Implémentation de la division sur FPGA


Pierre Langlois

http://creativecommons.org/licenses/by-nc-sa/2.5/ca/

Implémentation de la division sur FPGA
Sujets de ce thème

- Division par une constante, puissance de deux
- Division par une constante par multiplication par sa réciproque
- Division générale par multiplication par la réciproque, gardée dans une ROM
- Division au long
- Division par convergence, par multiplications successives

Définition des termes

- On a N / D = Q , où:
  - N est le dividende (numérateur)
  - D est le diviseur (dénominateur)
  - Q est le quotient
- Pour des nombres entiers, on peut aussi considérer le reste R et la relation

$$N = Q \times D + R$$

- Par exemple, on a 37 ÷ 5 = 7, reste 2, ou

$$37 = 7 \times 5 + 2$$

Division par une constante puissance de deux

- Dans le cas le plus simple, le diviseur est une puissance de deux, $D = 2^k$.
- Le quotient Q est obtenu en décalant le dividende vers la droite de *k* positions.
  - Il y a en général une perte de précision si le nombre de bits du quotient Q est le même que celui du dividende N
  - Si k est plus grand que le nombre de bits du dividende N, le quotient Q est 0
- Le reste R est obtenu en choisissant les *k* bits les moins significatifs.
- Les synthétiseurs de code VHDL peuvent traiter ces opérations.

$$\lfloor 46 \div 2^1 \rfloor = \lfloor 46 \div 2 \rfloor = 23$$
$$\lfloor 46 \div 2^2 \rfloor = \lfloor 46 \div 4 \rfloor = 11$$

$$\lfloor 101110_2 \div 2^1 \rfloor = 10111_2 = 23$$
$$\lfloor 101110_2 \div 2^2 \rfloor = 1011_2 = 11$$

$$247 \,\%\, 4 = 247 \,\%\, 2^2 = 3$$
$$247\% \; 16 = 247 \,\%\, 2^4 = 7$$
$$247 \,\%\, 64 = 247 \,\%\, 2^6 = 55$$

$$1111\ 0111_2 \,\%\, 2^2 = 11_2 = 3$$
$$1111\ 0111_2 \,\%\, 2^4 = 0111_2 = 7$$
$$1111\ 0111_2 \,\%\, 2^6 = 11\ 0111_2 = 55$$

Division par une constante via la multiplication par sa réciproque

- Pour faire la division d'un nombre par une constante, une approche consiste à exprimer le diviseur par une fraction dont le dénominateur est une puissance de deux.
- Par exemple, pour effectuer une division par 10 avec une précision de 8 bits, on a:

$$N / 10 = N \times (\text{arrondi}(2^8 / 10) / 2^8)$$
$$\cong (N \times 26) / 256)$$
$$= (N \times 26) >> 8$$

  - La multiplication produit un nombre de 16 bits dont on ne garde que les 8 bits les plus significatifs.
  - La précision du résultat est d'au moins 7 bits.

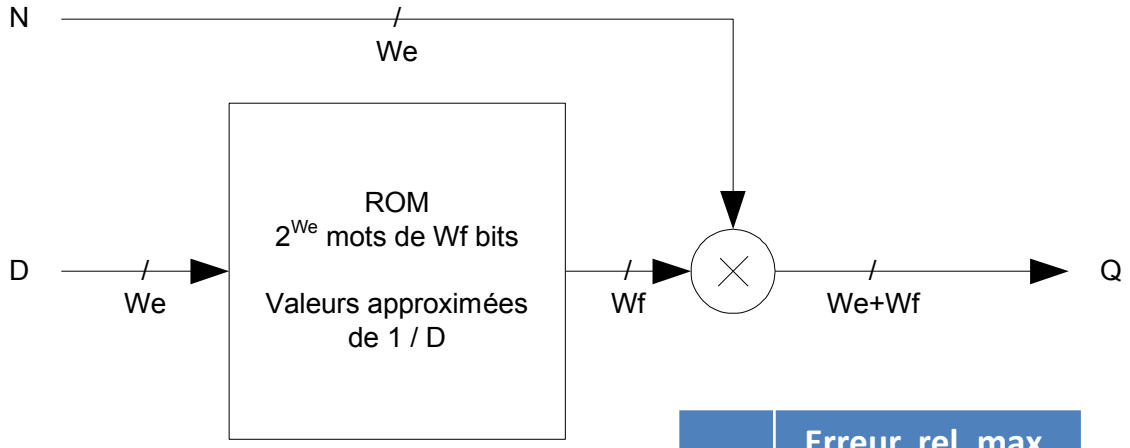
```
library ieee;
use ieee.numeric_std.all;

entity divisionPar10 is
   port (
      N : in unsigned(7 downto 0);
      Q : out unsigned(7 downto 0)
   );
end divisionPar10;

architecture arch1 of divisionPar10 is
begin

   process(N)
   -- 256 / 10 ~= 26
   constant facteur : unsigned(7 downto 0) := to_unsigned(26, 8);
   variable produit : unsigned(15 downto 0);
   variable Qvar : unsigned(7 downto 0);
   begin
      produit := N * facteur;
      Qvar := produit(15 downto 8);
      Q <= Qvar;
      report integer'image(to_integer(N)) & " / 10 =? "
           & integer'image(to_integer(Qvar));
   end process;

end arch1;
```

Division générale par multiplication par la réciproque

- Si le diviseur D n'est pas une constante mais qu'il peut s'exprimer sur peu de bits, on peut garder un tableau de réciproques en mémoire.
- L'opération de division correspond alors à chercher en mémoire la valeur de la réciproque 1/D, puis à multiplier cette valeur par le dividende.
- Cette approche peut être valable par exemple si on doit calculer le rapport des intensités de deux pixels, exprimés sur W_e = 8 bits.
- Le principe est simple. La complexité du code relève des alignements des bits des différentes quantités. Les types en virgule fixe simplifient grandement l'écriture du code en VHDL.
- Il faut traiter correctement le cas D = 0.

N et D sont des entiers exprimés avec W_e bits.

Le quotient Q est exprimé avec W_e bits pour la partie entière et W_f bits pour la partie fractionnaire.

