. En el desarrollo de programas para Arduino, haremos un gran uso de estas bibliotecas ya creadas, pero debemos acostumbrarnos también a escribir nuestros programas como conjuntos de funciones reutilizables, que acabarán constituyendo nuestra propia biblioteca personal.

Control electrónico de motores: El puente en “H”

Jesús Alonso Rodríguez

Si queremos automatizar nuestros montajes de Meccano, lo primero que necesitamos es una forma de controlar electrónicamente los motores. Vamos a estudiar los circuitos electrónicos que se utilizan para este menester.

El control electrónico de motores consiste en controlar la puesta en marcha de un motor, su sentido de giro e, incluso, su velocidad mediante señales eléctricas, en vez de mediante interruptores o pulsadores. Estas señales eléctricas pueden provenir, por ejemplo, de un microcontrolador, como el Arduino, lo que nos permitirá implementar sistemas de telemando, automatizaciones etc. Para controlar electrónicamente un motor, se utilizan circuitos derivados de lo que se denomina un “Puente en H”, así que vamos a empezar por estudiar esta topología.

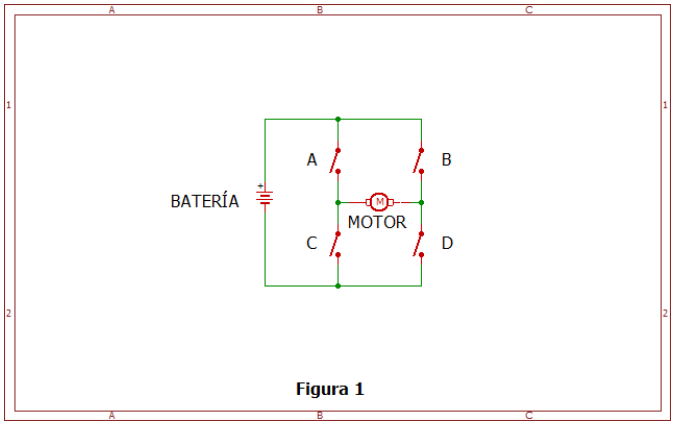


Figura 1

En la Figura 1 vemos, conceptualmente, lo que es un puente en H. Se llama así porque el motor y los cuatro interruptores forman la silueta de una letra “H”. Si cerramos el interruptor A y el D, el motor girará en un sentido mientras que, si cerramos el interruptor B y el C, girará en sentido contrario. El inconveniente de este circuito es que, si cerramos los interruptores A y C o los B y D, lo que tendremos será un montón de chispas y, probablemente, una batería de menos; es decir, un cortocircuito. Para evitarlo, podemos cambiar un poco el diseño:

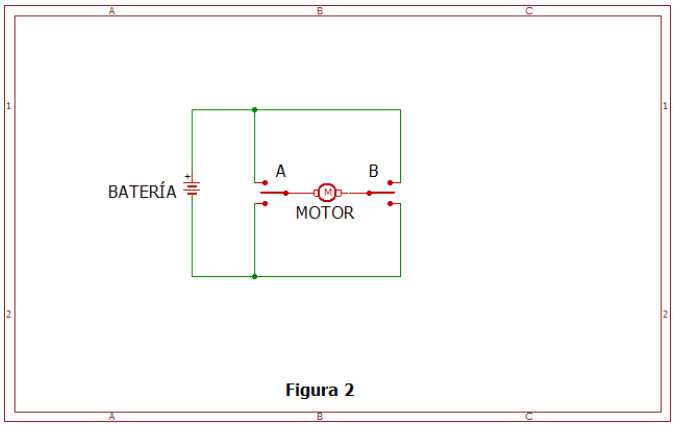


Figura 2

En la Figura 2, hemos cambiado los interruptores por conmutadores y ahora ya podemos hacer girar el motor en cualquier sentido sin arriesgarnos a provocar un cortocircuito. De hecho, ni siquiera es necesario que los conmutadores tengan una posición de "cero central" ya que, si ambos están arriba o ambos están abajo, el motor no recibirá corriente y, por tanto, no girará.
A estas alturas, el lector estará pensando que para este viaje no hacían falta alforjas, ¿dónde está la electrónica? Seguimos usando interruptores y conmutadores. Pues bien, vamos a volver al esquema de la Figura 1, pero sustituimos los interruptores por transistores:

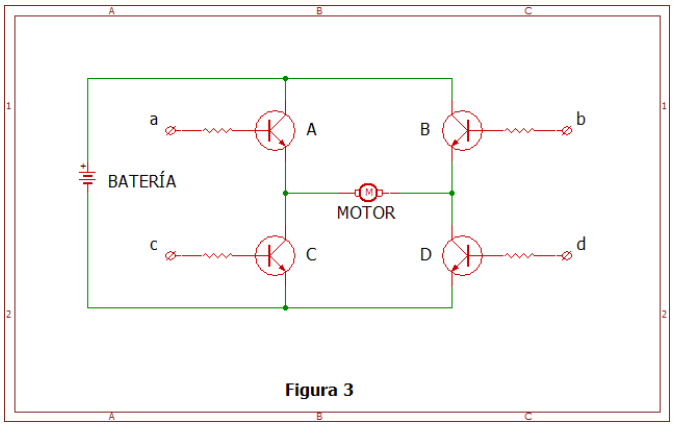


Figura 3

Un transistor puede trabajar en "Conmutación" y comportarse como un interruptor. En cada uno de los representados en la Figura 3, la pata de arriba se llama "Colector", la de abajo, la que tiene la flechita, se llama "Emisor" y la de en medio, la que está conectada a una resistencia, se llama "Base". Estos transistores conducen entre Colector y Emisor, si aplicamos una tensión entre Base y Emisor. Lo bueno es que la corriente (miliamperios) que circule entre Base y Emisor puede ser mucho más pequeña que la que circulará entre Colector y Emisor; por eso ponemos una resistencia en la Base, para limitar la corriente.

Así que ya tenemos un primer sistema de control electrónico. Si aplicamos una tensión positiva simultáneamente en "a" y en "d", el motor girará en un sentido y si la aplicamos en "b" y en "c", girará en sentido contrario. Aquí tenemos que hacer una observación importante: un motor es una carga inductiva, así que genera corrientes de auto-inducción que pueden llegar a ser de muchos voltios; por otro lado, un transistor es un componente delicado y estas corrientes podrían destruirlo. Para evitarlo, tenemos que poner diodos de protección:

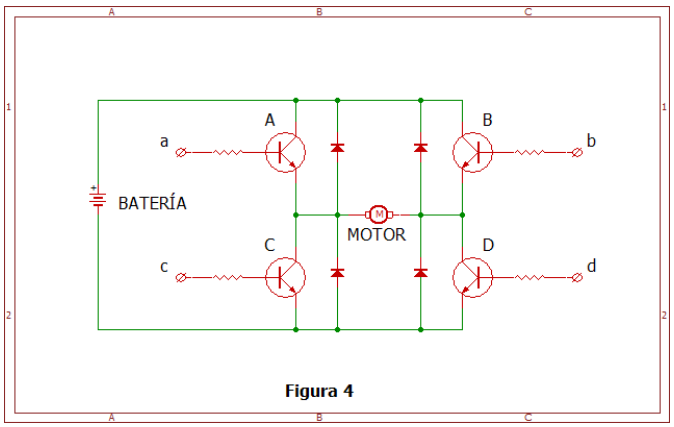


Figura 4

En la Figura 4, hemos puesto cuatro diodos de protección que absorben las corrientes de autoinducción generadas por el motor y protegen a los transistores. Ahora ya podríamos conectar esto a un Arduino y empezar a hacer magia... pero paciencia, todavía queda mucho camino por andar. Para empezar, este circuito presenta el mismo problema que el de la Figura 1: si aplicamos tensión en "a" y en "c" o en "b" y en "d", tendremos dos transistores de menos; los transistores se llevan muy mal con los cortocircuitos. De la misma forma que antes recurrimos a conmutadores, ahora vamos a sacarnos de la manga otro componente:

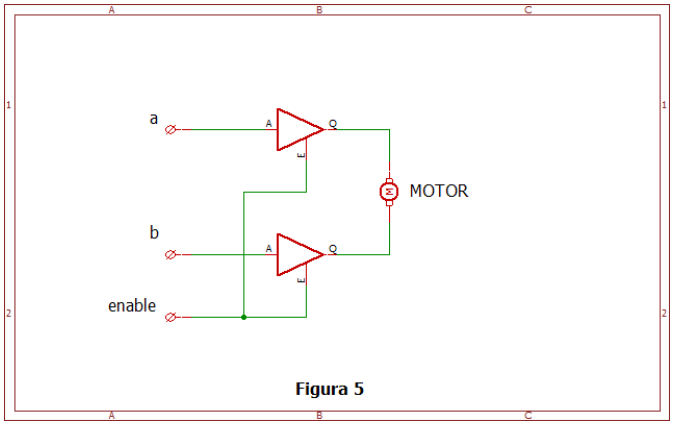


Figura 5

Y aquí hacemos una pausa para meter un poquito de teoría: cuando un transistor conduce entre Colector y Emisor, decimos que está en estado de "Saturación"; cuando no conduce, decimos que está en estado de "Corte". Así, un transistor en Saturación es equivalente a un interruptor cerrado, mientras que un transistor en Corte es equivalente a un interruptor abierto. Hay un tercer estado, intermedio a estos dos, que se denomina "Zona Activa" y que se utiliza en circuitos analógicos; aquí trabajaremos con circuitos digitales y los transistores sólo trabajarán entre Corte y Saturación. Cuando un transistor trabaja sólo entre Corte y Saturación, se dice que está trabajando en "Conmutación".

Un circuito en el que colocamos un transistor encima de otro, cómo en las Figuras 3 y 4, uniendo el emisor de uno con el colector de otro y sacando la salida de ese punto, se denomina: configuración "Totem-Pole" (en referencia a los tótems de los indígenas, que representaban figuras una encima de otra). Las salidas en configuración Totem-Pole son muy frecuentes en circuitos electrónicos, ya que permiten tanto entregar corriente desde positivo, como drenarla hacia negativo; es decir, funcionan como un conmutador. Por supuesto, cuando un circuito electrónico se diseña con salida Totem-Pole, se pone especial cuidado en que ambos transistores nunca puedan estar en Saturación al mismo tiempo.

Si ponemos, a ambos lados del motor, dos circuitos con salida Totem-Pole, lo que tendremos es un puente en H sin riesgo de cortocircuitos. Esto es lo que se suele hacer en la vida real: rara vez se utilizan puentes en H con transistores; en su lugar, se utilizan circuitos con salida Totem-Pole para controlar los motores. Los triangulitos que hemos representado en la Figura 5 son dos "Amplificadores de ganancia infinita con salida Totem-Pole", pero como esto suena muy largo, solemos llamarlos, simplemente: "Drivers" (pronunciado: "draibers").

Un Driver es un circuito que tiene, como mínimo, una entrada, que marcamos como "A", y una salida, que marcamos como "Q". Cuando la entrada A está conectada a positivo (recibe una tensión positiva), la salida Q estará también conectada a positivo y entregará corriente; cuando la entrada A no esté conectada a positivo, la salida Q estará conectada a negativo y, por tanto, drenará corriente. Lo bueno es que la corriente necesaria para la entrada puede ser de 1 miliamperio o menos, mientras que la salida puede entregar o drenar corrientes de cientos o de miles de miliamperios. Opcionalmente, puede haber una segunda entrada, que denominamos "Enable" (pronunciado: "ineibol") que significa: "habilitación" que, cuando está a positivo hace que el circuito funcione y cuando no, hace que no funcione. Luego veremos para qué podemos utilizar esta segunda entrada; si no la vamos a utilizar, simplemente, la conectamos a positivo.

Igual que podemos ir a una tienda de componentes electrónicos y comprar transistores, también podemos comprar circuitos integrados que contengan Drivers. No nos importa cómo están hechos por dentro los Drivers, eso es cosa del fabricante del circuito; para nosotros, son "cajas negras"; simplemente, sabemos cómo funcionan y los utilizamos. Algunos de ellos llevan, incluso, los diodos de protección ya integrados.

Un circuito integrado tiene un código que representa su funcionalidad y otro que representa cómo está encapsulado; estos códigos suelen ser los mismos para todos los fabricantes, así que, si vamos a una tienda de electrónica y pedimos un "L293D" en cápsula "DIL", nos dará igual si el circuito lo ha fabricado Fairchild, Texas Instruments o cualquier otro fabricante, el circuito será el mismo y nos servirá para lo mismo. Vamos a ver cómo podemos controlar un motor a partir de un puente en H realizado con Drivers:

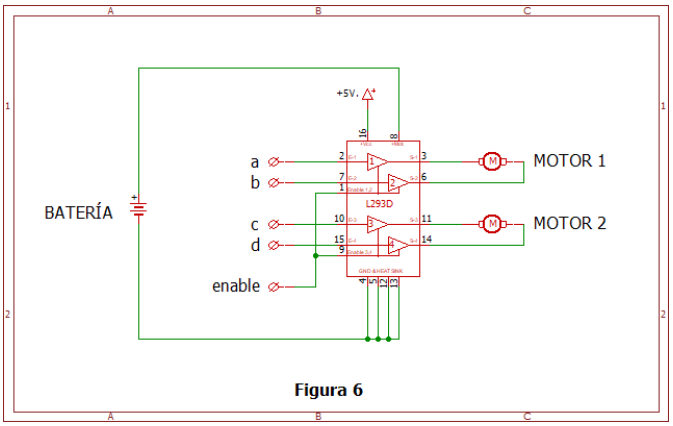


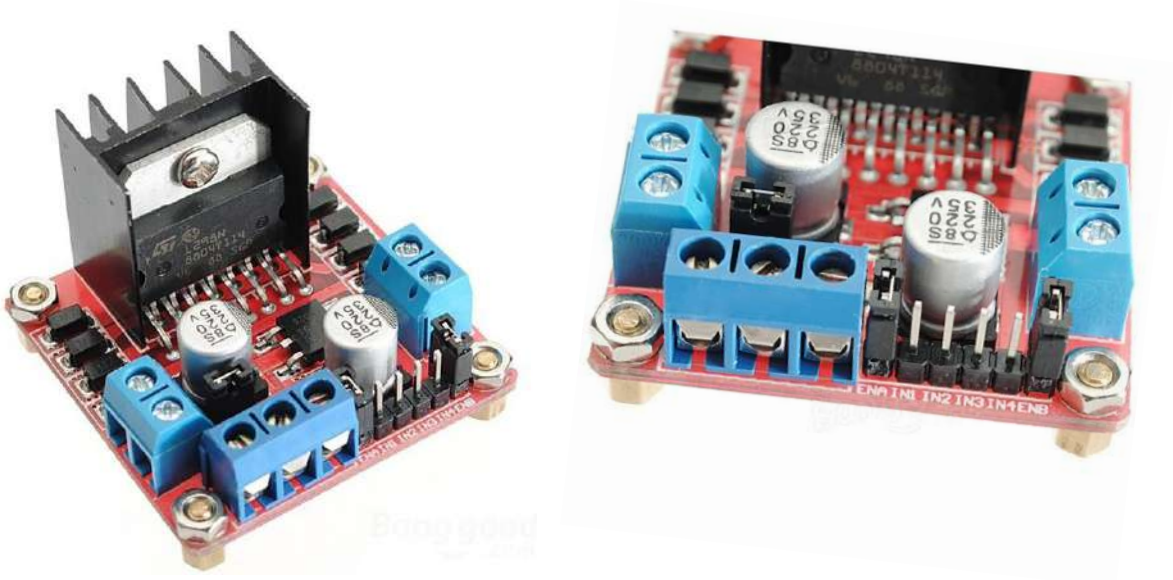
Figura 6

En la Figura 6, hemos utilizado un L293D. Como este circuito tiene cuatro Drivers, podemos usarlo para controlar dos motores. Es el mismo circuito que utilicé en el seguidor de línea que se publicó en el boletín número 27, de junio de 2016. Para no extenderme mucho, si alguien quiere ver qué aspecto tiene y cómo se conecta, recomiendo releer ese artículo. Este circuito puede manejar hasta 600 miliamperios en las salidas y lleva integrados los diodos de protección. El "L293D" es un circuito que integra cuatro Drivers, aunque los fabricantes lo suelen llamar: "Doble Puente en H"; ahora ya sabemos por qué. Ojo, también existe el "L293B" que es lo mismo, pero sin diodos de protección; si utilizamos este, los 8 diodos de protección tendremos que añadirlos nosotros fuera del circuito.

A primera vista, llama la atención que el circuito se alimenta con dos voltajes distintos: por un lado, en la pata 16 hay que meter +5 voltios con respecto a masa y por otro, en la pata 8 metemos la tensión positiva con la que vayamos a alimentar los motores; esta puede ser cualquier tensión entre 6 y 36 voltios. La mayor parte de los circuitos de conmutación, incluido el procesador del Arduino, se alimentan a +5 voltios con respecto a masa; sin embargo, nuestros motores suelen funcionar a voltajes superiores, digamos unos 12 voltios, así que esta dualidad en el voltaje de alimentación nos va a venir muy bien. Si estamos utilizando un Arduino, podremos alimentarlo a los 12 voltios de los motores y obtener los +5 voltios de una salida que entrega esta tensión, ya que el Arduino lleva, internamente, un reductor de tensión para poder alimentar el procesador y otros circuitos que también van a 5 voltios. Por supuesto, las señales de control que meteremos en las entradas “a”, “b”, “c”, “d” y “enable” serán de +5 voltios con respecto a masa, lo que nos viene muy bien porque son, precisamente, las señales que nos envían las salidas del Arduino.

¿Y qué pasa si tenemos que mover motores que consumen más de 600 miliamperios? Pues, como cabría esperar, existe un integrado, de código: “L298”, que puede manejar hasta 2 amperios (2.000 miliamperios) en cada una de sus salidas. El problema es que este integrado no se vende en cápsula DIL, sino en cápsulas “Multiwatt15” o “PowerSO20” que no se pueden enchufar en placas Breadboard. Además, hay que ponerle un radiador y no incluye los diodos de protección.

Afortunadamente, existe un módulo ya montado que incluye un L298, un reductor de voltaje a 5 voltios, los diodos de protección y clemas de conexión para los cables de alimentación y los motores. Además, y por si fuera poco, los tornillos para fijarlo, aunque son M3, están espaciados 1,5 pulgadas, así que parece que lo han pensado específicamente para el Meccano:



Este módulo se puede conseguir, por Internet, en tiendas chinas; por ejemplo: https://geek.wish.com

Y lo increíble es que, con gastos de envío y todo, sale a tres Euros.

