Reloj de bolas de acero

“NORIA”

Por Pedro Eisman Cabado, nov 2025 – ene 2026

- GUIA DE OPERACIÓN –

1. INTRODUCCIÓN

Sin duda alguna y por distintos motivos, el “Campanilla” ha sido el reloj más complejo de los 10 modelos que he fabricado hasta finales de 2025. Por un lado, fue el primer reloj en incorporar un sistema eléctrico controlado por un Arduino y, por otro lado, su diseño mecánico no fue nada fácil. Su construcción resultó laboriosa y no exenta de problemas. Por aquel entonces yo no contaba con la ayuda de la IA para escribir el código del programa para el Arduino. Tan complejo resultó aquel proyecto que no pude elaborar una guía de operación ya que tuve que ajustar partes de ese reloj tiempo después de haberlo terminado

Tras haber diseñado y fabricado varios relojes a base de leds y a sabiendas de que para cualquier cuestión electrónica ahora sí que puedo confiar en la inteligencia artificial, he considerado necesario volver a diseñar y construir un nuevo reloj haciendo hincapié en los aspectos estéticos y mecánicos. Básicamente la idea ha sido elaborar un nuevo modelo de un reloj de bolas de acero que pudiera igualar o superar en dificultad y complejidad al “Campanilla”. Este nuevo modelo ha sido mi reloj “Noria”.

Esta guía de operación la escribo con el reloj ya terminado. No he documentado con fotografías el proceso constructivo ya que en gran parte el modelo no deja de ser un prototipo que ha seguido el principio de “prueba y error” tanto en su diseño, pero sobre todo durante su construcción. Solo ahora que lo doy por terminado y que veo con satisfacción que funciona como esperaba es cuando redacto este documento.

2. PRINCIPIOS DE DISEÑO Y DE FUNCIONAMIENTO

Hay muchas versiones de relojes de bolas de acero que pueden encontrase en internet y todas ellas comparten el mismo principio de funcionamiento: un sistema de elevación de bolas y un conjunto de rampas por donde estas circulan. Muestro algunas fotografías de algunos de estos relojes.

Casi ninguno de estos modelos tiene planos o documentación disponibles. Solo he visto un modelo con documentación detallada en una versión para impresora 3D. Lo poco accesible son fotos o videos de algunos relojes en funcionamiento. Así que el diseño y elaboración de mi reloj había que hacerlo partiendo prácticamente de cero.

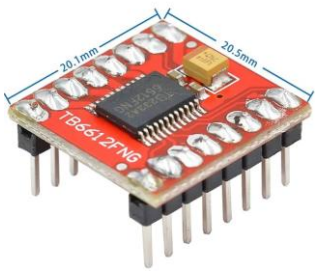
El segundo reloj que muestro en las fotos anteriores es el que me ha servido de inspiración. Del mismo solo hay disponible un vídeo con comentarios en alemán, muy poco útiles para mí. Pero aun no sabiendo cómo funciona ese reloj, su estética me gustó mucho ya que se aleja de las típicas versiones con rampas que se balancean. Y es ese modelo el punto de partida de mi reloj "Noria". Eso sí, aunque desconozco su funcionamiento, al menos lo que sí que sabía es que el mío daría las "campanadas". La gran mayoría de estos relojes de bolas de acero no disponen de un sistema de sonería.

Antes de adquirir una licencia de Fusion 360 estuve varios días dándole vueltas a la cabeza a cómo debía funcionar el reloj, empezando por el sistema de elevación. La mayor parte de los relojes que veo en internet llevan un sistema de elevación "continuo", que posiciona una bola en la parte superior del reloj cada minuto. No es muy diferente a cómo opera mi reloj "Campanilla", que funciona con dos motores paso a paso controlados por un Arduino. El código maneja el tiempo para que los ciclos de 60 segundos no tengan desviaciones sensibles. Pero el "Campanilla" solo gestiona bolas para la sonería, 12 bolas de 11 mm de diámetro con muy poco peso. Y a pesar de su diminuto tamaño dispuse de dos motores para asegurarme de que el par motor fuese el suficiente para elevar tantas bolas.

Mi nuevo reloj debía elevar no solo las bolas de la sonería sino también las bolas para marcar las horas. Son muchas bolas a la vez. Consulté con la IA si los motores paso a paso 28BYJ-48 que tengo en el "Campanilla" o en el reloj "Aire" podrían hacer ese trabajo y la respuesta fue clara: "no dan el par motor necesario". La alternativa entonces era adquirir un motor paso a paso más potente. Pero aun así había un problema que debía resolver: a las 12:00 el reloj debía haber subido 12 bolas para las campanadas. ¿Cómo llevar las bolas de las campanadas y las bolas de los minutos a la vez? ¿Podría subir 13 bolas en menos de un minuto? Para ello el motor no debería ser solo potente sino también rápido. La cuestión era compleja. Y con ese diseño y sin contar con un RTC todos los tiempos se debían controlar en el código del programa. Demasiada complejidad para que el reloj fuese preciso. Calibrar los tiempos en el "Campanilla" me llevó varios días.

Fue entonces cuando revisando el video del reloj alemán me percaté de que la bola que subía cada minuto se paraba en un punto de la rampa superior y que en un momento determinado dicha bola era liberada. Eso sin duda lo tenía que accionar un servomotor, pensé. En ese momento se hizo la luz ¡Eureka! Podría manejar tiempos con un RTC y accionar servos para realizar diferentes acciones. Por supuesto necesitaba interruptores o sensores ópticos para detectar las bolas.

No me llevó mucho tiempo diseñar un sistema de regulación automática que pudiera funcionar. Y ya no necesitaba un motor paso a paso. Podría servirme uno de los motores que utilicé en el "Swingtime". Lo consulté con ChatGPT y este me confirmo que ese motor efectivamente me daría el par motor suficiente. Mejor incluso que un motor paso a paso. Eso sí, necesitaba añadirle un controlador.



Me decanté por utilizar interruptores de final de carrera y no por otro tipo de sensores ópticos, al menos me pareció lo más sencillo. Pero para accionar un interruptor se necesita peso y una bola de 11 mm a buen seguro que era incapaz de hacerlo. Por otro lado, a mayor tamaño de bola mayor tamaño del reloj. Tras varias iteraciones con ChatGPT me decanté finalmente por unos interruptores SS-5GL y por bolas de acero de 18 mm de diámetro.

