Université de Jijel **Avril 2020**
Faculté des Sciences et de la Technologie
Département d'Automatique

Systèmes embarqués

Tp n°1

I. Objectifs du TP

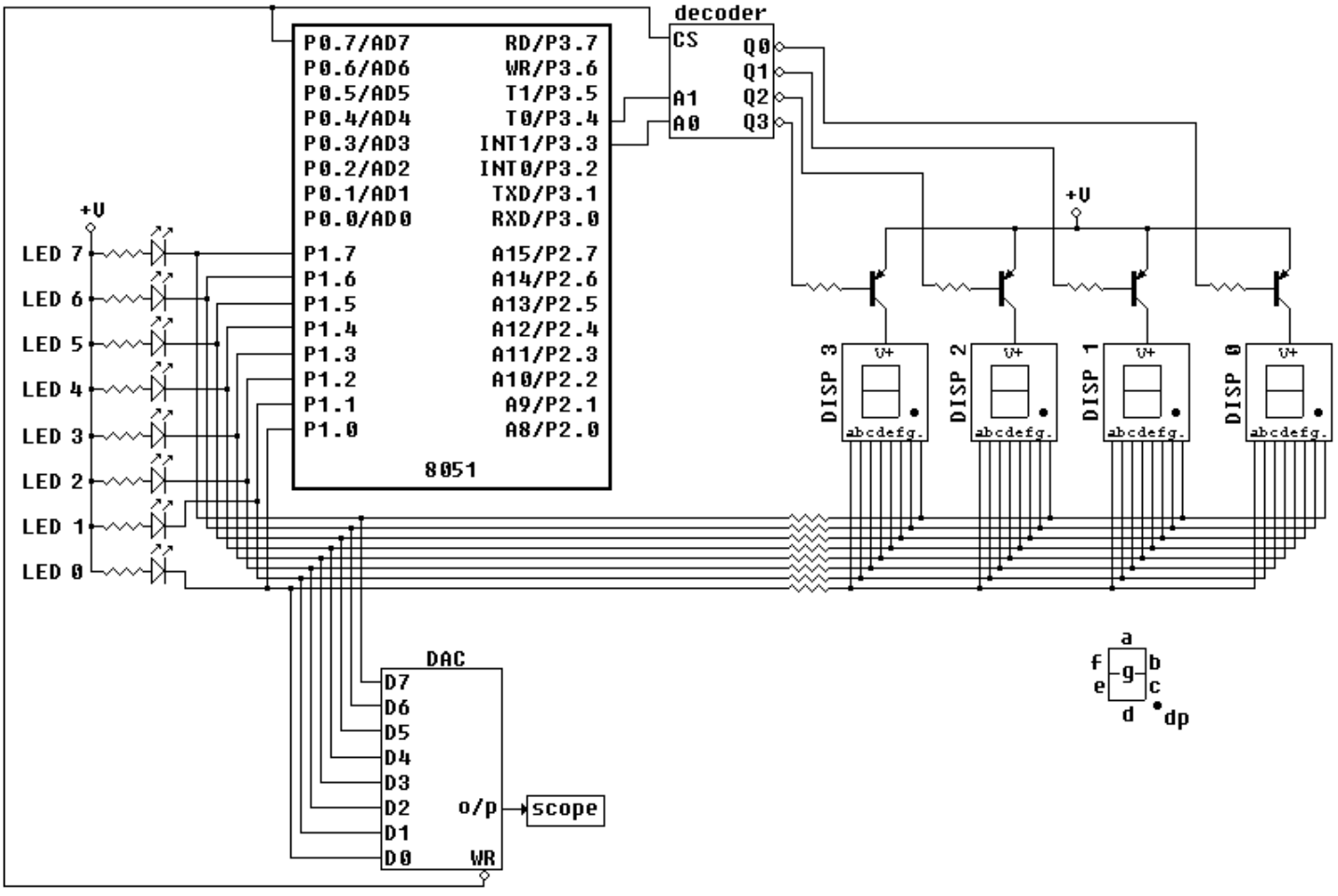
Familiariser l'étudiant aux concepts et aux principes de la programmation assembleur 8051 en utilisant le logiciel de simulation EdSim51.

