FLAG SOFTWARE ou Sémaphore ou Signalisation ...

l'idée est de créer une variable qui vaut 0 ou 1
pour prévenir de l'apparation d'un évènement software...

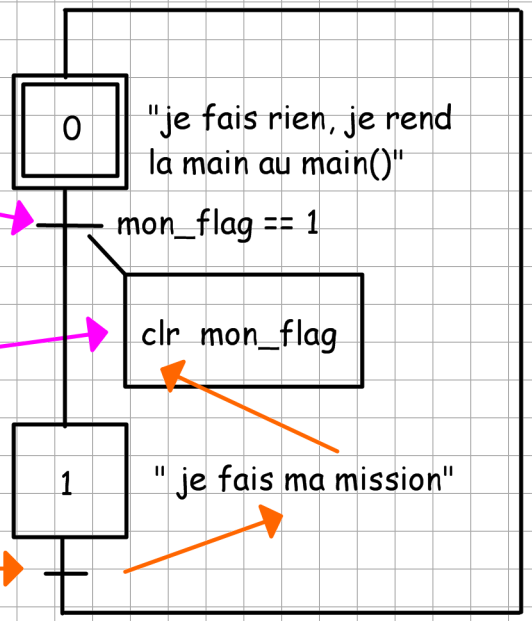
En général, c'est pour "débloquer",
pour faire avancer l'état d'une machine à état
ou pour faire un appel à une fonction

tache A

hep_reveille_toi !
en activant "mon_flag"
mon_flag =1 ;

tache B

Ok, j ai compris,
je gère.
Préviens moi à
nouveau si besoin
mon_flag=0;



TxCCR=0x02

acquittements FLAG hardware
n'ayant pas généré d'IT

T0MCR=0x03

acquittement FLAG hardware à
l'origine de l'IT

acquittement FLAG software
PAR REMISE A ZERO de la variable

comme la mission se finit de suite, on transforme en action
à la transition, et comme il reste un seul état --> simple test

Un sémaphore est une variable globale qui est initialisée à 0 pour ne pas déclencher une première action inutile. Si le flag sert à communiquer entre une interruption et le programme principal , il est nécessaire de déclarer la variable avec le mot clef "volatile"

volatile unsigned char mon_flag = 0;

cela indique au compilateur qu'il faut toujours aller relire la valeur mémoire de la variable (sous entendu elle peut avoir été modifiée dans ton dos en IT)

Tous les registres du processeur ont été déclarés en VOLATILE car le hardware peut changer leur état.

exemple 1 :

```
volatile int Flag_mes=0, mes =0;
void IT_mesure_teleme(void) IRQ
{mettre_a_jour_nb_mes_sec();
top_mesure();//générer le chronogramme pour faire un cri...
if(TxIR & (1<<FLAG_CAPTURE0))
    {mes=lecture_valeur_capturee();  Flag_mes=1;
     TxIR= (1<< FLAG_CAPTURE0);
     }
// acq IT
TxIR=1; VicVectAddr=0;
}
void main(void)
{init_proc(); init_var();
while(1)
{if (Flag_mes==1) {Flag_mes=0;calcul_dist_secu();}
.....
}
```

acquittements par
écriture d'un 1
pour effacer le bit levé...
On écrit sur le reset
d'une bascule

reste endormie tant que Flag_mes
ne la réveille pas