

Vega Terminal Mode Communications Guide

Terminal Mode API Version: 1.0.1.0

Document No: IJM061

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| REVISIONS | | | | | |
| Ltr | Description | Init | Date | DCN # |
| - | Initial Issue API version 1.0.0.0 | RHB | 9/25/23 |  |
| A | API version 1.0.1.0  Add command for IDS Enable for iM2 0xB9 IM2\_4\_IDS\_ENABLE  Add Ink empty status for IM2 response to the GET\_INK\_DATA command INK\_DATA\_REPLY [Ink\_empty\_alarm 1=active (out of ink); 0=inactive (still have ink left)]  Add support for using a single bi-directional port for TCP/IP communications. The port is 65100 for the Vega application server to accept connections. | RHP | 4/18/24 |  |
| B | API Version 1.0.1.0  Added appendix for management functions / messages.  Made typographical edits | MPM | 5/8/24 |  |

Document Revision Status:

Reference Documents (Latest Revisions Thereof):

|  |  |  |  |
| --- | --- | --- | --- |
| REFERENCES | | | |
| Ltr | Doc No | Title | Controller/Originator |
| a |  |  | Norwix |
| b |  |  | Norwix |

Table of Contents

[1 Communications Overview 14](#_Toc166055592)

[2 VEGA Remote Host Interface (Terminal Modes) Overview 15](#_Toc166055593)

[3 List of Terminal Mode Interfaces 15](#_Toc166055594)

[1. Normal Mode 15](#_Toc166055595)

[2. Terminal Mode 1 (TM1) 16](#_Toc166055596)

[3. Terminal Mode 3 (TM3) 16](#_Toc166055597)

[4. Index Mode 1 (IM1) 16](#_Toc166055598)

[5. Read-Print Mode 1 (RPM1) 16](#_Toc166055599)

[6. Windows Printer Driver Mode (WPD) 16](#_Toc166055600)

[4 List of Message IDs for Advanced Terminal Mode Interfaces 16](#_Toc166055601)

[5 Message Format for Advanced Terminal Mode Interfaces 20](#_Toc166055602)

[6 Message Structure 20](#_Toc166055603)

[7. Message (Positive) Acknowledgement (ACK) 22](#_Toc166055604)

[8. Message (Negative) Acknowledgement (NACK) 22](#_Toc166055605)

[7 Workflows for Advanced Terminal Mode Interfaces 24](#_Toc166055606)

[9. Setup and Configuration 24](#_Toc166055607)

[10. Job Directory Management and Editing Job Workflows 25](#_Toc166055608)

[11. Start Printing Sequence Workflows 26](#_Toc166055609)

[12. Determining length of Strings 27](#_Toc166055610)

[13. Database Files 28](#_Toc166055611)

[14. Updating Firmware Remotely 29](#_Toc166055612)

[15. Logging and Debugging 29](#_Toc166055613)

[16. Running Vega as a backend controller (hidden) but still using the job edit feature 30](#_Toc166055614)

[17. Using jobs that contain User Variables that require operator data input 30](#_Toc166055615)

[8 Message Specifications for Advanced Terminal Mode Interfaces 31](#_Toc166055616)

[18. Print Control Cycle Group 31](#_Toc166055617)

[8.1.1 START\_PRINTING 0x21 31](#_Toc166055618)

[8.1.2 QUICK\_START\_PRINTING 0Xc8 31](#_Toc166055619)

[8.1.3 STOP\_PRINTING 0x22 32](#_Toc166055620)

[8.1.4 ENABLE\_PRINTING 0x23 32](#_Toc166055621)

[8.1.5 READY\_TO\_PRINT 0x24 32](#_Toc166055622)

[8.1.6 SELECT\_VEGA\_PRINT\_BUTTON\_REMOTE 0x91 33](#_Toc166055623)

[19. Setup Messages 33](#_Toc166055624)

[8.1.7 SET\_TERMINAL\_MODE\_CONFIG\_REMOTE 0x71 33](#_Toc166055625)

[8.1.8 GET\_VERSIONS 0xB4 34](#_Toc166055626)

[8.1.9 VERSIONS\_REPLY 0xB5 35](#_Toc166055627)

[20. Job Control Group 36](#_Toc166055628)

[8.1.10 LOAD\_ PRINT\_JOB\_REMOTE 0x76 36](#_Toc166055629)

[8.1.11 LOAD\_JOB\_FOR\_EDIT 0x17 37](#_Toc166055630)

[8.1.12 GET\_LOADED\_PRINT\_JOB\_REMOTE 0x77 38](#_Toc166055631)

[8.1.13 LOADED\_PRINT\_JOB\_REPLY 0x78 38](#_Toc166055632)

[8.1.14 DELETE\_PRINT\_JOB 0x1c 39](#_Toc166055633)

[8.1.15 REQUEST\_PRINT\_JOB\_LIST 0x1A 39](#_Toc166055634)

[8.1.16 PRINT\_JOB\_LIST\_REPLY 0x1B 40](#_Toc166055635)

[8.1.17 EXPORT\_JOB 0x19 41](#_Toc166055636)

[8.1.18 IMPORT\_JOB 0x18 42](#_Toc166055637)

[8.1.19 REQUEST\_JOB\_THUMBNAIL\_IMAGE 0x1d 42](#_Toc166055638)

[8.1.20 JOB\_THUMBNAIL\_IMAGE\_REPLY 0x1E 43](#_Toc166055639)

[21. Database Control Group 44](#_Toc166055640)

[8.1.21 LOAD\_DYN\_DATA\_FILE\_REMOTE 0x79 44](#_Toc166055641)

[8.1.22 GET\_LOADED\_DYN\_DATA\_FILE\_REMOTE 0x7a 44](#_Toc166055642)

[8.1.23 LOADED\_DYN\_DATA\_FILE\_REPLY 0x7b 44](#_Toc166055643)

[22. System Configuration Group 45](#_Toc166055644)

[8.1.24 LOAD\_SYSTEM\_CONFIG\_REMOTE 0x97 45](#_Toc166055645)

[8.1.25 GET\_LOADED\_SYSTEM\_CONFIG\_REMOTE 0xA0 45](#_Toc166055646)

[8.1.26 LOADED\_SYSTEM\_CONFIG\_REPLY 0xA1 45](#_Toc166055647)

[8.1.27 TM\_SET\_VEGA\_WINDOW\_STATE 0x9e 46](#_Toc166055648)

[8.1.28 REQUEST\_UPDATE\_FIRMWARE 0x1F 46](#_Toc166055649)

[8.1.29 UPDATE\_FIRMWARE\_COMPLETE 0x20 47](#_Toc166055650)

[23. System Query Group 48](#_Toc166055651)

[8.1.30 SEND\_CONTACT\_IMAGER\_REMOTE 0x72 48](#_Toc166055652)

[8.1.31 CONTACT\_IMAGER\_REPLY 0x73 48](#_Toc166055653)

[8.1.32 GET\_PRINTER\_STATE\_REMOTE 0x74 48](#_Toc166055654)

[8.1.33 PRINTER\_STATE\_REPLY 0x75 49](#_Toc166055655)

[8.1.34 GET\_VEGA\_PRINT\_BUTTON\_STATE\_REMOTE 0x92 50](#_Toc166055656)

[8.1.35 VEGA\_PRINT\_BUTTON\_STATE\_REPLY 0x93 50](#_Toc166055657)

[8.1.36 TM\_VEGA\_PRINT\_BUTTON\_STATE\_CHANGED 0x9d 51](#_Toc166055658)

[8.1.37 GET\_STATS 0x10 51](#_Toc166055659)

[8.1.38 STATS\_REPLY 0x11 51](#_Toc166055660)

[8.1.39 GET\_PRINT\_PREVIEW 0x26 53](#_Toc166055661)

[8.1.40 PRINT\_PREVIEW\_RESPONSE 0x27 53](#_Toc166055662)

[8.1.41 USER\_ACTION\_NOTIFICATION 0x29 54](#_Toc166055663)

[24. Print Data Group 55](#_Toc166055664)

[8.1.42 SEND\_DYNAMIC\_DATA\_TM1\_REMOTE 0x7d 55](#_Toc166055665)

[8.1.43 SEND\_STATIC\_DATA\_IM1\_REMOTE 0xA3 56](#_Toc166055666)

[8.1.44 DYNAMIC\_DATA\_PRINTED\_TM1\_REPLY 0x7e 56](#_Toc166055667)

[8.1.45 SEND\_STATIC\_DATA\_TM1\_REMOTE 0xA2 57](#_Toc166055668)

[8.1.46 DATA\_BUFFER\_AVAILABLE 0x25 57](#_Toc166055669)

[8.1.47 WPD\_PAGE\_PRINTED\_REPLY 028 57](#_Toc166055670)

[8.1.48 SEND\_DYNAMIC\_DATA\_IM1\_REMOTE 0x81 58](#_Toc166055671)

[8.1.49 DYNAMIC\_DATA\_PRINTED\_IM1\_REPLY 0x82 58](#_Toc166055672)

[8.1.50 TM\_PRINT\_PURGE\_PAGE 0x9A 58](#_Toc166055673)

[8.1.51 TM\_PURGE\_PAGE\_REPLY 0x9B 59](#_Toc166055674)

[8.1.52 REQUEST\_USER\_DEFINED\_VARIABLE\_DATA 0x15 59](#_Toc166055675)

[8.1.53 USER\_DEFINED\_VARIABLE\_DATA\_REPLY 0x16 60](#_Toc166055676)

[25. Hardware IO Group 60](#_Toc166055677)

[8.1.54 SET\_IO\_OUTPUT\_RELAY\_REMOTE 0x8B 60](#_Toc166055678)

[8.1.55 GET\_IO\_INPUT\_STATE\_REMOTE 0x8C 61](#_Toc166055679)

[8.1.56 IO\_INPUT\_STATE\_REPLY 0x8d 61](#_Toc166055680)

[8.1.57 SET\_IO\_OUTPUT\_WHEN\_READY\_REMOTE 0x8E 62](#_Toc166055681)

[8.1.58 SET\_IO\_OUTPUT\_WHEN\_ERROR\_REMOTE 0x8F 63](#_Toc166055682)

[8.1.59 SET\_IO\_OUTPUT\_WHEN\_INKLOW\_REMOTE 0x90 63](#_Toc166055683)

[26. Ink Messages Group 64](#_Toc166055684)

[8.1.60 TM\_LOW\_INK 0x9C 64](#_Toc166055685)

[8.1.61 TM\_EMPTY\_INK\_ALARM 0x14 65](#_Toc166055686)

[8.1.62 GET\_INK\_LEVELS\_REMOTE 0x89 65](#_Toc166055687)

[8.1.63 INK\_LEVELS\_REPLY 0x8A 65](#_Toc166055688)

[8.1.64 GET\_INK\_DATA 0xB2 66](#_Toc166055689)

[8.1.65 INK\_DATA\_REPLY 0xB3 66](#_Toc166055690)

[8.1.66 PEN\_SPIT 0x12 70](#_Toc166055691)

[8.1.67 PEN\_SPIT\_DONE 0x13 70](#_Toc166055692)

[8.1.68 iM2\_4\_IDS\_ENABLE 0xC7 71](#_Toc166055693)

[27. Errors and Warnings Group 71](#_Toc166055694)

[8.1.69 ERROR\_MESSAGE\_REPLY 0x94 71](#_Toc166055695)

[8.1.70 WARNING\_MESSAGE\_REPLY 0xB6 73](#_Toc166055696)

[8.1.71 CLEAR\_ERROR\_AND\_WARNING\_MESSAGES 0xB7 74](#_Toc166055697)

[28. Buffer Management 75](#_Toc166055698)

[8.1.72 CLEAR\_IMAGER\_PRINT\_QUEUES 0x98 75](#_Toc166055699)

[8.1.73 DATA\_BUFFER\_AVAILABLE 0x25 75](#_Toc166055700)

[8.1.74 GET\_NUM\_LEFT\_TO\_PRINT\_REMOTE 0x85 76](#_Toc166055701)

[8.1.75 NUM\_LEFT\_TO\_PRINT\_REPLY 0x86 76](#_Toc166055702)

[9 Simple Message Type Interface Specifications 77](#_Toc166055703)

**[29.](#_Toc166055704)** [List of Message IDs for (TM3, RP1) 77](#_Toc166055704)

**[30.](#_Toc166055705)** [Message Format 78](#_Toc166055705)

[9.1.1 Terminal Mode 3 Message Wrapping 78](#_Toc166055706)

[9.1.2 Read Print Mode 1 Message Wrapping 79](#_Toc166055707)

**[31.](#_Toc166055708)** [Print Job Messages 79](#_Toc166055708)

[9.1.3 Load Job (0xF1) 79](#_Toc166055709)

[9.1.4 Start Job (0xFD) 79](#_Toc166055710)

[9.1.5 Ready to Print (0xFD) 80](#_Toc166055711)

[9.1.6 Stop Job (0xF3) 80](#_Toc166055712)

[9.1.7 Enable Printing (0xF2) 80](#_Toc166055713)

**[32.](#_Toc166055714)** [Print Data Messages 80](#_Toc166055714)

[9.1.8 Send Dynamic Data RP1 Remote 80](#_Toc166055715)

[9.1.9 Send Dynamic Data TM3 Remote 81](#_Toc166055716)

[9.1.10 Send Static Data RP1 Remote 81](#_Toc166055717)

[9.1.11 Send Static Data TM3 Remote 81](#_Toc166055718)

[9.1.12 Page Printed (0xF7) 82](#_Toc166055719)

[9.1.13 Buffer Available (0xA5) 82](#_Toc166055720)

**[33.](#_Toc166055721)** [Error Reportin](#_Toc166055721)**[g](#_Toc166055721)** [82](#_Toc166055721)

[9.1.14 Toggle Error Details (0xF8) 82](#_Toc166055722)

[9.1.15 Error Message (0xF3) 82](#_Toc166055723)

[9.1.16 Warning Message (0xF4) 83](#_Toc166055724)

[9.1.17 Low Ink Warning Message (0xFB) 84](#_Toc166055725)

**[34.](#_Toc166055726)** [System Status Messages 84](#_Toc166055726)

[9.1.18 Get System Status (0xF6) 84](#_Toc166055727)

[9.1.19 System Not Ready (0xF5) 85](#_Toc166055728)

[9.1.20 System Ready (0xF6) 85](#_Toc166055729)

[9.1.21 Command Accepted (0xFF) 85](#_Toc166055730)

**[35.](#_Toc166055731)** [Other Messages 86](#_Toc166055731)

[9.1.22 Insert Purge Page (0xF7) 86](#_Toc166055732)

[9.1.23 Purge Page Success (0xF9) 87](#_Toc166055733)

[9.1.24 Purge Page Error (0xFA) 87](#_Toc166055734)

[9.1.25 Clear Print Queues (0xF9) 87](#_Toc166055735)

[9.1.26 Clear Print Queues Success (0xFC) 87](#_Toc166055736)

[10 Software Error and Warnings 87](#_Toc166055737)

[36. Vega Errors and Warnings 87](#_Toc166055738)

[37. HP Printers 89](#_Toc166055739)

[10.1.1 HP Warning Codes 90](#_Toc166055740)

[38. NACK (Negative Acknowledge) Definitions 92](#_Toc166055741)

[11 Additional Monitoring and Configuration Messages 96](#_Toc166055742)

[39. GET\_LINE\_CONFIG 0xC0 97](#_Toc166055743)

[40. LINE\_CONFIG\_REPLY 0xC1 98](#_Toc166055744)

[41. RESET\_STATUS\_COUNTER 0xC2 101](#_Toc166055745)

[42. GET\_PRODUCTION\_LOG 0xC3 102](#_Toc166055746)

[43. PRODUCTION\_LOG\_REPLY 0xC4 103](#_Toc166055747)

[44. GET\_ALERTS 0xC5 104](#_Toc166055748)

[45. ALERTS\_REPLY 0xC6 105](#_Toc166055749)

[46. IM2\_4\_IDS\_ENABLE 0xC7 106](#_Toc166055750)

[47. CLEAR\_ERROR\_AND\_WARNING\_MESSAGES 0xB7 107](#_Toc166055751)

[48. GET\_APPLICATION\_NAME 0xCA 108](#_Toc166055752)

[49. APPLICATION\_NAME\_REPLY 0xCB 109](#_Toc166055753)

[50. SEND\_JOB 0xCC 110](#_Toc166055754)

[51. SEND\_JOB\_REPLY 0xCD 111](#_Toc166055755)

# Communications Overview

Terminal Mode (TM) is an API for communicating with the Vega application, giving the host control over all aspects print management during setup and runtime.

TCP/IP socket communication is used over an Ethernet network, or internally using Local Host (127.0.0.1).

The Vega application is the “Server” and the Host application is the “Client”. There are two independent options for creating the socket connection:

1. Legacy Two ports are used (for backward compatibility with existing host applications).

(Port 10001) for Sending Terminal Mode Messages to the Vega application.

(Port 10002) for Receiving Terminal Mode Messages from the Vega application.

1. A single port is used:

(Port 65100) for bi-directional Terminal Mode Messages between the Host and the Vega application.

1. Port 65000 is used for management / additional features (see [Appendix](#_Additional_Monitoring_and) for information)

Note: Vega will listen on both the localhost port and the Ethernet NIC port(s) and connect to the first request it receives.

Example for Single Socket implementation (on local host IP):

The following are successful connection attempts, and any errors will be reported.

Connection message in test application RemAppSim:

… Connection success from 127.0.0.1

Vega Log with Terminal Mode filter enabled:

…], [New Single Socket Terminal Mode connection established QHostAddress("127.0.0.1")]

Status bar in Vega:

Before host connects: “Terminal Mode 1: Waiting for a connection”

After host connects: “Connected to 127.0.0.1”

Example for Legacy two Socket implementation (On NIC):

The following are successful connection attempts, and any errors will be reported.

Connection message in test application RemAppSim:

… Connection success from 192.168.0.1 (receive)

… Connection success from 192.168.0.1 (transmit)

Vega Log with Terminal Mode filter enabled:

…],[TERMINAL],[INFO],[none],[New Tx Terminal Mode connection established QHostAddress("192.168.0.1")]

…],[TERMINAL],[INFO],[none],[New Rx Terminal Mode connection established QHostAddress("192.168.0.1")]

Status bar in Vega:

Before host connects: “Terminal Mode 1: Waiting for a connection”

After host connects: “Connected to 192.168.0.1”

# VEGA Remote Host Interface (Terminal Modes) Overview

The Vega application supports the following Terminal Modes (TM) for sending data and controlling Vega’s operation. The chart below shows a comparison between modes. Use this chart as a starting point to decide which TM is appropriate for your application.

# List of Terminal Mode Interfaces

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Mode Name** | **Messaging Type** | **Data Source** | **Data Encoding** | **Use** |
| [**Normal Mode**](file:///D:\Possum\Dropbox%20(Personal)\work\work\tm%20rebuild\Terminal%20Modes%20Read%20and%20Print%201%20and%20Terminal%20Mode%203Communications%20Specifications%20Guide.docx#_Normal_Mode) | N/A | Loaded In VEGA | ASCII/UTF-8 | VEGA only, No Remote Host |
| [**Terminal Mode 1**](file:///D:\Possum\Dropbox%20(Personal)\work\work\tm%20rebuild\Terminal%20Modes%20Read%20and%20Print%201%20and%20Terminal%20Mode%203Communications%20Specifications%20Guide.docx#_Terminal_Mode_1) **(TM1)** | Advanced | Remotely Located | ASCII/UTF-8 | Multi Field Records |
| [**Terminal Mode 3**](file:///D:\Possum\Dropbox%20(Personal)\work\work\tm%20rebuild\Terminal%20Modes%20Read%20and%20Print%201%20and%20Terminal%20Mode%203Communications%20Specifications%20Guide.docx#_Terminal_Mode_3) **(TM3)** | Simple | Remotely Located | ASCII Only | Print what you send |
| [**Index Mode 1**](file:///D:\Possum\Dropbox%20(Personal)\work\work\tm%20rebuild\Terminal%20Modes%20Read%20and%20Print%201%20and%20Terminal%20Mode%203Communications%20Specifications%20Guide.docx#_Index_Mode_1) **(IM1)** | Advanced | Remote or Loaded In VEGA | ASCII Only | DB Index Based Lookup |
| [**Read - Print Mode 1**](file:///D:\Possum\Dropbox%20(Personal)\work\work\tm%20rebuild\Terminal%20Modes%20Read%20and%20Print%201%20and%20Terminal%20Mode%203Communications%20Specifications%20Guide.docx#_Read-Print_Mode_1) **(RPM1)** | Simple | Loaded In VEGA | ASCII Only | DB String Based Lookup |
| Windows Printer Driver (WPD) | Advanced | WPD | WPD | When data comes from WPD |

## Normal Mode

In Normal Mode, Vega runs without any remote hosts and if a print job requires dynamic data, a database will need to be loaded into Vega prior to starting the print job. Dynamic data is printed sequentially from the database, beginning from a selected starting record number.

## Terminal Mode 1 (TM1)

In TM1, the host sends Vega dynamic data records, as well as commands to start/stop printing, load jobs/configurations, and monitor ink levels. This mode can be either ASCII or UTF-8 for the data.

## Terminal Mode 3 (TM3)

TM3 operates in a manner that allows direct ASCII printing capability by sending ASCII strings to Vega to be printed. A Preamble (if desired), and a Post-amble (or two, if desired), are defined in the Vega application. Commands for starting and stopping printing as well as loading jobs and configurations are not supported in this mode.

## Index Mode 1 (IM1)

IM1 uses a loaded database on Vega and prints the data record based on the index number sent from the host application. The host can send commands for starting and stopping printing as well as loading jobs and configurations. Databases can also be loaded by the host.

## Read-Print Mode 1 (RPM1)

RPM1 uses Vega’s loaded database along with data sent from the host to perform a string-based lookup. Commands for starting and stopping printing as well as loading jobs and configurations are not supported.

## Windows Printer Driver Mode (WPD)

WPD allows host to use many advanced TM commands except for the ones relating to managing jobs and data. The host can fully configure the printers and monitor Vega while printing.

