CELL-DYN Ruby® SYSTEM
Host Interface Specification

LIST NO. 09H05-01
REVISION C

ABBOTT DIAGNOSTICS DIVISION
Abbott Laboratories
Abbott Park, IL 60064

CELL-DYN Ruby® SYSTEM
Host Interface Specification

©2008 ABBOTT DIAGNOSTICS DIVISION
09H05-01C—July 2008

Abbott Laboratories
Abbott Park, IL 60064

NOTES

©2008 ABBOTT DIAGNOSTICS DIVISION

Abbott Laboratories
Abbott Park, IL 60064

TABLE OF CONTENTS

1.0 INTRODUCTION 1
1.1 Purpose 1
1.2 Scope 1
1.3 Definitions, Synonyms, and Abbreviations 1
1.4 References 1
1.5 Overview 1
2.0 STANDARD AND GENERAL CONFORMANCE 2
2.1 Applicable Standards 2
2.2 LIS1 Conformance 2
2.3 LIS2 Conformance 3
2.4 Independence of Instrument Message Transactions 4
3.0 SUPPORTED TRANSACTIONS 4
3.1 Overview 4
3.1.1 Serial Communications Link Test 4
3.1.2 Submission of Orders List Entries 4
3.1.3 Conversation for the Transfer of Test Results 5
3.1.4 Identification Requirements 5
3.1.5 Character Sets 5
3.2 Serial Communications Test (Link Test) 5
3.3 Submit Orders List Entries 6
3.3.1 The Submit Orders List Entry Message 6
3.3.2 Orders List Entry Request (issued by CELL-DYN Ruby, received by the host computer) 6
3.3.2.1 Header Record 6
3.3.2.1.1 Sender ID 7
3.3.2.1.2 Processing ID 8
3.3.2.1.3 CLSI Version No. 8
3.3.2.1.4 Example 8
3.3.2.2 Request Information Record (RIR) 8
3.3.2.2.1 Record ID 8
3.3.2.2.2 Request Information Record (RIR) Sequence Number 8
3.3.2.2.3 Starting Range ID Number 8
3.3.2.2.4 Examples 9
3.3.2.3 Message Terminator Record (MTR) 9
3.3.2.3.1 Record ID 9
3.3.2.3.2 Sequence Number 9
3.3.2.3.3 Termination Code 9
3.3.2.3.4 Example 9
3.3.3 Orders List Entry Submission (issued by the host computer, received by CELL-DYN Ruby) 9
3.3.3.1 Header Record 10
3.3.3.2 Patient Information Record (PIR) 10
3.3.3.2.1 Record ID 10
3.3.3.2.2 PIR Sequence Number 10
3.3.3.2.3 Laboratory Assigned Patient ID 11
3.3.3.2.4 Patient Name 11
3.3.3.2.5 Birthdate 11

3.3.3.2.6 Patient Sex 11
3.3.3.2.7 Attending Physician 11
3.3.3.2.8 Special Field 2 - User-Defined Demographics 12
3.3.3.2.9 Example 12
3.3.3.3 Test Order Record (TOR) 12
3.3.3.3.1 Record ID 12
3.3.3.3.2 TOR Sequence Number 13
3.3.3.3.3 Specimen ID 13
3.3.3.3.4 Universal Test ID 13
3.3.3.3.5 Specimen Collection Date and Time 14
3.3.3.3.6 Specimen Descriptor 15
3.3.3.3.7 Example 15
3.3.3.4 Message Terminator Record (MTR) 15
3.3.4 Orders List Entry Rejection (issued by CELL-DYN Ruby, received by the host computer) 15
3.3.4.1 Header Record 15
3.3.4.2 Patient Information Record (PIR) 15
3.3.4.3 Test Order Record (TOR) 15
3.3.4.4 Comment Record (CR) 16
3.3.4.4.1 Record ID 16
3.3.4.4.2 CR Sequence Number 16
3.3.4.4.3 Comment Source 16
3.3.4.4.4 Comment Text 16
3.3.4.4.5 Comment Type 18
3.3.4.4.6 Example 18
3.3.4.5 Message Terminator Record (MTR) 18
3.4 Transfer of Test Results 18
3.4.1 Transfer Test Results Conversation 18
3.4.2 Test Results Message (issued by CELL-DYN Ruby, received by the host computer) 19
3.4.2.1 Header Record 19
3.4.2.2 Patient Information Record (PIR) 19
3.4.2.3 Test Order Record (TOR) 19
3.4.2.3.1 Record ID 20
3.4.2.3.2 TOR Sequence Number 20
3.4.2.3.3 Specimen ID 20
3.4.2.3.4 Instrument Specimen ID 20
3.4.2.3.5 Universal Test ID 20
3.4.2.3.6 Specimen Collection Date & Time 20
3.4.2.3.7 Specimen Descriptor 21
3.4.2.3.8 Report Types 22
3.4.2.4 Results Record(s) 22
3.4.2.4.1 Record Type 23
3.4.2.4.2 RR Sequence Number 23
3.4.2.4.3 Universal Test ID 23
3.4.2.4.4 Data or Measurement Value 26
3.4.2.4.5 Result Status 26
3.4.2.4.6 Operator ID 27
3.4.2.4.7 Date/Time Test Started 27
3.4.2.4.8 Instrument Identification 27
3.4.2.4.9 Example of a Numerical Test Results Record 27
3.4.2.4.10 Example of a Flag Results Record 27