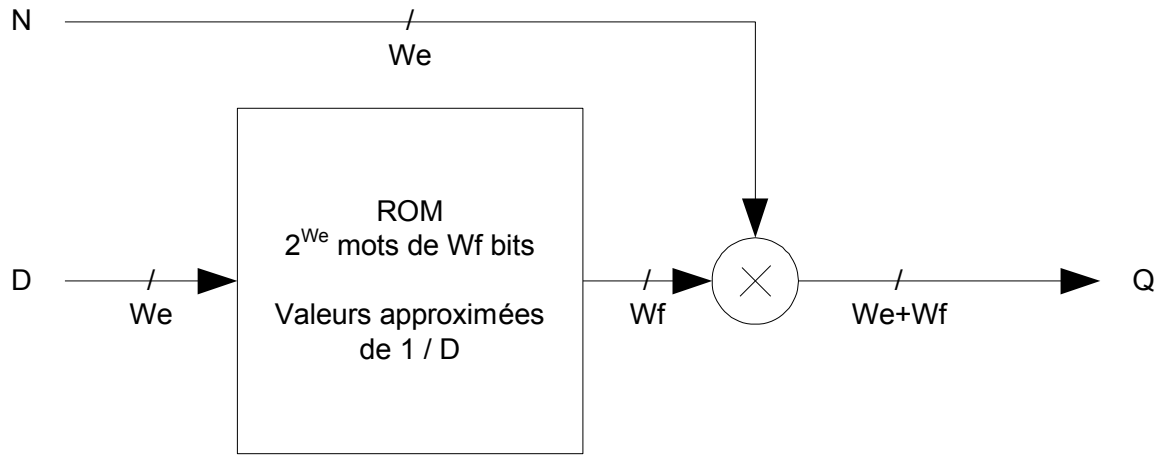


Wf	Erreur rel. max. pour We = 8
8	33.2%
10	10.9%
12	2.93%
14	0.77%
16	0.18%

Division générale par multiplication par la réciproque

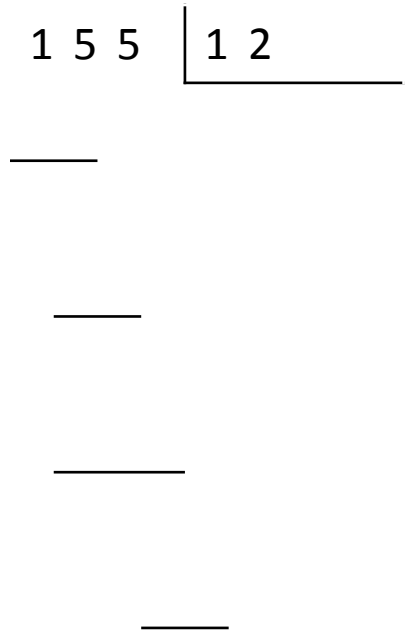
```
library ieee;
use ieee.fixed_pkg.all;
use ieee.std_logic_1164.all;

entity divisionparreciproque is
    generic(
        We : integer := 8; -- nombre de bits pour la partie entière
        Wf : integer := 12 -- nombre de bits pour la partie fractionnaire
    );
    port (
        N, D : in ufixed(We - 1 downto 0);
        Q : out ufixed(We - 1 downto -Wf);
        erreur : out std_logic -- '1' si D = 0, division par 0
    );
end divisionparreciproque;

architecture arch of divisionparreciproque is

type ROM_reciproque_type is array(0 to (2 ** We) - 1) of ufixed(0 downto -Wf);

-- La ROM entrepose les réciproques de D, exprimées sur Wf bits.
-- Les valeurs pour 0 et 1 sont sans importance - on retourne une erreur
-- si D == 0 et N si D == 1.
function init_mem return ROM_reciproque_type is
variable reciproque : ROM_reciproque_type;
begin
--  reciproque(0) := to_ufixed(0.0, reciproque(0));
--  reciproque(1) := to_ufixed(0.0, reciproque(1));
    for i in 2 to 2 ** We - 1 loop
        reciproque(i) := to_ufixed(1.0 / real(i), reciproque(i));
    end loop;
    return reciproque;
end init_mem;
```

```
-- la mémoire ROM est initialisée lors de l'instanciation du module
constant ROM_reciproque : ROM_reciproque_type := init_mem;

begin

    erreur <= '1' when D = 0 else '0';

    process(N, D)
    variable Qvar : ufixed(We downto -Wf);
    begin
        if D = 1 then
            Qvar := resize(N, Qvar);
        else
            Qvar := N * ROM_reciproque(to_integer(D));
        end if;
        Q <= Qvar(We - 1 downto -Wf);
    end process;

end arch;
```



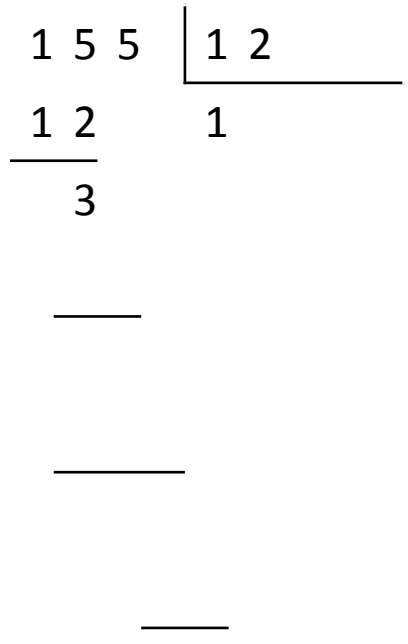
La division au long

- La division au long est un algorithme qui se prête bien aux calculs manuels avec papier et crayon.
- Les opérations arithmétiques à maîtriser sont la multiplication et la soustraction.
- En base 10, la difficulté est de déterminer combien de fois le diviseur se place dans une partie du dividende.



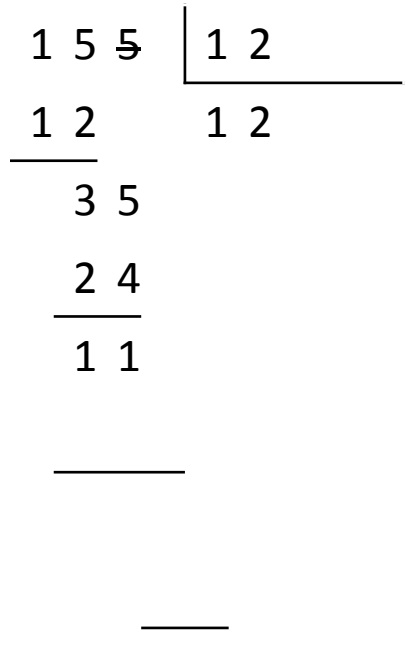
La division au long

- La division au long est un algorithme qui se prête bien aux calculs manuels avec papier et crayon.
- Les opérations arithmétiques à maîtriser sont la multiplication et la soustraction.
- En base 10, la difficulté est de déterminer combien de fois le diviseur se place dans une partie du dividende.



