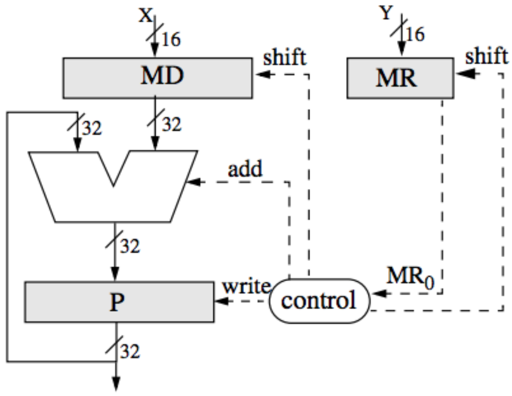
EC Examen de Problemes (SOLUCIONS)

Exercici 1 (Examen Parcial 2016/2017 Q2)

Sigui el circuit seqüencial per a la multiplicació de números naturals de 16 bits, anàleg a l'estudiat a classe, el qual calcula el producte en 32 bits:



Escriu usant el llenguatge C l'algorisme que descriu les operacions que ha de realitzar la unitat de control (anàleg al pseudocodi explicat a classe). És a dir, programa la següent funció multu que té els registres MD, MR i P com a variables locals, i els operands X, Y com a paràmetres, i on el producte és el resultat a retornar:

```
+------------------------------------------------------------+
| unsigned int multu(unsigned short X, unsigned short Y) {   |
| unsigned int MD, P;                                         |
| unsigned short MR;                                          |
|                                                             |
| int i;                                                      |
| MD =                                                        |
| MR =                                                        |
| P =                                                         |
| for (i=1; i<=16; i++)                                       |
| {                                                           |
|                                                             |
|    /* EL TEU CODI */                                        |
|                                                             |
|    return P;                                                |
| }                                                           |
+-------------------------------------------------------------
```

Exercici 2 (problema 5.29 de la col.lecció)

Suposem que \$f2=0x42000000 i \$f4=0x3d800000, i que executem la instrucció: mul.s \$f6, \$f2, \$f4. Suposant que el sumador té 1 bit de guarda, un d'arrodoniment i un de "sticky", i que arrodoneix al més pròxim (al parell en el cas equidistant) ¿quin és el valor final de \$f6 en hexadecimal?

```
0x42000000 = 0|100 0010 0|000 0000... = 1,0 x 2^5
0x3d800000 = 0|011 1101 1|000 0000... = 1,0 x 2^(-4)

1,0 x 2^5 * 1,0 x 2^(-4) = 1,0 x 2^1

= 0|100 0000 0|000 0000... = 0x4000
```

Exercici 3 (Examen Parcial 2016/2017 Q2)

Considera que el contingut dels registres \$f4 i \$f6 és 0xBE80000C i 0x40800000, respectivament i que s'executa la instrucció MIPS: add.s \$f0,\$f4,\$f6. Suposant que el sumador/restador té 1 bit de guarda, un d'arrodoniment i un de "sticky", i que arrodoneix al més pròxim (al parell en el cas equidistant), quin és el valor de \$f0 en hexadecimal després d'executar la instrucció? Quin és el el valor absolut de l'error de precisió comès en aquest càlcul?

```
Solució: 0x406FFFFE
Error: 2^-23
```

Exercici 4 (Examen Final 2011/2012 Q2)

a) Suposant que els valors inicials de \$f6 i \$f8 són \$f6=0x40D00003, \$f8=0xBE80000C, i que les operacions arrodoneixen el resultat al valor més pròxim ¿quin serà el valor de \$f10, en hexadecimal, després d?executar la instrucció: add.s \$f10, \$f6, \$f8?

b) Calcula l'error per pèrdua de precisió en el resultat anterior, expressant-lo en notació científica: error = $x * 2^y$ (on x i y són números en base 10):

Solució:

a) 0x40C80002

b) $1 * 2^{-23}$

Exercici 5 (Examen Final 2012/2013 Q1)

Considera que el contingut dels registres \$f2 i \$f4 és 0x01820003 i 0x81700003, respectivament i que s'executa la instrucció MIPS: add.s \$f0,\$f2,\$f4. Suposant que el sumador/restador té 1 bit de guarda, un d'arrodoniment i un de "sticky", i que arrodoneix al més pròxim (al parell en el cas equidistant), contesta a les següents preguntes:

1. Es pot representar el resultat en el format normalitzat de simple precisió (Si/No)? Per què?
2. Es pot representar el resultat en algun altre format de l'estàndar IEEE-754 en simple precisió (Si/No)? De quina manera?

Solució:

1. No. Perquè l'exponent del resultat és -128, i es troba fora del rang representable per als valors normalitzats en simple precisió, que és [-126, +127]. És a dir, que es produeix un "Underflow"
2. Sí. En format "Denormal" (exponent = -126, que es codifica amb 8 bits a zero). $Resultat = 0,01010000000000000000110 * 2^{-126} = 0x00280006$

Exercici 6 (problema 5.30 de la col·lecció)

Tradueix a assemblador MIPS la subrutina absdif:

```
float absdif (float a, float b)
{
    if (a>b)
        return a-b;
    else
        return b-a;
}
```

Sol:

```
absdif:
    c.lt.s    $f14, $f12            # bit de condició = (b<a)
    bc1f    else            # salta si bit de condició fals
    sub.s    $f0, $f12, $f14            # resultat = a-b
    b    fisi
else:
    sub.s    $f0, $f14, $f12            # resultat = b-a
fisi:
    jr    $ra
```