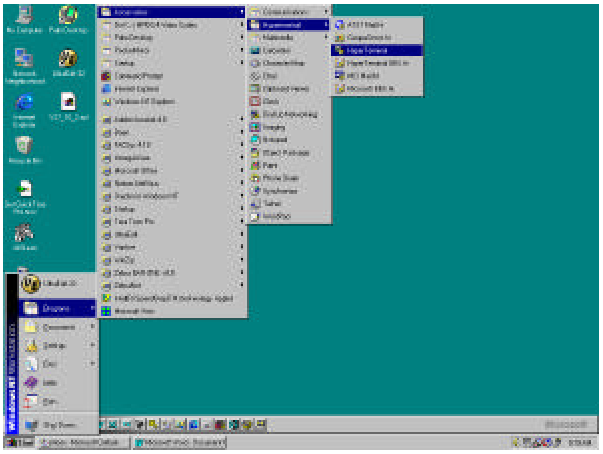
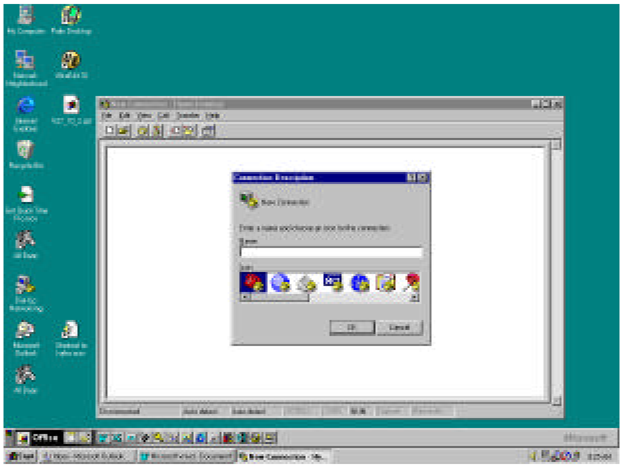
ZBI Program Downloading Procedure

ZBI Downloading Procedure

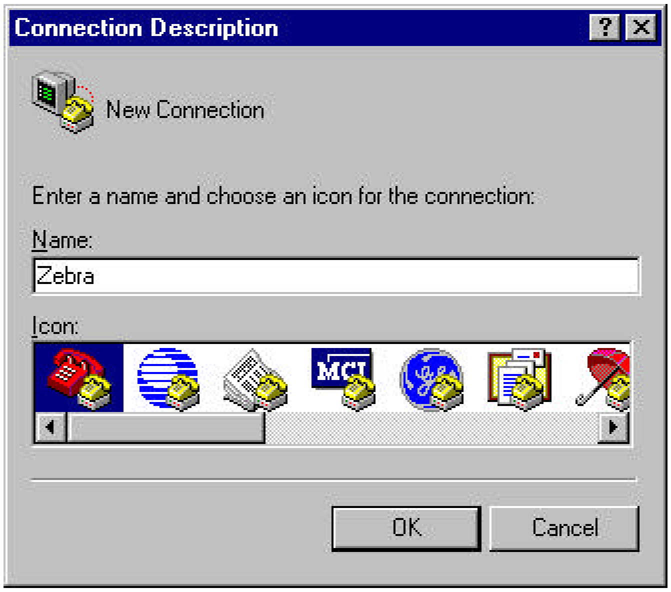
1. To download a ZBI program for the printer to use, the use of a terminal emulator is required such as Hyper Terminal in Windows. Terminal emulators are only available as serial communication connections, so the use of a null modem cable is required. Hyper terminal is usually located in the Start menu under programs and accessories.



2. If a terminal has not already been established previously on the computer, one will need to be made up from the following procedure.   After clicking on the hyper terminal item in the star menu, the following screen will appear.



3. Type in a name for the new connection. In this case the name will be Zebra.then press the ok button. Then click OK.



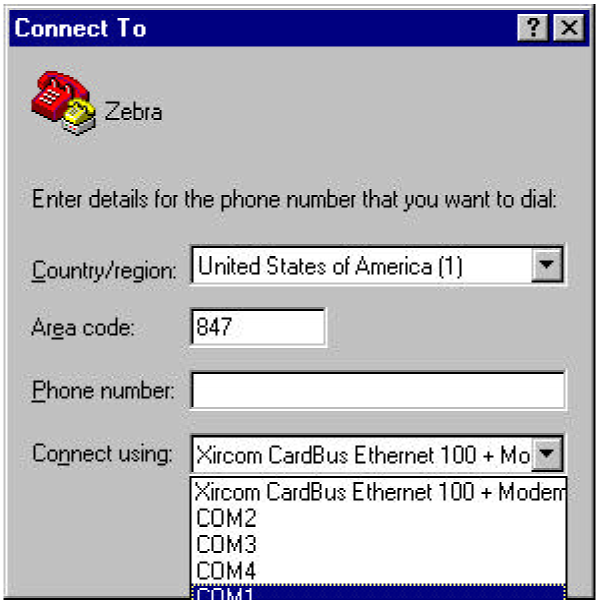
Connection Description

New Connection

Enter a name and choose an icon for the connection:

Name: Zebra

4. The program will then prompt for a connection type. There will be a drop down menu next to **connect using.** Select COM 1 or whatever COM port that is available. **Do not select Xircom CardBus Ethernet 100**. Then click OK



Connect To

Zebra

Enter details for the phone number that you want to dial:

