ENSEEIHT

Étude de la portabilité entre systèmes ARM et réalisation pratique

Rapport de stage

PRISCAL Cédric (enf65@etu.enseeiht.fr)
3ème EN CAMSI

Remerciements :

- Michel Cattoen, mon maître de stage.
- Gregory Estrade, un développeur GameBoy Advance, pour m'avoir assisté dans ce projet.
- Tous les membres des forums qui ont largement testé l'émulateur.
- Le réseau informatique des élèves, Net7, qui m'a permis d'utiliser les logiciels de développement sous Linux.

- Je remercie également les gens que j'aurais oublié dans cette liste...

Sommaire

Introduction

Contrairement à l'univers des PC où les processeurs Intel prédominent, les processeurs de la plupart des terminaux mobiles sont des processeurs ARM. Le secteur des terminaux mobiles est d'ailleurs actuellement en grande expansion avec les téléphones portables, les PDAs et les consoles de jeux. Aussi divers soient ces domaines, les produits deviennent de plus en plus polyvalents, et on peut maintenant trouver des téléphones portables qui peuvent aussi bien servir de console de jeu, d'appareil photo ou même de PDA.

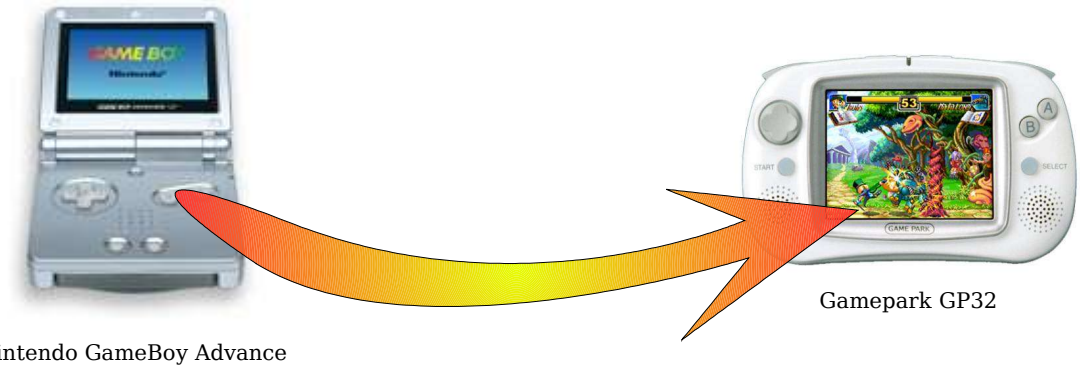
Le monde du jeux vidéo et en particulier les fabriquants de consoles de jeux ont souvent souffert du manque de jeux dès la sortie de leur console sur le marché. Ce problème va maintenant se généraliser à l'ensemble des appareils portatifs, c'est pourquoi les fabriquants doivent faire particulièrement attention à assurer le développement ou l'adaptation de jeux existants sur leur nouvelle console ou téléphone portable.

Sujet du stage

Le développement ou l'adaptation de jeux existants sur une nouvelle console n'étant pas toujours évident et surtout nécessitant un travail personnalisé pour chaque jeu, les fabriquants ont tout intérêt à assurer une compatibilité avec une autre console possédant déjà un large éventail de jeux.

Lorsque la compatibilité n'est pas totale au niveau matériel, il est parfois tout de même possible d'assurer la compatibilité en ajoutant une couche logiciel : un émulateur.

Nous verrons dans ce stage ce qu'est un émulateur et son fonctionnement. Nous étudierons ensuite un moyen d'améliorer les performances d'un émulateur dans un cas particulier : un émulateur Nintendo GameBoy Advance pour une Gamepark GP32, toutes deux étant des consoles portables utilisant un processeur ARM et reflétant assez bien le marché des terminaux mobiles.



Nintendo GameBoy Advance

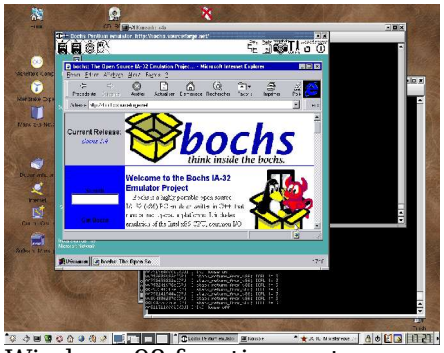
Gamepark GP32

Les émulateurs

Qu'est-ce qu'un émulateur ?

Un émulateur est en quelque sorte un simulateur temps réel. Contrairement à un simulateur qui sert à vérifier le fonctionnement d'un système pendant sa conception, le but d'un émulateur est lui d'utiliser ce système pour son usage normal. Un simulateur est donc en général utilisé pour étudier un système dans le détail, alors qu'un émulateur ne fait apparaître que les éléments qui sont utiles à la fonction du système, comme l'écran si c'est un émulateur de jeux vidéo.

Les émulateurs sont peu connus dans le monde industriel, ce pour des raisons que nous verrons par la suite. Il existe néanmoins des émulateurs comme VMWARE (http://www.vmware.com) ou Bochs (http://bochs.sourceforge.net) qui permettent d'émuler un ordinateur sur un ordinateur ou une station de travail. L'intérêt, s'il n'est pas évident à priori, est de pouvoir disposer de plusieurs ordinateurs virtuels possédant différents systèmes d'exploitation comme s'il s'agissait de simples applications. Cela permet également à des stations de travail Unix de pouvoir utiliser des programmes Windows.

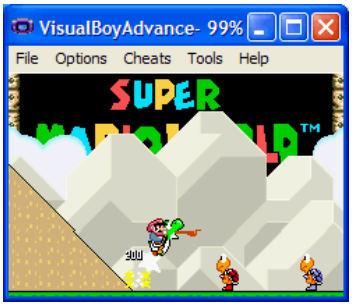


Windows 98 fonctionnant sous Linux


Les consoles 1ère génération.