3.4.2.5 Histogram Data Records 27
3.4.2.5.1 Record Type 28
3.4.2.5.2 MIR Sequence Number 28
3.4.2.5.3 Data Type 28
3.4.2.5.4 Graph Type 28
3.4.2.5.5 Measurement Labels 28
3.4.2.5.6 Discriminants 29
3.4.2.5.7 Cell Data 29
3.4.2.5.8 Example of a Histogram Data Record 29
3.4.2.6 Scattergram Data Records 29
3.4.2.6.1 Record Type 30
3.4.2.6.2 MIR Sequence Number 30
3.4.2.6.3 Data Type 30
3.4.2.6.4 Graph Type 30
3.4.2.6.5 Measurement Labels 30
3.4.2.6.6 Discriminants 31
3.4.2.6.7 Cell Data 31
3.4.2.6.8 Example of a Scattergram Data Record 32
3.4.2.7 Message Terminator Record (MTR) 32
3.4.3 Test Results Acknowledgement Message (issued by the host computer, received by CELL- DYN Ruby) 32
4.0 ERROR HANDLING 33
5.0 APPENDIX A - EXAMPLES 33
5.1 Example of the CELL-DYN Ruby sending Orders List Entry Request to the host computer (Request Record) 33
5.2 Examples of Orders List Entry Submission Messages (sent by the host computer to CELL-DYN Ruby in response to query) 33
5.3 Examples of a Orders List Entry Rejection message sent by the CELL-DYN Ruby to the host computer 34
5.4 Examples of Test Result Transfer Messages (sent by CELL-DYN Ruby to the host computer) 34
6.0 APPENDIX B – BASE 64 ALPHABET AND ENCODING/DECODING DESCRIPTION 37
6.1 Base64 Alphabet 37
6.2 Base64 Encoding/Decoding Description 37
7.0 APPENDIX C – BAUD RATE CONSIDERATIONS WHEN TRANSMITTING GRAPHICAL INFORMATION IN A HOST QUERY ENVIRONMENT 38
8.0 APPENDIX D – Interface Differences by Software Release 40
8.1 Differences Between the CELL-DYN Ruby Host Interface at launch and the CELL-DYN Ruby Host Interface for software Version 2.0 ML 40

NOTES

©2008 ABBOTT DIAGNOSTICS DIVISION

Abbott Laboratories
Abbott Park, IL 60064

1.0 INTRODUCTION

1.1 Purpose

The purpose of this document is to allow host computer system vendors to develop hardware and software which will connect to a CELL-DYN Ruby instrument and successfully exchange messages with a CELL-DYN Ruby instrument.

1.2 Scope

This document applies to the CELL-DYN Ruby instrument. It specifies the messages and behaviors of the initial version of the CELL-DYN Ruby Host Computer Interface.

1.3 Definitions, Synonyms, and Abbreviations

Instrument - a CELL-DYN Ruby instrument

LIS - Laboratory Information System

LIS1 - Document identification for document formerly known as ASTM Standard E 1381

LIS2 - Document identification for document formerly known as ASTM Standard E 1394

QCID - The facility that manages the runs of quality control specimens on the instrument, including an identification that matches the barcode expected to be read from the tube.

1.4 References

Clinical and Laboratory Standards Institute (CLSI, formerly NCCLS). Standard Specification for Low-Level Protocol to Transfer Messages Between Clinical Laboratory Instruments and Computer Systems. CLSI document LIS1- A, April 2003. Formerly known as ASTM Standard E 1381-02

Clinical and Laboratory Standards Institute (CLSI, formerly NCCLS). Standard Specification for Transferring Information Between Clinical Instruments and Computer Systems. CLSI document LIS2-A, April 2003. Formerly known as ASTM Standard E 1394-97

References to the Base64 alphabet and encoding/decoding algorithms for Microsoft Windows.

