
Software Interface Document



&



Ink Jet System

5760-113 Rev AD

TABLE OF CONTENTS

| | |
|--|-----------|
| Section 1: Overview | 4 |
| Section 2: Details | 5 |
| Serial data for the IJ/3000 | 5 |
| HTTP Server and Cgi's | 6 |
| Uploading files..... | 7 |
| Network Message List Mode..... | 8 |
| Network Notification Mode | 8 |
| Section 3: Status Page Information..... | 9 |
| Section 4: Setup | 10 |
| Structure of setup.cfg documents..... | 10 |
| The Cfg element | 10 |
| Shift Element | 10 |
| Net Element..... | 11 |
| Rollover Element | 11 |
| Encoder Element | 11 |
| Daisychain Element | 12 |
| Com Element | 13 |
| Options Element..... | 13 |
| Section 5: Product Documents..... | 14 |
| The Product Element..... | 14 |
| Fields..... | 15 |
| Text fields | 15 |
| Time fields | 15 |
| Date fields..... | 16 |
| Variable fields | 16 |
| Count fields..... | 16 |
| Logo fields..... | 17 |
| ALP Fields/Templates | 17 |
| BarCode Fields (only for Impulse Jet)..... | 17 |

IJ3000 & IJ4000

| | |
|--|-----------|
| Block Fields | 19 |
| Sample | 20 |
| Screen Shots | 20 |
| File Content | 20 |
| Section 6: Alphacodes | 21 |
| The Alphacodes Element | 21 |
| Section 7: Automatic Cleaning Cycle | 23 |
| Structure of AcsEvents.cfg document | 23 |
| Section 8: Font File Format..... | 24 |
| Section 9: Visual Basic Example for Transferring a Product to the IJ3000/IJ4000 | 25 |
| Appendix A: Glossary | 28 |

Section 1: Overview

Feature overview of the Network Controller:

- Controller receives http requests on port 80.
 - * Server uses html and cgi scripts written in Lua¹.
 - * Contains built-in, non-modifiable cgi scripts: upload.cgi, upprnt1.cgi, upprnt2.cgi and sys.cgi.
 - * Serial port also uses Lua scripts.
 - * Files are sent using the upload.cgi script, which uses http post method for uploading files to the controller using Ethernet.
 - * Print buffers on the controllers can be obtained as files from the controller as "printbuf1" or "printbuf2."
- Capable of sending http requests (URL is configurable) to an http server to obtain product lists and retrieve products from that http server over the network.
- Sends notification of Bootup, Photocell and IDS via UDP (address and port configurable).
- DHCP is currently not included, only static IP addressing.
- Controller does not currently utilize DNS or a default Gateway for either Network notification or Message list access.
- Controller is currently not accessible via OPC.
- Active ports that the controller listens to are: UDP/1025 and TCP/80.

¹Lua Software: Copyright © 1994-2015 Lua.org, PUC-Rio. All rights reserved.

Section 2: Details

Serial data for the IJ/3000

Data received by the controller through a serial port can be any valid sequence of ASCII characters between the space " "(0x20) and the tilde "~" (0x7E). This sequence would be followed by a CRLF (CR=0x0D and LF=0x0A) by default. If an end marker other than CRLF is needed, it can be changed to another two-byte sequence within the setup.cfg file in the COM element.

The serial ports on the controller are capable of serving one of three functions.

The first mode, "Message Look Up/Scanner," will select the next message to be printed on the interface corresponding to the serial port in use, where COM1 will print on Task 1 and COM2 will print on Task 2. The data that is received from the serial port must equal the message name of the desired message to be printed. If there is not a message with a name that equals the data sent to the serial port then the controller will cancel any printing message for the task that corresponds to that serial port.

The second mode is "External input". This mode is used when a printing message contains variable fields with a data source corresponding to the serial port used. The data from the serial port will be printed in the location of that field on the next photocell trip.

The third mode is "Command & Control." This mode allows the execution of a script on the controller. Arguments to the scripts are defined similar to URL arguments that are passed to the web server. For example, the call to the serial script would appear as:

serial.cgi?idx=0&nme=var.prd&fst=1&lst=1&d1=xyz. This would execute the serial.cgi script and set the following name value pairs for the variables:

idx=0 (idx is the task number; 0 is task 1, 1 is task 2)

nme=var.prd (nme is the product name to change to)

fst=1 (fst is the index of the first data-type variable field in the range being specified)

lst=1 (lst is the index of the last data-type variable field in the range being specified)

d1=xyz (assigns "xyz" to data-type variable field #1)

Following is another example where the user is printing the message test.prd which contains two data-type variable fields on task 2:

serial.cgi?idx=1&nme=test.prd&fst=1&lst=2&d1=abc&d2=123

To set data variable fields without specifying a message to print, leave out the "idx=x" and the "nme=<message name>" arguments. For example:

<http://x.x.x.x/serial.cgi?fst=1&lst=2&d1=abc&d2=123&net=1>

If the data variable field data is to include a **space character**, or any of the symbols:

; / ? : @ & = + \$, < > # % “ { } | \ ^ [] ‘

it must be replaced by a percent sign (%), followed by the hexadecimal ASCII value of the character. For example, if the variable data is "SELL BY", the URL would be:

<http://x.x.x.x/serial.cgi?fst=1&lst=1&d1=SELL%20BY&net=1>

IJ3000 & IJ4000

The hexadecimal ASCII values for the above symbols are:

| <u>Symbol</u> | <u>Value</u> | <u>Symbol</u> | <u>Value</u> | <u>Symbol</u> | <u>Value</u> |
|---------------|--------------|---------------|--------------|---------------|--------------|
| (space) | 20 | , | 2c | @ | 40 |
| " | 22 | / | 2f | [| 5b |
| # | 23 | : | 3a | \ | 5c |
| \$ | 24 | ; | 3b |] | 5d |
| % | 25 | < | 3c | ^ | 5e |
| & | 26 | = | 3d | { | 7b |
| ' | 27 | > | 3e | | 7c |
| + | 2b | ? | 3f | } | 7d |

To execute the serial.cgi script from a device that supports sending data over a TCP socket, format the message as follows:

```
GET /serial.cgi?idx=1&nme=test.prd&fst=1&lst=2&dl=abc&d2=123&net=1 HTTP/1.1↵
Host: 10.1.2.3 ↵
↵
↵
```

NOTE: ↵ = Carriage Return Line Feed (0x0D 0x0A)

NOTE: .prd files cannot be uploaded to the controller using "Command & Control".

HTTP Server and Cgi's

Cgi's built into the system have the following functions:

- upload.cgi - Upload a file to the controller. (See logo upload in 5760-121 Operations Manual.)
- sys.cgi - Allows controller to re-read config files, or to reboot the controller.
- upprnt1.cgi and upprnt2.cgi - Upload a product and print it on interface 1 or 2. A page will be returned that indicates success or failure of the operation. The connection will be closed. Once the page has been returned, the uploaded print message will be waiting for a currently printing product, if any, to finish before changing to the new product.
- Use the print.cgi to cancel printing.

Uploading files

If you wish to simultaneously upload and print a product, call the upprnt1.cgi or upprnt2.cgi. (The upprnt1.cgi will print on Task 1 and upprnt2.cgi will print on Task 2.) An example of this is below. Using a telnet application of your choice, input the following after connecting to the controller on port 80. (Details for uploading files can be found in rfc2616 - for http; rfc2854 and rfc2046 - form-based file upload.)

For details on any other method of connecting to the controller, use a network analyzer and a web browser to connect to the desired page.

If the system is using password protection, the authentication will also have to be included in the http header (i.e., Authorization: Basic dxNlcjpEaWFncmFwaA==) before the Content-type. The string following Basic is a base 64 encoded string of "user:password"; for this example, the password of "Diagraph" was used. (See RFC 3548.) The user name will always be "user"; the password will correspond to what is set on the controller. The authentication string is not required when not using password protection.

```
POST /upprnt1.cgi HTTP/1.1 ↴
Host: 10.1.2.3 ↴
Content-type: multipart/form-data; boundary=bound ↴
Content-Length: 241 ↴
↪
↪
bound ↴
Content-Disposition: form-data; name="fname"; filename="test.prd" ↴
Content-Type: application/octet-stream ↴
↪
<product len='1200' charwidth='5' name='test'>
<text cspc='1' txt='DIAGRAPH IJ3000' fnt='9b.fnt' />
</product> ↴
↪
bound-- ↴
```

NOTE: ↴ = Carriage Return Line Feed (0x0D 0x0A)

If everything was typed in correctly the following response would be sent from the controller:

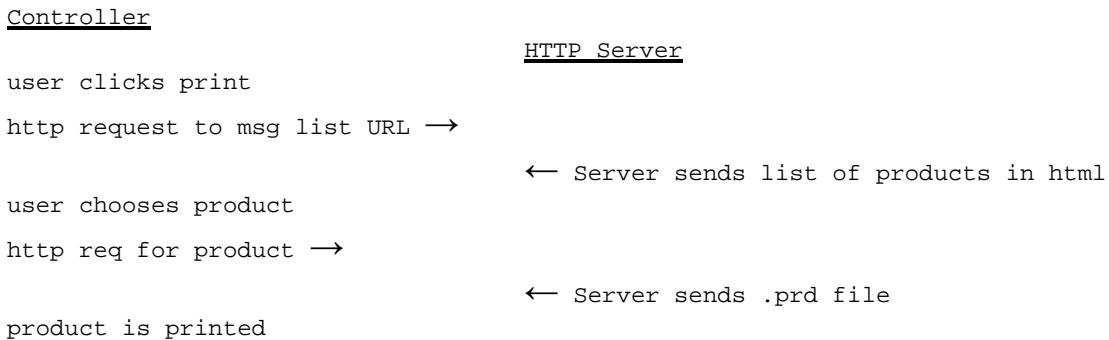
```
HTTP/1.0 200 OK
Date: Sat, 21 Dec 1996 12:00:00 GMT
Content-type: text/html
Content-length: 103

<html><body bgcolor='white'><img src='logo.gif'><p><font
face='arial'>File print
ed</font></body></html>
```

Network Message List Mode

When configured for network mode, the controller sends a request to the specified URL. The response from this request is then parsed for .prd references. If an http server returned the html test.prd, the file selection dialog would display "test". If selected for print, the controller would request the test.prd product from the http server. This product does not have to be a static file on the web server, nor does the list that was used. The list and the product itself could be dynamically generated from the http server.

Example:



Network Notification Mode

If the controller is configured for "Network notification," it will send notification through UDP to the address and port that is specified in the configuration of the system. If an interface is not present, the controller will not send notifications for that interface. The information sent will be in this format:

```
ifc=0 ↴  
cnt=53467 ↴  
prntng=test.prd ↴  
encspd=247 ↴  
state=100 ↴
```

NOTE: ↴ = Line Feed (0x0A)

Format Definitions:

- ifc - interface (task) that the information corresponds to.
 - cnt - is the current product count for that interface.
 - prntng - name of the current printing product for this interface.
 - encspd - encoder speed in feet per/min.
 - state - event that triggered the notification. Current values are: BOOT=1, PHOTOTRIP=100, IDS_STATUS=200

Section 3: Status Page Information

The status information that is returned from either an ethernet page request (status.html) or from command and control serial connection (status.cgi) contains information about the controller. It is presented as html and some of the data is also duplicated as html comments. These html comments are represented as tab delimited data sequences. The html contents can be seen by loading the html page. The following describes the contents of the tab delimited sequences.