De la première console de jeux jusqu'aux consoles de l'avant-dernière génération, les jeux étaient toujours contenus dans des cartouches, qui en fait étaient des simples mémoires mortes (ROM). Le coût de fabrication d'une ROM étant bien plus élevé que celui d'un CD ou d'un DVD, les jeux ne dépassaient presque jamais 4 Mo.

Avec le développement des PC, d'Internet, et des émulateurs, et de part la faible taille de ces anciens jeux, ils sont maintenant téléchargeables gratuitement mais illégalement sur Internet. Les jeux n'étant plus en vente, cette pratique a longtemps été tolérée.



Emulateur Game Boy Advance

Avec l'arrivée des derniers téléphones et consoles de jeux portables, l'enjeu des émulateurs est maintenant bien plus grand car tous ces anciens jeux s'avèrent être tout à fait adaptés aux téléphones portables, aussi bien au niveau technique que ludique. Des émulateurs commerciaux sont maintenant développés dans différentes sociétés de développement, mais la législation concernant les émulateurs est encore un champ de mines, et certaines société ont essuyé des procès assez lourd.

Comment fonctionne un émulateur ?

Contrairement à un circuit électronique analogique dans lequel un simulateur fonctionne en appliquant des modèles approchés des composants, les émulateurs s'appliquent eux uniquement à des systèmes informatiques dont le comportement est entièrement connu et défini précisément.

La plupart des émulateurs sont programmés en C ou en C++ et n'utilisent aucun matériel spécifique pour gérer l'émulation. L'avantage de ces émulateurs est qu'ils peuvent être compilés pour n'importe quelle processeur. Par exemple Bochs est un de ces émulateurs et il fonctionne aussi bien sous Windows, sous Linux que sur Macintosh. L'inconvénient est assez important dans notre cas car c'est le microprocesseur qui doit gérer entièrement l'émulation, et il en résulte une perte de perfomances considérable.

Des émulateurs plus performants utilisent un procédé qui en anglais s'appelle “hardware virtualization”, et qui tire parti du fait que le processeur émulé est le même que le processeur de la machine hôte. Ainsi ces émulateurs n'analysent pas chaque instruction du microprocesseur mais l'exécutent directement. Cette méthode est assez difficile à mettre en oeuvre car certaines instructions peuvent provoquer des actions non désirables comme des accès mémoires interdits; il est donc nécessaire de contrôler assez rigoureusement l'exécution de la machine émulée. Le gain en performances est par contre très important et essentiel dans l'émulateur que nous devons réaliser.

La virtualisation

Lorsque la rapidité de l'émulation est primordiale et que la machine à émuler est trop puissante, un émulateur classique va être beaucoup trop lent. Une solution dans le cas où les deux systèmes utilisent le même jeu d'instructions consiste à ne pas émuler les instructions. Ce procédé s'appelle “virtualisation”.

La virtualisation consiste à plonger le processeur de la machine réelle dans le même contexte que la machine émulé, c'est à dire qu'une instruction qui serait exécutée nativement sur le système émulé doit être exécutée de la même manière sur l'autre système. Il est alors nécessaire de distinguer les différents types d'instructions :

- une opération arithmétique : ces opérations sont des calculs élémentaires effectués sur des registres internes au microprocesseur, et leur exécution ne dépend donc pas de ce qui entoure celui-ci.
- un accès mémoire : ces opérations permettent de lire ou d'écrire la mémoire vive, et il faut donc que les plages mémoires correspondent. Cette correspondance sera réalisée grâce à la MMU (Memory Management Unit), qui permet d'interfacer les bus de données et d'adresse du microprocesseur avec les éléments extérieurs comme la mémoire vive. Il permet en particulier d'effectuer des translations d'adresse et ainsi de faire correspondre les plages mémoire de chaque console.
- les accès entrées/sorties : Il n'existe pas d'instructions spécifiques pour faire des accès IO sur les processeurs ARM, en effet les entrées/sorties sont interfacées sur le même bus de données que le bus mémoire. Les accès IO seront donc traités en partie comme les accès mémoire, à une différence près : les entrées/sorties sont des registres lisibles et modifiables par le microprocesseur, mais des organes extérieurs peuvent (le mot est faible) également les modifier, et l'émulateur va donc devoir mettre à jour ces registres avant toute tentative de lecture par une instruction d'accès IO. C'est encore la MMU qui va nous permettre de réaliser ceci grâce à des fonctions de surveillance de zones mémoires.
- Les appels BIOS ou interruptions logicielles : Le rôle du BIOS étant d'ajouter une couche logiciel

entre le matériel et le programme, il suffit d'adapter le BIOS en fonction du matériel pour que les appels BIOS fonctionnent aussi bien sur le système émulé que sur le système émulant.

Une émulation basée sur la virtualisation va donc être très différente d'une émulation classique. En effet le programme principal (la fonction main() d'un programme C) ne va pas être l'émulateur lui-même mais le code du programme émulé, et l'émulateur sera décomposé dans diverses routines d'interruptions qui vont émuler uniquement les périphériques du système.

Réalisation d'un émulateur

L'émulateur que nous allons réaliser va devoir émuler une Nintendo GameBoy Advance sur une Gamepark GP32. Ce sont deux consoles de jeu portables qui utilisent des microprocesseurs ARM. Ces microprocesseurs sont présents dans de nombreux téléphones portables et PDA actuels et l'émulateur pourrait être porté également sur ces différents appareils.

Présentation des consoles

Nintendo GameBoy Advance

- Historique :

Cette console existe en deux versions qui possèdent exactement la même architecture et sont donc entièrement compatibles. La première version est sortie en 2001, et le nombre de jeux disponibles sur cette plate-forme est maintenant de plus de 700 jeux.