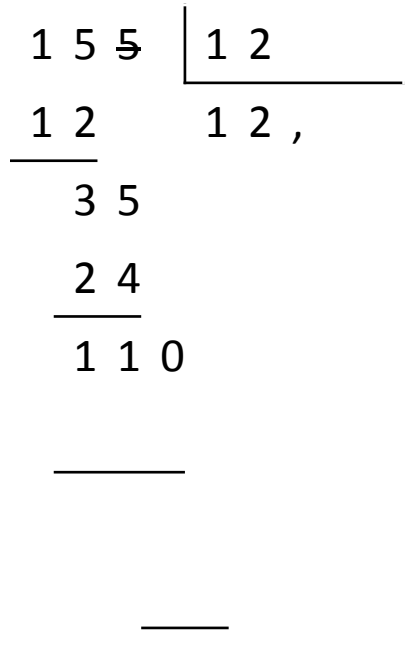
La division au long

- La division au long est un algorithme qui se prête bien aux calculs manuels avec papier et crayon.
- Les opérations arithmétiques à maîtriser sont la multiplication et la soustraction.
- En base 10, la difficulté est de déterminer combien de fois le diviseur se place dans une partie du dividende.



La division au long

- La division au long est un algorithme qui se prête bien aux calculs manuels avec papier et crayon.
- Les opérations arithmétiques à maîtriser sont la multiplication et la soustraction.
- En base 10, la difficulté est de déterminer combien de fois le diviseur se place dans une partie du dividende.



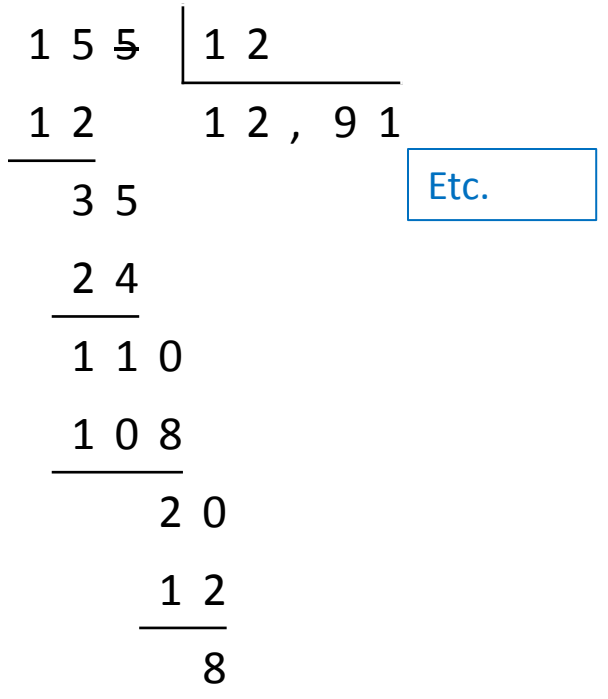
La division au long

- La division au long est un algorithme qui se prête bien aux calculs manuels avec papier et crayon.
- Les opérations arithmétiques à maîtriser sont la multiplication et la soustraction.
- En base 10, la difficulté est de déterminer combien de fois le diviseur se place dans une partie du dividende.

```
 1 5 5  | 1 2
 1 2    |------
 ------ | 1 2 , 9
   3 5
   2 4
 ------
   1 1 0
   1 0 8
 -------
       2
```

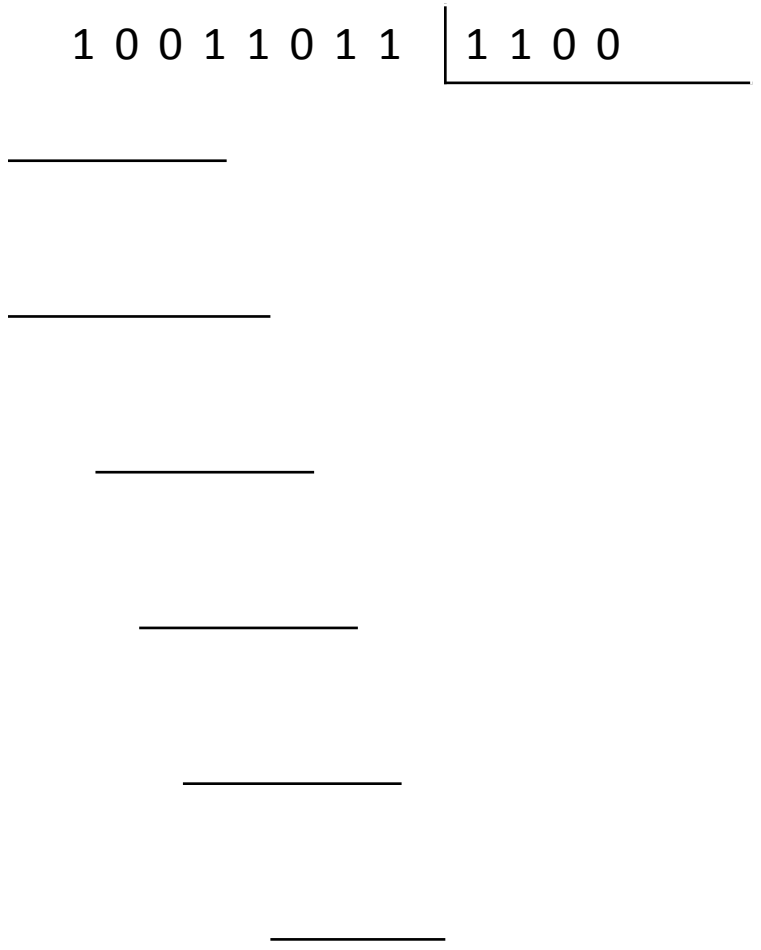
La division au long

- La division au long est un algorithme qui se prête bien aux calculs manuels avec papier et crayon.
- Les opérations arithmétiques à maîtriser sont la multiplication et la soustraction.
- En base 10, la difficulté est de déterminer combien de fois le diviseur se place dans une partie du dividende.
- À chaque étape on obtient un chiffre de plus du quotient.



