ZPL II®
プログラミング・ガイド

第 1 巻

文書名：45541LB-R3.pdf

所有権に関する声明

このマニュアルには、Zebra Technologies Corporation およびその子会社（Zebra Technologies）が所有する情報が含まれています。このマニュアルの唯一の目的は、記載されている機器を操作および保守するユーザーに情報を提供することです。Zebra Technologies の書面による許可なしに、その他の目的のためにこのような独自の情報を使用、複製、または他者に開示することは禁じられています。

製品の改善
製品を継続的に改善していくことは、Zebra Technologies のポリシーです。すべての仕様や設計は、通知なしに変更される場合があります。

責任の放棄
Zebra Technologies では、公開されているエンジニアリング仕様およびマニュアルに誤りが含まれていないよう、万全の対策を講じていますが、誤りが発生することもまれにあります。Zebra Technologies では、誤りが発見された場合にそれを補正し、その誤りから生じる責任を放棄する権利を有しています。

責任の制限
いかなる場合においても、Zebra Technologies、または付属の製品（ハードウェアおよびソフトウェアを含む）の作成、製造、または配布にかかわるその他の関係者は、本製品の使用、使用した結果、または使用できなかった結果から生じるすべての損害（業務利益の損失、業務の中断、または業務情報の損失を含む派生的損害を含むがそれに限定されない）に対し、Zebra Technologies がそのような損害の発生する可能性を通告されていた場合でも、一切責任を負いません。管轄区域によっては、付随的または派生的損害の除外または制限を認めていない場合があるため、上記の制限または除外はお客様に適用されないことがあります。

著作権

目次

機能別目次

序章

序章では、トピックを紹介し、このガイドで使用する標準について説明します。

目次

連絡先

Zebra の連絡先は次のとおりです。

Web サイト： http://www.zebra.com

郵送先住所：

Zebra Technologies Corporation
333 Corporate Woods Parkway
Vernon Hills, IL 60061.3109-3109 U.S.A.
電話 : +1 847.634.6700
ファックス : +1 847.913.8766

Zebra Technologies Europe Limited
Zebra House
The Valley Centre, Gordon Road
High Wycombe
Buckinghamshire HP13 6EQ, UK
電話 : +44 (0) 1494 472872
ファックス : +44 (0) 1494 450103

Asia Pacific, LLC
Bar Code & Card
1 Sims Lane, #06-11
Singapore 387355
電話 : +6 6858 0722
ファックス : +65 6885 0838

サポート

Zebra プログラミング言語 II（ZPL II）に関する質問や問題は、Zebra のサポートにお問い合わせください。

サポートの URL: http://www.zebra.com/SS/service_support.htm

電話 : +1.847.913.2259

本書について

*『ZPL II プログラミング・ガイド第 1 巻』*は次の章で構成されます。

章名	内容の説明
はじめに	Zebra プログラミング言語（ZPL）に関する概要を提供します。
ZPL II のコマンド	各 ZPL コマンドについてアルファベット順に詳しく説明します。
ZBI コマンド	Zebra Basic Interpreter （ZBI）についてアルファベット順に詳しく説明します。
ZBI -- チュートリアル	

文書の表記規則

本書では、特定の情報を提供するにあたって次の表記規則が使用されます。

「この章について」の項 これらの項では、章の主な項をそれぞれの最初のページ番号とともにリストし、説明します。この項は、このガイドの Adobe Acrobat .pdf バージョンのハイパーリンク・コンポーネントとして主に機能します。

代替色（オンラインのみ）相互参照には、このガイドの別の項にジャンプするためのホット・リンクが含まれています。このガイドを .pdf 形式でオンライン表示している場合に、相互参照（*青いテキスト*）をクリックすると、参照先に直接ジャンプします。

コマンド コマンドはすべて Courier New フォントで表示されます。たとえば、ZPL コマンドは、Courier New で表示されます。

ファイルとディレクトリ ファイル名とディレクトリはすべて Courier New フォントで表示されます。たとえば、Zebra<version number>.tar ファイルや /root ディレクトリなどのように表示されます。

以下は、本書を通して使用されている規則とその意味です。

重要、注記、および例 これらのトピックについては次の例で定義しています。

重要 • タスクを完了するために重要な情報を通知します。

注記 • 本文の要点を強調または補足する中立的情報または肯定的情報を示します。

例 • テキストの内容を明確にするための例やシナリオを提供します。

参考文献

*ZPL II® プログラミング・ガイド 第1 巻*に加えて、以下の文書が参考になります。

注記 • これらの文書のほとんどは、弊社の Web サイトから入手できます。

- *The ZebraLinka Solution Application white-paper*
- *ZBI Guide*
- *ZPL II® Programming Guide Volume Two: The X.10 Environment*


Zebra

はじめに

このガイドは、ファームウェアでサポートされるプログラミング・コマンドをアルファベット順に網羅した参考書です。

ファームウェア プリンタのファームウェア・バージョンは、設定ラベルを印刷すると確認できます。

注記 • ファームウェアのアップグレードは次のサイトから入手できます。www.zebra.com

前のバージョンの Zebra プリンタ・ファームウェアをご使用の場合、以前どおりに機能する同じコマンドもありますが、X.10 以前のバージョンのファームウェアでは認識されない新しいコマンドもほぼ同数あります。その他のコマンドは、以下のような革新技術に対応するために再設計され、強化されています。

- ZebraNet® ALERT
- リアルタイム・クロック
- Zebra BASIC Interpreter（ZBI）

このガイドに示した例を再生するには、ASCII 専用ファイルを作成できるワープロまたはテキスト・エディタならどれを使用してもかまいません。例の大半は、一連の命令ラインで構成されています。ラインを入力したら Enter を押してください。試している例のすべてのラインについて、引き続きこの処理を実行します。

詳細と相互参照を提供するために、*第 2 巻* で取り上げられている機能に直接関連するコマンドの場合は、「コメント」の見出しの後に、参照先の付録または項が記載されています。


Zebra

ZPL II のコマンド

この項には、ZPL II のコマンドのアルファベット順の全リストが含まれます。

説明 この見出しは、コマンドの使用方法、その機能、およびその特徴などの説明を提供します。

フォーマット フォーマットでは、コマンドの構文的な編成方法や、含まれるパラメータについて説明します。

例 ^B8 コマンドでは、EAN-8 バー・コードが印刷されます。^B8 コマンドのフォーマットは ^B8o,h,f,g です。これはキャレット記号（^）、コマンド・コード（B8）、およびパラメータ（o, h, f, g）で構成されています。^B8 の後に続く文字の o、h、f、g は、パラメータで、サポートされる値で置換されます。

パラメータ コマンドの機能をさらに特定するために定義できる値がコマンドに含まれる場合、それらはパラメータとして示されます。通常、パラメータには*有効値*と*デフォルト値*とがあります。

引き続き ^B8 の例を使用しますが、h パラメータは次のように定義されます。

h = バー・コードの高さ（ドット数）

*有効値：*1 ～ 32000

デフォルト値：^BY によって設定された値

たとえば ~JA（すべてキャンセル）のようにコマンドにパラメータがない場合、パラメータの見出しが削除され、コマンドのフォーマット（~JA）が有効な ZPL II コードであることを示しています。

例 • コンテキストを示すことによってコマンドを最も明確に説明できる場合は、ZPL II コード の例が提供されます。入力されたコードを正確に示すテキストが、見分けやすい Courier フォントで示されます。^B8 コマンドを使用したコード例は次のようになります。

```
^XA
^FO50,50
^B8N,100,Y,N
^FD1234567^FS
^XZ
```

^B8 のパラメータの文字がコマンドに適用する実際の値で置換されたことに注意してください。この例では、N、100、Y、N が入力されています。

コメント この項は、プログラマにとって貴重な注記、コマンド相互作用に関する警報、考慮の必要があるコマンド固有の情報などを記載するために使用します。

例 • コメントの例：**「このコマンドはプリンタがアイドル状態である場合にのみ機能します」**または**「値がパラメータの制限値を超えると、このコマンドは無視されます」**など。

コメントが特定の設定に直接適用する場合、それはパラメータの隣にも含められます。

ZPL の基本演習

これらの演習は、ZPL の基本的なコマンドを初級の ZPL ユーザーに紹介することを目的としています。

このチェックリストを必ず確認してください。

- プリンタに十分な大きさのラベルをセットします。
- 設定ラベルを印刷します（CANCEL テスト）。
- 設定ラベルで、LEFT POSITION が 000 に、LABEL TOP が 000 に設定されていることを確認します。
- プリンタの解像度を決定します。

  解像度は設定ラベルにリストされています。8/MM = 200 dpi、12/MM = 300 dpi、および 24/MM = 600 dpi

- ZPL ファイルを作成するには、DOS テキスト・エディタを使用します。
- ファイルは .txt ファイルとして保存し、DOS コマンド・ラインからプリンタにコピーできます。

始める前に

以下は、始める前に理解しておく必要のある重要点です。

- 200 dpi は印刷ヘッドの解像度が 200 ドット / インチであることを意味します。このプリンタで長さ 100 ドットのラインを描画するようにプログラムするとそれは半インチになります。 300 dpi のプリンタでは 100 ドットは 1/3 インチの長さのラインとして印刷されます。
- すべての座標が参照するホーム・ポジションは、プリンタからラベルが出てくる際、左側の後縁になります。（これには例外もあります。）

演習

演習は単純なものから始まり、 徐々に難度を増していきます。 これによって一般に使用されている ZPL のコマンドの多くを試す機会が与えられます。 すべてのコマンドが含まれるわけではありませんが、 代表的なコマンドは習得できます。 プリンタのファームウェアのバージョンにより、 サポートされていないコマンドもあります。

演習 1 この演習では、入力した名前の場所を指定する方法を示します。

1 ラベルに名前を印刷します。

2 モデルとして以下のフォーマットを使い、ラベルに名前のみを印刷することから始めます。

重要 • 名前は、コードの 2 行目の **xxxxxxxxxxx** の場所に挿入されます。

```
^XA

^FO50,50^ADN,36,20^xxxxxxxxxxxFD^FS

^XZ

Send the above format to the printer.

^XA every format must start with this command

^XZ every format must end with this command

^FD field data

^FS field separator

^FO field origin
```

3 ラベルが正しく印刷されたら、^FOx の後の最初の数値を変更し、印字位置への影響を確認します。次に ^FO50,x の後の 2 番目の数値を変更し、印字位置への影響を確認します。

フォント命令

^ADN

4 ^ADN,x,x コマンドの後の数値を変更します。

- 18,10 は **D** フォントの最小サイズです。
- 最初の数値はフォントの高さを、2 番目の数値は幅をそれぞれドット数で表したものです。
- 最大サイズには、最小サイズの 10 倍までの倍数を使用できます。

例 • 180,100 は **D** フォントの最大サイズです。

- 25,18 は有効なサイズではありません。プリンタは認識可能な次のサイズに値を丸めます。

5 *『ZPL II プログラミング・ガイド第 2 巻』* の付録 E にあるフォント・マトリックス表で、試すことのできるその他のフォントを確認してください。

6 ゼロ・スケーラブル・フォント ^A0N,x,x を試してみてください。

このフォントは、スケーラブルであるため任意の高さと幅を選択できます。

回転コマンド

7 ^ADN を ^ADR に、次に ^ADI に、そして ^ADB に変更します。

印字位置の変化を確認してください。

8 フィールドを追加します。

9 フォントサイズ ^ADN,36,20 を使用して、名前のすぐ下に印刷する次のフィールドを 2 つ追加します。

番地

市、州、郵便番号

10 以下で始まるコード・ラインを 2 つ追加する必要があります。

```
^XA

^FO50,50^ADN,36,20^FDxxxxxxxxxxx^FS

^FO     (fill in the rest)

^FO     (fill in the rest)

^XZ
```

これらのフィールドがすべて同じフォントとサイズで印刷されること、およびフィールドの左側に同じ縦の調整があることを確認してください。

名前

1200 W Main Street

Anytown, Il 60061

演習 2 ボックスと線

1. **演習 1** の住所フォーマットを使用します。
2. 既存のフォーマットに次の新しいラインを追加します。

   ```
   ^FO50,200^GB200,200,2^FS
   ```

   これにより、幅 1 インチ、長さ 1 インチ、線幅 2 ドットのボックスが印刷されます。

3. 名前と住所を均等に囲むようにボックスの位置とサイズを変更します。
4. 以下を追加して線を印刷します。

   ```
   ^FO50,300^GB400,0,4,^FS
   ```

   これにより、長さ 2 インチ、線幅 4 ドットの横方向の線が印刷されます。

5. 次のコードを使用して縦の線を印刷します。

   ```
   ^F0100,50^GBO,400,4^FS
   ```

演習 3 バー・コード – ^B3 Code 39 バー・コード

1. 以下のフォーマットを作成し、プリンタに送信します。

   ```
   ^XA

   ^FO50,50^B3N,N,100,Y,N^FD123456^FS

   ^XZ
   ```

2. ^B3 ストリング内の各パラメータを変更し、影響を確認してください。

重要 • 有効なパラメータについては、*^B3 ページ 25* を参照してください。

```
^B3o,e,h,f,g

^BY
```

3. ^B3 のすぐ前に ^BY コマンドを挿入すると、狭いバーの幅を変更できます。

   ```
   ^FO50,50^BY2^B3..etc ^BYx, acceptable values for x are
   1 through 10
   ```

4 太いバーに対する狭いバーの比率を変更します。

```
^FO50,50^BY2,3^B3..etc ^BY2,x acceptable values for x
are 2.1 through 3 in .1 increments
```

5 ^B3 バー・コードを 90° 回転させて、テキスト表示ラインがバー・コードの上になるようにして印刷します。

6 ^XZ のすぐ前に ^PQ を追加して、複数のラベルを印刷します。

```
^PQ4

^XZ
```

^PR 印字レート（1 秒あたりのインチ数）

7 フォーマットの先頭の ^XA の後に ^PR コマンドを追加し、印字レート（印字速度）を変更します。

```
^XA

^PR4 then try ^PR6 ^PRx acceptable values for x are 2
through 12 (check printer specs)
```

印字速度がバー・コードの印字品質にどのように影響するかを確認してください。印字速度が増すと、プリンタの濃度設定を上げる必要がある場合があります。

演習 4 ^SN – シリアル番号コマンド

1 次のフォーマットをプリンタに送信します。

```
^XA

^FO100,100^ADN,36,20^SN001,1,Y^FS

^PQ3

^XZ
```

^SNv,n,z を変更して増分 / 減分、および前ゼロ処理の機能演習を行うには、このガイドを参照してください。

シリアル番号に英数字が含まれる場合、このサンプル・シーケンスのようにデータの特定セグメントが真中にある場合でも、そのセグメントを増分 / 減分することができます。

```
ABCD1000EFGH, ABCD1001EFGH, ABCD1002EFGH
```

2 このファイルをプリンタに送信して、シリアル番号が増分されるのを確認してください。アルファベット文字の場合、^SF コマンドも有効です。

```
^XA

^FO100,100^ADN,36,20^FDABCD1000EFGH^SF%%%%dddd%%%%,10000^FS

^PQ15

^XZ
```

フィールド・データの文字のポジションが ^SF データ・ストリングと並んでいることに注意してください。

^	F	D	A	B	C	D	1	0	0	0	E	F	G	H
^	S	F	%	%	%	%	d	d	d	d	%	%	%	%
										1	0	0	0	0
										2	0	0	0	0
										3	0	0	0	0

続き ...

							1	0	1	4	0	0	0	0

最後のラベルには ABCD1014EFGH が印刷されます。

% は増分/減分したくない位置に使用し、d は小数、10000 は増分値を示します。

^SF の詳細については、*^SF ページ 311* を参照してください。

演習 5 テンプレートをメモリに保存する。 ^IS およびイメージ保存とイメージ読み込み。

^IS およびイメージ保存とイメージ読み込み。

注記 • この演習では多くのデータを入力するため、誤入力の危険性があります。ラベルのエラーを確認しながらコードをトラブルシューティングする練習にもなります。

1 次のフォーマットをプリンタに送信します。

```
^XA

^FO20,30^GB750,1100,4^FS

^FO20,30^GB750,200,4^FS

^FO20,30^GB750,400,4^FS

^FO20,30^GB750,700,4^FS

^FO20,226^GB325,204,4^FS

^FO30,40^ADN,36,20^FDShip to:^FS

^FO30,260^ADN,18,10^FDPart number #^FS

^FO360,260^ADN,18,10^FDDescription:^FS

^FO30,750^ADN,36,20^FDFrom:^FS

^ISR:SAMPLE.ZPL^FS

^XZ
```

2 次のフォーマットを送信します。

```
^XA

^ILR:SAMPLE.ZPL^FS

^FO150,125^ADN,36,20^FDAcme Printing^FS

^FO60,330^ADN,36,20^FD14042^FS

^FO400,330^ADN,36,20^FDScrew^FS

^FO70,480^BY4^B3N,,200^FD12345678^FS
```

```
^FO150,800^ADN,36,20^FDMacks Fabricating^FS

^XZ
```

この方法では、テンプレートをプリンタのメモリに一度送信する必要があるだけです。後続のフォーマットは、テンプレートを呼び出して、変数データをそれに結合して送信できます。この場合、ファイルはプリンタの揮発性の R: メモリに保存されていました。

DF および ^XF — ダウンロード・フォーマットと呼び出しフォーマット

^IS および ^IL コマンドと同様な概念。一般的に、^IS と ^IL は ^DF と ^XF よりもプリンタで速く処理されます。

以下は、^DF と ^XF のフォーマット構造によって、先ほど試した ^IS/^IL のサンプルと同様なラベルを生成する方法です。

```
^XA

^DFR:SAMPLE.ZPL^FS

^FO20,30^GB750,1100,4^FS

^FO20,30^GB750,200,4^FS

^FO20,30^GB750,400,4^FS

^FO20,30^GB750,700,4^FS

^FO20,226^GB325,204,4^FS

^FO30,40^ADN,36,20^FDShip to:^FS

^FO30,260^ADN,18,10^FDPart number #^FS

^FO360,260^ADN,18,10^FDDescription:^FS

^FO30,750^ADN,36,20^FDFrom:^FS

^FO150,125^ADN,36,20^FN1^FS (ship to)

^FO60,330^ADN,36,20^FN2^FS(part num)

^FO400,330^ADN,36,20^FN3^FS(description)
```

```
^FO70,480^BY4^B3N,,200^FN4^FS(barcode)

^FO150,800^ADN,36,20^FN5^FS (from)

^XZ

^XA

^XFR:SAMPLE.ZPL

^FN1^FDAcme Printing^FS

^FN2^FD14042^FS

^FN3^FDScrew^FS

^FN4^FD12345678^FS

^FN5^FDMacks Fabricating^FS

^XZ
```

^A

スケーラブル / ビットマップ・フォント

説明 ^A コマンドは、組み込みフォントまたは TrueType フォントを使用するスケーラブル / ビットマップ・フォントです。^A は、現在の ^FD ステートメントまたはフィールドのフォントを指定します。^A によって指定されたフォントは、その ^FD エントリに対して一度だけ使用されます。^A の値が再指定されないと、次の ^FD エントリに対してはデフォルトの ^CF フォントが使用されます。

フォーマット ^Afo,h,w

次の表では、このフォーマットのパラメータを示します。

コマンド	詳細
f = フォント名	*有効値 :* A ～ Z、および 1 ～ 9 *デフォルト値 :* A プリンタ内の任意のフォント（ダウンロードされたもの、EPROM、保存されたフォント、A ～ Z および 1 ～ 9 のフォント）。
o = フォントの向き	*有効値 :* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値 :* 最後に受け入れられた ^FW 値または ^FW のデフォルト

コマンド	詳細
h = 文字の高さ（ドット数）	スケーラブル *有効値：*10 ～ 32000 *デフォルト値：*15 または最後に受け入れられた ^CF 値 ビットマップ *有効値:*標準の高さの 2 ～ 10 倍の値で、1 ずつ増分します。 *デフォルト値：*指定のフォントの標準マトリックスの高さ
w = 幅（ドット数）	スケーラブル *有効値：*10 ～ 32000 *デフォルト値：*12 または最後に受け入れられた ^CF 値 ビットマップ *有効値:*標準幅の 2 ～ 10 倍の倍数で、1 ずつ増分します。 *デフォルト値:*指定のフォントの標準マトリックスの幅

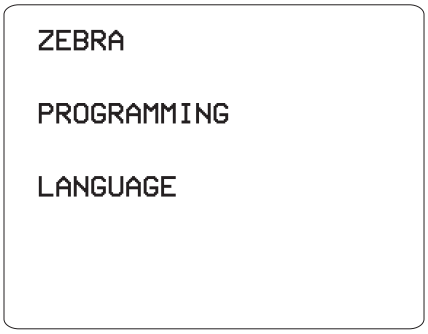
スケーラブル・フォント・コマンド

例 • 次はスケーラブル・フォント・コマンドの例です。

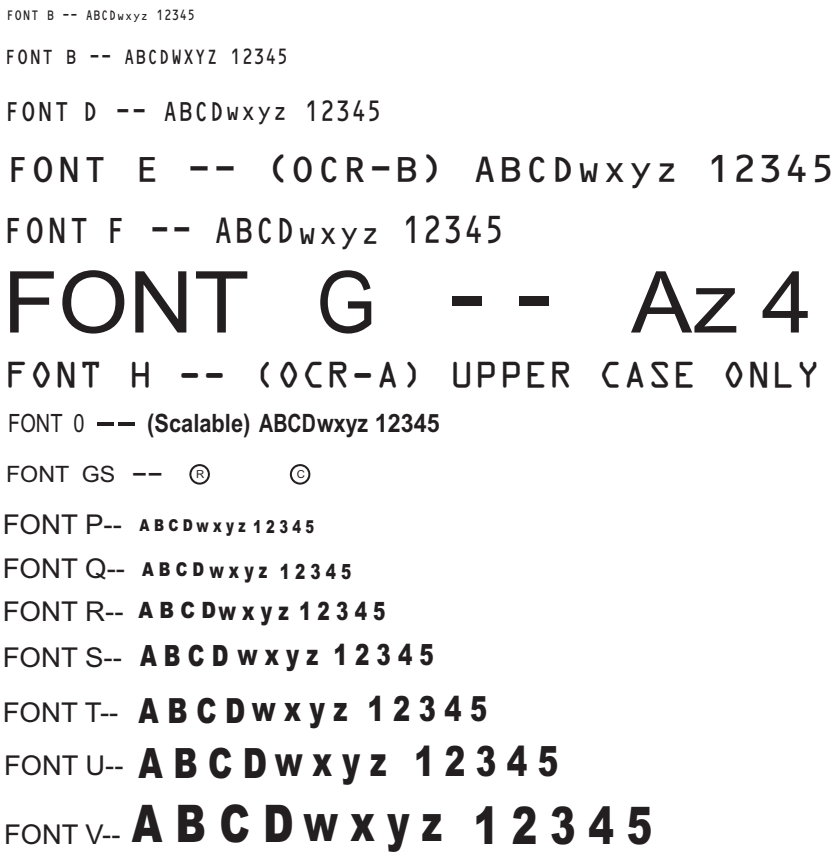
ZPL II CODE	GENERATED LABEL
^XA ^FO50,50 ^A0,32,25 ^FDZEBRA^FS ^FO50,150 ^A0,32,25 ^FDPROGRAMMING^FS ^FO50,250 ^A0,32,25^FDLANGUAGE^FS ^XZ	ZEBRA PROGRAMMING LANGUAGE

ビットマップ・フォント・コマンド

例 • 次はビットマット・フォント・コマンドの例です。

ZPL II CODE	GENERATED LABEL
^XA ^FO50,50 ^ADN,36,20 ^FDZEBRA^FS ^FO50,150 ^ADN,36,20 ^FDPROGRAMMING^FS ^FO50,250 ^ADN,36,20^FDLANGUAGE^FS ^XZ	

例 • 次は P-V フォントの例です。



コメント フォントは、幅に対する高さの比率の標準を定義するマトリックスを使用して構築されます。 高さと幅のどちらかの値のみを指定すると、そのフォントの標準マトリックスが自動的に他方の値を決定します。 値の指定がない場合や 0（ゼロ）を入力した場合は、 高さまたは幅は標準フォント・マトリックスによって決定されます。

^A@

フォント名を使用したフォント呼び出し

説明 ^A@ コマンドでは、^A で使用した指定文字ではなく、フォントの完全な名前を使用します。^A@ の値を一度定義すると、^A@ によって新しいフォント名が指定されるまで、その値がそのフォントを表します。

フォーマット ^A@o,h,w,d:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = フォントの向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*N または ^FW 値
h = 文字の高さ（ドット数）	*デフォルト値：***w**（文字幅）で指定した倍率、または最後に受け入れられた ^CF 値。指定がない場合は、基本の高さが使用されます。 **スケーラブル** 値は文字ブロック全体の高さをドット数で表したものです。文字はスケールされるため、倍率係数は不要です。 **ビットマップ** 値は、フォントの基本の高さに最も近い整数の倍数に四捨五入され、フォントの基本の高さで割って、制限値に最も近い倍数を出します。

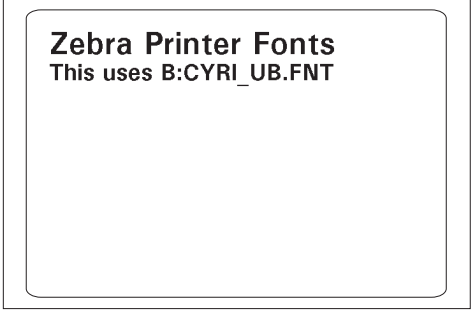
パラメータ	詳細
w = 幅(ドット数)	*デフォルト値：***h**（高さ）で指定された倍率、または最後に受け入れられた ^CF 値。指定がない場合は、基本の幅が使用されます。 **スケーラブル** 値は文字ブロック全体の幅をドット数で表したものです。文字はスケールされるため、倍率係数は不要です。 **ビットマップ** 値は、フォントの基本幅に最も近い整数の倍数に四捨五入され、フォントの基本幅で割って、制限値に最も近い倍数を出します。
d＝フォントのドライブの場所	*有効値：* R:、E:、B:、および A: *デフォルト値：* R:
o ＝フォント名	*有効値：*任意の有効フォント *デフォルト値：*無効な名前が入力された場合や、名前が入力されなかった場合は、^CF によって設定されたデフォルトが使用されます。^CF でフォントが指定されていない場合は、フォント A が使用されます。 ここで指定されたフォントは、今後フォント名の指定のないすべての ^A@ コマンドで使用されます。
x＝拡張子	*固定値：* .FTN

例 • この例の後にはコールアウトを説明する表が示されています。

ZPL II CODE

```
1 ^XA
2 ^A@N,50,50,B:CYRI_UB.FNT
3 ^FO100,100
4 ^FDZebra Printer Fonts^FS
5 ^A@N,40,40
6 ^FO100,150
7 ^FDThis uses B:CYRI_UB.FNT^FS
8 ^XZ
```

GENERATED LABEL



1	ラベル・フォーマットを開始します。
2	非揮発性のプリンタ・メモリで CYRI_UB.FNT を検索します (B:)。フォントが見つかると、^A@ コマンドによって印刷の向きが標準に、文字サイズが 50 ドット x 50 ドットに設定されます。
3	フィールドの基点を 100,100 に設定します。
4	フィールド・データ *Zebra Printer Fonts* をラベルに印刷します。
5	フォントを再度呼び出すと、文字サイズは 40 ドット x 40 ドットに縮小されます。
6	新しいフィールド基点を 100,150 に設定します。
7	フィールド・データ *This uses the B:CYRI_UB.FNT* をラベルに印刷します。
8	ラベル・フォーマットを終了します。

コメント スケーラブル / ビットマップ・フォントの詳細については、*『ZPL II プログラミング・ガイド第 2 巻』* を参照してください。

^B1

Code 11 バー・コード

説明 ^B1 コマンドは、USD-8 コードとしても知られる Code 11 バー・コードを生成します。Code 11 バー・コードでは、各文字は 3 本のバーと 2 本のスペースで構成され、文字セットには 10 桁とハイフンが含まれます。

- ^B1 は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。

フォーマット ^B1o,e,h,f,g

重要 • Code 11 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』*で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：現在の ^FW 値*
e = チェック・ディジット	*有効値：* Y（はい） = 1 桁 N（いいえ） = 2 桁 *デフォルト値：*N

パラメータ	詳細
h＝バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f ＝テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g ＝テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は Code 11 バー・コードの例です。

CODE 11 BAR CODE

ZPL II CODE

```
^XA
^FO100,100^BY3
^B1N,N,150,Y,N
^FD123456^FS
^XZ
```

CODE 11 BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9 -

Internal Start/Stop Character:△

When used as a stop character:

△ is used with 1 check digit
△ is used with 2 check digits

^B2

Interleaved 2 of 5 バー・コード

説明 ^B2 コマンドは、高密度の自己チェック式連続型の数値シンボル・コード体系である Interleaved 2 of 5 バー・コードを生成します。

Interleaved 2 of 5 バー・コードの各データ文字は、5 つのバー / スペースの 5 エレメントで構成されます。5 エレメントのうち 2 つは太く、3 つは細くなっています。このバー・コードは、すべてスペースの文字とすべてバーの文字をインターリーブ（交互配置）することによって形成されています。

- ^B2 は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。

フォーマット ^B2o,h,f,g,e

重要 • Interleaved 2 of 5 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f = テキスト表示ラインを印刷	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N
e = モジュラス 10 チェック・ディジットを計算し印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は Interleaved 2 of 5 バー・コードの例です。

INTERLEAVED 2 OF 5 BAR CODE	ZPL II CODE
	^XA ^FO100,100^BY3 ^B2N,150,Y,N,N ^FD123456^FS ^XZ

INTERLEAVED 2 OF 5 BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9
Start/Stop (internal)

コメント Interleaved 2 of 5 バー・コードの全桁数は偶数である必要があります。プリンタは奇数の桁数を受け取ると自動的に前ゼロを追加します。

Interleaved 2 of 5 バー・コードでは、エラー・チェックにモジュラス 10 チェック・ディジット・スキームを使用します。モジュラス 10 チェック・ディジットの詳細については、*『ZPL プログラミング・ガイド第 2 巻』* の付録 C を参照してください。

^B3

Code 39 バー・コード

説明 Code 39 バー・コードは、米国国防総省も含め、多くの産業における標準で、米国規格協会（ANSI）の標準 MHO10.8M-1983 で指定されている 3 種類のシンボル・コード体系の 1 つです。Code 39 は、USD-3 Code および 3 of 9 Code としても知られています。

Code 39 バー・コードの各文字は、5 本のバーと 4 本のスペースの 9 エレメント、および文字間ギャップで構成されています。9 エレメントのうち 3 本は太く、残りの 6 本は細くなっています。

- ^B3 は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。
- Code 39 は、スタート / ストップ文字（*）を自動的に生成します。
- テキスト表示ラインがオンの場合、スタート / ストップ文字のアステリスクがテキスト表示ラインに印刷されます。
- Code 39 は、フル ASCII セット 128 文字をエンコードできます。

フォーマット `^B3o,e,h,f,g`

重要 • Code 39 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
e = モジュラス 43 チェック・ディジット	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f = テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は Code 39 バー・コードの例です。

CODE 39 BAR CODE	ZPL II CODE
	^XA ^FO100,100^BY3 ^B3N,N,100,Y,N ^FD123ABC^FS ^XZ

CODE 39 BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
- . $ / + % Space

コメント 拡張 ASCII は、スキャナの機能であり、バー・コードの機能ではありません。この機能を使用するには、ご使用のスキャナで拡張 ASCII が有効になっている必要があります。Code 39 で拡張 ASCII を有効にするには、^FD ステートメントで *+$* をエンコードする必要があります。Code 39 で拡張 ASCII を無効にするには、^FD ステートメントで *-$* をエンコードする必要があります。

例 • 次の例では、ライン・フィードのあるキャリッジ・リターンを Code 39 バー・コードにエンコードします。

ZPL II CODE	GENERATED LABELS
^XA ^FO20,20 ^B3N,N,100,Y ^FDTEST+$$M$J-$^FS ^XZ	

Code 39 のフル ASCII モード

Code 39 は、次の表に示した文字の組み合わせを使用して、128 文字のフル ASCII セットを生成できます。

ASCII	Code 39
SOH	$A
STX	$B
ETX	$C
EOT	$D
ENQ	$E
ACK	$F
BEL	$G
BS	$H
HT	$I
LF	$J
VT	$K
FF	$L
CR	$M
SO	$N
SI	$O
DLE	$P
DC1	$Q
DC2	$R
DC3	$S
DC4	$T
NAK	$U
SYN	$V
ETB	$W
CAN	$X
EM	$Y
SUB	$Z
ESC	%A
FS	%B
FS	%C
RS	%D
US	%E

ASCII	Code 39
SP	Space
!	/A
“	/B
#	/C
$	/D
%	/E
&	/F
‘	/G
(	/H
)	/I
*	/J
++	/K
‘	/L
-	-
.	.
/	/O
0	O
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
:	/Z
;	%F
<	%G
=	%H
>	%I
?	%J

表 A Code 39 フル ASCII モード

ASCII	Code 39
@	%V
A	A
B	B
C	C
D	D
E	E
F	F
G	G
H	H
I	I
J	J
K	K
L	L
M	M
N	N
O	O
P	P
Q	Q
R	R
S	S
T	T
U	U
V	V
W	W
X	X
Y	Y
Z	Z
[	%K
\	%L
]	%M
^	%N
_	%O

ASCII	Code 39
‘	%W
a	+A
b	+B
c	+C
d	+D
e	+E
f	+F
g	+G
h	+H
I	+I
j	+J
k	+K
l	+L
m	+M
n	+N
o	+O
p	+P
q	+Q
r	+R
s	+S
t	+T
u	+U
v	+V
w	+W
x	+X
y	+Y
z	+Z
{	%P
|	%Q
}	%R
~	%S
DEL	%T, %X

表 B Code 39 フル ASCII モード

^B4

Code 49 バー・コード

説明 ^B4 コマンドは、複数行連続型、フル ASCII セット 128 文字エンコード可能な可変長シンボル・コード体系を作成します。このコードは、小さなスペースに大量のデータを必要とするアプリケーションに最適です。

コードは 2 ～ 8 行で構成されます。行は、先行クワイエット・ゾーン、8 コード文字をエンコードする 4 つの記号文字、ストップ・パターン、後続クワイエット・ゾーンで構成されます。行は、1 モジュールの高さの区切りバーによって区切られます。各シンボル文字は、Code 49 文字セットの 2 文字をエンコードします。

- ^B4 には固定印字比率があります。
- 行はどの順序でスキャンしてもかまいません。

フォーマット ^B4o,h,f,m

重要 • Code 49 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = 個々の行の高さの倍数	*有効値：*1 ～ラベルの高さ *デフォルト値：*^BY によって設定された値 この数値をモジュールで掛けると、個々の行の高さのドット数になります。この値に 1 を使用することはお薦めできません。

パラメータ	詳細
f ＝テキスト表示ラインを印刷する	*有効値：* N ＝ ラインを印刷しない A ＝テキスト表示ラインをコードの上に印刷する B ＝テキスト表示ラインをコードの下に印刷する *デフォルト値：*N フィールド・データが 2 行を超えると、テキスト表示ラインがバー・コード・シンボルの右端からはみ出します。
m＝開始モード	*有効値：* 0 ＝ 標準英数字モード 1 ＝ 複数読み取り英数字 2 ＝ 標準数字モード 3 ＝ グループ英数字モード 4 ＝ 標準英数字 Shift 1 5 ＝ 標準英数字 Shift 2 A ＝自動モードプリンタは、フィールド・データを分析して開始モードを決定します。 *デフォルト値：*A

例 • 次は Code 49 バー・コードの例です。

CODE 49 BAR CODE	ZPL II CODE
	^XA ^FO150,100^BY3 ^B4N,20,A,A ^FD12345ABCDE^FS ^XZ

Field Data Set	Unshifted Character Set	Shift 1 Character Set	Shift 2 Character Set
0	0	’	
1	1	ESC	;
2	2	FS	<
3	3	GS	=
4	4	RS	>
5	5	US	?
6	6	!	@
7	7	“	[
8	8	#	\
9	9	&	]
A	A	SOH	a
B	B	STX	b
C	C	ETX	c
D	D	EOT	d
E	E	ENQ	e
F	F	ACK	f
G	G	BEL	g
H	H	BS	h
I	I	HT	I
J	J	LF	j
K	K	VT	k
L	L	FF	l
M	M	CR	m
N	N	SO	n
O	O	SI	o
P	P	DLE	p
Q	Q	DC1	q
R	R	DC2	r
S	S	DC3	s
T	T	DC4	t
U	U	NAK	u
V	V	SYN	v
W	W	ETB	w
X	X	CAN	x
Y	Y	EM	y
Z	Z	SUB	z
-	-	(	_
.	.	)	‘
SPACE	SPACE	Null	DEL
$	$	*	{
/	/	,	|
++	++	:	}
%	%	reserved	~
< (Shift 1)			
> (Shift 2)			
: (N.A.)			
; (N.A.)			
? (N.A.)			
= (Numeric Shift)			

Code 49 Shift 1 and 2 Character Substitutions

表 C Code 49

Code 49 のフィールド・データ文字セット

0 ～ 5 の開始モードを使用する際に、プリンタに送られる ^FD データは、Code 49 国際文字セットです。これは前のページの Code 49 表の最初の列に示されています。以下は Code 49 コントロール文字です。

: ; < = > ?

0 ～ 5 のモードを使用する場合は、有効なフィールド・データを提供する必要があります。シフト文字はシフト文字 1 文字と、非シフト文字 1 文字のシーケンスとして送信されます。

例 • 小文字の **a** をエンコードするには、**a>**（Shift 2）に続けて、大文字の **A** を送信します。テキスト表示ラインの印刷を選択した場合、小文字の *a* がテキスト表示ラインに印刷されます。これにより、スキャナの出力が反映されます。Code 49 では上段の英数字のみが使用されます。

無効なシーケンスが検出されると、Code 49 フォーマッタはフィールド・データの解釈を停止し、無効なシーケンスの発生個所までのデータを含むシンボルを印刷します。以下は無効なシーケンスの例です。

- 0 ～ 9 以外の任意の文字または数字スペースで数字モードが終了されている場合。
- モード 4（標準英数字 Shift 1）で開始し、最初のフィールド・データ文字が Shift 1 セットではない場合。
- モード 5（標準英数字 Shift 2）で開始し、最初のフィールド・データ文字が Shift 2 セットではない場合。
- Shift 1 に続いて Shift 1 セットにない文字を送信した場合。
- Shift 2 に続いて Shift 2 セットにない文字を送信した場合。
- Shift 1 または Shift 2 コントロール文字を 2 つ送信した場合。

Code 49 自動モードの利点

デフォルト（自動モード）を使用すると、開始モードを選択したり、手動で文字シフトを実行したりする必要がまったくなくなります。自動モードでは、入力された ASCII ストリングを分析したうえで適切なモードを判断し、すべての文字シフトを実行し、最大限の効果を得るためにデータを圧縮します。

数字モードは、連続した数字が 5 個以上検出された場合にのみ選択またはシフトされます。8 文字以下で構成される数字のストリングの場合、数字パッケージにはスペース面での利点はありません。

^B5

Planet Code バー・コード

説明 ^B5 コマンドは、常駐バー・コードとしてすべてのプリンタでサポートされています。

フォーマット ^B5o,h,f,g

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向きのコード	*有効値 :* N = 標準 R = 回転 I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値 :* 現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値 :* 1 ～ 9999 *デフォルト値 :* ^BY によって設定された値
f = テキスト表示ライン	*有効値 :* N = デフォルトなし Y = はい
g = テキスト表示ラインがバー・コードの上に印刷されるかどうかを決定します。	*有効値 :* N = デフォルトなし Y = はい

例 • 次は Planet Code バー・コードの例です。

ZPL II CODE	GENERATED LABEL
^XA ^FO150,100^BY3 ^B5N,100,Y,0 ^FD12345678901$FS ^XZ	

^B7

PDF417 バー・コード

説明 ^B7 コマンドは、二次元の複数行連続型のスタック式シンボル・コード体系である PDF417 バー・コードを生成します。PDF417 は、1 バー・コードあたり 1,000 文字以上をエンコードできます。このコードは、バー・コードの読み取り時に大量のデータを必要とするアプリケーションに最適です。

このバー・コードは 3 ～ 90 のスタック式の行で構成されます。各行は、スタート / ストップ・パターンと*コード・ワード*と呼ばれるシンボル文字で構成されています。コード・ワードには 4 本のバーと 4 本のスペースが含まれます。1 行には最低 3 つのコード・ワードが必要です。

PDF417 バー・コードは、構造化結合オプション（^FM）を使用できるため、複数のバー・コードを印刷することにより、フィールド・データの制限を拡張することができます。構造化結合の使用の詳細については、*^FM ページ 174* を参照してください。

- PDF417 には固定印字比率があります。
- フィールド・データ（^FD）は文字データ 3K に限定されています。

フォーマット ^B7o,h,s,c,r,t

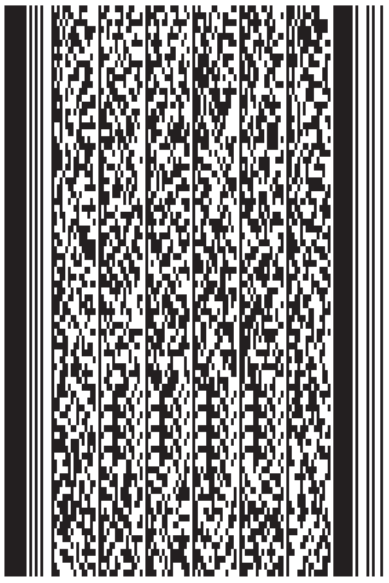
次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値 :* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値 :* 現在の ^FW 値
h = 個々の行のバー・コードの高さ（ドット数）	*有効値 :* 1 ～ラベルの高さ *デフォルト値 :* ^BY によって設定された値 この数値をモジュールで掛けると、個々の行の高さのドット数になります。この値に 1 を使用することはお薦めできません。
s = セキュリティ・レベル	*有効値 :* 1 ～ 8（エラー検出および訂正） *デフォルト値 :* 0（エラー検出のみ） これにより、シンボルに対して生成されるエラー検出 / 訂正コード・ワード数が決定します。デフォルトのレベルでは、訂正機能なしのエラー検出のみが提供されます。セキュリティ・レベルを上げると、エラー訂正のレベルも上がり、シンボル・サイズも大きくなります。
c = エンコードするデータ列の数	*有効値 :* 1 ～ 30 *デフォルト値 :* 1:2（行 : 列のアスペクト比率） コード・ワードの列数を指定することにより、シンボルの幅を制御できます。

パラメータ	詳細
r = エンコードする行の数	*有効値：* 3 ～ 90 *デフォルト値：* 1:2（行：列のアスペクト比率） シンボルの行数を指定することにより、シンボルの高さを制御できます。たとえば、行の値も列の値も入力されていない場合、72 コード・ワードは 6 列と 12 行のシンボルにエンコードされます。コード・ワードにより、アスペクト比率が多少異なる場合があります。
t = 右の行インジケータとストップパターンを切り捨てる	*有効値：* Y（切り捨て実行）および N （切り捨てなし） *デフォルト値：* N

例 1 • 次は PDF417 バー・コードの例です。

PDF417 BAR CODE



ZPL II CODE

```
^XA
^BY2,3
^FO10,10^B7N,5,5,,83,N
^FDZebra Technologies
Corporation strives to be
the expert supplier of
innovative solutions to
speciality demand labeling
and ticketing problems of
business and government.
We will attract and retain
the best people who will
understand our customer's
needs and provide them with
systems, hardware, software,
consumables and service
offering the best value,
high quality, and reliable
performance, all delivered
in a timely manner.
^FS^XZ
```

例 2 • 次は切り捨てを選択した場合と選択しない場合の PDF417 バー・コードの例です。

PDF417 without Truncation being selected

PDF417 with Truncation being selected

例 3 • 次の例は、フィールド 16 進（^FH）文字とともに使用した ^B7 コマンドを示しています。

ZPL II CODE	GENERATED LABEL
^XA ^FO50,50^BY3,3.0^B7N,8,5,7,21,N ^FH_^FD[)>_1E06_1DP12345678_1DQ160 _1D1JUN123456789A2B4C6D8E_1D20LA6-987 _1D21L54321 ZES_1D15KG1155 _1DBSC151208_1D7Q10GT_1E_04^FS ^XZ	

コメント 以下に箇条書きで示します。

- 列と行の両方を指定した場合、その積は 928 より小さい値である必要があります。
- 列と行の積が 928 より大きい場合はシンボルは印刷されません。
- コード・ワード合計が列と行の積より大きい場合はシンボルは印刷されません。
- このバー・コードでは連番指定はできません。
- 切り捨て機能は、ラベルの破損の可能性が少ない場合に使用できます。右行インジケータとストップ・パターンは単一のモジュール・バーの幅になります。切り捨てなしのバー・コードと切り捨てされたバー・コードの違いについては、次を参照してください。*次は切り捨てを選択した場合と選択しない場合の PDF417 バー・コードの例です。ページ 41*

PD417 使用時の ^BY に関する特別考慮事項

^B7 とともに使用する場合、^BY コマンドのパラメータは次のとおりです。

w = モジュール幅（ドット数）

*有効値：*2 ～ 10

*デフォルト値：*2

r = 比率

*固定値：*3（比率は PDF417 には影響しません)

h = バーの高さ（ドット数）

*有効値：*1 ～ 32000

*デフォルト値：*10

PDF417 は、^B7 h パラメータで行の高さが指定されていない場合にのみこれを使用します。

PD417 使用時の ^FD に関する特別考慮事項

^FD コマンドとともにプリンタに送信される文字セットには、プリンタに対して特別な意味のある文字を除き、フル ASCII セットが含まれています。

ページ番号 850 表

CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC	CHR	HEX	DEC
	20	32	0	30	48	@	40	64	P	50	80	‘	60	96	p	70	112	Ç	80	128
!	21	33	1	31	49	A	41	65	Q	51	81	a	61	97	q	71	113	ü	81	129
"	22	34	2	32	50	B	42	66	R	52	82	b	62	98	r	72	114	é	82	130
#	23	35	3	33	51	C	43	67	S	53	83	c	63	99	s	73	115	â	83	131
$	24	36	4	34	52	D	44	68	T	54	84	d	64	100	t	74	116	ä	84	132
%	25	37	5	35	53	E	45	69	U	55	85	e	65	101	u	75	117	à	85	133
&	26	38	6	36	54	F	46	70	V	56	86	f	66	102	v	76	118	å	86	134
’	27	39	7	37	55	G	47	71	W	57	87	g	67	103	w	77	119	ç	87	135
(	28	40	8	38	56	H	48	72	X	58	88	h	68	104	x	78	120	ê	88	136
)	29	41	9	39	57	I	49	73	Y	59	89	i	69	105	y	79	121	ë	89	137
*	2a	42	:	3a	58	J	4a	74	Z	5a	90	j	6a	106	z	7a	122	è	8a	138
+	2b	43	;	3b	59	K	4b	75	[	5b	91	k	6b	107	{	7b	123	ï	8b	139
,	2c	44	<	3c	60	L	4c	76	\	5c	92	l	6c	108	|	7c	124	î	8c	140
−	2d	45	=	3d	61	M	4d	77	]	5d	93	m	6d	109	}	7d	125	ì	8d	141
.	2e	46	>	3e	62	N	4e	78	^	5e	94	n	6e	110	~	7e	126	Ä	8e	142
/	2f	47	?	3f	63	O	4f	79	_	5f	95	o	6f	111	⌂	7f	127	Å	8f	143

CR と LF もすべての ^FD ステートメントに対して有効な文字です。次のスキームを使用します。

\& = キャリッジ・リターン / ライン・フィード

\\ = バックスラッシュ（\）

- バックスラッシュ（\）を印刷するには、^CI13 が選択されている必要があります。

^B8

EAN-8 バー・コード

説明 ^B8 コマンドは、EAN--13 バー・コードの短縮バージョンです。EAN は European Article Numbering の略称です。EAN-8 バー・コードの各文字は、2 本のバーと 2 本のスペースの 4 つのエレメントで構成されます。

- ^B8 は固定比率をサポートします。
- フィールド・データ（^FD）は 7 文字に制限されています。必要な文字数にするために、ZPL II は、自動的に左端でゼロを追加（パディング処理）したり切り捨てたりします。
- EAN-8 を特殊化したアプリケーションである JAN-8（Japanese Article Numbering）を使用する場合、プリンタに送信されるゼロ以外の最初の 2 桁は、常に 49 です。