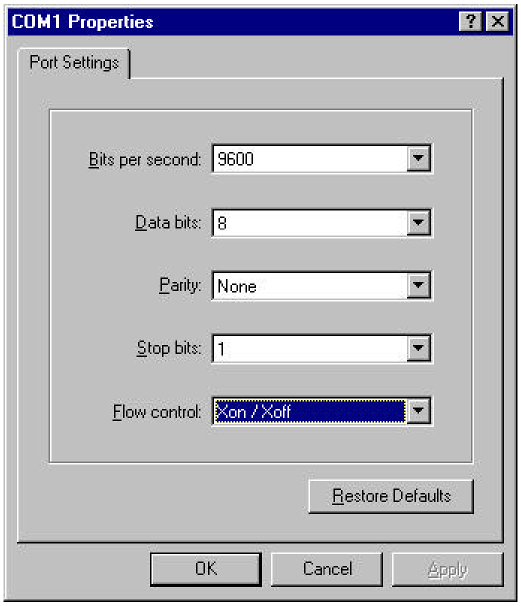
Country/region: United States of America (1)

Area code: 847

Phone number:

Connect using: Xircom CardBus Ethernet 100 + Mo

5. Hyper Terminal will then ask about the COM parameters. The Defaults for Zebra printers is 9600 baud, 8 data bits, 1 stop bit, no parity, and Xon\Xoff handshaking. Then click OK.



COM1 Properties

Port Settings

Bits per second: 9600

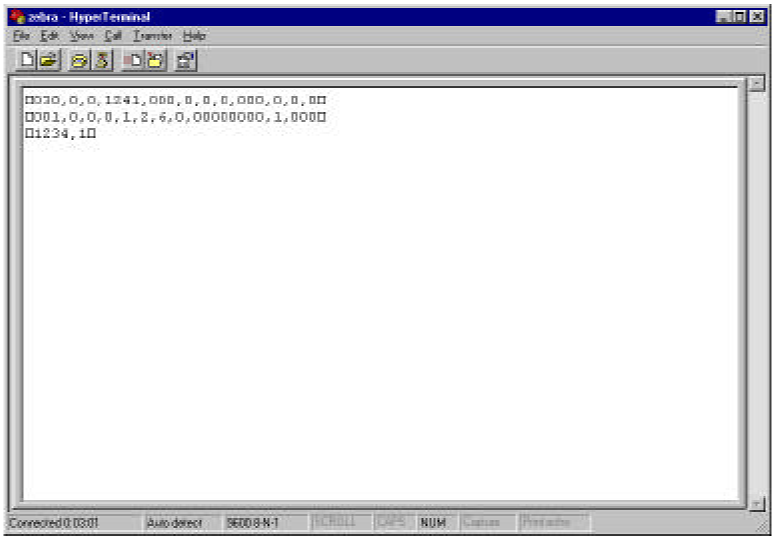
Data bits: 8

Parity: None

Stop bits: 1

Flow control: Xon / Xoff

6. At this point there will be a blank screen. Please note that **when a key is pressed on the keyboard, nothing will appear in the screen.** To check for proper communications with the printer, type in ~HS. The following string will appear on the screen.

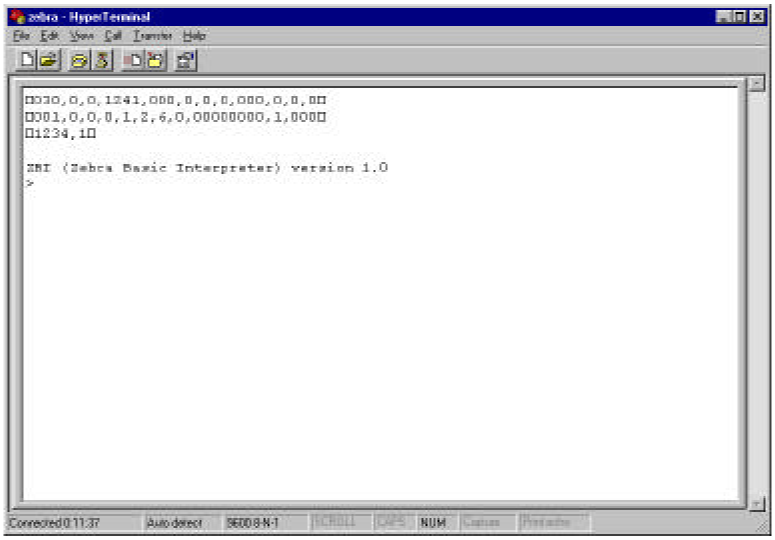


zebra - HyperTerminal

```
□030,0,0,1241,000,0,0,0,000,0,0,0□
□001,0,0,0,1,2,6,0,00000000,1,000□
□1234,1□
```