El esquema eléctrico de este módulo es el siguiente:

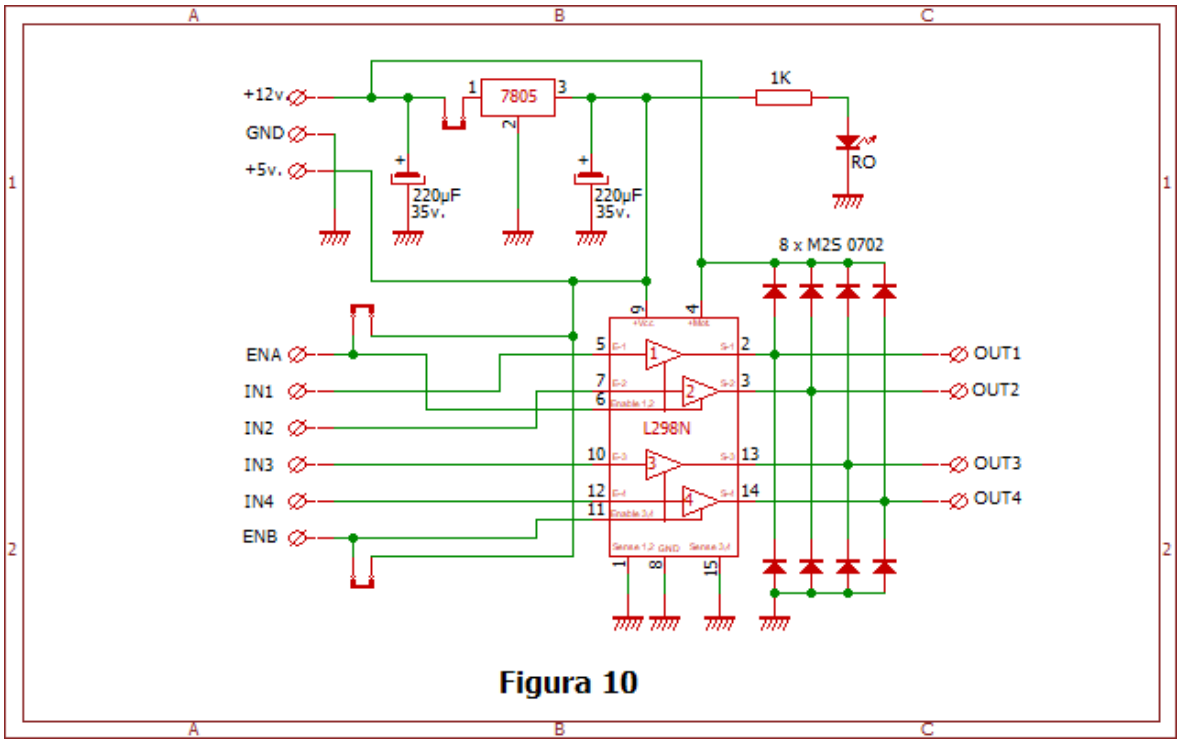


Figura 10

En la Figura 10, vemos que el módulo incluye un reductor de tensión (el integrado 7805), los diodos de protección y hasta un LED que nos indica que está funcionando. Junto al 7805 hay un jumper que deberemos quitar si le vamos a suministrar los +5 voltios desde fuera; en caso contrario, podemos utilizar la conexión marcada como "+5v." para alimentar otros circuitos, hasta un máximo de 500 miliamperios. Vemos también que lleva dos clemas de dos tornillos, una a cada lado; estas son las conexiones para los motores. La clema de tres tornillos es, de izquierda a derecha: +voltaje de motores, masa y + 5 voltios. Finalmente, vemos a la derecha un grupo de cuatro pines que son las entradas "a", "b", "c; y "d" (marcadas en el módulo como: "IN1", "IN2", "IN3", e "IN4") y dos "jumpers", uno a cada lado, que son las dos entradas "enable" (maracadas en el módulo como: "ENA" y "ENB") conectadas a +5v. Si no las vamos a usar, las dejamos así; si queremos usarlas, retiramos los jumpers. Este sería el esquema de conexión de este módulo:

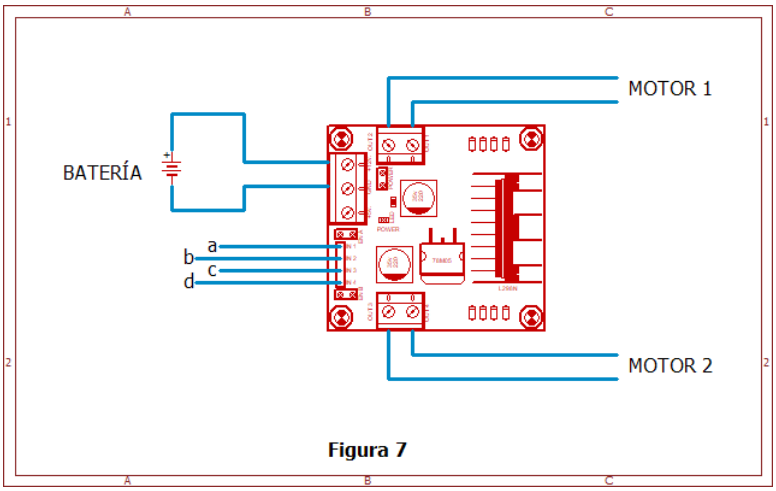


Figura 7

En la Figura 7, si ponemos a positivo (+5v.) la entrada “a” el motor 1 girará en un sentido; si ponemos a positivo la entrada “b”, girará en sentido contrario. Lo mismo ocurre con las entradas “c” y “d” para el motor 2.

O si queremos utilizar las entradas “enable”:

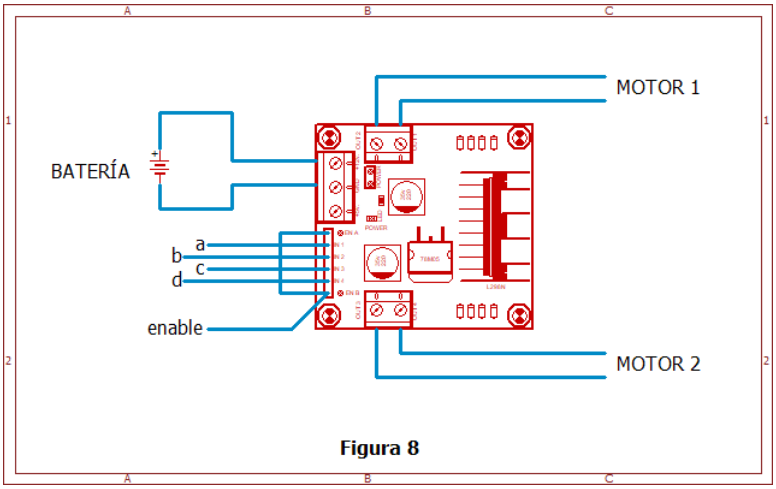


Figura 8

Pero, ¿para qué vamos a querer utilizar las entradas “enable”? Pues precisamente, a través de ellas vamos a controlar la velocidad de los motores. Para ello, inyectaremos por las entradas “enable” una señal de modulación por ancho de impulso (PWM). En el artículo que publiqué en el boletín número 26, de diciembre de 2015, explicaba cómo se controlaba la velocidad de un motor mediante modulación por ancho de impulso; ahora es el momento de irse a la pila de boletines antiguos y releer este artículo.

¿Ya de vuelta? Bueno, esta vez va a ser más sencillo porque el Arduino nos genera, directamente, la señal PWM, así que no necesitamos ningún circuito más. Ahora ya, por fin, vamos a conectar el módulo de control de motores con el Arduino:

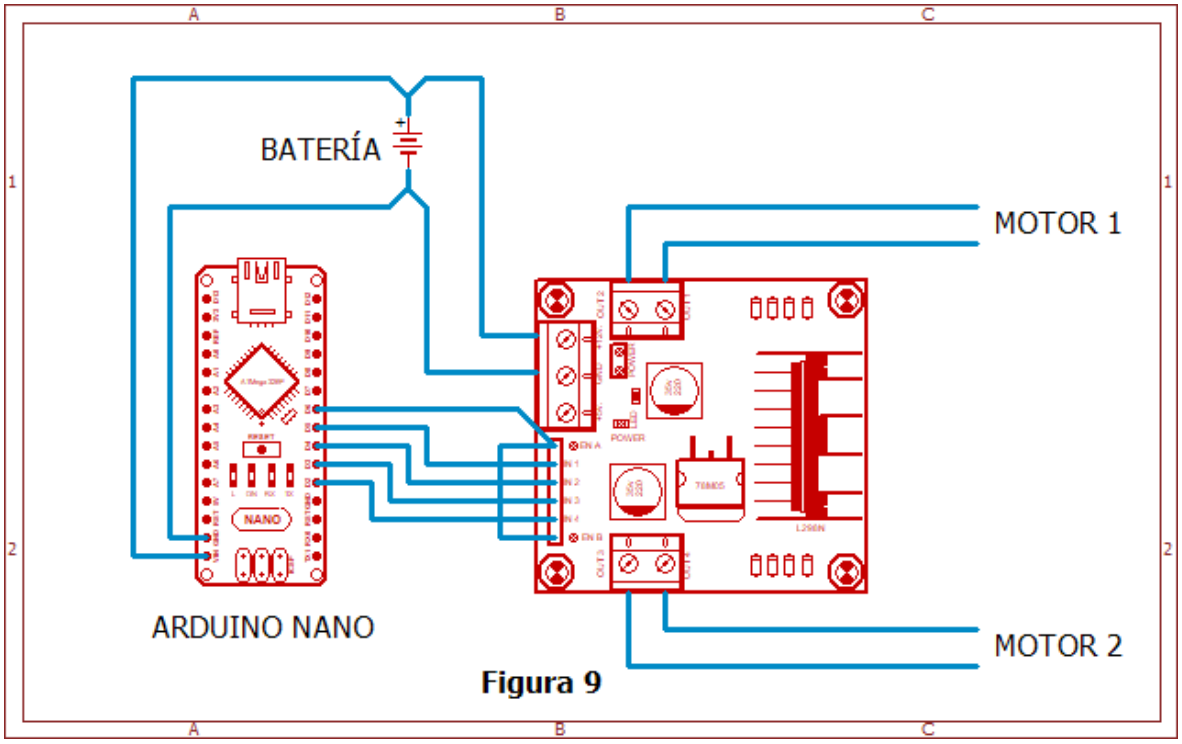


Figura 9

Lo primero que llama la atención en la Figura 9 es que he utilizado un Arduino Nano en vez de un Arduino Uno; a mí me gusta más: es más pequeño, más barato y, encima, tiene dos entradas analógicas más. Por cierto, en el artículo del número 27 metí la pata y dije que el Arduino Nano no tenía regulador de tensión; me equivoqué, sí lo tiene, mea culpa. En la misma tienda de Geek, se pueden encontrar zócalos para montar el Arduino Nano, tanto de tipo mini-Breadboard como con conexiones tipo clema, con tornillos. El mismo Arduino Nano se puede conseguir en Geek por unos cuatro Euros.

Como se ve en la Figura 9, conectamos el negativo de la batería a las dos masas, el positivo a voltaje de motores en el módulo (está marcado como +12v. pero admite cualquier tensión entre 7 y 24 voltios) y a "Vin" en el Arduino; ojo, a "Vin", no a "+5v", no la liemos. Finalmente, conectamos las entradas del módulo a salidas digitales del Arduino: "IN4" a "D2", "IN3" a "D3", "IN2" a "D4", "IN1" a "D5" y "ENA/ENB" a "D6". En este caso, hemos unido las dos entradas "enable" (marcadas "ENA" y "ENB") para controlar a la vez la velocidad de los dos motores; si quisiéramos controlarlas de forma independiente, podríamos conectar cada una a una salida PWM diferente del Arduino. Hay que tener en cuenta que no todas las salidas del Arduino pueden generar señal PWM; en el caso del Nano, sólo las salidas "D3", "D5", "D6", "D9", "D10" y "D11" pueden generar señal PWM.

Y ahora, vamos a ver cómo hacemos que esto funcione. Para empezar, declaramos una variable global, tipo byte, que almacene la velocidad y que contendrá un número entre 0 y 255. Antes de la función "void setup()" ponemos:

```
byte velocidad=128;
```

Que viene a ser la mitad. Ahora, dentro de "void setup()" configuramos las salidas y fijamos los valores iniciales:

```
void setup() {
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);
  pinMode(6,OUTPUT);
  digitalWrite(2,LOW);
  digitalWrite(3,LOW);
  digitalWrite(4,LOW);
  digitalWrite(5,LOW);
  analogWrite(6,velocidad);
}
```

A partir de aquí, y ya dentro de void loop(), cada vez que queramos que uno de los motores gire en un determinado sentido, haremos un "digitalWrite(n,HIGH)" en el pin correspondiente; por ejemplo, si queremos que el motor 1 se ponga a girar en un sentido, haremos:

```
  digitalWrite(5,HIGH);
  digitalWrite(4,LOW);
```

Si queremos que gire en sentido contrario:

```
digitalWrite(5,LOW);
digitalWrite(4,HIGH);
```

Y si queremos que se pare:

```
digitalWrite(5,LOW);
digitalWrite(4,LOW);
```

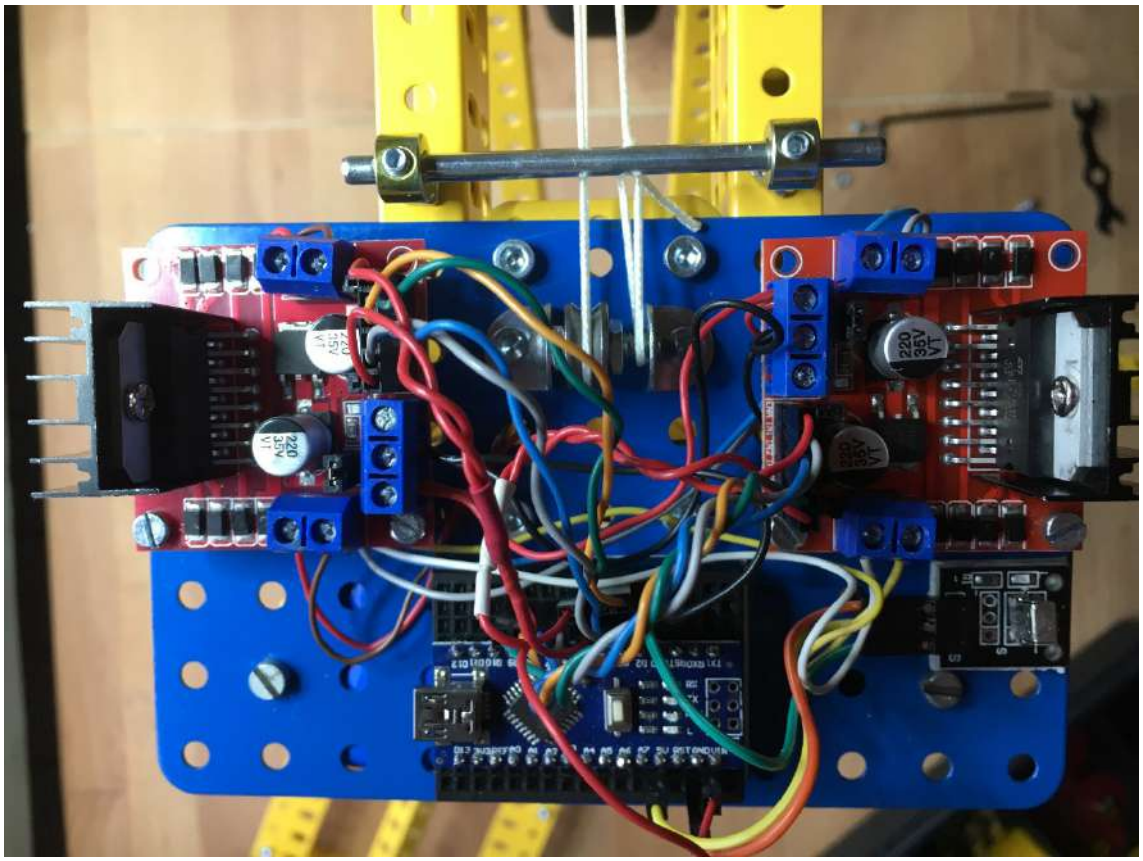
Cada vez que queramos cambiar la velocidad, cambiaremos el valor de la variable "velocidad" y lo escribiremos en la salida PWM; por ejemplo:

```
velocidad+=50;
analogWrite(6,velocidad);
```

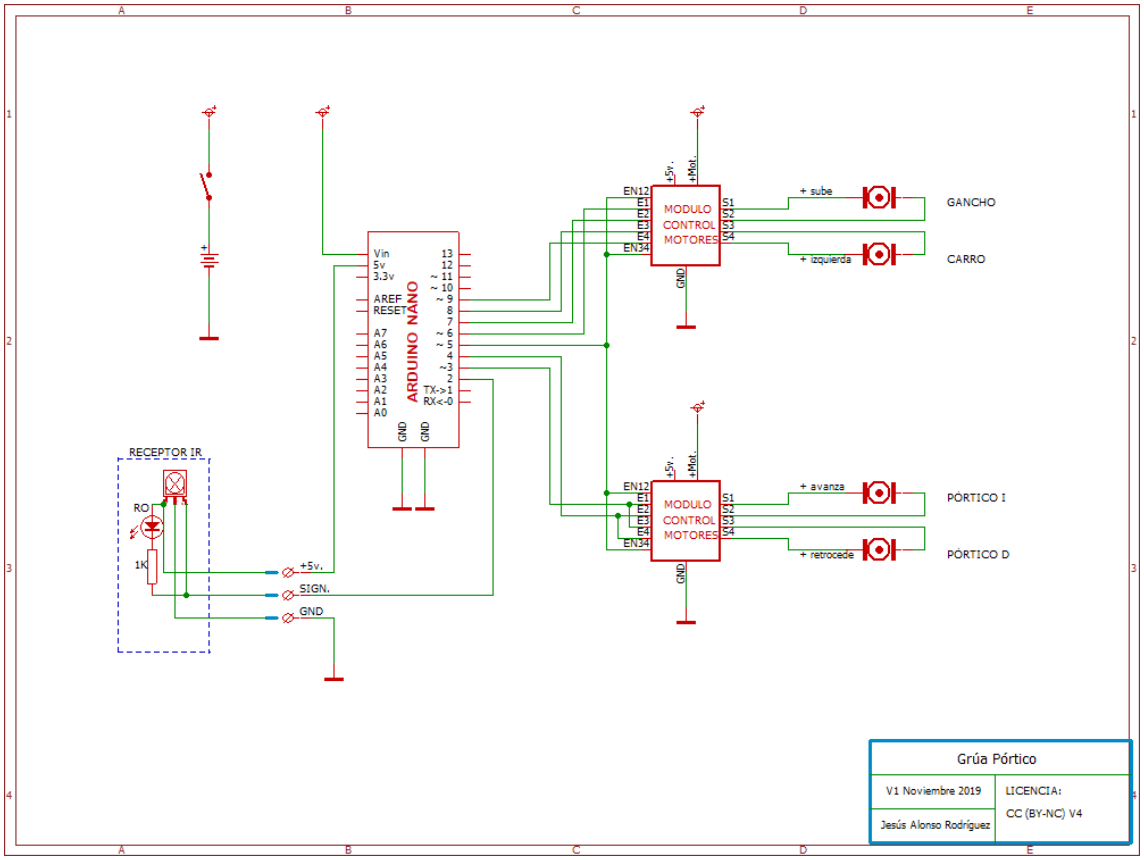
Que nos incrementará la velocidad en 50, pasando de 128 a 178.

Manejar los motores desde un Arduino nos va a permitir automatizar su funcionamiento, como en el caso del seguidor de línea del boletín número 27, o manejar los modelos desde un control remoto, bien sea vía radio o con un mando por infrarrojos. A continuación, se incluyen un par de fotografías de casos prácticos en los que se han utilizado dos módulos de control de motores manejados desde un Arduino Nano para implementar un control remoto mediante infrarrojos.