Interface card/task information:

```
<!-- ifcX d1 d2 d3 d4 d5 d6 d7 s8 -->
X - task number
d1 - task type
d2 - fpga status
d3 - photocell detect
d4 - line speed
d5 - non-printable count value
d6 - pause status
d7 - impulse status value
    bit0 - At temp
    bit1 - Waste full
    bit2 - Sleep on/off
    bit3 - Ink out
    bit4 - Ink Low
    bit5 - Pump on
    bit6 - Vacuum on
    bit7 - HV
    bit8 - Temp1
    bit9 - Temp2
    bit10 - Ink out 1
    bit11 - Ink out 2
    bit12 - Ink low 1
    bit13 - Ink low 2
    bit14 - HV 1
    bit15 - HV 2
s8 - message name
```

Integrated valve IDS information:

```
<!-- idsX d1 d2 d3 d4 d5 d6 -->
X - task number
d1 - Major version
d2 - Minor version
d3 - Ink status
d4 - Pressure
d5 - Vacuum
d6 - Broken Line
```

Section 4: Setup

NOTE: "sys.cgi?state=config" should be executed to have system re-read the setup.cgi file.

Structure of setup.cfg documents

```
<cfg>
...
<cfg>
```



```
<!ENTITY (%cfg.content) "(%net | %shift | %rollover | %daisychain
                           | %encoder | %com)">
<!ENTITY % num "CDATA"
-- a number specification: 012345 -- >
<!ENTITY % short "%num"
-- a number: 2 bytes wide (0..65535) -- >
<!ENTITY % char "%num"
-- a number: 1 bytes wide (0..255) -- >
<!ENTITY % long "%num"
-- a number: 4 bytes wide (0..0xffffffff) -- >
<!ENTITY % bool "%num"
-- a number: 1 bytes wide (0/1) -- >

<!ELEMENT CFG O O %cfg.content>
<!ATTLIST CFG
    lan CDATA          -- language to be used for unit (Default is
                         english.lan) --
    units %short        -- unit of measure for unit --
    mil %bool           -- 24 hour or 12 hour clock --
    usepwd %num          -- password protection --
    passwd CDATA         -- password to be used for password protection--
    grps %bool           -- use task grouping --
    save_cnts %bool      -- save counts dialog when cancelling print --
    >
```

Example:

```
<cfg lan="1" units="0" mil="0" usepwd="0" passwd="Diagraph">
```

Shift Element

```
<!ELEMENT SHIFT O O >
<!ATTLIST SHIFT
    num char          -- 4 of these are allowed --
    hh   CDATA         -- number of shift 0 through 3 --
    mm   CDATA         -- hours 0 through 23 --
    mm   CDATA         -- minutes 0 through 59 --
    code CDATA         -- up to 4 character designation --
    >
```

Example:

```
<shift num="0" hh="2" mm="4" code="ABCD" />
```

Net Element

```
<!ELEMENT NET O O >
<!ATTLIST NET
    ip CDATA          -- dotted ip address --
    ids1 CDATA        -- dotted first ids address --
    ids2 CDATA        -- dotted second ids address --
    sub CDATA         -- dotted subnet mask --
    acc short         -- 1 = local, 2 = network retrieve --
    map CDATA         -- list from map location --
    notify CDATA      -- url to retrieve network products location --
    alppmap CDATA     -- where to send notification of photocell trip --
    >
```

Example:

```
<net ip="10.1.2.4" ids1="10.1.2.2" ids2="0.0.0.0"
      sub="255.255.255.0" acc="1"
      map="http://10.1.2.5/prds.cgi" notify="udp://10.1.2.7:2048"/>
```

Rollover Element

```
<!ELEMENT ROLLOVER O O >
<!ATTLIST ROLLOVER
    hh   CDATA          -- hours 0 through 23--
    mm   CDATA          -- minutes 0 through 59 --
    >
```

Example:

```
<rollover hh='0' mm='0' />
```

Encoder Element

```
<!ELEMENT OPTIONS O O>
<!ATTLIST OPTIONS
    iface %char          -- 2 of these are allowed --
    ext   CDATA          -- number of interface 0 through 1 --
    speed %short         -- external(0)/internal(2) --
    share %char          -- line speed for internal encoder ft/min. --
    share CIDS           -- share = share encoder + share photocell + share
                           CIDS --
    minspeed %short      -- share encoder: no=0, yes=1 --
                           -- share photocell: no=0, yes=2 --
                           -- share CIDS: yes=0, no=4 --
                           -- minimum speed for photocell trip --
```

Example:

```
<encoder iface='1' ext='0' share='0' />
```

Daisychain Element

```
<!ELEMENT DAISYCHAIN O O (%ph) >
<!ATTLIST DAISYCHAIN
  iface char          -- 2 of these are allowed --
                      -- number of interface 0 through 1 --
  >
<!ELEMENT PH O O >
<!ATTLIST PH
  num char           -- 8 of these are allowed --
  style long          -- number of printhead --
  id char            -- type of printhead --
  offset short         -- printhead id --
                      -- printhead offset --
                      -- photocell to printhead distance,
                      -- in hundredths of an inch.
  >
```