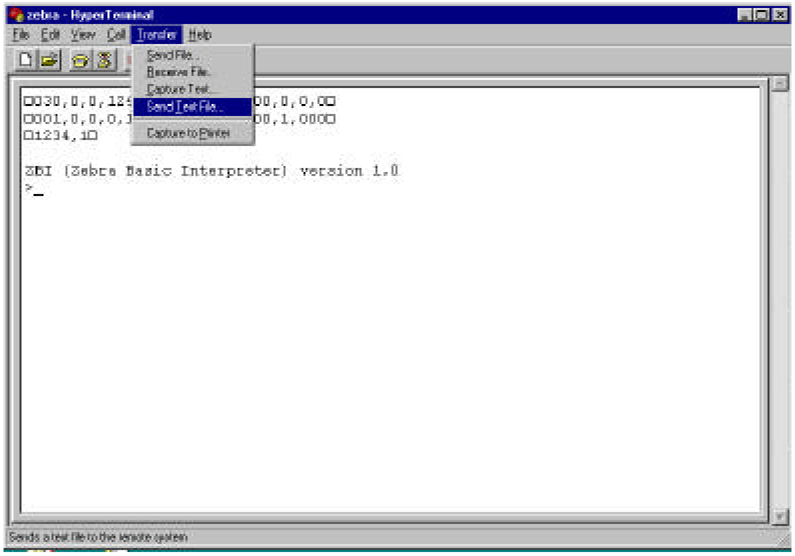
Connected 0:03:01 Auto detect 9600 8-N-1

7. Type in the ~JI command to start the ZBI interpreter. Please note **if the Zbi program that is being loaded onto the printer requires more than 50k of memory space, then the ^JI command must be used. Consult the ZPL programming manual.** The screen will look like this.



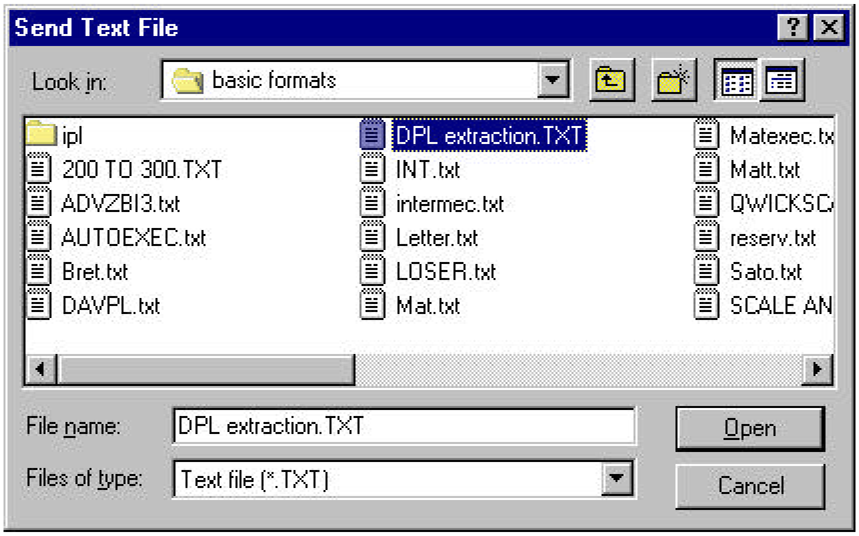
```
□030,0,0,1241,000,0,0,0,000,0,0,0□
□001,0,0,0,1,2,6,0,00000000,1,000□
□1234,1□

ZBI (Zebra Basic Interpreter) version 1.0
>
```

8. Once the connection is established and the ZBI version is on the string, a program can be downloaded to the printer. To select a program to send , go to the Transfer menu and select send text file.



```
□030,0,0,12                    00,0,0,0□
□001,0,0,0,1                   00,1,000□
□1234,1□

ZBI (Zebra Basic Interpreter) version 1.0
>_
```

9. Now go to the desired directory and select the file to be transferred.

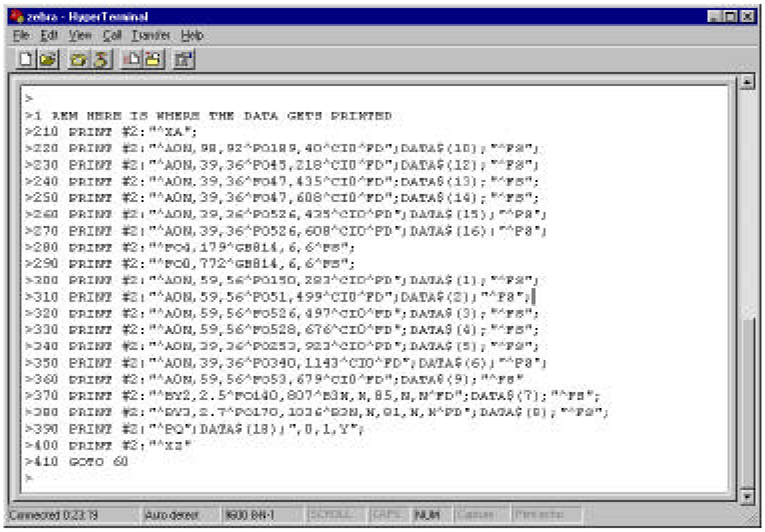


Look in: basic formats

File name: DPL extraction.TXT

Files of type: Text file (*.TXT)

10. Press the OPEN button and the file will be transferred. The text of the file will appear on the screen as it is being transferred to the printer.



