La carte à puce

La carte à puce

La carte à puce : introduction et principe

La carte à puce aujourd'hui

- Aujourd'hui : plus de 6 milliards de cartes
- **Monétique** :
  - Carte bancaire : Groupement Cartes Bancaires, nouvelles cartes EMV, etc.
  - Porte-monnaie : **Octopus**, **Moneo** en France, **Proton** en Belgique, **Geldkarte** en Allemagne
- **Identification** :

Cartes d'identité nationales (**eID** en Belgique), **E-passeports** (août 2006 en France), Passeport biométrique (depuis le fin Juin 2009)

- **Enseignement** (comme carte d'étudiant et/ou de restauration)

- **Téléphonie mobile** (carte **SIM**)

- **Secteur médical** (carte **Vitale** en France, carte **SIS** en Belgique).

- **Titre de transport** (**Passe Navigo** à Paris, **Oyster** à Londres, **Korrigo** (un seul titre de transport pour tous ses déplacements en transports en commun).

- **Sécurité informatique** (authentification forte et signature électronique): carte doté d'un cryptoprocesseur pour la génération des clés et le stockage de la clé privée).

Exemples des Passeports

Depuis 2006 en France :
Passeport électronique comporte une puce électronique qui stocke les données personnelles du détenteur : (son nom, sa date de naissance, sa nationalité, son numéro de passeport et la photo numérisée du titulaire).

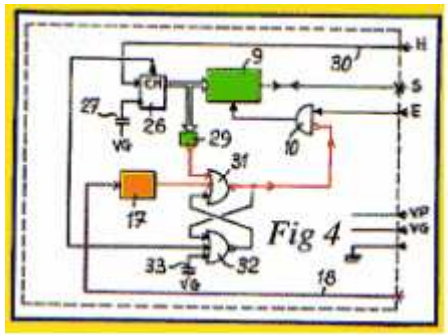
Depuis le 15 Juin 2009 :
Passeport biométrique : sur une puce RFID, qui permet de lire les informations à courte distance, sont enregistrés - outre les informations personnelles classiques et la photo numérisée - deux empreintes digitalisées des doigts du détenteur (à partir de l'âge de 6 ans).

Historique – Invention de la carte à puce

- **1967-1968: Jürgen Dethloff et Helmut Grötrupp ingénieurs allemands (déposent un brevet en 1969)**
- **1970: Kunitaka Arimura japonais dépose un brevet en mars 1970 au Japon**
- **1971: Paul Castrucci de IBM dépose aux USA un brevet intitulé *Information Card***
- **1974-1979 : Roland Moréno français dépose 47 brevets dans 11 pays (crée ensuite la société Innovatron)**
- **Implication industrielle de Bull et Schlumberger**

Première carte à puce

- **1979: 1ére carte à base de microcontrôleur**
  - ✓ Fabriquée par Motorola pour Bull CP8
  - ✓Possède une UC de type 6805 (micro-contrôleur 8 bits de Motorola)
  - ✓Avec une PROM de 1 Ko

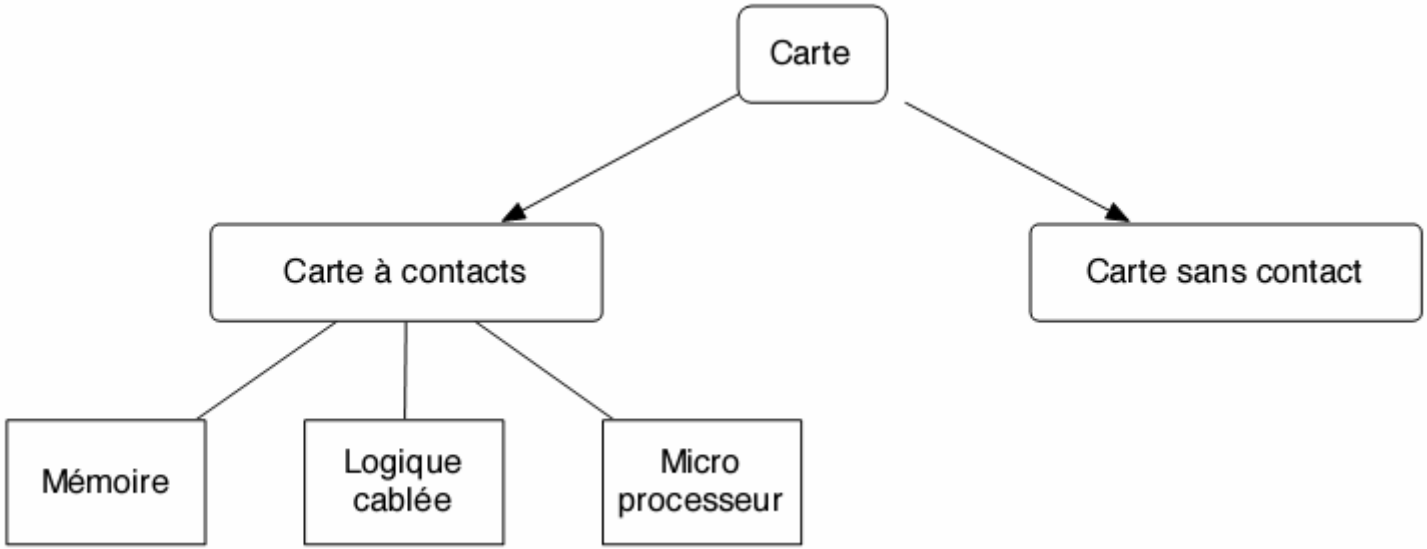


L'arrivée de la technologie Java Card

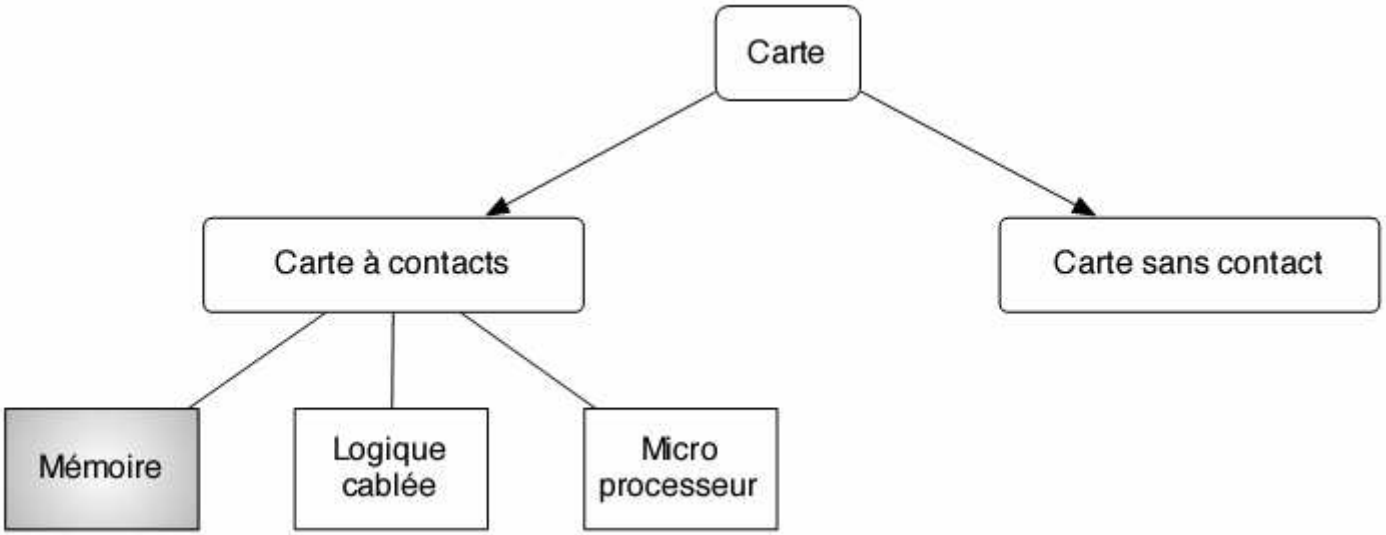
Année	Événement
1979	Première carte fabriquée par Motorola pour Bull CP8
1980-81	Premières expérimentations de télévision payante
1983	Premières cartes téléphoniques France Télecom
1984	Première version de la carte bleue à base de carte Bull CP8
1987	Publication des normes ISO 7816
1989	Premières cartes GSM pour téléphones mobiles (Gemplus)
1998	Premières cartes Java (Java Card)

Famille de produits

- **carte à mémoire**
- **carte logique câblée**
- **carte à microprocesseur**



La carte à mémoire



La carte à mémoire simple

- **1ère génération : fonction logique de stockage de données, pas de CPU.**

- **notion de zone:**
  - **zone à valeur fixe, non modifiable [identifiant émetteur],**
  - **zone pour l'écriture, modifiable [compteur d'unité],**
  - **notion de fusible, cycle de vie de la carte [par exemple la zone à valeur fixe devient non modifiable après que le fusible soit grillé : OTP (One Time Programing)].**

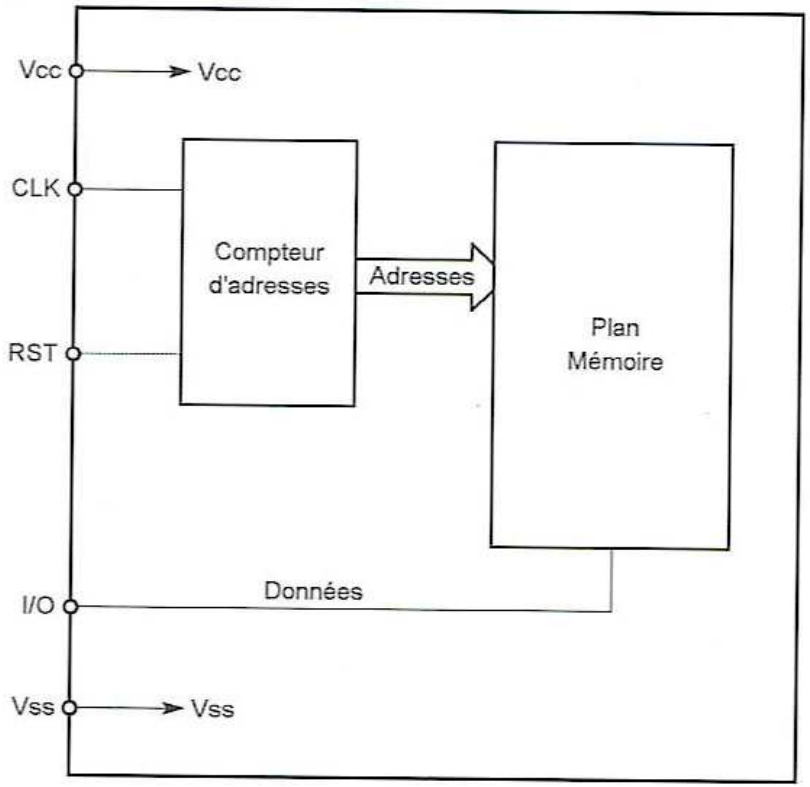
Exemple la télécarte

(mémoire de 256 bits = ZP (96) + ZL (150))

Av. : coût très bas. Inc. : "clonage" simple.

Cartes à mémoire simple

Exemple : carte téléphonique française 1ére génération



Les cartes à mémoire OTPROM

- **Cartes OTPROM (One Time PROM)**

  - **n'existent plus aujourd'hui**

  - **exemple : télécarte de France Télécom**

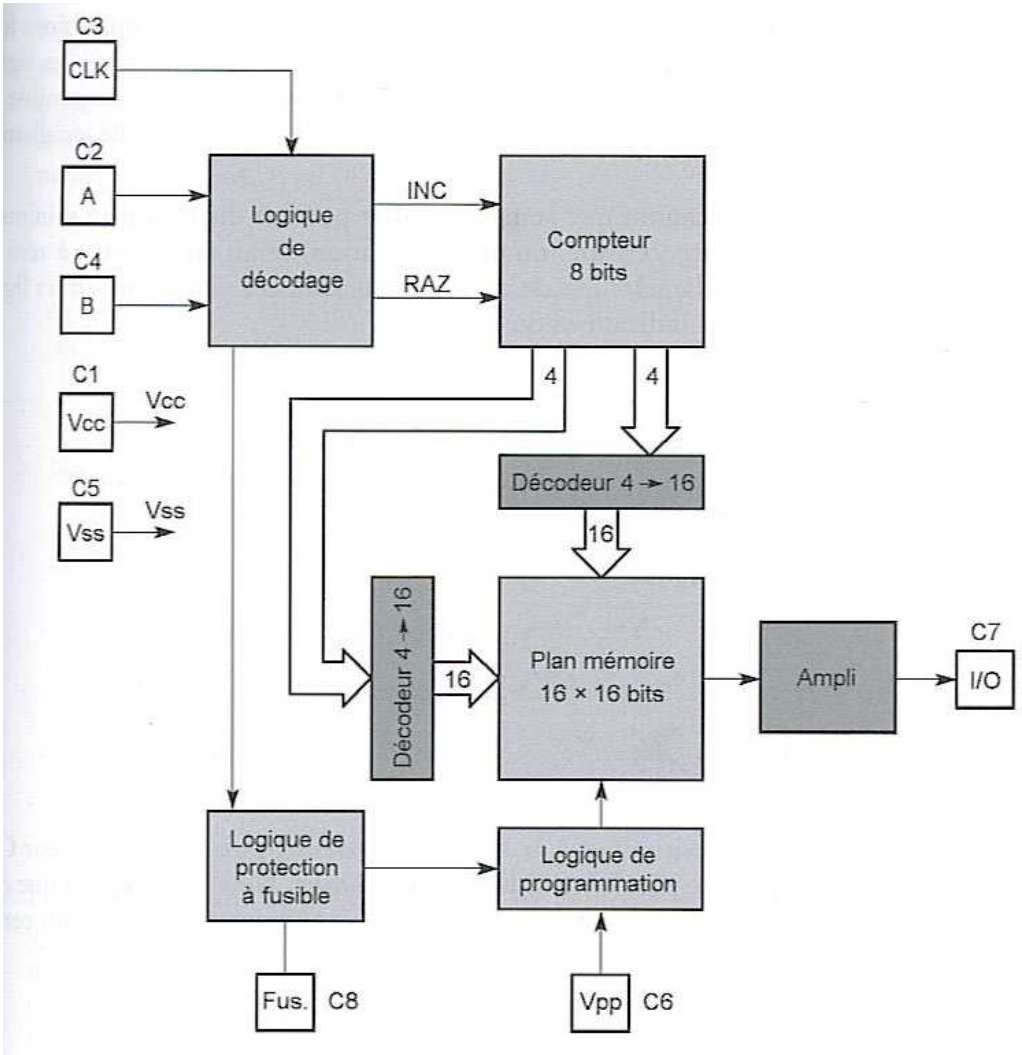
  - **technologie NMOS : haute tension (21 V) pour leur programmation (Vpp)**

- **Circuit intégré ST 1200 (de SGS Thomson) : programmable une seule fois (mise à 1)**

- **Utilisation parfaite pour les télécartes (ne sont pas rechargeables)**

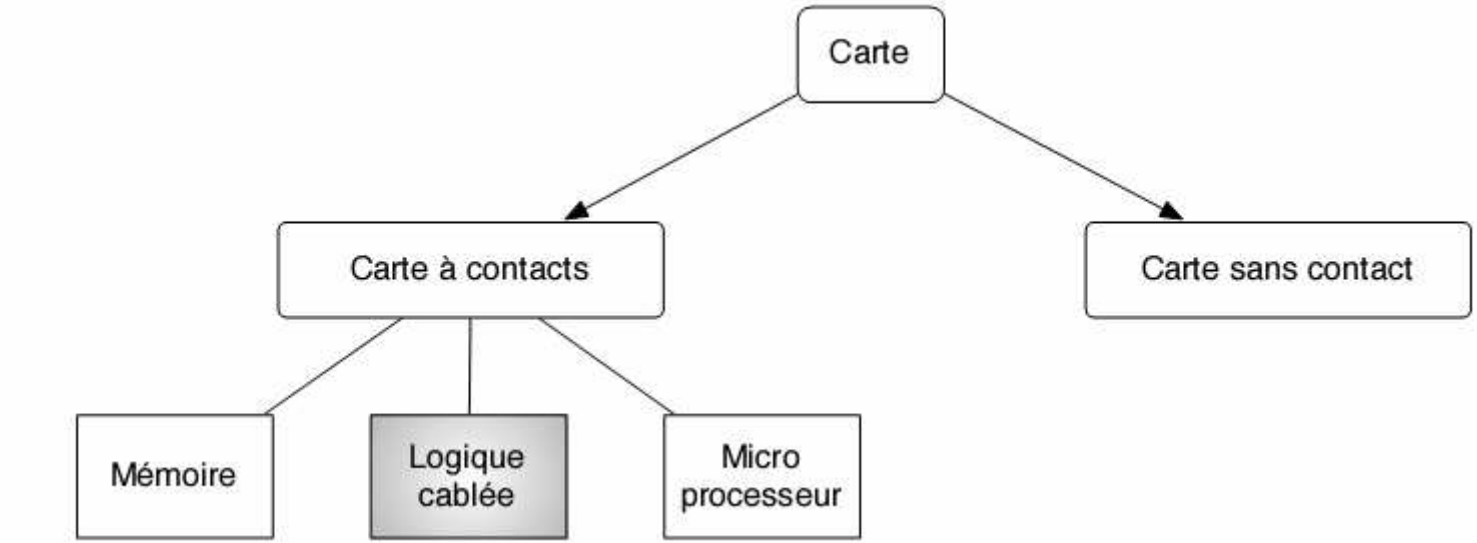
- **La sécurité repose sur la programmation une seule fois**

Les cartes à mémoire OTPROM



Synoptique interne d'une carte à mémoire OTPROM à base de ST 1200

La carte à logique câblée



La carte à logique câblée

- **Fonction : la carte comporte de la mémoire et des règles d'utilisation de celle-ci (certaines zones accessibles seulement en lecture)**

- **La logique des règles est implantée de manière physique dans le silicium du composant de la carte.**

Av. : simple à définir,
Inc. :
- Pas de souplesse, d'évolutivité de la carte : une caractéristique correspond à un ensemble de porte logiques,

- **Conduit à la carte à microprocesseur.**

- **Exemple : Produit ATMEL référencé AT88SC1608**

Caractéristiques des cartes à logique câblée

- **Plusieurs produits "génériques" :**
  - **Eurochip II,**
  - **T2G (Protocole allemand).**

- **Caractéristiques (cas GPM 375, T2G) :**
  - **Le passage de l'EPROM à l'EEPROM a permis l'introduction de compteur (ici 32 767 unités).**

