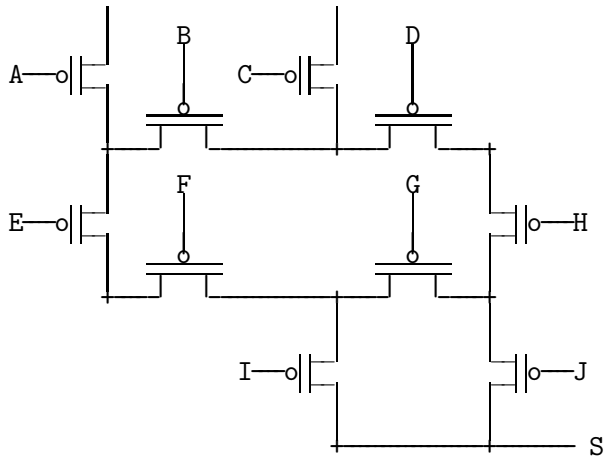
Université de Nanterre jeudi 22 juin 2017
architecture des ordinateurs contrôle

Que valent CF, OF, ZF et SF après les opérations 6+12, 1+9, 6-15, 11-8, 13+14, 5-4, 13-4, 4-5, 7-10, 10-7, 13-15, 5+5, 0+2, 9+9, sur des nombres codés sur 4 bits.

Complétez le bas du schéma avec autant de transistors.



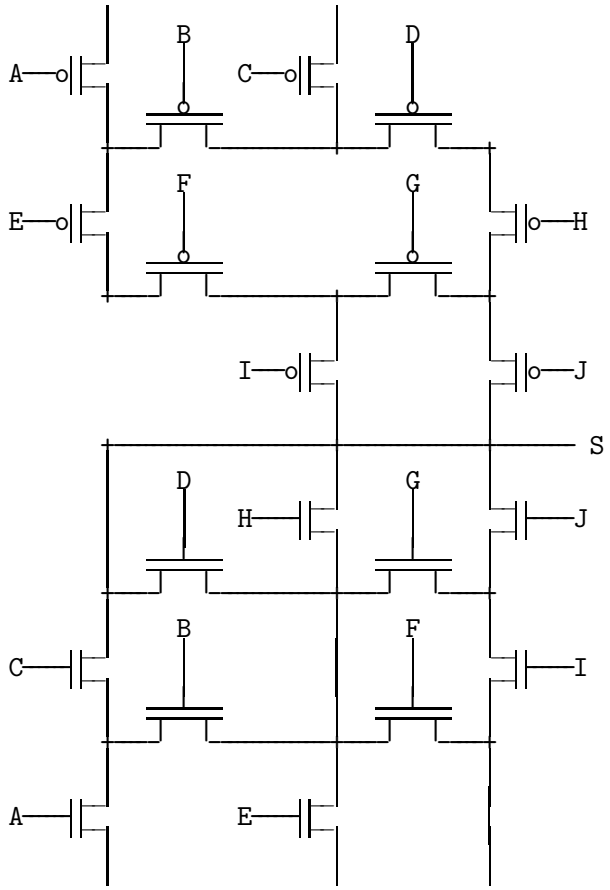

```
f:  loadimm16 r3,1
l1: sub   r0,r3,r0
    jc    l2
    load  r1,r4
    add   r2,r4,r2
    store r1,r2
    add   r1,r3,r1
    jmp   l1
l2: move  r2,r0
    ret
```

Donnez un équivalent simple en C de la fonction `f`.
Dans `int t[]={2,3,1,1}, x=f(3,t,1), y=f(3,t+1,2);` que valent `x` et `y` ?
Après leur calcul, que contient le tableau `t` ?

Ecrivez en assembleur la fonction `int g(int n, int *t);` qui rend la somme des carrés des `n` premiers éléments du tableau `t` sans modifier le contenu du tableau.

Corrigé

non signé	signé	CF	OF	ZF	SF
6+ 12= 2	6+ -4= 2	1	0	0	0
1+ 9=10	1+ -7=-6	0	0	0	1
6- 15= 7	6- -1= 7	1	0	0	0
11- 8= 3	-5- -8= 3	0	0	0	0
13+ 14=11	-3+ -2=-5	1	0	0	1
5- 4= 1	5- 4= 1	0	0	0	0
13- 4= 9	-3- 4=-7	0	0	0	1
4- 5=15	4- 5=-1	1	0	0	1
7- 10=13	7- -6=-3	1	1	0	1
10- 7= 3	-6- 7= 3	0	1	0	0
13- 15=14	-3- -1=-2	1	0	0	1
5+ 5=10	5+ 5=-6	0	1	0	1
0+ 2= 2	0+ 2= 2	0	0	0	0
9+ 9= 2	-7+ -7= 2	1	1	0	0



```
                  //         r0    r1     r2    r3   r4
f:  loadimm16 r3,1 //int f(int n,int*t,int s) //  1    x
l1: sub   r0,r3,r0 //{
    jc    l2       //  while(n--)
    load  r1,r4    //  { int x=*t;
    add   r2,r4,r2 //    s+=x;
    store r1,r2    //    *t=s;
    add   r1,r3,r1 //    t++;
    jmp   l1       //  }
l2: move  r2,r0    //  return s;
    ret            //}
int f(int n, int*t, int s) { while(n--) s+=*t, *t++=s; return s; }
```

Dans `int t[]={2,3,1,1}, x=f(3,t,1), y=f(3,t+1,2);`
`x` vaut 1+2+3+1=7 et après son calcul, `t` contient `{3,6,7,1}`.
`y` vaut 2+6+7+1=16 et après son calcul, `t` contient `{3,8,15,16}`.

```
                  //         r0    r1     r2    r3   r4
g:  loadimm16 r3,1 //int g(int n,int*t) //  s     1    x
    xor   r2,r2,r2 //{ int s=0;
l1: sub   r0,r3,r0 //
    jc    l2       //  while(n--)
    load  r1,r4    //  { int x=*t;
    mul   r4,r4,r4 //    x*=x;
    add   r2,r4,r2 //    s+=x;
    add   r1,r3,r1 //    t++;
    jmp   l1       //  }
l2: move  r2,r0    //  return s;
    ret            //}
```

Barème

1) 7pt=14x0.5pt

Chaque opération: 0 ou 0.5pt.

2) 4pt

-0.5pt pour toute erreur : Il manque 1 ou plusieurs !. Chaque lettre (commande) manquante ou en trop ou mal placée.

3) 6pt

`f` en C	2 pt-0.5pt par erreur comme argument manquant, test de boucle faux, incrément oublié
7	1 pt
16	1 pt
3 8 15 16	2 pt

4) 3pt

-0.5pt pour chaque erreur.