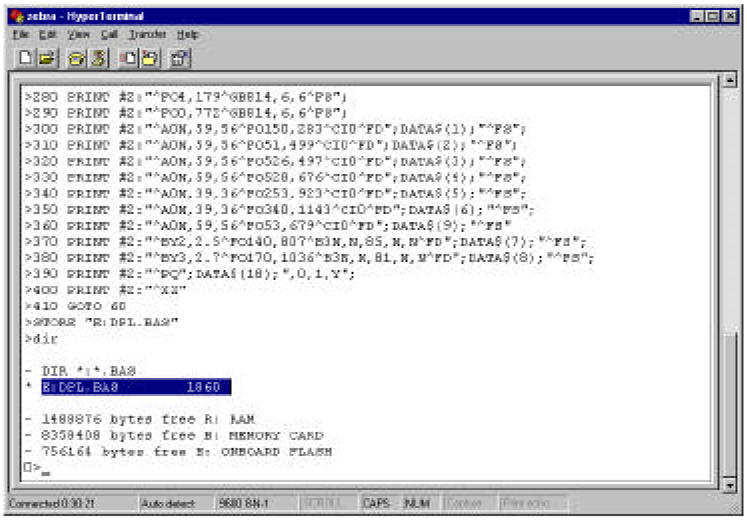
```
>
>1 REM HERE IS WHERE THE DATA GETS PRINTED
>210 PRINT #2:"^XA";
>220 PRINT #2:"^AON,98,92^FO189,40^CI0^FD";DATA$(10);"^FS";
>230 PRINT #2:"^AON,39,36^FO45,218^CI0^FD";DATA$(12);"^FS";
>240 PRINT #2:"^AON,39,36^FO47,435^CI0^FD";DATA$(13);"^FS";
>250 PRINT #2:"^AON,39,36^FO47,608^CI0^FD";DATA$(14);"^FS";
>260 PRINT #2:"^AON,39,36^FO526,435^CI0^FD";DATA$(15);"^FS";
>270 PRINT #2:"^AON,39,36^FO526,608^CI0^FD";DATA$(16);"^FS";
>280 PRINT #2:"^FO4,179^GB814,6,6^FS";
>290 PRINT #2:"^FO0,772^GB814,6,6^FS";
>300 PRINT #2:"^AON,59,56^FO150,283^CI0^FD";DATA$(1);"^FS";
>310 PRINT #2:"^AON,59,56^FO51,499^CI0^FD";DATA$(2);"^FS";
>320 PRINT #2:"^AON,59,56^FO526,497^CI0^FD";DATA$(3);"^FS";
>330 PRINT #2:"^AON,59,56^FO528,676^CI0^FD";DATA$(4);"^FS";
>340 PRINT #2:"^AON,39,36^FO253,923^CI0^FD";DATA$(5);"^FS";
>350 PRINT #2:"^AON,39,36^FO340,1143^CI0^FD";DATA$(6);"^FS";
>360 PRINT #2:"^AON,59,56^FO53,679^CI0^FD";DATA$(9);"^FS"
>370 PRINT #2:"^BY2,2.5^FO140,807^B3N,N,85,N,N^FD";DATA$(7);"^FS";
>380 PRINT #2:"^BY3,2.7^FO170,1036^B3N,N,91,N,N^FD";DATA$(8);"^FS";
>390 PRINT #2:"^PQ";DATA$(18);",0,1,Y";
>400 PRINT #2:"^XZ"
>410 GOTO 60
>
```

11. The program is now on the console and ready to be run, or stored in the printer. To store the file, use the store command with the following syntax.

STORE "E:FILENAME.BAS"

Please note that E; is the memory location in the printer and that FILENAME can be any name that is desired as long as it does not exceed 8 characters.

12 To verify that the program has been stored, type in DIR and the file should appear as follows. The highlighted area shows the file name which is this case is DPL.BAS

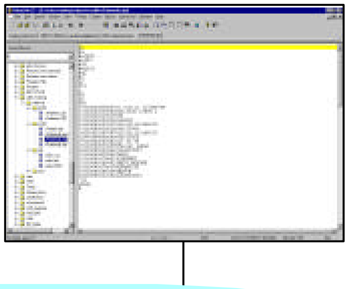


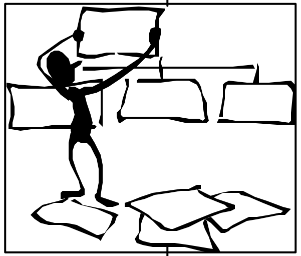
```
>280 PRINT #2:"^FO4,179^GB814,6,6^FS";
>290 PRINT #2:"^FO0,772^GB814,6,6^FS";
>300 PRINT #2:"^A0N,59,56^FO150,283^CI0^FD";DATA$(1);"^FS";
>310 PRINT #2:"^A0N,59,56^FO51,499^CI0^FD";DATA$(2);"^FS";
>320 PRINT #2:"^A0N,59,56^FO526,497^CI0^FD";DATA$(3);"^FS";
>330 PRINT #2:"^A0N,59,56^FO528,676^CI0^FD";DATA$(4);"^FS";
>340 PRINT #2:"^A0N,39,36^FO253,923^CI0^FD";DATA$(5);"^FS";
>350 PRINT #2:"^A0N,39,36^FO340,1143^CI0^FD";DATA$(6);"^FS";
>360 PRINT #2:"^A0N,59,56^FO53,679^CI0^FD";DATA$(9);"^FS"
>370 PRINT #2:"^BY2,2.5^FO140,807^B3N,N,85,N,N^FD";DATA$(7);"^FS";
>380 PRINT #2:"^BY3,2.7^FO170,1036^B3N,N,81,N,N^FD";DATA$(8);"^FS";
>390 PRINT #2:"^PQ";DATA$(10);",0,1,Y";
>400 PRINT #2:"^XZ"
>410 GOTO 60
>STORE "E:DPL.BAS"
>dir

- DIR *:*.BAS
* E:DPL.BAS       1860

- 1488876 bytes free R: RAM
- 8358408 bytes free B: MEMORY CARD
- 756164 bytes free E: ONBOARD FLASH
0>_
```

