

UNIVERSIDAD CARLOS III DE MADRID

Tema 5. Segmentación

Departamento de Ingeniería de Sistemas y
Automática

RAÚL PÉRULA MARTÍNEZ
LUIS ENRIQUE MORENO LORENTE
ALBERTO BRUNETE GONZALEZ
CESAR AUGUSTO ARISMENDI GUTIERREZ
DOMINGO MIGUEL GUINEA GARCIA ALEGRE



Universidad
Carlos III de Madrid



Esta obra se publica bajo una licencia Creative Commons Reconocimiento-NoComercial-CompartidIgual 3.0 España.

Nota: Considerar la arquitectura de la última página.

Ejercicio 1

Suponiendo que el siguiente código en C:

```
for (i=0;i<100;i++) {  
    if(X[i] > 0) {  
        Y[i] = X[i] + Y[i] + Y[i+1];  
    }  
    else {  
        Y[i] = X[i];  
    }  
}
```

Tiene la siguiente traducción a un cierto lenguaje ensamblador:

```
      L0:  ld    r3, #100  
          ld    r1, DIR_X  
          ld    r2, DIR_Y  
1     L1:  ld    f2, 0(r1)  
2          blt  f2, 0,L3  
3     L2:  ld    f4, 0(r2)  
4          ld    f6, 4(r2)  
5          add  f8, f2, f4  
6          add  f8, f8, f6  
7          store 0(r2), f8  
8          br   L4  
9     L3:  store 0(r2), f2  
10    L4:  add   r1, r1, #4  
11          add  r2, r2, #4  
12          sub  r3, r3, #-4  
13          bgt  r3, 0, L1
```

Donde se lista en primer lugar el operando de destino y a continuación los operandos fuente, y en el caso de las instrucciones de salto se listan primero los operandos a ser comparados y después la dirección de salto.

Se pide:

1. Obtener el cronograma de ejecución para los dos casos posibles del condicional suponiendo que la arquitectura del procesador es la DLX.



Ejercicio 2

Suponiendo que el siguiente código en C:

```
for (i=0;i<100;i++) {  
    if(X[i] > 0) {  
        Y[i] = X[i] * X[i] - Y[i] + Y[i+1];  
    }  
    else {  
        Y[i] = X[i] * X[i];  
    }  
}
```

Tiene la siguiente traducción a un cierto lenguaje ensamblador:

```
      L0:  ld    r3, #100  
          ld    r1, DIR_X  
          ld    r2, DIR_Y  
1     L1:  ld    f2, 0(r1)  
2          mul  f4, f2, f2  
3          blt  f2, 0, L3  
4     L2:  ld    f6, 0(r2)  
5          ld    f8, 4(r2)  
6          sub  f6, f4, f6  
7          add  f6, f8, f6  
8          store 0(r2), f6  
9          br   L4  
10    L3:  store 0(r2), f4  
11    L4:  add  r1, r1, #4  
12          add  r2, r2, #4  
13          sub  r3, r3, #-4  
14          bgt  r3, 0, L1
```

Donde se lista en primer lugar el operando de destino y a continuación los operandos fuente, y en el caso de las instrucciones de salto se listan primero los operandos a ser comparados y después la dirección de salto.

Se pide:

1. Obtener el cronograma de ejecución para los dos casos posibles del condicional suponiendo que la arquitectura del procesador es la DLX (con planificación estática).

Ejercicio 3

Suponiendo que el siguiente código en C:

```
for (i=0;i<100;i++) {  
    A[i] = B[i] + C[i];  
}
```