II. Préliminaires

a) Présentation d'EdSim51

Edsim51 est un logiciel didactique de simulation du microcontrôleur 8051. Il dispose d'un éditeur de texte pour écrire le code assembleur ainsi que quelques périphériques virtuels (convertisseur N/A, afficheurs 7 segments, clavier 4×3…etc).

b) Câblage Afficheur 7 segments sur 8051



Pour activer les afficheurs on utilise P3.4 et P3.3.

c) Exemple

Vérifier le contenu des registres et des mémoires code et données après chacune des instructions suivantes:

```
start:
SETB P3.3             met le bit P3.3 à 1
SETB P3.4
MOV P1, #11111001B    envoie le nombre binaire 11111001  au port P1
MOV P1, #0FFH
CLR P3.3              met le bit P3.3 à zéro
MOV P1, #10100100B
MOV P1, #0FFH
CLR P3.4
SETB P3.3
MOV P1, #10110000B
MOV P1, #0FFH
CLR P3.3
MOV P1, #10011001B
MOV P1, #0FFH
JMP start
```

III. Travail à faire

1. Ecrire le programme assembleur qui affiche sur les 4 afficheurs le mot anglais HELLO (on vous suggère d'utiliser un seul afficheur pour les 2 L).
2. Ecrire le programme assembleur qui affiche les chiffres de 0 à 9.

Université de Jijel **Avril 2020**
Faculté des Sciences et de la Technologie
Département d'Automatique

Systèmes embarqués

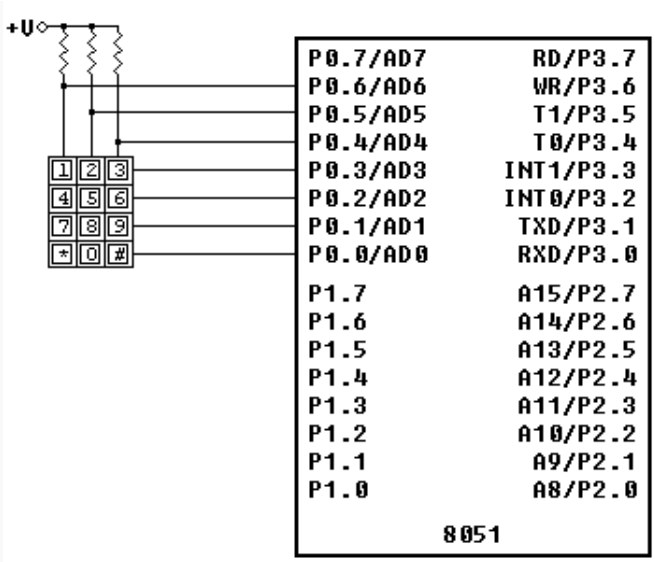
TP n°2

I. Objectifs du TP

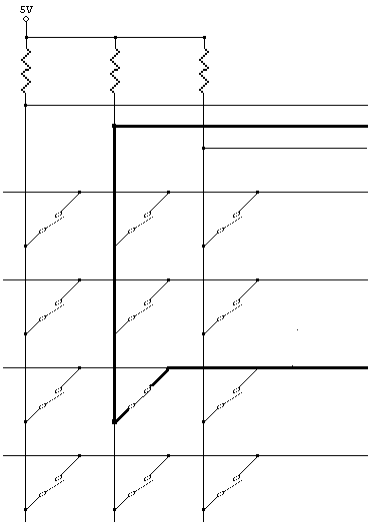
Gestion d'un clavier matriciel de 12 touches par le microcontrôleur 8051.

II. Préliminaires

Esim51 dispose d'un clavier virtuel formé d'une grille de 3 sorties et 4 entrées, organisées respectivement en colonnes et en lignes, avec une touche à chaque intersection. Cela prend sept broches du port d'entrée-sortie P0 (de P0.0 à P0.6).



