

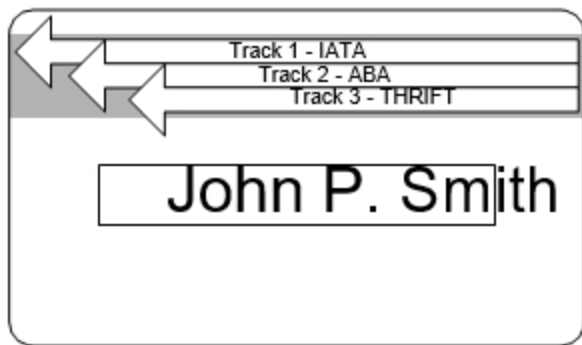
はじめに

時折、使いやすく堅牢で、多くの用途に適応する技術が登場します。磁気データストライプはまさにそのような技術の一つです。最も顕著な例はクレジットカードですが、他にも航空券、ショッピングクラブカード、ビデオレンタル店のIDカード、運転免許証、自動販売機のデビットカードなど、その用途は多岐にわたります。

本アプリケーションノートでは、Okika Devices社のフィールドプログラマブルアナログアレイ（FPAA）を用いて完全な磁気ストライプ読み取り増幅器／デコーダを構築する方法、およびホストプロセッサへの容易なインターフェース方法を簡潔に説明します。また、この種のアプリケーションにおけるプログラマブルアナログと固定機能の利点についても解説します。

標準カード

「標準の素晴らしい点は、選択肢が非常に多いことだ」。まさにその通りである。磁気ストライプ技術に関して議論する際、選択可能な標準は数多く存在するが、ここではそれらを全て列挙することはない。最も頻繁に参照される仕様はISO/IEC-7811であると言っておこう。ここでは、プラスチックカード用磁気ストライプ読み取りアプリケーション向けの読み取りヘッドとハウジングを使用していることを前提に、その要点をまとめる。こうした製品には多くの優良サプライヤーが存在するが、本稿ではMagtek製品を採用した。



この説明において、左図は一般的なプラスチック製クレジットカードの裏面を表現しています。図示のように磁気ストライプが上部にある場合、最初の（上部の）トラックは国際航空運送協会（IATA）が定めた形式でエンコードされており、通常はカード名義人の名前が記録されています。

第2トラック（これが実際の主力トラック）は米国銀行協会（ABA）が定義したフォーマットで、クレジットカード番号が記録されています。第3トラックのフォーマットはTHRIFTと呼ばれ、元々は現金自動預け払い機（ATM）での使用を想定していました。読み取り専用である第1・第2トラックとは異なり、THRIFTトラックは読み書き両方のアプリケーションを目的としていました。しかし実際には普及せず、データの書き戻しが設計要件に含まれるアプリケーション以外ではほとんど使用されません。典型的な例としては、複写機や自動販売機のデビットカードが挙げられます。

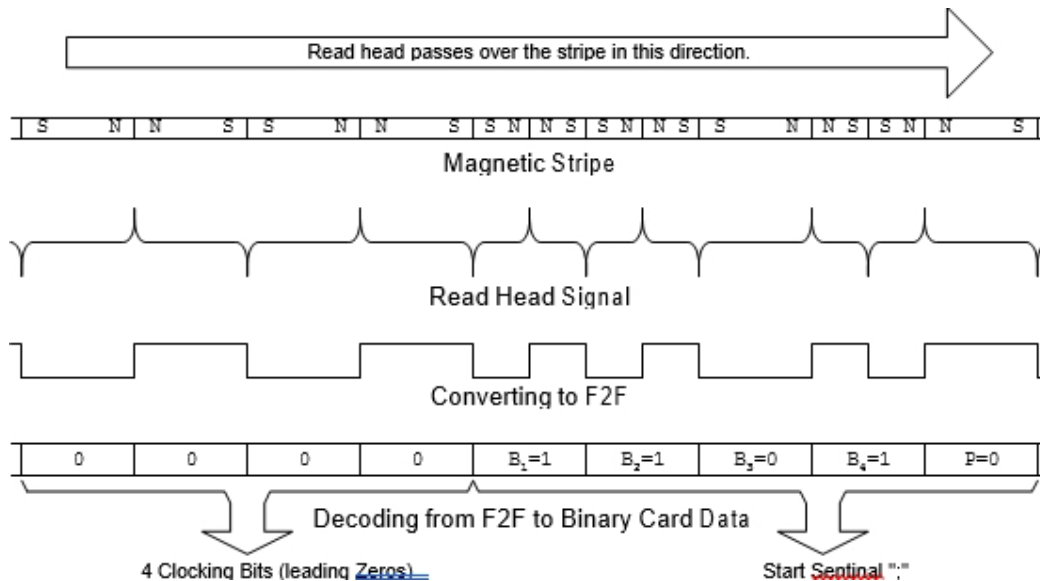
書き込みアプリケーションを想定していました。しかし実際には普及せず、データの書き込みが設計要件となる特定の用途（複写機や自動販売機のデビットカードなど）を除き、ほとんど使用されていません。

