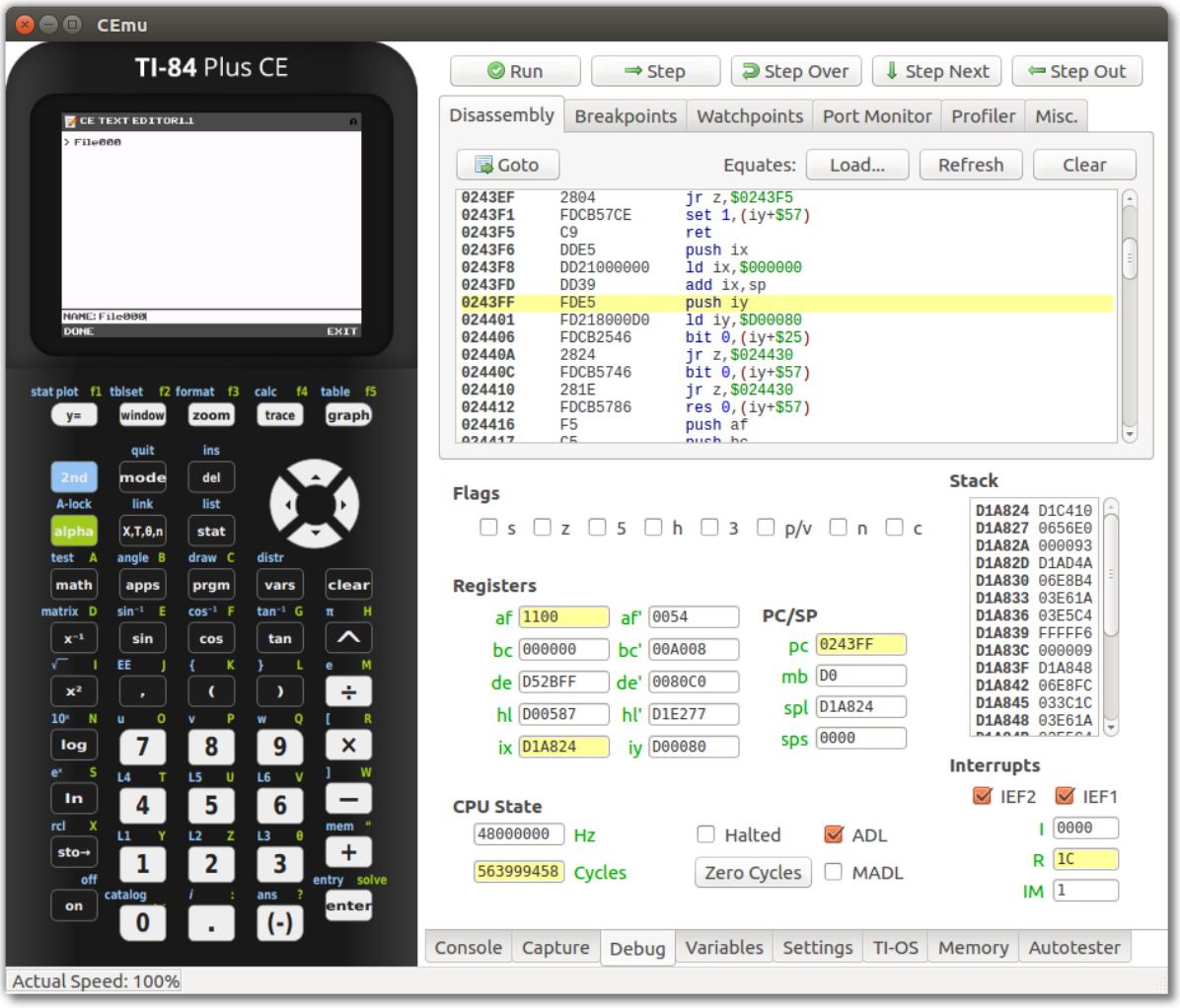
Rapport de Projet

P28 Adaptation d'un émulateur de calculatrice TI



POLYTECH LILLE

Université de Lille 1 SCIENCES ET TECHNOLOGIES

Rodolphe TOIN

Encadrants: Gilles Dhauzy
Xavier Redon

Sommaire

Introduction

Texas Instruments est un constructeur renommé de calculatrices adaptées à tous niveaux. Cependant, les logiciels joints avec leurs calculatrices sont limités et souvent peu flexibles avec l'environnement informatique, offrant un usage convenable dans un cadre individuel basique mais complexe dans un cadre éducationnel.

CEmu est un émulateur open-source réalisé par la communauté et visant à regrouper un maximum de fonctionnalités. Ce logiciel cherche à être plus flexible qu'un émulateur comme TI Smart View CE qui est officiel mais payant et restreint.

L'objectif de ce projet sera de faire au mieux pour enrichir l'émulateur CEmu afin de se rapprocher d'une émulation totale de la calculatrice TI 83 Premium CE.

Je tiens ici à remercier Adrien Bertrand, contributeur du projet CEmu, pour l'éclairage qu'il m'a apporté concernant le fonctionnement du logiciel.