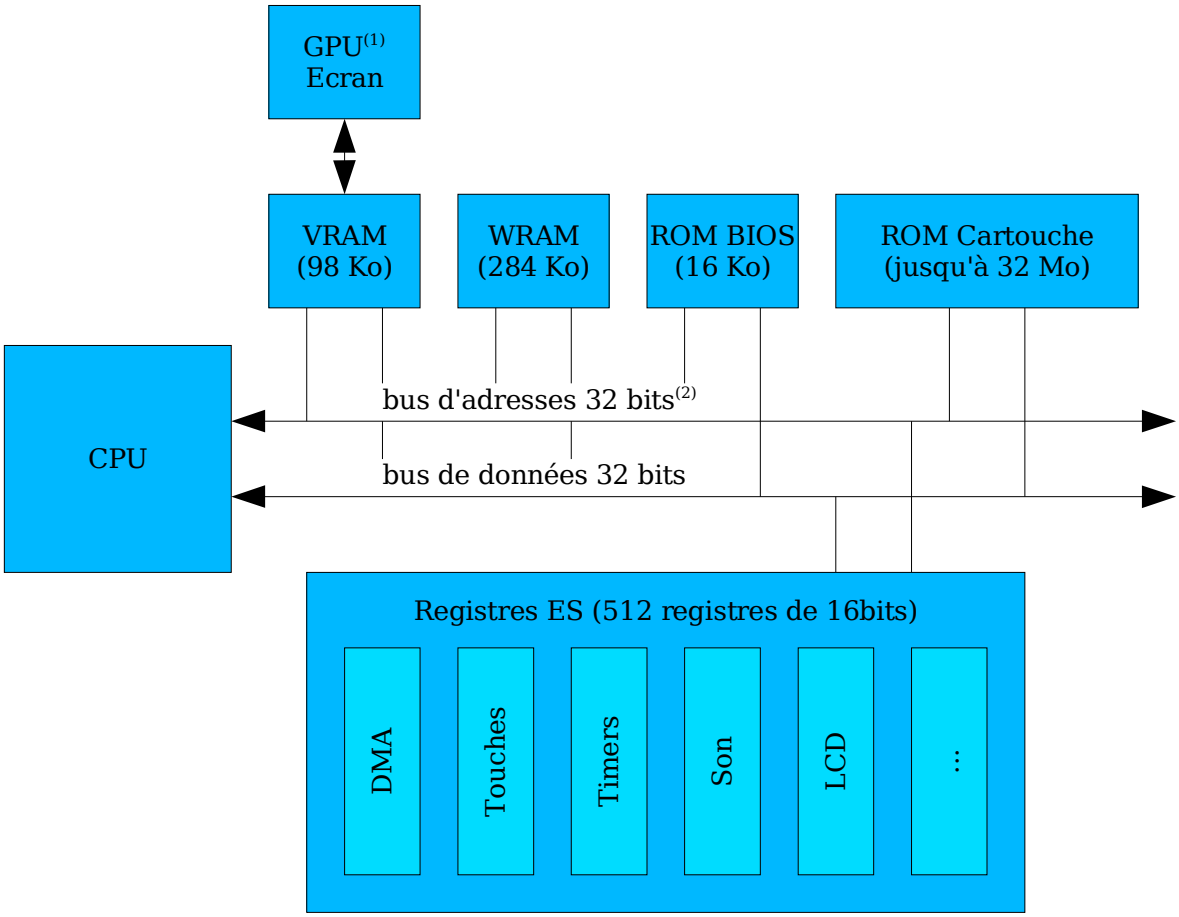
- Caractéristiques techniques :

  - CPU : 32-Bit ARM (ARM7TDMI cadencé à 16 MHz)
  - Processeur graphique : gestion des sprites et des plans
  - Mémoire vive : 32 Ko + 96 Ko VRAM (interne au CPU), 256 Ko WRAM (externe au CPU)
  - Ecran : Ecran TFT couleur de 240 x 160 pixels
  - Couleurs : Peut afficher 511 couleurs simultanément en mode sprites et 32,768 en mode bitmap
  - Support des jeux : cartouches de 4 à 32 Mo

- Environnement logiciel :

  - OS : Pas de système d'exploitation.
  - BIOS : Le BIOS n'offre principalement que des fonctions mathématiques.

- Architecture :



Remarques :

(1) GPU: Processeur graphique.

(2) seuls 28 bits du bus d'adressage sont utilisés.

Gamepark GP32

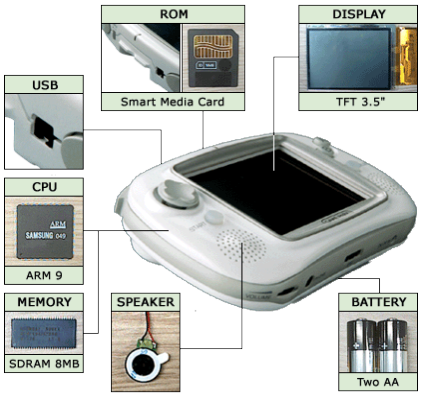
- Historique :

Cette console, plus récente que sa concurrente, a été fabriquée par une petite société coréenne qui se retrouve largement dans l'ombre de Nintendo. La console n'est vendue pour l'instant qu'en Asie et une date de sortie en France n'a pas été prévue.

Cette console possède pourtant des meilleures caractéristiques que son homologue.

