Ordinateurs, Structure et Applications

Simulateur d'ordinateur bas niveau

Guide de l'utilisateur

4 septembre 2015

Table des matières

1 Description du simulateur

Le simulateur d'ordinateur bas-niveau simule le fonctionnement d'un ordinateur rudimentaire à architecture Harvard avec entrées/sorties mappées en mémoire (Memory Mapped IO).

Le simulateur offre deux modes de fonctionnement différents, le mode démonstration et le mode interactif. Dans le mode démonstration, un CPU simulé exécute les instructions de la mémoire d'instruction. Dans le mode interactif, c'est l'utilisateur qui joue le rôle du CPU en utilisant les bus de contrôle, d'adresses et de données afin d'écrire ou de lire la mémoire ou les périphériques.

2 Plateformes supportées

Le simulateur a été testé sur les plateformes suivantes :
Linux : ArchLinux, Fedora 20, Ubuntu 14.10
Mac : Mac OSX 10.8, 10.9 et 10.10
Windows : Windows 7 et 8.1
Le fonctionnement du simulateur sur les autres plateformes n'est pas garanti.

3 Procédure de lancement du simulateur

Le simulateur est fourni avec toutes les librairies nécessaires pour son fonctionnement. Voici la procédure pour rouler le simulateur sur les différentes plateformes.

- Sur **Linux**, il suffit d'extraire l'archive tar.gz et d'exécuter le fichier exécutable s'y trouvant.
- Sur **Mac**, le simulateur est fourni dans le format application compressée (.dmg). Il est possible qu'il soit nécessaire de confirmer l'exécution en l'ouvrant avec CTRL + Click.
- Sur **Windows**,vous devez installer le *Microsoft Visual C++ Redistributable Package*. L'installateur est disponible sur le site suivant : http://www.microsoft.com/en-us/download/details.aspx?id=14632. Ensuite, il suffit d'extraire l'archive zip et d'exécuter le fichier exécutable s'y trouvant.

4 Composantes de l'ordinateur

4.1 Bus

L'ordinateur comprend un bus d'adresse, de données et de contrôle. Voici le rôle de chacun des bus.
Bus d'adresse : Contient l'adresse à l'intérieur de l'ordinateur qui doit être lu ou écrite.
Bus de données : Contient la donnée lue ou à écrire.
Bus de contrôle : Permet de sélectionner le type d'opération effectuée par le microprocesseur (Écriture ou Lecture).

4.2 Mémoires

L'ordinateur contient une *mémoire d'instructions* en lecture seule et *une mémoire de données* disponible en écriture et en lecture. Chaque emplacement mémoire contient une valeur sur 16 bits. Il est possible de charger le contenu initial des mémoires au début de la simulation en fournissant des fichiers de configuration. Si aucun fichier de configuration n'est fourni, la mémoire d'instruction contiendra des instructions vides à chaque adresse et celle de données la valeur 0x0.

4.3 Périphériques

L'ordinateur possède deux périphériques : un clavier et un écran.

Le clavier peut contenir, au maximum, deux caractères faisant parti de la table ASCII étendue. L'écran permet d'afficheur une valeur sur 16 bits en format hexadécimal.

4.4 Registres

L'ordinateur contient quatre registres généraux (R0,R1,R2 et R3) et le registre PC.