Example:

```
<daisychain iface='0'>
<ph num='1' style='4113' id='1' offset='0' />
<ph num='2' style='4113' id='2' offset='0' />
<ph num='3' style='17' id='10' offset='0' />
<ph num='4' style='17' id='11' offset='0' />
</daisychain>
```

| Print Head Style Number System | | | |
|--|-------------|---------------------------|-------------|
| For I.V. Type Print Heads | | For I.J. Type Print Heads | |
| Start at: | 16 | Start at: | 32 |
| If Print Head is: | Add: | If Print Head is: | Add: |
| 9-Dot 1/2" | 1 | IJ352 | 8 |
| 9-Dot 7/8" | 2 | IJ768 | 64 |
| 18-Dot 1" | 4 | IJ384 | 1,024 |
| 18-Dot 2" | 8 | NP384 | 512 |
| | | IJ96, 3/4" | 32,769 |
| For Serial Type Print Heads | | IJ96, 1-1/2" | 16,385 |
| Start at: | 0 | IJ96, 2" | 8,193 |
| If Print Head is: | Add: | IJ192, 1" | 32,770 |
| TJ500 | 1 | IJ192, 1-1/2" | 16,386 |
| TJ1000 | 2 | IJ192, 2" | 8,194 |
| IV12 | 4 | IJ224, 3/4" | 32,772 |
| | | IJ224, 1-1/2" | 16,388 |
| | | IJ224, 2" | 8,196 |
| If Non-ACS Head (IJ Print Heads only), add: | | | 256 |
| If ScanTrue II Head, add: | | | 2048 |
| If Bulk Ink Print Head (TJ Print Heads only), add: | | | 2048 |
| If printing reverse (product moves left to right past Print Head), add: | | | 4096 |

IJ3000 & IJ4000

Examples:

The style number for a 9-Dot 1/2" I.V. type Print Head printing reverse is:

16 + 1 + 4096 = 4113.

The style number for an IJ352 I.J. Print Head printing forward is:

32 + 8 = 40.

Com Element

```
<!ELEMENT COM O O >
<!ATTLIST COM
    port char          -- 2 of these are allowed --
    baud CDATA        -- number of COM 0 through 1 --
    device short      -- baud rate 2400, 4800, 9600, 19200, 38400, 57600 --
    device short      -- 50, 100, 200, 302 (only com 1 can have a 302), 303, 304, 400,
                        500, 501 --
    echo % bool       -- indicates whether characters should be echoed back --
    endmrk short     -- specifies the end of the data marker designated as a decimal
                        value. The value can be obtained by multiplying the first number
                        by 256, then adding the second number to it [i.e., CR = 13, LF =
                        10, so (13*256)+10 = 3338] -
    bmpdload % bool   -- auto download of bmps to serial print head off=0, on=1 -->
```

Example:

```
<com port='0' baud='57600' device='100' />
```

Options Element

```
<!ELEMENT OPTIONS O O>
<!ATTLIST OPTIONS
    iface %char         -- 2 of these are allowed --
    IDSdet %bool        -- number of interface 0 through 1 --
    pud %bool           -- detect CIDS: no=0, yes=1 --
    extPH %bool         -- print task upside down: no=0, yes=1 --
    IDStype %char       -- external photocell: no=0, yes=1 --
    >
```

Example:

```
<options iface='1' IDSdet='1' />
```

Section 5: Product Documents

Structure of Product Documents

For Single Task Product:

```
<product>
... product body
</product>
```

For Grouped Products:

```
<group>
    <product>
... product body
</product>
    <product>
... product body
</product>
</group>
```

The Product Element

```
<!ENTITY (%field-list.content ) "(%textfield | %timefield | %datefield | %barcode
|
%block | %countfield | %varfield | %logofield)">
<!ENTITY %num "CDATA" -- a number specification: 012345 -- >
<!ENTITY %short "%num" -- a number: 2 bytes wide (0..65535) -- >
<!ENTITY %char "%num" -- a number: 1 bytes wide (0..255) -- >
<!ENTITY %long "%num" -- a number: 4 bytes wide (0..0xffffffff) -- >
<!ENTITY %bool "%num" -- a number: 1 bytes wide (0/1) -- >
<!ENTITY %byte "%num" -- a number: 1 bytes wide (0..255) -- >
<!ENTITY %hspacing %short -- a number: 25, 33, 50, 66, 75, 100, 150, 200, 300,
400, 500, 600 -- >
<!ELEMENT PRODUCT O O %field-list.content "%alp">
<!ATTLIST PRODUCT
    len %short          -- length of the product, encoder ticks --
    margin %short        -- margin of front side (UOM is 1/100 in.) --
    margin2 %short       -- margin of back side (UOM is 1/100 in.) --
    charwidth %char      -- character width of product (4 = 25 dpi, 5 = 20
                           dpi...) --
    prntonce %bool       -- turn on prnt_n (below), or if no prnt_n print
                           once then remove from printing --
    prnt_n %long         -- print n times then remove from printing
                           (0 = prompt for n, 000001-999999 = print
                           quantity), prntonce must = 1 to use prnt_n --
    contprnt %bool        -- print continuously --
    mirror %bool          -- duplicate the front side on the back side --
    name CDATA            -- internal name of the product --
    hspc %hspacing         -- default product wide horizontal spacing --
```

IJ3000 & IJ4000

```
cspc %byte          -- character spacing, columns between characters --
prntupsdn %bool    -- print message upside down: no=0, yes=1 --
">>
```

Examples: <product len='1200' charwidth='5' cspc='2' name='test'>
<product len='4800' margin='300' prntonce='1', prnt_n='002500'
charwidth='3' hspc='200' name='Count'>

