Une BasicCard
Pour téléphones portables GSM

Découvrez les secrets de votre téléphone GSM

Patrick Gueulle

Une carte à puce du type BasicCard programmée constitue le « nerf » de cet article. Nous allons voir comment utiliser cet « émulateur de carte SIM » pour tester les principales fonctions accessibles par le « menu » normal et le « menu » de maintenance de certains portables GSM.

La carte à puce « SIM » (*Subscriber's Identification Module*) est le véritable « sésame » indispensable à l'utilisation de tout téléphone portable GSM. Contenant tout à la fois les identifiants secrets et les données personnelles de l'utilisateur, elle est émise par un opérateur de téléphonie mobile, qui la confie à son client mais en conserve généralement la propriété.
Une carte à puce à système d'exploitation ouvert, telle que la BasicCard, peut toutefois être transformée en carte SIM « de développement », autorisant une foule d'expérimentations en toute indépendance vis-à-vis des opérateurs.

La BasicCard « Professionnelle »

Depuis le lancement, en 1998, de la BasicCard « *compact* », ZeitControl a progressivement introduit des versions plus puissantes, dites « *enhanced* », puis tout récemment « *professional* ».
Avec la ZC 4.1, c'est une étape majeure qui vient d'être franchie, puisqu'il s'agit de la toute première BasicCard supportant le protocole « T=0 ».
Rappelons, en effet, que toutes les versions précédentes de la BasicCard fonctionnaient exclusivement en protocole « T=1 », particulièrement populaire en Allemagne.
Comme la spécification GSM 11.11 de l'ETSI impose le protocole « T=0 », d'ailleurs de très loin le plus courant, pour les cartes SIM, il n'était jusqu'à présent pas possible de faire fonctionner une BasicCard dans un téléphone portable.
Il faut savoir que la BasicCard « *professional* » est équipée de l'une des « puces » les plus performantes d'ATMEL, la AT90SC3232C.
Avec 32 Koctets de mémoire Flash pour le système d'exploitation, 32 Koctets d'EEPROM pour les programmes « utilisateur », 1 Koctet de RAM, et un processeur RISC « AVR » secondé par un coprocesseur cryptographique, ce composant parfaitement adapté à la production de cartes SIM « Phase 2+ » et « SIM Toolkit », n'est guère devancé que par sa variante AT90SC6464C, prévue notamment pour l'UMTS.
C'est dire qu'il s'agit d'une puce de

Tableau 1.
Le « dictionnaire » des commandes normalisées GSM 11.11.

```
20h : verify CHV
24h : change CHV
26h : disable CHV
28h : enable CHV
C0h : get response
32h : increase
04h : invalidate
B0h : read binary
B2h : read record
44h : rehabilitate
88h : run GSM algorithm
A2h : seek
A4h : select
FAh : sleep
F2h : status
2Ch : unblock CHV
D6h : update binary
DCh : update record
10h : terminal profile
C2h : envelope
12h : fetch
14h : terminal response
```

haut de gamme, dont l'originalité majeure est d'être entièrement réalisée en technologie « Flash EEPROM ».
Le système d'exploitation embarqué est ainsi téléchargé lors de la personnalisation de la carte, et non plus « masqué » dans une zone de ROM lors de la fabrication de la puce, ce qui est infiniment plus souple.
Dans une carte SIM conventionnelle, c'est généralement un système d'exploitation dédié qui est programmé dans cette zone.
La partie EEPROM accueille alors le système de fichiers normalisé GSM 11.11, et s'il s'agit d'une SIM « Phase 2+ », d'éventuelles applications « SIM Toolkit » (souvent des « *applets* » Java) implantées par l'opérateur ou par un fournisseur de services (banque, par exemple).
Dans une BasicCard, le système d'exploitation est de type « ouvert », tout comme celui des Javacard, Multos, et autres Windows for smart cards.
Il s'agit, en l'occurrence, d'un interpréteur permettant d'écrire les programmes applicatifs dans ce langage si populaire qu'est le BASIC, que l'on dirait d'ailleurs « taillé sur mesures » pour les cartes à puce.
Une fois mises au point à l'aide d'un outil de développement performant mais néanmoins gratuit (*http://www.basiccard.com*), les applications sont téléchargées dans la zone EEPROM.
Dans le présent projet, un programme BASIC de près de 450 lignes occupe environ 17% de la zone EEPROM, le solde étant partagé entre un système de fichiers compatible GSM 11.11 et un gros fichier « log », dans lequel la carte pourra enregistrer, sur demande, l'historique des commandes qu'elle recevra de l'extérieur.