¡Que decepción cuando me llegaron las bolas! A pesar de ser grandes no tenían el peso suficiente para accionar el interruptor. Pero eso ya lo tenía previsto…Habría que echar mano de Arquimides y alargar las pletinas de los interruptores para aumentar el efecto palanca. Al menos así lo decidí en aquel momento, confiando que esa solución daría resultado más adelante.

Una vez que había resuelto el sistema de gestión y control del reloj, era el momento de crear un código y testarlo en Wowki, paso fundamental antes de adquirir una licencia de Fusion 360 y empezar a dibujar los planos.

3. EL CIRCUITO Y SU PROGRAMA

Voy a describir en detalle el sistema que diseñé y las instrucciones que le di a ChatGPT para que me escribiera el código del programa para el Arduino Uno:

El motor gira continuamente salvo que se presione un pulsador y este se mantenga pulsado. Cuando el pulsador se libera inmediatamente el motor vuelve a ponerse en marcha. La idea era parar una bola justo encima del pulsador. La bola se pararía delante de un elemento que sería movido por un servomotor. Ese servo motor se accionaria cada minuto. El tiempo lo controlaría un RTC y no el Arduino con delays tal como hace el “Campanilla”. Así pues, el motor movería una rueda que sube bolas continuamente. El motor se mueve siempre que el pulsador no este presionado y cada minuto el RTC ordena al servomotor que libere la bola que está presionando con su peso el pulsador. Este simple y sencillo sistema es suficiente para que las bolas se gestionen correctamente en el reloj. Las rampas hacen el resto.

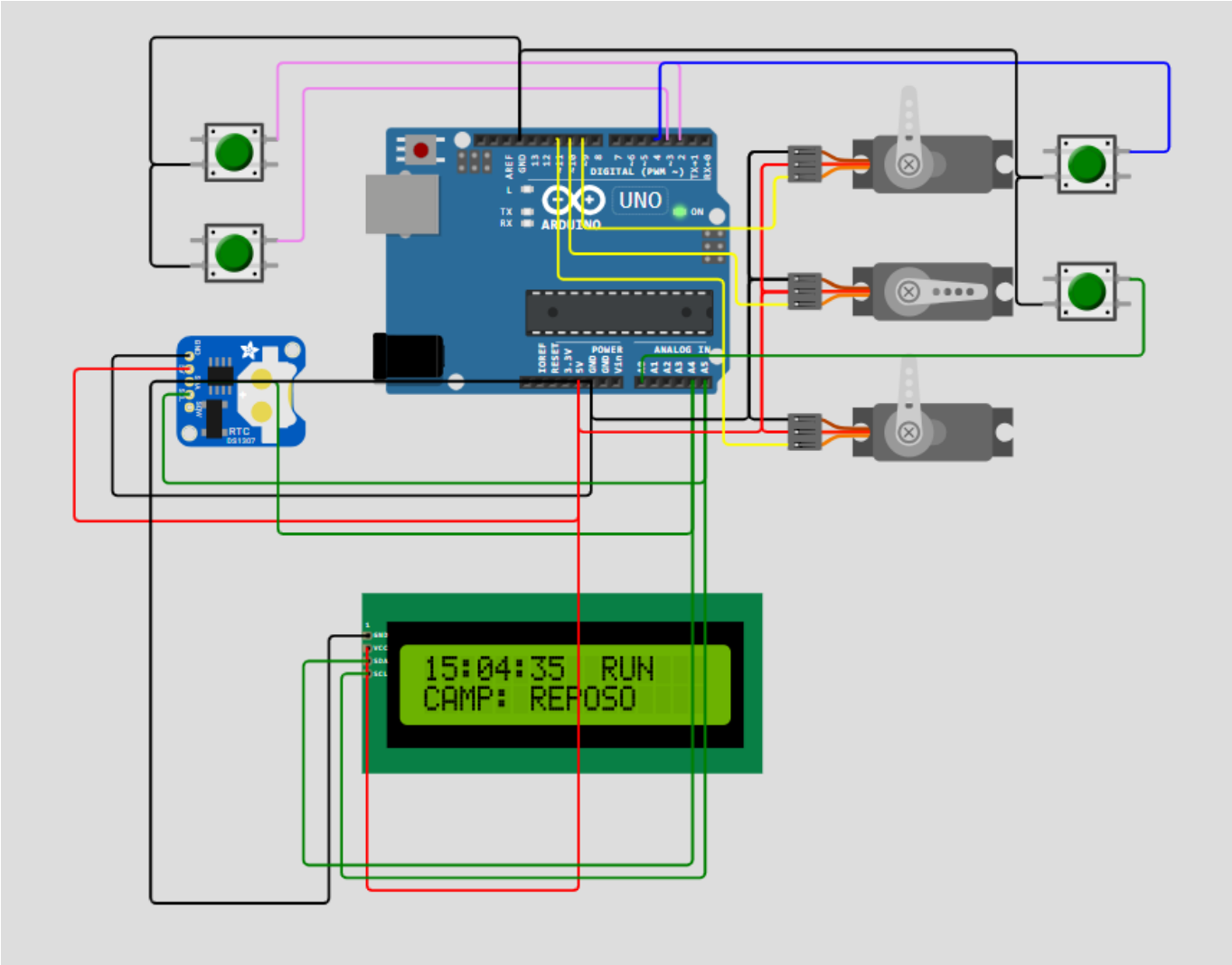
Ahora bien, faltaban las campanadas para las cuales pensé en disponer de rampas adicionales. Y no me valía la solución de usar las bolas de los minutos para dar unos golpes en una campana tal como hacen otros relojes. En cada hora había que dar las campanadas que correspondiesen. Solo necesitaba resolver como llevar las bolas a esas rampas diferenciadas, contarlas, pararlas y liberarlas en el momento oportuno.

Esta parte la resolví de la siguiente forma: en todas las horas xx:30, es decir en el minuto 30 de cada hora, un “flipper” segrega las bolas que sube el motor y las obliga a circular por una rampa diferente. En esa rampa hay otro pulsador que cuenta las bolas que pasan. Si son las 09:30 entonces el sistema debe contar 10 bolas (que son las campanadas que van a dar próximamente) y una vez estas han pasado el “Flipper” vuelve a su posición original.

Aquí se entiende por qué el motor debía girar rápido. En el peor de los casos debían cargarse 12 bolas en menos de un minuto para no perder la bola del minuto siguiente. Decidí “cargar” todas las bolas de las campanadas de golpe y con antelación suficiente por simplicidad.

