ENSSAT EII1

DS de Systèmes à microprocesseurs

Documents autorisés - Durée 1 heure

Mardi 6 juin 2006

1 Réalisation d'un transmetteur automatique morse (···/10pt)

Vous devez réaliser un transmetteur en morse! Le morse est un code rudimentaire composé de signaux courts (les points) et de signaux longs (les tirets). Par exemple, la lettre C en morse correspond à -.-. (long/court/long/court). En morse classique, l'utilisateur appuie sur un interrupteur pour générer les signaux courts et les signaux longs.

Ici, vous devez générer un signal morse automatique avec les durées suivantes pour les différents signaux :

- 0.2 sec pour un signal court.
- 0.6 sec pour un signal long.
- L'espace de temps entre les symboles morse d'une même lettre de l'alphabet est le même que pour un signal court (0.2 sec).
- L'espace de temps entre deux lettres est égal à 3 signaux courts (0.6 sec)
- L'espace de temps entre deux mots est équivalent à 7 signaux courts (1.4 sec).

La figure 1 représente ainsi le code morse du mot ENSSAT, et le détail de l'émission du N est donné par la figure 2.

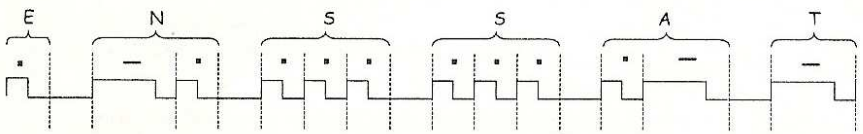

FIG. 1 – Code morse du mot ENSSAT

L'association entre la lettre de l'alphabet et la chaîne de symboles à générer est réalisée à l'aide d'une table présente en mémoire. On suppose que la gestion de cette association est faite par ailleurs et permet d'obtenir un tableau `chaine_symb` défini en variable globale avec les conventions suivantes : 0 pour un espace entre lettres, 1 pour un point et 2 pour un tiret. Pour l'émission du mot ENSSAT, par exemple, ce tableau est le suivant :

```
unsigned int chaine_symb[17]={1,0,2,1,0,1,1,1,0,1,1,1,0,1,2,0,2}.
```

Etant donné que les symboles successifs à transmettre sont souvent différents, il faut reprogrammer le timer en changeant entre autres les valeurs des registres de comparaison. Pour ce faire, une interruption est déclenchée avant la génération de chaque nouveau symbole (lignes pointillées sur les figures) et la reprogrammation est effectuée dans la routine d'interruption associée.

Q.1.1) Quel mode de fonctionnement du timer vous semble le plus approprié pour générer le code morse ?

Le rapport cyclique d'un signal logique est défini comme le rapport entre la durée du niveau haut et la période du signal.

Q.1.2) D'après la figure 2, qui représente le détail de la production d'un N, en déduire le rapport cyclique qui permet de générer un tiret. Même question pour un point.

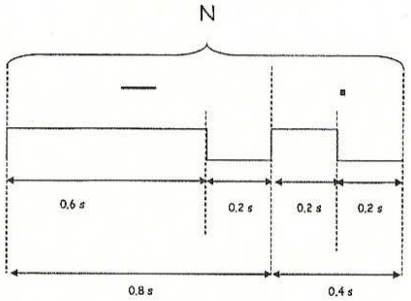

FIG. 2 – Code morse de la lettre N

Q.1.3) Déterminer les registres R_A, R_B ou R_C dont vous avez besoin et faire un schéma simple accompagné d'explications présentant le fonctionnement du timer d'un point de vue logique pour réaliser la fonctionnalité souhaitée.

Q.1.4) En considérant que le microprocesseur fournit une horloge de base `MCK` de 32 MHz, donner les valeurs des registres pour générer un point, ainsi que pour générer un tiret.

