Ministère de l'Enseignement Supérieur de la Recherche Scientifique et de la Technologie

Direction des Etudes Technologiques

Institut Supérieur des Etudes Technologiques de Sousse

Département Génie Electrique

TRAVAUX PRATIQUES

ATELIER ELECTRONIQUE

Niveau : L2- semestre 3, Tronc commun

Elaborés par Mrs:

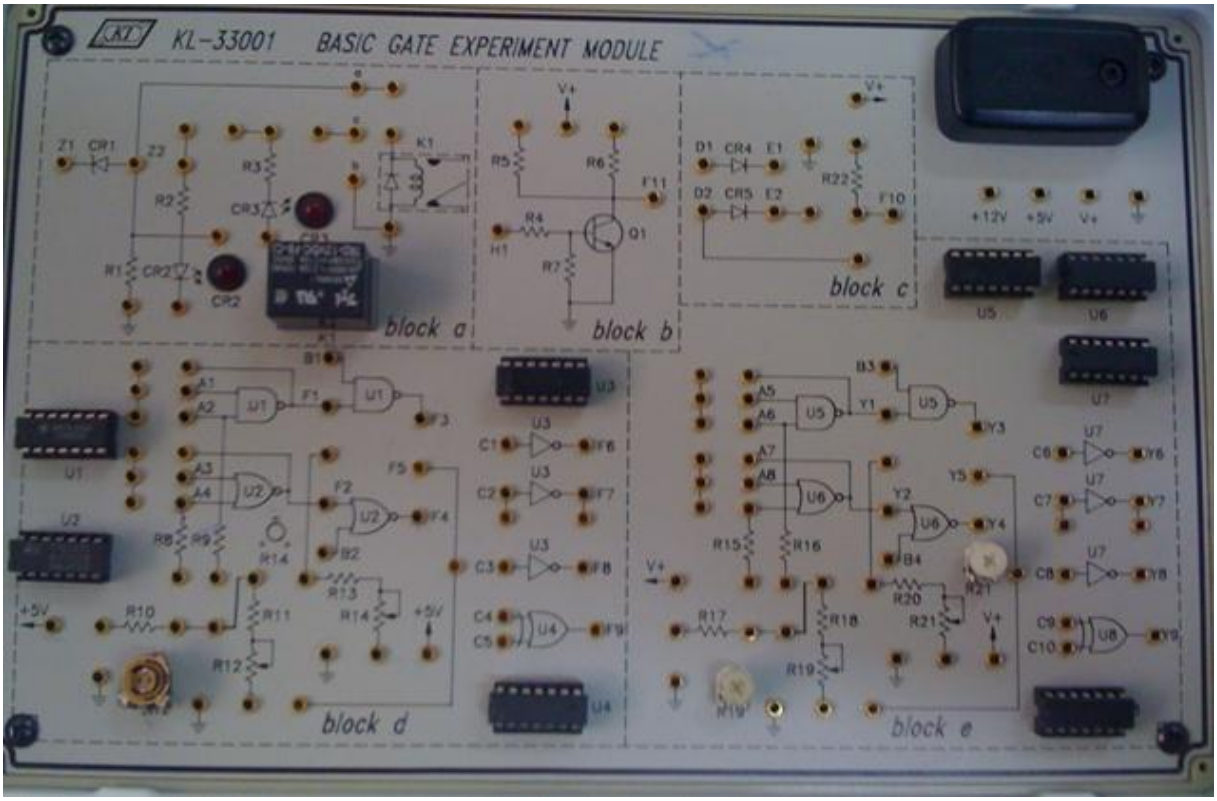
-GRIRI Faouzi

- TLILI Kais

Année Universitaire 2009/2010

TRAVAUX PRATIQUES

TP 1 : Les familles logiques TTL et CMOS



Module utilisé 33001

TP1 :

Les familles logiques TTL et CMOS

Objectifs :

- Etudier les caractéristiques courant tension des familles logiques TTL /CMOS.
- Déterminer le temps de retard pour les familles logiques TTL /CMOS.
- Comprendre la technique d'interfaçage TTL –CMOS et CMOS - TTL

Matériels utilisés : KL-31001, KL-33001, multimètre, oscilloscope numérique.

Manipulation :

A- Caractéristiques courant et tension pour les familles logique TTL/ CMOS

I- En utilisant le bloc d du module KL-31001 :

1- Insérer les connections comme indiqué dans la figure (1) :

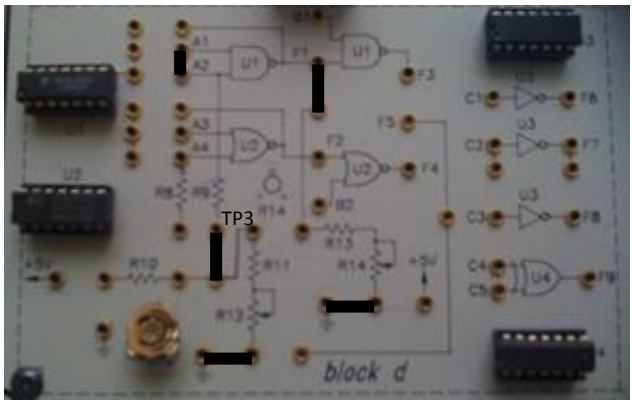


Figure 1 : Block d

2- En utilisant la porte logique U1a, ajuster le potentiomètre R_{12} pour avoir $V_{IL} = 0.8V$.

2-1- Mesurer la tension V_I entre A_2 et TP_3

2-2- En déduire le courant $I_{IL} = V_I / 100$ en (mA)

2-3- Ajuster R_{14} pour trouver V_{OH} et V_{OL}

3- Ajuster R_{14} pour avoir $V_{OH} = 2.4V$, débrancher la connexion entre R_{14} et la masse puis mesurer I_{OH} (mA).

4- Pour déterminer les caractéristiques courant et tension de l'U2a débrancher la connexion R_9 - TP3 d'une part et $F_1 – R_{13}$ d'autre part, et les placer respectivement entre R_8 - TP3 et $F_2 – R_{13}$.

Ajuster le potentiomètre R_{12} pour avoir $V_{IL} = 0.8V$.

4-1- Mesurer la tension V_2 entre A_3 et TP_3.

4-2- En déduire le courant $I_{IL} = V_2 / 100$ en (mA).

5- Ajuster R_{14} pour avoir $V_{OH} = 2.4V$, débrancher la connexion entre R_{14} et la masse puis mesurer I_{OH} (mA).

6- Insérer les connections comme indiqué dans la figure (2) :

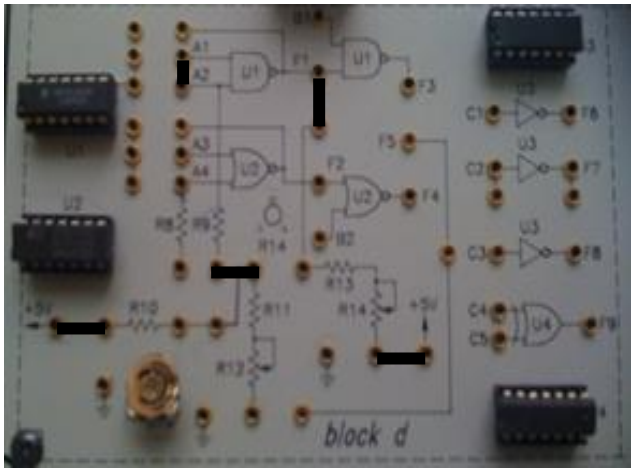


Figure 2 : Block d

6-1- Ajuster R_{12} pour avoir une tension $V_{IH} = 2V$ entre TP3 et la masse.