Las bolas se detendrían frente a otro elemento controlado por otro servomotor. Cada hora en punto el servomotor liberaría las bolas que posteriormente accionarían las campanadas.

Pero faltaba un elemento fundamental que nunca había utilizado. Necesitaba una pantalla LCD que me mostrara la hora que guardaba el RTC para saber lo que estaba calculando el programa en todo momento. Y por supuesto unos botones para poner el RTC en hora.



En el dibujo anterior se visualiza el esquema de los elementos en Wowki. Faltan el motor y su controlador, pero esos elementos no son críticos en la simulación.

Con todos estos detalles ChatGPT desarrolló un código de unas 250 líneas. Llevó su tiempo por supuesto (unos pocos días hasta que estuvo todo afinado), pero esta vez la experiencia no fue tan traumática como en ocasiones anteriores. Quizás sea porque voy aprendiendo cómo hay que manejar a la “bestia”.

Aquí muestro el código completo:

```
#include <Wire.h>
#include <RTClib.h>
#include <Servo.h>
#include <LiquidCrystal_I2C.h>

// ================================================
// PINES FÍSICOS
// ================================================
#define PIN_BTN_HORAS        2
#define PIN_BTN_MINUTOS      3
#define PIN_BTN_CUENTABOLAS  4
#define PIN_BTN_STOP         A0

#define PIN_SERVO_FLIPPER    9
#define PIN_SERVO_MOTOR      10
#define PIN_SERVO_CAMPANADAS 11

#define PIN_MOTOR_PWMA  5
#define PIN_MOTOR_AIN1  7
#define PIN_MOTOR_AIN2  8

// ================================================
// CONFIGURACIÓN
// ================================================
const uint8_t CAMPANADA_DELAY_SEG = 0;

const unsigned long DEBOUNCE_MS     = 80;
const unsigned long LCD_ROTATE_MS = 2000;

// ===== SERVO MOTOR =====
const int SERVO_MOTOR_REPOSO = 90;
const int SERVO_MOTOR_DELTA  = 50;

// ===== LCD =====
const unsigned long LCD_UPDATE_INTERVAL = 200;

// ================================================
// OBJETOS
// ================================================
RTC_DS3231 rtc;
LiquidCrystal_I2C lcd(0x27, 16, 2);

Servo servoFlipper;
Servo servoMotor;
Servo servoCampanadas;

// ================================================
// VARIABLES
// ================================================
enum { BTN_H, BTN_M, BTN_CB, BTN_STOP_I, BTN_TOTAL };

bool lastBtn[BTN_TOTAL];
unsigned long lastBtnTime[BTN_TOTAL];

unsigned long nowMs;
unsigned long servoMotorReturn = 0;
unsigned long servoCampReturn  = 0;
```

```
unsigned long lcdLastRotate    = 0;
unsigned long lcdUpdateMs      = 0;

int lastMinute = -1;
int lastHour   = -1;

bool flipperAbierto   = false;
bool flipperHechoHora = false;
int  bolasContadas    = 0;
int  bolasObjetivo    = 0;

bool campanadaHechaHora = false;
uint8_t lcdPage = 0;

// ================================================
// FUNCIONES
// ================================================
bool botonPulsado(uint8_t idx, uint8_t pin) {
  bool estado = digitalRead(pin);
  unsigned long ahora = millis();

  if (estado != lastBtn[idx]) {
    lastBtn[idx] = estado;
    lastBtnTime[idx] = ahora;
    return false;
  }

  if (estado == LOW && (ahora - lastBtnTime[idx]) > DEBOUNCE_MS) {
    lastBtnTime[idx] = ahora + 1;
    return true;
  }
  return false;
}

int bolasPorHora(int hora24) {
  int h = hora24 % 12;
  if (h == 0) h = 12;
  return h + 1;
}

// ================================================
// SETUP
// ================================================
void setup() {
  Wire.begin();
  rtc.begin();

  lcd.begin(16, 2);
  lcd.backlight();
  lcd.clear();

  pinMode(PIN_BTN_HORAS, INPUT_PULLUP);
  pinMode(PIN_BTN_MINUTOS, INPUT_PULLUP);
  pinMode(PIN_BTN_CUENTABOLAS, INPUT_PULLUP);
  pinMode(PIN_BTN_STOP, INPUT_PULLUP);

  for (int i = 0; i < BTN_TOTAL; i++) lastBtn[i] = HIGH;
```

```
servoFlipper.attach(PIN_SERVO_FLIPPER);
servoMotor.attach(PIN_SERVO_MOTOR);
servoCampanadas.attach(PIN_SERVO_CAMPANADAS);

servoFlipper.write(0);
servoMotor.write(SERVO_MOTOR_REPOSO);
servoCampanadas.write(0);

pinMode(PIN_MOTOR_PWMA, OUTPUT);
pinMode(PIN_MOTOR_AIN1, OUTPUT);
pinMode(PIN_MOTOR_AIN2, OUTPUT);

digitalWrite(PIN_MOTOR_AIN1, HIGH);
digitalWrite(PIN_MOTOR_AIN2, LOW);

DateTime now = rtc.now();
lastMinute = now.minute();
lastHour   = now.hour();

// ================================================
// LOOP
// ================================================
void loop() {

  DateTime now = rtc.now();
  nowMs = millis();

  // ===== AJUSTE RTC =====
  if (botonPulsado(BTN_H, PIN_BTN_HORAS)) {
    rtc.adjust(DateTime(now.year(), now.month(), now.day(),
                        (now.hour() + 1) % 24, 0, 0));
  }

  if (botonPulsado(BTN_M, PIN_BTN_MINUTOS)) {
    rtc.adjust(DateTime(now.year(), now.month(), now.day(),
                        now.hour(), (now.minute() + 1) % 60, 0));
  }

  // ===== MOTOR DC =====
  analogWrite(PIN_MOTOR_PWMA,
              digitalRead(PIN_BTN_STOP) == LOW ? 0 : 240);

  // ===== SERVO MOTOR =====
  if (now.minute() != lastMinute) {
    lastMinute = now.minute();
    servoMotor.write(SERVO_MOTOR_REPOSO - SERVO_MOTOR_DELTA);
    servoMotorReturn = nowMs + 2000;
  }

  if (servoMotorReturn && nowMs > servoMotorReturn) {
    servoMotor.write(SERVO_MOTOR_REPOSO);
    servoMotorReturn = 0;
  }

  // ===== FLIPPER =====
```

```
if (now.minute() == 30 && now.second() == 0 && !flipperHechoHora) {
  flipperHechoHora = true;
  flipperAbierto   = true;
  bolasContadas    = 0;
  bolasObjetivo    = bolasPorHora(now.hour());
  servoFlipper.write(53);
}

if (now.hour() != lastHour) {
  lastHour = now.hour();
  flipperHechoHora = false;
  campanadaHechaHora = false;
}

if (flipperAbierto && botonPulsado(BTN_CB, PIN_BTN_CUENTABOLAS)) {
  bolasContadas++;
  if (bolasContadas >= bolasObjetivo) {
    flipperAbierto = false;
    servoFlipper.write(0);
  }
}

// ===== CAMPANADAS =====
if (!campanadaHechaHora &&
    now.minute() == 0 &&
    now.second() == CAMPANADA_DELAY_SEG) {

  servoCampanadas.write(55);
  servoCampReturn = nowMs + 12000;
  campanadaHechaHora = true;
}

if (servoCampReturn && nowMs > servoCampReturn) {
  servoCampanadas.write(0);
  servoCampReturn = 0;
}

// ===== LCD PÁGINA =====
if (nowMs - lcdLastRotate > LCD_ROTATE_MS) {
  lcdLastRotate = nowMs;
  lcdPage = (lcdPage + 1) % 3;
}

// ===== LCD LÍNEA 0 =====
lcd.setCursor(0, 0);
char buf[17];
sprintf(buf, "%02d:%02d:%02d", now.hour(), now.minute(), now.second());
lcd.print(buf);

lcd.setCursor(10, 0);
lcd.print(digitalRead(PIN_BTN_STOP) == LOW ? "STOP" : "RUN ");

// ===== LCD LÍNEA 1 =====
// Actualizar LCD solo si NO estamos contando bolas
if (!flipperAbierto && nowMs - lcdUpdateMs > LCD_UPDATE_INTERVAL) {

lcd.setCursor(0, 1);
```