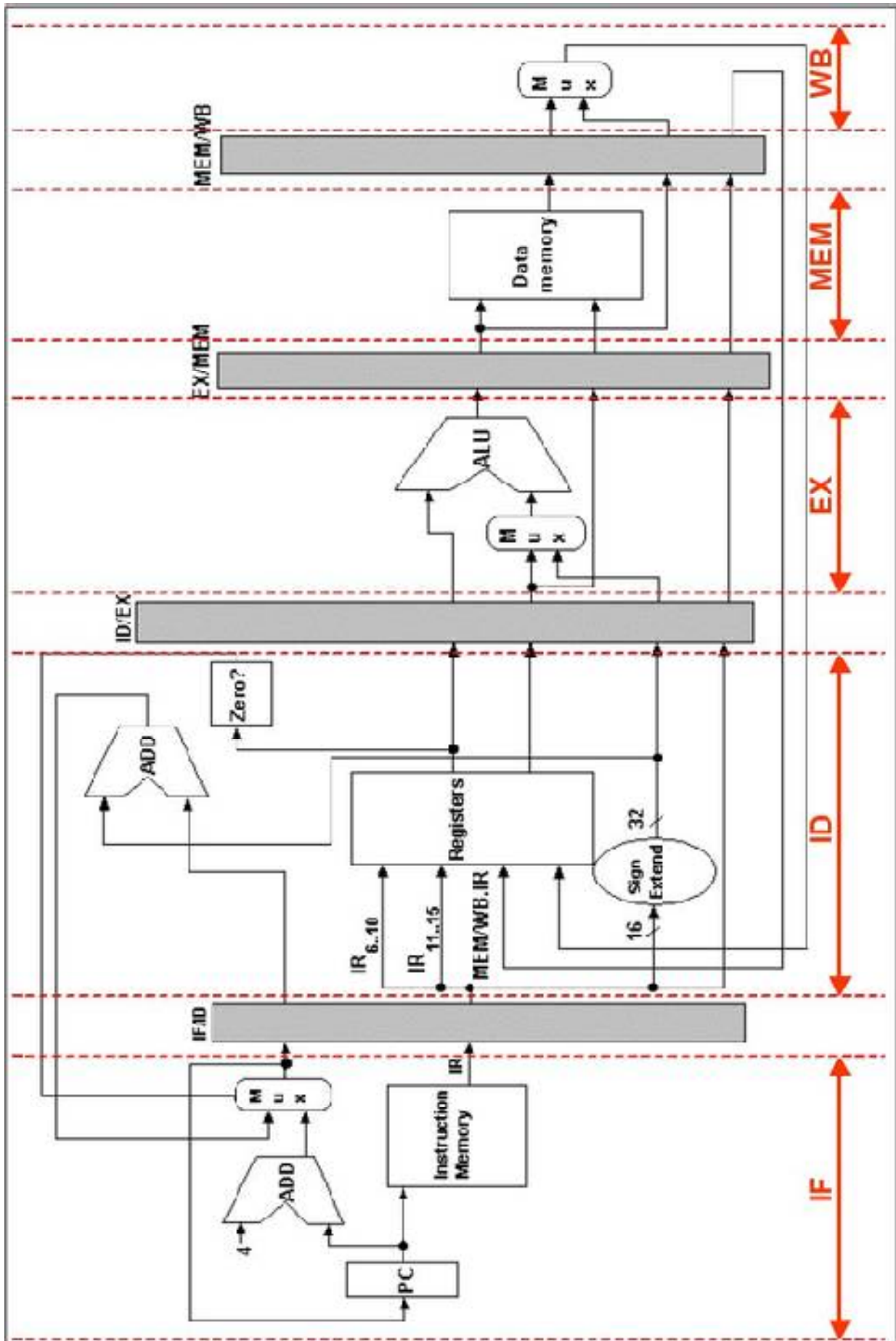
Tiene la siguiente traducción a un cierto lenguaje ensamblador:

```
L0:  ld    r4, #100  
      ld    r1, DIR_A  
      ld    r2, DIR_B  
      ld    r3, DIR_C  
1  L1:  ld    f2, 0(r2)  
2      ld    f4, 0(r3)  
3      add   f6, f4, f2  
4      store 0(r1), f6  
5      add   r1, r1, #4  
6      add   r2, r2, #4  
7      add   r3, r3, #4  
8      sub   r4, r4, #-4  
9      bgt  r4, 0, L1
```

Donde se lista en primer lugar el operando de destino y a continuación los operandos fuente, y en el caso de las instrucciones de salto se listan primero los operandos a ser comparados y después la dirección de salto.

Se pide:

1. Obtener el cronograma de ejecución para dos iteraciones sucesivas suponiendo que la arquitectura del procesador es la DLX (con planificación estática).



Copyright © 2000 - 2012 Mission Level Design GmbH. All Rights Reserved.