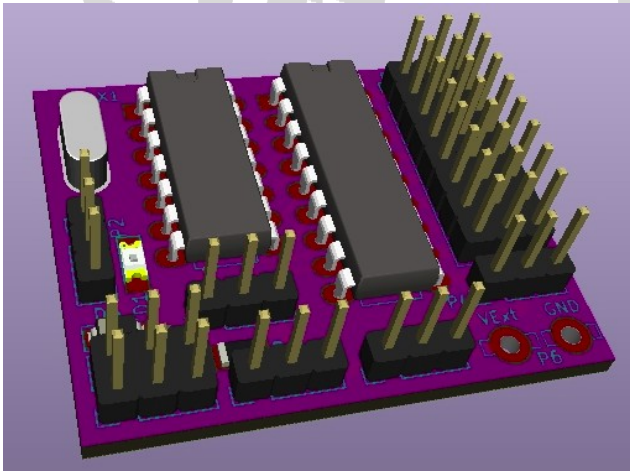


PulseSeq



**Un séquenceur
impulsionnel sur la
base du circuit
imprimé du
montage MS8-Xany
d'OpenAVRc**

Manuel Utilisateur

Séquenceur Impulsionnel PulseSeq



Copyright OpenAVRc 2022

Table des matières

1	CE DOCUMENT	3
1.1	Versions	3
1.2	Copyright	3
1.3	Avertissement	3
1.4	Contenu	3
2	PRESENTATION DE PULSESEQ	4
2.1	Vue d'ensemble	4
2.1.1	Piloté par un manche de l'émetteur et une voie du récepteur RC	4
2.1.2	Piloté par un bouton poussoir d'X-Any et une sortie de MS8-Xany	4
2.2	Spécifications du séquenceur impulsionnel PulseSeq	5
3	SCHEMA DU SEQUENCEUR IMPULSIONNEL (MEME SCHEMA QUE MS8-XANY)	7
4	REALISATION DU SEQUENCEUR PULSESEQ	8
4.1	Circuit imprimé (le même que pour MS8-Xany)	8
4.2	Montage des circuits intégrés sur support tulipe	8
4.2.1	Montage de l'ATtiny84 sur support tulipe	8
4.2.2	Montage de l'ULN2803 sur support tulipe	8
4.3	Chargement du Firmware et configuration des fusibles	9
4.3.1	Chargement du Firmware dans l'ATtiny84	9
4.3.2	Arduino UNO en programmeur ICSP sous Windows	10
4.3.3	Arduino UNO en programmeur ICSP sous Linux	11
4.3.4	Note très importante à propos de la valeur des fusibles	12
5	UTILISATION	13
5.1	Mode « salve PWM » ou mode « impulsion numérique »	13
5.2	Connexion au récepteur	13
5.2.1	Câblage des « utilisations » Tout-Ou-Rien sur les sorties	13
5.2.2	Sélection de la tension d'alimentation des sorties S1 à S8	14
5.2.3	Diodes de roue libre	14
5.2.4	Commande directe de relais avec diode intégrée à l'ULN2803	14
5.2.5	Commande de relais opto-isolés	15
5.2.6	Montage conseillé pour relais 5V opto-isolés	16
5.3	Configuration X-Any côté émetteur OpenAVRc	16
5.4	Mode avancé/Définition des séquences	17
5.4.1	Utilisation du port série de PulseSeq	17
5.4.2	Les messages de commande de PulseSeq	18
5.4.3	Exemple de configuration réelle	19

1 CE DOCUMENT

1.1 Versions

La version de ce document est la dernière listée dans la colonne **version** de la table ci-dessous :

Version	Date	Raison de l'évolution
0.1	16/01/2022	Création

1.2 Copyright

Ce document est Copyright © 2022 **OpenAVRc**.

1.3 Avertissement

L'équipe **OpenAVRc** n'est aucunement responsable des dommages qui pourraient découler de la mauvaise utilisation ou d'un éventuel dysfonctionnement de l'émetteur **OpenAVRc**, du module séquenceur impulsif **PulseSeq** et/ou des logiciels associés.

Il appartient donc à l'utilisateur final d'en mesurer, d'en assumer les risques et de respecter la législation en vigueur selon le pays d'utilisation.

1.4 Contenu

Ce document décrit la réalisation du module décodeur **PulseSeq** ainsi que le paramétrage pour son utilisation avec l'émetteur **OpenAVRc**.

PulseSeq est un module séquenceur impulsif disposant de 8 sorties « Tout-Ou-Rien ».

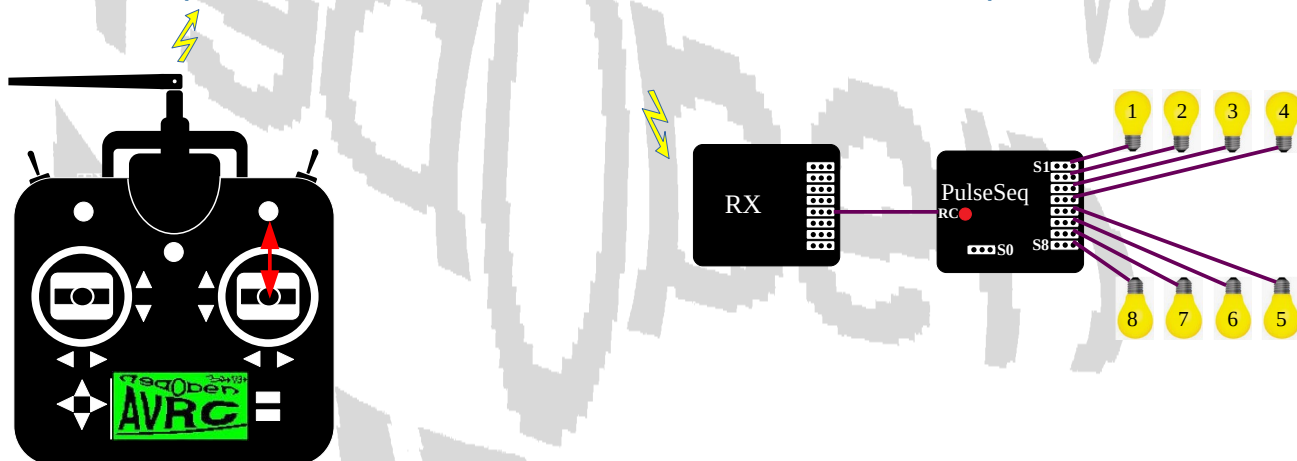
Il se commande soit :