La division au long en binaire

- En binaire, le processus est identique, mais simplifié parce que les seuls facteurs possibles sont 0 et 1.
- À chaque étape on obtient un chiffre de plus du quotient.



La division au long en binaire

- En binaire, le processus est identique, mais simplifié parce que les seuls facteurs possibles sont 0 et 1.
- À chaque étape on obtient un chiffre de plus du quotient.

```
  1 0 0 1 1 0 1 1  | 1 1 0 0
- 0 0 0 0          | 0
-------
  1 0 0 1 1
```

La division au long en binaire

- En binaire, le processus est identique, mais simplifié parce que les seuls facteurs possibles sont 0 et 1.
- À chaque étape on obtient un chiffre de plus du quotient.

```
  1 0 0 1 1 0 1 1  | 1 1 0 0
- 0 0 0 0          | 0 1
---------
  1 0 0 1 1
-   1 1 0 0
---------
    0 1 1 1 0
```

La division au long en binaire

- En binaire, le processus est identique, mais simplifié parce que les seuls facteurs possibles sont 0 et 1.
- À chaque étape on obtient un chiffre de plus du quotient.

```
  1 0 0 1 1 0 1 1  | 1 1 0 0
- 0 0 0 0          | 0 1 1
---------
  1 0 0 1 1
-   1 1 0 0
---------
    0 1 1 1 0
  -   1 1 0 0
  -------------
      0 0 1 0 1
```

La division au long en binaire

- En binaire, le processus est identique, mais simplifié parce que les seuls facteurs possibles sont 0 et 1.
- À chaque étape on obtient un chiffre de plus du quotient.

```
  1 0 0 1 1 0 1 1  | 1 1 0 0
- 0 0 0 0          | 0 1 1 0
---------
  1 0 0 1 1
- 1 1 0 0
---------
    0 1 1 1 0
  - 1 1 0 0
  -----------
      0 0 1 0 1
    - 0 0 0 0
    -----------
        0 1 0 1 1
```

La division au long en binaire

- En binaire, le processus est identique, mais simplifié parce que les seuls facteurs possibles sont 0 et 1.
- À chaque étape on obtient un chiffre de plus du quotient.

```
  1 0 0 1 1 0 1 1  | 1 1 0 0
- 0 0 0 0          | 0 1 1 0 0 ,
-------
  1 0 0 1 1
- 1 1 0 0
---------
    0 1 1 1 0
  - 1 1 0 0
  -------
      0 0 1 0 1
    - 0 0 0 0
    ---------
        0 1 0 1 1
      - 0 0 0 0
      ---------
          1 0 1 1 0
```

La division au long en binaire

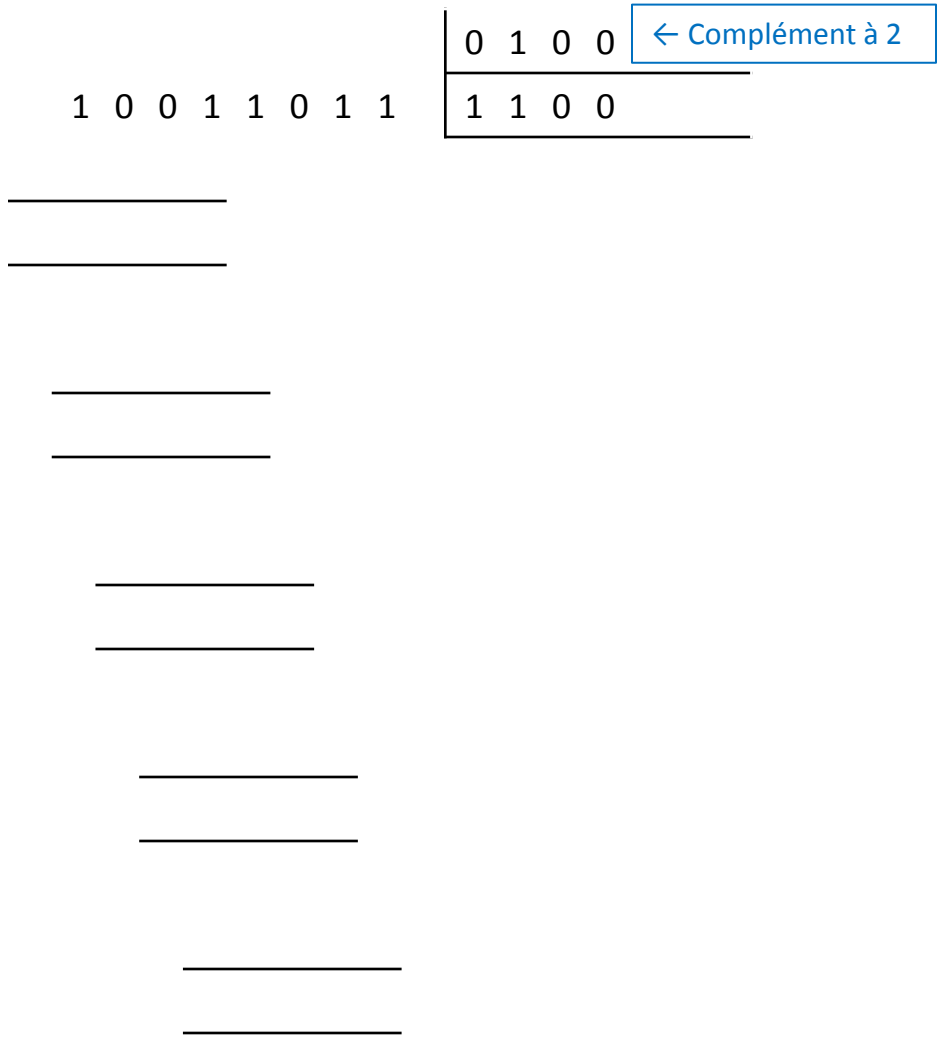
- En binaire, le processus est identique, mais simplifié parce que les seuls facteurs possibles sont 0 et 1.
- À chaque étape on obtient un chiffre de plus du quotient.

```
  1 0 0 1 1 0 1 1  | 1 1 0 0
- 0 0 0 0          -------
-------------       0 1 1 0 0 , 1
  1 0 0 1 1
- 1 1 0 0
---------
    0 1 1 1 0
  - 1 1 0 0
  ---------
      0 0 1 0 1
    - 0 0 0 0
    ---------
        0 1 0 1 1
      - 0 0 0 0
      ---------
          1 0 1 1 0
        - 1 1 0 0
        ---------
            1 0 1 0
```

Etc.

