Bulletin of Aichi Univ. of Education, 71（Natural Science）, pp. 17 − 23, March, 2022

非接触型ICカードの情報科学教育への利用について

安本 太一

情報教育講座

Use of Contactless Smart Cards for Education in Computer Sciences

Taichi YASUMOTO

Department of Information Sciences, Aichi University of Education, Kariya 448-8542, Japan

1　はじめに

電子マネーなどで使われている非接触型ICカード［1］を，高校生や大学生が情報科学を学ぶ際の題材とすることを提案する。

高校生や大学生が，2進法表現，ASCIIなどの文字コード，エンディアン，ビット演算など（以下，情報の表現や演算などという）を授業で習っても，実感がわかず，真に理解するのは困難だと推測される。筆者が勤務している大学のネットワークの授業で，高等学校の情報の授業で程度の差はあれ一度は学んだことになっているはずなのに，IPアドレスの2進法表現や文字コードを扱っても課題や試験で誤答をする大学生が少なからずいるからである。その原因は，他大学の教員による文献［2］でも述べられているように，情報の表現や演算などの授業において高校生や大学生に与えられているのは，教員からの説明だけであり，実習など手を動かすことがないからだと思われる。

そこで，本稿では，社会で使われている電子マネーなどの非接触型ICカードに格納されている履歴データなどを読み取り復号して表示するプログラミング実習を通じて，情報の表現や演算などの理解を深めることを提案する。文献［2］のアセンブリ言語プログラミング体験の授業アンケートにおいて，前向きな記述（positiveな記述）が約9割に達し，コンピュータの原理がわかった，0と1で全てを表しているのがわかったなどのコメントが寄せられていることから，コンピュータのハードウェアに近いことを扱う非接触型ICカードのデータの復号も同様の効果が期待できると考えたからである。

高校生と大学生を対象としているのは，新学習指導要領［3］の実施時期の関係で，現在，高校でプログラミングを学ぶ者とそうでない者が混在しているからである。また，以下，簡略化のため，高校生と大学生を総称して受講生という。

2　背景

現在，非接触型ICカードやプログラミングについて，次のような状況にある。

1. 非接触型ICカードを用いた交通系や流通系などの電子マネーや身分証明書の普及

2. マイナポータルや健康保険証機能の追加といった政府や自治体による個人番号カード（マイナンバーカード）の利用機会増の取り組みに伴う非接触型IC カードリーダー（PC/SC 対応品）の一般への普及の進行

3. 初心者から熟練者まで広く使われているプログラミング言語Pythonの普及

4. 筆者の研究室で開発を継続しているPythonのPC/SC対応モジュールpcscの存在

5. 上記の電子マネーなどにより異なる履歴データなどの表現形式

PC/SC（Personal Computer/Smart Card）は，異なるメーカ間でもICカードやリーダー/ライタをコンピュータで相互利用できる標準規格である［4］。PC/SCを用いてプログラムを開発するために，SDK（ソフトウェア開発キット）を購入したり秘密保持契約を結ぶ必要はない。

1と2からは，電子マネーなどと非接触型ICカードリーダーの双方を所有している世帯が一定数あり，これからも増え続けていくことが推測される。2，3，4からは，プログラムから電子マネーなどの履歴データにアクセスすることが容易になってきていると判断できる。5からは，バリエーションが豊富ということで，

電子マネーなどを搭載している非接触型ICカードが情報科学教育の題材として魅力的であることがうかがえる。以下, 簡略化のため, このような非接触型ICカードをIC電子マネーなどということがある。

3　筆者の研究室で開発している Python の pcsc モジュール

筆者の研究室で開発しているPythonからPC/SC対応の非接触型ICカードリーダーを使用するためのモジュールをpcscと名付けている。このモジュールの仕様を付録Aに示す。