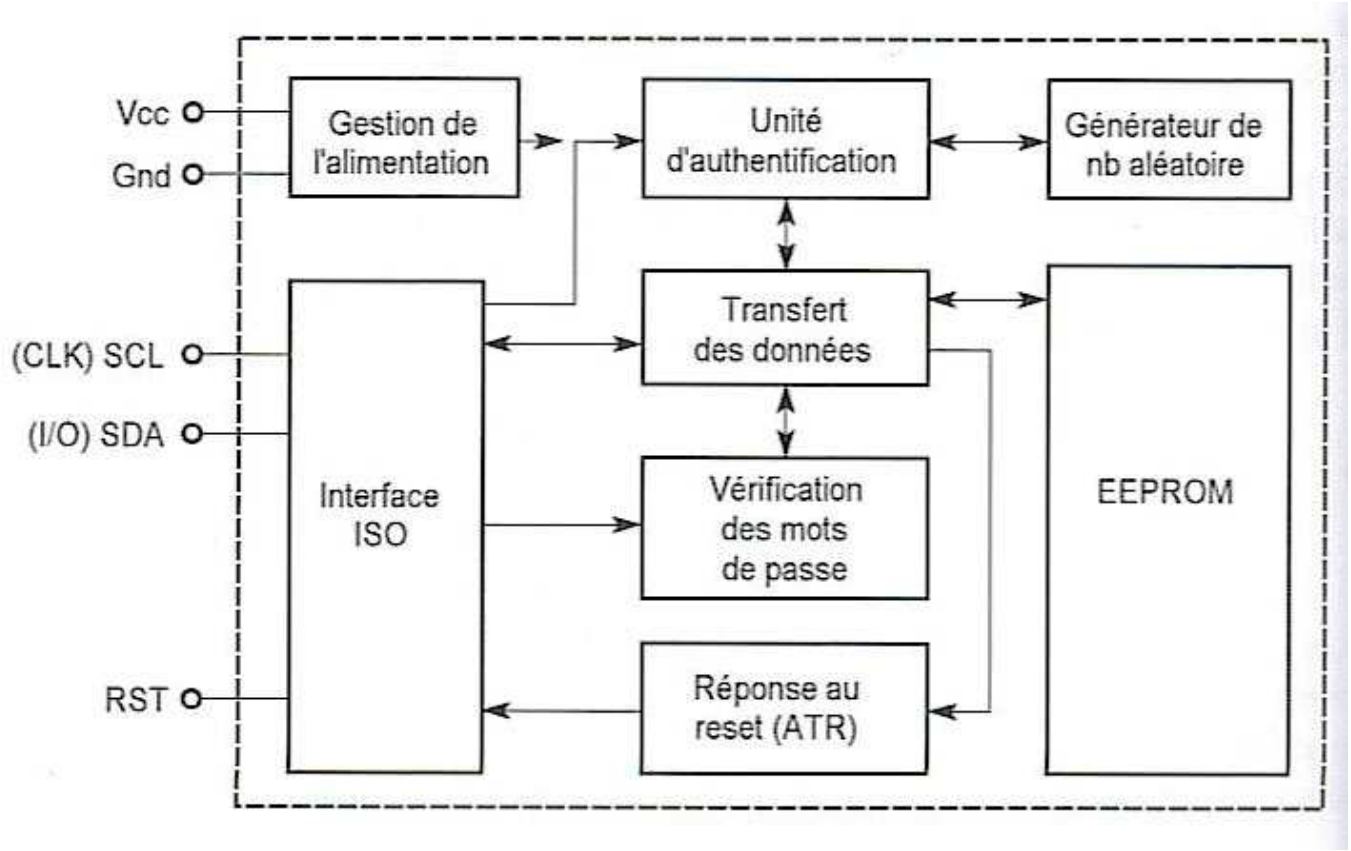
  - **Algorithme dynamique d'authentification (vérifié par un SAM = Security Access Module), procédure CBC (Cipher Block Chaining) pour lier authentification et signature de transaction.**

  - **Tension 5v, Protocole synchrone (5 contacts).**

Organisation de la mémoire

Adresse	Contenu	Rôle
0-15	Card manufacturer area	Identifies chip & card manufacturer
16-23	Issuer reference	Identifies issuer or application
24-63	Card identification area	Contains card serial number, secret key index, initial face value...
64-103	Abacus counter area	Holds balance of the card (units)
128-191	Authentication key	Contains card secret key
288-319	Pull-out flags	Enables to restore final counter value if transaction interrupted
320-375	56-bit user defined area	Erasable

Exemple d'une carte à logique câblée sécurisée



Sécurisation de la mémoire et des transactions

- **Carte mémoire = X zones mémoires (ex. X=8 dans le cas de AT88SC1608)**
  - zones à lecture seule ou à écriture (si la carte autorise l'application)

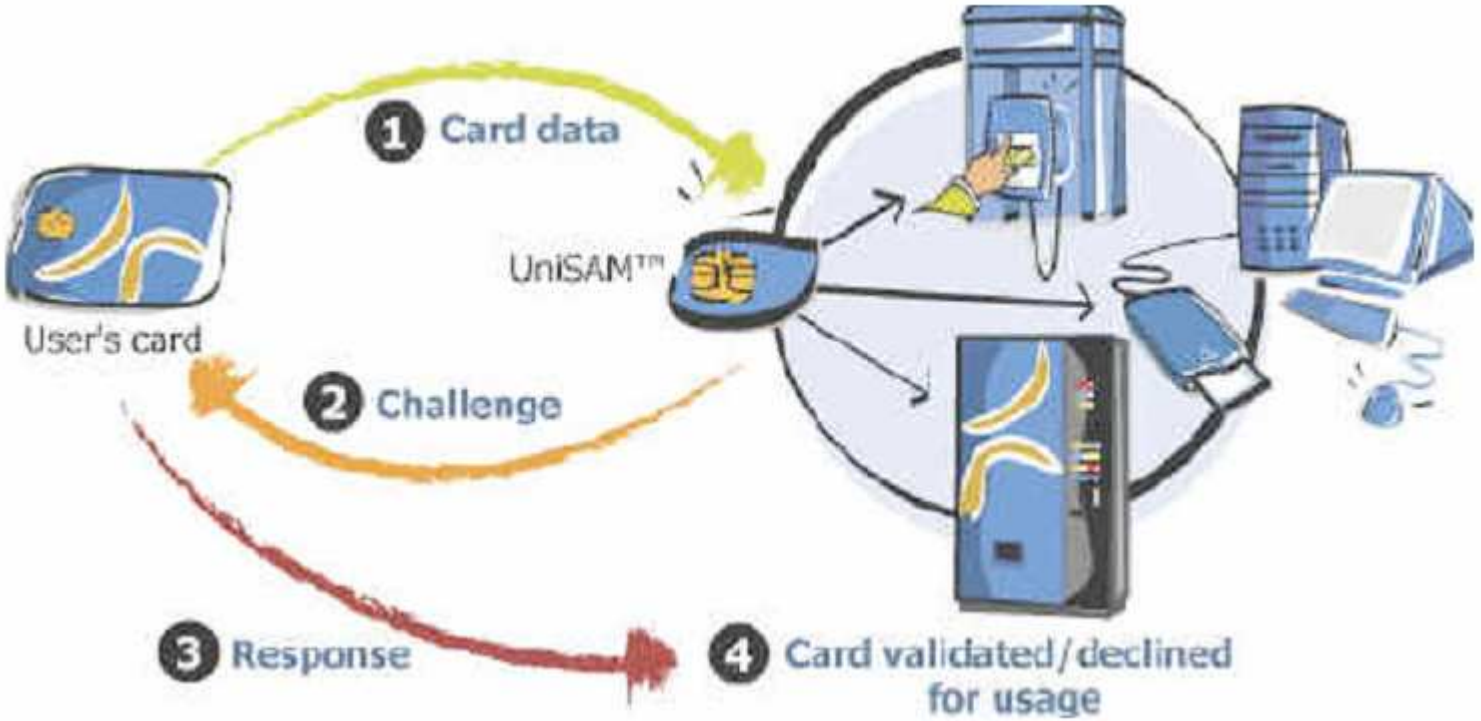
- **1er niveau de sécurisation**

  -Au niveau de chaque zone, on peut définir un mot de passe pour la la lecture et un mot de passe pour l'écriture

- **2ème niveau de sécurisation :**
  - authentification de la carte par rapport au lecteur
  - identification du lecteur par rapport à la carte
  => c'est l'authentification mutuelle utilisée dans les vraies cartes à puce

Télécarte de deuxième génération

- **principe : d'après http://gemalto.com**



Principe de l'authentification mutuelle

- Lorsque la carte est insérée dans le lecteur, le lecteur envoie une commande d'authentification

- La carte envoie au lecteur un numéro unique contenu dans un des registres

- Le lecteur traite ce nombre avec un algorithme cryptographique et retourne le résultat obtenu à la carte

- En même temps, la carte fait le même calcul

- La carte compare son résultat avec celui du lecteur, s'il y a égalité => le lecteur est identifié par la carte
- La carte utilise ce résultat pour calculer une nouvelle valeur à l'aide d'un algo cryptographique
- La carte envoie le nouveau résultat

- Le lecteur procède de la même manière, si égalité => il a identifié la carte

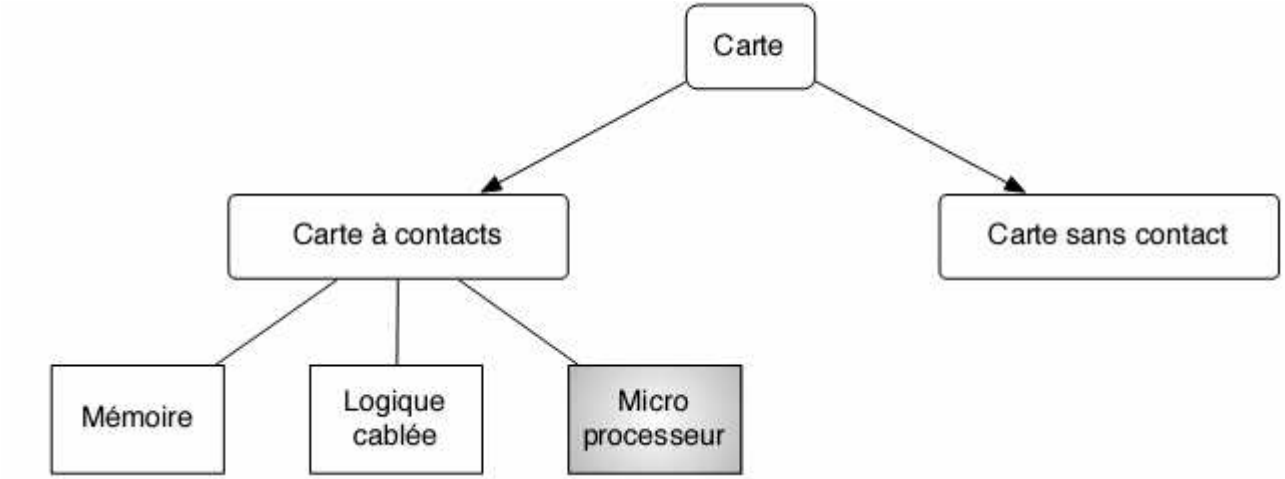
Développement d'une application

- **Carte mémoire = X zones mémoires**

- **Développer une application = > programmer un certain nombre de registres internes de la carte**

- **Chaque zone :**
  - **un octet : conditions d'accès à la zone (avec/sans mot de passe de Lect/ Ecrit/, avec/sans authentification)**
  - **données initiales pour l'authentification mutuelle (clé + semence du générateur de nbs aléatoires)**
  - **mots de passe en Lect/Ecrit**

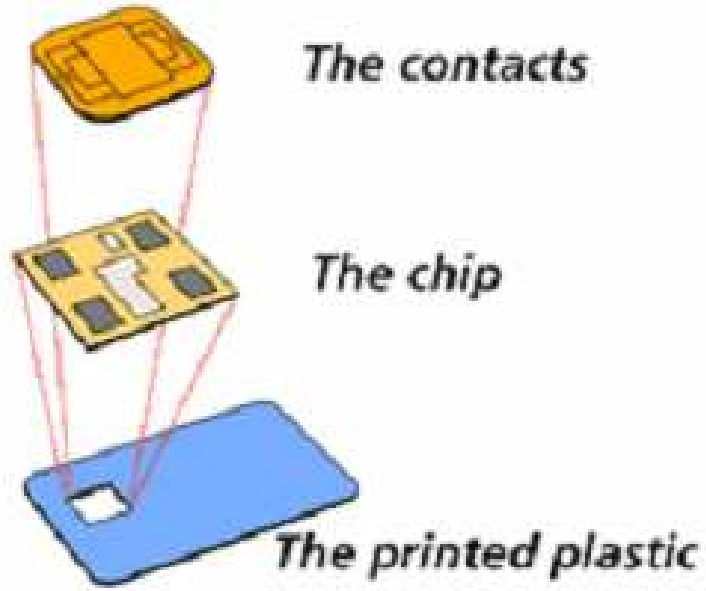
La carte à microprocesseur



1ère implémentation de la “smart card” (CP8)
RAM : 36 octets
EPROM : 1ko
ROM : 1,6 ko

Étapes de fabrication de la carte

- **Fabrication du support ou corps de la carte**
- **Plastique laminé**
- **Fabrication du composant (galette de silicium ou « wafer »)**
- **Fabrication du module :**
  - **Découpage/Sciage,**
  - **Contact/Binding,**
  - **Protection dans le module,**
  - **Collage.**



Cartes à microcontrôleur

- **Cartes à puce intelligentes**

- **Comportent un microcontrôleur :**

  - **UC**
  - **PROM**
  - **RAM**
  - **EEPROM**
  - **interface d'E/S**
  - **crypto processeur**

  } **dans le même circuit**

- **Processeur : 16 ou 32 bits**

- **EEPROM : de 1Ko à 128 Ko (256 Ko pour une Java Card ou une Basic Card)**

- **Interface série: UART simplifié**

Architecture logicielle

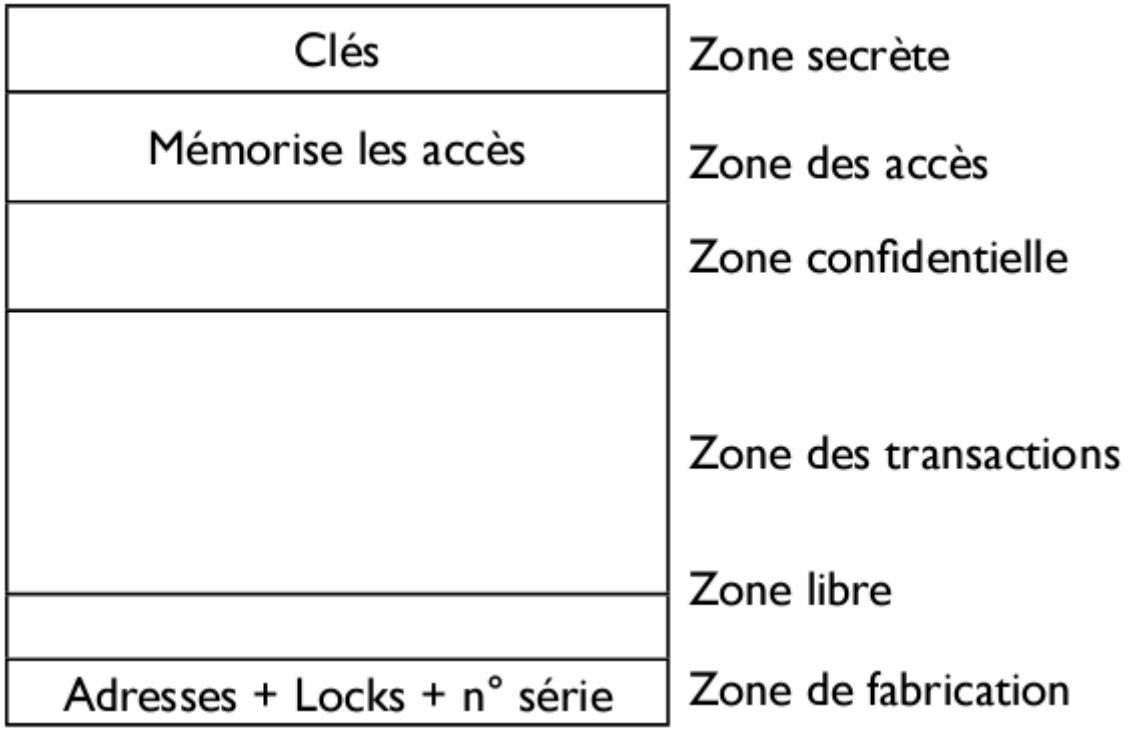
- **Gestion protocole d'E/S**
- **Gestion de la mémoire**
- **Fonction cryptographique**

Particularité du « système d'exploitation (SE)»

- **SE et application sont imbriqués (de manière inextricable) :**
  - **Pas de notion de SE.**
  - **Modèle absent ou rudimentaire :**
  - **Une structure de données = plan mémoire avec des zones et valeurs comme indicateurs de statut ou avancement dans le cycle de vie de la carte. (On parle de "mapping" carte).**

- **Une suite de fonctions = activables ou pas en fonction des valeurs de verrous (locks) qui agissent plus ou moins directement sur la mémoire.**

Exemple de « Mapping »

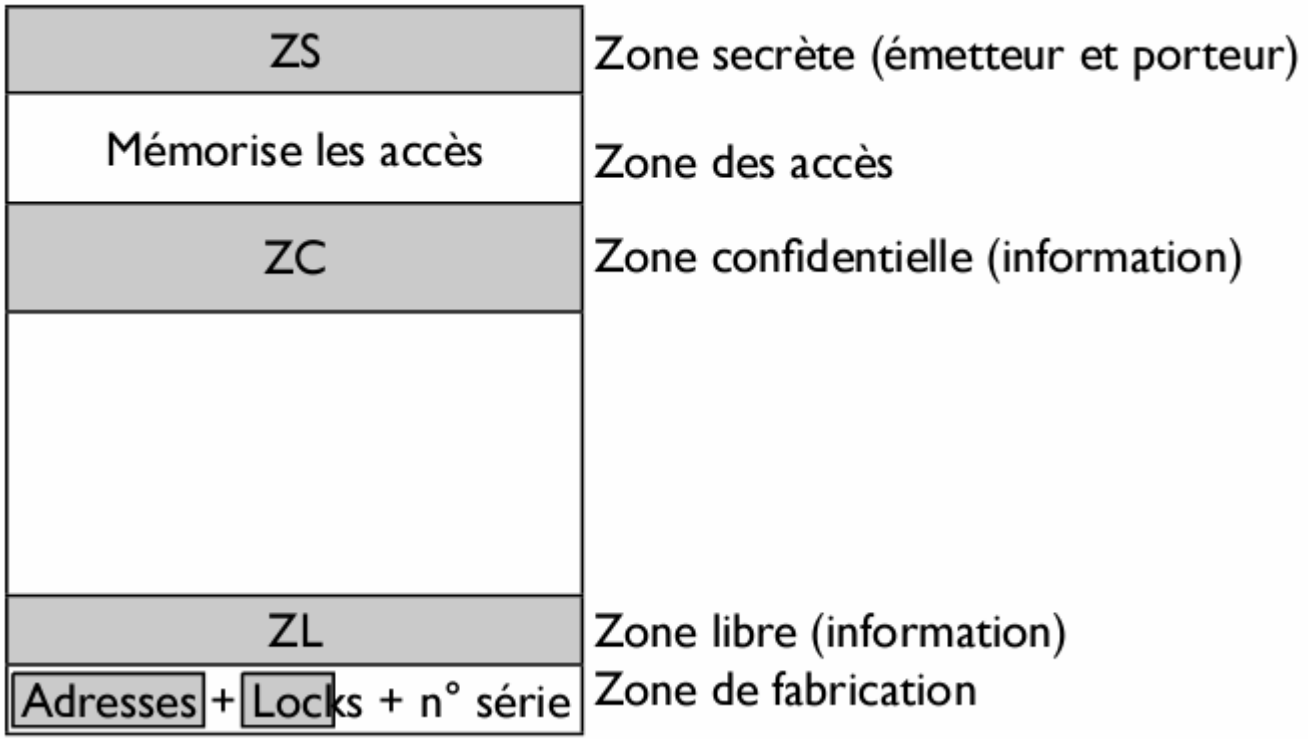


Fonctionnement (1)

A la pré-personnalisation (à la sortie de l'usine) : clé de fabrication

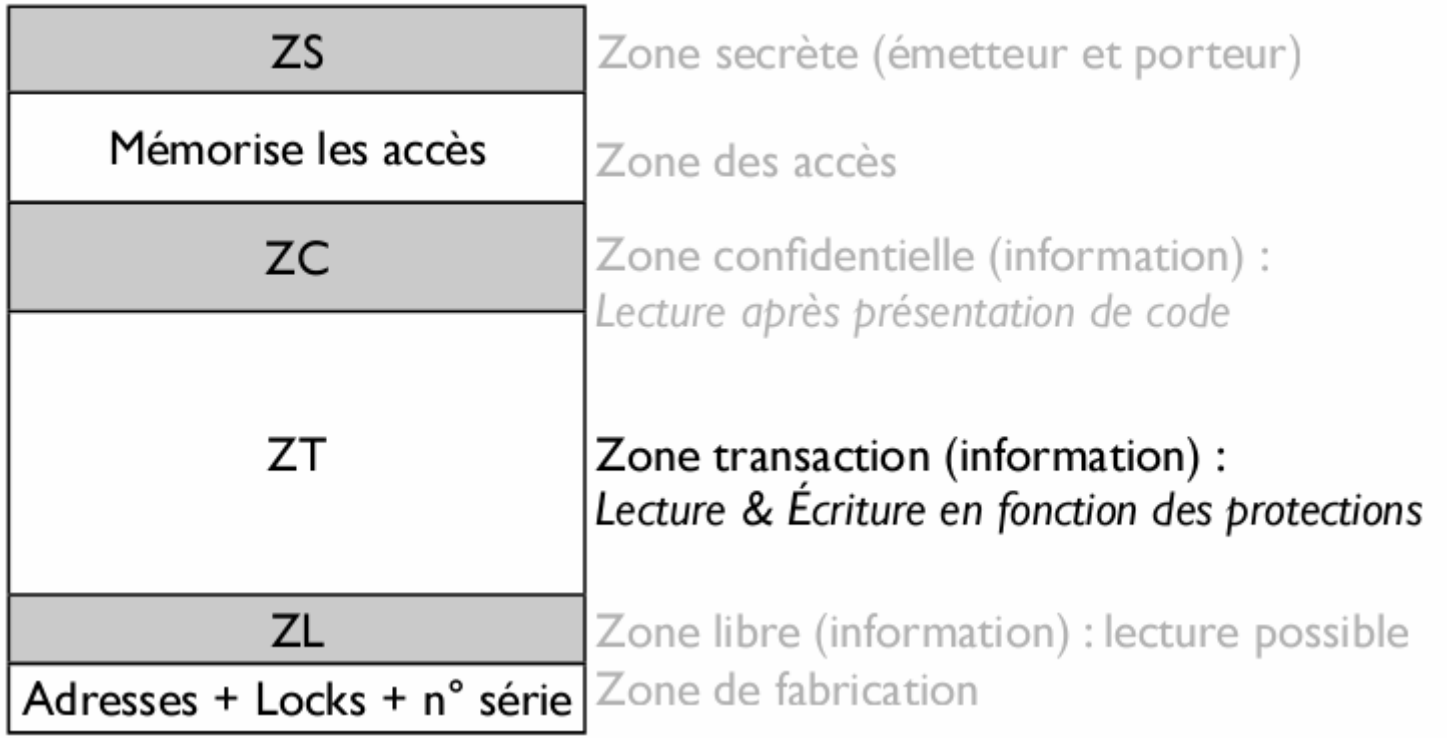
Fonctionnement (2)