6-2- Mesurer la tension V_2 entre A_2 et TP3.

6-3- En déduire le courant $I_{IH} = V_2 / 100$ en (mA).

7- Ajuster R_{12} et mesurer V_{OL} et V_{OH}.

7-1- Ajuster R_{14} pour avoir une tension V_{OL} = 0.4V, débrancher la connexion entre R_{14} et V_{CC} puis mesurer I_{OL} (mA) entre R_{14} et V_{CC}.

8- Remplacer les connexions $R_9 – R_{10}$ et $F_1 – R_{13}$ respectivement par R_8 - R_{10} et $F_2 – R_{13}$. Ajuster R_{12} pour avoir une tension V_{IH} = 2V

8-1- Mesurer la tension V_2 entre A_2 et TP_3.

8-2- En déduire $I_{IH} = V_2 /100$ en (mA).

9- Ajuster R_{14} pour avoir V_{OL} = 0.4V, débrancher la connexion entre R_{14} et la masse puis mesurer I_{OL} (mA).

II- En utilisant le bloc e du module KL-31001 :

1-1- En utilisant la section U5a, réaliser la connexion de la figure (3) en branchant V_{DD} à +12V.

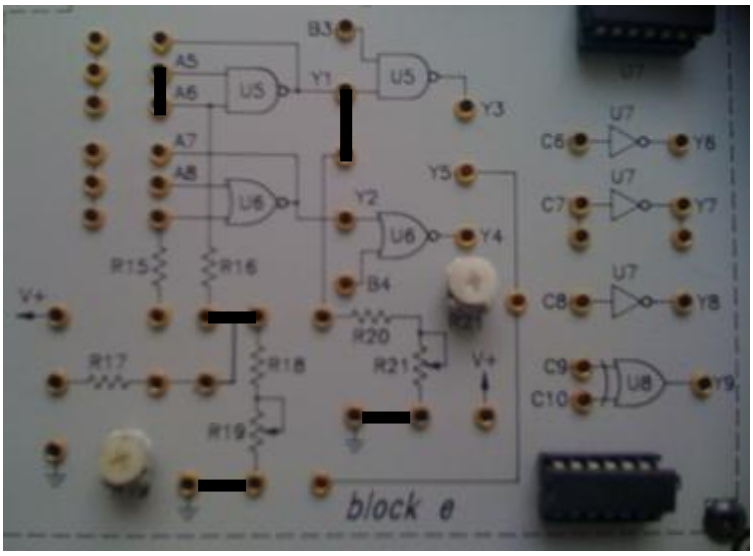


Figure3 : Block e

1-2- Ajuster R_{19} pour avoir une tension V_{IL} = 3.6V entre TP3 et la masse.

1-3- Ajuster R_{21} et mesurer la tension minimale V_{min} et maximale V_{max} au point Y_1.

1-4- Mesurer la tension V_3 entre R_{19} et TP3

1-5- En déduire $I_{IL} = V_3 / 100$ en (mA).

1-6- Ajuster R_{21} pour avoir une tension V_{OH} = 10.8V entre Y_1 et la masse.

1-7- Débrancher la connexion entre R_{21} et la masse, mesurer le courant I_{OL} (mA).

2-1- Ajuster R_{19} pour avoir une tension V_{IH} = 8.4V entre TP3 et la masse.

2-2- Débrancher la connexion entre R_{21} et la masse, la placer entre R_{21} et V_{DD}

2-3- Ajuster R_{21}, mesurer la tension minimale V_{min} et maximale V_{max} au point Y_1.

2-4- Mesurer la tension V_4 entre A_5 et TP3

2-5- En déduire $I_{IH} = V_4 / 100$ en (mA).

2-6- Débrancher la connexion entre R_{20} et Y_1, la placer entre R_{20} et Y_2.

2-7- Ajuster R_{21} pour avoir une tension V_{OL} = 1.2V entre Y_2 et la masse.

2-8- Mesurer le courant I_{OL}(mA) entre R_{21} et V_{CC}.

B- Mesure des temps de retard pour les familles logique TTL/ CMOS :

I- En utilisant le bloc d du module KL-31001 :

1- Insérer les connections comme indiqué dans la figure (4) :

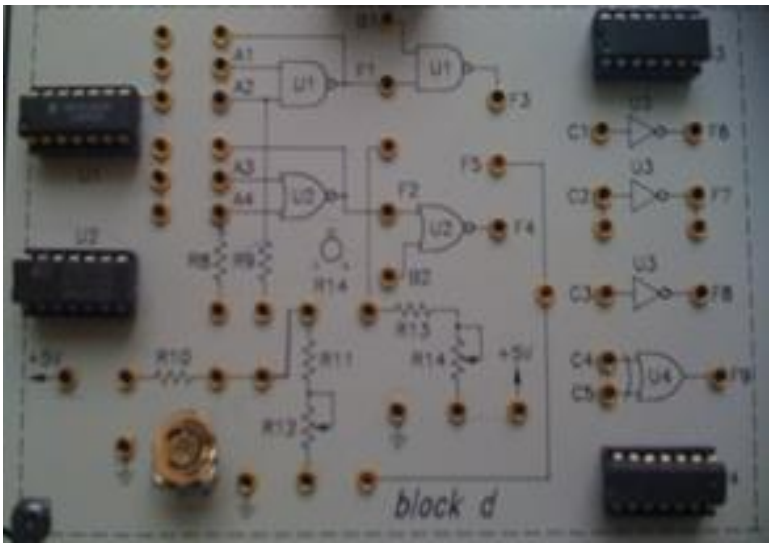


Figure 4 : Block d

2- En utilisant les portes logiques U1a et U1b, brancher l'entrée A_1 au générateur d'horloge à sortie TTL, régler la fréquence à 100KHz.

2-1- Relever les oscillogrammes de A1, F_1, F_3 avec l'oscilloscope.

2-2- Déterminer les temps de retard de F_1, F_3.

3- En utilisant les portes logiques U2a et U2b, brancher l'entrée A_3 au générateur d'horloge à sortie TTL, régler la fréquence à 100KHz.

3-1- Relever les oscillogrammes de A_3 , F_2, F_4 avec l'oscilloscope.

3-2- Déterminer les temps de retard de F_2, F_4.

4- En utilisant les portes logiques U3c et U3b, brancher l'entrée C_1 au générateur d'horloge à sortie TTL, régler la fréquence à 1MHz.

4-1- Relever les oscillogrammes de C_4, F_6, F_7.

4-2- Déterminer les temps de retard de F_6, F_7.

III- En utilisant le bloc e du module KL-31001 :

1- Insérer les connections comme indiqué dans la figure (5) et mettre V_{DD} à +12V :

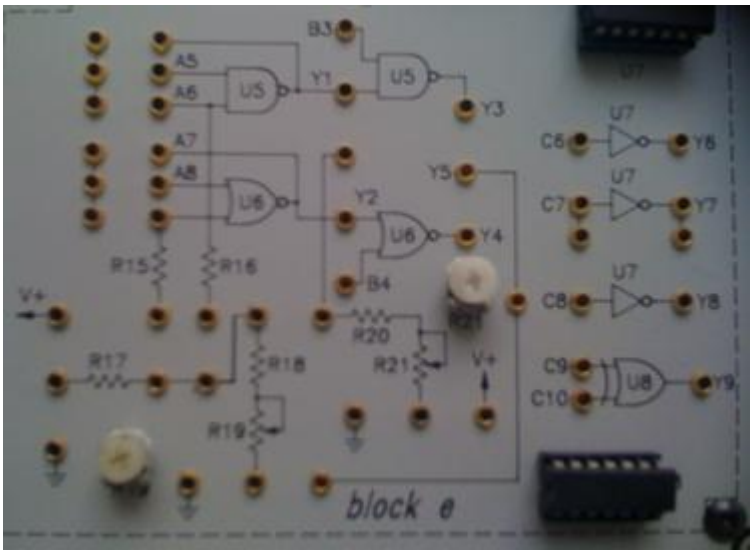


Figure 5 : Block e

2- En utilisant les portes logiques U5a et U5b, brancher l'entrée A_5 au générateur d'horloge à sortie CMOS, régler la fréquence à 100KHz.