- par une voie d'un récepteur RC (par coups de manche côté émetteur RC)
- par une sortie numérique (ex : bouton-poussoir d'un codeur/décodeur Multiswitch)

2 PRESENTATION DE PULSESEQ

2.1 Vue d'ensemble

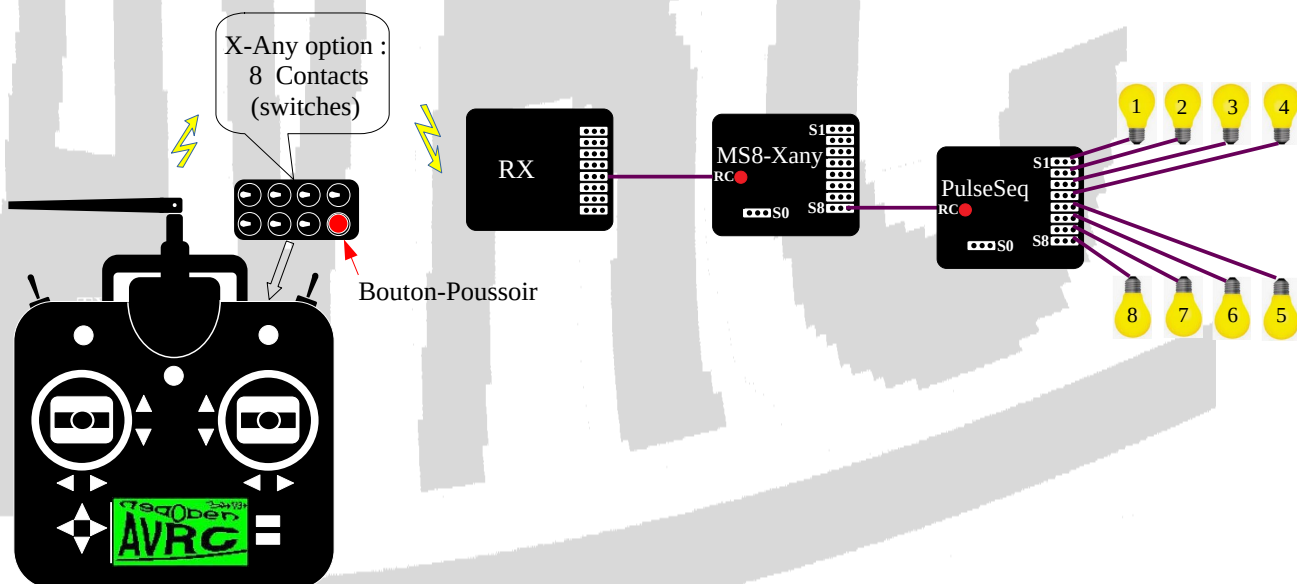
2.1.1 Piloté par un manche de l'émetteur et une voie du récepteur RC



PulseSeq peut s'utiliser avec n'importe quel ensemble RC à l'aide du manche associé à la voie du récepteur RC :

- Un coup de manche très bref (< 600 ms) fait avancer d'une étape,
- Un coup de manche bref (≥ 600 ms et < 2000 ms) fait reculer d'une étape,
- Un coup de manche long (≥ 2000 ms) fait passer à l'étape 0 (toutes les sorties désactivées).

2.1.2 Piloté par un bouton poussoir d'X-Any et une sortie de MS8-Xany



Cette solution plus ergonomique n'est possible qu'avec l'ensemble RC **OpenAVRc** qui supporte les décodeurs Multiswitches **MS8-Xany** et **MS16-Xany**. Le bouton-poussoir remplace le manche.

2.2 Spécifications du séquenceur impulsif PulseSeq

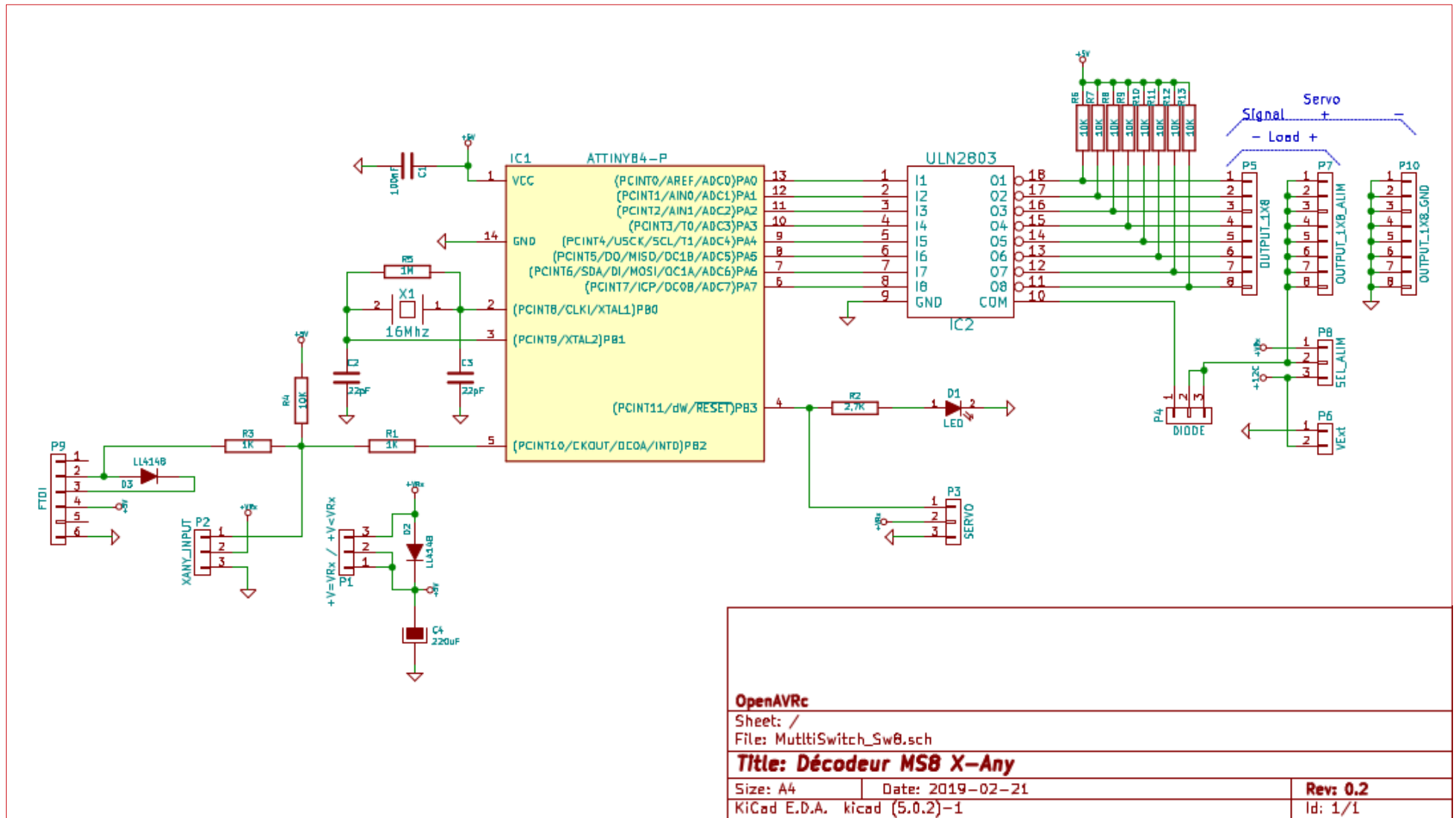
Spécification	Valeur / Caractéristique	Note
Alimentation	+3.3V à +6.6V	Mettre le cavalier « tension récepteur » sur « = » ou « ↓ » selon la valeur de la tension fournie par le récepteur
Commande	- en PWM : piloté par une sortie voie d'un récepteur Rcs - en numérique : piloté par une sortie à collecteur ouvert d'un décodeur Multiswitch ou tout simplement par un bouton-poussoir	
8 sorties Tout-Ou-Rien	- Commandées en Tout-Ou-Rien par sortie transistorisée	Sorties configurées en mode « numérique » (MultiSwitch)
Alimentation des sorties	- Interne : tension du récepteur - Externe : tension externe 5 à 24V	Sélectionnable par cavalier (commun aux 8 sorties)
Diode de roue libre	Diodes de roue libre des 8 sorties au + de l'alimentation des sorties	Sélectionnable par cavalier (commun aux 8 sorties)
LED rouge perte de signal	Au bout de 1.5 secondes sans signal	Gérée uniquement en mode PWM
Failsafe	- Toutes les sorties passent à 0 en cas de perte de signal RC	Synchrone avec LED rouge perte de signal
Port série TTL	Connecteur pour câble USB/FTDI TTL	Pour configuration avancée à l'aide d'une application type « Terminal série »
Dimensions (mm)	L x l x h : 39 x 33 x 12	