上記のカードを見ると、データは右から左へエンコードされています。エンコードは常に最下位ビットから始まります（右端のビットは、ストライプにおける最初の文字の最下位ビットです）。各トラックの定義は異なり、以下の表にまとめられています。

	Bits per Character	Bits Per Inch
Track 1 - IATA	Alpha - 6 Data + 1 Parity (odd)	210
Track 2 - ABA	BCD - 4 Data + 1 Parity (odd)	75
Track 3 - Thrift	BCD - 4 Data + 1 Parity (odd)	210

磁気ストライプからASCIIデータへの変換...

磁気読み取りヘッドの信号を、ほとんどのプロセッサが処理可能なASCIIデータに変換するには、いくつかのステップが必要です。以下の図は、上から下へ進むにつれてこれらの変換ステップを示しています。



磁気ストライプ

変換プロセスを理解するには、カード上のデータ保存方法を把握することから始めます。長い棒磁石の連鎖を想像してください。一部は1単位の長さ（常にペアで存在）で、残りはすべて2単位の長さです。さらに、これらの棒磁石はそれぞれ、本来の状態とは逆の方向に配置されています。つまり、北極は常に北極に、南極は常に南極に接触しています。このような同極が接する領域には、磁束線が集中しています。

読み取りヘッド信号

磁気ストライプが読み取りヘッドの前をスワイプされる。磁束密度が高い領域が通過するたびに電流が誘導される。N-N領域では一方に電流が誘導され、S-S領域では反対方向に電流が誘導される。結果として、読み取りヘッド信号の極性は重要ではない。F2F波形への変換に関連する要素は、遷移の時間間隔のみである。

F2Fまたはエイケン・バイフェーズ

現時点では、読み取りヘッド信号からF2F波形を生成する方法の詳細な議論は後回しにし、F2F符号化そのものに焦点を当てます。エイケン二相 (Aiken Biphase) ととも呼ばれるF2F符号化は、磁束遷移間の間隔を用いてビットを符号化します。2ユニットの棒磁石がゼロを、1ユニットの棒磁石のペアがワンを表します。

言い換えれば、各ビットはストライプ上で物理的に同じ長さを占有します。その長さの途中に「余分な」磁束遷移を持つビットが1と呼ばれます。繰り返しますが、これらの遷移の極性は任意であり、遷移間の相対的な間隔が1または0を示すのです。

カードデータからASCIIへの変換

各トラックは「クロッキングビット」で始まり、終わる。クロッキングビットとは単なるゼロの列に過ぎない。実際、手動でのカードスワイプはほぼ一定の速度で行われる。したがって、スワイプ開始時のビット周期はスワイプ終了時のビット周期と非常に近い。「クロッキングビット」は長さ（ビット周期）が解析され、そのスワイプのペースを設定する。磁束遷移間の経過時間がビット周期に近い場合、0と認識される。磁束遷移間の経過時間が半ビット周期に近い場合、1と認識される。

タイマー周辺機器を備えたプロセッサの場合、FPAA出力（カードスワイプ中に0Vと3V間でトグルする）をタイマーポートの入力ピンに接続できる。各遷移で割り込みが発生し、割り込みサービスルーチンでタイマー値を配列に格納するだけでよい。値の配列に対する簡単な後処理で、1と0を容易に判別できる。