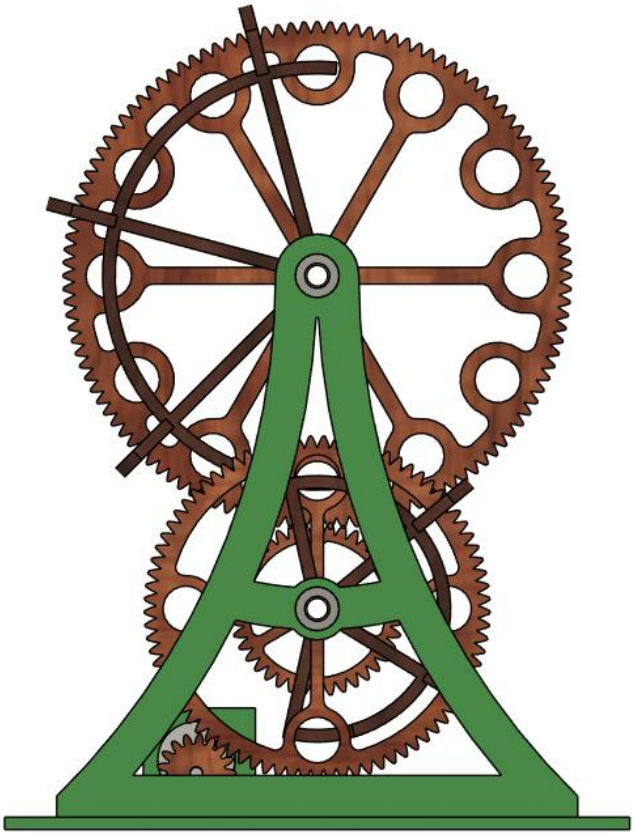
```
  lcd.print(flipperAbierto
            ? "FLIP: ABIERTO    "
            : "FLIP: CERRADO    ");
}
else if (lcdPage == 1) {
  char buf[17];
  snprintf(buf, sizeof(buf), "BOLAS %2d/%-2d      ",
           bolasContadas, bolasObjetivo);
  lcd.print(buf);
}
else {
  lcd.print(servoCampReturn
            ? "CAMP: ACTIVO     "
            : "CAMP: REPOSO     ");
}

lcdUpdateMs = nowMs;
```

4. LA CONSTRUCCION DEL RELOJ

A mediados de diciembre adquirí mi licencia de Fusion 360. Disponía de un mes para elaborar los planos, cortar, montar las piezas, hacer pruebas y rectificar en caso necesario. Parece mucho tiempo, pero este se pasa volando y además son fechas de muchas fiestas. Me dio tiempo a terminarlo, aunque el tiempo estuvo realmente justo.

Dado que no estaba muy seguro de si el proyecto saliese adelante decidí usar MDF que es un material relativamente barato. Obviamente el MDF lo cortaría con mi Sculpfun desde Lightburn, como en proyectos anteriores.

Lo primero fue diseñar el sistema de elevación. Pero ¿a cuanta altura? ¿de qué dimensiones estamos hablando? El reloj debería de tener al menos cinco rampas: tres para dar la hora, una cuarta para gestionar las campanadas y una quinta para alimentar el reloj. La inclinación de las rampas sería de 4 grados, pendiente similar al de las rampas del sistema de sonería del "Campanilla". Con estas dimensiones básicas diseñé un sistema de elevación que mide casi 40 cms de alto.



A la vista del dibujo el lector ya adivinará el origen del nombre del reloj. El sistema de elevación evoca a una Noria.

La parte más débil del "Campanilla" es su rueda de elevación que he tenido que retocar varias veces. Así que en el "Noria" no podía cometer el mismo error. En este nuevo reloj las ruedas son de 9 mm de espesor y se mueven sobre grandes rodamientos montados sobre ejes de tubo de cobre de 12 mm de diámetro. Las ruedas están sólidamente unidas a los tubos con tornillos pasantes. Toda la estructura es muy robusta.

El motor, situado en la parte inferior izquierda, gira a unas 30 revoluciones por minuto. El piñón que lleva conectado mueve el tren que eleva a las bolas de forma que 12 bolas tardan en subir poco más de 30 segundos. Todo calculado para que el sistema de sonería y el de marcación no interfiriesen y que el reloj marcase las horas correctamente.

Tenía dudas de si el motor daría el par motor necesario. Ya lo creo…el motor tiene una fuerza terrible. Me ha ocasionado más de una rotura con su fuerza al empujar alguna bola atascada. No hay quien lo pare cuando está en funcionamiento.