V3

Page left intentionnally blank
/

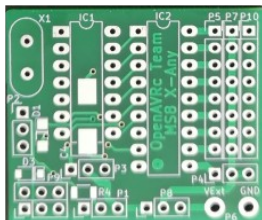
Page laissée intentionnellement blanche

3 SCHEMA DU SEQUENCEUR IMPULSIONNEL (MEME SCHEMA QUE MS8-XANY)



4 REALISATION DU SEQUENCEUR PULSESEQ

4.1 Circuit imprimé (le même que pour MS8-Xany)



4.2 Montage des circuits intégrés sur support tulipe

4.2.1 Montage de l'ATtiny84 sur support tulipe

ATTENTION :

L'**ATtiny84** doit être programmé **avant** son positionnement sur le circuit imprimé de **PulseSeq**.

C'est pourquoi, il est fortement recommandé de le monter sur un support tulipe de 14 points :



Il est également possible d'utiliser 2 portions (de 7 points) de barrette tulipe sécable :



Dans les 2 cas, bien vérifier l'orientation de l'**ATtiny84** avant de mettre sous tension **PulseSeq**.

4.2.2 Montage de l'ULN2803 sur support tulipe

En cas d'erreur de câblage sur les sorties, il est possible de détruire l'**ULN2803**.

Afin de faciliter son éventuel remplacement, il est fortement recommandé de le monter sur un support tulipe de 18 points :



Il est également possible d'utiliser 2 portions (de 9 points) de barrette tulipe sécable :



Dans les 2 cas, bien vérifier l'orientation de l'**ULN2803** avant de mettre sous tension **PulseSeq**.

4.3 Chargement du Firmware et configuration des fusibles

La programmation du microcontrôleur **ATtiny84** se fait en 2 phases (dans cet ordre) :

1. Chargement du Firmware
2. Configuration des fusibles

4.3.1 Chargement du Firmware dans l'ATtiny84

ATTENTION :

Le connecteur P9 (noté FTDI) **n'est pas** un connecteur de **programmation**, mais un connecteur de **paramétrage** de **PulseSeq** (Voir §Mode avancé/Définition des séquences)

Le firmware :

- **PulseSeq_Vx_y.hex** : **PulseSeq** est alors piloté par une sortie **PWM** du récepteur ou par une sortie numérique (Sortie à collecteur ouvert du décodeur **MS8-Xany** ou simple bouton-poussoir)

A l'aide d'un **programmeur ICSP** pour microcontrôleur AVR, charger le fichier **PulseSeq_Vx_y.hex** dans l'**ATtiny84**.

Pour cela nous allons utiliser un **arduino UNO** configuré en **programmeur ICSP** pour charger l'**ATtiny84** avec le **fichier HEX** et les bonnes **valeurs de fusibles** : voir le paragraphe suivant.

Toute la phase de programmation (**chargement du Firmware + configuration des fusibles**) se fait **avant** de placer l'**ATtiny84** sur son support tulipe sur le circuit imprimé de **PulseSeq**.