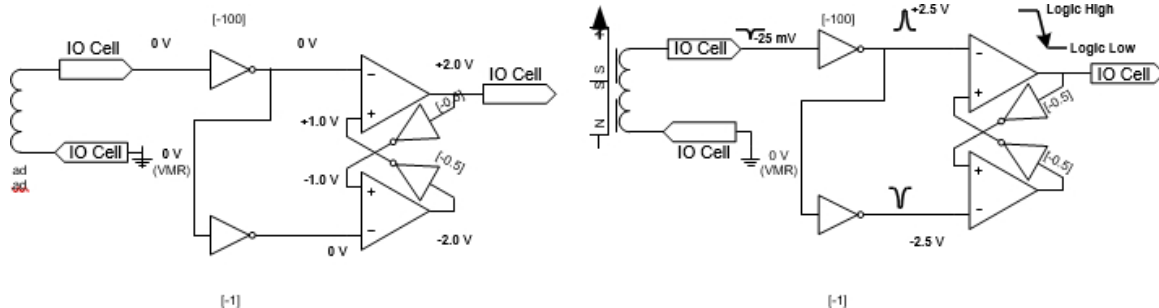
低コストプロセッサの場合、代わりにFPAA出力をプロセッサの汎用入力ピンに接続する必要があります。最初の遷移が検出されると（通常は割り込みとして）、プロセッサは時間を追跡し、入力ピンを定期的にポーリングし、遷移が発生した時刻を記録することに完全に専念しなければなりません。ここでも、タイムスタンプの配列を後処理することで、1と0を判別できます。

カードが逆方向にスワイプされた可能性があることに留意してください。データの反転処理はソフトウェアで容易に処理できます。

1と0の文字列が得られたら、データの開始位置を特定し、解釈方法を知る必要があります。この処理に関する概要については、本ノートの付録を参照してください。

AN231E04を磁気読み取りヘッド増幅器およびF2F発生器として適用

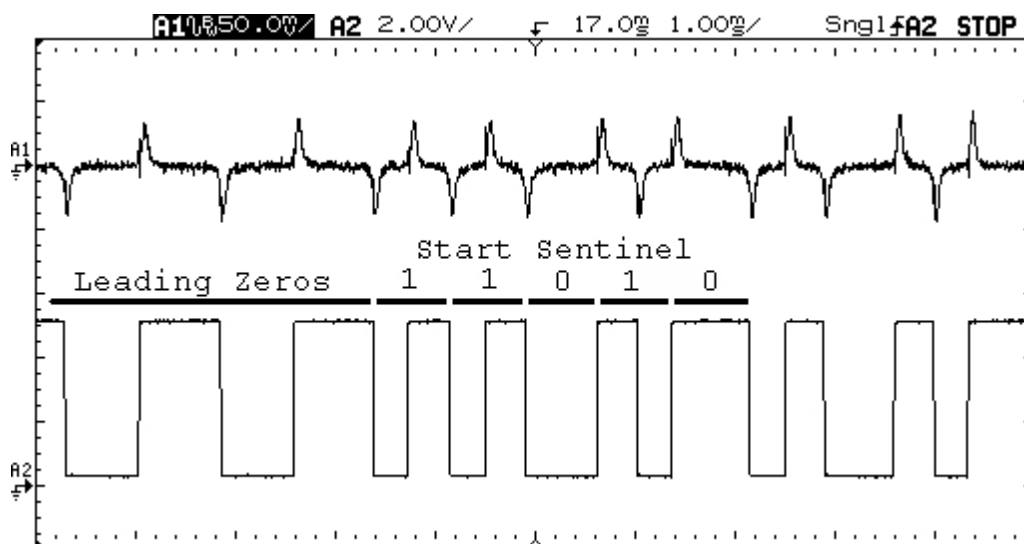
必要なのはゲイン反転回路とコンパレータCAMのみです。ここで示す回路は一種の「アナログS-Rフリップフロップ」です。コンパレータとそのフィードバック経路が双安定回路を形成します。この回路は2つの状態でのみ安定します。



左の回路図を参照。電圧は太字で示され、増幅器の利得は角括弧で表示されている。増幅器の出力には泡マークが付いており、これらが全て反転増幅器であることを視覚的に示すための記号である。カードが存在せず、全てのノード電圧が図示通りであると仮定する。（補足：FPAA内の全てのアナログ信号処理は差動方式であるため、上記図において+2Vの電圧はOxP - OxN = +2Vを意味する。さらに、コモンモード電圧は電圧中間レール（VMR）であり、慣例により0Vと表記される。VMRは実際にはチップのグラウンドより1.5V高い...したがって差動信号+2VはOxP = +2.5V、OxN = +0.5V）となります）。