# List of Message IDs for Advanced Terminal Mode Interfaces

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Dir** |  |  |  |
|  |  | **(Vega)** |  |  |  |
|  | **ID Hex** | **In Out** | **TM1** | **IM1** | **WPD** |
|  |  |  |  |  |  |
| [**Print Control**](#_Print_Control_Group) |  |  |  |  |  |
| [START\_PRINTING](#_START_PRINTING_0x21) | 21 | **â** | ü | ü | ü |
| [ENABLE\_PRINTING](#_ENABLE_PRINTING) | 23 | **â** | ü | ü |  |
| [READY\_TO\_PRINT](#_READY_TO_PRINT_0x24) | 24 | **á** | ü | ü | ü |
| [STOP\_PRINTING](#_STOP_PRINTING_0x22) | 22 | **â** | ü | ü | ü |
| [SELECT\_VEGA\_PRINT\_BUTTON\_REMOTE](#_SELECT_VEGA_PRINT_BUTTON_REMOTE_0x9) | **91** | **â** | ü | ü |  |
| [**Setup Messages**](#_Setup_Messages_1) |  |  |  |  |  |
| [SET\_TERMINAL\_MODE\_CONFIG\_REMOTE](#_SET_TERMINAL_MODE_CONFIG_REMOTE_1) | **71** | **â** | ü | ü | ü |
| [GET\_VERSIONS](#_GET_VERSIONS_0xB4) | B4 | **â** | ü | ü | ü |
| [VERSIONS\_REPLY](#_VERSIONS_REPLY_0xB5) | B5 | **á** | ü | ü | ü |
| [**Job**](#_Job_Control_Group) |  |  |  |  |  |
| [LOAD\_JOB\_TO\_PRINT](#_LOAD_JOB_TO_PRINT_0x76) | 76 | **â** | ü | ü |  |
| [LOAD\_JOB\_FOR\_EDIT](#_LOAD_JOB_FOR_EDIT_0x17) | 17 | **â** | ü | ü |  |
| [GET\_LOADED\_PRINT\_JOB\_REMOTE](#_GET_LOADED_PRINT_JOB_REMOTE_0x77) | 77 | **â** | ü | ü |  |
| [LOADED\_PRINT\_JOB\_REPLY](#_LOADED_PRINT_JOB_REPLY_0x78) | 78 | **á** | ü | ü |  |
| [DELETE\_PRINT\_JOB](#_DELETE_PRINT_JOB__0x1c) | 1C | **â** | ü | ü |  |
| [REQUEST\_PRINT\_JOB\_LIST](#_REQUEST_PRINT_JOB_LIST_0x1A) | 1A | **â** | ü | ü |  |
| [PRINT\_JOB\_LIST\_REPLY](#_PRINT_JOB_LIST_REPLY_0x1B) | 1B | **á** | ü | ü |  |
| [EXPORT\_JOB](#_EXPORT_JOB_0x19) | 19 | **â** | ü | ü |  |
| [IMPORT\_JOB](#_IMPORT_JOB_0x18) | 18 | **â** | ü | ü |  |
| [REQUEST\_JOB\_THUMBNAIL\_IMAGE](#_REQUEST_JOB_THUMBNAIL_IMAGE_0x1d) | 1D | **â** | ü | ü | ü |
| [JOB\_THUMBNAIL\_IMAGE\_REPLY](#_JOB_THUMBNAIL_IMAGE_REPLY_0x1E) | 1E | **á** | ü | ü | ü |
| [**Database**](#_Database_Control_Group) |  |  |  |  |  |
| [LOAD\_DYN\_DATA\_FILE\_REMOTE](#_LOAD_DYN_DATA_FILE_REMOTE_0x79) | **79** | **â** |  | ü |  |
| [GET\_LOADED\_DYN\_DATA\_FILE\_REMOTE](#_GET_LOADED_DYN_DATA_FILE_REMOTE_0x7) | **7A** | **â** |  | ü |  |
| [LOADED\_DYN\_DATA\_FILE\_REPLY](#_LOADED_DYN_DATA_FILE_REPLY_0x7b) | **7B** | **á** |  | ü |  |
| [**System Configuration**](#_System_Configuration_Group) |  |  |  |  |  |
| [LOAD\_SYSTEM\_CONFIG\_REMOTE](#_LOAD_SYSTEM_CONFIG_REMOTE_0x97) | **97** | **â** | ü | ü | ü |
| [GET\_LOADED\_SYSTEM\_CONFIG\_REMOTE](#_GET_LOADED_SYSTEM_CONFIG_REMOTE_0xA) | **A0** | **â** | ü | ü | ü |
| [LOADED\_SYSTEM\_CONFIG\_REPLY](#_LOADED_SYSTEM_CONFIG_REPLY_0xA1) | **A1** | **á** | ü | ü | ü |
| [TM\_SET\_VEGA\_WINDOW\_STATE](#_TM_SET_VEGA_WINDOW_STATE_0x9e) | 9E | **â** | ü | ü | ü |
| [REQUEST\_UPDATE\_FIRMWARE](#_REQUEST_UPDATE_FIRMWARE_1F) | 1F | **â** | ü | ü | ü |
| [UPDATE\_FIRMWARE\_COMPLETE](#_UPDATE_FIRMWARE_COMPLETE) | 20 | **á** | ü | ü | ü |
| [**System Query Messages**](#_System_Query_Group) |  |  |  |  |  |
| [SEND\_CONTACT\_IMAGER\_REMOTE](#_SEND_CONTACT_IMAGER_REMOTE_0x72) | **72** | **â** | ü | ü | ü |
| [CONTACT\_IMAGER\_REPLY](#_CONTACT_IMAGER_REPLY_0x73) | **73** | **á** | ü | ü | ü |
| [GET\_PRINTER\_STATE\_REMOTE](#_GET_PRINTER_STATE_REMOTE_0x74) | **74** | **â** | ü | ü | ü |
| [PRINTER\_STATE\_REPLY](#_PRINTER_STATE_REPLY_0x75) | **75** | **á** | ü | ü | ü |
| [GET\_VEGA\_PRINT\_BUTTON\_STATE\_REMOTE](#_GET_VEGA_PRINT_BUTTON_STATE_REMOTE_) | **92** | **â** | ü | ü | ü |
| [VEGA\_PRINT\_BUTTON\_STATE\_REPLY](#_VEGA_PRINT_BUTTON_STATE_REPLY_0x93) | **93** | **á** | ü | ü | ü |
| [TM\_VEGA\_PRINT\_BUTTON\_STATE\_CHANGED](#_TM_VEGA_PRINT_BUTTON_STATE_CHANGED_) | **9D** | **á** | ü | ü | ü |
| [GET\_STATS](#_GET_STATS_0x10) | 10 | **â** | ü | ü | ü |
| [GET\_STATS\_REPLY](#_STATS_REPLY_0x11) | 11 | **á** | ü | ü | ü |
| [GET\_PRINT\_PREVIEW](#_GET_PRINT_PREVIEW) | 26 | **â** | ü | ü | ü |
| [PRINT\_PREVIEW\_RESPONSE](#_PRINT_PREVIEW_RESPONSE_0x27) | 27 | **á** | ü | ü | ü |
| [USER\_ACTION\_NOTIFICATION](#_USER_ACTION_NOTIFICATION_0x29) | 29 | **á** | ü | ü |  |
| [**Print Data Messages**](#_Print_Data_Group) |  |  |  |  |  |
| [SEND\_DYNAMIC\_DATA\_TM1\_REMOTE](#_SEND_DYNAMIC_DATA_TM1_REMOTE_0x7d) | **7D** | **â** | ü |  |  |
| [SEND\_STATIC\_DATA\_TM1\_REMOTE](#_SEND_DYNAMIC_DATA_TM1_REMOTE) | **A2** | **â** | ü |  |  |
| [SEND\_STATIC\_DATA\_IM1\_REMOTE](#_SEND_STATIC_DATA_IM1_REMOTE_0xA2) | **A3** | **â** |  | ü |  |
| [DYNAMIC\_DATA\_PRINTED\_TM1\_REPLY](#_DYNAMIC_DATA_PRINTED_IM1_REPLY) | **7E** | **á** |  | ü |  |
| [SEND\_DYNAMIC\_DATA\_IM1\_REMOTE](#_SEND_DYNAMIC_DATA_IM1_REMOTE_0x81) | **81** | **â** |  | ü |  |
| [DYNAMIC\_DATA\_PRINTED\_IM1\_REPLY](#_DYNAMIC_DATA_PRINTED_IM1_REPLY_0x82) | **82** | **á** |  | ü |  |
| [DATA\_BUFFER\_AVAILABLE](#_DATA_BUFFER_AVAILABLE_0x25) | 25 | **á** | ü | ü |  |
| [WPD\_PAGE\_PRINTED\_REPLY](#_WPD_PAGE_PRINTED_REPLY_028) | 28 | **á** |  |  | ü |
| [TM\_PRINT\_PURGE\_PAGE](#_TM_PRINT_PURGE_PAGE_1) | 9A | **â** | ü | ü |  |
| [TM\_PURGE\_PAGE\_REPLY](#_TM_PURGE_PAGE_REPLY_0x9B) | 9B | **á** | ü | ü |  |
| [REQUEST\_USER\_DEFINED\_VARIABLE\_DATA](#_REQUEST_USER_DEFINED_VARIABLE_DATA_) | 15 | **á** | ü | ü |  |
| [USER\_DEFINED\_VARIABLE\_DATA\_REPLY](#_USER_DEFINED_VARIABLE_DATA_REPLY_0x) | 16 | **â** | ü | ü |  |
| [**Hardware IO Messages**](#_Hardware_IO_Group) |  |  |  |  |  |
| [SET\_IO\_OUTPUT\_RELAY\_REMOTE](#_SET_IO_OUTPUT_RELAY_REMOTE_0x8B) | **8B** | **â** | ü | ü | ü |
| [GET\_IO\_INPUT\_STATE\_REMOTE](#_GET_IO_INPUT_STATE_REMOTE_0x8C) | **8C** | **â** | ü | ü | ü |
| [IO\_INPUT\_STATE\_REPLY](#_IO_INPUT_STATE_REPLY_0x8d) | **8D** | **á** | ü | ü | ü |
| [SET\_IO\_OUTPUT\_WHEN\_READY\_REMOTE](#_SET_IO_OUTPUT_WHEN_READY_REMOTE_0x8) | **8E** | **â** | ü | ü | ü |
| [SET\_IO\_OUTPUT\_WHEN\_ERROR\_REMOTE](#_SET_IO_OUTPUT_WHEN_ERROR_REMOTE_0x8) | **8F** | **â** | ü | ü | ü |
| [SET\_IO\_OUTPUT\_WHEN\_INK\_LOW\_REMOTE](#_SET_IO_OUTPUT_WHEN_INKLOW_REMOTE_0x) | **90** | **â** | ü | ü | ü |
| [**Ink Messages**](#_Ink_Messages_Group) |  |  |  |  |  |
| [TM\_LOW\_INK](#_TM_LOW_INK_0x9C) | **9C** | **á** | ü | ü | ü |
| [TM\_EMPTY\_INK\_ALARM](#_TM_EMPTY_INK_ALARM_0x14) | 14 | **á** | ü | ü | ü |
| [GET\_INK\_LEVELS\_REMOTE](#_GET_INK_LEVELS_REMOTE_0x89) | 89 | **â** | ü | ü | ü |
| [INK\_LEVELS\_REPLY](#_INK_LEVELS_REPLY_0x8A) | 8A | **á** | ü | ü | ü |
| [GET\_INK\_DATA](#_GET_INK_DATA_0xB2) | B2 | **â** | ü | ü | ü |
| [INK\_DATA\_REPLY](#_INK_DATA_REPLY_0xB3) | B3 | **á** | ü | ü | ü |
| [PEN\_SPIT](#_PEN_SPIT_0x12) | 12 | **â** | ü | ü | ü |
| [PEN\_SPIT\_DONE](#_PEN_SPIT_DONE__0x13) | 13 | **á** | ü | ü | ü |
| [iM2\_4\_IDS\_ENABLE](#_iM2_4_IDS_ENABLE_0xC7) | C7 | **â** | ü | ü | ü |
| [**Errors and Warnings**](#_Errors_and_Warnings) |  |  |  |  |  |
| [ERROR\_MESSAGE\_REPLY](#_ERROR_MESSAGE_REPLY_0x94) | **94** | **á** | ü | ü | ü |
| [WARNING\_MESSAGE\_REPLY](#_WARNING_MESSAGE_REPLY_0xB6) | **B6** | **á** | ü | ü | ü |
| [CLEAR\_ERROR\_AND\_WARNING\_MESSAGES](#_CLEAR_ERROR_AND_WARNING_MESSAGES_0x) | **B7** | **â** | ü | ü | ü |
| [**Buffer Management**](#_Buffer_Management) |  |  |  |  |  |
| [CLEAR\_IMAGER\_PRINT\_QUEUES](#_CLEAR_IMAGER_PRINT_QUEUES_0x98) | **98** | **â** | ü | ü |  |
| [GET\_NUM\_LEFT\_TO\_PRINT\_REMOTE](#_GET_NUM_LEFT_TO_PRINT_REMOTE_0x85) | **85** | **â** | ü | ü | ü |
| [NUM\_LEFT\_TO\_PRINT\_REPLY](#_NUM_LEFT_TO_PRINT_REPLY_0x86) | **86** | **á** | ü | ü | ü |
| [**ACKs**](#_Message_Structure) |  |  |  |  |  |
| [REMOTE\_NEGATIVE\_ACKNOWLEDGE\_MSG](#_Message_(Negative)_Acknowledgement) | B8 | **á** | ü | ü | ü |
| [REMOTE\_ACKNOWLEDGE\_MSG](#_Message_(Positive)_Acknowledgement) | **99** | **á** | ü | ü | ü |

# Message Format for Advanced Terminal Mode Interfaces

In this document, C-programming language structures are used to illustrate the message formats, however, any programming language that can reproduce the binary structure of these messages can be used.

Note:

All structures must be “packed” to a 1 byte packing alignment (i.e. pragma pack(1)). All message components longer than 1 byte (such as unsigned long integers) must be in “little-endian” (Intel) format (as opposed to “big-endian” (Motorola) format

# Message Structure

Byte Function  
----------------------------------------------------  
1 Beginning of Message (STX = 0x02)  
2-5 Number of Bytes to Follow   
6 Checksum   
7 Transmit Sequence Number  
8 Message ID   
(9 ....) Data (if Message requires Data)  
Last Byte End of Message (ETX = 0x03)

typedef struct {  
 unsigned char ucSTX; // 0x02  
 unsigned long int uliNumBytesToFollow;  
 unsigned char ucChecksum;  
 unsigned char ucTransmitSequenceNum;  
 unsigned char ucMessageID;

// optional data   
 unsigned char ucETX; // 0x03  
} SMessageFormat;

**Message Format Description**

**Beginning of Message** (1 byte) - An STX (0x02) is the beginning of message delimiter.

**Number of Bytes to Follow** (4 bytes) - An unsigned long int (4 bytes) that represents the number of bytes that follow in the message. This consists of the Checksum byte, the Transmit Sequence Number byte, the Message ID byte, Data bytes (if the message requires data, in addition to the Message ID byte) and the End of Message byte.

Note: Number of Bytes to Follow = 4 + (number of bytes of Data, if any).

**Checksum** (1 byte) - Currently implemented as a simple sum, with resulting overflow and clipped result, of all the bytes that follow, up to and including the End of Message byte. Checksum implementation is OPTIONAL, and if not used, the Checksum byte should be set to ‘0’.

**Transmit Sequence Number** (1 byte) - This byte will increment with each message transmission, starting at a value of 0, incrementing to 255, back to 0 again, etc. Transmit Sequence Number implementation is OPTIONAL, and if not used, the Transmit Sequence Number byte should be set to ‘0’. Terminal mode supports an OPTIONAL Message Acknowledgement feature. A Message Acknowledgement message contains the Message ID byte and the Transmit Sequence Number byte of the message being acknowledged.

**Message ID** (1 byte) - This byte represents the identifier for the message or command. A Message Acknowledgement message contains the Message ID byte and the Transmit Sequence Number byte of the message being acknowledged.

**Data** (>= 0 bytes) - If a message requires data, it directly follows the Message ID byte.

**End of Message** (1 byte) - An ETX (0x03) is the end of message delimiter. The End of Message byte directly follows the Data byte(s) in messages that require Data, else it directly follows the Message ID byte.

Example:

The following breakdown is for the 0x71 command that contains a 6 Boolean byte payload:

<Sent> - SET\_TERMINAL\_MODE\_CONFIG\_REMOTE .......q....... [02 0A 00 00 00 00 00 71 00 01 01 01 01 01 03 ]

02 Start of Text Character

0A 00 00 00 Bytes to follow “little-endian” = 00 00 00 0A = decimal 10 (includes 03 at end)

00 Checksum

00 Transmit Sequence Number

71 Command ID

00 01 01 01 01 01 6-byte payload

03 End of Text Character

## Message (Positive) Acknowledgement (ACK)

ID: 0x99 Direction: *Both*: Vega 🡪 Host and 🡪 Host 🡪 Vega

Purpose: Acknowledges receipt of a message request and that it was accepted without error

Notes: The command request was processed

An error may be issued during execution of the command request. See: ERROR\_MESSAGE\_REPLY

Data:

typedef struct{  
 unsigned char ucMessageID;  
 unsigned char ucTransmitSequenceNum;  
} SRemoteAcknowledgeMsg;

Description:

ucMessageID - The ucMessageID of the message that is being Acknowledged.

ucTransmitSequenceNum - The ucTransmitSequenceNum of the message that is being acknowledged.

Example:

The following command 0x92 is ACK’d with the 0x99 for the 0x92 because it was processed (accepted) without error.

<Sent> - GET\_GUI\_PRINT\_BUTTON\_STATE\_REMOTE ......... [02 04 00 00 00 00 00 92 03 ]

<Received> - REMOTE\_ACKNOWLEDGE\_MSG ucMessageID: 146, ucTransmitSequenceNum: 0

… [02 06 00 00 00 00 01 99 92 00 03]

## Message (Negative) Acknowledgement (NACK)

ID: 0xB8 Direction: Vega 🡪 Host

Purpose: To indicate to Host that the command request was not successful.

Notes: This reply is immediate, and the command request is aborted

The error description is UTF-8

Must be enabled in SET\_TERMINAL\_MODE\_CONFIG\_REMOTE

Data:

typedef struct{  
 unsigned char ucMessageID;  
 unsigned char ucTransmitSequenceNum;

unsigned char ucReasonID;

char\* uft8ReasonDescription;

}SRemoteAcknowledgeMsg;

Description:

ucMessageID - The ucMessageID of the message that is being Acknowledged.

ucTransmitSequenceNum - The ucTransmitSequenceNum of the message that is being Acknowledged.

ucReasonID utf8ReasonDescription

0 Unknown

1 Not allowed while printing (ex: trying to load a job)

2 Must be on Home Screen (ex: trying to start a job when on edit screen)

3 Printer(s) not on-line (ex: get ink data)

4 No Job Loaded

5 Job not found

6 User edit in progress

7 directory not found

8 argument out of range

9 cannot send static data when not in print mode

10 cannot send dynamic data when not in print mode

Example:

Trying to start printing when there is no job loaded yet:

The command 0x91 cannot be executed so the NACK (0xb8) is returned with the error 5 for “job not found”

<Sent> - SELECT\_GUI\_PRINT\_BUTTON\_REMOTE ......... [02 04 00 00 00 00 00 91 03 ]

<Received> - REMOTE\_NEGATIVE\_ACKNOWLEDGE\_MSG Message ID: 145, Sequence Num: 0, Reason: 0, Description:

…[02 07 00 00 00 00 0d b8 91 05 00 03]

# Workflows for Advanced Terminal Mode Interfaces

## Setup and Configuration

Printer Hardware:

The Vega Printer Setup dialogs are used to configure the printer hardware, and there are no remote printer configuration commands in terminal mode.

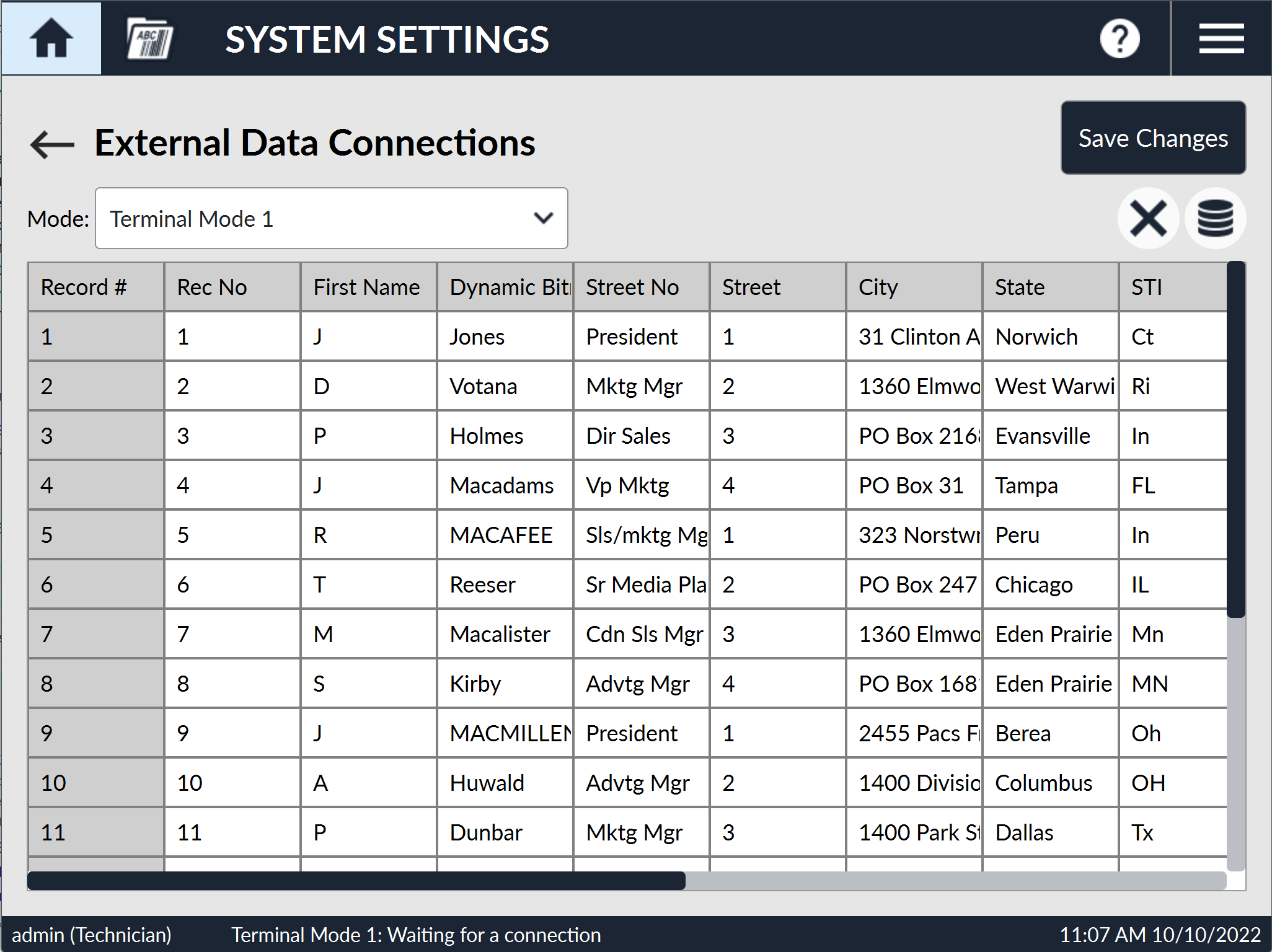
To Enable TM in Vega:

Under the Vega System Settings menu, you can enter the External Data Connections dialog where you will:

1. Enable the desired terminal mode type
2. For TM 1 a sample dummy database is configured that contains:
   1. Meaningful field names (to assist in selecting the field to use)
   2. Meaningful sample field data (for previewing to verify layout)

Example:

Graphical user interface, application

Description automatically generated 

To check that the Vega TM API revision is compatible with the Host’s application:

The GET\_VERSIONS 0xB4 is used to retrieve the following:

char[20] ApplicationVersion  
char[20] APIVersion  
char[20] InkTypesVersion

The versions are character strings with the following format consisting of 4 digits:

The first three digits represent the version number, usually displayed with “.” a dot between the numbers.

The fourth number is for the build number, where a “0” represents a production build.

Fully tested production builds will always end with a “0” and other builds represent interim development builds that occur between full production builds.

The API Version is contained in this document’s section for release notes. If a Host encounters a newer version of Vega it is very likely that the API will be backwards compatible (newer release has features that Host is not using and bug fixes). However, it is very important for the Host to investigate for possible side effects using a Vega / TM API interface version that is different from the one their application was originally targeted for.

Example:

Chart

Description automatically generatedGraphical user interface

Description automatically generated with medium confidence Note: the application version can also be found on the Vega About dialog.

The Terminal Mode API Version is only available in this message

The Inktypes version can also be found for HP pens in the

Printer Setup 🡪Ink dialog

To configure the optional flags in the messages between Vega and the Host:

The command SET\_TERMINAL\_MODE\_CONFIG\_REMOTE 0x71 must be sent every time the host connects to Vega.

Note: Vega does not save the communication flags.

These flags will be used to enable sequence numbers, error and warning notifications, and ACK messages.

Example: Screen shots to set these flags in the Remote Application Simulator (RemAppSim.exe)

Configuration button on left and the button on the operation tab to send the command for TM1 to the right.

Graphical user interface, application

Description automatically generated Graphical user interface, text, application

Description automatically generated

## Job Directory Management and Editing Job Workflows

The TM Host has **all** the capabilities that an operator does, to import, export, delete, and create/edit jobs.

Often the PC running Vega is the same PC that the Host application is running on. The Host application typically is involved in some process control tracking products, where the printing is not the focus for the operator (example: a mail inserting machine). In this case the Host will minimize the Vega application and run it in the background (not visible).

Managing the Vega Jobs Directory:

The Host can import, export, and delete jobs (all while the Vega application is hidden).

Note: Vega can only edit and print jobs that are loaded in its local directory named “Jobs”.

The commands IMPORT\_JOB 0x18 and EXPORT\_JOB 0x19 enable the host to COPY jobs in and out of the Jobs folder. Jobs no longer needed in the local Vega Jobs directory removed using the DELETE\_PRINT\_JOB 0x1C.

Loading an existing job, or creating a new job for edit:

The Host can open a job for edit and **ONLY** permit the operator to edit the job, blocking access to any other screen in the Vega application. The Host will send [LOAD\_JOB\_FOR\_EDIT 0x17](#_LOAD_JOB_FOR_EDIT) and the job edit screen will be displayed exclusively.

