Conception VLSI d'un accordeur pour instruments de musique

Frédéric Pétrot*
Architecture des Systèmes Intégrés et Micro-électronique
Université Pierre et Marie Curie
France

27 juillet 2004

1 Introduction

Ce texte est une introduction à la conception de circuits intégrés à l'aide d'outils de CAO. Je fais l'hypothèse que les outils utilisés sont ceux de la chaîne de CAO **Alliance**, et qu'ils sont installés correctement, ainsi que les pages de manuel, sur votre système.

L'implantation est totalement automatisé, et les seules interventions du concepteur consistent à écrire la spécification dans un langage exécutable, puis à lancer les différents outils en comprenant ce qu'il fait. C'est donc une vision idylique de la conception que je vous propose.

La première partie du document décrit la théorie du tempérament égal, et définit les spécification du circuit que je vous propose de réaliser. La seconde partie décrit l'architecture retenue pour faire les différentes fonctionnalités requises. La troisième partie décrit l'utilisation des outils pour implanter cette architecture sur le silicium.

Mes excuses aux non-francophones, mais je n'ai vraiment pas le temps de traduire cet exercice. Aussi si une bonne âme le trouve suffisament intéressant pour le traduire, welcome !

2 Spécifications

2.1 Généralités sur la fréquence des notes

Nous cherchons à construire un circuit électronique permettant d'accorder un piano possèdant 8 octaves. Conventionnellement, un tel instrument commence par un *la* grave et s'achevant par un *do* aigü. La fréquence du *la* le plus grave est de 27.5 Hz. La fréquence du *do* le plus aigü est de 4186 Hz. Passer à l'octave supérieur signifie multiplier par 2 la fréquence. Ainsi, le *la* d'octave supérieure[1] est de fréquence $27.5 \times 2 = 55$ Hz. La musique occidentale est basée sur 7 notes s'étalant 12 demi-tons. La répartition est :

$$\underset{\leftarrow}{do} \underset{\rightarrow\leftarrow}{1 \text{ ton}} \underset{\rightarrow}{ré} \underset{\leftarrow}{1 \text{ ton}} \underset{\leftarrow}{mi} \underset{\rightarrow\leftarrow}{\frac{1}{2} \text{ ton}} \underset{\rightarrow}{fa} \underset{\leftarrow}{1 \text{ ton}} \underset{\rightarrow}{sol} \underset{\leftarrow}{1 \text{ ton}} \underset{\rightarrow\leftarrow}{la} \underset{\rightarrow\times\leftarrow}{1 \text{ ton}} \underset{\rightarrow}{si} \underset{\leftarrow}{\frac{1}{2} \text{ ton}} \underset{\rightarrow}{do}$$

*Préparé, dans le cadre des TREX d'électronique de l'École Polytechnique, en vue d'une implantation VLSI.
[1]Octave : nom féminin.

Dans le « tempérament » égal, chaque demi-ton est distant (en fréqence) de son voisin d'un facteur multiplicatif équitablement réparti. Comme il y a 12 demi-tons entre deux notes identiques séparées d'une octave, ce facteur est donc $\sqrt[12]{2} \approx 1.0594630944$. Lorsqu'une note x est inférieure à sa voisine, y, d'un ton, on appelle la note supérieure à x d'un demi-ton $x^{\#}$, et celle inférieure à y d'un demi-ton $y^{\flat}$. Dans le tempérament égal, la fréquence de $x^{\#}$ est la même que celle de $y^{\flat}$. Ainsi la fréquence du *la*$^{\#}$ de la première octave est $27.5 \times \sqrt[12]{2} \approx 29.1$, et c'est aussi celle du *si*$^{\flat}$.