- Caractéristiques :

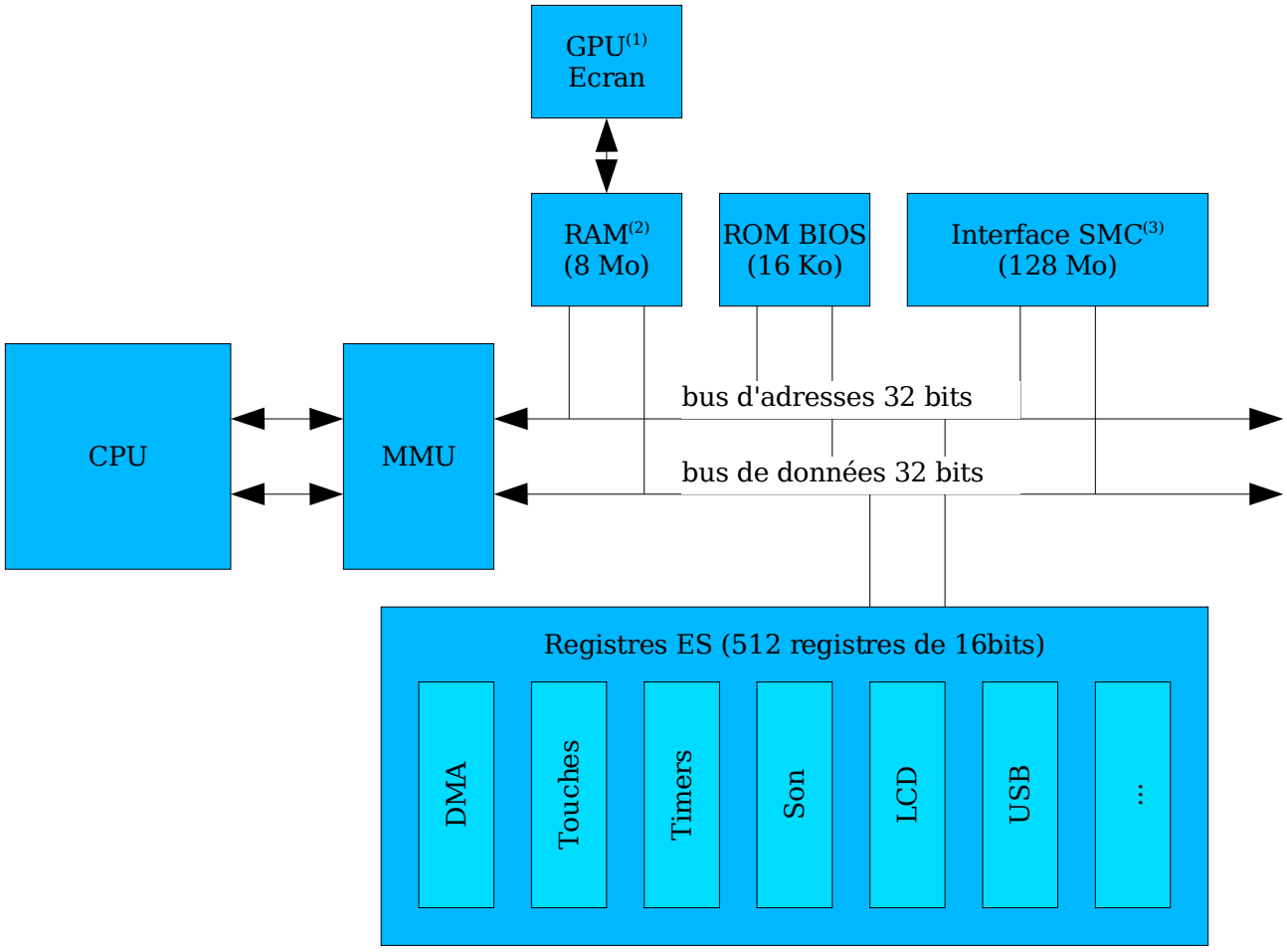
  - CPU : 32-Bit ARM (ARM9TDMI cadencé à 133 MHz)
  - Processeur graphique : modes bitmap uniquement
  - Mémoire vive : 8 Mo partagée avec la mémoire vidéo
  - Ecran : Ecran TFT couleur de 320 x 240 pixels
  - Couleurs : Peut afficher 65,536 couleurs en mode bitmap
  - Support des jeux : cartes Smart Media (SMC) de 128 Mo



- Environnement logiciel :

  - OS : Pas de système d'exploitation.
  - BIOS : Gestion de la MMU, des interruptions, de la carte Smart Media, d'un menu de démarrage.

- Architecture :



Remarques :

(1) GPU: Processeur graphique

(2) Contrairement à la GameBoy Advance, la mémoire vidéo est partagée avec la mémoire vive de la console.

(3) SMC: Smart Media Card, c'est un format de carte mémoire utilisé dans certains appareils photo numériques. Ces cartes peuvent contenir jusqu'à 128 Mo.

Comparatif des consoles

- Microprocesseur :

La première chose à remarquer concernant ces deux consoles est la nature de leurs microprocesseurs : ces deux cousins appartiennent à la même famille des microprocesseurs ARM. C'est cette caractéristique qui va être exploitée dans la réalisation de l'émulateur.

- Processeur graphique :

Le processeur graphique (GPU) de la GameBoy Advance permet de libérer le microprocesseur de la gestion des sprites (éléments graphiques animés possédant un contour particulier et une éventuelle transparence) et des plans.

Le GPU de la GP32 est quand à lui limité à un affichage « bitmap », ce qui signifie que l'écran est simplement divisé en tableaux de pixels, sachant qu'un pixel est codé sur 16 bits. Le déplacement d'un personnage sur l'écran doit donc être effectué par le microprocesseur en modifiant chaque pixel du personnage.

Cette limitation est compensée par le fait que le processeur de la GP32 est cadencé à plus de 10 fois la fréquence de son homologue.

- Mémoire vive :

La GameBoy Advance possède très peu de mémoire vive (256 Ko + 32 Ko), en comparaison à sa concurrente qui en possède 8 Mo. Cette différence s'explique par le choix du type de mémoire morte utilisée. En effet les jeux GameBoy Advance sont directement sur une ROM qui est interfacée directement sur le bus de données de la console, et les programmes peuvent être directement exécutés depuis la cartouche du jeu. Les jeux GP32 sont quand à eux présents sur des cartes mémoires flash assez lente et qui ne sont accessibles qu'à travers une interface matérielle particulière, ce qui signifie que les programmes doivent être d'abord copiés en RAM pour être exécutés.

Une cartouche GBA démontée : une ROM à gauche et une RAM avec sa pile de sauvegarde à droite.