A la personnalisation

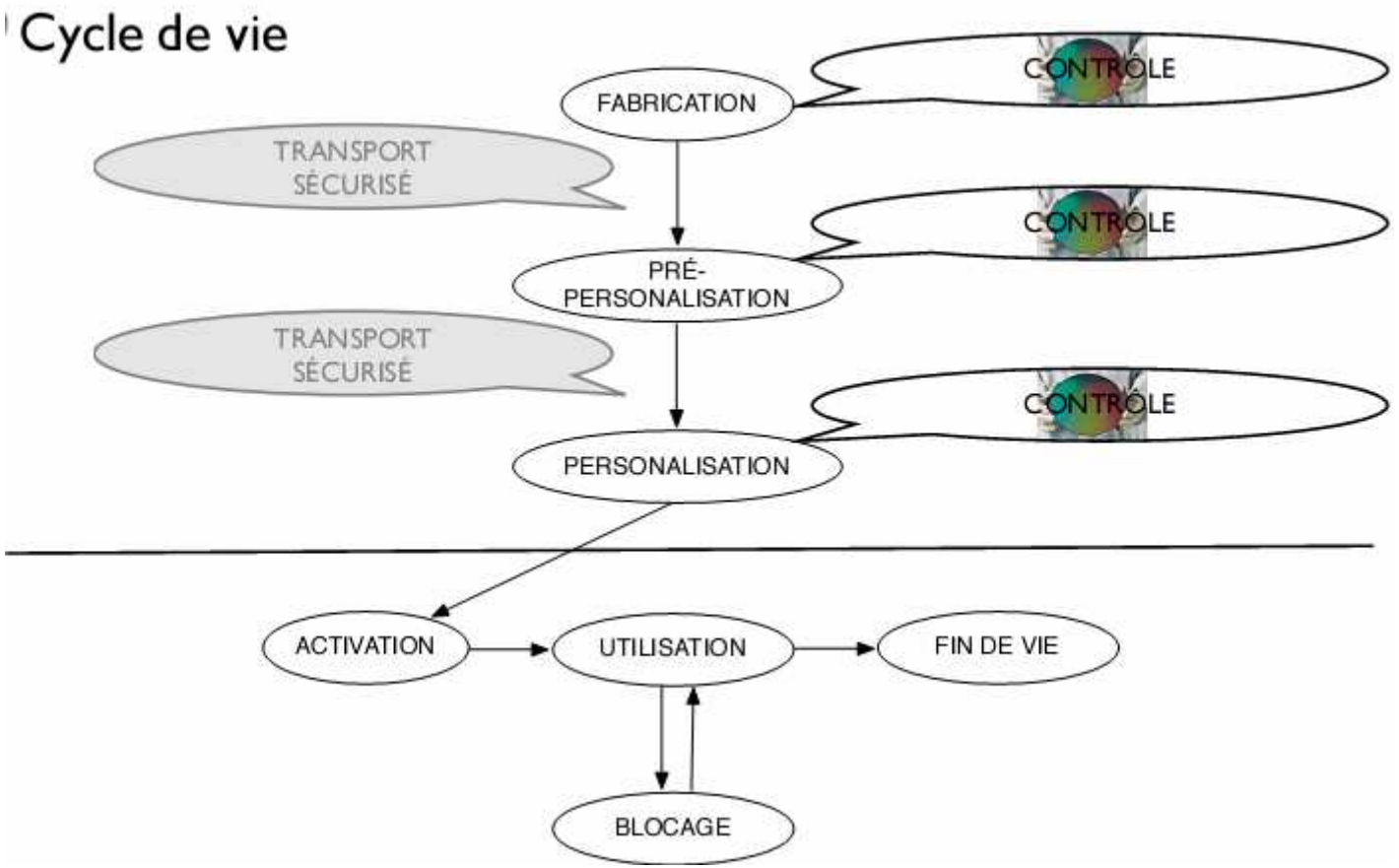


Fonctionnement (3)

A l'utilisation



Fonctionnement (4)



Les grandes familles de SE

- **Les prémisses (avant 1990):**
  - **M4, BO, COS,...**
    - **Mono-application**
    - **Plus ou moins figé dans les fonctions et structures**
    - **COS : possibilité d'ajouter des fonctions**

- **Les évolutions (1990-1995) :**
  - **MP, MP100, MCOS**
  - **Multi-application**
  - **CQL (1993), Basic Card,....**

- **En avant vers l'ouverture...**
  - **Java Card (depuis 1996)**
  - **.Net**

Architecture matérielle aujourd'hui

- **CPU : 8, 16 & 32 bits,**
  - **Coeur 8051, AVR, ARM, MIPS, propriétaire**

- **Mémoires :**
  - **RAM : 1 à 4 Ko**
  - **NVM (EEPROM/Flash) : 16 à 32 Ko**
  - **ROM : 32 à 64Ko**

- **Co-processeur**
  - **Java Card : exécution directe du Byte Code Java Card**

La normalisation

Les standards

Les normes liées à la carte :

- **ISO,**
- **ETSI, (télécommunications, GSM)**
- **EMV, (cartes de paiement)**
- **ICAO, (agence de l'ONU, biométrie, passeport)**
- **Santé,**

...

Normalisation parfaite

- **Quel que soit le fabricant de la carte à puce, celle-ci sera lue par n'importe quel distributeur dans le monde**

- **Pour garantir cette interopérabilité, la normalisation concerne au moins 3 points:**

-Des paramètres physiques : taille de la carte, position de la puce et ses contacts

-Des paramètres électriques : tension d'alimentation, niveaux électriques utilisés

-Des paramètres logiciels qui définissent le mode de dialogue avec la carte (commandes)

Normes principales des cartes à contact : l'ISO 7816

- **L'ISO 7816 « Idenfication cards – Integrated circuit cards with contacts »**
  - ✓publié par l'ISO (International Organisation for Standardisation)
  - ✓le plus important standard définissant les caractéristiques des cartes à puce qui fonctionnent avec un contact électrique
  - ✓15 normes sont proposées pour les cartes à contact.

Normes principales des cartes à contact

- **La norme ISO 7816-1 précise les caractéristiques physiques de la carte**

- **La norme ISO 7816-2 définit la position et le brochage des contacts de la carte**

- **La norme ISO 7816-3 définit les niveaux électriques utilisés pour le dialogue avec la carte**

- **La norme ISO 7816-4 définit les commandes de base des cartes à puce**

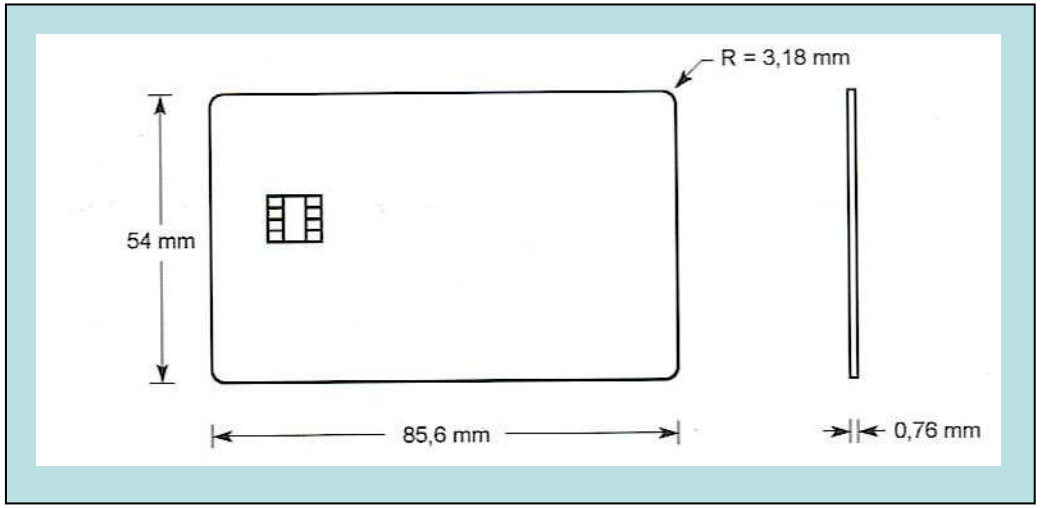
La norme ISO 7816-1

➢**ISO 7816-1 : révisé en mars 1998**
définit les caractéristiques physiques des cartes à puce à contact, ex : la géométrie, la résistance, les contacts, etc.

Caractéristiques mécaniques des cartes à puce

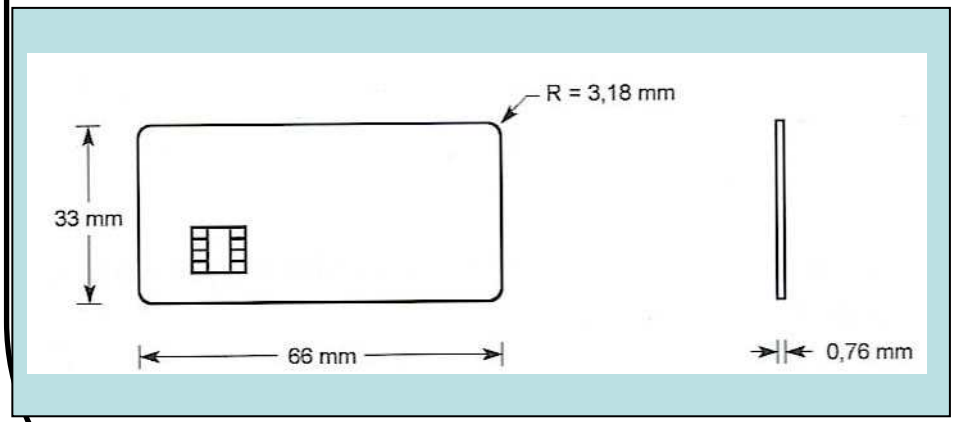
- **Même si on connaît en général deux formats de la carte à puce**
  - ✓**Celui de la carte bancaire**
  - ✓**Celui de la carte SIM**
- **3 formats normalisés : ID1, ID00 et ID000**

Les 3 formats

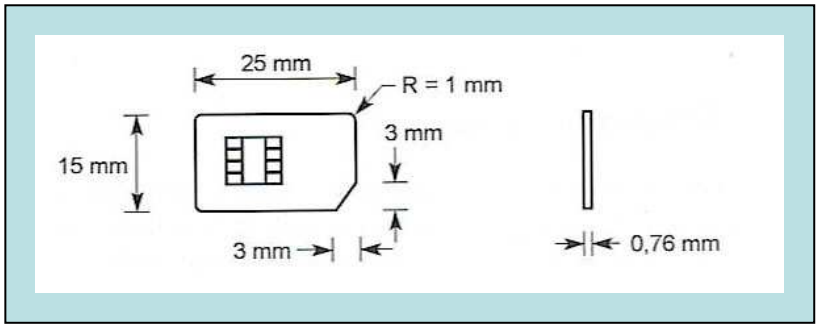


ID 01

ID 00

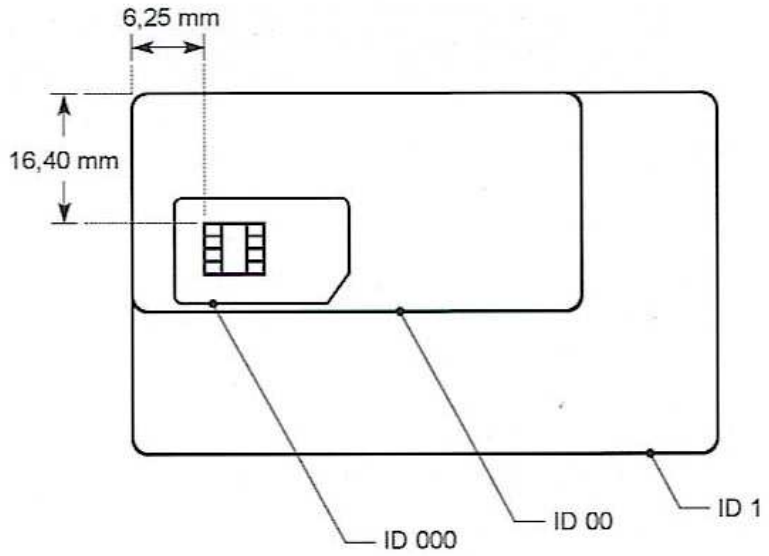


ID 000



Les 3 formats

➢**Le fabricant produit une seule taille (ID1), le client final pourra réduire ses dimensions au format ID00 ou ID000 (ex. carte SIM)**



Caractéristiques physiques

- La carte doit être opaque aux rayons UV (la puce insensible aux rayons UV)
- La carte doit résister aux détériorations de sa surface
- la carte doit protéger la puce lors de manipulation de stockage lors d'une utilisation normale
- La zone des contacts doit résister à la pression causée par une bille d'acier de 1,55 mm de diamètre appliquée avec une force intérieure ≤ 1,5 N.
- La puce doit résister aux rayons X
- La carte ne doit pas être endommagée par un champ magnétique statique de 79 500 A/m.

etc.

Normalisation AFNOR / ISO

- **La position des contacts : position AFNOR et position ISO**

Carte ISO

Carte AFNOR

L'ISO 7816-2

- **Elle spécifie le dimensionnement physique (extérieur) des contacts de la puce. 2 des 8 contacts réservés à une utilisation future (RFU) sont redéfinis pour l'utilisation USB dans la norme ISO 7816-12.**

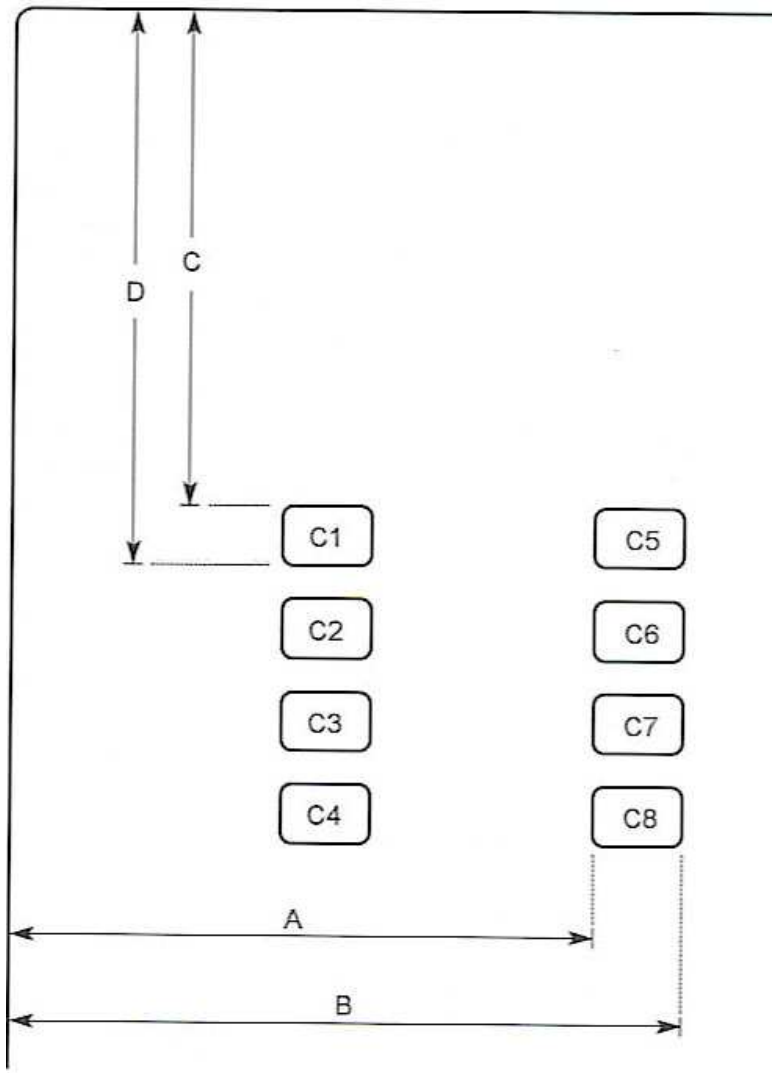
- **Dimension et emplacement des contacts, révisés en mars 1998.**

L'ISO 7816-2

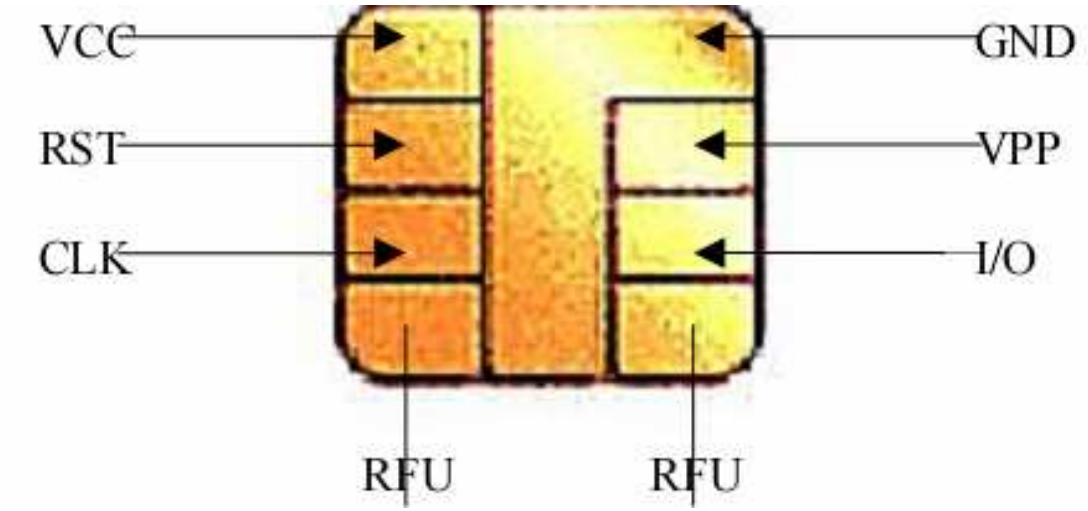
	A	B	C	D
C1	10,25	12,25	19,23	20,93
C2	10,25	12,25	21,77	23,47
C3	10,25	12,25	24,31	26,01
C4	10,25	12,25	26,85	28,55
C5	17,87	19,87	19,23	20,93
C6	17,87	19,87	21,77	23,47
C7	17,87	19,87	24,31	26,01
C8	17,87	19,87	28,85	28,55

Position ISO 7816

Valeurs en mm



L'ISO 7816-2



Vcc: tension électrique (3 à 5 V)
RST: c'est le « reset », initialise le microprocesseur (warm reset)
cold reset = coupure et rétablissement de l'alimentation
CLK: signal d'horloge car pas d'horloge sur la carte
GND: masse
Vpp: utilisé dans les anciens modèles pour avoir une autre source d'alimentation
I/O: utilisé pour le transfert des données et des commandes entre la carte et le lecteur. La communication half-duplex.