フォーマット ^B8o,h,f,g

重要 • EAN-8 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』*で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f = テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は EAN-8 バー・コードの例です。

EAN-8 BAR CODE

ZPL II CODE

```
^XA
^FO100,100^BY3
^B8N,100,Y,N
^FD1234567^FS
^XZ
```

EAN-8 BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9

^B9

UPC-E バー・コード

説明 ^B9 コマンドは、記数法 0 に使用される UPC シンボル・コード体系のバリエーションを生成します。これは UPC-A バー・コードを短縮したバージョンで、ゼロを抑制することにより、所要印刷スペースが少なくなっています。6 ドット /mm、12 ドット /mm、および 24 ドット /mm の印刷ヘッドにより、UPC および EAN のシンボル・コード体系が 100% のサイズで生成されます。しかし、8 ドット /mm の印刷ヘッドの場合は、UPC および EAN のシンボル・コード体系が 77% の倍率因数で生成されます。

UPC-E バー・コードの各文字は、2 本のバーと 2 本のスペースの 4 つのエレメントで構成されます。細いバーの幅を指定するためには、^BY コマンドを使用する必要があります。

- ^B9 は固定比率をサポートします。
- フィールド・データ（^FD）は、正確に 10 文字丁度に制限されており、5 桁のメーカー・コードと 5 桁の製品コードが必要です。
- ゼロを抑制した UPC バージョンを使用する場合、全 10 文字のシーケンスを入力する必要があります。ZPL II は短縮されたバージョンを計算し、印刷します。

フォーマット ^Bo,h,f,g,e

重要 • UPC-E バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f = テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N
e = チェック・ディジットを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y

例 • 次は UPC-E バー・コードの例です。

UPC-E BAR CODE

ZPL II CODE

```
^XA
^FO150,100^BY3
^B9N,100,Y,N,Y
^FD1230000045^FS
^XZ
```

UPC-E BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9

正しい製品コード番号の規則

- メーカーの番号の最後の 3 桁が 000、100、200 のいずれかである場合、有効な製品番号は 00000 ～ 00999 です。
- メーカーの番号の最後の 3 桁が 300、400、500、600、700、800、900 のいずれかである場合、有効な製品番号は 00000 ～ 00099 です。
- メーカーの番号の最後の 2 桁が 10、20、30、40、50、60、70、80、90 のいずれかである場合、有効な製品番号は 00000 ～ 00009 です。
- メーカーの番号が 0（ゼロ）で終わらない場合、有効な製品番号は 00005 ～ 00009 です。

^BA

Code 93 バー・コード

説明 ^BA コマンドは、可変長の連続型シンボル・コード体系を作成します。Code 93 バー・コードは、Code 39 と同じアプリケーションの多くで使用されています。このコードは、128 文字のフル ASCII セットを使用します。しかし、ZPL II は ASCII コントロール・コードもエスケープ・シーケンスもサポートしていません。以下の代替文字が使用されます。

コントロール・コード	ZPL II の代替
Ctrl $	&
Ctrl %	‘
Ctrl /	(
Ctrl +	)

Code 93 バー・コードの各文字は、3 本のバーと 3 本のスペースの 6 つのエレメントで構成されます。異なる方法で呼び出されますが、テキスト表示ラインはコントロール・コードを使用したかのように印刷されます。

- ^BA は固定印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。

フォーマット ^BAo,h,f,g,e

重要 • Code 93 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f = テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N
e = チェック・ディジットを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は Code 93 バー・コードの例です。

CODE 93 BAR CODE	ZPL II CODE
	^XA ^FO100,75^BY3 ^BAN,100,Y,N,N ^FD12345ABCDE^FS ^XZ

CODE 93 BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
- . $ / + % & ’ ( )
SPACE

□ *Denotes an internal start/stop character that must precede and follow every bar code message.*

コメント すべてのコントロール・コードはペアで使用されます。

さらに、Code 93 は 128 文字のフル ASCII セットをエンコードできます。詳細については、*53 ページの表 D* を参照してください。

Code 93 のフル ASCII モード

Code 93 は、表 C と D に示した文字の組み合わせを使用して、全 128 文字の ASCII セットを生成できます。

ASCII	Code 93
NUL	‘U
SOH	&A
STX	&B
ETX	&C
EOT	&D
ENQ	&E
ACK	&F
BEL	&G
BS	&H
HT	&I
LF	&J
VT	&K
FF	&L
CR	&M
SO	&N
SI	&O
DLE	&P
DC1	&Q
DC2	&R
DC3	&S
DC4	&T
NAK	&U
SYN	&V
ETB	&W
CAN	&X
EM	&Y
SUB	&Z
ESC	‘A
FS	‘B
FS	‘C
RS	‘D
US	‘E

ASCII	Code 93
SP	Space
!	(A
“	(B
#	(C
$	(D
%	(E
&	(F
‘	(G
(	(H
)	(I
*	(J
++	++
‘	(L
-	-
.	.
/	/
0	O
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
:	(Z
;	‘F
<	‘G
=	‘H
>	‘I
?	‘J

表 D Code 93 フル ASCII モード

ASCII	Code 93
@	‘V
A	A
B	B
C	C
D	D
E	E
F	F
G	G
H	H
I	I
J	J
K	K
L	L
M	M
N	N
O	O
P	P
Q	Q
R	R
S	S
T	T
U	U
V	V
W	W
X	X
Y	Y
Z	Z
[	‘K
\	‘L
]	‘M
^	‘N
_	‘O

ASCII	Code 93
‘	‘W
a	)A
b	)B
c	)C
d	)D
e	)E
f	)F
g	)G
h	)H
I	)I
j	)J
k	)K
l	)L
m	)M
n	)N
o	)O
p	)P
q	)Q
r	)R
s	)S
t	)T
u	)U
v	)V
w	)W
x	)X
y	)Y
z	)Z
{	‘P
|	‘Q
}	‘R
~	‘S
DEL	‘T

表 E Code 93 フル ASCII モード

^BB

CODABLOCK バー・コード

説明 ^BB コマンドは、2 次元の複数行のスタック式シンボル・コード体系を生成します。このコードは、大量の情報を必要とするアプリケーションに最適です。

選択したモードにより、このコードは 1 ～ 44 のスタック式の行で構成されます。各行はスタート・パターンで開始され、ストップ・パターンで終了されます。

- CODABLOCK A は、可変印字比率をサポートしています。
- CODABLOCK E と F は固定印字比率のみをサポートしています。

フォーマット ^BBo,h,s,c,r,m

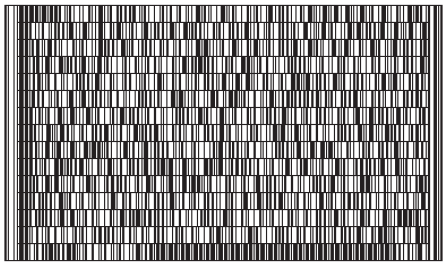
重要 • CODABLOCK バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*N
h = 個々の行のバー・コードの高さ（ドット数）	*有効値：*2 ～ 32000 *デフォルト値：*8 この数値をモジュールで掛けると、個々の行の高さのドット数になります。
s = セキュリティ・レベル	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y セキュリティ・レベルは、シンボル・チェックサムが生成されシンボルに追加されるかどうかを決定します。チェックサムは、単一行のシンボルに対して生成されることはありません。これは、パラメータ m が A に送信された場合はオフにすることができます。
c = 1 行あたりの文字数（データ列）	*有効値：*2 ～ 62 文字 これは CODABLOCK シンボルのエンコードに使用されます。このパラメータによって、シンボルの幅を制御できます。

パラメータ	詳細
r = エンコードする行の数	*有効値：* CODABLOCK A の場合：1 ～ 22 CODABLOCK E と F の場合：2 ～ 4 • c と r の値を指定しない場合、単一の行が生成されます。 • r の値を指定せず、c が最大範囲を超えている場合は、フィールド・データ長と同じ単一の行が生成されます。 • c の値を指定しない場合、1 行あたりの文字数はフィールド・データを r の値で割ることによって導かれます。 • 両方のパラメータを指定すると、フィールド・データの量は指定のパラメータの積より少なくなければなりません。フィールド・データが積の値を超えた場合、シンボルが印刷されないか、エラー・コードが印刷されます（^CV がアクティブな場合）。 • データ・フィールドに 1 次数字データが含まれる場合、指定の行数よりも少ない数の行が印刷される可能性があります。データ・フィールドにシフト文字とコード切り換え文字が含まれる場合、指定の行数よりも多い数の行が印刷される可能性があります。
m = モード	*有効値：*A、E、F CODABLOCK A は、Code 39 文字セットを使用します。 CODABLOCK F は、Code 128 文字セットを使用します。 CODABLOCK E は、Code 128 文字セットを使用し、自動的に FNC1 を追加します。 *デフォルト値：*F

例 • 次は CODABLOCK バー・コードの例です。

CODABLOCK BAR CODE	ZPL II CODE
	^XA ^BY2,3 ^FO10,10^BBN,30,,30,44,E ^FDZebra Technologies Corporation strives to be the expert supplier of innovative solutions to speciality demand labeling and ticketing problems of business and government. We will attract and retain the best people who will understand our customer's needs and provide them with systems, hardware, software, consumables and service offering the best value, high quality, and reliable performance, all delivered in a timely manner.^FS ^XZ

^BB 使用時の ^BY コマンドに関する特別考慮事項

^BB コードとともに使用する場合の ^BYw,r,h コマンドのパラメータは次のとおりです。

w = モジュール幅（ドット数）

*有効値：*2 ～ 10（CODABLOCK A のみ）

*デフォルト値：*2

r = 比率

固定値：3（比率は CODABLOCK E または F には影響しません）

h = バーの高さ（ドット数）

*有効値：*1 ～ 32.32000

*デフォルト値：*10

CODABLOCK は、^BB h パラメータで行の高さが指定されていない場合にのみ、これをシンボル全体の高さとして使用します。

^BB 使用時の ^FD 文字セットに関する特別考慮事項

プリンタに送信される文字セットは、パラメータ m で選択されているモードによって異なります。

CODABLOCK A : CODABLOCK A は、Code 39 と同じ文字セットを使用します。^FD ステートメントでその他の文字セットを使用した場合、バー・コードが印刷されないか、エラー・コードが印刷されます（^CV がアクティブな場合）。

CODABLOCK E : 自動モードには、プリンタに対して特別な意味のある文字を除いた フル ASCII セットが含まれます。機能コード または Code 128 サブセット A <nul> 文字は、^FH コマンドを使用して挿入できます。

```
<fnc1> = 80 hex  <fnc3> = 82 hex
<fnc2> = 81 hex  <fnc4> = 83 hex
<nul> = 84 hex
```

84 hex より上の文字の場合、バー・コードが印刷されないか、エラー・コードが印刷されます（^CV がアクティブな場合）。

CODABLOCK F : CODABLOCK F は、プリンタに対して特別な意味のある文字を除いた フル ASCII セットを使用します。機能コード または Code 128 サブセット A <nul> 文字は、^FH コマンドを使用して挿入できます。

```
<fnc1> = 80 hex  <fnc3> = 82 hex
<fnc2> = 81 hex  <fnc4> = 83 hex
<nul> = 84 hex
```

^BC

Code 128 バー・コード（サブセット A、B、C）

説明 ^BC コマンドは、高密度の可変長連続型英数字シンボル・コード体系である Code 128 バー・コードを生成します。このバー・コードは、複雑なエンコードの製品識別のために設計されたものです。

Code 128 には文字のサブセットが 3 つあります。各セットにはエンコードされた印刷用文字が 106 文字あり、使用する文字のサブセットにより、各文字には最高 3 つまで意味を持たせることができます。各 Code 128 文字は、3 本のバーと 3 本のスペースの 6 エレメントで構成されます。

- ^BC は固定印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。

フォーマット ^BCo,h,f,g,e,m

重要 • Code 128 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f = テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y テキスト表示ラインは、バー・コード・コマンドの前にフォント・コマンドを配置することにより、どのようなフォントによっても印刷できます。
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N
e = UCC チェック・ディジット	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

パラメータ	詳細
m = モード	*有効値：* N = モードの選択なし U = UCC Case モード A = 自動モード送信されたデータを分析し、最適な圧縮方法を自動的に判断します。^FD ステートメントでフル ASCII 文字セットを使用できます。すなわち、サブセットのシフトはプリンタによって決定されます。4 桁以上の数値ストリングの場合、サブセット C に自動的にシフトされます。 *デフォルト値：*N
d = データ挿入	

例 • 次は Code 128 バー・コードの例です。

CODE 128 BAR CODE	ZPL II CODE
	^XA ^FO100,100^BY3 ^BCN,100,Y,N,N ^FD123456^FS ^XZ

Value	Code A	Code B	Code C
0	SP	SP	00
1	!	!	01
2	"	"	02
3	#	#	03
4	$	$	04
5	%	%	05
6	&	&	06
7	'	'	07
8	(	(	08
9	)	)	09
10	*	*	10
11	++	++	11
12	,	,	12
13	-	-	13
14	.	.	14
15	/	/	15
16	0	0	16
17	1	1	17
18	2	2	18
19	3	3	19
20	4	4	20
21	5	5	21
22	6	6	22
23	7	7	23
24	8	8	24
25	9	9	25
26	:	:	26
27	;	;	27
28	<	<	28
29	=	=	29
30	>	>	30
31	?	?	31
32	@	@	32
33	A	A	33
34	B	B	34
35	C	C	35
36	D	D	36
37	E	E	37
38	F	F	38
39	G	G	39
40	H	H	40
41	I	I	41
42	J	J	42
43	K	K	43
44	L	L	44
45	M	M	45
46	N	N	46
47	O	O	47
48	P	P	48
49	Q	Q	49
50	R	R	50
51	S	S	51
52	T	T	52

Value	Code A	Code B	Code C
53	U	U	53
54	V	V	54
55	W	W	55
56	X	X	56
57	Y	Y	57
58	Z	Z	58
59	[	[	59
60	\	\	60
61	]	]	61
62	^	^	62
63	_	_	63
64	NUL	`	64
65	SOH	a	65
66	STX	b	66
67	ETX	c	67
68	EOT	d	68
69	ENQ	e	69
70	ACK	f	70
71	BEL	g	71
72	BS	h	72
73	HT	i	73
74	LF	j	74
75	VT	k	75
76	FF	l	76
77	CR	m	77
78	SO	n	78
79	SI	o	79
80	DLE	p	80
81	DC1	q	81
82	DC2	r	82
83	DC3	s	83
84	DC4	t	84
85	NAK	u	85
86	SYN	v	86
87	ETB	w	87
88	CAN	x	88
89	EM	y	89
90	SUB	z	90
91	ESC	{	91
92	FS	|	92
93	GS	}	93
94	RS	~	94
95	US	DEL	95
96	FNC3	FNC3	96
97	FNC2	FNC2	97
98	SHIFT	SHIFT	98
99	Code C	Code C	99
100	Code B	FNC4	Code B
101	FNC4	Code A	Code A
102	FNC1	FNC1	FNC1
103		*START (Code A)*	
104		*START (Code B)*	
105		*START (Code C)*	

表 F Code 128 の文字セット

UCC Case モードを選択した場合の特殊条件

- ^FD または ^SN で 19 桁を超えるデータは削除されます。
- ^FD または ^SN のデータが 19 桁以下の場合は、右端にゼロを加えて 19 桁になるようにします。これにより、無効なテキスト表示ラインが生成されます。

Code 128 のサブセット

Code 128 の文字サブセットは、サブセット A、サブセット B、サブセット C と呼ばれます。サブセットは次の方法で選択できます。

- バー・コードに関連付けられるフィールド・データ（^FD）ストリングに特殊な呼び出しコードを含めることができます。
- 希望のスタート・コードをフィールド・データの先頭に配置できます。スタート・コードを入力しない場合は、サブセット B が使用されます。

バー・コード内でサブセットを変更するには、フィールド・データ（^FD）ストリング内の適切な位置に呼び出しコードを配置します。新しいサブセットは、呼び出しコードで変更されるまで有効です。たとえば、サブセット C で、フィールド・データに >7 を含めるとサブセットが A に変更されます。

次の表には、これら 3 つのサブセットの Code 128 呼び出しコードとスタート文字が示されています。

Invocation Code	Decimal Value	Subset A Character	Subset B Character	Subset C Character
><	62			
>0	30	>	>	
>=	94		~	
>1	95	USQ	DEL	
>2	96	FNC 3	FNC 3	
>3	97	FNC 2	FNC 2	
>4	98	SHIFT	SHIFT	
>5	99	CODE C	CODE C	
>6	100	CODE B	FNC 4	CODE B
>7	101	FNC 4	CODE A	CODE A
>8	102	FNC 1	FNC 1	FNC 1
Start Characters				
>9	103	Start Code A	(Numeric Pairs give Alpha/Numerics)	
>:	104	Start Code B	(Normal Alpha/Numeric)	
		Start Code C		

表 G Code 128 呼び出し文字

例 • 次の図 A と B は、同一のバー・コードの例です。


```
^XA
^FO100,75
^BCN,100,Y,N,N
^FDCODE128^FS
^XZ
```

Figure A: Subset B with no start character


```
^XA
^FO100,75
^BCN,100,Y,N,N
^FD>:CODE128^FS
^XZ
```

Figure B: Subset B with start character

Code 128 サブセットは最もよく使用されるサブセットであるため、ZPL II ではデータ・ストリングでスタート文字が指定されていない場合、デフォルトとしてサブセット B を使用します。

ZPL II スクリプト内での ^BC

^XA - ラベル・フォーマットを開始する最初のコマンドです。

^FO100,75 - この 2 番目のコマンドは、フィールド基点を左上のコーナーから X 軸上 100 ドット、Y 軸上 75 ドットの位置に設定します。

^BCN,100,Y,N,N - この 3 番目のコマンドは、Code 128 バー・コードを呼び出し、回転なし（N）、高さ 100 ドットで印刷します。テキスト表示ラインは、バー・コードの下（N）に印刷されます（Y）。UCC チェック・ディジットは使用されません（N）。

^FDCODE128^FS（図 A）^FD>:CODE128^FS（図 B）- フィールド・データ・コマンドによりバー・コードの内容が指定されます。

^XZ - 最後のこのコマンドは、フィールド・データを終了し、ラベルの終わりを意味します。

テキスト表示ラインは、チェック・ディジットをオフにしてコードの下に印刷されます。

図 A の ^FD コマンドは、サブセットを指定していないため、サブセット B が使用されます。図 B では、^FD コマンドで >: スタート・コードを使用してサブセット B が呼び出されています。ZPL II はコード B をデフォルトとして使用しますが、コマンドには呼び出しコードを含める習慣をつけることをお勧めします。

Code 128 - サブセット B は、10 進数の 94 より大きい値と、呼び出しコードを使用してプログラムする必要のある一部の特殊文字を除き、ASCII テキストとして直接プログラムされます。それらの特殊文字には以下があります。

^ > ~

例 • Code 128 – サブセット A と C

Code 128 のサブセット A と C は、フィールド・データ・ストリング内の 2 桁の値（00 ～ 99）でプログラムされます。詳細については、*64 ページの表 F* を参照してください。

サブセット A では、2 桁の値により単一の文字がバー・コードにエンコードされ、サブセット C では文字は入力したとおりに印刷されます。下の図 E はサブセット A の例です（>9 がサブセット A のスタート・コードです）。

2 桁（D2）の最初の文字としてプログラムされた非整数値は無視されます。2 桁（2D）の 2 番目の文字としてプログラムされた非整数値は、2 桁すべてを無効にするため、このペアは無視されます。フィールド・データ・ストリング内でコード・シフトのすぐ前に余分な単独の桁がある場合は、それも無視されます。

下の図 C と 図 D は、サブセット C の例です。バー・コードが同一であることに注意してください。図 D のプログラム・コードでは、*D* が無視され、2 が 4 と組み合わされています。


```
^XA
^FO100,75^BY3
^BCN,100,Y,N,N
^FD>;382436^FS
^XZ
```

Figure C: Subset C with normal data


```
^XA
^FO100,75^BY3
^BCN,100,Y,N,N
^FD>;38D2436^FS
^XZ
```

Figure D: Subset C with ignored alpha character

Figure E: Subset A

^BD

UPS MaxiCode バー・コード

説明 ^BD コマンドは、2 次元の光学式読み取り（スキャン式ではない）コードを生成します。このシンボル・コード体系は、UPS（United Parcel Service）によって開発されました。

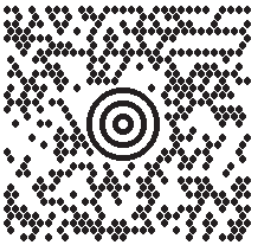
このコードにはその他のパラメータはなく、テキスト表示ラインも生成されないことに注意してください。^BY コマンドは UPS Maxi Code バー・コードには影響しません。しかし、^CV コマンドをアクティブにすることはできます。

フォーマット ^BDm,n,t

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
m = モード	*有効値：* 2 = 構造化された宅配業者メッセージ：数字の郵便番号（米国式） 3 = 構造化された宅配業者メッセージ：英数字の郵便番号（非米国式） 4 = 標準シンボル、セクレタリ 5 = フル EEC 6 = リーダー・プログラム、セクレタリ *デフォルト値：*2
n = シンボル番号	*有効値：*1 ～ 8 を構造化文書に追加できます。 *デフォルト値：*1
t = シンボルの合計数	*有効値：*このシーケンス内のシンボルの合計数を表す 1 ～ 8 *デフォルト値：*1

例 • 次は UPS MAXICODE - Mode 2 バー・コードの例です。

UPS MAXICODE - MODE 2	ZPL II CODE
	^XA ^FO50,50 ^CVY ^BD^FH^FD001840152382802 [)>_1E01_1D961Z00004951_1DUPSN_ 1D_06X610_1D159_1D1234567_1D1/1_ 1D_1DY_1D634 ALPHA DR_ 1DPITTSBURGH_1DPA_1E_04^FS ^FO30,300^A0,30,30^FDMode2^FS ^XZ

^BD 使用時の ^FD に関する特別考慮事項

^FD ステートメントは高優先度メッセージ(hpm)と低優先度メッセージ(lpm)の 2 つの部分に分けられます。高優先度のメッセージには 2 種類あります。その 1 つは米国式の郵便番号で、もう 1 つは非米国式の郵便番号です。これらの高優先度メッセージの構文は、どちらもここに示したものと正確に一致しなければ、エラー・メッセージが生成されます。

フォーマット ^FD <hpm><lpm>

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
<hpm> = 高優先度メッセージ（モード 2 と 3 の場合のみに適用可能）	*有効値：*0 ～ 9（別の指定がない場合） **米国式郵便番号（モード 2）** **<hpm>** = aaabbbccccddddd aaa = 3 桁のサービス・クラス bbb = 3 桁の国番号 ccccc = 5 桁の郵便番号 dddd = 4 桁の郵便番号拡張子（ない場合はゼロを 4 つ（0000）入力） **非米国式郵便番号（モード 3）** **<hpm>** = aaabbbccccc aaa = 3 桁のサービス・クラス bbb = 3 桁の国番号 ccccc = 6 桁の郵便番号（A ～ Z、または 0 ～ 9）
<lpm> = 低優先度メッセージ（モード 2 と 3 の場合のみに適用可能）	GS は、メッセージのフィールドを区切るために使用します(0x1D)。RS はフォーマット・タイプを区切るために使用します（Ox1E)。EOT は送信の終わりを示す文字です。 メッセージ・ヘッダー []>RS 輸送データ

フォーマット・ヘッダー	01GS96
管理番号 *	< 管理番号 >
SCAC*	GS<SCAC>
UPS 荷送人番号	GS< 荷送人番号 >
荷物収集日（ユリウス日）	GS< 荷物収集日 >
発送 ID 番号	GS< 発送 ID 番号 >
パッケージ n/x	GS<n/x>
パッケージ重量	GS< 重量 >
住所確認	GS< 確認 >
届け先住所	GS< 番地 >
届け先所在市	GS< 市 >
届け先所在州	GS< 州 >
RS	RS
メッセージの終わり	EOT
(* UPS の必須データ)	

コメント

- **<hpm>** と **<lpm>** のフォーマットは、モード 2 と 3 を使用する場合のみに適用されます。
  たとえば、モード 4 では ^FD コマンドで定義されたデータがシンボル内に配置されます。
- UPS は、特定のデータが定義した方法で存在することを必要とします。UPS の MaxiCode データをフォーマットする際は、必ず上段文字を使用してください。UPS の **<lpm>** で *フィールド* に入力する際は、『Guide to Bar Coding with UPS』で指定されたデータのサイズとタイプに従ってください。
- モードを選択しない場合、デフォルトはモード 2 です。非米国式の郵便番号を使用する場合、エラー・メッセージ（無効な文字やメッセージが短すぎるなど）が表示される可能性があります。非米国式のコードを使用する場合はモード 3 を使用してください。
- ZPL II は、郵便番号のフォーマットに基づいて自動的にモードを変更しません。
- GS、RS、EOT などの特殊文字を使用する場合は、^FH コマンドを使用して ZPL II で下線 (_) の後に 16 進の値を使用するように指示してください。

^BE

EAN-13 バー・コード

説明 ^BE コマンドは、UPC-A バー・コードと同様です。このコードは、ヨーロッパと日本の小売市場で広く使用されています。

EAN-13 バー・コードには 12 のデータ文字が含まれ、これは UPC-A コードよりも 1 つ多い文字数です。EAN-13 シンボルには、UPC-A と同じ数のバーが含まれますが、13 番目の桁を左端の 6 桁のパリティ・パターンにエンコードします。この 13 番目の桁は、12 番目の桁と組み合わせて国コードを表します。

- ^BE は固定印字比率をサポートします。
- フィールド・データ（^FD）は正確に 12 文字に制限されています。必要な文字数にするために、ZPL II は、自動的に左端でゼロを切り捨てたり追加（パディング処理）したりします。
- EAN-13 を特殊化したアプリケーションである JAN-13（Japanese Article Numbering）を使用する場合、プリンタに送信されるゼロ以外の最初の 2 桁は、常に 49 である必要があります。

フォーマット ^BEo,h,f,g

重要 • EAN-13 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f = テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は EAN-13 バー・コードの例です。

EAN-13 BAR CODE

ZPL II CODE

```
^XA
^FO100,100^BY3
^BEN,100,Y,N
^FD12345678^FS
^XZ
```

EAN-13 BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9

コメント EAN-13 バー・コードはエラー・チェックにモジュラス 10 チェック・ディジット・スキームを使用します。モジュラス 10 の詳細については、*『ZPL II プログラミング・ガイド第 2 巻』*の付録 C を参照してください。

^BF

Micro-PDF417 バー・コード

説明 ^BF コマンドは、PDF417 と同じ 2 次元の複数行連続型スタック式シンボル・コード体系を作成しますが、17 モジュール幅のスタート / ストップ・パターンと左 / 右の行インジケータを 10 モジュール幅の行のアドレス・パターンの一意なセットで置換します。これにより、シンボル全体の幅が狭まり、行の高さが 2X でも線形スキャンが可能になります。

Micro PDF417 は、エリア効率の改善を要するアプリケーション用に設計されていますが、PDF417 で要求される最大データ容量はありません。最大 4 データ列 x 44 行までの特定の行と列の組み合わせでのみ印刷できます。

フィールド・データ（^FD）とフィールド 16 進値（^FH）は次のように制限されています。

- 250 の 7 ビット文字
- 150 の 8 ビット文字
- 366 の 4 ビット数字

フォーマット ^BFo,h,m

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値 :* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値 :* 現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値 :* 1 ～ 9999 *デフォルト値 :* ^BY によって設定された値、または 10（^BY 値が存在しない場合）
m = モード	*有効値 :* 0 ～ 33（*79 ページの表 H*） *デフォルト値 :* 0 (*79 ページの表 H*).

例 • 次は MICRO-PDF417 バー・コードの例です。

MICRO-PDF417 BAR CODE	ZPL II CODE
	^XA ^FO100,100^BY6 ^BFN,8,3 ^FDABCDEFGHIJKLMNOPQRSTUV^FS ^XZ

データを Micro-PDF417 バー・コードにエンコードする手順

1. エンコードするデータのタイプを決定します（たとえば、ASCII 文字、数字、8 ビット・データ、これらの組み合わせなど）。
2. バー・コード内でエンコードするデータの最大量を決定します（たとえば、ASCII 文字の数、数字の数、8 ビット・データ文字の数など）。
3. バー・コード内で使用するチェック・ディジットのパーセントを決定します。使用するチェック・ディジットのパーセントが高いほど、バー・コードは耐破損性が高くなりますが、サイズは大きくなります。

4 上記の質問から得た情報と *79 ページの表 H* を使用して、バー・コードのモードを選択します。

Mode (M)	Number of Data Columns	Number of Data Rows	% of Cws for EC	Max Alpha Characters	Max Digits
0	1	11	64	6	8
1	1	14	50	12	17
2	1	17	41	18	26
3	1	20	40	22	32
4	1	24	33	30	44
5	1	28	29	38	55
6	2	8	50	14	20
7	2	11	41	24	35
8	2	14	32	36	52
9	2	17	29	46	67
10	2	20	28	56	82
11	2	23	28	64	93
12	2	26	29	72	105
13	3	6	67	10	14
14	3	8	58	18	26
15	3	10	53	26	38
16	3	12	50	34	49
17	3	15	47	46	67
18	3	20	43	66	96
19	3	26	41	90	132
20	3	32	40	114	167
21	3	38	39	138	202
22	3	44	38	162	237
23	4	6	50	22	32
24	4	8	44	34	49
25	4	10	40	46	67
26	4	12	38	58	85
27	4	15	35	76	111
28	4	20	33	106	155
29	4	26	31	142	208
30	4	32	30	178	261
31	4	38	29	214	313
32	4	44	28	250	366
33	4	4	50	14	20

表 H Micro-PDF417 モード

^BI

Industrial 2 of 5 バー・コード

説明 ^BI コマンドは、ディスクリートの自己チェック式連続型数値シンボル・コード体系です。Industrial 2 of 5 バー・コードは、2 of 5 系列のバー・コードでは最も長く使用されています。この系列のバー・コードの中では Standard 2 of 5（^BJ）と Interleaved 2 of 5（^B2) バー・コードも ZPL II で使用できます。

Industrial 2 of 5 では、すべての情報がバーに格納されます。このコードでは 2 種類のバー幅が採用されており、太いバーは細いバーの 3 倍の幅です。

- ^BI は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。

フォーマット ^BIo,h,f,g

重要 • Industrial 2 of 5 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f = テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は Industrial 2 of 5 バー・コードの例です。

INDUSTRIAL 2 OF 5 BAR CODE

ZPL II CODE

```
^XA
^FO100,100^BY3
^BIN,150,Y,N
^FD123456^FS
^XZ
```

INDUSTRIAL 2 OF 5 BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9

Start/Stop (internal)

^BJ

Standard 2 of 5 バー・コード

説明 ^BJ コマンドは、ディスクリートの自己チェック式連続型数値シンボル・コード体系です。

Standard 2 of 5 では、すべての情報がバーに格納されます。このコードでは 2 種類のバー幅が採用されており、太いバーは細いバーの 3 倍の幅です。

- ^BJ は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。

フォーマット ^BJo,h,f,g

重要 • Standard 2 of 5 バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値

パラメータ	詳細
f ＝テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g ＝テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は Standard 2 of 5 バー・コードの例です。

STANDARD 2 OF 5 BAR CODE

ZPL II CODE

```
^XA
^FO100,100^BY3
^BJN,150,Y,N
^FD123456^FS
^XZ
```

STANDARD 2 OF 5 BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9

Start/Stop (automatic)

^BK

ANSI Codabar バー・コード

説明 ANSI Codabar バー・コードは、図書館、医療産業、宅配便会社などのさまざまな情報処理アプリケーションに使用されています。このバー・コードは、USD-4 Code、NW-7、および 2 of 7 Code としても知られています。当初は小売店の価格ラベル用に開発されました。

このコードの各文字は、4 本のバーと 3 本のスペースの 7 エレメントで構成されます。Codabar バー・コードでは、数値とコントロール（スタート / ストップ）の 2 つの文字セットが使用されます。

- ^BK は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。

フォーマット ^BKo,e,h,f,g,k,l

重要 • Codabar バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o ＝向き	*有効値：* N ＝ 標準 R ＝ 90° 回転（時計方向） I ＝ 180° 反転 B ＝ 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
e＝チェック・ディジット	*固定値：*N
h＝バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f ＝テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g ＝テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N
k ＝スタート文字を指定	*有効値：*A、B、C、D *デフォルト値：*A
l ＝ストップ文字を指定	*有効値：*A、B、C、D *デフォルト値：*A

例 • 次は ANSI Codabar バー・コードの例です。

ANSI CODABAR BAR CODE	ZPL II CODE
	^XA ^FO100,100^BY3 ^BKN,N,150,Y,N,A,A ^FD123456^FS ^XZ

ANSI CODABAR BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9

Control Characters

- : . $ / +

Start/Stop Characters

A B C D

^BL

LOGMARS バー・コード

説明 ^BL コマンドは、国防総省で使用されている Code 39 の特殊なアプリケーションです。LOGMARS は、Logistics Applications of Automated Marking and Reading Symbols の略語です。

- ^BL は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。^FD ストリングの下段の文字は、サポートされる上段の LOGMARS 文字に変換されます。

フォーマット ^BLo,h,g

重要 • LOGMARS バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値

パラメータ	詳細
h＝バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
g ＝テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は LOGMARS バー・コードの例です。

LOGMARS BAR CODE

ZPL II CODE

```
^XA
^FO100,75^BY3
^BLN,100,N
^FD12AB^FS
^XZ
```

LOGMARS BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
- . $ / + %
SPACE

コメント LOGMARS バー・コードは、モジュラス 43 の計算を使用して*必須の*チェック・ディジットを生成します。モジュラス 43 の詳細については、*『ZPL II プログラミング・ガイド第 2 巻』*の付録 D を参照してください。

^BM

MSI バー・コード

説明 ^BM コマンドはパルス幅変調式、連続型、および自己チェック機能なしのシンボル・コード体系で、Plessey バー・コード（^BP）の変形です。

MSI バー・コードの各文字は、4 本のバーと隣接する 4 本のスペースの 8 エレメントで構成されます。

- ^BM は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- パラメータ e が B、C、または D の場合、バー・コードが有効であるためには、フィールド・データ（^FD）は 1 ～ 14 桁に制限されます。パラメータ e が A の場合、^FD は 1 ～ 13 桁にクワイエット・ゾーンを加えたものに制限されます。

フォーマット ^BMo,e,h,f,g,e2

重要 • MSI バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
e = チェック・ディジット選択	*有効値：* e = チェック・ディジットなし B = 1 モジュラス 10 C = 2 モジュラス 10 D = 1 モジュラス 10 と 1 モジュラス 11 *デフォルト値：*B
h = バー・コードの高さ（ドット数）	*有効値：* e = チェック・ディジットなし B = 1 モジュラス 10 C = 2 モジュラス 10 D = 1 モジュラス 10 と 1 モジュラス 11 *デフォルト値：*B
f = テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N
e2 = スタート文字を指定	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は MSI バー・コードの例です。

MSI BAR CODE	ZPL II CODE
	^XA ^FO100,100^BY3 ^BMN,B,100,Y,N,N ^FD123456^FS ^XZ

MSI BAR CODE CHARACTERS
1 2 3 4 5 6 7 8 9

^BP

Plessey バー・コード

説明 ^BP コマンドはパルス幅変調式、連続型、および自己チェック機能なしのシンボル・コード体系です。

Plessey バー・コードの各文字は、4 本のバーと隣接する 4 本のスペースの 8 エレメントで構成されます。

- ^BP は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。

フォーマット ^BPo,e,h,f,g

重要 • Plessey バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』*で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
e = チェック・ディジットを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

パラメータ	詳細
h＝バー・コードの高さ（ドット数）	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N
f ＝テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g ＝テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次は Plessey バー・コードの例です。

PLESSEY BAR CODE

ZPL II CODE

```
^XA
^FO100,100^BY3
^BPN,N,100,Y,N
^FD12345^FS
^XZ
```

PLESSEY BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9
A B C D E F

^BQ

QR Code バー・コード

説明 ^BQ コマンドは、正方形のパターンに並べた正方形のモジュールで構成されるマトリックス・シンボル・コード体系を生成します。4 つの角のうち 3 つに見られる一意なパターンにより、バー・コードのサイズ、位置、および傾斜が決まります。

4 つのレベルのエラー訂正機能とともに、広範囲に及ぶシンボル・サイズが実現しました。ユーザー指定によるモジュール寸法により、多様なシンボル作成技法を使用できます。

QR Code Model 1 は最初の仕様でしたが、QR Code Model 2 はその拡張バージョンです。Model 2 には追加機能が備わったほか、Model 1 と自動的に区別されます。

Model 2 が推奨されるモデルであり、通常はこちらを使用してください。

このバー・コードは、後続の ^FD ストリングで指定されるフィールド・データを使用して印刷されます。

エンコード可能な文字セットには、数字データ、英数字データ、8 ビット・バイト・データ、および漢字が含まれます。

フォーマット ^BQa,b,c

重要 • QR code バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = フィールド位置	*固定値：*標準（^FW は回転には影響しません）
b = モデル	*有効値：*1（最初のバージョン）および 2 （拡張バージョン - 推奨） *デフォルト値：*2
c = 倍率係数	*有効値：*1 ～ 10 *デフォルト値：* 1150 dpi プリンタの場合 2200 dpi プリンタの場合 3300 dpi プリンタの場合 6600 dpi プリンタの場合

例 • 次は QR Code バー・コードの例です。

QR CODE BAR CODE	ZPL II CODE
	^XA ^FO100,100 ^BQN,2,10 ^FDMM,AAC-42^FS ^XZ

後続のページでは、エンコードする情報を含む ^FD ステートメントでコマンドをフォーマットするための具体的なコマンドを紹介します。

QR コマンド使用時の ^FD に関する考慮事項

QR スイッチ（^FD フィールド・データにフォーマットされる）

混合モード <D>

D = 異なるタイプの文字モードを 1 つのコードに構成します。

コード番号 <01 16>

値 = 分割されたコードの N 番目の数値から差し引きます（2 桁）。

区分の数 <02 16>

区分の数（2 桁）。

パリティ・データ <1 バイト >

パリティ・データ値は、入力データ（EX-OR 操作でバイトごとに分割される前の最初の入力データ）で計算して取得します。

エラー訂正レベル <H、Q、M、L>

H = 極めて高い信頼性レベル

Q = 高信頼性レベル

M = 標準レベル（デフォルト）

L = 高密度レベル

文字モード <N、A、B、K>

N = 数字

A = 英数字

Bxxxx = 8 ビット・バイト・モードこのモードは、JIS X 0201（文字値 0x00 ～ 0xFF）に従って、8 ビットのラテン / 仮名文字セットを処理します。

xxxx = データ文字の数は 2 バイトの BCD コードで表わされます。

K = 漢字 -- JIS X 0208 に基づき Shift JIS システムに従って漢字のみを処理します。これは、文字モード *K* の後のすべてのパラメータは 16 ビットの文字である必要があることを意味します。8 ビットの文字（ASCII コードなど）があるとエラーが発生します。

データ文字ストリング < データ >

文字モードの後に配置されるか、^FD ステートメントの最後のスイッチです。

データ入力 <A, M>

A = 自動入力（デフォルト）データ文字ストリング JIS8 単位、Shift JIS。入力モードが自動入力の場合、0x89 ～ 0x9F、および 0xE0 ～ 0xFF のバイナリ・コードを設定することはできません。

M = 手動入力

データ入力モードには、自動（A）と手動（M）の 2 種類があります。A を指定した場合、文字モードを指定する必要はありません。M を指定した場合、文字モードを指定する必要があります。

^FD フィールド・データ（標準モード）

スイッチ付き自動データ入力（A）

```
^FD
< エラー訂正レベル >A
< データ文字ストリング >
^FS
```

例 • 自動データ入力による標準モードの QR Code

```
^XA
^FO20,20^BQ,2,10^FDQA,0123456789ABCD 2D
code^FS
^XZ
```

< エラー訂正レベル >	Q	（高）
< 入力モード >	A,	（自動設定）
< データ文字ストリング >		0123456789ABCD 2D code

スイッチ付き手動データ入力（M）

```
^FD
< エラー訂正レベル >M,
< 文字モード >< データ文字ストリング >
^FS
```

例 • 手動データ入力による標準モードの QR Code

```
^XA
^FO20,20^BQ,2,10
^FDHM,N123456789012345^FS
^XZ
```

＜エラー訂正レベル＞	H	（極めて高い信頼性レベル）
＜入力モード＞	M,	（手動入力）
＜文字モード＞	N	（数字データ）
＜データ文字ストリング＞		123456789012345

例 • 標準信頼性レベル、手動データ入力による標準モードの QR Code

```
^XA
^FO20,20^BQ,2,10^FDMM,AAC-42^FS
^XZ
```

＜エラー訂正レベル＞	M	（標準信頼性レベル）
＜入力モード＞	M,	（手動入力）
＜文字モード＞	A	（英数字データ）
＜データ文字ストリング＞		AC-42

^FD フィールド・データ（混合モード - 追加のスイッチが必要）

スイッチ付き自動データ入力（A）

```
^FD
<D>< = コード番号 >< 区分の数 > ＜パリティ・データ >,
＜エラー訂正レベル >A
＜データ文字ストリング >,
＜データ文字ストリング >,
＜  :  >,
＜データ文字ストリング  n**>
^FS
```

スイッチ付き手動データ入力（M）

```
^FD
＜コード番号 > ＜区分の数 > ＜パリティ・データ >,
```

```
< エラー訂正レベル >M,
< 文字モード  1>< データ文字ストリング  1>,
< 文字モード  2>< データ文字ストリング  2>,
< :   > <   :   >,
< 文字モード  n>< データ文字ストリング  n**>
^FS
```

混合モードで n** 最高 200 まで

例 • 手動データ入力による混合モードの QR Code

```
^XA
^FO,20,20^BQ,2,10
^FDD03048F,LM,N0123456789,A12AABB,B0006qrc
ode^FS
^XZ
```

< 混合モード識別子 >	D	(混合)
< コード番号 >	M	(コード番号)
< 区分の数 >	D	(区分)
< パリティ・データ >	M	(0x8F)
	‘	
< エラー訂正レベル >	L	(高密度レベル)
< 入力モード >	M	(手動入力)
	‘	
< 文字モード >	N	(数字データ)
< データ文字ストリング >		0123456789
	‘	
< 文字モード >	A	(英数字データ)
< データ文字ストリング >		12AABB
	‘	

< 文字モード >	B	(8 ビット・バイト・データ)
	0006	(バイト数)
< データ文字ストリング >		qrcode

例 • 次は、自動データ入力による混合モードの QR Code の例です。

```
^XA
^FO20,20^BQ,2,10
^FDD03048F,LA,012345678912AABBqrcode^FS
^XZ
```

< 混合モード識別子 >	D	(混合)
< コード番号 >	M	(コード番号)
< 区分の数 >	D	(区分)
< パリティ・データ >	M	(0x8F)
< エラー訂正レベル >	L	(高密度レベル)
< 入力モード >	A	(自動入力)
< データ文字ストリング >		012345678912AABBqrcode

^BR

RSS バー・コード

説明 このコードは、*XiIII*Plus プリンタの CompactFLASH 専用です。

フォーマット ^BRa,b,c,d,e,f

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 向き	*有効値：* N = 標準 R = 回転 I = 反転 B = 下から上 *デフォルト値：*R
b = RSS-14 ファミリのシンボル・コード体系タイプ	*有効値：* 1 = RSS14 2 = RSS14 Truncated 3 = RSS14 Stacked 4 = RSS14 Stacked Omnidirectional 5 = RSS Limited 6 = RSS Expanded 7 = UPC-A 8 = UPC-E 9 = EAN-13 10 = EAN-8 11 = UCC/EAN-128 および CC-A/B 12 = UCC/EAN-128 および CC-C *デフォルト値：*1

パラメータ	詳細
c = 倍率係数	*有効値 :* 1 ～ 10 *デフォルト値 :* 24 ドットは 6、12 ドットは 3、8 ドット以下は 2 12 ドットは 6、8 ドットは 3、8 ドット以下は 2
d = セパレータの高さ	*有効値 :* 1 または 2 *デフォルト値 :* 1
e = バー・コードの高さ	バー・コードの高さは、バー・コードの線形部分にのみ影響します。UCC/EAN および CC-A/B/C のみ。 *有効値 :* 1 ～ 32000 ドット *デフォルト値 :* 25
f = セグメント幅 (RSS expanded のみ)	*有効値 :* 2 ～ 22 (偶数のみ、ラインあたりのセグメント数) *デフォルト値 :* 22

演習 1 次はシンボル・コード体系 Type 7 - UPC-A の例です。

```
^XA

^FO10,10^BRN,7,5,2,100^FD12345678901|this is
composite info^FS

^XZ
```

例 • 次はシンボル・コード体系 Type 1 ñ RSS14 の例です。

```
^XA

^FO10,10^BRN,1,5,2,100^FD12345678901|this is
composite info^FS

^XZ
```

^BS

拡張 UPC/EAN

説明 ^BS コマンドは、ISBN（International Standard Book Numbers）のバー・コードを作成するために主に出版社で使用される 2 桁と 5 桁のアドオンです。これらの拡張コードは、別個のバー・コードとして扱われます。

^BS コマンドは、UPC-A バー・コード（^BU）と UPC-E バー・コード（^B9）と一緒に使用するように設計されています。

- ^BS は固定印字比率をサポートします。
- フィールド・データ（^FD）は 2 文字丁度か 5 文字丁度に制限されています。必要な文字数にするために、ZPL II は、自動的に左端でゼロを切り捨てたり追加（パディング処理）したりします。

フォーマット ^BSo,h,f,g

重要 • UPC/EAN バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値：*1 ～ 32000 *デフォルト値：*^BY によって設定された値
f = テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g = テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y

例 • 次は UPC/EAN の 2 桁のバー・コードの例です。

UPC/EAN 2-DIGIT BAR CODE

ZPL II CODE

```
^XA
^FO100,100^BY3
^BSN,100,Y,N
^FD12^FS
^XZ
```

UPC/EAN 2-DIGIT BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9

例 • 次は UPC/EAN の 5 桁のバー・コードの例です。

UPC/EAN 5-DIGIT BAR CODE	ZPL II CODE
	^XA ^FO100,100^BY3 ^BSN,100,Y,N ^FD12345^FS ^XZ

UPC/EAN 5-DIGIT BAR CODE CHARACTERS
0 1 2 3 4 5 6 7 8 9

最終的に作成される複合コードが UPC 仕様範囲内になるように、UPC-A または UPC-E コードに対する UPC/EAN アドオンの配置には注意が必要です。

モジュール幅が **2**（デフォルト）の UPC コードの場合、アドオンのフィールド基点オフセットは次のとおりです。

例 • 次は UPC-A の例です。

	Supplement Origin X - Offset	Adjustment Y - Offset
Normal	209 Dots	21 Dots
Rotated	0	209 Dots

次は UPC-E の例です。

	Supplement Origin X - Offset	Adjustment Y - Offset
Normal	122 Dots	21 Dots
Rotated	0	122 Dots

さらに、アドオンのバー・コードの高さは主コードよりも 27 ドット（0.137 インチ）低くなければなりません。主 UPC コードの高さが 183 ドット（0.900 インチ）の場合、アドオンの高さは 155 ドット（0.765 インチ）である必要があります。

例 • 次の例では、7000002198 という値の標準 UPC-A バー・コードを 04414 のアドオンを含めて作成する方法を示しています。

UPC-A BAR CODE WITH EXTENSION	ZPL II CODE
	^XA ^FO100,100^BY3 ^BUN,137 ^FD07000002198^FS ^FO400,121 ^BSN,117 ^FD04414^FS ^XZ

^BT

TLC39 バー・コード

説明 ^BT バー・コードは、TCIF の標準で、通信機器にタグを付けることができます。

Micro-PDF417 バー・コードである TCIF CLEI コードは、常に 4 列です。ファームウェアでは、エンコードする文字数に基づき、使用するモードを決定する必要があります。

