

# Examen partiel

Département de génie électrique et de génie informatique  
Systèmes VLSI - GIF19264

le 20 octobre 2004

Vous avez droit à tous les documents. SVP, *pas* d'ordinateurs portables.  
Durée de l'examen: 2 heures (13h30-15h30).

---

1. *Types de PLDs* (10 points) Indiquez, pour chacune des expressions suivantes, quelle technologie lui correspond le mieux dans la liste suivante: FPGA, CPLD, PAL, PLA, EPLD ou FPGA-plateforme.
  - (a) Dispositifs logiques programmables ayant la plus haute densité FPGA
  - (b) Dispositifs logiques programmables complexes le plus souvent configurables via une mémoire vive statique FPGA
  - (c) Dispositifs logiques programmables complexes plutôt combinatoires que séquentiels CPLD
  - (d) Dispositifs logiques programmables simples avec plan-ET et plan-OU programmables PLA
  - (e) Dispositifs logiques programmables complexes formés de ressources logiques programmables en plus de noyau(x) dur(s) de microprocesseur et / ou d'autres ressources spécialisées FPGA-plateforme
  - (f) Dispositifs logiques programmables complexes dont les délais entrée-sortie sont prévisibles CPLD
2. *syntaxe, sémantique, types, attributs* (20 points) Pour chacune des bribes de code VHDL ci-dessous, indiquez s'il y a une erreur et, s'il y a lieu, indiquez concrètement la nature de l'erreur. Assumez par ailleurs que le préambule commun s'applique et que les bribes de code sont situées dans le corps d'une architecture.

```
type opcode is (nop, change, entrepose, additionne, soustrait, negation, halte);  
subtype op_arith is opcode range additionne to negation;  
type tablo1 is array (255 downto 0) of std_logic;  
type tablo2 is array (7 downto 0) of std_ulogic;  
type tablo3 is array (0 to 7) of std_logic;  
type tablo4 is array (0 to 7) of bit;
```

```

signal s1: tablo1;
signal s2: tablo2;
signal s3: tablo3;
signal s4: tablo4;
signal a, b, y: bit;

```

(a) **variable** A: op\_arith := op\_arith'succ( additionne );

OK

(b) **variable** B: opcode := op\_arith'succ( negation );

Erreur: expression de droite invalide car le successeur de negation ne fait pas partie du sous-type op\_arith.

(c) **variable** C: op\_arith := op\_arith'base'succ( negation );

Erreur: expression de droite valide, mais incompatible avec le type de gauche.

(d) **variable** v1: tablo2 := "01101001";

OK

(e) s3 := v1;

Erreur: mauvais opérateur d'assignation (s3 est un signal).

(f) s2 <= tablo2(s3);

OK

(g) s2 <= tablo2(s4);

Erreur: une conversion entre les types tablo2 et tablo4 nécessite une fonction de conversion.

(h) s4 <= ('0', '1', '1', '0', **others** => 'z');

Erreur: l'élément 'z' ne fait pas partie du type bit.

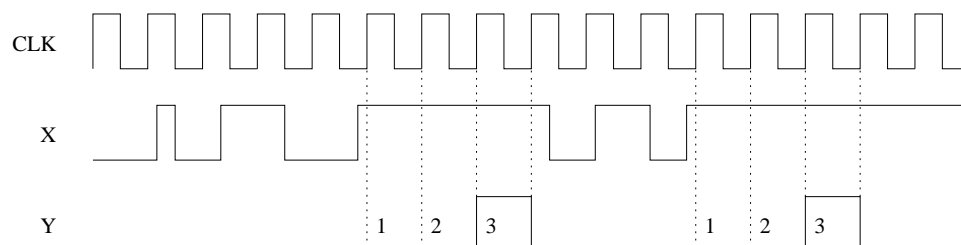
(i) v1 := tablo2( s1 );

Erreur: Plages incompatibles.

(j) y <= a **after** 5 ns;  
y <= a **xor** b **after** 15 ns;

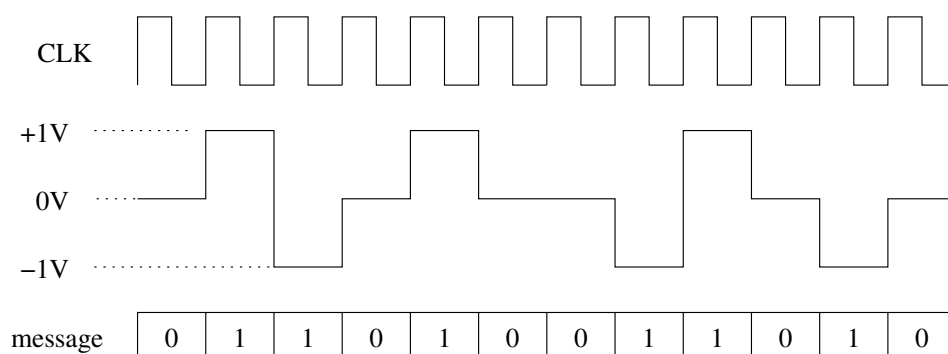
Erreur: Double assignation (conflit) sur signal y de type non-résolu.

3. *Conditionnement des entrées* (25 points) On désire réaliser un circuit d'amortissement synchrone qui, à partir d'un signal non-synchronisé x, produira en sortie y une et une seule impulsion d'une durée d'un coup d'horloge pour toute impulsion à l'entrée qui demeure stable à '1' pendant 3 coups d'horloge.

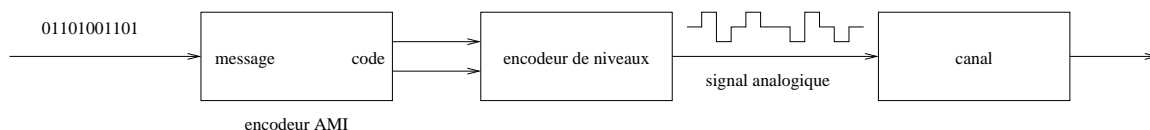


- (a) (10 points) Dessinez le graphe détaillé de la machine à états correspondante.
- (b) (15 points) Écrivez une description VHDL (entité et architecture) qui réalise le circuit d'amortissement.
4. *Design* (35 points) On utilise en communications numériques sur ligne câblée (e.g. paire torsadée, câble coaxial) divers “codes de ligne” pour communiquer en bande passante, c’est-à-dire sans moduler sur une porteuse. Ces codes constituent tout simplement une façon de représenter les bits à transmettre qui procure des caractéristiques spectrales intéressantes et / ou des propriétés faciliteront la synchronisation au récepteur.

Un tel code de ligne est le code “AMI” (Alternate Mark Inversion) qui utilise 3 niveaux de tension: -1, 0 et +1 volt. Pour encoder un zéro, on utilise la tension nulle (0 V). Pour encoder un '1', on utilise *en alternance* les niveaux +1 et -1. Ainsi, on a:



Une portion de la chaîne de communication correspondante est illustrée ici:



L’encodeur AMI convertit les bits à transmettre en code sur 2 bits. L’encodeur de niveaux produit à son tour un niveau analogique de -1, 0 ou +1 volt à partir de ce code sur 2 bits. La correspondance peut se faire selon le tableau suivant:

code	niveau (V)
00	-1
01	0
10	0
11	1

- (a) (20 points) Dessinez le graphe détaillé d’une machine à états réalisant l’encodeur AMI.
- (b) (15 points) Écrivez une description VHDL (entité et architecture) correspondante en séparant la partie synchrone et la partie combinatoire en des processus distincts.

*Bonne chance et bonne semaine de lecture!*