Este sistema de elevación lo completé en unos pocos días. Fue muy sencillo a la vez que muy gratificante su construcción. Ahora quedaba la parte más difícil: el sistema de rampas.

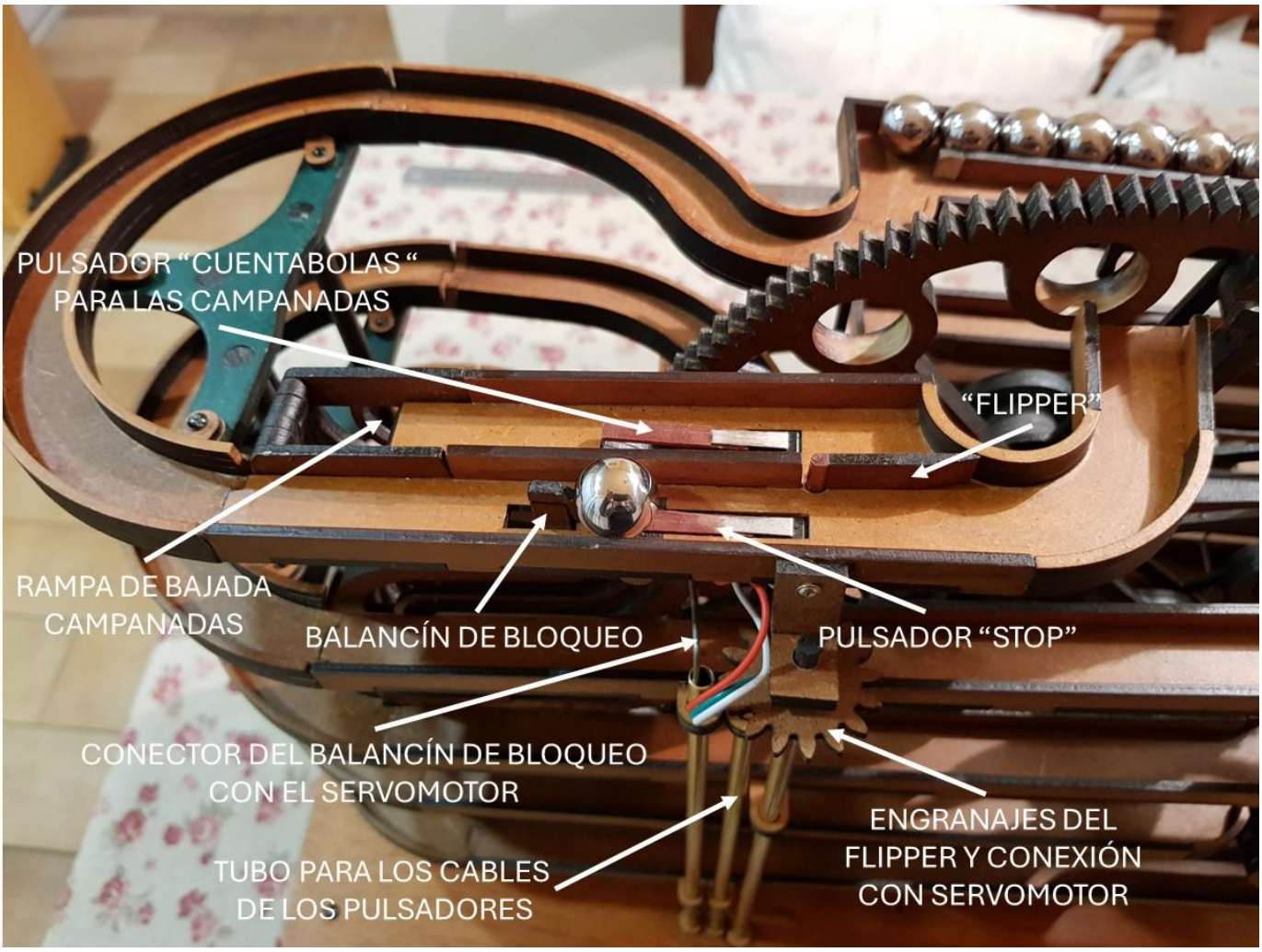
Antes de nada, explico cómo se leen las horas en el reloj.

La tercera rampa, contando de arriba hacia abajo, es la que marca las horas. En este caso son las cuatro porque hay cuatro bolas en ella. Como se aprecia en la fotografía no figura el número 12, solo caben 11 bolas en esta rampa. Las doce horas se visualizan cuando no hay ninguna bola en la rampa. Posteriormente se entenderá poque esto es así.

Las dos rampas que están por encima marcan los minutos. En este caso el reloj marca 56 minutos, 50 minutos en la rampa intermedia a los que hay que sumar los 6 minutos de la rampa superior.

Así pues, en la fotografía el reloj marca las 4:56. En este reloj no hay diferenciación AM/PM aunque se puede leer la hora en formato de 24 horas en la pantalla del LCD.

Como ya he explicado anteriormente todo comienza con una bola “bloqueada” en la parte superior del reloj. En la siguiente fotografía se muestran los elementos clave que operan en esta rampa.



La bola llega por medio del sistema de elevación a la rampa superior. Si el “Flipper” está cerrado (como lo está en la fotografía) la bola sigue recta hasta llegar al balancín de bloqueo. Al hacerlo pisa el pulsador “STOP” lo que hace que el motor del sistema de elevación se pare. Cada minuto ese balancín de bloqueo se mueve con un servomotor y deja pasar la bola. En el momento en que la bola se libera, esta deja de pisar el pulsador y hace que el sistema de elevación se mueva de nuevo. El balancín regresa a su posición anterior 2 segundos después de que haya sido liberada la bola de forma que la siguiente bola en aparecer será bloqueada, pisará el pulsador y el sistema de elevación se parará. Así una y otra vez, todo se repite cada minuto.

A las xx:30 se abre el “Flipper” y también se libera la bola que está en el balancín de bloqueo. El sistema de elevación empieza a subir bolas que ahora se dirigen a la rampa “Campanadas”. Allí un pulsador va contando las bolas que pasan. Cuando han pasado las bolas que correspondan entonces el “Flipper” se vuelve a cerrar.

Como se aprecia en la fotografía tuve que alargar los pulsadores. El “invento” ha funcionado bastante bien, aunque fue necesario ajustar con precisión la posición de cada uno de los pulsadores para dejarlos perfectamente al ras con la rampa.

Veamos ahora que pasa después de que el balancín de bloqueo libera la bola.