5 Fonctionnalités du simulateur

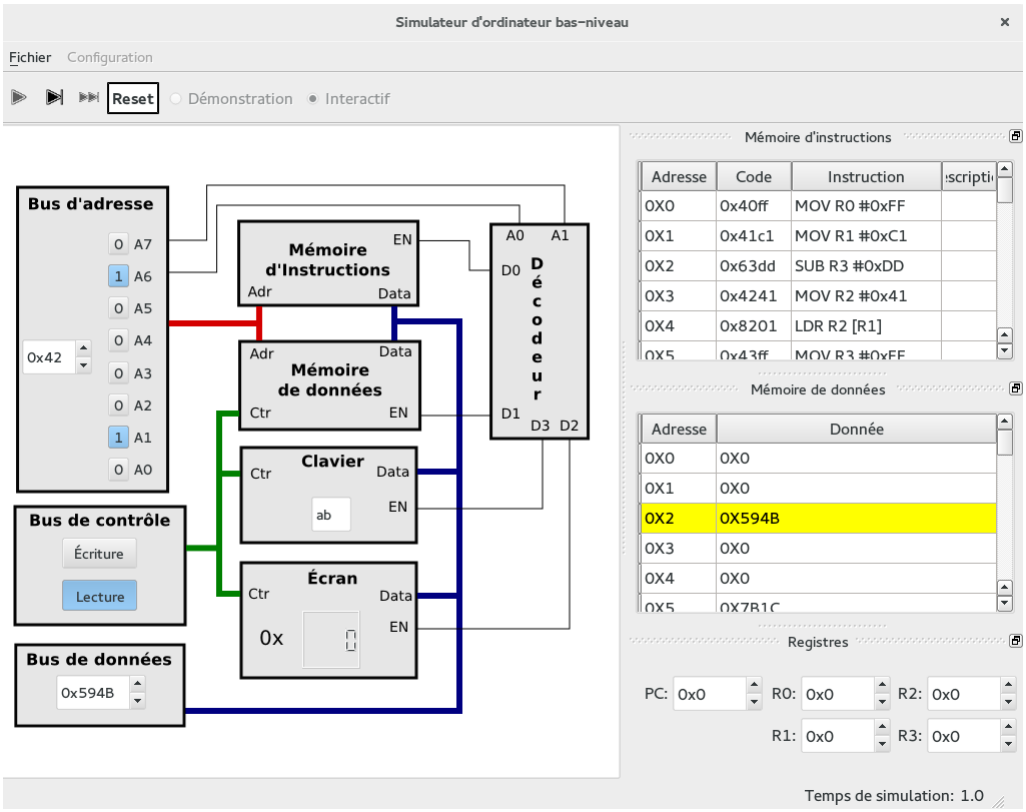

FIGURE 1 – Interface du simulateur

5.1 Sélection du mode du simulateur

Premièrement, le mode du simulateur peut être modifié à partir des boutons radio se trouvant dans la barre d'outils. Le mode peut seulement être modifié avant le début d'une simulation et quand un fichier Assembleur signé est chargé dans la mémoire d'instruction.

5.2 Modification de la valeur des bus et périphériques

Le dessin du circuit de l'ordinateur permet de modifier les valeurs des bus, des mémoires et des périphériques. Ces valeurs sont seulement modifiables en mode interactif.

5.3 Affichage des valeurs des mémoires et des registres

Une barre d'ancrage à la droite contient les fenêtres affichant les valeurs des mémoires et des registres. Il est possible de détacher chaque fenêtre en cliquant sur le bouton  dans le coin supérieur droit de la fenêtre. Lorsque détachée, une fenêtre peut être rattachée dans la barre d'ancrage en double-cliquant dans sa barre de titre.

Quand une valeur est lue ou écrite dans une mémoire, la rangée se trouvant à l'adresse fournie à cette mémoire sera surlignée en jaune dans la fenêtre correspondante. Les valeurs des registres sont seulement modifiables en mode interactif.

6 Contrôle de la simulation

Le simulateur permet d'exécuter une simulation en continu, pas à pas et jusqu'à la fin. La vitesse d'exécution en seconde entre les rafraîchissements de l'interface et le nombre de pas de simulation effectué par rafraîchissement peuvent être réglés dans le menu principal à partir de l'option "Préférences" sur Mac et "Options" sous Linux et Windows.

Les quatre boutons en haut à gauche de la fenêtre permettent de contrôler le déroulement de la simulation. Voici les fonctionnalités offertes par chaque bouton.

: Permet de réinitialiser la simulation.

 : Permet d'exécuter des pas de simulation en continu.

 : Permet de suspendre l'exécution de la simulation .

 : Permet d'exécuter un pas de simulation.

 : Permet d'exécuter tous les pas de simulation successivement.

7 Contenu des mémoires

7.1 Chargement

Le contenu des registres et des mémoires de données et d'instructions peut être chargés à partir du menu "Configuration". Un message d'erreur apparaîtra si le fichier sélectionné n'est pas valide. Après le chargement des fichiers de configuration, le contenu des fichiers est affiché dans les fenêtres détachables correspondantes.

7.2 Contenu initial de la mémoire de données

Le contenu initial de la mémoire de données peut être fourni à l'aide d'un fichier XML. Ce fichier doit utiliser cette structure :

```
<DataMemory>
    <Data Address="0x2f" Value="0xd948" />
    <Data Address="0x27" Value="0x5dc7" />
</DataMemory>
```

À l'intérieur d'une balise nommée *DataMemory*, un élément de type *Data* est ajouté pour chacune des adresses à configurer. Toutes les adresses qui ne possèdent pas d'élément dans le fichier se verront attribuer 0 comme valeur.

