Année universitaire 2020/2021

NOM :..... **PRENOM :.....**

Consignes relatives au déroulement de l'épreuve

Date : 20 janvier 2022

Devoir Module Bases des systèmes embarqués – Session 1 - Partie TP

Durée: 1H00

Professeurs responsables : François Joly

Documents : ☒ autorisés ☐ non autorisés

Si oui : type(s) de documents autorisés : Polycopié Fiche technique du 8051F020 uniquement

Calculatrices : ☒ autorisées ☐ non autorisées

Si oui : type(s) de calculatrices autorisées : alphanumériques

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

Les téléphones portables doivent être éteints pendant toute la durée de l'épreuve et rangés dans les cartables.

S'agissant de contrôle sans document, les trousses doivent être rangées dans les cartables.

Les cartables doivent être fermés et posés au sol.

Les oreilles des candidats doivent être dégagées.

Rappels importants sur la discipline lors 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.
Une absence non justifiée à un examen invalide automatiquement le module concerné.
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.
Toute fraude ou tentative de fraude est 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.

Devoir Session 1 – BSE - Partie TP
Durée de l'épreuve : 1H00

Contexte : Analyse de code.

Dans cette partie « TP » il va vous être demandé d'analyser un code qui a été produit pour répondre à la question 4 de l'examen « Cours ».
Le code à analyser est en annexe sur 3 pages. Ces 3 pages sont détachables pour vous faciliter la tâche.

Nous reproduisons ci-dessous, l'énoncé de cet exercice:

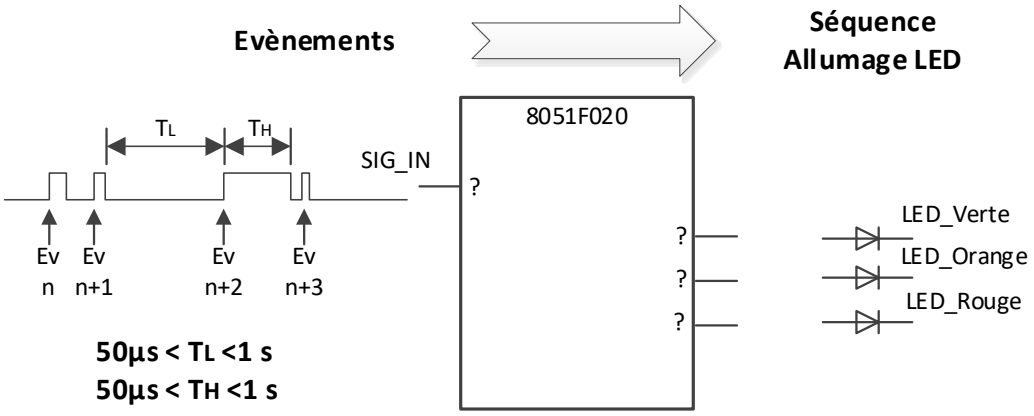
Fonctionnalité Comptage d'évènements (5 points).

Dans cet exercice, on se propose de mettre en place une solution de comptage d'évènements dans notre microcontrôleur. Ces « évènements » (le signal SIG_IN) arrivent au microcontrôleur sous forme d'impulsions positives de durée et de récurrence variables. Le niveau de tension est un niveau 0-3,3V.

En fonction du nombre d'évènements comptés, on pilotera 3 LED de couleur différentes : LED_Verte, LED_Orange et LED_Rouge
Le cycle de comptage sera le suivant :

- ***0 à 4999 évènements LED_Verte Cligno – LED_Orange éteinte – LED_Rouge éteinte***
- ***5000 à 5999 évènement LED_Verte allumée – LED_Orange éteinte – LED_Rouge éteinte***
- ***6000 à 6999 évènement LED_Verte éteinte – LED_Orange Cligno – LED_Rouge éteinte***
- ***7000 à 7999 évènement LED_Verte éteinte – LED_Orange allumée – LED_Rouge éteinte***
- ***8000 à 8999 évènement LED_Verte éteinte – LED_Orange éteinte – LED_Rouge Cligno***
- ***9000 à 9999 évènement LED_Verte éteinte – LED_Orange éteinte – LED_Rouge allumée***

En mode « Cligno » la LED s'allume pendant 50ms, puis s'éteint pendant 50ms.
Ce mode de comptage d'évènements se reproduit à l'infini. Après le 9999 ième évènement, on repart à zéro.



