06 interrupteur universel à télécommande

une nouvelle vie pour les vieilles télécommandes

Rob van Hest (Pays-Bas)

En apprenant les codes de n'importe quelle télécommande infrarouge, on peut réaliser un récepteur universel capable de commander des appareils avec de simples sorties on/off ou des impulsions. On peut presque tout commander avec les deux modèles à un et quatre canaux décrits ici.

Il y a bien longtemps, j'avais entrepris de transformer un vieux minuteur de jeu pour qu'il fonctionne avec une télécommande. Il fallait pour cela créer un récepteur capable d'effectuer quatre actions spécifiques : *démarrage*, *arrêt*, *remise à zéro* et *tour*, le tout déclenché par la télécommande. Une exigence de cette opération était la compatibilité avec presque toutes les télécommandes infrarouges. Avec l'abondance de télécommandes inutilisées dans nos foyers, ce projet offrait une solution élégante pour en réutiliser une. À l'époque, l'installation comportait un microcontrôleur PIC12F675 de Microchip Technology et une vieille télécommande utilisant le protocole NEC. Ce vieux projet m'a donné l'idée de refaire un module télécommandé similaire, avec cette fois un seul canal, en utilisant mon circuit imprimé universel à base de PIC [1]. La **figure 1** montre le résultat.

Récepteur télécommandé à un canal

Les télécommandes infrarouges fonctionnent en émettant des séquences d'impulsions codées, propres à chaque marque. L'appui sur un bouton émet un signal distinct, qu'un microcontrôleur déchiffre grâce à un phototransistor. Ce microcontrôleur commande ensuite un relais par le biais d'un transistor. Le récepteur présenté ici dispose de trois modes : *marche/arrêt*, *impulsion* et *paramétrage* – ce dernier permettant de synchroniser le récepteur avec la télécommande choisie.

- En mode *marche/arrêt*, un premier bouton active la sortie, et un second l'éteint.
- En mode *impulsion,* un bouton active la sortie qui s'éteint après une durée prédéterminée. Un deuxième bouton n'est pas nécessaire ici mais on peut enregistrer deux codes de bouton qui effectueront la même action.
- Le mode *paramétrage* permet au récepteur d'apprendre les codes de la télécommande.

Figure 1.
Le récepteur, ainsi qu'une télécommande reconvertie.

Fait remarquable, ce système ne nécessite que deux boutons de la télécommande, ce qui laisse une grande liberté pour choisir parmi les nombreuses télécommandes sous-utilisées à votre disposition. Mais comment assurer la compatibilité universelle avec les différentes télécommandes ?

Un récepteur universel

Devant la diversité de protocoles de télécommande, on a simplifié l'approche : au lieu de décoder entièrement chaque signal, le récepteur ne compte que les transitions des signaux et leur chronologie relative. Cette méthode a permis la compatibilité avec de nombreuses télécommandes. Lors des tests, diverses télécommandes - y compris des modèles RC-5 et NEC - ont fonctionné efficacement. Mais si vous utilisez une télécommande RC-5, vous constaterez que l'interrupteur ne répond pas toujours. Ceci à cause du bit de « bascule » du code RC-5, qui bascule entre 0 et 1 chaque fois que vous activez ou relâchez un bouton. C'est pourquoi il faut parfois appuyer une deuxième fois sur le bouton avant que le code soit reconnu. En mode *impulsion*, on peut résoudre ce problème en enregistrant deux fois le même code en mode *paramétrage* (voir ci-dessous).

Matériel

On peut réaliser ce projet grâce au circuit imprimé (PCB) PIC universel décrit dans un autre article [1], ou bien sur une plaque veroboard. Le schéma (**figure 2**) montre les numéros et les désignations des composants tels qu'utilisés sur le circuit imprimé PIC universel. Notez que R10 n'est pas repérée sur la sérigraphie ; il faut l'ajouter entre le nœud marqué X5 et le via le plus proche (la photo du PCB de la