Les fréquences des notes de l'octave de référence sont données ici :

Note	Fréquence	Note	Fréquence	Note	Fréquence	Note	Fréquence
do	261.6 Hz	*ré*$^{\#}$	311.1 Hz	*fa*$^{\#}$	370.0 Hz	*la*	440.0 Hz
do$^{\#}$	277.2 Hz	*mi*	329.7 Hz	*sol*	392.0 Hz	*la*$^{\#}$	466.2 Hz
ré	293.7 Hz	*fa*	349.2 Hz	*sol*$^{\#}$	415.3 Hz	*si*	493.9 Hz

2.2 Problème de l'accordage

Le problème que nous visons n'est pas de mesurer la fréquence d'une note, mais d'aider un musicien à accorder son instrument. Il faut donc être capable de discriminer deux fréquences proches pour indiquer sur un périphérique de visualisation de quelle note la note jouée est la plus « proche » et à quelle distance elle se trouve approximativement.

On se propose d'utiliser la méthode « à l'ancienne » et très peu chère à base de diodes et de diodes 7 segments.

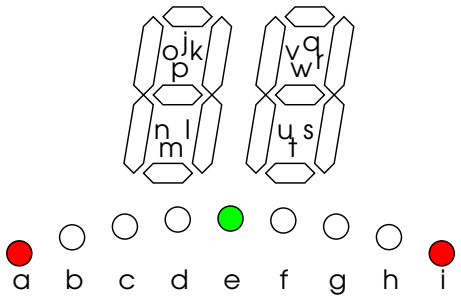

FIG. 1 – Dispositif de visualisation

Ce dispositif possède d diodes ($d = 9$ pour le dispositif de la figure 1). La diode centrale est allumée lorsque la note est exacte, et son nom est affiché, avec potentiellement un `b` derrière pour indiquer une note altérée d'un bémol. Si la fréquence est plus basse, alors la note reste affichée, mais la diode de gauche représentant la « distance » à la note est allumée. Si la fréquence continue de diminuer, et que l'on s'approche plus d'une autre note, c'est alors la diode à l'extrême droite qui s'allume et le nom de la note change. Le fonctionnement est inverse si la fréquence est plus élévée.

Si l'on a d diodes, alors un demi-ton est découpé en d parties égales (car la diode la plus à gauche est voisine de celle la plus à droite). Une octave est alors découpée en $d \times 12$ parties égales, et passer d'une partie à la suivante, c'est multiplier la fréquence par $\sqrt[12\times d]{2}$.

En partant sur $d = 9$, notre facteur vaut $\sqrt[111]{2} \approx 1.0064386691$. La note la plus grave sera un *la* fortement bémolisé à 26.8 Hz, et la note la plus aiguë un *do* fortement diésé à 4294.86 Hz

2.3 Analyse du problème

On fait l'hypothèse que le signal analogique sortant de l'instrument est transformé par un dispositif en signal carré numérique de la fréquence du signal initial. Ceci doit être fait de manière

subtile, car les signaux sortant des instruments sont assez torturés, mais sort du cadre de ce TREX.

L'idée est de mesurer le nombre de cycles d'un signal de période connue, période obtenue grâce à un quartz, qui s'écoulent entre deux front montant succéssif du signal d'entrée. Cette mesure est faite en temps réel, et suit donc l'évolution du signal d'entrée au cours du temps. On appelle le signal de période connue ck et le signal d'entrée in.

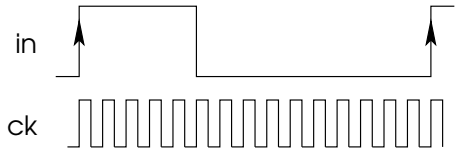

FIG. 2 – Décompte du nombre de périodes de ck dans une période de in.

Le problème est d'être capable de discriminer 2 périodes représentatives de 2 intervales successifs parmis les d formant un demi-ton. Nous avons $88 \times d$, soit 792 pour $d = 9$, tels intervales. Bien évidemment, les périodes les plus petites sont les plus difficiles à distinguer. La fréquence la plus élévée est 4294.86 Hz (période $.2328$ ms), et celle immédiatement inférieur, qu'il faut être capable de discriminer, est 4267.38 Hz (période $.2343$ ms). La différence de durée de ces périodes est donc de 1.5 μs. Par ailleurs, la période de la note la plus grave est de $\frac{1}{26.80} = 37.31$ ms.