Cette différence impose de posséder les jeux GameBoy Advance sous forme de fichier que l'on copiera sur une carte mémoire, et l'émulateur pourra ainsi placer le contenu du jeu GameBoy Advance en mémoire vive.

- Systèmes d'exploitation :

Aucune des deux consoles ne possède de système d'exploitation, ce qui est un avantage dans notre cas car cela nous donne accès directement au matériel, et en particulier à la MMU.

- Architecture

L'architecture est assez différente sur les deux systèmes, la différence majeure étant la présence de la MMU sur la GP32. C'est cet élément qui va réaliser l'émulation de l'architecture, c'est à dire qui va faire correspondre les plages d'adresses des deux consoles.

Emulateurs déjà existant

Il existe déjà plusieurs émulateurs GameBoy Advance fonctionnant sur ordinateur : Visual Boy Advance, Boycott Advance, BatGBA, etc. Mais rares sont les émulateurs GameBoy Advance qui fonctionnent sur GP32. Il n'en existe qu'un seul, Visual Boy Advance, qui en fait est une adaptation de la version PC. Cet émulateur n'utilise pas le procédé de virtualisation, et s'en retrouve fortement ralenti.

Visual Boy Advance étant cependant en sources libres, certaines parties du code ont été reprises pour gagner du temps de développement, quitte à reprendre ces parties plus tard pour optimiser la vitesse de l'émulateur. A l'heure actuelle, l'émulateur n'est pas encore assez stable pour commencer la phase d'optimisation.

Ressources utilisées

Documentations

La documentation dans ce projet est essentielle car nous faisons appel à des ressources matérielles de très bas niveau. De plus pour pouvoir émuler parfaitement un système, il est nécessaire de respecter très rigoureusement la documentation du système.

Toutes les documentations sont disponibles sur Internet gratuitement, au prix d'une petite recherche google ou mieux directement sur le site du projet (http://gpadvance.sourceforge.net).

- ARM920T Datasheet :

  Il s'agit du coeur du microprocesseur de la GP32, avec en particulier une documentation détaillée de la MMU.

- Samsung S3C2400X :

  Il s'agit du microprocesseur de la GP32 qui contient déjà la plupart des périphériques d'entrée/sortie, et même du processeur graphique.

- Mirko SDK :

  Il s'agit de la documentation fournie avec le kit de développement gratuit pour GP32.

- GBATEK :

  Il s'agit d'une documentation non officielle mais très détaillée sur la Nintendo GameBoy Advance, du jeu d'instructions ARM ou THUMB jusqu'à la tension d'alimentation de la console.

Forums et sites Internet

Les deux consoles étant très ouvertes au développement amateur, les sites internet et les forums de développement sur ces consoles sont très nombreux. Ces sites proposent de la documentation, des librairies, des tutoriels et également des jeux amateurs avec leurs sources. Les forums sont également des lieux d'échange qui permettent de trouver des informations rapidement.

- http://www.gbadev.org

Ce site offre toutes les ressources nécessaires pour développer sur GameBoy Advance.

- http://www.yaronet.com/posts.php?s=33561

Ce forum de développement sur GP32 est le forum français le plus actif pour cette console. Le lien ci-dessus pointe directement vers le premier article que j'avais envoyé dans lequel le projet a pris naissance. C'est l'engouement des internautes avec 250 réponses et plus de 13000 visiteurs qui m'a incité à continuer ce projet.

- http://www.gp32x.com

Ce site est un site d'information anglais sur la GP32, mais il offre également de nombreuses ressources pour les développeurs. Un forum est également disponible où il y figure d'ailleurs une section concernant l'émulateur (http://www.gp32x.com/board/index.php, section GP Advance.)

Il est à noter en particulier que de nombreux possesseurs de GP32 testent l'émulateur à chaque nouvelle version qui sort. Compte tenu du grand nombre de jeux GameBoy Advance, cela est très pratique car cela aide à cibler les différents bogues sans perdre de temps dans une longue phase de tests.

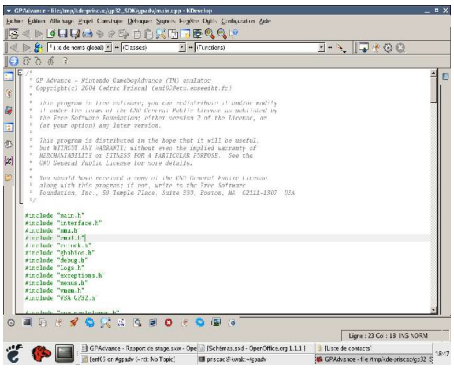
- http://sourceforge.net

Ce site est un fournisseur de services gratuits pour les développeurs qui programment en sources libres. Les services fournis sont les suivants : hébergement d'une page Internet, serveur CVS (voir dans « Logiciels » la définition d'un client/serveur CVS).
Ce site est très connu dans le monde du logiciel libre.

Logiciels

L'intégralité des logiciels utilisés dans ce stage sont gratuits, du système d'exploitation jusqu'au logiciel pour transférer le programme vers la console.

- KDevelop :



Kdevelop est un éditeur de fichier C/C++ qui gère la coloration syntaxique, les projets, l'aide en ligne, la compilation et la mise à jour CVS.