フォーマット ^BTo,w1,r1,h1,w2,h2

次の表では、このフォーマットのパラメータを示します。

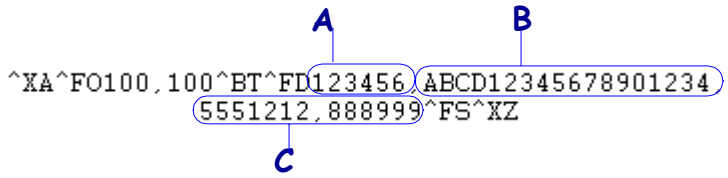
パラメータ	詳細
o = 向き	*有効値：* N = 標準 R = 回転 I = 反転 B = 下から上
w1 = Code 39 バー・コードの幅	*有効値（ドット数）：*1 ～ 10 *デフォルト値（600 dpi プリンタの場合）：*4 *デフォルト値（200 dpi プリンタの場合）：*2
r1 = Code 39 バー・コードの太いバーと細いバーの幅の比率	*有効値：*2.0 ～ 3.0（増分単位は 0.1） *デフォルト値：*2.0

パラメータ	詳細
h1 = Code 39 バー・コードの高さ	*有効値（ドット数）:* 1 ～ 9999 *デフォルト値（600 dpi プリンタの場合）:* 120 *デフォルト値（300 dpi プリンタの場合）:* 60 *デフォルト値（200 dpi プリンタの場合）:* 40
h2 = Micro-PDF417 バー・コードの行の高さ	*有効値（ドット数）:* 1 ～ 255 *デフォルト値（600 dpi プリンタの場合）:* 8 *デフォルト値（200 dpi と 300 dpi プリンタの場合）:* 4
w2= Micro-PDF417 バー・コードの細いバーの幅	*有効値（ドット数）:* 1 ～ 10 *デフォルト値（600 dpi プリンタの場合）:* 4 *デフォルト値（200 dpi と 300 dpi プリンタの場合）:* 2

例 • TLC39 バー・コード

次は TLC39 バー・コードを印刷する例です。主なコンポーネントはコールアウトによって示されており、下には詳しい説明が記載されています。

注記 • TCIF 産業標準に従った結果を得るには、印刷ヘッド密度にかかわらず、コマンドのデフォルトを使用してください。



A -- ECI 番号。 7 番目の文字が**カンマでない**場合は、Code 39 のみが印刷されます。これは、6 桁以上ある場合、最初の 6 桁に対して Code 39 が印刷され、Micro-PDF コードは印刷されないことを意味します。

- 6 桁である必要があります。
- 6 桁以外の桁数を検知すると、ファームウェアは無効な文字エラーを生成します。
- この数値はパディング処理できません。

A -- シリアル 番号。 シリアル番号は可変長で、最高 25 文字まで含めることができます。シリアル番号は Micro-PDF シンボルに保存されます。シリアル番号の後にカンマがある場合は、下の追加データを使用します。

- このコンポーネントが存在する場合は、英数字（句読点を含まない文字と数字）である必要があります。

この値は、ECI 番号の後にカンマがある場合に使用します。

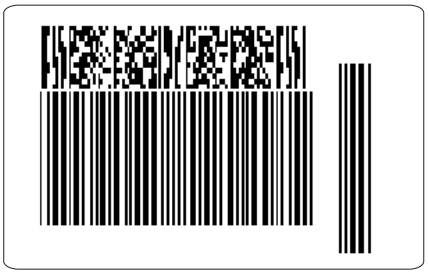
C -- 追加のデータ。このコンポーネントが存在する場合、国コードなどに使用します。

注記 • データは 150 バイトを超えてはなりません。これにはシリアル番号のカンマも含まれます。

- 追加のデータは Micro-PDF シンボルに保存され、シリアル番号の後に追加されます。それぞれ最高 25 文字までの追加フィールド間にはカンマが必要です。

各追加データ・フィールドには最高 25 文字の英数字を含めることができます。

結果は次のとおりです。

ZPL II CODE	GENERATED LABEL
^XA^FO100 100^BT^FD123456 ABCd12345678901234 5551212 88899 ^FS^XZ	

^BU

UPC-A バー・コード

説明 ^BU コマンドは、固定長の数値シンボル・コード体系を生成します。このコードは、主に小売産業でパッケージのラベルに使用されています。UPC-A バー・コードには 11 データ文字があります。6 ドット /mm、12 ドット /mm、24 ドット /mm の印刷ヘッドにより、UPC-A バー・コード（UPC/EAN シンボル・コード体系）が 100% のサイズで生成されます。しかし、8 ドット /mm の印刷ヘッドの場合は、UPC/EAN シンボル・コード体系が 77% の倍率因数で生成されます。

- ^BU は固定印字比率をサポートします。
- フィールド・データ（^FD）は 11 文字丁度に制限されています。必要な文字数にするために、ZPL II は、自動的に左端でゼロを切り捨てたり追加（パディング処理）したりします。

フォーマット ^BUo,h,f,g,e

重要 • UPC-A バー・コードに関して追加の詳細が必要な場合は、*『ZPL II プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照してください。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o ＝向き	*有効値：* N ＝ 標準 R ＝ 90° 回転（時計方向） I ＝ 180° 反転 B ＝ 下から上へ読み取り、270° *デフォルト値：*現在の ^FW 値
h＝バー・コードの高さ（ドット数）	*有効値：*1 ～ 9999 *デフォルト値：*^BY によって設定された値
f ＝テキスト表示ラインを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y
g ＝テキスト表示ラインをコードの上に印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N
e＝チェック・ディジットを印刷する	*有効値：*Y（はい）および N （いいえ） *デフォルト値：*Y

テキスト表示ラインのフォント・スタイルは、^BY で選択したモジュラス（細いバーの幅）によって異なります。

6 ドット /mm のプリンタ：2 ドット以上のモジュラスの場合、OCR-B テキスト表示ライン付きで印刷され、1 ドットのモジュラスの場合はフォント A で印刷されます。

8 ドット /mm のプリンタ：3 ドット以上のモジュラスの場合、OCR-B テキスト表示ライン付きで印刷され、1 または 2 ドットのモジュラスの場合はフォント A で印刷されます。

12 ドット /mm のプリンタ：5 ドット以上のモジュラスの場合、OCR-B テキスト表示ライン付きで印刷され、1、2、または 3 ドットのモジュラスの場合はフォント A で印刷されます。

24 ドット /mm のプリンタ：9 ドット以上のモジュラスの場合、OCR-B テキスト表示ライン付きで印刷され、1 ～ 8 ドットのモジュラスの場合はフォント A で印刷されます。

例 • 次はアドオン付きの UPC-A バー・コードの例です。

UPC-A BAR CODE WITH EXTENSION	ZPL II CODE
	^XA ^FO100,100^BY3 ^BUN,137 ^FD07000002198^FS ^FO400,121 ^BSN,117 ^FD04414^FS ^XZ

コメント UPC-A バー・コードはエラー・チェックにモジュラス 10 チェック・ディジット・スキームを使用します。モジュラス 10 の詳細については、*『ZPL プログラミング・ガイド第 2 巻』* の付録 C を参照してください。

^BX

Data Matrix バー・コード

説明 ^BX コマンドは、アラインメント・パターンとタイミング・セルの枠の中に配置された正方形で構成される 2 次元のマトリックス・シンボル・コード体系を作成します。

フォーマット ^BXo,h,s,c,r,f,g

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値 :* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値 :* 現在の ^FW 値
h = 個々のシンボル・エレメントの高さ	*有効値 :* 1 ～ ラベルの幅 個々のエレメントは正方形です。すなわち、このパラメータはモジュールと行の高さの両方を指定します。このパラメータがゼロ（または指定されていない）の場合は、^BY の h パラメータ（バーの高さ）がシンボルのおよその高さとして使用されます。

パラメータ	詳細
s = 品質レベル	*有効値：* 0, 50, 80, 100, 140, 200 *デフォルト値：* 0 *品質* は、エラー訂正のためにシンボルに追加されるデータ量です。AIM 仕様では、これは ECC 値と呼ばれます。ECC 50、ECC 80、ECC 100、および ECC 140 は畳み込み式エンコードを使用し、ECC 200 は Reed-Solomon エンコードを使用しています。新しいアプリケーションには ECC 200 をお勧めします。ECC 0000-140 は、単一のパーティがシンボルの生成と読み取りの両方を管理してシステム・パフォーマンス全般を担当しているクローズド・アプリケーションでのみ使用してください。
c = エンコードする列	*有効値：* 9 ～ 49 品質の値が 0 ～ 140（10 ～ 144）の場合は奇数値のみ、品質の値が 200 の場合は偶数値のみ。
r = エンコードする行	*有効値：* 9 ～ 49 品質の値が 0 ～ 140（10 ～ 144）の場合は奇数値のみ、品質の値が 200 の場合は偶数値のみ。シンボル内の行数と列数は自動的に決定されます。行数と列数の値に大きい値を強制することにより、一様なシンボル・サイズを実現できます。現在の実装では、品質 0 ～ 140 のシンボルは正方形で、行数または列数の大きい方の値が使用されシンボルはそのサイズになります。小さすぎるシンボルにデータを含めようとしても、シンボルは印刷されません。49 より大きい値を入力すると、行または列の値はゼロに設定され、サイズは通常の方法で決定されます。偶数を入力すると、INVALID-P（無効なパラメータ）が生成されます。9 より小さく 0 より大きい値を指定した場合、または強制されたサイズに対してデータが大きすぎる場合は、シンボルは印刷されません。さらに、^CV がアクティブであれば INVALID-L が印刷されます。

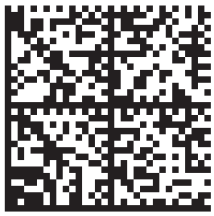
パラメータ	詳細
f = フォーマット ID (0 ～ 6) -- 品質が 200 に設定されている場合は使用されません。	*有効値：* 1 = フィールド・データは数字 + スペース（0..9,”）- \&” は使用しません。 2 = フィールド・データは上段の英数字 + スペース（A..Z,”）- \&” は使用しません。 3 = フィールド・データは上段の英数字 + スペース、ピリオド、カンマ、ダッシュ、およびスラッシュ（0..9,A..Z,“.-/”）です。 4 = フィールド・データは上段の英数字 + スペース（0..9,A..Z,”）- \&” は使用しません。 5 = フィールド・データは 128 文字のフル ASCII 7 ビット・セットです。 6 = フィールド・データは 256 文字のフル ISO 8 ビット・セットです。 *デフォルト値：* 6
g = エスケープ・シーケンス・コントロール文字	*有効値：* 任意の文字 *デフォルト値：* _（下線） このパラメータは、品質 200 が指定されている場合にのみ使用されます。フィールド・データ内に特殊なコントロール・シーケンスを埋め込むためのエスケープ文字です。^BX の使い方については、「フィールド・データ」（*^FD ページ 171*）を参照してください。

ECC LEVEL	ID = 1	ID = 2	ID = 3	ID = 4	ID = 5	ID = 6
0	596	452	394	413	310	271
50	457	333	291	305	228	200
80	402	293	256	268	201	176
100	300	218	190	200	150	131
140	144	105	91	96	72	63

Maximum Field Sizes

表 I 最大フィールド・サイズ

例 • 次は Data Matrix バー・コードの例です。

DATA MATRIX BAR CODE	ZPL II CODE
	^XA ^FO100,100 ^BXN,10,200 ^FDZEBRA TECHNOLOGIES CORP 333 CORPORATE WOODS PKWY VERNON HILLS, ILLINOIS 60061-3109^FS ^XZ

^BX に対する ^BY の影響

w = モジュール *（影響なし -- 個々のシンボル・エレメントの寸法を参照）*

r = 比率 *（影響なし）*

h = シンボルの高さ

個々のシンボル・エレメントの寸法が ^BD コマンドで指定されていない場合は、シンボル値の高さを必要な行数 / 列数で割り、四捨五入した値（最低 1）が個々のシンボル・エレメントの寸法として使用されます。

^BX のフィールド・データ（^FD）

品質 000 ～ 140

- PDF417 と同様に、*¥&* と ‖ は、キャリッジ・リターン、ライン・フィード、およびバックスラッシュの挿入に使用できます。コントロール文字内のその他の文字は、^FH を使用しなければ挿入できません。品質 *0* ～ *140* の場合、フィールド・データは 596 文字に制限されています。制限を超えるフィールド・データがあるとシンボルは印刷されず、^CV がアクティブな場合は INVALID-L が印刷されます。フィールド・データはユーザー指定によるフォーマット ID に対応している必要があり、対応していないとシンボルは印刷されず、^CV がアクティブな場合は INVALID-C が印刷されます。
- 品質 *0* ～ *140* のシンボルの最大フィールド・サイズは、**g** パラメータの表に示されています。

品質 200

- フィールド・データとして 3072 文字以上が入力された場合、3072 文字になるように切り捨てられます。これにより、仕様で許可される数字の Data Matrix シンボルの最大サイズは 3116 文字（数字）に制限されます。英数字の場合の最大許容サイズは 2335 文字で、8 ビット・バイトの場合の最大許容サイズは 1556 文字です。
- ^FH を使用した場合、次に説明するエスケープ・シーケンス処理が発生する前に、フィールドの 16 進数処理が行われます。
- 品質 200 のフィールド・データでは、下線がデフォルトのエスケープ・シーケンス・コントロール文字です。^BX コマンドでパラメータ g を使用することにより、異なるエスケープ・シーケンス・コントロール文字を選択することができます。

ASCII 95 の下線文字（_）、またはパラメータ g で入力した文字を使用して、入力ストリング・エスケープ・シーケンスを品質 200 のフィールド・データに埋め込むことができます。

- _X は、コントロール文字のシフト文字です（例：_@=NUL、_G=BEL、_0 = PAD など）。
- _1 ～ _3 は FNC 文字 1 ～ 3 に対応します（明示的 FNC4、上段シフトは使用できません）。
- FNC2（構造化結合）の後には、シンボル・シーケンスとファイル ID を表す 1 ～ 254 の値の 3 つの 3 桁の数値から成る 9 桁が必要です（たとえば、ファイル ID が 1001 のシンボル 3 of 7 は 2214001001 になります）。
- 5NNN はコード・ページ NNN で、NNN は 3 ページのコード・ページ値です（たとえばコード・ページ 9 は _5009 で表します）。
- _dNNN は、コード・ワードとして ASCII の 10 進値 NNN を作成します（3 桁）。
- データ内の _ は、__（2 つの下線）にエンコードされます。

^BY

バー・コード・フィールドのデフォルト

説明 ^BY コマンドは、モジュール幅（ドット数）、太いバーと細いバーの比率、およびバー・コードの高さのデフォルト値を変更するために使用されます。このコマンドは、ラベル・フォーマット内で必要な回数、何回でも使用できます。

フォーマット ^BYw,r,h

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
w＝モジュール幅（ドット数）	*有効値：*1 ～ 10 *パワーアップ時の初期値：*2
r ＝太いバーと細いバーの幅の比率	*有効値：*2.0 ～ 3.0（増分単位は 0.1） このパラメータは、固定比率のバー・コードには影響しません。
h＝バー・コードの高さ（ドット数）	*有効値：*2.0 ～ 3.0（増分単位は 0.1） *パワーアップ時の初期値：*10

パラメータ r の場合、生成される実際の比率は、パラメータ w （モジュール幅）のドット数の関数です。次のページの表を参照してください。

例 • モジュール幅（w）を 9 に、比率（r）を 2.4 に設定します。細いバーの幅は 9 ドットで、太いバーは 9 x 2.4、すなわち 21.6 ドットです。しかし、プリンタでは最も近いドット数に四捨五入されるため、太いバーは実際には 22 ドットとして印刷されます。

これにより、2.44（22 を 9 で割る）の比率のバー・コードが生成されます。フル・ドットのみが印刷されるため、この比率は 2.4 に最も近いものです。

モジュール幅と高さ（w と h）は、選択したシンボル・コード体系にかかわらず、^BY コマンドによっていつでも変更できます。

Ratio Selected (r)	Module Width in Dots (w)									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
2.0	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1
2.1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2:1	2.1:1
2.2	2:1	2:1	2:1	2:1	2.2:1	2.16:1	2.1:1	2.12:1	2.1:1	2.2:1
2.3	2:1	2:1	2.3:1	2.25:1	2.2:1	2.16:1	2.28:1	2.25:1	2.2:1	2.3:1
2.4	2:1	2:1	2.3:1	2.25:1	2.4:1	2.3:1	2.28:1	2.37:1	2.3:1	2.4:1
2.5	2:1	2.5:1	2.3:1	2.5:1	2.4:1	2.5:1	2.4:1	2.5:1	2.4:1	2.5:1
2.6	2:1	2.5:1	2.3:1	2.5:1	2.6:1	2.5:1	2.57:1	2.5:1	2.5:1	2.6:1
2.7	2:1	2.5:1	2.6:1	2.5:1	2.6:1	2.6:1	2.57:1	2.65:1	2.6:1	2.7:1
2.8	2:1	2.5:1	2.6:1	2.75:1	2.8:1	2.6:1	2.7:1	2.75:1	2.7:1	2.8:1
2.9	2:1	2.5:1	2.6:1	2.75:1	2.8:1	2.8:1	2.85:1	2.87:1	2.8:1	2.9:1
3.0	3:1	3:1	3:1	3:1	3:1	3:1	3:1	3:1	3:1	3:1

表 J モジュール幅の比率をドット数で示しています

コメント ラベル・フォーマットに ^BY コマンドを入力すると、それは別の ^BY コマンドを入力するまで有効です。

^BZ

POSTNET バー・コード

説明 POSTNET バー・コードは郵便物の処理の自動化に使用されます。POSTNET は、5 本のバー（長いバー 2 本、短いバー 3 本）を使用して 0 ～ 9 桁を表します。

- ^BZ は、2.0:1 ～ 3.0:1 の印字比率をサポートします。
- フィールド・データ（^FD）はラベルの幅に限定されます。

フォーマット ^BZo,h,f,g

重要 • POSTNET バー・コードに関して追加の詳細が必要な場合は、*『ZPL プログラミング・ガイド第 2 巻』* で AIM, Inc. の連絡先情報を参照するか、米国郵便公社に問い合わせ、POSTNET の仕様がすべて記載されている Publication 25『Designing Letter Mail』を請求してください。この Publication 25 は以下のサイトからもダウンロードできます。

http://pe.usps.gov/cpim/ftp/pubs/pub25/pub25.pdf

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = 向き	*有効値 :* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値 :* 現在の ^FW 値
h = バー・コードの高さ（ドット数）	*有効値 :* 1 ～ 32000 *デフォルト値 :* ^BY によって設定された値
f = テキスト表示ラインを印刷する	*有効値 :* Y（はい）または N （いいえ） *デフォルト値 :* N
g = テキスト表示ラインをコードの上に印刷する	*有効値 :* Y（はい）または N （いいえ） *デフォルト値 :* N

例 • 次は POSTNET バー・コードの例です。

POSTNET BAR CODE

ZPL II CODE

```
^XA
^FO100,100^BY3
^BZN,40,Y,N
^FD12345^FS
^XZ
```

POSTNET BAR CODE CHARACTERS

0 1 2 3 4 5 6 7 8 9

^CC

キャレットの変更 .

説明 ^CC コマンドは、フォーマット・コマンドのプレフィックスを変更する場合に使用します。デフォルトのプレフィックスはキャレット（^）です。

フォーマット ^CCx

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
x = キャレット文字の変更	*有効値 :* 任意の ASCII 文字 *デフォルト値 :* パラメータが必要です。パラメータを入力しないと、受け取った次の文字が新しいプレフィックス文字となります。

例 • 次は、フォーマット・プレフィックスを ^ から / に変更する例です。

```
^XA
^CC/
/XZ
```

コメント 上の例では、フォワード・スラッシュ（/）が新しいプレフィックスとして設定されています。終了タグの /XZ では新しく指定されたプレフィックス文字（/）が使われていることに注意してください。

~CC

キャレットの変更

説明 ~CC コマンドは、フォーマット・コマンドのプレフィックスを変更する場合に使用します。デフォルトのプレフィックスはキャレット（^）です。

フォーマット ~CCx

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
x＝キャレット文字の変更	*有効値：*任意の ASCII 文字 *デフォルト値：*パラメータが必要です。パラメータを入力しないと、受け取った次の文字が新しいプレフィックス文字となります。

例 • 次は、コマンド・プレフィックスを ~ から / に変更する例です。

```
~CC/
/XA/JUS/XZ
```

^CD ~CD

デリミタの変更

説明 ^CD と ~CD コマンドは、デリミタ文字を変更する場合に使用します。この文字は、複数の ZPL II コマンドに関連するパラメータ値を区切るために使用します。デフォルトのデリミタはカンマ（,）です。

フォーマット ^CDa または ~CDa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a＝デリミタ文字の変更	*有効値：*任意の ASCII 文字 *デフォルト値：*パラメータが必要です。パラメータを入力しないと、受け取った次の文字が新しいプレフィックス文字となります。

例 • 次の例は、文字のデリミタをピリオド（.）に変更する方法を示しています。

```
^XA
^CD.
^XZ
```

- 保存するには JUS コマンドが必要です。次は、JUS を使用した例です。

```
~CD.
^XA^JUS^XZ
```

^CF

英数字デフォルト・フォントの変更

説明 ^CF コマンドは、プリンタで使用されるデフォルト・フォントを設定します。^CF コマンドを使用すると、プログラムを単純化できます。

フォーマット ^CFf,h,w

次の表では、このフォーマットのパラメータを示します。

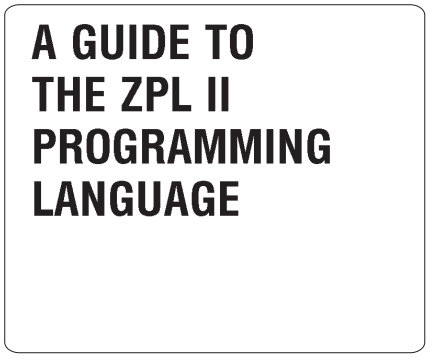
パラメータ	詳細
f = 指定されたデフォルト・フォント	*有効値：*A ～ Z、および 0 ～ 9 *パワーアップ時の初期値：*A
h = 個々の文字の高さ（ドット数）	*有効値：*0 ～ 32000 *パワーアップ時の初期値：*9
w = 個々の文字幅（ドット数）	*有効値：*0 ～ 32000 *パワーアップ時の初期値：*5 または最後に永久保存された値

パラメータ f は各英数字フィールドのデフォルト・フォントを指定します。パラメータ h は、各英数字フィールドのデフォルトの高さで、パラメータ 2 は各英数字フィールドのデフォルト幅の値です。

デフォルトの英数字フォントは A です。英数字のデフォルト・フォントを変更せず、英数字のフィールド・コマンド（^Af）を使用しなかった場合、または無効のフォント値を入力した場合は、指定したデータはフォント A で印刷されます。

高さか幅のどちらかのみを定義すると、倍率は定義されたパラメータに比例します。どちらの値も定義されていない場合、高さと幅に指定された最後の ^CF 値、またはデフォルトの ^CF 値が使用されます。

例 • 次は ^CF コードとその結果の例です。

ZPL II CODE	GENERATED LABEL
^XA ^CF0,89 ^FO20,50 ^FDA GUIDE TO^FS ^FO20,150 ^FDTHE ZPL II^FS ^FO20,250 ^FDPROGRAMMING^FS ^FO20,350 ^FDLANGUAGE^FS ^XZ	

コメント ^CW では、ダウンロードされたフォント、EPROM 保存フォント、A ～ Z および 0 ～ 9 のフォントを含むプリンタ内の任意のフォントも選択できます。

^CI

国際フォントの変更

説明 Zebra プリンタでは、アメリカ 1、アメリカ 2、イギリス、オランダ、デンマーク / ノルウェー、スウェーデン / フィンランド、ドイツ、フランス 1、フランス 2、イタリア、スペイン、おのびその他のセットを含む国際文字セットを使用して印刷できます。

ZPL II は国際文字の ISO 規格に従っています。

^CI コマンドを使用すると、印刷に使う国際文字を呼び出すことができます。ラベルでは文字セットを混合することができます。

このコマンドは文字の再マッピングを可能にします。フォント内の文字は、別の数値位置に再マッピングできます。

フォーマット `^CIa,s1,d1,s2,d2,...`

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 希望の文字セット	*有効値：* 0 = アメリカ 1 1 = アメリカ 2 2 = イギリス 3 = オランダ 4 = デンマーク / ノルウェー 5 = スウェーデン / フィンランド 6 = ドイツ 7 = フランス 1 8 = フランス 2 9 = イタリア 10 = スペイン 11 = その他 12 = 日本（円記号を含めた ASCII） 13 = IBM コード・ページ 850（*ページ 44* を参照） 14 = 16 ビット（Unicode）エンコード・スケーラブル・フォント * 15 = スケーラブル日本語フォント用 Shift-JIS** 16 = スケーラブル・フォント用 EUC- 漢字 17 = Unicode（Unicode エンコード・フォント用） 18 ～ 23 = 予約済み 24 = Unicode エンコード・フォントへの 8 ビット・アクセス *パワーアップ時の初期値：*0

パラメータ	詳細
s1 = マップ元 1（再マッピングする文字の位置）	*有効値：*10 進数 0 ～ 255
d1 = マップ先 1（s1 の文字の新しい位置）	*有効値：*10 進数 0 ～ 255
s2 = ソース 2（再マッピングする文字位置）	*有効値：*10 進数 0 ～ 255
d2 = マップ先 2（s2 の文字の新しい位置）	*有効値：*10 進数 0 ～ 255
... = パターンの継続	このコマンドではマップ元とマップ先の 256 の組み合わせを入力できます。

* エンコードは、変換表（*.DAT）によって制御されます。ZToolsa によって生成されたこの表は、TrueType フォントの内部コード（Unicode）です。

**Shift-JIS エンコードは、Shift-JIS を JIS に変換してから、JIS.DAT で JIS 変換をルックアップします。Shift_JIS が機能するためにはこの表が必要です。

例 1 • 次の例では、Euro 記号（21）をドル記号値（36）に再マッピングします。ドル記号文字がプリンタに送信されると、ユーロ記号が印刷されます。ユーロ記号値 15（16 進数）は 21（10 進数）と同等であり、ドル記号値 24（16 進数）は 36（10 進数）と同等になります。

```
^CI0,21,36
```

例 2 • ユーロ記号を印刷するために、16 進数値の 15 をフィールド・データ（^FD）コマンドに挿入します。^FD コマンドの前には、フィールド 16 進（^FH）コマンドが必要です。

ZPL II CODE	GENERATED LABEL
^XA ^FO10,10 ^A0,100 ^FH^FD_15^FS ^XZ	€

選択したフォントによって、印刷された記号の形と解像度が決定されます。

国際文字セット

Hex	23	30	40	5B	5C	5D	5E	60	7B	7C	7D	7E
CI0	#	0	@	[	¢	]	^	'	{	¦	}	~
CI1	#	0	@	⅓	¢	⅔	^	'	¼	½	¾	~
CI2	£	0	@	[	¢	]	^	'	{	¦	}	~
CI3	ƒ	0	§	[	IJ	]	^	'	{	ij	}	~
CI4	#	0	@	Æ	Ø	Å	^	'	æ	ø	å	~
CI5	Ü	0	É	Ä	Ö	Å	Ü	é	ä	ö	å	ü
CI6	#	0	§	Ä	Ö	Ü	^	'	ä	ö	ü	ß
CI7	£	0	à	[	ç	]	^	'	é	¦	ù	è
CI8	#	0	à	â	ç	ê	î	ô	é	ù	è	û
CI9	£	0	§	[	ç	é	^	ù	à	ò	è	ì
CI10	#	0	§	¡	Ñ	¿	^	'	{	ñ	ç	~
CI11	£	0	É	Ä	Ö	Ü	^	ä	ë	ï	ö	ü
CI12	#	0	@	[	¥	]	^	'	{	¦	}	~
CI13	#	0	@	[	\	]	^	'	{	¦	}	~

コメント どのフォントの場合も、スペース文字は再マッピングできません。

^CM

メモリの文字割り当ての変更

説明 ^CM コマンドを使用すると、プリンタのメモリ・デバイスに割り当てられた文字を変更できます。フォーマットがすでに存在する場合、フォーマットを変更したり作成しなおしたりする必要なく、メモリ・デバイスを対応する文字に割り当てなおすことができます。

このコマンドを使用すると、特定のメモリの場所を参照する後続のコマンドすべてに影響します。

フォーマット ^CMa,b,c,d

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = メモリのエイリアス文字割り当て	*有効値：*B:、E:、R:、A:、および NONE *デフォルト値：*B:
b = メモリのエイリアス文字割り当て	*有効値：*B:、E:、R:、A:、および NONE *デフォルト値：*E:
c = メモリのエイリアス文字割り当て	*有効値：*B:、E:、R:、A:、および NONE *デフォルト値：*R:
d = メモリのエイリアス文字割り当て	*有効値：*B:、E:、R:、A:、および NONE *デフォルト値：*A:

コメント 2 つ以上のパラメータが同じ文字指定子を指定している場合、すべての文字指定子はデフォルト値に設定されます。

パラメータが仕様外の場合、コマンドは無視されます。

例 次の例では、文字 E: が B: メモリ・デバイスを、文字 B: が E: メモリ・デバイスをポイントするように指定されています。

```
^XA
^CME,B,R,A
^JUS
^XA
```

次の例は、すべての文字指定がそれぞれ自らをポイントするようにリセットします。

```
^XA
^CME,B,B,A
^JUS
^XA
```

次の例は、すべての文字指定がそれぞれ自らをポイントするように設定します。

```
^XA
^CM,,R,A
^JUS
^XZ
```

コメント ^CM コマンドを入力した後は、^JUS を入力して、変更内容を EEPROM に保存することをお勧めします。重複したパラメータが入力されると、文字の指定はデフォルト設定に戻ります。

^CO

キャッシュ ON

説明 ^CO コマンドは、文字キャッシュのサイズを変更する場合に使用します。定義上、*文字キャッシュ*（キャッシュと呼びます）はスケーラブルな文字の保存のために予約されている DRAM の一部です。すべてのプリンタでは、22K のキャッシュが常にオンになっています。キャッシュのサイズを変更せずに保存可能な最大の単一文字サイズは、450 ドット x 450 ドットです。

Zebra プリンタでは、ビットマップとスケーラブルの 2 種類のフォントが使用されます。ビットマップ・フォントの文字、数字、記号は固定サイズ（たとえば 10 ポイント、12 ポイント、14 ポイント）です。それに対して、スケーラブル・フォントのサイズは固定していません。サイズはユーザーが選択します。

ビットマップ・フォントはサイズが固定されているため、ラベルに簡単に移動できます。それに対して、スケーラブル・フォントの場合は、ラベルに移動する前に、必要に応じて各文字を構築するため、かなりの時間がかかります。スケールした文字をキャッシュに保存することにより、それらをすばやく呼び出すことができます。

キャッシュに保存できる文字数は、キャッシュ（メモリ）のサイズと、保存される文字のサイズ（ポイント数）という 2 つの要因によって異なります。ポイント・サイズが大きいほど、キャッシュ内で使用される容量は多くなります。デフォルト・キャッシュでは、ラベルで使用するために要求されるスケーラブル文字のすべてが保存されます。同じ回転とサイズの同じ文字を再度使用する場合、それはキャッシュからすばやく取得されます。

しばらくすると、印刷キャッシュが一杯になる可能性があります。キャッシュが一杯になると、新しい文字用の容量は、印刷キャッシュの既存の文字を削除することによって取得されます。既存の文字は、使用頻度に基づき削除されます。これは自動的に行われます。たとえば、一度しか使われていない 28 ポイントの *Q* は、キャッシュから削除する文字の候補として適当です。

単一の印刷キャッシュ文字の最大サイズは 1500 ドット x 1500 ドットです。これには 300K のキャッシュが必要です。

キャッシュが希望のスタイルに対して小さすぎる場合は、小さい文字は表示されますが、大きい文字は表示されません。可能であれば、キャッシュのサイズを増やしてください。

フォーマット ^COa,b,c

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a ＝キャッシュ ON	*有効値：* Y（はい）または N （いいえ） *デフォルト値：* Y
b ＝キャッシュに追加するメモリ量（K 単位）	*有効値：* 使用可能なメモリ合計以内の任意のサイズ *デフォルト値：* 40
c ＝キャッシュ・タイプ	*有効値：* 0＝キャッシュ・バッファ（標準フォント） 1＝内部バッファ（アジア系フォントに推奨） *デフォルト値：* 0

例 • 次の例では、既存のキャッシュが 22K の場合に印刷キャッシュのサイズを 62K に変更します。

```
^COY,40
```

次の例では、既存のキャッシュが 22K の場合に印刷キャッシュのサイズを 100K に変更します。

```
^COY,78
```

印刷キャッシュのパフォーマンス

大きな文字を印刷する場合、^CO コマンドによってキャッシュに追加されたメモリは、プリンタにすでに存在する 22K のキャッシュに物理的に追加されるわけではありません。上の 2 番目の例では、追加後の 100K キャッシュは実際には 22K と 78K の 2 つの別々のブロックです。

大きな文字には連続したメモリ・ブロックが必要であるため、90K を要する文字は完全に保存されません。それは
100K のうちのどちらのブロックも十分な容量がないためです。したがって、大きな文字が必要な場合、^CO コマンドは必要なキャッシュの実際のサイズを反映する必要があります。

キャッシュのサイズを増やすと、スケーラブル・フォントのパフォーマンスが改善されます。しかし、キャッシュのサイズが大きくなり、格納した文字数が多すぎると、パフォーマンスは落ちます。各文字に対するキャッシュを探すために時間がかかるため、せっかく改善されたパフォーマンスも帳消しになってしまいます。

コメント キャッシュは、必要なときに何度でもサイズ変更できます。ただし、サイズ変更時のキャッシュ内の文字はすべて失われます。キャッシュにメモリが使用されるとともに、ラベル・ビットマップ、グラフィック、ダウンロード・フォントなどに利用できる容量は減ります。

アジア系フォントの一部では、通常のキャッシュよりもはるかに大きな内部バッファを必要とするものがあります。この大きなバッファはほとんどのフォントでは必要がないため、これは選択可能な設定オプションになりました。アジア系フォントで印刷すると、ラベル、グラフィックス、フォント、およびフォーマットなどに利用できるプリンタ・メモリが大幅に減少します。

^CT ~CT

ティルデの変更

説明 ^CT と ~CT コマンドは、コントロール・コマンドのプレフィックスを変更する場合に使用します。デフォルトのプレフィックスはティルデ (~) です。

フォーマット ^CTa または ~CTa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a＝コントロール・コマンド文字の変更	*有効値：*任意の ASCII 文字 *デフォルト値：*パラメータが必要です。パラメータを入力しないと、受け取った次の文字が新しいコントロール・コマンド文字となります。

例・ 次は、コントロール・コマンド・プレフィックスを ^ から ~ に変更する例です。

```
^XA
^CT+
^XZ
+DGR:GRAPHIC.GRF,04412,010
```

^CV

コード検証

説明 ^CV コマンドは、コード検証機能のオン / オフを切り替えるスイッチの役割をします。このコマンドがオンの場合、すべてのバー・コード・データで次のエラー状態が確認されます。

- 文字が文字セットにない
- チェック・ディジットが間違っている
- データ・フィールドが長すぎる（文字数が多すぎる）
- データ・フィールドが短かすぎる（文字数が少なすぎる）
- パラメータ・ストリングに不正なデータが含まれているかパラメータが欠如している

無効なデータが検出されると、バー・コードの代わりにエラー・メッセージとコードが反転イメージで印刷されます。メッセージは INVALID - X で、X は次のエラー・コードの 1 つで置換されます。

C = 文字が文字セットにない

E = チェック・ディジットが間違っている

L = データ・フィールドが長すぎる

S = データ・フィールドが短かすぎる

P = パラメータ・ストリングに間違ったデータが含まれる
（一部のバー・コードでのみ発生）

いったんオンにするとフォーマットが変わっても、^CV コマンドは別の ^CV コマンドでオフにするか、プリンタをオフにするまでアクティブのままになります。このコマンドは永久保存されません。

フォーマット ^CVa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = コード検証	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 下の例は、間違ったフィールド・データが入力されたときに ^CVY で生成されるエラー・ラベルです。*INVALID -* の後に続く文字を前のページのリストと比べてください。

ZPL II CODE	GENERATED LABEL
^XA ^CVY ^FO50,50 ^BEN,100,Y,N ^FD97823456 890^FS ^XZ	INVALID - C
^XA ^CVY ^FO50,50 ^BEN,100,Y,N ^FD9782345678907^FS ^XZ	INVALID - E
^XA ^CVY ^FO50,50 ^BEN,100,Y,N ^FD97823456789081^FS ^XZ	INVALID - L
^XA ^CVY ^FO50,50 ^BEN,100,Y,N ^FD97823456789^FS ^XZ	INVALID - S
^XA ^CVY ^FO50,50 ^BQN2,3 ^FDHM,BQRCODE-22^FS ^XZ	INVALID - P

コメント 複数のエラーが存在する場合、最初に検出されたエラーが表示されています。

^CV コマンドは、バー・コードにエンコードされたデータの完全性をテストします。このコマンドは、イメージやバー・コードのスキャンの完全性のテストに使用されるものではなく、それらと混同しないようにしてください。

^CW

フォント識別子

説明 組み込みフォントはすべて 1 文字の識別子を使用して参照されます。^CW コマンドは、DRAM、メモリ・カード、EPROM、Flash などに保存されたフォントに対して単一の英数字を割り当てます。

割り当てられた文字が組み込みフォントと同じ場合は、組み込みフォントの代わりにダウンロード・フォントが使用されます。フォーマットで組み込みフォントが必要な場合は、ラベルに新しいフォントが印刷されます。組み込みフォントの代わりに使用された場合、変更は電源を切ると無効になります。

割り当てられた文字が異なる場合、ダウンロード・フォントが追加のフォントとして使用されます。割り当ては、新しいコマンドが発行されるか、プリンタの電源を切るまで有効です。

フォーマット ^CWa,d:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a ＝既存のフォントの文字を置換、または新しいフォントを追加	*有効値：* A ～ Z、および 0 ～ 9 *デフォルト値：* 1 文字のエントリが必要です。
d＝フォントを保存するデバイス（オプション）	*有効値：* R:、E:、B:、および A: *デフォルト値：* R:

パラメータ	詳細
o = 組み込みフォントを置換する、または追加のフォントとしてのダウンロード・フォントの名前	*有効値：*最高 8 文字までの任意の名前 *デフォルト値：*名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子	*固定値：*.FNT

例 • 次の例は以下の使用法を示しています。

- フォーマットでフォント A が必要な場合、DRAM に保存された MYFONT.FNT

```
^XA
^CWA,R:MYFONT.FNT
^XZ
```

- DRAM に追加のフォント Q として保存された MYFONT.FNT

```
^XA
^CWQ,R:MYFONT.FNT
^XZ
```

- フォーマットでフォント F が必要な場合、DRAM に保存された NEWFONT.FNT

```
^XA
^CWF,R:NEWFONT.FNT
^XZ
```

```
DIRECTORY OF R:*.*

R:NEWFONT.FNT     65268
R:MYFONT.FNT      65268

582164 BYTES FREE R:
```

割り当て前のラベル・リスト

```
DIRECTORY OF R:*.*

F  R:NEWFONT.FNT     65268
AQ R:MYFONT.FNT      65268

582164 BYTES FREE R:
```

割り当て後のラベル・リスト

~DB

ビットマップ・フォントのダウンロード

説明 ~DB コマンドは、ダウンロードしたビットマップ・フォントを受け取るようにプリンタを設定し、ネイティブ・セルのサイズ、基準、スペース・サイズ、および著作権を定義します。

このコマンドは、フォントを定義する ZPLII コマンドと、フォントの各文字を定義する構造化データ・セグメントの 2 つの部分で構成されます。

フォーマット ~DBd:o.x,a,h,w,base,space,#char,©,data

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d＝フォントを保存するドライブ	*有効値：* R:、E:、B:、および A: *デフォルト値：* R:
o ＝フォント名	*有効値：* 1 ～ 8 文字の英数字 *デフォルト値：* 名前を指定しない場合は UNKNOWN が使用されます。
x＝拡張子	*固定値：* .FNT
a ＝ネイティブ・フォントの向き	*固定値：* 標準
h ＝ セルの最大高さ（ドット数）	*有効値：* 0 ～ 32000 *デフォルト値：* 値を指定する必要があります。
w ＝ セルの最大幅（ドット数）	*有効値：* 0 ～ 32000 *デフォルト値：* 値を指定する必要があります。

パラメータ	詳細
base ＝セルの一番上から文字のベースラインまでのドット数	*有効値：*0 ～ 32000 *デフォルト値：*値を指定する必要があります。
space ＝スペースまたは存在しない文字の幅	*有効値：*0 ～ 32000 *デフォルト値：*値を指定する必要があります。
#char ＝フォントの文字数	*有効値：*1 ～ 256（ダウンロードされる文字数と一致する必要があります）。 *デフォルト値：*値を指定する必要があります。
© ＝著作権保持者	*有効値：*1 ～ 63 文字の英数字 *デフォルト値：*値を指定する必要があります。
data ＝フォントの各文字を定義する構造化 ASCII データ	# 記号は、ピリオドで区切られる文字コード・パラメータを意味します。文字コードは 1 ～ 4 文字で、大きな国際文字セットをプリンタにダウンロードできます。 データ構造は次のとおりです。 #xxxx.h.w.x.y.i.data #xxxx = 文字コード h ＝ ビットマップの高さ（ドット行数） w ＝ ビットマップの幅（ドット行数） x ＝ x オフセット（ドット数） y ＝ y オフセット（ドット数） i = モーション置換のタイプセット （フォントの特定の文字の文字間ギャップを含む幅） data = 16 進ビットマップの説明

例 • 次は、~DB コマンドを使用する例です。DRAM にダウンロードするフォントの最初の 2 文字を示しています。

```
~DBR:TIMES.FNT,N,5,24,3,10,2,ZEBRA 1992,

   #0025.5.16.2.5.18.
   OOFF
   OOFF
   FFOO
   FFOO
   FFFF
   #0037.4.24.3.6.26.
   OOFFOO
   OFOOFO
   OFOOFO
   OOFFOO
```

^DE

エンコードのダウンロード

説明 TrueType Windows® フォントの標準のエンコードは常に Unicode です。ZPL II フィールド・データは、別のエンコードから Zebra プリンタが認識できる Unicode に変換する必要があります。必要な変換表は、ZTools for Windows で提供されており、~DE コマンドでダウンロードできます。

フォーマット ~DEd:o.x,s,data

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = 表の場所	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* R:
o = 表の名前	*有効値 :* 最高 8 文字までの有効な名前 *デフォルト値 :* 名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子	*固定値 :* .DAT
s = 表のサイズ	*有効値:* Zebra でダウンロード可能なフォントのフォーマットを保存するために必要なメモリのバイト数 *デフォルト値:* 間違った値を入力したり、値を入力しなかった場合は、このコマンドは無視されます。
data = データ・ストリング	*有効値 :* ASCII 16 進数値のストリング *デフォルト値 :* データを入力しなかった場合は、このコマンドは無視されます。

例 • 次は、必要な変換表をダウンロードする方法の例です。

```
~DER:JIS.DAT,27848,300021213001...
```

（27848 の 2 桁の 16 進数値）

コメント ZTools for Windows の詳細については、ソフトウェアに付属のプログラムのマニュアルを参照してください。

^DF

ダウンロード・フォーマット .