Une carte SIM de développement

La vocation d'une carte SIM de développement n'est évidemment pas de téléphoner (encore qu'en principe, elle puisse tout de même permettre d'appeler le 112), mais plutôt de rendre possibles toutes les manipulations normalement irréalisables avec une carte émise par un opérateur.
Dans une « vraie » carte SIM, en effet, les fichiers les plus « sensibles » sont protégés, en écriture et parfois même en lecture, pas des codes « administrateur » connus seulement de l'opérateur qui l'a émise.
Dans le cadre de ce projet, tous les codes confidentiels (y compris les « PIN » de l'utilisateur) sont désactivés une bonne fois pour toutes, ce qui libère à 100% l'accès à l'ensemble des fichiers existants.
On se gardera bien d'en déduire qu'il suffirait de recopier les identifiants d'une carte SIM valide, pour en créer un « clone » utilisable.
Deux « garde-fous » infranchissables évitent en effet que notre projet ne puisse être détourné vers des applications frauduleuses :
– Tout d'abord, l'algorithme cryptographique (A3/A8) assurant l'authentification des cartes SIM au travers des réseaux GSM est ici purement factice, et il ne saurait être question de le modifier sans avoir accès au code source.
– De plus, cloner une carte supposerait que l'on recopie, à l'identique, la clef secrète qui réside dans une zone de l'original rigoureusement verrouillée en lecture.
En revanche, l'écriture de valeurs bien particulières dans certains fichiers peut permettre, par exemple, d'utiliser pour nos manipulations des téléphones verrouillés (« Simlockés ») par l'opérateur qui en a sans doute « subventionné » la fourniture, et qui ne peuvent normalement fonctionner qu'avec leur carte SIM d'origine.
De même, notre carte donnera souvent accès aux menus « cachés » des téléphones (notamment de marque Motorola), réservés en principe aux seuls techniciens habilités.
Il n'en faut pas davantage, par exemple, pour obtenir l'affichage d'un code de sécurité malencontreusement oublié, ou pour activer des fonctions de repérage des relais des différents réseaux disponibles en un lieu donné.
Mais il y a mieux : la carte contient en effet suffisamment de mémoire EEPROM pour pouvoir enregistrer, dans un fichier interne, toutes les commandes que lui envoie le téléphone, ou n'importe quel autre « terminal » dans lequel on voudra bien l'introduire (par exemple un PC exécutant une application de gestion de cartes SIM).
Télédéchargé ensuite par un utilitaire approprié, ce fichier « LOG » pourra être analysé à loisir, à des fins didactiques ou de diagnostic.
Enfin, notre carte est compatible « Proactive SIM », ce qui signifie que, contrairement à une simple carte SIM de phase 1 ou 2, elle ne se contente pas d'exécuter les ordres que lui adresse le téléphone : elle peut, de sa propre initiative, prendre le contrôle de celui-ci, et notamment de son afficheur ou de son écouteur.
Une petite application indépendante a ainsi pu être implantée, qui ne s'exécute que si le téléphone est compatible « phase 2+ ». Moyennant quoi, l'écran affiche, au bout d'une minute environ, un mot hexadécimal (le « *Terminal profile* ») dont le décodage fournit un diagnostic détaillé des possibilités « SIM Toolkit » du mobile.
Par la suite, la carte s'immiscera toutes les minutes dans le fonctionnement du téléphone (sans pour autant le perturber le moins du monde) afin de lui faire afficher un court instant « BasicCard » en diffusant une petite mélodie.

Entrons dans le détail

Notre carte SIM de développement reconnaît, et c'est la moindre des choses, la totalité des commandes normalisées (voir **tableau 1**) qu'un mobile GSM est susceptible de lui envoyer dans l'état actuel de la normalisation (phases 1, 2, et 2+).
Reconnaître ne signifie pas nécessairement exécuter, en particulier lorsqu'il s'agit de commandes liées aux codes confidentiels.
Même si la carte ignore purement et simplement telle ou telle commande, elle y répondra toujours par un compte-rendu permettant la

