PAlib

Généré par Doxygen 1.6.1

Wed Jul 7 21 :08 :38 2010

Table des matières

Chapitre 1

PAlib 100707 Documentation

1.1 Introduction

Welcome to the PAlib documentation. Here you'll find information on how to use PAlib.

1.2 Core library

– **General functions** (p. 58)
– **Debugging utilities** (p. 42)
– **Math functions** (p. 75)
– **C++ wrappers** (p. 155)

1.3 Tiled backgrounds

– **Normal background functions** (p. 25)
– **Rotating background functions** (p. 22)
– **Dual background functions** (p. 140)
– **Text system** (p. 132)
– **Mode 7 functions** (p. 80)

1.4 Bitmapped backgrounds

– **Bitmapped background functions** (p. 44)
– **16-color bitmapped background functions** (p. 5)
– **Fake 16-bit background functions** (p. 54)
– **GIF functions** (p. 64)

1.5 Sprites

– **Sprite functions** (p. 97)
– **Dual sprite functions** (p. 119)
– **3D Sprite functions** (p. 11)

1.6 Palettes

– **Palette functions** (p. 84)
– **Dual palette functions** (p. 90)

1.7 Input

– **Pad and stylus functions** (p. 70)
– **Keyboard functions** (p. 67)
– **Handwriting recognition functions** (p. 93)
– **Microphone functions** (p. 79)
– **Special controller functions** (p. 74)

1.8 Sound

– **ASlib library** (p. 156)

1.9 Misc.

– **Special effects** (p. 94)

Chapitre 2

Liste des éléments obsolètes

Global(e) PA_16cCustomFont (p. **6**)

Global(e) PA_8bitCustomFont (p. **134**)

Global(e) PA_DualEasyBgLoad (p. **146**)

Global(e) PA_DualLoadBg (p. **144**)

Global(e) PA_DualLoadLargeBg (p. **145**)

Global(e) PA_DualLoadLargeBgEx (p. **145**)

Global(e) PA_DualLoadPAGfxLargeBg (p. **144**)

Global(e) PA_DualLoadRotBg (p. **143**)

Global(e) PA_DualLoadSimpleBg (p. **142**)

Global(e) PA_DualLoadTiledBg (p. **142**)

Global(e) PA_EasyBgLoad (p. **31**)

Global(e) PA_EasyBgLoadPtr (p. **31**)

Global(e) PA_InitCustomKeyboard (p. **68**)

Global(e) PA_InitCustomText (p. **134**)

Global(e) PA_LegacyIPCInit (p. **60**)

Global(e) PA_LoadBg (p. **30**)

Global(e) PA_LoadBgTiles (p. **28**)

Global(e) PA_LoadLargeBg (p. **18**)

Global(e) PA_LoadLargeBgEx (p. **19**)

Global(e) PA_LoadPAGfxLargeBg (p. **18**)

Global(e) PA_LoadPAGfxRotBg (p. **23**)

Global(e) PA_LoadRotBg (p. **22**)

Global(e) PA_LoadSimpleBg (p. **29**)

Global(e) PA_LoadTiledBg (p. **29**)

Chapitre 3

Documentation des modules

3.1 16color pseudo-bitmap mode

Macros

– #define **PA_16cCustomFont**(c16_slot, c16_font)
[DEPRECATED] Ajouter une police perso dans le systeme de texte 16c ! ! Doit être convertie avec PAGfx

Fonctions

– static void **PA_Init16cBg** (u8 screen, u8 bg)
Initialise le mode de dessin 16 couleurs, sur lequel on peut coller de petites images...

– void **PA_16cErase** (u8 screen)
Effacer un écran de 16 couleurs. Doit être utilisé juste après PA_WaitForVBL pour éviter des erreurs d'affichage.

– static void **PA_Dual16cErase** (void)
Effacer un écran de 16 couleurs sur les 2 écrans. Doit être utilisé juste après PA_WaitForVBL pour éviter des erreurs d'affichage.

– static void **PA_InitComplete16c** (u8 bg, void ∗Palette)
Initialise le mode de dessin 16 couleurs sur les 2 écrans avec une palette donnée.

– s16 **PA_16cText** (u8 screen, s16 basex, s16 basey, s16 maxx, s16 maxy, const char ∗text, u8 color, u8 size, s32 limit)
Cette fonction permet d'écrire du texte à chasse variable à l'écran.

– void **PA_Add16cFont** (int slot, const **PA_BgStruct** ∗font)
Ajouter une police perso dans le système de texte 16c.

– ALWAYSINLINE void **PA_16cPutPixel** (u8 screen, s16 x, s16 y, u32 color)
Afficher un pixel sur un fond 16c.

– ALWAYSINLINE void **PA_16c8X4** (u8 screen, s16 x, s16 y, u32 ∗image)

Afficher une image de 8x4 pixels à un endroit donné, fonction la plus rapide de copie...

– ALWAYSINLINE void **PA_16c8X6** (u8 screen, s16 x, s16 y, u32 ∗image)
Afficher une image de 8x6 pixels à un endroit donné, deuxième fonction la plus rapide de copie...

– ALWAYSINLINE void **PA_16c8X8** (u8 screen, s16 x, s16 y, u32 ∗image)
Afficher une image de 8x8 pixels à un endroit donné.

– ALWAYSINLINE void **PA_16c8Xi** (u8 screen, s16 x, s16 y, u32 ∗image, u8 i)
Afficher une image de 8x8 pixels à un endroit donné.

– void **PA_16cClearZone** (u8 screen, s16 x1, s16 y1, s16 x2, s16 y2)
Effacer une partie d'un fond 16c.

– static u8 **PA_16cGetPixel** (u8 screen, s16 x, s16 y)
Renvoie la valeur d'un pixel donné sur un fond 16c.

3.1.1 Description détaillée

Special 16color background on which you can paste images. Usefull to show shots in SHMUP !

3.1.2 Documentation des macros

3.1.2.1 #define PA_16cCustomFont(c16_slot, c16_font)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    bittext_maps[c16_slot] = (u16*)(void*)c16_font##_Map;\
    c16_tiles[c16_slot] = (u32*)(void*)c16_font##_Tiles;\
    pa_bittextdefaultsize[c16_slot] = (u8*)c16_font##_Sizes;\
    pa_bittextpoliceheight[c16_slot] = c16_font##_Height;\
}while(0)
```

[DEPRECATED] Ajouter une police perso dans le systeme de texte 16c ! ! Doit être convertie avec PAGfx

Obsolète

Paramètres:

c16_slot Slot pour ajouter la police. Les slots 0-4 sont utilisés pour les polices par défaut de PAlib, et 5-9 sont libres. On peut néanmoins charger par-dessus les polices PAlib si on veut.

c16_font Nom de la police...

3.1.3 Documentation des fonctions

3.1.3.1 static inline void PA_Init16cBg (u8 *screen*, u8 *bg*) [inline, static]

Initialise le mode de dessin 16 couleurs, sur lequel on peut coller de petites images... Initialise le mode de dessin 16 couleurs, sur lequel on peut coller de petites images... Utilisant la palette 0.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg

3.1.3.2 static inline void PA_16cErase (u8 *screen*)

Effacer un écran de 16 couleurs. Doit être utilisé juste après PA_WaitForVBL pour éviter des erreurs d'affichage.

Paramètres:

screen Choix de l'écran (0 ou 1)

3.1.3.3 static inline void PA_InitComplete16c (u8 *bg*, void * *Palette*) [inline, static]

Initialise le mode de dessin 16 couleurs sur les 2 écrans avec une palette donnée.

Paramètres:

bg Numéro du fond

Palette Palette de 16 couleurs

3.1.3.4 s16 PA_16cText (u8 *screen*, s16 *basex*, s16 *basey*, s16 *maxx*, s16 *maxy*, const char * *text*, u8 *color*, u8 *size*, s32 *limit*)

Cette fonction permet d'écrire du texte à chasse variable à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

basex Coordonnée X du coin supérieur gauche

basey Coordonnée Y du coin supérieur gauche

maxx Coordonnée X du coin inférieur droit

maxy Coordonnée Y du coin inférieur droit

text Texte, tel que "Hello World"

color Couleur de la palette à utiliser (0-255)

size Taille du texte, de 0 (vraiment petit) à 4 (assez grand)

limit On peut fixer une limite au nombre de caractères. Ceci peut etre utile pour dessiner un texte progressivement, en augmentant de 1 le nombre de caractères à chaque boucle....

3.1.3.5 void PA_Add16cFont (int *slot*, const PA_BgStruct * *font*)

Ajouter une police perso dans le système de texte 16c.
Paramètres:

slot Slot pour ajouter la police. Les slots 0-4 sont utilisés pour les polices par défaut de PAlib, et 5-9 sont libres. On peut néanmoins charger par-dessus les polices PAlib si on veut.

font Pointeur vers le police perso.

3.1.3.6 ALWAYSINLINE PA_16cPutPixel (u8 *screen*, s16 *x*, s16 *y*, u32 *color*)

Afficher un pixel sur un fond 16c.
Paramètres:

screen Ecran...

x Position X en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 263, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

y Position y en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 199, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

color Valeur du pixel (0-15, prend la couleur dans la palette chargée)

3.1.3.7 ALWAYSINLINE void PA_16c8X4 (u8 *screen*, s16 *x*, s16 *y*, u32 * *image*)

Afficher une image de 8x4 pixels à un endroit donné, fonction la plus rapide de copie...
Paramètres:

screen Ecran...

x Position X en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 255, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

y Position y en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 191, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

image Image en 16 couleurs à charger. Utiliser (u32*)NomImage en cas d'erreur de compilation

3.1.3.8 ALWAYSINLINE void PA_16c8X6 (u8 *screen*, s16 *x*, s16 *y*, u32 * *image*)

Afficher une image de 8x6 pixels à un endroit donné, deuxième fonction la plus rapide de copie...
Paramètres:

screen Ecran...

x Position X en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 255, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

y Position y en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 191, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

image Image en 16 couleurs à charger. Utiliser (u32∗)NomImage en cas d'erreur de compilation

3.1.3.9 ALWAYSINLINE void PA_16c8X8 (u8 *screen*, s16 *x*, s16 *y*, u32 ∗ *image*)

Afficher une image de 8x8 pixels à un endroit donné.

Paramètres:

screen Ecran...

x Position X en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 255, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

y Position y en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 191, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

image Image en 16 couleurs à charger. Utiliser (u32∗)NomImage en cas d'erreur de compilation

3.1.3.10 ALWAYSINLINE void PA_16c8Xi (u8 *screen*, s16 *x*, s16 *y*, u32 ∗ *image*, u8 *i*)

Afficher une image de 8x8 pixels à un endroit donné.

Paramètres:

screen Ecran...

x Position X en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 255, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

y Position y en pixels du coin supérieur gauche. A noter que celle-ci va de -8 à 191, afin de permettre des images à moitié sorties... NE JAMAIS DEPASSER DU CADRE, sous peine de gros bugs graphiques...

image Image en 16 couleurs à charger. Utiliser (u32∗)NomImage en cas d'erreur de compilation

i Nombre de lignes à dessiner

3.1.3.11 void PA_16cClearZone (u8 *screen*, s16 *x1*, s16 *y1*, s16 *x2*, s16 *y2*)

Effacer une partie d'un fond 16c.

Paramètres:

screen Ecran...

x1 Coin supérieur gauche...

y1 Coin supérieur gauche...

x2 Coin inférieur droit...

y2 Coin inférieur droit...

3.1.3.12 static inline u8 PA_16cGetPixel (u8 *screen*, s16 *x*, s16 *y*) [inline, static]

Renvoie la valeur d'un pixel donné sur un fond 16c.

Paramètres:

screen Ecran...

x Valeur X...

y Valeur Y...

3.2 3D Sprite System

Fonctions

– void **PA_Init3D** ()
Initializes 3D.

– void **PA_Init3D2Banks** ()
Initializes 3D taking two banks of VRAM.

– void **PA_3DProcess** ()
Renders the 3D sprites.

– s16 **PA_3DCreateTex** (void ∗obj_data, u16 width, u16 height, u8 type)
Creates a 3D texture.

– void **PA_3DCreateSpriteFromTex** (u16 sprite, u16 texture, u16 width, u16 height, u8 palette, s16 x, s16 y)
Creates a 3D sprite from a texture.

– void **PA_Reset3DSprites** ()
Resets the 3D system.

– void **PA_Reset3DSprites2Banks** ()
Resets the dual bank 3D system.

– static u16 **PA_3DCreateSprite** (u16 sprite, void ∗image, u16 width, u16 height, u8 type, u8 palette, s16 x, s16 y)
Creates a 3D sprite.

– void **PA_3DDeleteTex** (u32 tex_gfx)
Deletes a 3D texture.

– static void **PA_3DDeleteSprite** (u16 sprite)
Deletes a 3D sprite.

– static void **PA_3DSetSpriteX** (u16 sprite, s16 x)
Moves a 3D sprite in the X axis.

– static void **PA_3DSetSpriteY** (u16 sprite, s16 y)
Moves a 3D sprite in the Y axis.

– static void **PA_3DSetSpriteXY** (u16 sprite, s16 x, s16 y)
Moves a 3D sprite.

– static void **PA_3DSetSpriteRotateX** (u16 sprite, s16 rotateX)
Rotates a 3D sprite in the X axis.

– static void **PA_3DSetSpriteRotateY** (u16 sprite, s16 rotateY)
Rotates a 3D sprite in the Y axis.

– static void **PA_3DSetSpriteRotateZ** (u16 sprite, s16 rotate)

Rotates a 3D sprite in the Z axis.

– static void **PA_3DSetSpriteRotateXYZ** (u16 sprite, s16 rotateX, s16 rotateY, s16 rotateZ)
Rotates a 3D sprite.

– static void **PA_3DSetSpriteZoomX** (u16 sprite, float zoomx)
Zooms a 3D sprite horizontally.

– static void **PA_3DSetSpriteZoomY** (u16 sprite, float zoomy)
Zooms a 3D sprite vertically.

– static void **PA_3DSetSpriteZoomXY** (u16 sprite, float zoomx, float zoomy)
Zooms a 3D sprite.

– static void **PA_3DSetSpriteWidth** (u16 sprite, u16 width)
Changes the width of a 3D sprite.

– static void **PA_3DSetSpriteHeight** (u16 sprite, u16 height)
Changes the height of a 3D sprite.

– static void **PA_3DSetSpriteWidthHeight** (u16 sprite, u16 width, u16 height)
Changes the size of a 3D sprite.

– static void **PA_3DSetSpriteHflip** (u16 sprite, u8 hflip)
Sets the HFlip of a 3D sprite.

– static void **PA_3DSetSpriteVflip** (u16 sprite, u8 vflip)
Sets the VFlip of a 3D sprite.

– static u8 **PA_3DSpriteTouched** (u16 sprite)
Retrives if a 3D sprite is being touched by the stylus.

– static void **PA_3DSetSpriteTex** (u16 sprite, u16 texture)
Sets the texture of a 3D sprite.

– static void **PA_3DSetSpritePal** (u16 sprite, u16 palette)
Sets the palette of a 3D sprite.

– void **PA_3DSetSpriteFrame** (u16 sprite, u16 frame)
Sets the animation frame of a 3D sprite.

– static void **PA_3DSetSpriteTopLeft** (u16 sprite, s16 x, s16 y)
Sets the top left corner of a 3D sprite.

– static void **PA_3DSetSpriteTopRight** (u16 sprite, s16 x, s16 y)
Sets the top right corner of a 3D sprite.

– static void **PA_3DSetSpriteBottomLeft** (u16 sprite, s16 x, s16 y)
Sets the bottom left corner of a 3D sprite.

– static void **PA_3DSetSpriteBottomRight** (u16 sprite, s16 x, s16 y)

Sets the bottom right corner of a 3D sprite.

– static void **PA_3DSetSpritePrio** (u16 sprite, u16 priority)
Sets the priority of a 3D sprite.

– static void **PA_3DSetSpritePolyID** (u16 sprite, u8 polyID)
Sets the PolyID of a 3D sprite.

– static void **PA_3DSetSpriteAlpha** (u16 sprite, u8 alpha)
Sets the alpha value of a 3D sprite.

– void **PA_3DStartSpriteAnimEx** (u16 sprite, s16 firstframe, s16 lastframe, s16 speed, u8 type, s16 ncycles)
Démarre une animation de sprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

– static void **PA_3DStartSpriteAnim** (u16 sprite, s16 firstframe, s16 lastframe, s16 speed)
Démarre une animation de sprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

– static void **PA_3DStopSpriteAnim** (u16 sprite)
Arrêter une animation de sprite.

– static void **PA_3DSetSpriteAnimFrame** (u16 sprite, u16 frame)
Changer le numéro actuel de la frame d'animation.

– static u16 **PA_3DGetSpriteAnimFrame** (u16 sprite)
Renvoie le numéro actuel de la frame d'animation.

– static void **PA_3DSetSpriteAnimSpeed** (u16 sprite, s16 speed)
Changer la vitesse de l'animation.

– static u16 **PA_3DGetSpriteAnimSpeed** (u16 sprite)
Renvoie la vitesse de l'animation.

– static void **PA_3DSetSpriteNCycles** (u16 sprite, s16 NCycles)
Changer le nombre de cycles d'animation restant (-1 pour inifini).

– static u16 **PA_3DGetSpriteNCycles** (u16 sprite)
Renvoie le nombre de cycles d'animation restants.

– static void **PA_3DSpriteAnimPause** (u16 sprite, u8 pause)
Mettre en Pause en remettre en lecture une animation de sprite.

– static s32 **PA_3DGetSpriteX** (u16 sprite)
Gets the X value of a 3D sprite.

– static s32 **PA_3DGetSpriteY** (u16 sprite)
Gets the Y value of a 3D sprite.

– static void **PA_3DSetSpriteVisible** (u16 sprite, u8 visible)
Retrieves if a 3D sprite is visible.

3.2.1 Description détaillée

Sprites on one screen using the DS's 3D GPU

3.2.2 Documentation des fonctions

3.2.2.1 void PA_3DStartSpriteAnimEx (u16 *sprite*, s16 *firstframe*, s16 *lastframe*, s16 *speed*, u8 *type*, s16 *ncycles*)

Démarre une animation de sprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

firstframe Premières image de l'animation, généralement 0....

lastframe Dernière image à afficher. Une fois atteinte, ca retourne à la première

speed Vitesse, en frames par seconde (fps). 1 signifie donc 1 image par seconde...

type Défini de quelle manière on veut boucler. ANIM_LOOP (0) pour normal, et ANIM_UPDOWN (1) pour d'avant en arrière

ncycles Nombres de cycles d'animations avant l'arrêt. Si on utilise ANIM_UPDOWN, il faut 2 cycles pour que l'animation revienne à l'image de base

3.2.2.2 static inline void PA_3DStartSpriteAnim (u16 *sprite*, s16 *firstframe*, s16 *lastframe*, s16 *speed*) `[inline, static]`

Démarre une animation de sprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

firstframe Premières image de l'animation, généralement 0....

lastframe Dernière image à afficher. Une fois atteinte, ca retourne à la première

speed Vitesse, en frames par seconde (fps). 1 signifie donc 1 image par seconde...

3.2.2.3 static inline void PA_3DStopSpriteAnim (u16 *sprite*) `[inline, static]`

Arrêter une animation de sprite.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

3.2.2.4 static inline void PA_3DSetSpriteAnimFrame (u16 *sprite*, u16 *frame*) `[inline, static]`

Changer le numéro actuel de la frame d'animation.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

frame Numéro de frame...

3.2.2.5 static inline u16 PA_3DGetSpriteAnimFrame (u16 *sprite*) [inline, static]

Renvoie le numéro actuel de la frame d'animation.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

3.2.2.6 static inline void PA_3DSetSpriteAnimSpeed (u16 *sprite*, s16 *speed*) [inline, static]

Changer la vitesse de l'animation.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

speed Vitesse, en fps...

3.2.2.7 static inline u16 PA_3DGetSpriteAnimSpeed (u16 *sprite*) [inline, static]

Renvoie la vitesse de l'animation.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

3.2.2.8 static inline void PA_3DSetSpriteNCycles (u16 *sprite*, s16 *NCycles*) [inline, static]

Changer le nombre de cycles d'animation restant (-1 pour inifini).

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

NCycles Nombre de cycles

3.2.2.9 static inline u16 PA_3DGetSpriteNCycles (u16 *sprite*) [inline, static]

Renvoie le nombre de cycles d'animation restants.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

3.2.2.10 static inline u16 PA_3DSpriteAnimPause (u16 *sprite*, u8 *pause*) [inline, static]

Mettre en Pause en remettre en lecture une animation de sprite.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

pause 1 pour pause, 0 pour reprendre la lecture...

3.3 Old large background system

Macros

– #define **PA_LoadLargeBg**(screen, bg_select, bg_tiles, bg_map, color_mode, lx, ly)
[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large)

– #define **PA_LoadPAGfxLargeBg**(screen, bg_number, bg_name)
[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), converti avec PAGfx

– #define **PA_LoadLargeBgEx**(screen, bg_select, bg_tiles, tile_size, bg_map, color_mode, lx, ly)
[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), mais ici on met soi-meme la taille des tiles

Fonctions

– static void **PA_InfLargeScrollX** (u8 screen, u8 bg_select, s32 x)
Déplacer un fond à scrolling 'infini' horizontalement. Doit etre initialisé avec PA_LoadLargeBg.

– static void **PA_InfLargeScrollY** (u8 screen, u8 bg_select, s32 y)
Déplacer un fond à scrolling 'infini' verticalement. Doit etre initialisé avec PA_LoadLargeBg.

– static void **PA_InfLargeScrollXY** (u8 screen, u8 bg_select, s32 x, s32 y)
Déplacer un fond à scrolling 'infini' horizontalement et verticalement. Doit etre initialisé avec PA_LoadLargeBg.

– static void **PA_LargeScrollX** (u8 screen, u8 bg_select, s32 x)
Déplacer un grand fond à scrolling horizontalement.Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

– static void **PA_LargeScrollY** (u8 screen, u8 bg_select, s32 y)
Déplacer un grand fond à scrolling verticalement. Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

– static void **PA_LargeScrollXY** (u8 screen, u8 bg_select, s32 x, s32 y)
Déplacer un grand fond à scrolling horizontalement et verticalement. Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

3.3.1 Description détaillée

Old LargeMap functions, obsoleted by **PA_LoadBackground**() (p. 34)

3.3.2 Documentation des macros

3.3.2.1 #define PA_LoadLargeBg(screen, bg_select, bg_tiles, bg_map, color_mode, lx, ly)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_BgInfo[screen][bg_select].NTiles = SIZEOF_16BIT(bg_tiles)>>5;\
    if (PA_BgInfo[screen][bg_select].NTiles < MAX_TILES) {PA_LoadSimpleBg(screen,
       bg_select, bg_tiles, NULL, BG_512X256, 0, color_mode);}\
    else{PA_LoadTileEngine(screen, bg_select, (void*)bg_tiles);}\
    PA_InitLargeBg(screen, bg_select, lx, ly, (void*)bg_map);}while(0)
```

[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large)
Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

color_mode Nombre de couleurs : 0 pour 16 couleurs, 1 pour 256

lx Largeur, en tiles. Un fond de 512 pixels de large fera 64 tiles de large.

ly Hauteur, en tiles. Un fond de 512 pixels de hauy fera 64 tiles de haut.

3.3.2.2 #define PA_LoadPAGfxLargeBg(screen, bg_number, bg_name)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadBgPal(screen, bg_number, (void*)bg_name##_Pal); \
    PA_LoadLargeBg(screen, bg_number, bg_name##_Tiles, bg_name##_Map, 1, (bg_name
      ##_Info[1]) >> 3, (bg_name##_Info[2]) >> 3);}while(0)
```

[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), converti avec PAGfx
Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond que l'on veut charger (0-3)

bg_name Nom du fond dans PAGfx

3.3.2.3 #define PA_LoadLargeBgEx(screen, bg_select, bg_tiles, tile_size, bg_map, color_mode, lx, ly)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_BgInfo[screen][bg_select].NTiles = SIZEOF_16BIT(bg_tiles)>>5;\
    if (PA_BgInfo[screen][bg_select].NTiles < MAX_TILES) {PA_LoadBg(screen, bg_se
      lect, bg_tiles, tile_size, NULL, BG_512X256, 0, color_mode);}\
    else{PA_LoadTileEngine(screen, bg_select, bg_tiles);}\
    PA_InitLargeBg(screen, bg_select, lx, ly, (void*)bg_map);}while(0)
```

[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), mais ici on met soi-meme la taille des tiles
Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

tile_size Taille du tilset

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

color_mode Nombre de couleurs : 0 pour 16 couleurs, 1 pour 256

lx Largeur, en tiles. Un fond de 512 pixels de large fera 64 tiles de large.

ly Hauteur, en tiles. Un fond de 512 pixels de hauy fera 64 tiles de haut.

3.3.3 Documentation des fonctions

3.3.3.1 void PA_InfLargeScrollX (u8 *screen*, u8 *bg_select*, s32 *x*) [inline, static]

Déplacer un fond à scrolling 'infini' horizontalement. Doit etre initialisé avec PA_-LoadLargeBg.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

3.3.3.2 void PA_InfLargeScrollY (u8 *screen*, u8 *bg_select*, s32 *y*) [inline, static]

Déplacer un fond à scrolling 'infini' verticalement. Doit etre initialisé avec PA_-LoadLargeBg.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

y Valeur Y à déplacer

3.3.3.3 static inline void PA_InfLargeScrollXY (u8 *screen*, u8 *bg_select*, s32 *x*, s32 *y*) [inline, static]

Déplacer un fond à scrolling 'infini' horizontalement et verticalement. Doit etre initialisé avec PA_LoadLargeBg.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

y Valeur Y à déplacer

3.3.3.4 void PA_LargeScrollX (u8 *screen*, u8 *bg_select*, s32 *x*) [inline, static]

Déplacer un grand fond à scrolling horizontalement.Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

3.3.3.5 void PA_LargeScrollY (u8 *screen*, u8 *bg_select*, s32 *y*) [inline, static]

Déplacer un grand fond à scrolling verticalement. Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

y Valeur Y à déplacer

3.3.3.6 static inline void PA_LargeScrollXY (u8 *screen*, u8 *bg_select*, s32 *x*, s32 *y*) [inline, static]

Déplacer un grand fond à scrolling horizontalement et verticalement. Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

y Valeur Y à déplacer

3.4 Rotating Backgrounds

Macros

– #define **PA_LoadRotBg**(screen, bg_select, bg_tiles, bg_map, bg_size, wraparound)
[DEPRECATED] Charger un fond pour les rotations/zoom ! Attention, il faut avant utiliser PA_SetVideoMode avec 1 pour utiliser un fond rotatif (le fond 3 uniquement !), ou 2 pour 2 fonds (2 et 3). Le fond DOIT etre de 256 couleurs

– #define **PA_LoadPAGfxRotBg**(screen, bg_select, bg_name, wraparound)
[DEPRECATED] Charger un fond pour les rotations/zoom ! Attention, il faut avant utiliser PA_SetVideoMode avec 1 pour utiliser un fond rotatif (le fond 3 uniquement !), ou 2 pour 2 fonds (2 et 3). Le fond DOIT etre de 256 couleurs

Fonctions

– static void **PA_SetBgRot** (u8 screen, u8 bg_select, s32 x_scroll, s32 y_scroll, s32 x_rotcentre, s32 y_rotcentre, s16 bg_angle, s32 bg_zoom)
Faire tourner/zoomer un fond rotatif.

3.4.1 Description détaillée

Load rotating backgrounds, move, rotate, scale them

3.4.2 Documentation des macros

3.4.2.1 #define PA_LoadRotBg(screen, bg_select, bg_tiles, bg_map, bg_size, wraparound)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_DeleteBg(screen, bg_select);\
    PA_LoadBgTiles(screen, bg_select, bg_tiles); \
    PA_LoadRotBgMap(screen, bg_select, (void*)bg_map, bg_size); \
    PA_InitBg(screen, bg_select, bg_size, wraparound, 1);\
    PA_SetBgRot(screen, bg_select, 0, 0, 0, 0, 0, 256);\
    }while(0)
```

[DEPRECATED] Charger un fond pour les rotations/zoom ! Attention, il faut avant utiliser PA_SetVideoMode avec 1 pour utiliser un fond rotatif (le fond 3 uniquement !), ou 2 pour 2 fonds (2 et 3). Le fond DOIT etre de 256 couleurs

Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

bg_size Taille du fond. Utiliser les macros suivantes : BG_ROT_128X128, ou 256X256, 512X512, ou enfin 1024X1024

wraparound Si le fond boucle ou non.

3.4.2.2 #define PA_LoadPAGfxRotBg(screen, bg_select, bg_name, wraparound)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_Load8bitBgPal(screen, (void*)bg_name##_Pal);\
    PA_LoadRotBg(screen, bg_select, bg_name##_Tiles, bg_name##_Map, PA_GetPAGfxRo
      tBgSize(bg_name##_Info[1]), wraparound);\
    }while(0)
```

[DEPRECATED] Charger un fond pour les rotations/zoom ! Attention, il faut avant utiliser PA_SetVideoMode avec 1 pour utiliser un fond rotatif (le fond 3 uniquement !), ou 2 pour 2 fonds (2 et 3). Le fond DOIT etre de 256 couleurs

Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger

bg_name Nom du fond, comme bg0

wraparound Si le fond boucle ou non.

3.4.3 Documentation des fonctions

3.4.3.1 static inline void PA_SetBgRot (u8 *screen*, u8 *bg_select*, s32 *x_scroll*, s32 *y_scroll*, s32 *x_rotcentre*, s32 *y_rotcentre*, s16 *bg_angle*, s32 *bg_zoom*) [inline, static]

Faire tourner/zoomer un fond rotatif.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger

x_scroll Scroll x...

y_scroll Scroll y...

x_rotcentre Position X du centre de rotation

y_rotcentre Position Y du centre de rotation

bg_angle Angle de rotation (0-511)

bg_zoom Zoom (256 pour pas de zoom...)

3.5 Tiled Background Modes

Structures de données

– struct **PA_BgStruct**
Background structure.

Macros

– #define **_GFX_ALIGN** __attribute__((aligned (4)))
Graphics align define for PAGfx.

– #define **PA_HideBg**(screen, bg_select) _REG16(REG_BGSCREEN(screen)) &= ∼(0x100 << (bg_select))
Cacher un fond.

– #define **PA_ShowBg**(screen, bg_select) _REG16(REG_BGSCREEN(screen)) |= (0x100 << (bg_select))
Afficher un fond auparavant caché.

– #define **PA_ResetBg**(screen) _REG16(REG_BGSCREEN(screen)) &= ∼(0xF00)
Reinitialiser les fonds d'un écran. En fait ca ne fait que cacher tous les fonds.

– #define **PA_LoadBgTiles**(screen, bg_select, bg_tiles) PA_LoadBgTilesEx(screen, bg_select, (void∗)bg_tiles, SIZEOF_16BIT(bg_tiles))
[DEPRECATED] Charger un tileset en mémoire

– #define **PA_LoadTiledBg**(screen, bg_number, bg_name)
[DEPRECATED] On ne pourra jamais rendre ca plus simple... Charge un fond de type TiledBg converti avec PAGfx, en mettant les tiles, la map, et meme la palette ! Seulement en mode 256 couleurs

– #define **PA_LoadSimpleBg**(screen, bg_select, bg_tiles, bg_map, bg_size, wraparound, color_mode)
[DEPRECATED] Façon simple de cahrger un fond. Combine PA_InitBg, PA_LoadBgTiles, et PA_LoadBgMap

– #define **PA_LoadBg**(screen, bg_select, bg_tiles, tile_size, bg_map, bg_size, wraparound, color_mode)
[DEPRECATED] Facon la plus simple de cahrger un fond. Combine PA_InitBg, PA_LoadBgTiles, et PA_LoadBgMap

– #define **PA_SetMapTileAll**(screen, bg_select, x, y, tile_info) ∗(u16∗)(PA_BgInfo[screen][bg_select].Map + ((x) << 1) + ((y) << 6)) = (tile_info)
Change les infos tiles utilisée pour une tile donnée dans la map.

– #define **PA_EasyBgLoad**(screen, bg_number, bg_name)
[DEPRECATED] Moyen le plus simple de charger un fond créé avec PAGfx

– #define **PA_EasyBgLoadPtr**(screen, bg_number, bg_name)
[DEPRECATED] Moyen le plus simple de charger un fond créé avec PAGfx... Peut prendre des pointeurs !

Énumérations

– enum {
PA_BgInvalid, **PA_BgNormal**, **PA_BgLarge**, **PA_BgUnlimited**,
PA_BgRot, **PA_Font1bit**, **PA_Font4bit**, **PA_Font8bit** }
Types of background.

Fonctions

– void **PA_ResetBgSys** (void)
Reinitialise le systeme de fonds.

– void **PA_ResetBgSysScreen** (u8 screen)
Reinitialise le systeme de fonds pour 1 écran.

– void **PA_InitBg** (u8 screen, u8 bg_select, u8 bg_size, u8 wraparound, u8 color_mode)
Initialise un fond. A faire uniquement après avoir chargé un tileset et une map.

