TP4 : Opérations bit-à-bit

Exercice 1 : Opérations logiques de base

1.1 Indiquer ce qui est affiché par le programme suivant (justifier la réponse) :

```
int a=2;

a = a << 1;
printf("%d\n", a);
a <<= 1;
printf("%d\n", a);
a >>= 2;
printf("%d\n", a);
```

1.2 Ecrire une fonction, basée sur l'utilisation de l'opérateur logique de décalage et le ET, permettant d'afficher sur une seule ligne la valeur binaire d'une variable. L'idée est d'afficher les bits du nombre un à un, du MSB au LSB. La récupération du bit à afficher peut être obtenue par un décalage puis un masquage. *Rm* : la solution la plus simple ne prend que 2 lignes, pour l'affichage proprement dit. La fonction devra avoir le prototype suivant :

```
void printb(int a, int nb_bits);
```

1.3 On souhaite simuler un **chenillard** à 8 lampes (séquence 00000001, 00000010, 00000100, 00001000, 00010000, 00100000, 01000000, 10000000, 00000001, …), dont l'état est donné par une variable de 8 bits.

a) Compléter le programme suivant pour obtenir ce fonctionnement, en utilisant des opérations bit-à-bit. Pour l'affichage, on utilisera la fonction définie précédemment. *Rm* : pour que les affichages se produisent toujours sur la même ligne, on peut utiliser le caractère \r en fin de ligne.

```
char a=1;
while(1)
{
      ...
      getchar();        //attente d'appui sur une touche
}
```

b) Ajouter un affichage de la valeur entière (printf avec format "%d") de la variable utilisée, à chaque itération. Commenter les résultats observés.

1.4 Ecrire le programme ci-dessous et expliquer le résultat affiché.

```
unsigned short int a=0xFF00, b;
b = ~a;
printf("b=%d\n", b);
```

Réécrire la 2e ligne d'une manière différente : en utilisant l'opérateur OU-Exclusif (et une valeur numérique hexadécimale à 16 bits). Conclure sur l'utilité de ce dernier opérateur.

Exercice 2 : Programmation d'un microcontrôleur

Un microcontrôleur possède des entrées/sorties de type Tout-Ou-Rien (c'est-à-dire dont l'état est binaire). Ces entrées/sorties sont regroupés par ensembles appelés "ports".

On considère un microcontrôleur doté de 2 ports d'entrée/sortie à 8 bits, programmables par l'intermédiaire de 4 variables (de type `unsigned char`) :

- P1IN : entrées du port 1
- P1OUT : sorties du port 1
- P2IN : entrées du port 2
- P2OUT : sorties du port 2

On souhaite utiliser ce microcontrôleur pour gérer des lampes, à partir d'un certain nombre d'interrupteurs et boutons-poussoirs :

- Interrupteur 1 connecté au bit (d'indice) 0 du port 1
- Interrupteur 2 connecté au bit 0 du port 2
- Bouton-poussoir connecté au bit 1 du port 2
- Lampe 1 connectée au bit 7 du port 1
- Lampe 2 connectée au bit 4 du port 2
- Lampe 3 connectée au bit 5 du port 2
- Lampe 4 connectée au bit 6 du port 1

On souhaite avoir le fonctionnement suivant :

- Lampe 1 commandée par interrupteur 1
- Lampes 2 et 3 commandées par interrupteur 2
- Lampe 4 commandée par le bouton-poussoir, avec le mode suivant : chaque appui provoque alternativement allumage ou extinction

Ecrire le programme permettant d'obtenir ce fonctionnement.

Rm :

- un programme comportant une partie du code-source est disponible sur ENT, et doit être complété
- attention : le test de chacun des 3 cas d'utilisation devra être indépendant des 2 autres

Exercice 3 (bonus) Réalisation d'un chenillard à microcontrôleur

Programmer un chenillard à 8 lampes connectées au port 2 du microcontrôleur décrit dans l'exercice précédent.

Le sens d'évolution du chenillard devra alterner de l'un à l'autre à chaque appui sur un bouton-poussoir connecté sur un bit du port 1.