3.1　概要

pcscモジュールの開発は，2017年度卒業生の卒業研究と関連して始まり，現在も継続されている。

その1番目の特徴は，非接触型ICカードを扱うPythonのモジュールとして有名なnfcpyモジュール［5］に比べて，使用するのが簡単であることである。pcscモジュールは読み取りに特化しているなどnfcpyモジュールに比べて機能を大幅に絞っているが，pcscモジュールは各社の非接触型ICカードリーダーのPC/SCに対応させるためのドライバがOSにインストールされていれば，pcsc.pyをコピーしてすぐ使えることである。nfcpyの場合は，libusbというドライバが（Windowsの場合はwinusbも）必要であるが，Windowsにおけるlibusbのインストールはプログラミングを学び始めた頃のコンピュータの初心者には難しい。

2番目の特徴は, 今後出てくるあろう非接触型ICカードリーダーに対応できる可能性が高いことである。PC/SCは業界標準であり，macOS，Linux，Windowsにおいてサポートされている。今後発売される非接触型ICカードリーダーは，OSの PC/SCに対応するためのドライバが提供されると考えて差し支えない。pcscモジュールは，OSが提供するPC/SC のライブラリ（恐らくC言語で作成されたライブラリ）とPythonのctypesを使用して実装されているので，多くの会社の非接触型ICカードリーダーに対応できる可能性が高い。一方，nfcpyは，PC/SCが業界標準になる前の古い非接触型ICカードリーダーへの対応から始まったためか，PC/SCではなく，各社の非接触型ICカードリーダー固有のインタフェースを使っているようであり，拡張性は低い。実際のところ，2021年3月10日版のnfcpyのドキュメントには，ソニーのRC-S380より後に日本で家電量販店などで販売開始されている製品が対応機種として載っておらず，新しい非接触型ICカードリーダーには対応していない。

3.2　2021年度のpcscモジュールの機能拡張など

1. 新しいリーダーへの対応
   2020年7月に発売されたアイ・オー・データ機器のリーダー USB-NFC3ぴタッチ（AB Circle社のCIR215のOEM）に対応した。Felicaの汎用通信を行うためにコンピュータからリーダーに送信するAPDUのコマンドは，リーダーの製品によって多少異なる。pcsc モジュールは，非接触型ICカードがタッチされた時にリーダーの製品名を読み取り，その製品名の文字列によって分岐して，リーダーによって異なるAPDU のコマンドの内容を調節している。pcsc.pyはこのように実装されているので，新たなリーダー（製品名'Circle CIR215 PICC'）への対応は，短時間で行うことができた。実際に，販売終了になるリーダーがあり，その一方で新しいリーダーが発売されているので, 新たなリーダーへの対応が容易であることは重要である。
2. 複数のシステムを有している非接触型ICカードへの対応
   愛知教育大学の学生証・職員証は，愛知教育大学生活協同組合の電子マネーであるキャンパスペイも兼ねており，それぞれ，システムコードが異なる。これまで，pcscモジュールはデフォルトのシステムコード（学生証・職員証の方）しか扱うことができなかったが，今回，システムコードを指定して非接触型ICカードへポーリングすることによって，システムを明示的に指定する（システムを切り替える）機能を追加した。これにより，複数のシステムを有しているカードにも対応し，本学でいえば，学生証・職員証の学籍番号・職員番号などを読むことも，キャンパスペイの履歴なども読むこともできるようになった。
   カードへのポーリング後, そのカードのIDm（個々のカードに固有な書き換えができない製造番号のようなもの）も得られるようにし，非接触型ICカードを，IDカード（証明書）として利用することも容易になった。
3. Windows11のプレビュー版上での動作を確認
   本稿執筆時点での最新版Windows11Proバージョン 21H2ビルド22000.176上での動作を確認した。

4　IC 電子マネーなどにおける情報の表現と情報科学教育

表1 は，プログラミングの立場から，IC電子マネーなどにおける情報の表現形式の特徴をまとめたものである。表1を作成するにあたっては，文献［6，7］を参照したり，筆者が調査を行なった。IC電子マネーなどの表現形式の概要を付録Bから付録Hに示す。表1において，左側の列はデータの表現形式，右側の列