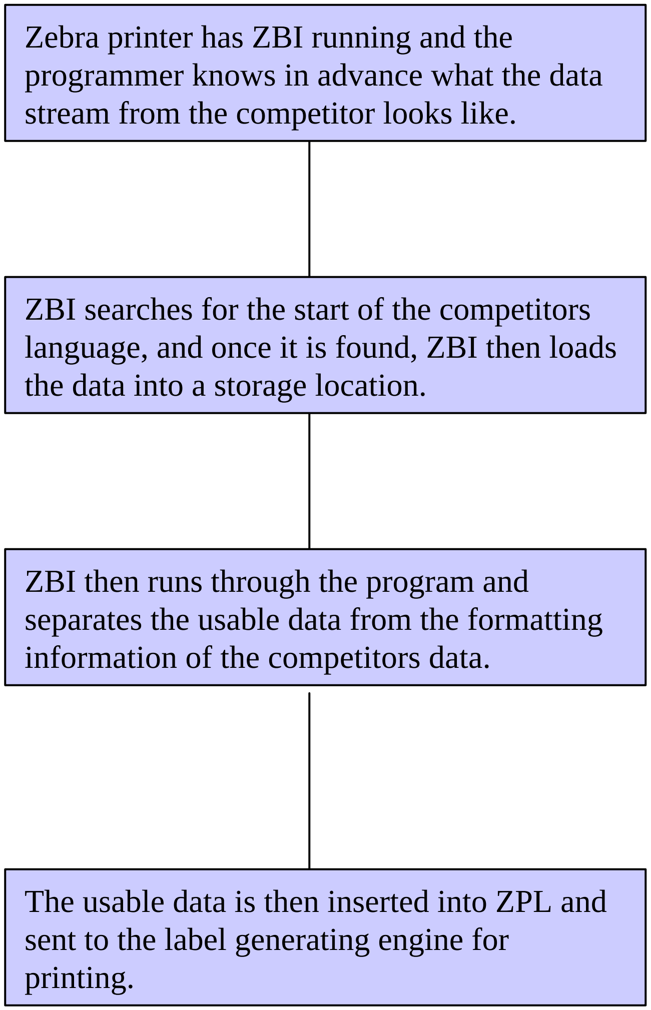
13. The program is stored and ready to run by using the run command.

DPL extraction Program

DPL Extraction Program







Zebra printer has ZBI running and the programmer knows in advance what the data stream from the competitor looks like.

ZBI searches for the start of the competitors language, and once it is found, ZBI then loads the data into a storage location.

ZBI then runs through the program and separates the usable data from the formatting information of the competitors data.

The usable data is then inserted into ZPL and sent to the label generating engine for printing.

DPL Extraction

```
1 REM THIS OPENS ALL OF THE REQUIRED PORTS

10 CLOSE #1
20 CLOSE #2
30 OPEN #1: NAME "SER"
40 OPEN #2: NAME "ZPL"

1 REM THIS SEARCHES FOR THE START OF THE FORMAT

50 LET START$ = CHR$(2) & "L"
60 LET JUNK$ = SEARCHTO$(1,START$,2)

1 REM THIS IS TO INPUT THE STUFF THAT IS NOT NEEDED

70 FOR I = 1 TO 7 STEP 1
80 INPUT #1: JUNK$
90 NEXT I

1 REM THIS INPUTS THE STRINGS FOR EXTRACTING

100 DECLARE STRING TEMP$(20)
110 FOR I = 1 TO 17 STEP 1
120 INPUT #1:  TEMP$(I)
130 NEXT I

1 REM THIS IS WERE THE DATA IS EXTRACTED

140 DECLARE STRING DATA$(20)
150 FOR I = 1 TO 17
160 LET DATA$(I) = TEMP$(I)(16:999)
170 NEXT I
180 INPUT #1:  JUNK$
190 INPUT #1:  Q$
200 LET DATA$(18) = Q$(2:999)


1 REM HERE IS WHERE THE DATA GETS PRINTED

210 PRINT #2:"^XA";
220 PRINT #2:"^A0N,98,92^FO189,40^CI0^FD";DATA$(10);"^FS";
230 PRINT #2:"^A0N,39,36^FO45,218^CI0^FD";DATA$(12);"^FS";
240 PRINT #2:"^A0N,39,36^FO47,435^CI0^FD";DATA$(13);"^FS";
250 PRINT #2:"^A0N,39,36^FO47,608^CI0^FD";DATA$(14);"^FS";
260 PRINT #2:"^A0N,39,36^FO526,435^CI0^FD";DATA$(15);"^FS";
270 PRINT #2:"^A0N,39,36^FO526,608^CI0^FD";DATA$(16):"^FS";
280 PRINT #2:"^FO4,179^GB814,6,6^FS";
290 PRINT #2:"^FO0,772^GB814,6,6^FS";
300 PRINT #2:"^A0N,59,56^FO150,283^CI0^FD";DATA$(1);"^FS";
310 PRINT #2:"^A0N,59,56^FO51,499^CI0^FD";DATA$(2);"^FS";
320 PRINT #2:"^A0N,59,56^FO526,497^CI0^FD";DATA$(3);"^FS";
330 PRINT #2:"^A0N,59,56^FO528,676^CI0^FD";DATA$(4);"^FS";
340 PRINT #2:"^A0N,39,36^FO253,923^CI0^FD";DATA$(5);"^FS";
350 PRINT #2:"^A0N,39,36^FO340,1143^CI0^FD";DATA$(6);"^FS";
360 PRINT #2:"^A0N,59,56^FO53,679^CI0^FD";DATA$(9);"^FS"
```