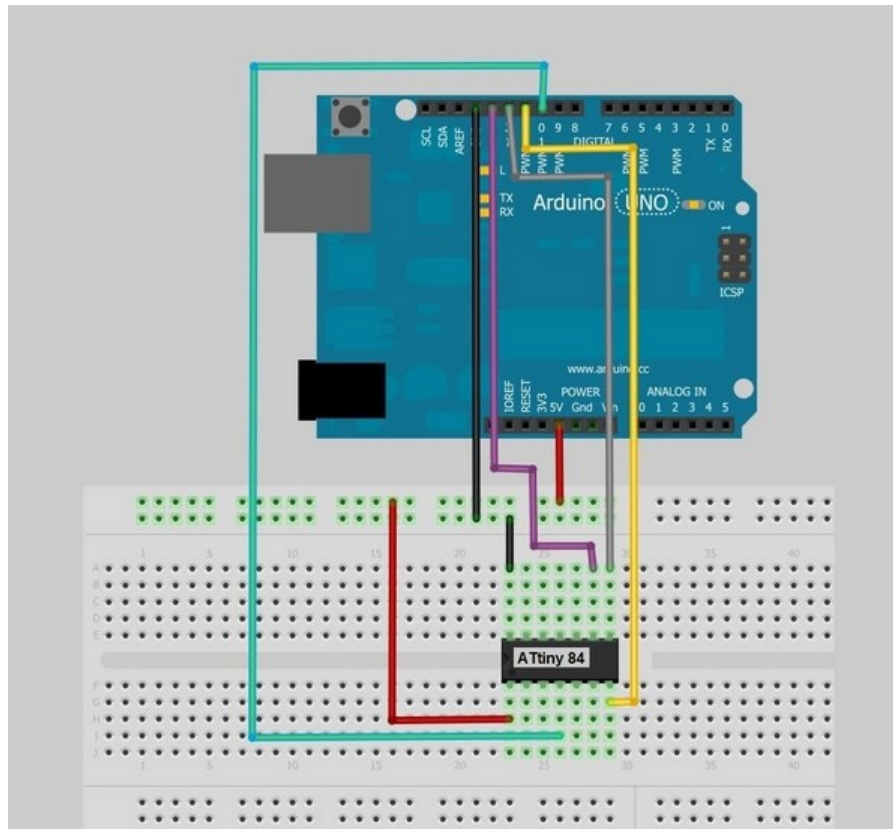
4.3.2 Arduino UNO en programmeur ICSP sous Windows

Connecter votre **arduino UNO** et à l'aide de l'**IDE arduino**, charger le sketch **ArduinoISP** par :

Fichiers → Exemples → 11.ArduinoISP

Cliquer sur **téléverser** : votre **arduino UNO** est désormais un **programmeur ICSP**.

Réaliser le câblage suivant pour programmer l'**ATtiny84** à l'aide de l'**arduino UNO** :



Copier le sous-répertoire **Windows** du répertoire [PROG](#) sur votre PC. Ouvrir une **console DOS** (invite de commande) et aller dans ce répertoire **Windows**.

Si nécessaire, éditer le fichier **prog_ms8.bat** à l'aide d'un éditeur de texte afin de redéfinir :

1. le bon **port série** utilisé par l'**arduino UNO** : **COMx**
2. le **fichier HEX** que vous désirez charger : **PulseSeq_Vx.y._HEX**

```
REM Adjust below the Serial Port used by your Arduino UNO (Look at Tools->Port, or Outils->Port in the Arduino IDE)
SET SERIAL_PORT = COM4
```

```
REM Define here the HEX file you want to load (this HEX file shall be in the current directory)
SET HEX_FILE = MS8_Xany_PWM_V0_4._HEX
```

Dans la **console DOS**, lancer la commande **prog_ms8.bat** et suivre les instructions.

Le **fichier HEX** sera chargé et les **fusibles** seront automatiquement programmés.

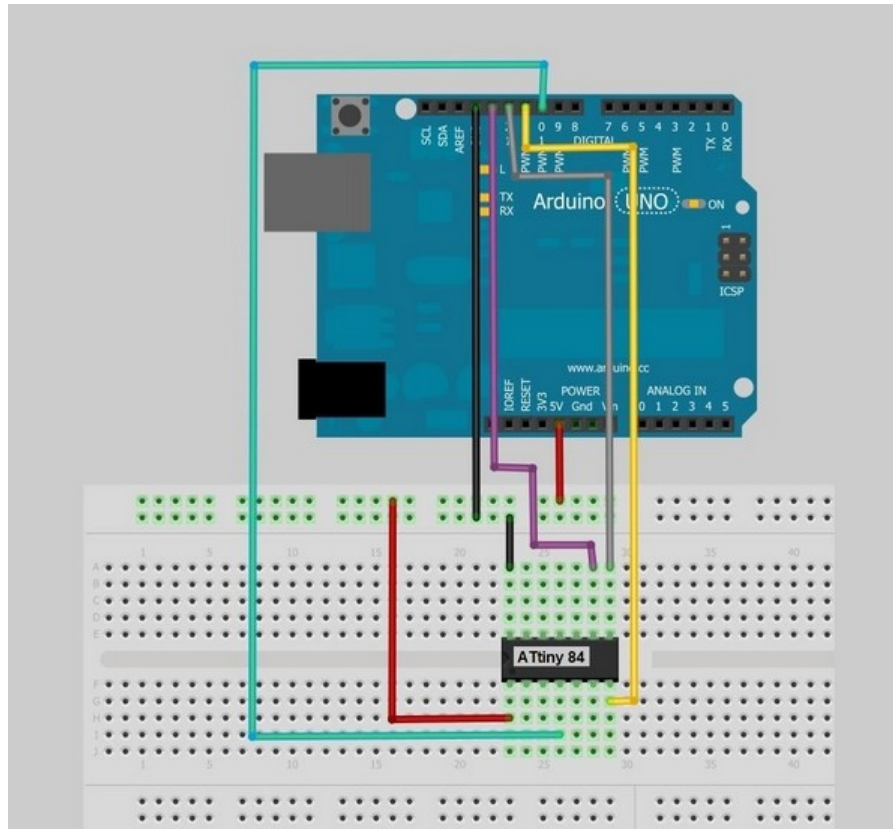
4.3.3 Arduino UNO en programmeur ICSP sous Linux

Connecter votre **arduino UNO** et à l'aide de l'**IDE arduino**, charger le sketch **ArduinoISP** par :

Fichiers → Exemples → 11.ArduinoISP

Cliquer sur **téléverser** : votre **arduino UNO** est désormais un **programmeur ICSP**.

Réaliser le câblage suivant pour programmer l'**ATTiny84** à l'aide de l'**arduino UNO** :



Copier le sous-répertoire **Linux** du répertoire [PROG](#) sur votre PC. Ouvrir une **console Linux** et aller dans ce répertoire **Linux**.