– void **PA_LoadBgTilesEx** (u8 screen, u8 bg_select, void ∗bg_tiles, u32 size)
Charger un tileset en mémoire avec une taille donnée.

– void **PA_ReLoadBgTiles** (u8 screen, u8 bg_select, void ∗bg_tiles)
ReCharger un tileset en mémoire.

– void **PA_DeleteTiles** (u8 screen, u8 bg_select)
Effacer un tileset en mémoire. A noter que charger un tileset efface automatiquement le tileset précédent, donc on n'aura pas souvent besoin de cette fonction...

– void **PA_DeleteMap** (u8 screen, u8 bg_select)
Effacer une map en mémoire. A noter que charger une map efface automatiquement la map précédent, donc on n'aura pas souvent besoin de cette fonction...

– static void **PA_DeleteBg** (u8 screen, u8 bg_select)
Effacer et reinitialiser un fond complètement.

– void **PA_LoadBgMap** (u8 screen, u8 bg_select, void ∗bg_map, u8 bg_size)
Charge la carte d'un fond.

– void **PA_LoadBackground** (u8 screen, u8 bg_select, const **PA_BgStruct** ∗bg_name)
Charger un fond (EasyBg ou RotBg).

– static void **PA_BGScrollX** (u8 screen, u8 bg_number, s32 x)
Scroll horizontal d'un fond de type Tiled.

– static void **PA_BGScrollY** (u8 screen, u8 bg_number, s32 y)
Scroll vertical d'un fond de type Tiled.

– static void **PA_SetMapTile** (u8 screen, u8 bg_select, s16 x, s16 y, s16 tile_number)
Change la tile gfx utilisée pour une tile donnée dans la map.

– static void **PA_SetLargeMapTile** (u8 screen, u8 bg_select, s32 x, s32 y, u32 tile_-info)
Change les infos tiles utilisée pour une tile donnée dans la map, seulement pour les grands fonds (512 de large ou haut).

– static void **PA_SetMapTileHflip** (u8 screen, u8 bg_select, u8 x, u8 y, u8 hflip)
Flipper une tile de la carte, horizontalement.

– static void **PA_SetMapTileVflip** (u8 screen, u8 bg_select, u8 x, u8 y, u8 vflip)
Flipper une tile de la carte, verticalement.

– static void **PA_SetMapTilePal** (u8 screen, u8 bg_select, u8 x, u8 y, u8 palette_-number)
Changer la palette de 16 couleurs utilisée par une tile de la carte. Marche uniquement en mode 16 couleurs pour le Bg.

– static void **PA_SetBgPrio** (u8 screen, u8 bg, u8 prio)
Changer la priorité d'un fond.

– static void **PA_SetBgPrioSeq** (u8 screen, u8 priority0, u8 priority1, u8 priority2, u8 priority3)
Changer la priorité des fonds pour qu'ils soient dans un ordre donné.

– static void **PA_ClearBg** (u8 screen, u8 bg_select)
Effacer un fond donné (juste la map).

– void **PA_EasyBgScrollX** (u8 screen, u8 bg_number, s32 x)
Scroll horizontal de n'importe quel fond.

– void **PA_EasyBgScrollY** (u8 screen, u8 bg_number, s32 y)
Scroll vertical de n'importe quel fond.

– static void **PA_EasyBgScrollXY** (u8 screen, u8 bg_number, s32 x, s32 y)
Scroll horizontal et vertical de n'importe quel fond.

– static u8 **PA_EasyBgGetPixel** (u8 screen, u8 bg_number, s32 x, s32 y)
Renvoie le numéro dans la palette du pixel à l'écran...

– static u16 **PA_EasyBgGetPixelCol** (u8 screen, u8 bg_number, s32 x, s32 y)
Renvoie la couleur (valeur u16) du pixel à l'écran...

– static void **PA_SetBgWrap** (u8 screen, u8 bg, u8 wrap)
Active ou non le wrapping des fonds (rotatifs, 8bit, et 16bit).

– static void **PA_InitParallaxX** (u8 screen, s32 bg0, s32 bg1, s32 bg2, s32 bg3)
Initialiser le Parallax Scrolling pour plusieurs fonds, horizontalement. Choix de la vitesse à laquelle les fonds vont défiler par rapport aux autres... Utiliser PA_ParallaxScrollX par la suite pour scroller.

– static void **PA_InitParallaxY** (u8 screen, s32 bg0, s32 bg1, s32 bg2, s32 bg3)
Initialiser le Parallax Scrolling pour plusieurs fonds, horizontalement. Choix de la vitesse à laquelle les fonds vont défiler par rapport aux autres... Utiliser PA_ParallaxScrollX par la suite pour scroller.

– static void **PA_ParallaxScrollX** (u8 screen, s32 x)
Déplacer les fonds activés pour le parallax...

– static void **PA_ParallaxScrollY** (u8 screen, s32 y)
Déplacer les fonds activés pour le parallax...

– static void **PA_ParallaxScrollXY** (u8 screen, s32 x, s32 y)
Déplacer les fonds activés pour le parallax...

3.5.1 Description détaillée

Load a background, scroll it, etc...

3.5.2 Documentation des macros

3.5.2.1 #define PA_HideBg(screen, bg_select) _REG16(REG_BGSCREEN(screen)) &= ∼(0x100 << (bg_select))

Cacher un fond.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

3.5.2.2 #define PA_ShowBg(screen, bg_select) _REG16(REG_BGSCREEN(screen)) |= (0x100 << (bg_select))

Afficher un fond auparavant caché.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

3.5.2.3 #define PA_ResetBg(screen) _REG16(REG_BGSCREEN(screen)) &= ∼(0xF00)

Reinitialiser les fonds d'un écran. En fait ca ne fait que cacher tous les fonds.
Paramètres:

screen Choix de l'écran (0 ou 1)

3.5.2.4 #define PA_LoadBgTiles(screen, bg_select, bg_tiles) PA_LoadBgTilesEx(screen, bg_select, (void∗)bg_tiles, SIZEOF_16BIT(bg_tiles))

[DEPRECATED] Charger un tileset en mémoire

Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

3.5.2.5 #define PA_LoadTiledBg(screen, bg_number, bg_name)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadBgPal(screen, bg_number, (void*)bg_name##_Pal); \
    PA_LoadSimpleBg(screen, bg_number, bg_name##_Tiles, bg_name##_Map, PA_GetPAGf
      xBgSize(bg_name##_Info[1], bg_name##_Info[2]), 0, 1);}while(0)
```

[DEPRECATED] On ne pourra jamais rendre ca plus simple... Charge un fond de type TiledBg converti avec PAGfx, en mettant les tiles, la map, et meme la palette ! Seulement en mode 256 couleurs

Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_name Nom du fond, comme bg0

3.5.2.6 #define PA_LoadSimpleBg(screen, bg_select, bg_tiles, bg_map, bg_size, wraparound, color_mode)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_DeleteBg(screen, bg_select);\
    PA_LoadBgTiles(screen, bg_select, bg_tiles); \
    PA_LoadBgMap(screen, bg_select, (void*)bg_map, bg_size); \
    PA_InitBg(screen, bg_select, bg_size, 0, color_mode);\
    PA_BGScrollXY(screen, bg_select, 0, 0);}while(0)
```

[DEPRECATED] Façon simple de cahrger un fond. Combine PA_InitBg, PA_-LoadBgTiles, et PA_LoadBgMap

Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

bg_size Taille du fond. Pour un fond normal, on utilise les macros BG_256X256, BG_256X512, etc...

wraparound Si le fond boucle ou non. C'est plus important pour les fonds rotatifs...

color_mode Nombre de couleurs : 0 pour 16 couleurs, 1 pour 256

3.5.2.7 #define PA_LoadBg(screen, bg_select, bg_tiles, tile_size, bg_map, bg_size, wraparound, color_mode)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadBgTilesEx(screen, bg_select, (void*)bg_tiles, tile_size); \
    PA_LoadBgMap(screen, bg_select, (void*)bg_map, bg_size); \
    PA_InitBg(screen, bg_select, bg_size, 0, color_mode);\
    PA_BGScrollXY(screen, bg_select, 0, 0);}while(0)
```

[DEPRECATED] Facon la plus simple de cahrger un fond. Combine PA_InitBg, PA_LoadBgTiles, et PA_LoadBgMap

Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

tile_size Taille du tilset

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

bg_size Taille du fond. Ceci est très important, car ça détermine aussi si le Bg est rotatif ou non. Pour un fond normal, on utilise les macros BG_256X256, BG_256X512, etc... Por un fond rotatif, il suffit d'utiliser BG_ROT_128X128...

wraparound Si le fond boucle ou non. C'est plus important pour les fonds rotatifs...

color_mode Nombre de couleurs : 0 pour 16 couleurs, 1 pour 256

3.5.2.8 #define PA_SetMapTileAll(screen, bg_select, x, y, tile_info) *(u16*)(PA_BgInfo[screen][bg_select].Map + ((x) << 1) + ((y) << 6)) = (tile_info)

Change les infos tiles utilisée pour une tile donnée dans la map.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut tourner (0-3)

x Valeur X de la tile à changer

y Valeur Y de la tile à changer dans la carte

tile_info Nouveau numéro de tile que l'on veut mettre (tile + palette + flips...)

3.5.2.9 #define PA_EasyBgLoad(screen, bg_number, bg_name)

Valeur :

```
do{PA_BgInfo[screen][bg_number].BgMode = bg_name##_Info[0];\
    PA_DEPRECATED_MACRO;\
    PA_StoreEasyBgInfos(screen, bg_number, bg_name##_Info[0], bg_name##_Info[1],
      bg_name##_Info[2], (void*)bg_name##_Tiles, SIZEOF_16BIT(bg_name##_Tiles), (void*)
      bg_name##_Map, SIZEOF_16BIT(bg_name##_Map), (void*)bg_name##_Pal);\
    if(PA_BgInfo[screen][bg_number].BgMode == BG_TILEDBG){  PA_LoadTiledBg(screen
      , bg_number, bg_name);}\
    else{PA_LoadPAGfxLargeBg(screen, bg_number, bg_name);}}while(0)
```

[DEPRECATED] Moyen le plus simple de charger un fond créé avec PAGfx
Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond... (0-3)

bg_name Nom du fond

3.5.2.10 #define PA_EasyBgLoadPtr(screen, bg_number, bg_name)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_EasyBgLoadEx(screen, bg_number, (u32*)bg_name->Info, bg_name->Tiles, bg_na
      me->TileSize, bg_name->Map, bg_name->MapSize, bg_name->Palette);\
}while(0)
```

[DEPRECATED] Moyen le plus simple de charger un fond créé avec PAGfx... Peut prendre des pointeurs !
Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond... (0-3)

bg_name Fond, par exemple &bg0

3.5.3 Documentation du type de l'énumération

3.5.3.1 anonymous enum

Types of background.

Valeurs énumérées :

PA_BgInvalid Invalid background.

PA_BgNormal Normal tiled background AKA TiledBg.

PA_BgLarge Large background AKA LargeMap.

PA_BgUnlimited Unlimited background AKA InfiniteMap.

PA_BgRot Rotational background.

PA_Font1bit 1-bit bitmap font

PA_Font4bit 4-bit bitmap font

PA_Font8bit 8-bit bitmap font

3.5.4 Documentation des fonctions

3.5.4.1 void PA_ResetBgSysScreen (u8 *screen*)

Reinitialise le systeme de fonds pour 1 écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

3.5.4.2 void PA_InitBg (u8 *screen*, u8 *bg_select*, u8 *bg_size*, u8 *wraparound*, u8 *color_mode*)

Initialise un fond. A faire uniquement après avoir chargé un tileset et une map.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_size Taille du fond. Ceci est très important, car ça détermine aussi si le Bg est rotatif ou non. Pour un fond normal, on utilise les macros BG_256X256, BG_256X512, etc... Pour un fond rotatif, il suffit d'utiliser BG_ROT_-128X128...

wraparound Si le fond boucle ou non. C'est plus important pour les fonds rotatifs...

color_mode Nombre de couleurs : 0 pour 16 couleurs, 1 pour 256

3.5.4.3 void PA_LoadBgTilesEx (u8 *screen*, u8 *bg_select*, void * *bg_tiles*, u32 *size*)

Charger un tileset en mémoire avec une taille donnée.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

size Taille en 16 bits...

3.5.4.4 void PA_ReLoadBgTiles (u8 *screen*, u8 *bg_select*, void * *bg_tiles*)

ReCharger un tileset en mémoire.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

3.5.4.5 void PA_DeleteTiles (u8 *screen*, u8 *bg_select*)

Effacer un tileset en mémoire. A noter que charger un tileset efface automatiquement le tileset précédent, donc on n'aura pas souvent besoin de cette fonction...

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

3.5.4.6 void PA_DeleteMap (u8 *screen*, u8 *bg_select*)

Effacer une map en mémoire. A noter que charger une map efface automatiquement la map précédent, donc on n'aura pas souvent besoin de cette fonction...

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

3.5.4.7 static inline void PA_DeleteBg (u8 *screen*, u8 *bg_select*) [inline, static]

Effacer et reinitialiser un fond complètement.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

3.5.4.8 void PA_LoadBgMap (u8 *screen*, u8 *bg_select*, void * *bg_map*, u8 *bg_size*)

Charge la carte d'un fond.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_map Nom du tableau contenant les infos sur la map (exemple : (void*)ship_-Map) n'oublie pas le void...

bg_size Taille du fond. Ceci est très important, car ça détermine aussi si le Bg est rotatif ou non. Pour un fond normal, on utilise les macros BG_256X256, BG_256X512, etc...

3.5.4.9 void PA_LoadBackground (u8 *screen*, u8 *bg_number*, const PA_BgStruct * *bg_name*)

Charger un fond (EasyBg ou RotBg).

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond... (0-3)

bg_name Pointeur vers le fond

Exemples :

Backgrounds/Effects/Mode7/source/main.c.

3.5.4.10 static inline void PA_BGScrollX (u8 *screen*, u8 *bg_number*, s32 *x*) [inline, static]

Scroll horizontal d'un fond de type Tiled.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond que l'on veut tourner (0-3)

x Valeur X à déplacer, horizontalement...

3.5.4.11 static inline void PA_BGScrollY (u8 *screen*, u8 *bg_number*, s32 *y*) [inline, static]

Scroll vertical d'un fond de type Tiled.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond que l'on veut tourner (0-3)

y Valeur Y à déplacer, verticalement...

3.5.4.12 static inline void PA_SetMapTile (u8 *screen*, u8 *bg_select*, s16 *x*, s16 *y*, s16 *tile_number*) [inline, static]

Change la tile gfx utilisée pour une tile donnée dans la map.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut tourner (0-3)

x Valeur X de la tile à changer

y Valeur Y de la tile à changer dans la carte

tile_number Nouveau numéro de tile que l'on veut mettre

3.5.4.13 static inline void PA_SetLargeMapTile (u8 *screen*, u8 *bg_select*, s32 *x*, s32 *y*, u32 *tile_info*) [inline, static]

Change les infos tiles utilisée pour une tile donnée dans la map, seulement pour les grands fonds (512 de large ou haut).

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut tourner (0-3)

x Valeur X de la tile à changer

y Valeur Y de la tile à changer dans la carte

tile_info Nouveau numéro de tile que l'on veut mettre (tile + palette + flips...)

3.5.4.14 void PA_SetMapTileHflip (u8 *screen*, u8 *bg_select*, u8 *x*, u8 *y*, u8 *hflip*) [inline, static]

Flipper une tile de la carte, horizontalement.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut tourner (0-3)

x Valeur X de la tile à changer

y Valeur Y de la tile à changer dans la carte

hflip Mettre la tile de la carte en flip horizontal

3.5.4.15 static inline void PA_SetMapTileVflip (u8 *screen*, u8 *bg_select*, u8 *x*, u8 *y*, u8 *vflip*) [inline, static]

Flipper une tile de la carte, verticalement.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut tourner (0-3)

x Valeur X de la tile à changer

y Valeur Y de la tile à changer dans la carte

vflip Mettre la tile de la carte en flip vertical

3.5.4.16 static inline void PA_SetMapTilePal (u8 *screen*, u8 *bg_select*, u8 *x*, u8 *y*, u8 *palette_number*) [inline, static]

Changer la palette de 16 couleurs utilisée par une tile de la carte. Marche uniquement en mode 16 couleurs pour le Bg.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut tourner (0-3)

x Valeur X de la tile à changer

y Valeur Y de la tile à changer dans la carte

palette_number Numéro de la palette (0-15)

3.5.4.17 static inline void PA_SetBgPrio (u8 *screen*, u8 *bg*, u8 *prio*) [inline, static]

Changer la priorité d'un fond.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg Numéro du fond...

prio Niveau de priorité, de 0 à 3, 0 étant priorité la plus élevée

3.5.4.18 static inline void PA_SetBgPrioSeq (u8 *screen*, u8 *priority0*, u8 *priority1*, u8 *priority2*, u8 *priority3*) [inline, static]

Changer la priorité des fonds pour qu'ils soient dans un ordre donné.
Paramètres:

screen Choix de l'écran (0 ou 1)

priority0 Fond à mettre en premier

priority1 Suivant...

priority2 Suivant...

priority3 Dernier...

3.5.4.19 static inline void PA_ClearBg (u8 *screen*, u8 *bg_select*) [inline, static]

Effacer un fond donné (juste la map).
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Fond...

3.5.4.20 void PA_EasyBgScrollX (u8 *screen*, u8 *bg_number*, s32 *x*)

Scroll horizontal de n'importe quel fond.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond que l'on veut tourner (0-3)

x Valeur X à déplacer, horizontalement...

3.5.4.21 void PA_EasyBgScrollY (u8 *screen*, u8 *bg_number*, s32 *y*)

Scroll vertical de n'importe quel fond.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond que l'on veut tourner (0-3)

y Valeur Y à déplacer, verticalement...

3.5.4.22 static inline void PA_EasyBgScrollXY (u8 *screen*, u8 *bg_number*, s32 *x*, s32 *y*) [inline, static]

Scroll horizontal et vertical de n'importe quel fond.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond que l'on veut tourner (0-3)

x Valeur X à déplacer, horizontalement...

y Valeur Y à déplacer, verticalement...

3.5.4.23 static inline u8 PA_EasyBgGetPixel (u8 *screen*, u8 *bg_number*, s32 *x*, s32 *y*) [inline, static]

Renvoie le numéro dans la palette du pixel à l'écran...
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond que l'on veut tourner (0-3)

x Valeur X du pixel à l'écran

y Valeur Y du pixel à l'écran

3.5.4.24 static inline u16 PA_EasyBgGetPixelCol (u8 *screen*, u8 *bg_number*, s32 *x*, s32 *y*) [inline, static]

Renvoie la couleur (valeur u16) du pixel à l'écran...
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_number Numéro du fond que l'on veut tourner (0-3)

x Valeur X du pixel à l'écran

y Valeur Y du pixel à l'écran

3.5.4.25 static inline void PA_SetBgWrap (u8 *screen*, u8 *bg*, u8 *wrap*) [inline, static]

Active ou non le wrapping des fonds (rotatifs, 8bit, et 16bit).
Paramètres:

screen Choix de l'écran (0 ou 1)

bg Numéro du fond que l'on veut tourner (0-3)

wrap Wrap activé ou désactivé...

Exemples :

Backgrounds/Effects/Mode7/source/main.c.

3.5.4.26 static inline void PA_InitParallaxX (u8 *screen*, s32 *bg0*, s32 *bg1*, s32 *bg2*, s32 *bg3*) [inline, static]

Initialiser le Parallax Scrolling pour plusieurs fonds, horizontalement. Choix de la vitesse à laquelle les fonds vont défiler par rapport aux autres... Utiliser PA_-ParallaxScrollX par la suite pour scroller.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg0 Valeur pour le premier fond (0). 256 met en vitesse normal, moins pour moins lent (128 pour moitié de vitesse), plus pour plus rapide (512 équivaut à 2 fois plus vite). On peut utiliser des valeurs négatives. 0 désactive le scrolling parallax pour ce fond

bg1 Idem, pour le Fond 1

bg2 Idem, pour le Fond 2

bg3 Idem, pour le Fond 3

3.5.4.27 static inline void PA_InitParallaxY (u8 *screen*, s32 *bg0*, s32 *bg1*, s32 *bg2*, s32 *bg3*) [inline, static]

Initialiser le Parallax Scrolling pour plusieurs fonds, horizontalement. Choix de la vitesse à laquelle les fonds vont défiler par rapport aux autres... Utiliser PA_-ParallaxScrollX par la suite pour scroller.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg0 Valeur pour le premier fond (0). 256 met en vitesse normal, moins pour moins lent (128 pour moitié de vitesse), plus pour plus rapide (512 équivaut à 2 fois plus vite). On peut utiliser des valeurs négatives. 0 désactive le scrolling parallax pour ce fond

bg1 Idem, pour le Fond 1

bg2 Idem, pour le Fond 2

bg3 Idem, pour le Fond 3

3.5.4.28 static inline void PA_ParallaxScrollX (u8 *screen*, s32 *x*) [inline, static]

Déplacer les fonds activés pour le parallax...

Paramètres:

screen Choix de l'écran (0 ou 1)

x Valeur X à déplacer

3.5.4.29 static inline void PA_ParallaxScrollY (u8 *screen*, s32 *y*) [inline, static]

Déplacer les fonds activés pour le parallax...

Paramètres:

screen Choix de l'écran (0 ou 1)

y Valeur Y à déplacer

3.5.4.30 static inline void PA_ParallaxScrollXY (u8 *screen*, s32 *x*, s32 *y*) [inline, static]

Déplacer les fonds activés pour le parallax...

Paramètres:

screen Choix de l'écran (0 ou 1)

x Valeur X à déplacer

y Valeur Y à déplacer

3.6 Background Transition Effects

Fonctions

– void **PA_InitBgTransEx** (u8 screen, u8 bg)
Initialise le systeme BgTrans sur le fond 0.

– static void **PA_InitBgTrans** (u8 screen)
Initialise le systeme BgTrans sur le fond 0.

– void **PA_BgTransUpDown** (u8 screen, u16 type, u8 vflip, s16 state)
Effet de transition en Haut/Bas.

– void **PA_BgTransLeftRight** (u8 screen, u16 type, u8 hflip, s16 state)
Effet de transition en Gauche/Droite.

– void **PA_BgTransDiag** (u8 screen, u16 type, u8 hflip, u8 vflip, s16 state)
Effet de transition en diagonale.

– void **PA_BgTransCenter** (u8 screen, u16 type, u8 invert, s16 state)
Effet de transition depuis le centre.

3.6.1 Description détaillée

All the different transition effects...

3.6.2 Documentation des fonctions

3.6.2.1 void PA_InitBgTransEx (u8 *screen*, u8 *bg*)

Initialise le systeme BgTrans sur le fond 0.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg Fond (0-3)

3.6.2.2 static inline void PA_InitBgTrans (u8 *screen*) `[inline, static]`

Initialise le systeme BgTrans sur le fond 0.
Paramètres:

screen Choix de l'écran (0 ou 1)

3.6.2.3 void PA_BgTransUpDown (u8 *screen*, u16 *type*, u8 *vflip*, s16 *state*)

Effet de transition en Haut/Bas.
Paramètres:

screen Choix de l'écran (0 ou 1)

type Type de transition... (0-4). Utiliser TRANS_ROUND, TRANS_DIAMOND , TRANS_CROSS, TRANS_LINES, ou TRANS_STAR

vflip Flip vertical...

state Etat, de 0 à TRANS_LENGTH. 0 pour visible, TRANS_LENGTH pour invisible...

3.6.2.4 void PA_BgTransLeftRight (u8 *screen*, u16 *type*, u8 *hflip*, s16 *state*)

Effet de transition en Gauche/Droite.

Paramètres:

screen Choix de l'écran (0 ou 1)

type Type de transition... (0-4). Utiliser TRANS_ROUND, TRANS_DIAMOND , TRANS_CROSS, TRANS_LINES, ou TRANS_STAR

hflip Flip horizontal...

state Etat, de 0 à TRANS_LENGTH. 0 pour visible, TRANS_LENGTH pour invisible...

3.6.2.5 void PA_BgTransDiag (u8 *screen*, u16 *type*, u8 *hflip*, u8 *vflip*, s16 *state*)

Effet de transition en diagonale.

Paramètres:

screen Choix de l'écran (0 ou 1)

type Type de transition... (0-4). Utiliser TRANS_ROUND, TRANS_DIAMOND , TRANS_CROSS, TRANS_LINES, ou TRANS_STAR

hflip Flip horizontal...

vflip Flip vertical...

state Etat, de 0 à TRANS_LENGTH. 0 pour visible, TRANS_LENGTH pour invisible...

3.6.2.6 void PA_BgTransCenter (u8 *screen*, u16 *type*, u8 *invert*, s16 *state*)

Effet de transition depuis le centre.

Paramètres:

screen Choix de l'écran (0 ou 1)

type Type de transition... (0-4). Utiliser TRANS_ROUND, TRANS_DIAMOND , TRANS_CROSS, TRANS_LINES, ou TRANS_STAR

invert Inverser dedans/dehors

state Etat, de 0 à TRANS_LENGTH. 0 pour visible, TRANS_LENGTH pour invisible...

3.7 Debugging utilities

Macros

– #define **PA_Assert**(c, m) ((c) ? ((void)0) : _PA_Assert(#c, m, __FILE__, __LINE_-_))
Afficher un erreur si la condition est faux.

Fonctions

– bool **PA_IsEmulator** ()
Détecte si le programme est exécuté sur un émulateur.

– void **PA_iDeaS_DebugOutput** (const char ∗str)
Sorties de texte à la iDeaS debugging console.

– void **PA_iDeaS_DebugPrintf** (const char ∗str,...)
Sorties de texte formaté à la iDeaS debugging console.

– void **PA_iDeaS_Breakpoint** ()
Mets un breakpoint sur iDeaS.

3.7.1 Description détaillée

Some debugging utilities like emulator detecting and iDeaS debug console printing

3.7.2 Documentation des macros

3.7.2.1 #define PA_Assert(c, m) ((c) ? ((void)0) : _PA_Assert(#c, m, __FILE__, __LINE__))

Afficher un erreur si la condition est faux.
Paramètres:

c Condition, comme MyVar $<$ 128

m Message d'erreur

3.7.3 Documentation des fonctions

3.7.3.1 void PA_iDeaS_DebugOutput (const char ∗ *str*)

Sorties de texte à la iDeaS debugging console.
Paramètres:

str Le texte

3.7.3.2 void PA_iDeaS_DebugPrintf (const char * *str*, ...)

Sorties de texte formaté à la iDeaS debugging console.

Paramètres:

str Le texte

3.8 Bitmap mode

Macros

- #define **PA_Get16bitPixel**(screen, x, y) PA_DrawBg[screen][(x) + ((y) << 8)]
  Récupérer la couleur d'un pixel, en mode dessin 16 bit.

- #define **PA_SetDrawSize**(screen, draw_size) PA_drawsize[screen] = draw_size ;
  Regler la taille du stylo quand on dessine.

- #define **PA_Load8bitBitmap**(screen, bitmap) DMA_Copy(bitmap, (void∗)PA_DrawBg[screen], 256∗96, DMA_16NOW)
  Charger une image à l'écran... pour une fond dessinable de 8 bits.

- #define **PA_Load16bitBitmap**(screen, bitmap)
  Charger une image à l'écran... pour une fond dessinable de 16 bits.

- #define **PA_Clear8bitBg**(screen) dmaFillWords(0, (void∗)PA_DrawBg[screen], 256∗96∗2) ;
  Efface l'écran... pour une fond dessinable de 8 bits.

- #define **PA_Clear16bitBg**(screen) dmaFillWords(0, (void∗)PA_DrawBg[screen], 256∗192∗2)
  Efface l'écran... pour une fond dessinable de 16 bits.

Fonctions

- void **PA_Init8bitBg** (u8 screen, u8 bg_priority)
  Initialise le mode de dessin 8 bit (avec palette). Il suffit de choisir l'écran et la priorité de ce font (de 0 à 3). Ce fond sera placé sur le fond 3 (le remplacant), et doit etre chargé avant tout autre fond ! Prend environ 3/8 de la VRAM.

- void **PA_InitBig8bitBg** (u8 screen, u8 bg_priority)
  Similaire à PA_Init8bitBg, mais avec une taille de 256x256. Ceci prend un peu plus de mémoire, mais autorise le scrolling vertical.

- void **PA_Init16bitBg** (u8 screen, u8 bg_priority)
  Initialise le mode de dessin 16 bit (sans palettes, couleurs RGB). Il suffit de choisir l'écran et la priorité de ce font (de 0 à 3). Ce fond sera placé sur le fond 3 (le remplacant), et doit etre chargé avant tout autre fond ! Prend environ 3/8 de la VRAM.

- static void **PA_Put8bitPixel** (u8 screen, s16 x, s16 y, u8 color)
  Dessine un pixel à l'écran, sur un fond de 8 bits.

- static void **PA_Put2_8bitPixels** (u8 screen, s16 x, s16 y, u16 colors)
  Dessine deux pixels à l'écran, sur un fond de 8 bits. Ces pixels sont contigus, et le premier doit avoir une position X pair. Beaucoup plus rapide que de dessiner les 2 pixels séparemment.

- static void **PA_PutDouble8bitPixels** (u8 screen, s16 x, s16 y, u8 color1, u8 color2)
  Dessine deux pixels à l'écran, sur un fond de 8 bits. Ces pixels sont contigus, et le premier doit avoir une position X pair. Beaucoup plus rapide que de dessiner les 2 pixels séparemment.

– static void **PA_Put4_8bitPixels** (u8 screen, s16 x, s16 y, u32 colors)
Dessine 4 pixels à l'écran, sur un fond de 8 bits. Ces pixels sont contigus, et le premier doit avoir une position X pair. Façon la plus rapide de dessiner à l'écran.

– static u8 **PA_Get8bitPixel** (u8 screen, u8 x, u8 y)
Récupérer la couleur d'un pixel, en mode dessin 8 bit.

– static void **PA_Put16bitPixel** (u8 screen, s16 x, s16 y, u16 color)
Dessine un pixel à l'écran, sur un fond de 16 bits.

– void **PA_Draw8bitLine** (u8 screen, u16 x1, u16 y1, u16 x2, u16 y2, u8 color)
Dessiner une ligne en mode dessin... pour le mode dessin 8 bit.

– void **PA_Draw16bitLine** (u8 screen, u16 x1, u16 y1, u16 x2, u16 y2, u16 color)
Dessiner une ligne en mode dessin... pour le mode dessin 16 bit.

– void **PA_Draw16bitLineEx** (u8 screen, s16 basex, s16 basey, s16 endx, s16 endy, u16 color, s8 size)
Dessiner une ligne épaisse en mode dessin... pour le mode dessin 16 bit.

– void **PA_Draw8bitLineEx** (u8 screen, s16 basex, s16 basey, s16 endx, s16 endy, u8 color, s8 size)
Dessiner une ligne épaisse en mode dessin... pour le mode dessin 8 bit.

– void **PA_Draw16bitRect** (u8 screen, s16 basex, s16 basey, s16 endx, s16 endy, u16 color)
Dessiner rectangle en mode dessin... pour le mode dessin 16 bit.

– void **PA_8bitDraw** (u8 screen, u8 color)
*Pour 8 bit : Jolie petite fonction qui dessine à l'écran ! Tout ce qu'il reste à faire, c'est de choisir la couleur. Si le VBL **PA** (p. 161) n'est pas initialiser, ne pas oublier de rafraichir le Stylet à chaque cycle (et non, pas avec des glacons !). Il suffit d'executer PA_Draw à chaque cycle pour dessiner...*

– void **PA_16bitDraw** (u8 screen, u16 color)
*Pour 16 bit : Jolie petite fonction qui dessine à l'écran ! Tout ce qu'il reste à faire, c'est de choisir la couleur. Si le VBL **PA** (p. 161) n'est pas initialiser, ne pas oublier de rafraichir le Stylet à chaque cycle (et non, pas avec des glacons !). Il suffit d'executer PA_Draw à chaque cycle pour dessiner...*

– static void **PA_LoadJpeg** (u8 screen, void ∗jpeg)
Charger un jpeg sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !

– void **PA_LoadBmpToBuffer** (u16 ∗Buffer, s16 x, s16 y, void ∗bmp, s16 SWidth)
Charger un BMP dans un buffer de 16 bit.

– static void **PA_LoadBmpEx** (u8 screen, s16 x, s16 y, void ∗bmp)
Charger un BMP sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !

– static void **PA_LoadBmp** (u8 screen, void ∗bmp)
Charger un BMP sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !

– static u16 **PA_GetBmpWidth** (void ∗bmpdata)
Récupérer la largeur d'un BMP en pixels.

– static u16 **PA_GetBmpHeight** (void ∗bmpdata)

Récupérer la hauteur d'un BMP en pixels.

3.8.1 Description détaillée

Draw on screen, either a pixel or a line, or anything ! Load a Bitmap, a Jpeg...