- Base64 alphabet and encoding/decoding description: http://www.ietf.org/rfc/rfc2045.txt (see Appendix B)
- Microsoft Windows Operating System encoding/decoding: see atlenc.h in the Active Template Library (ATL) in Visual Studio.NET 2003

1.5 Overview

The remainder of this document is divided into sections as follows:

Standards and General Conformance - indicates the standards to which the interface will conform, and specifies general implementation dependent behavior. Implementation dependent behavior which is context-specific (i.e., within a particular message) shall be specified in *Supported Transactions* under the appropriate context (i.e., message, field, etc.).

Supported Transactions - specifies all of the transactions supported by instruments, all messages which make up the transactions and their ordering, and the format of the messages. Implementation-dependent behavior is specified in the appropriate context (i.e., if the contents of a field in message 1 are different from the contents of the same field in message 2, the description of the field in message 1 contains only those contents allowable within the context of message 1).

Error Handling -the handling of semantic errors, and logical transmission errors.

Appendix A - contains one or more examples of each transaction message for reference purposes. Note that the content of example records in Appendix A and throughout the document does not necessarily reflect results from actual samples run on an instrument or requests from an actual host computer to an instrument.

Appendix B - contains the Base64 alphabet and encoding/decoding description.

Appendix C - contains baud rate considerations when transmitting graphical information in a host query environment.

Appendix D - contains a table detailing the differences between the launch version of the CELL-DYN Ruby Host Computer Interface and the version of the CELL-DYN Ruby Host Computer Interface described in this document (for CELL-DYN Ruby software release 2.0ML).

2.0 STANDARD AND GENERAL CONFORMANCE

2.1 Applicable Standards

The instrument host computer interface conforms to LIS1 and LIS2, except as noted below. General implementation dependent behavior (behavior not dependent on context) is described in the following subsections.

- The CELL-DYN Ruby instrument sends LIS2 records packed into full size frames (compatible with the CELL- DYN 4000 and CELL-DYN Sapphire™), rather than having each LIS2 record (defined as an LIS1 message) always begin in a new frame.
- The CELL-DYN Ruby instrument (compatible with the CELL-DYN 4000 and CELL-DYN Sapphire™) does not comply with LIS2 paragraph 5.2.2 Logical Transmission Error Recovery Requirements. The only re-transmission after a line failure is performed by the LIS1 interface at the frame level.

2.2 LIS1 Conformance

The interface shall conform to LIS1 with the exceptions and specifications for implementation dependent behavior noted in the remainder of this section.

The CELL-DYN Ruby host computer interface can be configured (in the CELL-DYN Ruby application) to support the character structure (LIS1 paragraph 5.2.2.5) options in the next table.

Character Structure Elements	Options available on CELL-DYN Ruby	Default Settings
Number of Bits per Character	Seven, Eight	Eight
Number of Stop Bits	One, Two	One
Parity	Even, Odd, None	None

Eight-bit characters shall be interpreted according to the ISO8559 encoding scheme (which uses ASCII encoding for characters in the range 0-127)

Only the content of the following fields may contain text in languages other than English (content in all other fields is as specified and independent of the language in effect on the instrument):

- Specimen ID
- Patient ID
- Patient Name, Attending Physician, Special Field 2 in Patient Information Record - Operator ID in Results Record

The CELL-DYN Ruby host computer interface can be configured in the CELL-DYN Ruby application to support data transmission rates of 4800, 9600, 19200, 38400, 57600, 115200 baud (LIS1 paragraphs 5.2.3.1, 5.2.3.2). The default setting is 9600 baud.

The CELL-DYN Ruby instrument serial card used for the host computer connection has an external connector different from that described in LIS1 paragraph 5.2.4. The host computer interface connector is a DB9, male, with the contact assignments shown in the next table.

(DB9) Contact Number	Description	Direction (from instrument)
2	RxD	Input
3	TxD	Output
5	Gnd	...
Connector contacts not listed are unused.		

The CELL-DYN Ruby accepts intermediate frames with termination sequences containing the <ETB> control character (followed by checksum, <CR>, <LF>). See LIS1 paragraph 6.3.1.2.

The CELL-DYN Ruby defaults for timeouts are per LIS1 paragraphs 6.5.2.1, 6.5.2.3, 6.5.2.4, 6.5.2.5.

The CELL-DYN Ruby defaults for wait periods are per LIS1 paragraphs 6.2.7.1, 6.2.6. The values are not editable.

The instrument shall specify use of the following delimiters:

- Field delimiter: vertical bar (|) Latin-1 (124)
- Component delimiter: caret (^) Latin-1 (94)
- Repeat delimiter: backslash (\ ) Latin-1 (96)
- Escape delimiter (specified, but not used): ampersand (&) Latin-1 (38)

2.3 LIS2 Conformance

The interface shall conform to LIS2 with the specifications for general implementation dependent behavior noted in the remainder of this section.