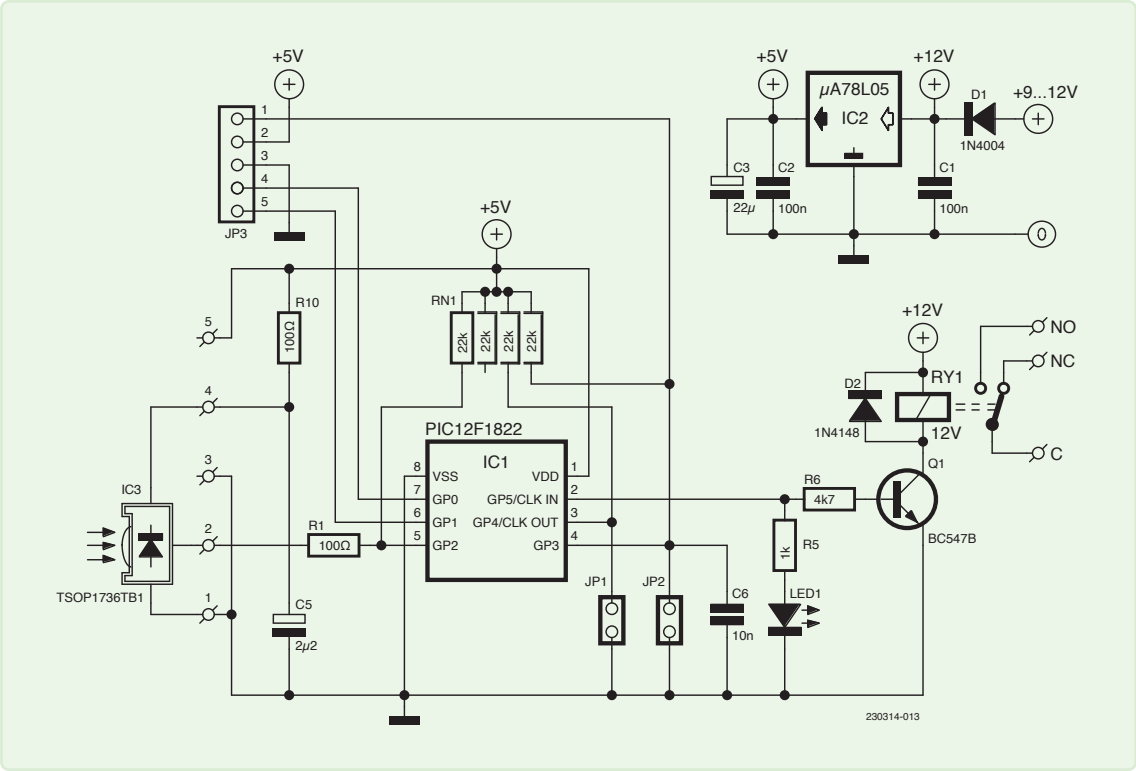

Figure 2. Schéma de la version monocanal.

figure 3 peut vous aider). Mon montage a été conçu pour fonctionner sous 12 V afin d'allumer des lampes LED de 12 V, car cette tension était facilement disponible à l'époque. Cependant, un fonctionnement sur 5 V est également possible si IC2 et D1 sont remplacés par des cavaliers et qu'un relais 5 V est utilisé. Dans ce cas, un vieux chargeur USB peut éventuellement servir d'alimentation. Voir la page du projet sur Elektor Labs [2].

À défaut d'un réseau de résistances approprié, on peut facilement remplacer RN1 par des résistances individuelles de 22 kΩ. J'ai utilisé une valeur de 100 Ω pour R1 afin de réduire le bruit, mais la valeur n'est pas critique, on peut aussi la remplacer par un cavalier de 0 Ω. Au lieu du TSOP1736, on peut utiliser d'autres récepteurs IR comme le TSOP1836, mais attention aux différences de brochage.

JP3 sert à programmer le PIC en utilisant le brochage standard du PICkit. Les broches sont, de 1 à 5, respectivement, MCLR, +5 V, GND, ICSP Data, ICSP Clock.

Logiciel

Lorsque l'état de la broche du signal IR passe de haut à bas, il déclenche une routine d'interruption. Cette même interruption est également déclenchée toutes les 4 ms lorsque le timer0 déborde. L'interruption est ainsi déclenchée dès la réception d'un signal IR. Chaque fois que l'état de la broche d'entrée change, on enregistre la valeur du timer0 (jusqu'à 48 valeurs). Si un trop grand nombre de dépassements du timer0 se produisent sans changement de l'état de la broche, ou si le nombre maximum d'échantillons est atteint, l'échantillonnage du code IR est interrompu. On détermine de cette façon le nombre de fronts et le délai entre deux fronts. En mode *paramétrage*, ces valeurs sont sauvegardées dans l'EEPROM ; dans les modes de fonctionnement normaux, on compare les valeurs reçues avec celles stockées dans l'EEPROM.