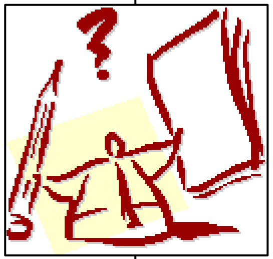
DPL Extraction

```
370 PRINT #2:"^BY2,2.5^FO140,807^B3N,N,85,N,N^FD";DATA$(7);"^FS";
380 PRINT #2:"^BY3,2.7^FO170,1036^B3N,N,81,N,N^FD";DATA$(8);"^FS";
390 PRINT #2:"^PQ";DATA$(18);",0,1,Y";
400 PRINT #2:"^XZ"
410 GOTO 60
```

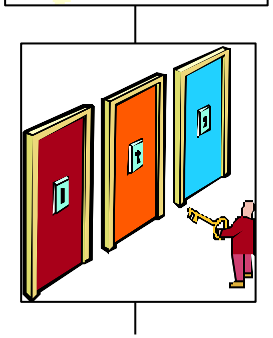
Scale and Scanner Application

Scale and Scanner Application

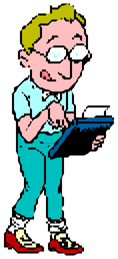
The Zebra printer uses a simple A/B serial switch to select between devices



The scale must be queried for a weight. So the ZBI program sends out a W for the scale and then looks to see if there is data on the port.



If data is seen on the port, ZBI inputs the data and checks to see if the data is from the scale or the scanner. If the data is a weight, ZBI checks to see if a price is in memory.



If there is a price in memory, ZBI Multiplies the weight with the price and prints a label. If no price has been scanned in, the program waits for the scanner input.

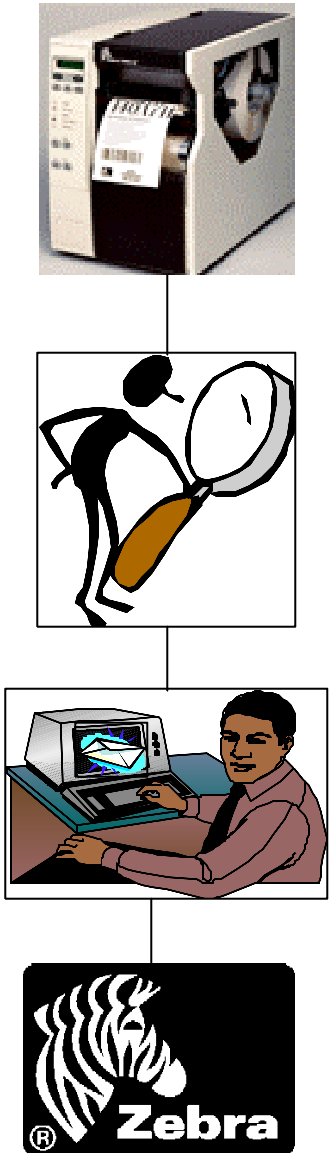
Scale and Scanner APP

```
1 REM OPENS PORTS FOR SERIAL AND FOR THE ZPL ENGINE

10 CLOSE #1
20 CLOSE #2
30 OPEN #1: NAME "SER"
40 OPEN #2: NAME "ZPL"

1 REM SETS COMM PARAMETERS FOR THE SERIAL PORT

50 PRINT #2: "^XA^SC9600,8,N,1,X,N^XZ"
55 LET COST = 0

1 REM SENDS COMMAND TO QUERY SCALE AND SEE IF THE RESPONSE IS FROM THE SCALE
OR FROM THE SCANNER

60 PRINT #1: "W";
70 IF DATAREADY(1) = 0 THEN
80 SLEEP 2
90 GOTO 60
100 END IF
110 INPUT #1: UNKNOWN$
120 IF POS(UNKNOWN$,"@") <> 0 THEN
130 GOTO 400 ! SCANNER PART OF THE PROGRAM
140 END IF
150 GOTO 600 !SCALE PART OF PROGRAM


1 REM SCANNER INPUT SECTION

400 DECLARE STRING ITEM$(20)
410 LET ITEM$(1) ="@SCREWS"
420 LET ITEM$(2) = "@NUTS"
430 IF UNKNOWN$ = ITEM$(1) THEN
450 LET COST = 139
451 CLOSE #1
452 OPEN #1: NAME "SER"
453 LET STORE$ = "SCREWS"
460 GOTO 60
470 END IF
480 IF UNKNOWN$ = ITEM$(2) THEN
490 LET COST = 120
491 CLOSE #1
492 OPEN #1: NAME "SER"
493 LET STORE$ = "NUTS"
500 GOTO 60
510 END IF
520 PRINT #2: "^XA^A0N,50,50^FDINVALID ITEM^FS^XZ"
523 CLOSE #1
525 OPEN #1: NAME "SER"
530 GOTO 60


1 REM SCALE INPUT AND COST CALCULATION SECTION
```

```
600 IF COST = 0 THEN
610 GOTO 60
620 END IF
630 IF POS ( UNKNOWN$ , "000.00" ) = 1 OR POS ( UNKNOWN$ , "?" ) = 1 THEN
640 GOTO 60
650 END IF
660 LET FIRST$ = UNKNOWN$(1:3)
670 LET SECOND$ = UNKNOWN$(5:6)
680 LET WEIGHT$ = FIRST$ & SECOND$
690 LET WEIGHT = VAL(WEIGHT$)
700 LET FINAL = WEIGHT * COST
710 LET FINAL$ = STR$(FINAL)
720 LET DEC = LEN(FINAL$)
730 LET POINT = DEC - 4
740 LET DOLLAR$ = FINAL$(1:POINT)
750 LET CENT$ = FINAL$(POINT +1:DEC)
760 LET MONEY$ = DOLLAR$ &"."&CENT$(1:2)
780 PRINT #2: "^XA";
790 PRINT #2: "^FO104,109^A0N,183,160^CI13^FR^FD";STORE$;"^FS";
800 PRINT #2: "^BY3,3.0^FO123,288^B3N,N,200,Y,N^FR^FD123456789^FS";
810 PRINT #2: "^FO109,663^A0N,37,33^CI13^FR^FDCost per
LBS.....................";
820 LET COST$ = STR$(COST)
830 PRINT #2: "$";COST$(1:1);".";COST$(2:999);"^FS";
840 PRINT #2: "^FO109,738^A0N,37,33^CI13^FR^FDWeight of
product..............";UNKNOWN$;" lbs^FS";
850 PRINT #2: "^FO109,829^A0N,37,33^CI13^FR^FDCost of this
unit.................$";MONEY$;"^FS";
860 PRINT #2: "^FO13,563^GB791,0,4^FS";
870 PRINT #2: "^PQ1,0,0,N";
880 PRINT #2: "^XZ";
890 LET COST = 0
900 CLOSE #1
910 OPEN #1: NAME "SER"
920 GOTO 60
```

