How-to

par Aurélien Lucchi

Date de publication : 06/07/2005

Dernière mise à jour :

Cet article explique les rudiments du projet SKAN. Il abordera les différents outils utilisés pour le projet (présentation, téléchargement, paramétrage,...) afin de vous permettre de compiler vous même votre propre version de SKAN.

Vous pouvez aussi consulter cet article au format PDF.

Si ce lien ne fonctionne pas chez vous, utilisez celui-ci.

I - Introduction

Nous commencerons par la présentation de certains outils indispensables pour le projet SKAN. Bochs est un émulateur PC très pratique pour les développeurs tandis que FASM est l'assembleur officiel utilisé pour le projet (il s'agit du résultat d'un vote des membres de l'équipe). Ensuite, une explication plus détaillée vous permettra de compiler les sources de SKAN.

II - Bochs

Site officiel de Bochs

II.A - Généralités

Bochs est un émulateur portable et open source (sous licence GNU/GPL) pour les architectures compatible x86 et AMD64. Il est écrit en C++ et il supporte l'émulation du processeur (incluant le mode protégé), de la mémoire, de l'ethernet, du BIOS, de la plupart des périphériques pour PC. Bochs est capable de faire fonctionner la plupart des systèmes d'exploitation comme Linux, Windows 95, Dos, Windows NT4, OS X sur Mac.

En tant qu'émulateur, Bochs est couramment utilisé pour le développement d'OS car en cas de crash complet de l'OS émulé, on ne crash pas son propre système. On ferme Bochs et on peut le relancer en quelques secondes. En plus de cela, les OS émulés peuvent être plus facilement débuggés.

Contrairement à d'autres logiciels (comme VMWare Workstation par exemple dont vous trouverez quelques informations dans la section outils), Bochs émule entièrement chaque instruction du processeur x86 et n'utilise pas les instructions du processeur existant.

- En théorie, Bochs fonctionne sur les plates-formes dont le processeur n'est pas x86.
- Bochs est beaucoup plus lent que les émulateurs qui n'émulent pas le processeur. Pour le développement de SKAN, cela n'est pas vraiment gênant, surtout quand on considère les avantages offerts par cet émulateur.

Il faut aussi savoir que les performances de Bochs dépendent en grande partie du paramétrage que vous avez fait. Enfin, sachez que la documentation de Bochs a été en grande partie écrite au format DocBook (il vous faudra donc l'outil jade qui permet de générer ce format si vous voulez recompiler Bochs).

II.B - Paramétrage

Par défaut Bochs cherche à lire le fichier ayant le nom "bochsrc" dans le répertoire d'installation de Bochs. Vous pouvez renommer le fichier "bochsrc-sample.txt" en "bochsrc" puis éditer ce fichier avec un éditeur de texte (comme le bloc note de windows ou vi sous Linux).

Conseil : Faites attention au saut de ligne dans votre fichier de configuration bochsrc. Au besoin éditer votre fichier de configuration avec un éditeur hexadécimal pour vérifier que les sauts de lignes sont traduits par 0x0D (en hexadécimal) sous Linux et 0x0D0A sous Windows.

- image ROM :

  Indiquez la localisation du fichier BIOS-bochs-latest présent dans le répertoire d'installation :

  romimage: file=chemin\BIOS-bochs-latest, address=0xf0000

- image ROM VGA :

Indiquez la localisation du fichier VGABIOS-lgpl-latest présent dans le répertoire d'installation :

vgaromimage: file=chemin\VGABIOS-lgpl-latest

- megs.

  Ce paramètre va déterminer la quantité de mémoire physique sui sera alloué à l'OS émulé. Pour l'instant, le paramètre par défaut (à savoir 32 megs) serait largement suffisant. Décommenter (supprimer le caractère # en début de ligne) la ligne qui détermine la quantité de mémoire que vous voulez utiliser.

- ata0-master.

  Ceci concerne les paramètres de votre disque dur maître, à savoir le nombres de têtes, pistes et cylindres de votre disque dur. Si ces paramètres sont faux, cela n'empêchera pas l'exécution de Bochs mais mieux vaut bien paramétrer Bochs pour éviter des plantages par la suite lorsque vous passerez à la véritable exécution de l'OS. Néanmoins si vous avez un disque avec une grosse capacité, il est possible que vous ne puissiez pas émuler complètement cette capacité car Bochs limite la valeur du nombre de cylindres par exemple. Si vous avez dépassé cette valeur, il vous avertira par le message d'erreur suivant :

  "numerical parameter ata-device:cylinders was set to X, which is out of range".

  ata0-master: type=disk, mode=flat, path="chemin\skan.bin", cylinders=15956, heads=16, spt=63

- boot :

  Ceci vous permet de définir votre secteur de boot. Si vous voulez booter sur le disque dur :

  #boot: floppy

  boot: disk

  La première ligne est commentée mais pas la deuxième.

  Si vous voulez booter sur disquette, assurez vous que la ligne floppya avec le chemin d'accès est exacte. Sous Windows, on a :

  floppya: 1_44=a:, status=inserted

  Sous Linux, le chemin sera /dev/fd0.