2-1- Relever les oscillogrammes de A_5, Y_1, Y_3 avec l'oscilloscope.

2-2- Déterminer les temps de retard de Y_1, Y_3.

3- En utilisant les portes logiques U6a et U6b, brancher l'entrée A_7 au générateur d'horloge à sortie CMOS, régler la fréquence à 100KHz.

3-1- Relever les oscillogrammes de A_7 , Y_2, Y_4 avec l'oscilloscope.

3-2- Déterminer les temps de retard de Y_2, Y_4.

4- En utilisant les portes logiques U7b et U7c, brancher l'entrée C_6 au générateur d'horloge à sortie CMOS, régler la fréquence à 100KHz.

4-1- Relever les oscillogrammes de C_6, Y_6, Y_7.

4-2- Déterminer les temps de retard de Y_6, Y_7.

C- Interfaçage TTL/ CMOS :

1- Interfaçage TTL - CMOS

En utilisant les blocs d et e on va piloter une porte CMOS par une TTL

Insérer les connexions comme dans la figure (6), la porte U1a qu'on va utiliser est de la série TTL standard.

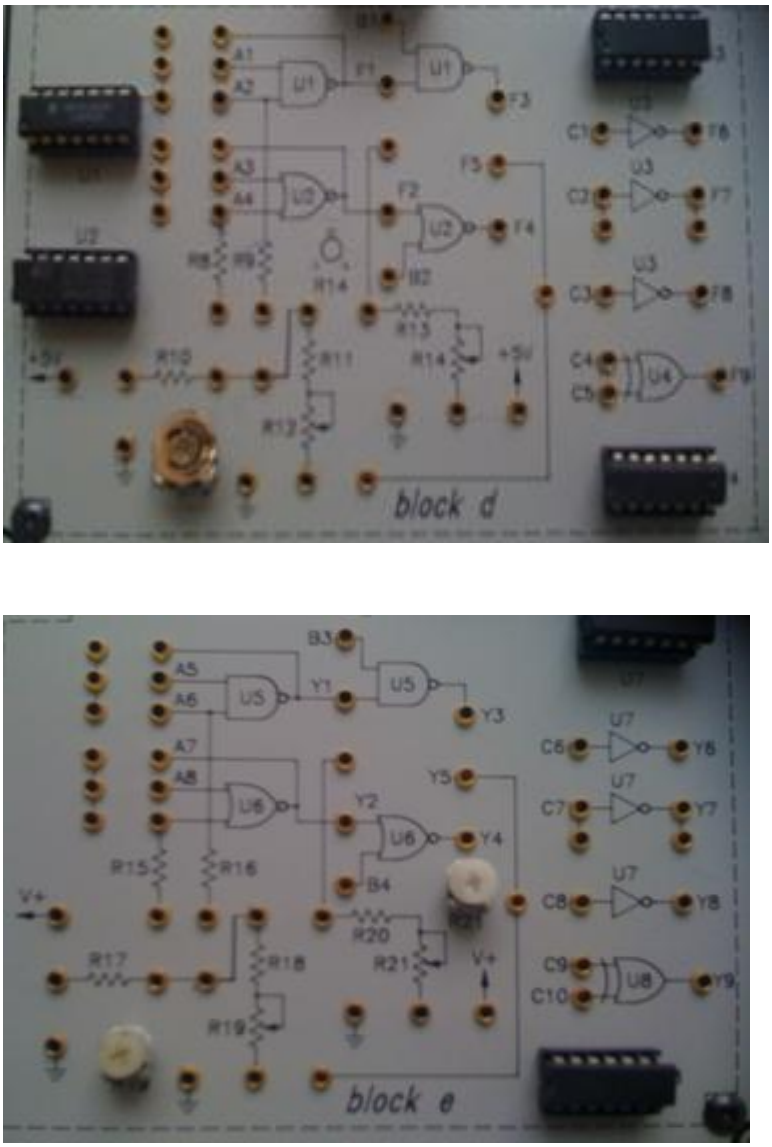


Figure 6 : Block d et e

1-1- En utilisant un multimètre ajuster R_{14} à 2.2KΩ.

1-2- Alimenter les deux portes logiques par +5V, brancher la sortie F_1 à l'entrée de U5a, brancher l'entée A_1 à la sortie TTL de switcher SW0 et mesurer les tensions aux points A_1 , F_1 , A_5 , Y_1 .

A_1	F_1	A_5	Y_1
0			
1			

1-3- Brancher F_1 à R_{13}, V_{CC} à R_{14} et mesurer les tensions aux points A_1 , F_1 , A_5 , Y_1 .

A_1	F_1	A_5	Y_1
0			
1			

Interfaçage CMOS- TTL

En utilisant les blocs d et e on va piloter une porte CMOS par une TTL

2-1- En utilisant U7a, U7b et U7c du bloc e de module KL 31001, insérer les connexions comme dans la figure ().

2-2- Brancher la sortie Y_8 à l'entrée A_1 de U1a et C_8 à la sortie TTL du switcher SW1, puis mesurer les tensions aux points Y_8 , A_1 , F_1.

C_8	Y_8	A_1	F_1
0			
1			

2-3- Brancher C_6, C_7, C_8 ensemble et répéter les mesures.

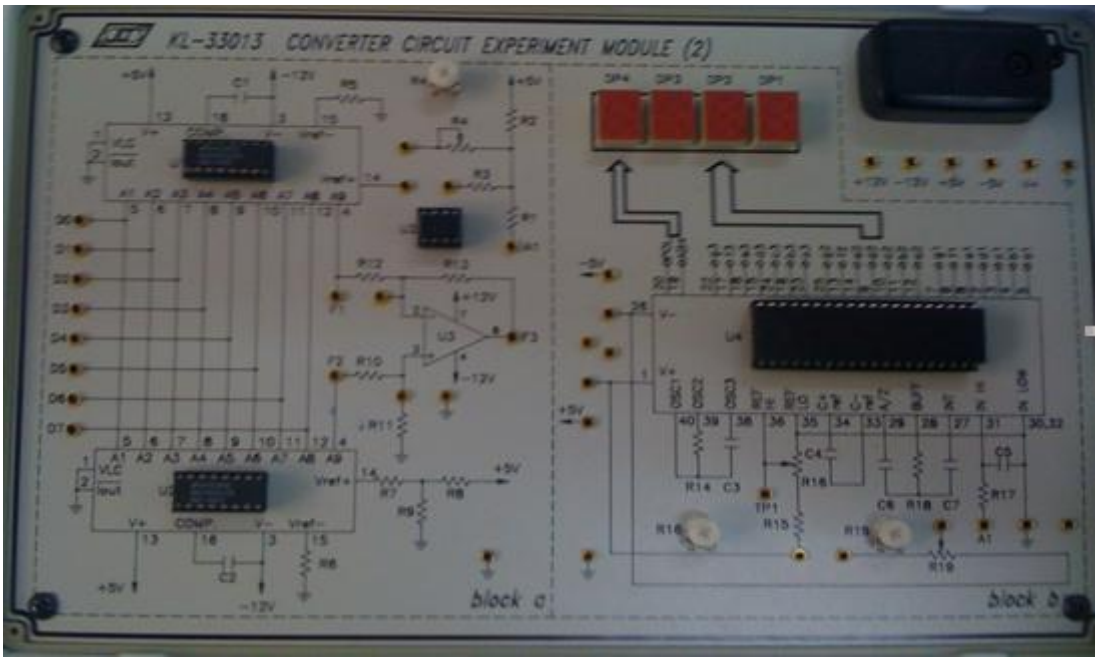
C_8	Y_8	A_1	F_1
0			
1			

2-4- Brancher Y_8 aux entrées C_1, C_2, C_3 de U3a, U3b et U3c, brancher ensemble les entrées C_1, C_2, C_3 d'une part et les sorties F_6 , F_7 , F_8 d'autre part et répéter les mesures