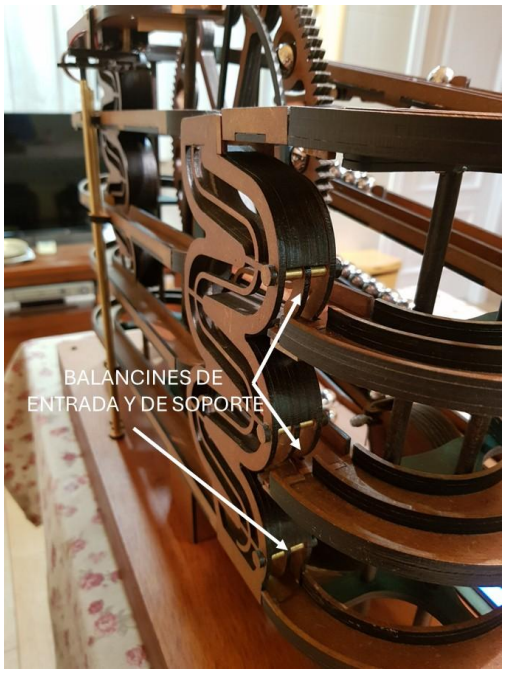
Como se aprecia en la fotografía la parte frontal de la rampa superior tiene dos carriles. Cuando la bola es liberada en la parte posterior, esta avanza y pasa por la curva "S". La forma de esta curva obliga a la bola a hacer una entrada casi perpendicular en la parte frontal de la rampa y la lleva a circular por el carril "B". En este carril la bola se encuentra con un balancín de bloqueo donde se para. Este balancín no está conectado a ningún motor, funciona con un contrapeso.

Cuando el carril "B" se llena con 9 bolas, la siguiente bola en llegar es obligada a circular por el carril "A" ya que no hay espacio disponible en el carril "B". Al avanzar la bola pisa una parte del balancín de bloqueo que hace bajar el bloqueo y libera todas las bolas del carril "B". Las bolas de este carril "B" avanzan hasta llegar a un orificio por el que bajan hasta el nivel base donde serán recogidas de nuevo por el sistema de elevación. La décima bola que fue obligada a circular por el carril principal continua hasta la rampa inferior donde entrará en el carril "B" correspondiente si este no está lleno. Esa bola añade 10 minutos al reloj. De igual forma funciona la rampa intermedia y la rampa de las horas.

Analizando el mecanismo de marcación anteriormente explicado, ahora se entenderá mejor porqué la rampa de las horas solo puede albergar 11 bolas.

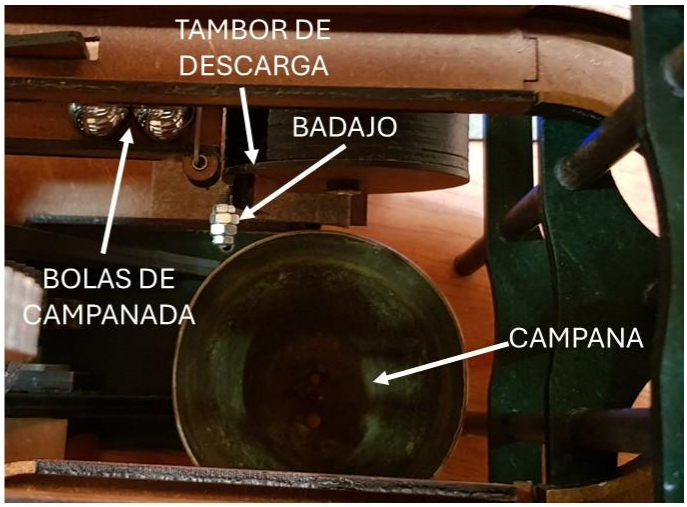
Elaborar las rampas y las curvas fue una labor complicada, aunque esta labor parezca sencilla a primera vista. Lo primero que hizo falta fue diseñar una estructura de columnas para sujetarlas. Una estructura lo suficientemente sólida y robusta pero que estéticamente no desmereciera al reloj. Todo el montaje y ajuste de estas rampas y curvas me llevó algunas semanas.

Me gustaría destacar las rampas en vertical tanto de descarga como la de las campanadas que también me dieron bastante guerra. A estas estructuras las he denominado como los “rizos”.



Los “rizos” de descarga son de particular interés dado que deben descargar las bolas de cuatro rampas. En cada una de esas rampas hay una entrada controlada por un balancín. Al entrar la bola en el rizo lateralmente el balancín se eleva y deja pasar la bola. Esa misma pieza no puede moverse hacia la rampa o exterior del rizo y así se evita que las bolas que bajan desde arriba se salgan por el orificio.

La última parte que merece la pena mencionar es la que alberga los mecanismos dispuestos en la rampa “Campanadas”.



Las bolas que bajan a esta rampa llegan a un mecanismo de retención que es accionado por un servomotor cada hora. Cuando el servomotor eleva el mecanismo de retención se permite que la bola avance hacia el tambor de descarga. Ese tambor recoge las bolas de una en una y las baja hasta la rampa de alimentación del sistema de elevación. Al entrar en el tambor la bola lo hace girar por su peso. El tambor al girar un ángulo determinado hace que la bola salga del mismo por gravedad. Dicho tambor lleva unos contrapesos que lo hacen retornar a su posición original y de esta forma puede recoger la siguiente bola.

El tambor en el giro provoca que el badajo que lleva acoplado golpee la campana. De esta manera se obtienen las “campanadas”. Mencionar como anecdótico que la campana me la facilitó mi querida esposa. La recuerda de toda la vida en casa de ius padres…así que es una campana con mucha “historia”.