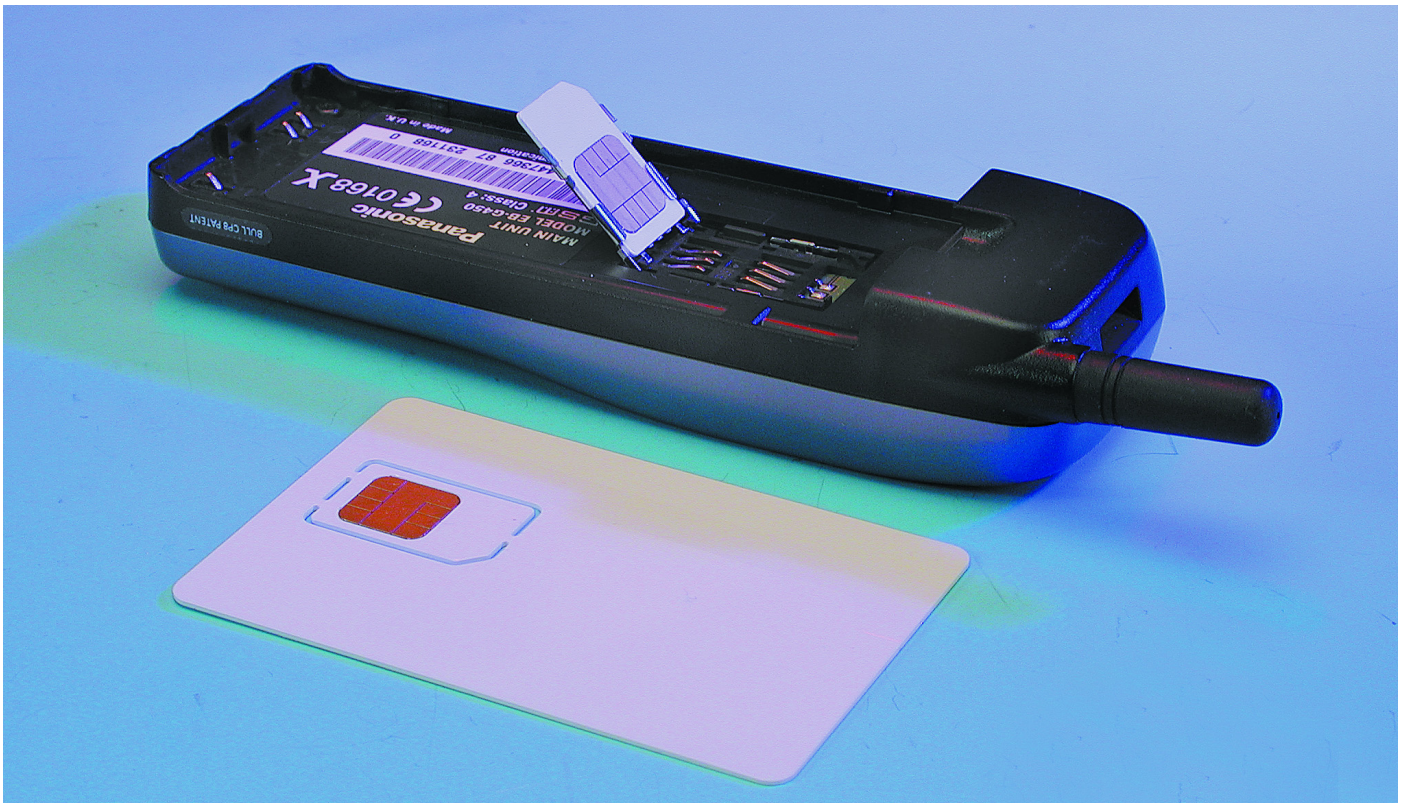
poursuite de la « session GSM » en cours.
Si, par exemple, on demande l'activation du code PIN, la carte répondra « bonne exécution » (9000h) mais ne réclamera pas, pour autant, ce code par la suite.
De même, si on tente de modifier le PIN, la carte répondra que l'opération est exécutée, mais continuera imperturbablement à considérer n'importe quel code comme étant le bon. Ce stratagème permet, et c'était précisément le but recherché, de jouer librement avec tous les fichiers de la carte.
Cela étant, il est indispensable de préciser ici comment est structurée une commande GSM. Non pas tant pour en construire de toutes pièces (ce sera plutôt le rôle de logiciels spécialisés), mais pour pouvoir interpréter, le moment venu, le contenu de ce fameux fichier « log ».
Les cartes SIM fonctionnant en protocole « T=0 », toute commande destinée à la carte commence par cinq octets d'en-tête (*header*) : CLA INS P1 P2 LEN.
CLA est la « classe ISO » de la commande, toujours égale à A0h pour une carte SIM.
INS est un code opératoire appartenant à la liste du tableau 1.
P1 et P2 sont deux paramètres dont la signification varie d'un code opératoire à l'autre, et qui restent très souvent, par défaut, à 00h.
LEN, enfin, indique la longueur du bloc de donnée qui suit, à moins qu'il n'y en ait pas (LEN = 00h).
Il faut distinguer les « commandes entrantes » (mobile vers carte), dont le bloc de données est émis à la suite de LEN, des « commandes sortantes » (carte vers mobile) où la carte est priée de renvoyer LEN octets de données.
Dans quasiment tous les cas, la carte retourne au minimum deux octets de « compte-rendu » (SW1 et SW2) qui, lorsque tout s'est bien passé, sont respectivement 90h et 00h.
Lorsque la carte a des données à renvoyer en réponse à une com-

Tableau 2.
Les fichiers disponibles dans notre carte.

```
Répertoire racine :
2FE2 (ICCID)

Répertoire Télécom :
6F3A (ADN, Abbreviated Dialling Numbers)
6F3B (FDN, Fixed Dialling Numbers)
6F3C (SMS, Short Messages)
6F40 (MSISDN, Own Numbers)
6F42 (SMSP, Short Messages Service Parameters)
6F43 (SMSS, SMS Status)
6F44 (LND, Last Number Dialed)

Répertoire GSM ou DCS :
6F05 (LP, Language preference)
6F07 (IMSI, International Mobile Subscriber Indentity)
6F20 (Kc, Ciphering Key)
6F30 (PLMN, Preferred PLMNs)
6F31 (HPLMN search period)
6F38 (SST, SIM Service Table)
6F3E (GID1, Group Identifier level 1)
6F3F (GID2, Group Identifier level 2)
6F74 (BCCH, Broadcast Control Channels)
6F78 (ACC, Access Control Class)
6F7B (FPLMN, Forbidden PLMNs)
6F7E (LOCI, Location Information)
6FAD (AD, Administrative Data)
6FAE (Phase)
```