C_8	Y_8	A_1	F_1
0			
1			

TRAVAUX PRATIQUES

TP2 : Les Convertisseurs analogiques numériques et numériques analogiques



Module utilisé 33013

TP2 :

Les Convertisseurs analogiques-numériques et numériques-analogiques

I- Objectifs :

- Comprendre les concepts de base et les théories des convertisseurs numérique-analogique
- Comprendre les concepts de base et les théories des convertisseurs analogique-numérique

II- Etude théorique :

II-1- Convertisseur numérique- analogique

Soit les montages sur la figure (1) des convertisseurs numériques-analogiques R-2R, à base d'AOP et à source du courant.

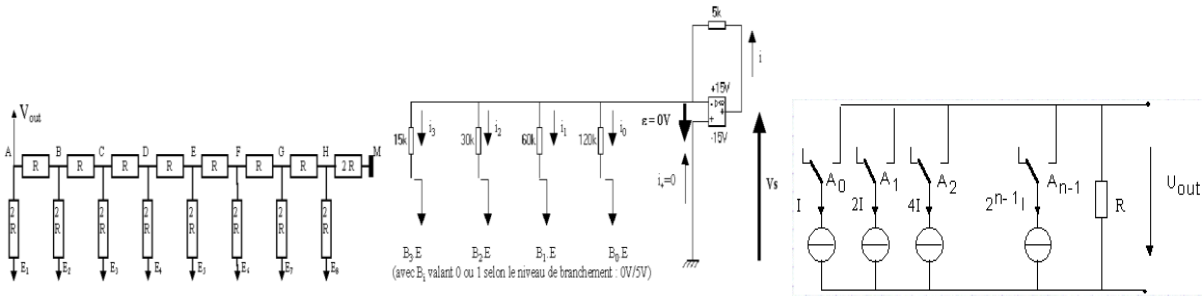


Figure 1 : Convertisseurs numériques.

II-1-1- quel est le nombre de bits pour chaque montage du convertisseur numérique analogique ?

II-1-2- Déterminer l'expression de V_0 pour les différents montages.

Soit le circuit convertisseur numérique-analogique DAC0808, donné par la figure suivante :

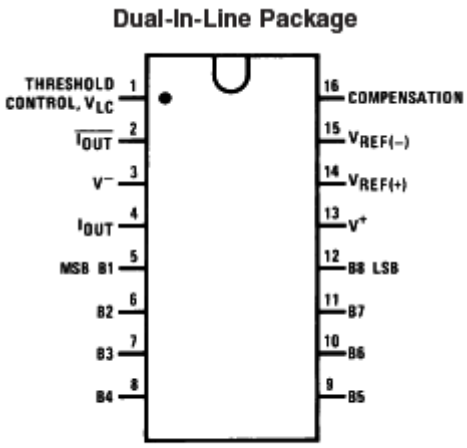


Figure 2 : DAC0808.

II-1-3- Quel est le nombre de bit pour le convertisseur : DAC0808.

II-1-4- Quel est le type de la sortie du convertisseur : DAC0808.

II-1-5- Quelle est la résolution du convertisseur DAC0808

II-2- Convertisseur analogique- numérique

Un convertisseur analogique numérique (ADC) est l'inverse d'un convertisseur numérique analogique (DAC), il possède une entrée analogique et une sortie binaire multiple. Le nombre des bits de sorties déterminent la résolution d'un ADC (l'ADC commercial possède de 4 à 20 bits de sortie).

Ils existent plusieurs types des ADC on trouve :

- ADC à rampe numérique
- ADC double rampe
- Convertisseur à approximation successive
- ADC à comparateur flash

- ADC à rampe numérique : C'est le plus simple et le moins utilisé des ADC, la tension à convertir est appliquée à l'entrée d'un comparateur et l'autre entrée à la sortie du DAC (Vout).
- Convertisseur à double rampe : Ce convertisseur converti une tension d'entrée inconnue dans un temps bien déterminé, il est constitué d'un intégrateur, comparateur, circuit de contrôle et un compteur binaire ou BCD, le cycle de conversion est donné par la figure suivante.

- Convertisseur à approximation successive : C'est une version améliorée et rapide du convertisseur à rampe numérique, seulement le circuit de commande du DAC est remplacé par un circuit séquentiel appelé « registre à approximation successive.
- Convertisseur à comparateur FLASH : C'est le convertisseur le plus rapide et le plus couteux.

II-2-1- Donner le principe de conversion de chaque type des ADC

L'ADC0804 donné par la figure suivante représente un convertisseur à approximation successive, il convertit une tension analogique de 0 à +5V sur 8 bits

III- Manipulation :

Matériel nécessaire : module KL-31001 ; module KL-33013 ; multimètre

III-1- Convertisseur numérique analogique (DAC) unipolaire :

III-1-1- Insérer les connections comme montre la figure () pour construire un DAC unipolaire

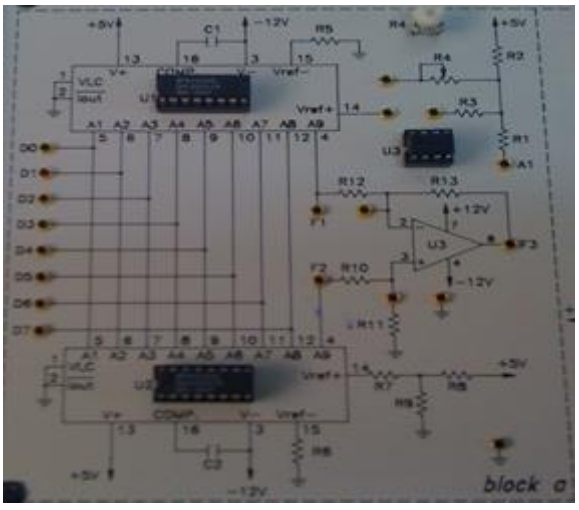


Figure 3 : Convertisseur numérique analogique unipolaire.

III-1-2- Compléter les connections des entrées

D7 à DIP Switch 1.7 D4 à DIP Switch 1.4 D1 à DIP Switch 1.1

D6 à DIP Switch 1.6 D3 à DIP Switch 1.3 D0 à DIP Switch 1.0

D5 à DIP Switch 1.5 D2 à DIP Switch 1.2

Mesurer la tension au point F3 avec le multimètre

III-1-3- Ajuster R4 et mesurer la résistance maximale entre A2 et Vcc

III-1-4- Pour la valeur maximale de résistance entre A2 et Vcc, mettre les entrées à «1 » logique et mesurer V0, $I_{réf}$ est le courant résultant de la division de la tension entre A2 Vcc par la résistance maximale entre A2 Vcc.