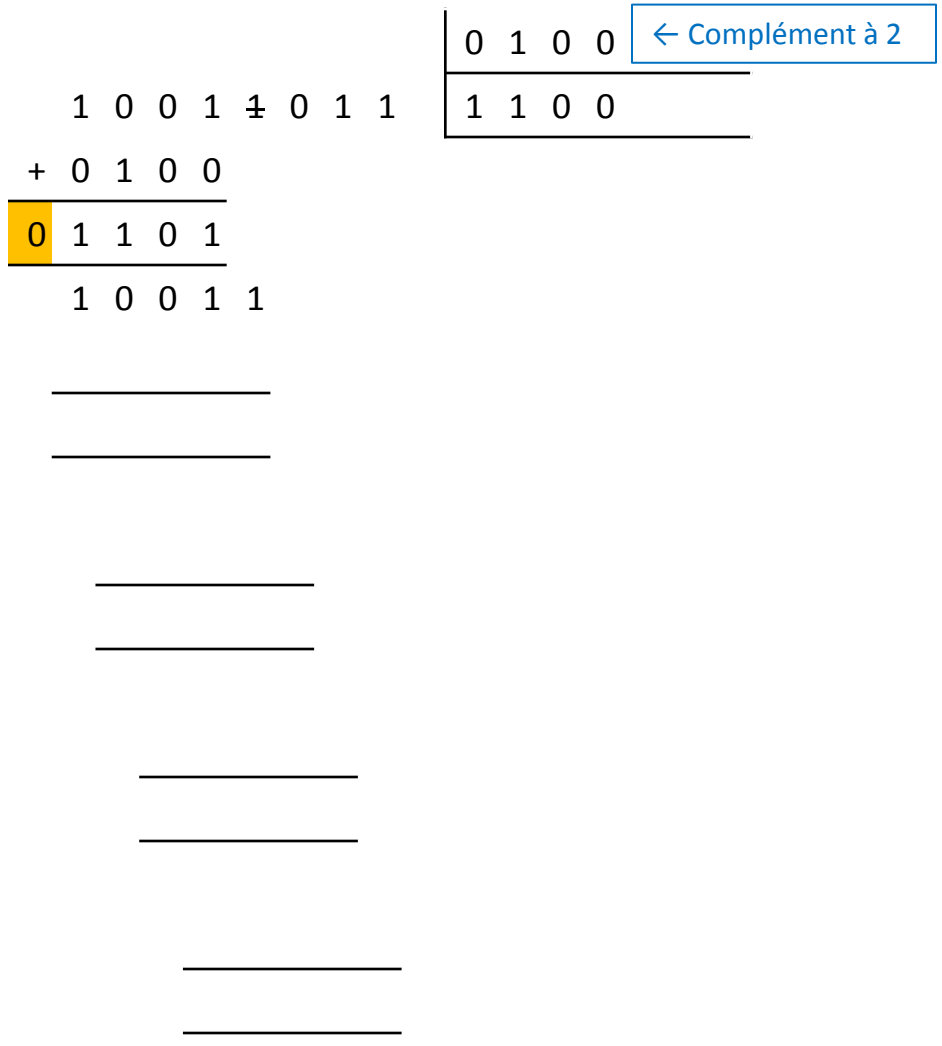
La division au long en binaire: méthode automatisée

- On peut automatiser le processus de division au long en binaire en ajoutant systématiquement le complément à deux du diviseur aux parties du dividende.
- Le reste de l'addition donne un chiffre du quotient à chaque étape, et indique si on doit garder la différence en cours ou non.



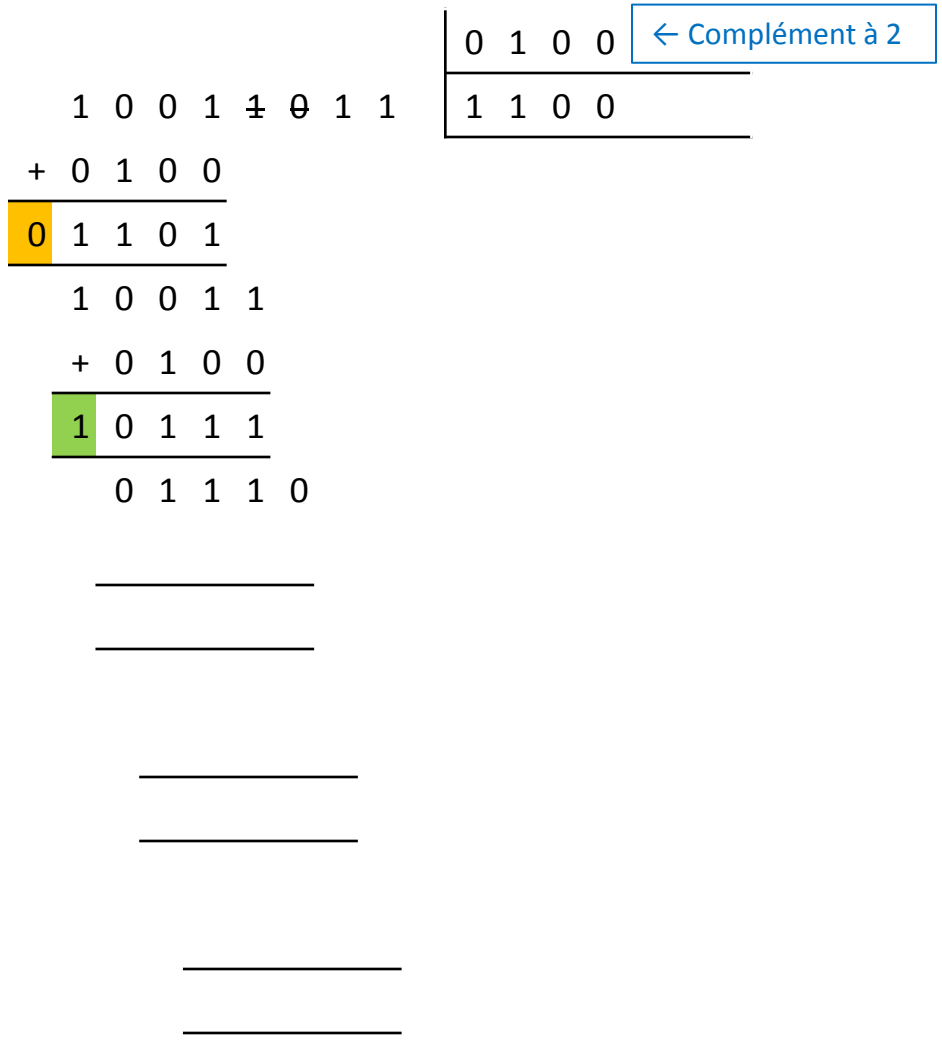
La division au long en binaire: méthode automatisée