Fields

```
<!ENTITY ( %field-attrs )  "
indent %short        -- indent value (x coordinate) UOM is print columns (0..8192) --
startdot %short      -- starting dot (y coordinate) UOM is dots (0..71)--
vflip %bool          -- upside down --
cspc %short          -- character spacing UOM is print columns (0..25) --
                      -- for PEL, (6..150) in multiples of 6 --
draft %bool          -- draft mode print, prints every other column --
hspc %hspacing        -- field horizontal spacing --
">>
```

Text fields

```
<!ELEMENT TEXT O O >
<!ENTITY ( %txt-field-attr )  "
  txt CDATA           -- textual data for field --
  fnt CDATA           -- font name for field --
  %field-attr          -- indent, startdot, vflip --
">>
```

```
<!ATTLIST TEXT %txt-field-attr >
```

Example:

```
<text indent='0' startDot='0' vflip='0' ljust='0' cspc='1'
txt='test' fnt='9b.fnt' />
```

Time fields

```
<!ELEMENT TIME O O >
<!ATTLIST TIME
  fmt %long            -- format value (0..7) --
  %txt-field-attr       -- txt, fnt, indent, startdot, vflip --
">>
```

Example:

```
<time fmt='4' indent='0' startDot='9' vflip='0' ljust='0'
cspc='1' txt='11:42PM' fnt='9b.fnt' />
```

IJ3000 & IJ4000

Date fields

```
<!ELEMENT DATE O O >
<!ATTLIST DATE
    fmt %long          -- format value (0..14) --
    offset %num         -- format value --
    %txt-field-attr     -- txt, fnt, indent, startdot, vflip --
>
```

Example:

```
<date fmt='5' offset='0' indent='68' startDot='9' vflip='0'
ljust='0' cspc='1' txt='12/04/02' fnt='9b.fnt'/>
```

Variable fields

```
<!ELEMENT VAR O O >
<!ATTLIST VAR
    length %long          -- length of field in 1/100 in. (0..8192) --
    prompt CDATA           -- prompt that is displayed when printed --
    src %char              -- indicates type of variable field --
    %txt-field-attr        -- txt, fnt, indent, startdot, vflip --
    align %byte             -- if 1, then center text within length (only applies to label
                               templates) --
>
```

Example:

```
<var length='100' prompt='' src='1' indent='218' startDot='9' vflip='0'
hflip='1' ljust='0' cspc='1' txt='XXXXXXXXXXXX' fnt='9b.fnt'/>
```

Count fields

```
<!ELEMENT COUNT O O >
<!ATTLIST COUNT
    cnt %long          -- count value (Should be padded with enough spaces
                           for maximum length value.) --
    start %long         -- start value --
    stop %long          -- stop value --
    inc %long           -- increment value --
    zeros %bool          -- indicates whether to prefill with zeros (0/1) --
    pallet %long         -- pallet count --
    psize %long          -- pallet size --
    picnt %long          -- pallet increment (Should be padded with
                           enough spaces      for maximum length value.)
                           --
    %txt-field-attr      -- txt, fnt, indent, startdot, vflip --
>
```

IJ3000 & IJ4000

Example:

```
<count cnt='1'          ' start='1' stop='999999' inc='1' zeros='1'
pallet='0' psiz='50' picnt='1'          ' indent='152' startDot='9'
vflip='0' ljust='0' cspc='1' txt='000001' fnt='9b.fnt' />
```

Logo fields

```
<!ELEMENT LOGO O O >
<!ATTLIST LOGO
  bmp CDATA          -- bmp file name --
  %field-attr         -- indent, startdot, vflip --
  >
```

Example:

```
<logo bmp='left.bmp' indent='0' startDot='0' vflip='0' />
```

ALP Fields/Templates

For Label Templates:

```
<alp>
....alp body
</alp>
<!ELEMENT ALP O O %field-list.content>
<!ATTLIST ALP
  src CDATA          -- file name --
  data CDATA          -- data --
  -- The following are only for ALP Templates: --
  hght %long          -- height in inches --
  wid %long          -- width in inches --
  rot %short          -- 0 or 180 --
  spd %byte           -- speed inches/sec; SATO: 4, 6, 8, 10 or 12 --
  drk %byte           -- darkness of the print; SATO: 1, 2 or 3 --
  >
```

BarCode Fields (only for Impulse Jet)

```
<!ELEMENT BARCODE O O >
<!ATTLIST BARCODE
  str CDATA          -- text --
  type %long          -- (1) UPCA --
                      -- (2) UPCE --
                      -- (3) EAN13 --
                      -- (4) EAN8 --
                      -- (5) CODE 39 --
                      -- (7) I 2 of 5 --
                      -- (8) CODE 128 --
```

IJ3000 & IJ4000

```
-- (9) GS1 DATA MATRIX --
-- (11) DATA MATRIX --
-- (12) QR CODE --
-- (14) SCC-14 UCC 128 --
-- (263) SCC-14 I 2 of 5 --
nbars %byte          -- For backwards compatibility only, Version 3.x or
                       -- earlier --
mil %byte            -- wide bar mil setting -- in increments of 5
human %bool           -- show human readable (0/1) --
fnt CDATA             -- font for human readable --
ecl %byte              -- (0) Low -- QR code error correction level
                         (1) Medium --
                         (2) Quartile --
                         (3) High --
hgt %byte            -- height --
bbhgt %byte           -- bearer bar height --
bbwid %byte            -- bearer bar width --
bleed %byte           -- bleed factor in 1/2 column increments --
src %char              -- (0) Fixed --
                         (1) Prompt --
                         (2) COM1 --
                         (3) COM2 --
                         (4) Database --
                         (5) Data variable field, index is in the upper 4 bits --
                         (6) Formula, uses prompt --
                         (7) Database formula --

prompt CDATA           -- prompt to show to the user --
                       -- For the formula source option prompt will consist of
                         following parameters --
-- Example: %o/%d/%c will print "10/31/2008" --
-- offset in days for dates is included between % and
                         character: --
-- if today is Jun 5 %30o/%30d will print "07/05" --
-- %% a literal % --
-- %A User Day --
-- %b Month MON --
-- %B User Month --
-- %c 4 digit Year --
-- %C User year --
-- %d Day of month --
-- %D User day of month --
-- %f Shift Code --
-- %g Date MDDMYY --
-- %G Date MDDMY --
-- %h Hour --
-- %H User Hour --
-- %j Julian day (001..366) --
-- %J Julian (AA-OB) -
```

