EVA

Association CodeAnon

14 octobre 2019

Table des matières

1 Remarque préliminaire 3

2 Introduction 5

3 Eva 6

4 Description technique 7

5 Éléments de conception 8

6 Implantation de la machine virtuelle 8

7 Disponibilité du projet 8

8 Projets connexes 9

1 Remarque préliminaire

Ce document est écris au présent. Il n'en reste pas moins une projection de ce que sera Eva au terme de son développement. Il faut donc le lire comme une description détaillée du projet final Eva et non comme une description de son état actuel.

2 Introduction

Le langage assembleur constitue la première abstraction à la programmation de circuits programmables. Aussi son apprentissage est une étape fondamentale pour l'étudiant curieux de comprendre le fonctionnement des ordinateurs. Mais d'autres applications nécessitent une bonne connaissance des langages d'assemblage. L'écriture des compilateurs par exemple repose assez largement sur une connaissance fine des différentes architectures existantes (ARM, elf, x86 ...) et des langages d'assemblage qui leurs sont associés.

Comment alors permettre aux étudiants désireux de s'initier à l'écriture de compilateur ou d'applications très bas niveau de réaliser leurs projets ? Face à l'étendue des architectures à cibler, à la complexité de certaines d'entre elles, ce type de projet peut rapidement devenir compliqué. Bien sur, des alternatives existent déjà. On peut citer le fameux ouvrage Structure And interpretation of Computer Programs [1] qui propose en dernier chapitre l'implémentation d'une machine abstraite programmable dans un langage d'assemblage simple. D'autres projets comme la machine virtuelle CHIP8 sont des sources intéressantes pour découvrir la programmation assembleur [2]. Ces outils déjà utilisés dans le but de rendre accessible aux étudiants les thématiques dites de "bas niveaux" restent néanmoins très spécifiques et limitent grandement le champs des possibles. C'est à ce carrefour entre facilité d'apprentissage et possibilités offertes ; ce delta entre outil pédagogique et problèmes en tailles réelles qu'essaye de s'intercaler le projet Eva.

3 Eva

Eva est une machine virtuelle développée par des étudiants pour les étudiants. Simple et légère, elle offre un terrain d'expérimentation particulièrement adapté à l'apprentissage de l'assembleur ou encore à l'écriture de compilateurs. Son code source publique, clair et documenté la rendent accessible et facilement modifiable. Eva est programmable dans un langage assembleur dédié facile d'accès et doté d'un jeu d'instructions succinct mais complet. Par complet nous entendons informellement "suffisant pour permettre d'implanter une application incluant des entrées, des sorties et éventuellement des fonctions de réseaux ou d'affichage graphique". Nous reviendrons en détail sur ce point dans la suite du document.

4 Description technique

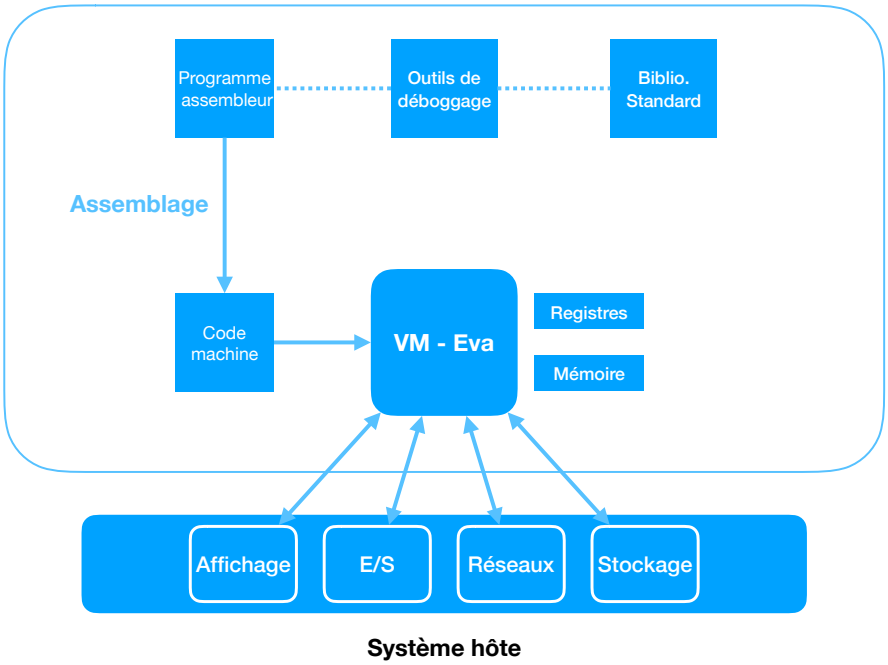

Figure 1 – Structure de la machine virtuelle Eva

L'ensemble du projet Eva se compose de plusieurs outils.

