Université Kasdi Merbah Ouargla
Faculté des nouvelles technologies de l'information et de télécommunications
Département de l'**électronique et de télécommunications**

**Année Universitaire :** 20**14** / 20**15**
**Spécialité :** Instrumentation pétrolière (**Master 1**)
**Durée : 1**h **30**min

# Examen Systèmes à Microprocesseurs

## Exercice 1 :

**Partie I / [05 points]**

1. Quel est l'**apport** de la logique programmée par rapport à la logique câblée ?
2. Donner un **schéma** qui illustre le traitement d'une interruption.
3. Expliquer la **différence** entre une interruption **IRQ** et une interruption **NMI**.

**Partie II / [05 points]**

Lors du traitement des sorties, les périphériques de sorties reçoivent des impulsions. Ces dernières (impulsions) doivent avoir une durée correcte (ou un **délai** calculé).

1. Citer deux périphériques de **sortie**.
2. Citer les méthodes qui permettent de **générer** les délais.
3. Ecrire **trois codes** qui permettent de **générer** les délais suivants : (**L'horloge étant égale à 01 MHZ**)
   - **32** micro secondes**.**
   - **2403** micro secondes**.**
   - **02** minutes **et 13,7** secondes.

## Exercice 2 : [05 points]

Ecrire un (**seul**) programme qui permet d'effectuer les opérations suivantes :

1. Effectuer une opération sur le **registre d'état** afin d'éviter les **IRQ**.
2. Charger l'accumulateur **A** avec le contenu de **[01F1]** et l'accumulateur **B** avec le contenu de **[01F2].**
3. Incrémenter l'accumulateur **A**.
4. Soustraire la valeur **05** de l'accumulateur **B**.
5. Ranger le contenu de l'accumulateur **A** dans $\mathbf{X_H}$ (Partie **haute** du **registre d'index**) et le contenu de l'accumulateur **B** dans $\mathbf{X_L}$ (Partie **basse** du **registre d'index**).
6. Sauvegarder le contenu du **registre d'état** dans **[0100].**
7. Exécuter le **sous programme** dont l'adresse est indiquée par le contenu du **registre d'index X**.
8. Restituer le contenu du **registre d'état**.
9. Autoriser les **IRQ**.
10. Effectuer un arrêt **logiciel**.

## Exercice 3 : [05 points]

Ecrire un programme qui permet de transférer le **bloc de données stockées en mémoire** de l'adresse [**0000**] à [**00FF**] vers l'emplacement mémoire [**0F00**] à [**0FFF**].

