



**Exercices en préparation de l'examen  
final du cours  
INF3610 Systèmes embarqués**

**Chapitre 3 – Bloc matériel**

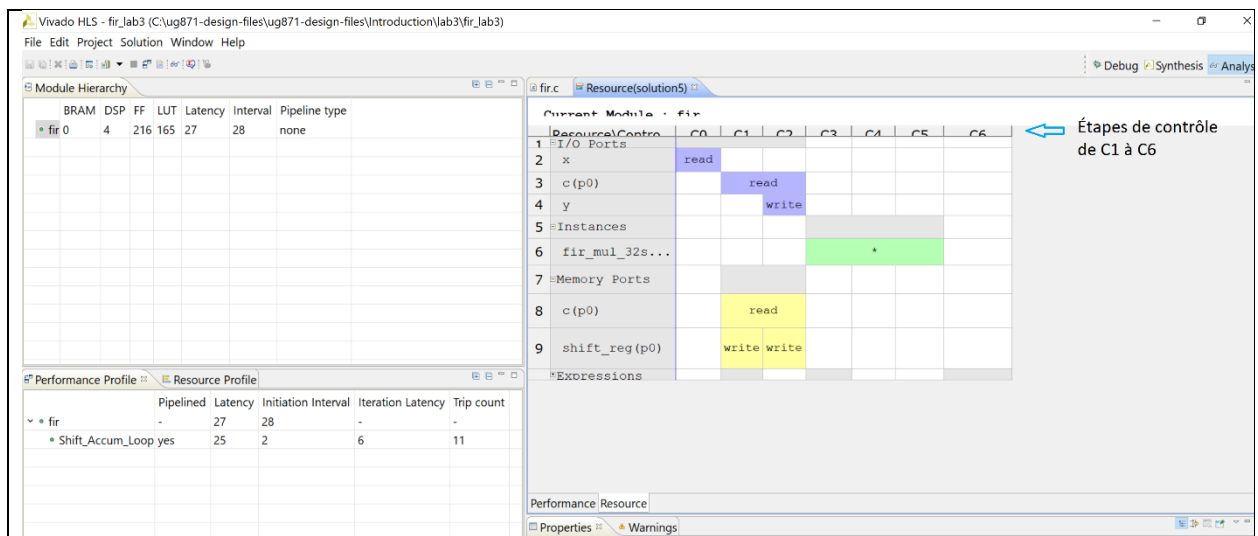
## Question 1 Optimisation pour synthèse HLS

La figure 1.1 représente le code du FIR de dimension 11 similaire à celui présenté en classe.

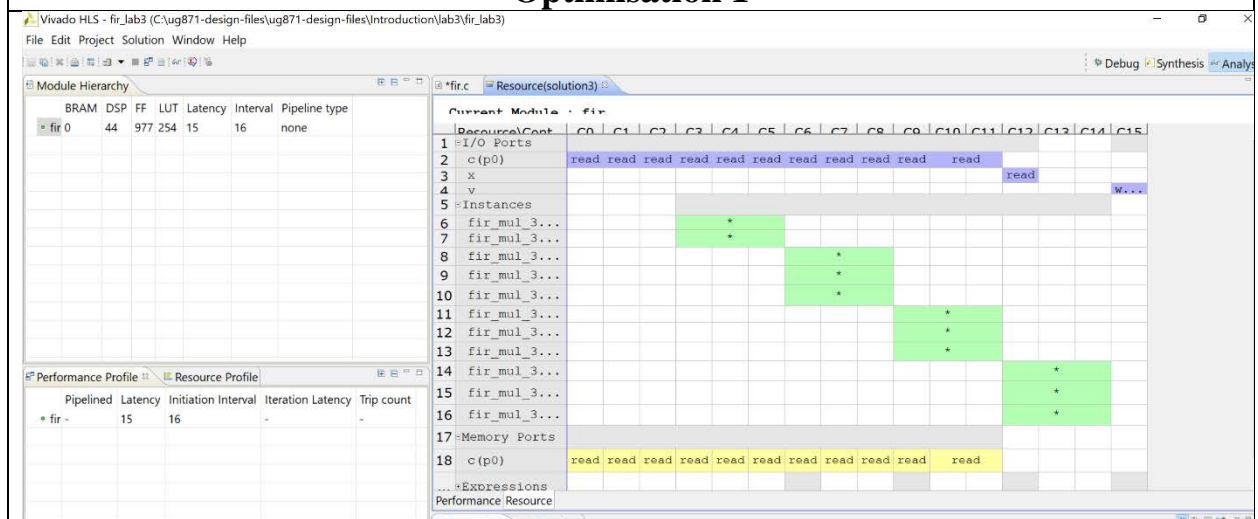
```
1 #include "fir.h"
2
3 void fir (
4     data_t *y,
5     coef_t c[N],
6     data_t x
7 ) {
8     #pragma HLS INTERFACE ap_vld port=y //ajout d'un protocole handshacking pour
9     #pragma HLS INTERFACE ap_vld port=x //synchorniser les données d'entrée
10    #pragma HLS RESOURCE variable=c core=RAM_1P_BRAM
11
12
13    static data_t shift_reg[N];
14    acc_t acc;
15    data_t data;
16    int i;
17
18    acc=0;
19    Shift_Accum_Loop: for (i=N-1;i>=0;i--) {
20        if (i==0){
21            shift_reg[0]=x;
22            data = x;
23        } else {
24            shift_reg[i]=shift_reg[i-1];
25            data = shift_reg[i];
26        }
27        acc+=data*c[i];;
28    }
29    *y=acc;
30 }
```

