partie 1 : 7,5/10
6/10
13,5/20

Année 2015/2016

A remplir obligatoirement par l'enseignant responsable du contrôle

Date : 07 janvier 2016
Contrôle de : Architecture des ordinateurs
Durée : 2 heures
Professeur responsable : N.ABOUCHI
Documents : ☑ autorisés ☐ non autorisés
Si oui : type(s) de documents autorisés :

Calculatrices alphanumériques : ☑ autorisées ☐ non autorisées

REPONDRE SUR LE SUJET : ☑ OUI NON

LES TELEPHONES PORTABLES ET AUTRES APPAREILS DE STOCKAGE DE DONNEES NUMERIQUES NE SONT PAS AUTORISES

A l'attention des élèves : rappels importants sur la discipline des examens

La présence à tous les examens est strictement obligatoire ; tout élève présent à une épreuve doit rendre une copie, même blanche, portant son nom, son prénom et la nature de l'épreuve.

Toute absence non justifiée est sanctionnée par un zéro.

Toute fraude ou tentative de fraude avérée est sanctionnée par un zéro à l'épreuve et portée à la connaissance de la direction des études qui pourra réunir le Conseil de Discipline. Les sanctions prises peuvent aller jusqu'à l'exclusion définitive du (des) élève(s) mis en cause.

Toute suspicion sur la régularité et le caractère équitable d'une épreuve est signalée à la direction des études qui pourra décider l'annulation de l'épreuve; tous les élèves concernés par l'épreuve sont alors convoqués à une épreuve de remplacement à une date fixée par le responsable d'année.

Exercice 1 : Généralités sur le 8051 (4 pts) plusieurs réponses possibles, le point est affecté uniquement si toutes les réponses sont exactes.

Question 1 (1 pt) : Le microprocesseur 8051:

- ☑ Est un microprocesseur 8 bits
- ☑ Est un microprocesseur de type Harvard
- ☐ Dispose de deux registres d'adresses
- ☑ Dispose d'un espace mémoire pour le code séparé de l'espace mémoire pour les données
- ☑ Dispose d'un seul bus d'adresses

0,5

Question 2 (1 pt) : les registres internes du 8051

- ☐ Sont au nombre de 32 (R0, R1, R2, R3, R4, ……, R31)
- ☐ Sont localisés dans la mémoire code
- ☐ Sont localisés dans la RAM externe
- ☐ Sont localisés dans l'espace mémoire SFR
- ☑ Sont au nombre de 32 (4 fois [R0, R1, R2, R3, R4, R5, R6, R7])

0

Question 3 (1 pt) : **gestion de la pile**

- ☐ Les instruction CALL et JMP sont équivalentes
- ☐ L'instruction PUSH place un octet dans la pile et incrémente de 2 le registre SP
- ☒ L'instruction CALL provoque la mémorisation automatique du registre pointeur de pile (SP) dans la pile et incrémente de 2 le registre SP
- ☒ Le registre SP est automatiquement incrémenté d'un octet suite à l'instruction JMP
- ☐ La pile est forcément une mémoire de type EPROM

0,5

Question 4 (1 pt) : **modes d'adressages**

- ☒ Seul l'adressage indirect est possible pour les accès en mémoires de données externe
- ☒ L'adressage immédiat ne permet pas d'écrire dans la mémoire code
- ☒ L'instruction CALL utilise l'adressage direct
- ☒ L'instruction JMP utilise l'adressage direct
- ☒ L'écriture dans les registres R0 à R7 n'utilise pas forcément l'adressage immédiat

1

Exercice 2 : Notion de programmation du 8051F020 (6 points)

- Le 8051F020 utilise les deux registres pointeur de code PC et pointeur de données DPTR pour accéder aux mémoires externes. Expliquer par l'instruction **MOVX @DPTR, A** comment et quand est-ce que ces deux registres sont utilisés et pourquoi ?