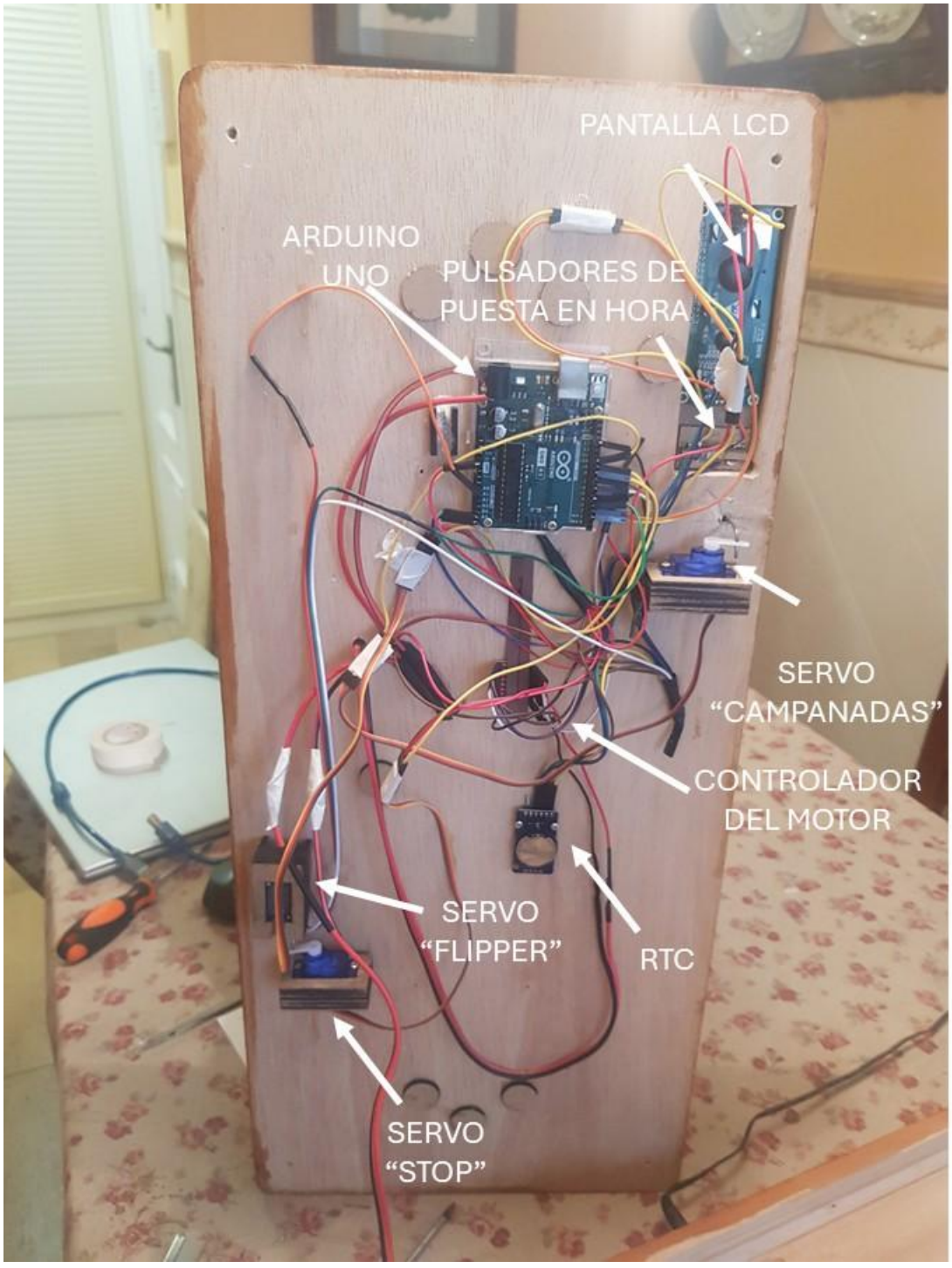
Poco más que mencionar del funcionamiento del reloj. Muestro algunas fotografías adicionales de diferentes partes del reloj para mejor referencia.

Una vez elaboradas y montadas todas las rampas tuve que construir una base donde alojar los diferentes componentes de control. Aunque en las fotos se ve todo terminado, dicha base solo la pude construir al final del proceso, cuando incluso ya me había caducado la licencia de Fusion 360 varios días atrás. Solo en ese momento pude montar los elementos mencionados.

Esa era la verdadera prueba de fuego. Todo lo que había sido teoría durante más de dos meses, ahora debía volverse realidad. Tardé unos tres días y la verdad es que fue casi “coser y cantar”.

Los elementos de control van fijados en el reverso de la base del reloj. Muestro una última foto de esta parte esencial de mi reloj.

Ahora ya puedo decir con orgullo que el “Noria” es el reloj más complejo y difícil que he diseñado y fabricado hasta la fecha. Es la nueva joya de la corona.



5. MEJORAS INCOROPRADAS EN FEBRERO DE 2026

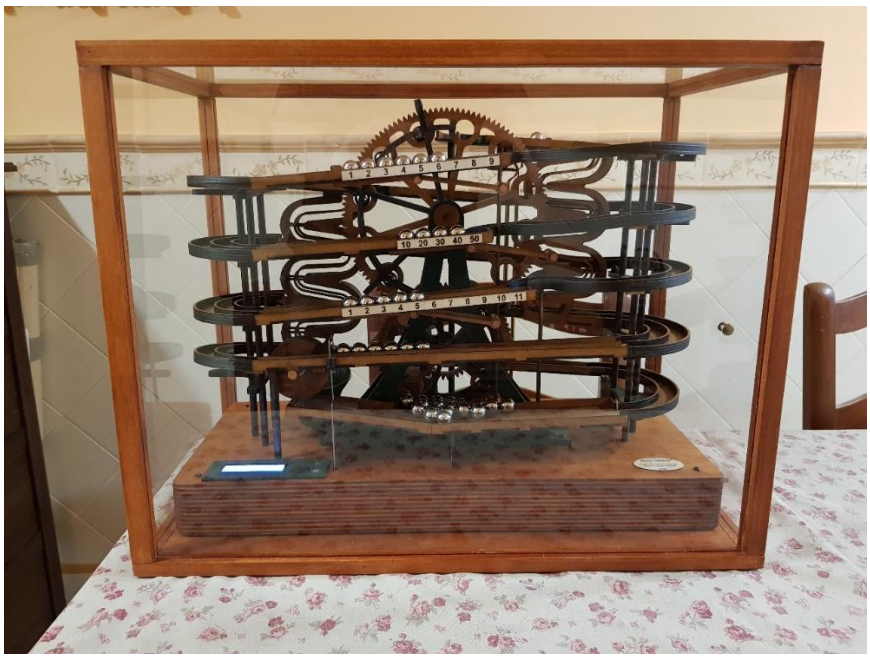
Tras algunos días de operación he observado que el reloj se ha “colgado” en distintos momentos y por razones diferentes. Para mitigar estos problemas he instalado un “step down” similar al que uso en mi reloj “Pulsar” para poder alimentar los servos directamente desde la fuente de alimentación sin tener que pasar por el Arduino. El Arduino ha pasado a ser alimentado a 5v desde el step down para aislarlo completamente del motor que es el único que se alimenta ahora a través de su controlador directamente desde la fuente de alimentación a 9v. También he añadido un condensador de 4700 µF en la salida del “step down” y otro de 1000 µF entre el VCC y el GND del Arduino.