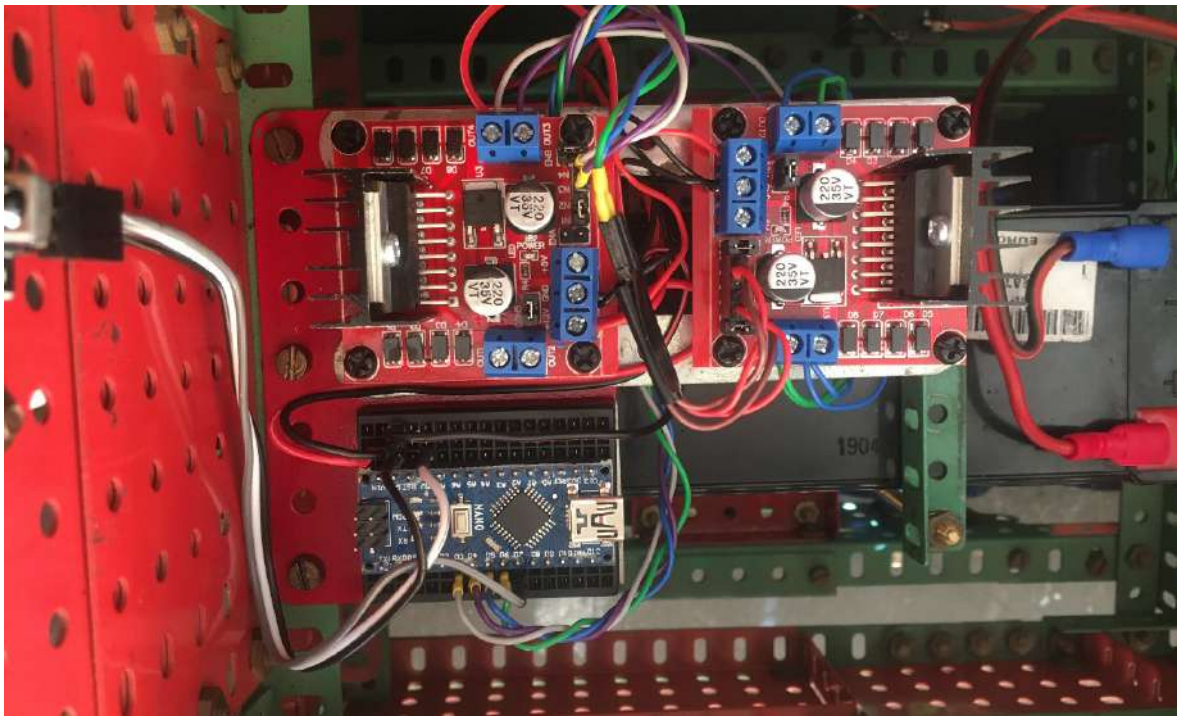
Este es un sistema para manejar cuatro motores en una grúa de pórtico:



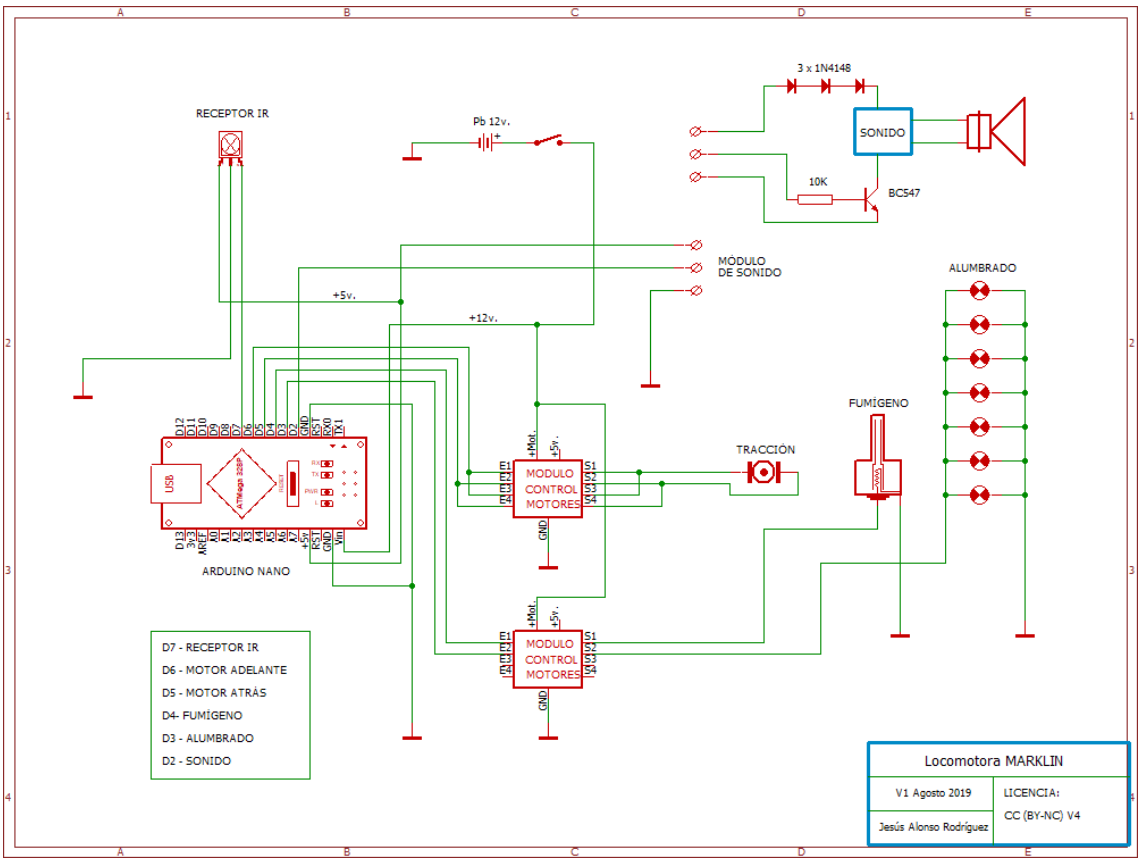
Y este es el esquema eléctrico:



En este otro caso, se trata de controlar, también mediante infrarrojos, el motor de una locomotora de vapor grande, el generador de humo, las luces y el sonido:



Y este sería el esquema eléctrico:



En este último caso, el motor que había que manejar consumía más de 2 amperios, así que se optó por montar los dos puentes en H en paralelo para sumar la corriente que los dos son capaces de manejar, pudiendo llegar hasta 4 amperios. Esto se puede hacer ya que así lo indica la hoja de características del fabricante del L298; no obstante, es peligroso porque, si no se es muy cuidadoso con el conexionado, es fácil cargarse el circuito: Es fundamental asegurarse de que se ponen en paralelo las entradas correspondientes a aquellas salidas que también se pongan en paralelo; de lo contrario, podría ocurrir que, de dos salidas que están en paralelo, una conmutara y la otra no, lo que provocaría un cortocircuito que quemaría el L298.

Se han utilizado otros dos drivers como amplificadores de corriente para el fumígeno y el alumbrado. Al fin y al cabo, los drivers no sólo sirven para montar puentes en H; en muchos casos, pueden ser útiles como simples amplificadores de corriente; hay que tener en cuenta que cada salida del Arduino sólo puede manejar 40 miliamperios y tanto el fumígeno como el alumbrado de esta locomotora consumen cerca de un amperio.

Y hasta aquí lo que tenía que explicar sobre puentes en H. En otro artículo veremos cómo podemos controlar motores paso-a-paso utilizando este mismo módulo. 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

Codificadores rotativos

Jesús Alonso Rodríguez

Hemos visto cómo un microcontrolador como Arduino puede accionar un motor para hacer girar un eje. Ahora vamos a ver unos dispositivos que sirven, exactamente, para lo contrario; es decir, para informar al microcontrolador sobre el movimiento de un eje.

Bucle abierto y bucle cerrado

Decimos que un sistema funciona en "bucle abierto", cuando no tiene realimentación, es decir, cuando no hay información transmitida desde el funcionamiento del sistema hacia la unidad de control. Por el contrario, decimos que funciona en "bucle cerrado" cuando el elemento que controla el funcionamiento recibe información sobre el estado del sistema. Veámoslo con un ejemplo: supongamos un guardagujas ferroviario que, desde una caseta de control, mueve una serie de desvíos; el guardagujas haría el papel de elemento controlador mientras que el posicionamiento de las agujas de los desvíos sería el sistema controlado. Cuando el guardagujas mueve una palanca, se supone que la aguja de un desvío cambia de posición. El accionamiento puede estar diseñado de forma tan segura que sea imposible que la palanca se mueva sin que la aguja cambie; en este caso, nos encontraríamos ante un sistema que funciona en "bucle abierto". Podemos querer hacer que nuestro sistema sea más seguro: que si el movimiento de la aguja no se produce por cualquier causa (una piedra que la atasque, por ejemplo) el guardagujas pueda darse cuenta, así que colocamos unos pequeños interruptores a ambos lados del desplazamiento de la aguja, que accionen dos lucecitas colocadas junto a la palanca de control; cada vez que el guardagujas accione una palanca, comprobará que la indicación de las luces ha cambiado y, por tanto, estará seguro de que el sistema ha respondido a su comando; en este caso, decimos que el sistema funciona en "bucle cerrado".

El funcionamiento en bucle cerrado es tan frecuente en sistemas electromecánicos que incluso se ha acuñado un término para denominar a los sistemas que se estabilizan funcionando en bucle cerrado; este término es: "servosistemas". El servofreno de un coche, la dirección asistida (también llamada servodirección) e incluso un servomotor de radiocontrol (lo que solemos llamar un "servo") son servosistemas. Algunos autores también lo denominan "funcionamiento en anillo". Probablemente, el primer servosistema de la Historia sea el famoso regulador centrífugo de una máquina de vapor.

Aparte del efecto de estabilización automática, un sistema que trabaje en bucle cerrado tiene también la ventaja de que asegura la exactitud del posicionamiento de algún elemento móvil. De nuevo, veámoslo con un ejemplo: supongamos que hemos construido un vehículo teledirigido y tenemos que mover la dirección. Podríamos utilizar un motor paso-a-paso que desplazara el volante a uno y otro lado y el

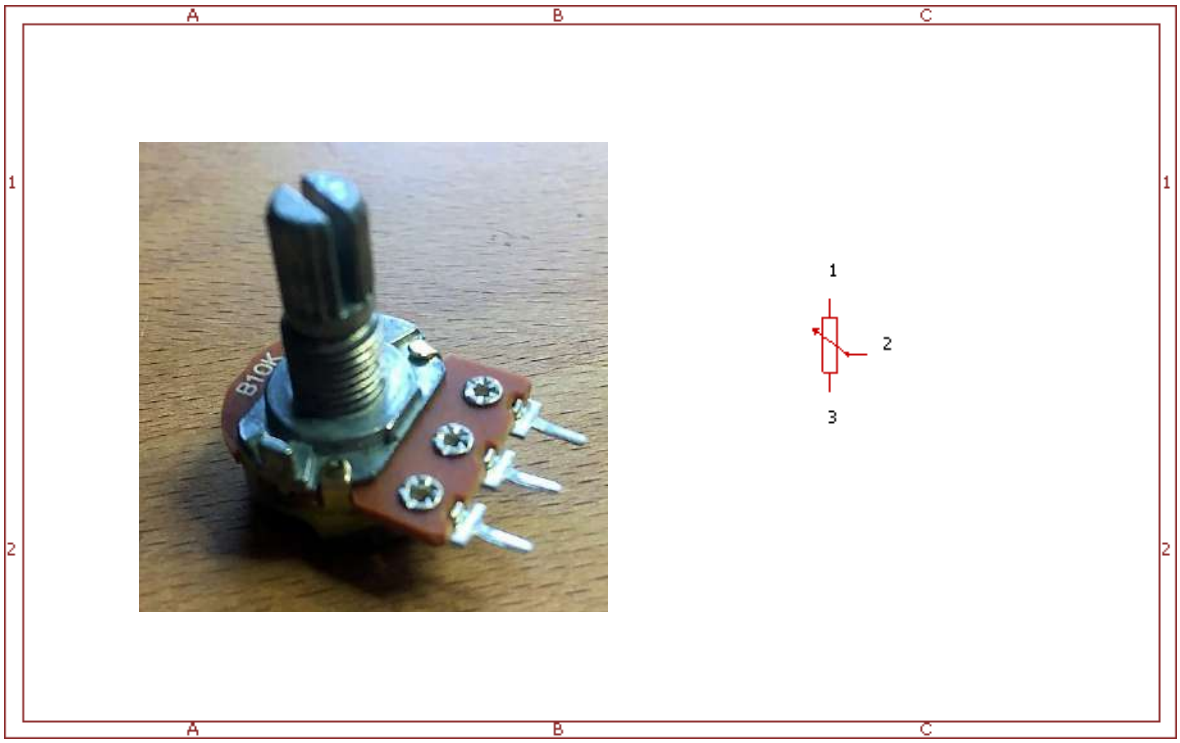
posicionamiento vendría dado por el número de pasos que se haya desplazado; si queremos volver a la posición central, simplemente le aplicamos el mismo número de pasos en sentido contrario. El problema es que un sistema así tiende a acumular error cada vez que se pierda un paso. Lo ideal es disponer de algo que nos informe (o más bien que informe a la unidad de control) de la posición de las ruedas en cada momento. Ese algo puede ser un codificador rotativo. También veremos que podemos utilizar codificadores rotativos como elementos de control y, de hecho, lo hacemos habitualmente. Cada vez que subimos o bajamos el botón de volumen de una radio o giramos un conmutador estamos actuando sobre un codificador rotativo.

Codificadores rotativos absolutos

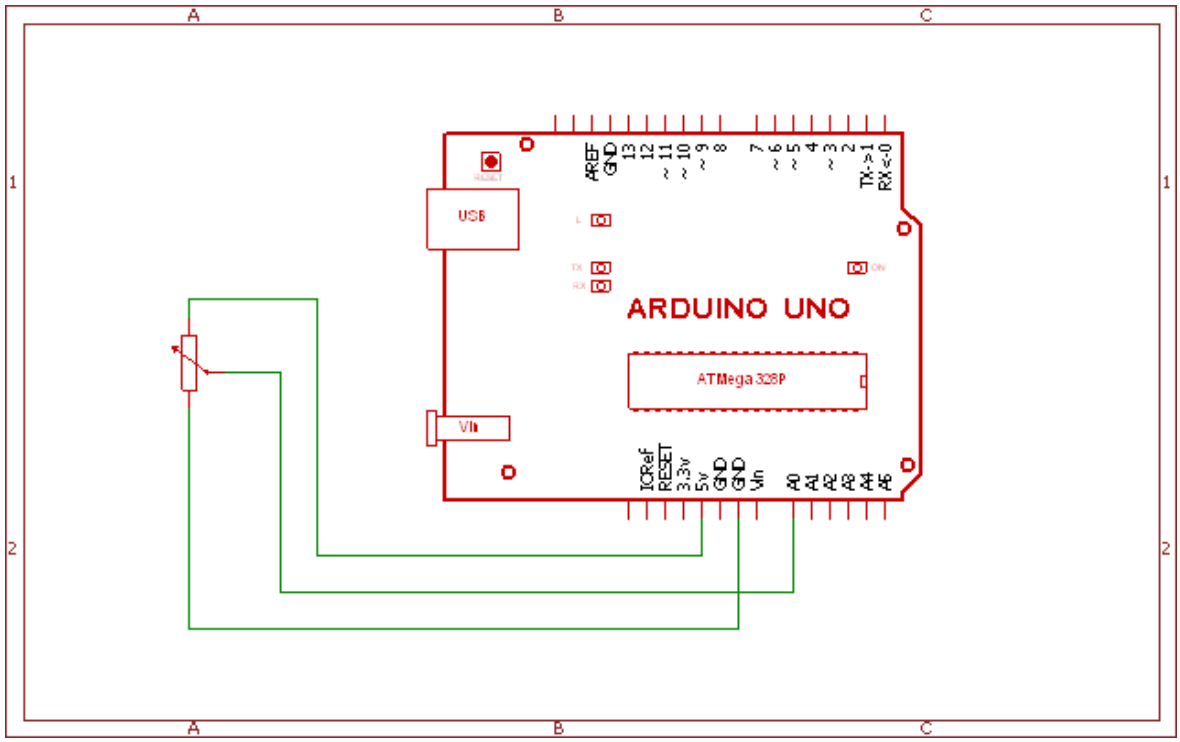
Decimos que un codificador rotativo es absoluto cuando nos informa exactamente de su posición angular, es decir, de cuantos grados está desplazado con respecto al punto de referencia; al mismo tiempo, un codificador rotativo absoluto puede ser continuo o discreto, según nos informe de los grados de desplazamiento o de una de entre varias posiciones concretas.

El codificador rotativo absoluto más conocido es un potenciómetro. Se trata de un dispositivo que tiene tres terminales; dos de ellos conectados a ambos extremos de una resistencia y el tercero a un cursor que se mueve sobre esa resistencia. Se trataría, por tanto, de un codificador rotativo absoluto continuo. La mayoría de los potenciómetros que existen en el mercado pueden girar 270 grados, pero también existen los denominados potenciómetros multivuelta que suelen poder girar 10 vueltas completas, es decir, 3.600 grados.

Para leer la posición de un potenciómetro con un Arduino, conectamos un extremo de la resistencia a +5v y el otro a masa; el cursor (terminal central) lo conectamos a una de las entradas analógicas (“A0” a “A5” en “Arduino Uno” o “A0” a “A7” en “Arduino Nano”). Como la resistencia está conectada entre +5v y masa, el cursor, al desplazarse sobre ella, recibirá un voltaje relativo a su posición; si está en el extremo conectado a masa, recibirá 0 voltios, mientras que si está en el extremo conectado a +5v recibirá 5 voltios; en posiciones intermedias recibirá voltajes intermedios. En este punto hay que comentar que, aunque la variación de resistencia con el movimiento angular en la mayoría de los potenciómetros es lineal, también existen algunos en los que esta variación es logarítmica y estos potenciómetros NO NOS VALEN como codificadores rotativos. Como referencia, un potenciómetro que se utilice para esto deberá tener un valor resistivo de entre 5.000 (5K) y 100.000 (100K) Ohmios. Menos de 5K consumiría demasiada corriente y más de 100K sería demasiado sensible al ruido. El valor ideal es de 10K (10.000 Ohmios). En la siguiente figura, vemos el aspecto que tiene un potenciómetro y su representación esquemática:



Y en esta figura, vemos cómo podríamos conectarlo a un Arduino:



Para leer este voltaje en un Arduino, utilizamos la función “analogRead(entrada)” (donde “entrada” es cualquiera de las entradas analógicas. Esta función nos devuelve un número, comprendido entre 0 y 1023, proporcional a ese voltaje. En un potenciómetro de 270 grados, cada grado de giro modificará ese número en 1023/270=3,8 (como lo que devuelve la función es un entero, digamos que 4). Si

queremos convertir la lectura de una entrada analógica en grados, podemos usar la función “map”:

```
int lectura=analogRead(A0);
int grados=map(lectura,0,1023,0,270);
```

Ojo que la función “map” es un poco puñetera y conviene aplicarla siempre sobre una variable y no sobre una expresión; por ejemplo:

```
int grados=map(analogRead(A0),0,1023,0,270);
```

Podría no funcionar correctamente en algunos casos.
Si se tratara de un potenciómetro multivuelta de 10 vueltas, podríamos hacer:

```
int lectura=analogRead(A0);
int gradosTotal=map(lectura,0,1023,0,3600);
int vueltas=gradosTotal/360;
int grados=gradosTotal%360;
```

Donde el signo “%” significa “módulo” y nos da el resto de la división.

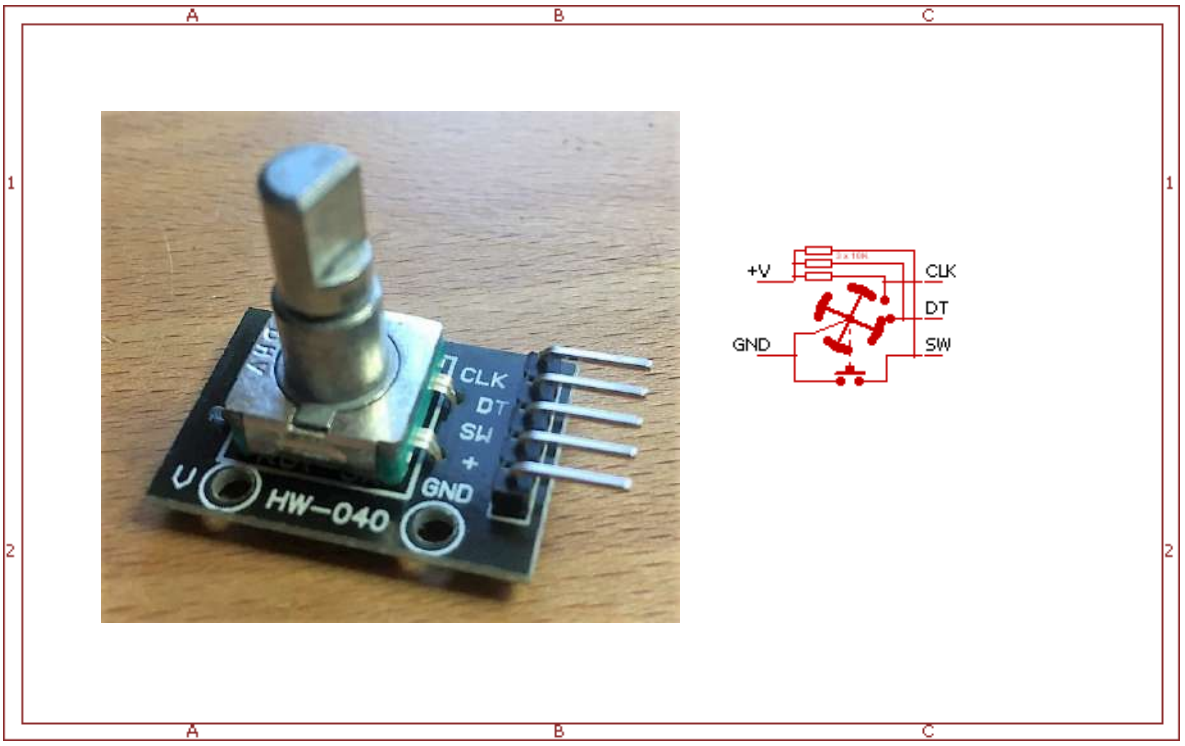
Un conmutador rotativo sería un codificador rotativo absoluto discreto; por ejemplo, el conmutador de 12 posiciones que se muestra en el artículo sobre motores paso-a-paso del boletín número 18 (diciembre de 2011) es un codificador rotativo en el que cada cambio de estado corresponde a un giro de 30 grados.

