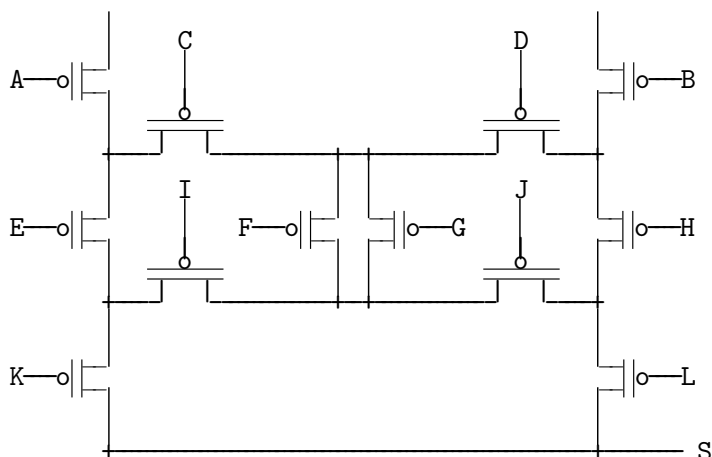


Que valent CF , OF , ZF et SF après les opérations 13-0, 3-5, 5-5, 15-11, 0+15, 1-10, 1+3, 7-15, 11-15, 9-2, 12+15, 6+14, 5+5, 10+11, sur des nombres codés sur 4 bits.

Complétez le bas du schéma avec autant de transistors.



```
f:  loadimm16 r3,1
    xor  r4,r4,r4
11:  sub  r0,r3,r0
    jc   12
    load r1,r5
    load r2,r6
    sub  r5,r6,r7
    sub  r6,r5,r6
    movcl r7,r6 // if(...*...) mov r7,r6
    add  r4,r6,r4
    add  r1,r3,r1
    add  r2,r3,r2
    jmp  11
12:  mov  r4,r0
    ret
```

Donnez un équivalent simple en C de la fonction f.

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

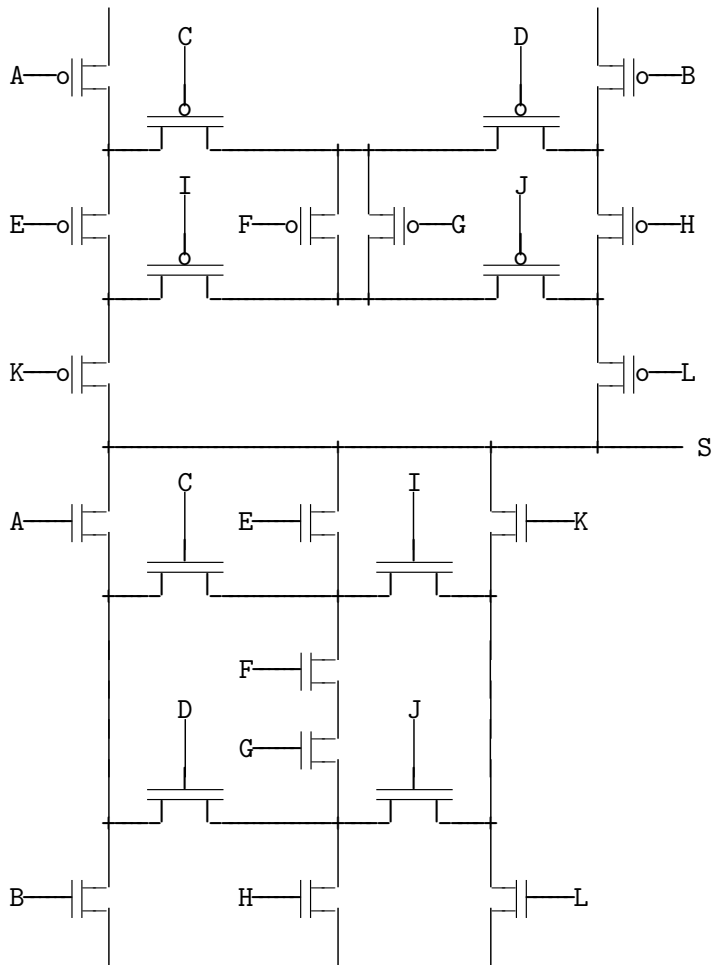
Dites en une phrase ce que calcule f.

