T.P. 2
Space Invaders (partie 5)

Étape 1

Soit le programme principal suivant qui déplace un envahisseur de la gauche vers la droite.

```
Main                ; Fait pointer A0 sur un envahisseur.
                    lea     InvaderA_Bitmap,a0

                    ; Place l'envahisseur sur le centre gauche.
                    move.w  #0,d1
                    move.w  #152,d2

\loop               ; Efface l'écran et affiche l'envahisseur.
                    jsr     ClearScreen
                    jsr     PrintBitmap

                    ; Incrémente l'abscisse de l'envahisseur.
                    addq.w  #1,d1

                    ; Reboucle tant que l'envahisseur
                    ; n'a pas atteint le centre droit.
                    cmpi.w  #456,d1
                    blt     \loop

                    illegal
```

Dans un premier temps, réalisez le sous-programme **`ClearScreen`** qui efface l'intégralité de l'écran (il pourra effectuer un appel à votre sous-programme **`FillScreen`**).

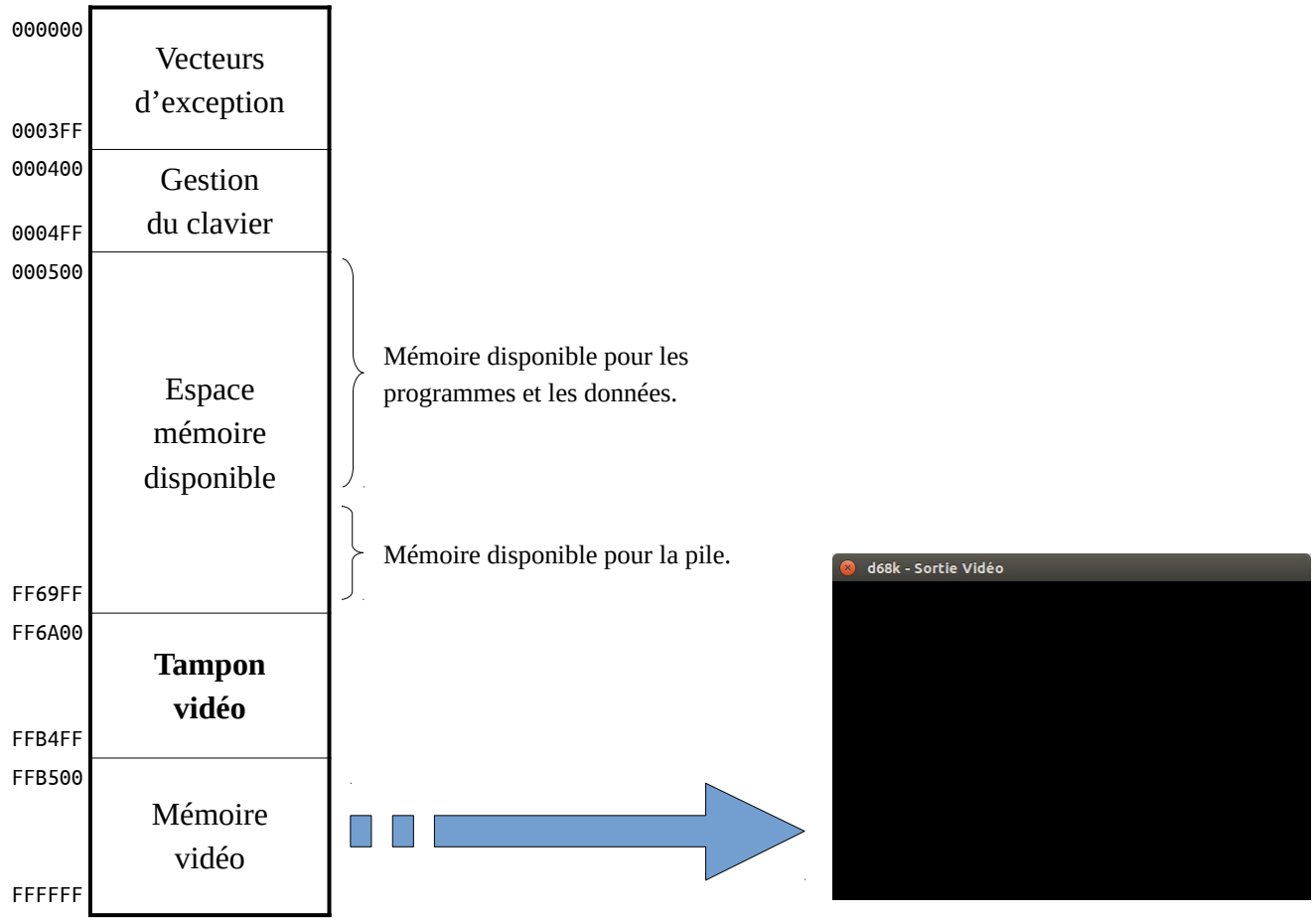
Lancez maintenant le programme principal ci-dessus et observez le déplacement de l'envahisseur. Vous pouvez constater qu'un scintillement apparaît. Ce scintillement est désagréable pour l'œil, il nous faudra le supprimer.

