Histoire de l'émulateur To7

par Sylvain Huet

L'émulateur To7 - version Unix, fév 94

Comme beaucoup de Français de mon âge, j'ai commencé l'informatique avec un ordinateur Thomson. C'était un To7, en 1982, avec 6Ko de mémoire utilisateur (comme on disait à l'époque) et un lecteur de cassettes. Je m'en suis servi jusqu'en 1989.

J'ai réalisé la première version de l'émulateur To7 en février 94. A l'époque, j'étais élève à l'Ecole Polytechnique, où nous avions la chance d'avoir une liaison internet à 10Mbits/s dans nos chambres. Nous étions un certains nombre à avoir des PC sous Linux, qui étaient comparables en terme de performance aux stations Unix de l'Ecole. C'est donc naturellement que j'ai réalisé la première version de l'émulateur sous Unix, la version Msdos n'étant venue que plus d'un an plus tard.

Je me suis fixé dès le début l'objectif de réaliser un émulateur totalement portable : il fallait qu'il fonctionne aussi bien sous Linux que sur les stations Sun, Dec et Hp de l'Ecole. J'ai donc choisi de le réaliser entièrement en C, en utilisant la LibX11 (la couche la plus basse de X-Window).

Le gros du développement a nécessité quatre jours.

La première étape consiste à récupérer les Roms du To7. Pour cela je comptais utiliser un programme que j'avais réalisé deux ans auparavant sur Atari ST, qui permettait de lire les disquettes Thomson. Malheureusement, le jour où j'ai ressorti mon To7 pour copier les Roms sur disquette, le transformateur du lecteur de disquettes a grillé. J'ai trouvé une solution de rechange, nettement moins efficace : en tendant trois fils entre le port manettes de jeux du To7, redirigé en sortie, et le port imprimante de l'Atari, redirigé en entrée, j'ai réalisé une liaison d'une qualité incompréhensiblement désastreuse (environ 80 bauds). Ce fut cependant suffisant pour transférer la Rom moniteur, les cartouches Basic et Assembleur, ainsi que le contenu de quelques cassettes (Eliminator et Pingo).

La seconde étape consiste à mettre en place la base de l'émulateur : il s'agit de réaliser les fonctions gérant une bande mémoire, le chargement de fichiers, l'édition d'octets dans cette bande, et surtout le désassembleur sans lequel rien n'est possible ensuite. Ceci demande une journée complète, et se révèle assez fastidieux : il faut entrer la correspondance de tous les OpCode vers le mnémonique et vers le type d'adressage.

La troisième étape est le cœur de l'émulateur : l'interpréteur de langage machine 6809. Le principe de fonctionnement est relativement simple : on dispose d'une bande mémoire représentant l'espace d'adressage du 6809, et de variables représentant les différents registres. A chaque boucle, on lit l'Opcode de l'instruction courante, une table donne la routine à exécuter, cette routine effectue l'instruction sur les registres ou en mémoire.
Il y a dans le 6809 (et plus généralement dans tout microprocesseur) un registre particulier appelé "registre d'état". Ce registre contient plusieurs bits qui sont susceptibles d'être modifiés à chaque instruction. Les quatres principaux sont :

. la retenue, qui indique un débordement suite par exemple à une addition

. le bit de signe, qui donne le signe du résultat de la dernière opération
. le bit de zéro, qui indique si le résultat est égal ou non à zéro
. le bit de débordement, qui indique un changement de signe inopiné

La gestion du registre d'état nécessite certes une documentation précise du jeu d'instruction mais aussi une stratégie claire d'émulation. On s'aperçoit en effet rapidement que l'émulation d'une instruction (l'addition par exemple) nécessite plusieurs lignes de C dont une seule code véritablement l'addition, tandis que les autres codent la gestion du registre d'état.
J'ai pris le parti de réaliser l'émulation du registre d'état sans le moindre branchement conditionnel, en utilisant uniquement des opérations logiques (And, Or, Eor, décalages de bits) sur des registres spéciaux. Ceci permet de ne pas créer de rupture dans le pipe-line, et accélère les choses, à mon avis, de manière importante.
Il faut un jour complet pour réaliser l'émulation du 6809 et, malheureusement, on travaille sans filet : comment tester les 200 ou 300 instructions émulées, avec les différents modes d'adressage indexés, indirects ou relatifs ? Le seul test valable est de lancer l'émulateur sur la Rom du To7 et d'avoir une sortie graphique, pour voir apparaître l'immonde page d'accueil sur fond bleu ciel. Ceci n'est pas possible avant le quatrième jour.