Si nécessaire, éditer le fichier **prog_ms8.sh** à l'aide d'un éditeur de texte afin de redéfinir :

3. le bon **port série** utilisé par l'**arduino UNO** : par exemple, **/dev/ttyACM0**
4. le **fichier HEX** que vous désirez charger : **PulseSeq_Vx.y._HEX**

```
# Adjust below the Serial Port used by your Arduino UNO (Look at Tools->Port, or Outils->Port in the Arduino IDE)
SERIAL_PORT=/dev/ttyACM0

# Define here the HEX file you want to load (this HEX file shall be in the current directory)
HEX_FILE=MS8_Xany_PWM_V0_4._HEX
```

Dans la **console Linux**, lancer la commande **./prog_ms8.sh** et suivre les instructions.

Le **fichier HEX** sera chargé et les **fusibles** seront automatiquement programmés.

4.3.4 Note très importante à propos de la valeur des fusibles

Une fois l'**ATtiny84** programmé (avec ses fusibles), il n'est plus possible de reprogrammer directement l'**ATtiny84** avec un programmeur **ICSP** puisque celui-ci a besoin de la fonction **Reset** de l'**ATtiny84**.

En effet, le programme **PulseSeq** a besoin d'utiliser la broche **Reset** mais **configurée en sortie** pour piloter la led « Signal Lost ».

Si, pour x raison, vous devez reprogrammer votre **ATtiny84**, il sera nécessaire de restaurer la fonction **Reset** de la broche **Reset**.

Pour cela il est nécessaire d'appliquer un **signal calibré de 12V** sur la broche **Reset**. Il faut alors utiliser un petit boîtier de type « **Fuse Resetter** ».

Après cette manipulation, l'**ATtiny84** redevient programmable à l'aide d'un **programmeur ICSP**.

5 UTILISATION

5.1 Mode « salve PWM » ou mode « impulsion numérique »

Après chargement du Firmware **PulseSeq**, au premier démarrage, le mode par défaut est « salve PWM » (PWM.BURST), c'est-à-dire que **PulseSeq** se pilote connecté à une voie d'un récepteur RC.

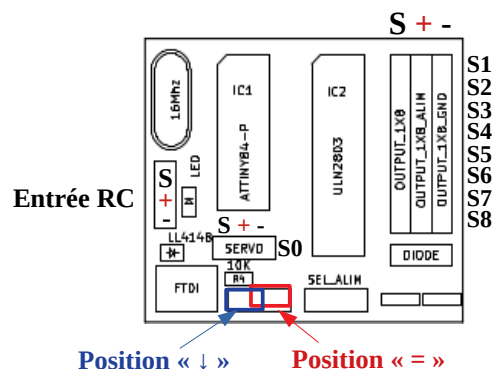
Pour changer le mode de commande en « impulsion numérique » (DIGITAL.PULSE), il est nécessaire de passer par le lien série (Cf §Mode avancé/Définition des séquences).

5.2 Connexion au récepteur

Avant de connecter **PulseSeq** au récepteur, il est impératif de mesurer la tension fournie par le récepteur.

Si la tension disponible entre les broches – et + du connecteur 3 points de la voie utilisée est :

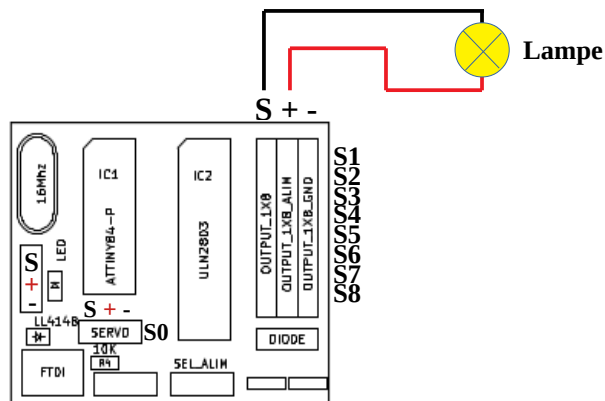
1. Inférieure à 5.7V, placer le cavalier « tension récepteur » sur « = »
2. Supérieure à 5.7V, placer le cavalier « tension récepteur » sur « ↓ »



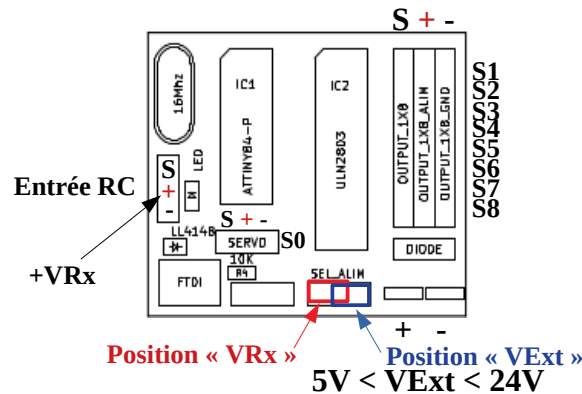
5.2.1 Câblage des « utilisations » Tout-Ou-Rien sur les sorties

Le **PulseSeq** s'utilise alors comme un module MultiSwitch standard :

- Les « utilisations » (ex : une lampe) se branchent sur les sorties **S1** à **S8** entre les broches « S » et « + », la rangée 8 points de « - » en bord de carte n'est pas utilisée dans ce cas.



5.2.2 Sélection de la tension d'alimentation des sorties S1 à S8

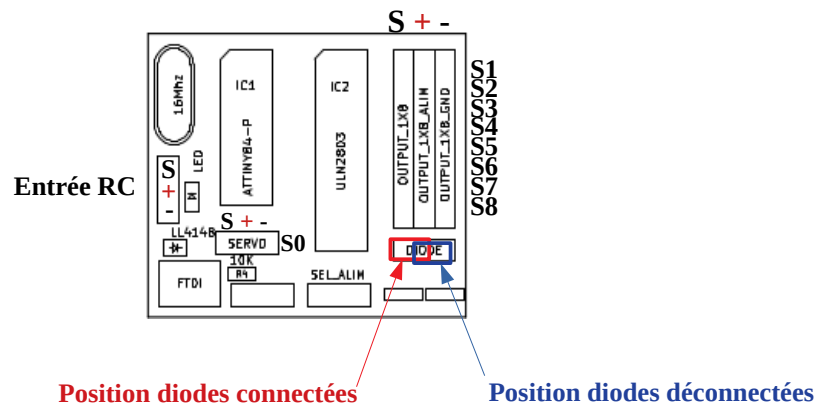


Il est possible d'alimenter les sorties **S1** à **S8** (fourniture du +) à partir :