En association avec le compilateur GCC, KDevelop devient un environnement de développement complet à l'image de Microsoft Visual C++ ou de Borland C++.

http://www.kdevelop.org/

- ARM GCC Crosscompiler :

Il s'agit du plus connu des compilateur C/C++ gratuits adapté aux processeurs ARM. Il est en développement constant, deux versions sont d'ailleurs sorties pendant la durée de mon stage.

http://gcc.gnu.org/

- Mirko SDK :

Mirko SDK est un ensemble de librairies pour développer en C/C++ sur GP32. Ce ne sont pas les librairies officielles mais celles-ci ont l'avantage de proposer également les sources des librairies, ce qui m'a permis de trouver l'origine de certains conflits entre ces librairie et mon émulateur (comme de s'approprier une interruption matérielle alors qu'une librairie l'utilise déjà).

http://home.t-online.de/home/mirkoroller/gp32/

- Un client CVS (Concurrent Versions System) :

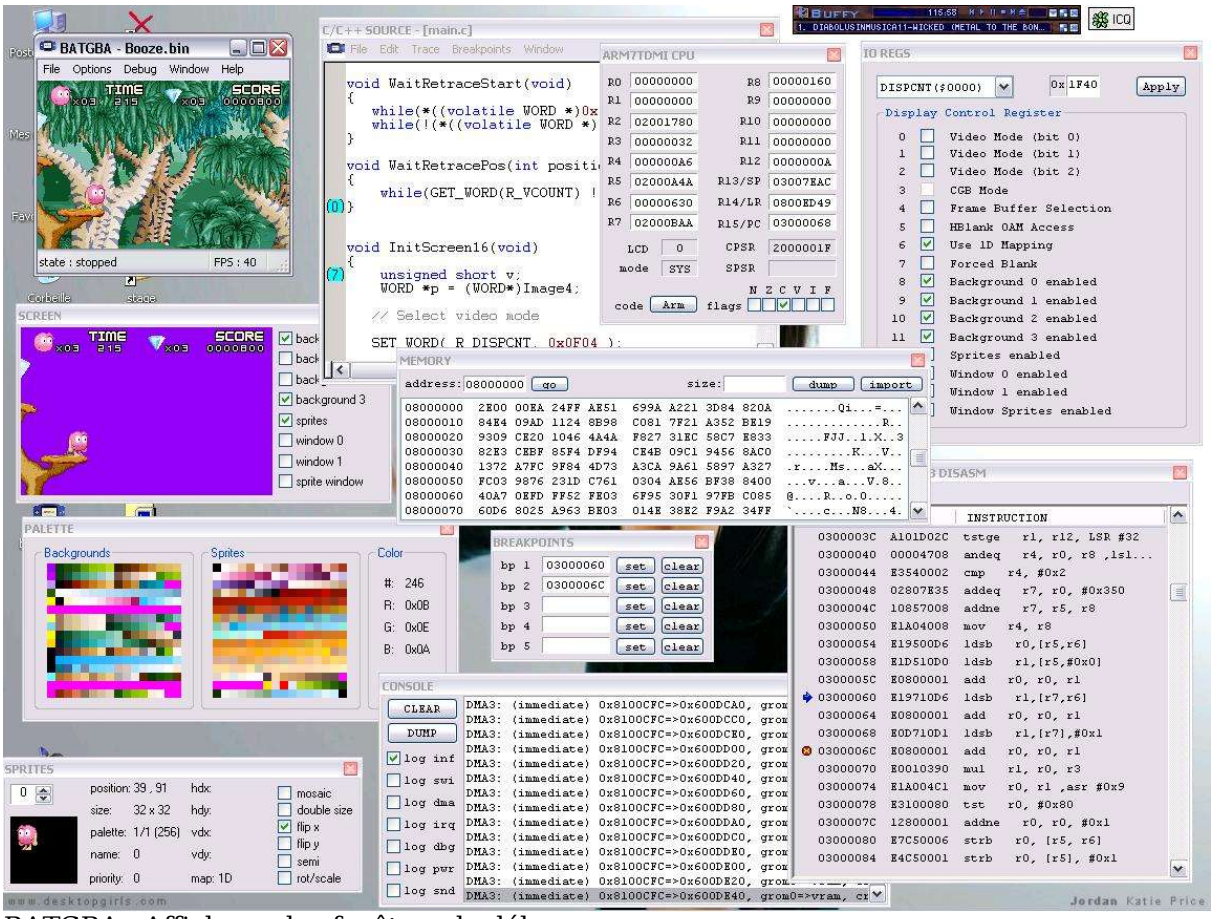
Le CVS est un système client-serveur permettant de gérer les modifications apportées à un code source. L'utilisation de ce procédé permet de stocker l'ensemble des sources d'un programme sur un unique serveur (sourceforge.net dans notre cas), et plusieurs clients CVS peuvent venir télécharger ou modifier ces sources. De plus, chaque version du logiciel est conservée et chaque modification est enregistrée avec le nom du développeur et la date de modification.

Le client CVS est ici incorporé dans KDevelop, et il suffit d'appuyer sur un bouton pour synchroniser ses fichiers sources avec ceux du serveur.

L'arbre CVS de l'émulateur est accessible à cette adresse :
http://cvs.sourceforge.net/viewcvs.py/gpadvance/gpadv

- BatGBA

  BatGBA est un émulateur GameBoy Advance fonctionnant sous Windows. Son principal intérêt n'est pas de pouvoir jouer aux jeux GameBoy Advance sur un PC, mais surtout de les déboguer. Il est ainsi possible de développer des jeux GameBoy Advance sans même utiliser la console.
  L'intérêt de programme pour mon émulateur est de savoir quel organe matériel utilise tel ou tel jeux GameBoy Advance (comme les Irq, Dma, Timers, mode graphique, appel Bios, etc).



BATGBA : Affichage des fenêtres de débogage

Matériel

- Gamepark GP32

  Cette console a été dès le départ conçue pour que le développement amateur soit possible sans accessoire particulier. En effet la console est vendue avec un câble USB permettant de la relier à un ordinateur et ainsi de pouvoir transférer les programmes sur carte mémoire ou directement en mémoire vive pour une exécution immédiate.

  De plus le type de support a été également choisi pour faciliter le développement sur la console. En effet les cartes mémoire utilisées sont des Smart Media Cards qui sont disponibles pour environ 30 € dans tous les grands magasins, et de plus ces cartes sont ré-inscriptibles un grand nombre de fois.