mande entrante, elle répond par SW1=9Fh, SW2 précisant alors le nombre d'octets qu'il faut venir chercher avec une commande « GET RESPONSE » (A0 C0 00 00 SW2).
C'est là une particularité importante du protocole « T=0 », de ne pas supporter les commandes à la fois entrantes et sortantes, possibles en protocole « T=1 » et naturellement moins lourdes à manier.
La commande la plus utilisée est sans aucun doute « SELECT », qui sert à se déplacer dans l'arborescence des répertoires et fichiers, un peu comme la commande « CD » (*Change Directory*) de MS-DOS.
Dans le répertoire racine (repéré 3F00h), notre carte dispose des sous-répertoires suivants :

7F10h (Télécom)
7F20h (GSM)
7F21h (DCS, simple « fantôme » du répertoire GSM, utilisé par les mobiles 1800 MHz)
Pour sélectionner le répertoire « Télécom » à partir du répertoire racine, on utilisera ainsi une commande de la forme

A0 A4 00 00 02 7F 10.

Le **tableau 2** fournit la liste limitative des fichiers que contiennent les différents répertoires de notre carte.
Cette sélection, très étudiée, se compose des quelques fichiers dont l'existence est indispensable pour que la carte puisse exécuter une « session GSM » (autrement dit pour qu'elle soit reconnue comme une SIM), et de ceux permettant de se livrer aux manipulations les plus intéressantes.
Après sélection d'un fichier donné, des commandes de lecture (*read binary*, *read record*...) ou d'écriture (*update binary*, *update record*...) permettront de manipuler à volonté les données qu'il contient.
Pour une utilisation fructueuse de toutes les possibilités de la carte, il est évidemment nécessaire de connaître le rôle de ces différents fichiers.
Bien que la seule référence officielle et exhaustive soit la spécification GSM (*www.etsi.org*), et en particulier sa section GSM 11.11, voici l'essentiel de ce qu'il faut savoir.
Intéressons nous donc en priorité au répertoire GSM (ou DCS), qui offre le plus d'opportunités d'expérimentations instructives.
Les fichiers Phase et SST (*SIM Service Table*), tout d'abord, jouent un rôle important, dans la mesure où ils renseignent le mobile sur les possibilités de la carte (le symétrique, en somme, du « Terminal profile » que peuvent émettre certains mobiles).
Le fichier Phase contient un seul octet, égal à 02h en phase 2 ou 03h en phase 2+, et carrément absent dans la plupart des SIM de phase 1.
Le fichier SST, pour sa part, contient un nombre d'octets d'autant plus important que la carte SIM est perfectionnée. Chaque « service » susceptible d'être supporté par la carte est en effet matérialisé par deux bits indiquant respectivement si le service est disponible ou non, et s'il est activé ou non, selon l'affectation du **tableau 3**.
D'habitude, seul l'opérateur a le pouvoir de modifier la SST (par exemple lorsque l'on souscrit un service supplémentaire).
Dans notre cas, la SST n'est pas protégée en écriture, mais il ne faut pas y écrire n'importe quoi pour autant !
Son contenu d'origine (DF 30 C3 F3 00 00 00 03) reflétant fidèlement les possibilités

Tableau 3.
Les services de la SIM Service Table.

Octet	Actif	Existe	SERVICES	GSM
	b2	b1	Service 1:	CHV (PIN)1 disable function
1	b4	b3	Service 2:	Abbreviated Dialling Numbers (ADN)
	b6	b5	Service 3:	Fixed Dialling Numbers (FDN)
	b8	b7	Service 4:	Short Message Storage (SMS)
	b2	b1	Service 5:	Advice of Charge (AoC)
2	b4	b3	Service 6:	Capability Config. Parameters (CCP)
	b6	b5	Service 7:	PLMN Selector
	b8	b7	Service 8:	Party Subaddress
	b2	b1	Service 9:	MSISDN
3	b4	b3	Service 10:	Extension 1
	b6	b5	Service 11:	Extension 2
	b8	b7	Service 12:	SMS Parameters
	b2	b1	Service 13:	Last Number Dialled (LND)
4	b4	b3	Service 14:	Cell Broadcast Message Identifier
	b6	b5	Service 15:	Group Identifier Level 1
	b8	b7	Service 16:	Group Identifier Level 2
	b2	b1	Service 17:	Service Provider Name
5	b4	b3	Service 18:	Service Dialling Numbers (SDN)
	b6	b7	Service 19:	Extension 3
	b8	b7	Service 20:	RFU
	b2	b1	Service 21:	VCGS Group Identifier List (EF VGCS and EF VGCSS)
6	b4	b3	Service 22:	VBS Group Identifier List (EF VBS and EF VBSS)
	b6	b5	Service 23:	Enhanced Multi-Level Precedence & Pre-emption Service
	b8	b7	Service 24:	Automatic Answer for eMLPP
	b2	b1	Service 25:	Data download via SMS-CB
7	b4	b3	Service 26:	Data download via SMS-PP
	b6	b5	Service 27:	Menu selection
	b8	b7	Service 28:	Call control
	b2	b1	Service 29:	Proactive SIM
8	b4	b3	Service 30:	Cell Broadcast Message Identifier Ranges
	b6	b5	Service 31:	Barred Dialling Numbers (BDN)
	b8	b7	Service 32:	Extension 4
	b2	b1	Service 33:	De-personalization Control Keys
9	b4	b3	Service 34:	Co-operative Network List