— **Cœur de la VM**
Il s'agit d'un interpréteur pour le langage machine de Eva. C'est ce programme qui fait l'interface entre l'utilisateur de Eva et sa machine.

— **Assembleur Eva**
Ce dernier permet la conversion d'un code assembleur en un fichier exécutable par la VM. Il est également accompagné d'un éditeur de lien permettant de composer des exécutables à partir de plusieurs modules assembleur.

— **Outil de débogage**
Cet outil permet aux utilisateurs de lancer l'exécution de leurs programmes en mode pas à pas. Ils peuvent ainsi observer en détail le comportement de leur programme et comment la VM le traite. On peut par ce biais contrôler le contenu de la mémoire et des registres de

la VM.

— **Bibliothèque Standard**
Le projet Eva est livré avec une collection de modules utilitaires. Il peut s'agir par exemple de fonctions d'affichages ou de calcul mathématique. Cette bibliothèque peut être explicitement chargés en mémoire avant le lancement de la VM, on pourra alors l'utiliser via des "appels systèmes". Une autre alternative étant de passer par une étape d'édition de lien pour embarquer une sélection de module dans un exécutable.

5 Éléments de conception

La machine virtuelle doit être minimaliste pour que la compréhension de son fonctionnement soit accessible, simple à programmer pour rester accessible aux étudiants désireux d'apprendre, mais également suffisamment performante pour supporter le développement d'applications en vrai grandeur. Eva étant conçue sur la base du « retour aux fondamentaux », elle va donc à l'essentiel et propose un jeu d'instructions à la fois concis et simple à prendre en main sans fonctionnalités superflue. Nous avons à ces fins sélectionné un sous-ensemble des instructions ARM tableau 1.

6 Implantation de la machine virtuelle

Afin d'aider au développement du projet, Eva va être implantée en C (norme 99) et ne doit dépendre d'aucun code source autre que la librairie standard. Une telle contrainte assure que l'implantation soit claire, le code source portable, et la compilation aisée.

Cette implantation minimale va de paire avec l'idéologie derrière Eva : prioriser l'accessibilité et la fiabilité. De plus, l'aspect sécurité est d'une grande importance pour le projet, où la correction de code aura toujours priorité sur l'ajout de nouvelles fonctionnalités.

7 Disponibilité du projet

Eva est un projet publique, dont le code et les binaires seront accessibles en open-source, suivant une licence MIT.

Cela veut dire que n'importe quel utilisateur peut télécharger, lire et modifier le code source. Le projet étant pédagogique, il va de soit que chaque composante du projet soit publique et accessible pour le plus grand nombre.

8 Projets connexes

En plus du développement de la machine virtuelle, Eva sera également le socle de plusieurs projets dont divers compilateurs ciblant EVA. Parmi ces compilateurs l'équipe d'EVA développera et maintiendra un langage de programmation de haut niveau spécifique à EVA. Ce dernier permettra de complémenter les possibilités offertes par le simple langage assembleur supporté nativement par la plate-forme.

Références

[1] Harold Abelson and Gerald Jay Sussman *Structure and Interpretation of Computer Programs* MIT, 1996 2

[2] Joseph Weisbecker *CHIP-8 programming language and virtual machine* 1978 2

Instruction	Description
ADD	Addition
ADC	Addition avec retenue
BIC	RAZ de bit
CMP	Comparaison
MOV	Écriture de valeur dans le registre
B	*Branch Jump*
BL	*Branch Link*
PUSH	Empiler
POP	Dépiler
SUB	Soustration avec retenue

Table 1 – Liste des op-codes de la machine virtuelle d'Eva.