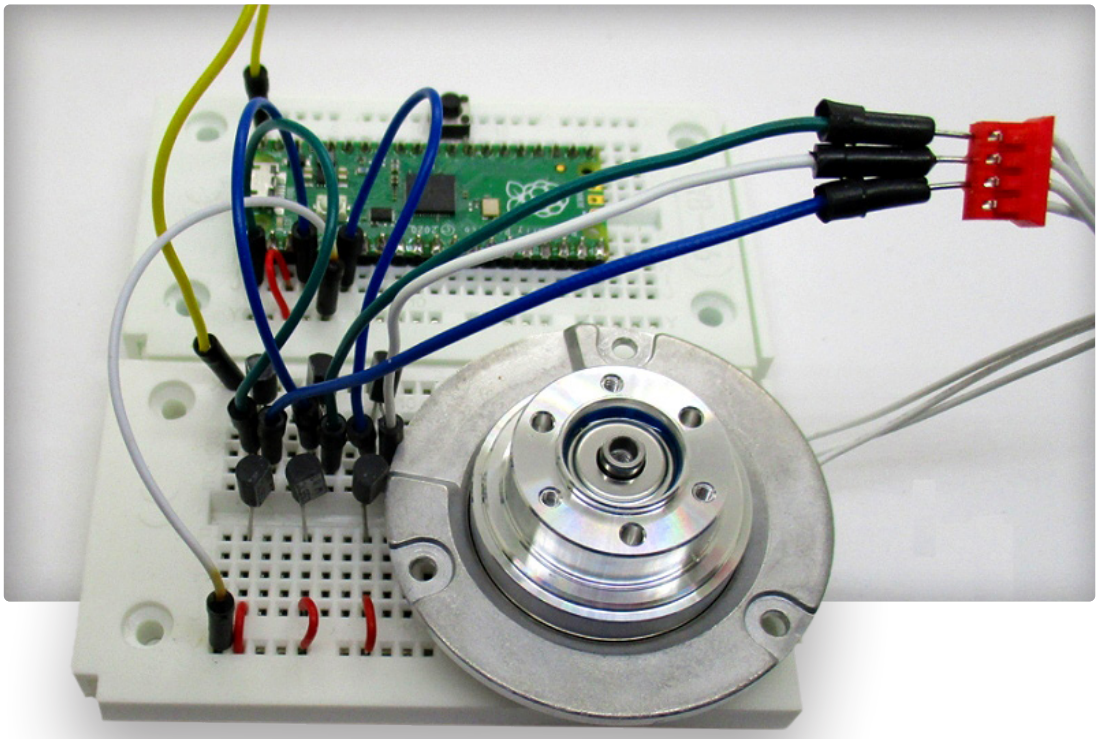
25 générateur triphasé

avec Raspberry Pi Pico



Burkhard Kainka (Allemagne)

En démontant un vieux disque dur, l'auteur a découvert un moteur triphasé. Celui-ci est bien monté et fonctionne parfaitement. C'est ainsi qu'est née l'idée de le faire tourner avec un générateur triphasé.

Le moteur étant synchrone, son démarrage doit être progressif. Pourquoi ne pas construire un petit générateur triphasé soi-même ? Un Raspberry Pi Pico, qui peut être programmé avec MicroPython, convient parfaitement. Trois signaux PWM avec des phases différentes peuvent être générés.

Le circuit de commande du moteur est illustré en **figure 1**. Il inclut trois émetteurs-suiveurs complémentaires, similaires à ceux utilisés dans les amplificateurs de puissance *push-pull*. Ici, ils doivent être contrôlés directement par les signaux PWM. L'inductance des bobines du moteur permet de réaliser un lissage du signal. Un inconvénient est la chute de tension d'environ 0,7 V par transistor, réduisant ainsi un signal PWM initial de 3,3 Vpp à 1,9 Vpp en sortie. Les bobines du moteur ayant une résistance d'environ 6 Ω, permettent un courant maximal de 300 mA, ce qui est à la limite de ce que les transistors peuvent gérer.

Logiciel

Le logiciel présenté dans le **listage 1** est disponible en téléchargement sur [1]. La table sinusoïdale utilisée consiste en 16 valeurs correspondant à une oscillation complète, ce qui permet d'isoler les quatre derniers bits du pointeur d'adresse avec `& 15`. Les valeurs de tension générées doivent être transmises aux trois sorties PWM pour maintenir des intervalles égaux, et ainsi assurer des déphasages de 120°. Cependant, comme 16 n'est pas divisible par 3, les positions 0, 5 et 10 sont utilisées (voir **figure 2**). Cette légère imprécision dans les déphasages ne nuit pas au fonctionnement du moteur.