Use Case: The Vega application is hidden, and the Host wishes to expose a job for the operator to edit. The host will load the job while Vega is hidden, then make Vega visible for the edit and save, and finally hide Vega again. This lets the Host use the Vega job editor as custom “plug-in” and keep the other aspects of the software out of the way.

When the Host needs to assist an operator in loading the desired job (again in the local Jobs folder) they can present a list of jobs using REQUEST\_PRINT\_JOB\_LIST 0x1A. For additional information they can display the job preview thumbnail for reference REQUEST\_JOB\_THUMBNAIL\_IMAGE 0x1D to confirm the correct job is selected visually.

To create a new job for edit the Host will send LOAD\_JOB\_FOR\_EDIT 0x17 with the flag for create new job set.

## Start Printing Sequence Workflows

The starting a job and beginning to print follows one of three different workflows:

1. Operator presses the start button and printing begins immediately
2. TM Host remotely “presses” the start button and printing begins immediately
3. TM Host starts the job, loads data buffers as desired, and then printing begins

Note 1: When terminal Mode is enabled (any of them) the host can send commands to start/ stop, and the operator can ALSO press the button on the screen to Start/ Stop.

Note 2: There is NO command to PAUSE for TM or the operator. When Vega has received a START\_PRINTING command from the TM host and is waiting for the host to send the ENABLE\_PRINT command Vega will be in State = 6 “Pause”.

This is NOT a valid button state for Vega when **any** of the terminal modes are enabled. 

Vega Terminal Mode and Print Button States

|  |  |  |  |
| --- | --- | --- | --- |
| ucPrintButton  State | **Print Button  Text** | **Print Button Bitmap** | **Description** |
| 0 | “Start Print Job” | A picture containing chart  Description automatically generated | Printing Stopped – idle |
| 1 | “Configuring…  PLEASE WAIT” | Icon  Description automatically generated | Configuring printer(s). |
| 2 | “Stop Print Job” | Chart  Description automatically generated | Printing |
| 5 | “Error” | A picture containing diagram  Description automatically generated | Printer off-line, or no job is loaded. |
| 6 | “Paused” |  | Paused |

Terminal Mode Manual Operator Job Start and Stop Workflow

Action State

1. No job loaded 🡪 5 = “Error”
2. Load Job 🡪 0 = “Start Print Job”
3. Press Start Button 🡪 1 = “Configuring”
4. Printing 🡪 2 = “Stop Print Job”
5. Press Stop Button 🡪 0 = “Printing Stopped – Idle”

Terminal Mode 1 (*Legacy*) Job Start and Stop Workflow

Action State

1. No job loaded 🡪 5 = “Error”
2. LOAD\_JOB\_TO\_PRINT 76 🡪 0 = “Start Print Job”
3. SELECT\_VEGA\_PRINT\_BUTTON\_REMOTE 91 🡪 1 = “Configuring”
4. Printing 🡪 2 = “Stop Print Job”
5. SELECT\_VEGA\_PRINT\_BUTTON\_REMOTE 91 🡪 0 = “Printing Stopped – Idle”

Terminal Mode 1 , Read Print 1, and Terminal Mode 3 Job Start and Stop Workflow

Action State

1. No job loaded 🡪 5 = “Error”
2. LOAD\_JOB\_TO\_PRINT 76 🡪 0 = “Start Print Job”
3. START\_PRINTING 21 🡪 1 = “Configuring”
4. Paused – Ready to Print 🡪 6 = “Paused”
5. *Receive* *READY\_TO\_PRINT 24*
6. *Send data records as desired …*
7. ENABLE\_PRINTING 23 🡪 2 = “Stop Print Job” Printing
8. STOP\_PRINTING 22 🡪 0 = “Printing Stopped – Idle”

## Determining length of Strings

UTF-8 strings are not NULL terminated, and the Host needs to reference the “Number of Bytes to Follow   
”. The start of the data is always at byte #9 and the length of the data is Number of Bytes to Follow – 1 (for ETX).

1 Beginning of Message (STX = 0x02)  
2-5  Number of Bytes to Follow   
6 Checksum   
7 Transmit Sequence Number  
8 Message ID   
(9 ....) Data (if Message requires Data)  
Last Byte End of Message (ETX = 0x03)

When messages contain multiple UFT-8 strings the will be separated with a specific delimiter.

For example the STATUS\_REPLY 0x11 message is:

“,” comma separated variable name- data pairs

“=” equals sign is used to separate the variable name from the data

Example data: “Printing=Yes,Speed=20.4,SpeedUnits= SPEED\_FEET\_PER\_MINUTE, …”

Note: the numeric values are passed as text string and the Host is responsible for converting to the appropriate integer or floating point.

## Database Files

**Index Mode 1**, (where an record index is sent for a loaded database lookup):

Database files must be copied into the Vega folder named “Databases” and the database cannot be run from a network drive.

The path on a Windows PC is: “C:\Users\Public\Vega\Databases”. (OEM branded versions “Vega” will be different).

Note: on the Embedded Linux platform Network Services are not supported and the user must import databases into the unit using the USB drive.

The host is responsible for copying the Database files into the local Vega folder and for deleting the files once they are no longer needed. (of course, operators could import and delete the files using Vega manually).

.Schema files:

Databases may be text files or binary, for example a text “.csv” file and a Microsoft Access ”.mbd” file.

When a database is first loaded into Vega the application will prompt the user to configure the correct field and record delimiters, and then it will create a companion file with exactly the same name with extension and then add a “.schema” to the end. For example: text database “sample.csv” will have a schema file named “sample.csv.schema”.

The .schema file is text file in XML format that contains all the configuration parameters for Vega to read and use the database.

When importing and deleting files using the filesystem, the .schema file needs to be treated as part of the database.

Note: when an operator imports a database, they will be prompted to configure the schema; however, IF when importing the database via the filesystem and the .schema file is not copied over there is no desire for a user prompt, so the host must be careful to always bring a .schema file for every database copied into Vega’s database directory.

Terminal Mode 1, (Where the entire data record is sent by the host and no database is loaded):

When the Host is sending the entire data record, Vega needs two things:

1) to know how data will be formatted, and

2) how to map the fields into the correct positions in a job.

To accomplish this, Vega requires that the user select a sample “Template” database when they enable the Terminal Mode 1 interface in Vega. By doing this, the .schema is defined for the data format, and when editing a job the field names (if they are not the default Field1 etc.) are available for the user to place on the layout with meaningful sample data shown in the print preview for reference.

## Updating Firmware Remotely

Note: Updating firmware by the host should only be used with caution as power cycling a printer can brick it.

The intended use case for this feature is for OEMs that run Vega hidden in the background and the operators are not familiar with how to update the firmware.

The Host should be sure to display some message on screen that the operator must agree to before the update is started (and the warning should remain up until the update is complete.)

## Logging and Debugging

Logging is an essential tool for debugging communication issues between Vega and the TM host. There are two sources of logs:

1. Vega menu 🡪 Logs 🡪 Log Settings 🡪 Terminal Mode Logging = Enabled (example below)

…],[TERMINAL],[DEBUG],[none],[Received "0xB4,GET\_VERSIONS"]

…],TERMINAL],[DEBUG],[none],[Generated message "0xB5,VERSIONS\_REPLY,1.1.10.6,0.0.0.0,1.0.40"]

…],[TERMINAL],[DEBUG],[none],["Added VERSIONS\_REPLY to Terminal Mode 1 message queue."]

…],[TERMINAL],[DEBUG],[none],[Sending "VERSIONS\_REPLY"]

…],[TERMINAL],[DEBUG],[none],[No messages left in queue]

1. Log screen when using the RemAppSim tester (see below: note sent messages are decoded in hex)

… <Sent> - GET\_VERSIONS ......... [02 04 00 00 00 00 00 B4 03 ]

… <Sent> - REMOTE\_ACKNOWLEDGE\_MSG ........... [02 06 00 00 00 00 00 99 B5 00 03 ]

…<Received> - VERSIONS\_REPLY

Versions: Application 1.1.10.6, API 0.0.0.0, Ink Types: 1.0.40

Debugging is performed using the RemAppSim.exe program, called Remote Application Simulator. This program will let the developer test the full Terminal Mode API and see the “intended” execution by example.

Note: See RemAppSim Quickstart.pdf

Example:

Screen shot showing Vega configured for Terminal Mode logging.

Graphical user interface, application

Description automatically generated Graphical user interface, text, application

Description automatically generated

The logs files will contain the tag “[TERMINAL]” and then the type tag “[DEBUG]”, “[INFO]”. “[WARNING]”, or “[ERROR’]”.

Note: Vega creates a single log file and there are not separate log file types for the printers or for terminal mode individually.

## Running Vega as a backend controller (hidden) but still using the job edit feature

Vega can be run as a “back end” controller with its user interface hidden from the operator. The command [TM\_SET\_VEGA\_WINDOW\_STATE 0x9e](#_TM_SET_VEGA_WINDOW_STATE) is used to hide the Vega application and let the Host application be the primary user facing interface. This is a common approach and works well; however, if there is a need to edit a job, the full Vega application would need to be exposed to the user, and this might not be desired.

The command [LOAD\_JOB\_FOR\_EDIT 0x17](#_LOAD_JOB_FOR_EDIT) is used by the host to expose the job edit dialog with all its functionality, and nothing else- the user cannot exit the job edit screen (simply edit and save the job).

When the user is done with the job edit operation, they will press the “X” close button in the upper right of the window, and this will signal the Host that the edit operation has completed. The command [USER\_ACTION\_NOTIFICATION 0x29](#_USER_ACTION_NOTIFICATION_(0x29)) is sent to the host as shown in the example below which gives the host the following workflow options:

1. Open another job for edit
2. Hide the Vega application and Load a job for printing (it could be the job just edited)

Example: Response after the user presses the “X” close window after the edit was completed.

USER\_ACTION\_NOTIFICATION 0x29 (UserName, UserRole, ActionID, ActionString, DescriptionString)

(Admin, Technician, 1, Save job, Label UDI 2x4)

where ActionID =1 means “Load job for print” followed by “the name of the job edited”

## Using jobs that contain User Variables that require operator data input

Vega can receive 3 distinct data commands from the Host:

1. Static Data Record [SEND\_STATIC\_DATA\_TM1\_REMOTE 0x7d](#_SEND_DYNAMIC_DATA_TM1_REMOTE)
2. Dynamic Data Record [SEND\_DYNAMIC\_DATA\_TM1\_REMOTE 0xa2](#_SEND_DYNAMIC_DATA_TM1_REMOTE)
3. User Variables [USER\_DEFINED\_VARIABLE\_DATA\_REPLY 0x16](#_USER_DEFINED_VARIABLE_DATA_REPLY)

All three external data sources can be in a job; however, this is not commonly done.

User variables are objects created in the job that will prompt the operator when the job is started- to enter some values given some information about the purpose, type, and range of the variable. These variables are entered once and used again and again.

Note: it is not possible to let the operator enter user variables at the pop-up prompts at the start of the job when in terminal mode (like you would when terminal mode is disabled). The idea is that all data entry is from the Host interface.

Vega requires that it receives the user variable data from the host before it will send the [READY\_TO\_PRINT 0x24](#_READY_TO_PRINT) to the host (see [Start Printing Sequence Workflows](#_Start_Printing_Sequence)). When the host receives the ready to print, they will know that the user variables have been RIPped without error and the [ENABLE\_PRINTING 0x23](#_ENABLE_PRINTING_0x23) can be sent.

# Message Specifications for Advanced Terminal Mode Interfaces

## Print Control Cycle Group

### START\_PRINTING 0x21

ID: 0x21 Direction: Host 🡪 Vega

Purpose: For Vega to initialize the current job and printer(s) for printing

Notes: There must be a job loaded else NACK 4 “No Job Loaded”

Vega must be on the Home Page else NACK 2 “Must be on Home Screen”

Requires Host sent [ENABLE\_PRINTING](#_ENABLE_PRINTING) to begin printing

Data: none

Response: [READY\_TO\_PRINT](#_READY_TO_PRINT) indicating host may pre-fill buffers or enable printing

Description: This command is used by the Host to control the [Start Printing Sequence](#_Start_Printing_Sequence)

The single command alternative is [SELECT\_VEGA\_PRINT\_BUTTON\_REMOTE](#_SELECT_VEGA_PRINT_BUTTON_REMOTE).

Example:

<Sent> - START\_PRINTING .......!. [02 04 00 00 00 00 00 21 03 ]

### QUICK\_START\_PRINTING 0Xc8

ID: 0x21 Direction: Host 🡪 Vega

Purpose: For Vega to switch to another job while currently printing

Notes: This is for quickly switching from a printing job to another job without having to stop and do pen tests etc.

There must be a job Printing else NACK ? “No Job Currently Printing”

Vega must be on the Home Page else NACK 2 “Must be on Home Screen”

If the job name is not found in the Vega Local jobs directory (i.e. job list) return NACK ? “Job Not found”

Data: char\* utf8JobName; // UTF-8

Response: [READY\_TO\_PRINT](#_READY_TO_PRINT) indicating host may pre-fill buffers or enable printing

Description: This command is used by the Host to not stop the currently printing job and switch to another job quickly.

Example:

<Sent> - QUICK\_START\_PRINTING .......!. [02 04 00 00 00 00 00 C8 03 ]

### STOP\_PRINTING 0x22

ID: 0x22 Direction: Host 🡪 Vega

Purpose: This will immediately stop the print cycle

Notes: All printing will stop and buffered pages in Vega and the printer will be deleted

Also, any data messages from the Host will be purged.

If Vega is not in print mode a NACK 11 “Not in Print Mode”

Data: none

Response: [VEGA\_PRINT\_BUTTON\_STATE\_REPLY](#_TM_VEGA_PRINT_BUTTON_STATE_CHANGED) = 1 “Idle”

Description: This command stops the job just as if the operator pressed the stop button manually.

Example:

<Sent> - STOP\_PRINTING .......". [02 04 00 00 00 00 00 22 03 ]

### ENABLE\_PRINTING 0x23

ID: 0x23 Direction: Host 🡪 Vega

Purpose: To let printing begin on the next “print go” sensor trigger after pre-filling any desired buffers

Notes: This command enables the printer hardware trigger sensor

This can be used to precisely time the start of sequenced print data

Data: none

Response: [VEGA\_PRINT\_BUTTON\_STATE\_REPLY](#_TM_VEGA_PRINT_BUTTON_STATE_CHANGED) = 2 “Printing”

Description: This command is used by the Host to control the [Start Printing Sequence](#_Start_Printing_Sequence)

Example:

<Sent> - ENABLE\_PRINTING .......#. [02 04 00 00 00 00 00 23 03 ]

### READY\_TO\_PRINT 0x24

ID: 0x24 Direction: Vega 🡪 Host

Purpose: Tell the host that the job was configured, and the printers initialized and waiting for data.

Notes: The job may or may not require Static or Dynamic Data

If data is required, it is the responsibility of Host to send it

Data: none

Action: Host sends first data record and will recieve [DATA\_BUFFER\_AVAILABLE](#_DATA_BUFFER_AVAILABLE)

Additional data records can be sent after recieveing data buffer available

Description: This command is used by the Host to control the [Start Printing Sequence](#_Start_Printing_Sequence)

Example:

<Received> - READY\_TO\_PRINT ......m$. [02 04 00 00 00 00 6D 24 03 ]

### SELECT\_VEGA\_PRINT\_BUTTON\_REMOTE 0x91

ID: 0x91 Direction: Host 🡪 Vega

Purpose: This message selects (remotely presses) the Vega <Print> Button

Notes: This is the same as an operator pressing the print button

The Host is responsible for knowing the current state

Data: None

Action: Host verifies the current state [GET\_VEGA\_PRINT\_BUTTON\_STATE\_REMOTE](#_GET_VEGA_PRINT_BUTTON_STATE_REMOTE)

Host then knows what the expected state change should be after the “press”

Description: This command is an ALTERNATIVE to the [Start Printing Sequence](#_Start_Printing_Sequence)

This is an older command that starts a job without data and enables the trigger sensor

It is expected that the Host has control over the product flow to prevent data underruns

Example:

<Sent> - SELECT\_VEGA\_PRINT\_BUTTON\_REMOTE......... [02 04 00 00 00 00 00 91 03 ]

… *Note: the original and new print button state will always be returned*

<Received> - TM\_GUI\_PRINT\_BUTTON\_STATE\_CHANGED………. [02 04 00 00 00 00 00 9D 00 02 03 ]

## Setup Messages

### SET\_TERMINAL\_MODE\_CONFIG\_REMOTE 0x71

ID: 0x71 Direction: Host 🡪 Vega

Purpose: Sets the Terminal mode configuration settings like error reporting and handshaking.

Notes: Must be sent at start of each session (Vega does not save values)

All text is UTF-8 (not 2-byte UTF-8)

ucEnableErrorMessages accepts 0,1, or a 2 for Verbose mode

Data:

typedef struct {  
 unsigned char ucSpare; // unused  
 unsigned char ucEnableChecksums;  
 unsigned char ucEnableTransmitNumbers;  
 unsigned char ucAckMessagesFromVegaApp;  
 unsigned char ucAckMessagesFromRemoteApp;  
 unsigned char ucEnableErrorMessages;  
} SSetTerminalModeConfigRemote;

Description:

ucSpare - ***unused*** *as all data is UTF-8 format*

ucEnableChecksums - Checksum generation/checking

0 = Disable, 1 = Enable

ucEnableTransmitNumbers - Transmit number generation/checking

0 = Disable, 1 = Enable

ucAckMessagesFromVegaApp - Host applications should Acknowledge messages

0 = Disable, 1 = Enable

ucAckMessagesFromRemoteApp - Vega application should Acknowledge messages

0 = Disable, 1 = Enable

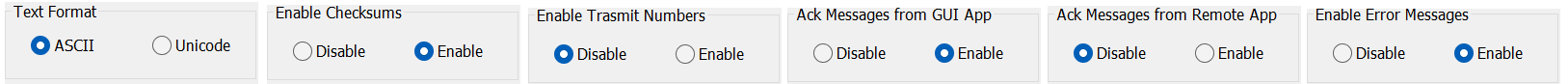
ucEnableErrorMessages - Transmission of Error Messages from the Vega to the Host

0 = Disable, 1 = Enable, 2 = Verbose mode

Verbose mode adds additional information (not just a number) to Error and Warning messages

See [ERROR\_MESSAGE\_REPLY 0x94](#_ERROR_MESSAGE_REPLY_0x94) and [WARNING\_MESSAGE\_REPLY 0xB6](#_WARNING_MESSAGE_REPLY_0xB6)

Example:



To set the above enables send the following:

<Sent> - SET\_TERMINAL\_MODE\_CONFIG\_REMOTE .......q....... [02 0A 00 00 00 00 00 71 00 01 00 01 00 01 03 ]

### GET\_VERSIONS 0xB4

ID: 0xB4 Direction: Host 🡪 Vega

Purpose: Host can request the version of the Vega application and the Terminal Mode interface specification.

Notes: The host should verify that the Vega TM API version matches (or is compatible)

In general, but not guaranteed, the Vega TM API is backwards compatible

You should be able to use a newer version of the Vega TM API than the one developed for.

Response: [VERSIONS\_REPLY](#_VERSIONS_REPLY)

Data: None

Description: This feature should eliminate side effects due to out of date software.

See: [Setup and Configuration](#_Setup_and_Configuration)

Example:

<Sent> - SET\_TERMINAL\_MODE\_CONFIG\_REMOTE .......q....... [02 0A 00 00 00 00 00 71 00 01 00 01 00 01 03 ]

… response summary (from RemAppSim):

Versions: Application 1.1.10.7, API 0.0.0.0, Ink Types: 1.0.40

Example:

### VERSIONS\_REPLY 0xB5

ID: 0xB5 Direction: Vega 🡪 Host

Purpose: This message provides the version numbers for those aspects of Vega that are versioned.

Notes: The host should verify that the Vega TM API version matches (or is compatible)

In general, but not guaranteed, the Vega TM API is backwards compatible

You should be able to use a newer version of the Vega TM API than the one developed for.

Requester: [GET\_VERSIONS](#_GET_VERSIONS_1)

Data:

char[20] ApplicationVersion  
char[20] APIVersion  
char[20] InkTypesVersion

Description:

Format Major.Mid.Minor.Build - Build = 0 for Production Release

See: [Setup and Configuration](#_Setup_and_Configuration)

Example:

The 0x40 (64 byte) message contains the three strings below with the 0x2e (‘.’ Character) between the digits

02 40 00 00 00 00 01 b5 31 2e 31 2e 31 30 2e 37 .@......1.1.10.7

00 00 00 00 00 00 00 00 00 00 00 00 30 2e 30 2e ............0.0.

30 2e 30 00 00 00 00 00 00 00 00 00 00 00 00 00 0.0.............

31 2e 30 2e 34 30 00 00 00 00 00 00 00 00 00 00 1.0.40..........

00 00 00 00 03 .....

## Job Control Group

### LOAD\_ PRINT\_JOB\_REMOTE 0x76

ID: 0x76 Direction: Host application 🡪 Vega application

Purpose: Loads a specified Job for printing and automatically switches to the Home Screen.

Notes: There must be a job in the local Vega “Jobs” directory. Otherwise a NACK 5 “Job Not Found” will be sent.

If Vega is currently editing a configuration then a NACK 6 “User Edit in Progress” will be sent.

Data: char\* utf8JobName; // UTF-8

Response: The job will load and [GET\_LOADED\_PRINT\_JOB\_REMOTE](#_GET_LOADED_PRINT_JOB_REMOTE) will return the job name

Description: The job name is the one shown in the Vega Job List.

The Host can remotely [IMPORT\_JOB](#_IMPORT_JOB) if necessary as it MUST be local

Example:

To load a job found in the Vega local jobs directory named “Terminal Mode 1 testing”

<Sent> - LOAD\_PRINT\_JOB\_REMOTE .......vTerminal.Mode.1.testing..............

0000 02 00 00 00 45 00 01 31 cb 21 40 00 80 06 00 00 ....E..1.!@.....

0010 7f 00 00 01 7f 00 00 01 fc 2a 27 11 38 46 ff 74 .........\*'.8F.t

0020 57 41 10 fd 50 18 20 fa 56 2f 00 00 02 04 01 00 WA..P. .V/......

0030 00 00 00 76 54 65 72 6d 69 6e 61 6c 20 4d 6f 64 ...vTerminal Mod

0040 65 20 31 20 74 65 73 74 69 6e 67 00 00 00 00 00 e 1 testing.....

0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................

0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................

0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................

*\* \* \* all zeros*

0120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................

0130 00 00 00 00 03 .....

### LOAD\_JOB\_FOR\_EDIT 0x17

ID: 0x17 Direction: Host 🡪 Vega

Purpose: Loads a specified Job to Edit and switches to the job edit screen.

Notes: Intended workflow is where the Vega applicaiton is hidden during printing and the host

wants to present the user with the job edit dialogs and nothting else

The Vega window cannot be resized or the application closed while in “Edit” mode.

There must be a job in Vega’s local “Jobs” directory else NACK 5 “Job Not Found”

If Vega is currently editing a configuration then NACK 6 “User Edit in Progress”

Data:

Unsigned char ucCreateNewJob;

char\* utf8JobName; // UTF-8

Workflow:

* Typically, Vega is being used as a back end and is hidden.
* The host sends the load job for edit command
* The host makes Vega visible
* The user can make edits
* When the user is done with the editing, they can save the job
* When the job is saved the command [USER\_ACTION\_NOTIFICATION 0x29](#_USER_ACTION_NOTIFICATION_(0x29)) informs host
* When the user wants to exit the edit process, they will need to press the X close button
* If there are unsaved edits the user will be prompted to save or cancel the edits.
* The command [USER\_ACTION\_NOTIFICATION 0x29](#_USER_ACTION_NOTIFICATION_(0x29)) is sent telling host that the edit operation is complete
* The host will then typically hide the application and then load a job for printing

Example:

<Sent> - LOAD\_JOB\_FOR\_EDIT .........test.job.a. [02 0F 00 00 00 00 00 17 00 74 65 73 74 20 6A 6F 62 20 61 03 ]

<Received> - USER\_ACTION\_NOTIFICATION .\*.....)admin,Technician,2,Save.job,test.job.a. [02 2A 00 00 00 00 0A 29 61 64 6D 69 6E 2C 54 65 63 68 6E 69 63 69 61 6E 2C 32 2C 53 61 76 65 20 6A 6F 62 2C 74 65 73 74 20 6A 6F 62 20 61 03 ] User Action : Name: admin Name: Technician Name: 2 Name: Save job Name: test job a

<Received> - USER\_ACTION\_NOTIFICATION .2.....)admin,Technician,7,Editing.complete,test.job.a. [02 32 00 00 00 00 0B 29 61 64 6D 69 6E 2C 54 65 63 68 6E 69 63 69 61 6E 2C 37 2C 45 64 69 74 69 6E 67 20 63 6F 6D 70 6C 65 74 65 2C 74 65 73 74 20 6A 6F 62 20 61 03 ]

User Action: Name: admin Name: Technician Name: 7 Name: Editing complete Name: test job a

<Sent> - REMOTE\_ACKNOWLEDGE\_MSG ........).. [02 06 00 00 00 00 00 99 29 0B 03 ]

Description:

ucCreateNewJob- 0=no 1=yes

If **no** then the job must be found from the stored jobs (an NACK if the specific job is not found).