También he puesto un condensador electrolítico de 220 µF en el LCD y varios condensadores cerámicos y resistencias en los botones de ajustes de las horas, en las señales del controlador del motor y en el propio motor para filtrar ruidos. Con tanto condensador le debería haber llamado reloj “Condensadores” más que reloj “Noria”. Bueno el caso es que todo lo anterior era necesario para filtrar señales, suprimir el ruido del motor y para reforzar el protocolo I2C.

En el código del programa he añadido unas líneas para incluir un “Watchdog” que monitorice comportamientos anómalos del Arduino Uno.

Además de todo lo anterior he incluido otro cambio más en el programa para parar el motor en caso de un problema mecánico. Si se atasca una bola o se rompe una pieza de los sujeta-bolas pueden salirse las bolas y hacer que no llegue ninguna al pulsador “STOP”. Para evitar que el motor gire continuamente he incluido un comando que lo bloquea si este ha estado girando continuamente dos minutos. He añadido un pulsador más para reiniciar el sistema en caso de que se produzca este fallo. El botón funciona también a modo de diagnóstico (pulsación corta de 1 s= diagnóstico, pulsación larga de 3s=reinicio).

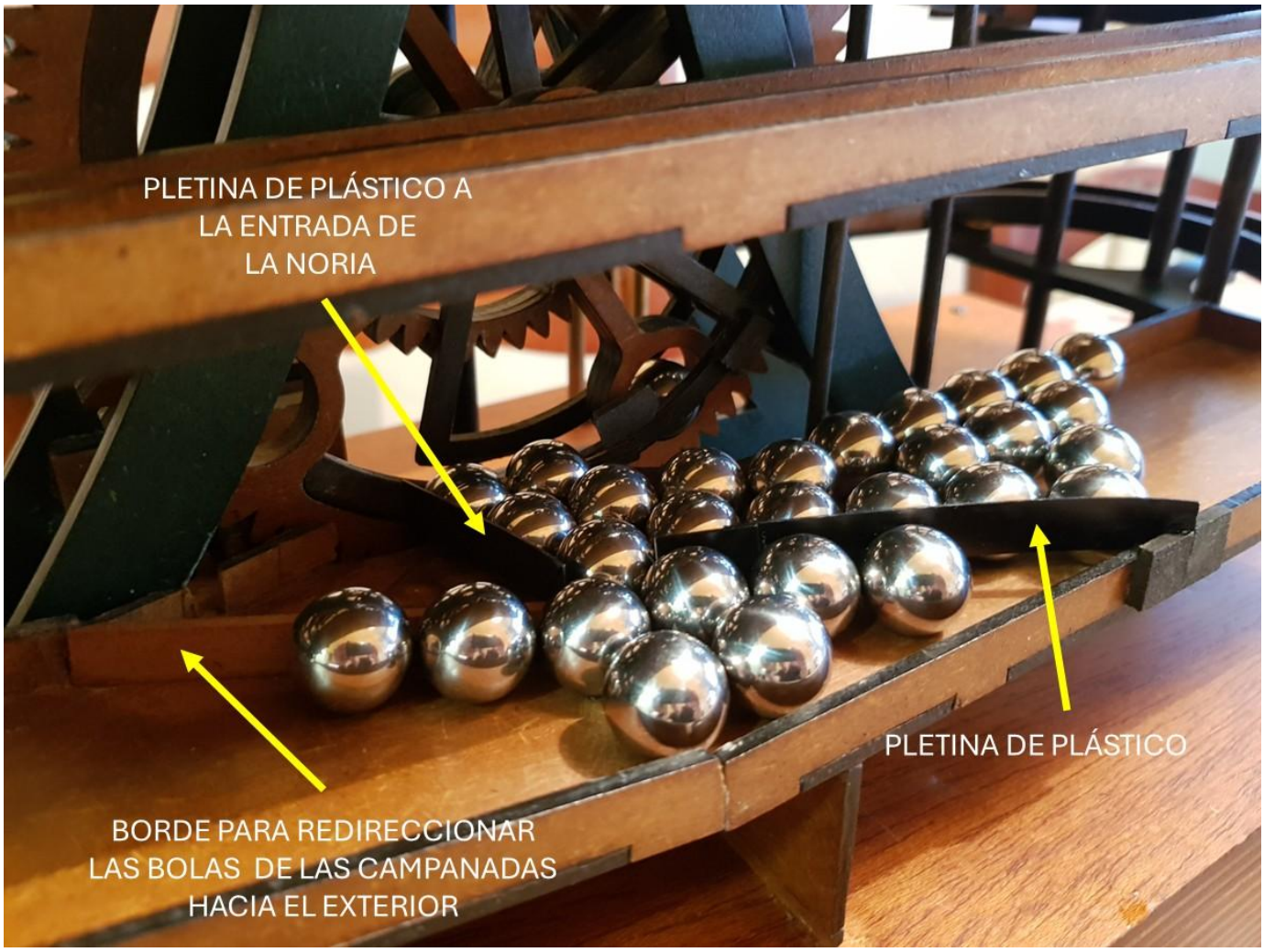
Por último, he aprovechado para fabricar una urna. La urna evita el polvo y que este frene las bolas. También mitiga considerablemente el ruido que genera el reloj.



Después de tener el reloj varios días en funcionamiento la pantalla LCD se ha quedado “a cuadros” varias veces y se ha bloqueado el programa de gestión. Así que he tenido que “doblar la rodilla” y sustituirla por una pequeña OLED Display I2C SSD1306 de 128 x 32 pixeles que ni necesita condensadores ni se pone nunca a cuadros.

Con esta nueva pantalla el reloj no se ha vuelto a colgar ni una vez en las últimas dos semanas.

Otro problema, esta vez mecánico, tenía que ver con el apelotonamiento de las bolas a la entrada de la noria que a veces formaban un arco bloqueante. Para subsanar ese fallo he forzado a las bolas de las campanadas a circular por el exterior de la bandeja y así evitar choques frontales con las bolas que bajan de las ramas. Para separar y armonizar flujos he dispuesto un par de pletinas flexibles de plástico.



Aquí digo lo mismo, el reloj lleva varios días funcionando y no se ha vuelto a generar ningún arco de bloqueo.