IJ3000 & IJ4000

```
-- %k Date YYMMDD --
-- %m Minutes --
-- %M User minutes --
-- %o month MM --
-- %n Month M (A..L) --
-- %N Month M no I --
-- %p AM PM --
-- %P AM=1, PM=2 --
-- %q Quarter Hour code -
-- %r Reverse Julian date --
-- %s Seconds SS -
-- %S Incrementing count --
-- %w Week 2 digit --
-- %W User week --
-- %u User day --
-- %v Week of: "... YY" --
-- %V Week of: "... YYYY" -
-- %X User prompted variable data, uses usrtxt --
-- %y Year 1 digit --
-- %Y Year 2 digit -
-- For Incrementing Count (%S), the number of digits in the
-- count is included between % and S -
-- Example: %0001S will print a 4-digit count -
-- For User Prompted Variable Data (%X), the number of
-- characters of variable data accepted is included between
-- $ and X -
-- Example: %8X will print up to 8 characters of variable data.
-- The number of characters may be 1 to 99 -
-- Prompt to show to the user for requesting variable data (%X)
-- add fnc1 + mod10 checkdigit to code 128 --

usrtxt CDATA
opt %byte

%field-attr

>
```

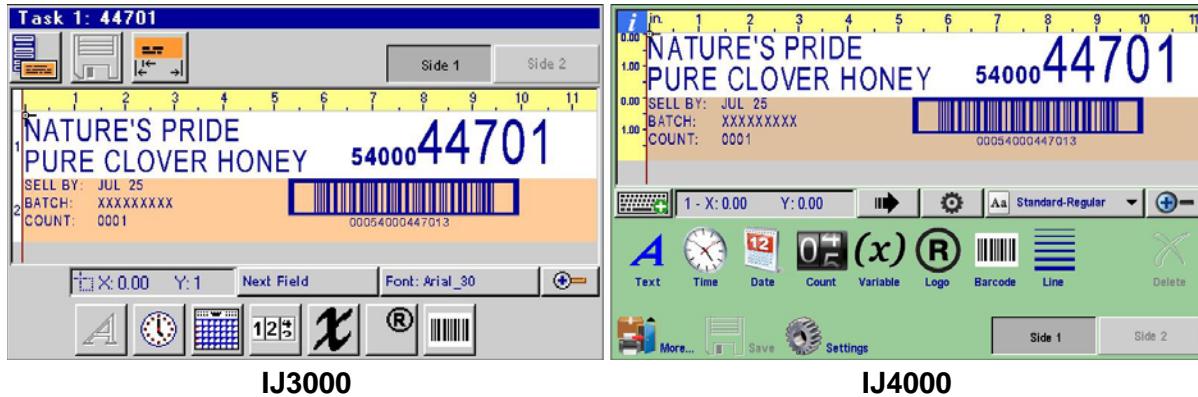
Block Fields

```
<!ELEMENT BLOCK OO >
<!ATTLIST BLOCK
    height %long          -- block height in dots --
    width %long           -- block width in inches or dots for ALP --
    %field-attr
    >
```

IJ3000 & IJ4000

Sample

Screen Shots



File Content

```
<product len='8400' charwidth='3' hspc='150' name='44701'>
<text indent='0' startDot='3' cspc='6' fnt='Arial_63.gfnt'>
NATURE'S PRIDE
</text>
<text indent='0' startDot='66' cspc='6' fnt='Arial_63.gfnt'>
PURE CLOVER HONEY
</text>
<text indent='1314' startDot='63' cspc='6' fnt='Arial_48_Bold.gfnt'>
54000
</text>
<text indent='1584' startDot='0' cspc='6' fnt='Arial_126.gfnt'>
44701
</text>
<text indent='0' startDot='129' cspc='6' fnt='Arial_30.gfnt'>
SELL BY:
</text>
<text indent='0' startDot='165' cspc='6' fnt='Arial_30.gfnt'>
BATCH:
</text>
<text indent='0' startDot='201' cspc='6' fnt='Arial_30.gfnt'>
COUNT:
</text>
<barcode mil='30' str='00054000447013' type='7' bleed='2' human='1'
fnt='Arial_24.gfnt' hgt='76' bbhgt='8' bbwid='24' indent='1062' startDot='129'
cspc='0' />
<date fmt='6' offset='60' type='2' indent='294' startDot='129' cspc='6'
fnt='Arial_30.gfnt' txt='JUL' />
<date fmt='0' offset='60' type='2' indent='420' startDot='129' cspc='6'
fnt='Arial_30.gfnt' txt='25' />
<var length='990' prompt='BATCH' src='0' indent='294' startDot='165' cspc='6'
fnt='Arial_30.gfnt' txt='XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX' />
<count cnt='0001' start='1' stop='9999' inc='1' zeros='1' pallet='0' psize='50'
picnt='0001' indent='294' startDot='201' cspc='6' fnt='Arial_30.gfnt' txt='0001' />
</product>
```