Pour effectuer la scrutation du clavier et lire la fermeture éventuelle d'une touche, on balaye le clavier en mettant chacune des 4 entrées P0.0 P0.1 P0.2 et P0.3 à l'état bat à tour de rôle et l'on vérifie si une sortie (P0.4 P0.5 P0.6) est à l'état bas (donc une touche est enfoncée). Dans ce cas, pour pouvoir identifier la touche enfoncée on ne peut fermer qu'une seule touche à la fois.



Exemple : Le programme suivant balaye les 2 lignes P0.0 et P0.1 et met F0 à 1 si une touche est enfoncée.

```
MOV R0, #0
Mov R1, #0                ; initialisation de R0 et R1 à 0
; Balayage de la ligne P0.0
CLR P0.0                  ; mise à 0 de P0.0
CALL Scan                 ; appel du sous-programme Scan
JB F0, Fin                ; saut vers l'étiquette Fin si F0=1

; Balayage de la ligne P0.1
INC R1
SETB P0.0                 ; remise à 1 de P0.0
CLR P0.1                  ; mise à 0 de P0.1
CALL Scan                 ; Appel du sous-programme Scan
JB F0, Fin                ; saut vers l'étiquette Fin si F0=1

Scan:
JNB P0.4, Key             ; saut vers Key si P0.4=0
INC R0
JNB P0.5, Key             ; saut vers Key si P0.5=0
INC R0
JNB P0.6, Key             ; saut vers Key si P0.6=0
INC R0
RET

Key:
SETB F0
RET
Fin :
```

III. Travail à faire

a) Questions

1) De combien de broches entrées-sorties on a besoin pour gérer un clavier de 4*5 touches ?
2) Peut-on effectuer l'opération de balayage en mettant P0.4 P0.5 P0.6 à l'état bat à tour de rôle et en scrutant P0.0 P0.1 P0.2 et P0.3 ?
3) Que se passe-t-il si plus qu'une touche est enfoncée sur la même colonne ?
4) Que se passe-t-il si plus qu'une touche est enfoncée sur la même ligne ?
5) Combien y a-t-il de sous-progs dans l'exemple précédent ?
6) Vers quelles instructions le retour est-il fait par les deux RET de l'exemple ?

b) Programmation

1- Compléter le programme de l'exemple pour que toutes les lignes du clavier soient balayées en permanence.

2- Ecrire le code supplémentaire qui affiche le symbole de la clé enfoncée sur l'un des afficheurs 7 segments. (Les symboles # et * ne peuvent être affichés sur un afficheur 7 segments, mais certains caractères spéciaux peuvent être inventés et affichés à leur place).

IV. Travail supplémentaire (facultatif)

Ecrire le programme qui gère le clavier même si plusieurs touches sont enfoncées simultanément.

Université de Jijel **Avril 2020**
Faculté des Sciences et de la Technologie
Département d'Automatique
Systèmes embarqués