If **yes** then the job must not be found in the stored jobs (an NACK if the specific job is found).

szJobName- Specifies the name of the job as it shows in the Vega Job list.

Example:

Note: host needs to determine the length of the string- see [Determining length of Strings](#_Determining_length_of)

<Sent> - LOAD\_JOB\_FOR\_EDIT .........Terminal.Mode.1.testing. [02 1C 00 00 00 00 00 17 00 54 65 72 6D 69 6E 61 6C 20

4D 6F 64 65 20 31 20 74 65 73 74 69 6E 67 03 ]

### GET\_LOADED\_PRINT\_JOB\_REMOTE 0x77

ID: 0x77 Direction: Host 🡪 Vega

Purpose: To check the name of the Print Job that is loaded (if any)

Notes: There must be a job loaded else NACK 5 “Job Not Found”

If Vega is currently editing a configuration then NACK 6 “User Edit in Progress”

Data: None

Response: [LOADED\_PRINT\_JOB\_REPLY](#_LOADED_DYN_DATA_FILE_REPLY)

Description: Verify which job is loaded for printing.

Example:

<Sent> - LOAD\_JOB\_FOR\_EDIT .........Terminal.Mode.1.testing. [02 1C 00 00 00 00 00 17 00 54 65 72 6D 69 6E 61 6C 20

4D 6F 64 65 20 31 20 74 65 73 74 69 6E 67 03 ]

Note: if no job is loaded

### LOADED\_PRINT\_JOB\_REPLY 0x78

ID: 0x78 Direction: Vega 🡪 Host

Purpose: To return the name of the currently loaded print job as UTF-8

Notes: There must be a job loaded else NACK 5 “Job Not Found”

If Vega is currently editing a configuration then NACK 6 “User Edit in Progress”

Requester: [GET\_LOADED\_PRINT\_JOB\_REMOTE](#_GET_LOADED_PRINT_JOB_REMOTE)

Data: unsigned short int[256] utf8JobName; //UTF-8

Description: The filename will not contain an extension and will be returned only if found else NACK.

szJobName - Specifies the name of the Print Job that is loaded.

Example:

<Received> - LOADED\_PRINT\_JOB\_REPLY .......xTerminal Mode 1 testing.

[ 02 1b 00 00 00 00 04 78 54 65 72 6d 69 6e 61 6c 20 4d 6f 64 65 20 31 20 74 65 73 74 69 6e 67 03 ]

### DELETE\_PRINT\_JOB 0x1c

ID: 0x1C Direction: Host 🡪 Vega

Purpose: Delete a specified Print Job from the Vega “Jobs” Directory

Notes: There must be a job loaded else NACK 5 “Job Not Found”

If Vega is currently editing a configuration then NACK 6 “User Edit in Progress”

Data: char\* utf8JobName; // UTF-8

Description: The filename will not contain an extension and will be returned only if found else NACK.

utf8JobName - This is the name of the job as found in the job list

Example: delete “dynamic.data.only”

<Sent> - DELETE\_PRINT\_JOB ........dynamic.data.only.

[02 15 00 00 00 00 00 1C 64 79 6E 61 6D 69 63 20 64 61 74 61 20 6F 6E 6C 79 03 ]

### REQUEST\_PRINT\_JOB\_LIST 0x1A

ID: 0x1A Direction: Host 🡪 Vega

Purpose: Get list of all jobs currently available within Vega (for printing, editing, or deleting)

Note: The Host can select the sort order of the returned list of local jobs

If no jobs are present locally then NACK 5 “Job Not Found”

If sort variables are out of range then NACK 8 “argument out of range”

Response: [PRINT\_JOB\_LIST\_REPLY](#_PRINT_JOB_LIST_REPLY)

Data:

Unsigned char ucSortType;

Unsigned char ucSortOrder;

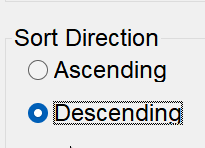
Description: the sort order of the list returned is defined below.

ucSortType - 1=Name, 2=Last Printed, 3=Last Edited

ucSortOrder - 1=Up, 2=Down

Example: Request list with sort by “Name” and “Descending”

Graphical user interface, text, application, chat or text message

Description automatically generated 

<Sent> - REQUEST\_PRINT\_JOB\_LIST ........... [02 06 00 00 00 00 00 1A 01 02 03 ]

### PRINT\_JOB\_LIST\_REPLY 0x1B

ID: 0x1B Direction: Vega 🡪 Host

Purpose: Provide a list of local jobs available

Notes: The job list is returned as delimited text

if no jobs exist then a FF (Form feed will be returned).

Requester: [REQUEST\_PRINT\_JOB\_LIST](#_REQUEST_PRINT_JOB_LIST)

Data: One comma separated string with integrated labels and a Form Feed at the end of each entry.

Description: The format is as follows with the “?” position containing the respective data.

Job name=?, Last printed date=?,Last printed time=?, Last edited date=?,Last edited time=?FF

...

Format for Date and time shown below:

Job name=Test Job,Last printed date=08/10/2022,Last printed time=04:14:52 PM,

Last edited date=08/08/2022,Last edited time=10:14:52 AMFF

Example: highlights are for the second job in the sort named “dynamic data only”

Note the 0x0C FF record breaks and the 0x2C for comma field delimeter and the 0x3d for equal sign

10/21/2022 15:26:02:507 <Received> - PRINT\_JOB\_LIST\_REPLY ......q.Job.name=b..............e,.Last.printed.date=10/19/2022,Last.printed.time=09:28:09.AM,.Last.edited.date=10/19/2022,Last.edited.time=01:38:24.PM.Job.name=dynamic.data.only,.Last.printed.date=10/21/2022,Last.printed.time=03:16:07.PM,.Last.edited.date=10/21/2022,Last.edited.time=02:20:07.PM.Job.name=static.data.only,.Last.printed.date=10/21/2022,Last.printed.time=02:17:58.PM,.Last.edited.date=10/21/2022,Last.edited.time=02:15:28.PM.Job.name=tm1.with.dynamic.and.static,.Last.printed.date=,Last.printed.time=,.Last.edited.date=10/21/2022,Last.edited.time=01:26:46.PM.Job.name=update.user.variable.in.TM1,.Last.printed.date=10/19/2022,Last.printed.time=09:28:09.AM,.Last.edited.date=10/20/2022,Last.edited.time=03:34:15.PM.. [02 D6 02 00 00 00 71 1B 4A 6F 62 20 6E 61 6D 65 3D 62 20 E6 9C 80 E5 B8 B8 E7 94 A8 E5 AD 97 20 65 2C 20 4C 61 73 74 20 70 72 69 6E 74 65 64 20 64 61 74 65 3D 31 30 2F 31 39 2F 32 30 32 32 2C 4C 61 73 74 20 70 72 69 6E 74 65 64 20 74 69 6D 65 3D 30 39 3A 32 38 3A 30 39 20 41 4D 2C 20 4C 61 73 74 20 65 64 69 74 65 64 20 64 61 74 65 3D 31 30 2F 31 39 2F 32 30 32 32 2C 4C 61 73 74 20 65 64 69 74 65 64 20 74 69 6D 65 3D 30 31 3A 33 38 3A 32 34 20 50 4D 0C 4A 6F 62 20 6E 61 6D 65 ***3D*** 64 79 6E 61 6D 69 63 20 64 61 74 61 20 6F 6E 6C 79 2C 20 4C 61 73 74 20 70 72 69 6E 74 65 64 20 64 61 74 65 ***3D*** 31 30 2F 32 31 2F 32 30 32 32 2C 4C 61 73 74 20 70 72 69 6E 74 65 64 20 74 69 6D 65 ***3D*** 30 33 3A 31 36 3A 30 37 20 50 4D 2C 20 4C 61 73 74 20 65 64 69 74 65 64 20 64 61 74 65 ***3D*** 31 30 2F 32 31 2F 32 30 32 32 2C 4C 61 73 74 20 65 64 69 74 65 64 20 74 69 6D 65 ***3D*** 30 32 3A 32 30 3A 30 37 20 50 4D 0C 4A 6F 62 20 6E 61 6D 65 3D 73 74 61 74 69 63 20 64 61 74 61 20 6F 6E 6C 79 2C 20 4C 61 73 74 20 70 72 69 6E 74 65 64 20 64 61 74 65 3D 31 30 2F 32 31 2F 32 30 32 32 2C 4C 61 73 74 20 70 72 69 6E 74 65 64 20 74 69 6D 65 3D 30 32 3A 31 37 3A 35 38 20 50 4D 2C 20 4C 61 73 74 20 65 64 69 74 65 64 20 64 61 74 65 3D 31 30 2F 32 31 2F 32 30 32 32 2C 4C 61 73 74 20 65 64 69 74 65 64 20 74 69 6D 65 3D 30 32 3A 31 35 3A 32 38 20 50 4D 0C 4A 6F 62 20 6E 61 6D 65 3D 74 6D 31 20 77 69 74 68 20 64 79 6E 61 6D 69 63 20 61 6E 64 20 73 74 61 74 69 63 2C 20 4C 61 73 74 20 70 72 69 6E 74 65 64 20 64 61 74 65 3D 2C 4C 61 73 74 20 70 72 69 6E 74 65 64 20 74 69 6D 65 3D 2C 20 4C 61 73 74 20 65 64 69 74 65 64 20 64 61 74 65 3D 31 30 2F 32 31 2F 32 30 32 32 2C 4C 61 73 74 20 65 64 69 74 65 64 20 74 69 6D 65 3D 30 31 3A 32 36 3A 34 36 20 50 4D 0C 4A 6F 62 20 6E 61 6D 65 3D 75 70 64 61 74 65 20 75 73 65 72 20 76 61 72 69 61 62 6C 65 20 69 6E 20 54 4D 31 2C 20 4C 61 73 74 20 70 72 69 6E 74 65 64 20 64 61 74 65 3D 31 30 2F 31 39 2F 32 30 32 32 2C 4C 61 73 74 20 70 72 69 6E 74 65 64 20 74 69 6D 65 3D 30 39 3A 32 38 3A 30 39 20 41 4D 2C 20 4C 61 73 74 20 65 64 69 74 65 64 20 64 61 74 65 3D 31 30 2F 32 30 2F 32 30 32 32 2C 4C 61 73 74 20 65 64 69 74 65 64 20 74 69 6D 65 3D 30 33 3A 33 34 3A 31 35 20 50 4D 0C 03 ]

Job name=b 最常用字 e, Last printed date=10/19/2022,Last printed time=09:28:09 AM, Last edited date=10/19/2022,Last edited time=01:38:24 PM

Job name=dynamic data only, Last printed date=10/21/2022,Last printed time=03:16:07 PM, Last edited date=10/21/2022,Last edited time=02:20:07 PM

Job name=static data only, Last printed date=10/21/2022,Last printed time=02:17:58 PM, Last edited date=10/21/2022,Last edited time=02:15:28 PM

Job name=tm1 with dynamic and static, Last printed date=,Last printed time=, Last edited date=10/21/2022,Last edited time=01:26:46 PM

Job name=update user variable in TM1, Last printed date=10/19/2022,Last printed time=09:28:09 AM, Last edited date=10/20/2022,Last edited time=03:34:15 PM

### EXPORT\_JOB 0x19

ID: 0x19 Direction: Host 🡪 Vega

Purpose: Copies a specified Vega Job (folder) to a full source path destination.

Notes: The job is copied (not moved)

The job can be deleted if desired [DELETE\_PRINT\_JOB](#_DELETE_PRINT_JOB)

If the job is not found in the Vega jobs list then NACK 5 “job not found”

If the destination path is not found, then a NACK 7 “directory not found”

Jobs can conversely be imported using [IMPORT\_JOB](#_IMPORT_JOB)

Strings are NOT NULL terminated

Data:

char\*utf8JobName; // UTF-8

const char delimiter = 0x44; // comma character at end of job name

char\*utf8JobPath; // UTF-8

Description: Both strings are UTF-8 and a comma character constant separates them

szJobName job name as found in Vega Jobs List

szJobPath destination file path

Example: export the job named “dynamic data only” to the folder named “d:\my jobs”

<Sent> - EXPORT\_JOB ........dynamic.data.only,D:\my.jobs.

[02 20 00 00 00 00 00 19 64 79 6E 61 6D 69 63 20 64 61 74 61 20 6F 6E 6C 79 2C 44 3A 5C 6D 79 20 6A 6F 62 73 03 ]

### IMPORT\_JOB 0x18

ID: 0x18 Direction: Host 🡪 Vega

Purpose: Copies a specified Job (folder) from a full source path to the Vega Jobs folder.

Notes: The job is copied (not moved)

If the Vega job exists then 12 “Cannot overwrite job must delete first”

If the destination path is not found, then a NACK 7 “directory not found”

Jobs can conversely be imported using [IMPORT\_JOB](#_IMPORT_JOB)

Strings are NOT NULL terminated

Data:

On

is folder with multiple files and the name of the

the folder is the name of the

Example: import job named “dynamic data only” from the folder “D:\my jobs”

<Sent> - IMPORT\_JOB ........dynamic.data.only,D:\my.jobs.

[02 20 00 00 00 00 00 18 64 79 6E 61 6D 69 63 20 64 61 74 61 20 6F 6E 6C 79 2C 44 3A 5C 6D 79 20 6A 6F 62 73 03 ]

### REQUEST\_JOB\_THUMBNAIL\_IMAGE 0x1d

ID: 0x1D Direction: Host 🡪 Vega

Purpose: Request the thumbnail image for a specific job

Notes: This is used in conjunction with [PRINT\_JOB\_LIST\_REPLY](#_PRINT_JOB_LIST_REPLY) for host to show user

If the job cannot be found, then NACK 5 “job not found”

Strings are NOT NULL terminated

Response: [JOB\_THUMBNAIL\_IMAGE\_REPLY](#_JOB_THUMBNAIL_IMAGE_REPLY)

Data: char\* utf8JobName; // UTF-8

Description: UTF-8

szJobName This is the name of the job as found in the job list

Example: request the thumbnail image for job named “dynamic data only”

It would display in the Vega Jobs list as

Graphical user interface, text

Description automatically generated with medium confidence

<Sent> - REQUEST\_JOB\_THUMBNAIL\_IMAGE ........dynamic.data.only.

[02 15 00 00 00 00 00 1D 64 79 6E 61 6D 69 63 20 64 61 74 61 20 6F 6E 6C 79 03 ]

### JOB\_THUMBNAIL\_IMAGE\_REPLY 0x1E

ID: 0x1E Direction: Vega 🡪 Host

Purpose: Request the thumbnail image for a specific job

Notes: This is used in conjunction with [PRINT\_JOB\_LIST\_REPLY](#_PRINT_JOB_LIST_REPLY) for host to show user

Requester: [REQUEST\_JOB\_THUMBNAIL\_IMAGE](#_REQUEST_JOB_THUMBNAIL_IMAGE)

Data: Unsigned char ucPNG\_Thumbnail[];

Description: The image will be returned as a .PNG (Portable Network Graphics)

ucPNG\_Thumbnail PNG bitmap

Example: the reply to the request the thumbnail image for job named “dynamic data only”

The message in hex is much to large and not useful to display here for the returned PNG image file.

Note: when sending the REQUEST\_JOB\_THUMBNAIL\_IMAGE command in the RemAppSim tester application you will receive the image in a pop-up window and the image can be saved to disk for reference.

Text

Description automatically generated

## Database Control Group

### LOAD\_DYN\_DATA\_FILE\_REMOTE 0x79

ID: 0x79 Direction: Host 🡪 Vega

Purpose: Loads a database that is stored locally in Vega Database folder for printing

Notes: Database Files must be stored locally (not mapped to a network drive)

If the database is not local it must be copied/ moved see: [Database Files](#_Database_Files)

If a job is currently printing, then NACK 1 “Not allowed while printing”

Data: unsigned char\* utf8DatabaseName;

Description:

utf8DatabaseName - The name of the database as shown in Vega in the database list.

Example: load the database named “samples”

<Sent> - LOAD\_DYN\_DATA\_FILE\_REMOTE .......ysamples. [02 0B 00 00 00 00 00 79 73 61 6D 70 6C 65 73 03 ]

### GET\_LOADED\_DYN\_DATA\_FILE\_REMOTE 0x7a

ID: 0x7a Direction: Host 🡪 Vega

Purpose: Requests the name of the database file that is loaded.

Notes: The file loaded is located locally

This is ONLY for Index Mode (not TM1)

Response: [LOADED\_DYN\_DATA\_FILE\_REPLY](#_LOADED_DYN_DATA_FILE_REPLY)

Data: None

Description: Request The name of the database as shown in Vega in the database list.

Example:

<Sent> - GET\_LOADED\_DYN\_DATA\_FILE\_REMOTE .......z. [02 04 00 00 00 00 00 7A 03 ]

### LOADED\_DYN\_DATA\_FILE\_REPLY 0x7b

ID: 0x7b Direction: Vega 🡪 Host

Purpose: Let the host verify the currently loaded database for printing

Notes: The name of the database does not a path

This is ONLY for Index Mode (not TM1)

Requester: [GET\_LOADED\_DYN\_DATA\_FILE\_REMOTE](#_GET_LOADED_DYN_DATA_FILE_REMOTE)

Data: unsigned char\* utf8DatabaseName;

Description:

utf8DatabaseName - The name of the currently loaded database.

Example: This is a loaded database with UTF-8 characters.

<Received> - LOADED\_DYN\_DATA\_FILE\_REPLY .......{.............txt. [02 14 00 00 00 00 01 7B E5 BA B7 E7 86 99 E5 AD 97 E5 85 B8 2E 74 78 74 03 ]

Loaded Datafile: 康熙字典.txt

## System Configuration Group

### LOAD\_SYSTEM\_CONFIG\_REMOTE 0x97

ID: 0x97 Direction: Host 🡪 Vega

Purpose: Loads a specified System Configuration that is in the Vega Configurations folder

Notes: This can only be executed when not printing else NACK 1 “Not allowed while printing”

If the configuration is not found then NACK 13 “No Configuration found/ Loaded”

Data: char\* utf8ConfigurationFilename

Description:

utf8ConfigurationFilename - Specifies the name of the System Configuration that is to be loaded.

Example:

<Sent> - LOAD\_SYSTEM\_CONFIG\_REMOTE ........test.config.

[02 0F 00 00 00 00 00 97 74 65 73 74 20 63 6F 6E 66 69 67 03 ]

### GET\_LOADED\_SYSTEM\_CONFIG\_REMOTE 0xA0

ID: 0xA0 Direction: Host 🡪 Vega

Purpose: Requests the name of the currently loaded System Configuration.

Response: [LOADED\_SYSTEM\_CONFIG\_REPLY](#_LOADED_SYSTEM_CONFIG_REPLY)

Notes: This can be requested anytime

If no configuration is loaded, then NACK 13 “No Configuration found/ Loaded”

Data: None

Description: This will get the UTF-8 name for the current Printer Configuration

Example:

<Sent> - GET\_LOADED\_SYSTEM\_CONFIG\_REMOTE ......... [02 04 00 00 00 00 00 A0 03 ]

### LOADED\_SYSTEM\_CONFIG\_REPLY 0xA1

ID: 0xA1 Direction: Vega 🡪 Host

Purpose: Returns the name of the currently loaded System Configuration.

Requester: [GET\_LOADED\_SYSTEM\_CONFIG\_REMOTE](#_GET_LOADED_SYSTEM_CONFIG_REMOTE)

Notes: If the request was NACKed this response will be a empty

Data: char\* utf8ConfigurationFilename

Description:

szConfiguration - Specifies the name of the currently loaded system configuration.

Example: note UTF-8 characters in middle of string

<Received> - LOADED\_SYSTEM\_CONFIG\_REPLY ........configStart............End.

[02 1E 00 00 00 00 01 A1 63 6F 6E 66 69 67 53 74 61 72 74 E5 A6 82 E6 9E 9C E6 88 91 E6 98 AF 45 6E 64 03 ]

Loaded Configuration: configStart如果我是End

### TM\_SET\_VEGA\_WINDOW\_STATE 0x9e

ID: 0x9E Direction: Host 🡪 Vega

Purpose: To hide the Vega application from the operator

Notes: This can be sent ONLY when on the Home page

If not on home page 🡪 NACK 2: Hide window only available on home page

Data: unsigned char windowState;

Description:

1= hide  
2= minimize  
3= restore  
4= maximize

When Vega is hidden and the restore or maximize command is sent, the Vega window will be left behind any other open windows. However, when Vega is minimized and the restore or maximize command is sent Vega will be placed in front of all other open windows.

Example: Hide command

<Sent> - TM\_SET\_GUI\_WINDOW\_STATE .......... [02 05 00 00 00 00 00 9E 01 03 ]

### REQUEST\_UPDATE\_FIRMWARE 0x1F

ID: 0x1F Direction: Host 🡪 Vega

Purpose: Request that a specific Printer ID have its firmware updated

Notes: See: [Updating Firmware Remotely](#_Updating_Firmware_Remotely)

If ucPrinterID is out of range then NACK 8 “argument out of range.”

Some printers have multiple boards and will use more than one ID

Vega must be in hidden mode, see [TM\_SET\_GUI\_WINDOW\_STATE 0x9e](#_TM_SET_VEGA_WINDOW_STATE)

Response: [UPDATE\_FIRMWARE\_COMPLETE](#_UPDATE_FIRMWARE_COMPLETE)

Data: Unsigned char ucPrinterID;

Description: ucPrinterID Printer ID starting at 1

The index is based on the printer “board”. For example a 4 inch wing has two internal “boards”.

Example:

<Sent> - REQUEST\_UPDATE\_FIRMWARE .......... [02 05 00 00 00 00 00 1F 01 03 ]

Example response if the application is not hidden:

<Received> - REMOTE\_NEGATIVE\_ACKNOWLEDGE\_MSG

Reason: 2, Description: Only allowed in hidden mode using TM\_SET\_VEGA\_WINDOW\_STATE to set state to 1

### UPDATE\_FIRMWARE\_COMPLETE 0x20

ID: 0x20 Direction: Host 🡪 Vega

Purpose: Result of Firmware update

Notes: See: [Updating Firmware Remotely](#_Updating_Firmware_Remotely)

Requester: [REQUEST\_UPDATE\_FIRMWARE](#_REQUEST_UPDATE_FIRMWARE)

Data:

Unsigned char ucResult;

String szReasonForFailure; // UTF-8

Description: ucPrinterID Printer ID starting at 1

ucResult 1=Success, 0=Failure

szReasonForFailure string describing reason for failure. ex. “Timeout”

Example:

<Received> - UPDATE\_FIRMWARE\_COMPLETE .R....z..Firmware.update.failed.for.printer.pc.board.1:.Timed.out.waiting.for.progress. [02 52 00 00 00 00 7A 20 00 46 69 72 6D 77 61 72 65 20 75 70 64 61 74 65 20 66 61 69 6C 65 64 20 66 6F 72 20 70 72 69 6E 74 65 72 20 70 63 20 62 6F 61 72 64 20 31 3A 20 54 69 6D 65 64 20 6F 75 74 20 77 61 69 74 69 6E 67 20 66 6F 72 20 70 72 6F 67 72 65 73 73 03 ]

Firmware update successful

## System Query Group

### SEND\_CONTACT\_IMAGER\_REMOTE 0x72

ID: 0x72 Direction: Host 🡪 Vega

Purpose: Verify that all the printers are on-line

Notes: This is generally used for debugging to see what printer is not responding on the network

Response: [CONTACT\_IMAGER\_REPLY](#_CONTACT_IMAGER_REPLY)

Data: None

Description: This is a one command that will trigger possible multiple responses

Example:

<Sent> - SEND\_CONTACT\_IMAGER\_REMOTE .......r. [02 04 00 00 00 00 00 72 03 ]

### CONTACT\_IMAGER\_REPLY 0x73

ID: 0x73 Direction: Vega 🡪 Host

Purpose: For each printer in the configuration to respond to verify that it is communicating with Vega

Notes: There will be a separate response message for EACH printer in the system

This command does not return a list of printers and their status

Requester: [SEND\_CONTACT\_IMAGER\_REMOTE](#_SEND_CONTACT_IMAGER_REMOTE)

Data:

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned long int uliFirmwareVersion;  
 unsigned long int uliGateArrayVersion;  
} SContactImagerReply;

Description:

ucImagerNum - (1 – 32) = Imager number that the data corresponds to.

uliFirmwareVersion - The Firmware Version number of the Imager.

uliFirmwareVersion - The Gate Array Version number of the Imager.