RFU = Réservé pour de Futurs Usages

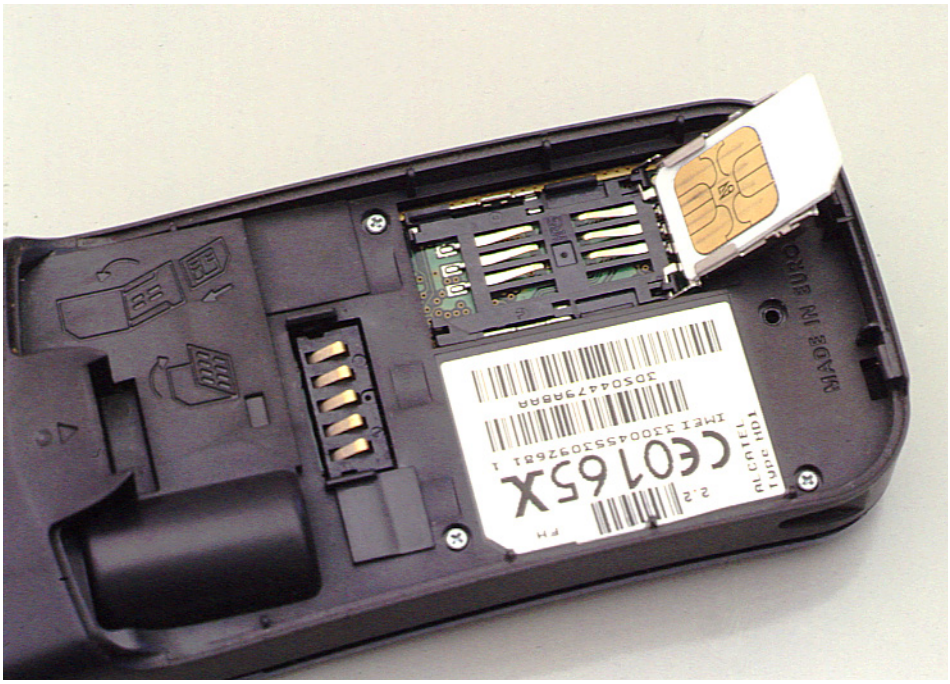
offertes par notre sélection de fichiers, il serait inopportun de tenter d'activer des services que la carte ne supporte pas.
On pourra, par contre, toujours désactiver occasionnellement tel ou tel service en mettant à zéro un ou plusieurs bits originellement à 1.
De même, on pourra faire passer temporairement la phase de 03h à 02h, et étudier les réactions qui en résultent (inhibition des fonctions « Proactive SIM », par exemple).
Si ICCID contient le numéro de série de la carte, nullement confidentiel, IMSI identifie l'utilisateur auprès de son opérateur. Son « numéro de compte », en somme...
Le code de l'opérateur est d'ailleurs incorporé dans IMSI, et c'est ce qui permet au « *roaming* » de fonctionner : lorsqu'un réseau reconnaît un IMSI étranger, il demande à l'opérateur émetteur de la SIM s'il autorise son client à utiliser ses services, et dans l'affirmative, son nom s'affiche sur l'écran.
Dans notre cas, le code réseau d'origine est 001-01, valeur correspondant à un réseau fictif et réservée aux SIM de test.
Dans le même ordre d'idées, nous avons également programmé, par défaut, le fichier AD avec une valeur propre aux cartes de test. Cela présente l'avantage que la carte ne sera pas refusée par un mobile « *simlocké* » par un opérateur ou un fournisseur de services, et pourra donc être utilisée sans restriction sur celui-ci.
Les fichiers GID1 et GID2 jouent d'ailleurs un rôle comparable, certains mobiles n'acceptant que les SIM contenant une valeur bien précise dans l'un ou l'autre de ces fichiers normalement inaccessibles en écriture.
Le contenu de LOCI se trouvera, pour sa part, mis à jour par chaque réseau sur lequel on tentera peut-être d'inscrire manuellement le mobile, tentative qui se soldera bien évidemment par un échec puisque les identifiants de la SIM sont factices.
On y retrouvera alors le code du dernier opérateur sollicité, ainsi que quelques détails concernant le lieu où la tentative a été effectuée et la raison de l'échec de l'inscription.
Simultanément, le code de chaque réseau ayant rejeté la SIM s'inscrit dans le fichier FPLMN, à concurrence d'un maximum de quatre.
Bien entendu, cette liste peut être vidée à volonté, en remettant tous les octets de FPLMN à FFh.
La plupart des fichiers du répertoire « Télécom » sont affectés aux données personnelles de l'utilisateur : répertoires de numéros de téléphone et mini-messages « SMS ».
Différentes options du menu de n'importe quel téléphone portable permettent d'intervenir sur le contenu de ces fichiers, mais il existe aussi des applications pour PC qui apportent un considérable surcroît de confort.