2.4 Solution envisagée

Pour une implantation VLSI, nous allons rechercher une solution la plus simple possible dans laquelle les seuls calculs doivent être des additions, sur aussi peu de bits que possible, des comparaisons, et des décalages. Les différents traitements doivent pouvoir s'exécuter en parallèle, car le décompte du nombre de périodes d'horloge à l'intérieur d'une période du signal d'entrée, que nous notons n à présent, doit se faire en continu.

Le principe retenu est donc :

1. puisque des fréquences – ou des périodes – doubles – ou moities – ont le même nom, nous allons « normaliser » n en le divisant par 2 pour qu'il soit représenté par un nombre p le plus petit possible, c.-à-d. celui correspondant aux fréquences les plus élévées. Ce choix est dicté par le nombre de bits nécessaire à la représentation de p, qui doit être le plus petit possible tout en permettant de discriminer les 2 périodes les plus courtes. Ce nombre p sera compris entre le nombre a de cycles pour le *ré* le plus bémol de l'octave supérieure (période $.4627$ ms) et le nombre b de cycle pour le *do* le plus diésé de cette même octave (période $.2328$ ms), qui est la dernière note qui nous intéresse ;
2. puisque $b \leq p \leq a$, nous pouvons faire une table dont l'indice sera $p - b$ et le contenu sera la commande de côntrole des diodes. La taille de cette table doit être minimisée. Pour cela, il faudra que la différence entre les nombres de cycles par transistion du signal d'entree soit la plus petite possible, soit 1 dans le meilleur des cas. Ce nombre dépend de la fréquence de ck, que nous ne pouvons choisir arbitrairement, car elle dépend des quartz disponibles sur le marché.

   Comme la différence entre les deux périodes les plus courtes est de 1.5 μs, la valeur idéale de la fréquence de ck est de 667 KHz. Le quartz horloger de base est à $32,768$ KHz, donc bien trop basse fréquence, et le premier autre quartz à vil prix est à 3.2768 MHz. En divisant cette fréquence par quatre, on obtient un circuit fonctionnant à 819.2 KHz, ce qui est raisonnablement proche de notre fréquence idéale ;

3. en partant sur cette hypothèse, ck a une période de 1.22 μs, donc $a = 381$ et $b = 191$. Le nombre d'entrées de la table est $a - b = 190$. Le nombre d'entrées correspondant exactement à une période d'une diode est de $12 \times d$ soit 108 dans notre cas. Nous devrons donc approximer 82 entrées par leur valeur la plus proche. La figure 3 représente la valeur approximée

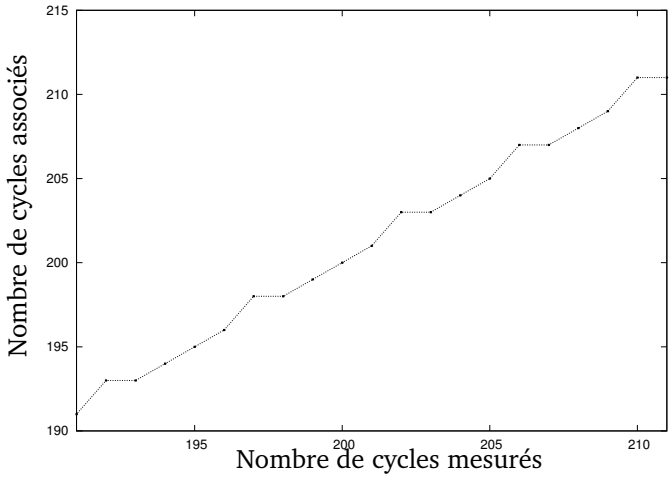

FIG. 3 – Exemple de répartition des entrées intermédiaires

lorsqu'une valeur mesurée ne correspond pas à une valeur précalculée.

De plus, p est lui même une approximation entière, donc nous aurons une petite erreur. Plus haute est la fréquence de ck, plus faible est l'erreur, mais plus élévé est le nombre de bits nécessaire, donc la surface, et la puissance consommée, ce qui est important si l'accordeur est alimenté par piles. La figure 4 présente l'erreur d'arrondi due au passage en entier pour notre plage de référence.