Objectifs et cahier des charges

Plusieurs objectifs à réaliser sont fixés:

- adapter l'interface graphique (de TI-84 plus CE vers TI-83 premium CE)
- permettre aux programmes en mémoire flash (archivés) de s'exécuter dans l'émulateur
- permettre la sauvegarde structurée de la mémoire sans utilisation d'utilitaire
- permettre la gestion du port USB
- donner la possibilité d'éditer des programmes
- contourner le mode examen par ajout d'une LED afin de fournir une information contradictoire sur son activation

Environnement de travail et compilation

Le logiciel CEmu possède une interface graphique développée avec le logiciel QT. Ce logiciel permet notamment d'utiliser des composants d'interface graphique (widgets) qui permettent de gérer les différentes parties du logiciel.

Afin de pouvoir compiler le projet CEmu, je travaille sous environnement linux possédant gcc pour la compilation et avec la version 5.8.0 du logiciel QT. Les fichiers du logiciel ont été récupéré avec un *git clone* depuis le dépôt git disponible en ligne. Après ajout du fichier bin/ de QT au PATH de l'environnement, la compilation peut simplement s'effectuer en se plaçant dans le dossier /gui/qt/ du projet et en exécutant la ligne de commande suivante:

qmake -r CEmu.pro && make

Le fichier CEmu.pro fait ici référence à un fichier projet réalisé sous QT.

Après compilation, nous obtenons un exécutable nommé CEmu. Le logiciel n'est cependant pas utilisable tel quel: il nécessite l'utilisation d'une ROM de la calculatrice. Cette image ROM m'a été difficile à acquérir: en effet, après une tentative infructueuse de récupération de la ROM via le logiciel open-source TiLP (logiciel de communication entre ordinateur et calculatrice TI) je n'ai pas non plus pu récupérer cette image de ROM via le logiciel officiel TI Connect CE. Une telle image de ROM étant protégé par des licences, il était donc impossible d'en trouver une en ligne et je suis donc resté plusieurs semaines sans pouvoir tester le logiciel. Il a fallu attendre que mon disque dur rende l'âme pour que je réinstalle mon système et que le logiciel TI Connect CE se mette à fonctionner.

Après récupération des images de ROM de la calculatrice (au nombre de 12), les fournir à CEmu permet de créer un seul fichier représentant la ROM de la calculatrice: ce fichier sera en permanence utilisé par l'émulateur afin de réaliser les fonctions habituelles de la calculatrice.

Architecture de CEmu

CEmu se compose de 2 éléments centraux:

- **le cœur d'émulation**, entièrement rédigé en C et qui consiste à exécuter des fonctions de la calculatrice en utilisant la ROM afin de reproduire le comportement de celle-ci ;

- **le GUI (ou Graphical User Interface)**, rédigé en C++ et en Qt et qui utilise des fonctions du cœur afin de réaliser un échange entre les demandes de l'utilisateur et la calculatrice.

La gestion de l'interface graphique est gérée par Qt, permettant un affichage par module et fenêtres ajustables. L'ensemble du logiciel tient sur plusieurs milliers de ligne et bien que la rédaction soit soignée, il n'y a que très peu de commentaires, voir pas du tout.

Interface graphique

Concernant l'interface graphique de la calculatrice c'est à dire véritablement la représentation de l'écran et des touches dans le logiciel CEmu, il a été simple d'appliquer des changements. La représentation de l'écran est réalisé avec une simple image PNG (située au chemin *\gui\qt\resources\skin*) qu'il suffit d'éditer avec un logiciel de retouche d'image.

Concernant les touches, celles-ci sont crées en C++ pour être gérées comme des objets. Les fichiers concernant les touches sont situés dans le répertoire *\gui\qt\keypad*. Les touches sont d'une part définies graphiquement: par exemple, les touches numériques sont définies dans le fichier *numkey.h*. Il est alors simple en ajustant quelques valeurs de modifier l'apparence des touches (comme visible sur la capture d'écran ci-dessus).

Chaque touche est traitée comme un objet et la référence des fonctions de chaque touche est implémentée dans le fichier *keymap.cpp* qui répertorie l'ensemble des actions réalisables par les touches.

Autres fonctionnalités et limitations

En cherchant à atteindre les autres objectifs de mon projet je me suis systématiquement confronté à des difficultés techniques qui m'ont parues insurmontables. J'aimerai par la suite expliquer chaque problème rencontré.

1) Gestion du port USB

Rien de concret n'est encore disponible concernant la gestion du port USB. Les ressources initialement disponibles dans CEmu ne réalisent qu'une bref initialisation de certaines variables mais le protocole USB est entièrement absent. Le module est encore à l'état de brainstorming entre les contributeurs de CEmu.

Cela signifie que lorsqu'on parlera d'échange entre calculatrice et ordinateur on parlera en réalité d'échange entre l'émulateur et l'ordinateur. CEmu ne possède pas de moyen de communication directe avec la calculatrice physique.

Cela signifie d'emblée qu'une gestion de CEmu via l'interface de la calculatrice physique n'est ici pas envisageable. Une telle fonctionnalité supposerait une gestion fonctionnelle des échanges USB entre l'émulateur et la calculatrice ainsi qu'un ensemble de fichiers permettant d'analyser les signaux de la calculatrice et d'appliquer les actions correspondantes dans CEmu.