Signification des contacts

- **Vcc : tension d'alimentation positive de la carte fournie par le lecteur**
  (4.75V ≤ Vcc ≤5.25V) Vcc=3.3V pour une carte SIM
- **RST: commande de reset de la carte, fournie par le lecteur**
  (entrée non obligatoire avec certaines cartes à mémoire)

- **CLK: Clock, horloge fournie à la carte par le lecteur**
  - rythme les échanges de données entre la carte et le lecteur

- **RFU (Reserved for Future Use) non utilisés**

- **GND masse électrique de la carte**

- **Vpp: tension de programmation de la carte fournie par le lecteur**
  -inutilisé aujourd'hui
  -21V nécessaire dans les premières cartes pour écrire dans des EPROM

- **I/O entrées/sorties des données**
  - ligne bidirectionnelle (carte => lecteur et lecteur => carte)

Caractéristiques électriques

- **Vcc : (4.75V ≤ Vcc ≤ 5.25V) Vcc=3.3V pour une carte SIM**

- **RST: valeur min = 4V ou Vcc-0.7V**

- **CLK: Min = 2.4V ou 0.7Vcc ou encore Vcc-0.7V**
  Max=Vcc

- **I/O : état haut (Z) : en mode réception de la carte**
  : état bas (A) : imposé par le lecteur
  - en fonctionnement normal, les 2 extrémités de la liaison ne doivent jamais être en mode émission simultanément

I: Min= 2V ou 0.7Vcc
Max=Vcc
O: Min=2.4V ou 3.8V
Max=Vcc

L'ISO 7816-3

- **Elle définit l'interface électrique et les protocoles de transmission :**

  - ✓**Les protocoles de transmission (TPDU, Transmission Protocol Data Unit)**
    T=0 : protocole orienté octet, T1 : protocole orienté paquet,
    T=14 : réservé pour les protocoles propriétaires,

  - ✓**La sélection d'un type de protocole,**

  - ✓**La réponse à un reset (ATR, ou Answer To Reset) qui correspond aux données envoyées par la carte immédiatement après la mise sous tension,**

  - ✓**Les signaux électriques, tels que le voltage, la fréquence d'horloge et la vitesse de communication.**

Insertion de la carte dans un lecteur

- **La norme ISO 7816-3 précise la mise sous tension de la carte et son arrêt**
- **Dans le lecteur, il y a un circuit d'interface :**
  - ✓ Connexion de la carte et activation de ses contacts par le circuit d'interface
  - ✓ Reset de la carte
  - ✓ Réponse au reset ou ATR émanant de la carte
  - ✓ Dialogue entre la carte et l'application via le circuit d'interface
  - ✓ désactivation des contacts par le circuit d'interface
  - ✓ Retrait de la carte

Étapes d'activation de l'interface avec la carte

➢ **Une fois la carte placée dans le lecteur, les opérations suivantes sont déclenchées :**

- ✓Mise au niveau bas de l'entrée RST
- ✓ Alimentation de la carte via son entrée Vcc
- ✓ Mise en mode réception de la ligne I/O du circuit d'interface du lecteur
- ✓ Mise au niveau repos de Vpp de la carte
- ✓ Génération d'une horloge stable sur l'entrée CLK de la carte
- ✓ Un reset est alors provoqué par le circuit d'interface

Désactivation de l'interface avec la carte

➢**La désactivation normale a lieu lorsque la transaction en cours se termine et le terminal vous invite à retirer votre carte. Avant l'affichage du message de retrait, il y a :**

- ✓ Mise au niveau bas de l'entrée RST
- ✓ Mise au niveau bas de CLK
- ✓ Mise au niveau inactif de Vpp
- ✓ Mise au niveau inactif de I/O
- ✓ Coupure de l'alimentation Vcc
- ✓ Retrait de la carte (le retrait de la carte avant la fin de la transaction doit être pris en charge par l'application).

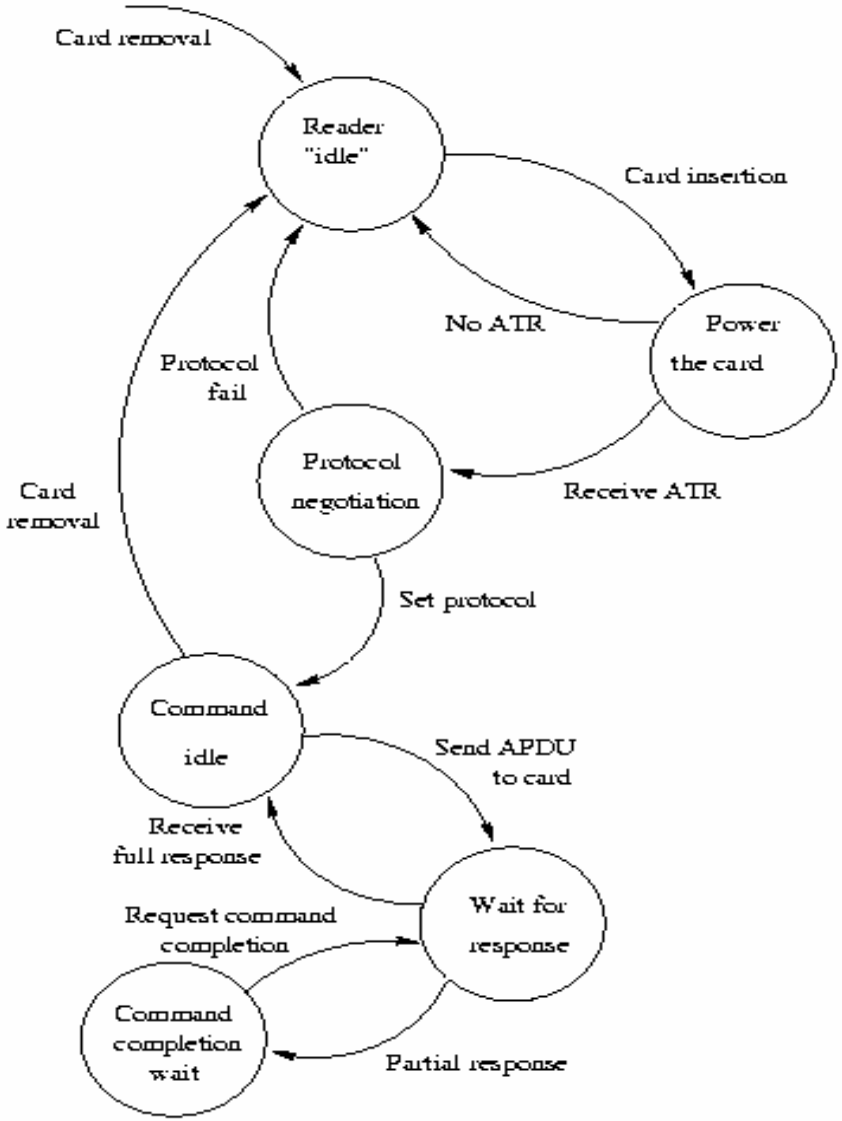
ATR défini dans l'ISO 7816-3

- **ATR (Answer To Reset):**
  - ✓**Dès que la carte est mise sous tension, elle envoie un message de réponse d'initialisation appelé ATR, il peut atteindre une taille maximale de 33 octets. Il indique à l'application cliente les paramètres nécessaires pour établir une communication avec elle.**

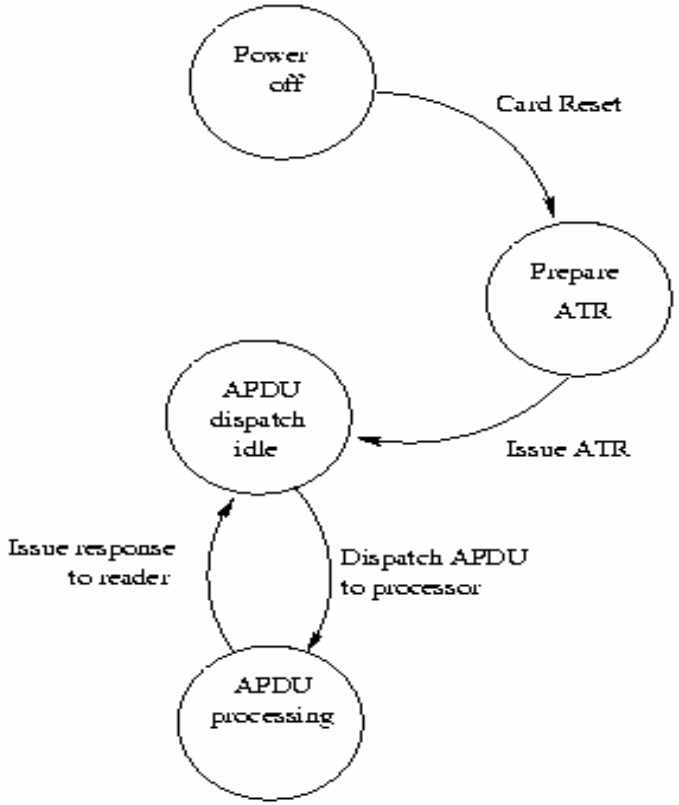
  - ✓**Paramètres envoyés par la carte :**

    - Le protocole de transport ;
    - Taux de transmission des données ;
    - Numéro de série de la puce …

Comportements de la carte et du lecteur lors d'un Reset



Reader State Diagram



Card State Diagram

Les cartes sans contact

Normes principales des cartes sans contact

- **La norme ISO 14443-1 précise les caractéristiques physiques de la carte**

- **La norme ISO 14443-2 indique comment télé-alimenter la carte et décrit les signaux électriques pour établir le dialogue avec la carte**

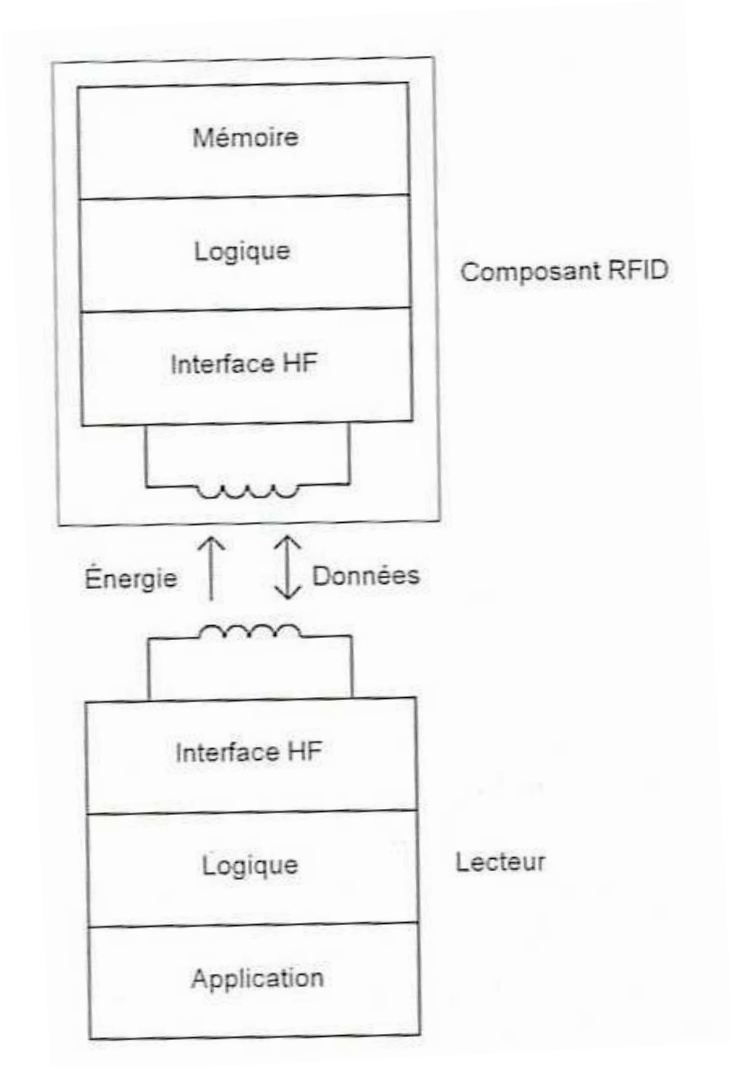
- **La norme ISO 14443-3 définit les phases d'initialisation des échanges et de gestion de collision**

- **La norme ISO 14443-4 définit le protocole de transmission**

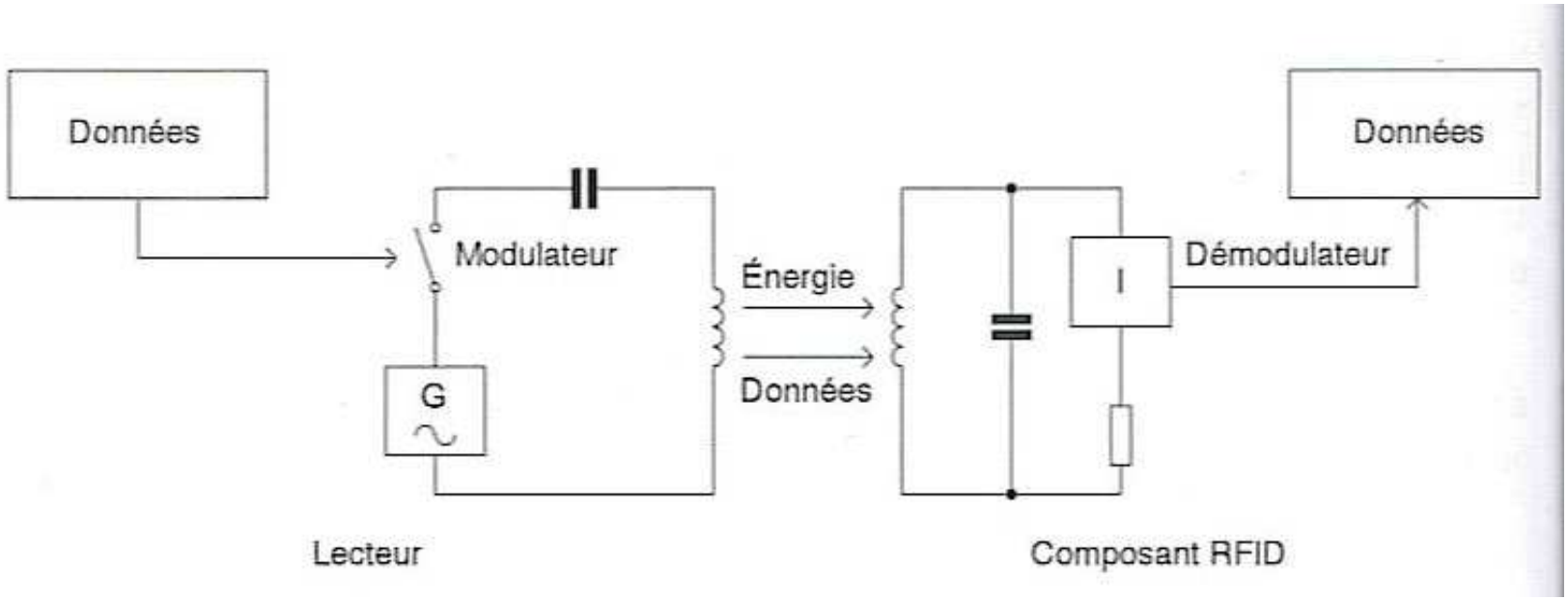
Caractéristiques des cartes sans contact

- **Les cartes à puce sans contact font partie des produits RFID (Radio Frequency Identification Devices)**

- **Le composant et le lecteur sont reliés par un champ électromagnétique haute fréquence, qui véhicule les données et l'alimentation du composant**



Liaison RFID Lecteur -> Composant



Liaison RFID Lecteur -> Composant

- **Le lecteur génère un champ HF de manière permanente**
- **Le composant RFID recueille l'énergie (très faible) destinée à l'alimenter**
- **Le champ HF généré par le lecteur est interrompu (réduit) en fonction des données à transmettre (champ modulé)**
- **Le composant RFID récupère les variations d'énergie du champ HF, par conséquent les données transmises**

Liaison RFID Composant -> Lecteur

- **Le composant ne peut pas réellement émettre**

- **Pour émettre ses données, il influe sur le champ HF reçu en faisant varier l'énergie qu'il absorbe**

Famille des cartes à puce sans contact

- **Cartes à puce à couplage très proche (quelques mm) (Close Coupling) : couvertes par les normes ISO 10536**

- **Cartes à puce à couplage de proximité (proximity coupling) (quelques cm) couvertes par les normes ISO 14 443.**

- **Cartes à puce à couplage éloigné ou de voisinage (vicinity coupling) (quelques dizaines de cm) couvertes par les normes ISO 15 693**

Caractéristiques mécaniques des cartes à puce sans contact

- **La norme ISO 14 443-1 s'inspire de la norme ISO 7816-1**

- **Les cartes de type ID1 (format carte bancaire)**

L'antenne est constituée de fils qui longent le périmètre externe

Caractéristiques électriques des cartes à puce sans contact

- **La norme ISO 14 443-2**

- **Alimentation de la carte par transfert de puissance entre la carte et le lecteur** au moyen d'un signal HF 13.56 MHz +/– 7 KHz

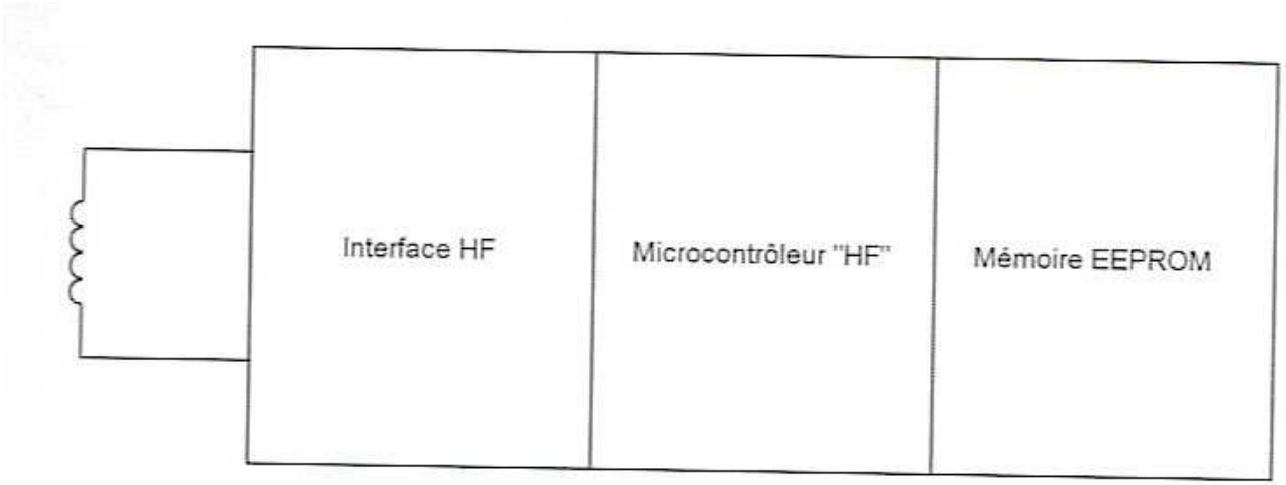
- **1.5 A/m ≤ le champ magnétique ≤ 6.5 A/m**

Les normes ISO 14 443-3 et 4

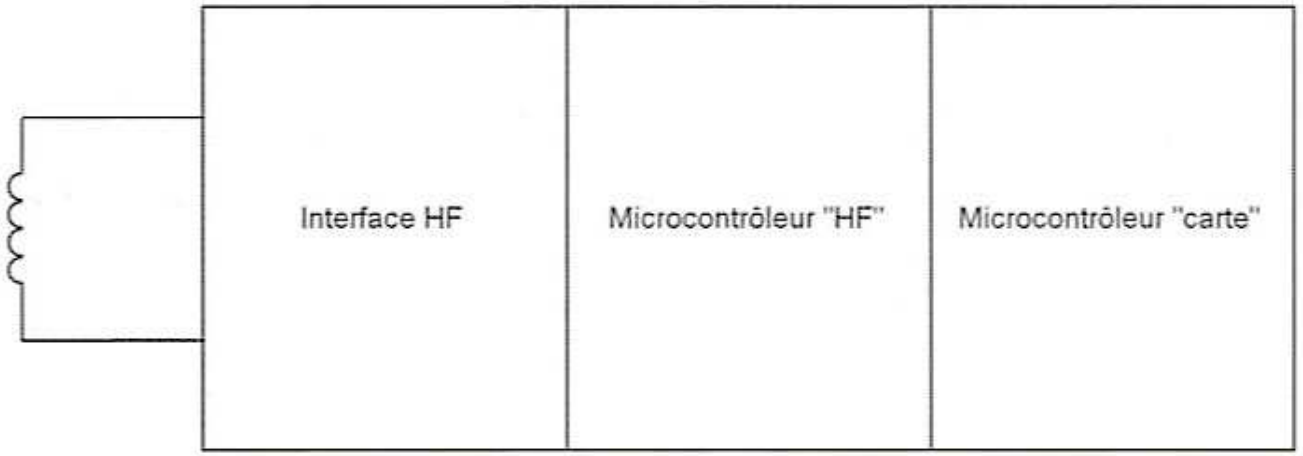
- **Si plusieurs cartes sont à proximité d'un lecteur, les cartes peuvent tenter un dialogue avec le lecteur**
- **Mélange des signaux émis => collision**
- **Mesure anti-collision pour la modulation A (déterministe)**
- **Mesure anti-collision pour la modulation B (probabiliste)**
- **Norme 14 443-4 : dialogue entre la carte et le lecteur**

