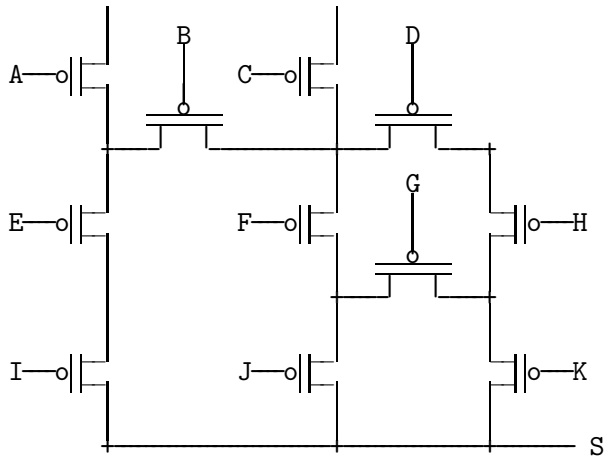
Université de Nanterre mardi 2 mai 2017
architecture des ordinateurs contrôle

Que valent CF, OF, ZF et SF après les opérations 1-7, 7+8, 6-3, 3+4, 13-4, 4-14, 6+15, 9-11, 6+6, 12-7, 15-9, 2-9, 15+15 et 8+9 sur des nombres codés sur 4 bits.

Complétez le bas du schéma avec autant de transistors.




```
f:  sub  r3,r2,r5
    loadimm16 r3,1
    xor  r4,r4,r4
l1: sub  r0,r3,r0
    jc   l2
    load r1,r6
    sub  r6,r2,r7
    mul  r6,r6,r6
    add  r6,r4,r6
    sub  r7,r5,r7
    movccbe r6,r4  // if(  <=  ) r4=r6   non signé
    add  r1,r3,r1
    jmp  l1
l2: mov  r4,r0
    ret
```

Donnez un équivalent simple en C de la fonction `f`.

Que vaut `x` dans `int t[]={3,6,1,5,3,4,2}, x=f(7,t,2,5);`

Donnez une formule donnant la valeur de `f(n,t,min,max)`. A défaut dîtes en une phrase ce que calcule `f`.

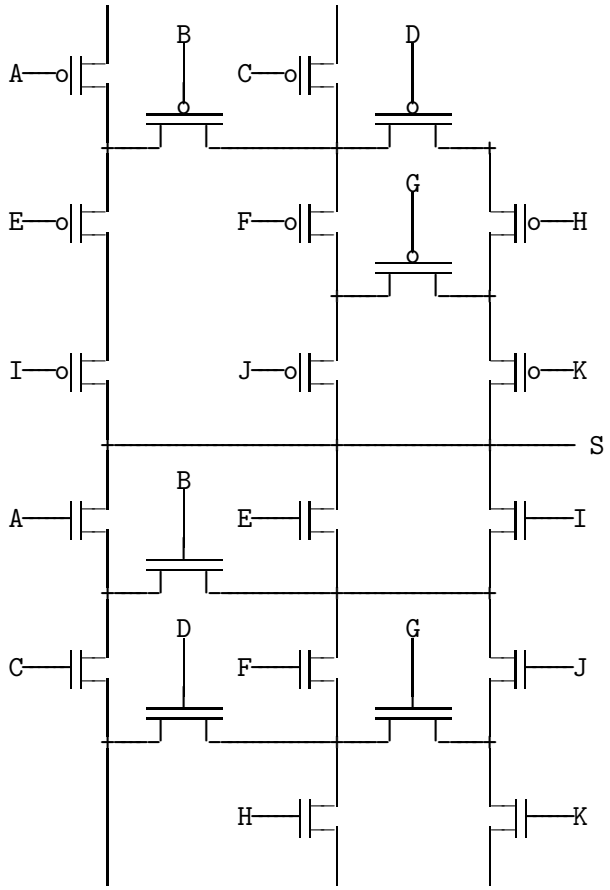
Même question si on remplace `movccbe` par `movccb`.

Redonnez la valeur de `x` et la formule de `f(n,t,min,max)` si on supprime la ligne `mul r6,r6,r6`.

Ecrivez en C puis en assembleur la fonction `int g(int n, int *t, int min, int max);` qui rend la somme des puissances quatrièmes des nombres qui parmi les `n` premiers éléments du tableau `t` ne sont pas compris dans l'intervalle [`min`, `max`].

Corrigé

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



```
f:  sub  r3,r2,r5 //         r0    r1     r2         r3   r4  r5       r6   r7
    loadimm16 r3,1//int f(int n,int*t,int min,int max)//1 s   d=       x    e=
    xor  r4,r4,r4 //{ int s=0;                          //     max-min  *t   *t-min
l1: sub  r0,r3,r0
    jc   l2       //  while(n--)             int f(int n, int*t,
    load r1,r6    //  { int x=*t,                  unsigned min, int max)
    sub  r6,r2,r7 //        e=x-min;         { while(n--)
    mul  r6,r6,r6 //    x*=x;                  { int x=*t++;
    add  r6,r4,r6 //    x+=s;                    if(x-min<=max-min) s+=x*x;
    sub  r7,r5,r7 //                           }
    movccbe r6,r4 //    if(e+0u<=d+0u) s=x;    return s;
    add  r1,r3,r1 //    t++;                 }
    jmp  l1       //  }
l2: mov  r4,r0    //  return s;
    ret           //}
```

$$x = 3^2 + 5^2 + 3^2 + 4^2 + 2^2 = 63$$

$$f(n, t, min, max) = \sum_{\substack{0 \le i < n \\ min \le t[i] \le max}} t[i]^2$$

C'est la somme des carrés des nombres contenus dans le tableau `t` de dimension `n` qui sont compris entre `min` et `max`.

$$x = 3^2 + 3^2 + 4^2 + 2^2 = 38$$

$$f(n, t, min, max) = \sum_{\substack{0 \le i < n \\ min \le t[i] < max}} t[i]^2$$

$$x = 3 + 5 + 3 + 4 + 2 = 17$$

$$f(n, t, min, max) = \sum_{\substack{0 \le i < n \\ min \le t[i] \le max}} t[i]$$

```
int g(int n, int*t, unsigned min, int max)
{ while(n--)
  { int x=*t++;
    if(x-min>max-min) s+=x*x*x;
  }
  return s;
}
```

Dans la fonction `f` donnée dans l'énoncé on remplace `movccbe` par `movcca` et la ligne

```
mul  r6,r6,r6
```

par

```
mul  r6,r6,r6
mul  r6,r6,r6
```

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	1.5 pt-0.5pt par erreur comme argument manquant, test de boucle faux, incrément oublié
63	0.75pt
formule	0.75pt
38	0.75pt
formule	0.75pt
17	0.75pt
formule	0.75pt

4) 3pt

`g` en C	1 pt
movccb	1 pt
2 `mul` pour x^4	1 pt

-0.5pt pour chaque erreur.