Codificadores rotativos relativos

Existe un tipo de codificadores rotativos que no nos informan de una posición absoluta, sino de que se ha producido un giro, del sentido de ese giro y, eventualmente, de su magnitud. Algunos disponen de un pequeño pulsador, axial o radial, accionable desde el mismo eje. El ejemplo quizá más conocido que utiliza uno de estos codificadores rotativos es la rueda de un ratón de ordenador. También los encontramos cada vez más en los botones de control de equipos de música y autorradios. La ventaja de estos codificadores rotativos es que no están limitados a un número determinado de grados.

Existen dos familias para este tipo de codificadores, dependiendo de la tecnología que utilicen: mecánicos y ópticos. Los mecánicos se basan en un doble conmutador en el que las dos conmutaciones están ligeramente desfasadas, mientras que los ópticos se basan en una rueda opaca con una serie de ranuras radiales (o una rueda transparente con una serie de líneas radiales) que tiene acoplados en su perímetro uno o dos fotoacopladores; los de un fotoacoplador se suelen utilizar para medir la velocidad angular de un motor, mientras que los de dos fotoacopladores informan también sobre el sentido de giro, ya que la conmutación de los dos fotoacopladores también está ligeramente desfasada.

En nuestros modelos con Arduino, utilizaremos principalmente codificadores rotativos mecánicos cuyo esquema se muestra a continuación, junto con el aspecto que suelen tener:



Para leer uno de estos codificadores rotativos, necesitaremos conectarlo a dos entradas del Arduino (a tres si también tiene pulsador axial o radial). Los codificadores rotativos que se pueden comprar en tiendas chinas suelen venir montados en una plaquita de circuito impreso y tienen cinco puntos de conexión:

“+V” o “+”: Conexión a +5 voltios.

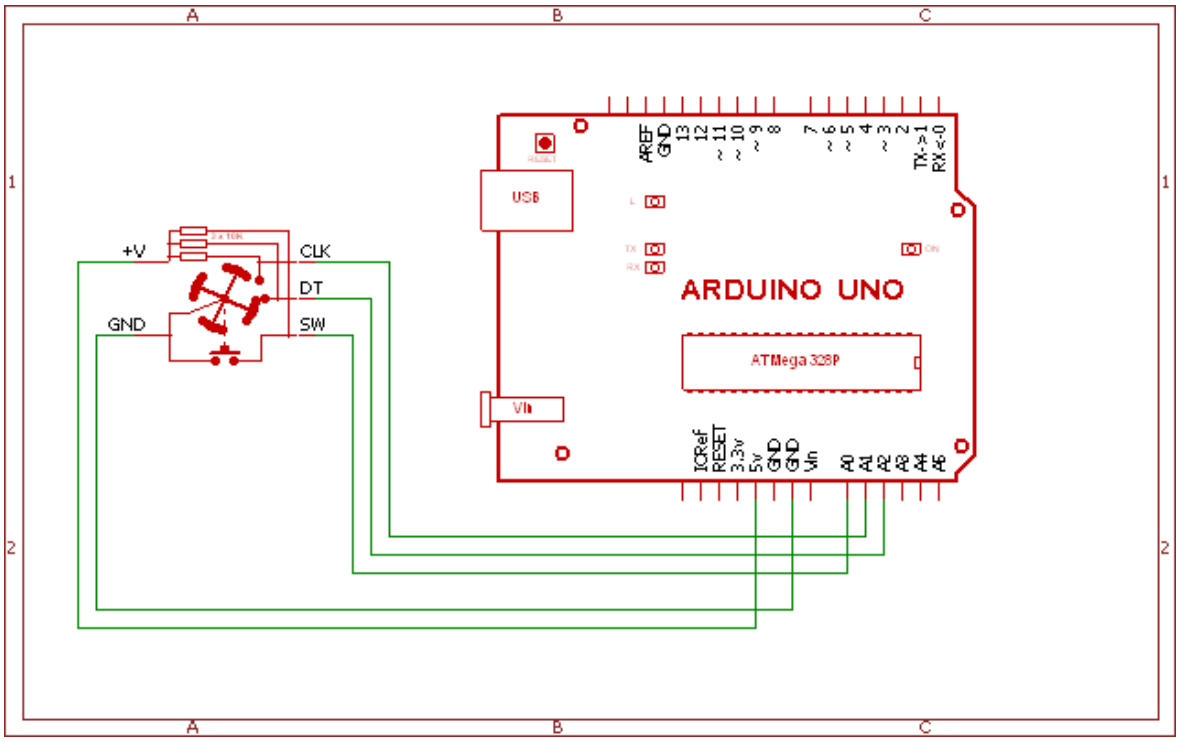
“GND”: Conexión a masa.

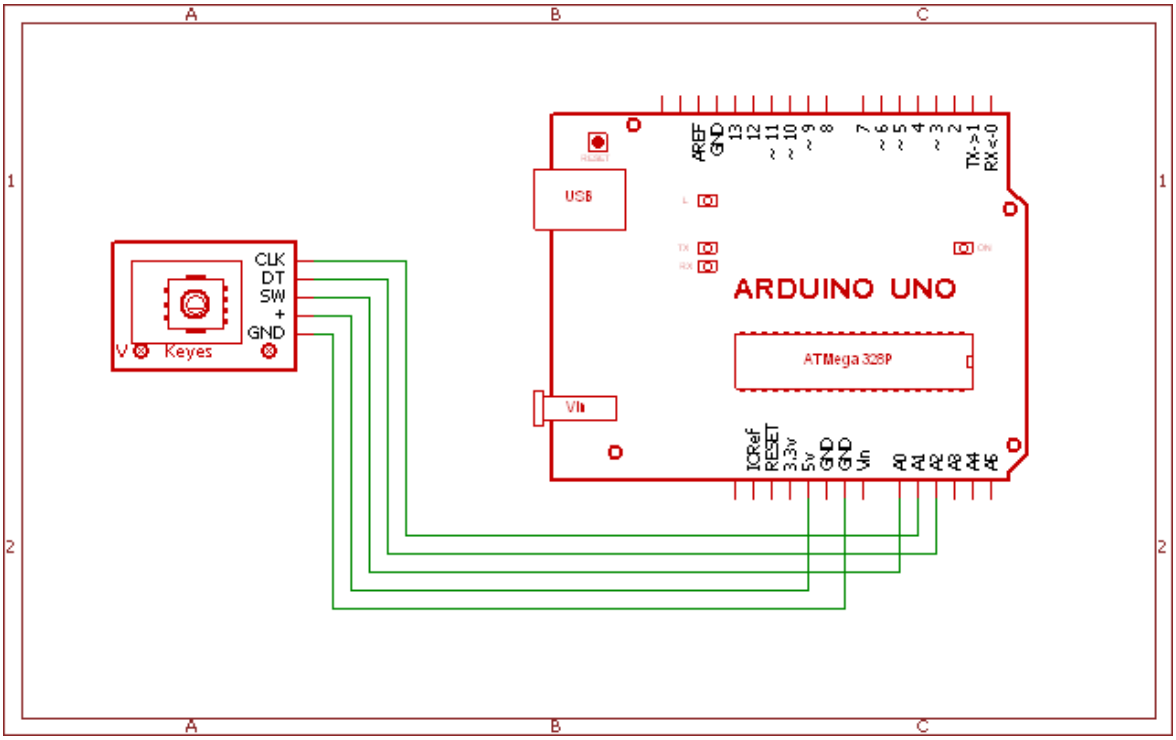
“SW”: “Switch”, pulsador axial.

“CK” o “CLK”: “Reloj”, uno de los conmutadores.

“DT”: “Datos”, el otro conmutador.

En el ejemplo que vamos a ver, lo conectaremos a tres entradas analógicas según los siguientes esquemas:





Definimos un umbral por debajo del cual consideramos que el pulsador o el conmutador están cerrados y también, etiquetas para las patas a las que conectamos las entradas:

```
#define UMBRAL 512
#define SW A0
#define CK A1
#define DT A2
```

También, necesitamos declarar una variable global que contenga el último estado:

```
boolean RELastState=(analogRead(CK)<UMBRAL);
```

Ahora ya, creamos una función que, al llamarla, nos devuelva un número según el movimiento del codificador rotativo:

```
byte RERead(){
  // Lee el Rotary Encoder y devuelve un código
  // según la lectura realizada:
  //  SIN MOVIMIENTO = 0
  //  GIRO HORARIO = 1
  //  GIRO ANTIHORARIO = 2
  //  PULSACIÓN = 4
  //  GIRO HORARIO PULSADO = 5
  //  GIRO ANTIHORARIO PULSADO = 6
  byte codRet=0;
  if(analogRead(SW)<UMBRAL){codRet+=4;}
  boolean REState=(analogRead(CK)<UMBRAL);
  if(REState!=RELastState){
    if((analogRead(DT)<UMBRAL)!=REState){codRet+=1;}
    else{codRet+=2;}
    RELastState=REState;
  }
  return codRet;
} // Fin de byte RERead().
```

Si en vez de conectar a entradas analógicas lo hiciéramos a entradas digitales (por ejemplo, 3, 4 y 5):

```
#define SW 3
#define CK 4
#define DT 5
```

Aquí no necesitamos umbral. Declaramos la variable global:

```
boolean RELastState=(digitalRead(CK)==LOW);
```

Y la función sería:

```
byte RERead(){
  // Lee el Rotary Encoder y devuelve un código
  // según la lectura realizada:
  //  SIN MOVIMIENTO = 0
  //  GIRO HORARIO = 1
  //  GIRO ANTIHORARIO = 2
  //  PULSACIÓN = 4
  //  GIRO HORARIO PULSADO = 5
```

```
    //  GIRO ANTIHORARIO PULSADO = 6
    byte codRet=0;
    if(digitalRead(SW)==LOW){codRet+=4;}
    boolean REState=(digitalRead(CK)==LOW);
    if(REState!=RELastState){
      if((digitalRead(DT)==LOW)!=REState){codRet+=1;}
      else{codRet+=2;}
      RELastState=REState;
    }
    return codRet;
  } // Fin de byte RERead().
```

Por ejemplo, se podría utilizar un codificador rotativo de este tipo en el mando de un coche teledirigido, para simular el volante y permitir pulsarlo para llevar la dirección al centro. En muchos casos, la utilización de un codificador rotativo con pulsador axial nos va a permitir realizar todo el manejo de un aparato sin necesitar teclados ni botoneras.

El esquema eléctrico para un codificador rotativo relativo que mostramos tiene 4 brazos de conmutación por lo que realizaría 8 conmutaciones por vuelta. En realidad, el que se muestra en la foto realiza 30 conmutaciones por vuelta; es decir, una conmutación cada 12 grados.

Los ejes tanto de los potenciómetros como de los conmutadores relativos suelen ser de 6 mm. de diámetro así que casi siempre que se utilicen con Meccano habrá que construir algún adaptador. Existen sin embargo, potenciómetros con eje de 4 mm. aunque son difíciles de encontrar.
También puede ser útil utilizar un codificador rotativo absoluto (por ejemplo, un potenciómetro) para limitar algún movimiento sin tener que recurrir a interruptores de fin de carrera.

Reducción armónica con engranajes

Esteban Orozco Vallejo

Introducción

Este es un tema de ingeniería mecánica muy conocido y que he ido viendo en la documentación indicada en el apartado siguiente. Hay dos variedades, en la primera se utilizan pares de engranajes que difieren en uno solo o en dos dientes y en la segunda se utiliza una rueda de erizo y una cadena con varios pasos, en general dos, superiores al número de dientes de la rueda de erizo.

En este modelo estudiamos la reducción armónica por engranajes utilizando ruedas dentadas de 56 y 57 dientes y piñones de 19 y 20 dientes. A su vez haremos dos tipos de montajes, el primero con giro epicíclico a mano y el segundo con el giro de un bloque epicíclico accionado por un piñón exterior.

Documentación

- The International Meccanoman nº 9, mayo de 1993
  "Harmonic Drive", por Michael Adler
- The International Meccanoman nº 65, marzo de 2012
  "T 755 : Powerful Epicyclic Reduction"
- Midlands Meccano Guild Gazette, abril de 1998
  "A look at Rod Rich´s Black Meccano System", por Mike Edkins
- "El libro del robot", de Richard Pawson
- CAM France Magazine nº 140, 4º trimestre de 2017
  "Roulement épicycloïdal", de Aubin Fanard
- Constructor Quarterly nº 20, junio de 1993
  "A Guide to Harmonic Drive, por Alan Partridge
- Constructor Quarterly nº 21, septiembre de 1993
  "More on Harmonic Drives", por Tony Rednall
- Constructor Quarterly nº 23, marzo de 1994
  "Harmonic Drive using Bevel Gear and Contrate", por Alan Partridge
- Internet. TAF : Transmisión armónica flexo-ondulatoria. ElectricBricks Lego
  "Harmonic Drive", por Akiyuki y Kavakuri
  "GBC module. Strain Wave Gears", "GBC Models of Cycloidal Drive"
- Internet. Reducción armónica con rueda y cadena (sólo para instrumentos musicales)

Tecnología Y Construcción . Montaje N º 1

Como hemos dicho se trata de utilizar pares de engranajes que difieran en uno solo o dos dientes. El mecanismo con esta tecnología se llama Paradoja de Ferguson, en referencia al astrónomo James Ferguson (1710 – 1776), que la utilizó para construir planetarios. Para aclarar el concepto de esta tecnología nos referiremos a las ruedas de 56 y 57 dientes. Hay un engranaje intermedio de 19 o 20 dientes que actúa a la vez sobre las ruedas de 56 y 57 dientes, que están en dos ejes independientes y no solidarios, aunque en la misma línea. Cuando este engranaje intermedio gira de modo epicicloidal, en una revolución ha recorrido 57 dientes (rueda fija y sin girar en el primer eje), pero en el segundo eje ha recorrido solo 56 de los 57 dientes. Con lo cual el eje de salida se habrá movido sólo un diente, dando una reducción 1 : 56. Véase la figura 1.

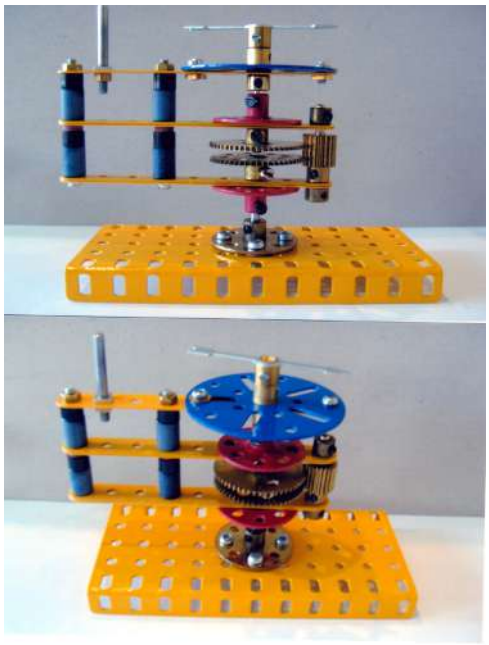


Figura 1. Paradoja de Ferguson

Tecnología Y Construcciön . Montaje Nº 2

Nos basamos en el esquema de la figura 2, tomada del International Meccanoman nº 65. Un piñón exterior de 19 dientes hace girar un bloque epicíclico para accionamiento de dos ruedas dentadas de 56 y 57 dientes.

A partir de dicho esquema construimos el modelo en Meccano (véanse las figuras 3 y 4). El eje de entrada tiene una manivela y un piñón exterior de 19 dientes que engrana, a través de una rueda dentada de 95 dientes, con un bloque epicíclico que lleva un eje en un agujero exterior con piñones de 20 y 19 dientes solidarios a dicho eje. Estos piñones engranan con ruedas de 56 y 57 dientes

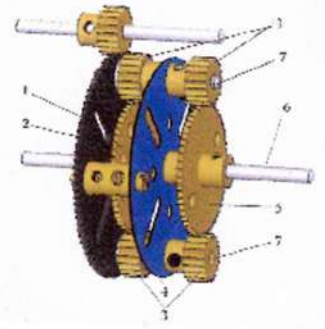


Figura 2

respectivamente. En el final de este eje de entrada está fija la rueda de 56 dientes. El segundo eje, independiente y no solidario con el primero aunque alineado con él lleva fija como comienzo la rueda de 57 dientes y tras una conexión a 90º con dos ruedas cónicas (pieza nº 30), con razón 1 : 1(véase figura 5) el giro aparece en el dial final de la placa frontal azul con 10 agujeros exteriores y una flecha indicativa. Respecto al esquema hemos cambiado la rueda de entrada de 57 a 56 dientes y añadido una tercera placa azul al tambor epicíclico que gira también libremente sobre el eje de salida.

Con solo 20 ó 19 dientes en los dos piñones, según se indica en la figura 2 el mecanismo no ha funcionado y sí, como hemos dicho antes, engranando el piñón de 20 dientes con la rueda de 56 dientes y el piñón de 19 dientes con la rueda de 57 dientes. Luego hablaremos de las posibles causas de ello.

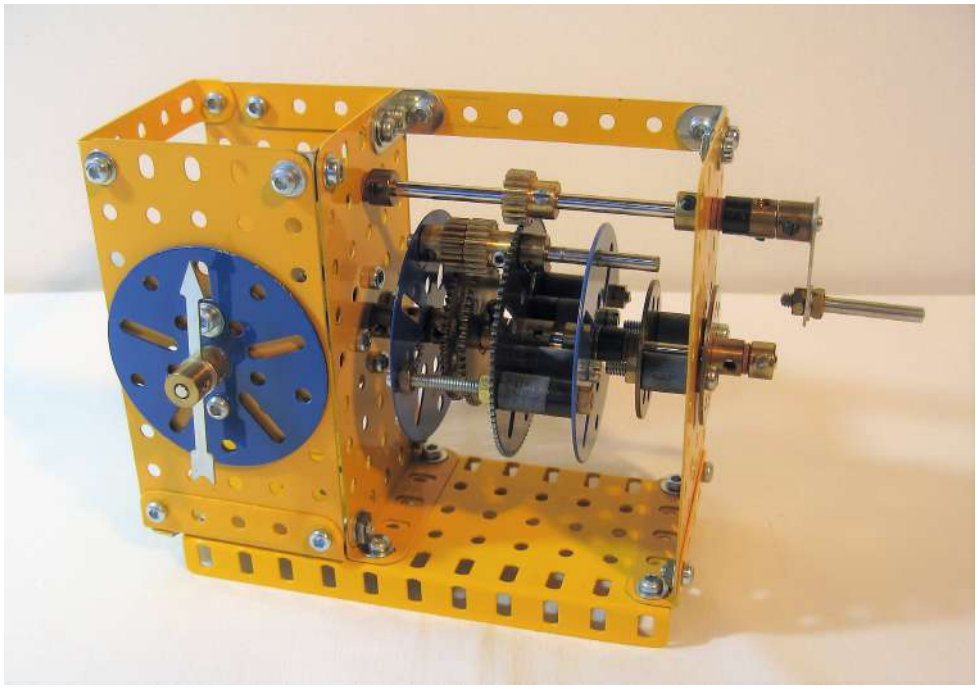


Figura 3 . Modelo en Meccano

Funcionamiento Del Montaje Nº 2

Actuamos de modo experimental. Superponemos al dial final un graduador transparente.

- Una vuelta (360º) de la manivela inicial

El eje final gira un poco menos de 4º, es decir 360º / 95 = 3º,8. La razón de reducción es 1 : 95.

- Cinco vueltas (5x360º = 1.800º) de la manivela inicial

Hemos dado una vuelta completa, a través del bloque epicíclico de 95 dientes, a la rueda fija de 56 dientes y obtendríamos en teoría allí una reducción de 1 : 57. En el dial final nos sale 19º = 1.800º / 95. Es decir la razón de reducción es 1 : 95.

- Diez vueltas (10x360º = 3.600º) de la manivela inicial

Nos sale 38º = 3.600 / 95 aproximados. La razón de reducción es también 1 : 95.

- Veinte vueltas (20x360º = 7.200º) de la manivela inicial

Nos sale 76º = 7.200 / 95 aproximados. La razón de reducción es también 1 : 95

Y así sucesivamente. Es probable que la justificación de esta razón de reducción 1 : 95 sea que tras mover en la parte de entrada del modelo los 95 dientes de la rueda del bloque epicíclico se logra que en la parte de salida se avance sólo un diente, derivado de la actuación conjunta sobre las ruedas de 56 y 57 dientes de los engranajes intermedios.

