Opérateurs sur les bits

Les principaux opérateurs sur les bits sont résumés dans le tableau suivant.

Condition	Signification	Condition	Signification
a & b	a 'et' b	a >> n	a décalé à droite de n bits
a | b	a 'ou' b	a << n	a décalé à gauche de n bits
a ^ b	a 'ou exclusif' b		

Ces opérateurs sont en général utilisés avec des variables de type entier ou des octets stockés dans des variables char. Ils agissent sur la forme binaire du nombre et sont souvent utilisés pour gérer des bits ou les tester séparément. Par exemple, s'il faut lire l'état de plusieurs drapeaux, ces derniers peuvent être stockés dans un seul entier.

En reprenant le tutoriel 2, il est possible d'afficher les valeurs décimales de chaque bit avec le programme ci-dessous :

```
#include <stdio.h>
int main() {
  int bit = 0, i = 1;
  while(i>0) { /* Boucle jusqu'à ce que le bit de signe soit fixé */
    printf(" pow(2,%2d) = %11d\n",bit,i);
    i = i<<1; /* Décale la valeur de i vers la gauche d'une position. */
    bit++; /* Incrémente le compteur. */
  }
  return 0; /* Retourne succès au système d'exploitation. */
}
```

Dans cet exemple, la valeur enregistrée dans la variable `i` est décalée d'une position vers la gauche. L'opérateur de décalage vers la gauche a pour effet de déplacer tous les bits de la variable `i` d'une position vers la gauche. Si un bit est déplacé en dehors de l'emplacement de la variable en mémoire, le bit est perdu. Dans ce cas, `i` contient seulement un. De plus, l'action de l'opérateur décalage à gauche est d'effectuer un passage à la puissance de deux suivante. Quand le bit de la variable `i` atteint le bit de signe, le nombre devient négatif ce qui provoque l'arrêt de la boucle while.

L'opérateur & est très utile pour tester si un bit est à un ou non. Combiné avec les opérateurs de décalages à gauche ou à droite, cela permet de tester chaque bit d'une variable entière :

```
#include <stdio.h>
int main() {
  char str[33]; /* Déclare un tableau de caractères pour conserver la sortie. */
  int bit, i = 235643; /* Déclare un nombre à convertir en binaire. */
  for(bit=31;bit>0;bit--) { /* Boucle de la gauche vers la droite */
    if(((1<<bit) & i) == 0) str[31-bit] = '0'; /* Faux */
    else str[31-bit] = '1'; /* Vrai */
  }
  str[32]='\0'; /* Ajout du terminateur de chaîne */
  printf("%d (décimal) = %s (binaire)\n", i, str);
  return 0; /* Retourne succès au système d'exploitation. */
}
```

Dans ce programme d'exemple, chaque caractère du tableau est défini selon la valeur binaire. Ensuite, pour compléter la chaîne de caractères, le terminateur de chaîne est ajouté. Enfin, la forme binaire du nombre entier est affichée.

Commandes système

Il peut être utile de pouvoir lancer des commandes shell ou d'autres programmes sans faire directement d'édition de lien avec la bibliothèque correspondante. Cela est réalisé grâce à la fonction `system` :

```
#include <stdlib.h>
int main() {
  system("ls ./"); /* Liste les fichiers du répertoire actuel. */
  return 0; /* Retourne succès au système d'exploitation. */
}
```