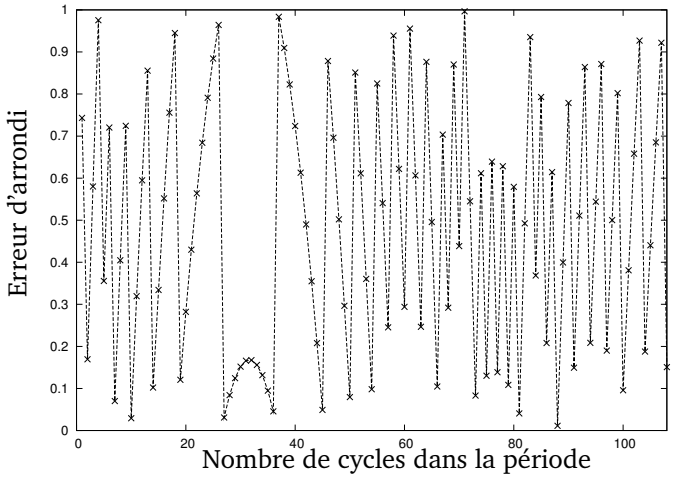

FIG. 4 – Erreurs sur les valeurs de p.

4. les indices d'accès à la table sont compris entre 0 et 188. Il faut donc $\lceil \log_2 188 \rceil = 8$ bits pour l'adresser.

3 Architecture

3.1 Interface logique

L'interface logique du dispositif que nous désirons concevoir est présentée sur la figure 5. On

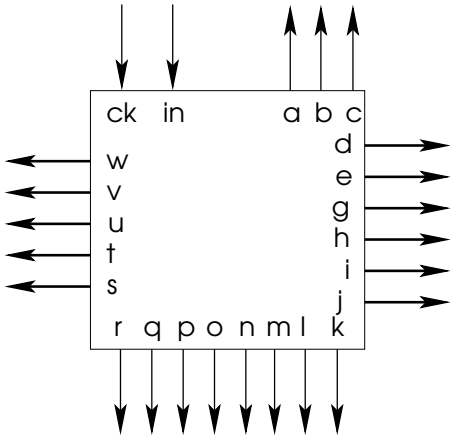

FIG. 5 – Interface logique du dispositif.

entend par interface logique le fait que cette interface ne contient pas les broches d'alimentation et qu'elle peut être amenée à être remodelée à cause des contraintes sur le nombre de broches disponibles dans les boitiers, ou de la taille du circuit. On peut par exemple être amené à multiplexer les sorties.

Le signal resetn fait une remise à zéro des registres. Le signal numérique issu du traitement analogique sera appliqué sur l'entrée in. Le signal de période connue sera appliqué sur l'entrée ck. Les autres signaux servent à commander les diodes (ou tout autre dispositif) pour l'affichage.

3.2 Matériel nécessaire

Les fonctionnalités nécessaires à l'implantation sont :

- un detecteur de front montant du signal in ;
- un incrémenteur comptant sur chaque front montant du signal ck. L'incrémenteur peut être amené à analyser la fréquence la plus grave que nous ayons, soit 26.8 Hz. Le nombre de cycles de ck est alors 30582, ce qui impose une taille de $\lceil \log_2 30582 \rceil = 15$ bits ;
- une unité de normalisation qui contient un diviseur par 2 itératif et un comparateur pour commander l'arrêt de la division. Le résultat $191 \leq p < 382$ tient sur 9 bits ;
- un soustracteur permettant de recadrer l'indice d'accès au tableau en calculant $p - 191$. Le resultat possède donc 8 bits significatifs ;
- une table de constantes contenant les valeurs à appliquer sur les sorties a à w et son décodeur. Cette table possède 190 entrées de 23 bits chacune.