Example:

<Received> - CONTACT\_IMAGER\_REPLY ......us..O....... [02 0D 00 00 00 00 75 73 00 00 4F FC C2 01 00 00 00 03 ]

ucImagerNum: 0, uliFirmwareVersion: -1023652096, uliGateArrayVersion: 1

### GET\_PRINTER\_STATE\_REMOTE 0x74

ID: 0x74 Direction: Host 🡪 Vega

Purpose: Requests the Printer State information.

Notes: This is a compact command for getting the system state (ie. What is printing?)

For more detailed state information use [GET\_STATS](#_GET_STATS)

Response: [PRINTER\_STATE\_REPLY](#_PRINTER_STATE_REPLY)

Data: None

Description: This is a compact message for finding out what, if anything, is printing

Example:

<Sent> - GET\_PRINTER\_STATE\_REMOTE .......t. [02 04 00 00 00 00 00 74 03 ]

### PRINTER\_STATE\_REPLY 0x75

ID: 0x75 Direction: Vega 🡪 Host

Purpose: Vega will return Printer State information in reply to the GET\_PRINTER\_STATE\_REMOTE message.

Notes:

Requester: [GET\_PRINTER\_STATE\_REMOTE](#_GET_PRINTER_STATE_REMOTE)

Data:

typedef struct {  
 unsigned char ucJobLoaded;  
 unsigned char ucDynamicDataFileLoaded;  
 unsigned char ucPrintButtonState;  
} SPrinterStateReply;

Description:

ucJobLoaded - 0 = No Job loaded, 1 = Job Loaded

ucDynamicDataFileLoaded - 0 = No File loaded, 1 = File Loaded

ucPrintButtonState -  
0 = Printing Stopped – idle

1 = Configuring printer(s)

2 = Printing

5 = Printer off-line, or no job is loaded

6 = Paused: Ready to Print or Stop

Example:

<Received> - PRINTER\_STATE\_REPLY ......ru.... [02 07 00 00 00 00 72 75 01 01 00 03 ]

ucJobLoaded: 1, ucDynamicDataFileLoaded: 1, ucPrintButtonState: 0

### GET\_VEGA\_PRINT\_BUTTON\_STATE\_REMOTE 0x92

ID: 0x92 Direction: Host 🡪 Vega

Purpose: This message requests the State of the Vega <Print> Button.

Notes: The host makes this request to get the overall state of the Vega system

The state of the print button is found in [PRINTER\_STATE\_REPLY](#_PRINTER_STATE_REPLY)

Reply: [VEGA\_PRINT\_BUTTON\_STATE\_REPLY](#_VEGA_PRINT_BUTTON_STATE_REPLY)

Data: None

Example:

<Sent> - GET\_GUI\_PRINT\_BUTTON\_STATE\_REMOTE ......... [02 04 00 00 00 00 00 92 03 ]

### VEGA\_PRINT\_BUTTON\_STATE\_REPLY 0x93

ID: 0x93 Direction: Vega 🡪 Host

Purpose: Proved the host with the Vega system state as shown by the print button

Notes: There is no text message associated with the state (see bitmaps below)

Disabled “Pause Printing” and “Print Buffers in Printer” see: [Start Printing Sequence](#_Start_Printing_Sequence)

Requester: [GET\_VEGA\_PRINT\_BUTTON\_STATE\_REMOTE](#_GET_NUM_LEFT_TO_PRINT_REMOTE)

Data: unsigned char ucPrintButtonState;

Description:

The ucPrintButtonState IDs, Print Button Bitmap, and Description are found in the table below.

|  |  |  |
| --- | --- | --- |
| ucPrintButton State | **Print Button Bitmap** | **Description** |
| 0 | A picture containing chart  Description automatically generated | Printing Stopped – idle |
| 1 | Icon  Description automatically generated | Configuring printer(s). |
| 2 | Chart  Description automatically generated | Printing |
| 5 | A picture containing diagram  Description automatically generated | Printer off-line, or no job is loaded. |
| 6 |  | Paused: Ready to Print or Stop |

Example:

<Received> - GUI\_PRINT\_BUTTON\_STATE\_REPLY ......p... [02 05 00 00 00 00 70 93 00 03 ]

ucPrintButtonState: 0= Idle

### TM\_VEGA\_PRINT\_BUTTON\_STATE\_CHANGED 0x9d

ID: 0x9D Direction: Vega 🡪 Host

Purpose: This message returns the previous and current Print Button State based on change.

Notes: This command is not requested it is sent on state change

Data: See: [VEGA\_PRINT\_BUTTON\_STATE\_REPLY](#_GUI_PRINT_BUTTON_STATE_REPLY)

unsigned char oldPrintMode;  
unsigned char newPrintMode;

Description: See: [VEGA\_PRINT\_BUTTON\_STATE\_REPLY](#_GUI_PRINT_BUTTON_STATE_REPLY)

Example:

<Received> - TM\_GUI\_PRINT\_BUTTON\_STATE\_CHANGED ......3.... [02 06 00 00 00 00 33 9D 01 00 03 ]

Received Print Mode Changed Message: oldPrintMode= Configuring, newPrintMode= Idle

### GET\_STATS 0x10

ID: 0x10 Direction: Host 🡪 Vega

Purpose: Request snapshot of the information shown on the Runtime Home Screen

Notes: This information is a text-based home screen

Data: none

Response: [STATS\_REPLY](#_STATS_REPLY)

Description: Vega will respond to the host with information that is found on the Home Run screen.

Example:

<Sent> - GET\_STATS ......... [02 04 00 00 00 00 00 10 03 ]

### STATS\_REPLY 0x11

ID: 0x11 Direction: Vega 🡪 Host

Purpose: Return snapshot of the information shown on the Runtime Home Screen

Requester: [GET\_STATS](#_GET_STATS)

Data: The data is formatted as a comma separated string

Description: One comma separated string with integrated labels.

*Note: there are NO CRLF in the message- shown here for clarity.*

“Printing=?,

Speed=?,SpeedUnits=?,

PiecesPerHour=?,LineCounter=?,JobCounter=?,SimpleCounter=?,

PCBuffers=?,PrinterBuffers=?,

InkTimeToEmpty=?,InkVolumeToEmpty=?,

PrinterStatusL1=?,PrinterStatusL2=?, PrinterStatusL3=?, PrinterStatusL4=?, PrinterStatusL5=?”

Example Data:

“Printing=?, yes | no

Speed=?, 20.4

SpeedUnits=?, SPEED\_INCHES\_PER\_SECOND | SPEED\_FEET\_PER\_MINUTE | SPEED\_METERS\_PER\_MINUTE

PiecesPerHour=?, 12,234

LineCounter=?, 234

JobCounter=?, 12321

SimpleCounter=?, 78

PCBuffers=?, 8

PrinterBuffers=?, 4

InkTimeToEmpty=?, 1:23

InkVolumeToEmpty=?, 55.43

InKLowAlarmActive=?, yes | no

InkEmptyAlarmActive=? yes | no

PrinterStatusL1=?, All Printers Ready to Print

PrinterStatusL2=?, ‘’

PrinterStatusL3=?, ‘’

PrinterStatusL4=?, ‘’

PrinterStatusL5=?” ‘’

Example:

<Received> - STATS\_REPLY .s....&.Printing=no,.Speed=19.96,.SpeedUnits=SPEED\_INCHES\_PER\_SECOND,.PiecesPerHour=0,.LineCounter=19114,.JobCounter=101,.SimpleCounter=0,.PCBuffers=0,PrinterBuffers=1,.InkTimeToEmpty=--:--,InkVolumeToEmpty=0.00,.InkLowAlarmActive=yes,.InkEmptyAlarmActive=yes,.PrinterStatusL1=All.Printers.Ready.to.Print,PrinterStatusL2=,.PrinterStatusL3=,.PrinterStatusL4=,.PrinterStatusL5=.

[02 73 01 00 00 00 26 11 50 72 69 6E 74 69 6E 67 3D 6E 6F 2C 20 53 70 65 65 64 3D 31 39 2E 39 36 2C 20 53 70 65 65 64 55 6E 69 74 73 3D 53 50 45 45 44 5F 49 4E 43 48 45 53 5F 50 45 52 5F 53 45 43 4F 4E 44 2C 20 50 69 65 63 65 73 50 65 72 48 6F 75 72 3D 30 2C 20 4C 69 6E 65 43 6F 75 6E 74 65 72 3D 31 39 31 31 34 2C 20 4A 6F 62 43 6F 75 6E 74 65 72 3D 31 30 31 2C 20 53 69 6D 70 6C 65 43 6F 75 6E 74 65 72 3D 30 2C 20 50 43 42 75 66 66 65 72 73 3D 30 2C 50 72 69 6E 74 65 72 42 75 66 66 65 72 73 3D 31 2C 20 49 6E 6B 54 69 6D 65 54 6F 45 6D 70 74 79 3D 2D 2D 3A 2D 2D 2C 49 6E 6B 56 6F 6C 75 6D 65 54 6F 45 6D 70 74 79 3D 30 2E 30 30 2C 20 49 6E 6B 4C 6F 77 41 6C 61 72 6D 41 63 74 69 76 65 3D 79 65 73 2C 20 49 6E 6B 45 6D 70 74 79 41 6C 61 72 6D 41 63 74 69 76 65 3D 79 65 73 2C 20 50 72 69 6E 74 65 72 53 74 61 74 75 73 4C 31 3D 41 6C 6C 20 50 72 69 6E 74 65 72 73 20 52 65 61 64 79 20 74 6F 20 50 72 69 6E 74 2C 50 72 69 6E 74 65 72 53 74 61 74 75 73 4C 32 3D 2C 20 50 72 69 6E 74 65 72 53 74 61 74 75 73 4C 33 3D 2C 20 50 72 69 6E 74 65 72 53 74 61 74 75 73 4C 34 3D 2C 20 50 72 69 6E 74 65 72 53 74 61 74 75 73 4C 35 3D 03 ]

Printing=no, Speed=19.96, SpeedUnits=SPEED\_INCHES\_PER\_SECOND, PiecesPerHour=0, LineCounter=19114, JobCounter=101,

SimpleCounter=0, PCBuffers=0, PrinterBuffers=1, InkTimeToEmpty=--:--,InkVolumeToEmpty=0.00,

InkLowAlarmActive=yes, InkEmptyAlarmActive=yes, PrinterStatusL1=All Printers Ready to Print,PrinterStatusL2=, PrinterStatusL3=, PrinterStatusL4=, PrinterStatusL5=

### GET\_PRINT\_PREVIEW 0x26

ID: 0x26 Direction: Host 🡪 Vega

Purpose: Request the preview for the currently loaded job

Response: [PRINT\_PREVIEW\_RESPONSE](#_PRINT_PREVIEW_RESPONSE)

Notes: If no job is loaded then NACK 4 “No Job Loaded”

Data: none

Description: This request will return a PNG (Portable Network Graphics) image

Example:

<Sent> - GET\_PRINT\_PREVIEW .......&. [02 04 00 00 00 00 00 26 03 ]

### PRINT\_PREVIEW\_RESPONSE 0x27

ID: 0x27 Direction: Vega 🡪 Host

Purpose: Return the first page image (bitmap) to Host for display for user

Requester: [GET\_PRINT\_PREVIEW](#_GET_PRINT_PREVIEW_1)

Notes: if “Save Image Preview” is enable the Vega local directory “Preview Image” will have a copy also

Data: unsigned char szJobPreview[];

Description: szJobPreview This is a PNG (Portable Network Graphics) image

Example: this is a preview from RemAppSim that show a user variable and dynamic record field.

This message is too large to decode here…

<Received> - PRINT\_PREVIEW\_RESPONSE

Text

Description automatically generated with medium confidence

### USER\_ACTION\_NOTIFICATION 0x29

ID: 0x29 Direction: Vega 🡪 Host

Purpose: the command alerts the host of specific user actions that at this time are related to file i/o; however, in the future other types of user operations could be added.

Notes: Currently only file I/O operations are reported but in the future additional actions may be added

7 Close Edit Job Screen action used with the command [LOAD\_JOB\_FOR\_EDIT 0x17](#_LOAD_JOB_FOR_EDIT)

Data: The command has the following structure:

USER\_ACTION\_NOTIFICATION 0x29 (UserName, UserRole, ActionID, ActionString, DescriptionString)

Where: a comma delimited string with 5 variable length arguments:

char\* utf8UserName

char\* utf8UserRole

char\* utf8ActionID

char\* utf8ActionString

char\* utf8DescriptionString

With the following descriptions:

UserName: this is the name of the currently logged in user

UserRole: this is the user’s login Role (privileges)

ActionID: the numeric value for the user action

ActionString: the text description for the ActionID

DescriptionString: additional information like filename

Example: (Admin, Technician, 1, Save job, Label UDI 2x4)

Argument Values:

ActionID ActionString DescriptionString

1 Load job for print name of job

2 Save job name of job

3 Save configuration name of config

4 Load Configuration name of config

5 Database loaded name of database

6 Database unloaded name of database

7 Close Edit Job Screen

## Print Data Group

### SEND\_DYNAMIC\_DATA\_TM1\_REMOTE 0x7d

ID: 0x7d Direction: Host 🡪 Vega

Purpose: Send a complete formatted Data Record for printing

Response: [DYNAMIC\_DATA\_PRINTED\_TM1\_REPLY](#_DYNAMIC_DATA_PRINTED_TM1_REPLY_1)

Notes: The field and record delimiters are defined in the sample database loaded in Vega

This data record must be sent for every piece

And this message can be combined with [SEND\_STATIC\_DATA\_TM1\_REMOTE](#_SEND_STATIC_DATA_TM1_REMOTE_1)

Data:

typedef struct {  
 unsigned long int uliRecordNumber;  
 unsigned long int uliNumTextChars;  
 szDynamicPrintData  
} SSendDynamicDataTM1Remote;

Description:

uliRecordNumber - The Record Number associated with the Dynamic Print Data contained in the message.

uliNumTextChars - The number of Text characters, contained in this message, to print.

szDynamicPrintData - The Dynamic Print Data to be printed for this record in UTF-8 format

Example:

<Sent> - SEND\_DYNAMIC\_DATA\_TM1\_REMOTE .......}........Record#20-Field#1.,Record#20-Field#2.,Record#20-Field#3.,Record#20-Field#4.,Record#20-Field#5.,Record#20-Field#6.,Record#20-Field#7.,Record#20-Field#8.... [02 A5 00 00 00 00 00 7D 14 00 00 00 99 00 00 00 52 65 63 6F 72 64 23 32 30 2D 46 69 65 6C 64 23 31 20 2C 52 65 63 6F 72 64 23 32 30 2D 46 69 65 6C 64 23 32 20 2C 52 65 63 6F 72 64 23 32 30 2D 46 69 65 6C 64 23 33 20 2C 52 65 63 6F 72 64 23 32 30 2D 46 69 65 6C 64 23 34 20 2C 52 65 63 6F 72 64 23 32 30 2D 46 69 65 6C 64 23 35 20 2C 52 65 63 6F 72 64 23 32 30 2D 46 69 65 6C 64 23 36 20 2C 52 65 63 6F 72 64 23 32 30 2D 46 69 65 6C 64 23 37 20 2C 52 65 63 6F 72 64 23 32 30 2D 46 69 65 6C 64 23 38 20 0D 0A 03 ]

### SEND\_STATIC\_DATA\_IM1\_REMOTE 0xA3

ID: 0xA3 Direction: Host 🡪 Vega

Purpose: Select a record from a loaded database by index to print forever

Response: [DYNAMIC\_DATA\_PRINTED\_IM1\_REPLY](#_DYNAMIC_DATA_PRINTED_IM1_REPLY)

Notes:

Data: unsigned long int uliIndexNumber;

Description:

uliIndexNumber- The 1-based Index number of the print data record to be printed.

Example: record index 2

<Sent> - SEND\_STATIC\_DATA\_IM1\_REMOTE ............. [02 08 00 00 00 00 00 A3 02 00 00 00 03 ]

### DYNAMIC\_DATA\_PRINTED\_TM1\_REPLY 0x7e

ID: 0x7e Direction: Vega 🡪 Host

Purpose: Let the host know that a page was printed

Requester: [SEND\_DYNAMIC\_DATA\_TM1\_REMOTE](#_SEND_DYNAMIC_DATA_TM1_REMOTE), or [SEND\_STATIC\_DATA\_TM1\_REMOTE](#_SEND_STATIC_DATA_TM1_REMOTE_1)

Notes: This message is sent for every page printed while in TM1

Even for pages that have no dynamic, static or [User Variable data](#_USER_DEFINED_VARIABLE_DATA_REPLY)

Data: unsigned long int uliRecordNumber;

Description:

uliRecordNumber - The Record Number of the Dynamic Print Data or Static Print Data that was printed.

Example: index number 5 returned

<Received> - DYNAMIC\_DATA\_PRINTED\_IM1\_REPLY ............. [02 08 00 00 00 00 1F 82 05 00 00 00 03 ]

uliIndexNumber: 5

### SEND\_STATIC\_DATA\_TM1\_REMOTE 0xA2

ID: 0xA2 Direction: Host 🡪 Vega

Purpose: A Static Print Data record to be printed, using Terminal Mode 1.

Notes: This record only needs to be sent once.

The host can resend this message (with different data) at any time

This message can be combined with [SEND\_DYNAMIC\_DATA\_TM1\_REMOTE](#_SEND_DYNAMIC_DATA_TM1_REMOTE)

Data:

typedef struct {  
 unsigned long int uliRecordNumber;  
 unsigned long int uliNumTextChars;  
 szDynamicPrintData  
} SSendDynamicDataTM1Remote;

Description:

uliRecordNumber - The Record Number associated with the Dynamic Print Data contained in the message.

uliNumTextChars - The number of Text characters, contained in this message, to print.

szDynamicPrintData - The Dynamic Print Data to be printed for this record in UTF-8 format

Example:

<Sent> - SEND\_STATIC\_DATA\_TM1\_REMOTE ................Record#2-Field#1.,Record#2-Field#2.,Record#2-Field#3.,Record#2-Field#4.,Record#2-Field#5.,Record#2-Field#6.,Record#2-Field#7.,Record#2-Field#8.... [02 9D 00 00 00 00 00 A2 02 00 00 00 91 00 00 00 52 65 63 6F 72 64 23 32 2D 46 69 65 6C 64 23 31 20 2C 52 65 63 6F 72 64 23 32 2D 46 69 65 6C 64 23 32 20 2C 52 65 63 6F 72 64 23 32 2D 46 69 65 6C 64 23 33 20 2C 52 65 63 6F 72 64 23 32 2D 46 69 65 6C 64 23 34 20 2C 52 65 63 6F 72 64 23 32 2D 46 69 65 6C 64 23 35 20 2C 52 65 63 6F 72 64 23 32 2D 46 69 65 6C 64 23 36 20 2C 52 65 63 6F 72 64 23 32 2D 46 69 65 6C 64 23 37 20 2C 52 65 63 6F 72 64 23 32 2D 46 69 65 6C 64 23 38 20 0D 0A 03 ]

### DATA\_BUFFER\_AVAILABLE 0x25

ID: 0x25 Direction: Vega 🡪 Host

Purpose: Tell Host that there is space in Vega/ Printers for another Print Data Record

Notes: See [Start Printing Sequence](#_Start_Printing_Sequence)

Description: Before the [ENABLE\_PRINTING](#_ENABLE_PRINTING) command is sent the host can prefill buffers

Example:

<Received> - DATA\_BUFFER\_AVAILABLE ......-%. [02 04 00 00 00 00 2D 25 03 ]

### WPD\_PAGE\_PRINTED\_REPLY 028

ID: 0x28 Direction: Vega 🡪 Host

Purpose: To let the host know that Vega printed a page that was sent from the Windows Printer Driver

Notes: Pages are printed without the host sending data

Data: unsigned long int uliPageNumber;

Description:

uliPageNumber The page number that started at 1 when the WPD started the job

Example:

### SEND\_DYNAMIC\_DATA\_IM1\_REMOTE 0x81

ID: 0x81 Direction: Host 🡪 Vega

Purpose: Select a record from a loaded database by index to print

Response: [DYNAMIC\_DATA\_PRINTED\_IM1\_REPLY](#_DYNAMIC_DATA_PRINTED_IM1_REPLY)

Notes:

Data: unsigned long int uliIndexNumber;

Description:

uliIndexNumber- The 1-based Index number of the print data record to be printed.

Example: record index 2

<Sent> - SEND\_DYNAMIC\_DATA\_IM1\_REMOTE ............. [02 08 00 00 00 00 00 81 02 00 00 00 03 ]

### DYNAMIC\_DATA\_PRINTED\_IM1\_REPLY 0x82

ID: 0x82 Direction: Vega 🡪 Host

Purpose: Let the host know that a page was printed with the corresponding index sent

Notes: This message is used in Index Mode 1.

Requester: [SEND\_DYNAMIC\_DATA\_IM1\_REMOTE](#_SEND_DYNAMIC_DATA_IM1_REMOTE_1)

Data: unsigned long int uliIndexNumber;

Description:

uliIndexNumber - The 1-based Index Number of the record that was printed.

Example:

### TM\_PRINT\_PURGE\_PAGE 0x9A

ID: 0x9A Direction: Host 🡪 Vega

Purpose: To insert a purge page into the data stream

Notes: The bitmap for the purge page is in the Vega Binary Directory: “Purge.bmp”

This bitmap can be changed but the name must remain the same

Response: [TM\_PURGE\_PAGE\_REPLY](#_TM_PURGE_PAGE_REPLY)

Data: None

Description: None

Example:

<Sent> - TM\_PRINT\_PURGE\_PAGE ......... [02 04 00 00 00 00 00 9A 03 ]

### TM\_PURGE\_PAGE\_REPLY 0x9B

ID: 0x9B Direction: Vega 🡪 Host

Purpose: Replies with either success or failure on printing the purge page.

Requester: [TM\_PRINT\_PURGE\_PAGE](#_TM_PRINT_PURGE_PAGE_1)

Data: unsigned char purgeStatus;

Description:

purgeStatus - 0 for success and 0xfa for failure.

Example:

<Received> - TM\_PURGE\_PAGE\_REPLY ......d... [02 05 00 00 00 00 64 9B 00 03 ]

PURGE -- SUCCESSFUL

### REQUEST\_USER\_DEFINED\_VARIABLE\_DATA 0x15

ID: 0x15 Direction: Vega 🡪 Host

Purpose: Request User Defined Variable data from host

Notes:

Response: [USER\_DEFINED\_VARIABLE\_DATA\_REPLY](#_USER_DEFINED_VARIABLE_DATA_REPLY)

Data:

DataType=?,VarName=?,DefaultValue=?,MinValue=?,MaxValue=?,OperatorPrompt=?FF

DataType=?,VarName=?,DefaultValue=?,MinValue=?,MaxValue=?,OperatorPrompt=?FF

DataType=?,VarName=?,DefaultValue=?,MinValue=?,MaxValue=?,OperatorPrompt=?FF

...

All data is delimited strings (= , FF) and converted by the Host application into defined data type.

DataType: integer, real, string, hex

If there are NO user variables the command will send a CR.

Example:

<Received> - REQUEST\_USER\_DEFINED\_VARIABLE\_DATA .u..../.DataType=String,VarName=test,DefaultValue=ssssssssssssssss,MinValue=0,MaxValue=54,OperatorPrompt=enter.some.text.. [02 75 00 00 00 00 2F 15 44 61 74 61 54 79 70 65 3D 53 74 72 69 6E 67 2C 56 61 72 4E 61 6D 65 3D 74 65 73 74 2C 44 65 66 61 75 6C 74 56 61 6C 75 65 3D 73 73 73 73 73 73 73 73 73 73 73 73 73 73 73 73 2C 4D 69 6E 56 61 6C 75 65 3D 30 2C 4D 61 78 56 61 6C 75 65 3D 35 34 2C 4F 70 65 72 61 74 6F 72 50 72 6F 6D 70 74 3D 65 6E 74 65 72 20 73 6F 6D 65 20 74 65 78 74 0C 03 ]

### USER\_DEFINED\_VARIABLE\_DATA\_REPLY 0x16

ID: 0x16 Direction: Host 🡪 Vega

Purpose: Response supplying the user defined variable data requested.