  Il n'a pas été usage d'un débogueur matériel lors du stage, en effet ces appareils sont très coûteux, et je ne suis même pas sûr qu'un tel appareil existe pour GP32. Afin de pouvoir déboguer mon programme, j'ai du incorporer un système de « log » dans lequel on ajoute toute sorte d'information et qui peut être affiché à tout moment, en particulier lors d'un plantage de l'émulateur.

- Nintendo GameBoy Advance

  La politique de Nintendo est quand à elle complètement différente de celle de Gamepark. En effet la GameBoy Advance n'a pas été conçue en faveur du développement amateur. Les outils de développement officiels sont réservés aux développeurs possédant une licence. Heureusement, avec le temps, de nombreuses sociétés tierce-partie ont développé des cartouches de développement avec leur interface à brancher sur n'importe quel ordinateur. Les cartouches normales étant simplement des mémoires mortes, les cartouches de développement ne sont en fait que des mémoires flash que l'on peut écrire à l'aide d'un PC.

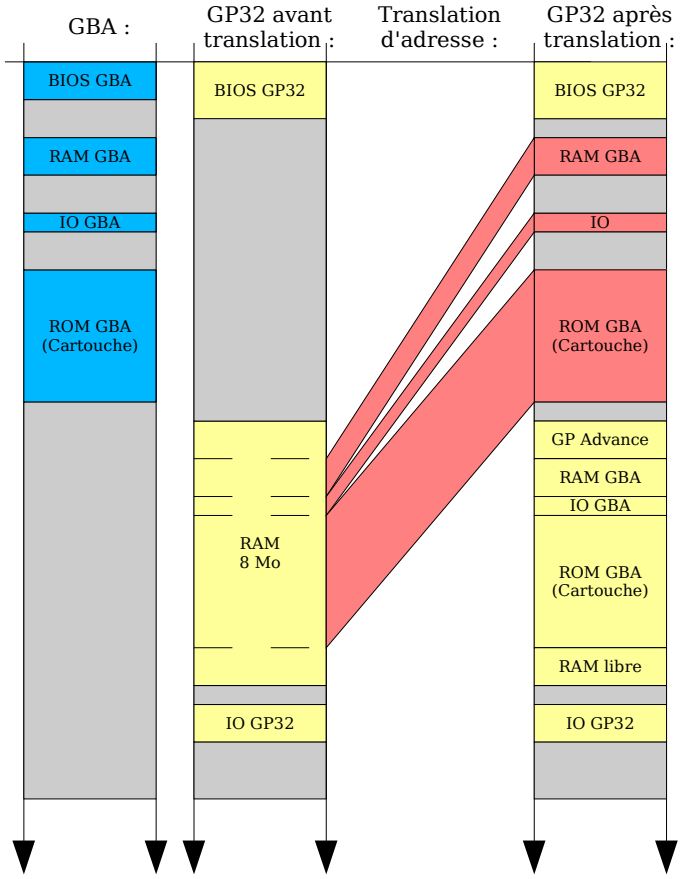
  Cependant, pour la réalisation de mon émulateur, il n'a pas été nécessaire de disposer d'un tel système de développement. L'utilisation de BatGBA a été largement suffisante.

Architecture de l'émulateur

Voici, sans rentrer dans le détail des fichiers sources, les différents modules de l'émulateur.

Initialisation de la MMU

Avant de pouvoir charger un jeu en mémoire, il est nécessaire de configurer la MMU. Cette étape va consister à effectuer des translations d'adresse pour que les adresses mémoire de la GameBoy Advance soient redirigées vers des blocs de mémoire vive réels de la GP32.



Modification des plages mémoires par la MMU

Il est à noter que ce schéma a été simplifié sur différents points :

- La GameBoy Advance possède différents types de RAM (mémoire vidéo, palette, mémoire rapide de faible taille et une mémoire vive d'usage général), et non pas un seul. Il y a en pratique plus de translations d'adresses.
- Il existe en réalité des zones de mémoire utilisées sur les deux consoles, et il a donc fallu également effectuer une translation d'adresse pour déplacer la position de la mémoire vive GP32.
- Les différentes puces de mémoire de la GameBoy Advance sont disposées pour que leur adresse de départ soit un multiple du méga octet. Chaque bloc de mémoire est donc contenu dans 1 Mo, si bien que comme sa taille est inférieure, ce bloc se retrouve répété modulo sa taille. Par exemple la zone WRAM (Work RAM, la mémoire vive lente) qui contient 256 Ko se retrouve dupliquée aux adresses 0200:0000, 0204:0000, 0208:0000 et 020C:0000. J'ai également reproduit ce phénomène en effectuant d'autres translations d'adresses.

Des zones de mémoire correspondant à celles de la GameBoy Advance existent maintenant dans la GP32. Ces zones de mémoires se comportent comme de la mémoire vive normale, mais ont du être allouée dans la mémoire vive réelle de la GP32. Il est donc possible d'accéder à ces zones de mémoire depuis deux adresses différentes, mais il faudra faire attention car la mémoire cache du microprocesseur considère ces deux zones comme étant indépendantes, et il peut en résulter des incohérences dans le cache. C'est pourquoi, par la suite, seuls les adresses mémoires de la GameBoy Advance seront utilisées.

Les menus

La première chose que l'on voit lorsque l'émulateur démarre sont les menus. Ceux-ci permettent de régler les options et de choisir le jeux à lancer. La programmation d'un menu n'est pas spécifique aux émulateurs, et n'a pas posé de problèmes.

La première version de GPAdvance

Le chargement d'un jeu

Une fois que les translations d'adresse sont effectuées, et que le jeu a été choisi dans le menu, il faut charger le jeu à la même adresse mémoire que là où il serait dans une vrai GameBoy Advance.