3.2.1 Détecteur de front montant

La détection d'un front montant se fait avec le dispositif suivant :

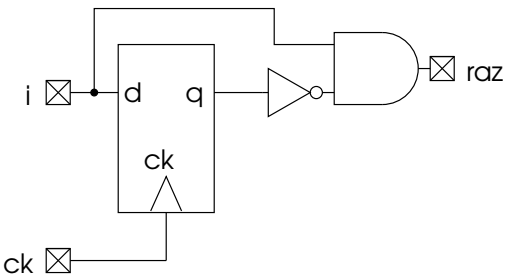

FIG. 6 – Détecteur de front montant.

3.2.2 Compteur

Nous avons besoin d'un incrémenteur, mais puisque nous aurons également l'usage d'un soustracteur, et que le temps de réalisation nous est compté, nous allons faire un simple additionneur à propagation de retenue.

Le compteur se contente d'incrémenter sa valeur de 1 à chaque cycle, sachant qu'il est systématiquement réinitialisé à 0 lors de la detection d'un front montant de l'entrée. La réinitialisation doit être synchrone avec l'horloge, afin que les registres qui échantillonnent la donnée puissent le faire avant cette remise à zéro.

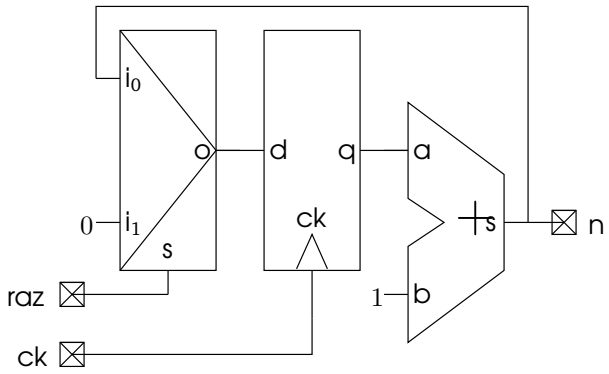

FIG. 7 – Compteur d'impulsions.

3.2.3 Unité de normalisation

L'unité de normalisation divise par deux le nombre n issu du compteur pour dans un premier temps le ramener à une valeur strictement inférieure à 382 et dans un second temps lui soustraire 191. Le nombre ainsi obtenu permettra d'indexer directement la table donnant les sorties en fonction de l'index.

3.2.4 Table de constantes

La table de constantes sera réalisée à l'aide d'un outil d'optimisation booléenne. En fait, nous allons exprimer chacune des sorties en fonction de toutes les entrées, par ex. la fonction f_a d'allumage de la diode a sera notée $f_a(\mathsf{idx}_0, \ldots, \mathsf{idx}_7)$. Comme il y a de nombreuses configurations des idx_i pour lesquel $f_a = 0$, on peut espérer qu'un outil d'optimisation booléenne va grandement minimiser la logique nécessaire. Cette table est purement combinatoire, cependant, afin d'avoir