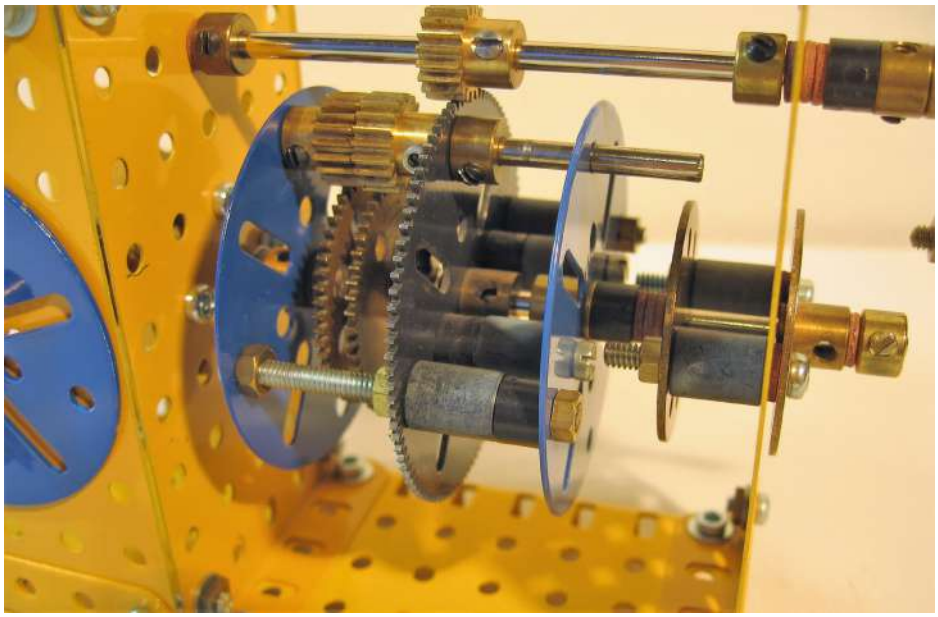


Figura 4 . Detalle del bloque epicíclico

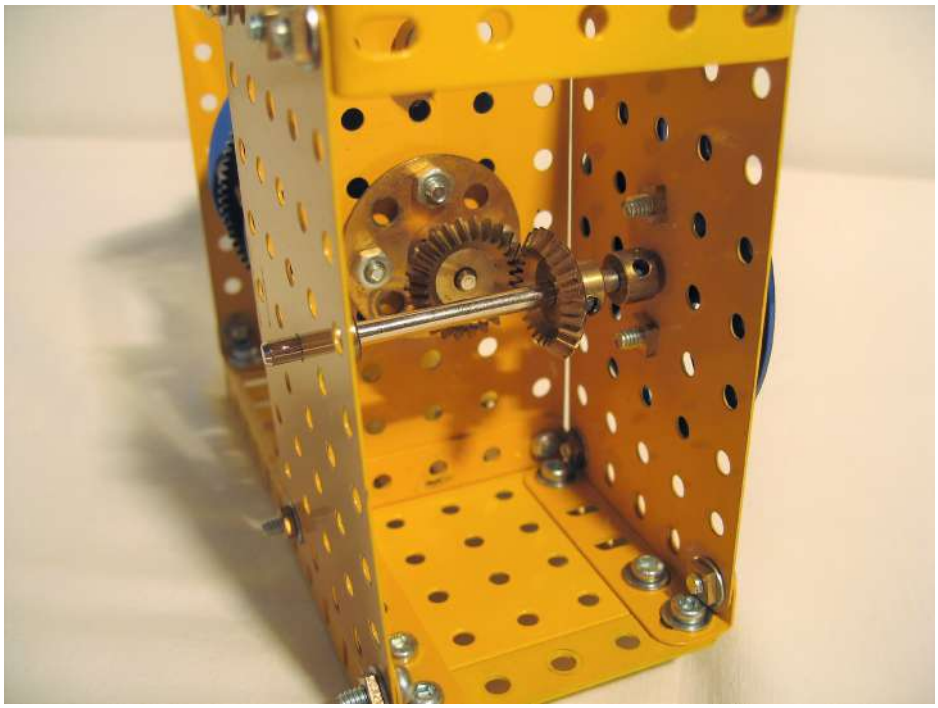


Figura 5 . Conexión en el eje de salida

Con una consideración purista dicen en varios artículos que tendríamos que operar con ruedas de 56 y 57 dientes de radio distinto porque, con el mismo paso de los dientes, las circunferencias son distintas. Como es sabido las ruedas dentadas de Meccano de 56 y 57 dientes tienen el mismo radio. Intento comprar en Metallus una rueda de 56 dientes, que creo tendrá un radio correcto y otro piñón de 20 dientes. Esto último para hacer un eje intermedio más, pues con dos el accionamiento sería más estable. Pero Metallus está en liquidación y no aparecen ya como disponibles. También es imposible pedir a Ashok una o dos piezas sueltas. Como funciona bien la disposición 56-20 y 57-19 me contento ya con ello.

Comentario General

Evidentemente este sistema, basado en la Paradoja de Ferguson, permite obtener grandes reducciones, ocupando el mecanismo un volumen pequeño.

En algunos de los documentos citados se dice que esta disfunción 56 / 57 dientes genera una onda (además “armónica”) que se propaga, a semejanza de otros casos de disfunción o error en Física. Pero esto es difícil de ver de modo directo o intuitivo.

Este tema de reducción armónica por engranajes o cadena tiene una gran variedad y complejidad. Puede construirse desde el modelo más sencillo hasta los más complicados, como los engranajes flexibles. A estos últimos se refiere el documento de Internet “GBC Strain Wave Gears” de ElectricBricks Lego.

Este modelo sobre la reducción armónica con engranajes que hemos construido nos ha exigido bastante esfuerzo y es un ejemplo de que una idea general no basta para construir un modelo que funcione. También hemos constatado que para modelos tecnológicos como este convendría que la precisión de las piezas de Meccano fuera mayor.

Es nuestro deseo construir en breve plazo un modelo de reducción armónica con rueda de erizo y cadena.

Maquina taladradora industrial de gran tamaño

José Enríquez Victoria

Taladro de agujeros multiples

El enorme progreso de la ingeniería que ha tenido lugar en los últimos 100 años ha requerido el correspondiente desarrollo en la maquinaria empleada en la preparación de los componentes de las máquinas, motores y estructuras de todo tipo.

Esto ha evolucionado con la mecánica de precisión, sin la cual las operaciones de hoy no podrían llevarse a cabo.

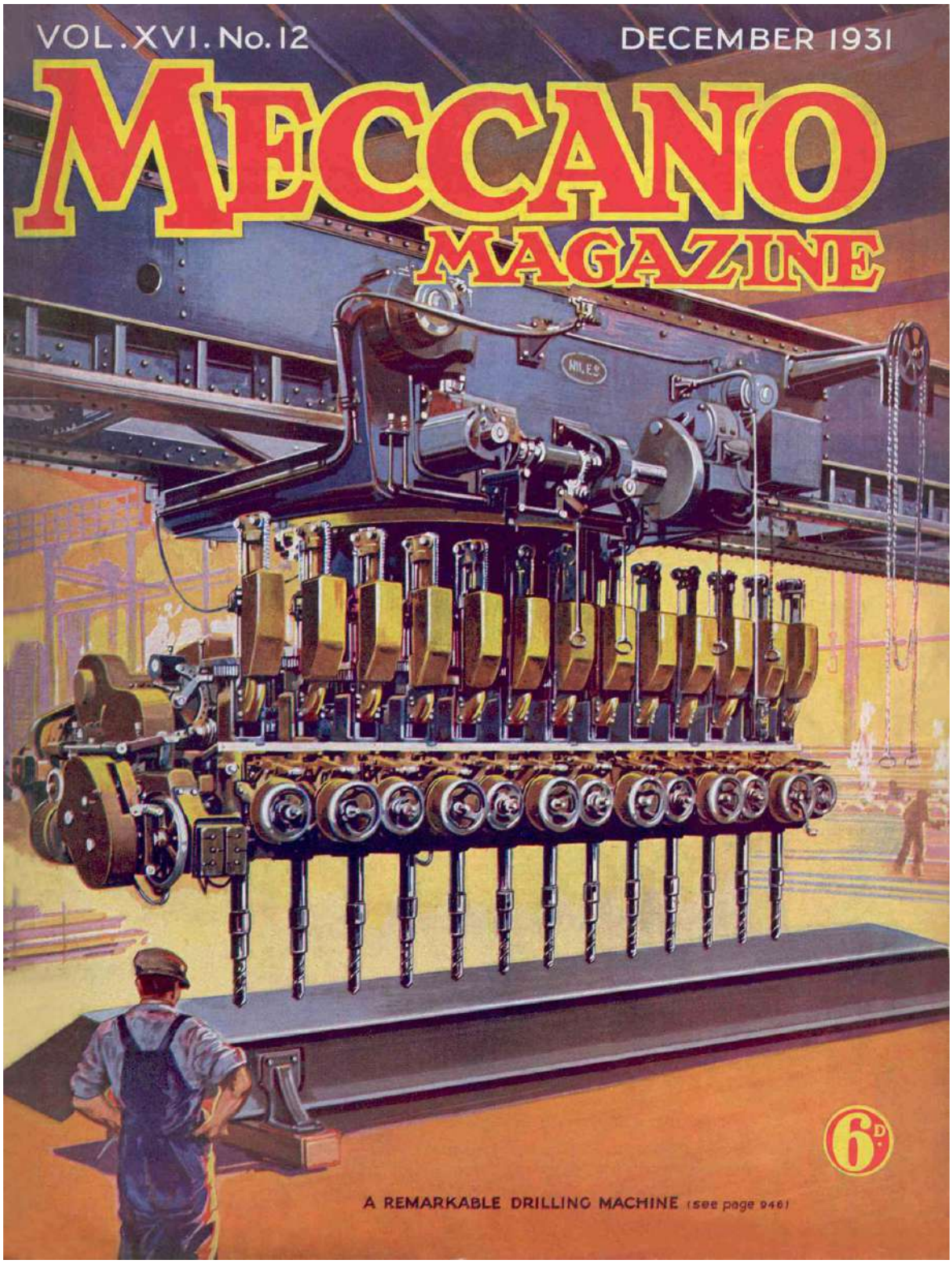
Debemos nuestras máquinas de precisión modernas a un pequeño número de pioneros en el arte del corte de metales, entre los cuales se pueden nombrar a: MAUDSLAY, WHITWORTH, CLEMENT entre otros. Se dijo de MAUDSLAY que se consideraba como el padre de las herramientas de precisión.

Henry Maudslay (1771-1831) fue un ingeniero e inventor británico gran innovador de las máquinas de precisión.
Comenzó a ayudar a su padre en la tienda de ultramarinos, pero a los 12 años comenzó a trabajar a una carpintería, y poco más tarde en una herrería.

En 1707 montó su propio taller de precisión. Su primera aportación fue construir una serie de 42 máquinas de carpintería para producir bloques de aparejo de madera destinados a la construcción de barcos para la Armada. El resultado constituyó el primer ejemplo conocido de maquinaria especializada en una cadena de montaje y fabricación en serie.

Aplicó la fórmula de piezas intercambiables y desarrolló el primer torno que mecanizaba tornillos. Estableció la estandarización sobre tamaño de rosca por primera vez, inventando el primer micrómetro de banco que era capaz de medir una diez milésima de pulgada y también se especializó en motores de vapor. Muchos ingenieros excepcionales se formaron en su taller, entre otros, Joseph Clement y sir Joseph Whitworth, etc.

Veamos una fotografía de donde he sacado el modelo Meccano.



Pasemos al modelo Meccano.
Este modelo tiene tres movimientos:

Traslado en ambos sentidos del puente.

El carro se traslada de izquierda a derecha.

Los taladros giran sobre sí mismos y las ruedas.

Veamos las fotos nº 1, 2 y 3.

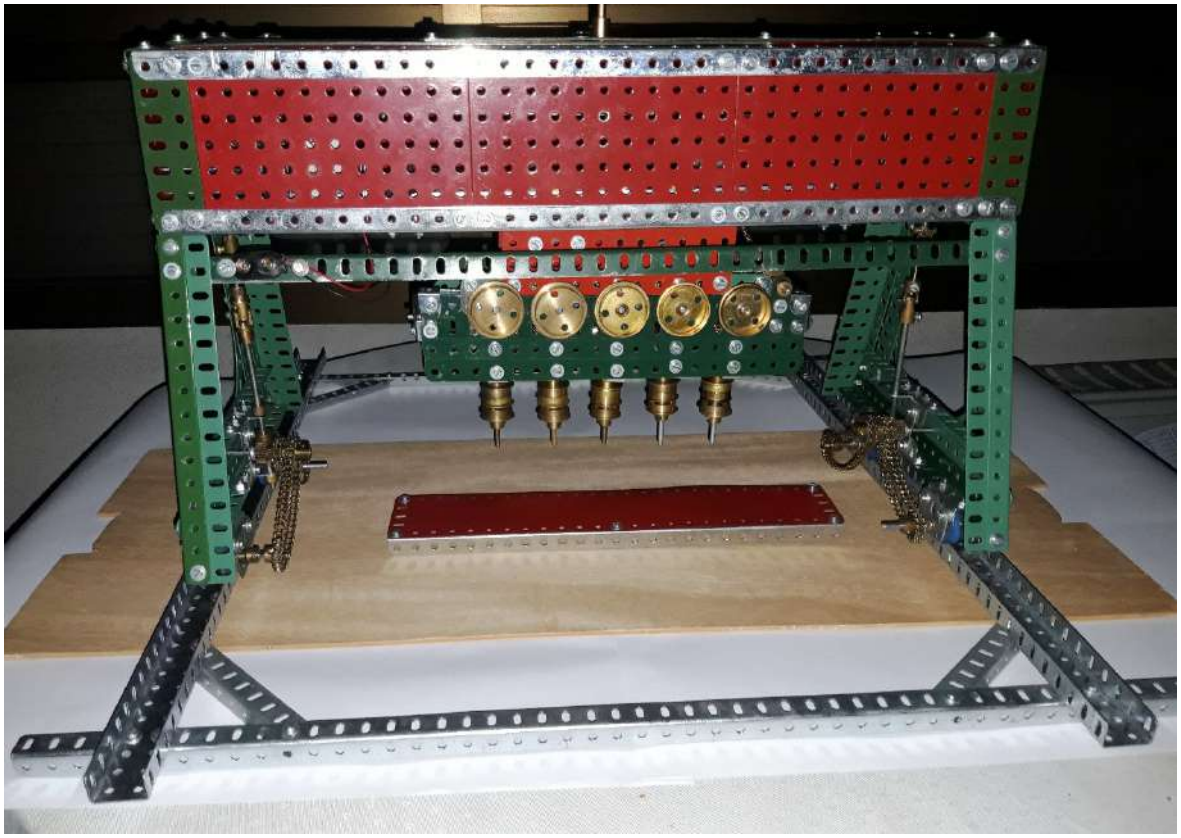


Foto nº 1. Vista general

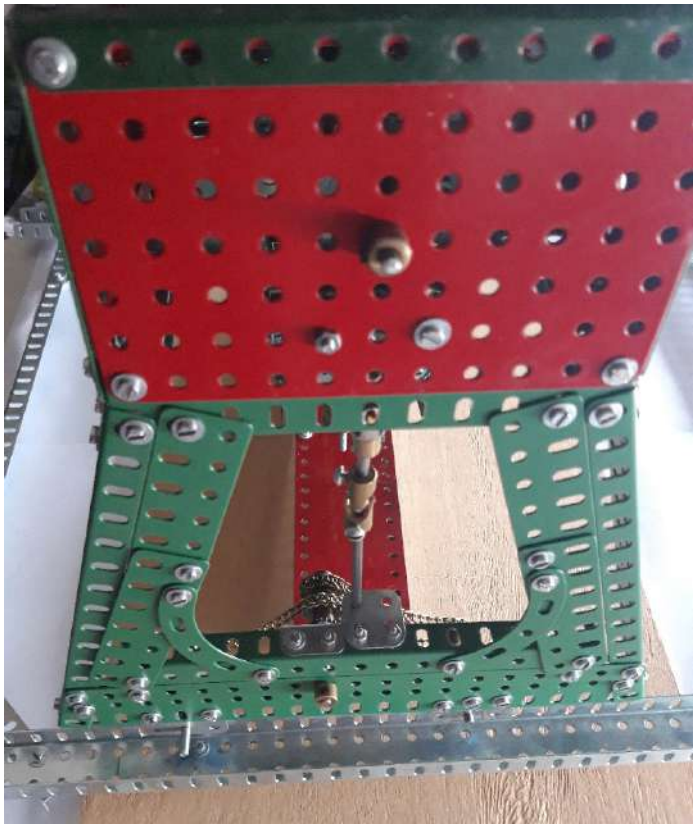


Foto nº 2. Vista lateral

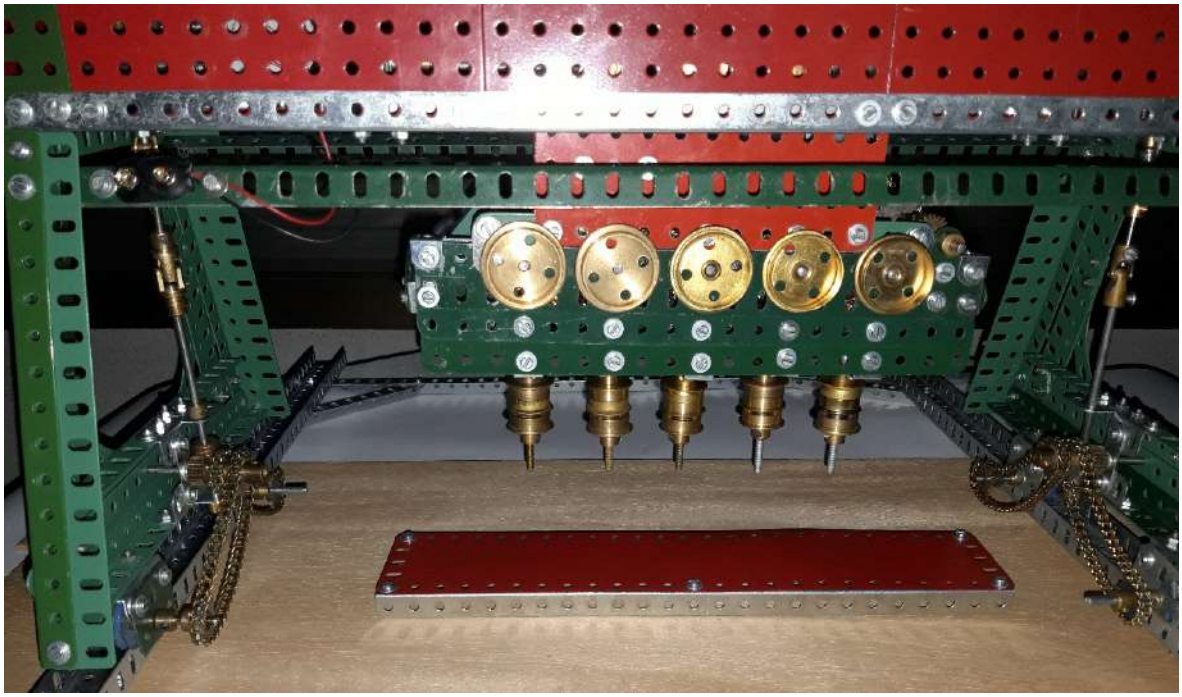


Foto nº 3. Vista general más cercana

En la foto nº 1 - Vista general, para el movimiento del traslado del puente me he fijado en el SUPERMODELO DE BLOQUES DE CEMENTO. He cambiado los rodamientos cónicos por cadenas, pero funcionaba bastante regular, así que decidí poner un engranaje sin fin y un piñón en el centro de las cadenas, a cada lado del puente con lo cual conseguí mucha más potencia, aunque por supuesto la velocidad queda notablemente reducida.

En cuanto a las ruedas he preferido poner poleas acanaladas de 25mm en lugar de las ruedas rebordeadas de 28mm, ya que estas ruedas ocasionan más roce que las poleas. Estas poleas pueden llevar una pequeña holgura en sus ejes, ya que sin esta holgura es muy difícil ajustar el puente.

