



DECODEUR RC / SWITCH POUR BATEAUX

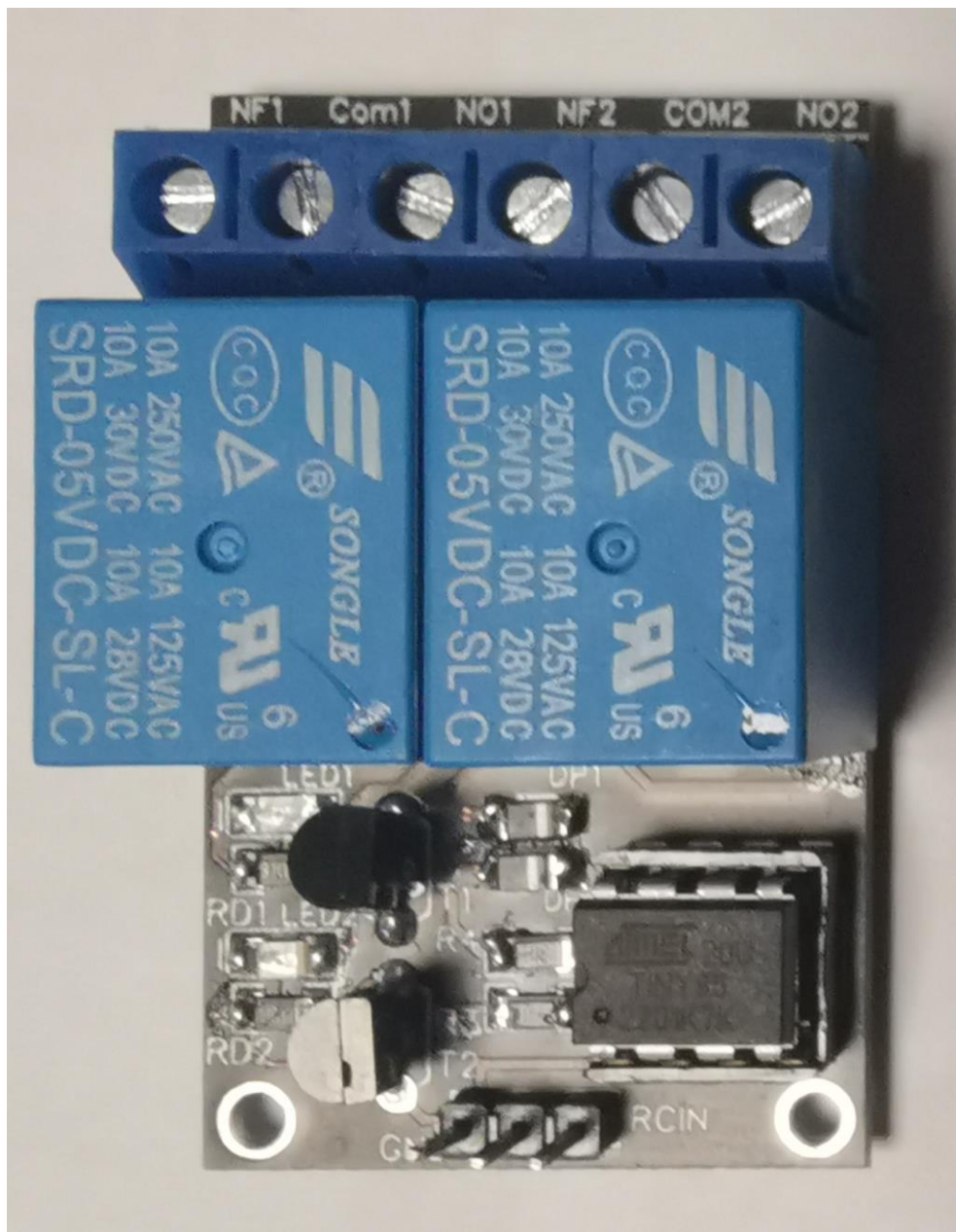


Disponible sur https://github.com/jfs59/Tiny_Decodeur/

Merci de bien lire le descriptif.