Les logiciels auxiliaires

N'importe quel lecteur de cartes à puce asynchrones permet, en principe, d'intervenir sur tous les fichiers de notre carte à partir du clavier d'un PC, cela par l'intermédiaire de commandes ISO 7816 de « bas niveau ».
Pour écrire 02h dans l'octet « Phase », par exemple, on pourrait enchaîner les commandes suivantes :

```
A0 A4 00 00 02 3F 00
A0 A4 00 00 01 7F 20
A0 A4 00 00 02 6F AE
A0 D6 00 00 01 02
```

Il faut pourtant bien reconnaître que les versions « Pro » des logiciels de gestion de cartes SIM facilitent infiniment les choses, grâce à leur « éditeur » incorporé.
L'un des meilleurs exemples en la matière est SIMSurf Profi, livré dans certains kits bâtis autour des lecteurs « ChipDrive » de Towitoko (*www.towitoko.de*).
Capables de fonctionner en mode PC/SC (moyennant l'installation des drivers ad-hoc), ces lecteurs très populaires en Europe sont également compatibles avec l'utilitaire *UTILPCSC.EXE* nécessaire pour exploiter le fichier « LOG » que notre carte est capable de mettre en oeuvre.
Ses versions *UTIL1.EXE* et *UTIL2.EXE*, pour leur part, sont destinées aux lecteurs « CyberMouse » (ACR20S ou ACR30S de chez ACS) fournis dans les kits BasicCard (*www.basiccard.com*).
Ces différents fichiers utilitaires évoqués dans l'encadré ainsi que le fichier *TP.IMG* à transférer sur la carte-SIM du type ZC4.1 sont disponibles au téléchargement sur le site d'Elektor (*www.elektor.fr*) et pour ceux qui n'auraient pas accès à la Toile, sur une disquette (**EPS010138-11**) disponible auprès des adresses habituelles.
Un driver spécial (disponible sur le site *http://www.acs.com.hk*) permet éventuellement l'utilisation de *UTILPCSC.EXE*, mais il est au moins aussi commode de rester en mode série « natif » !
Développé, tout comme la carte SIM, en ZCBasic version 4, ce logiciel prend en charge les quatre opérations de base relatives au fichier « LOG » : activation (*Activate*), désactivation (*Deactivate*), télédéchargement (*Download*), vidage (*Clear*).
En général, on activera le fichier



Réalisation de la carte

Bien que ce projet ait été entièrement développé au moyen d'un kit BasicCard version 4.12, il n'est aucunement nécessaire de disposer de l'intégralité de celui-ci pour réaliser et utiliser l'émulateur de carte SIM.
En pratique, il est nécessaire et suffisant de réunir les éléments suivants :
– l'utilitaire BCLOAD.EXE, inclus dans le kit de développement téléchargeable gratuitement sur www.basiccard.com ;
– un lecteur de cartes à puce compatible, qui pourra aussi bien être le « CyberMouse » fourni dans les kits BasicCard du commerce, que n'importe quel lecteur PC/SC correctement installé ;
– une BasicCard vierge compatible avec le fichier de programmation TP.IMG (en l'état actuel des choses, une ZC4.1 RSA [2001.09.28]).
Le lecteur de cartes à puce étant installé sur le PC selon les instructions de son fabricant, on ouvrira une fenêtre MS-DOS sur un répertoire de travail contenant le fichier TP.IMG et l'utilitaire BCLOAD.EXE.
On tapera alors simplement l'une ou l'autre des commandes suivantes :
– BCLOAD –D –P1 TP.IMG si on utilise un « CyberMouse » branché sur COM1:
– BCLOAD –D –P2 TP.IMG si on utilise un « CyberMouse » branché sur COM2:
– BCLOAD –D –P101 TP.IMG si on utilise un lecteur compatible PC/SC, quel que soit le port sur lequel il se trouve connecté.
Si le lecteur est correctement reconnu, il doit s'afficher une invitation à insérer la carte vierge (**écran A**).
Sauf incompatibilité entre la carte et le fichier, qui serait alors signalée par BCLOAD (**écran B**), le processus de programmation doit se traduire par le défilement, sur l'écran, d'une liste d'adresses dont on attendra évidemment la fin pour retirer la carte (**écran C**).
Il faut maintenant initialiser la carte, en exécutant simplement la fonction « Clear LOG file » de notre logiciel. Attention, cela peut prendre plusieurs secondes la première fois (**écran D**) !
Trois versions distinctes du logiciel sont fournies :
– UTIL1.EXE, destinée à un « CyberMouse » branché sur COM1:
– UTIL2.EXE, destinée à un « CyberMouse » branché sur COM2:
– UTILPCSC.EXE, destinée à un quelconque lecteur PC/SC.
Il ne restera plus qu'à découper la carte au format « SIM Micro » lorsque sera venu le moment de l'insérer dans un téléphone portable GSM.
La **figure 1** fournit les cotes permettant d'exécuter ce travail avec toute la précision voulue, si tant est que la BasicCard utilisée ne soit pas déjà prédécoupée (cela dépend des séries !). Si l'on sort alors la puce avec un minimum de soins on pourra toujours la remettre en place en dotant l'arrière de la carte, au niveau de la découpe, d'un morceau de scotch à effet temporaire. La carte pourra alors être réinsérée en toute sécurité dans le lecteur pour une nouvelle lecture des informations glanées dans le mobile.
Notons que l'on trouve facilement, chez les revendeurs de téléphones portables, des adaptateurs permettant de ramener la carte à son format primitif à chaque fois que l'on devra la réintroduire dans le lecteur du PC.