- Que fait l'instruction **MOV 00,#23H** ? quelle modes d'adressages sont utilisés pour les opérandes source et destination ?

Cette instruction place la valeur 23h à l'adresse 00 (soit R0 si on utilise la banque de registre 0). L'opérande source utilise l'adressage immédiat, l'opérande destination utilise l'adressage direct.

2

- Ecrire en assembleur 8051 les programmes suivants :
  - Copier le contenu de la mémoire 1234H (Espace XDATA) dans le registre DPH.

```
MOV DPTR, #1234h
MOV A, @DPTR
MOV DPH, A
```

2

  - Copier le contenu de la mémoire 0002H (Espace code) dans la mémoire 1234H (Espace XDATA).

```
MOV A, #00h
MOV DPTR, #0002h
MOVC A, @A+DPTR
MOV DPTR, #1234h
MOVX @DPTR, A
```

2

Gestion de la pile

Pour le programme suivant, représentez la pile (contenus et adresses) à 2 moments différents d'exécution du code (complétez les croquis ci-dessous).
Remarquez que ce programme ne fait rien de censé, il ne sert qu'à nous permettre de vérifier votre compréhension de la mise en œuvre de la pile.

```
89  ;********************
90  Prog_biden:
91          mov sp,#40H
92  ;       ...
93  bcl:    nop
94          call sp1
95  ;       ...
96          jmp  bcl
97  ;********************
98  sp1:    push ACC
99          push 10
100 ;       ...
101         call sp2
102 ;       ...
103         pop 10
104         pop ACC
105         ret
106 ;********************
107 sp2:    push DPH
108         push PSW
109 ;       ....
110         pop  PSW
111         pop  DPH
112         ret
113 ;************************
```

; ... cette ligne signifie qu'il peut y avoir plus de code, mais qu'il n'affecte pas la pile

Contenu de la pile et valeur de SP juste avant l'exécution de la ligne 109

Mémoire IDATA

Adresses:	Contenus:
48h	PSW
47h	DPH
46h	PC >
45h	PC <
44h	@10
43h	ACC
42h	PC >
41h	PC <

SP= 48h

Contenu de la pile et valeur de SP juste avant l'exécution de la ligne 96

Mémoire IDATA

Adresses:	Contenus:

SP= 40h

la pile ne contient rien juste avant l'execution de la ligne 96

Indiquez le contenu de chaque élément de la pile sous la forme d'un nom de registre, d'une adresse, d'une partie d'adresse, etc....

1/1

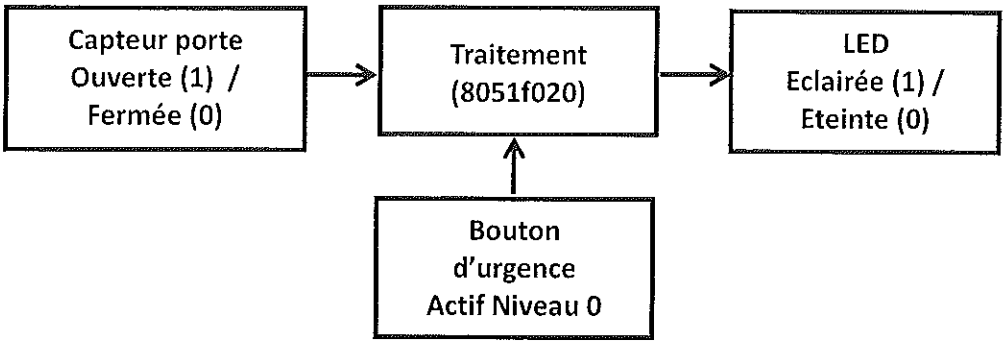
Problème : Codage en C et configuration de périphériques du 8051F020 (10 pts)

On souhaite utiliser un système basé sur le microcontrôleur 8051F020 pour surveiller l'accès à un local de stockage. L'état de la seule porte d'accès au local est en permanence affiché sur une LED. La LED est éclairée si la porte du local est ouverte et éteinte si la porte est fermée.