3.8.2 Documentation des macros

3.8.2.1 #define PA_Get16bitPixel(screen, x, y) PA_DrawBg[screen][(x) + ((y) << 8)]

Récupérer la couleur d'un pixel, en mode dessin 16 bit.

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y Position Y. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

3.8.2.2 #define PA_SetDrawSize(screen, draw_size) PA_drawsize[screen] = draw_size ;

Regler la taille du stylo quand on dessine.

Paramètres:

screen Choix de l'écran (0 ou 1)

draw_size Taille...

3.8.2.3 #define PA_Load8bitBitmap(screen, bitmap) DMA_Copy(bitmap, (void∗)PA_DrawBg[screen], 256∗96, DMA_16NOW)

Charger une image à l'écran... pour une fond dessinable de 8 bits.

Paramètres:

screen Choix de l'écran (0 ou 1)

bitmap Nom du bitmap

3.8.2.4 #define PA_Load16bitBitmap(screen, bitmap)

Valeur :

```
do{u32 PA_temp; \
    for (PA_temp = 0; PA_temp < 256*192; PA_temp++)\
    PA_DrawBg[screen][PA_temp] = bitmap[PA_temp] | (1 << 15);}while(0)
```

Charger une image à l'écran... pour une fond dessinable de 16 bits.
Paramètres:

screen Choix de l'écran (0 ou 1)

bitmap Nom du bitmap

3.8.2.5 #define PA_Clear8bitBg(screen) dmaFillWords(0, (void∗)PA_DrawBg[screen], 256∗96∗2) ;

Efface l'écran... pour une fond dessinable de 8 bits.
Paramètres:

screen Choix de l'écran (0 ou 1)

3.8.2.6 #define PA_Clear16bitBg(screen) dmaFillWords(0, (void∗)PA_DrawBg[screen], 256∗192∗2)

Efface l'écran... pour une fond dessinable de 16 bits.
Paramètres:

screen Choix de l'écran (0 ou 1)

3.8.3 Documentation des fonctions

3.8.3.1 void PA_Init8bitBg (u8 *screen*, u8 *bg_priority*)

Initialise le mode de dessin 8 bit (avec palette). Il suffit de choisir l'écran et la priorité de ce font (de 0 à 3). Ce fond sera placé sur le fond 3 (le remplacant), et doit etre chargé avant tout autre fond ! Prend environ 3/8 de la VRAM.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_priority

3.8.3.2 void PA_InitBig8bitBg (u8 *screen*, u8 *bg_priority*)

Similaire à PA_Init8bitBg, mais avec une taille de 256x256. Ceci prend un peu plus de mémoire, mais autorise le scrolling vertical.
Paramètres:

screen Choix de l'écran (0 ou 1)

bg_priority

3.8.3.3 void PA_Init16bitBg (u8 *screen*, u8 *bg_priority*)

Initialise le mode de dessin 16 bit (sans palettes, couleurs RGB). Il suffit de choisir l'écran et la priorité de ce font (de 0 à 3). Ce fond sera placé sur le fond 3 (le remplacant), et doit etre chargé avant tout autre fond ! Prend environ 3/8 de la VRAM.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_priority

3.8.3.4 static inline void PA_Put8bitPixel (u8 *screen*, s16 *x*, s16 *y*, u8 *color*) [inline, static]

Dessine un pixel à l'écran, sur un fond de 8 bits.

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X (0-255)

y Position Y (0-191)

color Couleur de la palette du fond (0-255)

3.8.3.5 static inline void PA_Put2_8bitPixels (u8 *screen*, s16 *x*, s16 *y*, u16 *colors*) [inline, static]

Dessine deux pixels à l'écran, sur un fond de 8 bits. Ces pixels sont contigus, et le premier doit avoir une position X pair. Beaucoup plus rapide que de dessiner les 2 pixels séparemment.

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X (0-254), doit etre PAIR

y Position Y (0-191)

colors Couleurs des premier et deuxième pixels (∗256 pour le deuxième)

3.8.3.6 static inline void PA_PutDouble8bitPixels (u8 *screen*, s16 *x*, s16 *y*, u8 *color1*, u8 *color2*) [inline, static]

Dessine deux pixels à l'écran, sur un fond de 8 bits. Ces pixels sont contigus, et le premier doit avoir une position X pair. Beaucoup plus rapide que de dessiner les 2 pixels séparemment.

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X (0-254), doit etre PAIR

y Position Y (0-191)

color1 Couleur de la palette du fond (0-255) pour le premier pixel

color2 Couleur de la palette du fond (0-255) pour le deuxième pixel

3.8.3.7 static inline void PA_Put4_8bitPixels (u8 *screen*, s16 *x*, s16 *y*, u32 *colors*) [inline, static]

Dessine 4 pixels à l'écran, sur un fond de 8 bits. Ces pixels sont contigus, et le premier doit avoir une position X pair. Façon la plus rapide de dessiner à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X (0-254), doit etre PAIR

y Position Y (0-191)

colors Couleurs des 4 pixels

3.8.3.8 static inline u8 PA_Get8bitPixel (u8 *screen*, u8 *x*, u8 *y*) [inline, static]

Récupérer la couleur d'un pixel, en mode dessin 8 bit.

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y Position Y. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

3.8.3.9 static inline void PA_Put16bitPixel (u8 *screen*, s16 *x*, s16 *y*, u16 *color*) [inline, static]

Dessine un pixel à l'écran, sur un fond de 16 bits.

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X (0-255)

y Position Y (0-191)

color Couleur de 16 bits, obtenue avec **PA_RGB(rouge, vert, bleu)** (p. 86)

3.8.3.10 void PA_Draw8bitLine (u8 *screen*, u16 *x1*, u16 *y1*, u16 *x2*, u16 *y2*, u8 *color*)

Dessiner une ligne en mode dessin... pour le mode dessin 8 bit.

Paramètres:

screen Choix de l'écran (0 ou 1)

x1 Position X du premier point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y1 Position Y du premier point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

x2 Position X du deuxième point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y2 Position Y du deuxième point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

color Couleur de la palette du fond (0-255)

3.8.3.11 void PA_Draw16bitLine (u8 *screen*, u16 *x1*, u16 *y1*, u16 *x2*, u16 *y2*, u16 *color*)

Dessiner une ligne en mode dessin... pour le mode dessin 16 bit.

Paramètres:

screen Choix de l'écran (0 ou 1)

x1 Position X du premier point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y1 Position Y du premier point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

x2 Position X du deuxième point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y2 Position Y du deuxième point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

color Couleur de 15 bits.On peut utiliser la macro PA_RGB pour entrer les valeurs RGB...

3.8.3.12 void PA_Draw16bitLineEx (u8 *screen*, s16 *basex*, s16 *basey*, s16 *endx*, s16 *endy*, u16 *color*, s8 *size*)

Dessiner une ligne épaisse en mode dessin... pour le mode dessin 16 bit.

Paramètres:

screen Choix de l'écran (0 ou 1)

basex Position X du premier point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

basey Position Y du premier point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

endx Position X du deuxième point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

endy Position Y du deuxième point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

color Couleur de 15 bits.On peut utiliser la macro PA_RGB pour entrer les valeurs RGB...

size Largeur du trait, en pixels

3.8.3.13 void PA_Draw8bitLineEx (u8 *screen*, s16 *basex*, s16 *basey*, s16 *endx*, s16 *endy*, u8 *color*, s8 *size*)

Dessiner une ligne épaisse en mode dessin... pour le mode dessin 8 bit.

Paramètres:

screen Choix de l'écran (0 ou 1)

basex Position X du premier point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

basey Position Y du premier point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

endx Position X du deuxième point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

endy Position Y du deuxième point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

color Couleur de 15 bits.On peut utiliser la macro PA_RGB pour entrer les valeurs RGB...

size Largeur du trait, en pixels

3.8.3.14 void PA_Draw16bitRect (u8 *screen*, s16 *basex*, s16 *basey*, s16 *endx*, s16 *endy*, u16 *color*)

Dessiner rectangle en mode dessin... pour le mode dessin 16 bit.

Paramètres:

screen Choix de l'écran (0 ou 1)

basex Position X du premier point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

basey Position Y du premier point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

endx Position X du deuxième point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

endy Position Y du deuxième point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

color Couleur de 15 bits.On peut utiliser la macro PA_RGB pour entrer les valeurs RGB...

3.8.3.15 PA_8bitDraw (u8 *screen*, u8 *color*)

Pour 8 bit : Jolie petite fonction qui dessine à l'écran ! Tout ce qu'il reste à faire, c'est de choisir la couleur. Si le VBL **PA** (p. 161) n'est pas initialiser, ne pas oublier de rafraichir le Stylet à chaque cycle (et non, pas avec des glacons !). Il suffit d'executer PA_Draw à chaque cycle pour dessiner...

Paramètres:

screen Choix de l'écran (0 ou 1)

color Numéro de la couleur dans la palette (0-255)

3.8.3.16 PA_16bitDraw (u8 *screen*, u16 *color*)

Pour 16 bit : Jolie petite fonction qui dessine à l'écran ! Tout ce qu'il reste à faire, c'est de choisir la couleur. Si le VBL **PA** (p. 161) n'est pas initialiser, ne pas oublier de rafraichir le Stylet à chaque cycle (et non, pas avec des glacons !). Il suffit d'executer PA_Draw à chaque cycle pour dessiner...

Paramètres:

screen Choix de l'écran (0 ou 1)

color Couleur de 15 bits.On peut utiliser la macro PA_RGB pour entrer les valeurs RGB...

3.8.3.17 static inline void PA_LoadJpeg (u8 *screen*, void * *jpeg*) [inline, static]

Charger un jpeg sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !

Paramètres:

screen Choix de l'écran (0 ou 1)

jpeg image au format jpeg...

3.8.3.18 void PA_LoadBmpToBuffer (u16 * *Buffer*, s16 *x*, s16 *y*, void * *bmp*, s16 *SWidth*)

Charger un BMP dans un buffer de 16 bit.

Paramètres:

Buffer Buffer...

x Position X du coin supérieur gauche

y Position Y du coin supérieur gauche

bmp image au format BMP...

SWidth Largeur du buffer, en pixels (256 pour la taille de l'écran...)

3.8.3.19 static inline void PA_LoadBmpEx (u8 *screen*, s16 *x*, s16 *y*, void * *bmp*) [inline, static]

Charger un BMP sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X du coin supérieur gauche

y Position Y du coin supérieur gauche

bmp image au format BMP...

3.8.3.20 static inline void PA_LoadBmp (u8 *screen*, void * *bmp*) [inline, static]

Charger un BMP sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !

Paramètres:

screen Choix de l'écran (0 ou 1)

bmp image au format BMP...

3.8.3.21 static inline u16 PA_GetBmpWidth (void * *bmp*) [inline, static]

Récupérer la largeur d'un BMP en pixels.

Paramètres:

bmp image au format BMP...

3.8.3.22 static inline u16 PA_GetBmpHeight (void * *bmp*) [inline, static]

Récupérer la hauteur d'un BMP en pixels.

Paramètres:

bmp image au format BMP...

3.9 Fake 16bit bitmap mode

Macros

– #define **PA_LoadFake16bitBitmap**(screen, bitmap) DMA_Copy(bitmap, (void∗)PA_DrawFake16[screen], 256∗192, DMA_16NOW)
[nothing]

– #define **PA_ClearFake16bitBg**(screen) dmaFillWords(0, (void∗)PA_DrawFake16[screen], 256∗192∗2)
[nothing]

– #define **PA_PutFake16bitPixel**(screen, x, y, color) PA_DrawFake16[screen][(x) + 256 ∗ (y)] = color
[nothing]

– #define **PA_GetFake16bitPixel**(screen, x, y) PA_DrawFake16[screen][(x) + 256 ∗ (y)]
[nothing]

– #define **PA_DrawFake16bitRect**(screen, x1, y1, x2, y2, color)
[nothing]

– #define **PA_Fake16bitLoadBmpEx**(screen, bmp, x, y) PA_LoadBmpToBuffer(PA_DrawFake16[screen], x, y, bmp, 256)
[nothing]

– #define **PA_Fake16bitLoadBmp**(screen, bmp) PA_Fake16bitLoadBmpEx(screen, bmp, 0, 0)
[nothing]

– #define **PA_Fake16bitLoadGif**(screen, gif) PA_Fake16bitLoadGifXY(screen, gif, 0, 0)
[nothing]

– #define **PA_Fake16bitLoadJpeg**(screen, jpeg) JPEG_DecompressImage((u8∗)jpeg, PA_DrawFake16[screen], 256, 192)
[nothing]

Fonctions

– void **PA_InitFake16bitBg** (u8 screen, u8 prio)
[nothing]

– void **PA_DrawFake16bitLine** (u8 screen, u16 x1, u16 y1, u16 x2, u16 y2, u16 color)
[nothing]

3.9.1 Description détaillée

Functions to handle fake 16 bit backgrounds that take up less memory than real ones !

3.9.2 Documentation des macros

3.9.2.1 #define PA_LoadFake16bitBitmap(screen, bitmap) DMA_Copy(bitmap, (void∗)PA_DrawFake16[screen], 256∗192, DMA_16NOW)

[nothing]
Paramètres:

screen [nothing]

bitmap [nothing]

3.9.2.2 #define PA_ClearFake16bitBg(screen) dmaFillWords(0, (void∗)PA_DrawFake16[screen], 256∗192∗2)

[nothing]
Paramètres:

screen [nothing]

3.9.2.3 #define PA_PutFake16bitPixel(screen, x, y, color) PA_DrawFake16[screen][(x) + 256 ∗ (y)] = color

[nothing]
Paramètres:

screen [nothing]

x Position X du point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y Position Y du point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

color Couleur de 15 bits.On peut utiliser la macro PA_RGB pour entrer les valeurs RGB...

3.9.2.4 #define PA_GetFake16bitPixel(screen, x, y) PA_DrawFake16[screen][(x) + 256 ∗ (y)]

[nothing]
Paramètres:

screen [nothing]

x Position X du point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y Position Y du point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

3.9.2.5 #define PA_DrawFake16bitRect(screen, x1, y1, x2, y2, color)

Valeur :

```
do{\
    PA_DrawFake16bitLine(screen, x1, y1, x2, y1, color);\
    PA_DrawFake16bitLine(screen, x1, y1, x1, y2, color);\
    PA_DrawFake16bitLine(screen, x2, y1, x2, y2, color);\
    PA_DrawFake16bitLine(screen, x1, y2, x2, y2, color);}while(0)
```

[nothing]

Paramètres:

screen [nothing]

x1 Position X du premier point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y1 Position Y du premier point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

x2 Position X du deuxième point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y2 Position Y du deuxième point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

color Couleur de 15 bits.On peut utiliser la macro PA_RGB pour entrer les valeurs RGB...

3.9.2.6 #define PA_Fake16bitLoadBmpEx(screen, bmp, x, y) PA_LoadBmpToBuffer(PA_DrawFake16[screen], x, y, bmp, 256)

[nothing]

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X du coin supérieur gauche

y Position Y du coin supérieur gauche

bmp image au format BMP...

3.9.2.7 #define PA_Fake16bitLoadBmp(screen, bmp) PA_-Fake16bitLoadBmpEx(screen, bmp, 0, 0)

[nothing]

Paramètres:

screen Choix de l'écran (0 ou 1)

bmp image au format BMP...

3.9.2.8 #define PA_Fake16bitLoadGif(screen, gif) PA_-Fake16bitLoadGifXY(screen, gif, 0, 0)

[nothing]

Paramètres:

screen Choix de l'écran (0 ou 1)

gif image au format Gif...

3.9.2.9 #define PA_Fake16bitLoadJpeg(screen, jpeg) JPEG_-DecompressImage((u8∗)jpeg, PA_DrawFake16[screen], 256, 192)

[nothing]

Paramètres:

screen Choix de l'écran (0 ou 1)

jpeg image au format jpeg...

3.9.3 Documentation des fonctions

3.9.3.1 void PA_InitFake16bitBg (u8 *screen*, u8 *prio*)

[nothing]

Paramètres:

screen [nothing]

prio [nothing]

3.9.3.2 void PA_DrawFake16bitLine (u8 *screen*, u16 *x1*, u16 *y1*, u16 *x2*, u16 *y2*, u16 *color*)

[nothing]

Paramètres:

screen [nothing]

x1 Position X du premier point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y1 Position Y du premier point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

x2 Position X du deuxième point. Attention, si X n'est pas compris entre 0 et 255, le résultat ne sera pas celui escompté

y2 Position Y du deuxième point. Attention, si Y n'est pas compris entre 0 et 191, le résultat ne sera pas celui escompté

color Couleur de 15 bits.On peut utiliser la macro PA_RGB pour entrer les valeurs RGB...

3.10 General Functions

Structures de données

– struct **PA_FifoMsg**
Represents a message sent through Fifo.

– struct **PA_TransferRegion**
PAlib transfer region type.

Macros

– #define **FIFO_PALIB** FIFO_SOUND
PAlib Fifo channel number...

– #define **PA_SendFifoMsg**(msg) fifoSendDatamsg(FIFO_PALIB, sizeof(**PA_FifoMsg**), (u8∗) &msg)
*Send a **PA_FifoMsg** (p. 170) structure to the other CPU.*

– #define **PA_SendFifoVal**(val) fifoSendValue32(FIFO_PALIB, val)
Send a 32bit value to the other CPU.

– #define **PA_SendFifoCmd** PA_SendFifoVal
Send a command value to the other CPU (same as PA_SendFifoVal but for readability).

– #define **PA_GetFifoMsg**(msg, bytes) fifoGetDatamsg(FIFO_PALIB, bytes, (u8∗) &msg)
*Receive a **PA_FifoMsg** (p. 170) structure from the other CPU.*

– #define **PA_FifoRetWait**() while(!fifoCheckValue32(FIFO_PALIB))
Wait for the other CPU to send a return value.

– #define **PA_FifoRetVal**() fifoGetValue32(FIFO_PALIB)
Get the other CPU's return value.

– #define **PA_Transfer** ((volatile **PA_TransferRegion**∗) 0x02FFF100)
PAlib transfer region (used for the storage of data coming from the ARM7). libnds also does this. As TransferRegion was removed we just skip the first 256 bytes.

– #define **PA_LegacyIPCInit**()
TODO.

– #define **PA_LidClosed**() _PA_LidDown
Vérifie si la DS est fermée. Renvoie 0 si ouverte, 1 si fermée.

– #define **PA_CloseLidSound**(close_sound)
Vérifie si la DS est fermée. Si fermée, ca met en pause la DS et joue un son.

– #define **PA_CloseLidSound2**(close_sound, open_sound)
Vérifie si la DS est fermée. Si fermée, ca met en pause la DS et joue un son.

– #define **PA_WaitFor**(something) do{while(!(something)) PA_-WaitForVBL() ;}while(0)
Attendre un événement précis...

Énumérations

– enum { **PA_MSG_INPUT** = 0x7000, **PA_MSG_MIC** = 0x7100, **PA_MSG_-DSLBRIGHT** = 0x7102, **PA_MSG_PSG** = 0x7103 }
PA_FifoMsg *(p. 170) message types.*

– enum { **PA_MSG_MICSTOP** = 0x7101 }
PA_SendFifoCmd() *(p. 58) commands.*

Fonctions

– static u32 **PA_FifoGetRetVal** ()
Inline function to ease the getting of the return value (wait + get).

– void **PA_Init** ()
Initialise la lib... Doit etre placé au début de main().

– void **PA_InitFifo** ()
Initialise le système Fifo. C'est fait automatiquement dans ***PA_Init()*** *(p. 59).*

– void **PA_Init2D** ()
Remet en mode 2D après avoir utilisé la 3D.

– void **PA_SetVideoMode** (u8 screen, u8 mode)
Changer de mode video... A utiliser avec précaution.

– void **PA_UpdateUserInfo** (void)
Met à jour les infos sur l'utilisateur... C'est fait automatiquement dans PA_Init. On peut ensuite récupérer toutes les infos avec PA_UserInfo.Color (couleur favorite), .BdayDay, .BdayMonth, .AlarmHour, .AlarmMinute, .Name, .NameLength, .Message, .MessageLength, .Language.

– void **PA_UpdateRTC** (void)
Met à jour les infos sur l'heure et la date. C'est automatiquement mis à jour dans le VBL ***PA*** *(p. 161)... On récupère les infos avec PA_RTC.Minutes, .Hour, .Seconds, .Day, .Month, et .Year.*

– static void **PA_SwitchScreens** ()
Echange les écrans du haut et du bas.

– static void **PA_SetAutoCheckLid** (u8 on)
Vérifie automatiquement si la DS est fermée dans PA_WaitForVBL.

– static void **PA_SetLedBlink** (u8 blink, u8 speed)
Régler le clignotement de la led.

– u8 **PA_CheckLid** ()
Vérifie si la DS est fermée. Si fermée, ca met en pause la DS et renvoie 1.

– static void **PA_WaitForVBL** ()
Attendre le vbl...

– static void **PA_SetScreenLight** (u8 screen, u8 light)
Allumer ou eteindre la lumière d'un écran.

– static void **PA_SetDSLBrightness** (u8 level)
Régler le niveau de lumière de la DS Lite !

– bool **PA_Locate** (char ∗start, char ∗target, bool isDir, int depth, char ∗result)
Find a directory in the file system within a given depth.

– void **PA_Error** (const char ∗text)
Displays an error message.

3.10.1 Description détaillée

Initialise the lib, and other general functions...

3.10.2 Documentation des macros

3.10.2.1 #define PA_LegacyIPCInit()

Valeur :

```
do{ \
    memset((void*) &PA_IPC, 0, sizeof(PA_IPCType)); \
    PA_Transfer->mailData = (u32)(&PA_IPC); \
}while(0)
```

TODO.
Obsolète

3.10.2.2 #define PA_CloseLidSound(close_sound)

Valeur :

```
do{\
    if(PA_LidClosed()){\
        PA_PlaySimpleSound(close_sound);\
        PA_CheckLid(); \
    }}while(0)
```

Vérifie si la DS est fermée. Si fermée, ca met en pause la DS et joue un son.
Paramètres:

close_sound Son à jouer, regarder la doc son si pas certain de quoi mettre...

3.10.2.3 #define PA_CloseLidSound2(close_sound, open_sound)

Valeur :

```
do{\
    if(PA_LidClosed()){\
        PA_PlaySimpleSound(close_sound);\
        PA_CheckLid(); \
        PA_PlaySimpleSound(open_sound); \
    }}while(0)
```

Vérifie si la DS est fermée. Si fermée, ca met en pause la DS et joue un son.

Paramètres:

close_sound Son à jouer quand se ferme, regarder la doc son si pas certain de quoi mettre...

open_sound Son à jouer quand s'ouvre, regarder la doc son si pas certain de quoi mettre...

3.10.2.4 #define PA_WaitFor(something) do{while(!(something)) PA_WaitForVBL() ;}while(0)

Attendre un événement précis...

Paramètres:

something Evénement à attendre, comme Pad.Newpress.A, ou Stylus.Newpress, etc...

3.10.3 Documentation du type de l'énumération

3.10.3.1 anonymous enum

PA_FifoMsg (p. 170) message types.

Valeurs énumérées :

PA_MSG_INPUT Input message (ARM7->ARM9).

PA_MSG_MIC Microphone record message (ARM9->ARM7).

PA_MSG_DSLBRIGHT DS lite screen brightness message (ARM9->ARM7).

PA_MSG_PSG PSG play message (ARM9->ARM7).

3.10.3.2 anonymous enum

PA_SendFifoCmd() (p. 58) commands.

Valeurs énumérées :

PA_MSG_MICSTOP Microphone stop recording message (ARM9->ARM7).

3.10.4 Documentation des fonctions

3.10.4.1 void PA_SetVideoMode (u8 *screen*, u8 *mode*)

Changer de mode video... A utiliser avec précaution.

Paramètres:

screen Ecran...

mode Mode 0 pour normal, 1 pour 1 fond rotatif, 2 pour 2

Exemples :

Backgrounds/Effects/Mode7/source/main.c.

3.10.4.2 static inline void PA_SetAutoCheckLid (u8 *on*) [inline, static]

Vérifie automatiquement si la DS est fermée dans PA_WaitForVBL.

Paramètres:

on 1 pour activer, 0 pour désactiver

3.10.4.3 static void PA_SetLedBlink (u8 *blink*, u8 *speed*) [inline, static]

Régler le clignotement de la led.

Paramètres:

blink 1 pour clignoter, 0 pour toujours allumé

speed Vitesse : 0 pour lent, 1 pour rapide

3.10.4.4 void PA_SetScreenLight (u8 *screen*, u8 *light*) [inline, static]

Allumer ou eteindre la lumière d'un écran.

Paramètres:

screen Ecran...

light Lumière, 1 pour allumé, 0 pour éteint

3.10.4.5 static inline void PA_SetDSLBrightness (u8 *level*) [inline, static]

Régler le niveau de lumière de la DS Lite !

Paramètres:

level Niveau de la lumière (0-3)

3.10.4.6 bool PA_Locate (char * *start*, char * *target*, bool *isDir*, int *depth*, char * *result*)

Find a directory in the file system within a given depth.

Paramètres:

start from which directory to start, use "/" to search from the root

target what to look for : the name of a file or directory

isDir look for a directory or a file ?

depth how much depth level (in number of directories) to traverse ; limiting this speeds up the search on crowded cards. A reasonable value is, for example, 3.

result pointer to a buffer where the result will be stored

Renvoie:

true if the target was found

3.11 Gif functions

Fonctions

- static u16 **PA_GetGifWidth** (void ∗gif)
  Récupérer la largeur d'un Gif en pixels.
- static u16 **PA_GetGifHeight** (void ∗gif)
  Récupérer la hauteur d'un Gif en pixels.
- static void **PA_LoadGifXY** (u8 screen, s16 x, s16 y, void ∗gif)
  Charger un Gif sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !
- static void **PA_LoadGif** (u8 screen, void ∗gif)
  Charger un Gif sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !
- static void **PA_GifAnimSpeed** (float speed)
  Changer la vitesse d'un gif.
- static void **PA_GifAnimStop** (void)
  Arrêter l'animation d'un gif.
- static void **PA_GifAnimPause** (void)
  Mettre en pause l'animation d'un gif.
- static void **PA_GifSetStartFrame** (s32 StartFrame)
  Régler à partir de quelle image commencer le gif.
- static void **PA_GifSetEndFrame** (s32 EndFrame)
  Régler à partir de quelle image arrêter le gif.
- static s32 **PA_GifGetFrame** (void)
  Renvoie le numéro d'image du gif en cours.

3.11.1 Description détaillée

Manages everything about gif files.

3.11.2 Documentation des fonctions

3.11.2.1 static inline u16 PA_GetGifWidth (void ∗ *gif*) [inline, static]

Récupérer la largeur d'un Gif en pixels.

Paramètres:

gif image au format Gif...

3.11.2.2 static inline u16 PA_GetGifHeight (void ∗ *gif*) [inline, static]

Récupérer la hauteur d'un Gif en pixels.

Paramètres:

gif image au format Gif...

3.11.2.3 static inline void PA_LoadGifXY (u8 *screen*, s16 *x*, s16 *y*, void * *gif*) [inline, static]

Charger un Gif sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !

Paramètres:

screen Choix de l'écran (0 ou 1)

x Position X à l'écran

y Position Y à l'écran

gif image au format Gif...

3.11.2.4 static inline void PA_LoadGif (u8 *screen*, void * *gif*) [inline, static]

Charger un Gif sur un fond de 16 bits... Faut pas oublier de charger ce fond avant !

Paramètres:

screen Choix de l'écran (0 ou 1)

gif image au format Gif...

3.11.2.5 static inline void PA_GifAnimSpeed (float *speed*) [inline, static]

Changer la vitesse d'un gif.

Paramètres:

speed 1 pour normal, 2 pour 2x, 0.5 pour la moitié...

3.11.2.6 static inline void PA_GifAnimStop (void) [inline, static]

Arrêter l'animation d'un gif. Reprendre l'animation d'un gif.

3.11.2.7 static inline void PA_GifSetStartFrame (s32 *StartFrame*) [inline, static]

Régler à partir de quelle image commencer le gif.

Paramètres:

StartFrame Image où démarrer... (0 pour le début)

3.11.2.8 static inline void PA_GifSetEndFrame (s32 *EndFrame*) [inline, static]

Régler à partir de quelle image arrêter le gif.

Paramètres:

EndFrame Image où démarrer... (100000 si vous voulez être sure de finir ^^)

3.12 Keyboard

Macros

– #define **PA_InitKeyboard** PA_LoadDefaultKeyboard
*Old name for **PA_LoadDefaultKeyboard()** (p. 68).*

– #define **PA_InitCustomKeyboard**(bg_number, keyb_custom)
[DEPRECATED] Initialiser un clavier perso sur un fond donné

– #define **PA_EraseLastKey**() PA_SetLetterPal(PA_Keyboard_Struct.oldX, PA_Keyboard_Struct.oldY, 15)
Effacer la dernière touche pressée, si ca ne le fait pas tout seul.

Fonctions

– void **PA_LoadDefaultKeyboard** (u8 bg_number)
Initialiser le calvier sur un fond donné. Utilise les palettes de 16 couleurs 14 et 15 (n'interfère pas avec le texte).

– void **PA_LoadKeyboard** (u8 bg_number, const **PA_BgStruct** ∗keyboard)
Initialiser un clavier perso sur un fond donné.

– char **PA_CheckKeyboard** (void)
Vérifie le clavier, s'il est utilisé, et renvoie la lettre appuyée (0 si pas de nouvel appuye). A utiliser tout le temps, même si le stylet ne touche pas l'écran.

– static void **PA_ScrollKeyboardX** (s16 x)
Placer le Clavier à la position X.

– static void **PA_ScrollKeyboardY** (s16 y)
Placer le Clavier à la position Y.

– static void **PA_ScrollKeyboardXY** (s16 x, s16 y)
Placer le Clavier à une position donnée.

– static void **PA_KeyboardIn** (s16 x, s16 y)
Faire entrer le clavier à la position (x, y) en glissant depuis le bas de l'écran.

– static void **PA_KeyboardOut** (void)
Faire sortir le clavier.

– void **PA_ReloadKeyboardCol** (void)
Recharge la palette du clavier, utile si on a changé de palette pour les fonds.

– static void **PA_SetKeyboardColor** (u8 color1, u8 color2)
On peut changer la couleur du clavier !

– static void **PA_SetKeyboardScreen** (u8 screen)
Régler l'écran du clavier. Doit être utilisé AVANT l'init du clavier.