Redonnez la valeur de x, si on remplace `movcl r7,r6` par `movcg r7,r6`
ou bien si on le remplace par `movcb r7,r6`

Ecrivez en C puis en assembleur la fonction `int g(int *t);` qui rend $|t[0] - t[1]| - |t[2] - t[3]|$.

Corrigé

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



```

//          r0    r1    r2    r3  r4 r5 r6  r7
f: loadimm16 r3,1 // int f(int n, int*t, int*u) // 1  s  x  y  d=x-y
xor  r4,r4,r4 // { int s=0;
11: sub  r0,r3,r0 // while(n--)
    jc  12
    load r1,r5 // { int x=*t,
    load r2,r6 //      y=*u,
    sub  r5,r6,r7 //      d=x-y;
    sub  r6,r5,r6 //      x=y<x? d : y-x;
    movcl r7,r6
    add  r4,r6,r4 //      s+=x;
    add  r1,r3,r1 //      t++;
    add  r2,r3,r2 //      u++;
    jmp  11 // }
12: mov  r4,r0 // return s;
    ret // }
int f(int n, int*t, int*u)
{ int s=0;
  while(n--) { int x=*t++, y=*u++; s+=y<x? x-y :y-x; }
  return s;
}

```

$$x = |1 - (-2)| + |5 - 3| + |-2 - 2| = 3 + 2 + 4 = 9$$

On passe à f deux tableaux d'entiers après leur longueur commune. $f(n, t, u) = \sum_{i=0}^{n-1} |t[i] - u[i]|$

est la somme des valeurs absolues des différences des éléments correspondants des deux tableaux.

Si on remplace `movcl r7,r6` par `movcg r7,r6`, la condition est inversée, et on prendra le minimum de $x-y$ et $x+y$ au lieu du maximum. Donc on prendra $-|x-y|$ au lieu $|x-y|$. Donc $x = -9$.

Si on remplace `movcl r7,r6` par `movcb r7,r6` la comparaison est entre des nombres non signés. Donc -2 est compris comme $2^{32} - 2$ et est plus grand que tout nombre positif. Donc le signe de la valeur absolue de la différence est changé quand on compare un nombre positif à un négatif. $x = -|1 - (-2)| + |5 - 3| - |-2 - 2| = -3 + 2 - 4 = -5$

```

g: loadimm16 r3,1 // int g(int *t)
    add  r0,r3,r1 // t+1
    add  r1,r3,r2 // t+2
    add  r2,r3,r3 // t+3
    load r0,r0 // t[0]
    load r1,r1 // t[1]
    load r2,r2 // t[2]
    load r3,r3 // t[3]
    sub  r0,r1,r4 // t[0]-t[1]
    sub  r1,r0,r5 // t[1]-t[0]
    movcl r4,r5 // |t[0]-t[1]|
    sub  r2,r3,r0 // t[2]-t[3]
    sub  r3,r2,r1 // t[3]-t[2]
    movcl r0,r1 // |t[2]-t[3]|
    sub  r5,r1,r0 // |t[0]-t[1]| - |t[2]-t[3]|
    ret // return y-b;
}

```

Barème

1) 4pt

Chaque opération partiellement fausse ou manquante : -0.3pt.

2) 6pt

-1pt pour un transistor dont une patte est mal raccordée.

-1.5pt pour un transistor dont les deux pattes sont mal raccordées.

3) 6pt

f en C 1.5pt-0.3pt par erreur comme argument manquant, test de boucle faux, incrément oublié

x=9 1pt

f en français 1.5pt

x=-9 1pt

x=-5 1pt

4) 4pt

g en C 1pt

g en assembleur 3pt

-0.3pt pour chaque instruction fausse ou manquante.