Proposition de correction
⋆ ⋆ ⋆
Agrégation interne 2020 – Partie 3

3. Le CAN (Convertisseur Analogique-Numérique)

Q.62. Chaque montage i est un comparateur.

Q.63. La tension $E_{\text{réf.}}$ valant 8 Volts, le montage proposé permet de soumettre chaque entrée V_i^- à :

Entrée inverseuse	V_1^-	V_2^-	V_3^-	V_4^-	V_5^-	V_6^-	V_7^-
tension correspondante	1 V.	2 V.	3 V.	4 V.	5 V.	6 V.	7 V.

et chaque tension V_i^+ à :

Entrée inverseuse	V_1^+	V_2^+	V_3^+	V_4^+	V_5^+	V_6^+	V_7^+
tension correspondante	e	e	e	e	e	e	e

On rappelle que quel que soit l'ALI i, $\begin{cases} \text{si } V_i^+ > V_i^- \Longrightarrow s_i = +V_{\text{sat}} \Longrightarrow \texttt{sortie}_i = 1 \\ \text{si } V_i^+ < V_i^- \Longrightarrow s_i = -V_{\text{sat}} \Longrightarrow \texttt{sortie}_i = 0 \end{cases}$

On en déduit donc le tableau suivant selon les valeurs de e :

Valeur de e en V. / État $\texttt{sortie}_i$ de l'ALI i	< 1	$[1;2[$	$[2;3[$	$[3;4[$	$[4;5[$	$[5;6[$	$[6;7[$	> 7
$\texttt{sortie}_7$	0	0	0	0	0	0	0	1
$\texttt{sortie}_6$	0	0	0	0	0	0	1	1
$\texttt{sortie}_5$	0	0	0	0	0	1	1	1
$\texttt{sortie}_4$	0	0	0	0	1	1	1	1
$\texttt{sortie}_3$	0	0	0	1	1	1	1	1
$\texttt{sortie}_2$	0	0	1	1	1	1	1	1
$\texttt{sortie}_1$	0	1	1	1	1	1	1	1
Sortie décodeur $a_2a_1a_0$	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1

Q.64. Pour un codage sur 3 bits $a_2a_1a_0$, $2^3 - 1$ (7) ALI sont nécessaires.

$\Longrightarrow$ Pour un coadage sur 16 bits $a_{15}a_{14}\ldots a_1a_0$: $\underline{2^{16} - 1}$ ALI sont nécessaires (65 535 ALI).

Q. 65.

Notre programme affichera deux choses (l'énoncé ne demande que les sorties du décodeur logique) :

— les valeurs des différentes sorties des ALI (en binaire)

— les sorties du décodeur logique

On propose le programme suivant que l'on tape directement en console pour pouvoir effectuer des tests à la suite (les tests ne sont pas demandés par l'énoncé) :

```
>>> def conversion(e) :
...     # on ajoute un nouveau tableau a2a1a0
...     # ce tableau contient la sortie du décodeur
...     if e <1 :
...         sorties=[0,0,0,0,0,0,0]
...         a2a1a0=[0,0,0] # rajouté
...     elif e>=1 and e<2 :
...         sorties=[0,0,0,0,0,0,1]
...         a2a1a0=[0,0,1] # rajouté
...     elif e>=2 and e<3 :
...         sorties=[0,0,0,0,0,1,1]
...         a2a1a0=[0,1,0] # rajouté
...     elif e>=3 and e<4 :
...         sorties=[0,0,0,0,1,1,1]
...         a2a1a0=[0,1,1] # rajouté
...     elif e>=4 and e<5 :
...         sorties=[0,0,0,1,1,1,1]
...         a2a1a0=[1,0,0] # rajouté
...     elif e>=5 and e<6 :
...         sorties=[0,0,1,1,1,1,1]
...         a2a1a0=[1,0,1] # rajouté
...     elif e>=6 and e<7 :
...         sorties=[0,1,1,1,1,1,1]
...         a2a1a0=[1,1,0] # rajouté
...     else :
...         sorties=[1,1,1,1,1,1,1]
...         a2a1a0=[1,1,1] # rajouté
...     # la fonction conversion retourne l'état des sorties
...     # et la sortie du décodeur
...     return sorties, a2a1a0 # modifié
...
>>> # fin de la définition de la fonction "conversion"
>>> #
>>> ############################################################
>>> # Effectuons quelques tests pour voir si ça fonctionne :
>>> conversion(0.32)
([0, 0, 0, 0, 0, 0, 0], [0, 0, 0])
>>> conversion(1.6)
([0, 0, 0, 0, 0, 0, 1], [0, 0, 1])
>>> conversion(2.1)
([0, 0, 0, 0, 0, 1, 1], [0, 1, 0])
>>> conversion(3)
([0, 0, 0, 0, 1, 1, 1], [0, 1, 1])
>>> conversion(4.7)
([0, 0, 0, 1, 1, 1, 1], [1, 0, 0])
>>> conversion(5.8)
([0, 0, 1, 1, 1, 1, 1], [1, 0, 1])
>>> conversion(6.9)
([0, 1, 1, 1, 1, 1, 1], [1, 1, 0])
>>> conversion(7.24)
([1, 1, 1, 1, 1, 1, 1], [1, 1, 1])
```