3.12.1 Description détaillée

Load a keyboard and have fun

3.12.2 Documentation des macros

3.12.2.1 #define PA_InitCustomKeyboard(bg_number, keyb_custom)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadBgPal(keyb_screen, bg_number, (void*)keyb_custom##_Pal);\
    PA_LoadSimpleBg(keyb_screen, bg_number, keyb_custom##_Tiles, keyb_custom##_Ma
      p, BG_256X512, 1, 1);\
    PA_Keyboard_Struct.Bg = bg_number;  PA_Keyboard_Struct.Type = 0;    PA_Keyboa
      rd_Struct.Repeat = 0;\
    PA_Keyboard_Struct.Custom = 1;\
    PA_BgInfo[keyb_screen][PA_Keyboard_Struct.Bg].Map = (u32)keyb_custom##_Map;\
}while(0)
```

[DEPRECATED] Initialiser un clavier perso sur un fond donné

Obsolète

Paramètres:

bg_number Numéro du fond que l'on veut tourner (0-3)

keyb_custom Claviet perso, converti comme EasyBg

3.12.3 Documentation des fonctions

3.12.3.1 void PA_LoadDefaultKeyboard (u8 *bg_number*)

Initialiser le calvier sur un fond donné. Utilise les palettes de 16 couleurs 14 et 15 (n'interfère pas avec le texte).

Paramètres:

bg_number Numéro du fond que l'on veut tourner (0-3)

3.12.3.2 void PA_LoadKeyboard (u8 *bg_number*, const PA_BgStruct * *keyboard*)

Initialiser un clavier perso sur un fond donné.

Paramètres:

bg_number Numéro du fond que l'on veut tourner (0-3)

keyboard Pointeur vers le fond du claviet perso, converti comme EasyBg

3.12.3.3 static inline void PA_ScrollKeyboardX (s16 *x*) [inline, static]

Placer le Clavier à la position X.

Paramètres:

x Position X

3.12.3.4 static inline void PA_ScrollKeyboardY (s16 *y*) [inline, static]

Placer le Clavier à la position Y.

Paramètres:

y Position Y

3.12.3.5 static inline void PA_ScrollKeyboardXY (s16 *x*, s16 *y*) [inline, static]

Placer le Clavier à une position donnée.

Paramètres:

x Position X

y Position Y

3.12.3.6 static inline void PA_KeyboardIn (s16 *x*, s16 *y*) [inline, static]

Faire entrer le clavier à la position (x, y) en glissant depuis le bas de l'écran.

Paramètres:

x Position X

y Position Y

3.12.3.7 static inline void PA_SetKeyboardColor (u8 *color1*, u8 *color2*) [inline, static]

On peut changer la couleur du clavier !

Paramètres:

color1 Couleur normale, 0 pour bleu, 1 pour rouge, 2 pour vert

color2 Couleur de la touche appuyée, 0 pour bleu, 1 pour rouge, 2 pour vert

3.12.3.8 static inline void PA_SetKeyboardScreen (u8 *screen*) [inline, static]

Régler l'écran du clavier. Doit être utilisé AVANT l'init du clavier.

Paramètres:

screen 0 (bas) or 1 (haut)

3.13 Key input system

Macros

– #define **PA_MoveSprite**(sprite) PA_MoveSpriteEx(PA_Screen, sprite, PA_-GetSpriteLx(0, sprite), PA_GetSpriteLy(0, sprite))

Déplacer un sprite en fonction du stylet. Le sprite sera accroché si le stylet passe aud-dessus, puis il sera déplacé en fonction... Donne 1 si on a déplacé ce sprite, sinon 0. On peut ensuite récupérer des infos avec PA_MovedSprite.Moving (1 si on déplace un sprite), .Sprite (numéro du sprite déplacé), .X (position X du centre du sprite), .Y (position Y du centre du sprite déplacé), .Vx (vitesse horizontale du sprite déplacé ! ! Utile si l'on veut que le sprite continue à se déplacer par la suite...), et .Vy.

– #define **PA_StylusInZone**(x1, y1, x2, y2) ((Stylus.X>=x1)&&(Stylus.Y>=y1)&&(Stylus.X<x2)&&(Stylus.

Vérifie si le stylet est dans une zone délimitée donnée... Renvoie 1 si oui, 0 sinon.

Fonctions

– void **PA_UpdatePad** ()

Permet de mettre à jour les touches appuyées. A utilisé une fois par frame (genre dans le vbl). On a ensuite accès aux touches pressées avec Pad.Held.A (ou Up, Down, L...), aux touches nouvellement pressées avec Pad.Newpress.R, et aux touches tout juste relachées avec Pad.Released.Up...

– void **PA_UpdateStylus** ()

Mettre à jour la position du stylet. On peut vérifier si le stylet est actuellement sur l'écran (Stylus.Held), tout just appuyé (Stylus.Newpress), ou relaché (Stylus.Released), et obtenir sa position (Stylus.X, Stylus.Y).

– u8 **PA_MoveSpritePix** (u8 sprite)

Déplacer un sprite en fonction du stylet, avec détection au pixel pret. Ceci est comme PA_MoveSprite, mais un peu plus lent, et nécessite PA_InitSpriteDraw(screen, sprite). Le sprite sera accroché si le stylet passe aud-dessus, puis il sera déplacé en fonction... Donne 1 si on a déplacé ce sprite, sinon 0. On peut ensuite récupérer des infos avec PA_-MovedSprite.Moving (1 si on déplace un sprite), .Sprite (numéro du sprite déplacé), .X (position X du coin sup gauche du sprite), .Y (position Y du point sup gauche du sprite déplacé), .Vx (vitesse horizontale du sprite déplacé ! ! Utile si l'on veut que le sprite continue à se déplacer par la suite...), et .Vy.

– u8 **PA_MoveSpriteEx** (u8 screen, u8 sprite, u8 lx, u8 ly)

Déplacer un sprite en fonction du stylet. Voir PA_MoveSprite pour plus de détails. La différence est qu'ici on précise la largeur et la hauteur du sprite, utile si le sprite ne fait pas vraiment la meme taille que la taille standard DS (genre si c'est un sprite de 20x20). Ceci limitera donc aussi la distance d'accrochage.

– static u8 **PA_MoveSpriteDistance** (u8 sprite, u8 distance)

Déplacer un sprite en fonction du stylet. Voir PA_MoveSprite pour plus de détails. La différence est qu'ici on précise la distance d'accrochage, en pixels.

– static u8 **PA_SpriteStylusOverEx** (u8 sprite, u8 lx, u8 ly)

Vérifie si le stylet est placé au-dessus d'un sprite donné (que le stylet touche l'écran ou non).

– static u8 **PA_SpriteTouchedEx** (u8 sprite, u8 lx, u8 ly)
Vérifie si l'on touche un sprite donné. Renvoie 1 si touché... On peut choisir la hauteur et la largeur autour du sprite.

– static u8 **PA_SpriteTouched** (u8 sprite)
Vérifie si l'on touche un sprite donné. Renvoie 1 si touché...

– static u8 **PA_SpriteStylusOver** (u8 sprite)
Vérifie si le stylet est placé au-dessus d'un sprite donné (que le stylet touche l'écran ou non).

3.13.1 Description détaillée

Check which keys are pressed...

3.13.2 Documentation des macros

3.13.2.1 #define PA_MoveSprite(sprite) PA_MoveSpriteEx(PA_Screen, sprite, PA_GetSpriteLx(0, sprite), PA_GetSpriteLy(0, sprite))

Déplacer un sprite en fonction du stylet. Le sprite sera accroché si le stylet passe aud-dessus, puis il sera déplacé en fonction... Donne 1 si on a déplacé ce sprite, sinon 0. On peut ensuite récupérer des infos avec PA_MovedSprite.Moving (1 si on déplace un sprite), .Sprite (numéro du sprite déplacé), .X (position X du centre du sprite), .Y (position Y du centre du sprite déplacé), .Vx (vitesse horizontale du sprite déplacé ! ! Utile si l'on veut que le sprite continue à se déplacer par la suite...), et .Vy.

Paramètres:

sprite Numéro de l'objet dans le systeme de sprite

3.13.2.2 #define PA_StylusInZone(x1, y1, x2, y2) ((Stylus.X>=x1)&&(Stylus.Y>=y1)&&(Stylus.X<x2)&&(Stylus.Y<y2))

Vérifie si le stylet est dans une zone délimitée donnée... Renvoie 1 si oui, 0 sinon.

Paramètres:

x1 Valeur X du coin supérieur gauche

y1 Valeur Y du coin supérieur gauche

x2 Valeur X du coin inférieur droit

y2 Valeur Y du coin inférieur droit

3.13.3 Documentation des fonctions

3.13.3.1 u8 PA_MoveSpritePix (u8 *sprite*)

Déplacer un sprite en fonction du stylet, avec détection au pixel pret. Ceci est comme PA_MoveSprite, mais un peu plus lent, et nécessite PA_InitSpriteDraw(screen, sprite). Le sprite sera accroché si le stylet passe aud-dessus, puis il sera déplacé en fonction... Donne 1 si on a déplacé ce sprite, sinon 0. On peut ensuite récupérer des infos avec PA_MovedSprite.Moving (1 si on déplace un sprite), .Sprite (numéro du sprite déplacé), .X (position X du coin sup gauche du sprite), .Y (position Y du point sup gauche du sprite déplacé), .Vx (vitesse horizontale du sprite déplacé ! ! Utile si l'on veut que le sprite continue à se déplacer par la suite...), et .Vy.

Paramètres:

sprite Numéro de l'objet dans le systeme de sprite

3.13.3.2 u8 PA_MoveSpriteEx (u8 *screen*, u8 *sprite*, u8 *lx*, u8 *ly*)

Déplacer un sprite en fonction du stylet. Voir PA_MoveSprite pour plus de détails. La différence est qu'ici on précise la largeur et la hauteur du sprite, utile si le sprite ne fait pas vraiment la meme taille que la taille standard DS (genre si c'est un sprite de 20x20). Ceci limitera donc aussi la distance d'accrochage.

Paramètres:

screen Sur quel écran le faire...

sprite Numéro de l'objet dans le systeme de sprite

lx Largeur du sprite

ly Hauteur du sprite

3.13.3.3 u8 PA_MoveSpriteDistance (u8 *sprite*, u8 *distance*) `[inline, static]`

Déplacer un sprite en fonction du stylet. Voir PA_MoveSprite pour plus de détails. La différence est qu'ici on précise la distance d'accrochage, en pixels.

Paramètres:

sprite Numéro de l'objet dans le systeme de sprite

distance Distance d'accrochage

3.13.3.4 static inline u8 PA_SpriteStylusOverEx (u8 *sprite*, u8 *lx*, u8 *ly*) `[inline, static]`

Vérifie si le stylet est placé au-dessus d'un sprite donné (que le stylet touche l'écran ou non).

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

lx Largeur

ly Hauter

3.13.3.5 static inline u8 PA_SpriteTouchedEx (u8 *sprite*, u8 *lx*, u8 *ly*) [inline, static]

Vérifie si l'on touche un sprite donné. Renvoie 1 si touché... On peut choisir la hauteur et la largeur autour du sprite.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

lx Largeur

ly Hauter

3.13.3.6 static inline u8 PA_SpriteTouched (u8 *sprite*) [inline, static]

Vérifie si l'on touche un sprite donné. Renvoie 1 si touché...

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

3.13.3.7 static inline u8 PA_SpriteStylusOver (u8 *sprite*) [inline, static]

Vérifie si le stylet est placé au-dessus d'un sprite donné (que le stylet touche l'écran ou non).

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

3.14 Special controllers

3.14.1 Description détaillée

Macros, variables, and prototypes needed for DS controller accessory (Guitar Hero Grip, Taito Paddle, ...) support.

3.15 Math functions

Structures de données

– struct **PA_Point**
Simple point structure.

Macros

– #define **PA_Cos**(angle) PA_SIN[((angle) + 128)&511]
Renvoie la valeur Cosinus d'un angle. Cette valeur est comprise entre -256 et 256... Attention : l'angle n'est pas en 360 degr ?s, mais en 512 !

– #define **PA_Sin**(angle) PA_SIN[((angle))&511]
Renvoie la valeur Sinus d'un angle. Cette valeur est comprise entre -256 et 256... Attention : l'angle n'est pas en 360 degr ?s, mais en 256 !

Fonctions

– static u32 **PA_Rand** ()
Donne un numéro aléatoire... Ceci est pris de Ham, je n'ai aucun m ?rite.

– static void **PA_InitRand** ()
Initialisation automatique du rand, basé sur l'horloge interne...

– static void **PA_SRand** (s32 r)
Initialiser le random avec un seed. Ceci est pris de Ham, je n'ai aucun mérite. J'ai juste raccourci/accéléré un peu le tout.

– static u32 **PA_RandMax** (u32 max)
Donne un num ?ro al ?atoire... Entre 0 et le nombre donn ? (inclus).

– static u32 **PA_RandMinMax** (u32 min, u32 max)
Donne un num ?ro al ?atoire... Entre les 2 nombres donn ?s (inclus).

– static u64 **PA_Distance** (s32 x1, s32 y1, s32 x2, s32 y2)
Calculer la distance (au carr ?) entre 2 points.

– static u64 **PA_TrueDistance** (s32 x1, s32 y1, s32 x2, s32 y2)
Calculer la vraie distance entre 2 points. Beaucoup plus lent que PA_Distance.

– u16 **PA_AdjustAngle** (u16 angle, s16 anglerot, s32 startx, s32 starty, s32 targetx, s32 targety)
Ajuster un angle, par exemple pour calculer la direction ? prendre par un vaisseau.

– static u16 **PA_GetAngle** (s32 startx, s32 starty, s32 targetx, s32 targety)
R ?cup ?rer l'angle, de 0 ? 511, par rapport ? l'horizontale...

– int **PA_mulf32** (int a, int b)
(TODO)

– int **PA_divf32** (int a, int b)
  (TODO)

– int **PA_modf32** (int a, int b)
  (TODO)

– int **PA_sqrtf32** (int a)
  (TODO)

3.15.1 Description détaillée

Adjust angles, get random values...

3.15.2 Documentation des fonctions

3.15.2.1 void PA_SRand (s32 *r*) [inline, static]

Initialiser le random avec un seed. Ceci est pris de Ham, je n'ai aucun mérite. J'ai juste raccourci/accéléré un peu le tout.

Paramètres:

r Seed...

3.15.2.2 static inline u32 PA_RandMax (u32 *max*) [inline, static]

Donne un num ?ro al ?atoire... Entre 0 et le nombre donn ? (inclus).

Paramètres:

max Valeur maximale incluse

3.15.2.3 static inline u32 PA_RandMinMax (u32 *min*, u32 *max*) [inline, static]

Donne un num ?ro al ?atoire... Entre les 2 nombres donn ?s (inclus).

Paramètres:

min Valeur minimale incluse

max Valeur maximale incluse

3.15.2.4 static inline u32 PA_Distance (s32 *x1*, s32 *y1*, s32 *x2*, s32 *y2*) [inline, static]

Calculer la distance (au carr ?) entre 2 points.

Paramètres:

x1 Coordonn ?e X du premier point

y1 Coordonn ?e Y du premier point

x2 Coordonn ?e X du deuxi ?me point

y2 Coordonn ?e Y du deuxi ?me point

3.15.2.5 static inline u32 PA_TrueDistance (s32 *x1*, s32 *y1*, s32 *x2*, s32 *y2*) [inline, static]

Calculer la vraie distance entre 2 points. Beaucoup plus lent que PA_Distance.

Paramètres:

x1 Coordonn ?e X du premier point

y1 Coordonn ?e Y du premier point

x2 Coordonn ?e X du deuxi ?me point

y2 Coordonn ?e Y du deuxi ?me point

3.15.2.6 u16 PA_AdjustAngle (u16 *angle*, s16 *anglerot*, s32 *startx*, s32 *starty*, s32 *targetx*, s32 *targety*)

Ajuster un angle, par exemple pour calculer la direction ? prendre par un vaisseau.

Paramètres:

angle Angle de base, de 0 ? 511

anglerot De combien tourner...

startx Coordonn ?e X de base

starty Coordonn ?e Y de base

targetx Coordonn ?e X de la cible

targety Coordonn ?e Y de la cible

3.15.2.7 static inline u16 PA_GetAngle (s32 *startx*, s32 *starty*, s32 *targetx*, s32 *targety*) [inline, static]

R ?cup ?rer l'angle, de 0 ? 511, par rapport ? l'horizontale...

Paramètres:

startx Coordonn ?e X de base

starty Coordonn ?e Y de base

targetx Coordonn ?e X de la cible

targety Coordonn ?e Y de la cible

3.15.2.8 int PA_mulf32 (int *a*, int *b*)

(TODO)

Paramètres:

a Premier nombre

b Deuxième nombre

3.15.2.9 int PA_divf32 (int *a*, int *b*)

(TODO)
Paramètres:

- ***a*** Premier nombre
- ***b*** Deuxième nombre

3.15.2.10 int PA_modf32 (int *a*, int *b*)

(TODO)
Paramètres:

- ***a*** Premier nombre
- ***b*** Deuxième nombre

3.15.2.11 int PA_sqrtf32 (int *a*)

(TODO)
Paramètres:

- ***a*** Nombre

3.16 Microphone

Macros

– #define **PA_MicGetVol**() PA_Transfer->micvol
Renvoie le volume du micro.

– #define **PA_MicStopRecording**() PA_SendFifoCmd(PA_MSG_MICSTOP)
Arrête l'enregistrement.

Fonctions

– static void **PA_MicStartRecording** (u8 *buffer, u32 length)
Commencer à enregistrer avec le microphone.

– static void **PA_MicReplay** (u8 *buffer, s32 length)
Rejouer un son enregistrer.

3.16.1 Description détaillée

Record a sound and replay it...

3.16.2 Documentation des fonctions

3.16.2.1 static inline void PA_MicStartRecording (u8 * *Buffer*, u32 *Length*) [inline, static]

Commencer à enregistrer avec le microphone.
Paramètres:

Buffer Buffer dans lequel enregistrer le son

Length Longueur du buffer.

3.16.2.2 static inline void PA_MicReplay (u8 * *Buffer*, s32 *Length*) [inline, static]

Rejouer un son enregistrer.
Paramètres:

Buffer Buffer dans lequel on a enregistré le son

Length Longueur du buffer

3.17 Mode 7 commands

Fonctions

- void **PA_InitMode7** (u8 bg_select)
  Initialise le Mode 7 pour un fond donné. Vous devez etre en mode 1 ou 2 impérativement !
- static void **PA_DeInitMode7** ()
  DesInitialise le Mode 7.
- static void **PA_Mode7Angle** (s16 angle)
  Définir l'angle.
- static void **PA_Mode7MoveLeftRight** (s16 x_deplac)
  Se déplacer latéralement.
- static void **PA_Mode7MoveForwardBack** (s16 z_deplac)
  Se déplacer latéralement.
- static void **PA_Mode7X** (s16 mode7x)
  Se déplacer en un point donné de la carte.
- static void **PA_Mode7Z** (s16 mode7z)
  Se déplacer en un point donné de la carte.
- static void **PA_Mode7SetPointXZ** (s16 mode7x, s16 mode7z)
  Se déplacer en un point donné de la carte (de coordonnées x, z).
- static void **PA_Mode7Height** (s16 mode7y)
  Régler la hauteur de la caméra.

3.17.1 Description détaillée

Different commands for Mode 7 :p A big thanks to TONC for these...

3.17.2 Documentation des fonctions

3.17.2.1 void PA_InitMode7 (u8 *bg_select*)

Initialise le Mode 7 pour un fond donné. Vous devez etre en mode 1 ou 2 impérativement !

Paramètres:

bg_select Numéro du fond. 2 en mode 1, 2 ou 3 en mode 2

Exemples :

Backgrounds/Effects/Mode7/source/main.c.

3.17.2.2 static inline void PA_Mode7Angle (s16 *angle*) [inline, static]

Définir l'angle.

Paramètres:

angle L'angle, qui va de 0 à 511...

Exemples :

Backgrounds/Effects/Mode7/source/main.c.

3.17.2.3 static inline void PA_Mode7MoveLeftRight (s16 *x_deplac*) [inline, static]

Se déplacer latéralement.

Paramètres:

x_deplac De combien de pixels se déplacer à gauche ou à droite

Exemples :

Backgrounds/Effects/Mode7/source/main.c.

3.17.2.4 static inline void PA_Mode7MoveForwardBack (s16 *z_deplac*) [inline, static]

Se déplacer latéralement.

Paramètres:

z_deplac De combien se déplacer en avant ou en arrière

Exemples :

Backgrounds/Effects/Mode7/source/main.c.

3.17.2.5 static inline void PA_Mode7X (s16 *mode7x*) [inline, static]

Se déplacer en un point donné de la carte.

Paramètres:

mode7x Position X sur la carte

3.17.2.6 static inline void PA_Mode7Z (s16 *mode7z*) [inline, static]

Se déplacer en un point donné de la carte.

Paramètres:

mode7z Position Z sur la carte

3.17.2.7 static inline void PA_Mode7SetPointXZ (s16 *mode7x*, s16 *mode7z*) [inline, static]

Se déplacer en un point donné de la carte (de coordonnées x, z).

Paramètres:

mode7x Position X sur la carte

mode7z Position Z sur la carte

3.17.2.8 static inline void PA_Mode7Height (s16 *mode7y*) [inline, static]

Régler la hauteur de la caméra.

Paramètres:

mode7y Hauteur... Par défaut, elle est de 8192. On peut la mettre de 0 à 40 000 (ou beaucoup plus, mais après ca commence à faire petit...

Exemples :

Backgrounds/Effects/Mode7/source/main.c.

3.18 DS Motion functions

Fonctions

– static void **PA_MotionInit** (void)
Turn on the accelerometer.

– static u8 **PA_CheckDSMotion** ()
Checks whether a DS Motion Card is plugged in.

– static void **PA_MotionToPad** (u8 enable)
Maps the DS Motion Card to the Pad structure (! !).

Variables

– motion_struct **Motion**
Motion struct.

3.18.1 Description détaillée

Easy enable and play around with your DS Motion !

3.19 Palette system

Macros

– #define **PA_LoadPal**(palette, source)
*Charger une palette de 256 couleurs pour les fonds ou les sprites pour l'écran 0 ou 1. Ex : **PA_LoadPal(PALETTE_BG1, bg_pal)** (p. 85) ;.*

– #define **PA_LoadPal16**(palette, n_palette, source) DMA_Copy((void∗)source, (void∗)(palette + (n_palette << 5)), 16, DMA_16NOW)
*Charger une palette de 16 couleurs pour les fonds ou les sprites pour l'écran 0 ou 1. Ex : **PA_LoadPal16(PALETTE_BG1, 4, bg_pal)** (p. 85) ;.*

– #define **PA_LoadSprite16cPal**(screen, n_palette, palette) PA_LoadPal16((PAL_-SPRITE0+(0x400∗screen)), (n_palette), palette)
Charger une palette de 16 couleurs pour les sprites.

– #define **PA_RGB**(r, g, b) ((1<<15) | (r) | ((g)<<5) | ((b)<<10))
Convertir une couleurs au format Rouge, Vert, Bleu en un nombre utilisable par le système de palette. Attention : sur Gba, les valeurs vont de 0 à 31...

– #define **PA_SetBgPalCol**(screen, color_number, colorRGB) BG_-PALETTE[color_number + ((screen) << 9)] = colorRGB
Changer la couleur d'une des couleurs de la palette des fonds. Ne plus utiliser.

Fonctions

– static void **PA_Load8bitBgPal** (u8 screen, void ∗Pal)
Charger une palette pour le fond 8bit.

– void **PA_SetBrightness** (u8 screen, s8 bright)
Régler la luminosité de l'écran.

– static void **PA_SetPalNeg** (u32 palette)
Négativer une palette donnée. Pour annuler, il suffit de négativer à nouveau.

– static void **PA_SetPal16Neg** (u32 palette, u8 n_palette)
Négativer une palette de 16 couleurs donnée. Pour annuler, il suffit de négativer à nouveau.

– void **PA_InitSpriteExtPal** ()
Initialise le mode 16 palettes pour sprites de 256 couleurs... Effectué par défaut.

– void **PA_InitBgExtPal** ()
Initialise le mode 16 palettes pour fonds de 256 couleurs...

– static void **PA_LoadSpritePal** (u8 screen, u8 palette_number, void ∗palette)
Charger une palette de 256 couleurs pour les sprites.

– void **PA_LoadBgPalN** (u8 screen, u8 bg_number, u8 pal_number, void ∗palette)
Charger une palette de 256 couleurs dans les palettes des fonds, à un slot donné.

– static void **PA_LoadBgPal** (u8 screen, u16 bg_number, void ∗palette)

Charger une palette de 256 couleurs dans les palettes des fonds.

– void **PA_SetBgPalNCol** (u8 screen, u8 bg_number, u8 pal_number, u8 color_number, u16 color)
Changer la couleur d'une des couleurs d'une palette d'un fonds.

– static void **PA_SetBgColor** (u8 screen, u16 color)
Changer la couleur de fond d'un écran.

– void **PA_SetSpritePalCol** (u8 screen, u8 pal_number, u8 color_number, u16 color)
Changer la couleur de fond d'un écran.

– void **PA_3DSetSpritePalCol** (u8 pal_number, u8 color_number, u16 color)
Changes a color in a 3d sprite palette.

3.19.1 Description détaillée

Load palettes, change palette colors, set the gamma, etc...

3.19.2 Documentation des macros

3.19.2.1 #define PA_LoadPal(palette, source)

Valeur :

```
do{\
    DMA_Copy((void*)source, (void*)palette, 256, DMA_16NOW);\
    if (palette == PAL_SPRITE0) PA_LoadSpritePal(0, 0, (void*)source);\
    if (palette == PAL_SPRITE1) PA_LoadSpritePal(1, 0, (void*)source);\
    if (palette == PAL_BG0) {u8 itemp; for (itemp = 0; itemp < 4; itemp++)
      PA_LoadBgPal(0, itemp, (void*)(source));}\
    if (palette == PAL_BG1) {u8 itemp; for (itemp = 0; itemp < 4; itemp++)
      PA_LoadBgPal(1, itemp, (void*)(source));}}while(0)
```

Charger une palette de 256 couleurs pour les fonds ou les sprites pour l'écran 0 ou 1. Ex : **PA_LoadPal(PALETTE_BG1, bg_pal)** (p. 85) ;.

Paramètres:

palette Charger pour les Bg ou les Sprites, sur l'écran 0 ou 1 : PAL_BG0, PAL_SPRITE0, PAL_BG1, ou PAL_SPRITE1

source Nom de la palette (ex : master_Palette)

3.19.2.2 #define PA_LoadPal16(palette, n_palette, source) DMA_Copy((void∗)source, (void∗)(palette + (n_palette << 5)), 16, DMA_16NOW)

Charger une palette de 16 couleurs pour les fonds ou les sprites pour l'écran 0 ou 1. Ex : **PA_LoadPal16(PALETTE_BG1, 4, bg_pal)** (p. 85) ;.

Paramètres:

palette Charger pour les Bg ou les Sprites, sur l'écran 0 ou 1 : PAL_BG0, PAL_-SPRITE0, PAL_BG1, ou PAL_SPRITE1

n_palette Numéro de la palette de 16 couleurs que l'on veut charger (0-15)

source Nom de la palette (ex : master_Palette)

3.19.2.3 #define PA_LoadSprite16cPal(screen, n_palette, palette) PA_LoadPal16((PAL_SPRITE0+(0x400∗screen)), (n_palette), palette)

Charger une palette de 16 couleurs pour les sprites.
Paramètres:

screen Ecran (0-1)

n_palette Numéro de la palette de 16 couleurs que l'on veut charger (0-15)

palette Nom de la palette (ex : Sprite_Pal)

3.19.2.4 #define PA_RGB(r, g, b) ((1<<15) | (r) | ((g)<<5) | ((b)<<10))

Convertir une couleurs au format Rouge, Vert, Bleu en un nombre utilisable par le système de palette. Attention : sur Gba, les valeurs vont de 0 à 31...
Paramètres:

r Rouge (0-31)

g Vert (0-31)

b Bleu (0-31)

3.19.2.5 #define PA_SetBgPalCol(screen, color_number, colorRGB) BG_PALETTE[color_number + ((screen) << 9)] = colorRGB

Changer la couleur d'une des couleurs de la palette des fonds. Ne plus utiliser.
Paramètres:

screen Ecran...

color_number Numéro de la couleur dans la palette (0-255)

colorRGB Valeur RGB, comme **PA_RGB(31, 31, 31)** (p. 86) pour blanc

3.19.3 Documentation des fonctions

3.19.3.1 static inline void PA_Load8bitBgPal (u8 *screen*, void ∗ *Pal*) [inline, static]

Charger une palette pour le fond 8bit.
Paramètres:

screen Ecran...

Pal Nom de la palette (ex : master_Palette)

3.19.3.2 void PA_SetBrightness (u8 *screen*, s8 *bright*)

Régler la luminosité de l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

bright Luminosité, de -32 à 32, 0 étant neutre

3.19.3.3 static inline void PA_SetPalNeg (u32 *palette*) `[inline, static]`

Négativer une palette donnée. Pour annuler, il suffit de négativer à nouveau.

Paramètres:

palette Charger pour les Bg ou les Sprites, sur l'écran 0 ou 1 : PAL_BG0, PAL_SPRITE0, PAL_BG1, ou PAL_SPRITE1

3.19.3.4 static inline void PA_SetPal16Neg (u32 *palette*, u8 *n_palette*) `[inline, static]`

Négativer une palette de 16 couleurs donnée. Pour annuler, il suffit de négativer à nouveau.

Paramètres:

palette Charger pour les Bg ou les Sprites, sur l'écran 0 ou 1 : PAL_BG0, PAL_SPRITE0, PAL_BG1, ou PAL_SPRITE1

n_palette Numéro de la palette de 16 couleurs (0-15)

3.19.3.5 void PA_LoadSpritePal (u8 *screen*, u8 *palette_number*, void * *palette*) `[inline, static]`

Charger une palette de 256 couleurs pour les sprites.

Paramètres:

screen Ecran...

palette_number Numéro de la palette (0-15)

palette Nom de la palette à charger ((void*)nom_palette)

3.19.3.6 void PA_LoadBgPalN (u8 *screen*, u8 *bg_number*, u8 *pal_number*, void * *palette*)

Charger une palette de 256 couleurs dans les palettes des fonds, à un slot donné. Charger une palette de 256 couleurs dans les palettes des fonds.

Paramètres:

screen Ecran...

bg_number Numéro du fond (0-3)

pal_number Numéro de palette

palette Nom de la palette à charger ((void∗)nom_palette)

screen Ecran...

bg_number Numéro du fond (0-3)

pal_number Numéro de la palette (0-15)

palette Nom de la palette à charger ((void∗)nom_palette)

3.19.3.7 void PA_LoadBgPal (u8 *screen*, u16 *bg_number*, void ∗ *palette*) [inline, static]

Charger une palette de 256 couleurs dans les palettes des fonds.
Paramètres:

screen Ecran...

bg_number Numéro du fond (0-3)

palette Nom de la palette à charger ((void∗)nom_palette)

3.19.3.8 void PA_SetBgPalNCol (u8 *screen*, u8 *bg_number*, u8 *pal_number*, u8 *color_number*, u16 *color*)

Changer la couleur d'une des couleurs d'une palette d'un fonds.
Paramètres:

screen Ecran...

bg_number Numéro du fond (0-3)

pal_number Numéro de palette (0-15), laisser à 0 si pas sur...

color_number Numéro de la couleur dans la palette (0-255)

color Valeur RGB, comme **PA_RGB(31, 31, 31)** (p. 86) pour blanc

3.19.3.9 static inline void PA_SetBgColor (u8 *screen*, u16 *color*) [inline, static]

Changer la couleur de fond d'un écran.
Paramètres:

screen Ecran...

color Valeur RGB, comme **PA_RGB(31, 31, 31)** (p. 86) pour blanc

3.19.3.10 void PA_SetSpritePalCol (u8 *screen*, u8 *pal_number*, u8 *color_number*, u16 *color*)

Changer la couleur de fond d'un écran.
Paramètres:

screen Ecran...

pal_number Numéro de la palette

color_number Numéro de la couleur

color Couleur (venant de PA_RGB...)

3.19.3.11 void PA_3DSetSpritePalCol (u8 *pal_number*, u8 *color_number*, u16 *color*)

Changes a color in a 3d sprite palette.

Paramètres:

pal_number Palette number

color_number Color number in the palette

color Color (given by PA_RGB...)

3.20 Palette system for Dual Screen

Macros

– #define **PA_DualLoadPal**(palette, source)
Charger une palette de 256 couleurs pour les fonds ou les sprites pour les 2 écrans.

– #define **PA_DualLoadPal16**(palette, n_palette, source)
Charger une palette de 16 couleurs pour les fonds ou les sprites pour les deux écrans.

Fonctions

– static void **PA_DualSetPalNeg** (u32 palette)
Négativer une palette donnée. Pour annuler, il suffit de négativer à nouveau.

– static void **PA_DualSetPal16Neg** (u32 palette, u8 n_palette)
Négativer une palette de 16 couleurs donnée. Pour annuler, il suffit de négativer à nouveau.

– static void **PA_DualLoadSpritePal** (u8 palette_number, void ∗palette)
Charger une palette de 256 couleurs dans les palettes des sprites.

– static void **PA_DualLoadBgPal** (u8 bg_number, void ∗palette)
Charger une palette de 256 couleurs pour un fond.

– static void **PA_DualSetBgColor** (u16 color)
Changer la couleur de fond des 2 écrans.

3.20.1 Description détaillée

Load palettes, change palette colors, set the gamma, etc... on both screens !

3.20.2 Documentation des macros

3.20.2.1 #define PA_DualLoadPal(palette, source)

Valeur :

```
do{\
    DMA_Copy((void*)source, (void*)palette, 256, DMA_16NOW);\
    DMA_Copy((void*)(source+1024), (void*)palette, 256, DMA_16NOW);\
    if(palette == PAL_SPRITE)\
        PA_DualLoadSpriteExtPal(0, (void*)palette);\
}while(0)
```

Charger une palette de 256 couleurs pour les fonds ou les sprites pour les 2 écrans.
Paramètres:

palette Charger pour les Bg ou les Sprites : PAL_BG ou PAL_SPRITE

source Nom de la palette (ex : master_Palette)

3.20.2.2 #define PA_DualLoadPal16(palette, n_palette, source)

Valeur :

```
do{\
    DMA_Copy((void*)source, (void*)(palette + (n_palette << 5)), 16, DMA_16NOW);\