Le débordement du timer0 est également utilisé dans la routine `delay_ms()`. Les interruptions ne sont activées que pour capturer les impulsions entrantes et sont désactivées sinon, ce qui est fait dans la boucle principale. Dans le code fourni dans *remote_uni.c* [2] l'interface série ne sert que pour le débogage, pour indiquer les nombres de fronts reçus et d'échantillons conformes aux valeurs de l'EEPROM. Sans interprétation des données reçues, aucune autre information sur le contenu du code IR n'est disponible. Vous pouvez voir un extrait du code dans le **listage 1**. Le code source complet est disponible à l'adresse [2].

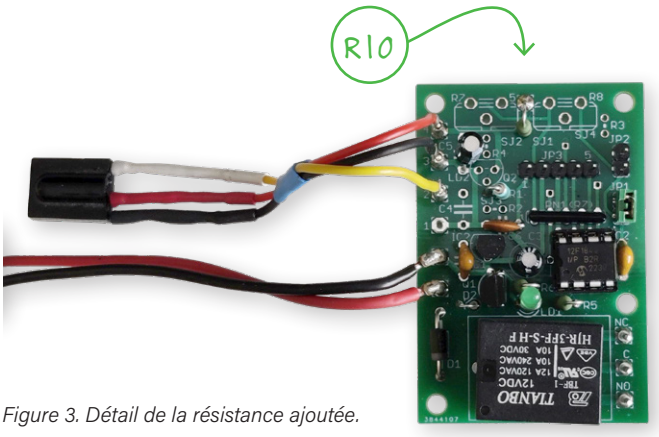

Figure 3. Détail de la résistance ajoutée.

Fonctionnement du récepteur

La configuration s'effectue avec deux cavaliers : JP1 pour le paramétrage et la sélection du mode, et JP2 pour la réinitialisation de l'appareil. L'installation de JP1 avant la mise sous tension met le récepteur en mode *marche/arrêt* ; le récepteur vérifie l'état de la broche GP4 (broche 3) au démarrage. L'absence de JP1 au démarrage met par défaut le récepteur en mode *impulsion.* Lorsque le récepteur est sous tension, l'insertion de JP1 permet d'entrer dans le mode de *paramétrage.* JP2 sert à réinitialiser l'appareil.

Appairage du récepteur avec la télécommande

Pour procéder à l'appairage, mettez le récepteur sous tension, passez en mode *paramétrage*, puis avec la télécommande, transmettez les signaux de marche (code 1) et d'arrêt (code 2), qui sont capturés et stockés. La LED s'allume et s'éteint pendant cette procédure, afin de fournir un retour visuel. À partir de ce moment, l'appareil est verrouillé jusqu'à la prochaine réinitialisation. Éteignez et rallumez l'appareil pour

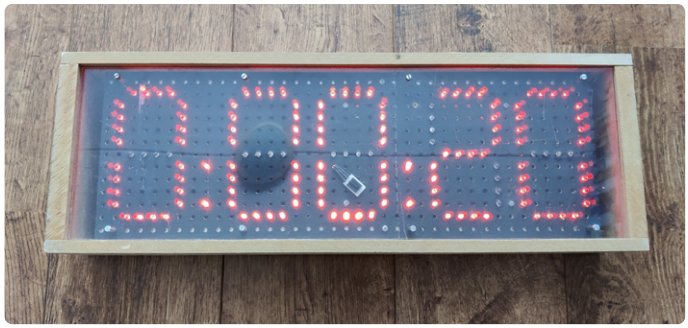

Figure 4. L'ancien minuteur de jeu, désormais télécommandé.

quitter le mode ***paramétrage***, avec ou sans JP1 selon votre préférence. En mode ***impulsion***, les deux codes IR déclenchent une brève activation de la sortie. Elle a été fixée à 500 ms ici et peut être modifiée à l'aide du paramètre `plslen`.

En mode ***marche/arrêt***, le code 1 active la sortie et le code 2 l'éteint. Les codes 1 et 2 peuvent être identiques, si bien qu'il est possible d'utiliser le même bouton pour activer et désactiver la sortie. Maintenir le bouton de la télécommande n'a aucun effet, car chaque réponse est suivie d'un délai pendant lequel aucun code IR ne peut être reçu. Il faut donc relâcher le bouton et appuyer à nouveau.