7.3 Contenu initial de la mémoire d'instruction

Le contenu initial de la mémoire d'instruction peut être fourni à l'aide d'un fichier Assembleur ayant l'extension .asm . Dans ce fichier, chaque ligne contient une seule instruction Assembleur respectant les contraintes du jeu d'instructions. Un commentaire peut être ajouté à la fin de chaque ligne en utilisant une apostrophe ('). Un dièse (#) permet d'insérer une constante comme opérande d'une instruction.

Pour que le mode démonstration soit accessible, une signature RSA valide doit se trouver dans le bas du fichier. La signature RSA permet de s'assurer que le fichier Assembleur n'a pas été modifié. Si le fichier est modifié de quelque façon que ce soit, la signature ne sera plus valide et le mode démonstration ne sera plus accessible pour ce fichier.

Voici un exemple de fichier Assembleur :

```
MOV R0 #0xC7
MOV R1 #0x59
MOV R2 #0xB0
LDR R3 [R0] 'Chargement de la valeur du clavier dans R3
STR R3 [R1] 'Ecriture dans la memoire de donnees
STR R3 [R2] 'Ecriture dans l'ecran
ADD R2 R1
SUB R0 R1

Signature : b'\x00\xc7\x83Z\xaeY\x7f\x03\xcbH!\xd0\xa2
\x08\xaf=\xd1P\x8e\xc3\xfd\x0c@DEk\xcb\x05\xf9/\x92
\xc2f*\xdc\xf0\xaf\x88\xa1M L\xa0\xce\xd7j\x87\xd4
\x8e=F\xa4\xd8D\xda\xe4\xea\x81\x12\xd9\xa4@GV'
```

7.4 Contenu initial des registres

Le contenu initial des registres peut être chargé à partir d'un fichier XML. Le fichier doit être organisé de la manière suivante :

```
<CPU>
<Register Name="PC" Value="0x56"/>
<Register Name="R0" Value="0x24"/>
<Register Name="R1" Value="0x5644"/>
<Register Name="R2" Value="0x5613"/>
<Register Name="R3" Value="0x11e"/>
</CPU>
```

À l'intérieur d'une balise *CPU*, des éléments de type *Register* doivent être présents pour chacun des registres à configurer. Les registres qui ne sont pas présents dans le fichier se verront attribuer la valeur 0.

8 Jeu d'instruction du micro-processeur

Toutes les instructions du microprocesseur sont sur 16 bits et se décomposent comme suit :

Bits 15 à 12 : Opcode de l'instruction

Bits 11 à 8 : Registre utilisé comme premier paramètre.

Bits 7 à 0 : Registre ou constante utilisés comme deuxième paramètre

Le microprocesseur possède quatre registres généraux nommés R0, R1, R2 et R3. En plus de ces quatre registres, un registre de pointeur d'instruction PC est disponible. Cependant, ce registre ne peut être utilisé qu'avec l'instruction MOV. Le nombre identifiant le registre PC est 0xF (15).

Le jeu d'instruction supporte les instructions suivantes où Rd est le registre destination, Rs le registre source et Rc le registre de condition :

Mnémonique	**Opcode**	**Description**
MOV Rd Rs	0000	Écriture de la valeur du registre Rs dans le registre Rd
MOV Rd Const	0100	Écriture d'une constante dans le registre Rd
ADD Rd Rs	0001	Addition des valeurs des registres Rd et Rs et insertion du résultat dans le registre Rd
ADD Rd Const	0101	Addition de la valeur du registre Rd avec une constante et insertion du résultat dans Rd
SUB Rd Rs	0010	Soustraction de la valeur Rs à l'intérieur de registre Rd.
SUB Rd Const	0110	Soustraction d'une constante à l'intérieur du registre Rd
LDR Rd [Rs]	1000	Chargement d'une valeur se trouvant à l'adresse Rs de l'ordinateur dans un registre.
STR Rd [Rs]	1001	Écriture de la valeur d'un registre à l'adresse Rs de l'ordinateur.
JZE Rc Const	1111	Saut à l'instruction située à l'adresse identifiée par la constante, mais seulement si Rc = 0 (sinon, cette instruction n'a aucun effet).
JZE Rc [Rs]	1011	Saut à l'instruction située à l'adresse Rs seulement si Rc = 0 (sinon, cette instruction n'a aucun effet).

Table 1 – Jeu d'instructions du microprocesseur