表 1：IC 電子マネーなどににおける情報の表現形式の特徴

	ビックエンディアン	リトルエンディアン	ASCIIコード	BCD	論理和	論理積	左シフト	右シフト
愛知教育大学 学生証・職員証 学籍番号・職員番号			○					
愛知教育大学 生活協同組合 キャンパスペイ履歴				○				
交通系ICカード 履歴	○	○			○	○	○	○
Edy nanaco WAON 会員番号 履歴 残額	○(履歴)	○(残額)		○(会員番号)	○(履歴)	○(履歴)	○(履歴)	○(履歴)
JAL ICカード 会員番号 会員名			○(会員名)	○(会員番号)				

は格納されている内容を人間可読（ヒューマンリーダブル）にするために必要なビット演算である。愛知教育大学学生証･職員証, 愛知教育大学生活協同組合キャンパスペイ，交通系ICカード（Suica, TOICA, manacaなど），Edy，nanaco，WAON，JAL ICカードの計7つというように多くの種類のIC電子マネーなどを扱えば，主要なデータの表現形式やビット演算を網羅できることがわかる。

コンピュータやプログラミングの授業では，社会に実際に存在するものを題材にする方が，授業内容の意義を受講生に理解してもらえると考えられるので，IC電子マネーなどの利用履歴などをプログラムで読み取ることを授業で行うことの提案に至った。

5　pcsc モジュールを用いた電子マネーなどの情報の読みとり

pcscモジュールの使用例として，交通系ICカードの履歴の最も新しいもの，すなわち，サービスコードの最初（0番目）のブロック（16バイト）を，

080200002af4a5560000c71c0007b740

のように読み取って，

履歴連番:**1975**　日付:**2021**年**7**月**20**日 残額:**7367**円

のように表示するプログラムの概要は図1のようになる。図1は説明のためのものであり，非接触型ICカードリーダーに接続し，交通系ICカードをタッチするまで待機するなどの部分を省いてあり，ここに掲載されている分だけでは動作しない。

clfは非接触型ICカードに対応するオブジェクトのインスタンスであり，そのメソッド**read_wo_encryption(sc, bc)**でサービス**sc**の中の**bc**番目のブロック（16バイト）を読み取り，**block_data**というバイト列に格納している。付録Dに示す履歴連番，日付，残額のフィールドの定義から，各フィールドを取り出して可読な形に変換する計算をし，画面表示している。データの表現形式やビット演算に加えて，配列や文字列も扱っており，プログラミングにある程度慣れた段階での題材として，適当と思われる。

例えば，このプログラムの

```
seq = '履歴連番:'+str(block_data[13]*256 \
                    + block_data[14]) + ' '
```

の部分を

```
import pcsc

SUICA_SYS = 0x0003
SUICA_SVC = 0x090f

# システムコードを指定してポーリング
sys_code = clf.make_systemcode(SUICA_SYS)
result = clf.polling(sys_code)

# サービスコードを示すデータの作成
sc = clf.make_servicecode(SUICA_SVC)

# 最初（0 番目）のブロックを示すデータの作成
no =  0
bc = clf.make_blockcode(no)

# 1 ブロック分のデータを得る
#bolck_data は，バイト列（1 つの要素が 8 ビットの整数の
# 配列）である．block_data[ 番目 ] のように参照できる．
block_data = clf.read_wo_encryption(sc, bc)

# 可読な形にして表示
year = str((block_data[4] >> 1) + 2000)+'年'
month = str((block_data[4] & 0x01) << 3 \
            | block_data[5] >> 5)+'月'
day = str(block_data[5] & 0x1f)+'日'
ymd ='日付:' + year + month + day + ' '
seq = '履歴連番:'+str(block_data[13]*256 \
                    + block_data[14]) + ' '
balance = '残額:'\
          + str(block_data[10] \
              + block_data[11] * 256)+'円'
print(seq + ymd + balance)
```