説明 ^DF コマンドは、後で ^XF を使用して可変データと結合するために、ZPL II フォーマット・コマンドをテキスト・ストリングとして保存します。保存するフォーマットには、呼び出し時に参照するフィールド番号（^FN）コマンドが含まれている場合があります。

保存されたフォーマットを使用することによって、送信時間は短縮されますが、フォーマットにかかる時間は節約されません。すなわち、このコマンドは印刷時にフォーマットされるテキスト・ストリングとして ZPL II を保存します。

^XA コマンドのすぐ後に ^DF 保存フォーマット・コマンドを入力してから、保存するフォーマット・コマンドを入力してください。

フォーマット ^DFd:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = イメージを保存するデバイス	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* R:
o = イメージ名	*有効値 :* 1 ～ 8 文字の英数字 *デフォルト値 :* 名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子	*固定値 :* ZPL

~DG

グラフィックスのダウンロード

説明 ~DG コマンドは、次の機能を実行します。

- プリンタをグラフィックス・モードにします。
- グラフィックに名前を付けます（この名前はラベルにそれを呼び出すときに使用されます）。
- グラフィックスのサイズを定義します。
- 16 進ストリングをプリンタにダウンロードします。

グラフィックスのダウンロードの際のその他の保存オプションや読み込みオプションについては、*~DY ページ 162* を参照してください。

フォーマット ~DGd:o.x,t,w,data

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = イメージを保存するデバイス	*有効値：* R:、E:、B:、および A: *デフォルト値：* R:
o = イメージ名	*有効値：* 1 ～ 8 文字の英数字 *デフォルト値：* 名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子	*固定値：* .GRF
t = グラフィックのバイト数合計	下の数式を参照してください。

パラメータ	詳細
w = 行あたりのバイト数	下の数式を参照してください。
data = イメージを定義する ASCII 16 進ストリング	データ・ストリングはイメージを定義し、そのイメージを ASCII 16 進で表現するものです。各文字は、4 つのドットで構成される横方向のニブルを表します。

例 • これらの例のキーは次のとおりです。

x = グラフィックの幅（ミリメータ）

y = グラフィックの高さ（ミリメータ）

z = ドット /mm = プログラムするプリンタの印字密度

8 = ビット / バイト

演習 1 t パラメータは、次の数式を使用して決定できます。

$$\frac{xz}{8} \times yz = totalbytes$$

演習 1 たとえば、幅 8 mm、高さ 16 mm、印字密度 8 ドット /mm のグラフィックの正しいパラメータを決定するための数式は次のようになります。

$$\left(\frac{8 \times 8}{8}\right) \times (16 \times 8) = bytes$$

$$8 \times 128 = 1024$$

$$t = 1024$$

部分的バイトは次の整数バイトに繰り上げます。

w パラメータ（行あたりのバイト数で表した幅）は、次の数式を使用して決定できます。

$$\left(\frac{xz}{8}\right) = totalbytes/row$$

$$w = 8$$

たとえば、幅 8 mm、印字密度 8 ドット /mm のグラフィックの正しいパラメータを決定するための数式は次のようになります。

$$\left(\frac{8 \times 8}{8}\right) = 8 bytes$$

$$w = 8$$

部分的バイトは次の整数バイトに繰り上げます。

パラメータ w は t の計算の最初の値です。

data パラメータは、グラフィック・イメージの表現として送信される 16 進数のストリングです。各 16 進文字は、4 つのドットで構成される横方向のニブルを表します。たとえば、作成するグラフィック・イメージの最初の 4 桁が白で次の 4 桁が黒の場合、ドット x ドットのバイナリ・コードは 00001111 になります。このバイナリ値の 16 進表現は、OF になります。グラフィック・イメージ全体がこの方法でエンコードされます。完全なグラフィック・イメージは、16 進値の連続した 1 つのストリングとして送信されます。

コメント グラフィックスの命名にはスペースもピリオドも使用しないでください。異なるグラフィックスには必ず異なる名前を付けてください。

同じ名前の 2 つのグラフィックスがプリンタに送られた場合、最初のグラフィックスは削除され、2 つのグラフィックスによって置換されます。

~DN

グラフィックのダウンロードの中止

説明 ^DG コマンドのパラメータ t のバイト数を解読し印刷したら、プリンタは標準印字モードに戻ります。~DN コマンドを使用すると、グラフィックス・モードを中止し、標準のプリンタ操作を再開できます。

フォーマット ~DN

コメント グラフィックのダウンロードを停止する必要がある場合は、ホスト・デバイスから送信を中止する必要があります。しかし、~DG コマンドをクリアするには、~DN コマンドを送信する必要があります。

~DS

スケーラブル・フォントのダウンロード

説明 ~DS コマンドは、ダウンロード可能なスケーラブル・フォントを受け取るようにプリンタを設定し、フォントのサイズ（バイト）を定義するために使用します。~DS は、プリンタに Intellifont データをダウンロードするために使用します。TrueType フォントのダウンロードについては、ページ *158* の ~DT コマンドを参照してください。

~DS コマンドとそれに関連するパラメータは、ベンダー提供によるフォントを Zebra プリンタ用に変換した結果です。変換は Zebra Technologies から入手可能な Zebra ユーティリティ・プログラムの Ztools for Windows を使用して行います。

フォーマット ~DSd:o.x,s,data

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d＝イメージを保存するデバイス	*有効値：*R:、E:、B:、および A: *デフォルト値：*R:
o ＝イメージ名	*有効値：*1 ～ 8 文字の英数字 *デフォルト値：*名前を指定しない場合は UNKNOWN が使用されます。
x＝拡張子	*固定値：*.FNT
s ＝フォントのサイズ（バイト）	*固定値：*この数値は ZTools によって生成されるもので、変更はできません。
data＝フォントを定義する ASCII 16 進ストリング	*固定値：*この数値は ZTools によって生成されるもので、変更はできません。

例 • 次の例は、ZTools for Windows プログラムを使用して変換し、プリンタにダウンロード可能になったスケーラブル・フォントの最初の 3 行を示しています。変換先とオブジェクト名は必要に応じて変更できます。

```
~DSB:CGTIMES.FNT,37080,
OOFFOOFFOOFFOOFF
FFOAECB28FFFOOFF
```

コメント ダウンロードされたスケーラブル・フォントの完全性はチェックされません。破損している場合は、プリンタで予想できない結果が発生します。

~DT

TrueType フォントのダウンロード

説明 TrueType フォントを Zebra にダウンロード可能なフォントに変換するには ZTools for Windows プログラムを使用する必要があります。ZTools for Windows は、~DT コマンドを含むダウンロード可能ファイルを作成します。Intellifont 情報の変換とダウンロードについては、*ページ 156* の ~DS コマンドを参照してください。

フォーマット ~DTd:o.x,s,data

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d ＝フォントの場所	*有効値：* R:、E:、B:、および A: *デフォルト値：* R:
o ＝フォント名	*有効値：* 最高 8 文字までの有効な TryeType 名 *デフォルト値：* 名前を指定しない場合は UNKNOWN が使用されます。
x＝拡張子	*固定値：* .DAT
s＝フォント・サイズ	*有効値:* Zebra でダウンロード可能なフォントのフォーマットを保存するために必要なメモリのバイト数 *デフォルト値:* 間違った値を入力したり、値を入力しなかった場合は、このコマンドは無視されます。
data ＝データ・ストリング	*有効値：* ASCII 16 進数値のストリング（2 桁の 16 進数 / バイト）2 桁の値の合計はパラメータ s と一致する必要があります。 *デフォルト値：* データを入力しなかった場合は、このコマンドは無視されます。

例 • 次は、true type フォントをダウンロードする例です。

```
~DTR:FONT,52010,00AF01B0C65E...
```

（52010 の 2 桁の 16 進数値）

~DU

バウンドなしの TrueType フォントのダウンロード

説明 国際フォントの一部には、アジア系フォントのように印刷可能な文字が 256 文字以上あるものがあります。これらのフォントは、*大型の TrueType フォント*としてサポートされ、~DU コマンドによってプリンタにダウンロードされます。大型の TrueType フォントを Zebra にダウンロード可能なフォーマットに変換するには、ZTools for Windows プログラムを使用する必要があります。

フィールド・ブロック（^FD）コマンドは、大型の TrueType フォントをサポートできません。

フォーマット ~DUd:o.x,s,data

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = フォントの場所	*有効値：* R:、E:、B:、および A: *デフォルト値：* R:
o = フォント名	*有効値：* 1 ～ 8 文字の英数字 *デフォルト値：* 名前を指定しない場合 UNKNOWN が使用されます。
x = 拡張子	*固定値：* .FNT

パラメータ	詳細
s = フォント・サイズ	*有効値:* Zebra でダウンロード可能なフォントのフォーマットを保存するために必要なメモリのバイト数 *デフォルト値 :* データを入力しなかった場合は、このコマンドは無視されます。
data = データ・ストリング	*有効値 :* ASCII 16 進数値のストリング（2 桁の 16 進数 / バイト）2 桁の値の合計はパラメータ s と一致する必要があります。 *デフォルト値 :* データを入力しなかった場合は、このコマンドは無視されます。

例 • 次は、バウンドなしの true type フォントをダウンロードする例です。

```
~DUR:KANJI,86753,60CA017B0CE7...
```

（86753 の 2 桁の 16 進数値）

~DY

グラフィックスのダウンロード

説明 ~DY は、サポートされるすべてのフォーマットのグラフィック・オブジェクトをプリンタにダウンロードします。このコマンドを ~DG の代わりに使用すると、保存と読み込みのオプションが増えます。

フォーマット ~DYf,b,x,t,w,data

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
f ＝フォント名	*有効値：*1 ～ 8 文字の英数字 *デフォルト値：*名前を指定しない場合 UNKNOWN が使用されます。
b ＝データ・フィールドにダウンロードされるフォーマット（f）	*有効値：* A＝圧縮されていないビットマップ（.GRF、ASCII） B＝圧縮されていないビットマップ（.GRF、バイナリ） C＝AR 圧縮ビットマップ（.GRF、圧縮バイナリ -- Zebra の BAR-ONE® ソフトウェアでのみ使用） P＝PNG イメージ（.PNG） *デフォルト値：*値を指定する必要があります
x＝保存されたグラフィックの拡張子	*有効値：* G＝生のビットマップ（.GRF） P＝圧縮ファイルとして保存（.PNG） *デフォルト値：*.GRF（パラメータ b が P（.PNG）に設定されている場合を除く
t＝グラフィックのバイト数合計	.GRF イメージ：メモリへの解凍後のサイズ .PNG イメージ：.PNG ファイルのサイズ

パラメータ	詳細
w = 行あたりのバイト数合計	.GRF イメージ : 行あたりのバイト数 .PNG イメージ : 値は無視 -- データは .PNG データに直接エンコードされます
data = データ	b により、ASCII 16 進エンコード、ZB64、またはバイナリ・データ a, p = ASCII 16 進または ZB64 b, c = バイナリ バイナリ・データが送信されると、グラフィック・フォーマットに必要な全バイト数を受け取るまで、すべてのコントロール・プレフィックスとフロー・コントロール文字は無視されます。

コメント ZB64 エンコードの詳細については、*『ZPL プログラミング・ガイド 第 2 巻』* の付録 I を参照してください。

~EF

保存されたフォーマットの消去 .

説明 ~EF コマンドは、保存されたフォーマットをすべて消去します。

フォーマット ~EF

コメント ~EF コマンドの使用はお勧めしません。^ID（オブジェクトの削除）コマンドを使用して、保存されたフォーマットを選択的に削除することをお勧めします。

~EG

ダウンロードしたグラフィックスの消去

^ID ページ 218 を参照してください。

^FB

フィールド・ブロック

説明 ^FB コマンドを使用すると、定義したブロック・タイプのフォーマットでテキストを印刷できます。このコマンドは、テキスト・ストリングに対して指定された基点、フォント、回転を使用して ^FD ストリング、または ^SN ストリングをテキストのブロックにフォーマットします。^FB コマンドには自動ワードラップ機能が含まれます。

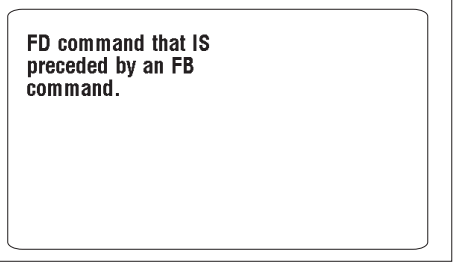
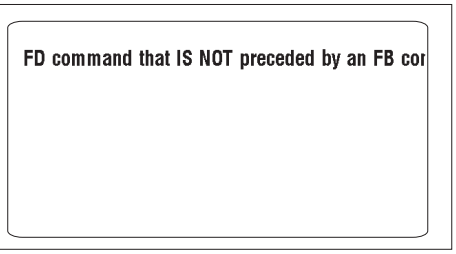
フォーマット ^FBa,b,c,d,e

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a＝テキスト・ブロック・ラインの幅（ドット数）	*有効値：*ラベルの幅（または 9999）に対して 0 *デフォルト値：*0 値がフォント幅より少ない場合、または指定されていない場合は、テキストは印刷されません。
b ＝ テキスト・ブロック内の最大ライン数	*有効値：*1 ～ 9999 *デフォルト値：*1 ラインの最大数を超えたテキストは、最後のラインを上書きします。フォント・サイズを変更すると、ブロックのサイズが自動的に増減します。
c ＝ライン間のスペースを追加または削除（ドット数）	*有効値：*-9999 ～ 9999 *デフォルト値：*0 マイナス符号がない限り、数値は正数とみなされます。正の値によりスペースが追加され、負の値によりスペースが削除されます。

パラメータ	詳細
d =テキスト調整	*有効値：*L（左）、C（中央）、R（右）、J（両端揃え） *デフォルト値：*L J を使用した場合、最後のラインは左揃えになります。
e =2 番目および残りのラインのハンギング・インデント（ドット数）	*有効値：*0 ～ 9999 *デフォルト値：*0

例 • 以下の例は ^FB コマンドがフィールド・データへ与える影響を示しています。

ZPL II CODE	GENERATED LABEL
^XA ^CF0,30,30^FO25,50 ^FB250,4,, ^FDFD command that IS preceded by an FB command.^FS ^XZ	
^XA ^CF0,30,30^FO25,50 ^FDFD command that IS NOT preceded by an FB command.^FS ^XZ	

^FB コマンドに関するコメント

このスキームは、以下のような特殊機能を可能にするために使用できます。

\& = キャリッジ・リターン / ライン・フィード

\(*) = ソフト・ハイフン（ダッシュによる単語の区切り）

\\ = バックスラッシュ（\）

項目 1 : バックスラッシュ（\）を印刷するには、^CI13 が選択されている必要があります。

項目 2 : ソフト・ハイフンをラインの終端近くに配置すると、ハイフンが印刷されます。ラインの終端近くに配置されない場合は、無視されます。

(*) = 任意の英数字

- 単語が 1 ラインに印刷するには長すぎる場合（そしてソフト・ハイフンが指定されていない場合）は、ブロックの右端の単語にハイフンが自動的に挿入されます。その単語の残りは次のラインに印刷されます。ハイフンの位置は、音節の区切りではなく、単語の長さによって決定されます。単語にソフト・ハイフンを挿入することにより、ハイフンの位置を制御することができます。
- データ・ストリングの最大長は 3K で、これにはコントロール文字、キャリッジ・リターン、ライン・フィードも含まれます。
- 通常のキャリッジ・リターン、ライン・フィード、および*ワード・スペース*は、改行時に破棄されます。
- ^FT（フィールド・タイプセット）を使用する場合、^FT は、最後のテキスト・ラインのベースライン基点を使用します。フォント・サイズを大きくすると、テキスト・ブロックのサイズが下端から上端へ大きくなります。これは、ラベルが上の余白を超えて印刷される原因となります。
- ^FO（フィールド基点）を使用する場合、フォント・サイズを大きくすると、テキスト・ブロックのサイズが上端から下端へ大きくなります。
- ^FD の代わりに ^SN を使用すると、フィールドは印刷されません。
- ^FS は ^FB コマンドを終了します。各ブロックは独自の ^FB コマンドを必要とします。

^FC

フィールド・クロック（リアルタイム・クロック用）

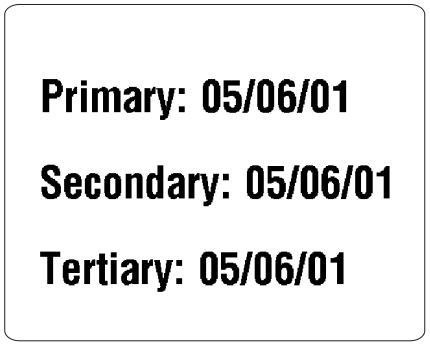
説明 ^FC コマンドは、リアルタイム・クロック・ハードウェアで使用するためにクロック・インジケータ（デリミタ）とクロック・モードを設定する場合に使用します。このコマンドは、フィールド内でリアルタイム・クロック値が必要になる都度、各ラベル・フィールド・コマンド・ストリングに含める必要があります。

フォーマット ^FCa,b,c

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 1 次クロック・インジケータ文字	*有効値 :* 任意の ASCII 文字 *デフォルト値 :* %
b = 2 次クロック・インジケータ文字	*有効値 :* 任意の ASCII 文字 *デフォルト値 :* なし -- この値は a または c と同じにすることはできません。
c = 3 次クロック・インジケータ文字	*有効値 :* 任意の ASCII 文字 *デフォルト値 :* なし -- この値は a または b と同じにすることはできません。

例 • これらの ZPL を入力すると、1 次クロック・インジケータが % に、2 次クロック・インジケータが { に、そして 3 次クロック・インジケータが # に設定されます。結果は、1 次、2 次、および 3 次をフィールド・データとするラベルに印刷されます。

ZPL II CODE	GENERATED LABEL
^XA ^FO10,100^A0N,50,50 ^FC%,{,# ^FDPrimary: %m/%d/%y^FS ^FO10,200^A0N,50,50 ^FC%,{,# ^FDSecondary: {m/{d/{y^FS ^FO10,300^A0N,50,50 ^FC%,{,# ^FDTertiary: #m/#d/#y^FS ^XZ	

コメント リアルタイム・ハードウェアがない場合は、^FC コマンドは無視されます。

^FD

フィールド・データ

説明 ^FD コマンドは、フィールドのデータ・ストリングを定義します。フィールド・データには、コマンド・プレフィックスで使用される文字（^ および ~）を除くすべての印刷可能文字を使用できます。

フォーマット ^FDa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 印刷するデータ	*有効値 :* 最高 3072 文字までの ASCII ストリング *デフォルト値 :* なし -- 文字のストリングを入力する必要があります。

コメント ^ と ~ 文字は、プレフィックス文字を変更することにより、印刷できます（*~CC ページ 126* と *^CT ~CT ページ 139* を参照してください。新しいプレフィックス文字は印刷できません。

127 以上のコードの文字、または ^ や ~ の文字は、^FH コマンド、および ^FD コマンドを使用して印刷できます。

- バックスラッシュ（\）を印刷するには、^CI13 が選択されている必要があります。

^FH

フィールド 16 進インジケータ

説明 ^FH コマンドを使用すると、どの文字に対しても 16 進値を ^FD ステートメントに直接入力できます。^FH コマンドは、フィールドに 16 進値を使用する各 ^FD コマンドの前に使用する必要があります。

^FD ステートメント内では、16 進インジケータは各 16 進値の前に挿入する必要があります。デフォルトの 16 進インジケータは _（下線）です。下線の後には最低 2 文字を指定する必要があります。異なる 16 進インジケータが必要な場合は、a パラメータを追加できます。

このコマンドは、フィールド・データを持つ任意のコマンド（すなわち ^FD、^FV（フィールド・データ）、および ^SN（連番データ）と一緒に使用できます。

有効な 16 進文字には以下があります。

0 1 2 3 4 5 6 7 8 9 A B C D E F a b c d e f

フォーマット ^FHa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 16 進インジケータ	*有効値：*現在のフォーマットおよびコントロール・プレフィックス以外の任意の文字（デフォルトでは ^ と ~） *デフォルト値：*_（下線）

例 • 次は、16 進値を ^FD ステートメントに直接入力する例です。

ZPL II CODE	GENERATED LABEL
^XA ^FO100,100 ^AD^FH ^FDTilde _7e used for HEX^FS ^XZ	Tilde ~ used for HEX
^XA ^FO100,100 ^AD^FH\ ^FDTilde \7E used for HEX^FS ^XZ	Tilde ~ used for HEX

^FM

複数のフィールド基点位置

説明 ^FM コマンドを使用すると、バー・コードの配置を制御できます。

構造化結合機能を使用する場合に、PDF417（^B7）と Micro-PDF417（^BF）バー・コードのフィールド位置を指定します。これにより、同じテキスト情報のセットから複数のバー・コードを印刷できます。

構造化結合機能は、両方のバー・コードのテキスト印刷機能を拡張する一方法です。ストリングがバー・コードのデータの制限を超える場合、1 of 3、2 of 3、3 of 3 のようにシリーズとして印刷できます。スキャナは情報を読み取り、セグメント化前の元のテキストに照合させます。

^FM コマンドは、^B7 および ^BF と使用した場合のみ、同じラベルでの複数のバー・コード印刷をトリガします。その他のコマンドと使用した場合は無視されます。

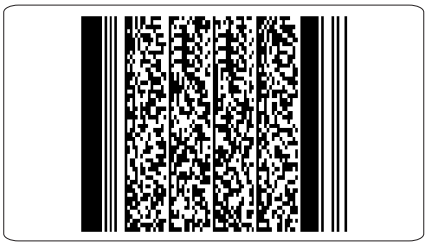
フォーマット ^FMx1,y1,x2,y2,...

次の表では、このフォーマットのパラメータを示します。

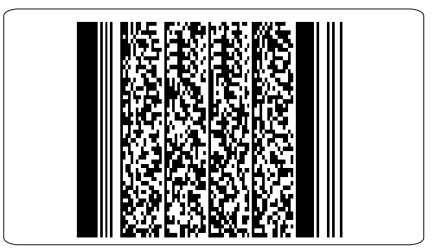
パラメータ	詳細
x1 = 最初のシンボルの x 軸の位置（ドット数）	*有効値：* 0 ～ 32000 e = このバー・コードを印刷から除外する *デフォルト値：*値を指定する必要があります
y1 = 最初のシンボルの y 軸の位置（ドット数）	*有効値：* 0 ～ 32000 e = このバー・コードを印刷から除外する *デフォルト値：*値を指定する必要があります

パラメータ	詳細
x2 = 2 番目のシンボルの x 軸の位置（ドット数）	x1 パラメータと同じ
y2 = 2 番目のシンボルの y 軸の位置（ドット数）	y1 パラメータと同じ
... = X、Y の組み合わせの継続	*組み合わせ最大数：60*

例 1 • 次の例では、最大 3 つのバー・コードを前提としています。

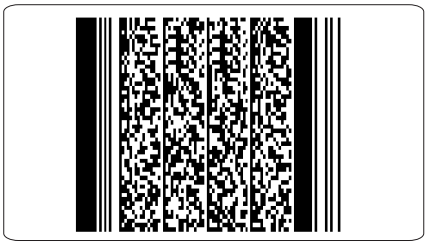
ZPL II CODE	GENERATED LABEL
^XA ^FM100,100,200,200,300,300 ^B7N,5,5,,50,N^FD<data>^FS ^XZ	

例 2 • 次の例は、最大 3 つのバー・コードを前提としていますが、バー・コード 2 of 3 は省略しています。

ZPL II CODE	GENERATED LABEL
^XA ^FM100,100,e,e,300,300 ^B7N,5,5,,50,N^FD<data>^FS ^FS	

x と y のいずれかの値に対して e を入力すると、バー・コードは印刷されません。

例 3 • 次の例でも、シンボル 2 of 3 は除外されています。

ZPL II CODE	GENERATED LABEL
^XA ^FM100,100,200,200,300,300 ^B7N,5,5,,50,N^FD<data>^FS ^XZ	

コメント 前のバー・コードのデータが制限値を超えると、後続のバー・コードが印刷されます。たとえば、1 of 3 が格納可能な最大データ量に達すると、バー・コード 2 of 3 が印刷されます。3 つのフィールドを指定しても、3 つのバー・コードが印刷されるわけではありません。3 つのバー・コード・フィールドを満たすのに十分なフィールド・データが必要です。

x と y の組み合わせの数は、生成されたバー・コードの数を超えてもかまいません。ただし、指定されたものが少なすぎる場合は、シンボルは印刷されません。

^FN

フィールド番号

説明 ^FN コマンドは、データ・フィールドに番号を付けるために使用します。このコマンドは、^DF（フォーマットの保存）と ^XF（フォーマットの呼び出し）コマンドの両方で使用されます。

保存されたフォーマットでは、通常 ^FD（フィールド・データ）コマンドを使用する場所に ^FN コマンドを使用します。保存されたフォーマットを呼び出す際、^FD と一緒に ^FN を使用してください。

フォーマット ^FN#

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
# = フィールドに割り当てる番号	*有効値：*0 ～ 9999 *デフォルト値：*0

例 • 次の例では、^DF を使って保存されたフォーマット（^XF）を呼び出し、フィールド番号データを挿入します。

ZPL II CODE	GENERATED LABEL
^XA ^XFR:STOREFMT.ZPL^FS ^FN1^FDZEBRA^FS ^FN2^FDLABEL^FS ^XZ	ZEBRA LABEL BUILT BY ZEBRA

コメント

- 複数の異なるフィールドに対して同じ ^FN 値を保存できます。

- ラベル・フォーマットに ^FN コマンドと ^FD コマンドを含む場合、そのフィールドのデータは同じ ^FN 値を含むその他のフィールドに対しても印刷されます。

^FO

フィールド基点

説明 ^FO コマンドは、ラベルのホーム（^LH）ポジションに対してフィールド基点を設定します。^FO は、回転にかかわらず、x 軸と y 軸に沿って点を定義することにより、フィールド領域の左上のコーナーを設定します。

フォーマット ^FOx,y

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
x = x 軸の位置（ドット数）	*有効値：*0 ～ 32000 *デフォルト値：*0
y = y 軸の位置（ドット数）	*有効値：*0 ～ 32000 *デフォルト値：*0

コメント x パラメータまたは y パラメータとして入力した値が大きすぎる場合は、フィールド基点がラベルから完全に外れて配置される可能性があります。

^FP

フィールド・パラメータ

説明 ^FP コマンドを使用すると、アジア系フォントで一般に使用されているフォント・フィールドの縦のフォーマットが可能になります。

フォーマット ^FPd,g

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = 方向	*有効値：* H = 縦方向印刷（左から右） V = 縦方向印刷（上から下） R = 逆方向印刷（右から左） *デフォルト値：*H
g = 追加の文字間ギャップ（ドット数）	*有効値：*0 ～ 9999 *デフォルト値：*値が入力されない場合は 0

例 • 次は、逆方向印刷と縦方向印刷を実装する例です。

ZPL II CODE	GENERATED LABEL
^XA ^FO100,50 ^FPV,10 ^AV ^FDvertical^FS ^XZ	v e r t i c a l
^XA ^FO350,50 ^FPR,10 ^AV ^FDreverse^FS ^XZ	esrever

コメント 逆方向印刷を使用する場合、^FT で指定された基点は右端のテキスト文字の左下コーナーです。

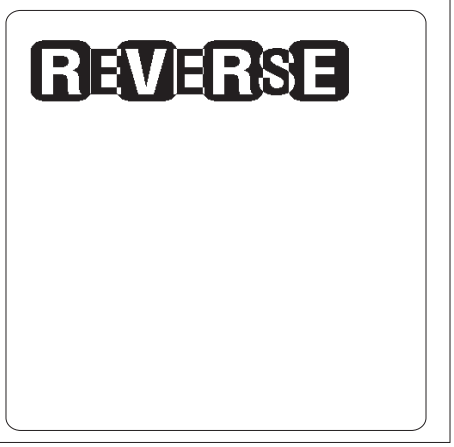
^FR

フィールド反転印刷

説明 ^FR コマンドを使用すると、フィールドを黒地に白、または白地に黒に表示できます。フィールドを印刷する場合、^FR コマンドを使用すると、出力の色は背景の逆になります。

フォーマット ^FR

例 • 次の例では、^GB コマンドが黒の領域を作成することにより、印刷対象が白くなるようになっています。

ZPL II CODE	GENERATED LABEL
^XA ^PR1 ^FO100,100 ^GB70,70,70,,3^FS ^FO200,100 ^GB70,70,70,,3^FS ^FO300,100 ^GB70,70,70,,3^FS ^FO400,100 ^GB70,70,70,,3^FS ^FO107,110^CF0,70,93 ^FR^FDREVERSE^FS ^XZ	

コメント ^FR コマンドは、1 つのフィールドにのみ適用し、毎回指定する必要があります。複数の ^FR コマンドを使用する場合、^LR コマンドを使用した方が便利な場合があります。

^FS

フィールド・セパレータ

説明 ^FS コマンドは、フィールド定義の終わりを示します。これに代わって、単一の ASCII コントロール・コード SI（Control-O、16 進 0F）として ^FS コマンドを発行することもできます。

フォーマット ^FS

^FT

フィールド・タイプセット

説明 ^FT コマンドは、^LH コマンドで指定したラベルのホーム・ポジションに対してフィールド位置を設定します。フィールドのタイプセット基点は、フィールドの内容に対して固定しており、回転によっては変更されません。

フォーマット ^FTx,y

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
x = x 軸の位置（ドット数）	*有効値：*0 ～ 32000 *デフォルト値:*最後にフォーマットされたテキスト・フィールドの位置
y = y 軸の位置（ドット数）	*有効値：*0 ～ 32000 *デフォルト値:*最後にフォーマットされたテキスト・フィールドの位置

テキスト 基点は、文字ストリングの先頭のフォントのベースラインにあります。通常ベースラインは、g や y などの下垂部のある文字を除きほとんどの文字の底辺です。

バー・コード バー・コードの下にテキスト表示がある場合も、バー・コードにガード・バーがある場合も、基点はバー・コードの底辺に位置します。

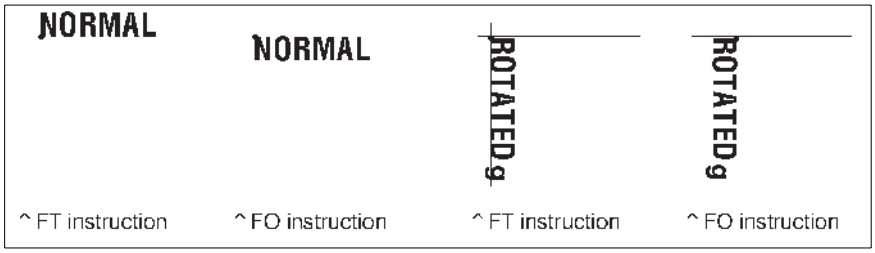
グラフィック・ボックス 基点はボックスの左下コーナーにあります。

イメージ 基点は、長方形のイメージ領域の左下コーナーにあります。

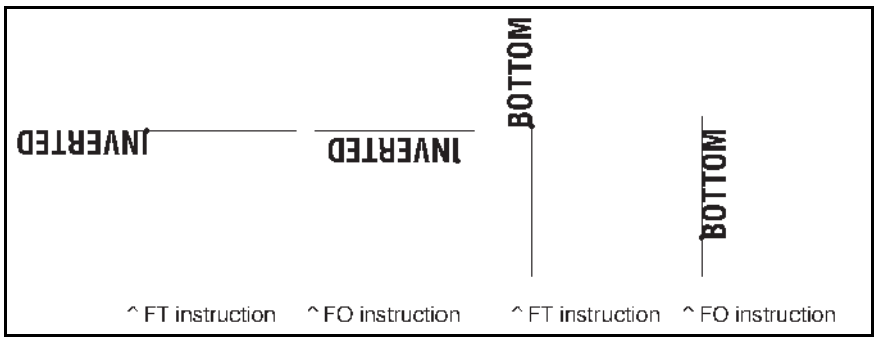
例 • 次の例では、^LH 位置に対して ^FT と ^FO を使用する場合のフォントの向きの違いを示しています。^FT コマンド使用時のフォントの基点は、常に、フィールドの最初のエレメントまたは文字のベースライン位置の左側になります。

標準の向きでは、すべての文字がベースラインに基づきます。回転状の向きでは、すべての文字はベースラインからラベルの右側に描かれます。反転した向きでは、すべての文字はベースラインから下に描かれ、左側に印刷されます。下向きでは、すべての文字はベースラインからラベルの左側に描かれ、右側に印刷されます。ドットは、^FT と ^FO の両方のフォントの向きの基点を示します。

スケーラブル標準フォントの向き、スケーラブル回転状フォントの向き



スケーラブル反転状フォントの向き、スケーラブルな下から上のフォントの向き



例 • 次は、^FT コマンドと連結の例です。

ZPL II CODE	GENERATED LABEL
^XA ^FT10,200^A0N,30,20^FDACME ^FS ^FT^GS^FDC^FS ^FT^A0N,30,20^FDSummer ^FS ^FT^A0N,60,50^FDClearance ^FS ^FT^A0N,120,100^FDSale ^FS ^XZ	

コメント 座標がない場合、最後にフォーマットされたフィールド位置が使用されます。この*記憶*によって、その他のフィールドの配置が単純化されます。最初のフィールドが配置されると、その他のフィールド位置は自動的に決定されます。

x および y パラメータを指定せずに ^FT コマンドを使用することをお勧めできない場合が次のようにいくつかあります。

- 最初のフィールドをラベル・フォーマットに配置する場合
- ^FN（フィールド番号）コマンドと一緒に使う場合
- ^SN（連番データ）コマンドの後に使用する場合

^FV

フィールド変数

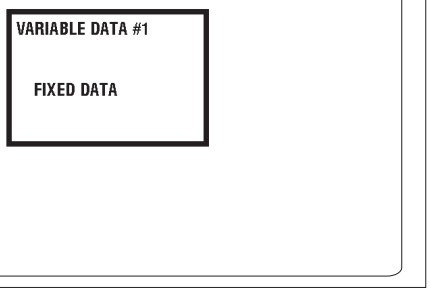
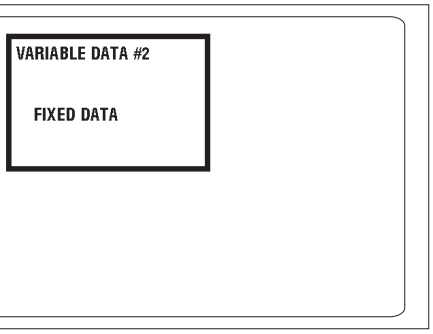
説明 ^FV は、フィールドが変数の場合、ラベル・フォーマットで ^FD（フィールド・データ）を置換します。

フォーマット ^FVa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a ＝印刷する変数フィールド・データ	*有効値：*0 ～ 3072 文字ストリング *デフォルト値：*データを入力しなかった場合は、このコマンドは無視されます。

例 • 次は、~MC コマンドと ^FV コマンドの使用例です。

ZPL II CODE	GENERATED LABEL
^XA ^FO40,40 ^GB300,203,8^FS ^FO55,60^CF0,25 ^FVVARIABLE DATA #1^FS ^FO80,150 ^FDFIXED DATA^FS ^MCN ^XZ	
^XA ^FO55,60^CF0,25 ^FVVARIABLE DATA #2^FS ^MCY ^XZ	

コメント ^FV フィールドは、ラベル印刷後に必ずクリアされます。^FD フィールドはクリアされません。

^FW

フィールドの向き

説明 ^FW コマンドは、向き（回転）パラメータを持つすべてのコマンド・フィールドのデフォルト向きを設定します。このコマンドを使用してフィールドを 0°、90°、180°、または 270° 時計方向に回転させることができます。

^FW コマンドは、その後に続くフィールドのみに影響します。いったん ^FW コマンドを発行すると、プリンタの電源を切るか、プリンタに新しい ^FW コマンドを送信するまでその設定が維持されます。

フォーマット ^FWr

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
r = フィールドの回転	*有効値：* N = 標準 R = 90° 回転 I = 180° 反転 B = 下から上へ読み取り、270° *パワーアップ時の初期値：*N

コメント ^FW コマンドは、r パラメータなしで入力された場合は無視されます。

^FW は、回転パラメータが明確に設定されていないコマンドの向きにのみ影響します。コマンドに特定の回転パラメータがある場合、その値が使用されます。

^FX

コメント

説明 ^FX コマンドは、ラベル・フォーマット内で*印刷対象外*の情報コメントまたはステートメントを追加したい場合に便利です。^FX コマンドの後、次のキャレット（^）またはティルディ（~）コマンドまでのデータは、ラベル・フォーマットには影響しません。したがって、^FX ステートメント内では、キャレット（^）またはティルディ（~）コマンドの使用は避けてください。

フォーマット ^FXc

次の表では、このフォーマットのパラメータを示します。

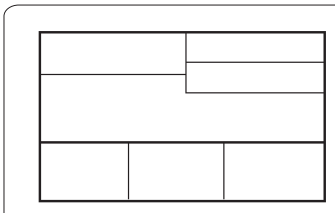
パラメータ	詳細
c = 印刷対象外のコメント	

例 • 次は、~FX コマンドを効果的に使用した例です。

ZPL II CODE

```
^XA
^LH100,100^FS
^FXSHIPPING LABEL^FS
^FO10,10^GB470,280,4^FS
^FO10,190^GB470,4,4^FS
^FO10,80^GB240,2,2^FS
^FO250,10^GB2,100,2^FS
^FO250,110^GB226,2,2^FS
^FO250,60^GB226,2,2^FS
^FO156,190^GB2,95,2^FS
^FO312,190^GB2,95,2^FS
^XZ
```

GENERATED LABEL



コメント ^FX コマンドの正しい使用法には、その後に ^FS コマンドを使用することが含まれます。

^GB

グラフィック・ボックス

説明 ^GB コマンドは、ラベル・フォーマットの一部としてボックスと線を描くために使用します。 ボックスと線は、 重要な情報を強調したり、 ラベルを別々の領域に区分したり、 ラベルの外観を改善したりするために使用します。 ボックスを描く場合も線を描く場合も同じフォーマット・コマンドが使用されます。

フォーマット ^GBw,h,t,c,r

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
w ＝ボックス幅（ドット数）	*有効値：* 1 ～ 32000 の値 *デフォルト値：* 高さに使用する値または 1
h ＝ボックスの高さ（ドット数）	*有効値：* 1 ～ 32000 の値 *デフォルト値：* 高さに使用する値または 1
t ＝線幅（ドット数）	*有効値：* 1 ～ 32000 *デフォルト値：* 1
c ＝線の色	*有効値：* B（黒）または W（白） *デフォルト値：* B
r ＝丸み付け角度	*有効値：* 0（丸み付けなし）～ 8（最大丸み付け） *デフォルト値：* 0

w と h パラメータについては、プリンタのデフォルトは 6、 8、 12、 24 ドット /mm のいずれかであることに留意してください。すなわち、インチあたりのドット数は 153、203、 300、 または 600 となります。 w と H の値を決定するには、 寸法をミリで計算し、 それぞれ 6、 8、 12、 24 で掛けてください。

幅と高さの指定がない場合は、t の値で指定した幅と高さの塗りつぶしボックスが印刷されます。

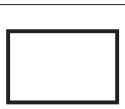
丸みインデックスを使用して、各ボックスの面取りの半径を決定します。数式:

丸み付け半径 = （丸みインデックス /8）＊（短辺 /2）

ここで、短辺とは幅と高さの短い方（最小値とデフォルト値に対しての調整後）です。

例 次にいくつかの例を示します。

幅：1.5 インチ、 高さ：1 インチ、 線幅み：10、 色：デフォルト、 丸み付け：デフォルト

ZPL II CODE	GENERATED LABEL
^XA ^FO50,50 ^GB300,200,10^FS ^XZ	

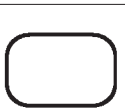
幅：0 インチ、高さ：1 インチ、線幅：20、色：デフォルト、丸み付け：デフォルト：

ZPL II CODE	GENERATED LABEL
^XA ^FO50,50 ^GB0,203,20^FS ^XZ	

幅：1 インチ、高さ：0 インチ、線幅：30、色：デフォルト、丸み付け：デフォルト

ZPL II CODE	GENERATED LABEL
^XA ^FO50,50 ^GB203,0,20^FS ^XZ	

幅：1.5 インチ、 高さ：1 インチ、 線幅：10、 色：デフォルト、 丸み付け：5

ZPL II CODE	GENERATED LABEL
^XA ^FO50,50 ^GB300,200,10,,5^FS ^XZ	

^GC

グラフィック円

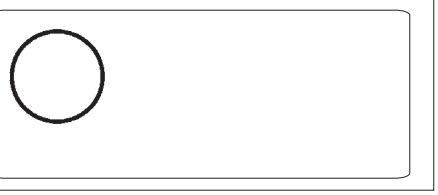
説明 ^GC コマンドは印刷ラベルに円を生成します。コマンド・パラメータは、円の直径（幅）、輪郭の線幅、および色を指定します。線幅は輪郭（外側）から内側へ増えます。

フォーマット ^GCd,t,c

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d＝円の直径（ドット数）	*有効値：*3 ～ 4095（大きい値は 4095 で置換します） *デフォルト値：*3
t＝線幅（ドット数）	*有効値：*2 ～ 4095 *デフォルト値：*1
c ＝線の色	*有効値：*B（黒）または W（白） *デフォルト値：*B

例 • 次は、印刷ラベルに円を作成する例です。

ZPL II CODE	GENERATED LABEL
^XA ^FO50,50 ^GC250,10,B^FS ^XZ	

^GD

グラフィック対角線

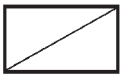
説明 ^GD コマンドは印刷ラベルに真っ直ぐの対角線を生成します。このコマンドを、その他のグラフィック・コマンドと一緒に使用することにより、さらに複雑な図形を作成できます。

フォーマット ^GBw,h,t,c,o

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
w = ボックス幅（ドット数）	*有効値：* 3 ～ 32000 *デフォルト値：* t （線幅）の値または 1
h = ボックスの高さ（ドット数）	*有効値：* 3 ～ 32000 *デフォルト値：* t （線幅）の値または 1
t = 線幅（ドット数）	*有効値：* 1 ～ 32000 *デフォルト値：* 1
c = 線の色	*有効値：* B（黒）または W（白） *デフォルト値：* B
o = 向き（対角線の方向）	*有効値：* R（または /）= 右に傾いた対角線 L（または \）= 左に傾いた対角線 *デフォルト値：* R

例 • 次は、印刷ラベル上のボックスで対角を結ぶ斜線を作成する例です。

ZPL II CODE	GENERATED LABEL
^XA ^FO150,100 ^GB350,203,10^FS ^FO155,110 ^GD330,183,10,,R^FS ^XZ	

^GE

グラフィック楕円

説明 ^GE コマンドはラベル・フォーマットに楕円を生成します。

フォーマット ^GEw,h,t,c

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
w = 楕円の幅（ドット数）	*有効値 :* 3 ～ 4095（大きい値は 4095 で置換します） *デフォルト値 :* 線幅（t）の値または 1
h = 楕円の高さ（ドット数）	*有効値 :* 3 ～ 4095 *デフォルト値 :* 線幅（t）の値または 1
t = 線幅（ドット数）	*有効値 :* 2 ～ 4095 *デフォルト値 :* 1
c = 線の色	*有効値 :* B（黒）または W（白） *デフォルト値 :* B

例 • 次は、印刷ラベルに楕円作成する例です。

ZPL II CODE	GENERATED LABEL
^XA ^FO100,100 ^GE300,100,10,B^FS ^XZ	

^GF

グラフィック・フィールド

説明 ^GF コマンドを使用すると、グラフィック・フィールド・データを直接プリンタのビットマップ保存領域にダウンロードできます。このコマンドは、その他のフィールドの規則に従います。すなわち、フィールドの向きが含まれます。グラフィック・フィールド・データはビットマップ・スペース内の任意の位置に配置できます。

フォーマット ^GFa,b,c,d,data

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 圧縮タイプ	*有効値：* A = ASCII 16 進（その他のダウンロード・コマンドのフォーマットに従います） B = バイナリ（c パラメータの後に送信されるデータは純バイナリです） C = 圧縮バイナリ（c パラメータの後に送信されるデータは圧縮バイナリ・フォーマットです。データは Zebra の圧縮アルゴリズムを使用してホスト側で圧縮されます。その後、データは解凍されビットマップに直接配置されます。） *デフォルト値：*A
b = バイナリ・バイト数	*有効値：*1 ～ 99999 これは、送信する全イメージのバイト数合計、またはパラメータ d の後に続くバイト数合計です。ASCII ダウンロードの場合、パラメータはパラメータ c と一致する必要があります。範囲外の値は最も近い限界値に設定されます。 *デフォルト値：*値を指定しなかった場合は、このコマンドは無視されます。

パラメータ	詳細
c = グラフィック・フィールド数	*有効値 :* 1 ～ 99999 これはパラメータ d として送信されるグラフィック・フォーマット（幅 x 高さ）を構成するバイト数合計です。この数値を 1 行あたりのバイト数で割ると、イメージ内のライン数が算出されます。この数値は、イメージのサイズを表すもので、必ずしもデータ・ストリームのサイズを表すものではありません（「d」を参照）。 *デフォルト値 :* 値を指定しなかった場合は、このコマンドは無視されます。
d = 1 行あたりのバイト数	*有効値 :* 1 ～ 99999 これはダウンロードされたデータのバイト数で、イメージの 1 行を構成します。 *デフォルト値 :* 値を指定しなかった場合は、このコマンドは無視されます。
data = データ	*有効値 :* ASCII 16 進データ：00 ～ FF ASCII 16 進数のストリング（1 イメージ・バイトあたり 2 桁）。読みやすくするための CR と LF を必要に応じて使用できます。2 桁の組み合わせの数は上の数と一致する必要があります。この数値が満たされた後に送信された数値は無視されます。データの中にカンマがある場合、現在のラインに *00*（余白）が追加され、送信されるデータが最小化されます。~DN またはキャレットやティルディ文字があると、ダウンロードが終了前に中断されます。 **バイナリ・データ**：純バイナリ・データがホストから送信されます。グラフィック・フォーマットに必要な全バイト数を受け取るまで、すべてのコントロール・プレフィックスは無視されます。

例 1 • 次の例では、合計 8,000 バイトのデータをダウンロードし、このグラフィック・データをビットマップの 100,100 の位置に配置します。プリンタに送信されるデータは ASCII フォーマットです。

```
^FO100,100^GFA,8000,8000,80,ASCII data
```

例 2 • 次の例では、合計 8,000 バイトのデータをダウンロードし、このグラフィック・データをビットマップの 100,100 の位置に配置します。プリンタに送信されるデータはバイナリ・フォーマットです。

```
^FO100,100^GFB,8000,8000,80,Binary data
```

^GS

グラフィック・シンボル

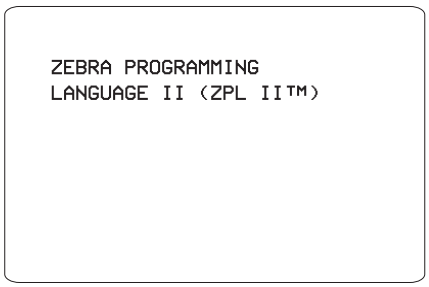
説明 ^GS コマンドを使用すると、登録商標、著作権シンボル、およびその他のシンボルを生成できます。

フォーマット ^GSo,h,w

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = フォントの向き	*有効値 :* N = 標準 R = 90° 回転（時計方向） I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値 :* N または ^FW 値
h = 幅に比例した文字の高さ（ドット数）	*有効値 :* 0 ～ 32000 *デフォルト値 :* 前回の ^CF 値
w = 高さに比例した文字の幅（ドット数）	*有効値 :* 0 ～ 32000 *デフォルト値 :* 前回の ^CF 値

例 • 希望の文字を生成するには、フィールド・データ内で ^GS コマンドの後に ^FD および適切な文字（A ～ E）を使用します。

ZPL II CODE	GENERATED LABEL
^XA^CFD ^FO50,50 ^FDZEBRA PROGRAMMING^FS ^FO50,75 ^FDLANGUAGE II (ZPL II)^FS ^FO280,75 ^GS^FDC^FS ^XZ	

A = ® (Registered Trade Mark)

B = © (Copyright)

C = ™ (Trade Mark)

D = (Underwriters Laboratories approval)

E = (Canadian Standards Association approval)

~HB

バッテリ・ステータス

説明 ~HB コマンドがプリンタに送信されると、データ・ストリングがホストに送り返されます。ストリングは、<STX> コントロール・コード・シーケンスで始まり、<ETX><CR><LF> コントロール・コード・シーケンスで終わります。

フォーマット ~HB

パラメータ：プリンタはコマンドを受け取ると、以下を返します。

<STX>bb.bb,hh.hh,bt<ETX><CR><LF>

<STX>	=	ASCII テキスト開始文字
bb.bb	=	1/4 ボルトに最も近い現在のバッテリ電圧
hh.hh	=	1/4 ボルトに最も近い現在のヘッド電圧
bt	=	バッテリ温度（摂氏）
<ETX>	=	ASCII テキスト終了
<CR>	=	ASCII キャリッジ・リターン
<LF>	=	ASCII ライン・フィード文字

コメント このコマンドは、プリンタの電源バッテリに使用するもので、バッテリ・バックアップによる RAM と混同しないようにしてください。

~HD

ヘッド温度情報

説明 ~HD コマンドは、端末エミュレータを使用して、電源とヘッド温度などのプリンタ・ステータス情報をエコーします。

フォーマット ~HD

例 • 次は、^HD コマンドの例です。

```
Head Temp = 29
Ambient Temp = 00
Head Test = Passed
Darkness Adjust = 23
Print Speed = 2
Slew Speed = 6
Backfeed Speed = 2
Static_pitch_length = 0521
Dynamic_pitch_length = 0540
Max_dynamic_pitch_length = 0540
Min_dynamic_pitch_length = 0537
COMMAND PFX = ~ : FORMAT PFX = ^ : DELIMITER = ,
P30 INTERFACE = None
P31 INTERFACE = None
P32 INTERFACE = Front Panel              Revision 5
P33 INTERFACE = None
P34 INTERFACE = None
P35 INTERFACE = None
Dynamic_top_position = 0008

No ribbon A/D = 0000
```

^HF

グラフィック・シンボル

説明 ^HF コマンドは、ZPL フォーマット命令のオブジェクトを ~DF でホストに送信します。

フォーマット ^HF,o,h,w

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
o = フォントの向き	*有効値：* N = 標準 R = 回転 I = 180° 反転 B = 下から上へ読み取り、270° *デフォルト値：*N または前回の ^FW 値
h = 幅に比例した文字の高さ（ドット数）	*有効値：*0 ～ 32000 *デフォルト値：*前回の ^CF 値
w = 高さに比例した文字の幅（ドット数）	*有効値：*0 ～ 32000 *デフォルト値：*前回の ^CF 値

^HG

ホスト・グラフィック

説明 ^HG コマンドは、ホストにグラフィックスをアップロードする場合に使用します。グラフィック・イメージは、将来使用するために保存することも、Zebra プリンタにダウンロードすることもできます。

フォーマット ^HGd:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = オブジェクトのデバイスの場所	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* 検索優先順序
o = オブジェクト名	*有効値 :* 1 ～ 8 文字の英数字 *デフォルト値 :* 名前を指定しない場合 UNKNOWN が使用されます。
x = 拡張子	*固定値 :* .GRF

コメント グラフィックスのアップロードの詳細については、*^HY ページ 215* を参照してください。

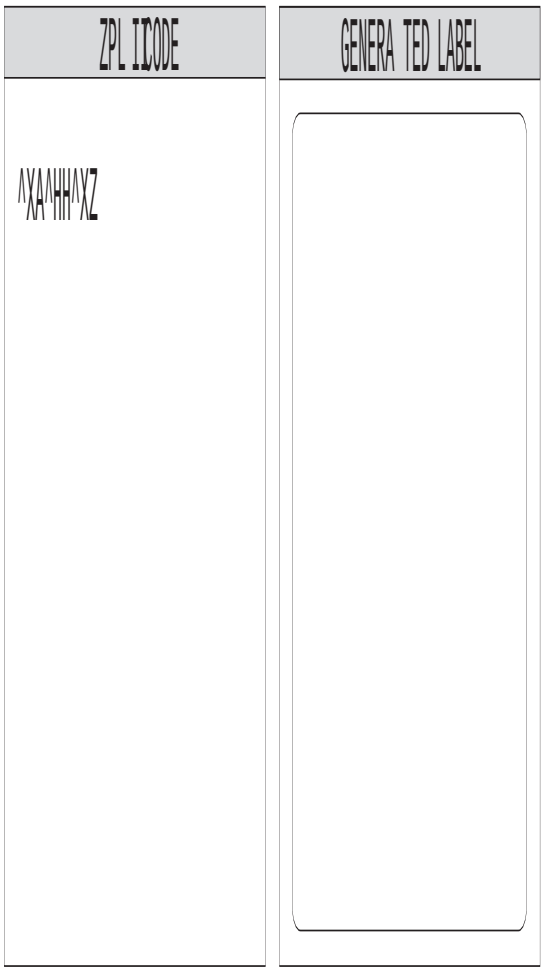
^HH

設定ラベルの返却

説明 ^HH コマンドは、端末エミュレータを使用してプリンタ設定をホストにエコー・バックします。

フォーマット ^HH

例 • 次は、^HH コマンドの例です。



~HI

ホスト識別

説明 ~HI コマンドは、ホストから Zebra プリンタに送信され情報を取得するように設計されています。受信と同時に、プリンタはモデル、ソフトウェア・バージョン、ドット /mm の設定、メモリ・サイズ、および検出オブジェクトなどの情報を返します。