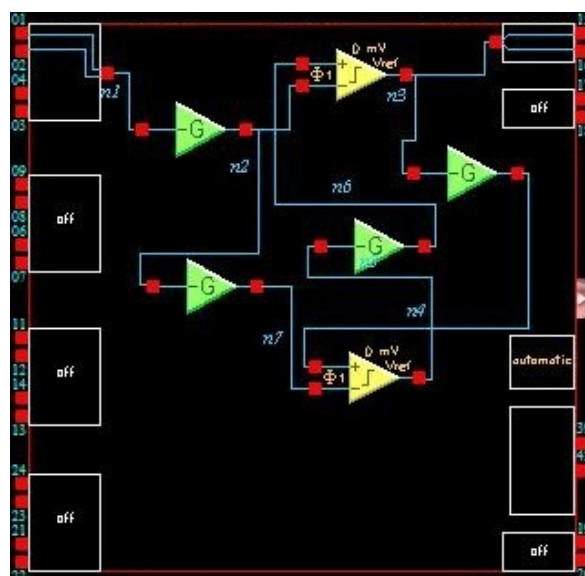
ノード電圧をしばらく観察すると、回路が安定した状態にあることがわかります。出力は論理ハイです。

では、カードが読み取りヘッドを通過した際に何が起こるかを考えてみましょう。右上の例では、S-S磁束が読み取りヘッドのギャップを通過しました。これにより、負方向の小さな信号スパイクが誘導されました。高利得入力段はこの微小スパイクをフルレール信号に変換します（振幅が25mVを超える入力信号は、図示のようにクリップされた信号となります）。この正パルスが上部コンパレータの負極に印加されると、出力波形の下降が示すように状態が変化します。下部のコンパレータでは逆の現象が発生し、回路全体が別の安定状態に移行します。出力は論理ローとなります。



上図は動作中の回路のオシロスコープ画像である。この特定のケースでは、読み取りヘッドがAN231E04に直接接続されている。上側のトレースはノイズを含む生の読み取りヘッド入力を示し、下側のトレースは回路の出力信号（0～3VのF2Fロジック波形）を示している。他の部品は一切使用されていない。これだけの話だ。プログラマブルアナログなら、これほど簡単なのである。

上記の回路はAN231E04内の4つのCABのうち2つしか消費していないため、AN231E04には2つ目の読み取りチャネルを追加する十分な余裕があります。下図はAnadigmDesigner2ソフトウェアでの回路実装例です（この回路内の全てのCAMクロックは4MHzであることに注意）。



プログラマブルアナログの利点

ここで説明する回路は十分に単純であり、もちろん20個ほどの標準部品で構成可能です。ではなぜプログラマブルアナログなのか？

明らかな答えは部品点数です。この応用例では一切の追加部品を使用していません。読み取りヘッドはFPAAに直接接続されました。しかし、真の利点は回路を実験室から取り出し、現実世界の過酷な環境に晒すまで明らかになりません。

現実世界では、プログラマブルなアナログ回路なしで構築されたクレジットカードの磁気ストライプリーダーは、本来あるべき姿とは程遠い。この事実を何度も目にしてきたため、おそらく感覚が麻痺しているでしょう。少し時間を取って、これまで目撃したレジ係の「スワイプ」テクニックを思い出してみてください。「ゆっくり慎重に」スワイプする手法。「素早くスワイプ」する手法。「逆方向にスワイプ」する手法。「前後に何度もスワイプ」する手法（個人的に特に気に入っている手法です）。「カードを拭いて再スワイプ」。「カードをビニール袋に包んで再スワイプ」。「別のカードリーダーを試す」テクニック。そしてもちろん究極の緊急対策として、レジ係がカードを光にかざして摩耗した凸字番号を認識し、手入力する（大抵は嫌そうな表情をあなたと共有しながら）。

