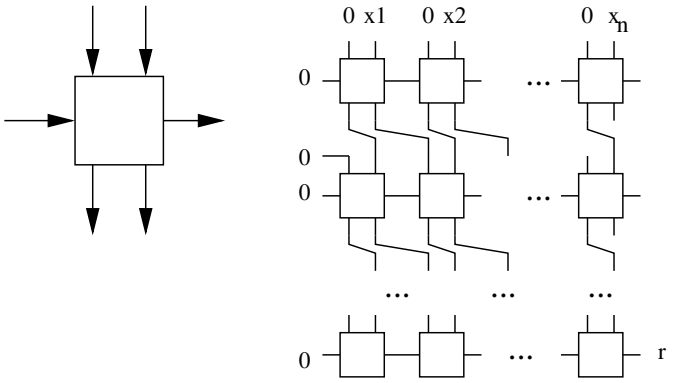
Partiel d'Architecture et Système

16 novembre 2014

Vous avez deux heures (de 13h45 à 15h45). Justifiez vos réponses.

1 Circuits

On considère des circuits formés par un rectangle de $k \times n$ *éléments*. Un élément est lui-même un circuit logique avec quelques entrées à gauche et en haut et quelques sorties à droite et en bas. Ci-dessous à gauche, vous voyez un tel élément avec trois entrées et trois sorties.



Un circuit peut contenir plusieurs types d'éléments. Ci-dessus à droite vous voyez un circuit qui contient des éléments comme à gauche sauf la derniére ligne, où les éléments possèdent trois entrées et une sortie.

1. Soit $m \leq n$ et $m+1$ le nombre de lignes dans le circuit ci-dessus. Dessinez les deux types d'éléments qu'il faut pour que le résultat r est 1 si et seulement si $\sum_{i=1}^{n} x_i \geq m$.

2. La profondeur du circuit donné est de $\mathcal{O}(n+m)$. Décrivez comment construire un circuit (de n'importe quelle forme) qui calcule la même fonction avec profondeur $\mathcal{O}(\log^2 n)$. (Vous pouvez utiliser n'importe quel circuit déjà discuté en cours.)

2 Opérations arithmétiques et logiques

Vous devez écrire quelques lignes de code en C. Rappel : les opérateurs "bit-wise" en C s'écrivent `&` (et), `|` (ou), `^` (ou exclusive), `~` (négation, unaire). L'opérateur de décalage à droite est `>>`. Dans le suivant on suppose que le type `int` consiste de 32 bits. Soit `x` un `int`.

L'opération de décalage à droite en C est dite *arithmétique*, c'est à dire qu'elle préserve le signe. Plus précisement, si `x` consiste des bits $x_{31} \cdots x_0$ où x_{31} contient le signe, le résultat de `y = x >> k` est $y_i = x_{i+k}$ pour $i < 31 - k$ et $y_i = x_{31}$ sinon.

1. En utilisant l'opérateur de décalage, écrire une expression qui donne `0` si `x` est zéro ou positif et `-1` sinon.

2. Voici une expression incomplète : `~(...)`. Complétez cette expression pour qu'elle donne `-x`.

3. On cherche une expression qui calcule la valeur absolue de `x` *sans branchement* (p.ex., sans `if` ni `?:`). Aide : En plus des idées développées en (a) et (b), l'opérateur `^` sera utile. Si vous n'avez pas réussi (a) ou (b), utilisez un symbole de fonction (`a,b`) dans sa place.

4. Pourquoi une expression sans branchement est-elle intéressant ?

3 Programme mystère

On considère le programme C suivant. Sur certains ordinateurs, ce programme affiche `x=0`. Pourquoi ?

```
void foo() {
   int a[1];
   a[2] += 8;
}

void main() {
   int x;
   x = 0;
   foo();
   x = 1;
   printf("x=%d",x);
}
```

4 Processus

Dessinez l'arbre généalogique des processus crées par le programme suivant :

```
int main() { fork() && (fork() || fork()); }
```