III-1-5- En respectant les séquences d'entrées D7 à D0 comme le tableau mesurer les valeurs de V0

D7	D6	D5	D4	D3	D2	D1	D0	V0
0	0	0	0	0	0	0	1	
0	0	0	0	0	0	1	0	
0	0	0	0	0	1	0	0	
0	0	0	0	1	0	0	0	
0	0	0	1	0	0	0	0	
0	0	1	0	0	0	0	0	
0	1	0	0	0	0	0	0	
1	0	0	0	0	0	0	0	
1	1	1	1	1	1	1	1	

III-1-6- Calculer le courant de sortie Iout

III-2- Convertisseur numérique analogique (DAC) bipolaire :

III-2-1- Insérer les connections comme montre la figure (5) pour construire un DAC bipolaire

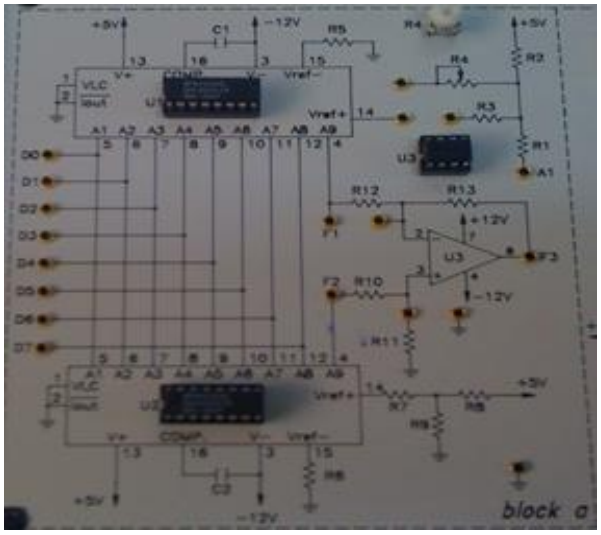


Figure 5: Convertisseur numérique analogique bipolaire.

III-2-2- Appliquer un tension de référence Vréf =(1/2)Vcc =2.5V au Pin14 du circuit U2, utiliser un signal sinusoïdale entre +2.5V et -2.5V comme V_i . Brancher respectivement les entrées D7 D0 au DIP Switch 1.7 à 1.0.

III-2-3- En respectant les séquences d'entrées D7 à D0 comme le tableau mesurer les valeurs de V0, V_i, F1 et F2

D7	D6	D5	D4	D3	D2	D1	D0	V0	F1	F2	V_i
0	0	0	0	1	1	1	1				
0	0	0	1	0	0	0	0				
0	0	1	0	0	0	0	0				
0	1	0	0	0	0	0	0				
1	0	0	0	0	0	0	0				

III-3- Convertisseur analogique numérique 8 bits (ADC):

Matériel nécessaire : module KL-31001 ; module KL-33012/ KL-33013 ; multimètre

III-3-1- Insérer les connections comme la figure (6)

III-3-2- Brancher les sorties D7 à D0 aux indicateurs logiques L1 à L7 et $\overline{INTR}$ à L8, mesurer l'entrée avec le multimètre.

III-3-3- Brancher $\overline{CS}$ et $\overline{RD}$ à la masse, $\overline{WR}$ au Switch SWA $\bar{Q}$ out, Incrémenter la tension d'entrée V_i d'un pas de 0.05V et compléter le tableau. A chaque fois si on incrémente V_i, SWA doit être actionné pour générer une impulsion négative qui valide le changement de la sortie. Une impulsion négative est générée par $\overline{INTR}$ si la sortie change et L_8 s'éteint momentanément

V_i	D7	D6	D5	D4	D3	D2	D1	D0	
0									
0.05									
0.1									
0.15									
0.2									
0.25									
0.3									
0.5									
0.7									
0.9									
1.0									
1.5									
2.0									
2.5									
3.0									
3.5									
4.0									
4.5									
5.0									

III-3-4- Augmenter la capacité de la Pin 4 à 0.1µF, combien du temps L_8 reste éteinte

III-3-5- Est-ce que la conversion est possible si $\overline{CS}$ et $\overline{RD}$ sont à la masse

III-4- Convertisseur analogique numérique 3 ½ digit (ADC):

III-4-1- Insérer les connections comme montrées dans la figure (7)

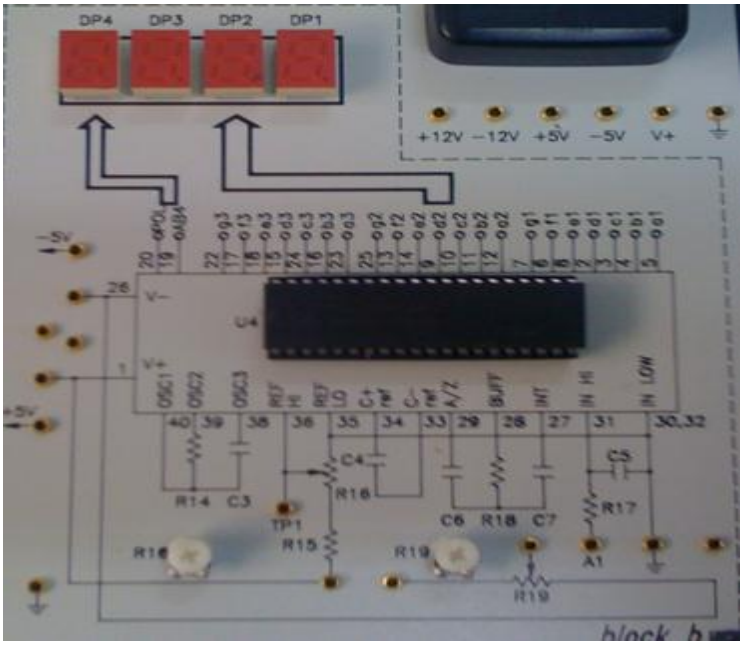


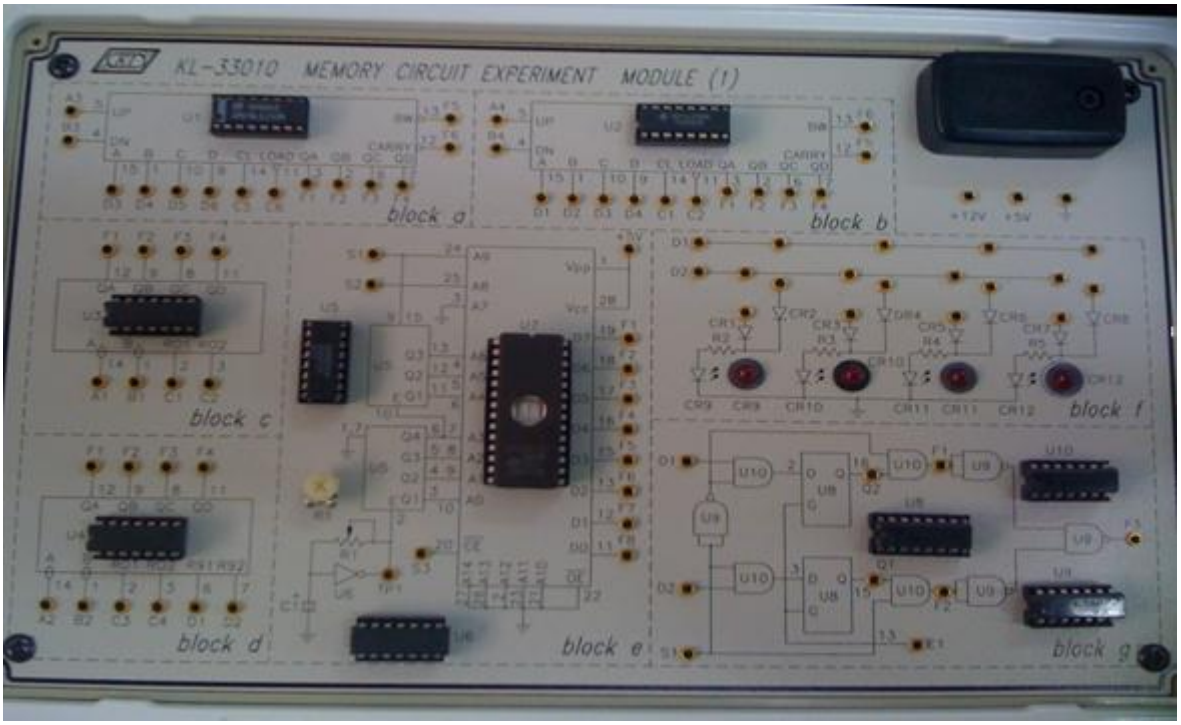
Figure 7: Convertisseur analogique numérique 3 ½ digit.