QUESTIONS (Chaque question vaut 1 point)

1. Qu'est censé contenir le fichier c8051F020.h ?

2. Compte tenu des codes de configuration, quelle est la fréquence de l'horloge système (SYSCLK) obtenue ? Justifiez.

3. Est-il possible d'avoir une horloge système plus précise ? Si oui, expliquez comment vous pourriez procéder et sur quelle partie du code vous agiriez.

4. Donnez les affectations de broches PIO liées à la configuration du CROSSBAR.

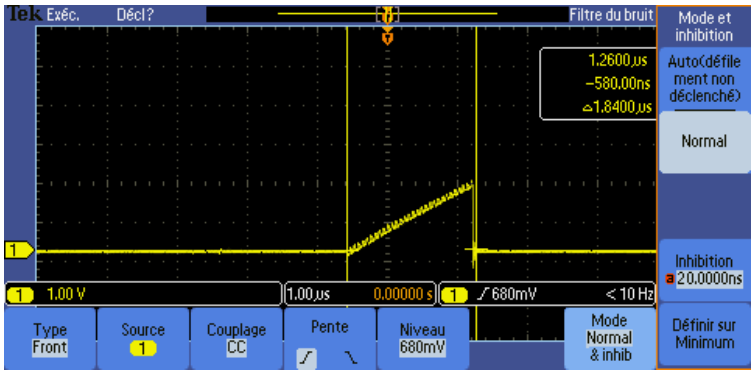
5. Quel est le rôle de la fonction `Reset_Sources_Init` dans ce code?

6. Quelle est la différence entre une déclaration de type « bit » et une déclaration de type « sbit » ?

7. Expliquez la signification de la ligne « `sfr16 RCAP2 = 0xca;` » ? Si on l'omet, par quoi devra-t-on la remplacer dans le code ?

8. Que se passe-t-il si on omet la ligne « `EA = 1;` » présente dans la fonction `main` ?

9. On a cherché à visualiser le signal VISU_TRT_D sur un oscilloscope. L'allure du signal obtenu est la suivante : Est-ce normal ? Expliquer, et proposer une solution pour corriger le problème.



Comme son nom semble l'indiquer, la fonction `Config_Timer_TimeBase` permet de configurer un périphérique Timer pour fonctionner en base de temps et donc produire des interruptions périodiquement.

10. Quel est le timer utilisé dans cette fonction `Config_Timer_TimeBase`?

.....
.....
.....

11. Dans quel mode est-il configuré ?

.....
.....
.....

12. Compte tenu de la configuration de ce Timer, exprimer la fréquence des interruptions en fonction de la fréquence de la SYSCLK

.....
.....
.....
.....
.....
.....
.....

13. Compte tenu de la configuration de ce Timer ; quel est l'évènement susceptible de déclencher une interruption ?

.....
.....
.....
.....
.....

14. Dans le code, quelle est la fonction d'interruption correspondant à la mise en œuvre de ce Timer? Justifiez.

.....
.....
.....
.....
.....

On s'intéresse désormais à la fonction `Config_B`

15. Quel est le périphérique mis en œuvre dans cette fonction ?

16. Donner des précisions sur la configuration de ce périphérique et donc sur le fonctionnement attendu.

17. Ce périphérique va-t-il pouvoir produire des interruptions ? Justifiez.

18. Quelle est la caractéristique spécifique des fonctions `TRT_C` et `TRT_D` ? Pourquoi ne sont-elles pas appelées dans le code ? Que signifient les numéro 5 et 16 derrière les noms de ces fonctions ?

19. Dans la ligne 171 « `if ((T4CON & (1<<6)) != 0) T4CON &= ~(1<<6);` » Expliquez avec des mots ce que fait ce code et à quoi il sert.

20. Sur quelles broches du microcontrôleur sont connectées les LED verte, orange et rouge ainsi que le signal SIG_IN ?

21. BONUS - Expliquer le fonctionnement global de ce code en précisant le rôle des différents périphériques mis en œuvre.

22. BONUS - Aurait-il été possible de remplacer le timer configuré et utilisé par la fonction `Config_Timer_TimeBase` par un autre timer. Expliquer et justifier.