Listage 1. Code MicroPython.

```
#DDS4.py Motortreiber
from machine import Pin, Timer, PWM
import time

pwm0 = PWM(Pin(0))
pwm0.freq(5000)
pwm1 = PWM(Pin(2))
pwm1.freq(5000)
pwm2 = PWM(Pin(4))
pwm2.freq(5000)

tim = Timer()
x =[128,176,217,245,254,245,217,176,128,80,39,11,2,11,38,80,128]
t=0
def tick(timer):
    global t
    global f
    t += f
    t1 = int(t / 100) & 15
    t2 = (t1 + 5) & 15
    t3 = (t1 + 10) & 15
    pwm0.duty_u16(256*x[t1])
    pwm1.duty_u16(256*x[t2])
    pwm2.duty_u16(256*x[t3])

tim.init(freq=1600, mode=Timer.PERIODIC,callback=tick)

for f in range(31):
    time.sleep(0.2)
    print(f, " Hz")
while 1:
    time.sleep (1)
```

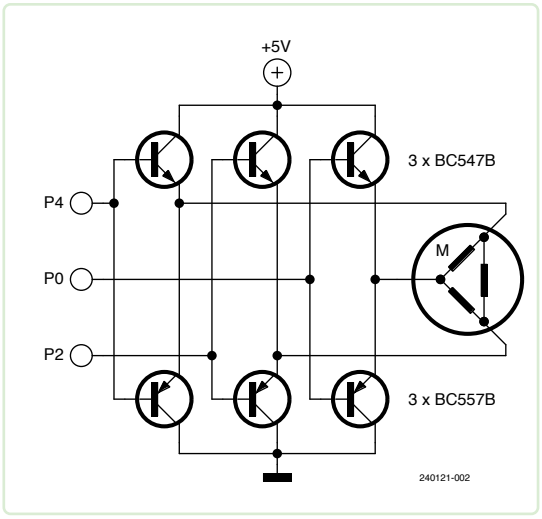


Figure 1. Le circuit est contrôlé par trois broches du Pico.

Dans la boucle, la fréquence augmente de 0 Hz à 30 Hz. À 1 Hz, le moteur présente encore des secousses visibles, mais, à partir de 5 Hz, il commence à fonctionner sans problème. La fréquence maximale atteignable est de 40 Hz, moment auquel la tension d'induction atteint un niveau équivaut à la tension générée. Pour atteindre des vitesses de rotation plus élevées, il faudrait fournir une tension plus élevée.

240121-04

Questions ou commentaires ?

Envoyez un courriel à l'auteur (b.kainka@t-online.de), ou contactez Elektor (redaction@elektor.fr).

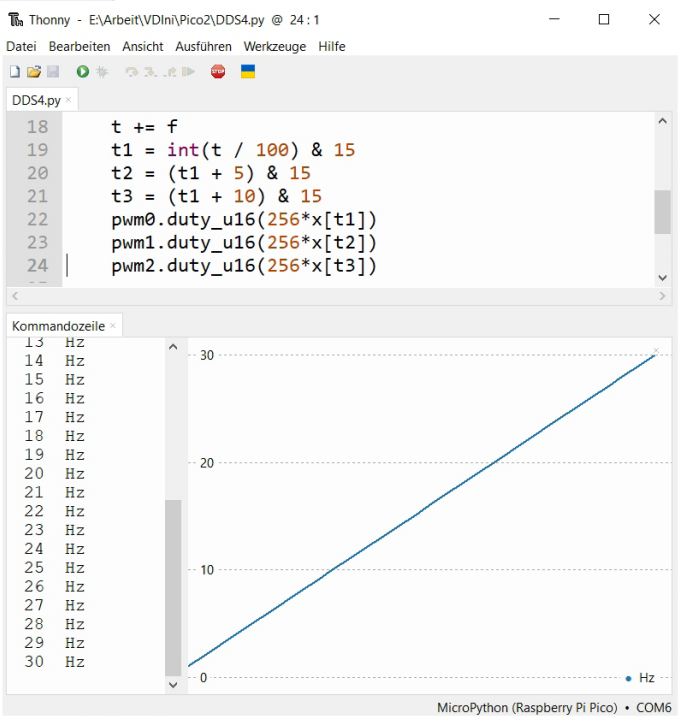


Figure 2. Dans le cadre des tests, la fréquence est incrémentée de 0 Hz à 30 Hz.

LIENS

[1] Page web de cet article : https://www.elektormagazine.fr/240121-04