図 1：pcsc モジュールの使用例の概略

```
seq = '履歴連番:'+str(block_data[13]<<8 \
                    | block_data[14]) + ' '
```

のように記述することもでき，2進数表現やコンピュータのハードウェアの理解を深めるのに役立つ。また，

```
seq = '履歴連番:'+str(block_data[13]<<8 \
                    + block_data[14]) + ' '
```

のような意図しない実行結果を招く誤りの例は，*，+，<<，|の優先順位の関係を題材にして，演算子の優先順位を理解することの重要性を説明するのに有用である。

6　授業における展開

非接触型ICカードの内容の読み取りにおいて，交通系ICカードの例を1つ示すだけでは情報の表現や演算などが受講者に定着するには十分ではないので，他のIC電子マネーなども扱う必要がある。交通系ICカードでは使用されていないデータ表現の形式（ASCIIコードやBCD）もあるので，これらのデータ表現の形式が使われている理由に触れる機会も得られる。受講生には，練習問題として，他のIC電子マネーなどの内容の読み取りに取り組んでもらうのが良い。しかしながら，受講生が，表1のIC電子マネーなどの全てを持っているわけではないので，図2のように，ブロックの内容に対応する16進数のリテラル表記のバイト列とそれを可読形式にしたものを対で示し，リテラル表記を解釈して可読形式に変換して画面表示するプログラムの作成を練習問題とするのである。作成したプログラムが正しいか否かは，画面表示を見れば一目瞭然なので，受講生にとって取り組みやすくかつ達成感のある課題であろう。16進数のリテラル表記のバイト列を手で入力することは現実的ではないので，練習問題をホームページに載せるなどして，受講生にはリテラル表記のバイト列をコピーアンドペーストしてもらうことが考えられる。

図1では1つのブロックに履歴連番，日付，（履歴の中の）残額という複数の項目が含まれる例を扱ったが，これが受講生に最初の例として難しいということであれば，1つのブロックの冒頭から1つの項目だけが含まれているもの例えば（履歴とは独立した最終の）残額の例から始めることが考えられる。

交通系ICカードの履歴など1つのブロックだけで完結するものと比べると，WAONの履歴のように複数のブロックを読まないと履歴が得られないのは難易度が高い。交通系ICカード，nanaco，WAONのように，日付を構成する年，月，日が分離されて格納されているものと比べると，Edyのように日付がある起点からの経過日数で表現で格納されているのも難易度が高い

```
以下は，Edy の履歴 5 件分のブロックである．
b0 = b'\x20\x00\x00\x05\x3b\x73\x28\x9f'\
     b'\x00\x00\x00\x82\x00\x00\x08\x6c'
b1 = b'\x04\x00\x00\x04\x3b\x33\x32\x8e'\
     b'\x00\x00\x03\xe8\x00\x00\x08\xee'
b2 = b'\x04\x00\x00\x03\x3a\x61\x27\xdb'\
     b'\x00\x00\x03\xe8\x00\x00\x05\x06'
b3 = b'\x20\x00\x00\x02\x3a\x14\xdb\x8f'\
     b'\x00\x00\x02\xca\x00\x00\x01\x1e'
b4 = b'\x04\x00\x00\x01\x38\xb5\x2b\x43'\
     b'\x00\x00\x03\xe8\x00\x00\x03\xe8'
blist = [b0,b1,b2,b3,b4]

次のように画面表示するプログラムを作成しなさい．
5 回目 2020-10-31 21:05:35 支払      処理額：  130 円 残額： 2156 円
4 回目 2020-09-29 21:47:58 ギフト    処理額： 1000 円 残額： 2286 円
3 回目 2020-06-16 21:02:19 ギフト    処理額： 1000 円 残額： 1286 円
2 回目 2020-05-09 15:36:47 支払      処理額：  714 円 残額：  286 円
1 回目 2019-11-15 21:16:51 ギフト    処理額： 1000 円 残額： 1000 円
```