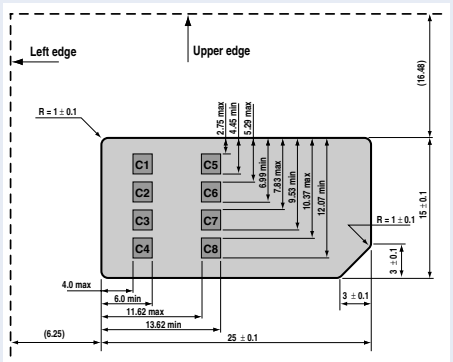
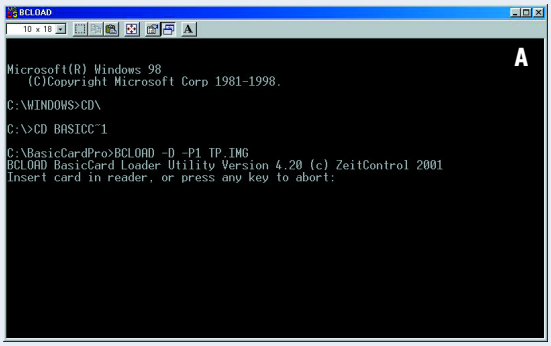
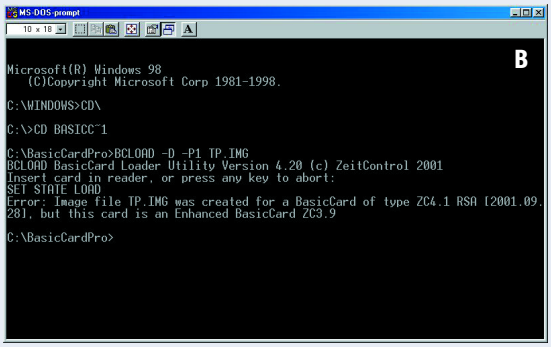
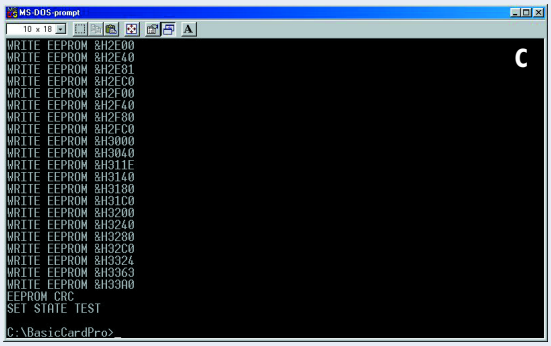


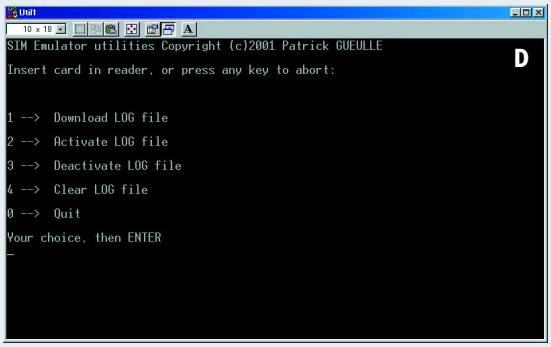
Figure 1. Cotes d'une carte SIM à mettre dans un mobile.

Notons que les programmes présents dans le répertoire BasicCardPro créé lors de l'installation de l'environnement de développement offrent nombre d'autres possibilités. Il est possible ainsi, d'obtenir des informations sur les cartes que l'on enfiche dans le lecteur (cf. **écran E**). Nous vous laissons le plaisir d'expérimenter avec cet outil à la palette de potentialités très riche.