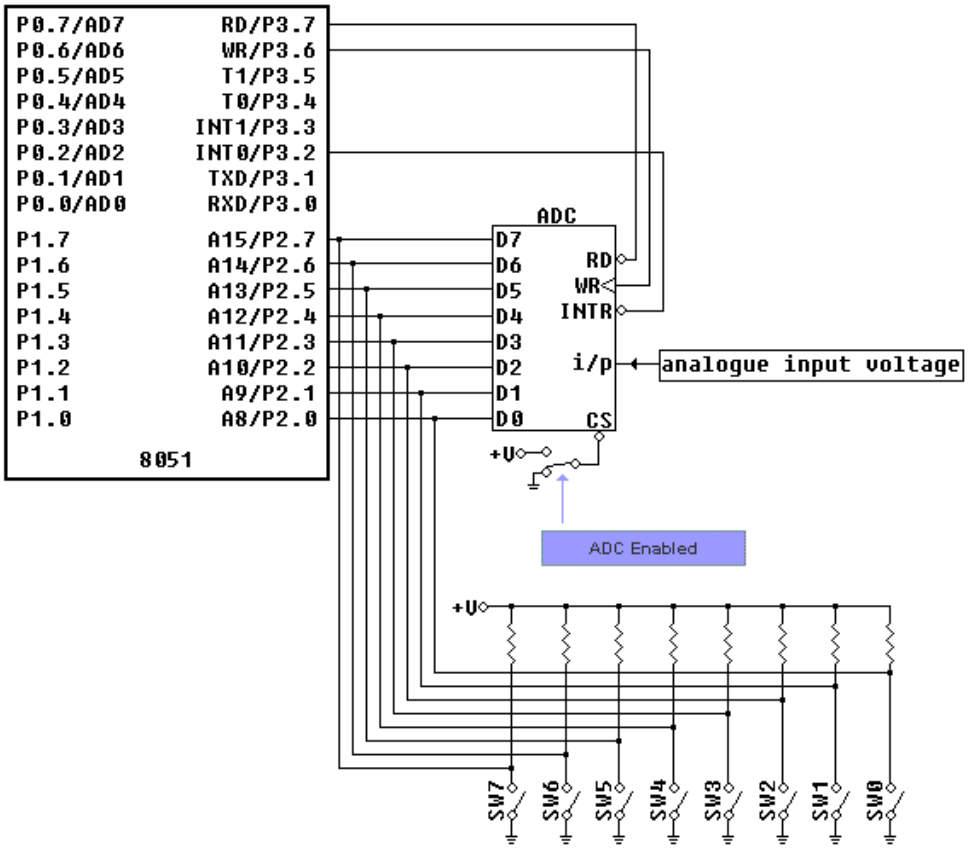
TP n°3

I. Objectifs du TP

Acquisition d'une tension analogique via un convertisseur analogique/numérique à l'aide du microcontrôleur 8051.

II. Préliminaires

Un convertisseur analogique - numérique transforme une grandeur physique (tension, courant) en une valeur numérique. La correspondance entre la grandeur analogique et la valeur numérique se fait selon les limites basses et les limites hautes des deux valeurs. Par exemple, pour un CAN 10 bits ayant une plage de tension d'entrée analogique limitée entre 0 V et +5 V correspond la plage de valeurs numériques comprise entre 0 et 1023.



EdSim51 dispose d'une source de tension commandée et d'un convertisseur A/N virtuel possédant :

- une entrée analogique i/p (tension),
- 8 sorties numériques D0-D7 connectées au port P2,
- une entrée WR (connectée à P3.6) qui permet d'initialiser la conversion. Elle doit être à l'état bas puis passe à l'état haut pour commencer la conversion (fonctionne avec un front mentant),
- une entrée RD (connectée à P3.7) qui doit être à l'état bas pour que la grandeur convertie apparaisse sur les sorties D0-D7,

- une sortie INTR qui passe à l'état bas lorsque la conversion A/N est terminée (reste à l'état bas jusqu'à ce que une autre conversion soit initialisée). Cette sortie est connectée à l'entrée d'interruption externe INT0.

Exemple

Dans cet exemple, le 8051 lit un échantillon à partir du CAN tous les 20us. ceci est réalisé par le biais du timer0 qui interrompre le programme principal tous les 20 us et active la conversion A/N. Lorsque la conversion est terminée, la ligne INTR du CAN passe au niveau bas. Cette ligne est connectée à ligne d'interruption externe INTR0 du 8051 ce qui déclenche l'interruption externe 0.

```
ORG 0    ;adresse 00h
JMP principal    ; saut à principal
ORG 3   ; vecteur de l'interruption INTR0
JMP ext ; saut à ext
ORG 0BH ; vecteur d'interruption du timer0
JMP timer0 ; saut à timer 0
ORG 30H ; prog principal commence à 30H
principal :
SETB IT0 ; met l'interruption externe0 sur front descendant
SETB EX0 ; valide l'interruption externe 0
MOV TMOD, #2
MOV TH0, #-20
MOV TL0, #-20
SETB TR0    ;demarre le timer0
SETB ET0   ; valide l'interruption du timer0
SETB EA    ; validation globale des interruptions
JMP $ ;ne rien faire
timer0 :
CLR P3.6
SETB P3.6 ; ces deux instructions servent à initialiser la conversion
RETI
ext:
CLR P3.7 ; permet la lecture de la grandeur convertie
MOV A, P2
SETB P3.7 ;
RETI
```