Création de mémoire virtuelle

Les jeux GameBoy Advance peuvent atteindre des tailles de 32 Mo, mais même si la plupart n'en fait que 8, les jeux ne pourront pas être contenus intégralement en mémoire vive. Une solution qui existe déjà sur la plupart des systèmes d'exploitation consiste à créer de la mémoire virtuelle. Le procédé consiste à utiliser de la mémoire morte tel un disque dur ou une carte mémoire dans notre cas pour stocker des zones de mémoires qui ne seront replacées en mémoire vive que lorsque le processeur essaie d'accéder à cette zone mémoire.

La mémoire virtuelle se comporte donc vis à vis d'un programme comme de mémoire vive classique, mais une interruption indiquant qu'une tentative d'accès a été faite dans telle ou telle zone de mémoire virtuelle. C'est cette interruption qui va devoir transférer la zone correspondante de la mémoire morte vers la mémoire vive et créer une translation d'adresse pour que la zone de mémoire virtuelle soit accessible avec les bonnes données. Le programme qui a tenté d'accéder à la mémoire virtuelle peut alors reprendre son exécution normale.

Dans mon cas, la zone de mémoire virtuelle correspond au jeu GameBoy Advance, et il restera donc stocké sous forme de fichier dans la carte mémoire. De plus, cette zone sera en lecture seule, ce qui simplifie grandement le fonctionnement de la mémoire virtuelle.

Emulation des périphériques

Bien qu'une grande partie du travail a déjà été faite avec tous les modules qui précèdent, il reste néanmoins une grande partie du travail que le processeur va devoir accomplir : l'émulation de tous les périphériques de la GameBoy Advance.

Ces organes fonctionnant en parallèle avec le microprocesseur, il faudrait idéalement que ce module soit exécuté également en parallèle. Ceci étant bien sûr impossible, je me suis encore inspiré des systèmes d'exploitation multitâches en réalisant une exécution pseudo-parallèle, c'est à dire en effectuant des interruptions grâce à une horloge du microprocesseur et en exécutant à tour de rôle ce module et le jeu GameBoy Advance.

Cette synchronisation est extrêmement critique car certaines entrées/sorties doivent être

émulées à intervalles bien déterminées. De plus à chaque interruption il n'est pas nécessaire de mettre à jour tous les périphériques, ce qui est assez important pour ne pas ralentir l'émulateur.

L'émulation des périphériques au sein de l'interruption d'horloge est la même que dans tous les émulateurs GameBoy Advance existants, et pour gagner du temps, nous avons utilisé les sources d'un autre émulateur, Visual Boy Advance.

Afin de ne pas mélanger mes fichiers de ceux de cet autre émulateur, les fichiers sources de Visual Boy Advance sont précédés de «VBA_ ».

L'utilisation du code de cet autre émulateur m'a permis de gagner beaucoup de temps et de me concentrer sur les problèmes liés à la virtualisation.

Gestion des interruptions

Les interruptions sont utilisée aussi bien sur GameBoy Advance que sur GP32 par de nombreux périphériques, comme les horloges, les touches, le rafraîchissement de l'écran, etc. Les périphériques GameBoy Advance étant émulé dans la GP32, les interruptions qu'ils pourraient provoquer sont également émulées, et ne se traduisent pas par une réelle interruption sur GP32.

Cela paraît donc assez simple en apparence, mais l'inconvénient majeur vient du fait qu'en pratique, la rentrée et la sortie d'une interruption sont gérées par le microprocesseur, et en particulier si l'on appelle simplement la fonction du BIOS GameBoy Advance qui doit gérer les interruptions, cette fonction va se terminer par une instruction en assembleur prévue pour sortir d'une interruption, alors que celle-ci n'avait pas été appelée depuis une vrai interruption. La virtualisation pose donc quelques problèmes concernant la gestion des interruptions, et il reste encore à résoudre ce problème.

Statut de l'émulateur

Les principales fonctions de l'émulateur ont été développées, mais il nécessite encore un temps considérable de débogage et d'optimisation pour fonctionner en temps réel avec un bon taux de compatibilité.

L'émulateur a été principalement testé avec des programmes amateurs car ceux-ci sont souvent disponibles avec leurs sources, et de plus ils utilisent très peu de périphériques de la GameBoy Advance. Ces programmes sont donc mieux adaptés à tester les différentes fonctions de l'émulateur séparément.

La compatibilité vis à vis des programmes amateur est d'ailleurs très bonne, par contre celle vis à vis des jeux commerciaux est bien moindre. Certains jeux fonctionnent néanmoins et d'autres sont même jouables.

Conclusion

La réalisation de cet émulateur, s'il n'est pas terminé, a tout de même bien répondu à la problématique du sujet : à condition de disposer d'un microprocesseur disposant d'une unité de gestion de la mémoire (MMU), il est tout à fait possible de faire fonctionner les programmes ARM d'un système sur un autre. La perte de performance sera elle directement liée à l'émulation des périphériques matériels.

De plus, cet émulateur m'a personnellement beaucoup appris sur l'architecture des machines et des systèmes informatiques, ainsi que sur le fonctionnement d'un système d'exploitation.

Enfin, le stage m'a également permis de gérer un projet et surtout de travailler en équipe avec des outils comme le CVS ou simplement des forums internet.

Annexes

Afin de ne pas surcharger le rapport avec des documentations techniques, et sachant que toutes les phases du stages ont été réalisées sur Internet (de l'idée de départ dans les forums jusqu'à chaque modification du code qui a été enregistrée et datée), voici les adresses Internet des différentes ressources :

- Les fichiers sources du stage :
  http://cvs.sourceforge.net/viewcvs.py/gpadvance/gpadv/

- La documentation technique de la MMU :
  http://gpadvance.sourceforge.net/docs/DDI0151C_920T_TRM.zip

- La documentation technique du microprocesseur de la GP32 :
  http://gpadvance.sourceforge.net/docs/um_s3c2400x_rev1.1.zip

- La documentation technique de la GameBoy Advance :
  http://www.work.de/nocash/gbatek.htm

- Un article sur l'avenir des émulateurs :
  http://www.wired.com/news/print/0,1294,64914,00.html