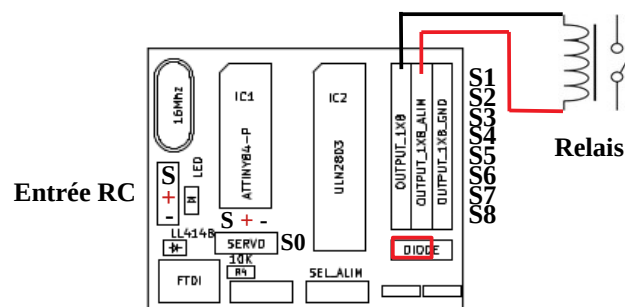
1. De la tension du **+VRx** fournie par le récepteur (Attention à la consommation sur les sorties !)
→ Cavalier de sélection alimentation sur « **VRx** »
2. D'une tension externe **VExt** (de **5V** à **24V**) appliquée sur le connecteur 2 points en bas à droite de la carte
→ Cavalier de sélection alimentation sur « **VExt** »

5.2.3 Diodes de roue libre

Si les « utilisations » connectées sur les sorties sont selfiques (ex :relais), il faut connecter les diodes internes de roue libre, sinon il y a risque de détruire l'ULN2803.



5.2.4 Commande directe de relais avec diode intégrée à l'ULN2803

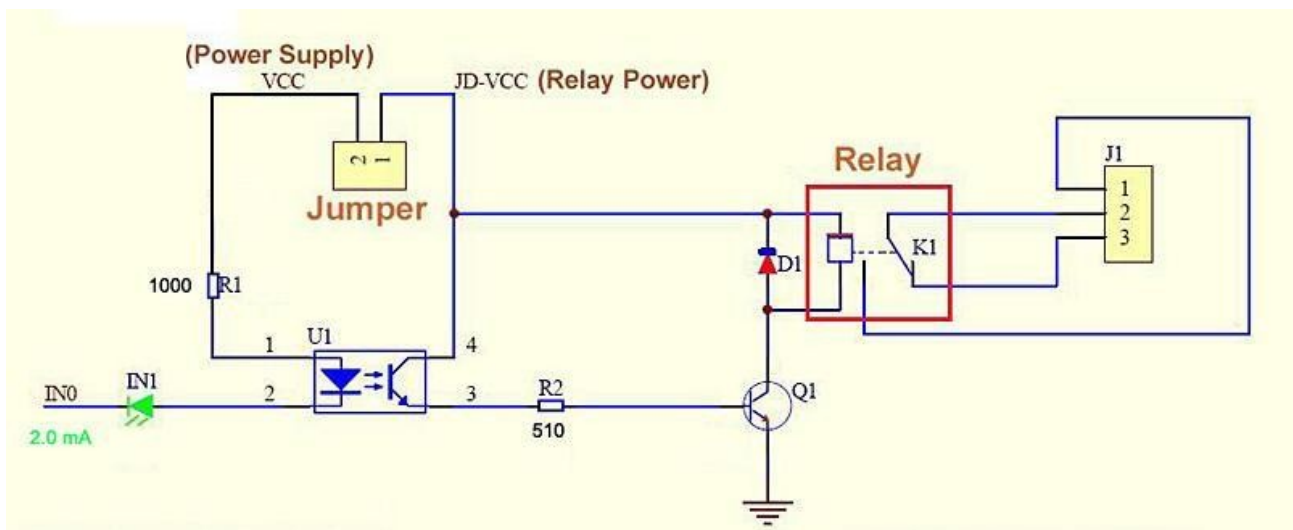


5.2.5 Commande de relais opto-isolés

Il existe des « modules relais » très bon marché souvent appelés « Arduino Relay Module ». Ces modules intègrent un opto-coupleur permettant une isolation totale entre la tension de commande et la tension d'alimentation des bobines des relais.



Le schéma équivalent d'une voie de ces modules est donné ci-dessous :

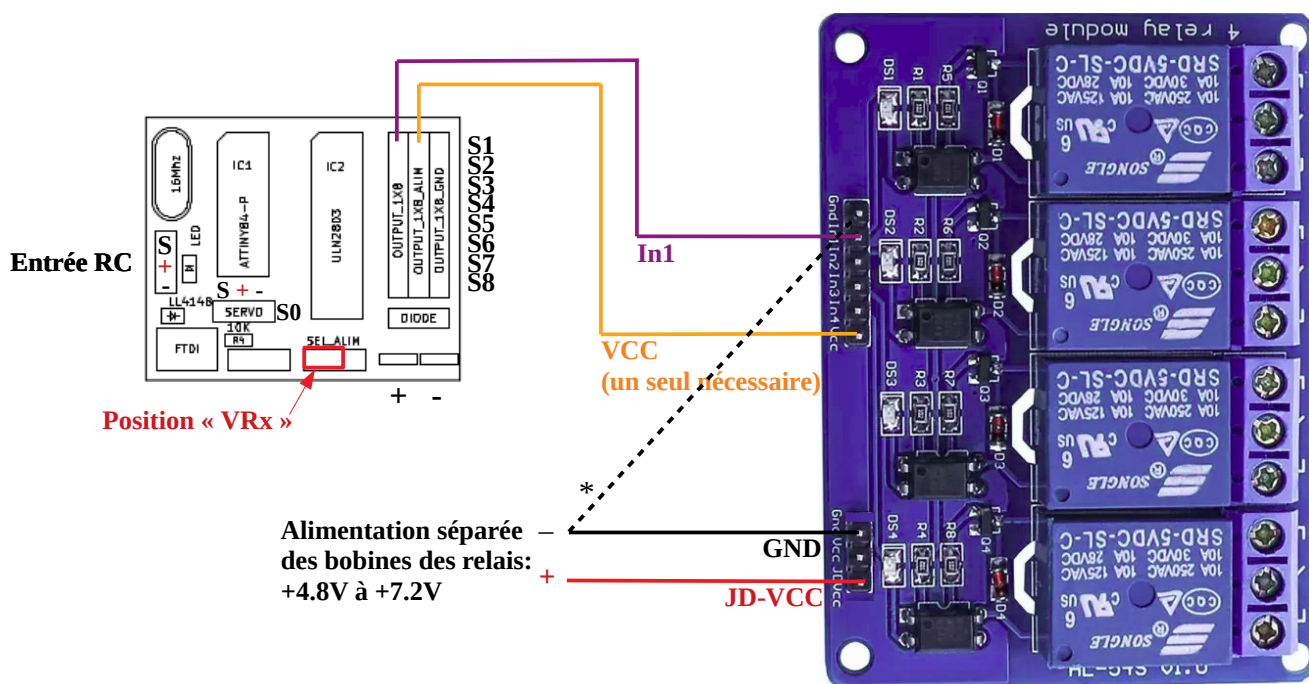


- ◆ **VCC** est la tension de commande des opto-coupleurs : accessible sur le connecteur de bord de carte
- ◆ **JD-VCC** est la tension de commande des bobines des relais : accessible sur un des points du connecteur « Jumper »

Note importante :

Pour bénéficier d'une isolation totale entre VCC et JD-VCC, le Jumper jaune (cavalier) doit être retiré.