***Remarque :*** ***Un seul document est autorisé = Le jeu d'instructions.***

***N. NASRI …***

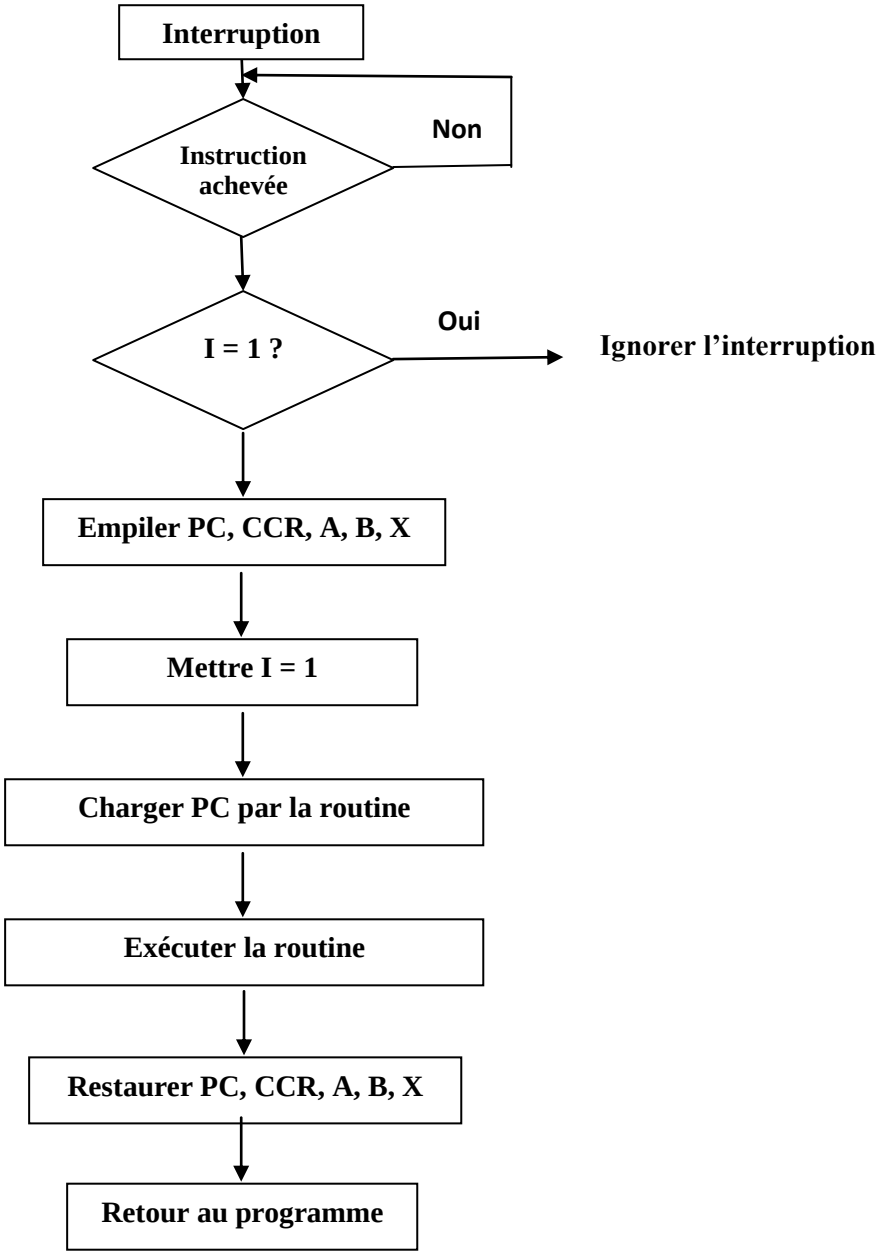
# Solution de l'examen Systèmes à Microprocesseurs 1/3

## Solution de l'exercice 1 :

**Partie I / [05 points]**

1. **Apport** de la logique programmée par rapport à la logique câblée :
   - Les systèmes programmés fonctionnent indépendamment du câblage.
   - Le fonctionnement est déterminé par un programme.
   - On peut modifier le fonctionnement en modifiant le programme.

2. **Schéma** qui illustre le traitement d'une interruption :




3. **Différence** entre une interruption **IRQ** et une interruption **NMI :**

| IRQ | NMI |
|---|---|
| - Ligne d'interruption normale<br>- Peut être masquée par **I = 1**<br>- Routine à l'adresse [**FFF8]** – [**FFF9]** | - Ligne d'interruption prioritaire<br>- Non masquable<br>- Routine à l'adresse [**FFFC]** – [**FFFD]** |

# Solution de l'examen Systèmes à Microprocesseurs 2/3

**Partie II / [05 points]**

**4.** Deux périphériques de **sortie** :

- Imprimante.
- Ecran.