La version à quatre canaux

Afin de commander le minuteur mentionné ci-dessus (**figure 4**), j'avais besoin d'un récepteur à quatre canaux pour simuler ses quatre boutons poussoirs, ce qui nécessitait quatre sorties à impulsions distinctes. Bien qu'initialement pas destinée à une diffusion publique, voici cette version améliorée avec un meilleur microcontrôleur, un PIC12F1840 doté d'un oscillateur interne supérieur. J'ai aussi ajouté un mode marche/arrêt à bascule et une fonction d'arrêt simultané des quatre canaux. La **figure 5** présente le schéma. Les broches PAD1 à PAD4 sont connectées aux quatre boutons poussoirs du minuteur. Une LED est connectée à PAD1 pour indiquer que le système fonctionne. En option, on peut connecter d'autres LED aux quatre sorties. La **figure 6** montre l'implémentation originale du minuteur télécommandé, avec un PIC12F675 au lieu du PIC12F1840 présenté ici.

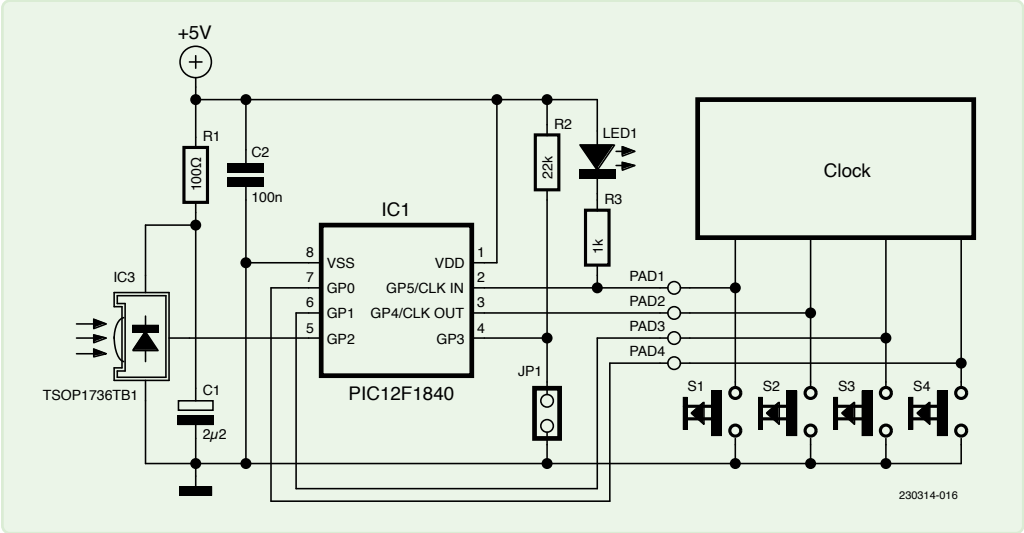

Figure 5. Schéma de la version à quatre canaux.

Variantes de sortie

Il y a deux configurations de sortie possibles pour cette version : collecteur ouvert et symétrique. Dans le premier cas, les broches de sortie du microcontrôleur sont ramenées à la masse lorsque l'état *On* est requis par l'appui sur un bouton de la télécommande. Pour l'état *Off*, les broches de sortie sont mises en haute impédance. Elles peuvent ainsi être connectées directement en parallèle avec les boutons-poussoirs du minuteur, sans relais supplémentaire. Ce mode permet également au récepteur d'être compatible avec des cartes chinoises bon marché dotées de quatre relais, par exemple pour commander quatre lampes. La seconde variante émet +5 V pour *On* et 0 V pour *Off* sur les broches de sortie du microcontrôleur, ce qui correspond au comportement du dispositif à canal unique décrit précédemment. Pour passer d'un mode à l'autre, une directive `#define KLOK` a été utilisée, qui peut être commentée si nécessaire. Si `KLOK` est défini, le récepteur fonctionne en mode collecteur ouvert. Sinon, il passe en mode symétrique. Le fichier ZIP disponible contient le fichier source C, à compiler avec CC5X. Comme tout le monde n'a pas accès à ce compilateur, des fichiers HEX ont été préparés pour les deux situations, avec et sans la directive `#define KLOK`.

Plus d'interrupteurs

Dans cette version à quatre canaux, le mode *On-Off* est géré un peu différemment. Cinq boutons de la télécommande sont désormais nécessaires. Une pression sur les boutons 1 à 4, qui envoient respectivement les codes 1 à 4, permet de basculer l'état de la sortie correspondante. En complément, une pression sur le bouton 5 désactive toutes les sorties en même temps. En mode ***impulsion***, comme dans la version monocanal, l'appui sur l'un des quatre premiers boutons active une sortie pour une durée limitée (définie par `plslen`), tandis que le bouton 5 n'est pas utilisé.