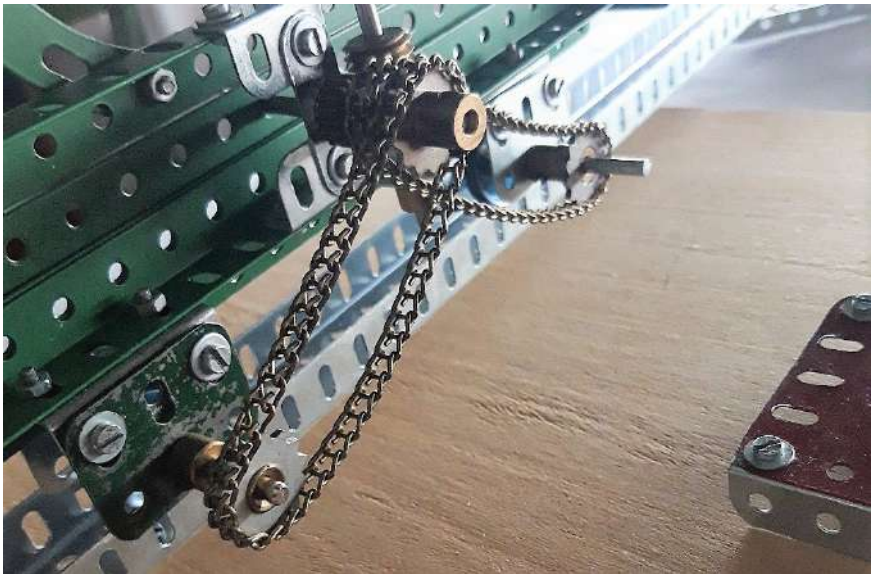


Foto nº 4.Vista de las cadenas para el arrastre del puente

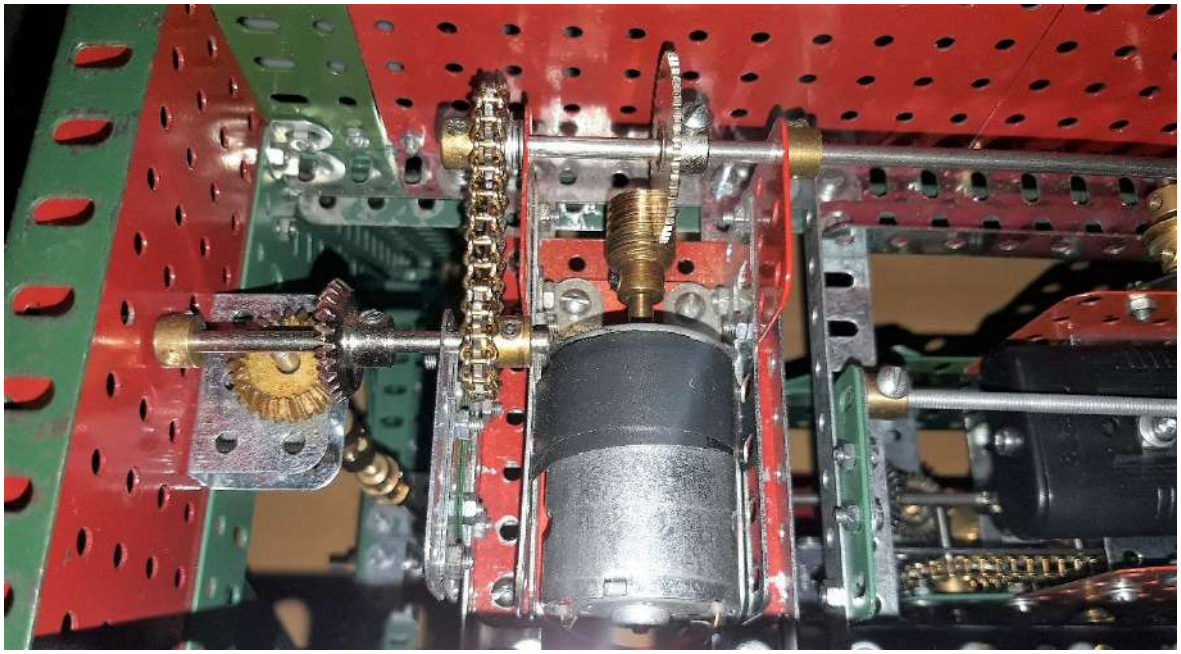


Foto nº 5. Transmisión del motor a las poleas / ruedas

El motor lleva un engranaje sin fin y una rueda dentada de 50 dientes, pequeña cadena con sus ruedas de erizo, a continuación ruedas cónicas. De este modo se logra el movimiento de las poleas. En la Foto nº 6 se ve como los ejes largos tienen cardán, pues sino la rigidez obstaculiza los giros.

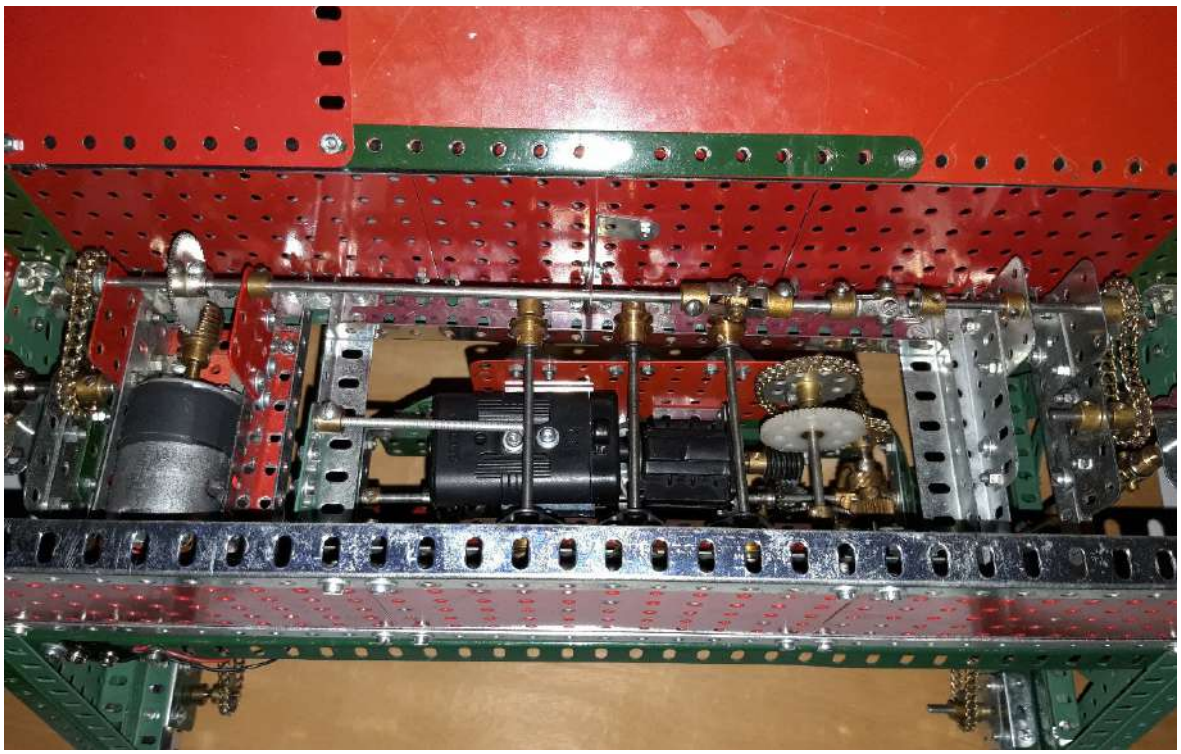


Foto nº 6. La transmisión a las poleas / ruedas del otro lado del motor

El otro lado de transmisión a las poleas es exactamente igual, cadenas, ruedas de erizo, y ruedas cónicas.

Si os fijáis el gran eje transversal lleva dos cardanes. Este eje atraviesa varios soportes, por lo que resultaba imposible que girase libremente, pero con los cardanes he suprimido este problema.

En la foto siguiente vemos que he puesto una tapa para que en caso de desajuste se pueda arreglar fácilmente.

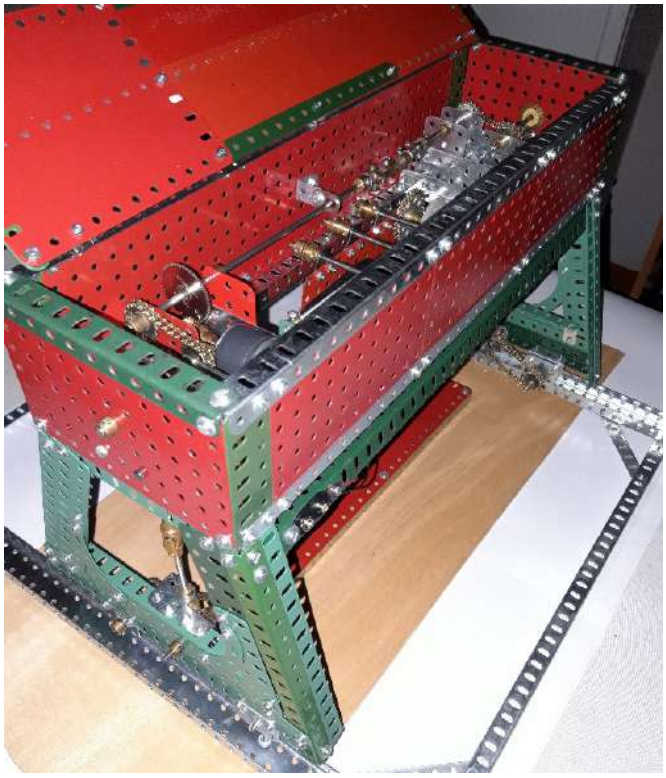


Foto nº 7.Tapa del modelo para poder abrir fácilmente en caso de desajuste.

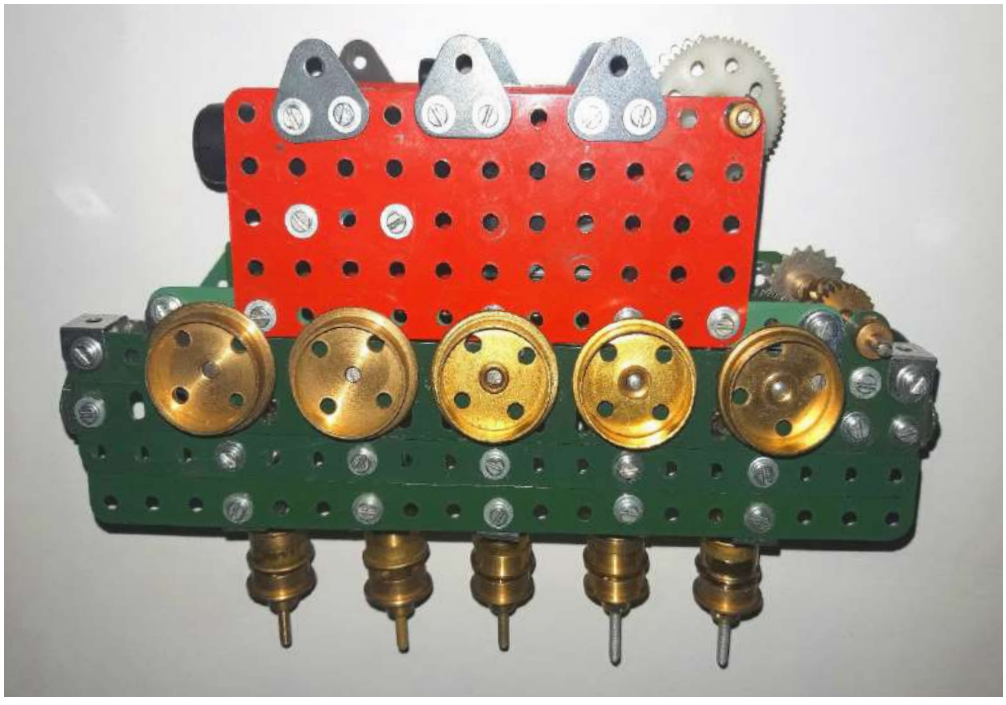


Foto nº 8.Vista general de los taladros

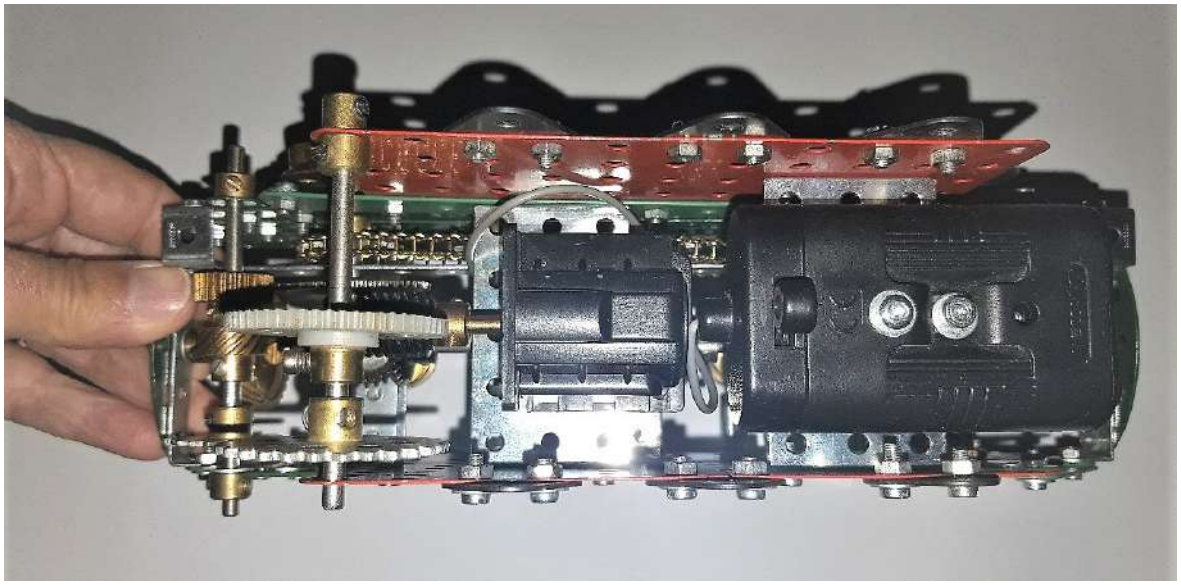


Foto nº 9. El movimiento de los taladros.

Foto nº 10. Perspectiva de los taladros.

En las fotos 9 y 10 observamos el movimiento de los taladros: un motor a pilas, distintos engranajes hasta llegar a un eje transversal y dos ruedas cónicas transmiten la rotación de cada taladro.

Las ruedas se mueven mediante una cadena, colocada entre el primer y quinto taladro y las demás giran por fricción.

Por ultimo para mover el carro he puesto un eje roscado.

Si hay alguna Exposición y la vuelvo hacer, el puente lo diseñaría más largo un mayor recorrido del carro.

Carretilla Elevadora

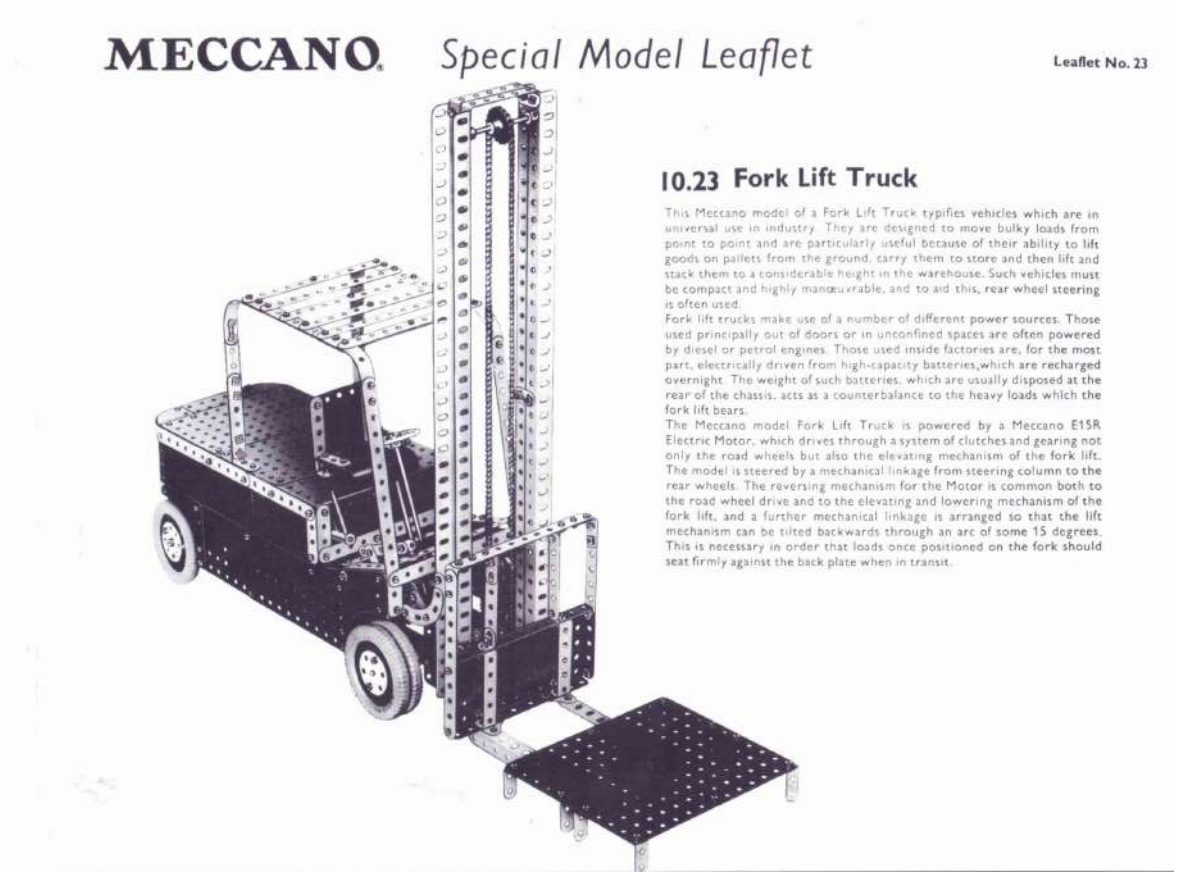
Una utilización conjunta de Meccano y Fischertechnik

Antonio Valero Aicua

Las carretillas elevadoras son vehículos ampliamente utilizados en la industria y en los almacenes, ya que facilitan el transporte de toda clase de objetos, normalmente embalados y colocados sobre palés, para situarlos en otros lugares o en estanterías mediante la elevación de la carga.

Hay varios modelos de carretillas elevadoras en los manuales Meccano y en revistas como Meccano Magazine, entre otras.

Para la construcción de esta carretilla elevadora me he orientado en el folleto nº 23 del equipo nº 10 que se publicó a partir de 1964.


MECCANO Special Model Leaflet

Leaflet No. 23

10.23 Fork Lift Truck

This Meccano model of a Fork Lift Truck typifies vehicles which are in universal use in industry. They are designed to move bulky loads from point to point and are particularly useful because of their ability to lift goods on pallets from the ground, carry them to store and then lift and stack them to a considerable height in the warehouse. Such vehicles must be compact and highly manœuvrable, and to aid this, rear wheel steering is often used.

Fork lift trucks make use of a number of different power sources. Those used principally out of doors or in unconfined spaces are often powered by diesel or petrol engines. Those used inside factories are, for the most part, electrically driven from high-capacity batteries,which are recharged overnight. The weight of such batteries, which are usually disposed at the rear of the chassis, acts as a counterbalance to the heavy loads which the fork lift bears.

The Meccano model Fork Lift Truck is powered by a Meccano E15R Electric Motor, which drives through a system of clutches and gearing not only the road wheels but also the elevating mechanism of the fork lift. The model is steered by a mechanical linkage from steering column to the rear wheels. The reversing mechanism for the Motor is common both to the road wheel drive and to the elevating and lowering mechanism of the fork lift, and a further mechanical linkage is arranged so that the lift mechanism can be tilted backwards through an arc of some 15 degrees. This is necessary in order that loads once positioned on the fork should seat firmly against the back plate when in transit.