Notes: To return the user data variables to Vega in the format that they were requested.

If a job does not require User Variables the request is sent with no data and a single CR

Requestor: [REQUEST\_USER\_DEFINED\_VARIABLE\_DATA](#_REQUEST_USER_DEFINED_VARIABLE_DATA)

Data: All data is delimited strings (= , FF).

Description: Variable name – Value pairs

VarName=?,Value=?FF

VarName=?,Value=?FF

VarName=?,Value=?FF

…

VarName is used for ease of debugging. Vega will throw an error if it does not get a valid name.

Value is the data from the host (as if it were entered by the operator).

Example:

<Sent> - USER\_DEFINED\_VARIABLE\_DATA\_REPLY

........VarName=test,Value=abc123..

[02 1E 00 00 00 00 00 16 56 61 72 4E 61 6D 65 3D 74 65 73 74 2C 56 61 6C 75 65 3D 61 62 63 31 32 33 0C 03 ]

## Hardware IO Group

### SET\_IO\_OUTPUT\_RELAY\_REMOTE 0x8B

ID: 0x8b Direction: Host 🡪 Vega

Purpose: Turns on or off a printer’s output relay

Notes:

Data:

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned char ucIOOutputRelayNum; // 1-based number  
 unsigned char ucIOOutputRelayState;   
} SSetIOOutputRelayRemote;

Description:

ucImagerNum - (1 – 32) = Send message to the specified Imager Number

ucIOOutputRelayNum - The 1-based Output Relay number.

ucIOOutputRelayState - 0 = Disabled, 1 = Enabled.

Example: Turn on the second relay on printer #1

<Sent> - SET\_IO\_OUTPUT\_RELAY\_REMOTE ............ [02 07 00 00 00 00 00 8B 01 01 01 03 ]

### GET\_IO\_INPUT\_STATE\_REMOTE 0x8C

ID: 0x8c Direction: Host 🡪 Vega

Purpose: Find out if an input is high or low

Response: [IO\_INPUT\_STATE\_REPLY](#_IO_INPUT_STATE_REPLY)

Notes: This is only for inputs (output relay states are not reported)

There will be a list of all the inputs returned

Data:

unsigned char ucImagerNum;

Description:

ucImagerNum- (1 – 32) = Send message to the specified Imager Number

Example: Get the input states for the first printer

<Sent> - GET\_IO\_INPUT\_STATE\_REMOTE .......... [02 05 00 00 00 00 00 8C 01 03 ]

### IO\_INPUT\_STATE\_REPLY 0x8d

ID: 0x8d Direction: Vega 🡪 Host

Purpose: This message returns the I/O Input Line state of the requested Input Line number.

Requester: [GET\_IO\_INPUT\_STATE\_REMOTE](#_GET_IO_INPUT_STATE_REMOTE)

Notes: The input state reported is after it has been debounced

Data:

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned long int uliIOInputLinesStates;  
}SIOInputStateReply;

Description:

ucImagerNum - (1 – 32) = Imager number that the data corresponds to.

ucIOInputLineStates- The State of the Input Lines, 0 = Low, 1 = High.

Bits 31..12 Reserved Bits  
 Bit 11 Input 5 Debounced  
 Bit 10 Input 4 Debounced  
 Bit 9 Input 3 Debounced  
 Bit 8 Input 2 Debounced  
 Bit 7 Input 1 Debounced  
 Bit 6 Input 0 Debounced  
 Bit 5 Input 5   
 Bit 4 Input 4   
 Bit 3 Input 3   
 Bit 2 Input 2   
 Bit 1 Input 1   
 Bit 0 Input 0

Example:

<Received> - IO\_INPUT\_STATE\_REPLY .............. [02 09 00 00 00 00 0F 8D 01 F7 0D 00 00 03 ]

Printer: 1 Inp 0: high Inp 1: high Inp 2: high Inp 3: low Inp 4: high Inp 5: high Inp 0 debounced: high Inp 1 debounced: high Inp 2 debounced: high Inp 3 debounced: low Inp 4 debounced: high Inp 5 debounced: high

### SET\_IO\_OUTPUT\_WHEN\_READY\_REMOTE 0x8E

ID: 0x8e Direction: Host 🡪 Vega

Purpose: Use an output relay to indicate if Vega is in print mode

Notes: This can also be done manually in the Vega I/O configuration dialog

Data:

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned char ucIOOutputRelayNum; // 1-based number  
 unsigned char ucIOOutputRelaySignaledState;   
} SSetIOOutputWhenSignaledRemote;

Description:

ucImagerNum - (1 – 32) = Send message to the specified Imager Number

ucIOOutputRelayNum - The 0-based Output Relay number.

ucIOOutputRelaySignaledState -  
 0 = Disable Feature.  
 1 = When “Ready to Print” Enable Relay, else Disable Relay.  
 2 = When “Ready to Print” Disable Relay, else Enable Relay.

Example: when ready to print, turn on the second relay on the first printer

<Sent> - SET\_IO\_OUTPUT\_WHEN\_READY\_REMOTE ............ [02 07 00 00 00 00 00 8E 01 01 01 03 ]

### SET\_IO\_OUTPUT\_WHEN\_ERROR\_REMOTE 0x8F

ID: 0x8f Direction: Host 🡪 Vega

Purpose: This message sets the specified I/O Output Relay to the specified state, when an Error has occurred in the Imaging System.

Notes: This can also be done manually in the Vega I/O configuration dialog

Data:

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned char ucIOOutputRelayNum; // 0-based number  
 unsigned char ucIOOutputRelaySignaledState;   
} SSetIOOutputWhenSignaledRemote;

Description:

ucImagerNum - (1 – 32) = Send message to the specified Imager Number

ucIOOutputRelayNum - The 1-based Output Relay number.

ucIOOutputRelaySignaledState -  
 0 = Disable Feature.  
 1 = When an Error has occurred Enable Relay, else Disable Relay.  
 2 = When an Error has occurred Disable Relay, else Enable Relay.

Example: Disable (remove from Vega) for imager #1 for the second relay

<Sent> - SET\_IO\_OUTPUT\_WHEN\_ERROR\_REMOTE ............ [02 07 00 00 00 00 00 8F 01 01 00 03 ]

### SET\_IO\_OUTPUT\_WHEN\_INKLOW\_REMOTE 0x90

ID: 0x90 Direction: Host application 🡪 Vega application

Purpose: This message sets the specified I/O Output Relay to the specified state, when a Pen in the Imaging System is in an “Ink Low” state.

Notes: This can also be done manually in the Vega I/O configuration dialog

Data:

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned char ucIOOutputRelayNum; // 0-based number  
 unsigned char ucIOOutputRelaySignaledState;   
}SSetIOOutputWhenSignaledRemote;

Description:

ucImagerNum - 1 – 32 = Send message to the specified Imager Number

ucIOOutputRelayNum - The 0-based Output Relay number.

ucIOOutputRelaySignaledState -  
 0 = Disable Feature.  
 1 = When “Ink Low” Enable Relay, else Disable Relay.  
 2 = When “Ink Low” Disable Relay, else Enable Relay.

Example: for imager #1, Output #0, and turn on when triggered.

<Sent> - SET\_IO\_OUTPUT\_WHEN\_INK\_LOW\_REMOTE ............ [02 07 00 00 00 00 00 90 01 00 01 03 ]

## Ink Messages Group

### TM\_LOW\_INK 0x9C

ID: 0x9C Direction: Vega 🡪 Host

Purpose: Message to host when Ink Low event is trigger, or active

Notes: There is also an output relay [SET\_IO\_OUTPUT\_WHEN\_INKLOW\_REMOTE](#_SET_IO_OUTPUT_WHEN_INKLOW_REMOTE)

Sent when triggered, when the job is started, and when Vega is started

“ink low” can be triggered by a

Data:

unsigned char imagerNumber;  
 unsigned char penNumber;

Description:

imagerNumber - the number of the imager that is low on ink. This number is OR’d with 0x80.

penNumber - the number of the pen that is low on ink. This number is OR’d with 0x80.

Note:

The OR’ing with 0x80 is to set the high bit in each byte.

Example: low ink on pen B for the first imager

<Received> - TM\_LOW\_INK ......f.... [02 06 00 00 00 00 66 9C 81 82 03 ]

Received Low Ink Message: Imager #1, Pen #2

### TM\_EMPTY\_INK\_ALARM 0x14

ID: 0x14 Direction: Vega 🡪 Host

Purpose: Informs the Host application that an imager and pen have reached the empty threshold.

Notes: THE EMPTY THRESHOLD IS ALWAYS ZERO (0) %; THERE IS NO ADJUSTMENT.

Data:

unsigned char imagerNumber;  
 unsigned char penNumber;

Description:

imagerNumber - the number of the imager that is out of ink. This number is OR’d with 0x80.

penNumber - the number of the pen that is low on ink. This number is OR’d with 0x80.

Note:

The OR’ing with 0x80 is to set the high bit in each byte.

This is applied to ALL terminal Modes.

Example:

<Received> - TM\_EMPTY\_INK ......d.... [02 06 00 00 00 00 64 14 81 82 03 ]

Imager Number: 1 Empty Pen: 2

### GET\_INK\_LEVELS\_REMOTE 0x89

ID: 0x89 Direction: Host 🡪 Vega

Purpose: This message requests the Ink Levels of the Pens/ cartridges in the Imaging system.

Note: This will request ONLY the level of ink remaining

Response: [INK\_LEVELS\_REPLY](#_INK_LEVELS_REPLY)

Notes: use [GET\_INK\_DATA](#_GET_INK_DATA_1) because it supports Smart Ink Card

Data: None

Example: Request ink levels

<Sent> - GET\_INK\_LEVELS\_REMOTE ......... [02 04 00 00 00 00 00 89 03 ]

### INK\_LEVELS\_REPLY 0x8A

ID: 0x8a Direction: Vega 🡪 Host

Purpose: For host to update active

Notes: To receive additional ink data use the [GET\_INK\_DATA](#_GET_INK_DATA_1) command

Requester: [GET\_INK\_LEVELS\_REMOTE](#_GET_NUM_LEFT_TO_PRINT_REMOTE)

Notes:

Data:

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned char ucTypeOfPens;  
 unsigned char ucPercentageInkLeftPen1;  
 unsigned char ucPercentageInkLeftPen2;  
 unsigned char ucPercentageInkLeftPen3;  
 unsigned char ucPercentageInkLeftPen4;  
} SInkLevelsReply;

Message Data Description:

ucImagerNum - (1 – 32) = Imager number that the data corresponds to.

ucTypeOfPens-

0 = Reservoir System  
 1 = Cartridge

ucPercentageInkLeftPen(1-4) - Percentage of Ink left in the Pen, rounded to the nearest integer.

Example: Response for printer # 1

<Received> - INK\_LEVELS\_REPLY ......I........ [02 0A 00 00 00 00 49 8A 01 01 00 00 00 00 03 ]

ucImagerNum: 0, ucTypeOfPens: 1, ucPercentageInkLeftPen1: 0, ucPercentageInkLeftPen2: 0, ucPercentageInkLeftPen3: 0, ucPercentageInkLeftPen4: 0

### GET\_INK\_DATA 0xB2

ID: 0xB2 Direction: Host 🡪 Vega

Purpose: Request information about the currently installed ink cartridges.

Notes: Use to get all information, including Smart Card, about ink cartridges/ printheads

Data: none

Description: Vega will respond to the host with the INK\_DATA\_REPLY command for all the printers.

Example: <Sent> - GET\_INK\_DATA ......... [02 04 00 00 00 00 00 B2 03 ]

### INK\_DATA\_REPLY 0xB3

ID: 0xB3 Direction: Vega 🡪 Host

Purpose: Information about the currently installed ink cartridges.

Note: Use “printerType “and printerName” to identify print cartridge group for specific printer

The information is different for HP, iM2, and PC22.

Data: char [] // length varies based on variable length strings.

Message Data Description:

The complete message is a UTF-8 string terminated with a Null character.

Each line of information is terminated with a 0x0D (carriage return) character

Each block of information for a printer is terminated with a 0x0C (Form Feed) character

Each variable contains this structure:

The Variable Name

an equal’s character “=”

UTF-8 data string (representing text or a numerical value)

a comma “,” between variables

All printer types have the following header line constructed of:

printerType=?printerName=? 0x0D

Where:

printerType “Core”,”Print Controller”,”Satellite Controller”,”V7 Print Controller”,”Wing”,”ISP”,”Uno”,”Uno Plus”,”IM2”,”PC22”

printerName string

For HP based Printers: (printerType: “Core”,”Print Controller”,”Satellite Controller”,”V7 Print Controller”,”Wing”,”ISP”,”Uno”,”Uno Plus”)

This message is for both smart chip (SC) and non-smart chip printers; however, there will be more information in the smart chip message. The ASCII string returned will be in the following format and is variable in length because of the Name field. To parse the information the string is comma separated with 10 fields for each pen and the first field is always the text “PenID=?” where ‘?’ being the capital letter A, B, C, or D.

Note: all HP printers will supply data for 4 pens (even for one pen V7)

Note: if a pen is disabled the data will be zeros (0)

Data structure for a pen: (“?” represents the data as a character string)

PenID=?. SC=?,GoodNozzles=?,inkID=?,name=?,InkVol=?,bulk=?,InkUsed=?,oemId=?,outOfInk=? 0x0D

|  |  |
| --- | --- |
| PenID | This is letter designation for the pen A, B, C, or D |
| SC | When the Smart Card is present this will be set to the character '1' and '0' otherwise. |
| goodNozzles | This is the result from the HP nozzle resistor test. |
| inkID | This ID will be read from the SC if available; otherwise, it will be read from the contacts. |
| name | This ID will be read from the SC if available; otherwise, it will be read from an inktypes table (if loaded in printer FLASH memory) |
| InkVol | This ID will be read from the SC if available; otherwise, it will be read from printer memory with one value for each of the 4 pen stalls A, B, C and D (position based). If a pen is moved between stalls it will only be valid if using the SC. Note: this value is updated only when the Stop Print command is sent; therefor, always properly stop printing (and do not turn the power off while printing.) |
| bulk | This ID will be read from the SC if available; otherwise, it will be read from the contacts. |
| InkUsed | This value is used with Bulk Pens only and is useful to determine how much use a bulk pen has seen when it starts to fail. It should only be relied upon when using a SC system. |
| oemId | Custom ID assigned to an OEM for their application to verify that the ink is from their distribution channel |
| outOfInk | This is set when the OEM wants to disable a cartridge because the "oemId" is not correct; or, when a non-bulk pen has printed 2.1 billion drops (to prevent refilling). |

Example:

<Received> - INK\_DATA\_REPLY ......G.PenID=A,SC=0,GoodNozzles=0,inkID=0,name=Unmanaged,InkVol=42000,bulk=0,InkUsed=0,oemId=0,outOfInk=0.PenID=B,SC=0,GoodNozzles=0,inkID=0,name=Unmanaged,InkVol=40000,bulk=0,InkUsed=0,oemId=0,outOfInk=0.PenID=C,SC=0,GoodNozzles=0,inkID=0,name=Unmanaged,InkVol=42000,bulk=0,InkUsed=0,oemId=0,outOfInk=0.PenID=D,SC=0,GoodNozzles=0,inkID=0,name=Unmanaged,InkVol=40000,bulk=0,InkUsed=0,oemId=0,outOfInk=0.. [02 90 01 00 00 00 47 B3 50 65 6E 49 44 3D 41 2C 53 43 3D 30 2C 47 6F 6F 64 4E 6F 7A 7A 6C 65 73 3D 30 2C 69 6E 6B 49 44 3D 30 2C 6E 61 6D 65 3D 55 6E 6D 61 6E 61 67 65 64 2C 49 6E 6B 56 6F 6C 3D 34 32 30 30 30 2C 62 75 6C 6B 3D 30 2C 49 6E 6B 55 73 65 64 3D 30 2C 6F 65 6D 49 64 3D 30 2C 6F 75 74 4F 66 49 6E 6B 3D 30 0D 50 65 6E 49 44 3D 42 2C 53 43 3D 30 2C 47 6F 6F 64 4E 6F 7A 7A 6C 65 73 3D 30 2C 69 6E 6B 49 44 3D 30 2C 6E 61 6D 65 3D 55 6E 6D 61 6E 61 67 65 64 2C 49 6E 6B 56 6F 6C 3D 34 30 30 30 30 2C 62 75 6C 6B 3D 30 2C 49 6E 6B 55 73 65 64 3D 30 2C 6F 65 6D 49 64 3D 30 2C 6F 75 74 4F 66 49 6E 6B 3D 30 0D 50 65 6E 49 44 3D 43 2C 53 43 3D 30 2C 47 6F 6F 64 4E 6F 7A 7A 6C 65 73 3D 30 2C 69 6E 6B 49 44 3D 30 2C 6E 61 6D 65 3D 55 6E 6D 61 6E 61 67 65 64 2C 49 6E 6B 56 6F 6C 3D 34 32 30 30 30 2C 62 75 6C 6B 3D 30 2C 49 6E 6B 55 73 65 64 3D 30 2C 6F 65 6D 49 64 3D 30 2C 6F 75 74 4F 66 49 6E 6B 3D 30 0D 50 65 6E 49 44 3D 44 2C 53 43 3D 30 2C 47 6F 6F 64 4E 6F 7A 7A 6C 65 73 3D 30 2C 69 6E 6B 49 44 3D 30 2C 6E 61 6D 65 3D 55 6E 6D 61 6E 61 67 65 64 2C 49 6E 6B 56 6F 6C 3D 34 30 30 30 30 2C 62 75 6C 6B 3D 30 2C 49 6E 6B 55 73 65 64 3D 30 2C 6F 65 6D 49 64 3D 30 2C 6F 75 74 4F 66 49 6E 6B 3D 30 00 03 ]

Printer 1

PenID=A SC=0 GoodNozzles=0 inkID=0 name=Unmanaged InkVol=42000 bulk=0 InkUsed=0 oemId=0 outOfInk=0

PenID=B SC=0 GoodNozzles=0 inkID=0 name=Unmanaged InkVol=40000 bulk=0 InkUsed=0 oemId=0 outOfInk=0

PenID=C SC=0 GoodNozzles=0 inkID=0 name=Unmanaged InkVol=42000 bulk=0 InkUsed=0 oemId=0 outOfInk=0

PenID=D SC=0 GoodNozzles=0 inkID=0 name=Unmanaged InkVol=40000 bulk=0 InkUsed=0 oemId=0 outOfInk=0

For iM2 based Printers: (printerType:” IM2”,”IM4”)

Data structure for a printhead: (“?” represents the data as a character string)

target\_HeadTemp (C)=?,current\_HeadTemp (C)=?, target\_ManifoldTemp (C)=?,current\_ManifoldTemp (C)=?

Where:

|  |  |
| --- | --- |
| Variable Name | Meaning |
| target\_HeadTemperature | Desired Head run temperature (In Celsius Range 0-50) set in Vega |
| target\_ManifoldTemperature | Desired Manifold run temperature (In Celsius Range 0-50) set in Vega |
| current\_HeadTemperature | Current head temperature, degrees Celsius, as read from sensor |
| current\_ManifoldTemperature | Current manifold temperature, degrees Celsius,as read from sensor |
| Ink\_empty\_alarm | 1=active (out of ink); 0=inactive (still have ink left) |

Note: additional information about the ink used may be added in the future.

For PC22 based Printers (printerType:”PC22”,”UC22”)

Data structure for a pen: (“?” represents the data as a character string)

selectType=?,stateChk=?,ErrChk=?,inkUsed (.1mm)=?,inkVol (.1mm)=?,%inkUsed=?,InsertCnt=?,name=?,pn=?,oemId=? 0x0D

Where:

|  |  |  |
| --- | --- | --- |
| Variable Name | Data | Definition |
|  |  |  |
| selectType | “Primary”, “Secondary” | Ink is drawn from the Primary Cartridge |
| stateChk | “NOT PRESENT”, “NOT VERIFIED”, “SC\_VALID”, “SC\_INVALID “ | Status of bulk ink cartridge- is it OK or not |
| ErrChk | “NoError”, or 1 or more: “Wrong Ink Type”, “SC Error”, “Wrong Supply Type”,” Faulty”, “Expired”, “Altered”, Out of Ink,” | 7 possilbe error states and a NoError State.  Note: it is possible to have more than one error state active at once. |
| inkUsed (.1mm) | Integer string | Ink consumed in tenths of a milliliter |
| inkVol (.1mm) | Integer string | Usable ink volume in tenths of a milliliter |
| %inkUsed | Integer string 0-100 | Percentage of ink remaining. Range of 0 to 100 |
| insertCnt | Integer string | Insertion count of the respective cartridge |
| name | string | Null terminated string of the ink name |
| pn | string | Null terminated string of the Norwix ink part number |
| oemId | Integer string | OEM ID of the respective cartridge |

Example:

selectType= Primary,stateChk= SC\_VALID,ErrChk= NoError,inkUsed (.1mm)=100?,inkVol (.1mm)=8000,%inkUsed=2,InsertCnt=2,name=Example ink name,pn=123456789,oemId=5

### PEN\_SPIT 0x12

ID: 0x12 Direction: Host 🡪 Vega

Purpose: Request all pens on all printers spit ink to clear nozzles.

Notes: Pen spit command allows user to purge all installed pens upon request.

Host must wait for purge completion status message, PEN\_SPIT\_DONE , **before** issuing new commands.

This has the same as pressing the “Purge” dropdown menu on the printer configuration card.

Data: none.

Example:

<Sent> - PEN\_SPIT ......... [02 04 00 00 00 00 00 12 03 ]

### PEN\_SPIT\_DONE 0x13

ID: 0x13 Direction: Vega 🡪 Host

Purpose: Request all pens on all printers spit ink to clear nozzles.

Requester: [PEN\_SPIT](#_PEN_SPIT)

Notes: Host must wait for purge completion status message, PEN\_SPIT\_DONE , **before** issuing new commands.