    DMA_Copy((void*)source, (void*)(palette + 1024 + (n_palette << 5)), 16, DMA_1
      6NOW);}while(0)
```

Charger une palette de 16 couleurs pour les fonds ou les sprites pour les deux écrans.

Paramètres:

palette Charger pour les Bg ou les Sprites : PAL_BG ou PAL_SPRITE

n_palette Numéro de la palette de 16 couleurs que l'on veut charger (0-15)

source Nom de la palette (ex : master_Palette)

3.20.3 Documentation des fonctions

3.20.3.1 static inline void PA_DualSetPalNeg (u32 *palette*) [inline, static]

Négativer une palette donnée. Pour annuler, il suffit de négativer à nouveau.

Paramètres:

palette Charger pour les Bg ou les Sprites : PAL_BG, PAL_SPRITE

3.20.3.2 static inline void PA_DualSetPal16Neg (u32 *palette*, u8 *n_palette*) [inline, static]

Négativer une palette de 16 couleurs donnée. Pour annuler, il suffit de négativer à nouveau.

Paramètres:

palette Charger pour les Bg ou les Sprites : PAL_BG, PAL_SPRITE

n_palette Numéro de la palette de 16 couleurs (0-15)

3.20.3.3 static inline void PA_DualLoadSpritePal (u8 *palette_number*, void * *palette*) [inline, static]

Charger une palette de 256 couleurs dans les palettes des sprites.

Paramètres:

palette_number Numéro de la palette (0-15)

palette Nom de la palette à charger ((void*)nom_palette)

3.20.3.4 static inline void PA_DualLoadBgPal (u8 *bg_number*, void * *palette*) [inline, static]

Charger une palette de 256 couleurs pour un fond.

Paramètres:

bg_number Numéro du fond (0-3)

palette Nom de la palette à charger ((void*)nom_palette)

3.20.3.5 static inline void PA_DualSetBgColor (u16 *color*) [inline, static]

Changer la couleur de fond des 2 écrans.

Paramètres:

color Valeur RGB, comme **PA_RGB(31, 31, 31)** (p. 86) pour blanc

3.21 Shape Recognition

Fonctions

– char **PA_CheckLetter** ()
Analyse la forme et renvoie une lettre correspondante. 0 si rien. La chaine représentative de la forme est copiée dans PA_RecoShape au Stylus Release. Copie des images dispos : `http://www.palib.info/Reco/PAGraffiti.gif`.

– static void **PA_RecoAddShape** (char letter, char ∗shape)
Ajouter une nouvelle forme au système de reconnaissance.

– static void **PA_ResetRecoSys** ()
Réinitialise le systeme de reconnaissance.

– static void **PA_UsePAGraffiti** (u8 use)
*Activer ou désactiver les lettres **PA** (p. 161) Graffiti. On voudra le désactiver quand on veut utiliser uniquement ses propres formes.*

3.21.1 Description détaillée

Draw a shape and have it recognized !

3.21.2 Documentation des fonctions

3.21.2.1 static inline void PA_RecoAddShape (char *letter*, char ∗ *shape*) [inline, static]

Ajouter une nouvelle forme au système de reconnaissance.

Paramètres:

letter Lettre renvoyée par le système de reconnaissance pour cette forme (peut être n'imorte quelle lettre, ou un nombre de 1 à 255)

shape Chaine de 15 caractères fournie par le systeme de reconnaissance in PA_-RecoShape

3.21.2.2 static inline void PA_UsePAGraffiti (u8 *use*) [inline, static]

Activer ou désactiver les lettres **PA** (p. 161) Graffiti. On voudra le désactiver quand on veut utiliser uniquement ses propres formes.

Paramètres:

use 1/0, on/off...

3.22 Special Effects

Macros

– #define **PA_EnableBgMosaic**(screen, bg) _REG16(REG_BGCNT(screen, bg)) |= (1 << 6)
Activer l'effet de mosaic pour un fond donné.

– #define **PA_DisableBgMosaic**(screen, bg) _REG16(REG_BGCNT(screen, bg)) &= ∼(1 << 6)
Désactiver l'effet de mosaic pour un fond donné.

– #define **PA_SetBgMosaicXY**(screen, h_size, v_size) do{PA_REG_MOSAIC(screen) &= 255; PA_REG_MOSAIC(screen) |= ((h_size) + ((v_size) << 4));}while(0)
Régler les paramètres de la mosaic pour les fonds.

– #define **PA_SetSpriteMosaicXY**(screen, h_size, v_size) do{PA_REG_MOSAIC(screen) &= (255 << 8); PA_REG_MOSAIC(screen) |= (((h_size) << 8) + ((v_size) << 12));}while(0)
Régler les paramètres de la mosaic pour les sprites.

– #define **PA_EnableSpecialFx**(screen, EffectType, FirstTarget, SecondTarget) PA_REG_BLDCNT(screen) = ((FirstTarget) + ((SecondTarget) << 8) + ((EffectType) << 6))
Activer les Effets Speciaux et choisir si les fonds et sprites l'utiliseront ou pas. On choisit aussi au passage quel Effet utiliser.

– #define **PA_DisableSpecialFx**(screen) PA_REG_BLDCNT(screen) = 0
Désactiver les Effets Speciaux.

– #define **PA_SetSFXAlpha**(screen, Coeff1, Coeff2) PA_REG_BLDALPHA(screen) = (Coeff1) + ((Coeff2) << 8)
Régler les paramètres pour l'Alpha-Blending.

3.22.1 Description détaillée

Set the sprite special effects (alpha-blending, luminosity, mosaic effects...)

3.22.2 Documentation des macros

3.22.2.1 #define PA_EnableBgMosaic(screen, bg) _REG16(REG_BGCNT(screen, bg)) |= (1 << 6)

Activer l'effet de mosaic pour un fond donné.

Paramètres:

screen Ecran du bg (0 ou 1)

bg Numéro du fond

3.22.2.2 #define PA_DisableBgMosaic(screen, bg) _-REG16(REG_BGCNT(screen, bg)) &= ∼(1 << 6)

Désactiver l'effet de mosaic pour un fond donné.

Paramètres:

screen Ecran du bg (0 ou 1)

bg Numéro du fond

3.22.2.3 #define PA_SetBgMosaicXY(screen, h_size, v_size) do{PA_REG_-MOSAIC(screen) &= 255 ; PA_REG_MOSAIC(screen) |= ((h_size) + ((v_size) << 4)) ;}while(0)

Régler les paramètres de la mosaic pour les fonds.

Paramètres:

screen Ecran...

h_size Taille horizontale de la mosaic (1 pour 1 pixel, 2 pour 2 pixels, etc...)

v_size Taille verticale de la mosaic (1 pour 1 pixel, 2 pour 2 pixels, etc...)

3.22.2.4 #define PA_SetSpriteMosaicXY(screen, h_size, v_size) do{PA_REG_MOSAIC(screen) &= (255 << 8) ; PA_REG_MOSAIC(screen) |= (((h_size) << 8) + ((v_size) << 12)) ;}while(0)

Régler les paramètres de la mosaic pour les sprites.

Paramètres:

screen Ecran...

h_size Taille horizontale de la mosaic (1 pour 1 pixel, 2 pour 2 pixels, etc...)

v_size Taille verticale de la mosaic (1 pour 1 pixel, 2 pour 2 pixels, etc...)

3.22.2.5 #define PA_EnableSpecialFx(screen, EffectType, FirstTarget, SecondTarget) PA_REG_BLDCNT(screen) = ((FirstTarget) + ((SecondTarget) << 8) + ((EffectType) << 6))

Activer les Effets Speciaux et choisir si les fonds et sprites l'utiliseront ou pas. On choisit aussi au passage quel Effet utiliser.

Paramètres:

screen Ecran...

EffectType Type d'effet. 0 pour aucun, 1 pour transparence, 2 pour augmentation de la luminosité, et 3 pour diminution de celle-ci... On peut utiliser les macors SFX_NONE, SFX_ALPHA, SFX_BRIGHTINC, SFX_BRIGHTDEC

FirstTarget Fond et sprites à afficher aevc l'effet spécial, que l'on choisi de la facon suivante : SFX_BG0 | SFX_BG1 | SFX_BG2 | SFX_BG3 | SFX_OBJ | SFX_BD (back drop)

SecondTarget Fond et sprites à afficher derrière la transparence, que l'on choisi de la facon suivante : SFX_BG0 | SFX_BG1 | SFX_BG2 | SFX_BG3 | SFX_-OBJ | SFX_BD (back drop)

3.22.2.6 #define PA_DisableSpecialFx(screen) PA_REG_BLDCNT(screen) = 0

Désactiver les Effets Speciaux.

Paramètres:

screen Ecran...

3.22.2.7 #define PA_SetSFXAlpha(screen, Coeff1, Coeff2) PA_-REG_BLDALPHA(screen) = (Coeff1) + ((Coeff2) << 8)

Régler les paramètres pour l'Alpha-Blending.

Paramètres:

screen Ecran...

Coeff1 Coefficient pour la première couche, de 0 à 31. A priori vaut mieux le mettre entre 0 et 16

Coeff2 Coefficient pour la deuxième couche, de 0 à 31. A priori vaut mieux le mettre entre 0 et 16

3.23 Sprite system

Macros

– #define **PA_UpdateOAM0**() DMA_Copy((void∗)PA_obj, (void∗)OAM0, 256, DMA_32NOW)

Mettre à jour les infos des sprites pour l'écran 0 uniquement. A faire dans le VBL.

– #define **PA_UpdateOAM1**() DMA_Copy((void∗)PA_obj + 256, (void∗)OAM1, 256, DMA_32NOW)

Mettre à jour les infos des sprites pour l'écran 1 uniquement. A faire dans le VBL.

– #define **PA_UpdateSpriteGfx**(screen, obj_number, obj_data) PA_UpdateGfx(screen, PA_GetSpriteGfx(screen, obj_number), obj_data)

Mettre à jour les Gfx d'un sprite donné.

– #define **PA_SetSpriteRotEnable**(screen, sprite, rotset) do{PA_obj[screen][sprite].atr0 |= OBJ_ROT; PA_obj[screen][sprite].atr1 = (PA_obj[screen][sprite].atr1 & ALL_BUT_ROTSET) + ((rotset) << 9);}while(0)

Faire tourner et zoomer un sprite.

– #define **PA_SetSpriteRotDisable**(screen, sprite) do{PA_obj[screen][sprite].atr0 &= ALL_BUT(OBJ_ROT); PA_obj[screen][sprite].atr1 &= ALL_BUT_ROTSET;}while(0)

Arreter de faire tourner et zoomer un sprite.

– #define **PA_SetSpriteX**(screen, obj, x) PA_obj[screen][obj].atr1 = (PA_obj[screen][obj].atr1 & ALL_BUT(PA_OBJ_X)) + ((x) & PA_OBJ_X)

Position X du sprite à l'écran.

– #define **PA_GetSpriteX**(screen, obj) (PA_obj[screen][obj].atr1 & (PA_OBJ_X))

Position X du sprite à l'écran.

– #define **PA_SetSpriteY**(screen, obj, y) PA_obj[screen][obj].atr0 = (PA_obj[screen][obj].atr0 & ALL_BUT(PA_OBJ_Y)) + ((y) & PA_OBJ_Y)

Position Y du sprite à l'écran.

– #define **PA_GetSpriteY**(screen, obj) (PA_obj[screen][obj].atr0 & PA_OBJ_Y)

Position Y du sprite à l'écran.

– #define **PA_SetSpritePal**(screen, obj, pal) PA_obj[screen][obj].atr2 = (PA_obj[screen][obj].atr2 & ALL_BUT_PAL) + ((pal) << 12)

Changer la palette d'un sprite.

– #define **PA_GetSpritePal**(screen, obj) (PA_obj[screen][obj].atr2 >> 12)

Palette d'un sprite.

– #define **PA_SetSpriteDblsize**(screen, obj, dblsize) PA_obj[screen][obj].atr0 = (PA_obj[screen][obj].atr0 & ALL_BUT(DBLSIZE)) + ((dblsize) << 9)

Activer ou désactiver le mode Doublesize pour un sprite.

– #define **PA_GetSpriteDblsize**(screen, obj) ((PA_obj[screen][obj].atr0 & DBLSIZE) >> 9)

Etat du mode Doublesize pour un sprite.

– #define **PA_SetSpriteColors**(screen, sprite, n_colors) PA_obj[screen][sprite].atr0 = (PA_obj[screen][sprite].atr0 & ALL_BUT(N_COLORS)) + ((n_colors) << 13)
Changer le mode de couleur du sprite.

– #define **PA_GetSpriteColors**(screen, sprite) ((PA_obj[screen][sprite].atr0 & N_COLORS) >> 13)
Mode de couleur d'un sprite.

– #define **PA_SetSpriteMode**(screen, sprite, obj_mode) PA_obj[screen][sprite].atr0 = (PA_obj[screen][sprite].atr0 & ALL_BUT(OBJ_MODE)) + ((obj_mode) << 10)
Régler le mode d'un sprite : 0 pour normal, 1 pour transparent, 2 pour fenetre.

– #define **PA_GetSpriteMode**(screen, obj) ((PA_obj[screen][obj].atr0 & OBJ_MODE) >> 10)
Mode d'un sprite : 0 pour normal, 1 pour transparent, 2 pour fenetre.

– #define **PA_SetSpriteMosaic**(screen, obj, mosaic) PA_obj[screen][obj].atr0 = (PA_obj[screen][obj].atr0 & ALL_BUT(OBJ_MOSAIC)) + ((mosaic) << 12)
Mettre ou non un sprite en mode mosaic.

– #define **PA_GetSpriteMosaic**(screen, obj) ((PA_obj[screen][obj].atr0 & OBJ_MOSAIC) >> 12)
Si un sprite est en mode mosaic.

– #define **PA_SetSpriteHflip**(screen, obj, hflip) PA_obj[screen][obj].atr1 = (PA_obj[screen][obj].atr1 & ALL_BUT(OBJ_HFLIP)) + ((hflip) << 12)
Utiliser ou non le flip horizontal pour un sprite.

– #define **PA_GetSpriteHflip**(screen, obj) ((PA_obj[screen][obj].atr1 & OBJ_HFLIP) >> 12)
S'il y a un flip horizontal pour un sprite.

– #define **PA_SetSpriteVflip**(screen, obj, vflip) PA_obj[screen][obj].atr1 = (PA_obj[screen][obj].atr1 & ALL_BUT(OBJ_VFLIP)) + ((vflip) << 13)
Utiliser ou non le flip vertical pour un sprite.

– #define **PA_GetSpriteVflip**(screen, obj) ((PA_obj[screen][obj].atr1 & OBJ_VFLIP) >> 13)
Si le flip vertical est utilisé ou non pour un sprite.

– #define **PA_SetSpriteGfx**(screen, obj, gfx) PA_obj[screen][obj].atr2 = (PA_obj[screen][obj].atr2 & ALL_BUT(OBJ_GFX)) + ((gfx) & OBJ_GFX)
Modifier les graphismes utilisés par un sprite.

– #define **PA_GetSpriteGfx**(screen, obj) (PA_obj[screen][obj].atr2 & OBJ_GFX)
Récupérer le gfx utilisés par un sprite.

– #define **PA_SetSpritePrio**(screen, obj, prio) PA_obj[screen][obj].atr2 = (PA_obj[screen][obj].atr2 & ALL_BUT(OBJ_PRIO)) + ((prio) << 10)
Régler la priorité d'un sprite par rapport au Bg.

– #define **PA_GetSpritePrio**(screen, obj) ((PA_obj[screen][obj].atr2 & OBJ_PRIO) >> 10)
Récupérer la priorité d'un sprite par rapport au Bg.

– #define **PA_GetSpriteLx**(screen, sprite) PA_size[PA_obj[screen][sprite].atr0 >> 14][PA_obj[screen][sprite].atr1 >> 14].lx
Récupérer la largeur d'un sprite.

– #define **PA_GetSpriteLy**(screen, sprite) PA_size[PA_obj[screen][sprite].atr0 >> 14][PA_obj[screen][sprite].atr1 >> 14].ly
Récupérer la hauteur d'un sprite.

– #define **PA_CloneSprite**(screen, obj, target) do{PA_obj[screen][obj].atr0 = PA_obj[screen][target].atr0 ; PA_obj[screen][obj].atr1 = PA_obj[screen][target].atr1 ; PA_obj[screen][obj].atr2 = PA_obj[screen][target].atr2 ; ++obj_per_gfx[screen][PA_GetSpriteGfx(screen, target)] ;}while(0)
Cloner un sprite. Marche uniquement pour les sprites sur un meme écran.

Fonctions

– void **PA_UpdateOAM** (void)
Mettre à jour les infos des sprites pour les 2 écrans. A faire dans le VBL.

– u16 **PA_CreateGfx** (u8 screen, void ∗obj_data, u8 obj_shape, u8 obj_size, u8 color_mode)
Charger en mémoire un gfx à utiliser plus tard pour un sprite. Renvoie le numéro en mémoire.

– void **PA_ResetSpriteSys** (void)
Remise à 0 du système de sprite, de la mémoire...

– static void **PA_CreateSprite** (u8 screen, u8 obj_number, void ∗obj_data, u8 obj_shape, u8 obj_size, u8 color_mode, u8 palette, s16 x, s16 y)
Creer un sprite avec ses gfx... Ceci est la version simple de la fonction.

– static void **PA_CreateSpriteEx** (u8 screen, u8 obj_number, void ∗obj_data, u8 obj_shape, u8 obj_size, u8 color_mode, u8 palette, u8 obj_mode, u8 mosaic, u8 hflip, u8 vflip, u8 prio, u8 dblsize, s16 x, s16 y)
Creer un sprite avec ses gfx... Ceci est la version complexe de la fonction.

– static void **PA_Create16bitSpriteEx** (u8 screen, u8 obj_number, void ∗obj_data, u8 obj_shape, u8 obj_size, u8 mosaic, u8 hflip, u8 vflip, u8 prio, u8 dblsize, s16 x, s16 y)
Creer un sprite de 16 bits avec ses gfx... Ceci est la version complexe de la fonction. Attention : un sprite de 16 bits DOIT etre large de 128 pixels, meme si ce sprite ne prend qu'une petite partie sur la gauche.

– static void **PA_Create16bitSpriteFromGfx** (u8 screen, u8 obj_number, u16 gfx, u8 obj_shape, u8 obj_size, s16 x, s16 y)
Creer un sprite de 16 bits à partir de gfx...

– static void **PA_Create16bitSprite** (u8 screen, u8 obj_number, void ∗obj_data, u8 obj_shape, u8 obj_size, s16 x, s16 y)
Creer un sprite de 16 bits avec ses gfx... Ceci est la version simple de la fonction. Attention : un sprite de 16 bits DOIT etre large de 128 pixels, meme si ce sprite ne prend qu'une petite partie sur la gauche.

– static void **PA_CreateSpriteFromGfx** (u8 screen, u8 obj_number, u16 obj_gfx, u8 obj_shape, u8 obj_size, u8 color_mode, u8 palette, s16 x, s16 y)
Creer un sprite avec ses gfx... Ceci est la version simple de la fonction.

– static void **PA_CreateSpriteExFromGfx** (u8 screen, u8 obj_number, u16 obj_gfx, u8 obj_shape, u8 obj_size, u8 color_mode, u8 palette, u8 obj_mode, u8 mosaic, u8 hflip, u8 vflip, u8 prio, u8 dblsize, s16 x, s16 y)
Creer un sprite avec ses gfx... Ceci est la version complexe de la fonction.

– static void **PA_UpdateGfx** (u8 screen, u16 gfx_number, void ∗obj_data)
Mettre à jour les Gfx donnés.

– static void **PA_UpdateGfxAndMem** (u8 screen, u8 gfx_number, void ∗obj_data)
Mettre à jour les Gfx donnés et le pointer d'animation dans PAlib... Uniquement pour utilisateurs avertis.

– void **PA_DeleteGfx** (u8 screen, u16 obj_gfx)
Effacer un Gfx. Si un sprite l'utilisait, il deviendra invisible...

– void **PA_DeleteSprite** (u8 screen, u8 obj_number)
Effacer un sprite. S'il était le seul à utiliser un gfx, il sera effacé lui aussi.

– static void **PA_SetRotset** (u8 screen, u8 rotset, s16 angle, u16 zoomx, u16 zoomy)
Faire tourner et zoomer un sprite.

– static void **PA_SetRotsetNoZoom** (u8 screen, u8 rotset, s16 angle)
Faire tourner un sprite sans zoomer. C'est un peu plus rapide que la fonction PA_-SetRotset.

– static void **PA_SetRotsetNoAngle** (u8 screen, u8 rotset, u16 zoomx, u16 zoomy)
Zoomer un sprite sans le faire tourner. C'est un peu plus rapide que la fonction PA_-SetRotset.

– static void **PA_SetSpriteXY** (u8 screen, u8 sprite, s16 x, s16 y)
Position X et Y du sprite à l'écran.

– static void **PA_Set16bitSpriteAlpha** (u8 screen, u8 sprite, u8 alpha)
Position X du sprite à l'écran.

– static void **PA_SetSpriteAnimEx** (u8 screen, u8 sprite, u8 lx, u8 ly, u8 ncolors, s16 animframe)
Régler l'image du sprite dans l'animation. Cette fonction est plus rapide que PA_-SetSpriteAnim parce qu'elle n'a pas à rechercher les dimensions du sprite.

– static void **PA_SetSpriteAnim** (u8 screen, u8 sprite, s16 animframe)
Régler l'image du sprite dans l'animation. Identique à PA_SetSpriteAnimEx, mais plus simple à utiliser, par contre plus lent.

– void **PA_StartSpriteAnimEx** (u8 screen, u8 sprite, s16 firstframe, s16 lastframe, s16 speed, u8 type, s16 ncycles)
Démarre une animation de sprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

– static void **PA_StartSpriteAnim** (u8 screen, u8 sprite, s16 firstframe, s16 lastframe, s16 speed)

Démarre une animation de sprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

– static void **PA_StopSpriteAnim** (u8 screen, u8 sprite)
Arrêter une animation de sprite.

– static void **PA_SetSpriteAnimFrame** (u8 screen, u8 sprite, u16 frame)
Changer le numéro actuel de la frame d'animation.

– static u16 **PA_GetSpriteAnimFrame** (u8 screen, u8 sprite)
Renvoie le numéro actuel de la frame d'animation.

– static void **PA_SetSpriteAnimSpeed** (u8 screen, u8 sprite, s16 speed)
Changer la vitesse de l'animation.

– static u16 **PA_GetSpriteAnimSpeed** (u8 screen, u8 sprite)
Renvoie la vitesse de l'animation.

– static void **PA_SetSpriteNCycles** (u8 screen, u8 sprite, s32 NCycles)
Changer le nombre de cycles d'animation restant (-1 pour inifini).

– static s32 **PA_GetSpriteNCycles** (u8 screen, u8 sprite)
Renvoie le nombre de cycles d'animation restants.

– static void **PA_SpriteAnimPause** (u8 screen, u8 sprite, u8 pause)
Mettre en Pause en remettre en lecture une animation de sprite.

– static void **PA_SetSpritePixel** (u8 screen, u8 sprite, u8 x, u8 y, u8 color)
Mettre un pixel d'un sprite à une couleur donnée. Comme PA_SetSpritePixelEx, avec moins d'options, mais un peu plus lent.

– static u8 **PA_GetSpritePixel** (u8 screen, u8 sprite, u8 x, u8 y)
Récupérer la couleur d'un pixel d'un sprite. Comme PA_GetSpritePixelEx, avec moins d'options, mais un peu plus lent.

– static u8 **PA_GetSprite16cPixel** (u8 screen, u8 sprite, u8 x, u8 y)
Récupérer la couleur d'un pixel d'un sprite de 16 couleurs.

– void **PA_InitSpriteDraw** (u8 screen, u8 sprite)
Initialise un sprite pour pouvoir dessiner dessus !

– static void **PA_InitAllSpriteDraw** (void)

Initialise tous les sprites à l'écran pour dessiner dessus.

– void **PA_InitSpriteExtPrio** (u8 SpritePrio)
Activer le systeme de priorité de sprites PAlib. Plus lent que le systeme normal, il permet d'avoir 256 niveaux de priorité (supplante la priorité par numéro de sprites).

3.23.1 Description détaillée

Load Sprite, move them around, rotate them...

3.23.2 Documentation des macros

3.23.2.1 #define PA_UpdateSpriteGfx(screen, obj_number, obj_data) PA_UpdateGfx(screen, PA_GetSpriteGfx(screen, obj_number), obj_data)

Mettre à jour les Gfx d'un sprite donné.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj_number Numéro de l'objet dans le systeme de sprite

obj_data Graphisme à charger

3.23.2.2 #define PA_SetSpriteRotEnable(screen, sprite, rotset) do{PA_obj[screen][sprite].atr0 |= OBJ_ROT ; PA_obj[screen][sprite].atr1 = (PA_obj[screen][sprite].atr1 & ALL_BUT_ROTSET) + ((rotset) << 9) ;}while(0)

Faire tourner et zoomer un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Sprite que l'on veut faire tourner

rotset Rotset que l'on veut pour un sprite donné (0-31). On peut a priori utiliser un rotset pour plusieurs sprites, s'ils sont zoomés/tournés pareil...

3.23.2.3 #define PA_SetSpriteRotDisable(screen, sprite) do{PA_obj[screen][sprite].atr0 &= ALL_BUT(OBJ_ROT) ; PA_obj[screen][sprite].atr1 &= ALL_BUT_ROTSET ;}while(0)

Arreter de faire tourner et zoomer un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Sprite que l'on veut faire tourner

3.23.2.4 #define PA_SetSpriteX(screen, obj, x) PA_obj[screen][obj].atr1 = (PA_obj[screen][obj].atr1 & ALL_BUT(PA_OBJ_X)) + ((x) & PA_OBJ_X)

Position X du sprite à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

x Position X

3.23.2.5 #define PA_GetSpriteX(screen, obj) (PA_obj[screen][obj].atr1 & (PA_OBJ_X))

Position X du sprite à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.6 #define PA_SetSpriteY(screen, obj, y) PA_obj[screen][obj].atr0 = (PA_obj[screen][obj].atr0 & ALL_BUT(PA_OBJ_Y)) + ((y) & PA_OBJ_Y)

Position Y du sprite à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

y Position Y

3.23.2.7 #define PA_GetSpriteY(screen, obj) (PA_obj[screen][obj].atr0 & PA_OBJ_Y)

Position Y du sprite à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.8 #define PA_SetSpritePal(screen, obj, pal) PA_obj[screen][obj].atr2 = (PA_obj[screen][obj].atr2 & ALL_BUT_PAL) + ((pal) << 12)

Changer la palette d'un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

pal Numéro de la palette (de 0 à 15)

3.23.2.9 #define PA_GetSpritePal(screen, obj) (PA_obj[screen][obj].atr2 >> 12)

Palette d'un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.10 #define PA_SetSpriteDblsize(screen, obj, dblsize) PA_-obj[screen][obj].atr0 = (PA_obj[screen][obj].atr0 & ALL_BUT(DBLSIZE)) + ((dblsize) << 9)

Activer ou désactiver le mode Doublesize pour un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

dblsize 1 pour l'activer, 0 pour l'inactiver

3.23.2.11 #define PA_GetSpriteDblsize(screen, obj) ((PA_obj[screen][obj].atr0 & DBLSIZE) >> 9)

Etat du mode Doublesize pour un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.12 #define PA_SetSpriteColors(screen, sprite, n_colors) PA_-obj[screen][sprite].atr0 = (PA_obj[screen][sprite].atr0 & ALL_BUT(N_COLORS)) + ((n_colors) << 13)

Changer le mode de couleur du sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro de l'objet dans le systeme de sprite

n_colors 0 pour 16 couleurs, 1 pour 256

3.23.2.13 #define PA_GetSpriteColors(screen, sprite) ((PA_-obj[screen][sprite].atr0 & N_COLORS) >> 13)

Mode de couleur d'un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro de l'objet dans le systeme de sprite

3.23.2.14 #define PA_SetSpriteMode(screen, sprite, obj_mode) PA_-obj[screen][sprite].atr0 = (PA_obj[screen][sprite].atr0 & ALL_BUT(OBJ_MODE)) + ((obj_mode) << 10)

Régler le mode d'un sprite : 0 pour normal, 1 pour transparent, 2 pour fenetre.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro de l'objet dans le systeme de sprite

obj_mode Mode : 0 pour normal, 1 pour transparent, 2 pour fenetre ; ne marche pas encore

3.23.2.15 #define PA_GetSpriteMode(screen, obj) ((PA_obj[screen][obj].atr0 & OBJ_MODE) >> 10)

Mode d'un sprite : 0 pour normal, 1 pour transparent, 2 pour fenetre.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.16 #define PA_SetSpriteMosaic(screen, obj, mosaic) PA_obj[screen][obj].atr0 = (PA_obj[screen][obj].atr0 & ALL_BUT(OBJ_MOSAIC)) + ((mosaic) << 12)

Mettre ou non un sprite en mode mosaic.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

mosaic Mode mosaic activé (1) ou désactivé (0)

3.23.2.17 #define PA_GetSpriteMosaic(screen, obj) ((PA_obj[screen][obj].atr0 & OBJ_MOSAIC) >> 12)

Si un sprite est en mode mosaic.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.18 #define PA_SetSpriteHflip(screen, obj, hflip) PA_obj[screen][obj].atr1 = (PA_obj[screen][obj].atr1 & ALL_BUT(OBJ_HFLIP)) + ((hflip) << 12)

Utiliser ou non le flip horizontal pour un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

hflip Flip horizontal, 1 pour oui, 0 pour non...

3.23.2.19 #define PA_GetSpriteHflip(screen, obj) ((PA_obj[screen][obj].atr1 & OBJ_HFLIP) >> 12)

S'il y a un flip horizontal pour un sprite.
Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.20 #define PA_SetSpriteVflip(screen, obj, vflip) PA_obj[screen][obj].atr1 = (PA_obj[screen][obj].atr1 & ALL_BUT(OBJ_VFLIP)) + ((vflip) << 13)

Utiliser ou non le flip vertical pour un sprite.
Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

vflip Flip vertical, 1 pour oui, 0 pour non...

3.23.2.21 #define PA_GetSpriteVflip(screen, obj) ((PA_obj[screen][obj].atr1 & OBJ_VFLIP) >> 13)

Si le flip vertical est utilisé ou non pour un sprite.
Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.22 #define PA_SetSpriteGfx(screen, obj, gfx) PA_obj[screen][obj].atr2 = (PA_obj[screen][obj].atr2 & ALL_BUT(OBJ_GFX)) + ((gfx) & OBJ_GFX)

Modifier les graphismes utilisés par un sprite.
Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

gfx Numéro du gfx en mémoire ; on peut obtenir un numéro avec PA_CreateGfx ou **PA_GetSpriteGfx(obj_number)** (p. 106) ;

3.23.2.23 #define PA_GetSpriteGfx(screen, obj) (PA_obj[screen][obj].atr2 & OBJ_GFX)

Récupérer le gfx utilisés par un sprite.
Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.24 #define PA_SetSpritePrio(screen, obj, prio) PA_obj[screen][obj].atr2 = (PA_obj[screen][obj].atr2 & ALL_BUT(OBJ_PRIO)) + ((prio) << 10)

Régler la priorité d'un sprite par rapport au Bg.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

prio Priorité du sprite : 0 est au-dessus du fond 0, 1 au-dessus du 1, etc... (0-3)

3.23.2.25 #define PA_GetSpritePrio(screen, obj) ((PA_obj[screen][obj].atr2 & OBJ_PRIO) >> 10)

Récupérer la priorité d'un sprite par rapport au Bg.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

3.23.2.26 #define PA_GetSpriteLx(screen, sprite) PA_size[PA_obj[screen][sprite].atr0 >> 14][PA_obj[screen][sprite].atr1 >> 14].lx

Récupérer la largeur d'un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro de l'objet dans le systeme de sprite

3.23.2.27 #define PA_GetSpriteLy(screen, sprite) PA_size[PA_obj[screen][sprite].atr0 >> 14][PA_obj[screen][sprite].atr1 >> 14].ly

Récupérer la hauteur d'un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro de l'objet dans le systeme de sprite

3.23.2.28 #define PA_CloneSprite(screen, obj, target) do{PA_obj[screen][obj].atr0 = PA_obj[screen][target].atr0 ; PA_obj[screen][obj].atr1 = PA_obj[screen][target].atr1 ; PA_obj[screen][obj].atr2 = PA_obj[screen][target].atr2 ; ++obj_per_gfx[screen][PA_GetSpriteGfx(screen, target)] ;}while(0)

Cloner un sprite. Marche uniquement pour les sprites sur un meme écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj Numéro de l'objet dans le systeme de sprite

target Numéro de la cible à cloner

3.23.3 Documentation des fonctions

3.23.3.1 u16 PA_CreateGfx (u8 *screen*, void * *obj_data*, u8 *obj_shape*, u8 *obj_size*, u8 *color_mode*)

Charger en mémoire un gfx à utiliser plus tard pour un sprite. Renvoie le numéro en mémoire.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj_data Gfx à charger

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

color_mode Mode 256 ou 16 couleurs (1 ou 0), ou 2 pour 16 bits

3.23.3.2 static inline void PA_CreateSprite (u8 *screen*, u8 *obj_number*, void * *obj_data*, u8 *obj_shape*, u8 *obj_size*, u8 *color_mode*, u8 *palette*, s16 *x*, s16 *y*) `[inline, static]`

Creer un sprite avec ses gfx... Ceci est la version simple de la fonction.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_data Gfx à charger

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

color_mode Mode 256 ou 16 couleurs (1 ou 0).

palette Palette à utiliser (0-15).

x Position X du sprite

y Position Y du sprite

3.23.3.3 static inline void PA_CreateSpriteEx (u8 *screen*, u8 *obj_number*, void * *obj_data*, u8 *obj_shape*, u8 *obj_size*, u8 *color_mode*, u8 *palette*, u8 *obj_mode*, u8 *mosaic*, u8 *hflip*, u8 *vflip*, u8 *prio*, u8 *dblsize*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite avec ses gfx... Ceci est la version complexe de la fonction.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_data Gfx à charger

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

color_mode Mode 256 ou 16 couleurs (1 ou 0).

palette Palette à utiliser (0-15).

obj_mode Mode du sprite (normal, transparent, fenetre). Pas encore opérationnel, laisser à 0...

mosaic Activer le mode mosaique pour ce sprite. Pas encore au point...

hflip Flip horizontal activé ou non.

vflip Flip vertical...

prio Priorité du sprite vis-à-vis des fonds : devant quel fond l'afficher... (0-3)

dblsize Doubler la taille possible du sprite. A activer uniquement si on compte grossir et faire tourner le sprite

x Position X du sprite

y Position Y du sprite

3.23.3.4 static inline void PA_Create16bitSpriteEx (u8 *screen*, u8 *obj_number*, void * *obj_data*, u8 *obj_shape*, u8 *obj_size*, u8 *mosaic*, u8 *hflip*, u8 *vflip*, u8 *prio*, u8 *dblsize*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite de 16 bits avec ses gfx... Ceci est la version complexe de la fonction. Attention : un sprite de 16 bits DOIT etre large de 128 pixels, meme si ce sprite ne prend qu'une petite partie sur la gauche.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_data Gfx à charger

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

mosaic Activer le mode mosaique pour ce sprite. Pas encore au point...

hflip Flip horizontal activé ou non.

vflip Flip vertical...

prio Priorité du sprite vis-à-vis des fonds : devant quel fond l'afficher... (0-3)

dblsize Doubler la taille possible du sprite. A activer uniquement si on compte grossir et faire tourner le sprite

x Position X du sprite

y Position Y du sprite

3.23.3.5 static inline void PA_Create16bitSpriteFromGfx (u8 *screen*, u8 *obj_number*, u16 *gfx*, u8 *obj_shape*, u8 *obj_size*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite de 16 bits à partir de gfx...

Paramètres:

screen Choix de l'écran (0 ou 1)

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

gfx Gfx à utiliser

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

x Position X du sprite

y Position Y du sprite

3.23.3.6 static inline void PA_Create16bitSprite (u8 *screen*, u8 *obj_number*, void * *obj_data*, u8 *obj_shape*, u8 *obj_size*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite de 16 bits avec ses gfx... Ceci est la version simple de la fonction. Attention : un sprite de 16 bits DOIT etre large de 128 pixels, meme si ce sprite ne prend qu'une petite partie sur la gauche.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_data Gfx à charger

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

x Position X du sprite

y Position Y du sprite

3.23.3.7 static inline void PA_CreateSpriteFromGfx (u8 *screen*, u8 *obj_number*, u16 *obj_gfx*, u8 *obj_shape*, u8 *obj_size*, u8 *color_mode*, u8 *palette*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite avec ses gfx... Ceci est la version simple de la fonction.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_gfx Gfx en mémoire à utiliser. On peut en avoir avec PA_GetSpriteGfx ou PA_CreateGfx

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

color_mode Mode 256 ou 16 couleurs (1 ou 0).

palette Palette à utiliser (0-15).

x Position X du sprite

y Position Y du sprite

3.23.3.8 static inline void PA_CreateSpriteExFromGfx (u8 *screen*, u8 *obj_number*, u16 *obj_gfx*, u8 *obj_shape*, u8 *obj_size*, u8 *color_mode*, u8 *palette*, u8 *obj_mode*, u8 *mosaic*, u8 *hflip*, u8 *vflip*, u8 *prio*, u8 *dblsize*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite avec ses gfx... Ceci est la version complexe de la fonction.

Paramètres:

screen Choix de l'écran (0 ou 1)

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_gfx Gfx en mémoire à utiliser. On peut en avoir avec PA_GetSpriteGfx ou PA_CreateGfx

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

color_mode Mode 256 ou 16 couleurs (1 ou 0).

palette Palette à utiliser (0-15).

obj_mode Mode du sprite (normal, transparent, fenetre). Pas encore opérationnel, laisser à 0...

mosaic Activer le mode mosaique pour ce sprite. Pas encore au point...

hflip Flip horizontal activé ou non.

vflip Flip vertical...

prio Priorité du sprite vis-à-vis des fonds : devant quel fond l'afficher... (0-3)

dblsize Doubler la taille possible du sprite. A activer uniquement si on compte grossir et faire tourner le sprite

x Position X du sprite

y Position Y du sprite

3.23.3.9 static inline void PA_UpdateGfx (u8 *screen*, u16 *gfx_number*, void * *obj_data*) [inline, static]

Mettre à jour les Gfx donnés.
Paramètres:

screen Choix de l'écran (0 ou 1)

gfx_number Numéro du Gfx en mémoire

obj_data Graphisme à charger

3.23.3.10 static inline void PA_UpdateGfxAndMem (u8 *screen*, u8 *gfx_number*, void * *obj_data*) [inline, static]

Mettre à jour les Gfx donnés et le pointer d'animation dans PAlib... Uniquement pour utilisateurs avertis.
Paramètres:

screen Choix de l'écran (0 ou 1)

gfx_number Numéro du Gfx en mémoire

obj_data Graphisme à charger

3.23.3.11 void PA_DeleteGfx (u8 *screen*, u16 *obj_gfx*)

Effacer un Gfx. Si un sprite l'utilisait, il deviendra invisible...
Paramètres:

screen Choix de l'écran (0 ou 1)

obj_gfx Numéro du Gfx en mémoire

3.23.3.12 void PA_DeleteSprite (u8 *screen*, u8 *obj_number*)

Effacer un sprite. S'il était le seul à utiliser un gfx, il sera effacé lui aussi.
Paramètres:

screen Choix de l'écran (0 ou 1)

obj_number Numéro du sprite

3.23.3.13 static inline void PA_SetRotset (u8 *screen*, u8 *rotset*, s16 *angle*, u16 *zoomx*, u16 *zoomy*) [inline, static]

Faire tourner et zoomer un sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

rotset Rotset que l'on veut changer. Pour attribuer un rotset à un sprite, utiliser PA_SetSpriteRotEnable

angle Angle, entre 0 et 512 (et non 360, attention !)

zoomx Zoom horizontal. 256 est pas de zoom, 512 2 fois plus petit, et 128 2 fois plus grand... Ajuster au mieux ! :p