Synoptique interne d'une carte à mémoire sans contact

Microcontrôleur HF : sert à la gestion du dialogue avec le lecteur, à l'anticollision et à la correction des erreurs de dialogue.
Exemple de la carte MF1 IC 550 (carte de la gamme Mifare Standard du groupe Philips)



Synoptique interne d'une carte à microcontrôleur sans contact



Établissement du dialogue avec une carte S/contact

- **Au contact avec une antenne d'un lecteur :**

- la carte fait un reset et attend une commande du lecteur
- les cartes situées dans la zone d'influence répondent par ATQA (Answer To Request Code)
- processus anti-collision : lecture de l'identifiant de la carte (numéro unique identifiant la carte lors de sa fabrication)

- élection d'une carte

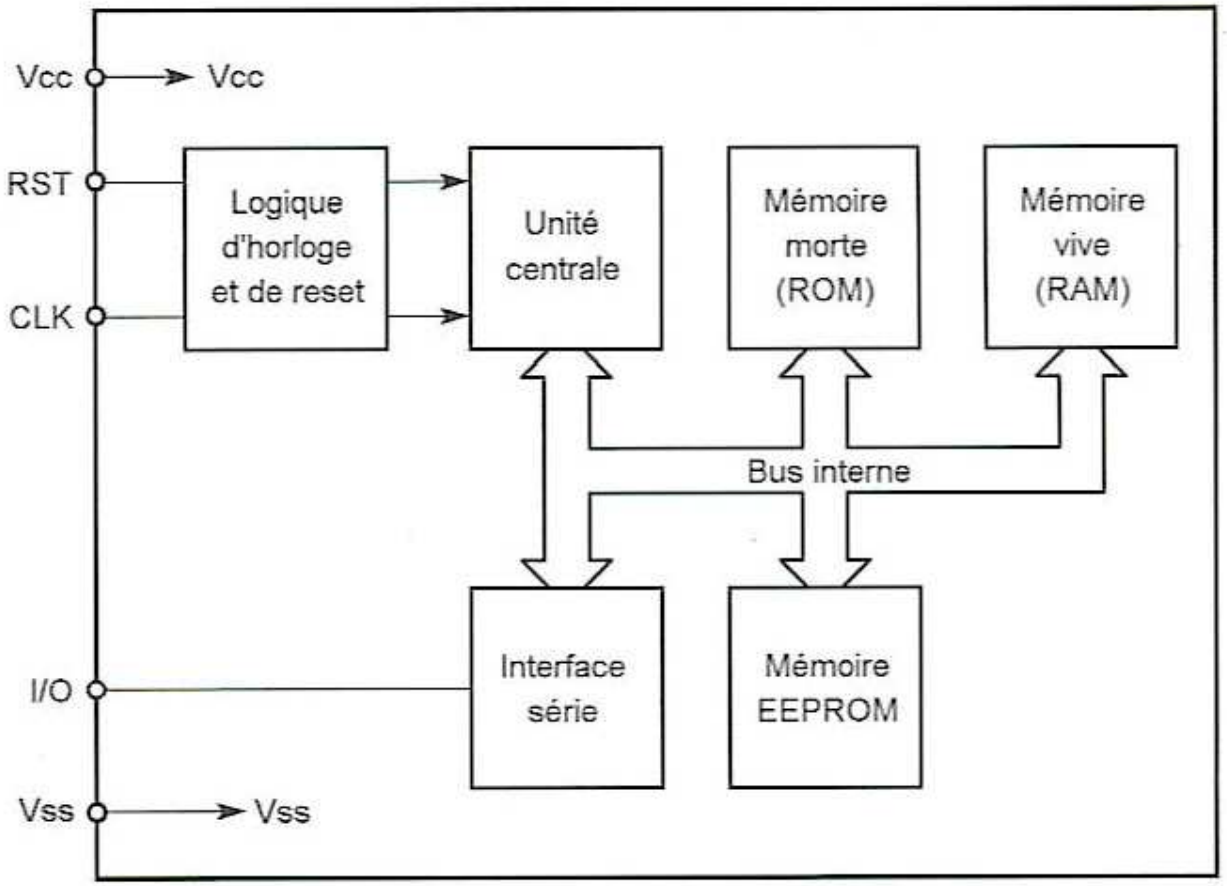
- le lecteur informe les autres cartes non sélectionnées qui retournent dans un état semi-passif

-la carte sélectionnée envoie une réponse ATS (Answer To Select) qui contient les caractéristiques de la carte

- Échange d'informations utiles

Les cartes à micro-contrôleur

Synoptique interne d'une carte à microcontrôleur



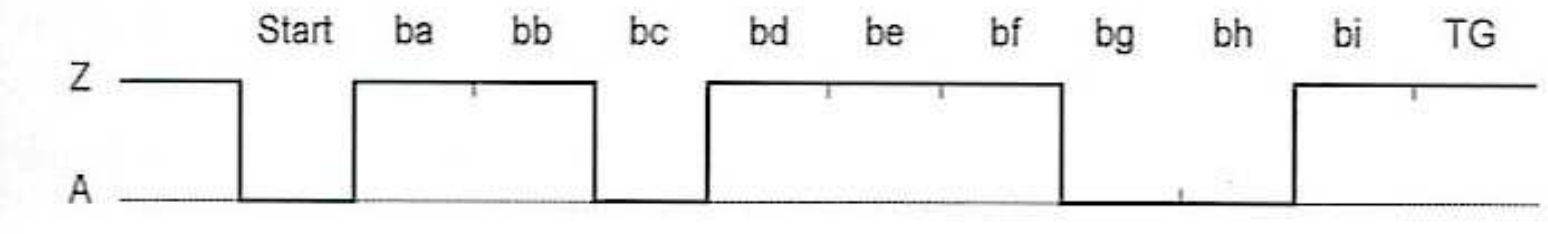
Les vraies cartes à puce

- **L'ATR est la réponse de la carte au reset du lecteur**
- **L'ATR diffère des cartes à mémoire aux cartes à microcontrôleur**
- **l'ATR au minimum = 2 octets, au maximum = 33 octets**
- **Transmission en mode asynchrone semi-duplex**
- **La fréquence d'horloge comprise ente 1 et 5 MHz pour permettre à n'importe quel lecteur de lire le 1er caractère**
- **Communication entre le lecteur et la carte via la ligne bidirectionnelle I/O**

Chronogramme de la réponse reset

- **Comm. Asynchrone : bit start + 8 bits de données (ba à bh) + bit de parité paire + TG (Temps de Garde = un ou plus bits Stop)**

- **A :niveau bas et Z niveau haut**
- **le délai entre 2 caractères est au moins de 12 etu et TG = 2 etu**



Caractère initial de l'ATR

- Premier caractère de l'ATR = TS (caractère initial)

- TS peut prendre 2 valeurs : (ZZAAAAAA)[1] ou (ZZAZZZAA)[2]

- 1: convention inverse :
  - niveau bas A = « un » logique
  - niveau haut Z = « zéro » logique
  - ba (bit transmis en premier) = bit 7 de poids fort
  - bh (bit transmis en dernier)=bit 0 de poids faible

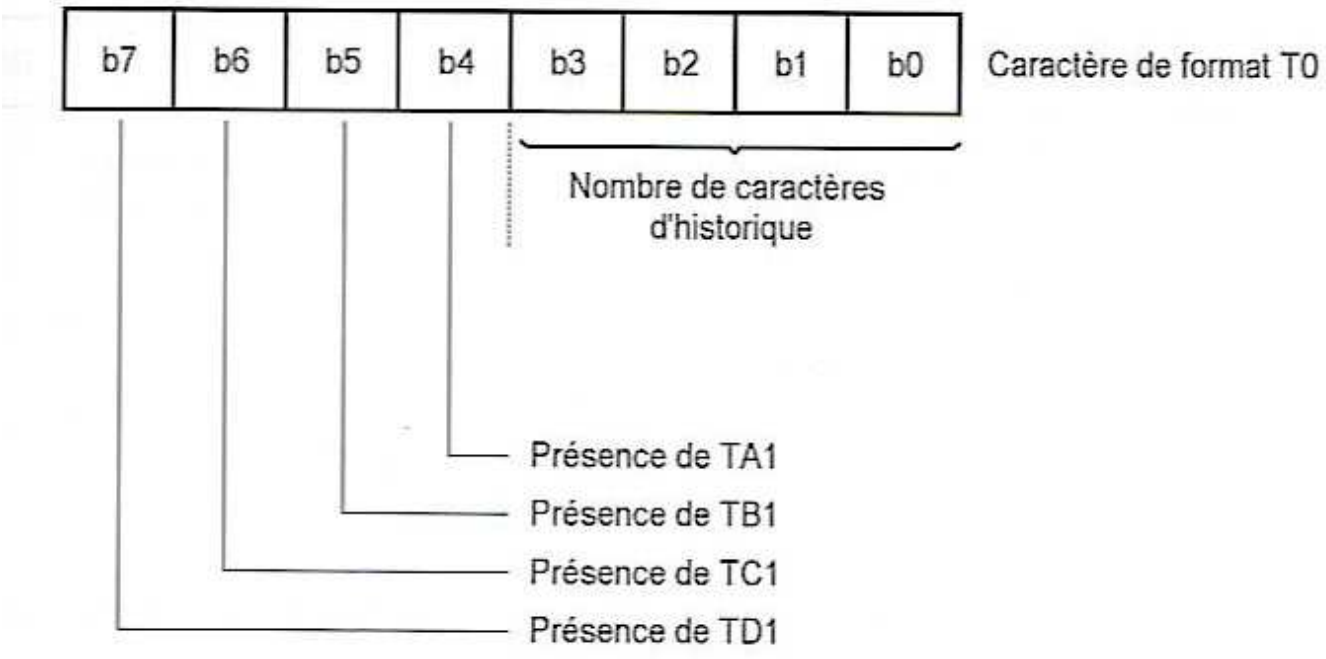
  TS = 0011 1111 (3F, en héxa)

- 2: convention directe :
  - niveau bas A = « 0 » logique
  - niveau haut Z = « 1 » logique
  - ba (bit transmis en premier) = bit 0 de poids faible
  - bh (bit transmis en dernier)=bit 7 de poids fort

  TS = 0011 1011 (3B, en héxa)

Caractère de format

- appelé aussi caractère T0
- 2ème caractère de l'ATR
- composé de :
  - -Partie Y1 (b_7 à b_4)
  - -Partie K (b_0 à b_3) facultative (n'est pas normalisée)

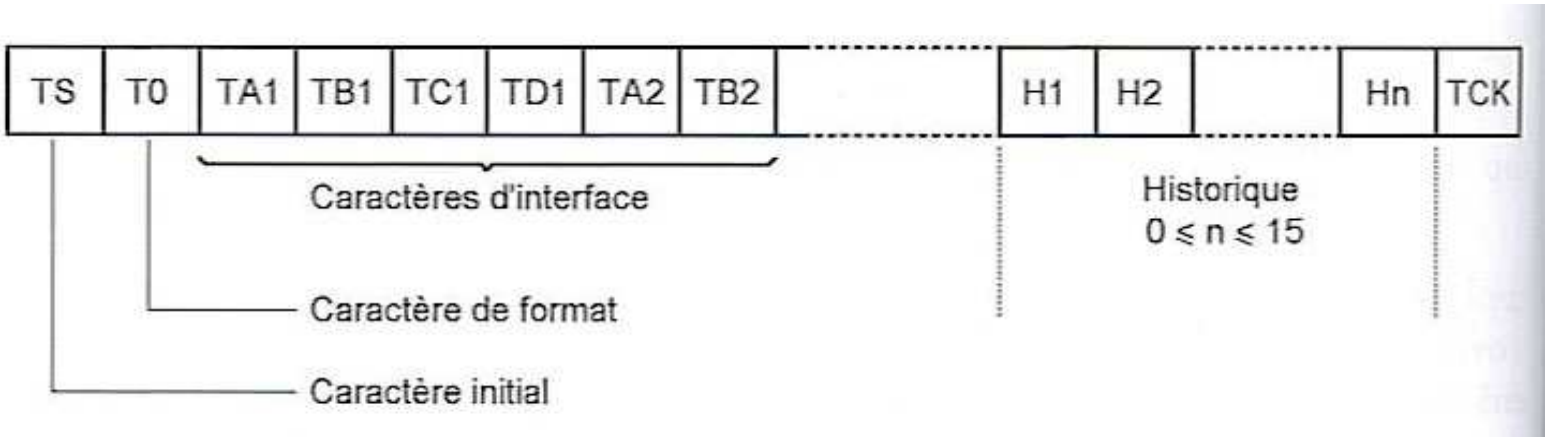


Caractère de format (suite)

- b_4=1 si le car TA_1 est transmis dans l'ATR
- b_5=1 si le car TB_1 est transmis dans l'ATR
- b_6=1 si le car TC_1 est transmis dans l'ATR
- b_7=1 si le car TD_1 est transmis dans l'ATR

- Les poids forts de TD_1 indique si les caractères supérieurs sont transmis dans l'ATR, par ex. :

  -Si TD_1 contient 1010 => TD_2 et TB_2 sont transmis



Caractère TA_1

TA1

b7	b6	b5	b4	b3	b2	b1	b0
F				D			

b7	b6	b5	b4	F	fs max (MHz)
0	0	0	0	Interne	-
0	0	0	1	371	5
0	0	1	0	558	6
0	0	1	1	744	8
0	1	0	0	1116	12
0	1	0	1	1488	16
0	1	1	0	1860	20
0	1	1	1	RFU	-
1	0	0	0	RFU	-
1	0	0	1	512	5
1	0	1	0	768	7,5
1	0	1	1	1024	10
1	1	0	0	1536	15
1	1	0	1	2048	20
1	1	1	0	RFU	-
1	1	1	1	RFU	-

RFU : Réservé pour un usage futur

b3	b2	b1	b0	D
0	0	0	0	RFU
0	0	0	1	1
0	0	1	0	2
0	0	1	1	4
0	1	0	0	8
0	1	0	1	16
0	1	1	0	RFU
0	1	1	1	RFU
1	0	0	0	RFU
1	0	0	1	RFU
1	0	1	0	1/2
1	0	1	1	1/4
1	1	0	0	1/8
1	1	0	1	1/16
1	1	1	0	1/32
1	1	1	1	1/64

RFU : Réservé pour un usage futur

Caractère TA_1 (suite)

- F et D définissent la vitesse de transmission utilisée après l'ATR
- Vitesse de transmission= $D*f_s/F$ bits/s avec f_s fréquence d'horloge en Hz
- Durée d'un bit (etu)= $F/(D*f_s)$ secondes
- Valeur min de f_s = 1 MHz (selon la norme)
- Valeur max de f_s : dictée par TA_1

Caractère TB_1 / TC_1

- **TB1 : de plus en plus rare d'utilisation**

- **Contient la valeur de la haute tension de programmation (tension à appliquer sur Vpp pour écrire sur la carte)**

- **TC1 code un paramètre N= temps de garde supplémentaire**

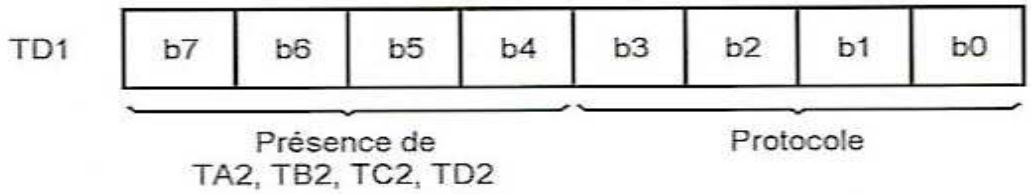
Si $0 \leq N \leq 254$ (FE en Hexa), TG=N*etu
Si N=255 (FF en Hexa), TG = 11*etu

- **Pour les caractères envoyés par la carte TG=2*etu.**

TC1 demandé par la carte permet un TG supplémentaire uniquement dans le sens Lecteur -> Carte

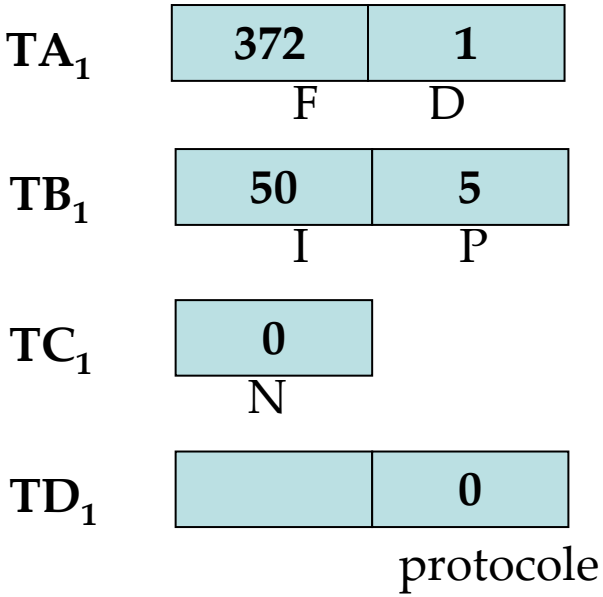
Caractère TD_1

- Code le caractère TA_2, TB_2, TC_2 et TD_2 (bits poids forts)
- bits de poids faible (numéro du protocole utilisé, T=0 et T=1)



b3	b2	b1	b0	Protocole T =
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

ATR par défaut



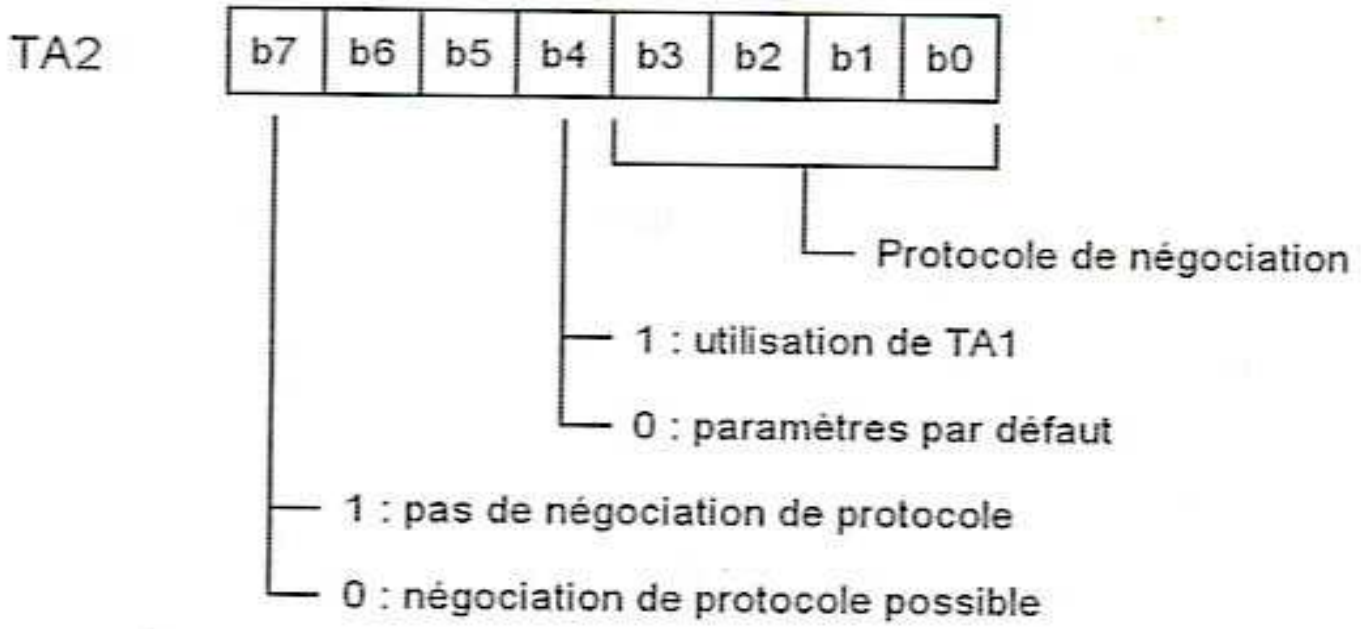
- **TCK présent si protocole ≠ 0 dans TD1**
- **TCK = valeur de sorte qu'un Ou exclusif entre les octets de T0 (inclus) et TCK (inclus) soit nul**