図 2：練習問題の例

（時刻も同様である）。経過日数から年月日を求める（経過秒数から時分秒を求める）コードを受講生に考えさせることに加えて，プログラミング言語におけるいわゆる日付型や時間型とその演算の存在を受講生に紹介することが考えられる。Pytyonでは，**datetime.datetime**や**datetime.timedelta**が利用できる。

表1に掲げるように多くの複数のIC電子マネーなどにおけるデータ表現について一通り触れた後，各電子マネーなどにおいて，どうしてそのようなデータ表現になっているか考察することは，受講生の情報科学のセンスを涵養するのに有用である。受講生自身が，将来，自ら情報システムを設計する時に，この考察で得たことを生かせれば，的外れなことはせず，良いものを作る方に時間を割くことができることが期待できる。

7　まとめ

非接触型ICカードの情報科学教育への利用について，提案した。今日では，メモリが潤沢な環境で高級プログラミング言語を使って，プログラミングを学ぶことが普通になっている。Python3では整数は無限長整数であり，オーバフローを経験することもない。その一方で，非接触型ICカードを扱う場合は，デバイスの価格を安価にするため制限された少ないデータ領域を有効に使用せざるを得ない制約があり，データ表現を考えさせられる機会が得られる。今日の受講生に，昔のワンボードマイコンの話をしても別世界の話になってしまうだろうが，今日身の回りで使われているデバイスならばそのような心配はない。

非接触型ICカードのIDmは本人確認のIDとしても使うことができ，今回紹介したpcscモジュールと併

せてPythonの豊富なモジュールを使えば，勤怠管理（出席管理）システム，電子錠システム，児童生徒が登下校時に非接触型ICカードをリーダーにタッチしたら保護者にメールを送るようなシステムなどを作ることは難しくない。受講者のさらに発展的に学びたいというきっかけになる可能性がある。

今後の課題は，3つある。1つ目は，本稿で触れていない他のIC電子マネーなどの表現形式をさらに収集し，練習問題を多様化するとともに，IC電子マネーなどの表現形式について考察するときの材料を増やすことである。

2つ目は，メモリ効率や実行効率が明確に悪くなるような，ずさんなデータ表現を用意し提示して，受講生にデータ表現の設計の重要さを理解してもらうことである。

3つ目としては，高等学校や大学で機会を得て，非接触型ICカードの内容を読み取ることを扱うプログラミングの授業を実際に行い，その評価を行うことがあげられる。

参考文献

［1］長谷川晴彦：PCから読み取る非接触型ICカード，オープンソースマガジン，7月号，pp.64-78（2006）.

［2］久野靖ほか：大学1年次コンピュータリテラシ科目でのアセンブリ言語プログラミング体験，情報処理学会情報教育シンポジウム SSS2017 報告集，pp.255-262（2017）.

［3］文部科学省：高等学校学習指導要領（平成30年告示），（2018）.

［4］PC/SC Workgroup, https://pcscworkgroup.com（2021.9.24 閲覧）.

［5］Tiedemann, S: nfcpy documentaion Release 1.0.3, https://nfcpy.readthedocs.io/（2021.9.24 閲覧）.

［6］ICカードのフォーマット解析，http://jennychan.web.fc2.com/format/（2021.9.24 閲覧）.

［7］felicalibプロジェクトWiki，https://ja.osdn.net/projects/felicalib/wiki/FrontPage（2021.9.24 閲覧）.