Section 6: Alphacodes

Structure of alphacodes.cfg document

```
<alphacodes>
... alphacodes body
</alphacodes>
```

The Alphacodes Element

```
<!ENTITY (%alpha.content) "(%hour | %minute | %day | %date
| %week | %month)">

<!ELEMENT ALPHACODES O O %alpha.content>

<!ELEMENT HOUR O O >
<!ATTLIST HOUR
    h00 CDATA          -- hour 0, 4 character allowed data --
    ...
    h23 CDATA          -- hour 23, 4 character allowed data --
>
```

Hour Example:

```
<hour h00='A' h01='B' h02='C' h03='D' h04='E' h05='F' h06='G' h07='H'
      h08='J' h09='K' h10='L' h11='M' h12='N' h13='P' h14='Q' h15='R' h16='S'
      h17='T' h18='U' h19='V' h20='W' h21='X' h22='Y' h23='Z' />
```

```
<!ELEMENT MINUTE O O >
<!ATTLIST MINUTE
    m00 CDATA          -- minute 0, 4 character allowed data --
    ...
    m59 CDATA          -- minute 23, 4 character allowed data --
>
```

Minute Example:

```
<minute m00='AA' m01='AB' m02='AC' m03='AD' m04='AE' m05='AF' m06='AG'
        m07='AH' m08='AJ' m09='AK' m10='AL' m11='AM' m12='AN' m13='AP' m14='AQ'
        m15='AR' m16='AS' m17='AT' m18='AU' m19='AV' m20='AW' m21='AX' m22='AY'
        m23='AZ' m24='BA' m25='BB' m26='BC' m27='BD' m28='BE' m29='BF' m30='BG'
        m31='BH' m32='BJ' m33='BK' m34='BL' m35='BM' m36='BN' m37='BP' m38='BQ'
        m39='BR' m40='BS' m41='BT' m42='BU' m43='BV' m44='BW' m45='BX' m46='BY'
        m47='BZ' m48='CA' m49='CB' m50='CC' m51='CD' m52='CE' m53='CF' m54='CG'
        m55='CH' m56='CJ' m57='CK' m58='CL' m59='CM' />
```

```
<!ELEMENT DAY O O >
<!ATTLIST DAY
    d01 CDATA -- day 1 of the week, 4 character allowed data --
    ...
    d7 CDATA -- day 7 of the week, 4 character allowed data --
>
```

IJ3000 & IJ4000

Day Example:

```
<day d1='SUN' d2='MON' d3='TUE' d4='WED' d5='THU' d6='FRI' d7='SAT' />
```

```
<!ELEMENT DATE O O >
<!ATTLIST DATE
    d01 CDATA -- date 1, 4 character allowed data --
    ...
    d31 CDATA -- date 31, 4 character allowed data --
>
```

Date Example:

```
<date d01='AA' d02='AB' d03='AC' d04='AD' d05='AE' d06='AF' d07='AG'
      d08='AH' d09='AJ' d10='AK' d11='AL' d12='AM' d13='AN' d14='AP' d15='AQ'
      d16='AR' d17='AS' d18='AT' d19='AU' d20='AV' d21='AW' d22='AX' d23='AY'
      d24='AZ' d25='BA' d26='BB' d27='BC' d28='BD' d29='BE' d30='BF' d31='BG' /<
```

```
<!ELEMENT WEEK O O >
<!ATTLIST WEEK
    w01 CDATA -- week 1, 4 character allowed data --
    ...
    w53 CDATA -- week 53, 4 character allowed data --
>
```

Week Example:

```
<week w01='AA' w02='AB' w03='AC' w04='AD' w05='AE' w06='AF' w07='AG'
      w08='AH' w09='AJ' w10='AK' w11='AL' w12='AM' w13='AN' w14='AP' w15='AQ'
      w16='AR' w17='AS' w18='AT' w19='AU' w20='AV' w21='AW' w22='AX' w23='AY'
      w24='AZ' w25='BA' w26='BB' w27='BC' w28='BD' w29='BE' w30='BF' w31='BG'
      w32='BH' w33='BJ' w34='BK' w35='BL' w36='BM' w37='BN' w38='BP' w39='BQ'
      w40='BR' w41='BS' w42='BT' w43='BU' w44='BV' w45='BW' w46='BX' w47='BY'
      w48='BZ' w49='CA' w50='CB' w51='CC' w52='CD' w53='CE' />
```

```
<!ELEMENT MONTH O O >
<!ATTLIST MONTH
    m01 CDATA -- month 1, 4 character allowed data --
    ...
    m12 CDATA -- month 12, 4 character allowed data --
>
```

Month Example:

```
<month m01='JAN' m02='FEB' m03='MAR' m04='APR' m05='MAY' m06='JUN'
      m07='JUL' m08='AUG' m09='SEP' m10='OCT' m11='NOV' m12='DEC' />
```

Section 7: Automatic Cleaning Cycle

Structure of AcsEvents.cfg document

```
<amsevents>
... acs body
</amsevents>

<!ENTITY (%acsevent.content) "($event)" >
<!ENTITY ACSEVENT %acsevent.content >
<!ATTLIST ACSEVENT onoff % bool
    ops % bool
    ihour % byte
    imin % byte
>

<!ELEMENT EVENT O O >
<!ATTLIST EVENT day % byte
    hour % byte
    minute % byte
>
```

Section 8: Font File Format

Following is a description of the font file format for printable fonts used with the IJ3000 Controller:

Header Section

| |
|----------------------------|
| short version |
| short dots |
| unsigned short start index |
| unsigned short end index |

Index Table Section (unsigned long 32 bits wide)

| |
|--|
| col[0+start] |
| col[1+start] + col[0+start] |
| col[1+start] + col[0+start] + ... + col[...] |
| ... |
| ... + col[end-1] |
| ... + col[end-1] + col[end] |

Note that the last column is not included as an offset, but for reference of the length of the last col[end].