汚れ、摩耗、温度、意図しない磁界への曝露が相まって、磁気ストライプカードの読み取りを困難にします。同様に、汚れ、摩耗、腐食は読み取りヘッドの効率を次第に低下させる。ではプログラマブルアナログ技術でどう解決するか？ホストプロセッサはわずか数マイクロ秒で回路パラメータを更新可能。あるいは100マイクロ秒未満でAN231E04に全く新しい回路構成をダウンロードできる。スワイプ失敗時、入力増幅器の利得を瞬時に調整できるのだ。

また、磁気ストライプの保磁力（コエリシビティ）が異なるカードを読み取るための回路調整も可能です。通常、読み取りヘッドでは重要ではありませんが、書き込みヘッドの駆動信号を調整することは、こうした保磁力差に対応するために不可欠です。

プログラマブルアナログを標準カードアプリケーションに適用する利点は以下の通りです。カスタムアプリケーションでは、プログラマブルアナログを使用する利点がさらに増えます。高セキュリティアプリケーションでは、FPAAをトーン生成や復号を含む様々なデータ符号化技術に対応するよう設定可能です。読み取り・書き込み機能など。

付録 - クレジットカードトラックの符号化とデータ規格

単純な磁気ストライプからデータを復元するには多くのステップが伴います。これまで、読み取りヘッド増幅、F2F波形の生成、およびそれを1と0の文字列へ解釈するプロセスについて説明してきました。復元プロセスの次のステップは、バイナリデータを文字データ（通常はASCII）に変換することです。データの解釈ルールは、読み取るトラックによって異なります。

Track Data Encoding and Parameters

Track	Standard	BPI	Character Type	Number of Characters	Encoding Scheme	Comments
1	IATA	210	Alphanumeric	79	6 Bits + Odd Parity	Read Only
2	ABA	75	BCD	40	4 Bits + Odd Parity	Read Only
3	Thrift	210	BCD	107	4 Bits + Odd Parity	Read & Write (uncommon usage)

一般的なトラックレイアウト

すべてのトラックに共通するのは「クロッキングビット」である。クロッキングビットは各トラックの両端に存在する。クロッキングビットは短いゼロ列で構成され、自己クロッキング式カードリーダーがスワイプ速度を確立し、ビットセル時間幅を設定する機会を与えることを目的としている。本アプリケーションでは、マイクロプロセッサが二進データをデコードする最初のステップとして、クロッキングビットの周期を検査する。

また、全トラックに共通する要素として、スタート・センチネル（SS）、エンド・センチネル（ES）、および縦方向冗長検査（LRC）文字があります。トラック1では、SS文字は「%」、ES文字は「?」です。トラック2および3では、SS文字は「;」、ES文字は「?」です。フィールド・セパレータ（FS）はトラック1と2に共通します。

最初にスタートセンチネルを認識できない場合、カードが逆向きにスワイプされた可能性があります。その場合はデコードアルゴリズムを適宜調整する必要があります。

LRCは、ストライプ上の全先行文字における先行Bnビットの合計値である（オーバーフローは無視される）。各文字に関連付けられた奇数パリティビットは、その文字で奇数ビット（例：1ビット）が誤りだった場合にのみ問題をフラグする。単一文字で2ビット誤りが発生する可能性は低いが、仮に発生しても奇数パリティチェックではこの事象を認識できない。したがってLRCの役割は、トラック全体に関連するデータストリームに対して、もう1層のエラーチェックを追加することです。

トラック1固有のレイアウト

ゼロ | SS | FC | プライマリアカウント (最大19文字) | 名前 (最大26文字) | FS | その他のデータ | ES | LRC | ゼロ

トラック2 特定レイアウト

ゼロ | SS | プライマリアカウント (最大19文字) | FS | その他のデータ | ES | LRC | ゼロ

トラック3

トラック3の非標準的な使用法は数えきれないほど存在します。端的に言えば、このトラックはほぼ廃れた存在であり、結果として特殊なシステムやカスタムシステムがデータを記録する場所となっている。本来の意図は、このトラックを読み書き可能なトラックとして使用し、実際に銀行口座の残高に関する暗号化された情報をここに記録することだった。これにより、ネットワークに接続されていないATMは、引き出しの瞬間に実際の残高を正確に把握せずに現金を払い出すことが可能となった。しかし銀行業界はすぐにATMネットワーク接続し、カード上の動的データを扱う構想を放棄するほうがはるかに優れていると気づくのに時間は要しなかった。よってトラック3は今や孤児になった。