III. Travail à faire

a) Questions

1) Quelle est la plage des valeurs numériques pour ce CAN ?
2) Supposant que l'ensemble 8051/CAN fait partie d'un asservissement numérique et que la tension d'entrée analogique du CAN provient du capteur d'un système continu. Quelle est, dans cas de l'exemple précédent, la période d'échantillonnage de l'asservissement ? peut-on réduire librement cette période d'échantillonnage ? Justifiez votre réponse.
3) Vers quelles instructions le retour est fait par les deux RETI de l'exemple?
4) Que se passe-t-il si on omit l'instruction `SETB P3.7` dans la routine `ext` ?
5) Quel est le rôle des instructions :
```
MOV TMOD, #2
MOV TH0, #-20
MOV TL0, #-20
```

b) Programmation

1) Compléter le programme de l'exemple pour stocker les échantillons convertis dans la RAM (on vous suggère de commencer à l'adresse 30h).
2) Ecrire le code supplémentaire qui affiche la valeur convertie sur 2 afficheurs 7 segments.

Université de Jijel **Mai 2020**
Faculté des Sciences et de la Technologie
Département d'Automatique
Systèmes embarqués

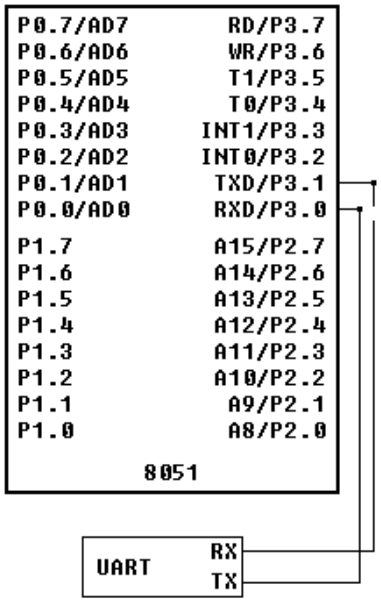
TP n°4

I. Objectifs du TP

Réception et transmission de données via le port série du microcontrôleur 8051.

II. Préliminaires

Edsim dispose d'un UART externe connecté aux ports TxD et RxD du 8051 comme le montre la figure ci-dessous.



Pour envoyer un caractère, il suffit de l'écrire dans le registre SBUF; cela entreprend automatiquement la procédure d'envoi du caractère. Lorsque tous les bits sont envoyés, le drapeau TI passe à un.

Lors de la réception d'un caractère, la procédure est enclenchée par la détection d'un franc descendant à la broche RxD. Lorsque tous les bits sont présents, le drapeau RI passe à un.

III. Travail à faire

Soit le programme suivant :

```
CLR SM0
SETB SM1
ORL PCON,#80h

MOV TMOD, #20H
MOV TH1, #243
MOV TL1, #243

MOV 30H, #'H'
MOV 31H, #'E'
MOV 32H, #'L'
```

```
MOV 33H, #'L'
MOV 34H, #'O'
MOV 35H, #0
MOV R0, #30H
SETB TR1
loop:
MOV A, @R0
JZ fin
MOV SBUF, A
INC R0
JNB TI, $
CLR TI
JMP loop
fin:
JMP $
```

a) Questions

1- Quel est le mode port série utilisé ?

2- Quel est l'effet de l'instruction `ORL PCON,#80h` ?

3- Calculer le Baud rate.

4- Que se passe-t-il si un caractère est envoyé avant que TI passe à un ?

5- Combien y a-t-il de sous-progs dans cet exemple ?

b) Programmation

1- Refaire le programme précédent en utilisant l'interruption port série.

2- Ecrire le programme qui reçoit en permanence des caractères sur RxD avec un baud rate de 4800, et les arrange dans la mémoire à partir de l'adresse 30H.