Data Section

| |
|------------------------------|
| bits for columns for 0+start |
| bits for columns for 1+start |
| bits for columns for ... |
| bits for columns for end-1 |

The index table contains the offset into the data section for a particular character. That character must be within the range of values specified by start/end indices.

Section 9: Visual Basic Example for Transferring a Product to the IJ3000/IJ4000

NOTE: Inet1 is msinet.ocx, which needs to be added to the project.

```
Dim bWaiting As Boolean
Dim sHdr As String, sBoundary As String, sEndBoundary As String

Private Sub cmdRun_Click()
    Dim sTxt As String
    If Not bWaiting Then
        Screen.MousePointer = vbHourglass
        bWaiting = True
        sTxt = getFormData(Text1.Text)
        Execute txtIPAddr.Text, sTxt
    End If
End Sub

Private Sub cmdPrtBuf_Click()
    Dim url As String, hdr As String
    Dim ipaddr As String
    ipaddr = txtIPAddr.Text
    hdr = "Host: " + ipaddr + vbCrLf
    url = "http://" + ipaddr + "/printbuf1"
    Inet1.Execute url, "GET"
End Sub

Private Sub Inet1_StateChanged(ByVal State As Integer)
    Select Case State
        Case icResponseCompleted ' 12
            bWaiting = False
            Screen.MousePointer = vbNormal
            Dim vtData As Variant ' Data variable.
            Dim strData As String: strData = ""
            Dim bDone As Boolean: bDone = False
            ' Get first chunk.
            vtData = Inet1.GetChunk(1024, icString)
            DoEvents
            Do While Not bDone
                strData = strData & vtData
                DoEvents
                ' Get next chunk.
                vtData = Inet1.GetChunk(1024, icString)
                If Len(vtData) = 0 Then
                    bDone = True
                End If
            Loop
            MsgBox StripHTML(strData)
    End Select
End Sub
```

IJ3000 & IJ4000

```
Private Function StripHTML(txt As String)
    Dim buf As String
    Dim i As Integer, j As Integer
    buf = txt
    i = 1
    i = InStr(i, txt, "<")

    While i > 0
        j = InStr(i, buf, ">")
        If CBool(j) Then
            buf = Mid(buf, 1, i - 1) + Mid(buf, j + 1)
        End If
        i = InStr(i, buf, "<")
    Wend
    StripHTML = buf
End Function

Private Sub Form_Load()
    bWaiting = False
    Left = (Screen.Width - Width) / 2
    Top = (Screen.Height - Height) / 2

    Dim bound As String
    bound = "XXXXXXXXXXXXXXXXXXXXXX"
    sBoundary = vbCrLf + bound + vbCrLf
    sEndBoundary = vbCrLf + bound + "--" + vbCrLf
    sHdr = "Content-type: multipart/form-data; boundary=" + bound + vbCrLf
End Sub

Private Function getProduct(txt As String) As String
    Dim sProd As String
    sProd = "<product len='1200' charwidth='5' name='Simple Test'>" +
vbCrLf
    sProd = sProd + "<text indent='0' startDot='0' cspc='1' txt='"
    sProd = sProd + txt + "' fnt='9b.fnt'/'>" + vbCrLf + "</product>" +
vbCrLf
    getProduct = sProd
End Function

Private Function getFormData(txt As String) As String
    Dim sTxt As String, name As String
    name = "Simple test"
    sTxt = sBoundary + "Content-Disposition: form-data; "
    sTxt = sTxt + "name=" + Chr(34) + "fname" + Chr(34) + "; "
    sTxt = sTxt + "filename=" + Chr(34) + name + ".prd" + Chr(34) + vbCrLf
    sTxt = sTxt + "Content-Type: application/octet-stream" + vbCrLf +
vbCrLf
    sTxt = sTxt + getProduct(txt) + sEndBoundary
    getFormData = sTxt
End Function
```

IJ3000 & IJ4000

```
Public Function Execute(ipaddr As String, sTxt As String)
    Dim url As String, hdr As String
    hdr = "Host: " + ipaddr + vbCrLf + sHdr
    url = "http://" + ipaddr + "/upload.cgi"
    Inet1.Execute url, "POST", sTxt, hdr
End Function
```

Appendix A: Glossary

CGI - Common Gateway Interface, a standard for http server dynamic scripting see <http://www.w3.org/CGI/>.

DHCP - Dynamic Host Configuration Protocol, rfc 2131.

DNS - Domain Name Server, rfc 1035.

HTML - Hypertext Markup Language, several different standards dependent on the client, see www.w3.org.

HTTP - HyperText Transfer Protocol, rfc 2616.

IP - Internet Protocol, rfc 791.

LUA - Scripting language.

MIME - Multipurpose Internet Mail Extensions, rfc 822, 2045-2049.

Network Analyzer - This is a program used to view network traffic. For Unix, tcpdump (usually included) can be used. For MS Windows® (<http://www.wireshark.org>) can be used.

Port - A number used to distinguish among multiple destinations within a given host computer.

RFC - Request For Comment, see <http://www.rfc-editor.org>.

Socket - A unique identifier to or from which information is transmitted in a network, rfc 147.

UDP - User Datagram Protocol, rfc 768.

URL - Uniform Resource Locator, rfc 2396.

IJ3000 & IJ4000

Revision History