Étape 2

Afin de supprimer le scintillement, nous allons utiliser la technique du *double buffering*. Cette technique consiste à afficher les bitmaps non pas directement à l'écran, mais d'abord dans un tampon vidéo (*video buffer*). Ce dernier sera ensuite copié intégralement dans la mémoire vidéo.

Un tampon vidéo n'est rien d'autre qu'un espace mémoire de taille identique à la mémoire vidéo. Pour cela, nous allons utiliser la mémoire située juste avant la mémoire vidéo.

Nous obtenons donc le nouveau découpage ci-dessous :



Afin d'utiliser ce tampon vidéo, vous devez dans un premier temps remplacer toutes les occurrences de `VIDEO_START` par `VIDEO_BUFFER` dans votre fichier source.

Ensuite, les étiquettes relatives à la mémoire vidéo devront être les suivantes :

```
                ; Mémoire vidéo
                ; ------------------------------

VIDEO_START     equ     $ffb500                     ; Adresse de départ
VIDEO_WIDTH     equ     480                         ; Largeur en pixels
VIDEO_HEIGHT    equ     320                         ; Hauteur en pixels
VIDEO_SIZE      equ     (VIDEO_WIDTH*VIDEO_HEIGHT/8)    ; Taille en octets
BYTE_PER_LINE   equ     (VIDEO_WIDTH/8)             ; Nombre d'octets par ligne
VIDEO_BUFFER    equ     (VIDEO_START-VIDEO_SIZE)    ; Tampon vidéo
```

L'adresse initiale de la pile doit également être mise à jour :

```
                    ; ==============================
                    ; Initialisation des vecteurs
                    ; ==============================

                    org     $0

vector_000          dc.l    VIDEO_BUFFER                ; Valeur initiale de A7
vector_001          dc.l    Main                        ; Valeur initiale du PC
```

Après ces différents changements, l'affichage ne se fait plus directement dans la mémoire vidéo, mais dans le tampon vidéo. Pour l'instant, votre programme principal fonctionne comme avant sauf que plus rien ne s'affiche dans la fenêtre de sortie vidéo.

Il faut donc maintenant copier le contenu du tampon vidéo dans la mémoire vidéo afin d'observer le déplacement de l'envahisseur. Modifiez votre programme principal de la façon suivante :

```
Main                ; Fait pointer A0 sur un envahisseur.
                    lea     InvaderA_Bitmap,a0

                    ; Place l'envahisseur sur le centre gauche.
                    move.w  #0,d1
                    move.w  #152,d2

\loop               ; Affiche l'envahisseur dans le tampon vidéo.
                    jsr     PrintBitmap

                    ; Copie le tampon vidéo dans la mémoire vidéo.
                    ; (Le contenu du tampon est effacé par la même occasion.)
                    jsr     BufferToScreen

                    ; Incrémente l'abscisse de l'envahisseur.
                    addq.w  #1,d1

                    ; Reboucle tant que l'envahisseur
                    ; n'a pas atteint le centre droit.
                    cmpi.w  #456,d1
                    blt     \loop

                    illegal
```

Réalisez ensuite le sous-programme **`BufferToScreen`** qui copie le tampon vidéo dans la mémoire vidéo (la copie se fera par mot long). Une fois qu'un mot long aura été copié du tampon vers la mémoire vidéo, il sera mis à zéro dans la mémoire tampon (cela permettra d'effacer la mémoire tampon avant le prochain affichage).

Testez maintenant votre sous-programme à l'aide du programme principal ci-dessus. Vérifiez que l'envahisseur se déplace bien de la gauche vers la droite et que le scintillement a complètement disparu.

Étape 3

Nous allons maintenant définir une structure de sprite. Un sprite est un petit élément graphique animé qui se déplace à l'écran. Dans un premier temps, nous gérerons les sprites uniquement de façon à mémoriser l'emplacement de tous les bitmaps affichés à l'écran. Puis, dans un second temps, nous y ajouterons une petite animation.

Commençons par définir la structure d'un sprite. Ce dernier possédera les cinq champs suivants :

Champ	Taille (bits)	Encodage	Description
STATE	16	Entier non signé	État d'affichage du sprite. Seulement deux valeurs possibles : HIDE = 0 ou SHOW = 1
X	16	Entier signé	Abscisse du sprite
Y	16	Entier signé	Ordonnée du sprite
BITMAP1	32	Entier non signé	Adresse du premier bitmap
BITMAP2	32	Entier non signé	Adresse du second bitmap

Les deux bitmaps serviront à l'animation du sprite, mais pour l'instant nous n'utiliserons pas le second bitmap. Nous nous occuperons de l'animation un peu plus tard. Nous utiliserons donc uniquement le premier bitmap pour l'affichage du sprite.