フォーマット ~HI

パラメータ プリンタはコマンドを受け取ると、以下を返します。

```
XXXXXX,V1.0.0,dpm,000KB,X
```

XXXXXX = Zebra プリンタのモデル

V1.0.0 = ソフトウェアのバージョン

dpm = ドット /mm

6、8、12、または 24 のドット /mm の印刷ヘッド

000KB = メモリ

512KB = 1/2 MB

1024KB = 1 MB

2,048KB = 2 MB

4,096KB = 4 MB

8,192KB = 8 MB

x = 認識可能オブジェクト

プリンタに固有のオプションのみが表示されます（カッター、オプションなど）。

~HM

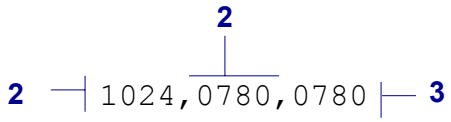
ホスト RAM ステータス

説明 プリンタに ~HM を送信すると、ホストに対してメモリ・ステータス・メッセージが即座に返されます。このコマンドは、プリンタの RAM ステータスを確認する必要がある場合に使用します。

~HM が Zebra プリンタに送信されると、メモリの全容量、最大容量、および使用可能容量に関する情報を含むデータのラインがホストに送り返されます。

フォーマット ~HM

例 • 次の例では、~HM がプリンタに送信されたときに、3 つの数値を含むデータがホストに送り返されます。各数値のセットは、下の表で識別し、説明しています。



1	プリンタにインストールされた RAM の全容量（キロバイト）。この例では、プリンタに 1024K の RAM がインストールされています。
2	ユーザーが使用できる RAM の最大容量（キロバイト）。この例では、プリンタで最大 780K の RAM が使用可能です。
3	現在ユーザーが使用できる RAM の容量（キロバイト）。この例では、現在プリンタにユーザーが使用できる RAM が 780K あります。

コメント 現在使用可能なメモリの値には、ビットマップが使用しているメモリが含まれています（^MCN のため）。

グラフィック・イメージ、フォントのダウンロードや、ビットマップの保存は、RAM の容量にのみ影響します。RAM の全容量と RAM の最大容量は、プリンタをオンにした後は変りません。

~HS

ホスト・ステータスの返却

説明 ~HS コマンドがプリンタに送信されると、3 つのデータ・ストリングがホストに送り返されます。各ストリングは、<STX> コントロール・コードで始まり、<ETX><CR><LF> コントロール・コード・シーケンスで終わります。混乱を避けるため、各ストリングはホストによって別々のラインに印刷されます。

ストリング 1

```
<STX>aaa,b,c,dddd,eee,f,g,h,iii,j,k,l<ETX><CR><LF>
```

aaa	=	通信（インターフェイス）設定 *
b	=	用紙切れフラグ（1 = 用紙切れ）
c	=	ポーズ・フラグ（1 = ポーズがアクティブ）
dddd	=	ラベル長（ドット数の値）
eee	=	受信バッファ内のフォーマット数
f	=	*バッファ満杯*フラグ（1 = 受信バッファが満杯）
g	=	*通信診断モード*フラグ（1 = 診断モードがアクティブ）
h	=	*部分フォーマット*フラグ（1 = 部分フォーマット進行中）
iii	=	未使用（常に 000）
j	=	*破損 RAM*フラグ（1 = 設定データ紛失）
k	=	温度範囲（1 = 過剰低温）
l	=	温度範囲（1 = 過剰高温）

* このパラメータは、プリンタのボー・レート、データ・ビット数、ストップ・ビット数、パリティ設定、およびハンドシェイクのタイプを指定します。この値は、8 ビットのバイナリ数値の 3 桁の 10 進表現です。このパラメータを評価するには、まず、10 進数をバイナリ数に変換します。

9 桁のバイナリ数は、次の表に従って読み取られます。

aaa = a^8 a^7 a^6 a^5 a^4 a^3 a^2 a^1 a^0	
a^7 = Handshake 0 = Xon/Xoff 1 = DTR	a^8 a^2 a^1 a^0 = Baud 0 000 = 110 0 001 = 300 0 010 = 600 0 011 = 1200 0 100 = 2400 0 101 = 4800 0 110 = 9600 0 111 = 19200 1 000 = 28800 *(available only on certain printer models)* 1 001 = 38400 *(available only on certain printer models)* 1 010 = 57600 *(available only on certain printer models)* 1 011 = 14400
a^6 = Parity Odd/Even 0 = Odd 1 = Even	
a^5 = Disable/Enable 0 = Disable 1 = Enable	
a^4 = Stop Bits 0 = 2 Bits 1 = 1 Bit	
a^3 = Data Bits 0 = 7 Bits 1 = 8 Bits	

ストリング 2

```
<STX>mmm,n,o,p,q,r,s,t,uuuuuuuu,v,www<ETX><CR><LF>
```

---

mmm = 機能設定 *

n = 未使用

o = *ヘッド上*フラグ（1 = ヘッドが上のポジション）

p = *リボン切れ*フラグ（1 = リボン切れ）

q = *熱転写モード*フラグ（1 = 熱転写モード選択済み）

r = 印字モード

0 = 巻き取り

1 = 剥離

2 = 切り取り

3 = カッター

4 = アプリケ - タ

---

s	=	印字幅モード
t	=	*ラベル待機中*フラグ（1 = ラベルが剥離モードで待機中）
uuuuu uuu	=	バッチの残りのラベル
v	=	*印刷中フォーマット*フラグ（常に 1）
www	=	メモリに保存されたグラフィック・イメージの数

＊このパラメータは、プリンタのメディア・タイプ、センサー・プロフィール・ステータス、および通信診断ステータスを指定します。ストリング 1 と同様に、これは 8 ビットのバイナリ数値の 3 桁の 10 進表現です。まず、10 進数をバイナリ数に変換します。

8 桁のバイナリ数は、次の表に従って読み取られます。

mmm = m7 m6 m5 m4 m3 m2 m1 m0	
m7 = Media Type 0 = Die-Cut 1 = Continuous	m4 m3 m2 m1 = Unused 0 = Off 1 = On
m6 = Sensor Profile 0 = Off	m0 = Print Mode 0 = Direct Thermal 1 = Thermal Transfer
m5 = Communications Diagnostics 0 = Off 1 = On	

ストリング 3

```
<STX>xxxx,y<ETX><CR><LF>
```

xxxx	=	パスワード
y	=	0（静的 RAM はインストールされていない）
		1（静的 RAM はインストールされている）

~HU

ZebraNet AlertConfiguration の返却

説明 このコマンドは、設定済みの ZebraNet Alertsettings をホストに返します。

フォーマット ~HU

例 • 既存の Alertmessages が電子メールおよび SNMP トラップに行くように設定されたプリンタに ~HU コマンドが、 送信された場合、 下の情報のようなデータが返されます。 個々のパラメータ設定の詳細については、*^SX ページ 329* を参照してください。

```
B,C,Y,Y,ADMIN@COMPANY.COM,0
J,F,Y,Y,,0
C,F,Y,Y,,0
D,F,Y,Y,,0
E,F,Y,N,,0
F,F,Y,N,,0
H,C,Y,N,ADMIN@COMPANY.COM,0
N,C,Y,Y,ADMIN@COMPANY.COM,0
O,C,Y,Y,ADMIN@COMPANY.COM,0
P,C,Y,Y,ADMIN@COMPANY.COM,0
```

最初のラインは、条件 B（リボン切れ）は宛先 C（電子メールアドレス）にルーティングされます。

次の 2 文字の Y と Y は、*条件設定* と *条件クリア* オプションが「はい」に設定されていることを意味します。

その次のエントリは、アラート電子メールの送り先（この例では admin@company.com）です。

最初のラインの最後の数値の 0 はポート番号です。

各ラインは、^SX コマンドで定義された異なるアラート条件の設定を示しています。

^HW

ホスト・ディレクトリ・リスト

説明 ^HW は、特定のメモリ領域内のオブジェクトのディレクトリ・リストをホスト・デバイスに送り返すために使用します。このコマンドは、オブジェクト名のフォーマット済み ASCII ストリングをホストに返します。

各オブジェクトは、ライン上にリストされ、固定長を持ちます。ラインの全長も固定しています。オブジェクトをリストする各ラインは、アステリスク（*）で始まり、空白のスペースが続きます。オブジェクト名には 8 つのスペースがあり、その後にピリオド、そして、拡張子用のスペースが 3 つ続きます。拡張子の後には 2 つの空白スペース、オブジェクト・サイズ用の 6 つのスペース、2 つの空白スペース、そしてオプション・フラグ用の 3 つのスペース（将来のために予約）が続きます。フォーマットは次のようになります。

```
<STX><CR><LF>
DIR R:<CR><LF>
*Name.ext(2sp.)(6 obj. sz.)(2sp.)(3 option flags)
*Name.ext(2sp.)(6 obj. sz.)(2sp.)(3 option flags)
<CR><LF>
-xxxxxxx bytes free
<CR><LF>
<ETX>
```

<STX> = テキストの開始

<CR><LR> = キャリッジ・リターン / ライン・フィード

<ETX> = テキストの終了

このコマンドは、スタンドアロンのファイルで使用して、いつでもプリンタに対して発行できます。プリンタは、コマンド受信時に実行していたその他のタスクに基づき、即刻ディレクトリ・リストを返します。

このコマンドは、すべての ^（キャレット）コマンドと同様に、プリンタでの受信順序に従って処理されます。

フォーマット ^HWd:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = オブジェクト・リストを取得する場所	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* R:
o = オブジェクト名	*有効値 :* 1 ～ 8 文字の英数字 *デフォルト値 :* アステリスク（*）疑問符（?）も使用できます。
x = 拡張子	*有効値 :* Zebra の規則に準拠したすべての拡張子 *デフォルト値 :* アステリスク（*）疑問符（?）も使用できます。

例 • 次は、R: から情報を取得する ^HW コマンドの例です。

```
^XA
^HWR:*.*
^XZ
```

例 • プリンタは、ホスト・ディレクトリ・リスト -DIR R:*.* としてこの情報を返しました。

```
*R:ARIALN1.FNT 49140
*R:ARIALN2.FNT 49140
*R:ARIALN3.FNT 49140
*R:ARIALN4.FNT 49140
*R:ARIALN.FNT 49140
*R:ZEBRA.GRF 8420

-794292 bytes free R:RAM
```

^HY

グラフィックのアップロード

説明 ^HY コマンドは ^HG コマンドを拡張したものです。^HY は、サポートされるすべてのフォーマットのグラフィック・オブジェクトをプリンタにアップロードするために使用します。

フォーマット ^HYd:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = オブジェクトの場所	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* 検索優先順序
o = オブジェクト名	*有効値 :* 1 ～ 8 文字の英数字 *デフォルト値 :* オブジェクト名を指定する必要があります
x = 拡張子	*有効値 :* G = .GRF（生のビットマップ・フォーマット） P = .PNG（圧縮ビットマップ・フォーマット） *デフォルト値 :* 保存されたイメージのフォーマット

コメント イメージは ~DY コマンドの形でアップロードされます。返された ~DY コマンドのデータ・フィールドは、常に ZB64 フォーマットでエンコードされます。

^HZ

説明情報の表示

説明 ^HZ コマンドは、プリンタの説明情報を XML フォーマットで返すために使用します。プリンタは、フォーマット・パラメータ、オブジェクト・ディレクトリ、個々のオブジェクト・データなどに関する情報、および印刷のステータス情報を返します。詳細については、*『ZPL プログラミング・ガイド第 2 巻』* の第 6 章（HML スーパー・ホストのステータス）を参照してください。

フォーマット ^HZd

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d＝返される表示内容	*有効値：* a ＝すべての情報を表示する f ＝プリンタのフォーマット設定情報を表示する l ＝オブジェクト・ディレクトリ・リスト情報を表示する o ＝個々のオブジェクト・データ情報を表示する オブジェクト情報は、R:、E:、B:、および A: から呼び出すことができます。オブジェクト名と拡張子は標準 Zebra 命名規則に従います（下のコメント参照）。このパラメータの後には、次のようにカンマ、ドライブの場所、オブジェクト名、およびオブジェクト拡張子が必要です。 ^XA ^HZO,R:SAMPLE.GRF ^XZ r ＝プリンタのステータス情報を表示する *デフォルト値：*値が欠落しているか無効な場合は、このコマンドは無視されます。

コメント オブジェクト（パラメータ 0）に対してサポートされる拡張子には以下があります。

- .FNT -- フォント
- .GRF -- グラフィック
- .PNG -- 圧縮グラフィック
- .ZPL -- 保存されたフォーマット
- .DAT -- エンコード表
- .ZOB -- ダウンロード可能オブジェクト
- .STO -- Alertdata ファイル
- .BAZ -- すべての ZBI 実行可能ファイルのセキュリティ
  - .BAZ の詳細については、*ZBI ファイルの暗号化 ページ 402* を参照してください。

^ID

オブジェクト削除

説明 ^ID コマンドは、保存領域からオブジェクト、グラフィックス、フォント、および保存されたフォーマットを削除します。オブジェクトは選択的にも、グループとしても削除できます。このコマンドは、新しいオブジェクトを保存する前に印刷フォーマット内で使用して、またはスタンドアロン・フォーマットで使用してオブジェクトを削除できます。

イメージ名と拡張子では、ワイルドカードとしてアステリスク（*）の使用がサポートされています。これにより、選択したオブジェクトのグループを簡単に削除できます。

フォーマット ^IDd:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = 保存されたオブジェクトの場所	*有効値：* R:、E:、B:、および A: *デフォルト値：* R:
o = オブジェクト名	*有効値：* 1 ～ 8 文字の任意の名前 *デフォルト値：* 名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子	*有効値：* Zebra の規則に準拠したすべての拡張子 *デフォルト値：* .GRF

例 1 • 保存されたフォーマットを DRAM から削除する場合

```
^XA
^IDR:*.ZPL^FS
^XZ
```

例 2 • 拡張子にかかわらず、DRAM から SAMPLE という名前のフォーマットとイメージを削除する場合

```
^XA
^IDR:SAMPLE.*^FS
^XZ
```

例 3 • SAMPLE2.GRF を保存する前にイメージ SAMPLE1.GRF を削除する場合

```
^XA
^FO25,25^AD,18,10
^FDDelete^FS
^FO25,45^AD,18,10
^FDthen Save^FS
^IDR:SAMPLE1.GRF^FS
^ISR:SAMPLE2.GRF^FS^XZ
```

例 4 • この例では、* はワイルドカードで、.GRF 拡張子の付いたすべてのオブジェクトが削除されることを意味します。

```
^XA
^IDR:*.GRF^FS
^XZ
```

コメント オブジェクトは、R: からが削除されると、使用できなくなり、メモリは保存用に使用可能になります。これは R: にのみ適用します。

さらに ^ID コマンドは、DRAM 内のオブジェクトの圧縮されていないバージョンを使用可能にします。

名前を *.ZOB として指定すると、ダウンロードされたバー・コード・フォント（またはその他のオブジェクト）のすべてが削除されます。

指定のダウンロード可能オブジェクトが R:、E:、B:、および A: デバイスに見つからない場合は、^ID コマンドは無視されます。

^IL

イメージ読み込み

説明 ^IL コマンドは、ラベル・フォーマットの開始時に使用して、フォーマットの保存されたイメージを読み込み、追加のデータと結合します。イメージは常に ^FO0,0 に配置されます。

重要 • *^IS ページ 224* を参照してください。

この技法を使用して不変情報のイメージを変数データでオーバーレイすると、ラベル・フォーマットのスループットが大きく改善されます。

フォーマット ^ILd:o.x

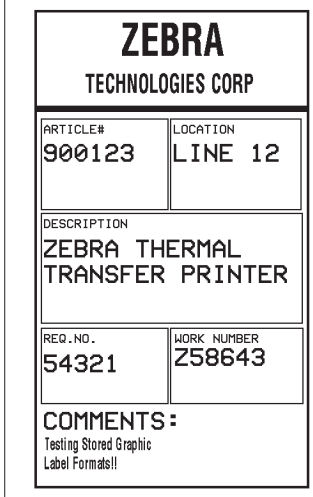
次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = 保存されたオブジェクトの場所	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* R:
o = オブジェクト名	*有効値 :* 1 ～ 8 文字の英数字 *デフォルト値 :* 名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子	*固定値 :* .GRF

例 • この例は、保存されたイメージ SAMPLE2.GRF を DRAM から呼び出し、追加のデータでオーバーレイします。グラフィックは ^IS コマンドを使用して保存されました。保存されたラベル・フォーマットについては、*^IS ページ 224* の コマンドを参照してください。

ZPL II CODE	GENERATED LABEL

```
^XA
^ILR:SAMPLE2.GRF^FS
^CFD,36,20
^FO15,210
^FD900123^FS
^FO218,210
^FDLINE 12^FS
^FO15,360^AD
^FDZEBRA THERMAL^FS
^FO15,400^AD
^FDTRANSFER PRINTER^FS
^FO15,540
^FD54321^FS
^FO220,530
^FDZ58643^FS
^FO15,670^A0,27,18
^FDTesting Stored Graphic^FS
^FO15,700^A0,27,18
^FDLabel Formats!!^FS
^XZ
```



^IM

イメージの移動

説明 ^IM コマンドは、保存領域からビットマップへのイメージの直接移動を実行します。コマンドは、^XG（グラフィックの呼び出し）コマンドと同じですが、サイズ・パラータがない点が異なります。

フォーマット ^IMd:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = 保存されたオブジェクトの場所	*有効値：* R:、E:、B:、および A: *デフォルト値：* 検索優先順序
o = オブジェクト名	*有効値：* 1 ～ 8 文字の英数字 *デフォルト値：* 名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子	*固定値：* .GRF

例 • 次の例では、DRAM からイメージ SAMPLE.GRF を移動して、複数の場所で元のサイズを使って印刷します。

```
^XA
^FO100,100^IMR:SAMPLE.GRF^FS
^FO100,200^IMR:SAMPLE.GRF^FS
^FO100,300^IMR:SAMPLE.GRF^FS
^FO100,400^IMR:SAMPLE.GRF^FS
^FO100,500^IMR:SAMPLE.GRF^FS
^XZ
```

コメント ^FO コマンドを使用することにより、ラベルのどこにでもグラフィック・イメージを配置できます。

^IM と ^XG の違いは次のとおりです。^IM には倍率がないため、フォーマットの所要時間が短くなる可能性があります。ただし、これを利用するには、イメージが 8、16、または 32 ビット・バウンダリである必要があります。

^IS

イメージの保存

説明 ^IS コマンドは、ラベル・フォーマット内で使用して、そのフォーマットを ZPL II スクリプトではなくグラフィック・イメージとして保存します。このコマンドは、通常スクリプトの終わりのほうで使用します。保存されたイメージは、実質上のフォーマット時間を要せずに、後で呼び出し、変数データでオーバーレイして完全なラベルを形成することができます。

この技法を使用して不変情報のイメージを変数データでオーバーレイすると、ラベル・フォーマットのスループットが大きく改善されます。

重要 • *^IL ページ 220* を参照してください。

フォーマット ^ISd:o.x,p

次の表では、このフォーマットのパラメータを示します。

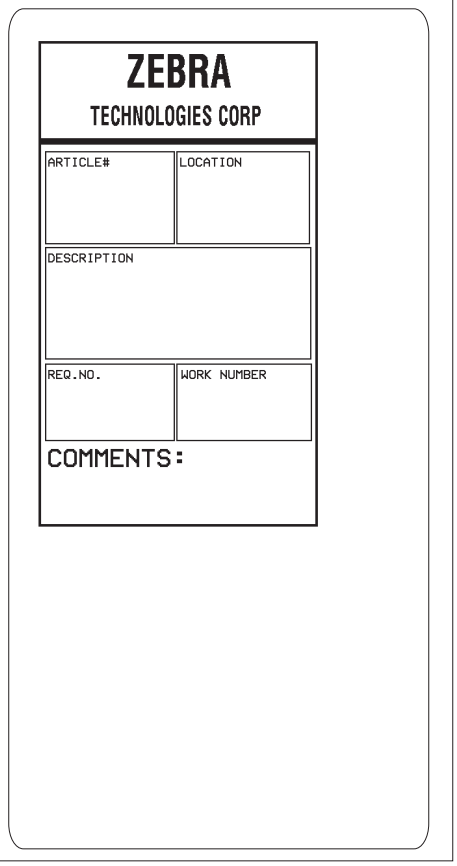
パラメータ	詳細
d = 保存されたオブジェクトの場所	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* R:
o = オブジェクト名	*有効値 :* 1 ～ 8 文字の英数字 *デフォルト値 :* 名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子	*有効値 :* .GRF または .PNG *デフォルト値 :* .GRF
P = 保存の後でイメージを印刷する	*有効値 :* Y（はい）または N （いいえ） *デフォルト値 :* Y

例 • 次は、^IS コマンドを使用してラベル・フォーマットを DRAM に保存する例です。グラフィックの保存に使用する名前は SAMPLE2.GRF です。

ZPL II CODE

```
^XA
^LH10,15^FWN^BY3,3,85^CFD,36
^GB430,750,4^FS
^FO10,170^GB200,144,2^FS
^FO10,318^GB410,174,2^FS
^FO212,170^GB206,144,2^FS
^FO10,498^GB200,120,2^FSR
^FO212,498^GB209,120,2^FS
^FO4,150^GB422,10,10^FS
^FO135,20^A0,70,60
^FDZEBRA^FS
^FO80,100^A0,40,30
^FDTECHNOLOGIES CORP^FS
^FO15,180^CFD,18,10^FS
^FDARTICLE#^FS
^FO218,180
^FDLOCATION^FS
^FO15,328
^FDDESCRIPTION^FS
^FO15,508
^FDREQ.NO.^FS
^FO220,508
^FDWORK NUMBER^FS
^FO15,630^AD,36,20
^FDCOMMENTS:^FS
^ISR:SAMPLE2.GRF,Y
^XZ
```

GENERATED LABEL



~JA

すべてキャンセル

説明 ^JA コマンドは、バッファ内にあるすべてのフォーマット・コマンドをキャンセルします。また、印刷中のすべてのバッチをキャンセルします。

現在のラベルの印刷が終了すると、プリンタは停止します。すべての内部バッファからデータがクリアされ、DATA LED がオフになります。

このコマンドをプリンタに送信すると、バッファがスキャンされ、入力バッファ内で ~JA の前のデータのみが削除されます。つまり、その他の ~JA コマンドを調べるために残りのバッファはスキャンされません。

フォーマット ~JA

^JB

フラッシュ・メモリの初期化

説明 ^JB コマンドは、Zebra プリンタで使用できる 2 種類のフラッシュ・メモリを初期化するために使用します。

フォーマット ^JBa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 初期化するデバイス	*有効値 :* B = フラッシュ・カード （PCMCIA） E = 内部フラッシュ・メモリ *デフォルト値 :* デバイスを指定する必要があります。

例 • 次は、異なる種類のフラッシュ・メモリを初期化する例です。

^JBE – オプションのフラッシュ・メモリを初期化します (プリンタにそれがインストールされている場合)。

^JBE – オプションのフラッシュ・メモリを初期化します (プリンタにそれがインストールされている場合)。

^JBA – コンパクト・フラッシュ・メモリを初期化します (プリンタにそれがインストールされている場合)。

~JB

オプションの・メモリのリセット

説明 ~JB コマンドは、次の条件に対して使用します。

- メモリ・カードに電源を供給しているバッテリが故障し、置換された場合に、~JB コマンドをプリンタに送信する必要があります。不良なバッテリの場合、プリンタの設定ラベルに「ﾊﾞｯﾃﾘｷﾚ」の状態が表示されます。
- ~JB コマンドは、B: を意図的にクリア（再初期化）するためにも使用できます。カードは、**書き込み禁止でない**必要があります。

フォーマット ~JB

コメント バッテリを置換しても、このコマンドをプリンタに送信しない場合、メモリ・カードは機能できません。

~JC

用紙センサーのキャリブレーションの設定

説明 ~JC コマンドは、ラベル長の測定を強制し、用紙センサーとリボン・センサーの値を調整します。

フォーマット ~JC

コメント 連続モードでは、用紙センサーとリボン・センサーのみがキャリブレートされます。

~JD

通信診断の有効化

説明 ~JD コマンドは、現在のラベル長とプリンタの最大幅を使用して、プリンタで受信されたすべての文字の ASCII 出力を生成する診断モードを開始します。出力には、ASCII 文字、16 進値、および通信エラーが含まれます。

フォーマット ~JD

~JE

診断の無効化

説明 ~JE コマンドは診断モードをキャンセルし、プリンタを標準のラベル印刷に戻します。

フォーマット ~JE

~JF

バッテリの状態の設定

説明 *PA/PT400*a プリンタで検知されるバッテリ減少の電圧レベルは 2 つあります。バッテリの電圧が最初のレベルよりも下がると、警告として緑色の LED が点滅を開始しますが、印刷は続行されます。この警告が発生したら、バッテリを再充電してください。

印刷を続けると、2 番目の電圧レベルに到達します。この時点で、緑色とオレンジ色の LED の両方が警告としてフラッシュし、印刷は自動的に一時停止されます。

低電圧時のポーズがアクティブ（~JFY）な場合に、バッテリの電圧レベルが 2 番目の*低電圧* レベルより下がると、印刷は一時停止され、エラー状態の表示により、プリンタをバッテリ充電器に接続する必要があることを示します。FEED を押すことにより、印刷はラベル単位で続行できますが、バッテリの電圧が継続的に減少しているため、ラベル・フォーマット情報を失う可能性があります。

低電圧時のポーズがアクティブではない場合（~JFN）に、バッテリの電圧レベルが 2 番目の*低電圧*レベルより下がっても、印刷が続行され、オレンジの LED も点灯しません。バッテリの電圧が引続き減少すると、ラベル情報が失われ、プリンタが停止する原因となります。このオプションは、プリンタが車のバッテリ・アダプタに接続している場合にのみ選択してください。プリンタは、バッテリの電圧が最初の*低電圧*レベル以下であることを検知することがありますが、車のバッテリの連続的な再充電のため、バッテリ電圧の減少は問題ではなく、印刷は続行されます。

車のバッテリ・アダプタを使用する際、このオプションが選択されていない場合は、FEED を押して、プリンタのポーズ・モードを取り消して、各ラベルを印刷する必要があることがあります。

フォーマット ~JFp

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
P = 低電圧時にポーズする	*有効値:* Y（低電圧時にポーズする）または N（ポーズしない） プリンタが車のバッテリ・アダプタを電源としている場合は、N をお勧めします。 *デフォルト値 :* Y

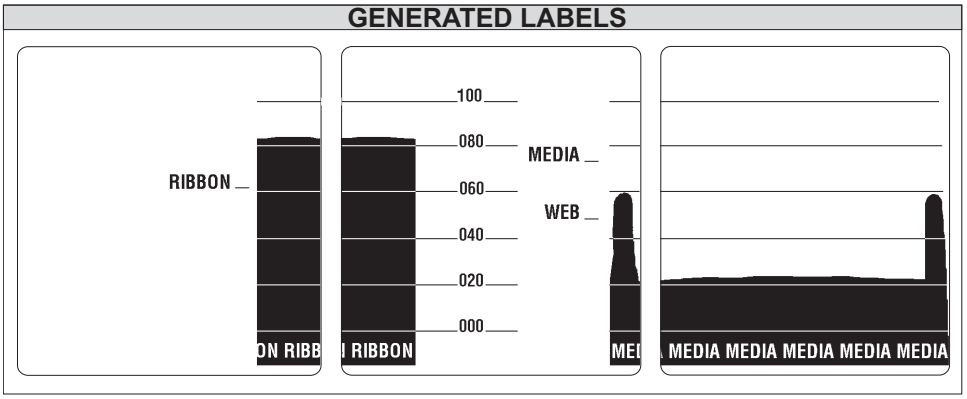
~JG

センサーのキャリブレーションのグラフ化

説明 ~JG コマンドは、ラベル長の測定を強制し、用紙センサーとリボン・センサーを再キャリブレーションし、センサーの値のグラフ（用紙センサー・プロフィール）を印刷する場合に使用します。

フォーマット ~JG

例 • ~JG コマンドをプリンタに送信すると、次のイメージに似た一連のラベルが生成されます。



^JI

ZBI（Zebra Basic Interpreter）の開始

説明 ^JI は、~JI コマンドとほぼ同様に機能します。どちらのコマンドも、Zebra BASIC Interpreter を初期化するためにプリンタに送信されます。

対話モードでは、^JI は通信ポート（シリアル、パラレル、または Ethernet）の 1 つを通して送信され、ZBI コマンドを受信するようにプリンタを初期化することができます。このコマンドは、ZTools などの Zebra ソフトウェア・ユーティリティの 1 つから、または Hyper terminal などの標準 PC プログラムから送信できます。

コマンドを受信すると、プリンタは ZBI ヘッダーをプログラム・バージョン番号とともにコンソールに送り返して応答します。これは、インタープリタがアクティブであることを示します。

フォーマット ^JId:o.x,b,c,d

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = 初期化後に実行するプログラムの場所	*有効値：* R:、E:、B:、および A: *デフォルト値：*場所を指定する必要があります
d = 初期化後に実行するプログラムの名前	*有効値：*任意の有効なプログラム名 *デフォルト値：*名前を指定する必要があります
x = 初期化後に実行するプログラムの拡張子	*固定値：* .BAS
b = コンソール・コントロール	*有効値：* Y（コンソールがオン）または N （コンソールがオフ） *デフォルト値：* Y

パラメータ	詳細
c = エコー・コントロール	*有効値 :* Y（エコーがオン）または N （エコーがオフ） *デフォルト値 :* Y
d = ZBI のメモリ割り当て	*有効値 :* 20K2 ～ 1024K *デフォルト値 :* 50K

コメント プリンタがオンの場合、ZPL II コマンドとラベル・フォーマットを受信できます。ただし、プリンタが ZBI コマンドとプログラムを認識するためには、^JI または ~JI を使って初期化されている必要があります。

プリンタ内で一度にアクティブにできる ZBI インタープリタは 1 つだけです。インタープリタの実行中に、2 番目の ^JI または ~JI コマンドを受信した場合は、コマンドは無視されます。

インタープリタは次のコマンドのどちらかを入力することにより停止されます。

ZBI プロンプトで ZPL

アクティブな ZPL ポートで ~JQ

~JI

ZBI（Zebra BASIC Interpreter）の開始

説明 ~JI は、^JI コマンドとほぼ同様に機能します。どちらのコマンドも、Zebra BASIC Interpreter を初期化するためにプリンタに送信されます。

対話モードでは、~JI は通信ポート（シリアル、パラレル、または Ethernet）の 1 つを通して送信され、ZBI コマンドを受信するようにプリンタを初期化することができます。このコマンドは、ZTools などの Zebra ソフトウェア・ユーティリティの 1 つから、または Hypercritical などの標準 PC プログラムから送信できます。

コマンドを受信すると、プリンタは ZBI ヘッダをプログラム・バージョン番号とともにコンソールに送り返して応答します。これは、インタープリタがアクティブであることを示します。

フォーマット ~JI

コメント コマンドの受信中、プリンタは受け取った文字を送信元にエコー・バックします。この機能は ZBI ECHO コマンドによってオン / オフを切り替えることができます。

プリンタがオンの場合、ZPL II コマンドとラベル・フォーマットを受信できます。ただし、ZBI コマンドとフォーマットを認識するためには、プリンタは ^JI または ~JI を使って初期化されている必要があります。

プリンタ内で一度にアクティブにできる ZBI インタープリタは 1 つだけです。インタープリタの実行中に、2 番目の ~JI または ^JI コマンドを受信した場合は、コマンドは無視されます。

インタープリタは次のコマンドのどちらかを入力することにより停止されます。

ZBI プロンプトで ZPL

アクティブな ZPL ポートで ~JQ

^JJ

補助ポートの設定

説明 ^JJ コマンドを使用すると、オンライン検証器またはアプリケータ・デバイスを制御できます。

フォーマット ^JJa,b,c,d,e,f

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 補助ポートの操作モード	*有効値：* 0 = オフ 1 = エラー発生時に再発行 -- プリンタは検証エラーのあるラベルで停止します。PAUSE ボタンを押すと、ラベルが再発行されます（^JZ が再発行に設定されている場合）。バー・コードがラベルの上端近くにある場合、ラベルはバー・コードを検証できる位置までフィードされ、その後、次のラベルの印刷と検証が可能な位置までバックフィードされます。 2 = 最大スループット -- 検証エラーが検出されるとプリンタは停止します。検証器が前のラベルを確認中に、プリンタは次のラベルの印刷を開始します。このモードは最大スループットを提供しますが、プリンタは検証エラーのあるラベルで即座に停止できません。 *デフォルト値：* 0
b = アプリケーション・モード	*有効値：* 0 = オフ 1 = 発行終了信号は通常はハイに設定され、プリンタがラベルを前に送り出す間のみローに設定されます。 2 = 発行終了信号は通常はローに設定され、プリンタがラベルを前に送り出す間のみハイに設定されます。 3 = 発行終了信号は通常はハイに設定され、ラベルが印刷され配置された場合 20 ms 間ローに設定されます。 4 = 発行終了信号は通常はローに設定され、ラベルが印刷され配置された場合 20 ms 間ハイに設定されます。 *デフォルト値：*0

パラメータ	詳細
c = アプリケーション・モード発行開始信号	*有効値：* p = パルス・モード - 発行開始信号は次のラベルに対してアサートされる前にデアサートする必要があります。 l = レベル・モード - 発行開始信号は、次のラベルを印刷するためにデアサートする必要はありません。発行開始信号がローで、ラベルがフォーマットされている限り、ラベルは印刷されます。 *デフォルト値：*0
d = アプリケーション・ラベル・エラー・モード	*有効値：* e = エラー・モード -- プリンタは、アプリケーション・ポートで *サービス要* 信号 (svce_req - pin 10) をアサートし、ポーズ・モードに入り、LCD にエラー・メッセージを表示します。 f = フィード・モード -- 予期した場所にプリンタを用紙とを同期させるためのウェブが見つからない場合、空白のラベルが印刷されます。 *デフォルト値：*f
e = 再発行モード	*有効値：* e = 有効 -- プリンタは再発行信号を無視します。 d = 無効 -- 信号がアサートされた後、最後のラベルが再発行されます。ラベルをキャンセルすると、再発行されるラベルもキャンセルされます。最後に発行されたラベルは再発行までリリースされないため、このモードはより多くのメモリを消費します。 *デフォルト値：*d
f = リボン少量モード	*有効値：* e = *有効* - リボンが少量になるとプリンタの警告が発行されます。 d = *無効* - リボンが少量になってもプリンタの警告は発行されません。 *デフォルト値：*e

~JL

ラベル長の設定

説明 ~JL コマンドは、ラベル長の設定に使用します。ラベルのサイズにより、プリンタは 1 つ以上の空白のラベルをフィードします。

フォーマット ~JL

^JM

ミリメートルあたりのドット数の設定

説明 ^JM コマンドは、印字密度を下げます。つまり、24 ドット /mm は 12 に、12 ドット /mm は 6 に、8 ドット /mm は 4 に、そして 6 ドット /mm は 3 になります。さらに、^JM は、ラベル上のフィールド基点（^FO）の配置にも影響します（例を参照）。

^JM コマンドをプリンタに送信すると、ラベルのフォーマット・サイズが倍になります。印刷ヘッドにより、Zebra プリンタの標準のミリメートルあたりのドット数は、12 ドット /mm（304 ドット / インチ）、8 ドット /mm（203 ドット / インチ）、または 6 ドット /mm（153 ドット / インチ）です。

このコマンドは、フォーマット内の最初の ^FS コマンドの前に入力する必要があります。^JM の影響は永続的なものです。

フォーマット ^JMn

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
n = ミリメートルあたりのドット数を設定する	*有効値：* A = 24 ドット /mm、12 ドット /mm、8 ドット /mm、6 ドット /mm のいずれか B = 12 ドット /mm、6 ドット /mm、4 ドット /mm、3 ドット /mm のいずれか *デフォルト値：*A

例 • 次の例は、ミリメートルあたりのドット数を変更した場合の影響を示しています。

ZPL II CODE	GENERATED LABEL
^XA ^JMA^FS ^FO100,100 ^B2N,50,Y,N,N ^FD1234567890^FS ^XZ	
^XA ^JMB^FS ^FO100,100 ^B2N,50,Y,N,N ^FD1234567890^FS ^XZ	

コメント ^JMB を使用すると、 UPS Maxicode バー・コードは規格外になります。

~JN

ヘッド・テスト（致命的）

説明 ~JN コマンドは、ヘッド・テスト・オプションをオンにします。~JN をアクティブにすると、ヘッド・テスト・エラーが発生した場合にプリンタは停止します。

エラーがいったん発生すると、ヘッド・テストをオフにする（~JO）まで、または電力が循環されるまでプリンタはエラー・モードのままになります。

フォーマット ~JN

コメント 通信バッファが一杯の場合、プリンタはデータを受信できません。その場合、~JO コマンドはプリンタで受信されません。

~JO

ヘッド・テスト（非致命的）

説明 ~JO コマンドは、ヘッド・テスト・オプションをオフにします。~JO はデフォルトの印刷ヘッド・テスト状態で、印刷ヘッド・エレメントのステータス・チェックのエラーを無視します。プリンタが ~JN（ヘッド・テスト（致命的））コマンドを受信すると、この状態は変更されます。~JO がアクティブな場合、印字ヘッド・テストではエラーは生成されません。

フォーマット ~JO

~JP

フォーマットのポーズとキャンセル

説明 ~JP コマンドは、現在処理中のフォーマットをクリアし、プリンタをポーズ・モードにします。

このコマンドにより印刷する次のフォーマット、またはバッファ内で最も古いフォーマットがクリアされます。後続の各 ~JP コマンドは、バッファに入れられたフォーマットを、バッファが空になるまで次々とクリアします。バッファが空でデータが送信されていない場合、DATA インジケータはオフになります。

~JP コマンドを発行することは、プリンタで CANCEL を使用するのと同じですが、プリンタの場合、最初にポーズ・モードにする必要はありません。

フォーマット ~JP

~JQ

Zebra BASIC Interpreter の終了

説明 ~JQ コマンドは、Zebra BASIC Interpreter がアクティブな場合に使用します。プリンタに ~JQ を送信すると ZBI セッションが終了されます。

フォーマット ~JQ

コメント コマンド・プロンプトで ZPL を入力しても ZBI セッションが終了されます。

~JR

電源オン・リセット

説明 ~JR コマンドは、プリンタのすべての内部ソフトウェアをリセットし、電源オン・セルフ・テスト（POST）を実行し、バッファと DRAM をクリアし、通信パラメータとデフォルト値をリセットします。~JR コマンドを発行すると、手動の電源オン・リセットと同じ機能が実行されます。

フォーマット ~JR

^JS

バックフィード手順の変更

説明 ^JS コマンドは、バックフィード手順の制御に使用します。このコマンドは、カッター付きのプリンタでもカッターなしのプリンタでも使用できます。

主なアプリケーションは次のとおりです。

- 連続用紙の裁断時の*静止位置* のプログラミングを可能にします。
- プリンタを印刷 / 適用アプリケーション設定で使用した場合、剥離の直後にバックフィードを行う。

このコマンドが有効なのは、プリンタの電源を切るまで、新しい ~JS コマンドが送信されるまで、またはフロント・パネルで設定が変更されるまでに限られます。~JS コマンドが発行されると、現在のフロント・パネルのバックフィード手順は無視されます。

バックフィードを消去する最も一般的な方法は、巻き取りモードで操作することです。巻き取りモードでは、バックフィードはまったく行われません。ラベルが印刷されたら、次のラベルのリーディング・エッジが印字ラインに配置されます。これにより、バックフィードの必要がなくなり、ラベルのリーディング・エッジまたは一番下に印字不可能領域が生成されることもありません。また、巻き取りモードではラベルは印刷ヘッドの下からフィードされないため、プリンタからラベルを取り出すことができません。

バックフィードをオフにして別のモードで操作すると、ラベルを取り外すことができるため、バックフィード手順に要する時間を節約できます。

フォーマット ~JSb

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
b = 印刷に関連したバックフィード手順	*有効値：* A = 印刷と切り取りの後は、100 パーセントのバックフィード B = 印刷と切り取りの後は、0 パーセントのバックフィード、および次のラベルの印刷前は 100 パーセントのバックフィード N = 標準 -- ラベル印刷後は 90 パーセントのバックフィード O = オフ -- バックフィードを完全にオフにする 10 ～ 90 = パーセント値 入力する値は 10 の倍数である必要があります。10 で割り切れない値は、最も近い有効値に四捨五入されます。たとえば、~JS55 は 50 パーセントのバックフィードとして受け入れられます。 *デフォルト値：*N

コメント 特定の値を使用する場合、次のラベルが印刷される前に、入力された値と 100% の差が計算されます。たとえば 40 の値は、ラベルが切り取られたか取り除かれた後に、40% のバックフィードが行われることを意味します。残りの 60% は、次のラベルが印刷される前に行われます。

このコマンドの値は、プリンタ設定ラベルのバックフィード・パラメータにも反映されます。

~JSN の場合 -- バックフィード・パラメータは DEFAULT としてリストされます。

~JSA または 100 の場合 -- バックフィード・パラメータは AFTER としてリストされます。

~JSB または 0 の場合 -- バックフィード・パラメータは BEFORE としてリストされます。

~JS10 ～ 90 の場合 -- バックフィード・パラメータは入力した値としてリストされます。

~JS

バックフィード手順の変更

説明 ~JS コマンドは、バックフィード手順の制御に使用します。このコマンドは、カッター付きのプリンタでもカッターなしのプリンタでも使用できます。

主なアプリケーションは次のとおりです。

- 連続用紙の裁断時の *静止位置*のプログラミングを可能にする。
- プリンタを印刷 / 適用アプリケーション設定で使用した場合、剥離の直後にバックフィードを行う。

このコマンドが有効なのは、プリンタの電源を切るまで、新しい ~JS コマンドが送信されるまで、またはフロント・パネルで設定が変更されるまでだけです。~JS コマンドが発行されると、現在のフロント・パネルのバックフィード手順は無視されます。

バックフィードを消去する最も一般的な方法は、巻き取りモードで操作することです。巻き取りモードでは、バックフィードはまったく行われません。ラベルが印刷されたら、次のラベルのリーディング・エッジが印字ラインに配置されます。これにより、バックフィードの必要がなくなり、ラベルのリーディング・エッジまたは一番下に印字不可能領域が生成されることもありません。また、巻き取りモードではラベルは印刷ヘッドの下からフィードされないため、プリンタからラベルを取り出すことができません。

バックフィードをオフにして別のモードで操作すると、ラベルを取り外すことができるため、バックフィード手順に要する時間を節約できます。

フォーマット ~JSb

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
b = 印刷に関連したバックフィード手順	*有効値：* A = 印刷と切り取りの後、100 パーセントのバックフィード B = 印刷と切り取りの後は、0 パーセントのバックフィード、および次のラベルの印刷前は 100 パーセントのバックフィード N = 標準 -- ラベル印刷後は 90 パーセントのバックフィード O = オフ -- バックフィードを完全にオフにする 10 ～ 90 = パーセント値 入力する値は 10 の倍数である必要があります。10 で割り切れない値は、最も近い有効値に四捨五入されます。たとえば、~JS55 は 50 パーセントのバックフィードとして受け入れられます。 *デフォルト値：*N

コメント 特定の値を使用する場合、次のラベルが印刷される前に、入力された値と 100% の差が計算されます。たとえば 40 の値は、ラベルが切り取られたか取り除かれた後に、40% のバックフィードが行われることを意味します。残りの 60% は、次のラベルが印刷される前に行われます。

このコマンドの値は、プリンタ設定ラベルのバックフィード・パラメータにも反映されます。

~JSN の場合 -- バックフィード・パラメータは DEFAULT としてリストされます。

~JSA または 100 の場合 -- バックフィード・パラメータは AFTER としてリストされます。

~JSB または 0 の場合 -- バックフィード・パラメータは BEFORE としてリストされます。

~JS10 ～ 90 の場合 -- バックフィード・パラメータは入力した値としてリストされます。

^JT

ヘッド・テスト間隔

説明 ^JT コマンドを使用すると、印刷ヘッド・テスト間隔を 100 ラベルごとから希望の間隔に変更できます。^JT コマンドでは、プリンタはラベルの印刷後にテストを実行できます。パラメータが定義されると、プリンタは一定数のラベルの印刷後にテストを実行します。

プリンタのデフォルトのヘッド・テスト状態はオフです。印刷ヘッドテスト実行のパラメータはユーザーによって決定されます。

フォーマット ^JT####,a,b,c

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
#### ＝ヘッド・テスト間に印刷される 4 桁のラベル数	*有効値：* 0000 ～ 9999 9999 より大きい値は無視されます。 *デフォルト値：*0000（オフ）
a ＝テストするエレメントの範囲を手動で選択する	*有効値：*Y（はい）または N （いいえ） *パワーアップ時の初期値：*N
b ＝パラメータ a が Y の場合に確認する最初のエレメント	*有効値：*0 ～ 9999 *パワーアップ時の初期値：*0
c ＝パラメータ a が Y の場合に確認する最後のエレメント	*有効値：*0 ～ 9999 *パワーアップ時の初期値：*9999

コメント ^JT コマンドは、一定範囲の印刷エレメントのテストをサポートします。プリンタは、前回のテスト後に使用されたエレメントを追跡することにより、テスト範囲を自動的に選択します。

さらに ^JT は、自動モードをオンにしてヘッド・テストの最初と最後のエレメントを指定します。これにより、ラベルの特定領域を選択することも、全印字幅を選択することも可能になります。

最後に選択したエレメントが、選択した印字幅よりも大きい場合は、選択した印字幅でテストが停止します。

ヘッド・テスト・コマンドを受信すると、値が 0（ゼロ）に設定されていない限り、次のラベルでヘッド・テストが実行されます。

~JU

設定の更新

説明 ^JU コマンドは、プリンタのアクティブ設定を指定します。

フォーマット ^JUa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = アクティブ設定	*有効値：* F = 工場出荷時の値を再読み込みする ^JUS で保存しないと、これらの値は電源を切った時点で失われます。 R = 最後に保存した値を呼び出す S = 現在の設定を保存する これらの値は、電源を入れたときに使用されます。 *デフォルト値：*値を指定する必要があります

^JW

リボン・テンションの設定

説明 ^JW は、送信先のプリンタのリボン・テンションを設定します。

フォーマット ^JWt

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
t = テンション	*有効値：* L = 低 M = 中 H = 高 *デフォルト値：*値を指定する必要があります

コメント ^JW は、*PAX* シリーズのプリンタの場合にのみ使用されます。

~JX

現在部分的に入力されているフォーマットのキャンセル

説明 ~JX コマンドは、現在プリンタに送信中のフォーマットをキャンセルします。現在印刷中のフォーマットにも、その後に送信されるフォーマットにも影響しません。

フォーマット ~JX

^JZ

エラー後再発行

説明 ^JZ コマンドは、リボン切れ、用紙切れ、ヘッド・オープンなどのエラーのために部分的に印刷されたラベルを再発行します。エラー状態が訂正されるとラベルはすぐに再発行されます。

このコマンドは、別の ^JZ コマンドがプリンタに送信されるか、プリンタの電源を切るまでアクティブです。

フォーマット ^JZa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = エラー後に再発行する	*有効値：*Y（はい）または N （いいえ） *パワーアップ時の初期値：*Y

コメント ^JZ はプリンタのエラー・モードを設定します。^JZ を変更した場合、変更後に印刷されたラベルのみに影響があります。

パラメータが欠落しているか不正な場合は、このコマンドは無視されます。

~KB

バッテリのキル（バッテリ放電モード）

説明 ポータブル・プリンタでの充電式バッテリのパフォーマンスを維持するために、バッテリは完全に放電してから定期的に再充電する必要があります。~KB コマンドは、プリンタをバッテリ放電モードにします。これにより、実際に印刷しなくてもバッテリの電力を抜き取ることができます。

フォーマット ~KB

コメント プリンタが放電モードの間は、緑色の電源 LED が 3 回ずつ連続でフラッシュします。

放電モードは、プリンタに印刷フォーマットを送信するか、フロント・パネル・キーのいずれかを押すことによって終了できます。

バッテリ充電器がプリンタに接続している場合、放電処理が完了するとバッテリは自動的に充電されます。

^KD

日付と時刻のフォーマットの選択（リアルタイム・クロック用）

説明 ^KD コマンドは、リアルタイム・クロックの日付と時刻の情報によって設定ラベルとして表示されるフォーマットを選択します。これは、*「ﾌﾟﾘﾝﾀ_ﾀｲｷﾁｭｳ」* LCD フロント・パネル・ディスプレイにも日付と時刻の設定中に表示されます。

フォーマット ^KDa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 日付と時刻のフォーマットの値	*有効値：* 0 = 標準、ファームウェアの*バージョン番号*を表示する 1 = MM/DD/YY（24 時間クロック） 2 = MM/DD/YY（12 時間クロック） 3 = DD/MM/YY（24 時間クロック） 4 = DD/MM/YY（12 時間クロック） *デフォルト値：*0