En effet, la quatrième étape consiste à émuler l'électronique qui se trouve autour du 6809 du To7. Le plus important est la sortie graphique, car elle permet de faire les premiers tests de l'émulateur. Vient ensuite l'émulation du clavier, qui permet de taper un programme en Basic. Puis le lecteur de cassettes, le crayon optique, et les manettes de jeu.
Sous Xwindow, il n'est pas possible d'écrire directement dans l'écran : j'ai utilisé ce qu'on appelle en jargon Xwindow des "images", c'est-à-dire des buffers graphiques situés et manipulés dans le programme, que l'on peut transférer vers l'écran (l'architecture client-serveur d'Xwindow s'avère ici un peu pénible). L'émulateur utilise un buffer monochrome, que l'on transfère vers l'écran en spécifiant couleur fond et couleur forme : cette idée vient de JF. Lozevis, un X qui venait de réaliser sous Unix un émulateur Spectrum.
L'affichage du To7 est cependant plus complexe que celui du Spectrum, puisqu'il y a sur le To7 un code couleur pour chaque segment de 8 points et non pas pour chaque carré de 8 sur 8. Ceci devait donc ralentir l'émulation puisque, mes "images" monochromes pouvant se limiter à des segments et non des carrés, l'émulateur To7 était susceptible d'envoyer huit fois plus de commandes graphiques que l'émulateur Spectrum.
Considérant le ridicule qui ne manquerait pas de s'abattre sur moi si j'essayais d'expliquer que mon émulateur ramait à cause des performances graphiques exceptionnelles du To7, je réalisai qu'en fait la possibilité d'avoir des codes couleurs différents à l'intérieur d'un même carré était rarement exploitée. En construisant la gestion graphique autour de cette idée, j'obtins des performances d'affichage satisfaisantes.

Au soir du quatrième jour, l'émulateur affichait la page de présentation du To7, qui se rappelait à mon souvenir après cinq ans d'absence.

Il fallut encore quelques jours pour mettre en place, entre deux cours, les derniers éléments, et une nuit blanche pour corriger un bug particulièrement déplaisant : sous l'émulateur, en Basic, on pouvait entrer une ligne de type "10 print"coucou":goto10", mais lorsqu'on tapait "LIST", au lieu de voir s'afficher la ligne, un message d'erreur "?SN Error" apparaissait, et l'émulateur continuait de fonctionner. Il s'agissait d'une erreur de gestion de la retenue dans l'instruction NEG, résultant d'une coquille dans ma documentation...

Grâce au lecteur de Roms d'un ami, je récupérai alors des cartouches supplémentaires : Airbus, Logicod et Blitz. Et sur une vieille disquette To8, je récupérai le mythique Aigle d'or,

pour lequel je réalisai une variante de l'émulateur, sorte d'hybride entre le To7 et le To7-70, histoire d'avoir assez de mémoire et 16 couleurs.

L'émulateur To7 - version Msdos, octobre 95

Après la version Unix, il était évident qu'une version Msdos permettrait une meilleure diffusion. Cependant par paresse, je ne souhaitais pas avoir à être confronté aux segments de 64ko du PC. Aussi j'attendis d'avoir un compilateur C pour dos4g. Une difficulté subsistait : comment gérer le clavier sous Msdos, et notamment, comment prendre connaissance des touches relâchées ? car l'émulateur en a besoin. Réécrire l'interruption n'est pas chose aisée dès lors que le programme oscille entre mode réel et mode protégé. Aussi j'ai attendu de savoir faire des interruptions bimodales pour écrire la version Msdos. C'était en octobre 95.
La version Msdos offre plusieurs avantages. Tout d'abord l'affichage est beaucoup plus simple à gérer, puisqu'on peut écrire directement dans la mémoire écran. Ensuite, le PC dispose d'un buzzer qui fonctionne exactement comme celui du To7 (sauf que celui du PC contient un oscillateur en plus). Il fut très facile de redonner la "parole" au To7 ainsi qu'au lecteur de cassettes, ce qui ne se fit pas sans une certaine émotion, la mémoire auditive étant assez fidèle.

Le convertisseur de cassettes - décembre 95

Pour tester un émulateur, le plus simple est d'essayer de lui faire charger le plus de programmes possible. La lenteur de ma liaison To7-Atari ST m'avait découragé de récupérer d'autres programmes que ceux cités plus haut.
Pour aller plus loin, il fallait pouvoir lire les cassettes. Le principe est là encore très simple : à partir d'une chaîne hifi reliée au PC, on digitalise le son d'une cassette sous la forme d'un fichier .Wav. La structure de ce fichier est déroutante de simplicité : après un en-tête relativement court et compréhensible, on trouve, sans traitement particulier, la liste des échantillons. A partir de ces échantillons, on reconnaît un signal sinusoïdal qui utilise deux fréquences : 5 périodes à 4,5 kHz pour coder le bit 0, et 7 périodes à 6,3 kHz pour le bit 1. En désassemblant la Rom du To7, on découvre que chaque octet est codé sur onze bits : un bit 0, les huit bits de l'octet à coder, et deux bits 1.
Il suffit de détecter les périodes du signal sur le .Wav, de reconnaître les bits, et de reconstituer les octets. Après avoir peiné pour réaliser le court programme qui effectue ce traîtement, je me suis aperçu qu'en utilisant l'équalizer sommaire de ma chaîne, j'obtenais un signal enregistré nettement meilleur, oscillant typiquement entre 64 et 192, alors que j'avais fait tout le développement avec un signal oscillant entre 127 et 129...