III-4-1- Brancher V- à -5V et V+ à +5V, régler TP1=1V, compléter le tableau ci-dessous

VA1	Afficheur
-3V	
-2.8V	
-2V	
-1.8V	
-1.65V	
-1.5V	
-1.43V	
-1.27V	
-1V	
-0.6V	
0	
1V	
1.8V	
1.97V	
2V	
2.23V	
2.38V	
3V	

TP3 :

LES MEMOIRES



Module utilisé 33010

TP3 :

LES MEMOIRES

Objectifs :

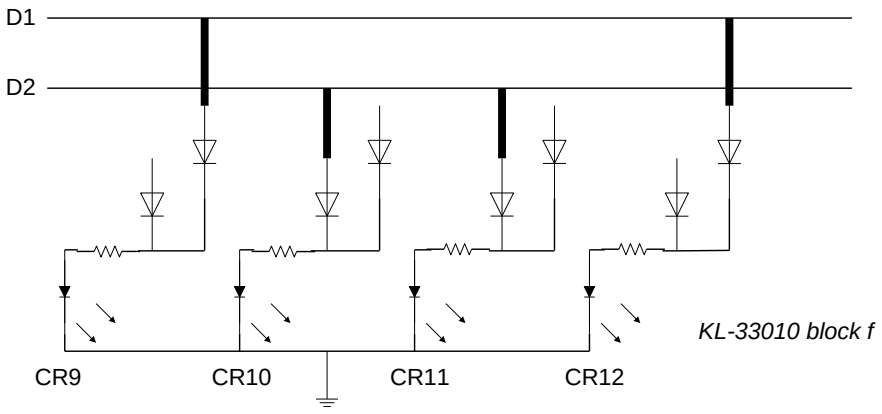
- Comprendre la structure d'une mémoire de type ROM et EPROM
- Comprendre la structure d'une mémoire de type RAM
- Applications des circuits mémoires

Matériel nécessaire : module alimentation KL-31001 et les modules KL-33010 et KL-33011

Manipulation

A- Etude des mémoires ROM et EPROM

1- Insérer les connections comme indiqué à la figure suivante.



2- Donner la sortie (CR9-CR10-CR11-CR12) correspondante aux entrées D2D1=01 et D2D1=10

3- Si l'alimentation est coupée ensuite rétablie est-ce que les sorties vont être les mêmes ? que peut-on dire de ce montage considéré comme l'équivalent à une mémoire ROM ?

4- Utiliser maintenant le bloc e du module KL-33010. La mémoire EPROM 27256 possède 15 lignes d'adresses et 8 lignes de données. Donner la capacité en Koctets de cette mémoire.

5- Connecter les entrées S1 à S3 du bloc e aux switchs TTL SW1 à SM3 et les sorties D0 à D7 aux Leds indicateurs L1 à L8.

6- Selon l'état des entrées S1S2, observer les sorties et commenter (S1S2 =00, 01,10 et 11)

7- Couper l'alimentation et la rétablir, est-ce que les sorties sont gardées ?

8- Quelle est la différence entre une ROM et une EPROM ?

B- Etude d'une mémoire RAM

1- Nous utilisons le bloc g du module KL-33010. Connecter E1, S1, D1, D2 aux switchs SW0 à SW3 (TTL).

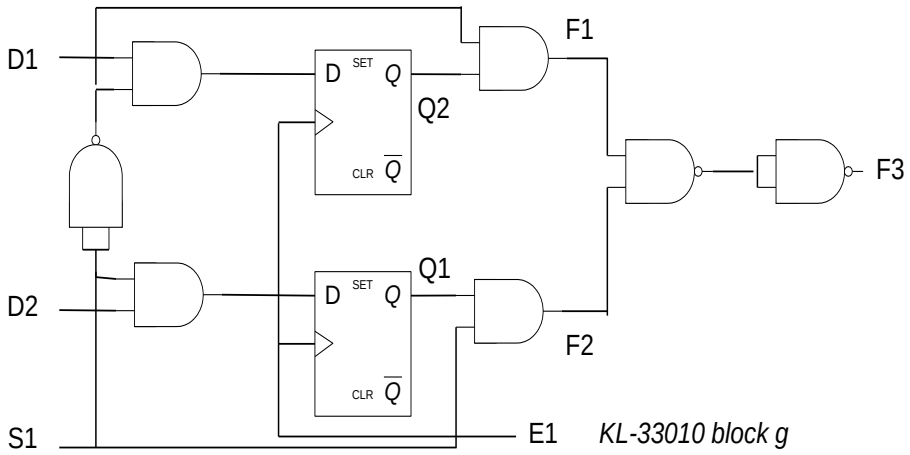


Figure 2 : KL33010 block g

Connecter les sorties F1, F2, F3 aux indicateurs L1, L2 et L3. Selon les séquences d'entrées du tableau 1, relever les sorties correspondantes. Que peut-on dire du fonctionnement de ce montage ? Ce montage est équivalent à quel type de mémoire ?

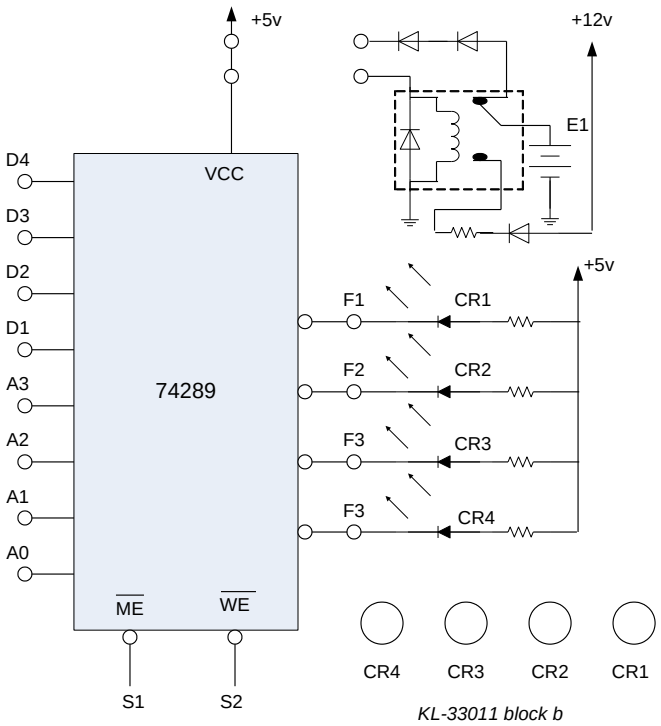
Entrée				Sortie		
E1	S1		D2	F1	F2	F3
D1						
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	0	0	0			
1	0	0	1			
1	0	1	0			
1	0	1	1			
1	1	0	0			
1	1	0	1			
1	1	1	0			
1	1	1	1			

Tableau 1

2- On utilise maintenant le module KL-33011. Le bloc b contient la mémoire RAM 64 bits (16x4) 74289. Le fonctionnement de la mémoire est réalisé selon le tableau suivant :

Entrées		Opération
ME/ (CS/)	WE/	
BAS	BAS	Ecriture
BAS	HAUT	Lecture
HAUT	X	Circuit non validée

Tableau 1



KL-33011 block b

Connecter les entrées D1 à D4 aux DIP Switch 1.0 à 1.3 et A0 à A3 aux DIP Switch 2.0 à 2.3. L'entrée S1 est connectée à l'entrée impulsion SWA et S2 à l'entrée Switch TTL SW0. Les sorties de la mémoire sont indiquées par CR1 à CR4. F1 à F4 sont initialement à 0. Le circuit

74289 est à sorties collecteurs ouverts. Connecter la batterie E1 (6v) à la tension 12v en alimentant le relais par 5v. Remplir le tableau 2 pour les séquences d'entrés 0000 à 1111. Les données écrites D1 à D4 varient aussi de 0000 à 1111.