Négociation de vitesse de dialogue

- **Mécanisme proposé depuis l'existence de la norme ISO 7816-3 en 1999**
- **Implanté plus récemment dans les cartes**
- **Idée : changer de vitesse au cours des échanges pour augmenter la sécurité en brouillant les simples amateurs de piratage**
- **Les cartes dialoguent à 9600 bits/s pendant l'ATR (vitesse connue de tout port série RS232 facilite la réalisation d'espions)**

Caractère TA_2

- **Si caractère TA2 existe => il indique les conditions de négociation**



Caractère TA_2 (suite)

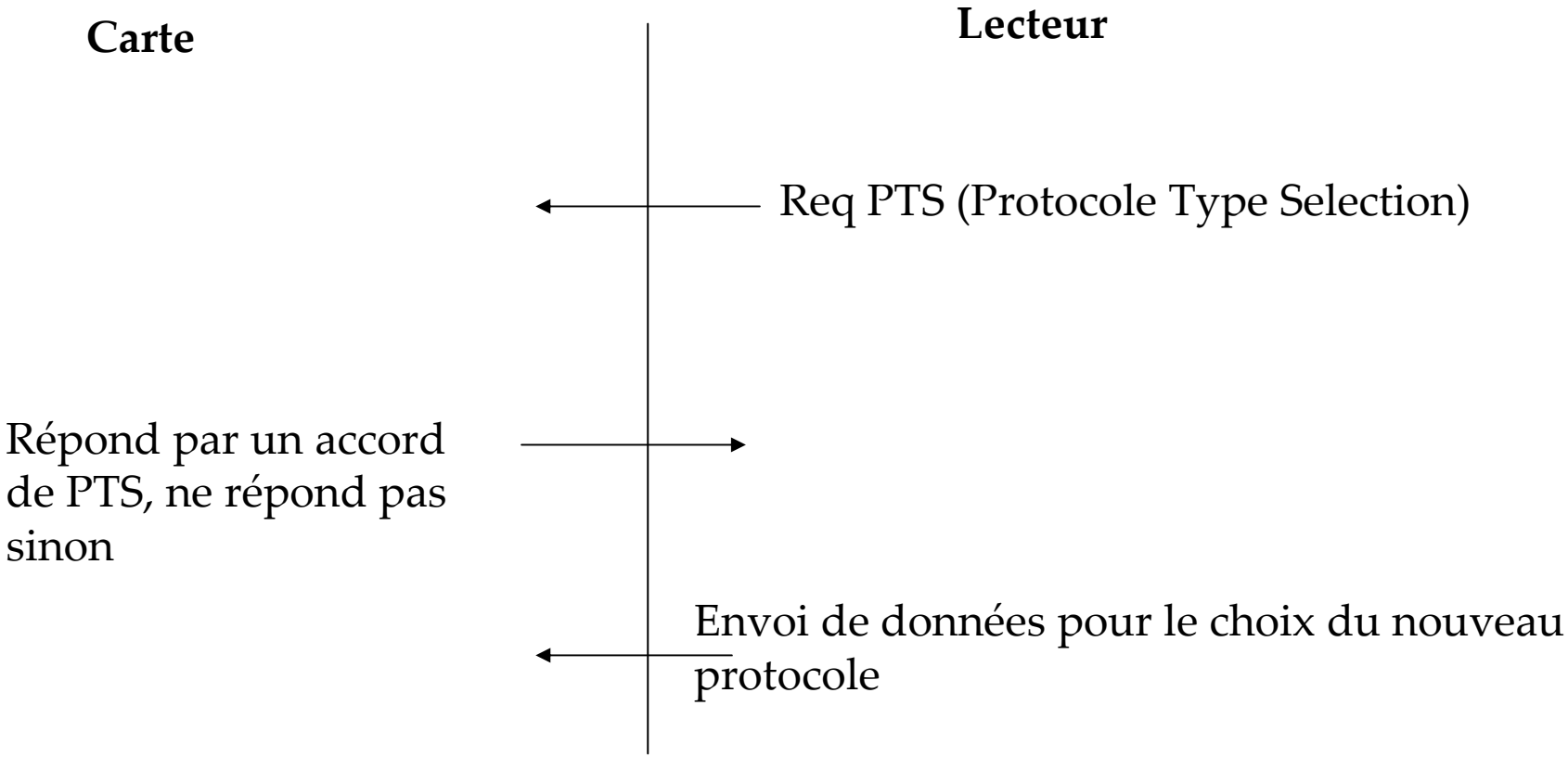
- **Si la carte supporte un seul protocole + Si paramètres par défaut utilisés (dictés par TA_1, fréquence) + Si N de TC_1 < 255 => dialogue commence dès la fin de l'ATR avec le protocole de TD_1**

- **Si la carte supporte plus de protocole et/ou TA_1 spécifie d'autres valeurs et/ou N=255 => négociation avec l'ATR pour déterminer le type de protocole**

- **Si la carte supporte plus d'un protocole alors que le lecteur ne supporte qu'un protocole différent de T=0, le lecteur rejette la carte.**

- **Si la carte supporte plusieurs protocoles dont T=0, c'est ce protocole qui doit être utilisé par défaut dans TD_1 si la négociation est absente.**

Étapes de la négociation de protocole



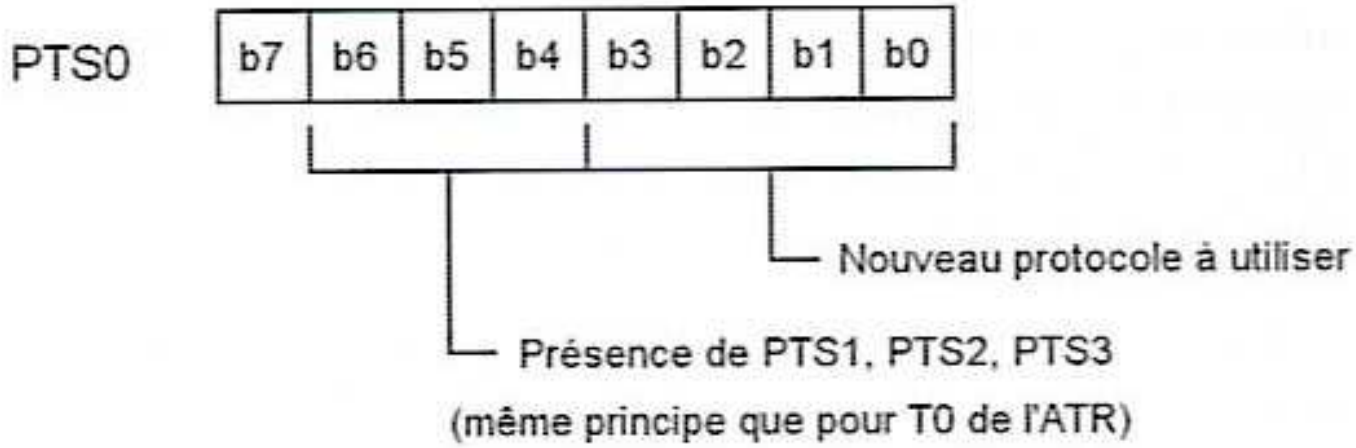
PTS

- **= 3 à 6 octets**

- 1er octet : PTSS (PTS Start) = FF

-2ème octet : caractère de format PTS0

-3 octets optionnels : PTS1, PTS2 et PTS3, dernier octet : PCK octet de contrôle



PTS (suite)

- **PTS3 : non défini par la norme**
- **PTS1 : paramètres D et F comme le car TA1 de l'ATR**
- **Si PTS1 absent, les valeurs par défaut (F=372 et D=1)**
- **PTS2=N (de TC1 de l'ATR)**
- **Si b0 de PTS2=1 la carte accepte que N=255 (tps de garde est réduit à 1 etu)**
- **Si b1=1 la carte peut utiliser un TG=12 etu entre chaque caractère envoyé au lecteur**
- **Si b2=0 valeur par défaut, aucun TG supplémentaire**
- **b3 à b7 non utilisés**
- **PCK contient une valeur de manière qu'un ou exclusif des bits allant de PTSS à PCK soit nul.**

Réponse au PTS par la carte

- **PTS Confirm (=FF) en réponse à PTSS**

- **rien à PTS0**

- **PTS1 en réponse à PTS1 (si pas de réponse, le lecteur déduit l'utilisation des valeurs par défaut F=372 et D=1)**

- **PTS2 en réponse à PTS2 (si pas de réponse à PTS2, la carte est rejetée)**

- **Conclusion**
  - **-Protocole un peu contraignant**
  - **-Ignoré sur la majorité des cartes actuelles en partie pour cette raison**

Les protocoles TPDU/APDU

Protocole TPDU T=0

- **Le caractère TD1 définit théoriquement 15 protocoles différents**

- **Il existe deux protocoles T=0 et T=1 (T=0 le plus utilisé)**

- **Protocole T=0 (TPDU : Transmission Protocol Data Unit)**
  - **est de type caractère**
  - **mode de fonctionnement de type commande/réponse**
  - **Le lecteur est l'initiateur des échanges**

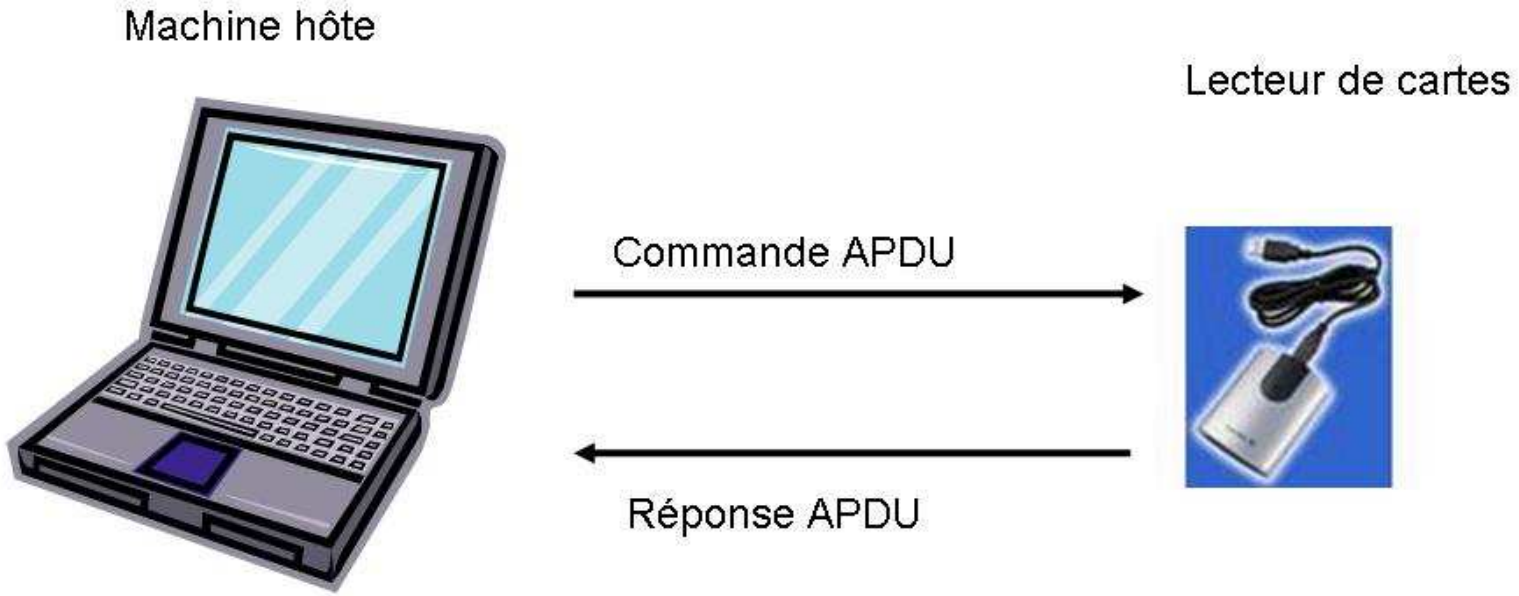
L'ISO 7816-4

- **Elle définit les messages APDU (Application Protocol Data Units) utilisés par les cartes à puce pour communiquer avec le lecteur.**

- **Les échanges s'effectuent en mode client-serveur,**

- **Le terminal est toujours l'initiateur de la communication.**

L'ISO 7816-4 : Le protocole APDU



Format des commandes APDU

Commande APDU						
Entête obligatoire				Corps optionnel		
CLA	INS	P1	P2	Lc	Data field	Le

- CLA (1 octet): Classe d'instructions --- indique la structure et le format pour une catégorie de commandes et de réponses APDU
- INS (1 octet): code d'instruction: spécifie l'instruction de la commande
- P1 (1 octet) et P2 (1 octet): paramètres de l'instruction
- Lc (1 octet): nombre d'octets présents dans le champ données de la commande
- Avec Le=0, - Si cde d'écriture => pas de données utiles
  - Si cde de lecture => la cde doit retourner 256 octets de données utiles
- Data field (octets dont le nombre est égal à la valeur de Lc): une séquence d'octets dans le champ données de la commande

Format des réponses APDU

Réponse APDU		
Corps optionnel	**Partie obligatoire**	
Data field	**SW1**	**SW2**
•Data field (longueur variable): une séquence d'octets reçus dans le champ données de la réponse •SW1 (1 octet) et SW2 (1 octet): Status words (Mots d'état)—état de traitement par la carte		

SW1 SW2 =	0x90 0x00	Succès
	0x6E 0x00	CLA error
	0x6D 0x00	INS error
	0x6B 0x00	P1, P2 error
	0x67 0x00	LEN error
	0x98 0x04	Bad PIN
	0x98 0x40	Card blocked

Exemples de classes : CLA

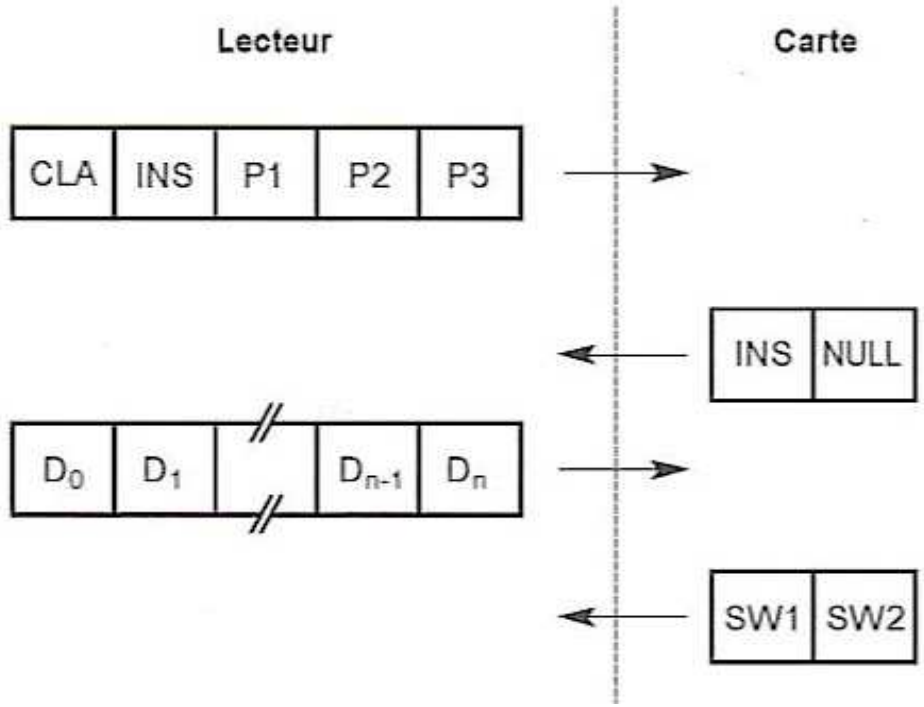
Classe (valeur de CLA)	Application
0x	Commandes standardisées conformes à la norme ISO 7816-4
80	Porte-monnaie électronique conforme à la norme ISO 1546-3
8x	Applications et commandes spécifiques
8x	Cartes de crédit avec des puces conformes à la norme EMV 2
A0	Téléphones GSM conformes à la norme ETSI GSM 11.11
BC	Cartes bancaires conformes à la norme EMV

Exemples de cartes

Champ de la commande APDU	Valeurs
CLA	BC = cartes de crédit françaises, cartes vitales françaises, A0 = cartes SIM (téléphonie) 00 = cartes Monéo (porte-monnaie en France), Mastercard, Visa
INS	20 = vérification du PIN, B0 = Lecture B2 = Lecture de record D0 = Écriture DC = Écriture de record A4 = Sélection du répertoire (directory) C0 = Demander une réponse (get an anwer)
P1, P2	paramètres contenant des adresses à lire
LEN	longueur prévue pour la réponse ou bien longueur de l'argument de l'instruction
ARG	contient LEN octets (octets à écrire, PIN à vérifier, etc.)