Le fonctionnement du local est le suivant :

- Le local est normalement fermé (LED éteinte),
- Si la seule porte d'accès au local est ouverte, la LED passe en mode éclairé,
- Si la porte reste ouverte pendant plus que 100 secondes, le LED clignote (100ms éteinte, 100 ms allumée)
- La LED reste en mode clignotement jusqu'à ce que la porte soit fermée.
- L'appui sur le bouton d'urgence place automatiquement la porte en position ouverte, par un mécanisme à ne pas considérer dans le cadre de ce problème, dans ce mode, la LED clignotera par les allumages brefs (100ms allumée, 1s éteinte).
- On sortira du mode « Urgence » par un redémarrage du système.

Schéma de principe de la réalisation :



Mise en œuvre :

- Le capteur d'état de la porte `Etat_porte` sera relié sur le port **P1.1**. Un niveau 0 signalera l'état « porte fermée »
- La LED `LED` sera reliée sur le port **P1.6.** Elle s'allumera avec un niveau haut
- Le bouton d'urgence `Urgence` sera relié sur l'entrée d'interruption **INT7 (P3.7)**. Ce bouton est actif sur un niveau bas.

L'objectif du problème est d'écrire un programme permettant de:

a. scruter en permanence l'état de la porte

b. mettre à jour l'état de la LED

c. vérifier que la durée de la porte ouverte soit inférieure à 100 secondes.

d. gérer les divers modes de clignotement de la LED

e. gérer le mode « Urgence ».

f. Mis à part le mode Urgence, toute la gestion du dispositif sera assurée par une interruption produite par un timer.

Questions :

Question 1 – Configuration globale du microcontrôleur (1 point) 0

Pour cette application, quels sont les éléments de configuration globale à gérer ?
Citez-les, expliquez. (Nous ne demandons pas de code dans cet exercice).

0

Question 2 – Configuration des broches d'entrée-sortie et de l'entrée d'interruption INT7 (2 points)

Configuration de la broche **Etat_Porte** (P1.1) en entrée. 2

Expliquez ce que vous devez faire

Nommer les registres sur lesquels vous devez agir et pourquoi.

Il faut configurer ce port comme étant une entrée. Pour cela, on doit le configurer en drain ouvert pour ouvrir le transistor TcH. De plus, il faut écrire 1 dans le port pour ouvrir le transistor TcL.

1

Les registres sur lesquelles on doit intervenir sont donc : P1NDOUT pour lequel on devra mettre le bit 1 à 0 afin de configurer le port en drain ouvert (TcH ouvert) et le registre P1 dans lequel on viendra écrire un 1 sur le bit 1 (TcL ouvert).

Configuration de la broche **LED** (P1.6) en sortie. Cette broche fait allumer un LED lorsque que la broche délivre un niveau haut. (Courant LED = 15mA).

Expliquez ce que vous devez faire

Nommer les registres sur lesquels vous devez agir et pourquoi.

Il faut configurer ce port en sortie, on choisira donc de le mettre en push pull. Pour cela, on interviendra sur le bit 6 du registre P1NDOUT, qu'on mettra à 1.

1

Coder la fonction `Config_GPIO()` . Cette fonction assurera la configuration des broches P1.1, P1.6 et P3.7 Ce code ne devra pas altérer la configuration des ports autres que P1.1, P1.6 et P3.7.

```
Config_GPIO() {
    P1NDOUT &= ~0x02;
    P1NDOUT |= 0x40;
    P1 |= 0x02;
    P3NDOUT &= ~0x80;
    P3 &= 0x80;
}
```

Question 3 – Configuration d'un Timer (3 points) (3)

Pour assurer la gestion du système, on fera en sorte de provoquer une interruption toutes les 10ms. Cette interruption sera produite par un Timer.
C'est dans ce programme d'interruption que l'on exécutera la lecture de l'état de la porte et le pilotage de la LED.
Pour cet exercice, nous vous imposons l'utilisation du Timer3.