付録 A　pcsc モジュールの仕様

- 対応しているOS
  macOS（M1は未確認），Linux（Intel, ARM），Windows（11のプレビュー版と10）。検証可能な環境があるものについては32ビット版と64ビット版の双方で確認。
- 対応している非接触型ICカードリーダー
  ソニー RC-S380（Windowsのみ），NTTコミュニケーションズ ACR1251CL-NTTCom，ACS ACR122U，ACS ACR1255U-J1，アイ・オー・データ機器USB-NFC3ぴタッチ（AB Circle CIR215の OEM）。
- 必要なソフトウェア
  使用する非接触型ICカードリーダーに対応するPC/SCデイバイスドライバ。OSに含まれていたり，接続時に自動ダウンロードされる場合は，明示的なインストールは不要。RC-S380がWindowsのみに対応しているのは，ソニーが一般向けにWindows 版のみPC/SCデバイスドライバを提供しているため。

付録 B　愛教育大学学生証・職員証の身分証明書

愛知教育大学の構成員以外が知ることは差し障りがあるかもしれないので，システムコードとサービスコードは伏せる。所有者である構成員はダンプすれば容易にわかる。

- システムコード：**XXXX**（**FE00**ではない）
- サービスコード：**YYYY**
- 0ブロック目
  0バイト目から，職員番号や学籍番号を格納。ASCIIコードによる表現で1桁1バイト。番号が終わると残りはNULL（0）。
- 3ブロック目
  0バイト目～：有効期間の始まり。8バイト目～：有効期限の終わり。ともに，**YYYYMMDD**の形式で，ASCIIコードによる表現で1桁1バイト。

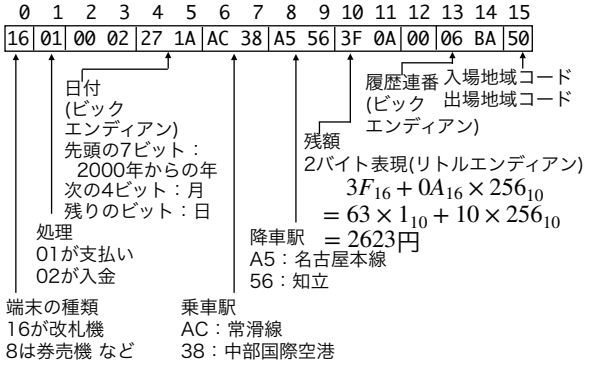
付録 C　愛教育大学学生証・職員証の大学生協キャンパスペイ

他の大学生協にも，同様のものがあるので記載する。

- システムコード：**FE00**
- サービスコード：**50CF**
- 1ブロックが1履歴で，10件の履歴が記録されている。
  0～6バイト目：**YYYYMMDDhhmmss**の形式で日付。
  7バイト目：処理で01_{16}が支払い05_{16}がチャージ。
  8～10バイト目：処理金額。11～13バイト目：残額。残りは未使用。日付，処理金額，残額は，BCDで，4ビットで1桁（1バイトで2桁）。

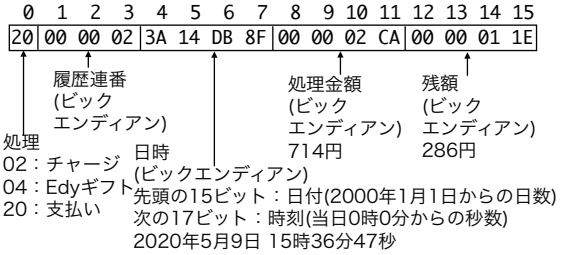
付録 D　交通系 IC カード

- システムコード：**090F**
- サービスコード：**0003**
- 1ブロックが1履歴で，0ブロック目から20件の履歴が記録されている。例で示す。



付録 E　Edy

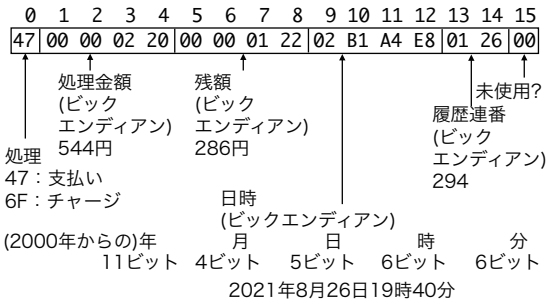
- システムコード：**FE00**
- Edy 番号（会員番号）
  - サービスコード：**110B**
  - 0ブロック目
    2バイト目から16桁のEdy番号を格納。BCDで，4ビットで1桁（1バイトで2桁）。
- 履歴
  - サービスコード：**170F**
  - 1ブロックが1履歴で，0ブロック目から6件の履歴が記録されている。例で示す。