コメント リアルタイム・クロック・ハードウェアがない場合は、表示モードは 0（バージョン番号）に設定されます。

表示モードを 0（バージョン番号）に設定して、リアルタイム・クロックがある場合は、設定ラベルの日付と時刻のフォーマットは、フォーマット 1 で表示されます。

表示モードを 0（バージョン番号）に設定して、リアルタイム・クロックがある場合は、フロント・パネルの日付と時刻のフォーマットは、フォーマット 1 で表示されます。

^KL

言語の定義

説明 ^KL コマンドは、フロント・パネルに表示される言語を選択します。

フォーマット ^KLa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 言語	*有効値：* 1 = 英語 2 = スペイン語 3 = フランス語 4 = ドイツ語 5 = イタリア語 6 = ノルウェー語 7 = ポルトガル語 8 = スウェーデン語 9 = デンマーク語 10 = スペイン語 2 11 = オランダ語 12 = フィンランド語 13 = 日本語 *デフォルト値：*1

^KN

プリンタ名の定義

説明 プリンタのネットワーク名と説明は ^KN コマンドを使用して設定できます。^KN は、Zebra プリンタをユーザーが識別しやすくするために設計されています。管理者が指定した名前は、プリンタによって生成された設定ラベルと Web ページにリストされています。

フォーマット ^KNa,b

次の表では、このフォーマットのパラメータを示します。

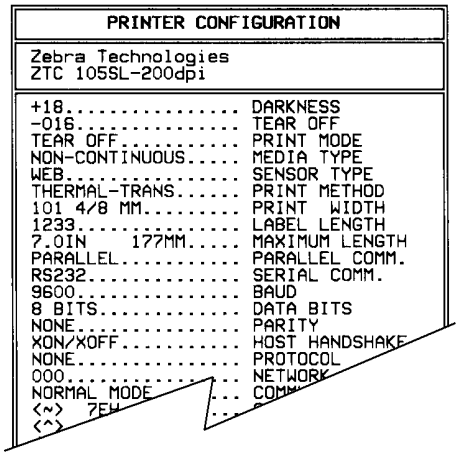
パラメータ	詳細
a ＝プリンタ名	*有効値：*最高 16 文字までの英数字 *デフォルト値：*値を入力しなかった場合は、このパラメータは無視されます。 16 文字以上を入力すると、最初の 16 文字のみが使用されます。
b ＝プリンタの説明	*有効値：*最高 35 文字までの英数字 *デフォルト値：*値を入力しなかった場合は、このパラメータは無視されます。 35 文字以上を入力すると、最初の 35 文字のみが使用されます。

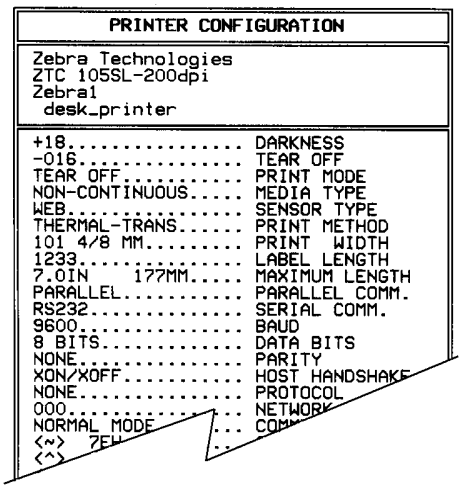
例 • 次は、プリンタのネットワーク名と説明を変更する例です。

以下は、このコマンドの使用前と使用後の設定です。

```
^XA
^KNZebra1,desk_printer
^XZ
```

このコマンドの使用前このコマンドの使用後





^KP

パスワードの定義

説明 ^KP コマンドは、フロント・パネルのスイッチと LCD セットアップ・モードにアクセスするために入力する必要のあるパスワードの定義に使用します。

フォーマット ^KP####

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
#### = 4 桁の必須のパスワード	*有効値：* = 任意の 4 桁の数値シーケンス *デフォルト値：* 1234

例 • 次は、新しいフロント・パネル・パスワードを設定する例です。

```
^XA
^KP5678
^XZ
```

コメント パスワードを忘れた場合は、プリンタをデフォルトのセットアップ・モードに戻すとデフォルトのパスワード *1234* が再び有効になります。しかし、これはプリンタの設定値もデフォルト値に戻すため、注意が必要です。

ZPL を使用してプリンタをデフォルトの工場出荷時設定に戻すには、以下を送信します。

```
^XA
^JUF
^XZ
```

コントロール・パネル・キーを使用してプリンタをデフォルトの工場出荷時設定に戻すには、プリンタのユーザー・ガイドで手順を参照してください。

^LH

ラベルのホーム

説明 ^LH コマンドはラベルのホーム・ポジションを設定します。

ラベルのデフォルト・ホーム・ポジションは左上コーナー（x 軸と y 軸に沿ってそれぞれ 0 の位置）です。これがラベルの軸基準点です。この点の下および右側の領域が印刷可能な領域です。^LH コマンドはこの基準点を変更します。たとえば、事前印刷されたラベルを扱う際、このコマンドを使用して基準点を事前印刷領域の下に移動します。

このコマンドは、その後に続くフィールドのみに影響します。^LH は、ラベル・フォーマット内の最初のコマンドの 1 つとして使用することをお勧めします。

フォーマット ^LHx,y

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
x = x 軸の位置（ドット数）	*有効値：*0 ～ 32000 *パワーアップ時の初期値:*0 または最後に永久保存された値
y = y 軸の位置（ドット数）	*有効値：*0 ～ 32000 *パワーアップ時の初期値:*0 または最後に永久保存された値

プリンタで使用している印刷ヘッドにより、x と y の値の決定には、以下の 1 つを使用してください。

6 ドット = 1 mm、152 ドット = 1 インチ

8 ドット = 1 mm、203 ドット = 1 インチ

11.8 ドット = 1 mm、300 ドット = 1 インチ

24 ドット = 1 mm、608 ドット = 1 インチ

コメント 既存のプリンタとの互換性を持たせるため、このコマンドは最初の ^FS（フィールド・セパレータ）コマンドの前に挿入する必要があります。いったん ^LH コマンドを発行すると、プリンタの電源を切るか、プリンタに新しい ^LH コマンドを送信するまでその設定が維持されます。

^LL

ラベル長

説明 ^LL コマンドは、ラベルの長さを定義します。このコマンドは、連続用紙（切れ目、スペース、切り込み、スロット、穴などで別々のラベルに分割されていない用紙）を使用する場合に必要です。

現在のラベルに適用するため、および既存のプリンタとの互換性を持たせるため、^LL コマンドは最初の ^FS（フィールド・セパレータ）コマンドの前に挿入する必要があります。いったん ^LL コマンドを発行すると、プリンタの電源を切るか、新しい ^LL コマンドを送信するまでその設定が維持されます。

フォーマット ^LLy

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
y = y 軸の位置（ドット数）	*有効値：*1 ～ 32000（最大ラベル・サイズを超えない） プリンタはこのパラメータとしてどの値でも受け入れますが、インストールされているメモリの量によってラベルの最大長が決定します。 *デフォルト値：*通常、LCD（適用する場合）によって設定されるか、プリンタの最大ラベル長に設定されます。

コメント y の値を決定するためには次の数式を使用できます。

6 ドット /mm の印刷ヘッドの場合：	ラベル長（インチ）x 153.4（ドット / インチ）= y
8 ドット /mm の印刷ヘッドの場合：	ラベル長（インチ）x 203.2（ドット / インチ）= y
12 ドット /mm の印刷ヘッドの場合：	ラベル長（インチ）x 304.8（ドット / インチ）= y
24 ドット /mm の印刷ヘッドの場合：	ラベル長（インチ）x 609.6（ドット / インチ）= y

y の値はメモリ・サイズによって異なります。y として入力した値が有効な限界値を超える場合、ラベルの一番下が切り取られます。また、ラベルは上から下にシフト・ダウンします。

同じラベル・フォーマットで複数の ^LL コマンドが発行された場合、最後の ^LL コマンドが 最初の ^FS 前に挿入されていない限り、それは次のラベルに影響します。

^LR

ラベル反転印刷

説明 ^LR コマンドはラベル・フォーマット内のすべてのフィールドの印刷を反転します。この コマンドを使用すると、フィールドを黒地に白、または白地に黒に表示できます。

^LR を使用することは、現在および後続のすべてのフィールドに ^FR を挿入することと同じです。

フォーマット ^LRa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = すべてのフィールドを反転印刷	*有効値：* Y（はい）または N （いいえ） *パワーアップ時の初期値:* N または最後に永久保存された値

例 • 次は、黒地に白、および白地に黒を印刷する例です。

ZPL II CODE	GENERATED LABEL
^XA^LRY ^FO100,50 ^GB195,203,195^FS ^FO180,110^CFG ^FDLABEL^FS ^FO130,170 ^FDREVERSE^FS ^XZ	

コメント ^LR 設定は ^LRN によってオフにするか、プリンタの電源を切るまでアクティブです。

注記 • ^GB は ^LR と一緒に使用する必要があります。

このコマンドの後に続くフィールドのみに影響があります。

^LS

ラベルのシフト

説明 ^LS コマンドは、最大ラベル幅より少なく設定されている Z-130 プリンタ・フォーマットとの互換性を提供します。このコマンドは、Z-130 または Z-220 プリンタで使用されるのと同じコマンドを Zebra プリンタで使用できるように、すべてのフィールド位置を左にシフトします。

^LS コマンドの値を決定するには次の数式を使用します。

Z-130 and Z-220 values for ^LHx + ^FOx

（ラベルのエッジからの距離）= ^LSa のプリンタ値

印字位置が 0 よりも少ない場合、^LS を 0 に設定してください。

フォーマット ^LSa

重要 • ^LS コマンドを保存できるかどうかは、ファームウェアのバージョンによります。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 左へシフトする値（ドット数）	*有効値：*-9999 ～ 9999 *パワーアップ時の初期値：*0

コメント 正の値を入力する場合、+ 符号を使用する必要はありません。マイナス符号（-）がない限り、数値は正数とみなされます。

既存の Zebra プリンタとの互換性を持たせるため、このコマンドは最初の ^FS（フィールド・セパレータ）コマンドの前に挿入する必要があります。いったん ^LS コマンドを発行すると、プリンタの電源を切るか、プリンタに新しい ^LS コマンドを送信するまでその設定が維持されます。

^LT

Y 印字基点

説明 ^LT コマンドは、ラベルの上端に関して、ラベル・フォーマット全体を現在の位置から最大 120 ドット上または下に移動します。負の値はフォーマットをラベルの上端の方向に、正の値は上端と反対の方向に移動します。

このコマンドを使用すると、完成したラベルの位置を、既存のパラメータを変更せずに微調整できます。

フォーマット ^LTx

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
x = Y 印字基点（ドット行数）	*有効値：* -120 ～ 120 *デフォルト値：* 値を指定しないとコマンドは無視されます。

コメント x の有効値の範囲はプリンタのプラットフォームによっては上記よりも小さい場合があります。

^LT コマンドによって用紙の静止位置は変更されません。

^MC

マップ・クリア

説明 通常の操作では、フォーマットの印刷後にビットマップはクリアされます。^MC コマンドは、現在のビットマップを維持する場合に使用します。これは、^MCY によってクリアされるまで、現在と後続のラベルに適用します。

フォーマット ^MCa

重要 • ラベルテンプレートを生成するには、^MC を ^FV と一緒に使用する必要があります。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = マップ・クリア	*有効値：* Y（ビットマップをクリアする）または N（ビットマップをクリアしない） *パワーアップ時の初期値：* Y

コメント ^MC コマンドは、フォーマットの後、現在のラベルのイメージを維持します。それは、次に印刷されるラベルの背景に表示されます。

^MD

用紙の濃度

説明 ^MD コマンドは、現在の濃度設定に関連して濃度を調整します。

フォーマット ^MDa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 用紙濃度レベル	*有効値：*-30 ～ 30（現在の値による） *パワーアップ時の初期値：*0 値を入力しなかった場合は、このコマンドは無視されます。

例 以下は、プリンタを異なる濃度レベルに設定する例です。

- 現在の値（設定ラベル上の値）が 16 の場合、^MD-9 コマンドを入力すると値が 7 に減ります。
- 現在の値（設定ラベル上の値）が 1 の場合、^MD15 コマンドを入力すると値が 16 に増えます。
- 現在の値（設定ラベル上の値）が 25 の場合、^MD10 コマンドを入力しても値は許容される最大値の 30 にしかなりません。

各 ^MD コマンドは、設定ラベルに印刷された現在の値に関して別々に処理されます。

重要 • XiIIIPlus の濃度設定範囲は 1 ～ 30 で、増分単位は .1 です。
ファームウェアは、ZPL の 2 つの濃度コマンド（^MD, ~SD）がその範囲の設定を受け入れるようにセットアップされています。

例 • 以下は XiIIIPlus の濃度設定例です。

```
^MD8.3

~SD8.3
```

例 • たとえば、2 つの ^MD コマンドを受信した場合を考えます。

現在の値は 15 だとします。現在の値を 9 に変更する ^MD-6 コマンドが受信されます。別のコマンド ^MD2 が受信されます。現在の値は 17 に変更されます。

2 つの ^MD コマンドは、現在の値 15 に関して別々に処理されます。

コメント ~SD コマンド値は ^MD コマンドに追加されます（該当する場合）。

^MF

用紙のフィード

説明 ^MF コマンドは、パワーアップ時およびエラー解消後ヘッドを閉めるときの用紙の処理を決定します。

フォーマット ^MFp,h

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
p = パワーアップ時にフィード・アクション	*有効値：* F = センサーの後の最初のウェブにフィードする C = （*~JC ページ 229* の定義を参照） L = （*~JL ページ 241* の定義を参照） N = 用紙フィードなし *デフォルト値：*プラットフォームによる
p = 印刷ヘッドを閉じた後にフィード・アクション	*有効値：* F = センサーの後の最初のウェブにフィードする C = （*~JC ページ 229* の定義を参照） L = （*~JL ページ 241* の定義を参照） N = 用紙フィードなし *デフォルト値：*プラットフォームによる

コメント N の設定を選択する場合、プリンタは用紙、および印刷ヘッドに関連したその位置は電源を切った前、または印刷ヘッドを開く前のものと同じであると推定することに留意することが重要です。変更を保存するには ^JU コマンドを使用してください。

^ML

最大ラベル長

説明 ^ML コマンドを使用すると、最大ラベル長を調節できます。

フォーマット ^MLa,b,c,d

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 最大ラベル長（ドット行数）	*有効値：*0 ～ラベルの最大長 *デフォルト値：*最後に永久保存された値
b = 最大論理用紙切れカウンタ	*有効値：*実際のラベル長より大きい必要があります。そうでない場合は、各ラベルの後にプリンタで「ﾖｳｼｷﾞﾚ」エラーが表示されます。 *デフォルト値：*ラベル長の 2 倍の長さより 1 インチ長く設定します。
c = 最大物理用紙切れカウンタ	*有効値：*実際の切れ込みまたは穴より大きい必要があります。そうでない場合は、各ラベルの後にプリンタで「ﾖｳｼｷﾞﾚ」エラーが表示されます。 *デフォルト値：*半インチに設定されます。
d = 最大リボン切れカウンタ	*有効値：*リボン・センサーがエラー状態を起こさずにリボンを誤読できる値。 *デフォルト値：*半ミリメートルに設定されます。 **重要 •** プリンタは長さが半ミリメートル以下のリボンの表示は無視します。

コメント キャリブレーションが正しく機能するためには、実際のラベル長以上の最大ラベル長を設定する必要があります。

^MM

印字モード

説明 ^MM コマンドは、ラベル、またはラベルのグループの印刷後のプリンタの動作を決定します。次の箇条書きのリストは、操作のさまざまなモードを示しています。

- 切り取り -- 印刷後、ラベルはウェブが切り取りバーの上になるように送られます。ラベルは、台紙をつけたまま手動で切り取ることができます。
- 剥離 -- 印刷後、ラベルは前に送られラベル検出センサーがアクティブになります。印刷は、プリンタからラベルが手動で除去されるまで、停止します。

  自動巻き取り剥離 - オプションの巻き取りスピンドルを使って、台紙が自動的に巻き取られます。

  簡易巻き取り剥離 - 台紙がプリンタのフロントにフィード・ダウンされ、手動で除り除かれます。

  事前剥離 - 各ラベルを手動で取り除いた後、プリンタは次のラベルを前にフィードし、そのラベルの小さな部分を台紙から事前剥離します。その後、プリンタはそのラベルをバックフィードし、印刷します。事前剥離機能は、一部の用紙タイプの正しい剥離操作を補助します。
- 巻き取り -- ラベルと台紙は外付け巻き取りデバイスで巻き取られます。次のラベルは、印刷ヘッドの下に配置されます（バックフィード動作なし）。
- アプリケータ -- アプリケーション・デバイスと一緒に使用した場合、アプリケータで取り除いてアイテムに貼ることが可能な場所までラベルが前に送られます。
- カッター -- 印刷後、用紙は前に送られ、事前に決定した長さに切り分けられます。

フォーマット `^MMa,b`

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 希望のモード	*有効値：* T = 切り取り P = 剥離（*S*-300 では利用不可） R = 巻き取り A = アプリケータ（プリンタ・モデルによる） C = カッター *デフォルト値：*T パラメータ a として使用できる値は、使用するプリンタと、それがこのオプションをサポートしているかどうかによって異なります。
b = 事前剥離選択	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*Y パラメータが欠落しているか無効な場合は、このコマンドは無視されます。コマンドの現在の値は変りません。

コメント 予期しない結果を招かないために、使用する印刷モードに対して適切な値を選択してください。

^MN

用紙の管理

説明 ^MN コマンドは、管理の目的で、使用している用紙のタイプ（連続または短票）をプリンタに伝えます。次の箇条書きリストは、このコマンドに関連する用紙のタイプを示しています。

- 連続メディア - このメディアには、ラベルを区切る物理特性（ウェブ、切れ込み、目打ち、マークなど）がありません。ラベル長は ^LL コマンドによって決定されます。
- 単票メディア - このメディアには、ラベルを区切る何らかの物理特性（ウェブ、切れ込み、目打ち、マークなど）があります。

フォーマット ^MNa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 使用中の用紙	*有効値：* N = 連続用紙 *Y = 単票用紙ウェブ検知 *W = 単票用紙ウェブ検知 M = 単票用紙マーク検知 *デフォルト値：*値を指定しないとコマンドは無視されます。

* 同じ結果を提供します。

^MP

モード保護

説明 ^MP コマンドは、フロント・パネルのさまざまなモード機能を無効にするために使用します。いったん無効にすると、特定のモード機能の設定は変更できなくなり、その機能と関連する LED は点灯しません。

このコマンドにはパラメータが 1 つしかないため、個別の ^MP コマンドを使用して各モードを無効にする必要があります。

フォーマット ^MPa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a ＝保護するモード	*有効値：* D＝濃度モードの無効化 P＝ポジション・モードの無効化 C＝キャリブレーション・モードの無効化 E ＝すべてのモードを有効化 S＝すべてのモード保存の無効化（モードは調整できますが、値は保存されません）。 W＝ポーズの無効化 F＝フィードの無効化 X＝キャンセルの無効化 M＝メニュー変更の無効化 *デフォルト値：*値を指定しないとコマンドは無視されます。

例 • 次の例は、D と C のモードを無効にします。

```
^XA
^MPD
^MPC
^XZ
```

^MT

用紙タイプ

説明 ^MT コマンドは、プリンタで使用される用紙のタイプを選択します。次の 2 つのタイプから選択できます。

- 熱転写用紙 - この用紙は、高炭素の黒または色付きリボンを使用します。リボンのインクは用紙に付着します。
- ダイレクト・サーマル用紙 - この用紙は感熱性で、リボンを必要としません。

フォーマット ^MTa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 使用する用紙のタイプ	*有効値 :* T = 熱転写用紙 D = ダイレクト・サーマル用紙 *デフォルト値 :* 値を指定しないとコマンドは無視されます。

^MU

測定単位の設定

説明 ^MU コマンドは、プリンタで使用する測定単位を設定します。^MU はフィールド単位で機能します。単位のモードは、いったん設定すると、新しい単位のモードを入力するまでフィールドからフィールドへと継続されます。

さらに、^MU は、低めの解像度での印刷も可能にします。すなわち、600 dpi のプリンタは 300、200、および 150 dpi での印刷が可能で、300 dpi プリンタは 150 dpi の印刷が可能です。

フォーマット ^MUa,b,c

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 単位	*有効値：* d = ドット数 I = インチ M = ミリメートル *デフォルト値：*D
b = フォーマット・ベース（インチあたりのドット数）	*有効値：*150, 200, 300 *デフォルト値：*値を指定しないとコマンドは無視されます。
c = 希望のインチあたりのドット数への変換	*有効値：*300, 600 *デフォルト値：*値を指定しないとコマンドは無視されます。

演習 1 次は単位の設定の例です。

8 ドット / ミリメートル（203 ドット / インチ）プリンタを想定します。

ドット数に基づくフィールド：

```
^MUd^FO100,100^GB1024,128,128^FS
```

ミリメートルに基づくフィールド：

```
^MUm^FO12.5,12.5^GB128,16,16^FS
```

インチに基づくフィールド：

```
^MUi^FO.493,.493^GB5.044,.631,.631^FS
```

例 2 • 次は dpi 値の変換の例です。

ドット単位を基本として、150 dpi フォーマットを 300 dpi フォーマットに変換します。

```
^MUd,150,300
```

ドット単位を基本として、150 dpi フォーマットを 600 dpi フォーマットに変換します。

```
^MUd,150,600
```

ドット単位を基本として、200 dpi フォーマットを 600 dpi フォーマットに変換します。

```
^MUd,200,600
```

変換係数を元のフォーマットにリセットするには、パラメータ b と c に同じ値を入力します。

```
^MUd,150,150
^MUd,200,200
^MUd,300,300
^MUd,600,600
```

コメント 正しい ZPL II フォーマットにするには、このコマンドをラベル・フォーマットの先頭に含める必要があります。

変換をオフにするには、パラメータ b と c に同じ値を入力します。

^MW

見出しの警告の変更

説明 操作環境に基づき、ヘッド低温警告インジケータを設定できます。

フォーマット ^MWy

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = ヘッド低温警告を有効にする	*有効値：* y = ヘッド低温警告を有効にする n = ヘッド低温警告を無効にする

重要 • パラメータを**指定しない**と、この命令は*無視されます*。

~NC

ネットワーク接続

説明 ~NC コマンドは、特定のプリンタのネットワーク ID 番号を呼出して、そのプリンタをネットワークに接続するために使用します。

フォーマット ~NC###

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
### ＝割り当てられたネットワーク ID 番号（3 桁入力必要）	*有効値：* 001 ～ 999 *デフォルト値：* 0000（なし）

コメント このコマンドは、ラベル・フォーマットの先頭で使用して、ネットワーク上のどのプリンタを使用するかを指定します。いったん設定したプリンタは、別の ~NC コマンドによって変更されるまで、続けて使用されます。プリンタを*目覚めさせる*には、ラベル・フォーマットにこのコマンドを含める必要があります。

コマンド ^MW、~NC、^NI、~NR、および ~NT は ZNET RS-485 プリンタ・ネットワークでのみ使用されます。

^NI

ネットワーク ID 番号

説明 ^NI コマンドは、ネットワーク ID 番号をプリンタに割り当てるために使用します。これはプリンタをネットワークで使用するために必要な手順です。

フォーマット ^NI###

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
### = 割り当てられたネットワーク ID 番号（3 桁入力必要）	*有効値：* 001 ～ 999 *デフォルト値：* 0000（なし）

コメント システムで認識されるのは、最後に設定されたネットワーク ID 番号です。

コマンド ~NC、^NI、~NR、および ~NT は ZNET RS-485 プリンタ・ネットワークでのみ使用されます。

~NR

すべてのネットワーク・プリンタを透過に設定

説明 ~NR コマンドは、ID や現在のモードにかかわらず、ネットワーク内のすべてのプリンタを透過に設定します。

フォーマット ~NR

コメント コマンド ~NC、^NI、~NR、および ~NT は ZNET RS-485 プリンタ・ネットワークでのみ使用されます。

^NS

ネットワーク設定の変更

説明 ^NS コマンドは、ネットワーク設定の変更に使用します。

フォーマット ^NSa,b,c,d

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = ネットワーク設定	*有効値：* IP 解像度 a（すべて）、b（BOOTP）、c（DHCP と BOOTP）、d（DHCP）、g（収集のみ）、r（RARP）、p（永久）
b = IP アドレス	*有効値：*0 ～ 255
c = サブネット・マスク	*有効値：*0 ～ 255
d = デフォルト・ゲートウェイ	*有効値：*0 ～ 255

~NT

現在接続しているプリンタを透過に設定

説明 ~NT コマンドは、現在接続しているネットワーク・プリンタを透過に設定します。

フォーマット ~NT

コメント *Z* Series™ プリンタでは、~NT コマンドは ~NR コマンドと同様に機能します。ネットワーク状のすべての *Z* Series プリンタは通信を受信します。

コマンド ~NC、^NI、~NR、および ~NT は ZNET RS-485 プリンタ・ネットワークでのみ使用されます。

^PF

スリューを適用するドット行数

説明 ^PF コマンドを使用すると、プリンタでラベルの一番下から指定のドット行数だけラベルがスリュー（印刷せずにラベルを高速で動かすこと）されます。ラベルの下の部分が空白の場合、このコマンドによって印刷が加速されます。

フォーマット ^PF#

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
# = スリューするドット行数	*有効値：* 0 ～ 32000 *デフォルト値：* 値を指定しないとコマンドは無視されます。

^PF ~PF

ホーム・ポジションまでスリュー

説明 ^PH または ~PH コマンドを使用すると、プリンタで空白のラベルが 1 枚フィードされます。

~PH コマンドは、プリンタをポーズ状態にすると、現在印刷中のフォーマットの終了後、ラベルを 1 枚フィードします。

^PH コマンドは、現在のフォーマットの印刷後、1 枚の空白ラベルをフィードします。

フォーマット ^PH または ~PH

^PM

ラベルのミラー・イメージの印刷

説明 ^PM コマンドは、ラベルの印刷可能領域全体をミラー・イメージとして印刷します。このコマンドは、イメージを左から右へ反転します。

フォーマット ^PMa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = ラベル全体のミラー・イメージを印刷する	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次はラベルにミラー・イメージを印刷する例です。

ZPL II CODE	GENERATED LABEL
^XA^PMY ^FO100,100 ^CFG ^FDMIRROR^FS ^FO100,160 ^FDIMAGE^FS ^XZ	

コメント パラメータが欠落しているか無効な場合は、このコマンドは無視されます。^PM コマンドは、いったん入力すると、^PMN を受信するかプリンタの電源を切るまでアクティブです。

^PO

印刷の向き

説明 ^PO コマンドはラベル・フォーマットを 180° 反転します。ラベルは逆さまに印刷されます。 元のラベルに ^LL、 ^LS、 ^LT、 および ^PF などのコマンドが含まれる場合、 反転したラベルの出力への影響は異なります。

フォーマット ^POa

次の表では、このフォーマットのパラメータを示します。

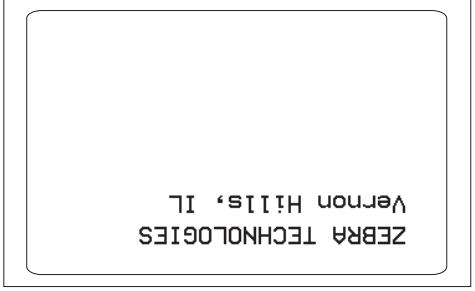
パラメータ	詳細
a = ラベルを 180° 反転	*有効値：*N（標準）または I（反転） *デフォルト値：*N

例 • 次はラベルを 180° 反転して印刷する例です。

ZPL II CODE

```
^XA^CFD
^POI
^LH330,10
^FO50,50
^FDZEBRA TECHNOLOGIES^FS
^FO50,75
^FDVernon Hills, IL^FS
^XZ
```

GENERATED LABEL



^POI コマンドは、x, y 座標を反転します。イメージの配置はすべて反転した座標に相対します。したがって、印刷をラベルに戻すために異なる ^LH（ラベル・ホーム）を使用できます。

コメント 同じラベル・フォーマット内で複数の ^PO コマンドが発行された場合、プリンタに最後に送信されたコマンドのみが使用されます。

^PO コマンドをいったん送信すると、別の ^PO コマンドを受信するかプリンタの電源を切るまで、その設定が維持されます。

^PP ~PP

プログラム可能ポーズ

説明 ~PP コマンドは、現在のラベルを終了すると（印刷中のものがある場合）印刷を停止し、プリンタをポーズ状態にします。

^PP コマンドは即座に実行されるものではありません。したがって、ポーズが実行される前に複数のラベルが印刷される場合もあります。このコマンドは、現在のフォーマットが印刷された後ポーズします。

操作は、プリンタのフロント・パネルで PAUSE を押すのと同じです。プリンタは PAUSE を押すまで、または ~PS（印刷開始）コマンドがプリンタに送信されるまでポーズを維持します。

フォーマット ^PP または ~PP

^PQ

印刷枚数設定

説明 ^PQ コマンドを使用するといくつかの印刷操作を制御できます。印刷するラベル枚数、プリンタのポーズ前に印刷されるラベル枚数、および各シリアル番号の複写枚数を制御できます。

フォーマット ^PQq,p,r,o

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
q ＝印刷するラベルの枚数合計	*有効値：*1 ～ 99,999,999 *デフォルト値：*1
p ＝ポーズ & 切断値（ポーズ間のラベル枚数）	*有効値：*1 ～ 99,999,999 *デフォルト値：*0（ポーズなし）
r ＝各シリアル番号の複写枚数	*有効値：*0 ～ 99,999,999（複写枚数） *デフォルト値：*0（複写なし）
o ＝ポーズ間隔を無視	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

o パラメータを Y に設定すると、プリンタは切断しますが、ポーズはしません。さらに、各グループのラベル枚数が印刷された後もプリンタは **ポーズしません**。o パラメータが N（デフォルト）に設定されている場合、プリンタはラベルの各グループのラベル枚数が印刷される都度、ポーズします。

例 • 次の例は、印刷操作の制御を示しています。

^PQ50,10,1,Y: この例は、合計 50 のラベルを印刷します。各シリアル番号に対して生成される複写枚数は 1 枚です。枚数合計を 10 ずつのグループに分けて印刷しますが、各グループの後にはポーズしません。

^PQ50,10,1,N: この例は、合計 50 のラベルを印刷します。各シリアル番号に対して生成される複写枚数は 1 枚です。枚数合計を 10 ずつのグループに分け、各グループの後にポーズしながら印刷します。

^PR

印字レート

説明 ^PR コマンドは、印刷中の用紙とスリュー速度（空白のラベルのフィード）を決定します。

プリンタは、設定を再発行するか、プリンタの電源を切るまで、選択した速度で動作します。

印字スピードはアプリケーションによって異なります。印字品質は、用紙、リボン、印字速度、およびプリンタ操作モードに影響を受けるため、各アプリケーションのテストを実行することが非常に大切です。

重要 • 電源を切るとデフォルトの印字速度になるモデルもあります。

フォーマット ^PRp,s,b

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
p = 印字速度	*有効値：*

A または 2	50.8 mm/ 秒	(2 インチ / 秒)
B または 3	76.2 mm/ 秒	(3 インチ / 秒)
C または 4	101.6 mm/ 秒	(4 インチ / 秒)
5	127 mm/ 秒	(5 インチ / 秒)
D または 6	152.4 mm/ 秒	(6 インチ / 秒)
E または 8	203.2 mm/ 秒	(8 インチ / 秒)

9	220.5 mm/ 秒	(9 インチ / 秒)
10	245 mm/ 秒	(10 インチ / 秒)
11	269.5 mm/ 秒	(11 インチ / 秒)
12	304.8 mm/ 秒	(12 インチ / 秒)

デフォルト値 : A

s = *スリュー速度*

有効値 :

A または 2	50.8 mm/ 秒	(2 インチ / 秒)
B または 3	76.2 mm/ 秒	(3 インチ / 秒)
C または 4	101.6 mm/ 秒	(4 インチ / 秒)
5	127 mm/ 秒	(5 インチ / 秒)
D または 6	152.4 mm/ 秒	(6 インチ / 秒)
E または 8	203.2 mm/ 秒	(8 インチ / 秒)
9	220.5 mm/ 秒	(9 インチ / 秒)
10	245 mm/ 秒	(10 インチ / 秒)
11	269.5 mm/ 秒	(11 インチ / 秒)
12	304.8 mm/ 秒	(12 インチ / 秒)

デフォルト値 : D

b = *バックフィード速度*

有効値 :

A または 2	50.8 mm/ 秒	(2 インチ / 秒)
B または 3	76.2 mm/ 秒	(3 インチ / 秒)
C または 4	101.6 mm/ 秒	(4 インチ / 秒)

5	127 mm/ 秒	（5 インチ / 秒）
D または 6	152.4 mm/ 秒	（6 インチ / 秒）
E または 8	203.2 mm/ 秒	（8 インチ / 秒）
9	220.5 mm/ 秒	（9 インチ / 秒）
10	245 mm/ 秒	（10 インチ / 秒）
11	269.5 mm/ 秒	（11 インチ / 秒）
12	304.8 mm/ 秒	（12 インチ / 秒）

*デフォルト値：*A

コメント p、s、および b の速度設定は、プリンタの制約によります。特定のプリンタが、6 ips（1 秒あたりのインチ）のレートに制限されている場合、12 という値を入力できてもプリンタは 6 ips のレートでしか印刷できません。パフォーマンスの詳細については、プリンタのユーザー・ガイドを参照してください。

~PR

アプリケータ再発行

説明 ~PR コマンドは、*PAX* および *PAX*2-Series プリンタでのみサポートされています。~PR コマンドが有効な場合（*^JJ ページ 238* を参照）、アプリケータがアプリケータ・ポートで再発行信号をアサートするのと同様に、最後に発行されたラベルが再発行されます。

フォーマット ~PR

コメント フロント・パネルで PREVIOUS を押しても、最後のラベルが再発行されます。

~PS

印刷再開

説明 ~PS コマンドは、ポーズ・モードのプリンタで印刷を再開します。この操作は、プリンタがすでにポーズ・モードになっている場合にプリンタのフロント・パネルで PAUSE を押すのと同じです。

フォーマット ~PS

^PW

印字幅

説明 ^PW コマンドを使用すると、印字幅を設定できます。

フォーマット ^PWa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = ラベル幅（ドット数）	*有効値：*2（ラベルの幅以下） 値がラベルの幅を超えた場合、幅はラベルの最大サイズに設定されます。 *デフォルト値：*最後に永久保存された値

制約 すべての Zebra プリンタが ^PW コマンドをサポートしているわけではありません。

~RO

高度機能を搭載したカウンタのリセット

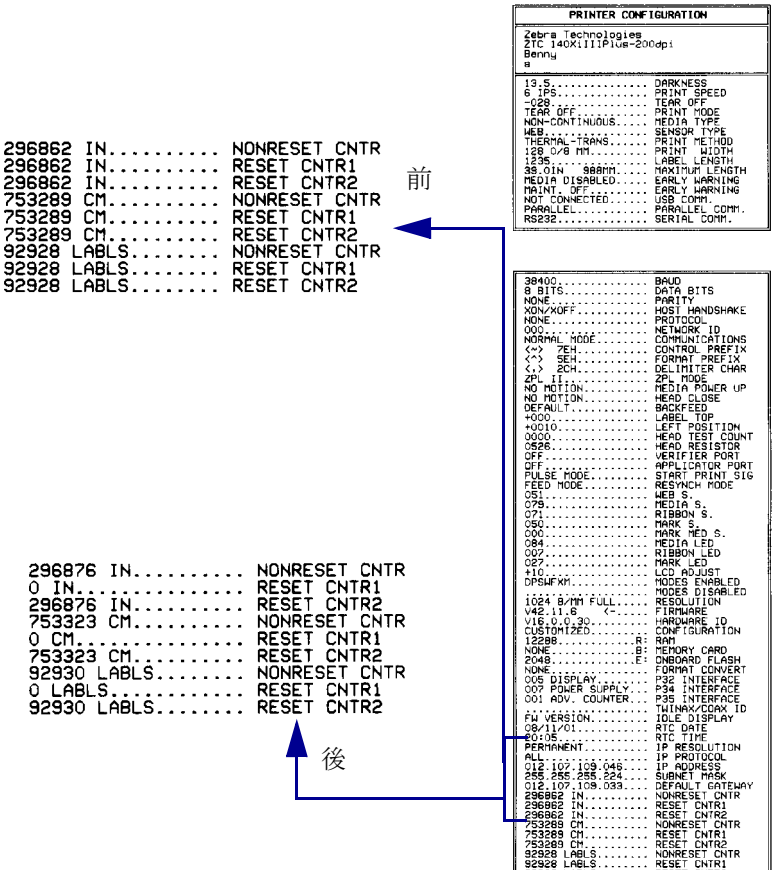
説明 ~RO コマンドは、プリンタでのインチおよびセンチでのラベルの生成やラベルの数などを監視するために使用する高度機能を搭載したカウンタをリセットします。リセット可能なカウンタは 2 つあります。

フォーマット ~ROc

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
c = カウンタ番号	*有効値：*1 または 2 *デフォルト値：*値を指定しないとコマンドは無視されます。

例 • 次は ~RO コマンドの例です。



^SC

シリアル通信の設定

説明 ^SC コマンドを使用すると、使用しているシリアル通信のパラメータを変更できます。

フォーマット ^SCa,b,c,d,e,f

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a ＝ボー・レート	*有効値：* 110; 600; 1,200; 2400; 4800; 9600; 14400; 19200; 28800; 38400; または 57600; 115000 *デフォルト値：*指定しないとパラメータは無視されます。
b ＝単語の長さ（データ・ビット）	*有効値：*7 または 8 *デフォルト値：*指定する必要があります
c＝パリティ	*有効値：*N（なし）、E（偶数）、または O（奇数） *デフォルト値：*指定する必要があります
d ＝ストップ・ビット	*有効値：*1 または 2 *デフォルト値：*指定する必要があります
e ＝プロトコル・モード	*有効値：*X（XON/XOFF）、D（DTR/DSR）、または R（RTS） *デフォルト値：*指定する必要があります
f ＝Zebra プロトコル	*有効値：* A＝ACK/NAK N＝なし Z＝Zebra *デフォルト値：*指定する必要があります

コメント パラメータが欠落している場合、仕様外の場合、特定のプリンタでサポートされていない場合、または ZPL 無視 DIP スイッチ・セットがある場合は、コマンドは無視されます。

^JUS コマンドを使用すると、通信モードの変更がパワーアップやソフトウェア・リセットを通して持続されます。

~SD

濃度の設定

説明 ~SD コマンドを使用すると、印字濃度を設定できます。~SD はフロント・パネル・ディスプレイの濃度設定パラメータに相当します。

フォーマット ~SD##

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
## ＝希望の濃度設定（2 桁の数値）	*有効値：* 00 ～ 30 *デフォルト値：*最後に永久保存された値

例 • 以下は XiIIIPlus の濃度設定例です。

```
^MD8.3

~SD8.3
```

コメント ^MD コマンド値は ~SD　コマンドに追加されます（該当する場合）。

^SE

エンコードの選択

説明 ^SE コマンドは、希望の ZPL または ZPL II エンコード表を選択するために作成されました。

フォーマット ^SEd:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = エンコード表の場所	*有効値：* R:、E:、B:、および A: *デフォルト値：* R:
o = エンコード表の名前	*有効値：* 1 ～ 8 文字の英数字 *デフォルト値：* 値を指定する必要があります
x = 拡張子	*固定値：* .DAT

^SF

連番フィールド（標準 ^FD ストリング使用）

説明 ^SF コマンドを使用すると、標準の ^FD ストリングを連番指定することができます。このコマンドで連番指定されたフィールドは、右揃えか、ストリングの最後の文字で終わります。増分ストリングは、右端の位置からマスクと位置合わせされます。マスクと増分ストリングの最大サイズは合計 3K です。

フォーマット ^SFa,b

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a ＝マスク・ストリング	マスク・ストリングは連番スキームを設定します。ストリング・マスクの長さは、連番指定する現在の ^FD ストリング内の文字数を定義します。マスクは、^FD ストリング内の文字に対して右端から位置合わせされます。 *マスク・ストリングのプレース・ホルダー：* D または d - 10 進数値 0 ～ 9 H または h - 16 進数 0 ～ 9 および a ～ f または A ～ F O または o - 8 進数 0 ～ 7 A または a - 英字 a ～ z または A ～ Z N または n - 英数字 0 ～ 9 および a ～ z または A ～ Z % - 文字を無視するかスキップする
b ＝増分ストリング	増分ストリングとは、各レベルのフィールドに追加する値のことです。デフォルト値は、10 進数値の 0 に相当します。このストリングは、連番ストリングで定義された任意の文字で構成されます。無効な文字は、その文字位置におけるゼロの値です。 英字のストリングの増分値は、'A' または 'a' をゼロのプレースホルダーとして始まります。これは、英字を 1 で増分するためには、'B' または 'b' の値が増分ストリングに存在する必要があることを意味します。

増分されない文字の場合は、増分ストリングに % 文字を 追加する必要があります。

例 • 次は ^FD ストリングを連番にする例です。

ZPL II CODE	GENERATED LABELS
^XA ^FO100,100 ^CF0,100 ^FD12A^SFnnA,F^FS ^PQ3 ^XZ *Note: The ZPL II code above will generate three separate labels, seen to the right.*	12A 12F 12K

このマスクでは、最初の文字が英数字（nn = 12）で、最後の桁が大文字の英字（A）になっています。増分数値の 10 進値は 5（F）です。生成されるラベルの数は、^PQ コマンドで設定された数によって決定されます。

同様な例として、^FD ストリングは、以下のどちらかの ^FD ストリングで置換して、^PQ で決定される一連のラベルを生成できます。

```
^FDBL0000^SFAAdddd,1
```

この一連のラベルの印刷シーケンスは次のとおりです。

```
BL0000, BL0001,...BL0009, BL0010,...

BL0099, BL0100,...BL9999, BM0000...
```

```
^FDBL00-0^SFAAdd%d,1%1
```

この一連のラベルの印刷シーケンスは次のとおりです。

```
BL00-0, BL01-1, BL02-2,...BL09-9,

BL11-0, BL12-1...
```

^SL

モードと言語の設定（リアル・タイム・クロック用）

説明 ^SL コマンドは、リアルタイム・クロックの操作モードと情報印刷のための言語の指定に使用します。

フォーマット ^SLa,b

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = モード	*有効値：* S = 開始時刻モード。ラベル・フォーマットの開始時（^XA の受信時）にリアルタイム・クロックから読み取られる時刻です。最初のラベルには最後のラベルと同じ時刻が印刷されます。 T = 現時刻モード。印刷するラベルが印刷キューに入れられた時にリアルタイム・クロックから読み取られる時刻です。*現時刻* は、連番指定した時刻 / 日付フィールドと同様です。 *デフォルト値：*S
b = 言語	*有効値：* 1 = 英語 2 = スペイン語 3 = フランス語 4 = ドイツ語 5 = イタリア語 6 = ノルウェー語 7 = ポルトガル語 8 = スウェーデン語 9 = デンマーク語 10 = スペイン語 2 11 = オランダ語 12 = フィンランド語 *デフォルト値：*^KL またはフロント・パネルで選択された言語

^SN

連番データ

説明 ^SN コマンドを使用すると、選択した増分値または減分値によってデータ・フィールドにインデックスを付けて、ラベルが印刷される都度、データ・フィールドが増分または減分するようにすることができます。これは指定のフォーマットの 100 ～ 150 のフィールドに対して実行することができるほか、英数字フィールドとバー・コード・フィールドの両方で実行できます。最大 12 までの右端の整数がインデックス付けの対象となります。右から左へスキャンして検出された最初の整数が、データ・フィールドのインデックス部分を開始します。

インデックス付けする英数字フィールドが英字で終わる場合、数値文字が検出されるまで、データは 1 文字ずつ右から左にスキャンされます。連番指定は、最初に検出された数値を使用して行われます。

フォーマット ^SNv,n,z

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
v = 開始値	*有効値：*インデックスを付ける部分に対して最大 12 桁 *デフォルト値：*1
n = 増分値または減分値	*有効値：*最大 12 桁 *デフォルト値：*1 減分値を示すには、マイナス（-）符号を前に付けます。
z = 前ゼロを追加する（必要な場合）	*有効値：*Y（はい）または N （いいえ） *デフォルト値：*N

例 • 次の例は、特定の値による増分を示しています。

ZPL II CODE	GENERATED LABELS
^XA ^FO260,110 ^CFG ^SN001,1,Y^FS ^PQ3 ^XZ *Note: The ZPL II code above will generate three separate labels, seen to the right.*	001 002 003

コメント ^PQ（印刷枚数設定）コマンドの r パラメータで指定した複写枚数が各シリアル番号に対してすべて印刷されると、各連番フィールドの増分と減分が実行されます。

連番ラベルの印刷過程でプリンタが用紙切れまたはリボン切れになると、用紙またはリボンの取り替え後最初に印刷されたラベルは、*用紙切れ / リボン切れ* の状態が発生する前に *部分的* に印刷されたラベルと同じシリアル番号になります。これは、*用紙切れ / リボン切れ* の状態になる前の最後のラベルが完全に印刷されなかった場合のためです。これは ^JZ コマンドによって制御されます。

前ゼロの使用

^SN コマンドでは、z パラメータによって前ゼロを印字するか抑制するかが決定されます。使用する値（Y = 前ゼロを印刷する、N = 前ゼロを印刷しない）によって、プリンタは前ゼロ処理を決定します。

このパラメータのデフォルト値は、N（前ゼロを印刷しない）です。

前ゼロ印字

スタート値は、右端からの連続した数字で構成されます。幅（シーケンス内の桁数）は、右から左へスキャンして最初に検出した数字以外の文字（スペースまたは英字）によって決まります。特定の幅を作成するには、必要に応じて手動で前ゼロを配置してください。

前ゼロの抑制

スタート値は、先行スペースも含め、右端からの連続した数字で構成されます。幅（シーケンス内の桁数）は、右から左へスキャンして最初に検出した英字（スペースは除く）によって決まります。特定の幅を作成するには、必要に応じて手動で先行スペースまたは前ゼロを配置してください。抑制したゼロはスペースで置換されます。連番指定過程で、全数値がすべてゼロを含んでいる場合、最後のゼロは抑制されません。

^SN コマンドは、ラベル・フォーマット・プログラム内でフィールド・データ（^FD）コマンドの代わりに使用します。

^SO

オフセットの設定（リアルタイム・クロック用）

説明 ^SO コマンドは、1 次リアルタイム・クロックから 2 次、3 次のオフセットを設定する場合に使用します。

フォーマット ^SOa,b,c,d,e,f,g

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a＝クロック・セット	*有効値：*2（2 次）または 3（3 次） *デフォルト値：*値を指定する必要があります
b ＝月のオフセット	*有効値：*-32000 ～ 32000 *デフォルト値：*0
c ＝日のオフセット	*有効値：* -32000 ～ 32000 *デフォルト値：*0
d ＝年のオフセット	*有効値：* -32000 ～ 32000 *デフォルト値：*0
e ＝時間のオフセット	*有効値：* -32000 ～ 32000 *デフォルト値：*0
f ＝分のオフセット	*有効値：* -32000 ～ 32000 *デフォルト値：*0
g ＝秒のオフセット	*有効値：*-32000 ～ 32000 *デフォルト値：*0

^SP

印刷の開始

説明 ^SP コマンドを使用すると、ラベル全体が完全にフォーマットされる前に、指定の時点でラベルの印刷を開始できます。非常に複雑なラベルの場合、このコマンドを使用すると印刷の全体のスループットを改善できます。

このコマンドは次のように機能します。まず、^SP コマンドを開始するドット行を指定します。これにより、ラベル・*セグメント*が作成されます。^SP コマンドが処理されると、そのセグメントの全情報が印刷されます。印刷処理中、^SP の後に続くコマンドは、すべて継続してプリンタで受信され、処理されます。

^SP コマンドの後のセグメント（またはラベルの残り）の準備ができていれば、印刷は停止せずに継続されます。次のセグメントの準備ができていない場合は、プリンタはラベルの途中で停止し、次のセグメントの準備完了を待ちます。^SP コマンドの正確な位置指定は、印字速度やラベルの複雑さなどに左右されるため、試行錯誤が必要になります。