Protocoles TPDU/APDU

- **TPDU : Transmission Protocol Data Unit**
- **APDU: Application Protocol Data Unit**
- **Les octets (INS,NULL) sont encapsulés dans les commandes APDU**



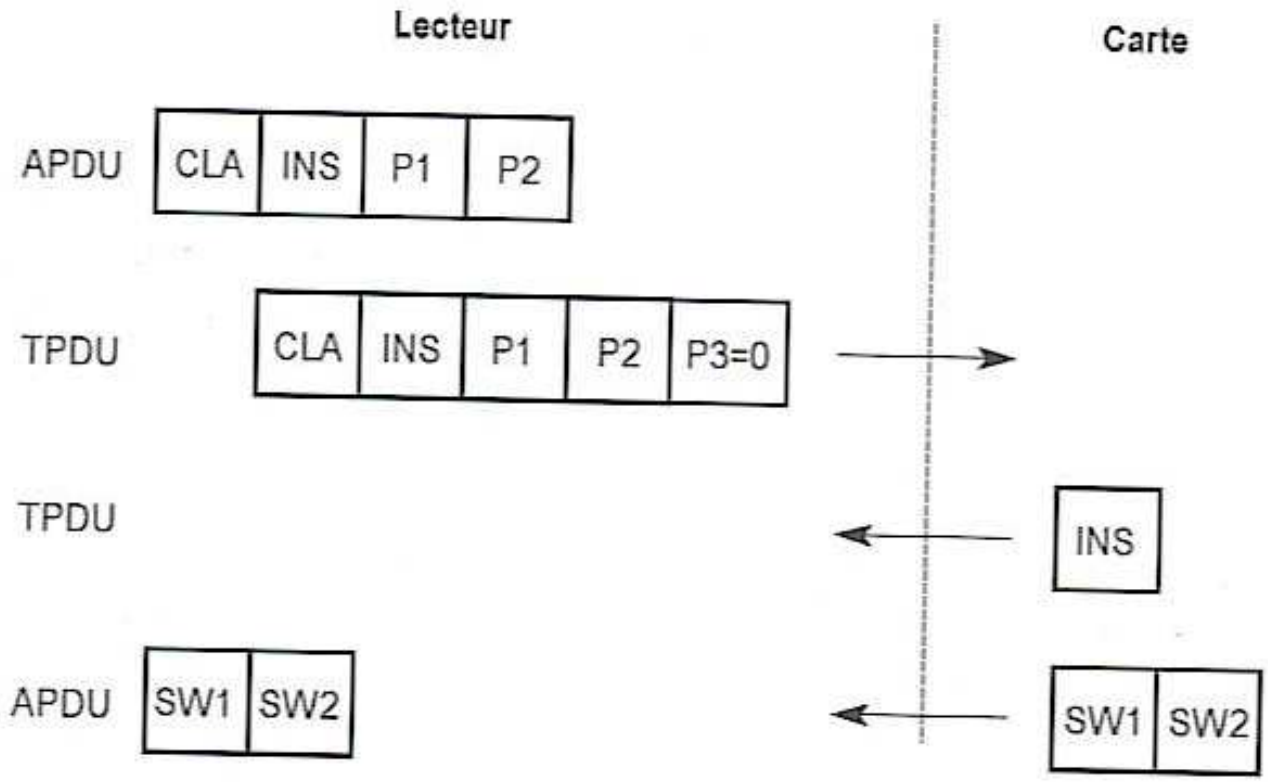
Protocole APDU (ISO 7816-4)

- **La norme relative aux cartes à puce a essayé de se conformer au modèle OSI**

- **Couche application (APDU) n'est pas vraiment séparée de la couche transport (TPDU)**

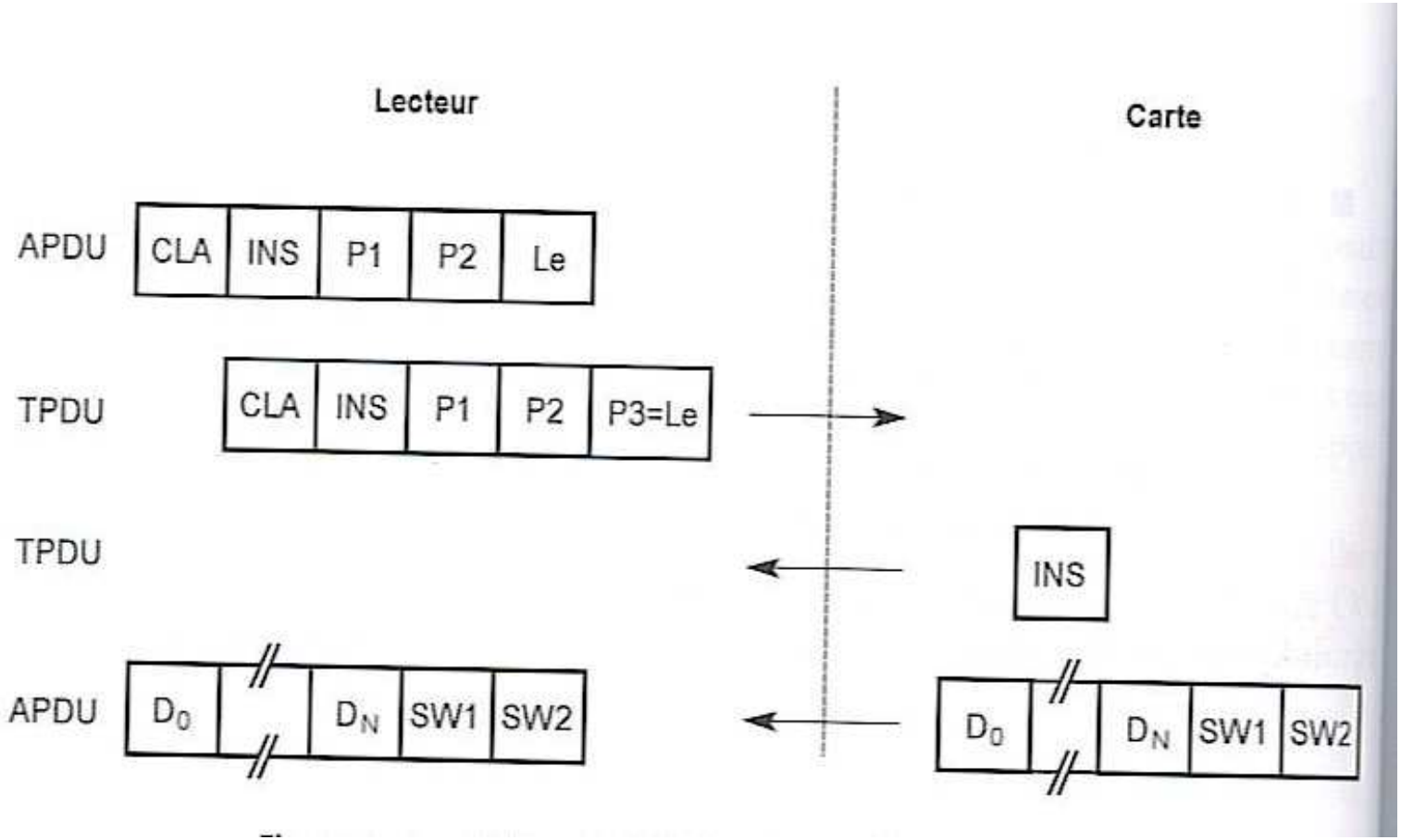
- **Il existe 5 types de commandes APDU selon qu'il y a ou non échange de données utiles**

Envoi d'une commande sans données utiles



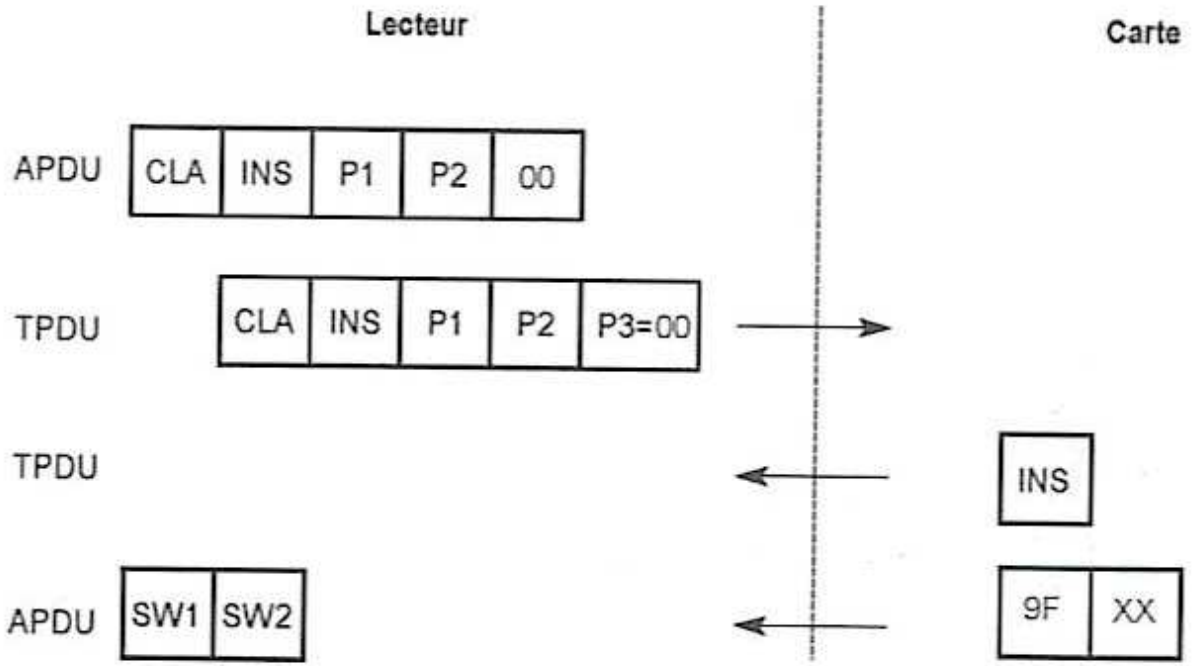
Commande avec réception de données utiles depuis la carte

- **Le nb d'octets retournés peut être différent de *Le***



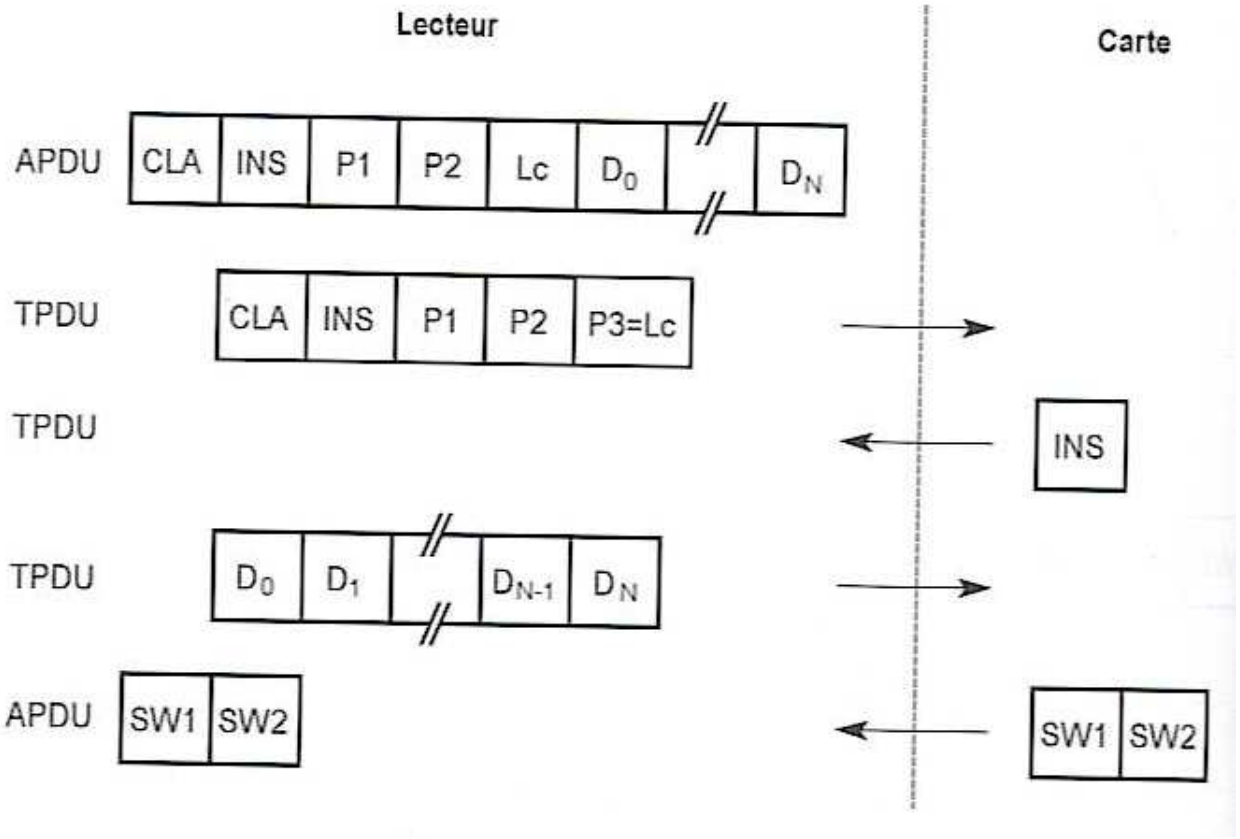
Commande avec invitation à lire des données depuis la carte

- ***Le=0*** **car le nb d'octets attendus est inconnu**
- **SW1 SW2 = 9F XX (avec XX le nb d'octets disponibles pour cette cde**
- **Le lecteur doit refaire sa commande avec ce nb d'octets**

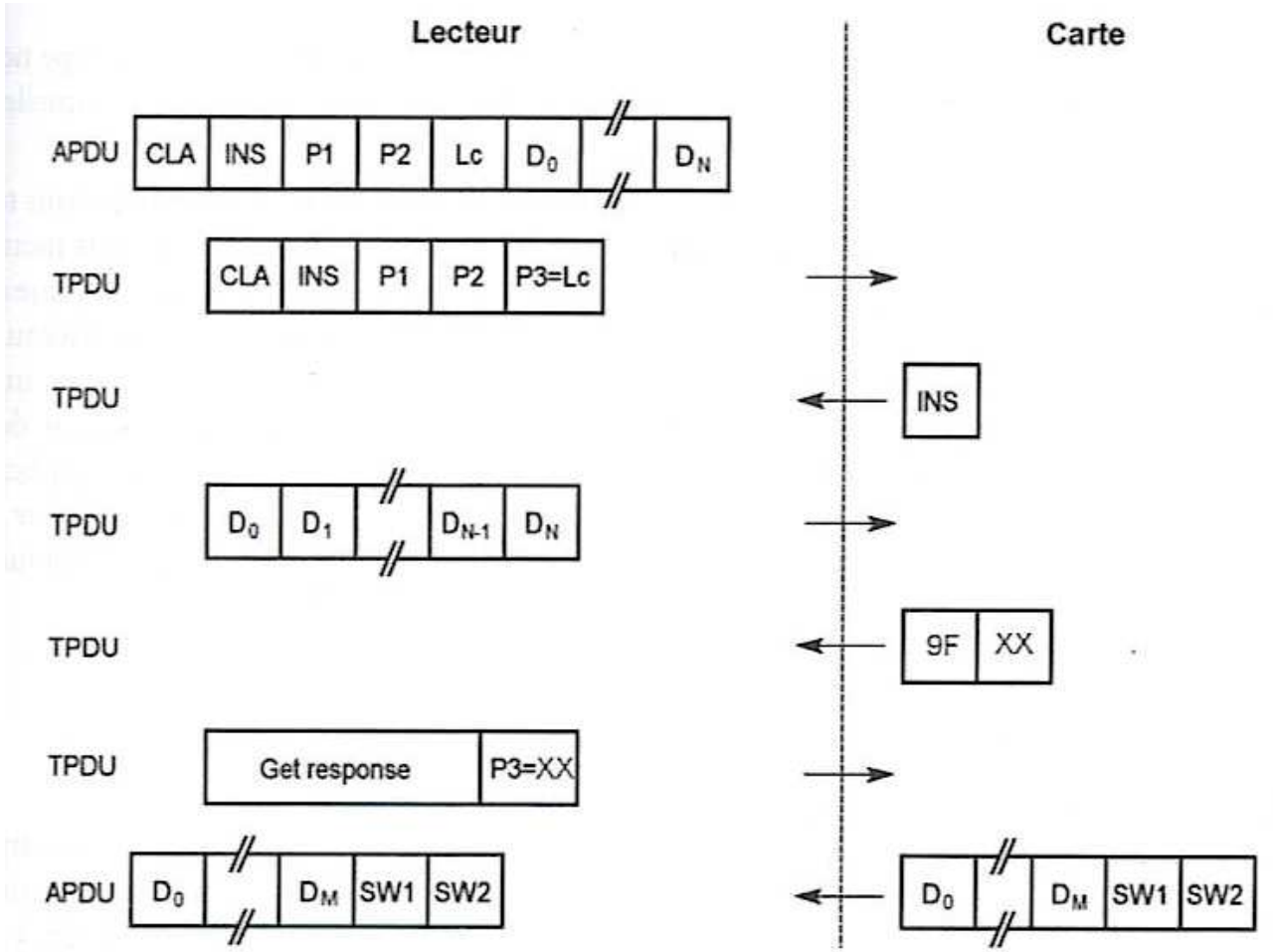


Commande avec envoi de données utiles vers la carte

- ***Lc*: est le nb d'octets envoyés**



Commande avec envoi et réception de données utiles



Le protocole T=1 (ISO7816-3)

Protocole T=1

- **Protocole ambitieux**
- **Protocole peu utilisé**
- **Protocole plus proche du modèle OSI**
- **Échange de blocs structurés**

Structure d'un bloc

- **Chaque bloc commence par un champ obligatoire**
  - **prologue field**
  - **données**
  - **champ de contrôle**

-NAD = (adr du destinataire, adr émetteur)

-LEN: nb d'octets de données (tte la cde APDU)

-PCB: octet de contrôle (b_7b_6=11 si bloc supérieur, =00 si bloc d'infos, =10 si réception prête)

-LRC: résultat du OuX de tous les octets le précédant

Prologue			Information	Épilogue
NAD	PCB	LEN	« Données » (APDU)	LRC
1 octet	1 octet	1 octet	0 à 254 octets	1 octet

L'ISO 7816-5

- **Elle définit la procédure d'enregistrement et d'attribution des identifiants des applications (AID, ou *Application IDentifier*).**

- **Un unique AID est associé à chaque application = {RID, PIX}**

- **RID : le numéro d'enregistrement du fournisseur d'application attribué par l'ISO.**
  Le RID doit être le même pour le paquetage et l'applet.

Application identifier (AID)	
National registered application provider (RID)	Proprietary application identifier extension (PIX)
5 octets	0 to 11 octets

La gestion des fichiers d'une carte à puce

Les fichiers

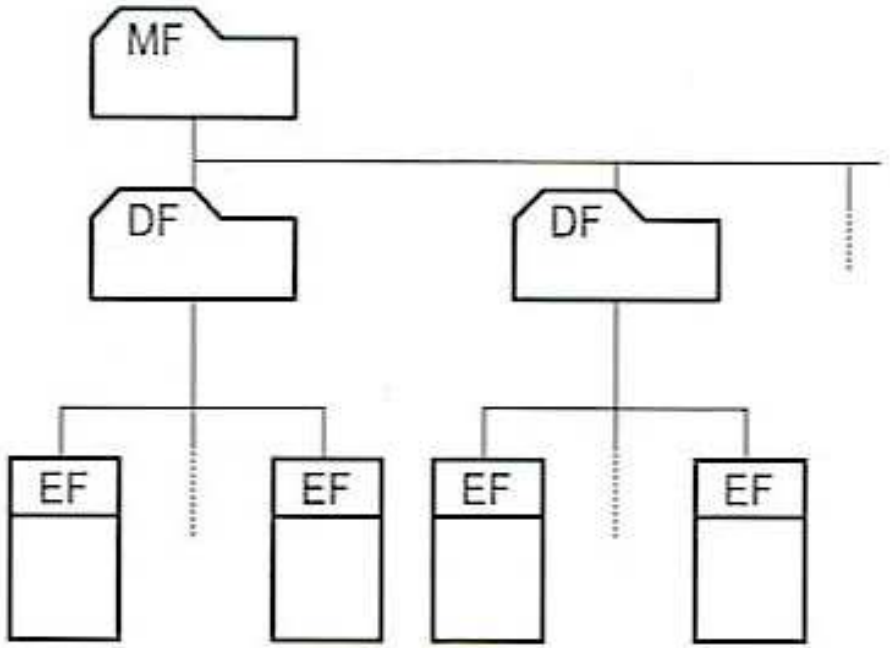
- **Cartes à mémoire: composées de zones mémoire destinées à la lect/écrit**

- **Cartes à puce: comportent un véritable système de fichiers**

  - création/destruction de fichiers
  - attribuer un nom à un fichier
  - définir des restrictions d'accès en Letc/écrit
  - lire/écrire dans un fichier
  - le plus contraignant est l'échange en binaire entre le lecteur et la carte

Arborescence des fichiers et répertoires

- La norme ISO 7816-4 définit l'arborescence de fichiers
- Vocabulaire propre aux cartes à puce
  - EF (Elementary Files): fichiers élémentaires (feuilles de l'arbre)
  - DF (Dedicated Files) : qu'on peut appeler Directory Files (répertoires)
  - MF (Master File): fichier maître, c'est le répertoire racine



Identification et nommage des fichiers

- **Plusieurs façons d'identifier un fichier dans la norme 7816-4**

a - un DF ou un EF peut être référencé à l'aide d'un FID (File IDentifier) sur 2 octets

b - un EF peut être référencé à l'aide d'un chemin d'accès : concaténation des FID en commençant de MF

c - un EF peut être référencé à l'aide d'un short FID sur 5 bits (valeurs autorisées 0 à 30)

d - un DF peut être référencé par un nom codé sur 1 à 16 octets

La méthode la plus utilisée sur les cartes actuelles est la méthode (a)

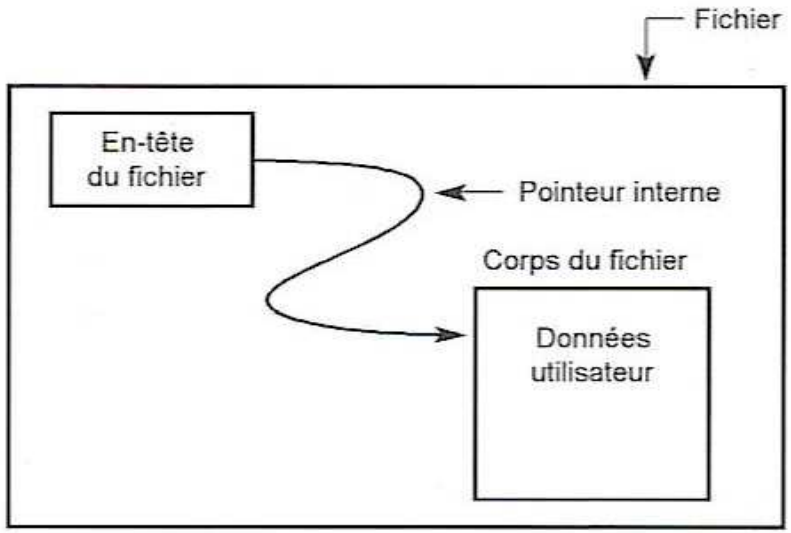
Contraintes pour la méthode (a)