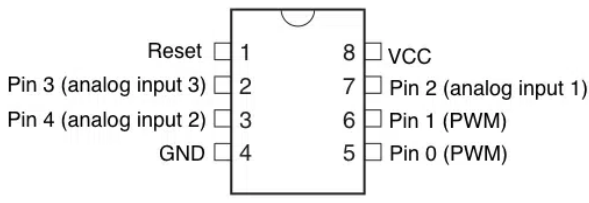
Toujours charger la dernière version de ce document ici :

Mise en garde : Bien que ne comportant pas de difficultés majeures la réalisation n'est pas facile ! Si vraiment grand débutant faites-vous aider.



JFS59 @ 2022-2023

Table des matières

Réalisation d'un décodeur 1 voie 2 relais.....	3
Préambule :	3
Principe :	3
Schema	3
ATTINY 85 5V / 8MHz DIP8.....	3
	
.....	3
Circuit imprimé	4
Composants et sous-ensembles.	4
Borniers a vis clipsables,	4
ATTiny 85 en boitier DIL 8	4
Chronologie de câblage :	5
Interconnexions :	5
Flash du programme	5
Utiliser un usbap ou Arduino as isp	5
Structure et extraits du programme AtTiny.....	6
Variables de configurations :	6
Confiuration par défaut :	6
Gestion des entrées RC par interruption :	6

Réalisation d'un décodeur 1 voie 2 relais.

Préambule :

Principe :

L'AtTiny décode le signal RC PPM et génère les niveaux nécessaires à la commande de deux relais.

L'ensemble est configurable l'aide d'un programme spécifique développé pour Windows.

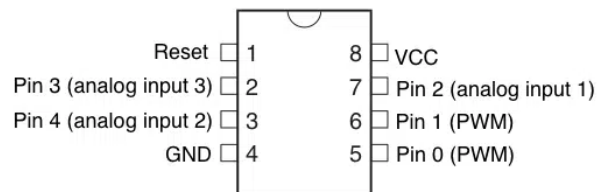
Sont configurables

Les valeurs de basculement.

Le fonctionnement monostable ou bistable des deux relais.