トラック1 文字セット - 6ビットアルファ (奇数パリティ)

Character	P	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	ASCII
Space	1	0	0	0	0	0	0	20
(undefined)								21
								22
(OG)	1	0	0	0	0	1	1	23
\$	0	0	0	0	1	0	0	24
% (SS)	1	0	0	0	1	0	1	25
								26
								27
(	0	0	0	1	0	0	0	28
)	1	0	0	1	0	0	1	29
								2A
								2B
								2C
-	0	0	0	1	1	0	1	2D
.	0	0	0	1	1	1	0	2E
/	1	0	0	1	1	1	1	2F
0	0	0	1	0	0	0	0	30
1	1	0	1	0	0	0	1	31
2	1	0	1	0	0	1	0	32
3	0	0	1	0	0	1	1	33
4	1	0	1	0	1	0	0	34
5	0	0	1	0	1	0	1	35
6	0	0	1	0	1	1	0	36
7	1	0	1	0	1	1	1	37
8	1	0	1	1	0	0	0	38
9	0	0	1	1	0	0	1	39
								3A
								3B
								3C
=	1	0	1	1	1	0	1	3D
								3E
2 (ES)	0	0	1	1	1	1	1	3F

Character	P	B ₆	B ₅	B ₄	B ₃	B ₂	B ₁	ASCII
								40
A	1	1	0	0	0	0	1	41
B	1	1	0	0	0	1	0	42
C	0	1	0	0	0	1	1	43
D	1	1	0	0	1	0	0	44
E	0	1	0	0	1	0	1	45
F	0	1	0	0	1	1	0	46
G	1	1	0	0	1	1	1	47
H	1	1	0	1	0	0	0	48
I	0	1	0	1	0	0	1	49
J	0	1	0	1	0	1	0	4A
K	1	1	0	1	0	1	1	4B
L	0	1	0	1	1	0	0	4C
M	1	1	0	1	1	0	1	4D
N	1	1	0	1	1	1	0	4E
O	0	1	0	1	1	1	1	4F
P	1	1	1	0	0	0	0	50
Q	0	1	1	0	0	0	1	51
R	0	1	1	0	0	1	0	52
S	1	1	1	0	0	1	1	53
T	0	1	1	0	1	0	0	54
U	1	1	1	0	1	0	1	55
V	1	1	1	0	1	1	0	56
W	0	1	1	0	1	1	1	57
X	0	1	1	1	0	0	0	58
Y	1	1	1	1	0	0	1	59
Z	1	1	1	1	0	1	0	5A
								5B
								5C
								5D
^ (FS)	0	1	1	1	1	1	0	5E
								5F

ASCII値に変換するには、P (パリティビット) を無視し、B[6:0]の16進値に20を加算する。

例 : /文字の場合、0F + 20 =2F

トラック2および3の文字セット - 奇数パリティ付き4ビットBCD

Character	P	B ₄	B ₃	B ₂	B ₁	ASCII
0	1	0	0	0	0	30
1	0	0	0	0	1	31
2	0	0	0	1	0	32
3	1	0	0	1	1	33
4	0	0	1	0	0	34
5	1	0	1	0	1	35
6	1	0	1	1	0	36
7	0	0	1	1	1	37

Character	P	B ₄	B ₃	B ₂	B ₁	ASCII
8	0	1	0	0	0	38
9	1	1	0	0	1	39
: (AS)	1	1	0	1	0	3A
; (SS)	0	1	0	1	1	3B
<	1	1	1	0	0	3C
= (FS)	0	1	1	0	1	3D
>	0	1	1	1	0	3E
2 (ES)	1	1	1	1	1	3F

To convert to the ASCII value, ignore the P (parity bit) and add the hex value of B[4:0] to 30 hex.
For example for the : character, B + 30 = 3B

P = Parity Bit, usually odd

SS = Start Sentinel

ES = End Sentinel

FS = Field Separator

AS = Account Separator (Track 3 Only)

OG = Option Graphic

詳細は www.okikadevices.com をご覧いただくか、Okika Devices テクニカルサポートまで
メールでお問い合わせください。 contact@okikadevices.com