Q.1.5) Dans le cas de 2 symboles successifs d'une même lettre, déterminer la configuration du registre `TC_CMR` (TC Channel Mode Register) en choisissant pour chaque champ présenté ci-dessous, la valeur correcte. **Vous devez répondre sur cette feuille en cochant la case correspondante :**

- Nom : ...
- Prénom : ...

- Horloge en entrée du compteur (TCCLKS : Clock Selection)
  - ☐ MCK/2
  - ☐ MCK/8
  - ☐ MCK/32
  - ☐ MCK/128
  - ☐ MCK/1024
  - ☐ XC0
  - ☐ XC1
  - ☐ XC2

- Polarité de l'horloge, CLKI : Clock Invert
  - ☐ 0 = Counter is incremented on rising edge of the clock.
  - ☐ 1 = Counter is incremented on falling edge of the clock.

- BURST : Burst Signal Selection
  - ☐ The clock is not gated by an external signal.
  - ☐ XC0 is ANDed with the selected clock.
  - ☐ XC1 is ANDed with the selected clock.
  - ☐ XC2 is ANDed with the selected clock.

- CPCSTOP : Counter Clock Stopped with RC Compare
  - ☐ 0 = Counter clock is not stopped when counter reaches RC

- ☐ 1 = Counter clock is stopped when counter reaches RC

- CPCDIS : Counter Clock Disable with RC Compare
  - ☐ 0 = Counter clock is not disabled when counter reaches RC
  - ☐ 1 = Counter clock is disabled when counter reaches RC

- ETRGEDG : External Trigger Edge Selection
  - ☐ None
  - ☐ Rising edge
  - ☐ Falling edge
  - ☐ Each edge

- EEVT : External Event Selection
  - ☐ TIOB Input
  - ☐ XC0 Output
  - ☐ XC1 Output
  - ☐ XC2 Output

- ENETRG : External Event Trigger Enable
  - ☐ 0 = The external event has no effect on the counter and its clock. In this case, the selected external event only controls the TIOA output.
  - ☐ 1 = The external event resets the counter and starts the counter clock.

- CPCTRG : RC Compare Trigger Enable
  - ☐ 0 = RC Compare has no effect on the counter and its clock.
  - ☐ 1 = RC Compare resets the counter and starts the counter clock.

- WAVE
  - ☐ 0 = Waveform Mode is disabled (Capture Mode is enabled).
  - ☐ 1 = Waveform Mode is enabled.

- ACPA : RA Compare Effect on TIOA
  - ☐ None
  - ☐ Set
  - ☐ Clear
  - ☐ Toggle

- ACPC : RC Compare Effect on TIOA
  - ☐ None
  - ☐ Set
  - ☐ Clear
  - ☐ Toggle

- AEEVT : External Event Effect on TIOA
  - ☐ None
  - ☐ Set
  - ☐ Clear
  - ☐ Toggle

- ASWTRG : Software Trigger Effect on TIOA
  - ☐ None
  - ☐ Set

  - ☐ Clear
  - ☐ Toggle

- BCPB : RB Compare Effect on TIOB
  - ☐ None
  - ☐ Set
  - ☐ Clear
  - ☐ Toggle

- BCPC : RC Compare Effect on TIOB
  - ☐ None
  - ☐ Set
  - ☐ Clear
  - ☐ Toggle

- BEEVT : External Event Effect on TIOB
  - ☐ None
  - ☐ Set
  - ☐ Clear
  - ☐ Toggle

- BSWTRG : Software Trigger Effect on TIOB
  - ☐ None
  - ☐ Set
  - ☐ Clear
  - ☐ Toggle

Q.1.6) Développez en C la routine d'initialisation du timer `Init_Timer` pour réaliser la fonctionnalité souhaitée (vous utiliserez le timer 0).

Q.1.7) Développez en C la routine d'interruption `Inter_Symbole` (la reprogrammation doit tenir compte non seulement du caractère à venir, mais aussi du suivant, notamment dans le cas de l'espace entre lettres).