**Figure 1.1**

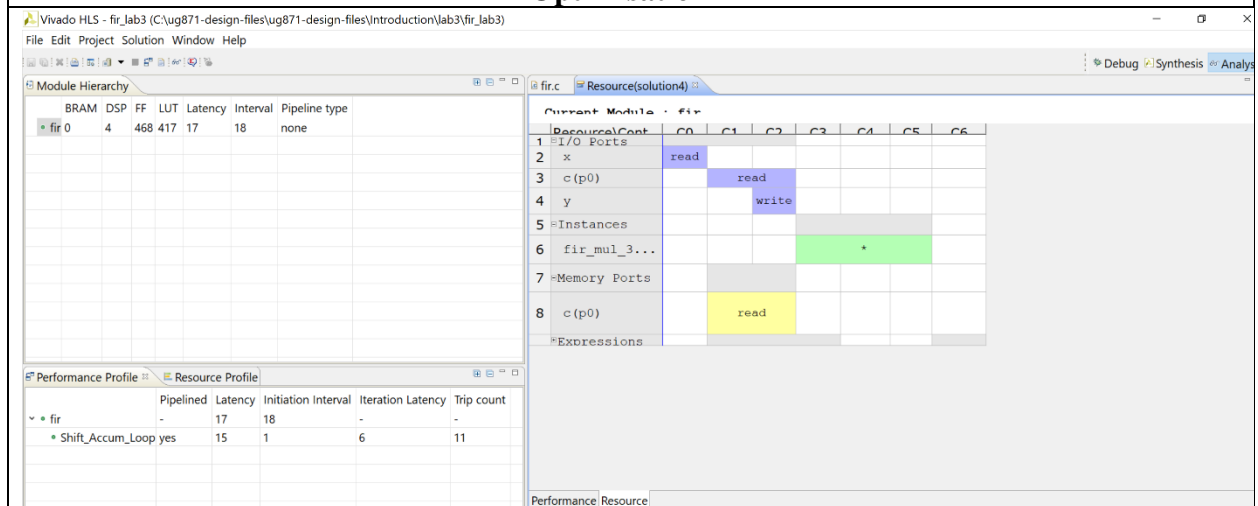
On décide d'optimiser ce code dans HLS Vivado. Pour chacune des 3 optimisations de la page suivante, indiquez-la (les) directive(s) (pragma) qui a (ont) été utilisé(es). Justifiez bien votre réponse. Voir l'annexe au besoin.



## Optimisation 1



## Optimisation 2



## Optimisation 3

## Question 2 Synthèse de haut niveau

- a) En classe, j'ai souvent dit que les architectures RISC de type superscalaire, ou encore VLIW, étaient utiles pour une parallélisation à gros grains alors que le FPGA était utile pour une parallélisation à grains fins. Expliquez dans vos mots ce que cela signifie.
- b) Expliquez le rôle du code suivant (Figure 2.1) et de sa transformation avec HLS Vivado (Figure 2.2). Expliquez également l'efficacité de ce pragma, c'est-à-dire obtient-on les résultats escomptés à chaque fois?

```
for(int i = 0; i < X; i++) {  
    pragma HLS unroll factor=2  
    a[i] = b[i] + c[i];  
}
```

Figure 2.1

```
for(int i = 0; i < X; i += 2) {  
    a[i] = b[i] + c[i];  
    if (i+1 >= X) break;  
    a[i+1] = b[i+1] + c[i+1];  
}
```

Figure 2.2