2) Permettre aux programmes en mémoire flash (archivés) de s'exécuter dans l'émulateur

Dans la calculatrice, la distinction est faite en l'espace RAM directement accessible pour l'exécution et la mémoire Flash appelée Archive. Un programme stocké en Archive ne peut, dans des conditions normales d'utilisation, pas être exécuté.

Il s'avère que rendre possible l'exécution de programmes en mémoire flash est une tâche complexe. Après de nombreuses recherches sur le sujet, la méthode la plus adéquate semble l'utilisation de programmes pour la calculatrice. PHASM et Cesium sont deux programmes permettant l'exécution d'autres programmes archivés.

https://github.com/MattWaltz/cesium

https://github.com/MathisLav/PHASM

Afin d'utiliser PHASM par exemple, il faut charger les deux éléments du programme dans la calculatrice: un fichier .8xp qui se placera dans la RAM et un fichier .8xg qui se placera en archive. Le programme se lance alors avec la commande assembleur Asm(pgrmPHASM) permettant une exécution continue du programme tout en pouvant continuer d'utiliser normalement la calculatrice. L'exécution de programmes archivés devient alors possible et le programme offre également des possibilités d'édition de mémoire.

Néanmoins, un tel programme est codé en format .z80 (respectivement en .asm pour Cesium) qui correspond à un langage assembleur. Celui-ci comporte plusieurs centaines de lignes de code assembleur.

Un compromis pourrait être de scripter l'envoi et l'exécution de ce programme par CEmu. Le seul élément permettant la réalisation d'une telle méthode est l'autotesteur du logiciel. Cette fonctionnalité est cependant récente et se base sur l'utilisation de scripts rédigés en .json (JavaScript Object Notation). Outre le fait que je ne maitrise pas une telle rédaction, CEmu resterai toujours dépendant d'un programme externe et cette fonction d'autotest reste une fonctionnalité du mode développeur.

3) Donner la possibilité d'éditer des programmes

En soi l'édition de programmes est disponible dans CEmu: il est en effet possible d'éditer un programme en utilisant les touches de la calculatrice et donc une navigation classique.

Cependant, pour permettre une édition au clavier de l'ordinateur il faudrait pouvoir réaliser une conversion entre un format assimilable à du texte. Les programmes de la calculatrice sont, comme déjà évoqué, au format 8xp. Ce format n'est pas éditable par le moyen d'un éditeur de texte classique. Ils sont le fruit de la compilation d'un code en langage assembleur type .asm ou .z80 comme nous l'avons vu précédemment.

Pouvoir taper un programmer au clavier dans CEmu semble dès lors très complexe étant donnée que la calculatrice possède un catalogue de fonctions très vaste et dont la syntaxe nécessiterai une certaine adaptation. Une telle fonctionnalité nécessiterai la mise en place d'un système de traduction complexe afin de pouvoir convertir des fichiers éditables classiques en programme lisibles par la TI.

J'ai cherché à m'inspirer du mode "Variables" de CEmu qui permet une visualisation des variables de la calculatrice et ainsi des programmes stockés. Cette fonctionnalité passe cependant par des fonctions internes de la calculatrice et il m'a été impossible de déterminer quelle fonctions étaient réellement à l'œuvre.

4) Contournement du mode examen par ajout d'une LED afin de fournir une information contradictoire sur son activation

Compte-tenu de l'absence de documentation constructeur sur la TI 83 Premium CE (pour des raisons évidentes de protection de la technologie) et compte-tenu de la complexité du circuit imprimé et de l'absence d'accès clair à la LED, cette réalisation a été rapidement abandonnée.

Conclusion

Ce projet a été pour moi un véritable échec. Peu importe l'objectif souhaité, je me suis toujours retrouvé confronté à des difficultés techniques dues soit à la complexité du langage, soit à la complexité du code lui-même et à la difficulté de comprendre comme chaque fonction intervient et doit être utilisée. En soi, mon projet n'a pas de réalisation finale, outre les petites modifications de l'interface graphique qui m'ont été accessibles suite aux cours de Java de cette année et je suis profondément déçu de ne pas avoir abouti à du code fonctionnel.

Je regrette aussi d'avoir été confronté à ce problème de récupération de ROM qui ne m'a permis de tester le logiciel que tardivement et de me rendre compte trop tard de la complexité du travail à réaliser. J'aurai du signaler ces difficultés le plus tôt possible.

Souhaitant orienter ma recherche de stage pour la 5ème année dans le développement logiciel, ce travail m'a tout de même permis de comprendre la complexité d'un tel projet, la polyvalence dont il faut savoir faire preuve (en terme de langages par exemple) et aussi la nécessité de détailler le fonctionnement d'un code. J'ai pu être un peu conseillé par des contributeurs de CEmu mais n'ayant pas moi-même rédigé le code du logiciel il aurait presque fallut que je sois en contact constant avec ces personnes (ce qui n'était pas possible pour eux).