Alert for Media Inventory

Alert for Media Inventory



The printer with an advanced counter option installed, prints labels and records the number of labels that have been printed.

ZBI watches the advanced counter increment until a certain number, that was defined by the user ,is achieved.

Once the the number of labels printed reaches the target value, ZBI sends an Email to the required people. In this case the Email would be sent to Zip Zebra.

The email contains the labels and ribbon numbers that the customer is using, and also the quantity of labels that have been printed.

Alert for Counter

```
1 REM OPENS PORTS AND INITIALIZES SEARCH ARRAYS

10 CLOSE #1
20 CLOSE #0
30 OPEN #1:NAME "ZPL"
40 DECLARE STRING SEARCH$(3)
50 LET CHECKINT = 780
60 LET SEARCH$(1) = "RAM"
70 LET SEARCH$(2) = ":LABEL.ZPL"
80 LET SEARCH$(3) = "FLASH"
90 DECLARE STRING XML$(2)
100 LET XML$(1) = "LABELS"
110 LET XML$(2) = "COUNTER2"


1 REM CHECKS R:MEMORY FOR A ZPL FILE TO ASSIGN THE CHECK INTERVAL

120 PRINT #1: "^XA^HW^XZ"
130 LET START$ = SEARCHTO$(1,SEARCH$)


1 REM IF THERE IS NO FILE THEN CHECK E: MEMORY

140 IF START$ = "RAM" THEN
150 GOSUB 2000
160  SLEEP 10
170 GOTO 1000
180 END IF


1 REM IF THERE IS A FILE IN R: TEHN TRANSFER TO E: AND CHECK MEMORY AVAILABLE

190 PRINT #1 : "^XA^HFR:LABEL.ZPL^XZ"
200 LET TEMP$ = SEARCHTO$(1,"^FX")
210 LET CHECKINT$ = EXTRACT$(1,"","^FS")
220 LET CHECKINT = VAL(CHECKINT$)
230 PRINT #1 : "^XA^TOR:LABEL.ZPL,E:LABEL.ZPL^XZ"
240 Print #1 :"^XA^IDR:LABEL.ZPL^FS^XZ"
250 GOSUB 3000



1 REM CHECK TO SEE HOW MANY LABELS HAVE BEEN PRINTED

1000 PRINT #1 : "^XA^HZR^XZ"
1030 LET GARBAGE$ = SEARCHTO$ (1,"COUNTER1")
1040 LET GARBAGE$ = SEARCHTO$ (1,XML$)
1050 IF GARBAGE$ = "COUNTER2" THEN
1060 GOTO 1500
1065 END IF
1070 INPUT #1 : EXTRA$
1080 LET COUNT$ = EXTRACT$(EXTRA$,">","<")
1090 LET COUNT = VAL(COUNT$)
```

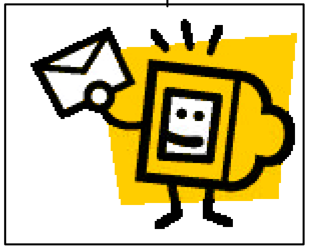
Zbi Alert for Front Panel Changes

Alert for Front Panel Change

The printer needs to have a front panel LCD control and a ZebraNet II PrintSever installed.

ZBI sets all of the front panel settings to a user selected default. ZBI then loads the front panel settings into its memory for comparisons.

Once known good settings are loaded into ZBI the program then checks to see if an item has been changed by comparing the known good config to the current config every 2 seconds.



If there is a difference detected in ZBI, an email is sent to the administrator showing what fields have been changed.