Data: none

<Received> - PEN\_SPIT\_DONE ......F.. [02 04 00 00 00 00 46 13 03 ]

### iM2\_4\_IDS\_ENABLE 0xC7

ID: 0xC7 Direction: Host application 🡪 Vega application

Purpose: To enable or disable the IDS (ink delivery system) in All or individual iM2/ iM4 printers

Note: If the printer accessed is not an iM2 or iM4 the issue NACK “Not correct

Data: None for “ALL” or a 1 based index for a particular Alert.

unsigned char ucEnable; // 0 for off 1 for on

ucEnable 1 = Enable, 0 = Disable

Turn ON IDS:

Enable<Sent> - IM2\_4\_IDS\_ENABLE (V) ........... [02 06 00 00 00 00 00 B9 00 01 03 ]  
<Received> - REMOTE\_ACKNOWLEDGE\_MSG ........... [02 06 00 00 00 00 00 99 B9 00 03 ]

Turn OFF IDS:  
<Sent> - IM2\_4\_IDS\_ENABLE (V) ........... [02 06 00 00 00 00 00 B9 00 00 03 ]  
<Received> - REMOTE\_ACKNOWLEDGE\_MSG ........... [02 06 00 00 00 00 00 99 B9 00 03 ]

RemAppSim dialog:

A screenshot of a computer screen

Description automatically generated

## Errors and Warnings Group

### ERROR\_MESSAGE\_REPLY 0x94

ID: 0x94 Direction: Vega 🡪 Host

Purpose: Notify the host of an error as it occurs

Notes: see SET\_TERMINAL\_MODE\_CONFIG\_REMOTE to receive this message

The errors are sent as they are declared

There are errors for the printer(s) and higher level declared one’s by the Vega application

Data:

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned char ucErrorNumber;  
}SErrorMessageReply;

or if [SET\_TERMINAL\_MODE\_CONFIG\_REMOTE 0x71](#_SET_TERMINAL_MODE_CONFIG_REMOTE_0x7)🡪 ucEnableErrorMessages = 2 (Verbose mode)

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned char ucErrorNumber;

char\* Utf8DescriptiveString;  
}SErrorMessageReplyVerbose;

Message Data Description:

ucImagerNum -  
 0 = Error not imager related (Vega generated).  
 1 – 32 = Imager number that the source of the error.

ucErrorNumber -  
 1 = (list TBD)  
 2 = …  
 n =

Example: Data underrun Error

<Received> - ERROR\_MESSAGE\_REPLY ......B.... [02 06 00 00 00 00 42 94 01 01 03 ]

Imager Number: 1 Error Code: 1

Graphical user interface, application

Description automatically generated

Example: for Verbose mode where Utf8DescriptiveString

“Error,Printer:pc,Title:Connection Error, There is an error with TCP/IP connection, and the connection has been closed. Any current job in progress has been stopped”

Text

Description automatically generated with medium confidence

### WARNING\_MESSAGE\_REPLY 0xB6

ID: 0xB6 Direction: Vega 🡪 Host

Purpose: Notify the host of a warning as it occurs

Notes: see [SET\_TERMINAL\_MODE\_CONFIG\_REMOTE](#_SET_TERMINAL_MODE_CONFIG_REMOTE_1) to receive this message

The warnings are sent as they are declared

There are warning for the printer(s) and higher level declared one’s by the Vega application

Data:

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned char ucWarningNumber;  
} SErrorMessageReply;

or if [SET\_TERMINAL\_MODE\_CONFIG\_REMOTE 0x71](#_SET_TERMINAL_MODE_CONFIG_REMOTE_0x7)🡪 ucEnableErrorMessages = 2 (Verbose mode)

typedef struct {  
 unsigned char ucImagerNum;  
 unsigned char ucWarningNumber;

char\* Utf8DescriptiveString;  
}SErrorMessageReplyVerbose;

Utf8DescriptiveString-

4 comma separated strings

Type, Source, Title, Description

Message Data Description:

ucImagerNum -  
 0 = Warning not imager related (Vega generated).  
 1 – 32 = Imager number that the data corresponds to.

ucWarningNumber -  
 1 = (list TBD)  
 2 = …  
 n =

Example: the first imager is issuing a printer error #4 (Pen Power-Up Failed)

<Received> - WARNING\_MESSAGE\_REPLY ......&.... [02 06 00 00 00 00 26 B6 01 04 03 ]

Imager Number: 1 Warning Code: 4

Text

Description automatically generated

Example: for Verbose mode where Utf8DescriptiveString

“Warning,Printer:pc,Title:Pen failed to power up,Warning Pen A power-up failed”

Text

Description automatically generated

Example: of Verbose mode

<Received> - WARNING\_MESSAGE\_REPLY .&........Warning,.Pen.C.power-up.failed...

[02 26 00 00 00 00 0B B6 01 04 57 61 72 6E 69 6E 67 2C 20 50 65 6E 20 43 20 70 6F 77 65 72 2D 75 70 20 66 61 69 6C 65 64 2E 0A 03 ]

Imager Number: 1 Warning Code: 4 Description Warning, Pen A power-up failed.

### CLEAR\_ERROR\_AND\_WARNING\_MESSAGES 0xB7

ID: 0XB7 Direction: Host application 🡪 Vega application

Purpose: Clear out error and warning messages from Vega.

Data: None

Description:

If errors are warnings were detected there will be a yellow alert triangle in the upper right hand corner of the screen. Vega can automatically clear some (but not all) error or warning messages. The user can clear the messages using the Alert dialog while this command lets the TM Host do the same.

A screenshot of a computer

Description automatically generated with low confidenceGraphical user interface, application

Description automatically generated

Note:

Vega will attempt to clear ALL the error and warning messages. However, some conditions may still be present and if so, the alert will be reissued. This will then send the a WARNING\_MESSAGE\_REPLY, an ERROR\_MESSAGE\_REPLY, or both.

Example:

<Sent> - CLEAR\_ERROR\_AND\_WARNING\_MESSAGES ......... [02 04 00 00 00 00 00 B7 03 ]

<Received> - REMOTE\_ACKNOWLEDGE\_MSG

ucMessageID: 183, ucTransmitSequenceNum: 0

## Buffer Management

### CLEAR\_IMAGER\_PRINT\_QUEUES 0x98

ID: 0x98 Direction: Host 🡪 Vega

Purpose: Remove current and queued page buffers in the printers and in Vega.

Note: Any data records not Ripped in Vega will also be removed.

Hardware sensor triggers queued (for products that have not reached the printer yet) are also purged

See [GET\_NUM\_LEFT\_TO\_PRINT\_REMOTE](#_GET_NUM_LEFT_TO_PRINT_REMOTE_1) to investigate anytime to check on # of available buffers to print

Data: None

Example:

<Sent> - CLEAR\_IMAGER\_PRINTING\_REMOTE ......... [02 04 00 00 00 00 00 98 03 ]

<Received> - REMOTE\_ACKNOWLEDGE\_MSG

ucMessageID: 152, ucTransmitSequenceNum: 0

### DATA\_BUFFER\_AVAILABLE 0x25

ID: 0x25 Dir: Vega 🡪 Host

Purpose: tell host that there are buffers available to prefill

Note: When the host sends data, it is Ripped *if* there is a free buffer to place the image.

The total number of buffers = printer buffers + Vega buffers.

Additional data records, received by Vega that cannot be RIPped are queued – waiting for the next available buffer to free up.

Data: None

Description: In Vega under the Application Settings dialog there are the following places to set the desired number of buffers. Printer buffers defaults to 4 and GUI (Vega) buffers defaults to 10.

A picture containing square

Description automatically generated

Vega will send the command once for each available buffer.

If the Host wishes to Pre-Load buffers before sending the enable print command, they can count the “buffer available” requests and then send that many data records.

See: [Start Printing Sequence Workflows](#_Start_Printing_Sequence)

Example:

<Received> - DATA\_BUFFER\_AVAILABLE ......-%. [02 04 00 00 00 00 2D 25 03 ]

### GET\_NUM\_LEFT\_TO\_PRINT\_REMOTE 0x85

ID: 0x85 Direction: Host 🡪 Vega

Purpose: To find out how may pages are buffered in Vega + Printer buffers.

Response: [NUM\_LEFT\_TO\_PRINT\_REPLY](#_NUM_LEFT_TO_PRINT_REPLY)

Notes: Host can check that all buffers have been cleared after [CLEAR\_IMAGER\_PRINT\_QUEUES](#_CLEAR_IMAGER_PRINT_QUEUES)

Data: None

Example:

<Sent> - GET\_NUM\_LEFT\_TO\_PRINT\_REMOTE ......... [02 04 00 00 00 00 00 85 03 ]

### NUM\_LEFT\_TO\_PRINT\_REPLY 0x86

ID: 0x86 Direction: Vega 🡪 Host

Purpose: Return the number of print data records received that have not yet printed yet.

Requester: [GET\_NUM\_LEFT\_TO\_PRINT\_REMOTE](#_GET_NUM_LEFT_TO_PRINT_REMOTE_1)

Notes: This is printer buffers + Vega Ripped Buffers+ Vega Queued Buffers

Data: unsigned char uliNumLeftToPrint;

Description:

uliNumLeftToPrint - The number of print data records that have been sent to the Vega application that have not yet printed.

Example: three buffers left to print

<Received> - NUM\_LEFT\_TO\_PRINT\_REPLY ......I... [02 05 00 00 00 00 49 86 03 03 ]

uliNumLeftToPrint: 3

# Simple Message Type Interface Specifications

The TM3 and RPM1 modes use a very simple format for communicating between the host and Vega. These modes are more suited for interfacing Vega to Programmable Logic Controller (PLC) or other simple non-PC devices like barcode scanners where the programming power of a PC is not available, or not desired for the remote host design.

## List of Message IDs for (TM3, RP1)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Message** | **ID hex** | **Dir** | **TM3** | **RP1** | **Reply** | **Notes (TM3 does not use STX / ETX)** |
| **Job** | | | | | | |
| [Load\_Job](#_Load_Job_(0xF1)) | **F1** | **>** | X | X | FF | Load job to print |
| [Start\_Job](#_Start_Job_(0xF2)) | **FD** | **>** | X | X | FF | Start loaded job printing |
| [Stop\_Job](#_Stop_Job_(0xF3)) | **FE** | **>** | X | X | FF | Stop printing job |
| [Enable Printing](#_Enable_Printing_(0xF3)) | **F2** | **>** | X | X |  | Enable sensor to start printing |
| [Ready to Print](#_Ready_to_Print_1) | **FD** | **<** | X | X |  | Ready for data buffer prefill |
| **Data** | | | | | | |
| [Send Dynamic Data RP1 Remote](#_Send_Dynamic_Data) | n/a | **>** |  | X | F7 or F3 | [STX]UTF-8 Data[ETX] |
| [Send Dynamic Data TM3 Remote](#_Send_Dynamic_Data_1) | n/a | **>** | X |  | F7 or F3 | [PreA]UTF-8 Data[PstA1][PstA2] |
| [Send Static RP1 Remote](#_Send_Static_Data) | n/a | **>** | X |  |  |  |
| [Send Static Data TM3 Remote](#_Send_Static_Data) | n/a | **>** | X |  | F7 or F3 | [PreA]***[1A]***UTF-8 Data[PstA1][PstA2] |
| [Page Printed](#_Page_Printed_(0xF7)_2) | **F7** | **<** | X | X |  | [STX][F7]UTF-8 Data[ETX] |
| [Data Buffer Available](#_Buffer_Available_(0x25)) | **A5** | **<** | X | X |  |  |
| **Error/ Warning** | | | | | | |
| [Toggle Error Details](#_Toggle_Error_Details_(0xF8)) | **F8** | **>** | X | X | FF | Toggle Level of Error msg details |
| [Error Message](#_Simple_Error_Message) | **F3** | **<** | X | X |  | Error message |
| [Warning\_Message](#_Simple_Warning_Message) | **F4** | **<** | X | X |  | Warning message |
| [Low Ink](#_Low_Ink_Warning)\_Warning\_Message | **FB** | **<** | X | X |  | Low ink |
| **Status** | | | | | | |
| Get [System](#_Get_System_Status_1) Status | **F6** | **>** | X | X | F5 or F6 | Queries state of Start Print button |
| [System Not Ready](#_Insert_Purge_Page_(0xF7)) | **F5** | **<** | X | X |  | Start Print button = Green or Error |
| [System Ready](#_System_Ready_(0xF6)) | **F6** | **<** | X | X |  | Start Print button = Red & No Error |
| [Command Accepted](#_Page_Printed_(0xF7)) | **FF** | < | X | X |  | [STX][FF][new value][ETX] |
| **Purge** | | | | | | |
| [Insert Purge Page](#_Insert_Purge_Page) | **F7** | **>** | X | X | F9 or FA | Insert a Purge page in middle of print job. Responses from each Printer are given. |
| [Purge Page Success](#_Purge_Page_Success) | **F9** | **<** | X | X |  |
| [Purge Page Error](#_Purge_Page_Error) | **FA** | **<** | X | X |  |
| **Print Queues** | | | | | | |
| [Clear Print Queues](#_Clear_Print_Queues_1) | **F9** | **>** | X | X | FC | Clears page data and queued sensor triggers |
| [Clear Print Queues Success](#_Clear_Print_Queues_2) | **FC** | **<** | X | X |  |

## Message Format

The simple message terminal modes, TM3 and RPM1 use the formatting described below.

### Terminal Mode 3 Message Wrapping

This mode uses user defined message wrapping which is configured in Vega.

[PreAmb]UTF-8 Data[PostAmb1][PostAmb2]

* [PreAmb] = Optional setting in VEGA - Pre-Amble (hex)
* Graphical user interface, application

  Description automatically generatedData = UTF-8 data
* [PostAmb1] = Required setting in VEGA - Post-Amble 1 (hex)
* [PostAmb2] = Optional setting in VEGA - Post-Amble 2 (hex)

Note: when passing **UTF-8 data** it is suggested that the Host use the (optional) Preamble.

4-byte UTF-8 characters will begin with “F” as the first nibble and this will be decoded as a message (not data).

Using a preamble will ensure that the UTF-8 characters are decoded correctly,

Table

Description automatically generated

### Read Print Mode 1 Message Wrapping

This mode uses STX and ETX characters are required for this mode.

[STX]UTF-8 Data[ETX]

* [STX] = Start of text character (hex 02)
* Data = UTF-8 data
* [ETX] = End of text character (hex 03)

## Print Job Messages

### Load Job (0xF1)

This message is sent by the host to load a specific job by name.

RPM1: [STX][F1]*job name*[ETX] TM3: [F1]*job name*

*job name* is the job name as it appears in the Vega job list. This is an UTF-8 string of characters that is case sensitive and does not include any filename extension. (Note: be sure to pay attention to the UPPER and lower case characters or the job name will not be found.)

### Start Job (0xFD)

This message is sent by the host to start printing for the currently loaded job.

RPM1: [STX][FD][ETX] TM3:[FD]

### Ready to Print (0xFD)

This message is sent by the host when the job was loaded without error and waiting for data.

RPM1: [STX][FD][ETX] TM3:[FD]

### Stop Job (0xF3)

This message is sent by the host to stop printing for the currently printing job.

RPM1: [STX][F3][ETX] TM3:[F3]

### Enable Printing (0xF2)

This message is sent by the host after desired buffers have been pre-filled so actual printing may begin.

RPM1: [STX][F2][ETX] TM3:[F2]

## Print Data Messages

Print Data messages does not require a command byte in front of the actual data.

### Send Dynamic Data RP1 Remote

In RPM1 the data string sent is used to lookup a record in the database loaded in Vega. The first character is “F7” folled by the data string in UTF-8 and must be framed by the start of text/end of text pair.

[STX]F7UTF-8 Data[ETX]

Note**:** If the data command is empty (or NULL) it will print a blank page. The Page Printed reply from Vega will be NULL as well (<STX>F7<ETX>). That is the “F7” without any following string. No warning will be sent, and the blank page will be printed to keep the buffers in sequence with the printed product stream.

A “F7” by itself, is also sent in the case where the lookup failed to find a match or there was any other RIP error that caused the print image to not be created; however, two additional warnings are sent (described below). Bottom line, if the host expected something to print and it receives a “F7” without any data this should indicate that there was a problem.

If the string data sent resulted in a database lookup of “no match” (or for any other reason the page could not be RIPped [ ex: invalid barcode data]) a blank page will be generated.

At the time of RIPping a warning will be generated and at the time of print an additional warning will be generated as shown in the images below (for page #7 is this example).

Text

Description automatically generated at time of RIP

<Received> <STX><ETX> - IMAGER\_WARNING (RP) .... [F4 80 80 07 ]  
     Source: Vega Message: 0x7

Text

Description automatically generated at time of Print.

<Received> <STX><ETX> - IMAGER\_WARNING (RP) .... [F4 80 80 08 ]  
     Source: Vega Message: 0x8

### Send Dynamic Data TM3 Remote

In TM3, the data string that is sent is used as the print record. Only a Postamble is required and is defaulted to Form Feed (0x0C). If multiple fields are sent in the data record, they must be delimited by a Carriage Return (0x0D). Optionally, a Preamble and second Postamble can be sent. See Vega’s system settings tab for configuring these values.

[PreA]UTF-8 Data[PstA1][PstA2]

### Send Static Data RP1 Remote

The command is the same as the Send Dynamic Data RP1 Remote with the addition of an added first character in the UTF-8 data string. This additional character is a hex 0x1A which is the Substitute non-printing control character and indicates to Vega that this data is to be repeated for each record to be printed.

When Vega receives a static data record it will continue to use this data without receiving any dynamic data from the host for the fields that use the static data.

[STX]UTF-8 Data[ETX]

### Send Static Data TM3 Remote

The command is the same as the Send Dynamic Data TM3 Remote with the addition of an added first character in the UTF-8 data string. This additional character is a hex 0x1A which is the Substitute non-printing control character and indicates to Vega that this data is to be repeated for each record to be printed.

When Vega receives a static data record it will continue to use this data without receiving any dynamic data from the host for the fields that use the static data.

[PreA][1A]UTF-8 Data[PstA1] [PstA2]

### Page Printed (0xF7)

Issued when a page is printed.

RPM1: [STX][F7] UTF-8 Data string host sent [ETX] TM3:[F7]

### Buffer Available (0xA5)

Issued when a buffer is loaded BEFORE the Enable Print message is sent

RPM1: [STX][A5][ETX] TM3:[F5]

## Error Reportin**g**

### Toggle Error Details (0xF8)

Vega starts in simple error reporting mode by default every time the program is launched. The detailed errorreporting mode tells Vega to send 3 bytes of data consisting of ID numbers to identify the source of the error either Vega or a printer, which printer (if not Vega), and the error/warning.

RPM1: [STX][F8][ETX] TM2:[F8]

The response to this command is [Command Accepted (0xFF)](#_Page_Printed_(0xF7)) message indicating the “toggled” state.

RPM1: [STX][FF][F8][80][ETX] TM3:[STX][FF][F8][80]

80 = Detailed Error and Warning Reporting Disabled (Default)

RPM1: [STX][FF][F8][81][ETX]TM3:[STX][FF][F8][81]

81 = Detailed Error and Warning Reporting Enabled

### Error Message (0xF3)

This message will be sent to the host for each error condition declared by a printer or Vega.

Simple Error Reporting: RPM1: [STX][F3][ETX] TM3:[F3]

Verbose Error Reporting: RPM1: [STX][F3][SrcID][DevID][MsgID][ETX] TM3:[F3][SrcID][DevID][MsgID]

* SrcID = Error or Warning Source ID (80 = VEGA, 81 = Printer)
* DevID = Device ID (VEGA = Unused (80), Printer = Printer ID + 80 (1 Based)
* MsgID = Error or Warning Message ID (0x80 to 0xFF)

**Printer Errors:**

* 81 = (TBD)
* 82 =
* 83 =

**Vega Errors:**

* 81 = (TBD)
* 82 =
* 83 =

Example: verbose error

<Received> - ERROR\_MESSAGE\_REPLY .[....\*...The.printer.speed.(196.08.in/sec).has.exceeded.the.maximum.allowed.speed.(55.in/sec).. [02 5B 00 00 00 00 2A 94 01 00 54 68 65 20 70 72 69 6E 74 65 72 20 73 70 65 65 64 20 28 31 39 36 2E 30 38 20 69 6E 2F 73 65 63 29 20 68 61 73 20 65 78 63 65 65 64 65 64 20 74 68 65 20 6D 61 78 69 6D 75 6D 20 61 6C 6C 6F 77 65 64 20 73 70 65 65 64 20 28 35 35 20 69 6E 2F 73 65 63 29 2E 03 ]

Imager Number: 1 Error Code: 0 Description The printer speed (196.08 in/sec) has exceeded the maximum allowed speed (55 in/sec).

### Warning Message (0xF4)

This message will be sent to the host for each Warning condition declared by a printer or Vega.

Simple Warning Reporting: RPM1: [STX][F3][ETX] TM3:[F3]

Verbose Warning Reporting: RPM1: [STX][F3][SrcID][DevID][MsgID][ETX] TM3:[F3][SrcID][DevID][MsgID]

* SrcID = Error or Warning Source ID (80 = VEGA, 81 = Printer)
* DevID = Device ID (VEGA = Unused (80), Printer = Printer ID + 80 (1 Based)
* MsgID = Error or Warning Message ID (0x80 to 0xFF)

**Printer Warnings:**

* 81 = (TBD)
* 82 =
* 83 =

**Vega Warnings:**

* 81 = (TBD)
* 82 =
* 83 =

### Low Ink Warning Message (0xFB)

This message is sent when one or more ink cartridges entering the low ink threshold set in Vega (Pen Status > Set Pen Parameters > Cartridge System %). This is a 3-byte message not including wrapping. The message format is as follows:

* The **i** is the number of the printer reporting a pen low on ink (0 based).
* The **p** is the number of the pen that is reporting to be low on ink (0 based).

RPM1: [STX][FB][8i][8p][ETX] TM3:[FB][8i][8p]

Note: Both the printer number and the pen number values have the high order bit set to avoid possible confusion with the wrapping bytes.

## System Status Messages

The System Status messages are sent from VEGA to the Remote App/Host and Wrapped with STX and ETX.

### Get System Status (0xF6)

Use this command to query Vega if it is ready to receive dynamic data or index lookups (i.e. in print mode)

RPM1: [STX][F6][ETX] TM3:[F6]

### System Not Ready (0xF5)

Response to Get System Status (0xF6) indicating not in print mode because of error or printing was not started. Do not send dynamic data from your Host to Vega if this message is received by the host.

RPM1: [STX][F5][ETX] TM3:[F5]

### System Ready (0xF6)

Response to Get System Status (0xF6) indicating that the job is in print mode ready for data. Look for this message to be received by the host prior to sending dynamic data.

RPM1: [STX][F6][ETX] TM3:[F6]

### Command Accepted (0xFF)

Send to host the result of the received command action.

The message format: RPM1: [STX][FF][id][rc][ETX] TM3:[FF][id][rc]

* **id** = Message ID of command received
* **rc** = Result code

For the Error Detail Toggle message, the response indicates the “toggled” state:

* *Toggle Error Details (0xF8)* 
  + *id = F8*
  + *rc* 
    - *80 =* Detailed error and warning reporting is disabled (Default)
    - *81 =* Detailed error and warning reporting is enabled

The following commands the response will be an ACK/ NACK telling the host if the command was accepted (meaning that it was a valid request based on Vega’s current state).

* *Load Job (0xF1)*
* *Start Job (0xF2)*
* *Stop Job (0xF3)*
* *Insert Purge Page (0xF7)* (note: when purge page is printed 🡪Purge Page Success (0xF9))
* *Clear Print Queues (0xF9)* (note: when clear completed 🡪 Clear Print Queues Success (0xFC))
  + *id = F1, F2, F3, F7, or F9*
  + *rc*
    - *FF = ACK 🡪 Success*
    - *80 = NACK (0x80 + NACK Error Code) (note: see* *NACK (Negative Acknowledge) Definitions*

Example:

If the command, *Start Job (0xF2)* is sent, and the job is started without error then the response would be

RPM1: [STX][FF][F2][FF][ETX] TM3:[FF][F2][FF] (note: “FF” = Success!)

If the command, *Start Job (0xF2)* is sent, and the job is currently printing the response would be

RPM1: [STX][FF][F2][FE][ETX] TM3:[FF][F2][FE]

|  |  |  |  |
| --- | --- | --- | --- |
| START\_JOB | 0xFE | 1 | Currently printing |

## Other Messages

### Insert Purge Page (0xF7)

This is system level command. Use this command to insert a purge page into the next available buffer (not the next product unless the buffers are empty). A Purge Page Success (0xF9) or Purge Page Error (0xFA) will be sent to the host in response for each connected printer.

RPM1: [STX][F7][ETX] TM3:[F7]

### Purge Page Success (0xF9)

This command is sent to the host for each printer upon completion of the Purge Page process after an Insert Purge Page (0xF7) is received. This happens when a purge page has printed successfully.

RPM1: [STX][F9][ETX] TM3:[F9]

### Purge Page Error (0xFA)

This command is sent to the host for each printer upon the event of a Purge Page related error after an Insert Purge Page (0xF7) is received. This error will occur when the print job is stopped by a user.

RPM1: [STX][FA][ETX] TM3:[FA]

### Clear Print Queues (0xF9)

This command clears the printer buffers and clears the tracking of any products seen by a printers’ print sensor.

RPM1: [STX][F9][ETX] TM3:[F9]

### Clear Print Queues Success (0xFC)

This message is sent to the host once the Clear Print Queues (0xF9) command has successfully completed.

RPM1: [STX][FC][ETX] TM3:[FC]

# Software Error and Warnings

## Vega Errors and Warnings

The following is a list of errors and warnings generated from the Vega application.

Note: there currently are no Error or Warning IDs.