Fig. 1

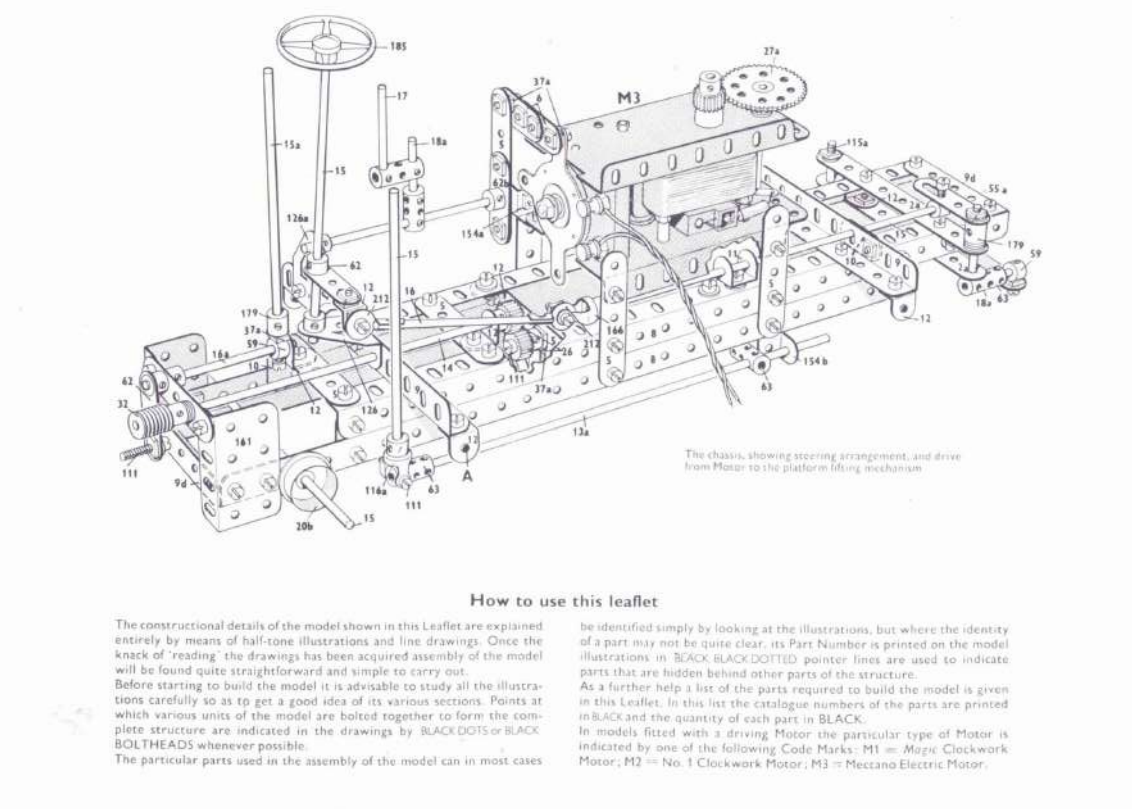


Fig. 2

Los folletos 21 a 30 aparecieron con los equipos amarillo, negro (después azul) y plata (después cincado). Inicialmente eran folletos del equipo núm. 9, pero cuando disminuyó el contenido de ese equipo pasaron a formar parte de los folletos del equipo nº 10, puesto que esos modelos del equipo nº 9 no se podían construir con el nuevo y reducido equipo nº 9.

Como figuras 1 y 2 se reproduce la portada y una de las páginas del mencionado folleto.

El modelo que he construido sigue en líneas generales esa carretilla elevadora, pero he introducido muchas modificaciones y mejoras, así:

En lugar de una cadena he puesto dos cadenas, una a cada lado, de las vías por donde sube y baja los dos “cuernos” que se introducen bajo los palés, con ello se obtiene más estabilidad del dispositivo.

He eliminado el motor eléctrico Meccano E15 y le he sustituido por un potente motor eléctrico cilíndrico, que es mucho menos ruidoso que el motor Meccano, además con ello he hecho sitio al receptor de radio control y los dos servos Fischertechnik.

Aunque no es fácil compatibilizar las piezas de Meccano con las de Fischertechnik, ya que las únicas piezas iguales son los ejes que en uno y otro sistema son de 4mm, he podido acoplar el receptor de radio control, dos servos, la batería de la radio y el conmutador, atornillándolos a las piezas Meccano, utilizando también algunos bloques de Fischertechnik, que tienen agujeros de 4mm.

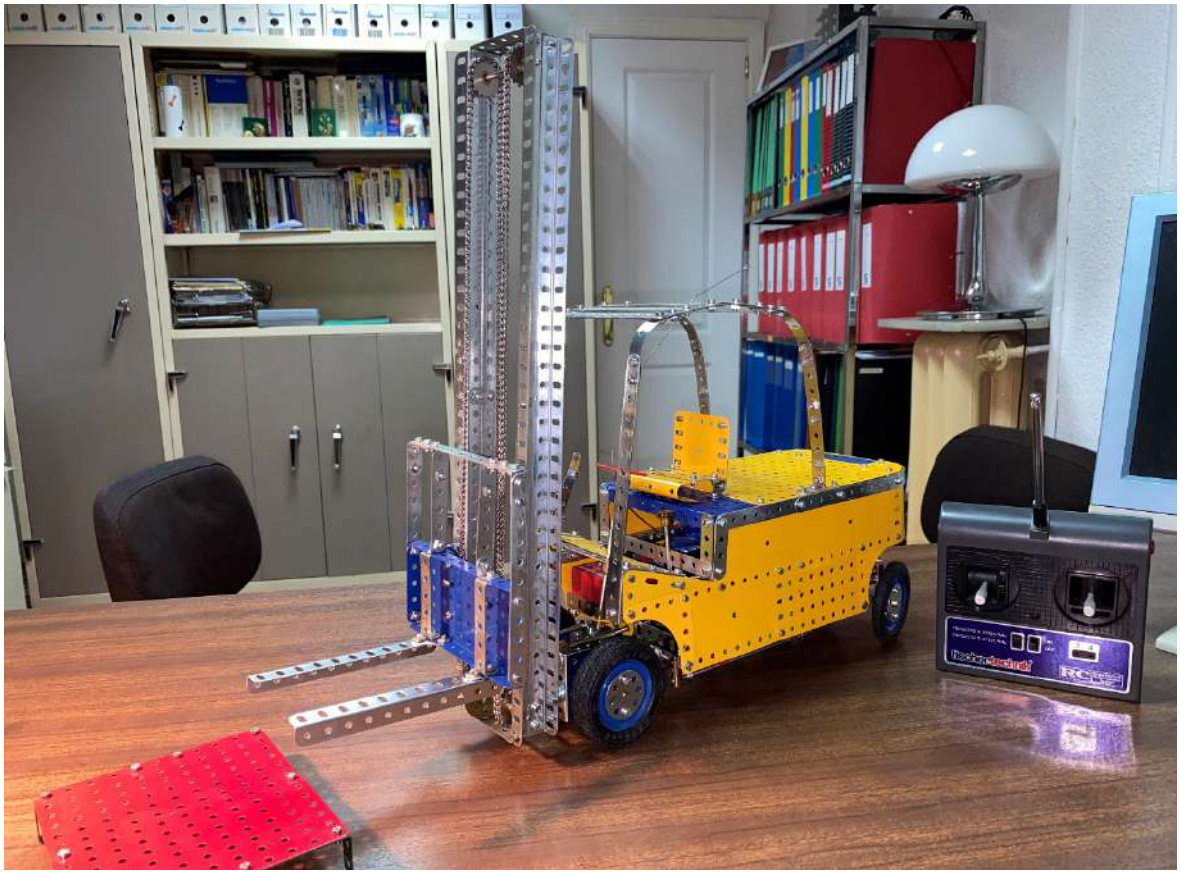


Fig. 3

La imagen nº 3 muestra todo el conjunto del modelo con el emisor de radio control de Fischertechnik y un palé.

Otras vistas exteriores del modelo aparecen como imágenes 4 y 5.

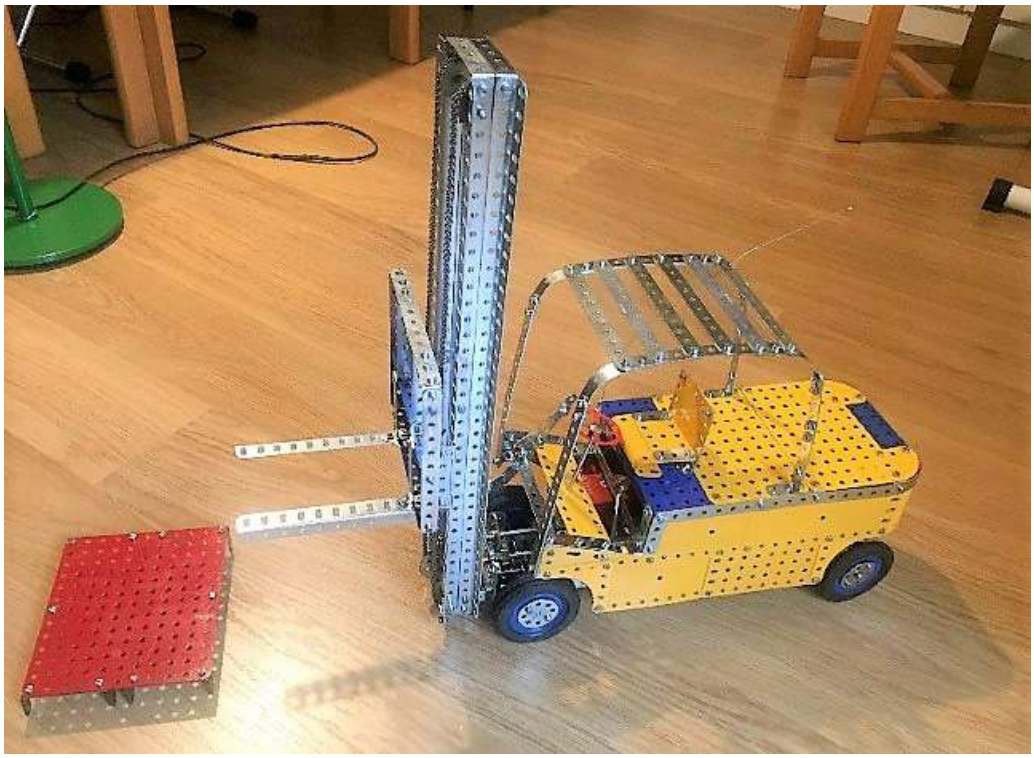


Fig. 4

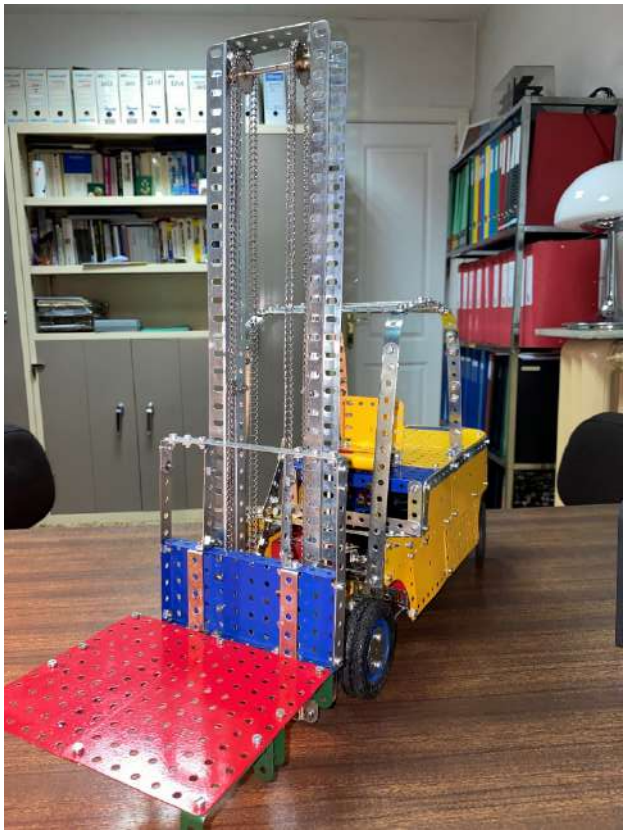


Fig. 5

El chasis del modelo y la colocación del receptor de radio y dos servos (piezas de color negro) se aprecian en la imagen nº 6. Igualmente se colocan en el chasis baterías recargables de Ni Cd, que alimentan el motor, y la batería recargable roja de Fischertechnik es la que alimenta el receptor de radio control y a los dos servos.

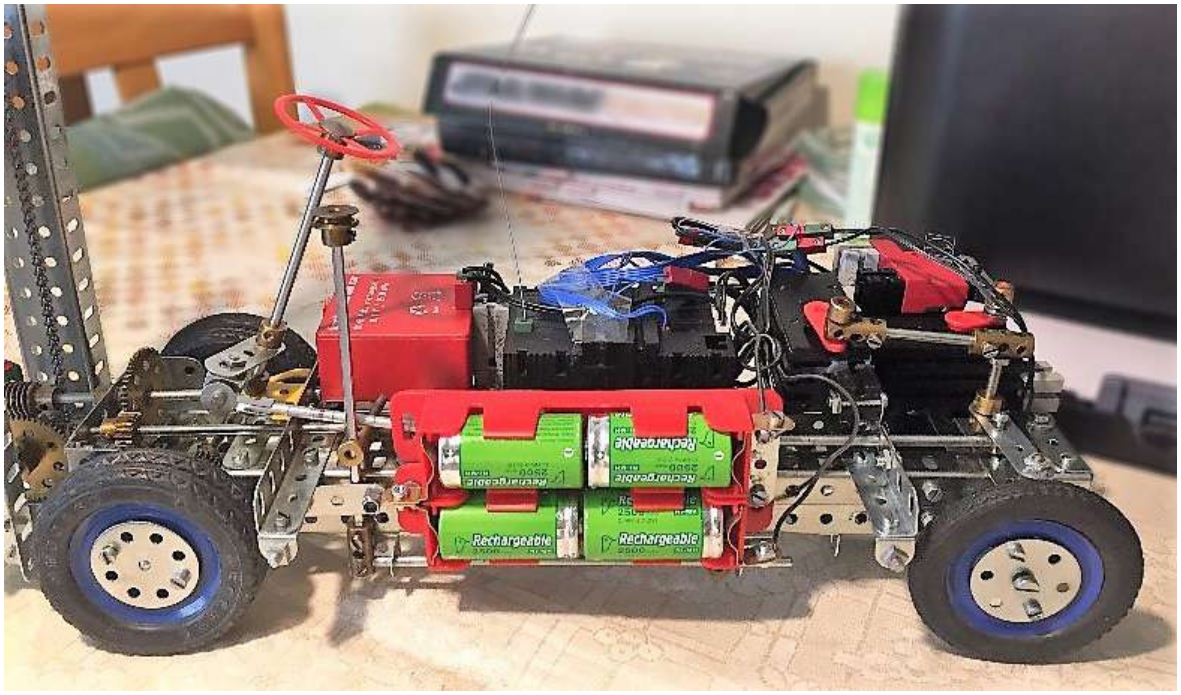


Fig. 6

En la imagen nº 7 se muestra el modelo por debajo, el motor eléctrico de 6 voltios y la transmisión a las ruedas motrices y a los engranajes de subida y bajada de la plataforma.

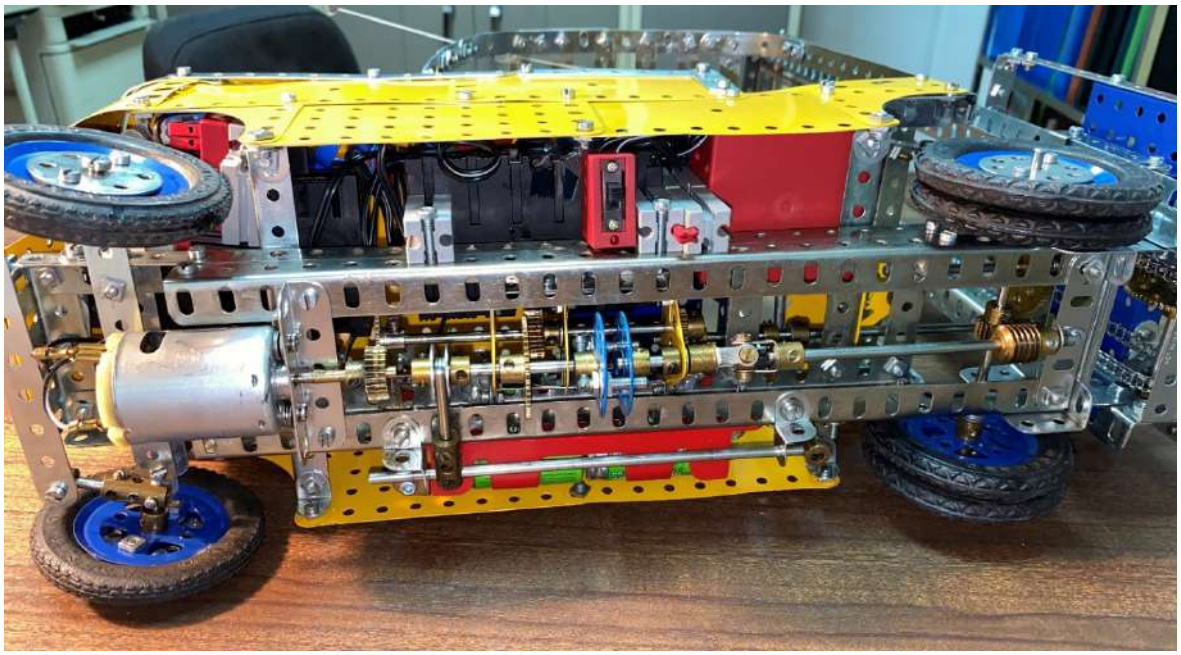


Fig. 7

En la imagen nº 8 se aprecian los dos servos Fischertechnik. Con el de la derecha actúa la dirección y el volante, y el de la izquierda mueve la palanca del conmutador Fischertechnik para la marcha adelante, atrás y paro, o cuando se engrana, con la palanca que está junto al eje del volante se controla la subida, la bajada y paro de la plataforma.

La foto nº 9 muestra los mecanismos de la tracción y de la elevación y bajada de la plataforma.

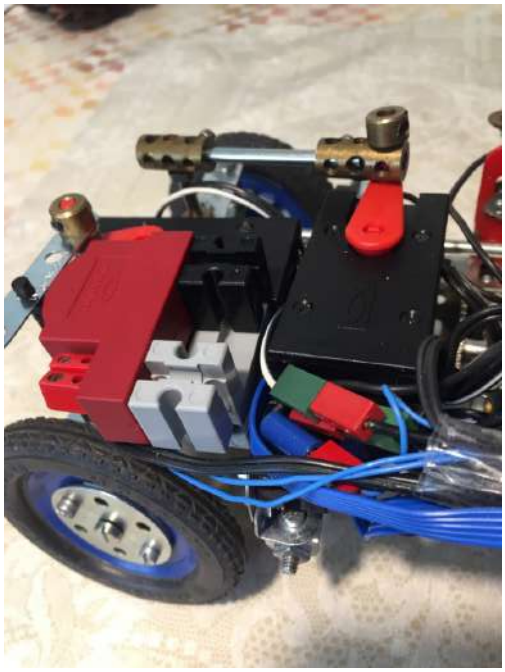


Fig. 8

Fig. 9

Un video de esta carretilla elevadora está subido a YouTube. Se accede al mismo en el siguiente enlace: https://youtu.be/bQVy_voVZf0

Este modelo no es el primer modelo en el que he combinado piezas de Meccano y Fischertechnik, hace algunos años construí un PinBall en el que se utiliza el mini comprensor los pistones neumáticos y el mini computador de Fischertechnik, que controlan los flasps y marca los tantos obtenidos.