- On peut automatiser le processus de division au long en binaire en ajoutant systématiquement le complément à deux du diviseur aux parties du dividende.
- Le reste de l'addition donne un chiffre du quotient à chaque étape, et indique si on doit garder la différence en cours ou non.



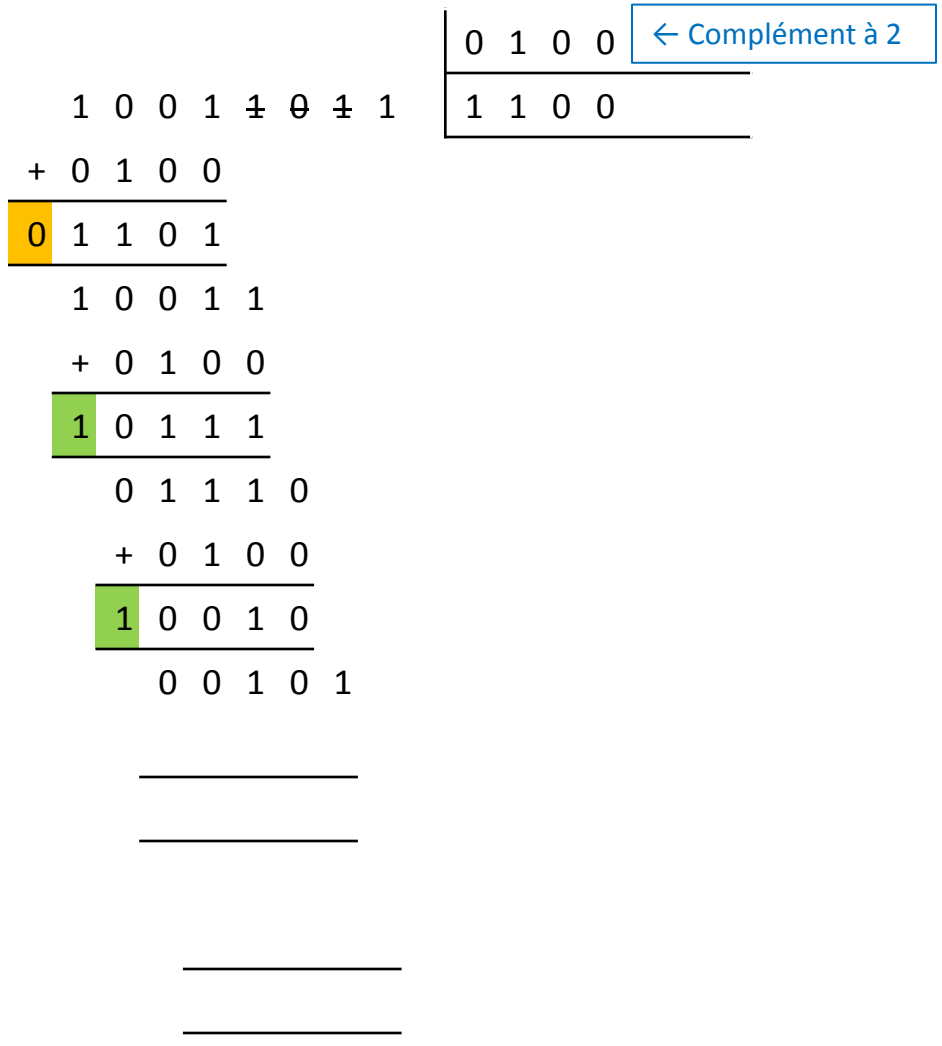
La division au long en binaire: méthode automatisée

- On peut automatiser le processus de division au long en binaire en ajoutant systématiquement le complément à deux du diviseur aux parties du dividende.
- Le reste de l'addition donne un chiffre du quotient à chaque étape, et indique si on doit garder la différence en cours ou non.



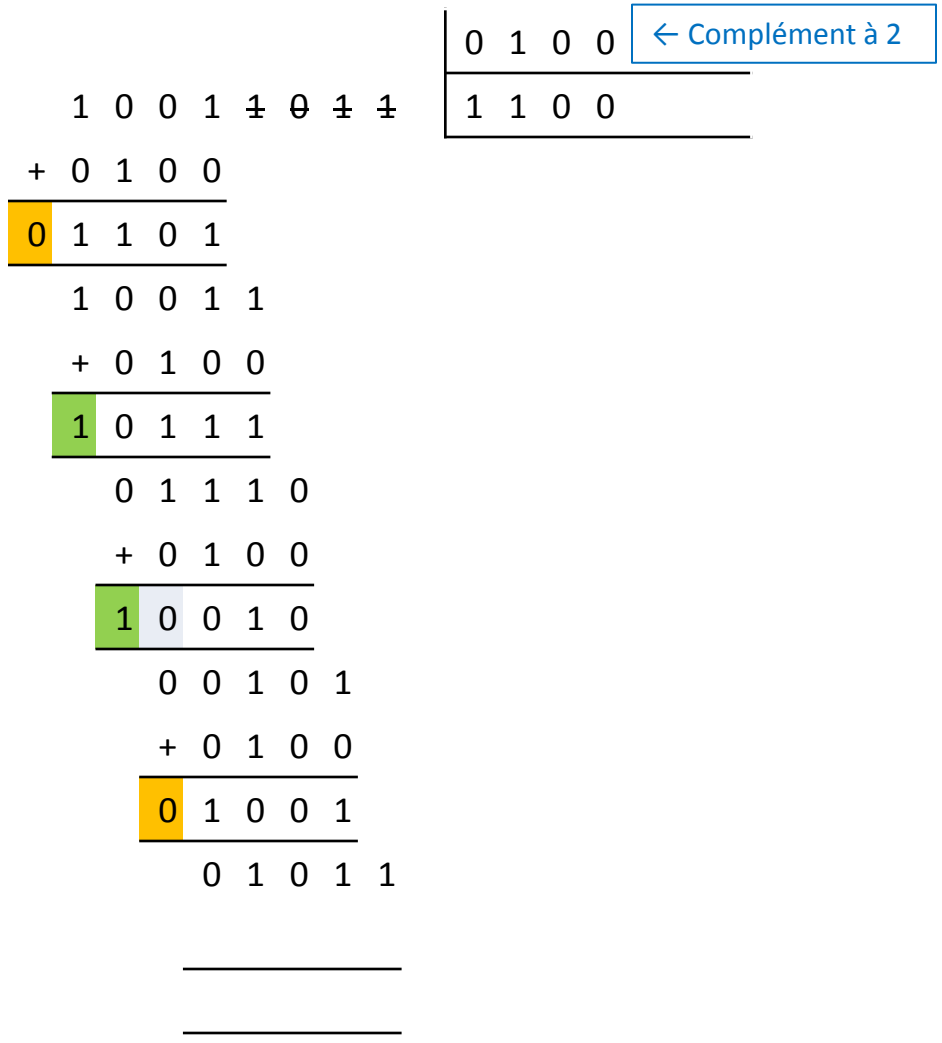
La division au long en binaire: méthode automatisée