zoomy Zoom vertical. 256 est pas de zoom, 512 2 fois plus petit, et 128 2 fois plus grand... Ajuster au mieux ! :p

3.23.3.14 static inline void PA_SetRotsetNoZoom (u8 *screen*, u8 *rotset*, s16 *angle*) [inline, static]

Faire tourner un sprite sans zoomer. C'est un peu plus rapide que la fonction PA_-SetRotset.

Paramètres:

screen Choix de l'écran (0 ou 1)

rotset Rotset que l'on veut changer. Pour attribuer un rotset à un sprite, utiliser PA_SetSpriteRotEnable

angle Angle, entre 0 et 512 (et non 360, attention !)

3.23.3.15 static inline void PA_SetRotsetNoAngle (u8 *screen*, u8 *rotset*, u16 *zoomx*, u16 *zoomy*) [inline, static]

Zoomer un sprite sans le faire tourner. C'est un peu plus rapide que la fonction PA_-SetRotset.

Paramètres:

screen Choix de l'écran (0 ou 1)

rotset Rotset que l'on veut changer. Pour attribuer un rotset à un sprite, utiliser PA_SetSpriteRotEnable

zoomx Zoom horizontal. 256 est pas de zoom, 512 2 fois plus petit, et 128 2 fois plus grand... Ajuster au mieux ! :p

zoomy Zoom vertical. 256 est pas de zoom, 512 2 fois plus petit, et 128 2 fois plus grand... Ajuster au mieux ! :p

3.23.3.16 static inline void PA_SetSpriteXY (u8 *screen*, u8 *sprite*, s16 *x*, s16 *y*) [inline, static]

Position X et Y du sprite à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

x Position Y

y Position Y

3.23.3.17 static inline void PA_Set16bitSpriteAlpha (u8 *screen*, u8 *sprite*, u8 *alpha*) [inline, static]

Position X du sprite à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro de l'objet dans le systeme de sprite, uniquement pour les sprites 16bit

alpha Paramèter alpha, 0-15

3.23.3.18 static inline void PA_SetSpriteAnimEx (u8 *screen*, u8 *sprite*, u8 *lx*, u8 *ly*, u8 *ncolors*, s16 *animframe*) [inline, static]

Régler l'image du sprite dans l'animation. Cette fonction est plus rapide que PA_SetSpriteAnim parce qu'elle n'a pas à rechercher les dimensions du sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

lx Largeur du sprite (8, 16, 32, 64)

ly Hauteur du sprite (8, 16, 32, 64)

ncolors Mode couleur du sprite (0 pour 16 couleurs, 1 pour 256)

animframe Frame de l'animation du sprite (0, 1, 2, etc...)

3.23.3.19 static inline void PA_SetSpriteAnim (u8 *screen*, u8 *sprite*, s16 *animframe*) [inline, static]

Régler l'image du sprite dans l'animation. Identique à PA_SetSpriteAnimEx, mais plus simple à utiliser, par contre plus lent.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

animframe Frame de l'animation du sprite (0, 1, 2, etc...)

3.23.3.20 void PA_StartSpriteAnimEx (u8 *screen*, u8 *sprite*, s16 *firstframe*, s16 *lastframe*, s16 *speed*, u8 *type*, s16 *ncycles*)

Démarre une animation de sprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

firstframe Premières image de l'animation, généralement 0....

lastframe Dernière image à afficher. Une fois atteinte, ca retourne à la première

speed Vitesse, en frames par seconde (fps). 1 signifie donc 1 image par seconde...

type Défini de quelle manière on veut boucler. ANIM_LOOP (0) pour normal, et ANIM_UPDOWN (1) pour d'avant en arrière

ncycles Nombres de cycles d'animations avant l'arrêt. Si on utilise ANIM_-UPDOWN, il faut 2 cycles pour que l'animation revienne à l'image de base

3.23.3.21 static inline void PA_StartSpriteAnim (u8 *screen*, u8 *sprite*, s16 *firstframe*, s16 *lastframe*, s16 *speed*) `[inline, static]`

Démarre une animation de sprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

firstframe Premières image de l'animation, généralement 0....

lastframe Dernière image à afficher. Une fois atteinte, ca retourne à la première

speed Vitesse, en frames par seconde (fps). 1 signifie donc 1 image par seconde...

3.23.3.22 static inline void PA_StopSpriteAnim (u8 *screen*, u8 *sprite*) `[inline, static]`

Arrêter une animation de sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

3.23.3.23 static inline void PA_SetSpriteAnimFrame (u8 *screen*, u8 *sprite*, u16 *frame*) `[inline, static]`

Changer le numéro actuel de la frame d'animation.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

frame Numéro de frame...

3.23.3.24 static inline u16 PA_GetSpriteAnimFrame (u8 *screen*, u8 *sprite*) [inline, static]

Renvoie le numéro actuel de la frame d'animation.
Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

3.23.3.25 static inline void PA_SetSpriteAnimSpeed (u8 *screen*, u8 *sprite*, s16 *speed*) [inline, static]

Changer la vitesse de l'animation.
Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

speed Vitesse, en fps...

3.23.3.26 static inline u16 PA_GetSpriteAnimSpeed (u8 *screen*, u8 *sprite*) [inline, static]

Renvoie la vitesse de l'animation.
Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

3.23.3.27 static inline void PA_SetSpriteNCycles (u8 *screen*, u8 *sprite*, s32 *NCycles*) [inline, static]

Changer le nombre de cycles d'animation restant (-1 pour inifini).
Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

NCycles Nombre de cycles

3.23.3.28 static inline s32 PA_GetSpriteNCycles (u8 *screen*, u8 *sprite*) [inline, static]

Renvoie le nombre de cycles d'animation restants.
Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

3.23.3.29 static inline u16 PA_SpriteAnimPause (u8 *screen*, u8 *sprite*, u8 *pause*) [inline, static]

Mettre en Pause en remettre en lecture une animation de sprite.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

pause 1 pour pause, 0 pour reprendre la lecture...

3.23.3.30 static inline void PA_SetSpritePixel (u8 *screen*, u8 *sprite*, u8 *x*, u8 *y*, u8 *color*) [inline, static]

Mettre un pixel d'un sprite à une couleur donnée. Comme PA_SetSpritePixelEx, avec moins d'options, mais un peu plus lent.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

x Coordonnée X du pixel à changer

y Coordonnée Y du pixel à changer

color Nouvelle couleur de la palette à metrre

3.23.3.31 static inline u8 PA_GetSpritePixel (u8 *screen*, u8 *sprite*, u8 *x*, u8 *y*) [inline, static]

Récupérer la couleur d'un pixel d'un sprite. Comme PA_GetSpritePixelEx, avec moins d'options, mais un peu plus lent.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

x Coordonnée X du pixel

y Coordonnée Y du pixel

3.23.3.32 static inline u8 PA_GetSprite16cPixel (u8 *screen*, u8 *sprite*, u8 *x*, u8 *y*) [inline, static]

Récupérer la couleur d'un pixel d'un sprite de 16 couleurs.

Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

x Coordonnée X du pixel

y Coordonnée Y du pixel

3.23.3.33 void PA_InitSpriteDraw (u8 *screen*, u8 *sprite*)

Initialise un sprite pour pouvoir dessiner dessus !
Paramètres:

screen Choix de l'écran (0 ou 1)

sprite Numéro du sprite dans le systeme de sprite

3.23.3.34 void PA_InitSpriteExtPrio (u8 *SpritePrio*)

Activer le systeme de priorité de sprites PAlib. Plus lent que le systeme normal, il permet d'avoir 256 niveaux de priorité (supplante la priorité par numéro de sprites).
Paramètres:

SpritePrio 1 pour on, 0 pour off...

3.24 Sprite system for Dual Screen

Fonctions

– static void **PA_SetScreenSpace** (s16 ScreenSpace)
Désigner l'espace entre les 2 écrans, 48 pixels par défaut.

– static void **PA_DualSetSpriteX** (u8 obj, s16 x)
Position X du sprite à l'écran.

– static void **PA_DualSetSpriteY** (u8 obj, s16 y)
Position Y du sprite à l'écran.

– static void **PA_DualSetSpriteXY** (u8 sprite, s16 x, s16 y)
Position X et Y du sprite à l'écran.

– static void **PA_DualCreateSprite** (u8 obj_number, void ∗obj_data, u8 obj_shape, u8 obj_size, u8 color_mode, u8 palette, s16 x, s16 y)
Creer un sprite avec ses gfxsur les 2 écrans.

– static void **PA_DualCreateSpriteEx** (u8 obj_number, void ∗obj_data, u8 obj_shape, u8 obj_size, u8 color_mode, u8 palette, u8 obj_mode, u8 mosaic, u8 hflip, u8 vflip, u8 prio, u8 dblsize, s16 x, s16 y)
Creer un sprite avec ses gfx... Ceci est la version complexe de la fonction.

– static void **PA_DualCreate16bitSpriteEx** (u8 obj_number, void ∗obj_data, u8 obj_shape, u8 obj_size, u8 mosaic, u8 hflip, u8 vflip, u8 prio, u8 dblsize, s16 x, s16 y)
Creer un sprite de 16 bits avec ses gfx... Ceci est la version complexe de la fonction. Attention : un sprite de 16 bits DOIT etre large de 128 pixels, meme si ce sprite ne prend qu'une petite partie sur la gauche.

– static void **PA_DualCreate16bitSprite** (u8 obj_number, void ∗obj_data, u8 obj_shape, u8 obj_size, s16 x, s16 y)
Creer un sprite de 16 bits avec ses gfx... Ceci est la version simple de la fonction. Attention : un sprite de 16 bits DOIT etre large de 128 pixels, meme si ce sprite ne prend qu'une petite partie sur la gauche.

– static void **PA_DualCreateSpriteFromGfx** (u8 obj_number, u16 ∗obj_gfx, u8 obj_shape, u8 obj_size, u8 color_mode, u8 palette, s16 x, s16 y)
Creer un sprite avec ses gfx... Ceci est la version simple de la fonction.

– static void **PA_DualCreateSpriteExFromGfx** (u8 obj_number, u16 ∗obj_gfx, u8 obj_shape, u8 obj_size, u8 color_mode, u8 palette, u8 obj_mode, u8 mosaic, u8 hflip, u8 vflip, u8 prio, u8 dblsize, s16 x, s16 y)
Creer un sprite avec ses gfx... Ceci est la version complexe de la fonction.

– static void **PA_DualUpdateSpriteGfx** (u8 obj_number, void ∗obj_data)
Mettre à jour les Gfx d'un sprite donné.

– static void **PA_DualUpdateGfx** (u16 gfx_number, void ∗obj_data)
Mettre à jour les Gfx d'un sprite donné.

– static void **PA_DualDeleteSprite** (u8 obj_number)
Effacer un sprite. S'il était le seul à utiliser un gfx, il sera effacé lui aussi.

– static void **PA_DualSetSpriteRotEnable** (u8 sprite, u8 rotset)
Faire tourner et zoomer un sprite.

– static void **PA_DualSetSpriteRotDisable** (u8 sprite)
Arreter de faire tourner et zoomer un sprite.

– static void **PA_DualSetRotset** (u8 rotset, s16 angle, u16 zoomx, u16 zoomy)
Faire tourner et zoomer un sprite.

– static void **PA_DualSetRotsetNoZoom** (u8 rotset, s16 angle)
Faire tourner un sprite sans zoomer. C'est un peu plus rapide que la fonction PA_-SetRotset.

– static void **PA_DualSetRotsetNoAngle** (u8 rotset, u16 zoomx, u16 zoomy)
Zoomer un sprite sans le faire tourner. C'est un peu plus rapide que la fonction PA_-SetRotset.

– static void **PA_DualSetSpritePal** (u8 obj, u8 pal)
Changer la palette d'un sprite.

– static void **PA_DualSetSpriteDblsize** (u8 obj, u8 dblsize)
Activer ou désactiver le mode Doublesize pour un sprite.

– static void **PA_DualSetSpriteColors** (u8 sprite, u8 n_colors)
Changer le mode de couleur du sprite.

– static void **PA_DualSetSpriteMode** (u8 sprite, u8 obj_mode)
Régler le mode d'un sprite : 0 pour normal, 1 pour transparent, 2 pour fenetre.

– static void **PA_DualSetSpriteMosaic** (u8 obj, u8 mosaic)
Mettre ou non un sprite en mode mosaic.

– static void **PA_DualSetSpriteHflip** (u8 obj, u8 hflip)
Utiliser ou non le flip horizontal pour un sprite.

– static void **PA_DualSetSpriteVflip** (u8 obj, u8 vflip)
Utiliser ou non le flip vertical pour un sprite.

– static void **PA_DualSetSpriteGfx** (u8 obj, u16 ∗gfx)
Modifier les graphismes utilisés par un sprite.

– static void **PA_DualSetSpritePrio** (u8 obj, u8 prio)
Régler la priorité d'un sprite par rapport au Bg.

– static void **PA_DualCloneSprite** (u8 obj, u8 target)
Cloner un sprite. Marche uniquement pour les sprites sur un meme écran.

– static void **PA_DualSetSpriteAnimEx** (u8 sprite, u8 lx, u8 ly, u8 ncolors, s16 animframe)
Régler l'image du sprite dans l'animation. Cette fonction est plus rapide que PA_-SetSpriteAnim parce qu'elle n'a pas à rechercher les dimensions du sprite.

– static void **PA_DualSetSpriteAnim** (u8 sprite, s16 animframe)
Régler l'image du sprite dans l'animation. Identique à PA_SetSpriteAnimEx, mais plus simple à utiliser, par contre plus lent.

– static void **PA_DualStartSpriteAnimEx** (u8 sprite, s16 firstframe, s16 lastframe,

s16 speed, u8 type, s16 ncycles)
Démarre une animation de sprite pour DualSprites. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

– static void **PA_DualStartSpriteAnim** (u8 sprite, s16 firstframe, s16 lastframe, s16 speed)
Démarre une animation de sprite pour DualSprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

– static void **PA_DualStopSpriteAnim** (u8 sprite)
Arrêter une animation de sprite pour les DualSprites.

– static void **PA_DualSetSpriteAnimFrame** (u8 sprite, u16 frame)
Changer le numéro actuel de la frame d'animation pour les DualSprites.

– static u16 **PA_DualGetSpriteAnimFrame** (u8 sprite)
Renvoie le numéro actuel de la frame d'animation pour les DualSprites.

– static void **PA_DualSetSpriteAnimSpeed** (u8 sprite, s16 speed)
Changer la vitesse de l'animation pour les DualSprites.

– static u16 **PA_DualGetSpriteAnimSpeed** (u8 sprite)
Renvoie la vitesse de l'animation pour les DualSprites.

– static void **PA_DualSpriteAnimPause** (u8 sprite, u8 pause)
Mettre en Pause en remettre en lecture une animation de sprite pour les DualSprites.

3.24.1 Description détaillée

Load Sprite, move them around, rotate them...

3.24.2 Documentation des fonctions

3.24.2.1 static inline void PA_SetScreenSpace (s16 *ScreenSpace*) [inline, static]

Désigner l'espace entre les 2 écrans, 48 pixels par défaut.

Paramètres:

ScreenSpace Espace en pixels

3.24.2.2 static inline void PA_DualSetSpriteX (u8 *obj*, s16 *x*) [inline, static]

Position X du sprite à l'écran.

Paramètres:

obj Numéro de l'objet dans le systeme de sprite

x Position X

3.24.2.3 static inline void PA_DualSetSpriteY (u8 *obj*, s16 *y*) [inline, static]

Position Y du sprite à l'écran.
Paramètres:

obj Numéro de l'objet dans le systeme de sprite

y Position Y

3.24.2.4 static inline void PA_DualSetSpriteXY (u8 *sprite*, s16 *x*, s16 *y*) [inline, static]

Position X et Y du sprite à l'écran.
Paramètres:

sprite Numéro du sprite dans le systeme de sprite

x Position Y

y Position Y

3.24.2.5 static inline void PA_DualCreateSprite (u8 *obj_number*, void * *obj_data*, u8 *obj_shape*, u8 *obj_size*, u8 *color_mode*, u8 *palette*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite avec ses gfxsur les 2 écrans.
Paramètres:

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_data Gfx à charger

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

color_mode Mode 256 ou 16 couleurs (1 ou 0).

palette Palette à utiliser (0-15).

x Position X du sprite

y Position Y du sprite

3.24.2.6 static inline void PA_DualCreateSpriteEx (u8 *obj_number*, void * *obj_data*, u8 *obj_shape*, u8 *obj_size*, u8 *color_mode*, u8 *palette*, u8 *obj_mode*, u8 *mosaic*, u8 *hflip*, u8 *vflip*, u8 *prio*, u8 *dblsize*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite avec ses gfx... Ceci est la version complexe de la fonction.
Paramètres:

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_data Gfx à charger

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

color_mode Mode 256 ou 16 couleurs (1 ou 0).

palette Palette à utiliser (0-15).

obj_mode Mode du sprite (normal, transparent, fenetre). Pas encore opérationnel, laisser à 0...

mosaic Activer le mode mosaique pour ce sprite. Pas encore au point...

hflip Flip horizontal activé ou non.

vflip Flip vertical...

prio Priorité du sprite vis-à-vis des fonds : devant quel fond l'afficher... (0-3)

dblsize Doubler la taille possible du sprite. A activer uniquement si on compte grossir et faire tourner le sprite

x Position X du sprite

y Position Y du sprite

3.24.2.7 static inline void PA_DualCreate16bitSpriteEx (u8 *obj_number*, void * *obj_data*, u8 *obj_shape*, u8 *obj_size*, u8 *mosaic*, u8 *hflip*, u8 *vflip*, u8 *prio*, u8 *dblsize*, s16 *x*, s16 *y*) `[inline, static]`

Creer un sprite de 16 bits avec ses gfx... Ceci est la version complexe de la fonction. Attention : un sprite de 16 bits DOIT etre large de 128 pixels, meme si ce sprite ne prend qu'une petite partie sur la gauche.

Paramètres:

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_data Gfx à charger

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

mosaic Activer le mode mosaique pour ce sprite. Pas encore au point...

hflip Flip horizontal activé ou non.

vflip Flip vertical...

prio Priorité du sprite vis-à-vis des fonds : devant quel fond l'afficher... (0-3)

dblsize Doubler la taille possible du sprite. A activer uniquement si on compte grossir et faire tourner le sprite

x Position X du sprite

y Position Y du sprite

3.24.2.8 static inline void PA_DualCreate16bitSprite (u8 *obj_number*, void * *obj_data*, u8 *obj_shape*, u8 *obj_size*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite de 16 bits avec ses gfx... Ceci est la version simple de la fonction. Attention : un sprite de 16 bits DOIT etre large de 128 pixels, meme si ce sprite ne prend qu'une petite partie sur la gauche.

Paramètres:

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_data Gfx à charger

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

x Position X du sprite

y Position Y du sprite

3.24.2.9 static inline void PA_DualCreateSpriteFromGfx (u8 *obj_number*, u16 * *obj_gfx*, u8 *obj_shape*, u8 *obj_size*, u8 *color_mode*, u8 *palette*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite avec ses gfx... Ceci est la version simple de la fonction.

Paramètres:

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_gfx Gfx en mémoire à utiliser. On peut en avoir avec PA_GetSpriteGfx ou PA_CreateGfx

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

color_mode Mode 256 ou 16 couleurs (1 ou 0).

palette Palette à utiliser (0-15).

x Position X du sprite

y Position Y du sprite

3.24.2.10 static inline void PA_DualCreateSpriteExFromGfx (u8 *obj_number*, u16 * *obj_gfx*, u8 *obj_shape*, u8 *obj_size*, u8 *color_mode*, u8 *palette*, u8 *obj_mode*, u8 *mosaic*, u8 *hflip*, u8 *vflip*, u8 *prio*, u8 *dblsize*, s16 *x*, s16 *y*) [inline, static]

Creer un sprite avec ses gfx... Ceci est la version complexe de la fonction.

Paramètres:

obj_number Numéro du sprite que vous voulez utiliser (de 0 à 127 pour chaque écran séparemment).

obj_gfx Gfx en mémoire à utiliser. On peut en avoir avec PA_GetSpriteGfx ou PA_CreateGfx

obj_shape Forme du sprite à charger, de 0 à 2. Utiliser la macro OBJ_SIZE_-32X32 (...) pour charger la forme et la taille...

obj_size Taille du sprite. Utiliser la macro OBJ_SIZE_32X32 (...) pour charger la forme et la taille...

color_mode Mode 256 ou 16 couleurs (1 ou 0).

palette Palette à utiliser (0-15).

obj_mode Mode du sprite (normal, transparent, fenetre). Pas encore opérationnel, laisser à 0...

mosaic Activer le mode mosaique pour ce sprite. Pas encore au point...

hflip Flip horizontal activé ou non.

vflip Flip vertical...

prio Priorité du sprite vis-à-vis des fonds : devant quel fond l'afficher... (0-3)

dblsize Doubler la taille possible du sprite. A activer uniquement si on compte grossir et faire tourner le sprite

x Position X du sprite

y Position Y du sprite

3.24.2.11 static inline void PA_DualUpdateSpriteGfx (u8 *obj_number*, void * *obj_data*) [inline, static]

Mettre à jour les Gfx d'un sprite donné.
Paramètres:

obj_number Numéro de l'objet dans le systeme de sprite

obj_data Graphisme à charger

3.24.2.12 static inline void PA_DualUpdateGfx (u16 *gfx_number*, void * *obj_data*) [inline, static]

Mettre à jour les Gfx d'un sprite donné.
Paramètres:

gfx_number Numéro du Gfx en mémoire

obj_data Graphisme à charger

3.24.2.13 static inline void PA_DualDeleteSprite (u8 *obj_number*) [inline, static]

Effacer un sprite. S'il était le seul à utiliser un gfx, il sera effacé lui aussi.
Paramètres:

obj_number Numéro du sprite

3.24.2.14 static inline void PA_DualSetSpriteRotEnable (u8 *sprite*, u8 *rotset*) [inline, static]

Faire tourner et zoomer un sprite.

Paramètres:

sprite Sprite que l'on veut faire tourner

rotset Rotset que l'on veut pour un sprite donné (0-31). On peut a priori utiliser un rotset pour plusieurs sprites, s'ils sont zoomés/tournés pareil...

3.24.2.15 static inline void PA_DualSetSpriteRotDisable (u8 *sprite*) [inline, static]

Arreter de faire tourner et zoomer un sprite.

Paramètres:

sprite Sprite que l'on veut faire tourner

3.24.2.16 static inline void PA_DualSetRotset (u8 *rotset*, s16 *angle*, u16 *zoomx*, u16 *zoomy*) [inline, static]

Faire tourner et zoomer un sprite.

Paramètres:

rotset Rotset que l'on veut changer. Pour attribuer un rotset à un sprite, utiliser PA_SetSpriteRotEnable

angle Angle, entre 0 et 512 (et non 360, attention !)

zoomx Zoom horizontal. 256 est pas de zoom, 512 2 fois plus petit, et 128 2 fois plus grand... Ajuster au mieux ! :p

zoomy Zoom vertical. 256 est pas de zoom, 512 2 fois plus petit, et 128 2 fois plus grand... Ajuster au mieux ! :p

3.24.2.17 static inline void PA_DualSetRotsetNoZoom (u8 *rotset*, s16 *angle*) [inline, static]

Faire tourner un sprite sans zoomer. C'est un peu plus rapide que la fonction PA_SetRotset.

Paramètres:

rotset Rotset que l'on veut changer. Pour attribuer un rotset à un sprite, utiliser PA_SetSpriteRotEnable

angle Angle, entre 0 et 512 (et non 360, attention !)

3.24.2.18 static inline void PA_DualSetRotsetNoAngle (u8 *rotset*, u16 *zoomx*, u16 *zoomy*) [inline, static]

Zoomer un sprite sans le faire tourner. C'est un peu plus rapide que la fonction PA_SetRotset.

Paramètres:

rotset Rotset que l'on veut changer. Pour attribuer un rotset à un sprite, utiliser PA_SetSpriteRotEnable

zoomx Zoom horizontal. 256 est pas de zoom, 512 2 fois plus petit, et 128 2 fois plus grand... Ajuster au mieux ! :p

zoomy Zoom vertical. 256 est pas de zoom, 512 2 fois plus petit, et 128 2 fois plus grand... Ajuster au mieux ! :p

3.24.2.19 static inline void PA_DualSetSpritePal (u8 *obj*, u8 *pal*) [inline, static]

Changer la palette d'un sprite.

Paramètres:

obj Numéro de l'objet dans le systeme de sprite

pal Numéro de la palette (de 0 à 15)

3.24.2.20 static inline void PA_DualSetSpriteDblsize (u8 *obj*, u8 *dblsize*) [inline, static]

Activer ou désactiver le mode Doublesize pour un sprite.

Paramètres:

obj Numéro de l'objet dans le systeme de sprite

dblsize 1 pour l'activer, 0 pour l'inactiver

3.24.2.21 static inline void PA_DualSetSpriteColors (u8 *sprite*, u8 *n_colors*) [inline, static]

Changer le mode de couleur du sprite.

Paramètres:

sprite Numéro de l'objet dans le systeme de sprite

n_colors 0 pour 16 couleurs, 1 pour 256

3.24.2.22 static inline void PA_DualSetSpriteMode (u8 *sprite*, u8 *obj_mode*) [inline, static]

Régler le mode d'un sprite : 0 pour normal, 1 pour transparent, 2 pour fenetre.

Paramètres:

sprite Numéro de l'objet dans le systeme de sprite

obj_mode Mode : 0 pour normal, 1 pour transparent, 2 pour fenetre ; ne marche pas encore

3.24.2.23 static inline void PA_DualSetSpriteMosaic (u8 *obj*, u8 *mosaic*) [inline, static]

Mettre ou non un sprite en mode mosaic.

Paramètres:

obj Numéro de l'objet dans le systeme de sprite

mosaic Mode mosaic activé (1) ou désactivé (0)

3.24.2.24 static inline void PA_DualSetSpriteHflip (u8 *obj*, u8 *hflip*) [inline, static]

Utiliser ou non le flip horizontal pour un sprite.

Paramètres:

obj Numéro de l'objet dans le systeme de sprite

hflip Flip horizontal, 1 pour oui, 0 pour non...

3.24.2.25 static inline void PA_DualSetSpriteVflip (u8 *obj*, u8 *vflip*) [inline, static]

Utiliser ou non le flip vertical pour un sprite.

Paramètres:

obj Numéro de l'objet dans le systeme de sprite

vflip Flip vertical, 1 pour oui, 0 pour non...

3.24.2.26 static inline void PA_DualSetSpriteGfx (u8 *obj*, u16 * *gfx*) [inline, static]

Modifier les graphismes utilisés par un sprite.

Paramètres:

obj Numéro de l'objet dans le systeme de sprite

gfx Numéro du gfx en mémoire ; on peut obtenir un numéro avec PA_CreateGfx ou **PA_GetSpriteGfx(obj_number)** (p. 106) ;

3.24.2.27 static inline void PA_DualSetSpritePrio (u8 *obj*, u8 *prio*) [inline, static]

Régler la priorité d'un sprite par rapport au Bg.

Paramètres:

obj Numéro de l'objet dans le systeme de sprite

prio Priorité du sprite : 0 est au-dessus du fond 0, 1 au-dessus du 1, etc... (0-3)

3.24.2.28 static inline void PA_DualCloneSprite (u8 *obj*, u8 *target*) [inline, static]

Cloner un sprite. Marche uniquement pour les sprites sur un meme écran.
Paramètres:

obj Numéro de l'objet dans le systeme de sprite

target Numéro de la cible à cloner

3.24.2.29 static inline void PA_DualSetSpriteAnimEx (u8 *sprite*, u8 *lx*, u8 *ly*, u8 *ncolors*, s16 *animframe*) [inline, static]

Régler l'image du sprite dans l'animation. Cette fonction est plus rapide que PA_-SetSpriteAnim parce qu'elle n'a pas à rechercher les dimensions du sprite.
Paramètres:

sprite Numéro du sprite dans le systeme de sprite

lx Largeur du sprite (8, 16, 32, 64)

ly Hauteur du sprite (8, 16, 32, 64)

ncolors Mode couleur du sprite (0 pour 16 couleurs, 1 pour 256)

animframe Frame de l'animation du sprite (0, 1, 2, etc...)

3.24.2.30 static inline void PA_DualSetSpriteAnim (u8 *sprite*, s16 *animframe*) [inline, static]

Régler l'image du sprite dans l'animation. Identique à PA_SetSpriteAnimEx, mais plus simple à utiliser, par contre plus lent.
Paramètres:

sprite Numéro du sprite dans le systeme de sprite

animframe Frame de l'animation du sprite (0, 1, 2, etc...)

3.24.2.31 static inline void PA_DualStartSpriteAnimEx (u8 *sprite*, s16 *firstframe*, s16 *lastframe*, s16 *speed*, u8 *type*, s16 *ncycles*) [inline, static]

Démarre une animation de sprite pour DualSprites. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !
Paramètres:

sprite Numéro du sprite dans le systeme de sprite

firstframe Premières image de l'animation, généralement 0....

lastframe Dernière image à afficher. Une fois atteinte, ca retourne à la première

speed Vitesse, en frames par seconde (fps). 1 signifie donc 1 image par seconde...

type Défini de quelle manière on veut boucler. ANIM_LOOP (0) pour normal, et ANIM_UPDOWN (1) pour d'avant en arrière

ncycles Nombres de cycles d'animations avant l'arrêt. Si on utilise ANIM_-UPDOWN, il faut 2 cycles pour que l'animation revienne à l'image de base

3.24.2.32 static inline void PA_DualStartSpriteAnim (u8 *sprite*, s16 *firstframe*, s16 *lastframe*, s16 *speed*) [inline, static]

Démarre une animation de sprite pour DualSprite. Une fois démarrée, elle continue tant qu'on ne l'arrête pas !

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

firstframe Premières image de l'animation, généralement 0....

lastframe Dernière image à afficher. Une fois atteinte, ca retourne à la première

speed Vitesse, en frames par seconde (fps). 1 signifie donc 1 image par seconde...

3.24.2.33 static inline void PA_DualStopSpriteAnim (u8 *sprite*) [inline, static]

Arrêter une animation de sprite pour les DualSprites.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

3.24.2.34 static inline void PA_DualSetSpriteAnimFrame (u8 *sprite*, u16 *frame*) [inline, static]

Changer le numéro actuel de la frame d'animation pour les DualSprites.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

frame Numéro de frame...

3.24.2.35 static inline u16 PA_DualGetSpriteAnimFrame (u8 *sprite*) [inline, static]

Renvoie le numéro actuel de la frame d'animation pour les DualSprites.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

3.24.2.36 static inline void PA_DualSetSpriteAnimSpeed (u8 *sprite*, s16 *speed*) [inline, static]

Changer la vitesse de l'animation pour les DualSprites.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

speed Vitesse, en fps...

3.24.2.37 static inline u16 PA_DualGetSpriteAnimSpeed (u8 *sprite*) [inline, static]

Renvoie la vitesse de l'animation pour les DualSprites.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

3.24.2.38 static inline void PA_DualSpriteAnimPause (u8 *sprite*, u8 *pause*) [inline, static]

Mettre en Pause en remettre en lecture une animation de sprite pour les DualSprites.

Paramètres:

sprite Numéro du sprite dans le systeme de sprite

pause 1 pour pause, 0 pour reprendre la lecture...

3.25 Text output system

Macros

– #define **PA_InitText** PA_LoadDefaultText
*Old name for **PA_LoadDefaultText()** (p. 135).*

– #define **PA_SetTileLetter**(screen, x, y, letter) PA_SetMapTileAll(screen, PAbgtext[screen], x, y, (PA_textmap[screen][(u16)letter]&((1<<12)-1)) + (PAtext_pal[screen] << 12))
Ecrire une lettre à l'écran.

– #define **PA_InitCustomText**(screen, bg_select, text) PA_InitCustomTextEx(screen, bg_select, text##_Tiles, text##_Map, text##_Pal)
[DEPRECATED] Initialiser le texte en utilisant une police perso

– #define **PA_ShowFont**(screen) PA_LoadBgMap(screen, PAbgtext[screen], (void*)PA_textmap[screen], BG_256X256)
Affiche la police utilisée. C'est juste utile pour du débuggage, aucun intéret autrement.

– #define **PA_8bitCustomFont**(bit8_slot, bit8_font)
[DEPRECATED] Ajouter une police perso dans le systeme de texte 8bit ! ! Doit être convertie avec PAGfx

Fonctions

– void **PA_LoadDefaultText** (u8 screen, u8 bg_select)
Charger et initialiser le texte. Ne marche qu'en modes 0-2.

– static void **PA_SetTextTileCol** (u8 screen, u8 color)
Change la couleur du texte à écrire (ne change pas la couleur du texte déjà écrit).

– void **PA_OutputText** (u8 screen, u16 x, u16 y, const char *text,...)
Ecrire du texte à l'écran. Ne marche qu'en modes 0-2.

– u16 **PA_OutputSimpleText** (u8 screen, u16 x, u16 y, const char *text)
Ecrire du texte tout simple à l'écran. Ne marche qu'en modes 0-2. Beaucoup plus rapide que PA_OutputText, masi aussi beaucoup plus limité... Renvoie le nombre de lettres.

– u32 **PA_BoxText** (u8 screen, u16 basex, u16 basey, u16 maxx, u16 maxy, const char *text, u32 limit)
Permet d'écrire du texte à l'écran, dans une boite délimitée au choix, et en choisissant le nombre de lettres à afficher (peut être utile pour afficher du texte en train de se taper, sinon suffit de mettre 10000 pour afficher tout d'un coup) Renvoie le nmobre de lettre écrites.

– u32 **PA_BoxTextNoWrap** (u8 screen, u16 basex, u16 basey, u16 maxx, u16 maxy, const char *text, u32 limit)
Permet d'écrire du texte à l'écran, dans une boite délimitée au choix, et en choisissant le nombre de lettres à afficher (peut être utile pour afficher du texte en train de se taper, sinon suffit de mettre 10000 pour afficher tout d'un coup) Renvoie le nombre de lettre écrites. Cette fonction coupe les mots...

– static void **PA_SetTextCol** (u8 screen, u16 r, u16 g, u16 b)

Changer la couleur de base du texte à l'écran.

– void **PA_LoadText** (u8 screen, u8 bg_number, const **PA_BgStruct** ∗font)
Initialiser le texte en utilisant une police perso.

– s16 **PA_8bitText** (u8 screen, s16 basex, s16 basey, s16 maxx, s16 maxy, const char ∗text, u8 color, u8 size, u8 transp, s32 limit)
Cette fonction permet d'écrire du texte à chasse variable à l'écran. Elle nécessite d'avoir un fond dessinable de 8 bits (cf PA_Init8bitBg). Les options sont la taille, la transparence, et les limites, ainsi que la couleur. Seul inconvénient : il n'accepte pas les commande comme d, etc... La fonction renvoie le nombre de charactères écrits.

– s16 **PA_CenterSmartText** (u8 screen, s16 basex, s16 basey, s16 maxx, s16 maxy, const char ∗text, u8 color, u8 size, u8 transp)
En gros la meme chose que SmartText, mais en centré...

– void **PA_AddBitmapFont** (int slot, const **PA_BgStruct** ∗font)
Ajouter une police perso dans le système de texte 8bit/16bit.

– void **PA_InitTextBorders** (u8 screen, u8 x1, u8 y1, u8 x2, u8 y2)
Initialise une boite à texte, avec la bordure. Ceci rend l'utilisation des textes délimités bien plus simple.

– void **PA_EraseTextBox** (u8 screen)
Efface le text d'un boite à texte... Nécessite qu'il ait été initialisé avec PA_InitTextBorders.

– static u32 **PA_SimpleBoxText** (u8 screen, const char ∗text, u32 limit)
Ecrit du texte dans une zone délimitée. Similaire à PA_BoxText, mais sans avoir besoin de délimiter.

– void **PA_ClearTextBg** (u8 screen)
Effacer tout le texte sur un écran donné.

– void **PA_Print** (u8 screen, const char ∗text,...)
Ecrire du texte à l'écran. Marche comme la fonction printf.

– static void **PA_PrintLetter** (u8 screen, char letter)
Comme PA_Print, mais juste pour une lettre.

3.25.1 Description détaillée

Allows you to output text...

3.25.2 Documentation des macros

3.25.2.1 #define PA_SetTileLetter(screen, x, y, letter) PA_SetMapTileAll(screen, PAbgtext[screen], x, y, (PA_textmap[screen][(u16)letter]&((1<<12)-1)) + (PAtext_pal[screen] << 12))

Ecrire une lettre à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

x Coordonnée X en TILES (0-31) où afficher la lettre

y Coordonnée Y en TILES (0-19) où afficher la lettre

letter Lettre... 'a', 'Z', etc...

3.25.2.2 #define PA_InitCustomText(screen, bg_select, text) PA_InitCustomTextEx(screen, bg_select, text##_Tiles, text##_Map, text##_Pal)

[DEPRECATED] Initialiser le texte en utilisant une police perso
Obsolète

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond...

text Image de la police, converti avec PAGfx

3.25.2.3 #define PA_ShowFont(screen) PA_LoadBgMap(screen, PAbgtext[screen], (void∗)PA_textmap[screen], BG_256X256)

Affiche la police utilisée. C'est juste utile pour du débuggage, aucun intéret autrement.
Paramètres:

screen Choix de l'écran (0 ou 1)

3.25.2.4 #define PA_8bitCustomFont(bit8_slot, bit8_font)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    bittext_maps[bit8_slot] = (u16*)(void*)bit8_font##_Map; \
    bit8_tiles[bit8_slot] = (u8*)bit8_font##_Tiles; \
    pa_bittextdefaultsize[bit8_slot] = (u8*)bit8_font##_Sizes;  \
    pa_bittextpoliceheight[bit8_slot] = bit8_font##_Height;\
}while(0)
```