J'ai alors récupéré tout ce que j'avais sous la main : les quelques jeux que j'avais (notamment PulsarII), mais aussi tous mes propres programmes. Je suis vraiment étonné de la qualité d'enregistrement des cassettes : plus de 10 après, j'ai pu tout récupérer.

Du site Web au To7-70 - printemps 96

La diffusion de l'émulateur était restée jusque là confidentielle : seuls quelques amis disposaient d'une version, et dans le forum électronique de l'Ecole Polytechnique, un groupe évoquait encore l'existence de ce programme. En mars 96, un élève posta dans le forum un

message demandant où il pouvait se procurer l'émulateur. Ceci aboutit à la création de ma page Web, à l'Inria (Institut national de recherche en informatique et en automatique) où j'effectuais un stage. A cette occasion, je découvris les News, et postai un message qui allait accélérer les choses. Quelle ne fut pas ma surprise de trouver un pointeur vers ma page web sur le site d'un certain Chrissalo, avec le commentaire : "There are many computers where I never expected that an emulator would be written for, but this is (so far) the only emulator I know, where I hadn't even heard of the computer which is emulated. I think it is a french computer."

Pcteam inclut l'émulateur dans le CD de juin 96. Je souhaitais réaliser une version To7-70, mais pour cela j'avais besoin d'un To7-70 et plus particulièrement de sa Rom. Grace aux News, je fis la connaissance de David Winter, un collectionneur de vieilles machines qui me prêta gracieusement l'une d'elles, ainsi qu'un stock respectable de cassettes et de cartouches.

Cette fois, grâce au convertisseur de cassettes, la récupération de la Rom ne prit que cinq minutes. La conversion prit une courte après-midi, les deux machines étant très proches, et l'émulateur To7 disposant déjà d'un mode évolué 16 couleurs.

Je rencontrai cependant une difficulté inattendue, en essayant de charger les programmes d'Infogrames. Androides, Sortileges, Vampire, Vera Cruz, ..., utilisent un loader particulier : il suffit de taper "LOADM" pour charger et lancer le programme. N'arrivant pas à comprendre pourquoi cette simple commande permettait de lancer le programme, je contactai une vieille connaissance, Eric Mottet, d'Infogrames, qui m'orienta vers William Hennebois, l'auteur d'Androides, qui travaille encore aujourd'hui dans la société de Villeurbanne. Malheureusement, sa mémoire n'était pas très précise, le temps ayant accompli son œuvre dévastatrice. La difficulté d'un tel obstacle réside dans le fait que le problème est difficile à localiser a priori : erreur dans la lecture des cassettes, bug dans l'émulation d'une instruction du 6809, absence d'émulation d'une fonctionnalité de l'électronique du To7-70. Tout est possible, ce qui est un peu décourageant. Le plus ennuyeux aurait été que le loader d'Infogrames n'utilise pas la routine Rom de lecture d'un octet sur cassettes (car l'émulateur gère les cassettes en "patchant" cette routine), mais une routine "maison".

Finalement, début juin, je compris que le loader de William utilisait une instruction non documentée du 6809, d'Opcode 01, qui visiblement se comporte comme un NOP, mais de taille 2 (en fait comme un BRN), alors que l'émulateur considérait les instructions inconnues comme des NOP de taille 1.
Galvanisé par cette découverte, je remis tout au propre, et programmai une émulation fine des PIA 6821 (avec gestion des bits de direction). Cela aboutit à la version de l'émulateur disponible depuis juin 1996 sur ma page Web (http://pauillac.inria.fr/~shuet).

Et maintenant ?

Il reste maintenant à faire l'émulateur du To8 ; je compte m'y mettre dès que je trouverai un peu de temps.En effet je souhaiterais refaire marcher l'adaptation de "macadam bumper" que j'avais réalisée pour Ere Informatique en 1989.
Les deux principales difficultés de la version To8 sont l'émulation des modes graphiques, et celle du lecteur de disquettes. Je pense que l'émulation fine des deux est difficilement envisageable. Notamment, la gestion du lecteur de disquettes doit être "patchée" au niveau de la routine 'lire secteur', ce qui risque de poser des problèmes avec les protections disquettes qui manipulent directement le contrôleur de disque. Quant à l'écran, la gestion fine des

signaux de contrôle est particulièrement délicate, surtout si l'on souhaite qu'ils correspondent à ceux de l'écran du Pc (notamment le retour vertical du rayon).

23 nov 1996.