- On peut automatiser le processus de division au long en binaire en ajoutant systématiquement le complément à deux du diviseur aux parties du dividende.
- Le reste de l'addition donne un chiffre du quotient à chaque étape, et indique si on doit garder la différence en cours ou non.



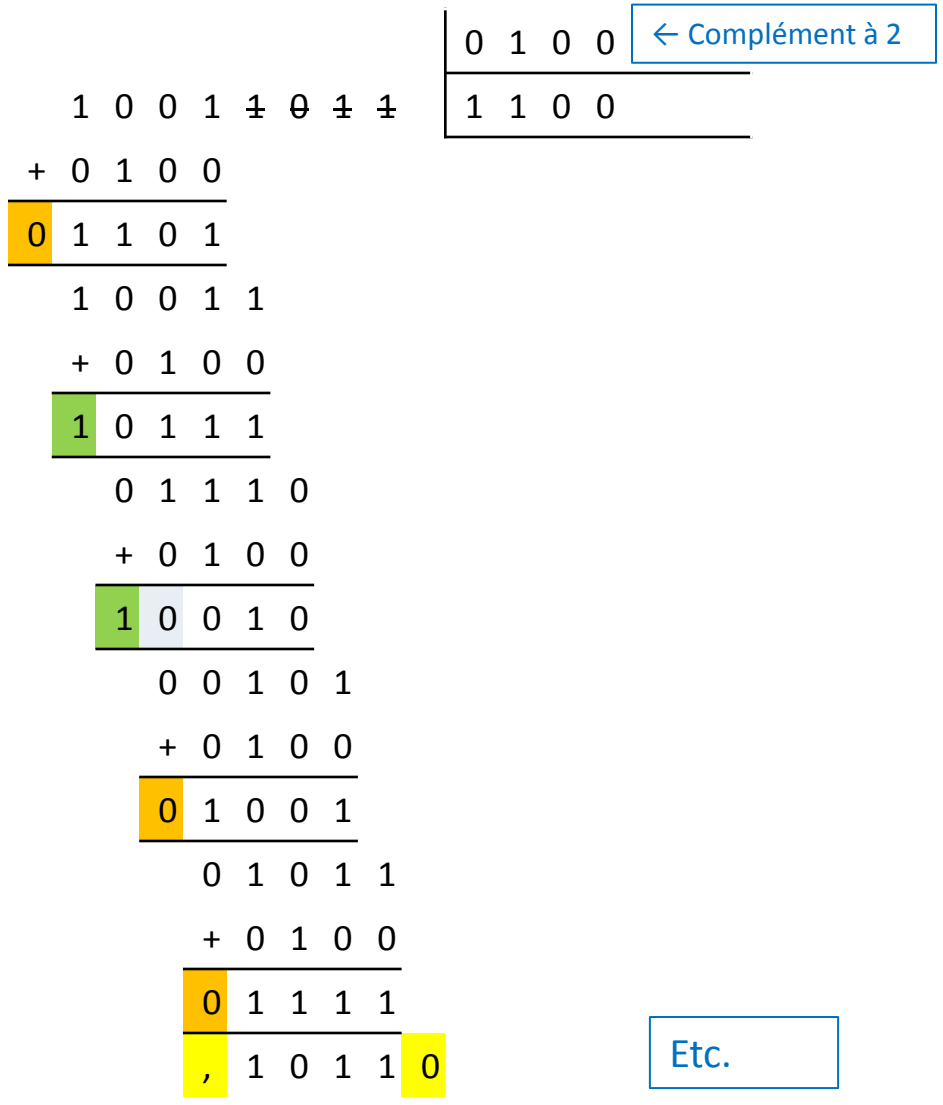
La division au long en binaire: méthode automatisée

- On peut automatiser le processus de division au long en binaire en ajoutant systématiquement le complément à deux du diviseur aux parties du dividende.
- Le reste de l'addition donne un chiffre du quotient à chaque étape, et indique si on doit garder la différence en cours ou non.