Errors and Warnings

|  |  |  |  |
| --- | --- | --- | --- |
| Code # | Message | Cause | Resolution |
| XXXXXX | Pen Test Warning | This warning event is generated when any nozzle in a print cartridge fails the pen test. | Turned off by default. If bad nozzle count is 14 or 22 (or multiples of) one or more of the contacts isn’t being made. Reseat the cartridge and verify the pogo pins are in the up position/cartridge contacts are clean. |
| XXXXXX | Pen Test Error | This error event is generated when all nozzles in a print cartridge fail test. This usually indicates no print cartridge is installed | Turned off by default. If enabled and cartridge is installed try removing the cartridge and reinserting. Then recheck the cartridge status. If problem persists, replace the cartridge. |
| XXXXXX | Printer Configuration Errors |  |  |
| XXXXXX | Exceeding Maximum Speed | Encoder speed is too fast for the current job resolution. The printer could not keep up. | Lower the job resolution or slow the transport speed. |
| XXXXXX | Cartridge ID Misread | Could not read cartridge ID. Some of the traces on the cartridge may be damaged. |  |
| XXXXXX | Encoder Mismatch | This error indicates that the encoder speed returned by different printers on the same line did not match. This error can occur when the speed of a transport is adjusted. Printing is not affected when the error occurs, but the operator should verify proper system operation in response to this error event before clearing the event. | Verify daisy chain cables are connected properly. If needed replace daisy chain cable. If transport indexes it is recommended to disable this error for false occurrences. |
|  | Unable to Start Print Job | The current job contains a user defined variable which is not valid while using Terminal Mode | A job with user defined variables cannot be started from the Vega application. Only the Terminal Mode host can start the job because it can remotely update the user variables. |
|  | Page Sequence Error | Board # of Printer # to Board # of Printer #, Page# to Page#, MaxPageDiff # | A printer did not print a page, likely because it was not getting the electrical signals for the encoder or the sensor. Check the electrical connections. |
|  | Timeout Error | WAIT FOR ACK: TIMEOUT ERROR | A printer stopped responding on the network. Check the network connections. |

## HP Printers

The following are the error and warning codes relevant to the HP style printers (HP Error Codes for Core, Print Controller, Satellite Controller, V7 Print Controller, Wing, iSP, Uno, Uno Plus, and MC Imager).

Error Codes

|  |  |  |  |
| --- | --- | --- | --- |
| Code # | Message | Cause | Resolution |
| 1 | Pen Data Underrun | Attempting to print with no data in buffer. | Send print data prior top of form. |
| 2 | Illegal Command | The command does not exist, incorrectly formatted, printer is in the wrong state to accept the command, or may have failed for unknown reasons. | Printer firmware and software version might not be compatible. If sending data or commands to the application verify command structure is correct. |
| 3 | External I/O Device Failure | Cannot configure or use external IO device. | Check external device connections/configurations. |
| 5 | Pen Buffer Out of Sync | If the printer ends a print using a different buffer from the current buffer. | Stop and start print. If problem persists, check pen stitching values to different values. Multiple pens cannot have the same stitching value (need to be at least 1 tick apart). Refer to stitching for details on adjusting the stitching. |
| 6 | Raster Buffer Load Error | Error loading buffer or trying to load a non-existent buffer. | Stop and start print. |
| 9 | Command Execution Error | Failure to process a command. Could be due to an update in the firmware changing the command structure or format. | Printer firmware and software version might not be compatible. If the firmware or UI software was recently updated verify the software and firmware are compatible. |
| 11 | System Overload | Printer cannot keep up with message activity. | Check the signals being sent to the printer. If too many messages/signals are sent in a short period of time it could overload the printer. |
| 23 | Out of Buffers Error | Printer received print data but all print buffers were full. | Before sending data to fill buffers verify that there are buffers available. |
| 26 | Communications Error | Message header was received but the body of the message could not be received or got a bad ETX character. | Verify the command/message being sent is correctly formatted. |
| 30 | Out of Memory Error | Printer has run out of RAM. | Reboot the printer. If problems persist replace hardware. |
| 10 | Communications Buffer Overflow | Message being sent is too long. | Verify the command/message being sent is correctly formatted. |
| 28 | XML Translation Error | An error occurred while parsing an XML file. | Verify the XML file being parsed is correctly formatted. |
| 4 | Pen Buffer Write Error |  |  |
| 21 | Partial Page Skip | Data Underrun occurred but not on all pens. Data printed on the next page will be out of synch between pens. | Increase the time of when the data is sent and when print occurs. (Increase sensor delay, send data sooner, etc…) |

### HP Warning Codes

Warning Codes

|  |  |  |  |
| --- | --- | --- | --- |
| Code # | Message | Cause | Resolution |
| 1 | No Response from Pen Driver | Pen driver board is not communicating with the printer board. | Pen stall/cable isn’t connected to printer or hardware could be faulty and needs to be replaced. |
| 3 | Pen Out of Sync Check Disabled | Notification/Confirmation that Pen Out of Sync Check has been disabled. Form width of job is shorter than the length of all the pens so multiple pages can be printing at the same time. | Make the form width wider or disable the warning in XXXXXX. |
| 4 | Pen Power-Up Failed | Print mode enabled but no cartridge was installed. | Insert cartridge before entering print mode. |
| 5 | Pen Early Page Terminiation | Sensor detected a smaller form than current job form width. This will only occur when Track Page Size is enabled. | Make sure the sensor is triggered for the entire width of the substrate. The form width in the job setup needs to be the width of the actual substrate. |
| 16 | Non-Managed Ink Cartridge Database Loaded | A non-programed cartridge is being read. | Use a managed print cartridge. |
| 17 | Failed to Load Managed Ink Params | Failed to load parameters from ink database in printer memory. | Could be the database is corrupt or formatted incorrectly. Verify database structure and resend database to printer. |
| 18 | Managed Ink, No Database Found | Failed to load ink database because file is missing | Load ink database in printer memory. |
| 20 | Draft Mode Selected | A non-managed cartridge is installed into a managed printer. Draft mode will cause the printer to print at 12.5% density. | Use a managed print cartridge. |
| 21 | Ink Cartridge Swapped | Can occur when stop print is initiated. The cartridge ID from smart card being read is differnet than the ID when printing was initiated. | Don’t swap the ink cartridge while in print mode. |
| 22 | Ink Cartridge Out of Ink | Ink cartridge is being read as empty | Replace cartridge. |
| 7 | Data Buffer Overwrite | Data buffer is being overwritten with new data. | If sending data, clear buffer then send data. |

## NACK (Negative Acknowledge) Definitions

|  |  |  |  |
| --- | --- | --- | --- |
| **Command** | **ID** | **ACK #** | **Scenario** |
| N/A | N/A | 0 | Timeout parsing command |
|  |  | 0 | Missing start character for message |
|  |  | 0 | Unknown message ID |
|  |  | 0 | Message data size > 1024 bytes |
|  |  | 0 | Incomplete message |
|  |  | 0 | Command ID not recognized |
|  |  | 0 | Ack timeout |
| TM3?? Select vega print button | ?? | 2 | Currently editing |
|  |  | 4 | No print job loaded |
|  |  | 2 | Must be on home screen |
|  |  | 0 | Failed to use print button |
| LOAD\_JOB\_FOR\_EDIT | 0x17 | 1 | Currently printing |
|  |  | 6 | Currently editing |
|  |  | 13 | User doesn't have job edit permissions |
|  |  | 5 | New job already exists |
|  |  | 8 | New job name invalid |
|  |  | 5 | Existing job not found |
|  |  | 0 | Load of existing job failed |
| LOAD\_PRINT\_JOB\_REMOTE | 0x76 | 1 | Currently printing |
|  |  | 6 | Currently editing |
|  |  | 5 | Existing job not found |
|  |  | 0 | Load of existing job failed |
| SEND\_DYNAMIC\_DATA\_TM1\_REMOTE | 0x7D | 10 | Not yet printing |
|  |  | 0 | Job doesn’t use dynamic data |
| SEND\_STATIC\_DATA\_TM1\_REMOTE | 0xA2 | 9 | Not yet printing |
|  |  | 0 | Job doesn't use static data |
| TM\_SET\_VEGA\_WINDOW\_STATE | 0x9E | 2 | Trying to hide when not on home page |
| SEND\_DYNAMIC\_DATA\_IM1\_REMOTE | 0x81 | 10 | Not yet printing |
| GET\_INK\_DATA | 0xB2 | 0 | No hardware config |
|  |  | 3 | No printers in hardware config |
|  |  | 3 | No printers online or of the correct type |
| GET\_VERSIONS | 0xB4 | 0 | No hardware config |
|  |  | 3 | No printers in config |
|  |  | 3 | No printers online or of the correct type |
| REQUEST\_PRINT\_JOB\_LIST | 0x1A | 8 | Sort field out of range 1-3 |
|  |  | 8 | Sort order out of range 1-2 |
|  |  | 0 | Couldn't get job list model |
| REQUEST\_JOB\_THUMBNAIL\_IMAGE | 0x1D | 5 | Existing job not found |
|  |  | 0 | Couldn't get job list model |
|  |  | 5 | Couldn't load thumbnail file |
| GET\_PRINT\_PREVIEW | 0x26 | 4 | No job loaded |
| DELETE\_PRINT\_JOB | 0x1C | 5 | Existing job not found |
|  |  | 6 | Job to delete is currently loaded for editing |
|  |  | 4 | Job to delete is currently loaded for printing |
| REQUEST\_FIRMWARE\_UPDATE | 0x1F | 1 | Currently printing |
|  |  | 6 | Currently editing |
|  |  | 3 | No hardware config |
|  |  | 2 | Not in "Hidden" window state |
|  |  | 8 | Printer board index is out of range |
|  |  | 0 | No valid firmware file for the selected printer board |
|  |  | 3 | Printer is not connected |
|  |  | 3 | Firmware update already in progress |
| IMPORT\_JOB | 0x18 | 5 | Import path doesn’t exist |
|  |  | 5 | Job doesn't exist in import path |
|  |  | 12 | Job already exists in Vega job directory |
|  |  | 0 | Failed to create new job folder |
|  |  | 5 | Job file is corrupted |
|  |  | 5 | Job resource mismatch |
|  |  | 5 | Job resource is corrupted |
|  |  | 0 | Not enough space on disk to import job |
|  |  | 0 | Couldn't clean job space |
|  |  | 0 | Job not found after import |
|  |  | 0 | Failed to copy job resource |
| EXPORT\_JOB | 0x19 | 5 | Existing job not found |
|  |  | 7 | Export directory does not exist |
|  |  | 5 | Job file is corrupted |
|  |  | 5 | Job resource mismatch |
|  |  | 5 | Job resource is corrupted |
|  |  | 0 | Not enough space on disk to export job |
|  |  | 12 | Job already exists in export directory |
|  |  | 0 | Cannot access export dir |
|  |  | 5 | Copy job failed |
|  |  | 5 | Unable to clean export space |
|  |  | 5 | Unable to copy job resources |
|  |  | 5 | Job resource missing |
| START\_PRINTING  START\_JOB | 0x20  0xFE | 1 | Currently printing |
|  |  | 4 | No print job loaded |
|  |  | 3 | No hardware config |
|  |  | 3 | Hardware cannot print |
|  |  | 0 | Cannot start job |
|  |  | 0 | Failed to start print job |
| ENABLE\_PRINTING  ENABLE\_PRINTING | 0x23  0xF3 | 0 | Not in correct state to enable (must be in paused state) |
| USER\_DEFINED\_VARIABLE\_REPLY | 0x16 | 11 | Not currently printing |
|  |  | 11 | Failed to process user variables |
| SET\_IO\_OUTPUT\_RELAY\_REMOTE | 0x8B | 0 | No hardware config |
|  |  | 3 | Printer/board not online |
|  |  | 8 | board number out of range |
| GET\_IO\_INPUT\_STATE\_REMOTE | 0xC | 0 | No hardware config |
|  |  | 8 | board number out of range |
| GET\_INK\_LEVELS\_REMOTE | 0x89 | 0 | No hardware config |
|  |  | 3 | No printers in config/no ink data messages sent |
| SET\_IO\_OUTPUT\_WHEN\_READY\_REMOTE | 0x8E | 8 | Relay state out of range 0-3 |
| SET\_IO\_OUTPUT\_WHEN\_ERROR\_REMOTE | 0x8F | 0 | No hardware config |
| SET\_IO\_OUTPUT\_WHEN\_INKLOW\_REMOTE | 0x90 | 0 | No IO Config |
|  |  | 8 | Imager num out of range |
|  |  | 0 | No IO Label Model |
|  |  | 0 | No grouped event model |
|  |  | 0 | Template event had empty action list |
| LOAD\_SYSTEM\_CONFIG\_REMOTE | 0x97 | 1 | Currently printing |
|  |  | 14 | file not found |
| LOAD\_DYN\_DATA\_FILE\_REMOTE | 0x79 | 16 | Not in index mode 1 |
|  |  | 1 | Currently Printing |
|  |  | 4 | No print job loaded |
|  |  | 14 | file not found |
| GET\_LOADED\_DYN\_DATA\_FILE | 0x7A | 4 | No print job loaded |
|  |  | 15 | Not dynamic data loaded |
| IM2\_4\_IDS\_ENABLE | 0xC7 | 0 | No IM2 or IM4 in the current configuration |
|  |  | 1 | The specified printer is not an IM2 or IM4 |

# Additional Monitoring and Configuration Messages

Vega has additional advanced messaging that are only available on TCP port 65000. Primarily, these messages are concerned with management functions of the Vega application from some external host software.

|  |  |  |  |
| --- | --- | --- | --- |
| **Command** | **ID** | **Direction** | **Description** |
| GET\_LINE\_CONFIG | 0xC0 | Host -> Vega | Request current config .xml and settings.xml |
| LINE\_CONFIG\_REPLY | 0xC1 | Vega -> Host |  |
| RESET\_STATUS\_COUNTER | 0xC2 | Host -> Vega |  |
| GET\_PRODUCTION\_LOG | 0xC3 | Host -> Vega |  |
| PRODUCTION\_LOG\_REPLY | 0xC4 | Vega -> Host |  |
| GET\_ALERTS | 0xC5 | Host -> Vega |  |
| ALERTS\_REPLY | 0xC6 | Vega -> Host |  |
| IM2\_4\_IDS\_ENABLE | 0xC7 | Host -> Vega |  |
|  | 0xC8 |  | Reserved |
|  | 0xC9 |  | Reserved |
| GET\_APPLICATION\_NAME | 0xCA | Host -> Vega |  |
| APPLICATION\_NAME\_REPLY | 0xCB | Vega -> Host |  |
| SEND\_JOB | 0xCC | Host -> Vega |  |
| SEND\_JOB\_REPLY | 0xCD | Vega -> Host |  |
|  | 0xCE |  |  |
|  | 0xCF |  |  |

## GET\_LINE\_CONFIG 0xC0

|  |  |  |  |
| --- | --- | --- | --- |
| ID: | 0xC0 | | |
| Direction: | Host -> Vega | | |
| Purpose: | Host application requests that connected Vega return both: the current line configuration, as well as the settings.xml file | | |
| Requester: |  | | |
| Response: | LINE\_CONFIG\_REPLY 0xC1 | | |
| Data: |  |  | |
| Description: |  | |
| Notes: | Upon receipt of LINE\_CONFIG\_REPLY, the host will need to parse the returned data. | | |
| Example: | Message:  0000 02 04 00 00 00 00 06 c0 03 ········ ·    Data:  NONE | |

## LINE\_CONFIG\_REPLY 0xC1

|  |  |  |  |
| --- | --- | --- | --- |
| ID: | 0xC1 | | |
| Direction: | Vega -> Host | | |
| Purpose: | Data returned from Vega concerning the line configuration, name, and associated data | | |
| Requester: | GET\_LINE\_CONFIG 0xC0 | | |
| Response: |  | | |
| Data: | char\* utf8CurrentConfiguration;  const char 0x0C  char\* utf8SystemSettings; | // UTF-8  // File Delimeter  // UTF-8 | |
| Description: | The host management application needs details on the printhead names and overall print height for the job layout and preview that is contained in the Vega current line configuration file.  In addition, the host management application needs to display the name of the line and store the line description which is stored in the global “System Settings.xml” file. | | |
| Notes: | The length of the returned char\* for the current line configuration is variable.  It is terminated by the 0x0C byte (form feed)  This is followed by the contents of the Vega “settings.xml” file  The host application will need to parse the contents of the returned xml files. | |
| Example: | Message (full message truncated for space):  0000 02 07 29 00 00 00 06 c1 3c 3f 78 6d 6c 20 76 65 ··)····· <?xml ve  0010 72 73 69 6f 6e 3d 22 31 2e 30 22 20 65 6e 63 6f rsion="1 .0" enco  0020 64 69 6e 67 3d 22 55 54 46 2d 38 22 3f 3e 0a 3c ding="UT F-8"?>·<  0030 48 57 5f 43 4f 4e 46 49 47 3e 0a 20 20 20 20 3c HW\_CONFI G>· <  0040 43 4f 4e 46 49 47 5f 50 41 52 41 4d 45 54 45 52 CONFIG\_P ARAMETER  0050 53 20 64 6f 75 62 6c 65 53 70 65 65 64 3d 22 66 S double Speed="f  0060 61 6c 73 65 22 20 70 61 67 65 53 79 6e 63 3d 22 alse" pa geSync="  0070 66 61 6c 73 65 22 20 64 69 73 61 62 6c 65 50 61 false" d isablePa  0080 75 73 65 50 72 69 6e 74 3d 22 66 61 6c 73 65 22 usePrint ="false"  0090 3e 0a 20 20 20 20 20 20 20 20 3c 41 43 54 49 56 >· <ACTIV | | |

## RESET\_STATUS\_COUNTER 0xC2

|  |  |  |
| --- | --- | --- |
| ID: | 0xC2 | |
| Direction: | Host -> Vega | |
| Purpose: | Reset one of the three Vega counters: simple, job, or line | |
| Requester: |  | |
| Response: |  | |
| Data: | Unsigned char ucResetCounterType; | // Value 0x01, 0x02, 0x03 |
| Description: | Host request to Vega to reset the count on one of the three counters – depending on the payload byte. | |
| Notes: | Values:  const unsigned char ucSimpleCounter = 1  const unsigned char ucJobCounter = 2  const unsigned char ucLineCounter = 3 | |
| Example: | Message:  Data: | |

## GET\_PRODUCTION\_LOG 0xC3

|  |  |  |
| --- | --- | --- |
| ID: | 0xC3 | |
| Direction: | Host -> Vega | |
| Purpose: | Requests the production log from Vega | |
| Requester: |  | |
| Response: | PRODUCTION\_LOG\_REPLY 0xC4 | |
| Data: |  |  |
| Description: | This command will request the Production Log from Vega  If no logs are present a NAK will be generated to warn the Host. | |
| Notes: | The production log on Vega is a single file that can be locally deleted if desired | |
| Example: | Message:  Data: | |

## PRODUCTION\_LOG\_REPLY 0xC4

|  |  |  |
| --- | --- | --- |
| ID: | 0xC4 | |
| Direction: | Vega -> Host | |
| Purpose: | Provides the production logs for the last number of requested hours | |
| Requester: | GET\_PRODUCTION\_LOG 0xC3 | |
| Response: |  | |
| Data: | char []; | // Variable length |
| Description: | The complete message is a UTF-8 string.  Each item within a line is a key=value pair separated by a comma  Each line of items is terminated by a 0x0D, e.g.:  VariableName=UTF-8String, | |
| Notes: | Note that the last key=value pair does not have a comma, just 0x0D | |
| Example: | Message:  Data:  JobName=?,Prints=?,StartTime=?,StopTime=?,StopReason=?,UserID=?0x0D | |

## GET\_ALERTS 0xC5

|  |  |  |
| --- | --- | --- |
| ID: | 0xC5 | |
| Direction: | Host -> Vega | |
| Purpose: | Requests the current list of Alerts (errors & warnings) | |
| Requester: |  | |
| Response: | ALERTS\_REPLY 0xC6 | |
| Data: |  |  |
| Description: | The command directs Vega to send the active alerts to the host.  The host receives WARNING\_MESSAGE\_REPLY 0xB6 and ERROR\_MESSAGE\_REPLY 0x94 messages when an alert is issued and can clear all the alerts using CLEAR\_ERROR\_AND\_WARNING\_MESSAGES 0xB7 command. | |
| Notes: | If there are no active alerts the ALERTS\_REPLY data ucNumberOfAlerts will be zero | |
| Example: | Message:  Data: | |

## ALERTS\_REPLY 0xC6

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID: | 0xC6 | | | |
| Direction: | Vega -> Host | | | |
| Purpose: | Data on all current, active alerts (errors & warnings) | | | |
| Requester: | GET\_ALERTS 0xC5 | | | |
| Response: |  | | | |
| Data: | unsigned char ucNumberOfAlerts;  char []; | | // num alerts in msg  // the alerts | |
| Description: | Vega sends this message to the host in response to a GET\_ALERTS 0xC5 message.  The complete message is a UTF-8 string.  Each line of information is terminated with a 0x0D (carriage return) byte  Each line contains key, value pairs in the form:  Key=value, | | | |
| Notes: | The current key=value pairs are:  Index:  Type:  Source:  Title:  Description:  ID: | position in the list  one of: warning or error  printer or system name  name given for the alert  description of the alert  (future use for referencing by error / warning ID number) | | |
|  | The last item in a line of data will not have a comma  If any of the values contain a comma as part of the string, then the comma must be escaped with a backslash (‘\’) character. | | | |
| Example: | Message:  Data:  Index=1,Type=Warning,Source=Front Printer,Title=?,Description=?,ID0x0D  Index=2,Type=Error,Source=Back Printer,Title=?,Description=?,ID0x0D  Index=3,Type=Warning,Source=Vega,Title=?,Description=?,ID0x0D | | |

## IM2\_4\_IDS\_ENABLE 0xC7

|  |  |  |
| --- | --- | --- |
| ID: | 0xC7 | |
| Direction: | Host -> Vega | |
| Purpose: | Tell Vega to send the IDS\_ENABLE command to all, or individual IM2 or IM4 printers in configuration | |
| Requester: |  | |
| Response: |  | |
| Data: | unsigned char ucEnable; | // 0 for OFF, 1 for ON |
| Description: | The command directs Vega to enable the Ink Delivery System (IDS) on all connected IM2/4 printers for the given line/configuration.  The data byte ucEnable – if present, will be used on a particular printer.  The data byte ucEnable – if **not** present, means Vega should send the command to all printers | |
| Notes: | If the printer being accessed is not an IM2 or IM4 type, a NACK should be issued by Vega for “not correct” | |
| Example: | Message:  Data: | |

## CLEAR\_ERROR\_AND\_WARNING\_MESSAGES 0xB7

|  |  |  |
| --- | --- | --- |
| ID: | 0xB7 | |
| Direction: | Host -> Vega | |
| Purpose: | Tell a particular Vega to clear all error and warning messages. | |
| Requester: |  | |
| Response: |  | |
| Data: | “No payload”  Index byte | // if there is no data payload – clear all errors and warnings  // if index byte exists, clear the error/warning at that index |
| Description: | If errors or warnings are detected there will be a yellow alert triangle in the upper right hand corner of the screen. Vega can automatically clear some (but not all) errors and warnings. The user can clear the messages using the alert dialog while this command lets the host do the same. | |
| Notes: | If the message is sent with no payload data, then Vega should clear everything.  If the message is sent with a byte of payload data, then Vega should interpret that byte to be the index of the error or warning at that position.  The index byte value corresponds with the index byte value returned for a given error or warning message in the ALERTS\_REPLY 0xC6 message payload  If Vega clears a given error or warning – it may repopulate (depending on the nature of the message). In this case, the alert will be re-raised by Vega. Vega will then send a WARNING\_MESSAGE\_REPLY, an ERROR\_MESSAGE\_REPLY or both. | |
| Example: | Message:  Data: | |

## GET\_APPLICATION\_NAME 0xCA

|  |  |  |
| --- | --- | --- |
| ID: | 0xCA | |
| Direction: | Host -> Vega | |
| Purpose: | Request the | |
| Requester: |  | |
| Response: | APPLICATION\_NAME\_REPLY 0xCB | |
| Data: |  |  |
| Description: |  | |
| Notes: |  | |
| Example: | Message:  Data: | |

|  |  |
| --- | --- |
|  |  |

## APPLICATION\_NAME\_REPLY 0xCB

|  |  |  |
| --- | --- | --- |
| ID: | 0xCB | |
| Direction: | Vega -> Host | |
| Purpose: |  | |
| Requester: | GET\_APPLICATION\_NAME 0xCA | |
| Response: |  | |
| Data: |  |  |
| Description: |  | |
| Notes: |  | |
| Example: | Message:  Data: | |

## SEND\_JOB 0xCC

|  |  |  |
| --- | --- | --- |
| ID: | 0xCC | |
| Direction: | Host -> Vega | |
| Purpose: |  | |
| Requester: |  | |
| Response: | SEND\_JOB\_REPLY 0xCC | |
| Data: |  |  |
| Description: |  | |
| Notes: |  | |
| Example: | Message:  Data: | |

## SEND\_JOB\_REPLY 0xCD

|  |  |  |
| --- | --- | --- |
| ID: | 0xCD | |
| Direction: | Vega -> Host | |
| Purpose: |  | |
| Requester: | SEND\_JOB 0xCC | |
| Response: |  | |
| Data: |  |  |
| Description: |  | |
| Notes: |  | |
| Example: | Message:  Data: | |