^SP コマンドを効果的に使用して最悪の印字品質を決定することができます。この手順を使用することにより、そのアプリケーションで ^SP コマンドを使用することが適切であるかどうかを判断できます。

最初の ^SP コマンドまでのラベル・フォーマットを送信したら、印刷が停止するまで次のセグメントを送信するのを待ちます。印刷されたラベルは最悪の印字品質のサンプルになります。異常のあるフィールドはすべてドロップされます。

上記の手順を使用した場合、ラベル・フォーマットの終わりは次のようになります。

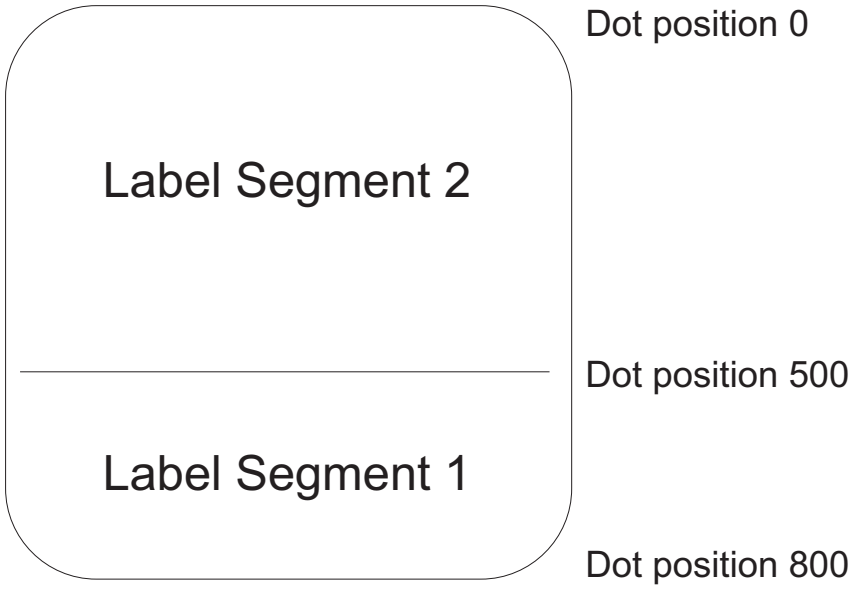
```
^SP#^FS
```

フォーマット ^SPa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = 印刷を開始するドット行	*有効値：*0 ～ 32000 *デフォルト値：*0

例 • 次の例では、長さが 800 ドット行のラベルで ^SP500 を使用しています。セグメント 1 は、セグメント 2 のコマンドの受信 / フォーマット中に印刷されます。



^SQ

ZebraNet Alert の停止

説明 ^SQ コマンドは、ZebraNet Alert オプションの停止に使用します。

フォーマット ^SQa,b,c

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a ＝状態タイプ	*有効値：* A＝用紙切れ B ＝リボン切れ C＝印刷ヘッド過剰高温 C＝印刷ヘッド過剰低温 E ＝ヘッド・オープン C＝印刷ヘッド過剰高温 G＝リボンあり警告（ダイレクト・サーマル・モード） H＝巻き取りがフル I＝デフォルト・プリンタ J ＝切り取りエラー K ＝プリンタ・ポーズ L＝PQ ジョブ完了 M＝ラベル剥離 N ＝ ヘッド・エレメントなし O＝ZBI（Zebra BASIC Interpreter）ランタイム・エラー P＝ZBI（Zebra BASIC Interpreter）強制エラー Q＝電源オン R＝すべてのエラー *＝すべての状態に対してアラートを止めるワイルドカード

パラメータ	詳細
b =宛先	*有効値：* A = シリアル・ポート B = パラレル・ポート C = 電子メールアドレス D = TCP/IP E = UDP/IP F = SNMP トラップ * = すべての宛先に対してアラートを止めるワイルドカード
c = メッセージの停止	*有効値：*Y（メッセージの停止）または N （メッセージの開始） *デフォルト値：*Y

^SR

印刷ヘッドの抵抗値の設定

説明 ^SR コマンドを使用すると、印刷ヘッドの抵抗値を設定できます。

フォーマット ^SR####

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
#### ＝抵抗値 (4 桁の数値)	*有効値：* 0488 ～ 1175 *デフォルト値：*最後に永久保存された値

コメント 印刷ヘッドの損傷を防ぐため、この値は使用している印刷ヘッドで示されている値以下である必要があります。高い値を設定すると、印刷ヘッドが損傷するおそれがあります。

注記 • 新しいモデルではヘッドの抵抗値は自動的に設定されます。

^SS

メディア・センサーの設定

説明 ^SS コマンドは、用紙のキャリブレーション処理で設定された用紙、ウェブ、リボン、およびラベル長の値の変更に使用します。用紙のキャリブレーション処理は、ご使用のプリンタのユーザー・ガイドで説明されています。

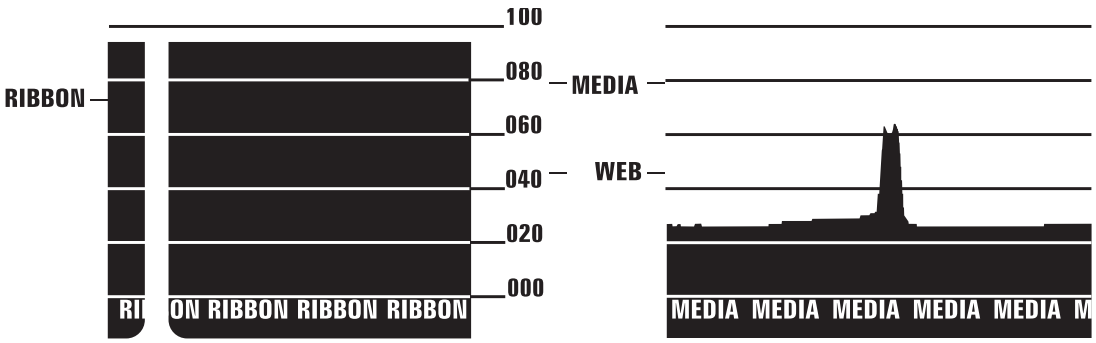
フォーマット ^SSw,m,r,l,m2,r2,a,b,c

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
w = ウェブ（3 桁の値）	*有効値：* 000 ～ 100 *デフォルト値：*用紙センサー・プロフィールまたは設定ラベルに示された値
m = 用紙（3 桁の値）	*有効値：* 000 ～ 100 *デフォルト値：*用紙センサー・プロフィールまたは設定ラベルに示された値
r = リボン（3 桁の値）	*有効値：*001 ～ 100 *デフォルト値：*用紙センサー・プロフィールまたは設定ラベルに示された値
l= ラベル長（ドット数で表す 4 桁の値）	*有効値：* 0001 ～ 32000 *デフォルト値：*キャリブレーション処理で計算された値
m2 = 用紙 LED 光力（3 桁の値）	*有効値：* 000 ～ 100 *デフォルト値：*キャリブレーション処理で計算された値
r2 = リボン LED 光力（3 桁の値）	*有効値：* 000 ～ 100 *デフォルト値：*キャリブレーション処理で計算された値

パラメータ	詳細
a = マーク検知（3 桁の値）	*有効値：* 000 ～ 100 *デフォルト値：*キャリブレーション処理で計算された値
b = マーク用紙検知（3 桁の値）	*有効値：* 000 ～ 100 *デフォルト値：*キャリブレーション処理で計算された値
c = マーク LED 検知（3 桁の値）	*有効値：* 000 ～ 100 *デフォルト値：*キャリブレーション処理で計算された値

例 • 次は用紙センサー・プロフィールの例です。000 ～ 100 の数値に並んで WEB、MEDIA、RIBBON が表示されています。また、縦方向には黒い突起状の図形が表示されています。これは、プリンタが用紙 - ウェブ - 用紙への移行を検知した部分を表しています。



生成される用紙とセンサーのプロフィールは、プリンタ間で異なります。

コメント m2 と r2 パラメータは、Stripe® *S*-300 と *S*-500 のプリンタには影響しません。

パラメータの最大値は、使用するプリンタ・プラットフォームによって異なります。

^ST

日付と時刻の設定（リアル・タイム・クロック用）

説明 ^ST コマンドは、リアルタイム・クロックの日付と時刻を設定します。

フォーマット ^STa,b,c,d,e,f,g

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a ＝月	*有効値：*01 ～ 12 *デフォルト値：*現在の月
b ＝日	*有効値：*01 ～ 31 *デフォルト値：*現在の日
c＝年	*有効値：*1998 ～ 2097 *デフォルト値：*現在の年
d＝時間	*有効値：* 00 ～ 23 *デフォルト値：*現在の時間
e ＝分	*有効値：* 00 ～ 59 *デフォルト値：*現在の分
f ＝秒	*有効値：* 00 ～ 59 *デフォルト値：*現在の秒
g ＝フォーマット	*有効値：* A＝a.m. P＝p.m. M＝24 時間式 *デフォルト値：*M

^SX

ZebraNet Alert の設定

説明 ^SX コマンドは、ZebraNet Alert システムの設定に使用します。

フォーマット ^SXa,b,c,d,e,f

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a ＝状態タイプ	*有効値：* A＝用紙切れ B ＝リボン切れ C＝印刷ヘッド過剰高温 C＝印刷ヘッド過剰低温 E ＝ヘッド・オープン F＝電源過剰高温 G＝リボンあり警告（ダイレクト・サーマル・モード） H＝巻き取りがフル I＝デフォルト・プリンタ J ＝切り取りエラー K ＝プリンタ・ポーズ L＝PQ ジョブ完了 M＝ラベル剥離 N ＝ ヘッド・エレメントなし O＝ZBI（Zebra BASIC Interpreter）ランタイム・エラー P＝ZBI（Zebra BASIC Interpreter）強制エラー Q＝電源オン R＝すべてのエラー *デフォルト値：*パラメータが欠落しているか無効な場合は、このコマンドは無視されます。
b ＝ ルーティング先アラート	*有効値：* A＝シリアル・ポート B*＝パラレル・ポート C＝電子メールアドレス D＝TCP/IP E＝UDP/IP F＝SNMP トラップ *デフォルト値：*パラメータが欠落しているか無効な場合は、このコマンドは無視されます。 *双方向通信を**必要**とします。

パラメータ	詳細
c = この宛先への*条件設定*アラートを有効にする	*有効値 :* Y（はい）または N （いいえ） *デフォルト値 :* Y または前に設定された値
d = この宛先への*条件クリア*アラートを有効にする	*有効値 :* Y（はい）または N （いいえ） *デフォルト値 :* N または前に設定された値
	e と f のパラメータは宛先に基づくサブオプションです。サブオプションが欠落しているか無効な場合は、これらのパラメータは無視されます。
e = 宛先設定	*有効値 :* インターネット電子メール・アドレス（user@company.com など） IP アドレス（10.1.2.123 など） SNMP トラップ IP または IPX アドレス
f = ポート番号	*有効値 :* TCP ポート番号（0 ～ 65535） UPD ポート番号（0 ～ 65535） **例** シリアル：^SXA,A,Y,Y パラレル：^SXA,B,Y,Y 電子メール：^SXA,C,Y,Y,admin@company.com TCP：^SXA,D,Y,Y,123.45.67.89,1234 UDP：^SXA,E,Y,Y,123.45.67.89,1234 SNMP トラップ：^SXA,F,Y,Y,255.255.255.255

コメント 上の SNMP トラップの例では、255.255.255.255 を入力するとネットワーク上のすべての SNMP マネージャに通知がブロードキャストされます。デバイスを単一の SNMP マネージャにルーティングするには、特定のアドレス（123.45.67.89）を入力します。

^SZ

ZPL の設定

説明 ^SZ コマンドは、プリンタで使用するプログラミング言語の選択に使用します。このコマンドにより、ZPL と ZPL II の両方のフォーマットのラベルを印刷できます。

このコマンドは、別の ^SZ コマンドがプリンタに送信されるか、プリンタの電源を切るまでアクティブです。

フォーマット ^SZa

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
a = ZPL バージョン	*有効値：* 1 = ZPL 2 = ZPL II *デフォルト値：*2

コメント パラメータが欠落しているか無効な場合は、このコマンドは無視されます。

~TA

切り取り位置の調整

説明 ~TA コマンドを使用すると、ラベル印刷後の用紙の静止位置を調整できます。ラベルはこの位置で切り取られます。

フォーマット ~TA###

重要 • 以下はこのコマンドに関する注意点です。

- 600 dpi プリンタの場合は、ステップ・サイズが倍になります。
- 文字数が 3 文字**より少ない**と、このコマンドは無視されます。

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
### = 用紙の静止位置の変更 **重要** • ドット行数を表す 3 桁の値を使用する必要があります。	*有効値：* -120 ～ 120 *デフォルト値：*最後に永久保存された値

コメント パラメータが欠落しているか無効な場合は、このコマンドは無視されます。

^TO

オブジェクト・コピー

説明 ^TO コマンドは、オブジェクトまたはオブジェクトのグループを 1 つのストレージ・デバイスから別のストレージ・デバイスにコピーする場合に使用します。PC で使用するコピー機能と同様です。

コピー元とコピー先のデバイスの指定が必要で、それらが異なるデバイスであること、および指定のアクションに対して有効であることが必要です。無効なパラメータを指定するとこのコマンドは無視されます。

アステリスク（*）は、オブジェクト名と拡張子のワイルドカードとして使用できます。たとえば、ZEBRA.* や *.GRF は ^TO コマンドで使用できる有効な形式です。

コピー元パラメータ（d、o、または x）およびコピー先パラメータ（s, o、x）をそれぞれ少なくとも 1 つ指定する必要があります。^TO しか入力しなかった場合は、このコマンドは無視されます。

フォーマット ^TOd:o.x,s:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = 保存されたオブジェクトのコピー元デバイス	*有効値：* R:、E:、B:、および A: *デフォルト値：* ドライバを指定しないと、パラメータ s で設定されたドライブにすべてのオブジェクトがコピーされます。
o = 保存されたオブジェクト名	*有効値：* Zebra の規則に準拠した既存のすべてのオブジェクト *デフォルト値：* 名前を指定しない場合 * が使用されます。すなわち、すべてのオブジェクトが選択されます。

パラメータ	詳細
x = 拡張子	*有効値 :* Zebra の規則に準拠したすべての拡張子 *デフォルト値 :* 拡張子を指定しない場合 * が使用されます。すなわち、すべての拡張子が選択されます。
s = 保存されたオブジェクトのコピー先デバイス	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* コピー先を指定する必要があります
o = コピー先のオブジェクトの名前	*有効値 :* 最高 8 文字までの英数字 *デフォルト値 :* 名前を指定しない場合、既存のオブジェクトの名前が使用されます。
x = 拡張子	*有効値 :* Zebra の規則に準拠したすべての拡張子 *デフォルト値 :* 拡張子を指定しない場合、既存のオブジェクトの拡張子が使用されます。

コメント パラメータ o、x、および s ではワイルドカード（*）の使用がサポートされています。

コピー先デバイスにコピーするオブジェクトを保存するために十分な空き容量がない場合、このコマンドはキャンセルされます。

Zebra ファイル（Z:*.*）はコピーできません。これらのファイルの著作権は Zebra Technologies が所有しています。

オブジェクトのコピー

次は ^TO コマンドの使用例です。

DRAM にある ZLOGO.GRF というオブジェクトを ZLOGO1.GRF という名前に変更して、オプションのメモリ・カードにコピーするには、次のフォーマットを使用します。

```
^XA
^TOR:ZLOGO.GRF,B:ZLOGO1.GRF
^XZ
```

メモリ・カードにある SAMPLE.GRF というオブジェクトを同じ名前で DRAM にコピーするには、次のフォーマットを使用します。

```
^^XA
^TOB:SAMPLE.GRF,R:SAMPLE.GRF
^XZ
```

複数オブジェクトのコピー

アステリスク（*）を使用すると、複数のオブジェクト・ファイル（*.FNT 以外）を DRAM からメモリ・カードにコピーすることができます。たとえば、ロゴを含んだ複数のオブジェクト・ファイルがあるとします。これらのファイルは、LOGO1.GRF、LOGO2.GRF、および LOGO3.GRF だとします。

これらのファイルをすべて、LOGO ではなくて NEW という名前を使ってメモリ・カードにコピーするには、コピー・コマンドで NEW と LOGO の後にアステリスクを挿入します。これにより、1 つのコマンドで LOGO で始まるすべてのファイルがコピーされます。

```
^XA
^TOR:LOGO*.GRF,B:NEW*.GRF
^XZ
```

複数ファイルのコピー中、大きすぎてメモリ・カードに保存できないファイルはスキップされます。残りのすべてのファイルのコピーが試みられます。容量の制約内で保存できるすべてのファイルがコピーされ、保存できないファイルは無視されます。

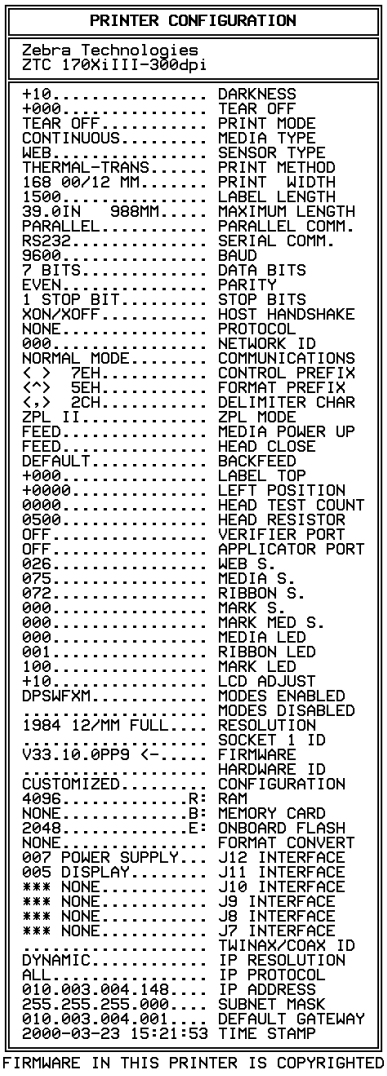
~WC

設定ラベルの印刷

説明 ~WC コマンドは、プリンタ設定ラベルの生成に使用します。プリンタ設定ラベルには、センサー・タイプ、ネットワーク ID、ZPL モード、ファームウェア・バージョン、および R:、E:、B:、A: デバイスに関する記述データなど、プリンタのセットアップに関する情報が含まれます。

フォーマット ~WC

コメント このコマンドはプリンタがアイドル状態の場合にのみ機能します。



```
PRINTER CONFIGURATION

Zebra Technologies
ZTC 170XiIII-300dpi

+10.................. DARKNESS
+000................. TEAR OFF
TEAR OFF............. PRINT MODE
CONTINUOUS........... MEDIA TYPE
WEB.................. SENSOR TYPE
THERMAL-TRANS........ PRINT METHOD
168 00/12 MM........ PRINT  WIDTH
1500................. LABEL LENGTH
39.0IN   988MM..... MAXIMUM LENGTH
PARALLEL............. PARALLEL COMM.
RS232................ SERIAL COMM.
9600................. BAUD
7 BITS............... DATA BITS
EVEN................. PARITY
1 STOP BIT........... STOP BITS
XON/XOFF............. HOST HANDSHAKE
NONE................. PROTOCOL
000.................. NETWORK ID
NORMAL MODE.......... COMMUNICATIONS
<~>  7EH............. CONTROL PREFIX
<^>  5EH............. FORMAT PREFIX
<,>  2CH............. DELIMITER CHAR
ZPL II............... ZPL MODE
FEED................. MEDIA POWER UP
FEED................. HEAD CLOSE
DEFAULT.............. BACKFEED
+000................. LABEL TOP
+0000................ LEFT POSITION
0000................. HEAD TEST COUNT
0500................. HEAD RESISTOR
OFF.................. VERIFIER PORT
OFF.................. APPLICATOR PORT
026.................. WEB S.
075.................. MEDIA S.
072.................. RIBBON S.
000.................. MARK S.
000.................. MARK MED S.
000.................. MEDIA LED
001.................. RIBBON LED
100.................. MARK LED
+10.................. LCD ADJUST
DPSWFXM.............. MODES ENABLED
..................... MODES DISABLED
1984 12/MM FULL...... RESOLUTION
..................... SOCKET 1 ID
V33.10.0PP9 <-...... FIRMWARE
..................... HARDWARE ID
CUSTOMIZED.......... CONFIGURATION
4096..............R: RAM
NONE..............B: MEMORY CARD
2048..............E: ONBOARD FLASH
NONE................. FORMAT CONVERT
007 POWER SUPPLY.... J12 INTERFACE
005 DISPLAY......... J11 INTERFACE
*** NONE............ J10 INTERFACE
*** NONE............ J9 INTERFACE
*** NONE............ J8 INTERFACE
*** NONE............ J7 INTERFACE
..................... TWINAX/COAX ID
DYNAMIC.............. IP RESOLUTION
ALL.................. IP PROTOCOL
010.003.004.148..... IP ADDRESS
255.255.255.000..... SUBNET MASK
010.003.004.001..... DEFAULT GATEWAY
2000-03-23 15:21:53 TIME STAMP

FIRMWARE IN THIS PRINTER IS COPYRIGHTED
```

^WD

ディレクトリ・ラベルの印刷

説明 ^WD コマンドは、ラベル・リスト・バー・コード、DRAM に保存されたオブジェクト、またはフォントなどの印刷に使用します。

バー・コードの場合、バー・コード名がリストされます。フォントの場合、フォント名、^Af コマンドで使用する数値、およびサイズがリストされます。DRAM に保存されたオブジェクトの場合、オブジェクト名、拡張子、サイズ、およびオプション・フラグがリストされます。すべてのリストは 2 重線のボックスで囲まれています。

フォーマット ~WDd:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = ソースのデバイス（オプション）	*有効値：*R:、E:、B:、A: および Z: *デフォルト値：*R:
o = オブジェクト名（オプション）	*有効値：*1 ～ 8 文字の英数字 *デフォルト値：** ?（疑問符）の使用も可能です。

パラメータ	詳細
x = 拡張子 （オプション）	*有効値：* Zebra の規則に準拠したすべての拡張子 .FNT = フォント .BAR = バー・コード .ZPL = 保存された ZPL フォーマット .GRF = GRF グラフィック .CO = メモリ・キャッシュ .DAT = フォント・エンコード .BAS = ZBI プログラム .STO = データ・ストレージ .PNG = PNG グラフィック * = すべてのオブジェクト *デフォルト値：* * ?（疑問符）の使用も可能です。

例 1 • DRAM にあるオブジェクトをすべてリストしたラベルを印刷するには、次のように入力します。

```
^XA
^WDR:*.*
^XZ
```

例 2 • 常駐するすべてのバー・コードをリストしたラベルを印刷するには、次のように入力します。

```
^XA
^WDZ:*.BAR
^XZ
```

例 3 • 常駐するすべてのフォントをリストしたラベルを印刷するには、次のように入力します。

```
^XA
^WDZ:*.FNT
^XZ
```

^XA

開始フォーマット

説明 ^XA コマンドは、ZPL II コードの先頭に使用します。これは新しいラベル・フォーマットの始めを表す左ブラケットです。このコマンドの代わりに単一の ASCII コントロール文字 STX（Control-B、16 進 02）を使用することもできます。

フォーマット ^XA

コメント 有効な ZPL フォーマットでは、ラベル・フォーマットが ^XA コマンドで始まり、^XZ コマンドで終わることが必要です。

^XB

バックフィードの抑制

説明 ^XB コマンドは、現在のプリンタ・モードにより、切り取り位置までの用紙のフィードを抑制します。フィードが発生しない場合、次のラベルの印刷前のバックフィードは不要となるため、スループットが改善されます。ラベルのバッチを印刷する場合、最後のラベルにはこのコマンドを含めないようにしてください。

フォーマット ^XB

切り取りモードの ^XB

標準操作 : バックフィード、印刷、切り取り位置までフィード

^XB *操作 :* 印刷（巻き取りモード）

剥離モードの ^XB

標準操作 : バックフィード、印刷、切り取り位置までフィード

^XB *操作 :* 印刷（巻き取りモード）

^XF

フォーマット呼び出し .

説明 ^XF コマンドは保存されたフォーマットを呼び出し、変数データと結合できます。1 つのフォーマットで、複数の ^XF コマンドを使用することができ、コード内の任意の位置に配置できます。

保存されたフォーマットを呼び出して、^FN（フィールド番号）機能を使用してデータを結合する際、データを正しく結合するためには、呼び出しのフォーマットに ^FN コマンドが含まれている必要があります。

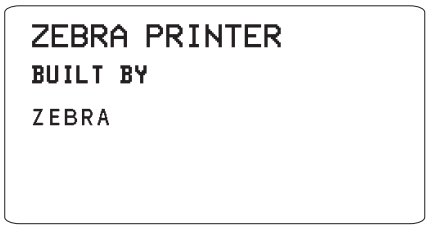
保存されたフォーマットを使用すると送信時間は短縮されますが、フォーマットにかかる時間は短縮されません。呼び出される ZPL II フォーマットはテキスト・ストリングとして保存されるため、印刷時にフォーマットする必要があります。

フォーマット ^XFd:o.x

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = 保存されたイメージのソース・デバイス	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* 検索優先順序（R:、E:、B:、および A:）
o = 保存されたイメージの名前	*有効値 :* 1 ～ 8 文字の英数字 *デフォルト値 :* 名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子 1	*固定値 :* ZPL

例 • 次は、^XF コマンドを使用して DRAM からフォーマット STOREFMT.ZPL を呼び出し、 ^FN フィールドに新しい参照データを挿入する例です。

ZPL II CODE	GENERATED LABEL
^XA ^XFR:STOREFMT.ZPL^FS ^FN1^FDZEBRA^FS ^FN2^FDLABEL^FS ^XZ	

^XG

グラフィックの呼び出し .

説明 ^XG コマンドは、1 つまたは複数のグラフィック・イメージを呼び出して印刷するために使用します。このコマンドは、ラベル・フォーマットで使用し、会社のロゴなどのグラフィックスをテキスト・データと結合し、完全なラベルを作成できます。

イメージは、各フォーマットで必要に応じて何度でも呼び出し、サイズ変更できます。その他のイメージとデータもフォーマットに追加できます。

フォーマット ^XGd:o.x,mx,my

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
d = 保存されたイメージのソース・デバイス	*有効値 :* R:、E:、B:、および A: *デフォルト値 :* 検索優先順序（R:、E:、B:、および A:）
o = 保存されたイメージの名前	*有効値 :* 1 ～ 8 文字の英数字 *デフォルト値 :* 名前を指定しない場合は UNKNOWN が使用されます。
x = 拡張子 l	*固定値 :* .GRF
mx = x 軸上の倍率係数	*有効値 :* 1 ～ 10 *デフォルト値 :* 1
my = y 軸上の倍率係数	*有効値 :* 1 ～ 10 *デフォルト値 :* 1

例 • 次は、^XG コマンドを使用して DRAM からイメージ SAMPLE.GRF を呼び出し、5 つの異なるサイズで 5 つの異なる場所に同じラベルを印刷する例です。

```
^XA
^FO100,100^XGR:SAMPLE.GRF,1,1^FS
^FO100,200^XGR:SAMPLE.GRF,2,2^FS
^FO100,300^XGR:SAMPLE.GRF,3,3^FS
^FO100,400^XGR:SAMPLE.GRF,4,4^FS
^FO100,500^XGR:SAMPLE.GRF,5,5^FS
^XZ
```

^XZ

終了フォーマット

説明 ^XZ コマンドは終わりを表す右ブラケットです。ラベル・フォーマットの終わりを意味します。このコマンドを受け取るとラベルが印刷されます。このコマンドは、単一の ASCII コントロール文字 ETX（Control-C、16 進 03）を発行することもできます。

フォーマット ^XZ

コメント 有効な ZPL フォーマットでは、ラベル・フォーマットが ^XA コマンドで始まり、^XZ コマンドで終わることが必要です。

^ZZ

プリンタ・スリープ

説明 ^ZZ コマンドは、プリンタをアイドルまたはシャットダウン・モードにします。

フォーマット ^ZZt,b

次の表では、このフォーマットのパラメータを示します。

パラメータ	詳細
t = シャットダウン前の秒数（アイドル時間）	*有効値：* 0 ～ 999999（0 に設定すると、自動シャットダウンが無効になります） *デフォルト値：* 最後に永久保存された値または 0
b = シャットダウン時のステータス	*有効値：* Y = ラベルがまだキューにあるときにシャットダウンすることを意味します。 N = シャットダウン前にすべてのラベルを印刷する必要があることを意味します。 *デフォルト値：* N

コメント ^ZZ コマンドは PA400 と PT400 バッテリ電源式のプリンタでのみ有効です。


Zebra

ZBI コマンド

この項には、ZPL のコマンドのアルファベット順のリストが含まれます。ZBI コードが認識されるかどうかは、ご使用のファームウェアのバージョンによって決まります。

この項のテキストは、次の見出しの下に編成されています。

説明 この見出しの項では、コマンドの使用方法、その機能、およびその特徴などの説明を提供します。

フォーマット フォーマットでは、コマンドの編成方法や、含まれるパラメータについて説明します。たとえば、AUTONUM コマンドは自動付番オプションを開始します。このコマンドのフォーマットは AUTONUM <A>,<B> です。<A> と <B> はこのコマンドのパラメータで、ユーザーが決定する値によって置換されます。

パラメータ コマンドの機能をさらに特定するために定義できる値がコマンドに含まれる場合、それらはこの見出しの下にリストされます。引き続き AUTONUM の例を使用しますが、<A> パラメータは次のように定義されます。

<A> = 自動付番シーケンスの開始に使用する数値

例 コンテキストを示すことによってコマンドを最も明確に説明できる場合は、ZPL コードの例が提供されます。パラメータ、入力する正確なコード、ホストから返されたデータなどのテキストが見分けやすい Courier フォントで印刷されます。

例 • AUTONUM コードの例：

```
AUTONUM 10,5
10 PRINT "HELLO WORLD"
15 GOTO 10
```

例では、> 記号は入力するコードのラインを示します。

コメント この項は、プログラマにとって貴重な注記、コマンド相互作用に関する警報、考慮の必要があるコマンド固有の情報などのために予約されています。コメントの例：「これはプログラム・コマンドであるため、ライン番号が前に示される必要があります。」

AND

説明 AND ステートメントは論理演算子です。両方の式が真である場合に結果は真となり、それ以外の場合は偽となります。

フォーマット < 論理式 > AND < 論理式 >

コメント これは論理式と一緒に使用する論理演算子です。

Arrays

説明 配列は、プログラムで使用する値の集合です。配列には次の特徴があります。

- 配列には、整数変数とストリング変数の両方を使用できます。
- 配列インデックスには括弧を使ってアクセスできます。
- 配列インデックスは、1 で始まり、配列の長さで終わります（たとえば、変数 3 は変数配列の 3 番目の位置にある値を返します）。
- 1 次元と 2 次元の配列を使用できます。2 次元配列は、括弧内のカンマで区切った 2 つのインデックスで参照されます。
- 配列は、元の配列を破壊する DECLARE を呼び出すことによってのみ次元変更することができます。
- 配列サイズは、割り当てられたメモリ・ヒープによってのみ制限されます。
- 配列を割り当てることができない場合、エラー・メッセージ（Error: Heap overflow.）が表示されます。
- 制限範囲を超える配列にアクセスしようとすると、エラーメッセージ（Error: Invalid array access）が表示されます。.

AUTONUM

説明 このコマンドは、連番のプログラム・ライン番号を自動生成します。この機能は、現在のライン番号を上書きして希望の対話モード・コマンドを入力するか、ラインを空白にすることによって無効にできます。

フォーマット AUTONUM A,B

パラメータ：

A = 自動付番シーケンスの開始に使用する数値

B = 新しいライン番号間の自動増分

例 • 次の例は、開始ライン番号と新しい行間の増分を指定した例です。

```
AUTONUM 10,5
10 PRINT "HELLO WORLD"
15 GOTO 10
```

コメント 2 つのラインは AUTONUM のパラメータによって自動的に開始されます。ここでは、最初のラインは 10 で開始され、後続のラインはそれぞれ 5 ずつ増分されていきます。

これは対話式コマンドで、プリンタで受信されると同時に実行されます。

Boolean Expression

説明 論理式では 0（ゼロ）が偽、ゼロ以外が真とされます。

フォーマット < ストリング式 > < 論理比較 > < ストリング式 >
<# 式 > < 論理比較 > <# 式 >
(< 論理式 >)

コメント # は数式を意味します。数式は、ストリング式とは比較できません。比較を試みるとエラー・メッセージ（Error: Poorly formed expression.）が表示されます。

数式は、値 0（ゼロ）が偽をゼロ以外が真を表す論理式の代わりに使用できます。

<、<=、>、>=、= の優先順位は、すべて同じで、その後に NOT、AND、OR がこの順序で続きます。同じ優先順位のアイテムは左から右へと処理されます。

BREAK

説明 このコマンドは、DEBUG 機能がアクティブな場合にのみ使用できます。DEBUG がオンの場合に BREAK によって処理が停止されます。RUN はプログラムを初めから開始します。RESTART を使用すると、停止した所からプログラムを再開できます。

フォーマット BREAK

コメント これはライン番号が前に示されているプログラム・コマンドです。

Channels

説明 I/O コマンドはチャネルに対して発行し、コマンドを特定のポートに適用することができます。データ入力ポートは、ZBI インタープリタを開始する ZPL コマンドによって指定されます。すべての対話式通信はこのポートを介して行われます。

フォーマット #< チャネル式 >

パラメータ：

#< チャネル式 >

*有効値：*0 ～ 9

デフォルト値：0

CLOSE

説明 このコマンドは、使用中の特定のポートを閉じるために実装されています。チャネル上でポートが開いている場合、CLOSE コマンドを入力するとポートは閉じられ、ZPL バッファとの通信に戻ります。

フォーマット CLOSE < チャネル式 >

パラメータ：

< チャネル式 >

*有効値：*0 ～ 9

*デフォルト値：*使用されません。

例 • 次はチャネル 1 を閉じる例です。

```
>CLOSE #1
```

コメント これは、プリンタで受信されると同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

CLRERR

説明 このコマンドは、エラー・フラグをクリアするメッセージをプリンタに送信します。エラーが永久的なものでなければ、エラー・フラグはリセットされます。

フォーマット `10 CLRERR`

コメント これはライン番号が前に示されているプログラム・コマンドです。

CTRL-C

説明 03 をポート 0 に送信するか、Ctrl キーを押しながら C を押すと、現在実行中の ZBI プログラムがすべて終了されます。

DEBUG

説明 `DEBUG` がオンの場合、`TRACE` と `BREAK` オプションが有効になります。`DEBUG` がオフの場合、`TRACE` と `BREAK` オプションは無効になります。

フォーマット `DEBUG <ON/OFF>`

パラメータ：

`<ON/OFF>` = デバッグ・モードのオン / オフを切り替える

コメント これは、プリンタで受信されると同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

Declaration

説明 宣言に共通する特徴は次のとおりです。

- 変数の名前は、通常 ZBI プログラムの冒頭で設定されます。宣言すると整数変数は 0 の初期値に設定されます。宣言するとストリング変数は空のストリング（“ ”）を初期値として設定されます。
- 前に宣言された変数は、明示的宣言によって削除され再割り当てされます。
- 黙示的な宣言では、前に定義されていない変数名が使用されます。この変数の値は定義されていません。
- 非配列ストリング変数と整数変数は、黙示的にも明示的にも宣言できます。
- 配列は明示的に定義する必要があります。

DECLARE

説明 これは変数を宣言する明示的方法で、通常はプログラムの冒頭で実行されます。配列は、割り当てる配列サイズの数式の前後に括弧を挿入して指定します。

フォーマット DECLARE < タイプ > < 変数名 > [,< 変数名 >]*

パラメータ：

< データ型 > = 数値型またはストリング型

< 変数名 > = 宣言する変数（配列である可能性もあります）

例 • 次は 変数を宣言する例です。

```
10 DECLARE NUMERIC A !Declare an integer
20 DECLARE STRING A$ !Declare a string
30 DECLARE NUMERIC My_Array(10) !Declare an integer
array of size 10
40 DECLARE STRING My_Str_Array$(10) !Declare a string
array with 10 strings
50 DECLARE NUMERIC A, B(10), C !Declare multiple integer
variables
60 DECLARE STRING A$, B$, C$ !Declare multiple string
variables
70 DECLARE STRING A2D$(5,5) !Declare A2-D array
```

コメント これは、プリンタで受信されると同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

DELETE

説明 このコマンドは、プリンタのメモリから指定のファイルを削除します。

フォーマット DELETE <" ファイル名 ">

パラメータ <" ファイル名 "> = 削除するファイルの名前ドライブの場所とファイル名は引用符で囲む必要があります。

例 • 次はプリンタ・メモリから指定のファイルを削除する例です。

```
>DELETE "E:PROGRAM1.BAS"
```

コメント これは対話式コマンドで、プリンタで受信されると同時に実行されます。

DIR

説明 このコマンドをフィルタを含まずに使用すると、プリンタ内のすべてのメモリの場所に常駐するすべての ZBI プログラムがリストされます。

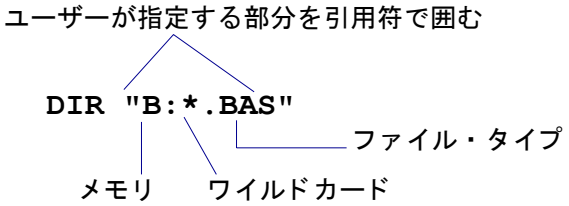
フィルタを含めると、プリンタで検索が限定されます。ドライブの場所を含めると、プリンタではその場所のみが検索されます。

アステリスク (*) はワイルドカードとして使用されます。ワイルとカード (*) を使用すると、特定の要求すべての事例を検出できます。たとえば、DIR "B:*.BAS" では、B: のメモリで .BAS の拡張子を持つすべてのファイルが検索されます。

フォーマット DIR ["フィルタ"]

パラメータ ["フィルタ"] = アクセスするファイルの名前（オプション）ドライブの場所とファイル名は引用符で囲む必要があります。

重要 • ユーザーが指定する部分は引用符で囲む必要があります。以下は、ワイルドカード（*）を使って、B: のメモリにあるすべての .BAS ファイルを検索する方法を示しています。



コメント これは対話式コマンドで、プリンタで受信されると同時に実行されます。

DO-LOOP

説明 ループの処理は DO または LOOP ラインにある <WHILE/UNTIL> 式によって制御されます。

WHILE ステートメントの処理は、DO ラインでも LOOP ラインでも同じです。論理式が評価され、ステートメントが真の場合は、DO ステートメントの後のラインで LOOP が続行されます。それ以外の場合は、対応する LOOP の後のラインが次に処理されるラインです。

UNTIL ステートメントの処理は、DO ラインでも LOOP ラインでも同じです。論理式が評価され、ステートメントが偽の場合は、DO ステートメントの後のラインで LOOP が続行されます。それ以外の場合は、対応する LOOP の後のラインが次に処理されるラインです。

<WHILE/UNTIL> が LOOP ラインにある場合、論理式の評価の前にループの BODY が実行されます。

DO ラインにも LOOP ラインにも <WHILE/UNTIL> ステートメントがない場合は、ループは無期限に続行されます。

DO-LOOP はネストできますが、重なることはできません。DO-LOOP コマンドには次の 2 つのフォーマットがあります。

フォーマット 1

```
DO <WHILE/UNTIL> < 論理式 >
~~BODY~~
LOOP
```

形式例 1 • 次は、DO-LOOP コマンドを使用する例です。

```
10 DO WHILE A$="70"
20 INPUT A$
30 LOOP
```

フォーマット 2

```
DO
~~BODY~~
LOOP <WHILE/UNTIL> < 論理式 >
```

形式例 2 • 次は、DO UNTIL LOOP コマンドを使用する例です。

```
10 DO
20 INPUT A$
30 LOOP UNTIL A$="EXIT"
```

コメント これはライン番号が前に示されているプログラム・コマンドです。

ECHO

説明 このコマンドは、コンソール・モードが有効になっている場合に、プリンタが文字を通信ポートにエコー・バックするかどうかを決定します。ECHO ON を入力すると、キー操作の結果が画面に返されます。ECHO OFF を入力すると、キー操作の結果は画面に返されません。

フォーマット ECHO <ON/OFF>

パラメータ <ON/OFF> = ECHO コマンドのオン / オフを切り替える

コメント これは、プリンタで受信されると同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

END

説明 END コマンドは、現在実行されているプログラムをすべて終了します。END コマンドを受信すると、インタープリタは対話モードを返します。

フォーマット END

例 • 次は、END コマンドを使用する例です。

```
10 PRINT "THIS PROGRAM WILL TERMINATE"
20 PRINT "WHEN THE END COMMAND IS RECEIVED"
30 END
```

コメント これはライン番号が前に示されているプログラム・コマンドです。

! (EXCLAMATION MARK)

説明 感嘆符は、番号が付けられたプログラミング・ラインの最後にコメントを追加するためのマーカーです。ラインまたはコマンドが処理されると！の後にあるテキストはすべて無視されます。

フォーマット ！[コメント・テキスト]

例 • 次は、！（コメント）コマンドを使用する例です。

```
10 LET A=10 !Indicates number of labels to
print
```

EXIT

説明 このコマンドは、DO ループおよび FOR ループを終了するために使用します。

フォーマット EXIT <DO/FOR>

コメント これはライン番号が前に示されているプログラム・コマンドです。

FOR-LOOP

説明 FOR ループは最初の式の値に数値変数を割り当てます。NEXT ラインは、その数値変数を STEP の値で増分します。

数値変数が NEXT ラインで増分後 2 番目の式よりも少ない場合、プログラムは FOR ラインの後のラインで実行を継続します。

STEP 部分を省略すると、STEP の値は 1 になります。ただし、2 番目の式が最初の式よりも大きい場合は、デフォルトの ñ1 が使用されます。

フォーマット

```
FOR <# 変数> = <# 式> TO <# 式>
STEP <# 式>]
~~BODY~~
NEXT <# 変数>
```

パラメータ

<# 変数> = 数値変数が使用されることを意味します。

<# 式> = 数式が使用されることを意味します。

例 • 次は、FOR LOOP コマンドを使用する例です。

```
10 FOR X=1 TO 10 STEP 1
20 PRINT X; ":ZBI IS FUN"
30 NEXT X
```

コメント FOR LOOP はネストする必要がありますが、重なることはできません。変数はネストしたループで再使用できません。これはライン番号が前に示されているプログラム・コマンドです。

Functions

説明 このインタープリタに組み込まれた関数は、式でのみ使用できます。関数名では大文字と小文字を区別しません。

入力パラメータが存在する場合、括弧で囲みます。入力パラメータが存在しない場合は、括弧は使用しません。

関数で参照する変数の代わりに、同じタイプの関数または式を使用できます。関数名が $ で終わる場合、ストリング型の値が返されます。それ以外の場合は、数値型の値が返されます。

間違ったタイプのパラメータを指定すると、Syntax Error（構文エラー）または Poorly formed expression error（無効な式のエラー）が返されます。

Integer Functions

以下にリストした整数関数は、数値型の値を返します。

DATE

この関数は現在の日付を YYYYDDD の形式で返します。ここで、YYY は年、DDD は年の初めから数えた日数です。リアルタイム・クロックがインストールされていない場合は、0 が返されます。

例 • 次の例は、現在の日付が 2003 年 1 月 1 日であると仮定しています。

```
10 PRINT DATE
RUN
```

結果は次のとおりです。

```
2003001
```

DATAREADY(A)

この関数は、ポート A でデータが準備完了している場合は 数字の 1 を、データがない場合は 0 を返します。

例 • 次は、ポートにデータがあるかどうかを確認する例です。

```
10 PRINT DATAREADY(0)

RUN
```

待機中のデータがない場合、結果は次のとおりです。

```
   0
```

LEN(A$)

この関数は、ストリング A$ の長さを返します。

例 • 次の例では、ストリングの長さが文字数で示されています。Hello World は 10 文字です。

```
10 LET A$="Hello World"

20 PRINT LEN(A$)

RUN
```

結果は次のとおりです。

```
      11
```

MAX(X,Y)

この関数は、X または Y の最大代数値を返します。X が Y よりも大きい場合は X の値が返されます。X が Y よりも小さい場合は Y が返されます。

例 • 次は、MAX(X,Y) コマンドを使用する例です。

```
10 LET A=-2
20 LET B=1
30 PRINT MAX(A,B)
RUN
```

結果は次のとおりです。

```
1
```

MAXLEN(V$)

この関数は、ストリング V$ の最大長を返します。これは常に 255 です。この値は、V$ には左右されませんが、ANSI 規格との準拠のために必要です。

例 • 次は、MAXLEN(V$) コマンドを使用する例です。

```
10 LET A$="Hello"
20 PRINT MAXLEN(A$)
RUN
```

結果は次のとおりです。

```
255
```

MAXNUM

この関数は、このマシンで表すことのできる最大値 2,147,483,647 を返します。

例 • 次は、MAXNUM コマンドを使用する例です。

```
10 PRINT MAXNUM
RUN
```

結果は次のとおりです。

```
2147483647
```

MIN(X,Y)

この関数は、X または Y の最小代数値を返します。X が Y よりも小さい場合は X の値が返されます。X が Y よりも大きい場合は Y が返されます。

例 • 次は、MIN(X,Y) コマンドを使用する例です。

```
10 LET A=-2
20 LET B=0
30 PRINT MIN(A,B)
RUN
```

結果は次のとおりです。

```
-2
```

MOD(X,Y)

この関数は X モジュロ Y（X/Y の残りと同じ）を返します。

例 • 次は、MOD(X,Y) コマンドを使用する例です。

```
10 LET A=9
20 LET B=2
30 LET C=-2
40 PRINT MOD(A,B)
50 PRINT MOD(C,A)
RUN
```

結果は次のとおりです。

```
1
-2
```

ORD(A$)

この関数は、ストリング A$ の最初の文字の ASCII 値を返します。

例 • 次は、ORD(A$) コマンドを使用する例です。

```
10 LET A$="ABC"
20 PRINT ORD(A$)
RUN
```

結果は次のとおりです。

```
65
POS(A$,B$)
```

この関数は、A$ で B$ が最初に発生する場所を返します。発生しない場合、結果は 0 です。

例 • 次は、ORD(A$) コマンドを使用する例です。

```
10 LET A$="ABCDD"
20 LET B$="D"
30 PRINT POS(A$,B$)
RUN
```

結果は次のとおりです。

```
4
```

POS(A$,B$,M)

この関数は、A$ の M の位置から開始して B$ が最初に発生する場所を返します。発生しない場合、結果は 0 です。

例 • 次は、 POS(A$,B$,M) コマンドを使用する例です。

```
10 LET A$="Hello World"
20 LET B$="o"
30 PRINT POS(A$,B$,6)
RUN
```

結果は次のとおりです。

```
8
```

TIME

この関数は、零時（2400h）を過ぎた時間を秒数で返します。リアルタイム・クロックがインストールされていない場合は、0 が返されます。

例 • 次は、TIME コマンドを使用する例です。

```
10 PRINT TIME
RUN
```

時刻が零時 1 分だとすると、結果は次のとおりです。

```
60
```

VAL(A$)

この関数は、A$ で表す数値型の値を返します。変換は、INPUT コマンドと同じ方法で行われます。

例 • 次は、VAL(A$) コマンドを使用する例です。

```
10 LET A$="123"
20 LET C=VAL(A$)
30 PRINT C
RUN
```

結果は次のとおりです。

```
123
```

String Functions

以下にリストしたストリング関数は、ストリング型の値を返します。

CHR$(M)

この関数は、拡張 ASCII 表の M の位置の文字を返します。M の値として 255 よりも大きい値を使用すると、255 に調整されます。M を 0 にすることはできません。数字の 1 で置換されます。

例 • 次は、CHR$(M) コマンドを使用する例です。

```
10 LET A=97
20 PRINT CHR$(A)
30 PRINT CHR$(353) !Note that 353 is
modulated to 97
RUN
```

結果は次のとおりです。

```
a
a
```

DATE$

この関数は、現在の日付をストリング形式 YYYYMMDD で返します。リアルタイム・クロックがインストールされていない場合は、データは返されません。

例 • 次は、DATE$ コマンドを使用する例です。

```
10 PRINT DATE$
RUN
```

日付が 2003 年 1 月 1 日だと仮定すると、結果は次のとおりです。

```
20030101
```

EXTRACT$

この関数はストリングを返します。

- **抽出元**は、ポートまたは変数ストリング名です。
- **開始ストリング**は、最初に出現してコマンドをトリガするストリングまたは文字です。
- **終了ストリング**は、最後に出現してコマンドを終了するストリングまたは文字です。