À partir de cette structure, nous allons définir de nouvelles constantes afin de faciliter la gestion des sprites. Ajoutez les lignes suivantes dans la partie *Définition des constantes* de votre code source.

```
                    ; Sprites
                    ; -----------------------------

STATE               equ     0                       ; État de l'affichage
X                   equ     2                       ; Abscisse
Y                   equ     4                       ; Ordonnée
BITMAP1             equ     6                       ; Bitmap no 1
BITMAP2             equ     10                      ; Bitmap no 2

HIDE                equ     0                       ; Ne pas afficher le sprite
SHOW                equ     1                       ; Afficher le sprite
```

Ajoutez également votre premier sprite dans la partie *Données* de votre code source. Il s'agira d'un envahisseur placé dans le centre gauche de la fenêtre de sortie vidéo.

```
Invader             dc.w    SHOW                    ; Afficher le sprite
                    dc.w    0,152                   ; X = 0, Y = 152
                    dc.l    InvaderA_Bitmap         ; Bitmap à afficher
                    dc.l    0                       ; Inutilisé
```

Supposons maintenant que le registre **A1** contienne la valeur de l'adresse `Invader`, on pourra facilement accéder aux différents champs du sprite.

Par exemple :

```
                    move.w  STATE(a1),d0    ; État de l'affichage -> D0.W
                    move.w  X(a1),d1        ; X                   -> D1.W
                    move.w  Y(a1),d2        ; Y                   -> D2.W
                    movea.l BITMAP1(a1),a0  ; Adresse du bitmap 1 -> A0.L
```

Réalisez le sous-programme **`PrintSprite`** qui affiche un sprite dans le tampon vidéo. Attention si l'état d'affichage d'un sprite est HIDE, celui-ci ne doit pas être affiché. Ce sous-programme possède une seule entrée :
Entrée : **A1.L** = Adresse du sprite.

Vous testerez votre sous-programme à l'aide du programme principal suivant :

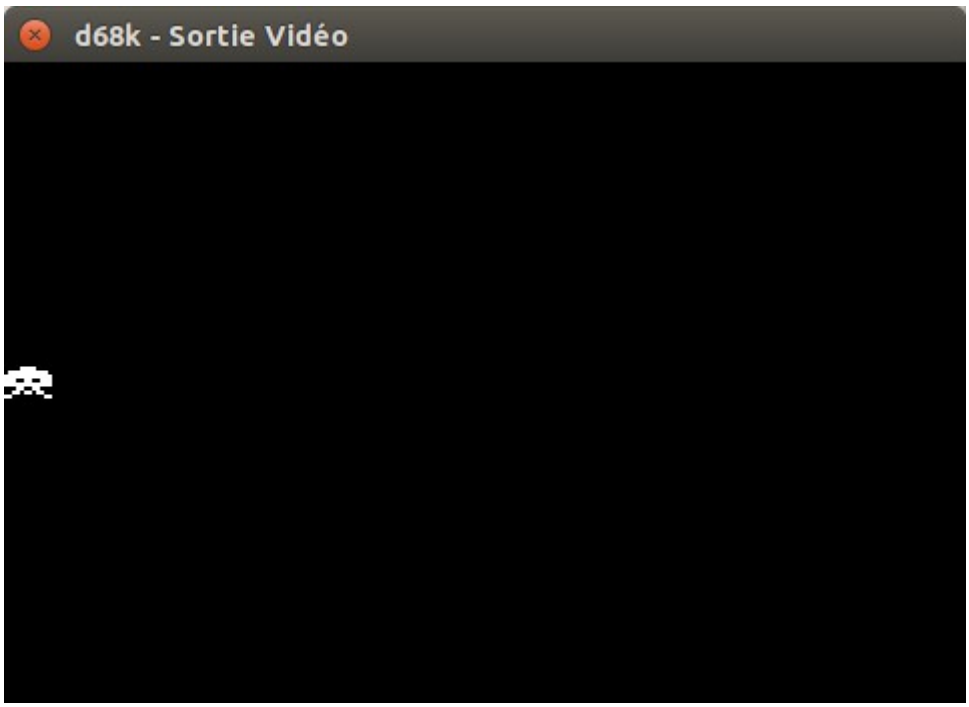
```
Main                ; Fait pointer A1 sur un sprite.
                    lea     Invader,a1

                    ; Affiche le sprite dans le tampon vidéo.
                    jsr     PrintSprite

                    ; Copie le tampon vidéo dans la mémoire vidéo.
                    jsr     BufferToScreen

                    illegal
```

Capture d'écran du résultat attendu :



Positionnez maintenant l'état d'affichage du sprite à HIDE et relancez votre programme principal. Vérifiez bien que le sprite ne s'affiche pas.