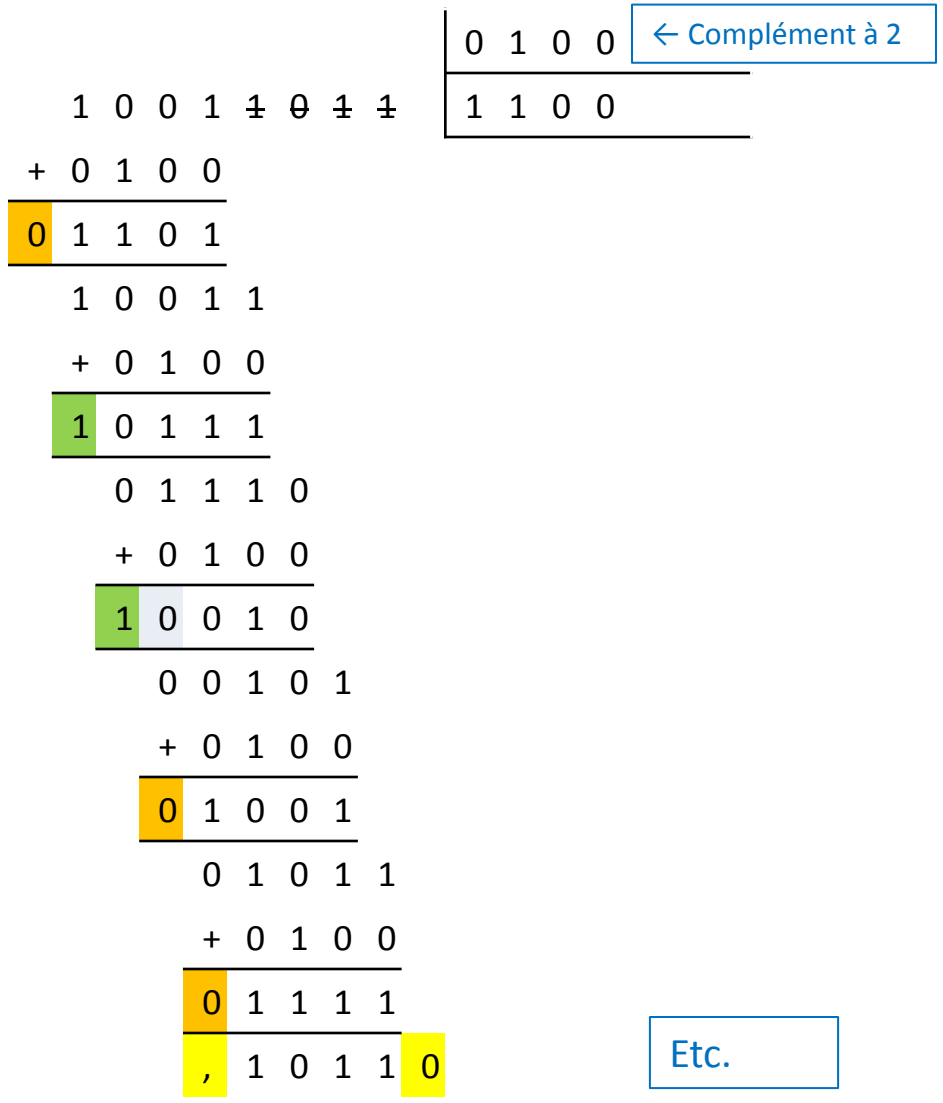
La division au long en binaire: méthode automatisée

- On peut automatiser le processus de division au long en binaire en ajoutant systématiquement le complément à deux du diviseur aux parties du dividende.
- Le reste de l'addition donne un chiffre du quotient à chaque étape, et indique si on doit garder la différence en cours ou non.



La division au long en binaire: méthode automatisée

- À chaque étape, il faut donc:
  - Conserver la différence courante dans un registre à décalage
  - Y ajouter le complément à deux du diviseur
  - Inspecter le bit le plus significatif:
    - Si c'est un 0, remplacer le contenu du registre par la différence précédente
  - Décaler le contenu du registre vers la gauche et insérer un nouveau bit du dividende à droite
  - Conserver les bits de retenue
- En exercice: donner un chemin des données pour effectuer ces opérations.
  - Additionneur à 5 bits, registre à décalage, multiplexeur …



Division par convergence, par multiplications successives (méthode de Goldschmidt)

- La méthode de division par multiplications successives consiste à multiplier le dividende N et le diviseur D par une suite de nombres X_k.

$$Q = \frac{N}{D} \times \frac{X_1}{X_1} \times \frac{X_2}{X_2} \times \frac{X_3}{X_3} \times \ldots \times \frac{X_k}{X_k} = \frac{Q}{1}$$

- Les nombres X_k sont choisis afin que le produit $D \times X_1 \times X_2 \times X_3 \ldots$ converge vers 1.
- Le produit $N \times X_1 \times X_2 \times X_3 \ldots$ converge alors vers Q.

- À chaque étape on calcule:

$$\frac{N_{k+1}}{D_{k+1}} = \frac{N_k}{D_k} \times \frac{X_{k+1}}{X_{k+1}}$$

- Avec D_0 dans l'intervalle]0.5, 1] (suite à une normalisation préalable de N et D), on choisit

$$X_{k+1} = 2 - D_k$$

- La division au long nécessite *m* étapes pour obtenir *m* bits du quotient. La méthode de Goldschmidt nécessite plutôt $\log_2 m$ étapes.

Division par convergence, par multiplications successives (méthode de Goldschmidt) - exemple

- Calculer 227 / 5
- Tout d'abord, il faut normaliser le diviseur D pour qu'il soit dans l'intervalle]0.5, 1].
- On choisit une puissance de deux: 8. On a alors:

  227 / 5 = 28,375 / 0,625
- À chaque étape on calcule:

$$\frac{N_{k+1}}{D_{k+1}} = \frac{N_k}{D_k} \times \frac{X_{k+1}}{X_{k+1}}$$

- Et on choisit:

$$X_{k+1} = 2 - D_k$$

k	N_k	D_k	X_{k+1}
0	28,375	0,625	1,375
1	39,015625	0,859375	1,140625
2	44,5021973	0,98022461	1,01977539
3	45,3822456	0,99960893	1,00039107
4	45,3999931	0,99999985	1,00000015
5	45,4	1	

Vous devriez maintenant être capable de …

- Expliquer et appliquer différentes façons d'implémenter la division dans un système numérique. (B2, B3)
- Choisir l'approche la plus appropriée selon les conditions du problème. (B3)
- Évaluer la complexité de différentes approches de division en termes des ressources de calcul nécessaires et du nombre d'opérations à effectuer pour obtenir un quotient d'une précision désirée. (B3)
- Modéliser en VHDL différentes approches pour la division. (B3)

Code	Niveau (http://fr.wikipedia.org/wiki/Taxonomie_de_Bloom)
B1	Connaissance – mémoriser de l'information.
B2	Compréhension – interpréter l'information.
B3	Application – confronter les connaissances à des cas pratiques simples.
B4	Analyse – décomposer un problème, cas pratiques plus complexes.
B5	Synthèse – expression personnelle, cas pratiques plus complexes.