5.2.6 Montage conseillé pour relais 5V opto-isolés



- ◆ Les opto-coupleurs sont alimentés par la tension du récepteur (conso : quelques x10 mA)
- ◆ VCC de la carte relais est relié à un des points de OUTPUT_1X8_ALIM
- ◆ In1 de la carte relais est reliée à la sortie S1
- ◆ In2 de la carte relais est reliée à la sortie S2, etc
- ◆ Les bobines des relais sont alimentées par une alimentation séparée (+4.8V à +7.2V)

→ Les alimentations étant complètement isolées, la tension du récepteur ne sera pas perturbée lors des commutations des relais : aucun risque de perdre le contrôle RC.

* : sur certains modèles de « module relais », il n'y a pas de broche **GND** à proximité du **JD-VCC**. Dans ce cas, utiliser la broche **GND** à proximité immédiate de la broche **In1** sur l'autre connecteur.

5.3 Configuration X-Any côté émetteur OpenAVRc

Pour piloter facilement le séquenceur impulsif **PulseSeq** à l'aide de la solution **MS8-Xany**, il faudra qu'au moins un des contacts côté émetteur soit de type bouton-poussoir.

Reportez vous au manuel de **MS8-Xany** du projet **OpenAVRc** pour configurer X-Any.

5.4 Mode avancé/Définition des séquences

Le séquenceur impulsif **PulseSeq** dispose d'un accès pour les configurations avancées : un port série TTL.

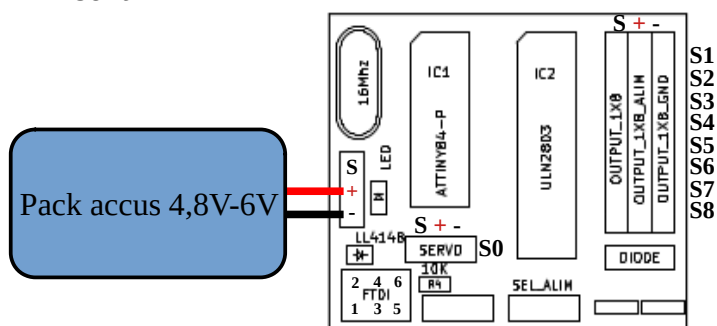
C'est cet accès série qui va permettre de définir le mode de commande ainsi que les séquences pilotant les sorties **S1** à **S8**.

5.4.1 Utilisation du port série de PulseSeq

Pour accéder au port série de **PulseSeq**, il faut un câble USB/Série TTL 5V par exemple de type « FTDI ».

Les points nécessaires sur le câble USB/Série TTL sont :

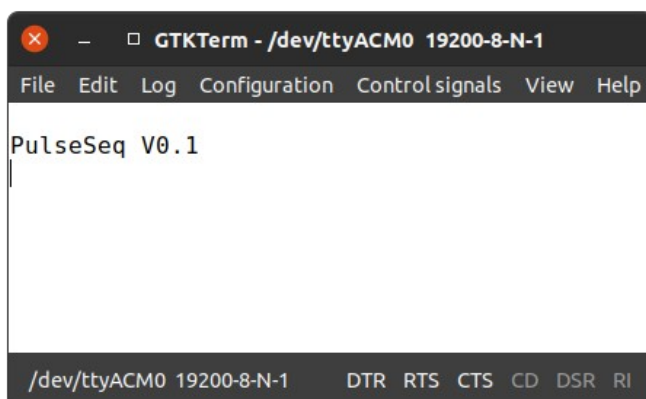
- GND « FTDI » → Point 6 de P9
- +5V « FTDI » → Non connecté
- TX « FTDI » → Point 3 de P9
- RX « FTDI » → Point 2 de P9



1. Connecter le câble USB/Série TTL sur sur le port USB d'un PC pour le côté USB
2. Sur le PC, ouvrir un Terminal série, par exemple, PuTTY, TeraTerm, HyperTerminal, GtkTerm, ou encore CoolTerm avec les paramètres suivants : 19200 bauds, 8 bits de données, 1 bit de stop, pas de parité.

Selon le Terminal série, il peut être nécessaire d'activer les retours à la ligne automatique sur réception de CR/LF pour avoir un bon affichage.

3. Connecter un **pack d'accus externe de 4,8V à 6V** sur le connecteur RC de **PulseSeq**, c'est ce **pack** qui va fournir l'alimentation.
4. **Dans les 3 secondes** après le branchement du pack d'accus sur **PulseSeq**, appuyer sur la touche « Entrée » de votre clavier, le message « PulseSeq VX.Y » doit apparaître sur le Terminal série comme illustré ci-dessous. Si ce n'est pas le cas, débrancher le **pack d'accus** et recommencer à l'étape 3 ci-dessus.



Exemple de connexion avec le Terminal GtkTerm sous Linux

5.4.2 Les messages de commande de PulseSeq

La liste des messages supportés par **PulseSeq** est donnée dans la table suivante :