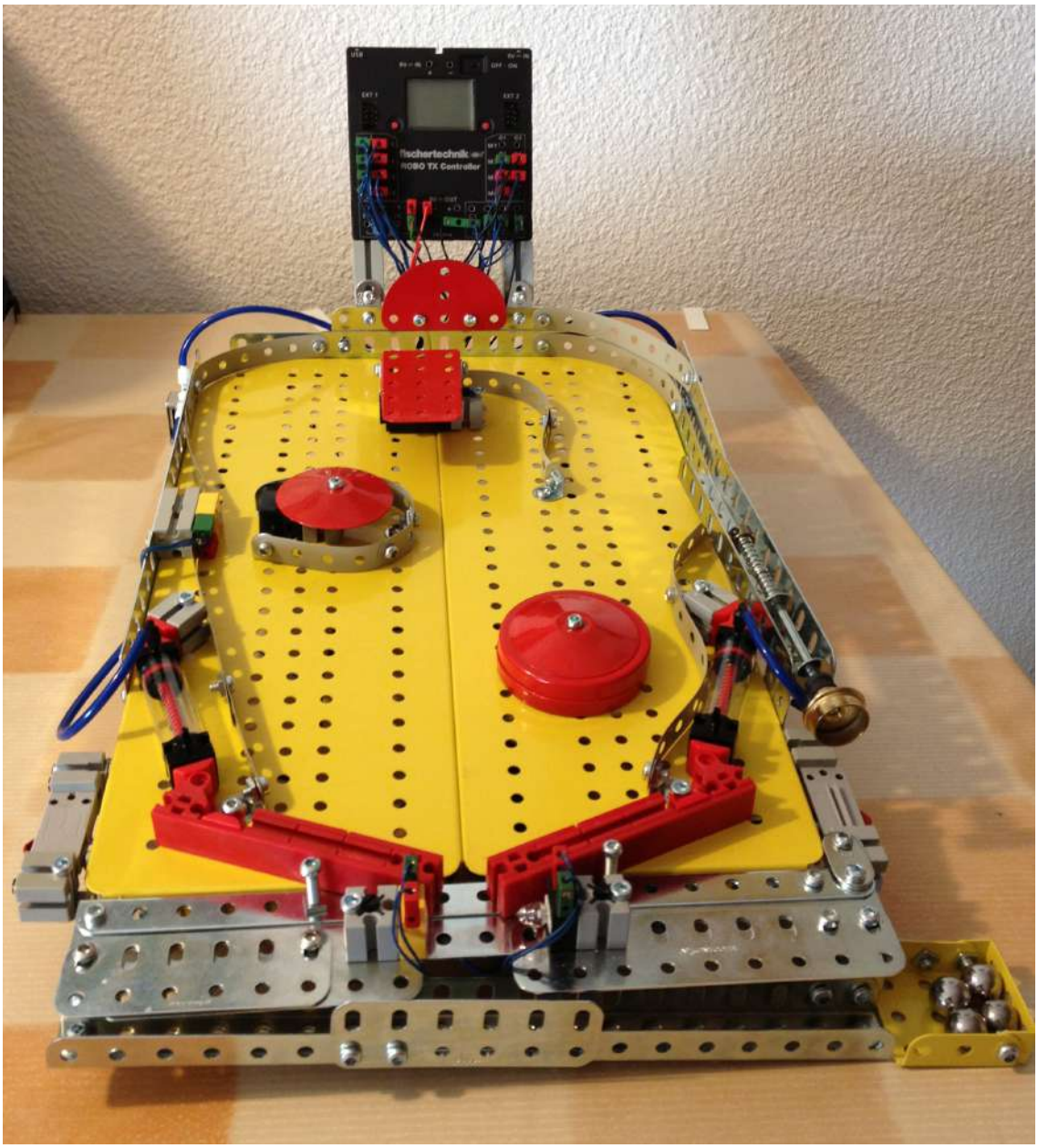


Fig. 10

Locomotora de W. Hedley Wylam Ocho Ruedas - 7 Engranajes

José Enríquez Victoria

W. Hedley Wylam (1779-1843) Reino Unido

Fue uno de los principales ingenieros industriales del siglo XIX, realizó importantes innovaciones en el desarrollo ferroviario.
Construyó la primera locomotora práctica.

Antes de la época de Hedley, eran demasiado pesadas, la mayoría de las líneas utilizaban cables con motores estacionarios.

En1804 Richard Trevithick realizó la primera locomotora de vapor. En 1813 William Brunton, Escocia (1777-1851) construyó su “Viajero Mecánico”.

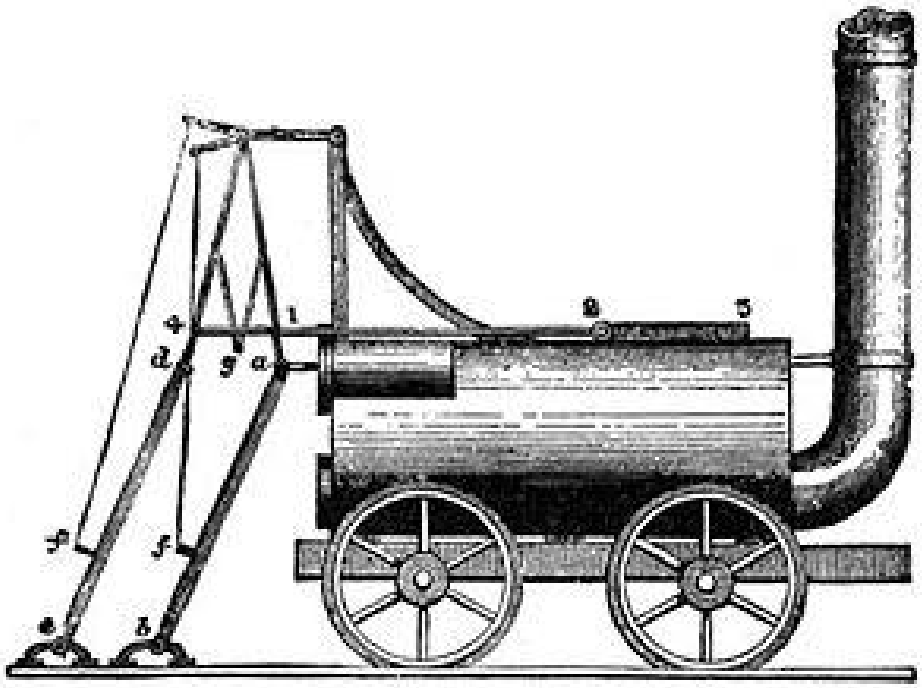
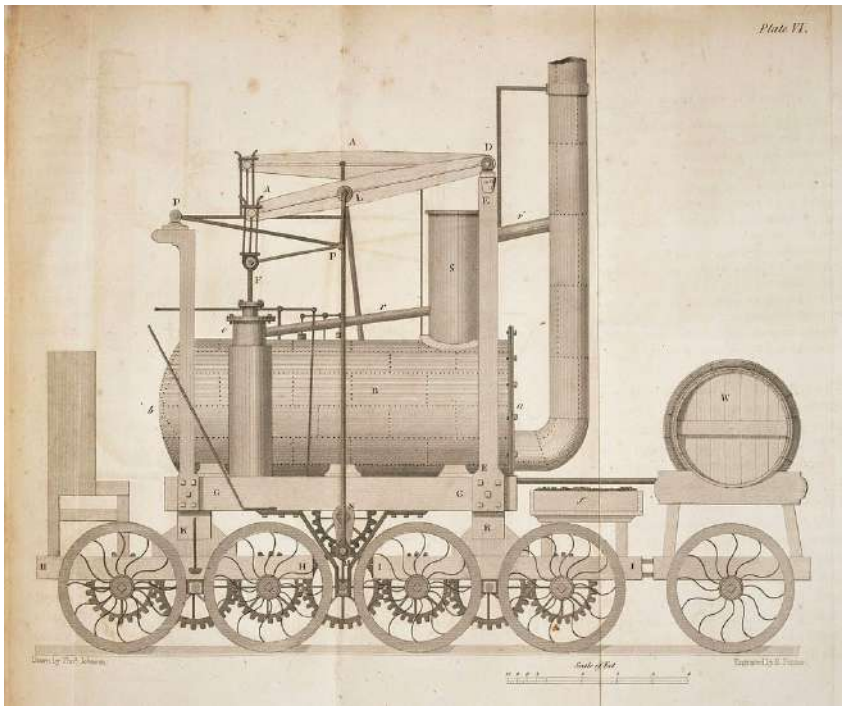


FIG. 1. BRUNTON'S TRAVELLER, 1813.

Tenía un par de patas mecánicas con pies que agarraban a los rieles en la parte trasera del motor para empujarlos hacia adelante a unos tres kilómetros por hora.
Nadie creía que las ruedas de acero sobre rieles darían suficientes adherencias hasta que Robert Stephenson y William Hedley demostraron lo contrario. Hay otros muchos ejemplos, pero he puesto este modelo por ser bastante curioso.

Cuando vi por internet esta fotografía con bastantes engranajes, 7 en cada lado y cuatro ruedas más 1 de apoyo pensé que podía ser un buen modelo Meccano.



Veamos estas 3 fotos generales.
En estas 3 fotos se aprecian bastante bien, los pistones, las bielas y las chimeneas, en realidad es un modelo bastante fácil para un meccanero que ha realizado muchas locomotoras.

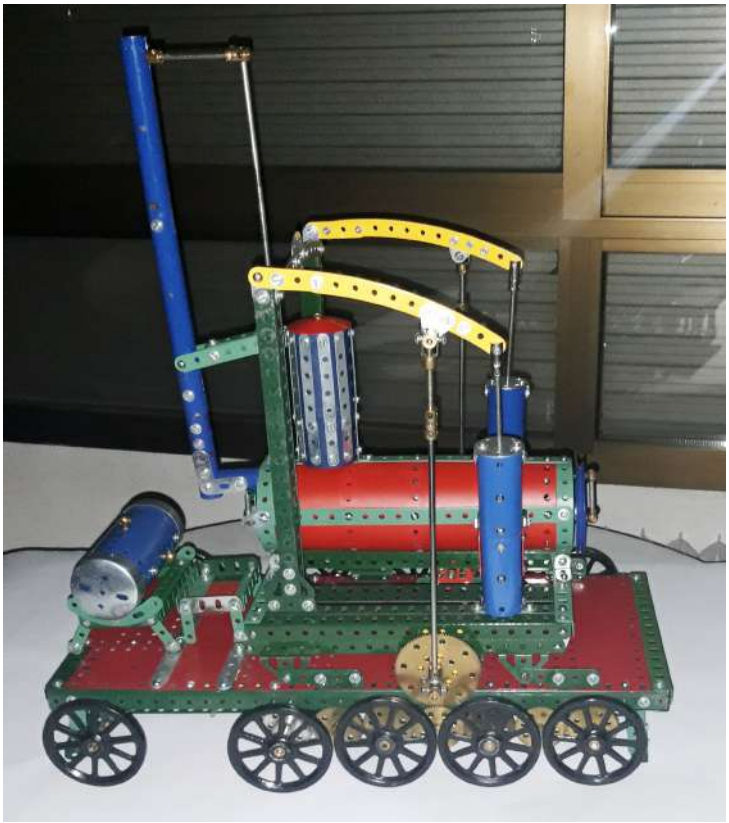


Foto nº 1.Vista general

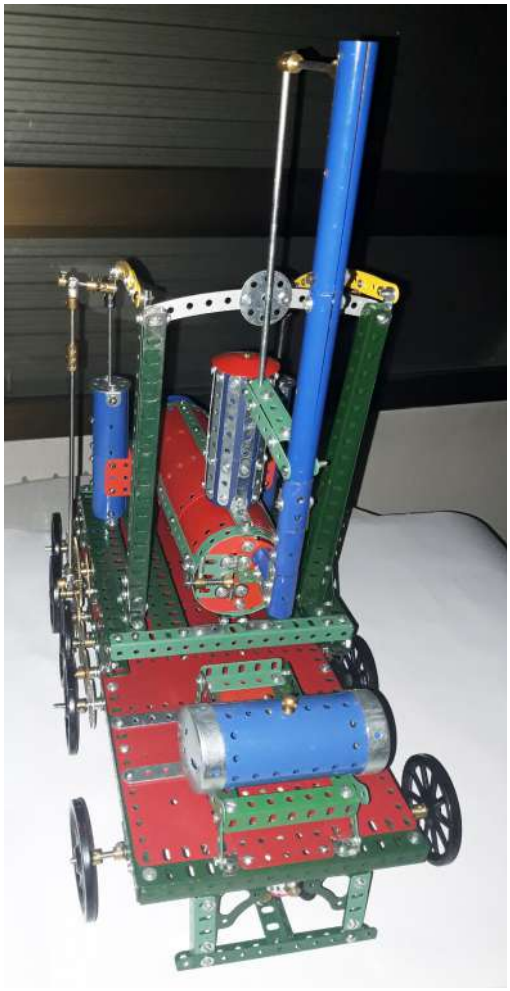


Foto nº 2.Vista posterior

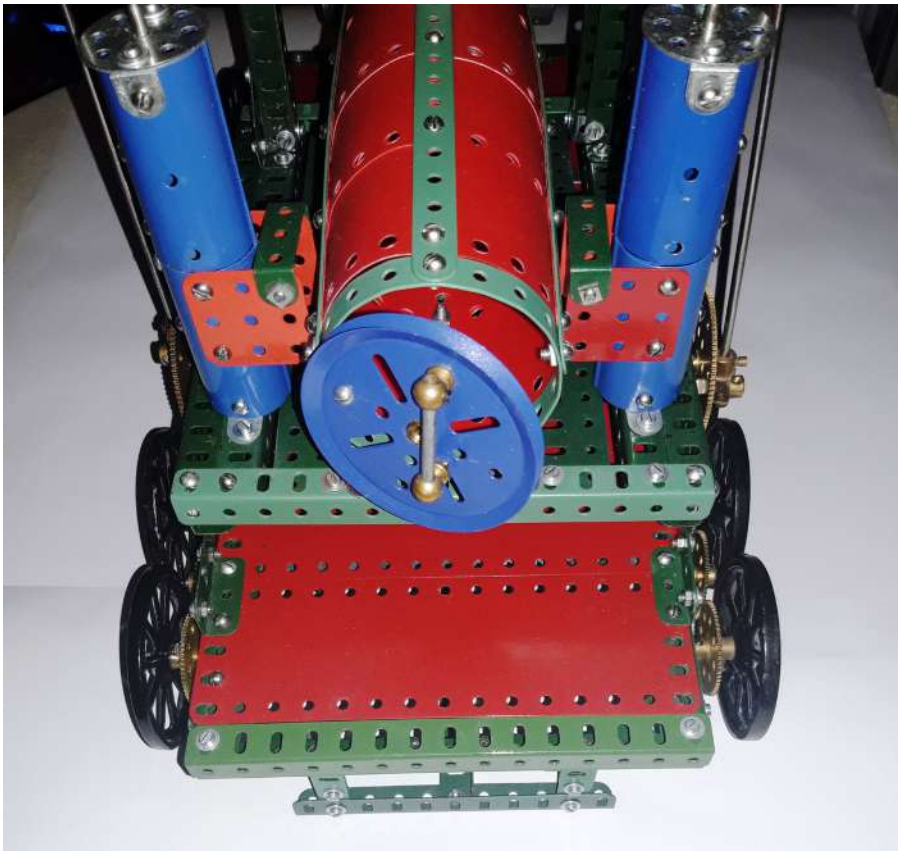


Foto nº 3. Detalle frontal

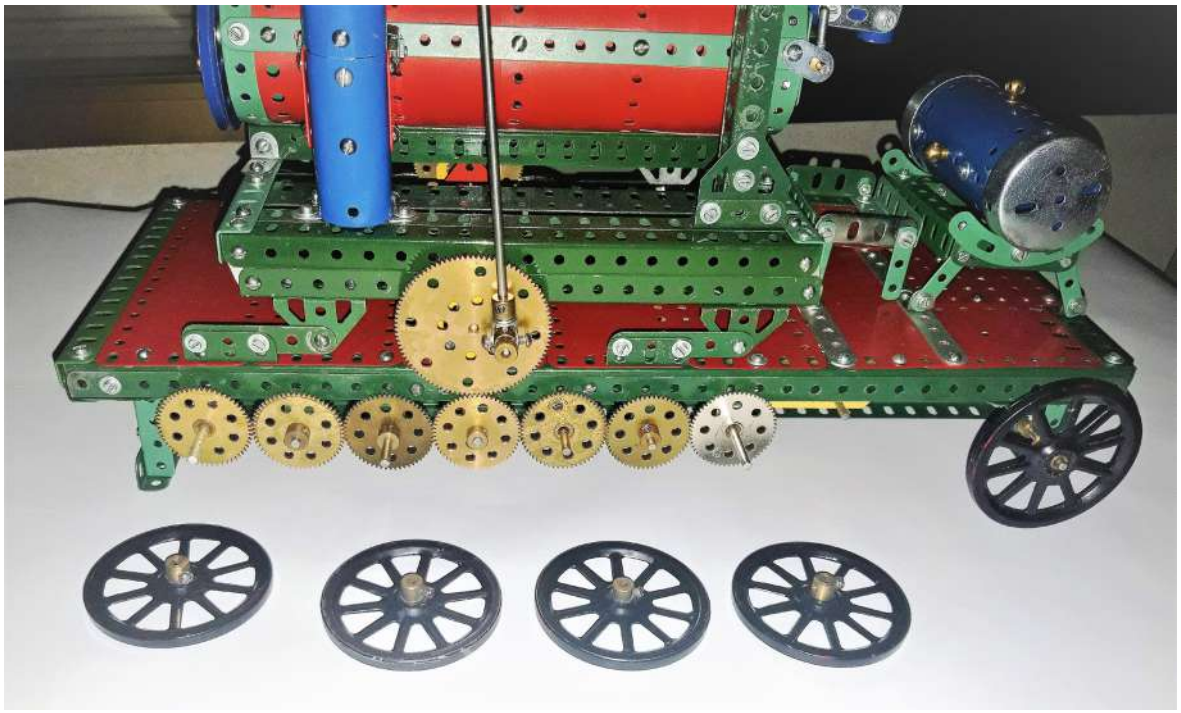


Foto nº 4. Los engranajes

En esta foto se aprecia una vez desmontadas las ruedas los 7 engranajes, son ruedas dentadas de 57 dientes, tienen que ser impares para que todas las ruedas giren en el mismo sentido. La rueda central engrana con otra rueda más grande que hace mover las distintas bielas, en un principio pensé en poner las ruedas grandes en lugar de las pequeñas, pero las dimensiones del modelo se iban de tamaño, aparte de poder tener 14 ruedas.

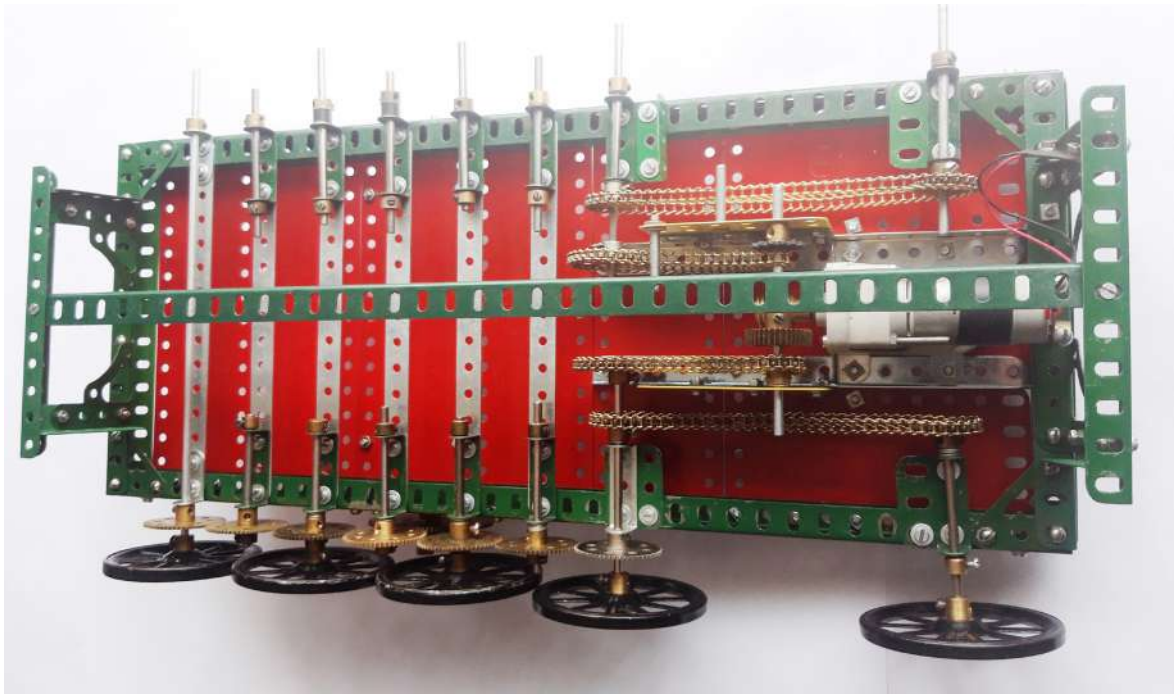


Foto nº 5. La transmisión del motor a los engranajes

Cuando empecé a pensar como trasmitía el movimiento a las ruedas dentadas me parecía sencillo, motor al primer eje y a sus correspondiente ruedas, luego a los seis restantes, pero es cuando te das cuenta de las imperfecciones de las piezas, siempre hay algún engranaje que se atascaba.

Lo primero que hice es atornillar las dos primeras ruedas y las demás por roce unas con otras se moverían, pero sin atornillar. Después decidí poner ejes con tiras dobladas de tres agujeros, con esto se consigue que el ajuste sea más perfecto. El inconveniente de esto ee que hay que poner cadena en los dos lados. Cuando hablamos de ruedas me refiero a dentadas.

El modelo al estar en alzado las dos ruedas que no llevan engranajes no se mueven, esto queda bastante raro así que he tenido que engranarlas con cadenas.

Y con esto está dicho todo de este modelo tan extraño.

Expositor de piezas Meccano de la época níquel, 1912 a 1927