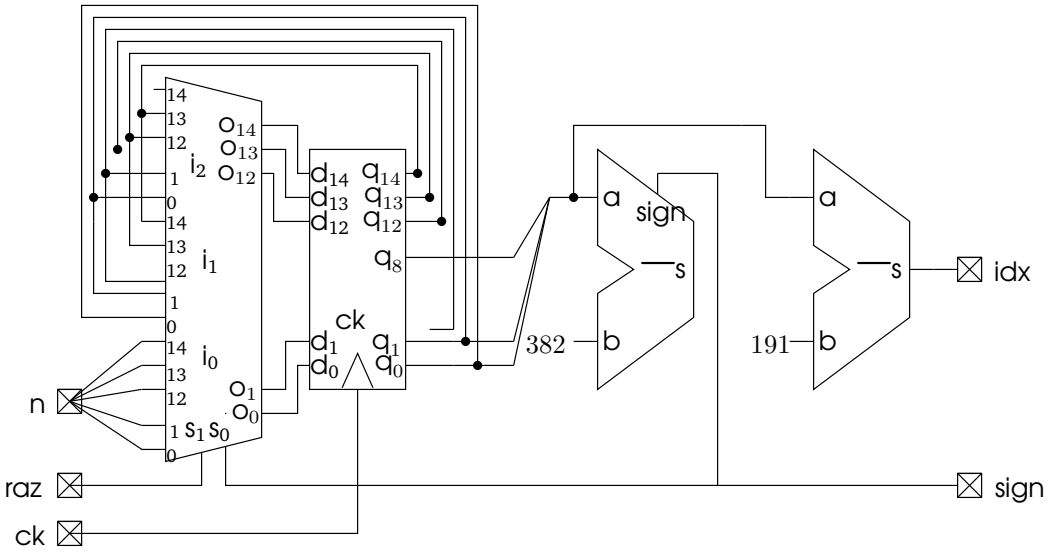

FIG. 8 – Normalisation du nombre de cycles comptés.

une entrée stable, nous plaçons devant-elle un registre qui échantillonne la valeur finale de la normalisation.

La difficulté est qu'un tel outil a besoin de cellules logiques classiques pour générer ces résultats, et nous devrons donc dessiner des et, ou, non et, non ou, non.

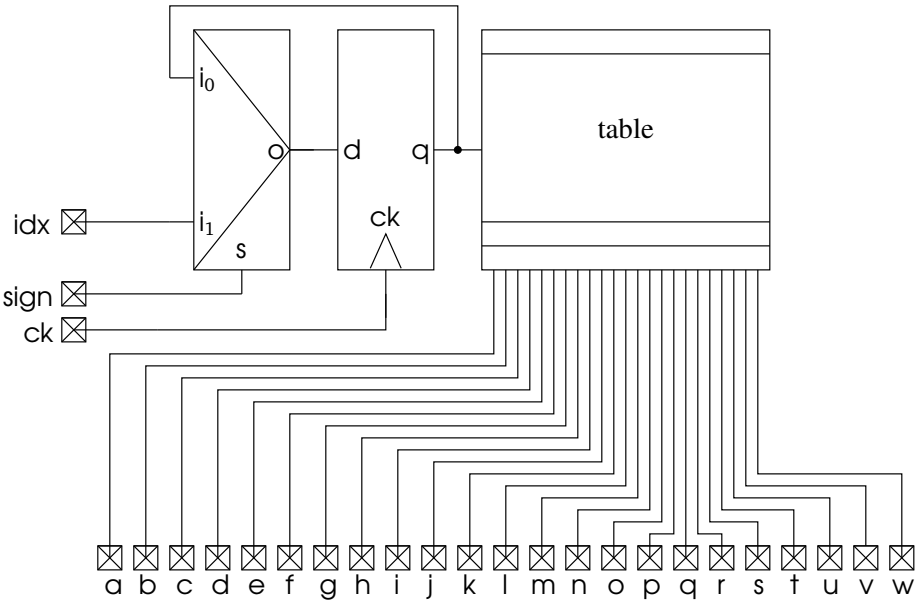

FIG. 9 – Décodeur du nombre d'implusions en note

4 Implantation VLSI

Pour cette partie, je vous propose de lire :

1. le fichier `tuner.vbe` qui décrit le comportement du circuit dans sa totalité. On regardera

avec intérêt les additionneurs (qui se font grâce à l'opérateur `'+'` dans `vasy(1)`. La table se trouvant à la fin a été générée grâce à un petit programme C, car il y a des limites au stackanovisme.

2. le fichier `build_tuner` qui appelle les différents outils avec les paramètres. Assurez vous que vous avez bien compris ce que sont les entrées et les sorties des outils. Il y a quelques sources C à écrire, d'abord pour les vecteurs de test (qui mériteraient d'être largement augmentés) puis pour la description des connexions entre le core et les plots. Il est nécessaire de savoir comment connecter ces plots, et le dessin de la figure 10 tente de vous y aider. Il faut aussi décrire les fichiers de placement des entrées/sorties pour `ocp` et `ring`. Je vous incite à consulter ces pages de manuels pour en comprendre la syntaxe et l'utilité.

Bon courage, et bons *designs* avec **Alliance**.