[DEPRECATED] Ajouter une police perso dans le systeme de texte 8bit ! ! Doit être convertie avec PAGfx
Obsolète

Paramètres:

bit8_slot Slot pour ajouter la police. Les slots 0-4 sont utilisés pour les polices par défaut de PAlib, et 5-9 sont libres. On peut néanmoins charger par-dessus les polices PAlib si on veut

bit8_font Nom de la police...

3.25.3 Documentation des fonctions

3.25.3.1 void PA_LoadDefaultText (u8 *screen*, u8 *bg_select*)

Charger et initialiser le texte. Ne marche qu'en modes 0-2.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond que l'on veut tourner (0-3)

Exemples :

Backgrounds/Effects/Mode7/source/main.c, et **Text/Normal/HelloWorld/-source/main.c**.

3.25.3.2 static inline void PA_SetTextTileCol (u8 *screen*, u8 *color*) [inline, static]

Change la couleur du texte à écrire (ne change pas la couleur du texte déjà écrit).

Paramètres:

screen Choix de l'écran (0 ou 1)

color Couleur de 0 à 6, suffit de tester pour voir le résultat :)

3.25.3.3 void PA_OutputText (u8 *screen*, u16 *x*, u16 *y*, const char * *text*, ...)

Ecrire du texte à l'écran. Ne marche qu'en modes 0-2.

Paramètres:

screen Choix de l'écran (0 ou 1)

x Coordonnée X en TILES (0-31) où commencer à afficher le text

y Coordonnée Y en TILES (0-19) où commencer à afficher le text

text Chaine de caractère à écrire. On dispose des commandes suivantes : %s pour une autre chaine de caractères, %d pour écrire la valeur d'une variables, %fX pour afficher un nombre avec X chiffres après la virgule, \n pour aller à la ligne. Voici un exemple : PA_OutputText(0, 0, 1, "Mon nom est %s et je n'ai que %d dents...", "Mollusk", 20) ;

Exemples :

Backgrounds/Effects/Mode7/source/main.c.

3.25.3.4 u16 PA_OutputSimpleText (u8 *screen*, u16 *x*, u16 *y*, const char * *text*)

Ecrire du texte tout simple à l'écran. Ne marche qu'en modes 0-2. Beaucoup plus rapide que PA_OutputText, masi aussi beaucoup plus limité... Renvoie le nombre de lettres.
Paramètres:

screen Choix de l'écran (0 ou 1)

x Coordonnée X en TILES (0-31) où commencer à afficher le text

y Coordonnée Y en TILES (0-19) où commencer à afficher le text

text Chaine de caractère à écrire.

Exemples :

Text/Normal/HelloWorld/source/main.c.

3.25.3.5 u32 PA_BoxText (u8 *screen*, u16 *basex*, u16 *basey*, u16 *maxx*, u16 *maxy*, const char * *text*, u32 *limit*)

Permet d'écrire du texte à l'écran, dans une boite délimitée au choix, et en choisissant le nombre de lettres à afficher (peut être utile pour afficher du texte en train de se taper, sinon suffit de mettre 10000 pour afficher tout d'un coup) Renvoie le nmobre de lettre écrites.
Paramètres:

screen Choix de l'écran (0 ou 1)

basex Coordonnée X en TILES (0-31) où commencer à afficher le text

basey Coordonnée Y en TILES (0-19) où commencer à afficher le text

maxx Coordonnée X en TILES (0-31) où finir d'afficher le text

maxy Coordonnée Y en TILES (0-19) où finir d'afficher le text

text Chaine de caractère à écrire.

limit Nombre maximum de lettres à afficher pour ce coup-ci

3.25.3.6 u32 PA_BoxTextNoWrap (u8 *screen*, u16 *basex*, u16 *basey*, u16 *maxx*, u16 *maxy*, const char * *text*, u32 *limit*)

Permet d'écrire du texte à l'écran, dans une boite délimitée au choix, et en choisissant le nombre de lettres à afficher (peut être utile pour afficher du texte en train de se taper, sinon suffit de mettre 10000 pour afficher tout d'un coup) Renvoie le nombre de lettre écrites. Cette fonction coupe les mots...
Paramètres:

screen Choix de l'écran (0 ou 1)

basex Coordonnée X en TILES (0-31) où commencer à afficher le text

basey Coordonnée Y en TILES (0-19) où commencer à afficher le text

maxx Coordonnée X en TILES (0-31) où finir d'afficher le text

maxy Coordonnée Y en TILES (0-19) où finir d'afficher le text

text Chaine de caractère à écrire.

limit Nombre maximum de lettres à afficher pour ce coup-ci

3.25.3.7 static inline void PA_SetTextCol (u8 *screen*, u16 *r*, u16 *g*, u16 *b*) [inline, static]

Changer la couleur de base du texte à l'écran.

Paramètres:

screen Choix de l'écran (0 ou 1)

r Quantité de rouge (0-31)

g Quantité de vert (0-31)

b Quantité de bleu (0-31)

3.25.3.8 void PA_LoadText (u8 *screen*, u8 *bg_select*, const PA_BgStruct * *font*)

Initialiser le texte en utilisant une police perso.

Paramètres:

screen Choix de l'écran (0 ou 1)

bg_select Numéro du fond...

font Pointeur vers la police

3.25.3.9 s16 PA_8bitText (u8 *screen*, s16 *basex*, s16 *basey*, s16 *maxx*, s16 *maxy*, const char * *text*, u8 *color*, u8 *size*, u8 *transp*, s32 *limit*)

Cette fonction permet d'écrire du texte à chasse variable à l'écran. Elle nécessite d'avoir un fond dessinable de 8 bits (cf PA_Init8bitBg). Les options sont la taille, la transparence, et les limites, ainsi que la couleur. Seul inconvénient : il n'accepte pas les commande comme d, etc... La fonction renvoie le nombre de charactères écrits.

Paramètres:

screen Choix de l'écran (0 ou 1)

basex Coordonnée X du coin supérieur gauche

basey Coordonnée Y du coin supérieur gauche

maxx Coordonnée X du coin inférieur droit

maxy Coordonnée Y du coin inférieur droit

text Texte, tel que "Hello World"

color Couleur de la palette à utiliser (0-255)

size Taille du texte, de 0 (vraiment petit) à 4 (assez grand)

transp Transparence. Mettre à 0 effecera tout dessin de la zone de texte. 1 écrira le texte par-dessus le dessin sans l'effacer. 2 n'écrira rien (juste pour compter les lettres). 3 fera un texte tourné à 90°. 4 est un texte tourné dans l'autre sens.

limit On peut fixer une limite au nombre de caractères. Ceci peut etre utile pour dessiner un texte progressivement, en augmentant de 1 le nombre de caractères à chaque boucle....

3.25.3.10 s16 PA_CenterSmartText (u8 *screen*, s16 *basex*, s16 *basey*, s16 *maxx*, s16 *maxy*, const char * *text*, u8 *color*, u8 *size*, u8 *transp*)

En gros la meme chose que SmartText, mais en centré...
Paramètres:

screen Choix de l'écran (0 ou 1)

basex Coordonnée X du coin supérieur gauche

basey Coordonnée Y du coin supérieur gauche

maxx Coordonnée X du coin inférieur droit

maxy Coordonnée Y du coin inférieur droit

text Texte, tel que "Hello World"

color Couleur de la palette à utiliser (0-255)

size Taille du texte, de 0 (vraiment petit) à 4 (assez grand)

transp Transparence. Mettre à 0 effecera tout dessin de la zone de texte. 1 écrira le texte par-dessus le dessin sans l'effacer. 2 n'écrira rien (juste pour compter les lettres). 3 fera un texte tourné à 90°. 4 est un texte tourné dans l'autre sens.

3.25.3.11 void PA_AddBitmapFont (int *slot*, const PA_BgStruct * *font*)

Ajouter une police perso dans le système de texte 8bit/16bit.
Paramètres:

slot Slot pour ajouter la police. Les slots 0-4 sont utilisés pour les polices par défaut de PAlib, et 5-9 sont libres. On peut néanmoins charger par-dessus les polices PAlib si on veut.

font Pointeur vers le police perso.

3.25.3.12 void PA_InitTextBorders (u8 *screen*, u8 *x1*, u8 *y1*, u8 *x2*, u8 *y2*)

Initialise une boite à texte, avec la bordure. Ceci rend l'utilisation des textes délimités bien plus simple.
Paramètres:

screen Choix de l'écran (0 ou 1)

x1 Limite gauche en tiles

y1 Haut

x2 Droite

y2 Bas

3.25.3.13 void PA_EraseTextBox (u8 *screen*)

Efface le text d'un boite à texte... Nécessite qu'il ait été initialisé avec PA_InitTextBorders.
Paramètres:

screen Choix de l'écran (0 ou 1)

3.25.3.14 static inline u32 PA_SimpleBoxText (u8 *screen*, const char * *text*, u32 *limit*) [inline, static]

Ecrit du texte dans une zone délimitée. Similaire à PA_BoxText, mais sans avoir besoin de délimiter.

Paramètres:

screen Choix de l'écran (0 ou 1)

text Chaine de caractère à écrire.

limit Nombre maximum de lettres à afficher pour ce coup-ci

3.25.3.15 void PA_ClearTextBg (u8 *screen*)

Effacer tout le texte sur un écran donné.

Paramètres:

screen Choix de l'écran (0 ou 1)

3.25.3.16 void PA_Print (u8 *screen*, const char * *text*, ...)

Ecrire du texte à l'écran. Marche comme la fonction printf.

Paramètres:

screen Choix de l'écran (0 ou 1)

text Chaine de caractère à écrire. On dispose des commandes suivantes : %s pour une autre chaine de caractères, %d pour écrire la valeur d'une variables, %fX pour afficher un nombre avec X chiffres après la virgule, \n pour aller à la ligne. Voici un exemple : PA_OutputText(0, 0, 1, "Mon nom est %s et je n'ai que %d dents...", "Mollusk", 20) ;

3.25.3.17 static inline void PA_PrintLetter (u8 *screen*, char *letter*) [inline, static]

Comme PA_Print, mais juste pour une lettre.

Paramètres:

screen Choix de l'écran (0 ou 1)

letter Une lettre...

3.26 Bg Modes on 2 Screens

Macros

– #define **PA_DualLoadTiledBg**(bg_number, bg_name)
[DEPRECATED] On ne pourra jamais rendre ca plus simple... Charge un fond de type TiledBg converti avec PAGfx, en mettant les tiles, la map, et meme la palette ! Seulement en mode 256 couleurs. Sur 2 écrans, comme un seul grand

– #define **PA_DualLoadSimpleBg**(bg_select, bg_tiles, bg_map, bg_size, wraparound, color_mode)
[DEPRECATED] Facon la plus simple de cahrger un fond sur les 2 écrans

– #define **PA_DualLoadRotBg**(bg_select, bg_tiles, bg_map, bg_size, wraparound)
[DEPRECATED] Charger un fond pour les rotations/zoom ! Attention, il faut avant utiliser PA_SetVideoMode avec 1 pour utiliser un fond rotatif (le fond 3 uniquement !), ou 2 pour 2 fonds (2 et 3). Le fond DOIT etre de 256 couleurs

– #define **PA_DualLoadBg**(bg_select, bg_tiles, tile_size, bg_map, bg_size, wraparound, color_mode)
[DEPRECATED] Facon la plus simple de cahrger un fond. Combine PA_InitBg, PA_LoadBgTiles, et PA_LoadBgMap

– #define **PA_DualLoadPAGfxLargeBg**(bg_number, bg_name)
[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), converti avec PAGfx. Fond sur les 2 écrans comme un seul

– #define **PA_DualLoadLargeBg**(bg_select, bg_tiles, bg_map, color_mode, lx, ly)
[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), sur les 2 écrans

– #define **PA_DualLoadLargeBgEx**(bg_select, bg_tiles, tile_size, bg_map, color_mode, lx, ly)
[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), mais ici on met soi-meme la taille des tiles

– #define **PA_DualEasyBgLoad**(bg_number, bg_name)
[DEPRECATED] Chargement de fond EasyBg, mais pour le Dual Screen...

Fonctions

– static void **PA_DualHideBg** (u8 bg_select)
Cacher un fond sur les 2 écrans.

– static void **PA_DualShowBg** (u8 bg_select)
Afficher un fond auparavant caché sur les 2 écrans.

– static void **PA_DualResetBg** (void)
Reinitialiser les fonds d'un écran. En fait ca ne fait que cacher tous les fonds.

– static void **PA_DualDeleteBg** (u8 bg_select)
Effacer un fond complètement (tiles + map + cacher).

– static void **PA_DualBGScrollX** (u8 bg_number, s16 x)
Scroll horizontal de n'importe quel fond, sur les 2 écrans.

– static void **PA_DualBGScrollY** (u8 bg_number, s16 y)
Scroll vertical de n'importe quel fond.

– static void **PA_DualBGScrollXY** (u8 bg_number, s16 x, s16 y)
Scroll horizontal et vertical de n'importe quel fond.

– static void **PA_DualEasyBgScrollX** (u8 bg_select, s32 x)
Déplacer un fond EasyBg horizontalement. Doit etre initialisé avec PA_LoadLargeBg.

– static void **PA_DualEasyBgScrollY** (u8 bg_select, s32 y)
Déplacer un fond EasyBg verticalement.

– static void **PA_DualLoadBackground** (u8 bg_number, const **PA_BgStruct** ∗bg)
Charger un fond (EasyBg, RotBg or UnlimitedBg), mais pour le Dual Screen...

– static void **PA_DualEasyBgScrollXY** (u8 bg_select, s32 x, s32 y)
Déplacer un fond EasyBg en Dual Screen.

– static void **PA_DualInfLargeScrollX** (u8 bg_select, s32 x)
Déplacer un fond à scrolling 'infini' horizontalement. Doit etre initialisé avec PA_-LoadLargeBg.

– static void **PA_DualInfLargeScrollY** (u8 bg_select, s32 y)
Déplacer un fond à scrolling 'infini' verticalement. Doit etre initialisé avec PA_-LoadLargeBg.

– static void **PA_DualInfLargeScrollXY** (u8 bg_select, s32 x, s32 y)
Déplacer un fond à scrolling 'infini' horizontalement et verticalement. Doit etre initialisé avec PA_LoadLargeBg.

– static void **PA_DualLargeScrollX** (u8 bg_select, s32 x)
Déplacer un grand fond à scrolling horizontalement.Doit etre initialisé avec PA_-LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

– static void **PA_DualLargeScrollY** (u8 bg_select, s32 y)
Déplacer un grand fond à scrolling verticalement. Doit etre initialisé avec PA_-LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

– static void **PA_DualLargeScrollXY** (u8 bg_select, s32 x, s32 y)
Déplacer un grand fond à scrolling horizontalement et verticalement. Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

– static void **PA_DualInitParallaxX** (s32 bg0, s32 bg1, s32 bg2, s32 bg3)
Initialiser le Parallax Scrolling pour plusieurs fonds, horizontalement. Choix de la vitesse à laquelle les fonds vont défiler par rapport aux autres... Utiliser PA_ParallaxScrollX par la suite pour scroller.

– static void **PA_DualInitParallaxY** (s32 bg0, s32 bg1, s32 bg2, s32 bg3)
Initialiser le Parallax Scrolling pour plusieurs fonds, horizontalement. Choix de la vitesse à laquelle les fonds vont défiler par rapport aux autres... Utiliser PA_ParallaxScrollX par

la suite pour scroller.

– static void **PA_DualParallaxScrollX** (s32 x)
Déplacer les fonds activés pour le parallax...

– static void **PA_DualParallaxScrollY** (s32 y)
Déplacer les fonds activés pour le parallax...

– static void **PA_DualParallaxScrollXY** (s32 x, s32 y)
Déplacer les fonds activés pour le parallax...

– static void **PA_DualSetBgPrio** (u8 bg, u8 prio)
Changer la priorité d'un fond.

3.26.1 Description détaillée

Load tiles, a map, scroll it... and 2 screens automatically

3.26.2 Documentation des macros

3.26.2.1 #define PA_DualLoadTiledBg(bg_number, bg_name)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadTiledBg(0, bg_number, bg_name);\
    PA_LoadTiledBg(1, bg_number, bg_name);\
    PA_DualBGScrollY(bg_number, 0);}while(0)
```

[DEPRECATED] On ne pourra jamais rendre ca plus simple... Charge un fond de type TiledBg converti avec PAGfx, en mettant les tiles, la map, et meme la palette ! Seulement en mode 256 couleurs. Sur 2 écrans, comme un seul grand

Obsolète

Paramètres:

bg_number Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_name Nom du fond, comme bg0

3.26.2.2 #define PA_DualLoadSimpleBg(bg_select, bg_tiles, bg_map, bg_size, wraparound, color_mode)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
```

```
PA_LoadSimpleBg(0, bg_select, bg_tiles, bg_map, bg_size, wraparound, color_mo
  de);\
PA_LoadSimpleBg(1, bg_select, bg_tiles, bg_map, bg_size, wraparound, color_mo
  de);\
PA_DualBGScrollY(bg_select, 0);}while(0)
```

[DEPRECATED] Facon la plus simple de cahrger un fond sur les 2 écrans

Obsolète

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

bg_size Taille du fond. Pour un fond normal, on utilise les macros BG_256X256, BG_256X512, etc...

wraparound Si le fond boucle ou non. C'est plus important pour les fonds rotatifs...

color_mode Nombre de couleurs : 0 pour 16 couleurs, 1 pour 256

3.26.2.3 #define PA_DualLoadRotBg(bg_select, bg_tiles, bg_map, bg_size, wraparound)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadRotBg(0, bg_select, bg_tiles, bg_map, bg_size, wraparound);\
    PA_LoadRotBg(1, bg_select, bg_tiles, bg_map, bg_size, wraparound);\
    PA_DualBGScrollY(bg_select, 0);}while(0)
```

[DEPRECATED] Charger un fond pour les rotations/zoom ! Attention, il faut avant utiliser PA_SetVideoMode avec 1 pour utiliser un fond rotatif (le fond 3 uniquement !), ou 2 pour 2 fonds (2 et 3). Le fond DOIT etre de 256 couleurs

Obsolète

Paramètres:

bg_select Numéro du fond que l'on veut charger

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

bg_size Taille du fond. Utiliser les macros suivantes : BG_ROT_128X128, ou 256X256, 512X512, ou enfin 1024X1024

wraparound Si le fond boucle ou non.

3.26.2.4 #define PA_DualLoadBg(bg_select, bg_tiles, tile_size, bg_map, bg_size, wraparound, color_mode)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadBg(0, bg_select, bg_tiles, tile_size, bg_map, bg_size, wraparound, col
      or_mode);\
    PA_LoadBg(1, bg_select, bg_tiles, tile_size, bg_map, bg_size, wraparound, col
      or_mode);\
    PA_DualBGScrollY(bg_select, 0);}while(0)
```

[DEPRECATED] Facon la plus simple de cahrger un fond. Combine PA_InitBg, PA_-LoadBgTiles, et PA_LoadBgMap

Obsolète

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

tile_size Taille du tilset

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

bg_size Taille du fond. Ceci est très important, car ça détermine aussi si le Bg est rotatif ou non. Pour un fond normal, on utilise les macros BG_-256X256, BG_256X512, etc... Por un fond rotatif, il suffit d'utiliser BG_-ROT_128X128...

wraparound Si le fond boucle ou non. C'est plus important pour les fonds rotatifs...

color_mode Nombre de couleurs : 0 pour 16 couleurs, 1 pour 256

3.26.2.5 #define PA_DualLoadPAGfxLargeBg(bg_number, bg_name)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadPAGfxLargeBg(0, bg_number, bg_name);\
    PA_LoadPAGfxLargeBg(1, bg_number, bg_name);\
    PA_DualInfLargeScrollY(bg_number, 0);}while(0)
```

[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), converti avec PAGfx. Fond sur les 2 écrans comme un seul

Obsolète

Paramètres:

bg_number Numéro du fond que l'on veut charger (0-3)

bg_name Nom du fond dans PAGfx

3.26.2.6 #define PA_DualLoadLargeBg(bg_select, bg_tiles, bg_map, color_mode, lx, ly)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadLargeBg(0, bg_select, bg_tiles, bg_map, color_mode, lx, ly);\
    PA_LoadLargeBg(1, bg_select, bg_tiles, bg_map, color_mode, lx, ly);\
    PA_DualInfLargeScrollY(bg_select, 0);}while(0)
```

[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), sur les 2 écrans

Obsolète

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

color_mode Nombre de couleurs : 0 pour 16 couleurs, 1 pour 256

lx Largeur, en tiles. Un fond de 512 pixels de large fera 64 tiles de large.

ly Hauteur, en tiles. Un fond de 512 pixels de hauy fera 64 tiles de haut.