- **Le FID de MF est toujours égal à 3F 00 (imposé par la norme)**
- **Le FID FFFF est inutilisé (RFU)**
- **Le FID 3FFF n'est pas utilisé par un EF ou DF car il fait référence au DF courant**

Divers FID sont réservés en fonction des applications visées. Il y a des normes par domaine d'applications

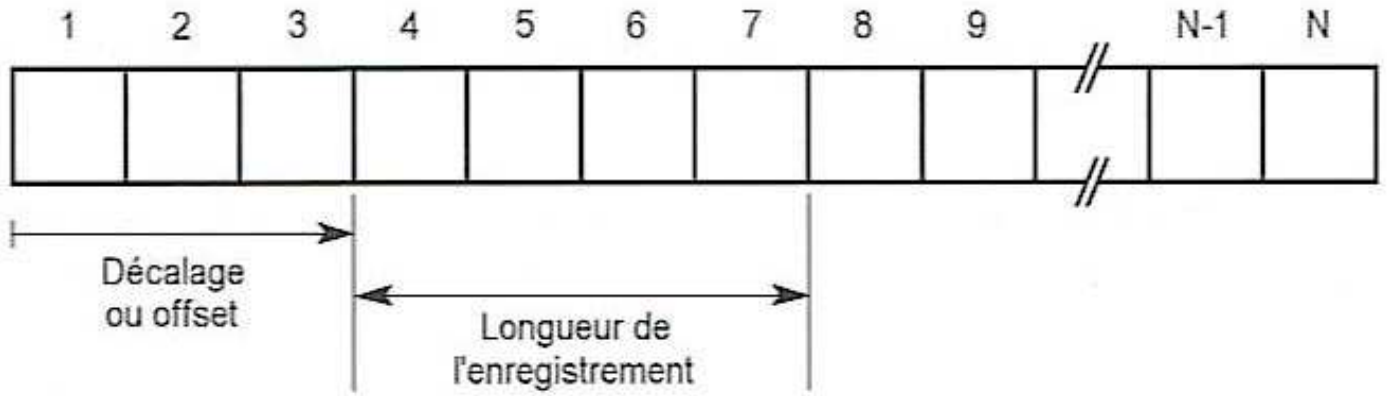
Structure des fichiers

- **Entête : contient le FID du fichier + autorisation d'accès**
- **Corps : données utiles du fichier**
- **Les entêtes sont stockées sur une page mémoire**
- **Les données sont hébergées sur une autre page**



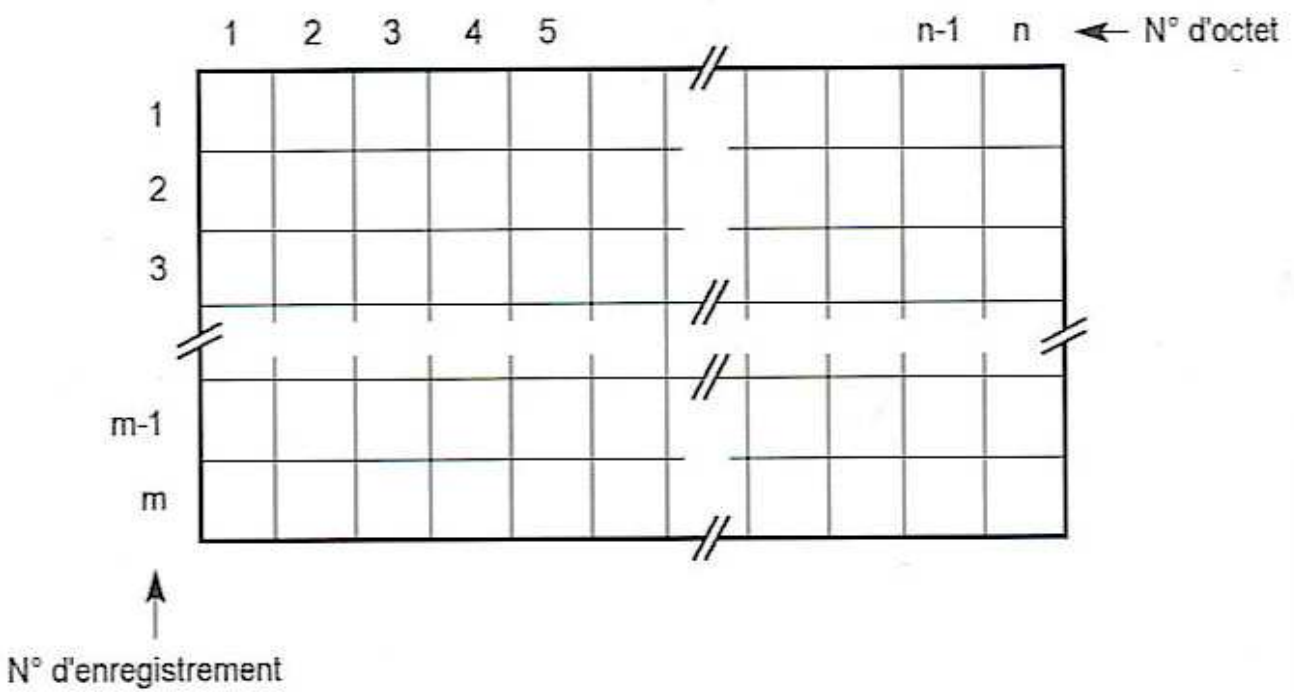
Fichiers à structure transparente

- **Fichier: suite d'octets permettant de stocker une faible quantité de données**
- **Taille min=1 octet, taille max: 255 octets**



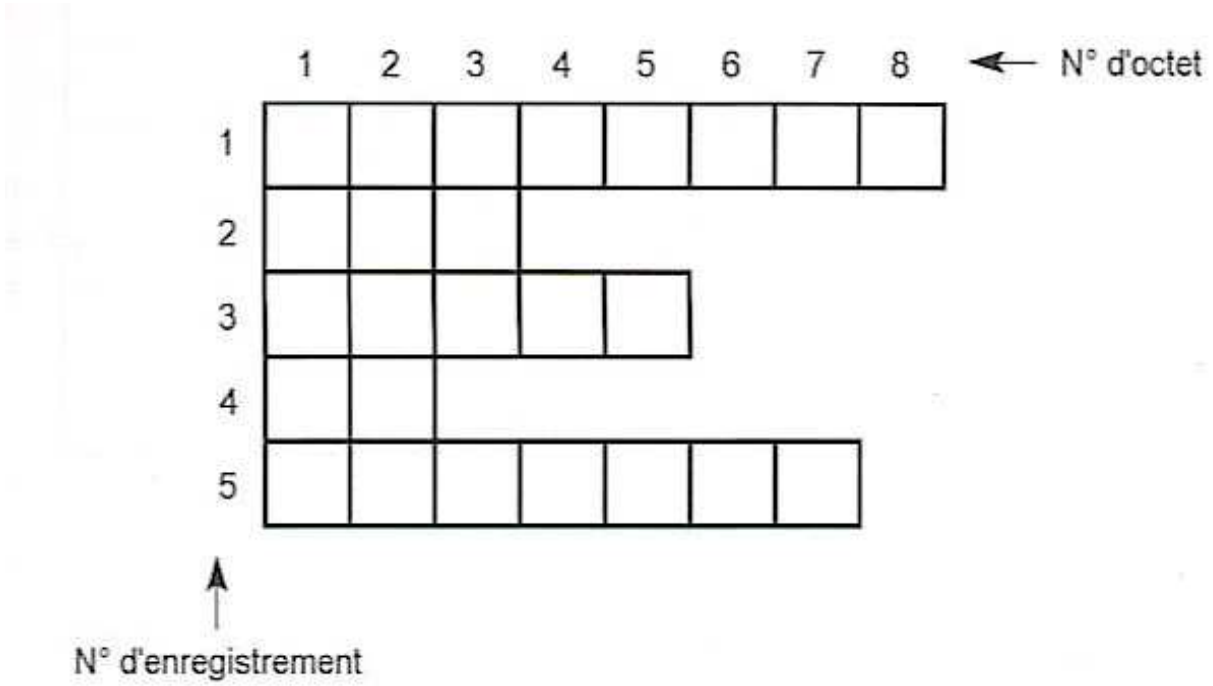
Fichiers à structure linéaire fixe

- **Fichier: suite d'enregistrements**
- **1 enregistrement : nb d'octets fixe**
- **L'accès à un enregistrement se fait sur le numéro qui doit toujours commencer à partir de 1**
- **Taille min=1 octet, taille max: 254 octets**



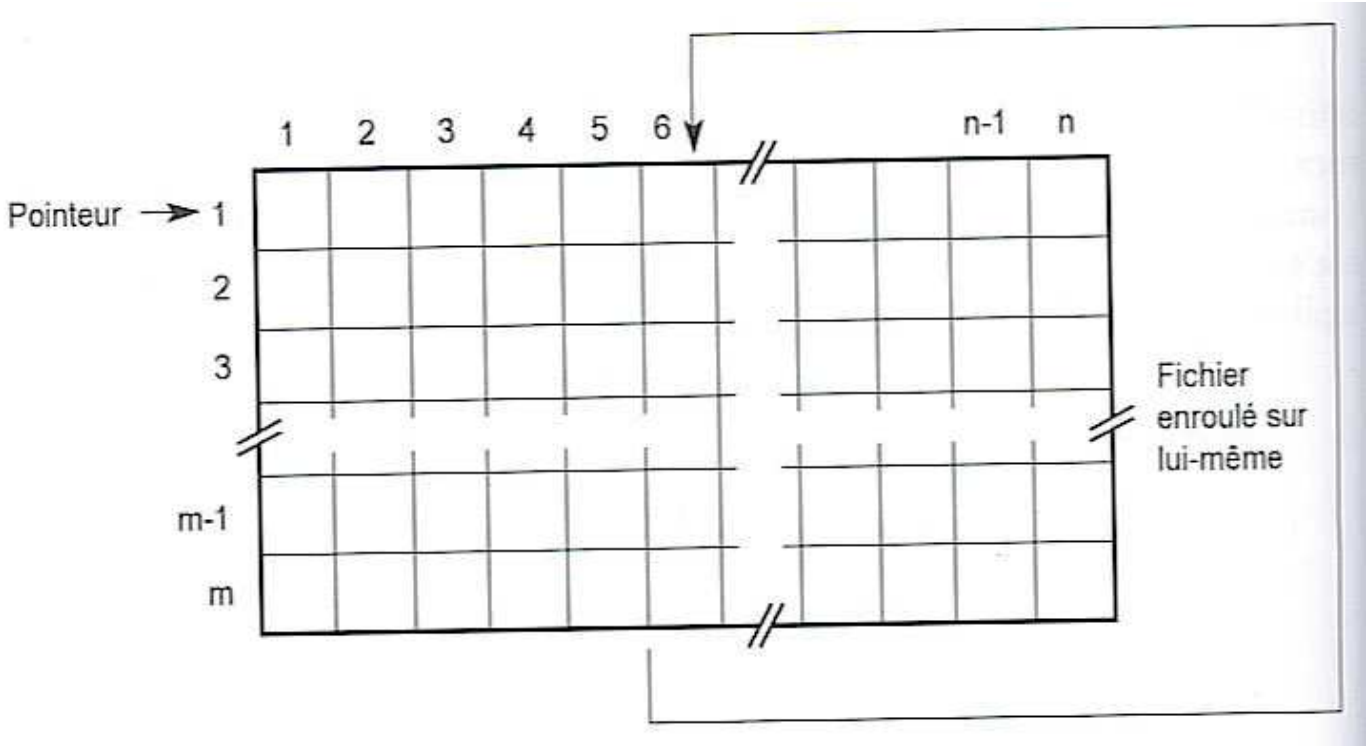
Fichiers à structure linéaire variable

- **Utilisée pour stocker des noms dans un répertoire téléphonique (carte SIM)**
- **Un enregistrement: un nb d'octets variable, l'indice commence à 1**
- **Taille min=1 octet, taille max: 254 octets**



Fichiers à structure linéaire cyclique

- C'est la structure linéaire fixe fermée
- L'indice 1 correspond toujours au dernier enregistrement écrit
- L'indice 2 correspond toujours à l'enregistrement écrit juste avant, etc.
- Taille max: 254 octets



Instructions normalisées et Messages d'erreur

Instructions normalisées

- **La norme ISO 7816-4 définit des commandes**
- **La présence de toutes ces instructions n'est pas obligatoire même si la carte est compatible à la norme**
- **Si ces commandes sont utilisées par une carte, elles doivent respecter la norme**
- **Une carte normalisée peut disposer d'instructions spécifiques (inexistantes dans la norme)**

Commandes de gestion de fichiers

Les commandes de gestion de fichiers

- **Binary (read, write, search, erase)**
- **Record (read, write, update, append, search, erase)**
- **Get/Put Data**

Commande SELECT

- **permet de sélectionner un fichier ou répertoire**

Commande APDU						
Entête obligatoire				**Corps optionnel**		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**
•CLA: dépend de l'application •INS: "A4" •P1 : mode de sélection (sélection par nom, par chemin d'accès, etc.) •P2: sélection d'un enregistrement dans un fichier •Lc: peut être absent •Data field: varie selon le mode de sélection						

Commande READ BINARY

➢ **permet de lire le contenu d'un fichier à structure transparente**
Ex: C0 B0 00 06 10

Commande APDU						
Entête obligatoire				**Corps optionnel**		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**
•CLA: dépend de l'application •INS: "B0" •P1 : Si MSB (b7)=1, les bits b0 à b2 = FID d'un fichier EF. P2 contient l'adresse de l'octet concerné Si b7=0, il s'agit du fichier EF déjà sélectionné alors (P1 P2) contiennent le décalage du premier octet à lire •Si Le=0, alors lecture des données comprises entre l'octet défini par P1 et la fin du fichier						

Commande WRITE BINARY

➢ **permet d'écrire dans un fichier à structure transparente (écriture, OU/ET logique avec les données du fichier)**
Ex: C0 D0 01 02 01 FF

Commande APDU						
Entête obligatoire				**Corps optionnel**		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**

- CLA: dépend de l'application
- INS: "D0"
- P1 : Si MSB (b7)=1, les bits b0 à b2 = FID d'un fichier EF. P2 contient l'adresse de l'octet concerné

Si b7=0, il s'agit du fichier EF déjà sélectionné alors (P1 P2) contiennent le décalage du premier octet à lire

- Lc: contient le nombre d'octets à écrire
- Data field : données à écrire (dont la taille = Lc)

Commande UPDATE BINARY

➢ **permet de mettre à jour le contenu d'un fichier à structure transparente (écriture normale)**
Ex: C0 D6 00 03 02 AA 55

Commande APDU						
Entête obligatoire				**Corps optionnel**		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**

•CLA: dépend de l'application
•INS: "D6"
•P1 : Si MSB (b7)=1, les bits b0 à b2 = FID d'un fichier EF. P2 contient l'adresse de l'octet concerné
Si b7=0, il s'agit du fichier EF déjà sélectionné alors (P1 P2) contiennent le décalage du premier octet à lire
•Lc: contient le nb d'octets à écrire dans le fichier
•Data field : données à écrire (dont la taille = Lc)

Commande ERASE BINARY

➢ **permet de supprimer les données (mise à 0) d'un fichier à structure transparente**

Ex: C0 0E 00 03 01 06

Commande APDU						
Entête obligatoire				**Corps optionnel**		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**

- CLA: dépend de l'application
- INS: "0E"
- P1 : Si MSB (b7)=1, les bits b0 à b2 = FID d'un fichier EF. P2 contient l'adresse de l'octet concerné

Si b7=0, il s'agit du fichier EF déjà sélectionné alors (P1 P2) contiennent le décalage du premier octet à lire

- Lc=0 effacement de l'octet désigné par (P1 P2) jusqu'à la fin du fichier
- Lc=01 ou 02
- Data field : contient le dernier octet à partir duquel on arrête l'effacement

Commande READ RECORD

➢ **permet de lire un ou plusieurs enregistrements d'un fichier à structure linéaire fixe, variable ou cyclique**
Ex: C0 B2 06 04 10

Commande APDU						
Entête obligatoire				**Corps optionnel**		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**

•CLA: dépend de l'application
•INS: "B2"
•P1 = 0, lire l'enregistrement en cours.
Si "01"<=P1<="FF" alors lire la valeur de P2
•P2 contient ($b_2b_1b_0$)=100 (lire numéro dicté par P1), =101 (lire de P1 jusqu'à la fin du fichier), =110 (lire de la fin du fichier jusqu'à P1), etc.

Commande WRITE RECORD

- **permet d'écrire dans un fichier à structure linéaire fixe, variable ou cyclique (écriture, OU/ET logique avec les données du fichier)**

Ex: C0 D2 06 04 0B 2E 54 61 76 65 72

Commande APDU						
Entête obligatoire				Corps optionnel		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**

- CLA: dépend de l'application
- INS: "D2"
- P1 = 0, écrire dans l'enregistrement en cours.

Si "01"<=P1<="FF" alors lire la valeur de P2

- P2 contient ($b_2b_1b_0$)=000 (premier enreg.), =001 (dernier enreg.), =010 enreg. Suivant, 011 (enreg. précédent), numéro d'enreg dans P1, etc.
- Data field : contient les données à écrire

Commande UPDATE RECORD

➢ **permet d'écrire dans un fichier à structure linéaire fixe, variable ou cyclique (écriture normale)**
Ex: C0 DC 00 00 05 44 75 6E 6F 64

Commande APDU						
Entête obligatoire				**Corps optionnel**		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**

•CLA: dépend de l'application
•INS: "DC"
•P1 = 0, écrire dans l'enregistrement en cours.
Si "01"<=P1<="FF" alors lire la valeur de P2
•P2 contient ($b_2b_1b_0$)=000 (premier enreg.), =001 (dernier enreg.), =010 enreg. Suivant, 011 (enreg. précédent), numéro d'enreg dans P1, etc.
•Data field : contient les données à écrire

Commande APPEND RECORD

➢ **permet d'ajouter un enreg dans un fichier à structure linéaire fixe ou variable, ou bien d'écrire le 1er enreg d'un fichier à structure linéaire cyclique**

Ex: C0 E2 00 00 05 63 61 72 74 65

Commande APDU						
Entête obligatoire				**Corps optionnel**		
CLA	**INS**	**P1**	**P2**	**Lc**	**Data field**	**Le**

- CLA: dépend de l'application
- INS: "E2"
- P1 =0
- P2 : contient le FID court du fichier EF concerné
- Lc: taille de l'enreg à écrire dans un fichier à structure linéaire variable
- Data field : données à écrire

Bibliographie

1. Technology for smart cards: architecture and programmer's guide, Zhiqun Chen, Addison Wesley, sept. 2000
2. Les Cartes à puce: théorie et mise en œuvre, Christian Tavernier, 2ème édition, Ed. Dunod, 2007.
3. Understanding Java Card 2.0, Zhiqun Chen & Rinaldo Di Giorgio
4. http://www.javaworld.com/javaworld/jw-03-1998/jw-03-javadev.html
5. http://javacardforum.org
6. Zhiqun Chen, "**How to write a Java Card applet: A developer's guide**", http://www.javaworld.com/javaworld/jw-07-1999/jw-07-javacard.html.
7. Pierre Paradinas, Support de cours sur « **la Carte à puce** » et « **Java Card** », UE de Systèmes Enfouis et Embarqués, Valeur C, Laboratoire CEDRIC, CNAM. http://deptinfo.cnam.fr/~paradinas/cours/
8. **Global Platform, Card Specification :** http://www.globalplatform.org/specificationform2.asp?id=archived
9. **API Java Card** : http://java.sun.com/products/javacard/htmldoc
10. Eric Vétillard : http://javacard.vetilles.com/2006/09/17/hello-world-smart-card/

Webographie

http://java.sun.com/products/javacard/
http://www.gemalto.com
http://www.oberthur.com
http://www.globalplatform.org
http://www.javacardforum.org
http://www.opencard.org
http://www.linuxnet.com/
http://www.iso.org
http://www.ttfn.net/techno/smartcards/
http://eurekaweb.free.fr/ih1-carte_a_puce.htm
http://membres.lycos.fr/dbon/historique.htm
http://apte.net/info-e/pubs.htm
http://www.eurosmart.com/index.htm