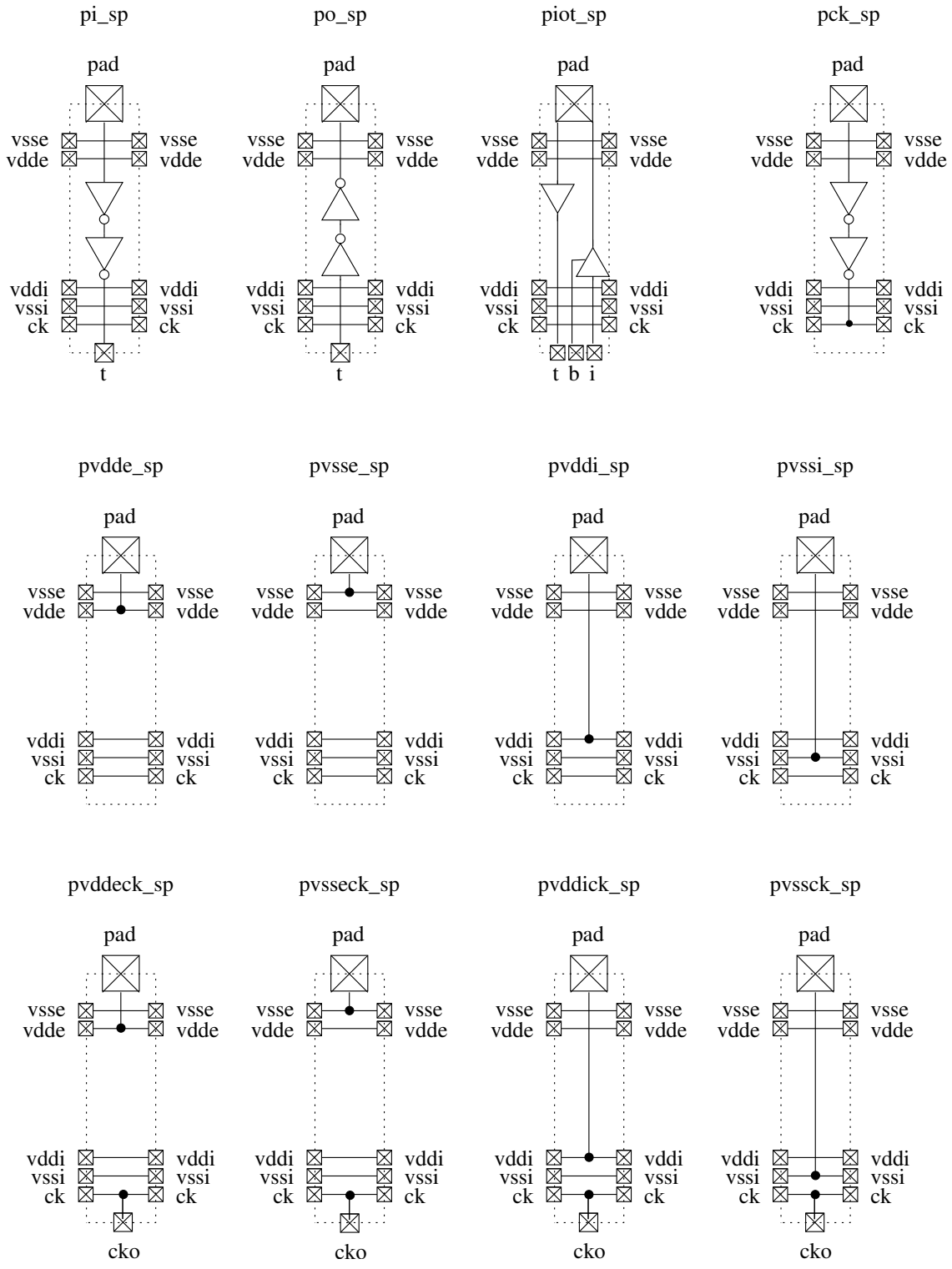

FIG. 10 – Schéma fonctionnel des plots d'entrée/sortie