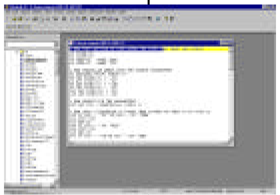
Front Panel Check for ZBI

```
1    REM Automatic Configuration Monitor v2.1 (Richard Powlesland)
10   GOSUB 9500  ! Setup variables and ports

1 REM  STORE 1ST CONFIG LABEL

100  LET config = 1
105  GOSUB 9600
110  LET config_old$(1) = data$
115    DO
120      INPUT #1: data$
125      LET b = POS(data$, "CONVERT")
130      LET config_old$(c_old) = data$
135      LET c_old = c_old + 1
140  LOOP WHILE b = 0
145  CLOSE #1

1    REM ADJUSTABLE SLEEP PERIOD
1    REM WITH AN INSTALLED RTC THIS COULD BE SET TO A SPECIFC TIME OR DATE

200  SLEEP 5

1    REM STORE 2ND CONFIG LABEL

300  LET config = 2
305  GOSUB 9600
310  LET config_new$(1) = data$
315    DO
320      INPUT #1: data$
325      LET b = POS(data$, "CONVERT")
330      LET config_new$(c_new) = data$
335      LET c_new = c_new + 1
340  LOOP WHILE b = 0
345  CLOSE #1

1    REM COMPARE CONFIG_OLD$ WITH CONFIG_NEW$

400  FOR n = 1 TO c_new - 1
405    IF config_old$(n) <> config_new$(n) THEN
410      GOSUB 9700  !Error notification
415    END IF
420  NEXT n

500  LET c_new = 2
505  GOTO 200  !Return to SLEEP

3000 END  !Abort program

1 REM *******  Start of SUB-ROUTINES  *******


9400 Print #0: "Error occured"
9499 return

1 REM SUBROUTINE FOR SETTING DEFAULT PARMETERS
```

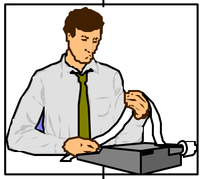
200 to 300 DPI Conversion for ZBI

200 to 300 dpi conversion

ZPL comes from the computer as a 200 dpi format



ZBI searches for all of the values behind ^FO, ^A0, and ^GB. These values define the size and the positions of fonts and boxes.



After the values of the selected commands has been found, they are then multiplied by 1.5 in order to convert them from a 200 dpi value to a 300 dpi value.

The converted values then inserted back into the ZPL and are sent to the ZPL engine for processing and printing

200 to 300 DPI Converter

```
1 REM THIS SECTION OF CODE OPENS THE PORTS FOR INPUT AND OUTPUT
10 CLOSE #1
20 CLOSE #2
30 OPEN #1 : NAME "PAR"
40 OPEN #2 : NAME "ZPL"

1 REM CREATE AN ARRAY WITH THE SEARCH PARAMETERS
50 DECLARE STRING FIND$(20)
60 LET FIND$(1) = "^FO"
70 LET FIND$(2) = "^A0"
80 LET FIND$(3) = "^GB"
90 LET FIND$(4) = "^XZ"
100 LET FIND$(5) = "FO"
110 LET FIND$(6) = "GB"

1 REM SEARCH FOR THE PARAMETERS
300 LET IN$ = SEARCHTO$(1,FIND$,2)

1 REM ONCE A PARAMETER IS FOUND THEN FIGURE OUT WHAT TO DO WITH IT
310 IF IN$ = "^FO" OR IN$ = "FO" THEN
320 GOTO 500
330 END IF
340 IF IN$ = "^A0" THEN
350 GOTO 700
360 END IF
370 IF IN$ = "^GB"OR IN$ = "GB" THEN
380 GOTO 900
390 END IF
400 IF IN$ = "^XZ" THEN
410 PRINT #2: IN$;
420 GOTO 300
430 END IF

1 REM THIS IS THE SECTION THAT CONVERTS THE ^FO FROM 200 TO 300
500 INBYTE #1: A$
510 LET A = ORD(A$)
520 IF A >= 65 THEN
530 PRINT #2: IN$&A$;
540 GOTO 300
550 END IF
560 LET X$ = EXTRACT$(1,"",",")
570 LET X2$ = A$&X$
580 LET Y$ = EXTRACT$(1,"","^")
590 LET X = VAL(X2$)
600 LET Y = VAL(Y$)
610 LET X2 = (X/2)+X
620 LET Y2 = (Y/2)+Y
630 PRINT #2: IN$;X2;",";Y2;"^";
640 GOTO 300


1 REM THIS IS THE SECTION THAT CONVERTS THE ^A0 FROM 200 TO 300
700 LET H$ = EXTRACT$(1,",",",")
```

```
710 LET W$ = EXTRACT$(1,"","^")
720 LET H = VAL(H$)
730 LET W = VAL(W$)
740 LET H2 = (H/2)+H
750 LET W2 = (W/2)+W
760 PRINT #2: IN$&"N,";H2;",";W2;"^";
770 GOTO 300

1 REM THIS IS THE SECTION THAT CONVERTS THE ^GB FROM 200 TO 300
900 LET W$ = EXTRACT$(1,"",",")
910 LET H$ = EXTRACT$(1,"",",")
920 LET T$ = EXTRACT$(1,"","^")
930 LET H = VAL(H$)
940 LET W = VAL(W$)
950 LET T = VAL(T$)
960 LET H2 = (H/2)+H
970 LET W2 = (W/2)+W
980 LET T2 = (T/2)+T
990 PRINT #2: IN$;W2;",";H2;",";T2;"^";
1000 GOTO 300
```