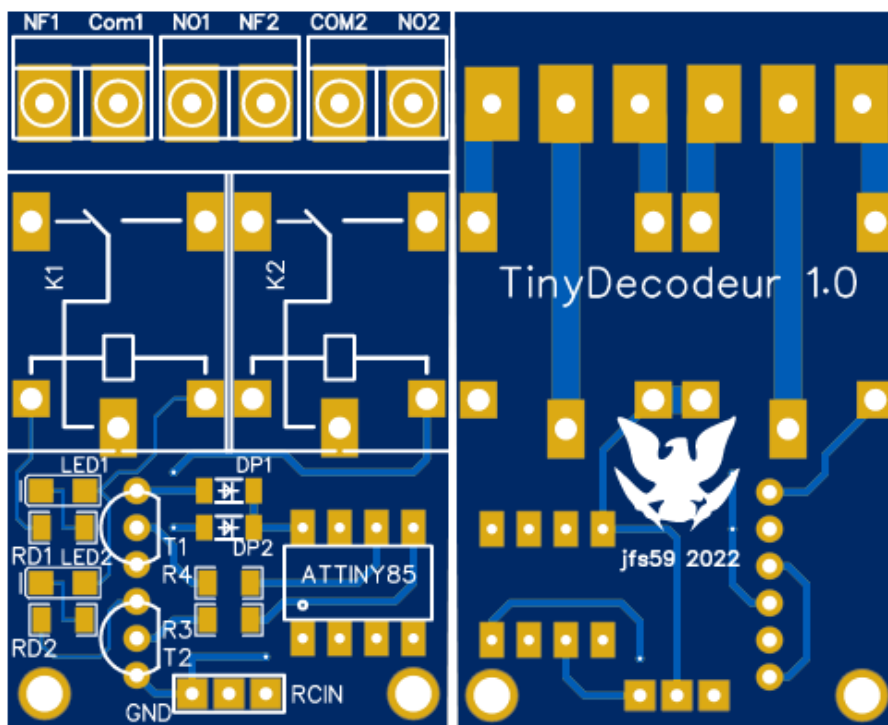
Schema

ATTINY 85 5V / 8MHz DIP8



Circuit imprimé

Circuit double face, trous métallisés, vernis épargne, et sérigraphie. Dessiné et routé par mes soins. Réalisé par une entreprise. (Le résultat donne un CI professionnel) Le routage et le design sont donnés ci-dessous mais les fichiers Gerber sont et resteront ma propriété. Le circuit imprimé est disponible pour la somme de 2.5 € couvrant les frais de réalisation et transport. (Me contacter au club de Raismes.)



Composants et sous-ensembles.

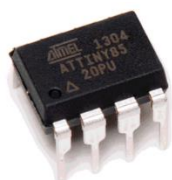
Borniers a vis clipsables,

Par 2 ou 3 il en faut 6 : 2 X3 ou 3 X2



ATTiny 85 en boîtier DIL 8

(Avec ou sans bootloader peu important)



Chronologie de câblage :

Souder les deux résistances cms 820 Ohm et vérifier en testant sur le circuit. (Consulter un tuto de soudage cms)

Souder les deux LED cms , Vérifier qu'elles s'allument en injectant du 5v dans le bon sens ! (sur les pastilles relais)

Souder les diodes de protection.

Souder les résistances de base.

Souder les transistors.

Souder la ligne de bornier.

Souder le support de l'attiny.(Support nécessaire pour pouvoir le reflasher)

Charger le sketch (programme) dans l'Attiny.

Fixer l'attiny dans le support. (Vérifier le sens !)

Souder les connecteurs de l'entrée RCin

Brancher un récepteur alimenté par un bec sur RCin

Faire varier le signal les led doivent s'allumer et changer d'état suivant la valeur du signal.

Souder les relais.

Souder les connecteurs (si nécessaires)

Interconnexions :

Le PCB est alimenté en 5V par le récepteur.

Les sorties sont des simples interrupteurs inverseurs à contacts et se branchent donc comme tel.

Flash du programme

Utiliser un usbap ou Arduino as isp

Se référer à la procédure de flash soit avec un ISP soit avec un Arduino câblé en ISP et le programme (Arduino as ISP).

Utiliser avrdude et avrdudess. (lire un tuto si nécessaire)

Structure et extraits du programme AtTiny

Le code complet ne sera pas publié par contre je laisse libre accès au fichier compilé et donc au fichier hexadécimal à programmer.
Je peux sous certaines conditions faire pour vous la mise à jour.

Variables de configurations :

```
static const unsigned long STRUCT_FLAG = 11223344;

struct __attribute__((packed)) Decode_Config {

    unsigned int  Val_Max;
    unsigned int  Val_Min;
    bool Mono_Rel1;
    bool Mono_Rel2;
    uint8_t Lissage;

    unsigned long Flag; // controle d'integrité sauvegarde
};
```

Configuration par défaut :

```
void DefinirConfigDefaut() {

    Configuration.Val_Max = 1800;
    Configuration.Val_Min = 1300;
    Configuration.Mono_Rel1 = true;
    Configuration.Mono_Rel2 = false;
    Configuration.Lissage = 6;

    Configuration.Flag = STRUCT_FLAG;
}
```

Gestion des entrées RC par interruption :

```
ISR(PCINT0_vect) {

#define RC_LEVEL (PINB & (1 << PINB3)) >> PINB3    // "digitalRead(PWM_INPUT_PIN)",

    unsigned long timestamp = micros();           //

    if (RC_LEVEL == LOW && prevRClevel == HIGH) {
        RC_PWM = timestamp - RCpulsestart;
        prevRClevel = LOW;
    }
    else if (RC_LEVEL == HIGH && prevRClevel == LOW) {
        RCpulsestart = timestamp;
        prevRClevel = HIGH;
    }
}
```