重要 • EXTRACT$ コマンドが、開始文字または終了文字に遭遇する前に、キャリッジ・リターン / ライン・フィードに遭遇すると、null が返されます。

EXTRACT$ コマンドは、2 つの既知のポイント間でデータを取得するために使用します。たとえば、カンマ区切りファイルのデータでは、レコード間に必ずカンマがあります。

例 • 次の例は、Zebra, Technologies, Corporation というストリングから Technology という単語を抽出する方法を示しています。

プログラムは次のようになります。

```
10 LET A$ = Zebra,Technologies,Corporation,
20 LET DATA$ = EXTRACT$(A$,",",",",",")
```

例 • 次の例は、開いたポートから EXTRACT$ コマンドがどのように機能するかを示しています。

```
10 OPEN #1:NAME "SER"
20 LET DATA$ = EXTRACT$(1,",",",",",")
```

リテラル文字（この場合はカンマ）には引用符が使用されていることに注意してください。

例 • 次の例は、スタート / ストップ・ポイントが変数であること、リテラルの代わりに変数名が使用されていることを示しています。

```
10 LET B$ = ",”
20 LET A$ = Zebra,Technologies,Corporation
30 LET DATA$ = EXTRACT$(A$,B$,B$)
```

LTRIM$(A$)

この関数は、すべての先行スペースを削除したストリング A$ を返します。

例 • 次は、LTRIM$(A$) コマンドを使用する例です。

```
10 LET A$=" Hello"
20 PRINT LTRIM$(A$)
RUN
```

結果は次のとおりです。

```
Hello
```

REPEAT$(A$,M)

この関数は、A$ の M 個のコピーを含んだストリングを返します。

例 • 次は、REPEATS$(A$,M) コマンドを使用する例です。

```
10 LET X=3
20 LET A$="Hello"
30 PRINT REPEAT$(A$,X)
RUN
```

結果は次のとおりです。

```
HelloHelloHello
```

RTRIM$(A$)

この関数は、後続スペースを削除したストリング A$ を返します。

例 • 次は、RTRIM$(A$) コマンドを使用する例です。

```
10 LET A$="Hello "
20 LET B$="World"
30 PRINT RTRIM$(A$);
40 PRINT B$
RUN
```

結果は次のとおりです。

```
HelloWorld
```

STR$(X)

この関数は数値型の X をストリング型に変換します。

例 • 次は、STR$(X) コマンドを使用する例です。

```
10 LET A=53
20 PRINT STR$(A)
RUN
```

結果は次のとおりです。

```
53
```

TIME$

この関数は、HH:MM:SS（時間 : 分 : 秒）の形式で時刻を返します。リアルタイム・クロックがインストールされていない場合は、データは返されません。

例 • 次は、TIME$ コマンドを使用する例です。

```
10 PRINT TIME$
RUN
```

10 a.m の場合、結果は次のとおりです。

```
10:00:00
```

UCASE$(A$)

この関数は、A$ から作成されたストリングをすべて大文字で返します。文字以外の値は変更されません。

例 • 次は、UCASE$(A$) コマンドを使用する例です。

```
10 LET A$="Zebra Technologies"
20 PRINT UCASE$(A$)
RUN
```

結果は次のとおりです。

```
ZEBRA TECHNOLOGIES
```

GOTO

説明 GOTO ステートメントは、インタープリタを特定のライン番号に移動させるために使用します。GOTO の後には、プログラムで次に処理するライン番号が示されます。GOTO ステートメントを実行すると、インタープリタは、GOTO の後に指定されたライン番号から実行を継続します。参照先のライン番号がない場合はエラー・メッセージ（Error: Line does not exist）が表示されます。

フォーマット GOTO

例 • 次は、GOTO コマンドを使用する例です。

```
10 PRINT "Zebra Printers"
20 GOTO 10
```

コメント プログラムが停止するまで、結果として Zebra Printers が表示されます。これはプログラム・コマンドであるため、ライン番号が前に示される必要があります。

GOSUB-RETURN

説明 GOSUB の後には、プログラムで次に処理するライン番号が示されます。GOSUB ステートメントを実行すると、インタープリタは、GOSUB の後に指定されたライン番号から実行を継続します。参照先のライン番号がない場合はエラー・メッセージ（Error: Line does not exist）が表示されます。

次のラインを実行する前に、GOSUB コマンドは GOSUB ラインのライン番号を保存します。RETURN ステートメントを呼び出すと、プログラムは GOSUB の後の次のラインに戻ります。

対応する GOSUB ステートメントなしで RETURN ステートメントを実行すると、エラー・メッセージ（Error: Invalid RETURN statement.）が表示されます。

GOSUB ステートメントはネストできます。

フォーマット

```
GOSUB
RETURN
```

例 • 次は、GOSUB-RETURN コマンドを使用する例です。

```
10 PRINT "Call Subroutine"
20 GOSUB 1000
30 PRINT "Returned from Subroutine"
40 END
1000 PRINT "In Subroutine"
1010 RETURN
```

コメント これらはプログラム・コマンドであるため、ライン番号が前に示される必要があります。

IF Statements

説明 IF ステートメントの < 論理式 > の値が真で、キーワード THEN の後にプログラム・ラインがある場合、そのプログラム・ラインが実行されます。論理式の値が偽で、キーワード ELSE の後にプログラム・ラインがある場合、そのプログラム・ラインが実行されます。ELSE がない場合、実行は END IF ステートメントに続くラインで、順に続行されます。

ブロックのネストは可能ですが、DO-LOOP と同じネスト制約（ブロックが重なってはならない）が適用されます。

ELSE IF ステートメントは、後ろに IF ラインが続く ELSE ラインとして処理されますが、IF は、元の IF ステートメントの END IF ラインを共有します。

フォーマット

```
IF < 論理式 > THEN
~~BODY~~
[ELSE IF < 論理式 > THEN
~~BODY~~]*
[ELSE
~~BODY~~]
END IF
```

例 • 次は、IF コマンドを使用する例です。

```
10 IF A$="0" THEN
20 PRINT "ZBI IS FUN"
30 ELSE IF A$="1" THEN
40 PRINT "ZBI IS EASY"
50 ELSE
60 PRINT "X=0"
70 END IF
```

コメント これはライン番号が前に示されているプログラム・コマンドです。

INBYTE

説明 このコマンドは、データが利用できるようになるまでインタープリタを強制的に一時停止します。ポートにデータがあるかどうかを判断するには、DATAREADY 機能を使用してください。

フォーマット INBYTE [チャネル式 :]<A>

パラメータ <A> = 数式またはストリング式受け取った値は、変数に使用されている現在の値を置換します。

例 • 次は、INBYTE ステートメント・コマンドを使用する例です。

```
10 INBYTE A$ !takes one byte (char) from
port #1

20 PRINT A$ !prints the character to the
console
```

コメント この例では、データが入力されるまでインタープリタは一時停止し、その後処理を続けます。このコマンドは、コントロール・コードを含めストリングまたは整数のすべてのバイトを入力します。

これは、プリンタで受信されると同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

INPUT

説明 変数が数値型で、入力した値を数値に変換できない場合、それは 0 として書き込まれます。この操作は、データを左から右へスキャンし、すべての数値を変数にシフトします。入力を終了する返却文字、またはプログラムを終了する Ctrl-C（^C）以外は無視されます。この変数はストリング型の場合も数値型の場合もあります。

フォーマット

```
INPUT [<チャネル式>:]<変数>
[,変数]*
```

[<チャネル式>:] を省略すると、デフォルト・ポートは 0 になります。無効なポートを指定すると、Error: Invalid port （エラー : 無効なポート）という形でエラー状態が返されます。

例 1 • 次は、INPUT コマンドを使用する例です。

```
10 INPUT A

20 PRINT A
```

1234567891011 を入力した場合、ディスプレイに送られる数値は MAXNUM 値で調整されます。

例 2 • 次は、INPUT コマンドを使用する例です。

```
10 OPEN #1:NAME "ZPL"
20 PRINT #1:"~HS"
30 FOR I = 1 TO 3
40 INPUT #1:A$
50 PRINT A$
60 NEXT I
```

例 2 では、ZPL ポートにホスト・ステータス要求 ~HS を送信した後、ホスト・ステータスがコンソールに出力されます。

ZBI インタープリタの INPUT/OUTPUT コマンドは、通信ポートに限定されます。ファイル I/O はサポートされていません。

コメント これは、プリンタで受信と同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

LET

説明 LET コマンドは、特定の変数に値を割り当てるために使用します。式は評価され、変数リストの書く変数に割り当てられます。

フォーマット LET <変数> [,<変数>]* = <式>

変数タイプは式のタイプに一致する必要があり、一致しない場合はエラー・メッセージ（Error: Variable types must be the same.）が表示されます。

値がサブストリング修飾子のあるストリング変数に割り当てられると、サブストリング修飾子はその値によって置換されます。ストリング変数の値の長さは、この置換の結果変更される可能性があります。

例 次は、LET コマンドを使用する例です。

```
10 LET A$= "1234"
15 LET A$(2:3)= "55" !A$ NOW = 1554
20 LET A$(2:3)= "" !A$ NOW = 14

10 LET A$= "1234"
15 LET A$(2:3)= A$(1:2) !A$ NOW = 1124

10 LET A$= "1234"
20 LET A$(2:1)= "5" !A$ NOW = 15234
```

コメント これは、プリンタで受信と同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

LIST

説明 このコマンドは、現在メモリにあるプログラム・ラインを番号順にリストします。

フォーマット LIST [RANGE]

パラメータ

[RANGE] = コードの特定ライン（1 つまたは複数）の要求（オプション）[RANGE] はハイフンで区切られた 1 対の数値です。

例 次は、LIST コマンドを使用する例です。

```
LIST 20
LIST 90-135
```

プリンタはライン 20 を返します。また、2 番目の例ではライン 90 ～ 135 を返します。

コメント これは対話式コマンドで、プリンタで受信されると同時に実行されます。

LOAD

説明 このコマンドは、前にプリンタのメモリに保存されたプログラム・ファイルを転送し、ZBI プログラム・メモリで開きます。

プログラム・ファイルが存在しない場合、ZBI プログラム・メモリはクリアされ、プログラムは開かれません。エラー・メッセージ（Error: Invalid file name.）が表示されます。

フォーマット LOAD <" ファイル名 ">

パラメータ <" ファイル名 "> = メモリに読み込むファイルの名前ドライブの場所とファイル名は引用符で囲む必要があります。

例 • 次は、LOAD コマンドを使用する例です。

```
LOAD "PROGRAM1.BAS"
LOAD "E:PROGRAM1.BAS"
```

コメント これは対話式コマンドで、プリンタで受信されると同時に実行されます。

NEW

説明 このコマンドは、ライン・バッファと変数を含むインタープリタのメモリをクリアします。開いたポートは含まれません。

フォーマット NEW

例 • 次は、NEW コマンドを使用する例です。

```
NEW
```

コメント これは対話式コマンドで、プリンタで受信されると同時に実行されます。

NOT

説明 NOT ステートメントは論理演算子です。論理式が真の場合、結果は偽です。論理式が偽の場合、結果は真です。

フォーマット NOT < 論理式 >

例 • 次は、NOT コマンドを使用する例です。

```
10 LET A=1
20 IF NOT A=0 THEN
30 PRINT A
40 END IF
RUN
```

コメント これは論理式と一緒に使用する論理演算子です。

Numeric Expressions

説明：基本数式は定数、 変数、 または括弧に入った別の数式のいずれかである可能性があります。 オーバーフローは検出できません。 結果は定義されていません。使用される 5 つのタイプ (加算、減算、乗算、除算、累乗) を下にリストします。

1 + (加算) 加算式は次のフォーマットを使用します。

< 数式 >+< 数式 >

2 - (減算) 減算式は次のフォーマットを使用します。

< 数式 >-< 数式 >

3 * (乗算) 乗算式は次のフォーマットを使用します。

< 数式 >*< 数式 >

4 / (除算) 除算式は次のフォーマットを使用します。

< 数式 >/< 数式 >

5 ^ (累乗) 累乗式は次のフォーマットを使用します。

< 数式 >^< 数式 >

数学では、優先順序によって式を処理する順序が決定されます。すべての式には、事前に定義された優先順序があります。

優先順序は、^、* /、+ - です。

* と /、および + と - はそれぞれ同じ優先順位です。同じ優先順位のアイテムは左から右へと処理されます。

たとえば、5+(8+2)/5 の式は 8+2、10/5=2、そして 5+2 のように処理され、結果は 7 になります。

関数と括弧は常に優先順序が最高で、それらが最初に処理されることを意味します。残りのアイテムは、式のタイプ（論理式、数式、ストリング式など）によって異なります。

最大値 : 2,147,483,647

最小値 : -2,147,483,648

コメント

- 浮動小数点はサポートされません。
- 変数名は文字で始まる必要があり、一連の文字、数字、および下線を含めることができます。
- 関数名とコマンド名は変数名として使用できません。
- 変数名では大文字と小文字は区別されませんが、インタープリタによって大文字に変換されます。
- 除算を使用する場合、数値は常に切り捨てられます。たとえば、5/2=2 です。

ON ERROR

説明 ON ERROR コマンドを使用すると、エラー発生時にプログラムが停止するのを防ぐことができます。プログラム実行中、前のラインでエラーが発生すると、ON ERROR ステートメントによって GOTO または GOSUB ステートメントが呼び出され、プログラムが続行できます。

フォーマット ON ERROR <GOTO/GOSUB> LN

エラーがない場合は、このラインは無視されます。

例 • 次は、ON ERROR コマンドを使用する例です。

```
30 LET A = B/C
40 ON ERROR GOTO 100
...
100 PRINT "DIVIDE BY ZERO OCCURRED"
110 LET A=0
120 GOTO 50
...
```

コメント これはライン番号が前に示されているプログラム・コマンドです。

Open

説明 このコマンドは、データの送受信のためにポートを開く場合に使用します。

フォーマット

```
OPEN #<チャネル式>:NAME <ストリング式> [, ACCESS
<ACCESS type>]
```

パラメータ

<チャネル式> =

有効値 : 0 ～ 9

デフォルト値 : ポートを指定する必要があります。

<ストリング式> = 開くポート名（SER、PAR、または ZPL）

<ACCESS type> =

受信専用は INPUT

送信専用は OUTPUT

送受信用は OUTIN

チャネルの指定が必要です。デフォルト値は使用できません。アクセス・タイプを指定しないと、OUTIN が使用されます。

例 • 次は、OPEN コマンドを使用する例です。

```
10 OPEN #1:NAME"ZPL"
```

ポートを開く再に競合があると、エラー・メッセージ（Error: Unable to open port.）が表示されます。ポートが既に開いている場合は、エラー・メッセージ（Error: Port already opened.）が表示されます。

開いているポートでは、データは直接バッファに入ることができなくなります。接続が切断され、インタープリタがデータ・フローを制御するようになります。

すでにバッファにあるデータはバッファに留まります。

コメント これは、プリンタで受信されると同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

OR

説明 OR ステートメントは論理演算子です。どちらかの式が真である場合に結果は真となり、それ以外の場合は偽となります。

フォーマット

< 論理式 > OR < 論理式 >

コメント これは論理式と一緒に使用する論理演算子です。

OUTBYTE

説明 このコマンドは、コントロール・コードを含めストリングのすべてのバイトを出力します。

フォーマット OUTBYTE [< チャネル式 >:]<A>

パラメータ <A> = 数式またはストリング式

パラメータが数式の場合、0 ～ 255 の値である必要があります。そうでない場合は、切り捨てられます。ストリングの場合、最初の文字が使用されます。NULL ストリングの場合は 0 が送信されます。

例 • 次は、OUTBYTE コマンドを使用する例です。

```
Let A$="Hello"
OUTBYTE A$
```

結果は次のとおりです。

```
H
```

コメント これは、プリンタで受信されると同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

Ports <CHANNEL EXPRESSION>

説明 各プリンタには、インタープリタで開くことのできるポートのリストがあります。

各ポートには、そのポートを開けるために OPEN コマンドと一緒に使用する 3 文字の略語が与えられています。標準のポート名は、シリアル・ポートの場合は SER、パラレル・ポートの場合は PAR です。ポート参照 ZPL によって、プリンタのフォーマット・エンジンに対してチャネルが開かれます。このチャネルは、ZPI インタープリタによって制御されないポートとして機能します。

PRINT

説明 このコマンドは印刷するデータをプリンタに送信します。

フォーマット PRINT [チャネル式 :]< 式 > [,or; < 式 >]* [;]

式はストリング式の場合も数式の場合もあります。

,（カンマ）を使用して式を区切ると、式と式の間にスペースが挿入されます。

;（セミコロン）を使用して式を区切ると、式と式の間にスペースは挿入されません。

;（セミコロン）をラインの終わりに使用すると、新しいラインなしで PRINT ステートメントが終了されます。

例 • 次は、PRINT コマンドを使用する例です。

```
10 LET A$= "This is an example"
20 LET B$= "of the PRINT Command."
30 PRINT A$, B$ ! adds a space between
expressions
40 PRINT A$; B$ ! no space added
RUN
```

結果は次のとおりです。

```
This is an example of the PRINT Command.
This is an exampleof the PRINT Command.
```

コメント これは、プリンタで受信されると同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。

REM

説明 番号の付いた**備考**ラインは、REM で始まり、その後に任意の形式の任意のテキストを含むことができます。このラインはインタープリタによって無視されます。

フォーマット REM [コメント]

例 • 次は、REM コマンドを使用する例です。

```
10 REM COMMAND LINES 20-100 PRINT A LABEL
```

コメント 備考は、プログラムの説明に使用でき、別個のプログラム・ラインとして含めたり、プログラム・ラインの後ろに追加したりすることができます。また、内部のコメントには感嘆符（!）ステートメントを使用することもできます。

RENUM

説明 このコマンドは、編集中のプログラムのラインに再付番します。RENUM は、ライン番号が等間隔でなくなった場合にコードを再編成できます。GOTO と GOSUB ステートメントの後のライン参照が定数の値である場合、それらは再付番されます。ライン番号が 1 ～ 10000 の範囲外の場合は再付番は行われません。

フォーマット RENUM <A>,<B>

パラメータ

<A> = 再付番シーケンスの開始に使用する数値

<B> = 新しいライン番号間の自動増分

例 • 次は、RENUM コマンドを使用する例です。

```
13 LET A=6
15 LET B=10
17 GOTO 13

>RENUM 10,5

0 LET A=6
15 LET B=10
20 GOTO 10
```

コメント これは対話式コマンドで、プリンタで受信されると同時に実行されます。

RESTART

説明 プログラムが Ctrl-C または BREAK コマンドによって停止された場合、RESTART コマンドを使用することにより、停止した場所からプログラムを再開することができます。RESTART は RUN と同様に機能しますが、最後に終了した場所から再開を試みるという点で異なります。また、このコマンドは STEP コマンドと一緒に機能し、STEP が終了した場所から実行を開始します。

フォーマット RESTART

コメント プログラムが実行されていないか終了していない場合は、RESTART は最初からプログラムを実行します。これは対話式コマンドで、プリンタで受信されると同時に実行されます。

RUN

説明 このコマンドを使用すると、編集中のプログラムをアクティブにし、最も下のライン番号から開始できます。ラインで次に処理するラインが指定されていない場合は、次の番号のラインが処理されます。それ以上のライン番号がなくなると、RUN コマンドは停止します。

フォーマット RUN

例 • 次は、RUN コマンドを使用する例です。

```
10 PRINT "ZBI"
15 PRINT "Programming"
RUN
```

結果は次のとおりです。

```
ZBI
Programming
```

コメント アプリケーションを起動したときに開いているポートは、アプリケーション終了後も開いたままになります。また、変数もアプリケーション終了後に残ります。

これは対話式コマンドで、プリンタで受信されると同時に実行されます。

SEARCHTO$ (A,B$)

説明 この関数は、ポート A で B$ によって定義されたストリングまで検索を実行します。検索結果のストリングが表示されます。

フォーマット SEARCHTO$ (A,B$)

パラメータ

A ＝要求されたデータの送信先のポート番号（0 ～ 9）

B$ ＝ストリング型変数またはストリング型配列。B$ が配列の場合、このコマンドは、B$ 配列内の null 値以外のすべてのストリングを検索します。

例 • 次は、SEARCHTO$ (A,B$) コマンドを使用する例です。

```
10 LET A$=SEARCHTO$(0,"Hello")
```

Hello World がこの例で検索されているテキスト・ストリングであることを表示する方法は複数あります。A$ は Hello で、プログラムは検索を実行すると、World のみがポートに残ります。

SEARCHTO$ (A,B$,C)

説明 この関数は、SEARCHTO$(A,B$) と同様に機能し、ポート A で B$ によって定義されたストリングまで検索を実行します。検索結果のストリングが表示されます。検索ストリングまでの未使用の文字はポート C に送られます。

フォーマット SEARCHTO$ (A,B$,C)

パラメータ

A ＝要求されたストリングの送信先のポート番号（0 ～ 9）

B$ ＝ストリング型変数またはストリング型配列。B$ が配列の場合、このコマンドは、B$ 配列内の null 値以外のすべてのストリングを検索します。

C ＝未使用の文字の送信先のポート番号（0 ～ 9）

例 • 次は、SEARCHTO$ (A,B$,C) コマンドを使用する例です。

```
LET B$ = "Hello"
10 LET A$ = SEARCHTO$(0,"Hello",1)
```

Hello World がこの例で検索されているテキスト・ストリングであることを表示する方法は複数あります。A$ は Hello で、プログラムは検索を実行すると、World のみがポートに残ります。未使用の文字はポート 1 に送られます。

例 • 変数の配列宣言した次の例も上の例と同様に機能しますが、使用可能な変数のリストが拡張されています。

```
DECLARE STRING B$(4)
LET B$(1) = "Hello"
LET B$(2) = "HELLO"
LET B$(3) = "hello"
LET B$(4) = "Hi"
10 LET A$ = SEARCHTO$(0,B$,0)
```

SETERR

説明 このコマンドは、エラー・フラグを設定するメッセージをプリンタに送信します。プリンタで論理インタープリタ・フラグがトリガされます。このエラーは、BASIC Forced Error（BASIC 強制エラー）として参照されます。

フォーマット SETERR

コメント これはプログラム・コマンドであるため、ライン番号が前に示される必要があります。

SLEEP

説明 このコマンドは、ラベル生成が優先されるようにインタープリタが一時停止する時間を指定します。このコマンドは、印刷するラベル・フォーマットの送信後にプリンタに送信できます。インタープリタは、指定の時間だけ処理を一時停止します。

フォーマット SLEEP <A>

パラメータ <A> = インタープリタが一時停止する秒数（0 ～ 500）

例 • 次は、SLEEP コマンドを使用する例です。

```
10 SLEEP 450
```

コメント これはプログラム・コマンドであるため、ライン番号が前に示される必要があります。

STEP

説明 プログラムが BREAK コマンドによって停止された場合、STEP は最後に終了したラインの次のラインからプログラムの実行を試みます。プログラムが実行されていないか終了していない場合、最も低い番号のラインから実行されます。

フォーマット STEP

例 • 次は、STEP コマンドを使用する例です。

```
10 PRINT "Hello World"
20 BREAK
30 PRINT "30"
```

STEP を入力するとライン 10 が実行されます。

STEP の前に RUN を入力すると次の結果になります。

RUN -- ライン 20 まで実行して停止します。

STEP -- ライン 30 まで実行して停止します。

コメント これは対話式コマンドで、プリンタで受信されると同時に実行されます。

STORE

説明 このコマンドは、現在メモリにあるプログラムを指定のファイル名で保存します。次のフォーマットを使用します。

```
STORE <" ファイル名 ">
```

パラメータ <" ファイル名 "> = 保存するファイルの名前ドライブの場所とファイル名は引用符で囲む必要があります。

例 • 次は、STORE コマンドを使用する例です。

```
STORE "E:ZEBRA1.BAS"
```

コメント ファイル名は 8.3 の規則（ファイル名 は 8 文字以下で拡張子は 3 文字以下）に従っている必要があります。ここでは、拡張子は常に .BAS です（たとえば MAXIMUM8.BAS）。

これは対話式コマンドで、プリンタで受信されると同時に実行されます。

STRING CONCATENATION (&)

説明 基本ストリング式は、定数の場合も変数の場合もあり、連結（&）もサポートされています。

連結演算子を使用すると、最初のストリングに 2 番目のストリングが追加されます。

フォーマット ＜ストリング式＞ & ＜ストリング式＞

例 • 次は、STRING CONCATENATION (&) コマンドを使用する例です。

```
10 LET A$= "ZBI "
20 LET B$= "Programming"
30 LET C$= A$ & B$
40 PRINT C$
```

結果は次のとおりです。

```
ZBI Programming
```

コメント 連結によってストリングが最大長（255 文字）を超える場合は、最初のストリングが返され、エラー・メッセージ（Error: String size limit exceeded.）が表示されます。

STRING VARIABLE

説明 最大長：255 文字

変数名は文字で始まる必要があり、一連の文字、数字、および下線を含めることができます。変数は $ で終わります。

関数名とコマンド名は変数名として使用できません。

変数名では大文字と小文字は区別されませんが、インタープリタによって大文字に変換されます。

UB-STRINGS

説明 ストリングでサブストリング演算子を使用すると、ストリングの特定部分にアクセスできます。使用するストリング部分の座標を決定するには、ストリングの先頭から終わりまでのスペースを含めた文字間隔を数えます。

フォーマット StringVariable$(A:B)

パラメータ

A＝希望のストリングの最初の文字の位置

B＝希望のストリングの最後の文字の位置

例• 次は、UB-STRINGS コマンドを使用する例です。

```
10 LET A$="Zebra Quality Printers"
20 LET B$=A$(1:13)
30 PRINT B$
```

結果は次のとおりです。

```
Zebra Quality
```

上のライン 10 の Zebra Quality Printers の位置を計算すると、Z が 1 の位置、e が 2 の位置、b が 3 の位置、r が 4 の位置、a が 5 の位置、スペースが 6 の位置というようになります。

コメント A パラメータが 1 より小さい場合、自動的に値 1 が割り当てられます。ストリングの計算は 1 から始まるため、A パラメータを 1 より小さい値に設定することはできません。

B がストリング長よりも大きい場合、その値に代わってストリング長が使用されます。この例では、B パラメータは 22 以下でなければなりません。

A が B より大きい場合、A の位置またはストリングの終わりをポイントする NULL ストリング（“ ”）が返されます。これは、ストリングの真中に文字を削除せずに別のストリングを追加する場合に使用します。LET コマンドを参照してください。

TRACE

説明 このコマンドは、DEBUG 機能がアクティブな場合にのみ有効です。

フォーマット TRACE <ON/OFF>

パラメータ

<ON/OFF> = TRACE が有効（オン）か無効（オフ）かを決定します。

DEBUG が有効で、TRACE コマンドがオンの場合、追跡詳細が表示されます。変数を変更すると、新しい値は次のように表示されます。

Variable$ = 新しい値

処理されるラインにはそれぞれライン番号が印刷されます。

DEBUG がオンの場合、コマンド・ラインが実行される前にライン番号が印刷され、変数は値が割り当てられると同時に印刷されます。

```
10 LET A=5
20 GOTO 40
30 PRINT "Error"
40 PRINT A
RUN
```

出力は次のとおりです。

```
<TRACE> 10
<TRACE> A=5
<TRACE> 20
<TRACE> 40
5
```

コメント これは、プリンタで受信されると同時に実行される対話式コマンドの場合も、ライン番号が前に示されたプログラム・コマンドの場合もあります。


Zebra

ZBI -- チュートリアル

このチュートリアルでは、バー・コード印刷で Zebra BASIC Interpreter（ZBI）と Zebra プログラミング言語 II（ZPL II）を併用する例を提供します。

インタープリタがご使用のプリンタでどのように機能するかを理解するため、これらの例を実際に実行してみることをお勧めします。

ZBI を使用すると、特定のニーズのために作成されたカスタム・プログラムを使って、バー・コード印刷を最大化できます。

幸い ZBI はその他の BASIC インタープリタの多くと非常によく似ており、ZBI の機能を利用するために 1 から新しい言語を学ぶ必要はありません。

ZBI の開始

ZBI は、通信ポート（シリアル、パラレル、Ethernet）の 1 つを介して対話モードで ZPL II コマンドをプリンタに送信することにより有効になります。この通信ポートをコンソールと呼びます。プリンタと通信するには、Hyper Terminal for Windows などのプログラムを使用します。

ZPL II コマンドの ~JI と ^JI は ZBI Interpreter を開始するために使用できます。プリンタに電源が入っている場合、ZPL II コマンドとラベル・フォーマットを受信します。ただし、プリンタで ZBI コマンドを受信するためには、^JI または ~JI を使ってそれが初期化されている必要があります。

コンソール・アプリケーションが開いた状態で、次のコマンドの 1 つをプリンタに送ります。

1 **~JI** -- このコマンドを受信すると、プリンタはプログラム・バージョン、および入力プロンプト（>）を含んだ新しいラインとともに ZBI ヘッダーを画面に送り返します。これは ZBI プログラム・ラインまたはコマンドをプリンタに送信できることを意味します。プログラミング・コマンドの受信中、プリンタは受け取った文字を送信元に‘エコー’・バックします。この機能は ECHO コマンドを使ってオフにすることができます。

2 **^JId:o.x,b,c,d** -- このコマンドをプリンタに送信すると ZBI セッションがアクティブになります。パラメータには次の関数があります。

d ＝初期化後に実行するプログラムの場所

有効値： R:, E: および B:
*デフォルト値：*場所を指定する必要があります

d ＝初期化後に実行するプログラムの名前

*有効値：*任意の有効なプログラム名
*デフォルト値：*名前を指定する必要があります。

x ＝初期化後に実行するプログラムの拡張子

固定値： .BAS

b ＝コンソール・コントロール

有効値： Y（コンソールがオン）または N （コンソールがオフ）
デフォルト値： Y

c ＝エコー・コントロール

有効値： Y（エコーがオン）または N （エコーがオフ）
デフォルト値： Y

d = ZBI のメモリ割り当て

有効値： 20K ～ 1024K
デフォルト値： 50K

インタープリタの実行中に、2 番目の ~JI または ^JI コマンドを受信した場合は、コマンドは無視されます。

ZBI Interpreter モード

ZBI Interpreter は次のモードで実行されます。

- 対話モードでは、コマンドを入力してすぐに処理できます。ライン番号なしで入力されたコマンド・ラインは対話式で、プリンタでの受信と同時に処理されます。

例 • 次は、NEW コマンドを使用する例です。

```
NEW
```

- プログラム・モードでは、一連のコマンド・ラインをメモリに保存し、RUN コマンドを入力したときに番号順に処理できます。ライン番号は 0 より大きく 10,000 より小さい必要があります。

例 • 次はその例です。

```
10 LET A=10
20 LET B=3
```

プログラムは RUN コマンドを入力することにより、最初の番号のラインからアクティブになります。プログラムの実行中、コンソールからプリンタに Control-C（^C）コマンドを送ることにより、停止することができます。中断したプログラムは、RESTART コマンドの入力により、中断した場所から再開されます。

ZBI セッションの終了

アクティブな ZBI セッションは次の方法のいずれかで停止できます。

- 入力プロンプト（>）で ZPL を送る。

- 入力プロンプト（>）で ~JQ を送る。

ZBI ファイルの暗号化

.baz 拡張子は、すべての ZBI 実行可能ファイルに対してセキュリティを提供します。.baz 拡張子は、ZBI の STORE コマンドを使用すると作成されます。

^HZO = .baz ファイルをプリンタにアップロードします。

重要 • セキュリティのため、ユーザーは以下のことを行わないでください。

- .baz= ファイルを開く
- .baz ファイルを .bas ファイル形式にアップロードまたはダウンロードする

例 • 次は .baz ファイルを生成する例です。

1. 下の単一ラインのプログラムのような基本的なプログラムを作成します。
2. 10 PRINT "ZPL" と入力します。
3. Store E:ZEBRA1.BAZ というファイル名でファイルを保存します。
4. 次のように ^HZO コマンドを使用して、新しく作成したファイルをプリンタから呼び出します。

   ^XA^HZO,E:ZEBRA1.BAZ^XZ

5. 新しいファイルを使用するには、^HZO コマンドの後の受信データを任意のプリンタにダウンロードします。

制約 次のコマンドは、ZBI 実行可能ファイルの読み込み時には機能しません。

- List
- Trace
- Debug

基本的な ZBI の例

この項では、ZBI の基本的な例を提供します。

例 1 • 次の例では、設定ラベルの印刷、テキストの印刷、プリンタのキャリブレーションなどの選択肢が示されています。ここで選択する項目はプリンタに送信される ZPL に対応します。

```
 5 REM PORTS SHOULD BE CLOSED, BY ROUTINE, BEFORE OPENING
10 CLOSE #1
20 CLOSE #2
25 REM OPENING PORT
40 OPEN # 1 :NAME "ZPL"
45 REM THE FOLLOWING WILL BE DISPLAYED ON THE TERMINAL
50 PRINT "YOU HAVE THREE CHOICES"
51 PRINT "PRESS 1 TO PRINT CONFIG LABEL" !prints to the console
52 PRINT "PRESS 2 TO PRINT TEXT "
53 PRINT "PRESS 3 TO CALIBRATE"
54 PRINT "OR PRESS 7 TO EXIT"
58 REM THE FOLLOWING ARE THE POSSIBLE INPUTS
60 INPUT A !PRINTER IS EXPECTING AN INPUT
61 IF A = 1 THEN
62 GOTO 120
63 ELSE
65 IF A = 2 THEN
66 GOTO 130
67 ELSE
70 IF A = 3 THEN
71 GOTO 140
72 ELSE
75 IF A = 7 THEN
76 GOTO 300
80 END IF
81 GOTO 51
85 REM IF THERE IS AN IF STATEMENT ALWAYS HAVE AN ENDIF
95 REM THE FOLLOWING WILL PRINT A CONFIGURATION LABEL
```

```
120 PRINT # 1 :"~WC" !~WC TELLS THE PRINTER TO PRINT CONFIGURATION
    LABEL
125 GOTO 50
130 PRINT "THIS TASK WILL SEND SAMPLE TEXT TO THE PRINTER"
131 PRINT "INPUT YOUR SAMPLE TEXT"
132 INPUT Y$
133 REM THE FOLLOWING STATEMENT WILL PRINT THE Y$ STRING VALUE
135 PRINT # 1 :"^XA^FO20,20^A0N50,50^FD" ; Y$ ; "^XZ"
136 GOTO 50
139 REM THE FOLLOWING COMMAND WILL TELL THE PRINTER TO CALIBRATE
140 PRINT # 1 :"~JG" !~JG ZPL COMMAND BEING SENT TO THE PRINTER
141 GOTO 50
300 PRINT "BEFORE CLOSING THE PRINTER WILL NOW PAUSE FOR 10 SECONDS"
301 SLEEP 10 !THE SLEEP COMMAND PAUSES THE PRINTER
310 END
```

例 2 • 次の例では、EXTRACT および SEARCHTO コマンドを使用した ZBI コードを示しています。

```
 10 REM ALWAYS CLOSE PORTS BEFORE OPENING (ROUTINE)
 20 CLOSE #1
 30 CLOSE #2
 60 OPEN # 1 :NAME "ZPL"
 69 REM SPECIFYING VALUE FOR STRING BELOW
 70 LET B$ = "START"
 79 REM USING SEARCHTO$ COMMAND TO SEARCH STREAMING DATA FOR B$ OR
    THE WORD "START"
 80 LET Z$ = SEARCHTO$ ( 0 , B$ )
 89 REM SPECIFYING VALUES FOR THE STRINGS F$ AND G$ ("T" AND "W ")
 90 LET F$ = "T "
100 LET G$ = " W"
108 REM USING THE EXTRACT$ COMMAND TO SEARCH STREAMING DATA FOR F$
    AND G$
109 REM X$ WILL BE THE DATA BETWEEN F$ AND G$
110 LET X$ = EXTRACT$ ( 0 , F$ , G$ )
250 PRINT # 1 :"^XA^F025,25^A050,50^FD" & X$ &"^FS^XZ"
1000 END
```

例 3 • 次の例では、ARRAYS を使用した ZBI コードを示しています。

```
  5 REM PORTS SHOULD BE CLOSED, BY ROUTINE, PRIOR TO OPENING
 10 CLOSE #1
 20 CLOSE #2
 40 OPEN # 1 :NAME "ZPL"
 45 REM WHEN USING ARRAYS, THE ARRAY HAS TO BE DECLARED BEFORE
    USING
 49 REM THIS ARRAY "NAME$" IS BEING DEFINED WITH 5 SPOTS FOR DATA
 50 DECLARE STRING NAME$ (5)
 57 REM THE FOLLOWING ALLOWS ONE TO INPUT 5 DIFFERENT VALUES AND
    THE VALUES
 58 REM ARE PLACED INTO THE ARRAY IN ORDER (1-5)
 59 PRINT "INPUT YOUR FIRST 5 NAMES"
 60 FOR INDEX = 1 TO 5 STEP 1
 70 INPUT NAME$(INDEX)
 80 NEXT INDEX
 89 REM THE FOLLOWING DECLARES AN ARRAY (NUMBERS$) AND ALLOWS 5
    SPOTS FOR DATA
 90 DECLARE STRING NUMBERS$ (5)
 99 PRINT "INPUT YOUR FIRST 5 NUMBERS"
100 FOR INDEX = 1 TO 5 STEP 1
110 INPUT NUMBERS$(INDEX)
120 NEXT INDEX
128 REM THE FOLLOWING PRINTS OUT THE VALUES THAT HAVE BEEN STORED
    IN THE ARRAYS
130 PRINT # 1 :"^XA^FO60,60^A0N50,50^FD"&NAME$(1)&"^FS";
140 PRINT # 1 :"^FO60,120^A0N50,50^FD"&NUMBERS$(1)&"^FS^XZ"
150 PRINT # 1 :"^XA^FO60,60^A0N50,50^FD"&NAME$(2)&"^FS";
160 PRINT # 1 :"^FO60,120^A0N50,50^FD"&NUMBERS$(2)&"^FS^XZ"
170 PRINT # 1 :"^XA^FO60,60^A0N50,50^FD"&NAME$(3)&"^FS";
180 PRINT # 1 :"^FO60,120^A0N50,50^FD"&NUMBERS$(3)&"^FS^XZ"
190 PRINT # 1 :"^XA^FO60,60^A0N50,50^FD"&NAME$(4)&"^FS";
200 PRINT # 1 :"^FO60,120^A0N50,50^FD"&NUMBERS$(4)&"^FS^XZ"
210 PRINT # 1 :"^XA^FO60,60^A0N50,50^FD"&NAME$(5)&"^FS";
220 PRINT # 1 :"^FO60,120^A0N50,50^FD"&NUMBERS$(5)&"^FS^XZ"
300 END
```

例 4 • 次の ZBI コードは整数関数を示しています。

```
10 REM THIS PROGRAM WILL SHOW VARIOUS INTEGER FUNCTIONS
11 CLOSE #1
12 CLOSE #2
20 PRINT "ENTER YOUR LAST NAME"
25 INPUT A$ !PRINTER IS EXPECTING A STRING INPUT
30 LET X = LEN ( A$ ) !THIS IS WHERE THE LENGTH IS DETERMINED
35 PRINT "YOUR LAST NAME (" ; A$ ; ") CONTAINS " ; X ; " CHARACTERS"
36 PRINT
40 PRINT "ENTER 2 DIFFERENT NUMBERS EACH FOLLOWED BY THE ENTER KEY"
45 INPUT A
46 INPUT B
55 LET Y = MIN ( A , B ) !THIS IS WHERE THE SMALLEST NUMBER IS
   DETERMINED
60 PRINT "THE SMALLEST NUMBER THAT YOU ENTERED WAS " ; Y
61 PRINT
65 PRINT "ENTER ANY ASCII CHARACTER FROM YOUR KEYBOARD"
66 INPUT B$
70 LET U = ORD ( B$ ) !THIS IS WHERE THE ASCII EQUIVELANT IS
   DETERMINED
71 LET R = 34
75 PRINT "THE ASCII EQUIVELANT OF " , CHR$ ( R ) , B$ , CHR$ ( 290 )
   , "IS" , U !THIS LINE SHOWS TWO DIFFERENT WAYS TO PRINT THE "
   CHARACTER USING THE CHR$ FUNCTION
76 REM IN LINE 75 I AM USING "," UNLIKE THE ";" THAT WAS USED IN LINE
   35
78 REM WHEN SEPERATING WITH THE "," CHARACTER A SPACE WILL BE
   INCLUDED
79 REM IF USING THE ";" NO SPACE WILL BE INCLUDED AS IN LINE 34
80 END
```

例 5 • 次の例は、数式を示す ZBI コードです。

```
 10 REM This program will demonstrate the use of Numeric
    expressions
 11 CLOSE #1
 12 CLOSE #2
 20 REM SIMPLE PROGRAM SHOWING HOW TO USE NUMERIC EXPRESSIONS
 28 PRINT
 30 PRINT "The first example will be addition, subtraction,
    mulitiplication , division and exponentiation"
 31 PRINT " Input your first number"
 32 INPUT A
 33 PRINT " Input your second number"
 35 INPUT B
 38 LET X = A + B !simple math (addition)
 39 LET Y = A - B !simple math (subtraction)
 40 PRINT "If you add" , A , "plus" , B , "you get" , X
 43 PRINT "If you subtract" , A , "minus" , B , "you get" , Y
 50 LET Z = A * B !simple math (multiplication)
 70 PRINT
 71 PRINT "IF YOU MULTIPLY" , A , "AND" , B , "YOUR RESULT IS" , Z
 75 LET J = A / B !simple math (division)
 76 ON ERROR GOTO 100 !This statement allows the program to
    continue
 78 PRINT "IF YOU DIVIDE" , A , "BY" , B , "YOU GET" , J
 79 IF J = 0 THEN
 80 GOSUB 400
 82 GOTO 200
100 PRINT "THERE HAS BEEN A DIVIDE BY ZERO ERROR"
121 PRINT "THE SECOND NUMBER YOU ENTERED WAS A ZERO"
122 PRINT "THE ERROR WAS CAUSED BY TRYING TO DIVIDE" , A , "BY" , B
200 PRINT
201 PRINT " EXPONENTIATION USES THE ^ SYMBOL, X^2 WOULD BE X
    SQUARED"
202 LET G = A ^ B !exponentiation
205 PRINT A , "^" , B , "RESULTS IN" , G
300 END
400 PRINT
```

```
401 PRINT "YOUR RESULT WHEN YOU DIVIDED" , A , "BY" , B , "WAS ZERO"
405 PRINT "REMEMBER WHEN USING DIVISION THE NUMBER WILL ALWAYS BE
    ROUNDED DOWN"
407 RETURN
```

例 6 • 電子メール・サーバーを使用した ZBI の例

```
 10 CLOSE #1
 20 CLOSE #2
 30 OPEN # 1 :NAME "EML" !PORT SET FOR EML FOR EMAIL SERVER
 31 OPEN # 2 :NAME "ZPL"
 32 PRINT "ENTER YOUR FIRST NAME"
 35 INPUT A$
 36 PRINT "ENTER YOUR LAST NAME"
 37 INPUT B$
 40 PRINT "ENTER YOUR EMPLOYEE NUMBER"
 42 INPUT C$
 45 PRINT "Hello ";A$; " We are now ready to proceed"
400 PRINT # 1 :"JDOE@ZEBRA.COM";CHR$(4); !SENDING MESSAGE TO JDOE
405 PRINT # 1 :"FROM:PRINTER #1" !MAIL FROM:DATA ("PRINTER #1")
410 PRINT # 1 :"TO: MAIN LOG SERVER (EMAIL)" ! MAIL TO: DATA ("MAIN
    LOG SERVER")
500 PRINT #1 : "SUBJECT: EMPLOYEE "; B$ ;" ( ";C$;" ) LOGGED IN"!
    SUBJECT : DATA
505 PRINT # 1 : ""
510 PRINT #1 : "EMPLOYEE "; B$; " LOGGED IN" ! MAIL MESSAGE
550 PRINT #1 : CHR$(4) ! IMPORTANT TO END WITH EOT
600 CLOSE #1
700 CLOSE #2
    In this program an e-mail message will be sent to
    JDOE@ZEBRA.COM <mailto:JDOE@ZEBRA.COM> that indicates that an
    employee has logged in
```

例 7 • TCP ポートを使用した ZBI

```
 10 CLOSE #1
 20 CLOSE #2
 30 OPEN # 1 :NAME "TCP"
 31 OPEN # 2 :NAME "ZPL"
 32 PRINT "ENTER THE PRODUCT NAME"
 35 INPUT A$
 36 PRINT "ENTER QUANTITY"
 37 INPUT B
 40 PRINT "ENTER THE LOCATION AISLE/SHELF # EXAMPLE 3/4 "
 42 INPUT C$
 45 PRINT # 2 :"^XA^FO100,100^A0N,50,50^FD";A$;"^FS"; !SENDING ZPL
    LOCALLY
 46 PRINT # 2 :"^FO100,200^A0N,50,50^FD";B;"^XZ"
400 PRINT # 1 :"10.3.50.79 6101";CHR$(4); !OPENING TCP PORT
405 PRINT # 1 :"! 0 200 200 400 1" !CPCL STARTS
410 PRINT # 1 :"JOURNAL"
415 PRINT # 1 :"PAGE-WIDTH 400"
420 PRINT # 1 :"T 4 0 100 20 AISLE/SHELF"
500 PRINT # 1 :"T 4 0 100 100 ";C$
501 PRINT # 1 :"T 4 0 100 150 QUANTITY"
505 PRINT # 1 :"T 4 0 100 200 ";B
510 PRINT # 1 :"T 4 0 100 250 DESCRIPTION"
520 PRINT # 1 :"T 4 0 100 300 ";A$
530 PRINT # 1 :"PRINT" !CPCL ENDS WITH PRINT
550 PRINT # 1 :CHR$(4);
600 CLOSE #1
700 CLOSE #2
```

ユーザーは、製品名、数量、通路 / 棚の場所、および価格を入力します。ローカル・ラベルには説明（製品名）、数量、および価格が印刷されますが、TCP ポート 6101 を使用する IP アドレス 10.3.50.79 のリモート・プリンタ（CPCL を使用したポータブル・プリンタ）では、製品名、数量、および製品の場所（通路 / 棚）が印刷されます。

例 8 • UDP ポートを使用した ZBI。次の例では、アイテム、数量、価格、プリンタ番号がプリンタから送信されます。ログにより月、日付、時刻、および IP アドレスが自動的に生成されます。

サーバーからのテキスト・ファイル

Jun	30	13:59:45	10.3.9.195	4	45	1.99	PRINTER5
Jun	30	14:00:01	10.3.9.195	3	23	8.69	PRINTER5
Jun	30	14:00:14	10.3.9.195	40	3	1.99	PRINTER5
Jun	30	14:00:24	10.3.9.195	5	65	5.99	PRINTER5

```
 10 CLOSE #1
 20 CLOSE #2
 30 OPEN #1:NAME "ZPL"
 40 OPEN #2:NAME "UDP"
 45 LET C$ = "PRINTER5"
 50 PRINT "ENTER PRODUCT NUMBER 1-45"
 55 INPUT A
 60 PRINT "ENTER QUANTITY"
 65 INPUT B
 70 PRINT "ENTER $ AMOUNT EXAMPLE 6.99"
 75 INPUT A$
 80 PRINT #2 :"10.3.50.79 514";CHR$(4);
 90 PRINT #2 :A ;" ";B;" ";A$;" ";" ";" ";C$;
 91 REM The extra spaces in above line are for
 92 REM columns in ACCESS database where no data is sent or
    needed from the printer
 95 PRINT #2 :CHR$(4);
100 CLOSE #1
105 CLOSE #2
```

索引

Symbols

C

D

E

I

L

M

P

Q

T

U

Z

あ

い

え

お

か

き

く

け

こ

さ

し

す

せ

そ

た

つ

て

と

ね

の

は

ひ

ふ

へ

ほ

ま

み

め

も

よ

ら

り

れ

索引

Zebra Technologies Corporation
333 Corporate Woods Parkway
Vernon Hills, IL 60061-3109 U.S.A.
電話 : +1 847.634.6700
ファックス : +1 847.913.8766

Zebra Technologies Europe Limited
Zebra House
The Valley Centre, Gordon Road
High Wycombe
Buckinghamshire HP13 6EQ, UK
電話 : +44 (0) 1494 472872
ファックス : +44 (0) 1494 450103

顧客注文番号 45541L
製造部品番号 45541LB-Rev3