**5.** Méthodes qui permettent de **générer** les délais :

- Méthode hardware (Temporisateur).
- Méthode software (Délais programmées)

**6.** **Codes** qui permettent de **générer** les délais suivants : (**L'horloge étant égale à 01 MHZ**)

- **32** micro secondes **:**

| | | |
|---|---|---|
| | **LDA A #**$N_1$ | **2** cycles |
| loop | **DEC A** | **2** cycles |
| | **BNE** loop | **4** cycles |

On pose $N_1 = 5$ alors : $2 + (2 + 4) \times N_1 = 2 + 6 \times 5 =$ **32 µs**

- **2403** micro secondes **:**

| | | |
|---|---|---|
| | **LDX #012C** | **3** cycles |
| loop | **DEX** | **4** cycles |
| | **BNE** loop | **4** cycles |

On pose $N_1 = (\mathbf{012C})_{\mathbf{hex}} = (\mathbf{300})_{\mathbf{dec}}$ alors : $3 + (4 + 4) \times 300 = 3 + 8 \times 300 =$ **2403 µs**

- **02** minutes **et 13,7** secondes :

| | | | |
|---|---|---|---|
| | **LDA A #**$N_1$ | **2** cycles | |
| loop2 | **DEC A** | **2** cycles | |
| | **LDX #**$N_2$ | **3** cycles | $H = 3 + 8 \times N_2$ |
| loop1 | **DEX** | **4** cycles | |
| | **BNE** $loop_1$ | **4** cycles | |
| | **BNE** $loop_2$ | **4** cycles | |

On pose $N_1 = (\mathbf{FF})_{\mathbf{hex}} = (\mathbf{255})_{\mathbf{dec}}$ et $N_2 = (\mathbf{FFFF})_{\mathbf{hex}} = (\mathbf{65535})_{\mathbf{dec}}$ alors :

$2 + [\ 2 + 4 + H\ ] \times N_1 = 133{,}69$ **S = 2 Min 13,7 S**

# Solution de l'examen Systèmes à Microprocesseurs 3/3

**Solution de l'exercice 2 : [05 points]**

**Important :** Afin de pouvoir exécuter le programme sur Moto6800, il faux avoir : des données sauvegardées aux adresses [**01F1**], [**01F2**], un sous programme en mémoire ayant pour adresse de départ le contenu de **X** et le programme se terminera par END au lieu de SWI.

```
SEI          % Forcer le flag I à 1 donc les IRQ seront masquées.

LDA A 01F1   % Charger A et B.
LDA B 01F2

INC A        % Incrémenter A.

SUB B #05    % Soustraire 05 de B.

STA A 01F3   % Charger XH et XL (Contenu A et B stockés dans des adresses différentes).
STA B 01F4
LDX 01F3

TPA          % Sauvegarder le contenu du CCR dans 0100.
STA A 0100

JSR  X ,00   % Exécute le sous programme.

LDA A 0100   % Restituer le contenu du CCR.
TAP

CLI          % Autorise les IRQ.

SWI          % Arrêt logiciel.
```

**Solution de l'exercice 3 : [05 points]**

**Important :** Afin d'exécuter le programme sur Moto6800, les données doivent être déjà stockées en mémoire.

```
LDX #0000    Charger le registre X par l'adresse de la 1ere valeur (emplacement du départ).
LDS #0F00    Charger le registre SP par l'adresse d'arrivée ou sera transférée la 1ere valeur.
loop:
LDA A X, 00  Charger l'accumulateur A par la 1ere valeur à transférer.
PSH A        Utiliser la pile pour enregistrer dans l'adresse d'arrivée.
INX          Incrémenter registre X afin de lire la valeur suivante à transférer.
INS          Incrémenter registre SP.
INS          Incrémenter registre SP encore une fois afin d'avoir l'adresse suivante.
CPX #0100    Comparer le contenu du registre X à l'adresse qui vient après la dernière valeur.
BNE loop     Refaire la boucle tant qu'on a pas dépassé la dernière valeur à transférer.
END
```

- L'utilisation du registre **X** permet de faire une **lecture répétée** des valeurs présentes en mémoire.
- L'utilisation du registre **SP** permet de **gérer** les adresses de transfert.
- Incrémenter registre **SP deux fois** car **SP** se **décrémente** à chaque écriture.

***N. NASRI …***