	Ecriture			Lecture		
	ME/	WE/	D4 D3 D2 D1	ME/	WE/	F4 F3 F2 F1
A3 A2 A1 A0						
0 0 0 0	↓	0		↓	1	
0 0 0 1	↓	0		↓	1	
0 0 1 0	↓	0		↓	1	
0 0 1 1	↓	0		↓	1	
0 1 0 0	↓	0		↓	1	
0 1 0 1	↓	0		↓	1	
0 1 1 0	↓	0		↓	1	
0 1 1 1	↓	0		↓	1	
1 0 0 0	↓	0		↓	1	
1 0 0 1	↓	0		↓	1	
1 0 1 0	↓	0		↓	1	
1 0 1 1	↓	0		↓	1	
1 1 0 0	↓	0		↓	1	
1 1 0 1	↓	0		↓	1	
1 1 1 0	↓	0		↓	1	
1 1 1 1	↓	0		↓	1	
	↓			↓		

Tableau 3.

Après avoir remplie la RAM, reprendre les adresses depuis 0000 et remplir la deuxième partie du tableau. Commenter.

3- Connecter VCC à la sortie de la batterie en désexcitant le relais. Couper l'alimentation environ 20s et rétablir la tension est-ce que les données sont sauvegardées dans la RAM ?
4- Déconnecter la batterie de VCC et couper l'alimentation de nouveau. Est-ce que le contenu est toujours sauvegardé ? Quelles sont les différences majeures entre une RAM ?

TP4 :
Initiation au microprocesseur 80x86, utilisation du DEBUGGER

Objectifs :

- Connaissance des registres internes du Microprocesseur 80x86
- Se familiariser à l'utilisation du Debugger sous DOS
- Initiation à la programmation assembleur

A/ Initiation aux commandes sous DEBUG

Debug est un logiciel d'aide à la mise au point des programmes. Le terme "bug" est resté pour désigner une erreur inexpliquée dans un programme, le debuggage désignant la recherche des erreurs. Pour faire le débuggage, il existe de nombreux logiciels, les environnements intégrés de programmation incluant leurs propres débuggeurs. Les débuggers proposent trois fonctions principales :

- l'affichage des informations (variables)
- la simulation de l'exécution d'une partie d'un programme. Pour cette fonctionnalité, on peut procéder à une exécution pas à pas (exécution d'une instruction du programme et retour au débuggeur) ou partielle jusqu'à un point d'arrêt (breakpoints).
- l'exécution "instruction par instruction, le chargement, la modification et l'assemblage de petits programmes permettant ainsi de les tester sans avoir recours au compilateur.

Souvent les processeurs disposent aussi d'un mode d'exécution pas à pas, utilisé pour vérifier leur fonctionnement ou les programmes de très pas niveau (assembleur). Le programme DEBUG, disponible avec le système DOS (ou système similaire), est un logiciel qui met le

processeur en mode d'exécution pas à pas, ce qui permet de tester un programme machine, ou de lire les registres ou les données en mémoire.

Les commandes principales de DEBUG sont :

La commande R (pour Registres) affiche et permet de modifier le contenu d'un registre. Si aucun nom n'est spécifié, elle affiche tous les registres, ainsi que l'adresse de l'instruction suivante à exécuter, avec sa syntaxe hexadécimale et assembleur. Le registre des flags s'appelle F. si on donne RF, l'état des flags s'affiche et on peut changer les flags individuellement. OV, DN, EI, NG, ZR, AC, PE, et CY permettent de mettre à un les flags OVerflow, Direction, Interruption, Signe, Zéro, Retenue auxiliaire, Parité et Retenue. NV, UP, DI, PL, NZ, NA, PO et NC permettent des les remettre à zéro.

1 - Afficher le contenu de tous les registres, ……

2 - Ecrire dans le registre BX la valeur 1234H ……

3 - Afficher à l'écran l'état des indicateurs ……

4 - Essayer de changer la retenu intermédiaire ……

La commande D (pour Dump) affiche sur l'écran le contenu d'une zone mémoire (instructions, données, pile, ROM,...) dont les adresses de début et de fin sont spécifiées (ou bien le début et la longueur). L'affichage se présente en deux parties : une partie à 16 octets en hexadécimal et une partie pour la correspondance ASCII (c'est possible pour les données mais c'est sans signification pour les instructions).

Tapez les commandes suivantes :

5 - D ↵

……

……

6 - D 100 ↵

……

……

7 - D 100 120 ↵

……

……

8 - D 123 L10 ↵

……

……

La commande F (pour Fill) Remplit un bloc avec une suite d'octets. La suite est répétée pour tout remplir.

Syntaxe : F <bloc> <liste d'octets>

9 - Taper la commande suivante : F 200 300 20, 11, 33 ↵

……

……

10 - Visualiser le contenu du bloc à l'aide de la commande Dump

……

……

11 - Mettre des 00 dans une zone mémoire (100 cases) à partir de DS:200.

……

……

La commande A (pour Assemble) permet d'écrire, en mémoire, un programme en langage assembleur; qui pourra être désassemblé par la commande U.

Syntaxe : A <adresse>

Syntaxe : U ou U <adresse> ou U <bloc>

La commande T (pour Trace) exécute le nombre des instructions précisés à partir de l'adresse CS:IP. Après chaque instruction elle affiche les mêmes informations que la commande R.

Syntaxe : T [=<adresse>] [<Nb. Instructions>]

B/ Programmation sous Debug

Exercice 1:

1°) Lancez la commande A 100 et écrivez le programme suivant:

```
MOV WORD PTR [200] , 56E3
MOV AX , 0BF7
MOV BX ,C01E
XCHG AL , BL
XCHG AH , AL
XCHG [200] , AX
```

Afin de valider l'écriture du programme, tapez sur la touche [entrer] deux fois. Lancez la commande R.

Exécutez le programme pas à pas en lançant la commande T et vérifier le contenu des registres internes du microprocesseur à chaque exécution puis interprétez les différentes lignes du programme.