Expliquez le fonctionnement du timer dans cet exercice.

Dans cet exercice, on utilisera le timer 3 en mode compteur/timer avec auto-rechargement. Le timer est préchargé avec une valeur. Lorsque on lance le timer, il commence à compter depuis sa valeur de rechargement jusqu'à FFFFh + 1 soit 0000h. À ce moment là, une interruption d'déclenchera son sous-programme d'interruption qui effectuera le traitement souhaité. De plus le timer est automatiquement rechargé. (1)

Inventoriez tous les registres sur lesquels vous allez devoir agir pour programmer le timer3 afin qu'il génère une interruption toutes les 10ms.
Pour chaque registre, indiquez son rôle.

Il va tout d'abord falloir agir sur le registre TMR3CN qui est le registre de contrôle du Timer 3. Le bit 0 devra être à 0 car nous souhaitons utiliser un horloge interne, le bit 1 devra être à 1 afin d'utiliser la fréquence d'horloge divisée par 12. Les registres TMR3RLH et TMR3RLL sont les registres de préchargement du timer, on y mettra la valeur qui nous intéresse. Il faudra également précharger cette valeur dans les registres TMR3H et TMR3L, qui sont les registres compteurs, afin d'avoir une première période égale aux autres. Il faut aussi activer l'interruption dans le registre EIE2, et activer la validation d'interruption global EA. Enfin, on lance le timer avec le bit 2 du registre TMR3CN (1)

Codez la fonction `Config_Timer3()`. Cette fonction configure le Timer3 pour générer une interruption toutes les 10ms (on considère que le 8051F020 fonctionne avec une fréquence SYSCLK de 22,1184 MHz)

```
Config_Timer3(){
    TMR3CN &= 0x01;
    TMR3CN |= 0x02;
    TMR3RLH = 0xB8;
    TMR3RLL = 0xCC;
    TMR3H = 0xB8;
    TMR3L = 0xCC;
    EIE2 |= 0x01;
    EA = 1;
}TMR3CN |= 0x02;
```

(1)

Question 4 – Configuration de l'interruption externe INT7 (P3.7). (1 point)

Expliquez le mode retenu pour le fonctionnement de INT7.
Coder la fonction `Config_INT7()`

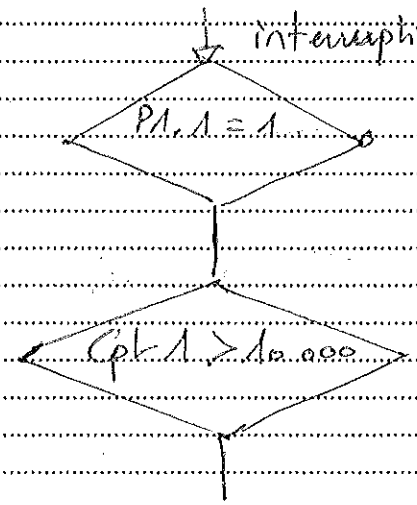
On choisit d'utiliser un mode de fonctionnement sur front descendant car le bouton est actif au niveau logique 0.

```
Config_INT7(){
    P3IF &= 0x04;
    EIE2 |= 0x20;
}
```

Question 5 – Programme d'interruption du Timer 3 (3 points)

C'est dans ce programme que toute la gestion de l'application est faite (la boucle infinie dans la fonction **main** est vide).

Ecrire sous forme de pseudo-code ou d'algorigramme ce programme d'interruption.



Question 6 – Amélioration du dispositif. (Bonus 2 points)

Pour améliorer le dispositif, on souhaite envoyer l'information à distance à l'aide d'une liaison série asynchrone.
Expliquez comment vous pourriez procéder. Citez les paramètres dont vous pourriez avoir besoin.
Quels sont les périphériques à mettre en œuvre pour cette application.
Expliquez les configurations à faire et comment vous comptez utiliser ce périphérique.