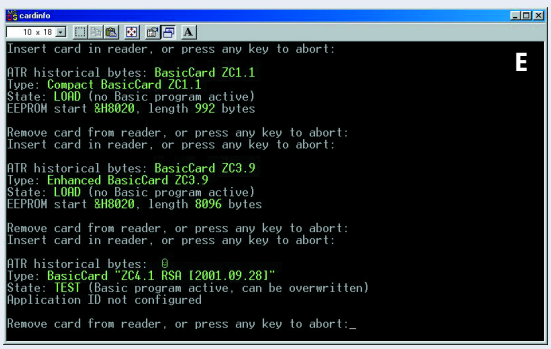


Tableau 4.
Éléments d'interprétation du « Terminal profile ».

```
Octet 1:
Bit 1 : Profile download
Bit 2 : SMS-PP data download
Bit 3 : Cell Broadcast data download
Bit 4 : Menu selection

Octet 2 :
Bit 1 : Command result
Bit 2 : Call Control by SIM

Octet 3 (Proactive SIM) :
Bit 1 : Display Text
Bit 2 : Get Inkey
Bit 3 : Get Input
Bit 4 : More Time
Bit 5 : Play Tone
Bit 6 : Poll Interval
Bit 7 : Polling Off
Bit 8 : Refresh

Octet 4 (Proactive SIM) :
Bit 1 : Select Item
Bit 2 : Send Short message
Bit 3 : Send SS
Bit 4 : Send USSD
Bit 5 : Setup Call
Bit 6 : Setup Menu
Bit 7 : Provide Local Information
```

« LOG » juste avant de placer la SIM dans un téléphone GSM, et on le désactivera aussitôt après l'en avoir retirée, afin d'écarter tout risque de saturation ultérieure.
On télédéchargera alors le fichier (qui se retrouvera sur le disque dur du PC, sous le nom *CARD.LOG*), puis on le videra afin de libérer de la place pour de nouvelles manipulations.
CARD.LOG est un fichier ASCII, lisible à l'aide de n'importe quel éditeur de texte, et contenant des valeurs hexadécimales.
Une ligne distincte est affectée à chaque commande reçue par la carte, tandis que toute ligne vide indique un RESET de la carte à cet instant précis (arrêt et remise en route du téléphone, par exemple).
Voici un court extrait enregistré lors des premiers instants de fonctionnement d'un mobile compatible Phase 2+, à la fin duquel on ne manquera pas de remarquer la commande « Terminal profile » par laquelle le téléphone informe la carte SIM de l'étendue de ses propres possibilités

```
A0 A4 00 00 02 7F 20
A0 C0 00 00 16
A0 A4 00 00 02 6F AE
A0 C0 00 00 0F
A0 B0 00 00 01
A0 A4 00 00 02 6F 05
A0 C0 00 00 0F
A0 B0 00 00 03
A0 A4 00 00 02 6F 05
A0 C0 00 00 0F
A0 A4 00 00 02 7F 20
A0 C0 00 00 16
A0 A4 00 00 02 6F AE
A0 C0 00 00 0F
A0 B0 00 00 01
A0 10 00 00 04 0F 03 FF F7
```

C'est d'ailleurs ce mot hexadécimal 0F 03 FF F7 qui s'affichera sur l'écran du mobile lorsqu'il aura terminé sa procédure d'initialisation (ces quelques longs instants pendant lesquels on obtient souvent le message « Essayez plus tard » si l'on tente d'appeler une fonction un tant soit peu gourmande en ressources machine).
Son décodage (**tableau 4**) d'après la spécification GSM 11.14 indique que ce mobile supporte la totalité des fonctionnalités SIM Toolkit actuelles, à l'exception de la fonction « Send USSD ».

Cela permet de dépister, par exemple, l'incompatibilité de cet appareil avec diverses fonctions avancées (envoi de SMS, d'E-mails, etc.) d'une étonnante carte SIM prépayée internationale, la « GSM card easyRoam » de Swisscom (*http://www.easy-roam.com*), ce que ne laisserait aucunement soupçonner l'étude de la notice du constructeur !
Mais nous arrivons là au stade du cas particulier : chaque modèle de mobile donnera des résultats différents, d'ailleurs largement variables en fonction des changements que l'on aura jugé utile d'apporter au contenu par défaut des fichiers de la carte SIM.
Bien entendu, l'auteur ne peut prétendre avoir « fait le tour » de tous les types de téléphones portables en circulation, bien qu'il ait expérimenté sa carte SIM avec des modèles aussi bien récents qu'anciens de Motorola, Nokia, Alcatel, Panasonic, Sagem, etc.
Il n'a pas davantage épuisé toutes les découvertes que permet potentiellement un tel outil, et invite cordialement ses lecteurs à pousser toujours plus loin l'aventure !

(010138)

NdlR : le seul petit problème que nous ayons rencontré pour le moment avec ce projet est le coût relativement élevé de ces cartes, près de 10 $US par carte (ce qui se justifie par sa complexité) avec un minimum, pour l'instant, de 10 cartes (l'impact des frais de port se justifie difficilement pour un nombre plus faible).