- 残額
  - サービスコード：**1317**
  - 0ブロック目
    0 ～ 3バイト目：32ビットの残額をリトルエンディアンで格納。

付録 F　nanaco

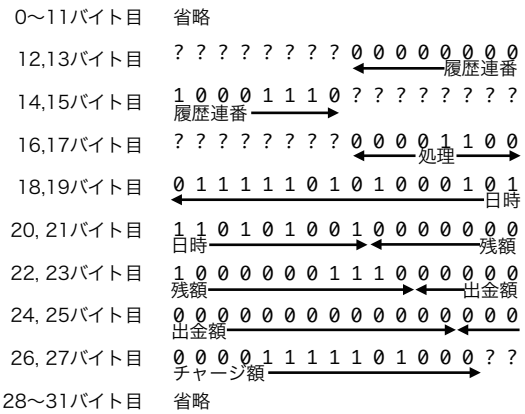
- システムコード：**FE00**
- nanaco 番号（会員番号）
  - サービスコード：**558B**
  - 0ブロック目
    0バイト目から16桁のnanaco番号を格納。BCDで，4ビットで1桁（1バイトで2桁）。
- 履歴
  - サービスコード：**564F**
  - 1ブロックが1履歴で，0ブロック目から5件の履歴が記録されている。例で示す。



- 残額
  - サービスコード：**5597**
  - 0ブロック目
    0 ～ 3バイト目：32ビットの残額をリトルエンディアンで格納。

付録 G　WAON

- システムコード：**FE00**
- WAON 番号（会員番号）
  - サービスコード：**67CF**
  - 0ブロック目と1ブロック目
    0ブロック目の12 ～ 15バイト目に16桁のWAON番号始めの8桁を格納。1ブロック目の0 ～ 3バイト目に残り8桁を格納。BCDで，4ビットで1桁（1バイトで2桁）。
- 履歴
  - サービスコード：**680B**
  - 2ブロック（32バイト）が1履歴で，各項目がバイトの境界で区切れていない。0ブロック目から3件の履歴が格納されている。例で示す。



履歴連番（ビックエンディアン），
処理（04_{16}：支払い，$0C_{16}$：チャージ），
日時（順に，年：2005年からの年で5ビット，月：4ビット，日：5ビット，時：5ビット，分：6ビット，ビックエンディアン），
残額，出金額，チャージ額はビックエンディアン。
本例では，142回，チャージ，2020年10月17日14時41分，1038円（残額），0円（出金額），

1000円（チャージ額）。

- 残額
  - サービスコード：**6817**
  - 0ブロック目
    0 ～ 1バイト目：16ビットの残額をリトルエンディアンで格納。

付録 H　JAL IC カード

- システムコード：**FE00**
- サービスコード：**2F4B**
- 0 ～ 2ブロック目にJMBお客様番号（会員番号）と氏名を格納。
  0ブロックの11 ～ 15バイト目に9桁のJMBお客様番号を，1バイト2桁（BCDで4ビット）ずつ，15バイト目から11バイト目の順（リトルエンディアンのような形）で格納。11バイト目の上位4ビットはお客様番号の最後の桁，下位4ビットは1。7バイト目から10バイト目は，順に，1016，1316，3216，0116が格納されている。
  1ブロック目の0 ～ 15バイト目と2ブロック目の6 ～ 15バイト目に英字表記（ローマ字表記）の氏名をASCIIコードで格納。氏名を逆順にしたものを，1ブロック目の15バイト目から格納し，1ブロック目の0バイト目に達したら，2ブロック目の15バイト目から6バイト目に向かって格納。氏名の格納が終わったら，残りは空白（20_{16}）で埋める。

（2021年9月24日受理）