23. BONUS - Que devriez-vous changer pour avoir des cycles de comptage de 100 000 évènements (0 à 49999, 50000 à 59999, , 90000 à 99999) ? Le comptage à 10^6 serait-il aussi possible ? Expliquer.

ANNEXES pour Examen BSE TP 20/01/22

Pour faciliter votre travail, cette feuille peut être détachée.
Elle n'est pas à rendre à la fin de l'examen.

```
//-------------------------------------------------------------------------
// Examen BSE 20-01-22.c
//-------------------------------------------------------------------------
// AUTH: FJ
// DATE: 10/01/22
// Target: C8051F02x
// Tool chain: KEIL Microvision5
//
//-------------------------------------------------------------------------
// Fichiers d'entête
#include<c8051F020.h>
//-------------------------------------------------------------------------
// Prototypes de fonctions
//-------------------------------------------------------------------------
void Oscillator_Init();
void Config_Timer_TimeBase(void);
void Config_B(void);
void Init_Device (void);
void Reset_Sources_Init();
void Port_IO_Init();
void Gestion_Etat_LED(unsigned char);
void Gestion_Cde_LED(void);
//-------------------------------------------------------------------------
// Déclaration des MACROS
#define LED_ON 1
#define LED_OFF 0
#define ON 1
#define OFF 0
#define CLIGNO 2
//-------------------------------------------------------------------------
//
sbit LED_Verte = P2^0;
sbit LED_Orange = P2^1;
sbit LED_Rouge = P2^2;
sbit VISU_TRT_C = P2^4;
sbit VISU_TRT_D = P2^5;

sfr16 RCAP2    = 0xca;
sfr16 T2       = 0xcc;
sfr16 RCAP4    = 0xe4;
sfr16 T4       = 0xf4;
//-------------------------------------------------------------------------
// Variables globales
bit New_CP_1000;
unsigned char Etat_LED_Verte;
unsigned char Etat_LED_Orange;
unsigned char Etat_LED_Rouge;
//-------------------------------------------------------------------------
// MAIN Routine
//-------------------------------------------------------------------------
void main (void)
{
        Init_Device();
        Config_Timer_TimeBase();
        Config_B();
        Etat_LED_Verte = LED_OFF;
        Etat_LED_Orange = LED_OFF;
        Etat_LED_Rouge = LED_OFF;
        EA = 1;
        while(1);
}
//-------------------------------------------------------------------------
//-------------------------------------------------------------------------
void Reset_Sources_Init()
{
    WDTCN = 0xDE;
    WDTCN = 0XAD;
}
//*************************************************************************
// Oscillator_Init()
//*************************************************************************
void Oscillator_Init()
{
    OSCICN    = 0x07;
}
//-------------------------------------------------------------------------
//-------------------------------------------------------------------------
void Port_IO_Init()
{
    XBR0      = 0x04;
    XBR1      = 0xF4;
    XBR2      = 0x58;

    LED_Verte = LED_OFF;
    LED_Orange = LED_OFF;
    LED_Rouge = LED_OFF;
    VISU_TRT_C = OFF;
    VISU_TRT_D = OFF;
}

