Concours Technic Tic Toc
FreeLUG - Automne 2021

1) Motorisation

La première décision a été de réaliser une horloge avec moteur, solution "efficace". Afin de permettre une horloge exacte et compacte, le hub City Powered Up s'impose, et donc le petit moteur auxiliaire du set Boost avec son encodeur de rotation. L'ensemble des deux permet de programmer précisément la vitesse de rotation du moteur.

2) Programmation

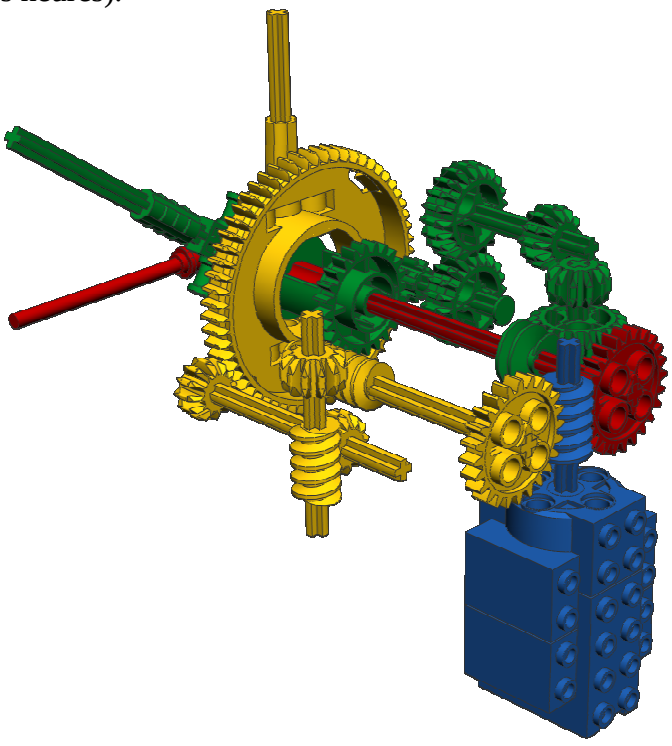
L'appli Powered Up ne peut pas pour le moment de fonctionnement autonome (un smartphone ou un tablette est requis), je me suis donc tourné vers Pybricks avec lequel on peut télécharger le code dans le hub.

3) Aiguilles concentriques

Afin de mettre un peu d'animation dans le fonctionnement, j'ai ajouté une trotteuse, même si celle-ci n'était pas requise dans les règles. La solution simple pour obtenir les 3 mouvements concentriques est une plaque tournante (heures), traversé par un boîtier de différentiel (minutes) et un axe central (trotteuse). La difficulté est de fixer (ou du moins bloquer en rotation) l'aiguille sur le différentiel, j'ai utilisé des cornes placées de part et d'autre de l'engrenage 16 dents de ce dernier.

4) Divisions de vitesse

Il n'est pas possible de faire fonctionner le moteur à une vitesse trop faible, une première réduction 1:24 (vis sans fin/engrenage 24 dents) est utilisée à la sortie du moteur. En faisant tourner le moteur à 24 tours/minute, on obtient le mouvement de la trotteuse à 1 tour/minute. On utilise en général le schéma secondes divisés par 60 pour les minutes lui même par 12 pour obtenir les heures, mais en raison de la division par 5 (12:60) au niveau de l'entraînement de la plaque tournante, j'ai opté pour deux divisions directes (1:20 / 1:3 pour les minutes, 1:12 / 1:12 / 12:60 pour les heures).



5) Cadran

Les engrenages 1/4 de couronne étaient une solution évidente pour obtenir un cadran rond de bon diamètre. Mais les marques d'heure ont posé plus de problème. J'ai tenté des éléments ronds (petites roues) qui ne posaient pas de problème d'orientation mais n'étaient pas très esthétiques. Je me suis donc orienté vers des tiles 1x2, et pour bloquer le stud en rotation (les 1/2 pin n'étaient pas une option car tournant librement) j'ai utilisé des axes avec stud et des connecteur d'axe caoutchouc compressés par des bagues.

6) Père Noël

L'idée de faire un Père Noël qui tourne la manivelle pour "faire fonctionner l'horloge" m'est venue rapidement, mais la réalisation pratique était délicate... il me fallait deux articulations pour permettre le mouvement, mais les hanches de minifig ont beaucoup trop de friction ! Heureusement les hanches à ressort des minifig des sets basket des années 90 ont fourni la solution. Pour des questions de disponibilité dans mon stock, le Père Noël a maintenant un pantalon bleu ! L'entraînement est fait directement sur la sortie moteur (24 tours/minute) pour obtenir un mouvement à vitesse correcte.

7) Ciel étoilé

Pourquoi laisser une sortie inutilisée sur le hub ? Ajouter une LED Powered Up semblait une bonne idée, mais pour éclairer quoi ? Le ciel bien sûr ! Avec l'aide de fibre optiques (sets Exo-Force) je pouvais démultiplier les étoiles dans le ciel. Le scintillement est obtenu de deux façons, variation de luminosité des LEDs par programme (mais cela affecte toutes les étoiles en même temps, pas très naturel !). D'où l'ajout d'une bielle à trous passant entre les LEDs et les fibres optiques pour obtenir des différences entre étoiles.

8) Proportion de pièces Technic

Une évaluation du nombre de pièces me donne 402 pièces au total, dont 249 sont de la catégorie Technic, soit 61.5% (ouf, c'est pas passé loin !)

9) Programme

Celui-ci est écrit en Python avec Pybricks, afin d'obtenir un fonctionnement autonome de l'horloge. Après avoir allumé le hub, une seconde pression démarre le programme. Un délai de quelques secondes permet éventuellement d'appuyer une 3ème fois sur le bouton pour lancer le mode de test dans lequel le moteur tourne 5 fois plus rapidement.

```
from pybricks.hubs import CityHub
from pybricks.pupdevices import Motor

from pybricks.pupdevices import Light

from pybricks.parameters import Port, Color, Button
from pybricks.tools import wait, StopWatch
from urandom import randint

# initialise le hub
hub=CityHub()

# initialise le moteur sur le port A
ClockMotor = Motor(Port.A)
```

```
# initialise la LED sur le port B
StarLED = Light(Port.B)

# Permet l'utilisation du bouton par programme
hub.system.set_stop_button(None)


# initialise un temporisation
watch=StopWatch()

# Allume le hub en jaune pendant qu'on attend une pression
# pour passer en mode test
hub.light.on(Color.YELLOW)
speed=1
# Attente pendant 3 secondes
while watch.time()<3000:
    if hub.button.pressed():
        # Appui sur le bouton: horloge rapide 5x!
        hub.light.on(Color.RED)
        speed=5
if speed==1:
    # Vitesse normale: hub allumé en vert
    hub.light.on(Color.GREEN)

# Autorise de nouveau l'interruption du programme par le bouton
hub.system.set_stop_button(Button.CENTER)

# redémarre la temporisation
watch.reset()
while True:
    time = watch.time()
    # le moteur doit tourner à 24 tours/seconde
    angle = time / 1000 * 6 * 24 * speed
    # asservit l'angle de rotation du moteur au temps passé
    ClockMotor.track_target(-angle)
    # varie aléatoirement la luminosité de la LED (scintillement)
    if(int(time/100) % 10 == 1):
        StarLED.on(randint(1,100))
    wait(10)
```