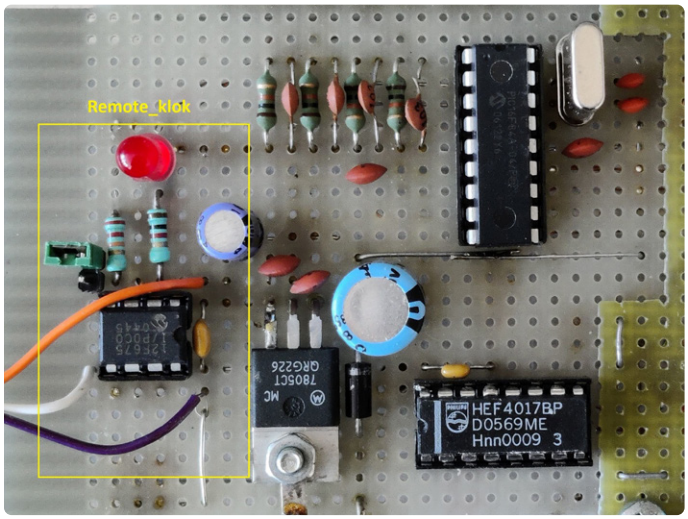

Figure 6. Implantation pratique dans le minuteur.

Listage 1. La fonction qui associe le code reçu au code stocké.

```
uns8 chkcod(uns8 num) {
     uns8 hlpvar, cntr, retval=false, memofs;
     memofs = num * maxcnt;            // starting address in EEPROM
     for (cntr=0; cntr < maxcnt; cntr++) {
         hlpvar = GetEE(memofs);
         memofs++;
         if (hlpvar == 0 || timr0[cntr] == 0)
             break;
         if (timr0[cntr]<(hlpvar-timtol) || timr0[cntr]>(hlpvar+timtol))
             break;
     }

     if (hlpvar == 0 && timr0[cntr] == 0)
         retval = true;
     return retval;
 }
```

Le processus de paramétrage reste en grande partie inchangé. Il commence par l'enregistrement du code 1 dans l'EEPROM, que le microcontrôleur confirme en activant momentanément la sortie 1. Cette procédure est répétée pour les codes 2 à 4, l'enregistrement de chaque code étant confirmé par une brève impulsion sur les sorties respectives. L'enregistrement du code 5 conclut la configuration et déclenche une dernière impulsion sur la sortie 1. Le paramétrage est alors terminé et le récepteur doit être éteint et rallumé pour passer en mode normal. Enfin, en raison de la manière dont le système est construit et programmé, les différents codes peuvent provenir de télécommandes différentes, même si celles-ci n'utilisent pas le même protocole à l'origine. Par exemple, s'il vous faut une télécommande par canal et que vous en avez plusieurs de différents modèles, le code 1 peut être envoyé à partir d'une télécommande RC-5, le code 2 à partir d'une télécommande NEC, et ainsi de suite.

VF : Denis Lafourcade — 230314-04

Questions ou commentaires ?

Envoyez un courriel à l'auteur (trainer99@ziggo.nl) ou contactez Elektor (redaction@elektor.fr).

À propos de l'auteur

Rob van Hest a étudié l'ingénierie électrique à l'université de Twente dans les années 1970, date à laquelle il publie aussi ses premiers articles pour des magazines d'électronique. À l'époque, il construit lui-même ses premiers PC, d'abord un système CP/M 8080, puis une carte Z80. Au cours de sa carrière professionnelle, M. van Hest s'est orienté vers le développement de logiciels, mais il a toujours continué à bricoler dans le domaine de l'électronique. Maintenant qu'il est à la retraite, il a encore plus de temps pour le faire !

Liste des composants

Résistances

RN1 = 22 kΩ
R1 = 100 Ω
R5 = 1 kΩ
R6 = 4,7 kΩ
R10 = 100 Ω (entre X5 et via)

Condensateurs

C1 = 100 nF
C2 = 100 nF
C3 = 22 µF
C5 = 2,2 µF
C6 = 10 nF

Divers

IC1 = PIC12F1822 (ou PIC12F1840)
IC2 = 78L05
IC3 = TSOP1736 (Brochage : X2 = signal, X3 = GND, X4 = Vcc)
LD1 = LED verte, 3 mm
Q1 = BC547B
Ry1 = Relais 12 V SPDT

Produits

- **Bert van Dam, *50 PIC Microcontroller Projects*, Elektor, 2010 (E-book)**
  www.elektor.fr/18091

LIENS

[1] R. van Hest, « Carte pour les projets à base de microcontrôleur », Elektor 7-8/2024 : http://elektormagazine.fr/230175-04
[2] Page du projet sur Elektor Labs : https://www.elektormagazine.fr/labs/universal-ir-remote-switch