```

```
91
92  //-------------------------------------------------------------------------
93  //-------------------------------------------------------------------------
94  void Init_Device(void)
95  {
96      Reset_Sources_Init();
97      Port_IO_Init();
98      Oscillator_Init();
99  }
100 //*************************************************************************
101 //*************************************************************************
102 void Config_Timer_TimeBase(void)
103 {
104   CKCON &= ~(1<<5);
105   TF2 = 0;
106   EXF2 = 0;
107   RCLK0 = 0;
108   TCLK0 = 0;
109   CPRL2 = 0;
110   EXEN2 = 0;
111   CT2 = 0;
112   RCAP2 =  0xCBEB;
113   T2 = RCAP2;
114   TR2 = 1;
115   PT2 = 1;
116   ET2 = 1;
117 }
118 //*************************************************************************
119 // void Config_B(void)
120 //*************************************************************************
121 void Config_B(void)
122 {
123   T4CON = 0x02;
124   RCAP4 = 64536;
125   T4 = RCAP4;
126   T4CON |= (1<<2);
127   EIP2 |= (1<<2);
128   EIE2 |= (1<<2);
129 }
130 //*************************************************************************
131 // void TRT_C(void)
132 //*************************************************************************
133 void TRT_C (void) interrupt 5
134 {
135 static unsigned char CP_New_CP_1000;
136 static unsigned char CP_50ms;
137
138     VISU_TRT_C = 1;
139     if (TF2 == 1)
140     {
141         TF2 = 0;
142         if (New_CP_1000 == ON)
143         {
144             New_CP_1000 = OFF;
145             CP_New_CP_1000++;
146             if (CP_New_CP_1000 >= 10) CP_New_CP_1000 = 0;
147             Gestion_Etat_LED(CP_New_CP_1000);
148         }
149
150         CP_50ms++;
151         if (CP_50ms >= 5)
152         {
153             CP_50ms = 0;
154             Gestion_Cde_LED();
155         }
156     }
157     if (EXF2 == 1) EXF2 = 0;
158     VISU_TRT_C = 0;
159 }
160 //*************************************************************************
161 // TRD_D
162 //*************************************************************************
163 void TRT_D (void) interrupt 16
164 {
165     VISU_TRT_D = 1;
166     if ((T4CON & (1<<7)) != 0)
167     {
168         T4CON &= ~(1<<7);
169         New_CP_1000 = ON;
170     }
171     if ((T4CON & (1<<6)) != 0) T4CON &= ~(1<<6);
172     VISU_TRT_D = 0;
173 }
174
175
176
177
178
179
180
```

Pour faciliter votre travail, cette feuille peut être détachée.

Elle n'est pas à rendre à la fin de l'examen.

```
181
182  //*****************************************************************************
183  // void Gestion Etat LED(unsigned char CP)
184  //*****************************************************************************
185  void Gestion_Etat_LED(unsigned char CP)
186   {
187       switch (CP)
188      {
189        case 0:
190        case 1:
191        case 2:
192        case 3:
193        case 4:
194          Etat_LED_Verte = CLIGNO;
195          Etat_LED_Orange = LED_OFF;
196          Etat_LED_Rouge = LED_OFF;
197        break;
198        case 5:
199          Etat_LED_Verte = LED_ON;
200          Etat_LED_Orange = LED_OFF;
201          Etat_LED_Rouge = LED_OFF;
202        break;
203        case 6:
204          Etat_LED_Verte = LED_OFF;
205          Etat_LED_Orange = CLIGNO;
206          Etat_LED_Rouge = LED_OFF;
207        break;
208        case 7:
209          Etat_LED_Verte = LED_OFF;
210          Etat_LED_Orange = LED_ON;
211          Etat_LED_Rouge = LED_OFF;
212        break;
213        case 8:
214          Etat_LED_Verte = LED_OFF;
215          Etat_LED_Orange = LED_OFF;
216          Etat_LED_Rouge = CLIGNO;
217        break;
218        case 9:
219          Etat_LED_Verte = LED_OFF;
220          Etat_LED_Orange = LED_OFF;
221          Etat_LED_Rouge = LED_ON;
222        break;
223        default:
224          Etat_LED_Verte = LED_OFF;
225          Etat_LED_Orange = LED_OFF;
226          Etat_LED_Rouge = LED_OFF;
227        break;
228     }
229  }
230  //*****************************************************************************
231  // void Gestion_Cde_LED()
232  //*****************************************************************************
233   void Gestion_Cde_LED()
234  {
235      if (Etat_LED_Verte == CLIGNO) LED_Verte = !LED_Verte;
236      else if (Etat_LED_Verte == LED_ON) LED_Verte = LED_ON;
237      else LED_Verte = LED_OFF;
238      if (Etat_LED_Orange == CLIGNO) LED_Orange = !LED_Orange;
239      else if (Etat_LED_Orange == LED_ON) LED_Orange = LED_ON;
240      else LED_Orange = LED_OFF;
241      if (Etat_LED_Rouge == CLIGNO) LED_Rouge = !LED_Rouge;
242      else if (Etat_LED_Rouge == LED_ON) LED_Rouge = LED_ON;
243      else LED_Rouge = LED_OFF;
244  }
245
```

Pour faciliter votre travail, cette feuille peut être détachée.

Elle n'est pas à rendre à la fin de l'examen.