- IPS (Emulated Instructions Per Second).

  Regarder les commentaires du fichier "bochsrc-sample.txt" pour adapter ce nombre selon la puissance de votre pc.

Pour plus d'informations, n'hésitez pas à vous reportez à la documentation de SKAN qui est disponible en ligne sur le site http://bochs.sourceforge.net

II.C - Installation sous Linux

1 Récupérer les dernières sources sur le site de Bochs :

http://bochs.sourceforge.net

2 Décompresser l'archive puis compiler :

$ tar xvfz bochs-2.1.tar.gz

$ cd bochs-2.1

$ ./configure

$ make

3 Installer en tant que root :

$ sudo make install

- Formater une disquette (ext2) : mkfs -t ext2 /dev/fd0 1440
- Copie vers une disquette : dd if=skan.img of=/dev/fd0

Ensuite, vous n'avez plus qu'à lancer Bochs. Sous Linux, le fichier de paramétrage se nomme .boschsrc car il s'agit d'un fichier caché.

III - FASM

Site officiel de FASM

III.A - Généralités

FASM est un assembleur 32 bits. Le nom vient de "flat assembler". Fasm est lui même est écrit en assembleur. Il est disponible pour Windows, Linux et Menuet (un OS écrit avec FASM : http://www.menuetos.org). Il possède certaines caractéristiques avancées que nous verrons dans la prochaine partie.

Actuellement, FASM supporte le jeu d'instructions 8086 de base ainsi que les extensions du 80486 et du Pentium (MMX, SSE, SSE2, SSE3 et 3DNow!). Le 64 bits n'est pas en reste puisque FASM prend également en charge les extensions x86-64 (les deux spécificités majeures du x86-64 sont l'extension 64-bits appelée long mode et les extensions du registre), que ce soit celle de l'AMD64 ou du EM64T (Intel Extended Memory 64 Technology).

La syntaxe de FASM ressemble beaucoup à celle de NASM et les 2 incluent certaines caractéristiques communes comme la création de binaires au format flat, ELF (format exécutable pour Linux), PE (format exécutable pour Windows), ou encore COFF (Common Object File Format ou Format de fichier objet, il s'agit d'un format UNIX à la base).

III.B - Caractéristiques

Adresse courante : "$"

Une des caractéristiques intéressantes de ces 2 assembleurs est le dollar ("$"). Le signe "$" correspond à l'adresse courante (lors de l'exécution du programme). Ceci est vraiment très utile pour déterminer la taille d'un bloc de code ou de données.

Voici un exemple simple d'utilisation du "$" :

maChaine db "Ceci est ma chaine", 0

lgMaChaine equ $- maChaine

Labels

Une autre des caractéristiques couramment utilisées sont les labels locaux, commençant par un ".".

Exemple :

labelglobal :

.labellocal:

Macros

Les macros sous FASM ressemblent à celles du C. La syntaxe est la suivante :

macro (nom) (paramètres) { code }

Voici un exemple qui permet d'utiliser l'instruction mov avec 3 paramètres :

```
macro mov op1,op2,op3
{
 if op3 eq
   mov   op1,op2
 else
   mov   op1,op2
   mov   op2,op3
 end if
}
```

III.C - Installation

L'installation est très simple. Il suffit de vous rendre sur http://flatassembler.net/ dans la rubrique download et de télécharger la version souhaitée parmi les 3 disponibles (Windows, Linux, ou DOS). Vous n'avez plus qu'à décompresser l'archive obtenue et récupérer le code source de SKAN (cf. section suivante) pour la compilation.

IV - Compilation du code source

Après avoir téléchargé l'archive sur le serveur FTP, décompresser le contenu de l'archive. Ensuite, il faut compiler skan.asm qui contient toutes les inclusions nécessaires à la compilation des autres fichiers. Pour cela, il faudra que vous ayez préalablement installé FASM (référez vous à la première partie de cet article pour plus d'informations sur FASM).

Une fois FASM installé, voici la commande à utiliser sous Linux pour compiler le fichier skan.asm :

fasm src/skan.asm bin/skan.img

(en supposant que vous soyez dans le répertoire racine du projet).

Sous Windows ou sous DOS, vous n'avez qu'à remplacer les "/" par des "\". Aucun changement n'est nécessaire pour passer d'une plate-forme à l'autre car FASM est entièrement portable pour Windows et Linux.

Le fichier skan.img produit peut alors être utilisé comme une image de disque sous bochs ou copié sur une disquette (avec rawrite, gratuit, open source et illimité) :

http://uranus.it.swin.edu.au/~jn/linux/rawwrite.htm

Sur disquette :

Il suffit de bien paramétrer le fichier boshsrc comme indiqué précédemment. Voici un exemple de fichier de paramétrage pour booter sur disquette :

Voir le fichier (si ce lien ne fonctionne pas chez vous, utilisez celui-ci).

Conseil :

Pour une utilisation simple, vous pouvez par exemple configurer un démarrage de SKAN à partir du disque dur en lui indiquant le chemin de skan.bin dans le répertoire ou vous l'avez généré grâce avec FASM.