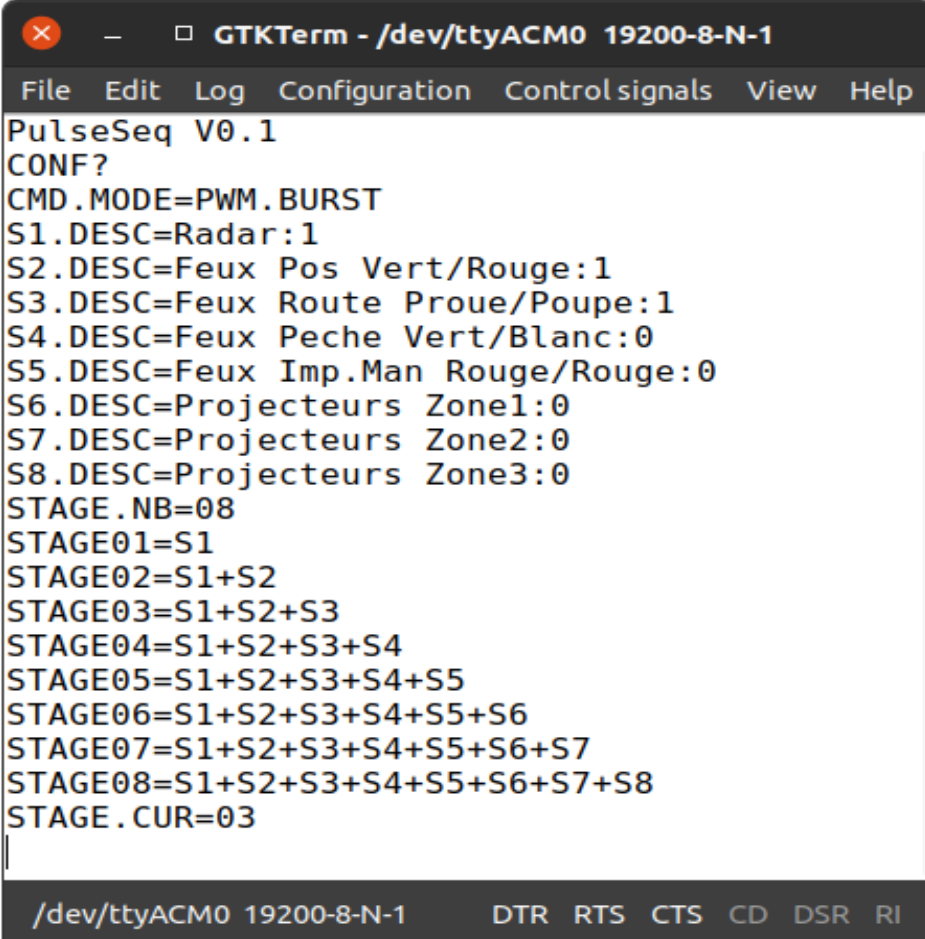
← Commande/ → Réponse	Action	Remarque
← Enter → PulseSeq Vx.y	Si envoyée pendant les 3 secondes après la mise sous tension, passe en mode Terminal	Si échec et 3 secondes écoulées, débrancher puis rebrancher le connecteur 6 points du câble USB/Série TTL
← CMD.MODE? → CMD.MODE=PWM.BURST ou → CMD.MODE=DIGITAL.PULSE	Retourne le mode de commande : - PWM.BURST : PulseSeq est commandé par une voie d'un récepteur RC - DIGITAL.PULSE : PulseSeq est commandé par un signal numérique, tel q'une sortie de décodeur Multiswitch ou un simple bouton-poussoir (test sur table sans ensemble RC)	
← CMD.MODE=PWM.BURST ← CMD.MODE=DIGITAL.PULSE → CMD.MODE	Définit le mode de commande : - PWM.BURST - DIGITAL.PULSE	Voir détail en PWM.BURST et DIGITAL.PULSE ci-dessus.
← Sx.DESC? → Sx=Description:C	Si x est compris entre 1 et 8, retourne la description de la sortie N°x ainsi que l'état « C » de la Commande courante associée (0 ou 1)	Renvoie ERR, si - Valeur x non comprise entre 1 et 8
← Sx.DESC=Description → Sx.DESC	Définit la description de la sortie x	Exemple de commandes : S1.DESC=Radar S2.DESC=Projecteur Zone 1
← STAGE.NB? → STAGE.NB=StageNb	Retourne le nombre d'étape du séquenceur (de 1 à 16)	
← STAGE.NB=StageNb → STAGE.NB	Définit le nombre d'étape du séquenceur (de 1 à 16)	Renvoie ERR, si - Valeur x non comprise entre 1 et 16
← STAGE.nn? → STAGE.nn=Sa[+Sb+...]	Retourne les sorties Sx que doit activer la séquence nn	Exemple de commandes : STAGE01? STAGE01=S3+S5
← STAGE.nn=Sa[+Sb+...] → STAGE.nn	Définit les sorties Sx que doit activer la séquence nn	Si STAGE.NB=3, il faut définir STAGE01 à STAGE03
← STAGE.CUR? → STAGE.CUR=xx	Retourne le numéro de la séquence courante	
← STAGE.CUR=Action → STAGE.CUR	Modifie le numéro de la séquence courante : Action=+ : incrémente Action=- : décrémente Action=0 : remet à zéro	Exemple de commandes : STAGE.CUR=+ STAGE.CUR=- STAGE.CUR=0 Très pratique pour tests sur table sans ensemble RC
← CONF? → La configuration	Retourne la configuration complète avec état 0/1 des sorties	Idéal pour avoir une vue rapide de la configuration et des états
← ARCHIVE?	Retourne la configuration complète	Idéal pour sauvegarder la

→ La configuration	pour archivage (sans l'état 0/1 des sorties)	configuration dans un fichier texte
--------------------	--	-------------------------------------

5.4.3 Exemple de configuration réelle

Dans l'exemple ci-dessous :

- Le mode commande est « salve PWM » (PWM.BURST) : **PulseSeq** se connecte sur une voie d'un récepteur RC
- Les 8 sorties sont utilisées
- 8 séquences sont définies (on peut en déclarer jusqu'à 16)
- La séquence courante est la N°3 : les sorties S1, S2 et S3 sont donc actives



```

GTKTerm - /dev/ttyACM0 19200-8-N-1
File Edit Log Configuration Control signals View Help
PulseSeq V0.1
CONF?
CMD.MODE=PWM.BURST
S1.DESC=Radars:1
S2.DESC=Feux Pos Vert/Rouge:1
S3.DESC=Feux Route Proue/Poupe:1
S4.DESC=Feux Peche Vert/Blanc:0
S5.DESC=Feux Imp.Man Rouge/Rouge:0
S6.DESC=Projecteurs Zone1:0
S7.DESC=Projecteurs Zone2:0
S8.DESC=Projecteurs Zone3:0
STAGE.NB=08
STAGE01=S1
STAGE02=S1+S2
STAGE03=S1+S2+S3
STAGE04=S1+S2+S3+S4
STAGE05=S1+S2+S3+S4+S5
STAGE06=S1+S2+S3+S4+S5+S6
STAGE07=S1+S2+S3+S4+S5+S6+S7
STAGE08=S1+S2+S3+S4+S5+S6+S7+S8
STAGE.CUR=03
  
```