Logiciel de copiage des trame télécommande clim Mitsubishi

Solar Management System for air conditioners

Table des matières

1. Principe

Les trames sont espionné à l'oscilloscope avec un récepteur (une simple photodiode).

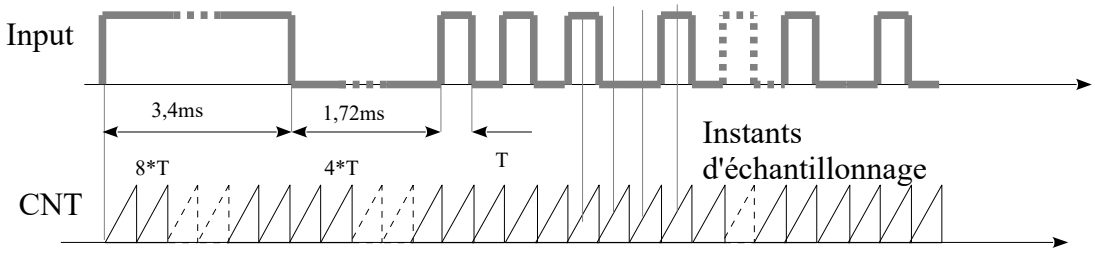
La trame est longue, environ 115 bits. Chaque bit 0/1 est en fait codé sous la forme d'une impulsion :

La trame commence par un '1' long et un '0" long :

T correspond au timing le plus court, T # 430µs

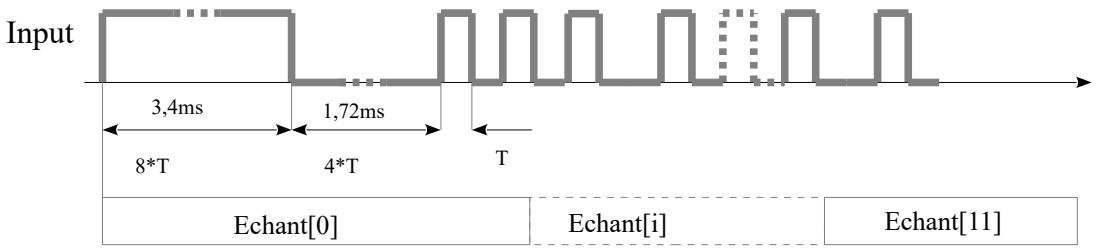
l'idée est de lancer un timer de périodicité 430µs , de programmer la pin d'entrée du train d'impulsions en IT front montant et descendant, de venir mettre à 0 le timer lors de ces IT.
Ensuite, en plein milieu du comptage, on programme une seconde IT (avec un CCR = ARR/2) qui permet de donner un ordre d'échantillonnage au bon moment.

Ensuite, il faut détecter les 8 premiers '1' et 4 '0' qui suivent pour démarrer le sampling.
Comme on a 115 bits à analyser, on peut compter le nombre de fronts montants pour stopper l'acquisition.



Donnée récupérée
La longueur de trame peut varier en longueur. Le maximum correspond à une suite de 115 bits à 0, soit 3 échantillons / bit ce qui donne 115*3 = 345bits. Il faut ajouter les 12 échantillons de frame start ce qui donne 357 échantillons.
Pour cela nous utilisons des mots de 32 bits, ce qui donne 11,1 mots soit **12 mots de 32bits**.

Ainsi nous allons utiliser une tables de 12 éléments, Echant[12] que l'on va exploiter de la sorte :



La durée maximale pour jouer la séquence est donc de :
Taille Mot*Nbre Mots*T = 32*12*430µs = **165ms**.

Le stockage mémoire pour 50 commandes est de :
50*12 = 600 Mots de 32 bits, soit **2.4kO**

2. Algorithme

Init

Start =0;

FrontUp=0

NbEch=0

IndiceTab=0

char Tab_Ech[48]=0

….....

While(1)

Si Start = 1 , Valider le Timer

Sinon l'invalider

….....

IT Front up/Dwn

Si Front up

Si (Start =0 & FrontUp=0) alors Start =1

Si FrontUp=120

Start =0;

else

FrontUp++

finsi

Fin si

CNT = 0

…....

IT CCR

IndiceTab=NbEch/8;

Tab_Ech[IndiceTab]= Tab_Ech[IndiceTab]<<1

Si Input = '1'

Tab_Ech[IndiceTab]=Tab_Ech[IndiceTab]|0x1;

FinSi (si '0' on ne fait rien)

NbEch++

3. Code C: Obtention des codes télécommande

3.1. Architecture logicielle

3.1.1. Config CubeMx

GPIO :
PC10 en Ext interrupt (l'entrée des signaux IR conditionnés)
PC11 en output ppull (pour voir les instants d'échantillonnage)

Timer :
Global interrupt activé
NB : appel au handler IT timer 2 commenté dans le fichier `stm32f1xx_it.c`
de manière à récupérer l'interruption à la main dans MyIT.c

UART2 :
configurée par défaut

3.1.2. le fichier main

→ Config Timer 2 complétée
ARR et PSC réglés à 430µs
Activation locale d'IT sur CCR1 (mi parcours ARR)

→ Config IT
Appel à MyIT_Init() cf MyIT.h

3.1.3. Le fichier MyIT.c/h

→ interruption externe PC10

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
```

C'est au départ une fonction weak écrite par HAL dans `stm32f1xx_hal_gpio.c`
Celle-ci est appelée par HAL dans `void HAL_GPIO_EXTI_IRQHandler(uint16_t GPIO_Pin)`
toujours dans `stm32f1xx_hal_gpio.c`

Cette dernière fonction est appelée par `void EXTI15_10_IRQHandler(void)` dans `stm32f1xx_it.c`

Rem : il aurait été bienvenu de procéder comme pour tim2, à savoir commenter EXTI15_10_IRQHandler et de le reprendre dans MyIT.c.

→ interruption Tim 2

```
void TIM2_IRQHandler(void)
```

3.2. Utilisation du logiciel

3.2.1. Préparation analyse de la télécommande

1- Observer plusieurs trames de télécommande à l'oscilloscope avec l'électronique de mesure (ampli transimpédance + photodiode).

2- Relever la durée du pulse le plus petit, T_{min}. C'est la durée à caler pour TIM2. Typiquement 433µs (*Mitsubishi* ou *Hitachi*)

3- Relever la durée de trame. Faire une estimation du nombre de bits nécessaires :

$$Nb\ Bit = \frac{Durée\ totale}{T_{min}}$$, ajouter 30% par sécurité

Exemple :

Mitsubishi, durée totale = 126ms, soit Nb bits = 290 bits, +30% donne 378 bits.

Hitachi, durée totale = 335ms, soit NB bits = 773 bits, +30% donne 1000 bits

4- calculer le nombre d'octets nécessaires

$$Nb\ Octet = \frac{Nb\ bits}{8}$$

Exemple :

Mitsubishi, NbOctets = 48

Hitachi, NbOctets = 125

5- Modifier dans *main.h* , `#define Nb_Octets (48)`

y mettre la bonne valeur.

6- Charger le code, connecter PC10 à la sortie du transimpédance. On peut observer les pics de synchronisation sur PC11.

Lancer XCTU sur le COM ST de la sonde ST link (UART2), et le régler à 115200 Bauds.

Ouvrir le terminal,

Lancer le code sur la nucléo 103

Activer la télécommande en face de la photodiode

Les codes s'affichent en série sur le terminal.

Vérifier bien que les derniers codes sont bien à 0 pour vérifier que la fenêtre d'échantillonnage est correcte.

Y a plus qu'à copier coller …

(commit `1da5934 suivant`)