3.26.2.7 #define PA_DualLoadLargeBgEx(bg_select, bg_tiles, tile_size, bg_map, color_mode, lx, ly)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_LoadLargeBgEx(0, bg_select, bg_tiles, tile_size, bg_map, color_mode, lx, l
      y);\
    PA_LoadLargeBgEx(1, bg_select, bg_tiles, tile_size, bg_map, color_mode, lx, l
      y);\
    PA_DualInfLargeScrollY(bg_select, 0);}while(0)
```

[DEPRECATED] Charger et initialiser un fond pour le scrolling infini (pour les fonds de plus de 512 pixels de haut ou de large), mais ici on met soi-meme la taille des tiles

Obsolète

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

bg_tiles Nom du tableau contenant les tiles (exemple : ship_Tiles)

tile_size Taille du tilset

bg_map Nom du tableau contenant les infos sur la map (exemple : ship_Map)

color_mode Nombre de couleurs : 0 pour 16 couleurs, 1 pour 256

lx Largeur, en tiles. Un fond de 512 pixels de large fera 64 tiles de large.

ly Hauteur, en tiles. Un fond de 512 pixels de hauy fera 64 tiles de haut.

3.26.2.8 #define PA_DualEasyBgLoad(bg_number, bg_name)

Valeur :

```
do{\
    PA_DEPRECATED_MACRO;\
    PA_EasyBgLoad(0, bg_number, bg_name);\
    PA_EasyBgLoad(1, bg_number, bg_name);\
    PA_DualEasyBgScrollY(bg_number, 0);}while(0)
```

[DEPRECATED] Chargement de fond EasyBg, mais pour le Dual Screen...
Obsolète

Paramètres:

bg_number Numéro du fond que l'on veut charger (0-3)

bg_name Nom du fond dans PAGfx

3.26.3 Documentation des fonctions

3.26.3.1 static inline void PA_DualHideBg (u8 *bg_select*) [inline, static]

Cacher un fond sur les 2 écrans.
Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

3.26.3.2 static inline void PA_DualShowBg (u8 *bg_select*) [inline, static]

Afficher un fond auparavant caché sur les 2 écrans.
Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

3.26.3.3 static inline void PA_DualDeleteBg (u8 *bg_select*) [inline, static]

Effacer un fond complètement (tiles + map + cacher).

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

3.26.3.4 static inline void PA_DualBGScrollX (u8 *bg_number*, s16 *x*) [inline, static]

Scroll horizontal de n'importe quel fond, sur les 2 écrans.

Paramètres:

bg_number Numéro du fond que l'on veut tourner (0-3)

x Valeur X à déplacer, horizontalement...

3.26.3.5 static inline void PA_DualBGScrollY (u8 *bg_number*, s16 *y*) [inline, static]

Scroll vertical de n'importe quel fond.

Paramètres:

bg_number Numéro du fond que l'on veut tourner (0-3)

y Valeur Y à déplacer, verticalement...

3.26.3.6 static inline void PA_DualBGScrollXY (u8 *bg_number*, s16 *x*, s16 *y*) [inline, static]

Scroll horizontal et vertical de n'importe quel fond.

Paramètres:

bg_number Numéro du fond que l'on veut tourner (0-3)

x Valeur X à déplacer, horizontalement...

y Valeur Y à déplacer, verticalement...

3.26.3.7 static inline void PA_DualEasyBgScrollX (u8 *bg_select*, s32 *x*) [inline, static]

Déplacer un fond EasyBg horizontalement. Doit etre initialisé avec PA_LoadLargeBg.

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

3.26.3.8 static inline void PA_DualEasyBgScrollY (u8 *bg_select*, s32 *y*) [inline, static]

Déplacer un fond EasyBg verticalement.

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

y Valeur Y à déplacer

3.26.3.9 static inline void PA_DualLoadBackground (u8 *bg_number*, const PA_BgStruct * *bg*) [inline, static]

Charger un fond (EasyBg, RotBg or UnlimitedBg), mais pour le Dual Screen...

Paramètres:

bg_number Numéro du fond que l'on veut charger (0-3)

bg Pointeur vers le fond

3.26.3.10 static inline void PA_DualEasyBgScrollXY (u8 *bg_select*, s32 *x*, s32 *y*) [inline, static]

Déplacer un fond EasyBg en Dual Screen.

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

y Valeur Y à déplacer

3.26.3.11 static inline void PA_DualInfLargeScrollX (u8 *bg_select*, s32 *x*) [inline, static]

Déplacer un fond à scrolling 'infini' horizontalement. Doit etre initialisé avec PA_-LoadLargeBg.

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

3.26.3.12 static inline void PA_DualInfLargeScrollY (u8 *bg_select*, s32 *y*) [inline, static]

Déplacer un fond à scrolling 'infini' verticalement. Doit etre initialisé avec PA_-LoadLargeBg.

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

y Valeur Y à déplacer

3.26.3.13 static inline void PA_DualInfLargeScrollXY (u8 *bg_select*, s32 *x*, s32 *y*) [inline, static]

Déplacer un fond à scrolling 'infini' horizontalement et verticalement. Doit etre initialisé avec PA_LoadLargeBg.

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

y Valeur Y à déplacer

3.26.3.14 static inline void PA_DualLargeScrollX (u8 *bg_select*, s32 *x*) [inline, static]

Déplacer un grand fond à scrolling horizontalement.Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

3.26.3.15 static inline void PA_DualLargeScrollY (u8 *bg_select*, s32 *y*) [inline, static]

Déplacer un grand fond à scrolling verticalement. Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

y Valeur Y à déplacer

3.26.3.16 static inline void PA_DualLargeScrollXY (u8 *bg_select*, s32 *x*, s32 *y*) [inline, static]

Déplacer un grand fond à scrolling horizontalement et verticalement. Doit etre initialisé avec PA_LoadLargeBg. Cette fonction ne permet pas au fond de 'boucler' sur lui-meme, mais est bien plus rapide que InfLargeScroll...

Paramètres:

bg_select Numéro du fond que l'on veut charger (de 0 à 3 en mode 0, uniquement 2 et 3 en mode 2)

x Valeur X à déplacer

y Valeur Y à déplacer

3.26.3.17 static inline void PA_DualInitParallaxX (s32 *bg0*, s32 *bg1*, s32 *bg2*, s32 *bg3*) [inline, static]

Initialiser le Parallax Scrolling pour plusieurs fonds, horizontalement. Choix de la vitesse à laquelle les fonds vont défiler par rapport aux autres... Utiliser PA_-ParallaxScrollX par la suite pour scroller.

Paramètres:

bg0 Valeur pour le premier fond (0). 256 met en vitesse normal, moins pour moins lent (128 pour moitié de vitesse), plus pour plus rapide (512 équivaut à 2 fois plus vite). On peut utiliser des valeurs négatives. 0 désactive le scrolling parallax pour ce fond

bg1 Idem, pour le Fond 1

bg2 Idem, pour le Fond 2

bg3 Idem, pour le Fond 3

3.26.3.18 static inline void PA_DualInitParallaxY (s32 *bg0*, s32 *bg1*, s32 *bg2*, s32 *bg3*) [inline, static]

Initialiser le Parallax Scrolling pour plusieurs fonds, horizontalement. Choix de la vitesse à laquelle les fonds vont défiler par rapport aux autres... Utiliser PA_-ParallaxScrollX par la suite pour scroller.

Paramètres:

bg0 Valeur pour le premier fond (0). 256 met en vitesse normal, moins pour moins lent (128 pour moitié de vitesse), plus pour plus rapide (512 équivaut à 2 fois plus vite). On peut utiliser des valeurs négatives. 0 désactive le scrolling parallax pour ce fond

bg1 Idem, pour le Fond 1

bg2 Idem, pour le Fond 2

bg3 Idem, pour le Fond 3

3.26.3.19 static inline void PA_DualParallaxScrollX (s32 *x*) [inline, static]

Déplacer les fonds activés pour le parallax...

Paramètres:

x Valeur X à déplacer

3.26.3.20 static inline void PA_DualParallaxScrollY (s32 *y*) [inline, static]

Déplacer les fonds activés pour le parallax...

Paramètres:

y Valeur Y à déplacer

3.26.3.21 static inline void PA_DualParallaxScrollXY (s32 *x*, s32 *y*) [inline, static]

Déplacer les fonds activés pour le parallax...

Paramètres:

x Valeur X à déplacer

y Valeur Y à déplacer

3.26.3.22 static inline void PA_DualSetBgPrio (u8 *bg*, u8 *prio*) [inline, static]

Changer la priorité d'un fond.

Paramètres:

bg Numéro du fond...

prio Niveau de priorité, de 0 à 3, 0 étant priorité la plus élevée

3.27 Window system

Macros

– #define **PA_SetWin1XY**(screen, x1, y1, x2, y2) do{WIN1X(screen) = x2 + ((x1) << 8) ; WIN1Y(screen) = y2 + ((y1) << 8) ;}while(0)
Régler les coordonnées X et Y du rectangle de la deuxième fenetre. Il faudra aussi utiliser PA_SetWin1 pour choisir quels fonds sont visibles et si les sprites le sont ou non...

– #define **PA_EnableWin0**(screen, bg_sprites) do{DISPCNTL(screen) |= WINDOW0 ; WININ(screen) &= 255 ; WININ(screen) |= (bg_sprites) ;}while(0)
Activer et choisir quels fonds seront visibles dans la fenetre, et si les sprites le seront ou pas, pour la fenetre 0. Il faudra ensuite régler la taille avec PA_SetWin0XY.

– #define **PA_DisableWin0**(screen) DISPCNTL(screen) &= ∼WINDOW0
Désactiver la première fenetre...

– #define **PA_EnableWin1**(screen, bg_sprites) do{DISPCNTL(screen) |= WINDOW1 ; WININ(screen) &= 255 ; WININ(screen) |= ((bg_sprites) << 8) ;}while(0)
Activer et choisir quels fonds seront visibles dans la fenetre, et si les sprites le seront ou pas, pour la fenetre 1. Il faudra ensuite régler la taille avec PA_SetWin1X.

– #define **PA_DisableWin1**(screen) DISPCNTL(screen) &= ∼WINDOW1
Désactiver la deuxième fenetre...

– #define **PA_DisableWinObj**(screen) DISPCNTL(screen) &= ∼WINDOWOBJ
Désactiver la fenetre objet...

– #define **PA_SetOutWin**(screen, bg_sprites) do{WINOUT(screen) &= ∼255 ; WINOUT(screen) |= bg_sprites ;}while(0)
Choisir quels fonds seront visibles dans la fenetre, et si les sprites le seront ou pas, en dehors des fenetres.

Fonctions

– static void **PA_EnableWinObj** (u8 screen, u16 bg_sprites)
Activer et choisir quels fonds seront visibles dans la fenetre, et si les sprites le seront ou pas, pour la fenetre objets (créée à partir des sprites en mode Window).

– static void **PA_WindowFade** (u8 screen, u8 type, u8 time)
Permet de faire des entrées/sorties en utilisant le systeme de fenetre.

3.27.1 Description détaillée

Set up 2 windows and a possible object window...

3.27.2 Documentation des macros

3.27.2.1 #define PA_SetWin1XY(screen, x1, y1, x2, y2) do{WIN1X(screen) = x2 + ((x1) << 8) ; WIN1Y(screen) = y2 + ((y1) << 8) ;}while(0)

Régler les coordonnées X et Y du rectangle de la deuxième fenetre. Il faudra aussi utiliser PA_SetWin1 pour choisir quels fonds sont visibles et si les sprites le sont ou non...

Paramètres:

screen Ecran...

x1 Coordonnée X du coin supérieur gauche

y1 Coordonnée Y du coin supérieur gauche

x2 Coordonnée X du coin inférieur droit

y2 Coordonnée Y du coin inférieur droit

3.27.2.2 #define PA_EnableWin0(screen, bg_sprites) do{DISPCNTL(screen) |= WINDOW0 ; WININ(screen) &= 255 ; WININ(screen) |= (bg_sprites) ;}while(0)

Activer et choisir quels fonds seront visibles dans la fenetre, et si les sprites le seront ou pas, pour la fenetre 0. Il faudra ensuite régler la taille avec PA_SetWin0XY.

Paramètres:

screen Ecran...

bg_sprites Fond et sprites à afficher, que l'on choisi de la facon suivante : WIN_BG0 | WIN_BG1 | WIN_BG2 | WIN_BG3 | WIN_OBJ | WIN_SFX (pour les effets spéciaux)

3.27.2.3 #define PA_DisableWin0(screen) DISPCNTL(screen) &= ∼WINDOW0

Désactiver la première fenetre...

Paramètres:

screen Ecran...

3.27.2.4 #define PA_EnableWin1(screen, bg_sprites) do{DISPCNTL(screen) |= WINDOW1 ; WININ(screen) &= 255 ; WININ(screen) |= ((bg_sprites) << 8) ;}while(0)

Activer et choisir quels fonds seront visibles dans la fenetre, et si les sprites le seront ou pas, pour la fenetre 1. Il faudra ensuite régler la taille avec PA_SetWin1X.

Paramètres:

screen Ecran...

bg_sprites Fond et sprites à afficher, que l'on choisi de la facon suivante : WIN_BG0 | WIN_BG1 | WIN_BG2 | WIN_BG3 | WIN_OBJ | WIN_SFX (pour les effets spéciaux)

3.27.2.5 #define PA_DisableWin1(screen) DISPCNTL(screen) &= ∼WINDOW1

Désactiver la deuxième fenetre...
Paramètres:

screen Ecran...

3.27.2.6 #define PA_DisableWinObj(screen) DISPCNTL(screen) &= ∼WINDOWOBJ

Désactiver la fenetre objet...
Paramètres:

screen Ecran...

3.27.2.7 #define PA_SetOutWin(screen, bg_sprites) do{WINOUT(screen) &= ∼255 ; WINOUT(screen) |= bg_sprites ;}while(0)

Choisir quels fonds seront visibles dans la fenetre, et si les sprites le seront ou pas, en dehors des fenetres.
Paramètres:

screen Ecran...

bg_sprites Fond et sprites à afficher, que l'on choisi de la facon suivante : WIN_BG0 | WIN_BG1 | WIN_BG2 | WIN_BG3 | WIN_OBJ

3.27.3 Documentation des fonctions

3.27.3.1 static inline void PA_EnableWinObj (u8 *screen*, u16 *bg_sprites*) [inline, static]

Activer et choisir quels fonds seront visibles dans la fenetre, et si les sprites le seront ou pas, pour la fenetre objets (créée à partir des sprites en mode Window).
Paramètres:

screen Ecran...

bg_sprites Fond et sprites à afficher, que l'on choisi de la facon suivante : WIN_BG0 | WIN_BG1 | WIN_BG2 | WIN_BG3 | WIN_OBJ | WIN_SFX (pour les effets spéciaux)

3.27.3.2 static inline void PA_WindowFade (u8 *screen*, u8 *type*, u8 *time*) [inline, static]

Permet de faire des entrées/sorties en utilisant le systeme de fenetre.
Paramètres:

screen Ecran...

type Type... 8 différents types sont disponibles (0-7)

time Temps, de 0 à 32 (inclus). 0 est écran visisble, 32 invisible

3.28 C++ wrappers

Espaces de nommage

– namespace **PA**
PAlib C++ namespace.

Fonctions

– void ∗ **operator new** (size_t size)
Lightweight new operator.

– void ∗ **operator new[ ]** (size_t size)
Lightweight new operator.

– void **operator delete** (void ∗p)
Lightweight delete operator.

– void **operator delete[ ]** (void ∗p)
Lightweight delete operator.

3.28.1 Description détaillée

C++ wrappers for PAlib

3.29 ASlib functions

Structures de données

– struct **SoundInfo**
Sound info.

Macros

– #define **AS_SoundQuickPlay**(name) AS_SoundDefaultPlay((u8∗)name, (u32)name##_size, 127, 64, false, 0)
Easiest way to play a sound, using default settings.

Énumérations

– enum **MP3Command** { ,
MP3CMD_INIT = 8, **MP3CMD_STOP** = 16, **MP3CMD_PLAY** = 32, **MP3CMD_PAUSE** = 64,
MP3CMD_SETRATE = 128 }
MP3 commands.

– enum **SoundCommand** { ,
SNDCMD_STOP = 2, **SNDCMD_PLAY** = 4, **SNDCMD_SETVOLUME** = 8, **SNDCMD_SETPAN** = 16,
SNDCMD_SETRATE = 32, **SNDCMD_SETMASTERVOLUME** = 64 }
Sound commands.

– enum **MP3Status** {
MP3ST_STOPPED = 0, **MP3ST_PLAYING** = 1, **MP3ST_PAUSED** = 2, **MP3ST_OUT_OF_DATA** = 4,
MP3ST_DECODE_ERROR = 8, **MP3ST_INITFAILED** = 16 }
MP3 states.

– enum **AS_MODE** { **AS_MODE_MP3** = 1, **AS_MODE_SURROUND** = 2, **AS_MODE_16CH** = 4, **AS_MODE_8CH** = 8 }
ASlib modes.

– enum **AS_DELAY** { **AS_NO_DELAY** = 0, **AS_SURROUND** = 1, **AS_REVERB** = 4 }
Delay values.

– enum **AS_SOUNDFORMAT** { **AS_PCM_8BIT** = 0, **AS_PCM_16BIT** = 1, **AS_ADPCM** = 2 }
Sound formats.

Fonctions

– void **AS_Init** (u8 mode)

Initialize ASlib.

– static void **AS_ReserveChannel** (u8 channel)
Reserve a particular DS channel (so it won't be used for the sound pool).

– static void **AS_SetMasterVolume** (u8 volume)
Set the master volume (0..127).

– static void **AS_SetDefaultSettings** (u8 format, s32 rate, u8 delay)
Set the default sound settings.

– int **AS_SoundPlay** (**SoundInfo** sound)
Play a sound using the priority system. Returns the sound channel allocated or -1 if the sound was skipped.

– static int **AS_SoundDefaultPlay** (u8 ∗data, u32 size, u8 volume, u8 pan, u8 loop, u8 prio)
Play a sound using the priority system with the default settings. Returns the sound channel allocated or -1 if the sound was skipped.

– void **AS_SetSoundPan** (u8 chan, u8 pan)
Set the panning of a sound (0=left, 64=center, 127=right).

– void **AS_SetSoundVolume** (u8 chan, u8 volume)
Set the volume of a sound (0..127).

– void **AS_SetSoundRate** (u8 chan, u32 rate)
Set the sound sample rate.

– static void **AS_SoundStop** (u8 chan)
Stop playing a sound.

– void **AS_SoundDirectPlay** (u8 chan, **SoundInfo** sound)
Play a sound directly using the given channel.

– void **AS_MP3DirectPlay** (u8 ∗buffer, u32 size)
Play a MP3 directly from memory.

– void **AS_MP3StreamPlay** (const char ∗path)
Play a MP3 stream.

– static void **AS_MP3Pause** ()
Pause a MP3.

– static void **AS_MP3Unpause** ()
Unpause a MP3.

– static void **AS_MP3Stop** ()
Stop a MP3.

– static int **AS_GetMP3Status** ()
Get the current MP3 status.

– static void **AS_SetMP3Volume** (u8 volume)
Set the MP3 volume (0..127).

– void **AS_SetMP3Pan** (u8 pan)
Set the MP3 panning (0=left, 64=center, 127=right).

– static void **AS_SetMP3Delay** (u8 delay)
Set the default MP3 delay mode (warning : high values can cause glitches).

– static void **AS_SetMP3Loop** (u8 loop)
Set the MP3 loop mode (false = one shot, true = loop indefinitely).

– static void **AS_SetMP3Rate** (s32 rate)
Set the MP3 sample rate.

– void **AS_SoundVBL** ()
Regenerate buffers for MP3 stream. Must be called each VBlank (only needed if mp3 is used).

3.29.1 Description détaillée

Functions to play RAW sounds and ∗shrug∗ MP3s.

3.29.2 Documentation du type de l'énumération

3.29.2.1 enum MP3Command

MP3 commands.
Valeurs énumérées :

MP3CMD_INIT Initialize.
MP3CMD_STOP Stop.
MP3CMD_PLAY Play.
MP3CMD_PAUSE Pause.
MP3CMD_SETRATE Set rate.

3.29.2.2 enum SoundCommand

Sound commands.
Valeurs énumérées :

SNDCMD_STOP Stop.
SNDCMD_PLAY Play.
SNDCMD_SETVOLUME Set volume.
SNDCMD_SETPAN Set pan.
SNDCMD_SETRATE Set rate.
SNDCMD_SETMASTERVOLUME Set master volume.

3.29.2.3 enum MP3Status

MP3 states.
Valeurs énumérées :

MP3ST_STOPPED Stopped.

MP3ST_PLAYING Playing.

MP3ST_PAUSED Paused.

MP3ST_OUT_OF_DATA Out of data.

MP3ST_DECODE_ERROR Decoding error.

MP3ST_INITFAILED Initialization failed.

3.29.2.4 enum AS_MODE

ASlib modes.
Valeurs énumérées :

AS_MODE_MP3 use mp3

AS_MODE_SURROUND use surround

AS_MODE_16CH use all DS channels

AS_MODE_8CH use DS channels 1-8 only

3.29.2.5 enum AS_DELAY

Delay values.
Valeurs énumérées :

AS_NO_DELAY 0 ms delay

AS_SURROUND 16 ms delay

AS_REVERB 66 ms delay

3.29.2.6 enum AS_SOUNDFORMAT

Sound formats.
Valeurs énumérées :

AS_PCM_8BIT 8-bit PCM

AS_PCM_16BIT 16-bit PCM

AS_ADPCM 4-bit ADPCM

Chapitre 4

Documentation des espaces de nommage

4.1 Référence de l'espace de nommage PA

PAlib C++ namespace.

Structures de données

- class **Application**
  Simple application abstraction layer for PAlib C++ programs.
- class **Fixed**
  Fixed-point wrapper class.
- class **Point**
  Fixed-point point class.
- class **Sprite**
  Wrapper class for sprites.
- class **HandleProvider**
  Handle provider, use it to get dynamic sprite numbers for example.

4.1.1 Description détaillée

PAlib C++ namespace.

Chapitre 5

Documentation des structures de données

5.1 Référence de la classe PA : :Application

Simple application abstraction layer for PAlib C++ programs.

Fonctions membres publiques

– void **run** ()
  Runs the application.

Fonctions membres protégées

– virtual void **init** ()
  Initialization function.

– virtual bool **update** ()
  Update function.

– virtual void **render** ()
  Render function.

– virtual void **cleanup** ()
  Cleanup function (optional).

5.1.1 Description détaillée

Simple application abstraction layer for PAlib C++ programs.

5.2 Référence de la classe PA : :Fixed

Fixed-point wrapper class.

Fonctions membres publiques

- **Fixed** ()
  Empty constructor.
- **Fixed** (const **Fixed** &a)
  Copy constructor.
- **Fixed** (int a)
  int constructor.
- **Fixed** (float a)
  float constructor.
- **operator int** () const
  int cast.
- **operator float** () const
  float cast.
- **operator bool** () const
  bool cast.
- **operator char** () const
  char cast.
- **operator short** () const
  short cast.
- **operator long long** () const
  long long cast.
- **Fixed** & **operator=** (const **Fixed** &a)
  Assignment operator.
- **Fixed operator+** (const **Fixed** &a) const
  Addition operator. int and float versions also available.
- **Fixed operator-** (const **Fixed** &a) const
  Subtraction operator. int and float versions also available.
- **Fixed operator**$*$ (const **Fixed** &a) const
  Multiplication operator. int and float versions also available.
- **Fixed operator/** (const **Fixed** &a) const
  Division operator. int and float versions also available.

– **Fixed operator%** (const **Fixed** &a) const
Modulo operator. int and float versions also available.

– **Fixed operator++** ()
Pre-increment operator.

– **Fixed operator--** ()
Pre-decrement operator.

– **Fixed operator++** (int)
Post-increment operator.

– **Fixed operator--** (int)
Post-decrement operator.

– **Fixed operator-** () const
Negation operator.

– **Fixed operator**$\sim$ () const
Binary negation operator.

– **Fixed** & **operator+=** (const **Fixed** &a)
Addition and assignment operator. int and float versions also available.

– **Fixed** & **operator-=** (const **Fixed** &a)
Subtraction and assignment operator. int and float versions also available.

– **Fixed** & **operator**$*$**=** (const **Fixed** &a)
Multiplication and assignment operator. int and float versions also available.

– **Fixed** & **operator%=** (const **Fixed** &a)
Modulo and assignment operator. int and float versions also available.

– bool **operator==** (const **Fixed** &a) const
Equals operator. int and float versions also available.

– bool **operator !=** (const **Fixed** &a) const
Not-equals operator. int and float versions also available.

– bool **operator**$<$**=** (const **Fixed** &a) const
Less-or-equal operator. int and float versions also available.

– bool **operator**$>$**=** (const **Fixed** &a) const
Greater-or-equal operator. int and float versions also available.

– bool **operator**$<$ (const **Fixed** &a) const
Less-than operator. int and float versions also available.

– bool **operator**$>$ (const **Fixed** &a) const
Greater-than operator. int and float versions also available.

– **Fixed operator**$<<$ (int a) const

Left shift operator.

– **Fixed operator**>> (int a) const
Right shift operator.

– **Fixed** & **operator**<<**=** (int a)
Left shift and assign operator.

– **Fixed** & **operator**>>**=** (int a)
Right shift and assign operator.

– **Fixed operator&** (u32 a) const
Binary AND operator, u32 version.

– **Fixed operator**| (u32 a) const
Binary OR operator, u32 version.

– **Fixed operator**^ (u32 a) const
Binary XOR operator, u32 version.

– **Fixed operator&** (const **Fixed** &a) const
*Binary AND operator, **Fixed** (p. 164) version.*

– **Fixed operator**| (const **Fixed** &a) const
*Binary OR operator, **Fixed** (p. 164) version.*

– **Fixed operator**^ (const **Fixed** &a) const
*Binary XOR operator, **Fixed** (p. 164) version.*

– **Fixed** & **operator&=** (u32 a)
Binary AND assignment, u32 version.

– **Fixed** & **operator**|**=** (u32 a)
Binary OR assignment, u32 version.

– **Fixed** & **operator**^**=** (u32 a)
Binary XOR assignment, u32 version.

– **Fixed** & **operator&=** (const **Fixed** &a)
*Binary AND assignment, **Fixed** (p. 164) version.*

– **Fixed** & **operator**|**=** (const **Fixed** &a)
*Binary OR assignment, **Fixed** (p. 164) version.*

– **Fixed** & **operator**^**=** (const **Fixed** &a)
*Binary XOR assignment, **Fixed** (p. 164) version.*

– **Fixed sqrt** () const
Gets the square root.

– **Fixed abs** () const
Gets the absolute value.

– int **raw** () const
Gets the raw Q12 fixed point number.

Fonctions membres publiques statiques

– static **Fixed r2f** (int a)
*Creates a **Fixed** (p. 164) object using a raw Q12 fixed point number.*

5.2.1 Description détaillée

Fixed-point wrapper class.

5.3 Référence de la classe PA : :HandleProvider< NHANDLES > (modèle)

Handle provider, use it to get dynamic sprite numbers for example.

Fonctions membres publiques

- **HandleProvider** ()
  Constructor.
- int **newhandle** ()
  Get a new handle.
- void **deletehandle** (int handle)
  Delete a handle.

5.3.1 Description détaillée

template<int NHANDLES> class PA : :HandleProvider< NHANDLES >

Handle provider, use it to get dynamic sprite numbers for example.

5.4 Référence de la structure PA_BgStruct

Background structure.

Champs de données

– int **BgType**
Type of background.

– int **width**
Width of background in pixels.

– int **height**
Height of background in pixels.

– const void ∗ **BgTiles**
Pointer to background tiles.

– const void ∗ **BgMap**
Pointer to background map.

– size_t **BgTiles_size**
Size of tiles in bytes.

– const void ∗ **BgPalette**
Pointer to palette.

– const void ∗ **FontSizes**
Pointer to font sizes.

– size_t **BgMap_size**
Size of the map in bytes.

– int **FontHeight**
Height of the font in pixels.

5.4.1 Description détaillée

Background structure.

5.5 Référence de la structure PA_FifoMsg

Represents a message sent through Fifo.

Champs de données

– u32 **type**
  Type of message.

– union {
  struct {
    u16 **tdiode1**
      TSC temperature diode 1.
    u16 **tdiode2**
      TSC temperature diode 1.
    u32 **temperature**
      TSC computed temperature.
    u16 **battery**
      TSC battery.
    u8 **micvol**
      Microphone volume.
    u8 **extra**
      Extra byte - used as padding for now.
  } **InputMsg**
    Input message data.
  struct {
    u8 * **buffer**
      Buffer to record microphone data.
    u32 **length**
      Length of the buffer in bytes.
  } **MicMsg**
    Microphone record message data.
  struct {
    u8 **brightness**
      Brightness of the lights (0-3).
  } **DSLBrightMsg**
    DS lite brightness message data.
  struct {
    u32 **freq**
      Frequency (in hertz).
    u8 **chan**
      Channel.
    u8 **vol**
      Volume (0-127).
    u8 **pan**
      Pan (0-64-127).
    u8 **duty**
      Duty (0-7).
  } **PSGMsg**

PSG play message data.
} ;

--

5.5.1 Description détaillée

Represents a message sent through Fifo.

5.6 Référence de la structure PA_Point

Simple point structure.

Champs de données

– int **x**
X value.

– int **y**
Y value.

5.6.1 Description détaillée

Simple point structure.

5.7 Référence de la structure PA_TransferRegion

PAlib transfer region type.

Champs de données

– vuint16 **tdiode1**
TSC temperature diode 1.

– vuint16 **tdiode2**
TSC temperature diode 2.

– vuint32 **temperature**
TSC computed temperature.

– vuint16 **battery**
TSC battery.

– vuint8 **micvol**
Microphone volume.

– vuint8 **extra**
Extra field - used as padding for now.

– LEGACY vuint32 **mailData**
Legacy IPC field.

5.7.1 Description détaillée

PAlib transfer region type.

5.8 Référence de la classe PA : :Point

Fixed-point point class.

Fonctions membres publiques

– **operator PA_Point** () const
*Convert the object to a **PA_Point*** (p. *172*) *structure.*

Champs de données

– **Fixed x**
X value.

– **Fixed y**
Y value.

5.8.1 Description détaillée

Fixed-point point class.

5.9 Référence de la structure SoundInfo

Sound info.

Champs de données

– u8 * **data**
Pointer to data.

– u32 **size**
Size in bytes.

– u8 **format**
Format (see AS_SOUNDFORMAT).

– s32 **rate**
Rate in Hz.

– u8 **volume**
Volume (0-127).

– s8 **pan**
Pan (0-64-127).

– u8 **loop**
Loop (0 or 1).

– u8 **priority**
Priority.

– u8 **delay**
Delay.

5.9.1 Description détaillée

Sound info.

5.10 Référence de la classe PA : :Sprite

Wrapper class for sprites.

Fonctions membres publiques

– **Sprite** ()
Empty constructor.

– **Sprite** (int scr, int sprn)
Normal constructor.

– void **init** (int scr, int sprn)
Initialize function.

– void **create** (void ∗gfx, int shape, int size, int paln)
Create sprite.

– void **create** (u16 gfx, int shape, int size, int paln)
Create sprite from existing GFX.

– void **remove** ()
Delete sprite.

– void **setpalette** (int paln)
Set palette.

– void **setgfx** (int gfxn)
Set GFX.

– void **render** ()
Render (more like update position).

– void **move** (const **Fixed** &x, const **Fixed** &y)
Move (fixed point version).

– void **move** (int x, int y)
Move (integer version).

– void **hflip** (bool flip)
Set HFlip.

– void **vflip** (bool flip)
Set VFlip.

– void **dblsize** (bool dblsize)
Set doublesize.

– void **priority** (int prio)
Set priority.

– void **bindrotset** (int rotset)
Bind rotset.

– void **debindrotset** ()
Debind rotset.

– void **rotate** (int angle)
Rotate.

– void **zoom** (int zx, int zy)
Zoom.

– void **rotozoom** (int angle, int zx, int zy)
Rotate and zoom.

– void **frame** (int frame)
Set frame.

– void **startanim** (int begin, int end, int speed, int animtype=ANIM_LOOP, int ncycles=-1)
Start animation.

– void **pauseanim** (bool pause=true)
Pause animation.

– void **stopanim** ()
Stop animation.

– void **animspeed** (int speed)
Set animation speed.

5.10.1 Description détaillée

Wrapper class for sprites.

Chapitre 6

Documentation des exemples

6.1 Backgrounds/Effects/Mode7/source/main.c

```
// Mode 7 example.

// Includes
#include <PA9.h>

#include "all_gfx.h"

int main(){
    PA_Init();

    PA_SetVideoMode(0, 2);  //screen, mode
    PA_SetVideoMode(1, 2);  //screen, mode

    // Yup, we use the standard bg load function!
    PA_LoadBackground(0, //screen
                      3, // background number
                      &Rot); // background name in PAGfx
    PA_LoadBackground(1, 3, &Rot);

    // Wraparound (!)
    PA_SetBgWrap(0, 3, 1);
    PA_SetBgWrap(1, 3, 1);

    PA_LoadDefaultText(1, 0);

    PA_InitMode7(3);

    u16 angle = 0;
    u16 height = 8192;

    while(true){
        // Change the angle
        angle += Pad.Held.Right - Pad.Held.Left;
        angle &= 511;
        PA_Mode7Angle(angle);

        // Move left/right
        PA_Mode7MoveLeftRight(Pad.Held.A - Pad.Held.Y);
```

```
        // Move Forward/backward
        PA_Mode7MoveForwardBack(Pad.Held.Up - Pad.Held.Down);

        // Height
        height += (Pad.Held.X - Pad.Held.B)<<7;
        PA_Mode7Height(height);

        PA_OutputText(1, 0, 0, "Angle  : %d    ", angle);
        PA_OutputText(1, 0, 1, "Height : %d    ", height);

        PA_WaitForVBL();
    }
}
```

6.2 Text/Normal/HelloWorld/source/main.c

```
// Hello World Program //

// Lines starting with two slashes are ignored by the compiler
// Basically you can use them to comment what are you doing
// In fact, this kind of lines are called comments :P

// Include PAlib so that you can use it
#include <PA9.h>

int main(){
    // Initialize PAlib
    PA_Init();

    // Load the default text font
    PA_LoadDefaultText(1, // Top screen
                       2); // Background #2

    // Write the text "Hello World"
    PA_OutputSimpleText(1, // Top screen
                        1, // X position 1*8 = 8
                        1, // Y position 1*8 = 8
                        "Hello World");

    // Infinite loop to keep the program running
    while(true){
        // Wait until the next frame.
        // The DS runs at 60 frames per second.
        PA_WaitForVBL();
    }
}
```

Index