2°) En exécutant les instructions suivantes, indiquer la valeur de SS :SP ainsi que le contenu des registres AX, BX CX DX

```
MOV AX, 0BF7
MOV BX, C01E
MOV CX, C01E
```

```
MOV DX, C01E

PUSH AX

PUSH BX
POP AX
POP DX
```

Exercice 2:

1°) Exécuter les instructions suivantes et donner le contenu des registres associés puis vérifier le résultat manuellement:

a)
```
MOV AX, 0AFB

MOV BX, C01E

OR  AX, BX

OR  AX, FFFF

AND AX , 9999

XOR BX , BX

NOT AX
```

b)
```
MOV CX, C01E

MOV DX, C01E

NEG   CX
AND DX , CX
```

TP5 :

Programmation du kit MTS-86C

Principaux composants

Le kit MTS-86C est composé d´un microprocesseur 16 bits (8086 d'Intel) auquel sont connectés tous les constituants principaux de l'informatique industrielle :

- RAM (64Ko) 2 x 62256
- EPROM Moniteur (64Ko) 2 x 27256
- EPROM Utilisateur (64Ko) 2 x 27256
- Interface parallèle 3 x 8255
- Interface série 2 x 8251
- Contrôleur de clavier 8279
- Contrôleur d'interruption 8259
- Timer programmable 8253
- ADC (8 bits – 8 entrées) ADC809
- DAC (8 bits) DAC0808

Cartographie de la mémoire du kit

L'espace mémoire utilisé est divisé en trois parties (Figure **0-1**). Une zone pour le programme moniteur et exemples d'application, la seconde pour le programme utilisateur et les vecteurs d'interruptions et la troisième, une zone extensible, elle peut être du type ROM ou RAM.

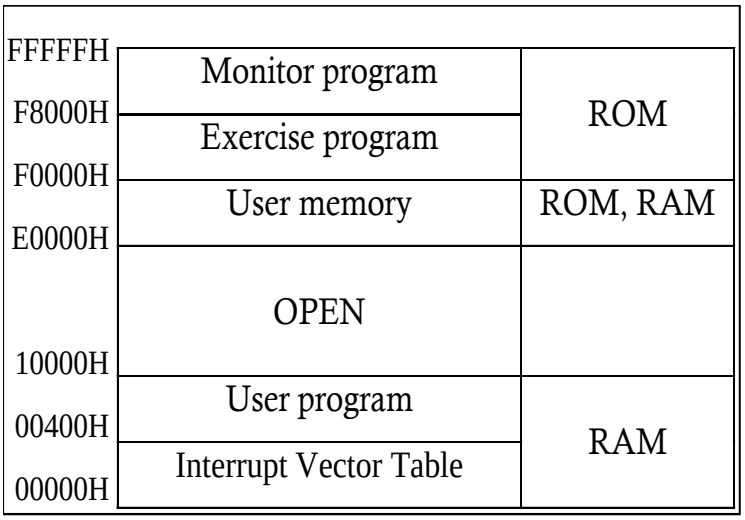


Figure 0-1 : Cartographie de la mémoire du kit MTS-86C

Adressage des entrées-sorties

Le schéma de la Figure **0-2** présente le décodage d'adresses des principaux périphériques à étudier. En se référant au schéma du décodage, déterminer l'adresse de base de chaque composant (remplir le Tableau **0-1**).

Tableau 0-1 : Adressage des Entrées-sorties

composants	Adresse de base en binaire				@ de base en Hexa.
	A_{15} A_{14} A_{13} A_{12}	A_{11} A_{10} A_9 A_8	A_7 A_6 A_5 A_4	A_3 A_2 A_1 A_0	
ADC809					
8255-3					
DAC0808					
Aff-7Seg					
8259					
8251-2					
8253					

8279					
8251-1					
8255-2					

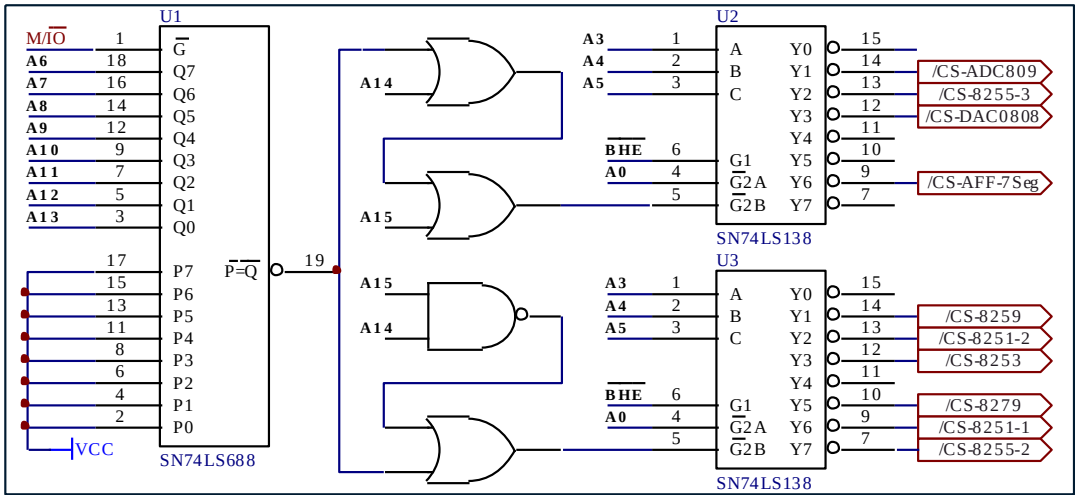


Figure 0-2 : Décodage d'adresses

Ecriture et exécution d'un programme sur le KIT MTS-86C

Ecriture du programme

Pour écrire un programme, il faut se servir d'un éditeur ASCII, tel que le programme EDIT du dos, puis l'enregistrer avec l'extension « .asm ». (exemple MonPrg.asm).

Taper le programme suivant et l'enregistrer sous le nom Tp0.asm

```
;============== Exemple d'un programme Assembleur ===============

CNT3        EQU         3FD6H

PB3         EQU         3FD2H


CODE        SEGMENT

        ASSUME  CS : CODE, DS : CODE
```

```
        ORG 0
        MOV SP, 4000H
        MOV AL, 90H
        MOV DX, CNT3
        OUT DX, AL

        MOV AL, 01H
        MOV DX, PB3
@ :     OUT DX, AL
        MOV CX, 0A000H
        LOOP $
        ROL AL, 1
        JMP @
CODE ENDS
        END
```

Génération du fichier HEX

Le kit MTS-86C ne supporte que les fichiers de type .hex d'Intel. Pour créer ce fichier à partir du ficher source, il faut faire appel à des programmes de compilation, d'édition de lien et de conversion.

a) Le programme «MASM» de Microsoft ou « TASM » de Borland, permettent de créer un fichier intermédiaire appelé « fichier Objet ». la commande suivante génère la ficher .obj.
b) Tapez la commande suivante:

C:\ASM86> masm Tp0 ;

c) Le programme «LINK» de Microsoft ou « TLINK » de Borland créent à partir du fichier .obj. un fichier .exe.

d) Tapez la commande suivante:

C:\ASM86> link Tp0 ;

e) Le programme «EXE2BIN» convertit le fichier .exe en un fichier .bin. Tapez la commande suivante:

C:\ASM86> exe2bin Tp0

f) Enfin le programme «BIN2HEX» convertit Le fichier .bin en un ficher .hex. Tapez la commande suivante:

C:\ASM86> bin2exe Tp0.bin

Chargement et exécution du programme

a) Connectez le Kit au port série du PC. Pour charger le programme dans le kit ouvrir l'hyper terminal de Windows, avec la configuration suivante : 19200 bps, 8bits de données, pas de parité, sans contrôle de flux.

b) Appuyez sur les boutons RESET puis F du kit, pour initialiser la communication. Le moniteur répond avec un message d'invite et vous donne la main. Tapez H si vous voulez afficher toutes les commandes.

c) Tapez L (pour Load) puis ↵ , le moniteur répond avec le message « Sending your file in INTEL HEX format. »

d) Dans le menu Transfert, cliquez sur Envoyer un fichier texte…, puis sélectionnez le fichier .HEX à charger (Tp0.hex par exemple). Si tous va bien le moniteur vous répond « Thank you for your cooperation »

e) Pour exécuter votre programme. Tapez G, puis Y pour confirmer l'adresse d'exécution proposée.