Records containing repeat delimiters (LIS2 paragraph 6.4.4) must not be sent to the instrument. Instrument response to records containing repeat delimiters depends on where they are used and may or may not result in a rejection of the message. Repeat delimiters are used by the instrument in the Manufacturer Information Records used to transfer graphical results information.

Instruments shall accept the escape delimiter and use it to correctly parse fields within the record (see note about use of delimiters in the Specimen ID field in Specimen ID paragraph 3.3.3.3.3.). Instruments shall not decode nor create escape sequences.

The CELL-DYN Ruby shall interpret a null field value to mean that the contents of that field are unknown or irrelevant (LIS2 paragraphs 6.4.10.1 - 3). Only unsupported or optional fields may take on null values. If the CELL- DYN Ruby must have a value for the information assigned to an optional field in order to process the specimen, receipt of a null value shall cause the CELL-DYN Ruby to use its default setting for that information. The CELL- DYN Ruby does not support host computer directed deletion of any information that it has received.

Comment records must not be sent to the CELL-DYN Ruby (LIS2 section 11.0). However, comment records *are generated* by the CELL-DYN Ruby as part of the Orders List Entry Rejection message from the CELL-DYN Ruby to the host computer.

Request information records must not be sent to the CELL-DYN Ruby (LIS2 section 12.0). The CELL-DYN Ruby shall *generate* a single form of request information record in order to elicit

Orders List Entries from the host computer.

Scientific Records must not be sent to the instrument and are never generated by the instrument (LIS2 section 14.0).

Manufacturer Information Records must not be sent to the instrument (LIS2 section 15.0). However, Manufacturer Information Records *are generated* by the CELL-DYN Ruby to transfer graphical results information.

2.4 Independence of Instrument Message Transactions

Message transactions initiated by the instrument, such as an Orders List Entry Request by Specimen ID, or a Test Results message, are initiated from independent processes in the instrument. This may result in a scenario where the instrument sends Test Results messages right after it has sent an Orders List Entry Request message. The instrument does not wait for the response to the Orders List Entry Request before sending the Test Results messages.

3.0 SUPPORTED TRANSACTIONS

3.1 Overview

There are 3 conversations that can be held between the host computer and the CELL-DYN Ruby instrument. The CELL-DYN Ruby instrument maintains coordination between separate send and receive threads; however all conversations are funneled through one serial port driver. The conversations are:

- Serial Communications Link Test
- Submission of Orders List Entries
- Transfer of Test Results

3.1.1 Serial Communications Link Test

The Serial Communications Link Test is used to determine if the connection between the CELL-DYN Ruby instrument and the host computer is operational. It can also be held between the CELL-DYN Ruby instrument and Serial Loopback Device in order to check that the CELL-DYN Ruby Data Station hardware supporting the host computer link is functional. If this complete conversation takes place, the CELL-DYN Ruby Data Station considers the link to be up; otherwise the link is considered to be down and the CELL-DYN Ruby and host computer will be unable to converse with each other.

3.1.2 Submission of Orders List Entries

The Submit Orders List Entries request shall be initiated by the CELL-DYN Ruby instrument at a programmable interval (default value = every 60 seconds) whenever the CELL-DYN Ruby instrument's polling is enabled and the interval is non-zero. The host computer may send Orders List Entries at any time, independent of whether it has received a Submit Orders List Entries request. The CELL-DYN Ruby checks each received Orders List entry and sends a rejection message for each entry that is not accepted. If the CELL-DYN Ruby rejects an Orders List Entry, the host computer will have to re-submit the entry. No acknowledgement is sent for Orders List Entries that are accepted.

The CELL-DYN Ruby instrument accepts Orders List Entries pertaining to the same specimen identifier if they meet the requirements for unique entries. Therefore, if the first Orders List Entry received for Patient A (with Specimen ID = M34566) was semantically and syntactically correct, but not what the doctor ordered, the host computer may send a second Orders List Entry for Patient A (Specimen ID = M34566). Both Orders List Entries will co-exist in the CELL-DYN Ruby instrument's Orders List.

Orders List Entries are deleted from the CELL-DYN Ruby instrument by the CELL-DYN Ruby operator, by the instrument after the run, or by the instrument after the Orders List Entry reaches a certain age.

3.1.3 Conversation for the Transfer of Test Results

The Transfer of Test Results conversation shall be initiated by the CELL-DYN Ruby instrument. If the CELL-DYN Ruby instrument is set to “auto-transmit,” the transfer is initiated as the test results become available and qualify for transmission. The transfer of test results may also be initiated by the CELL-DYN Ruby operator via a manual transmit action. Each conversation of this type contains information about a single aspiration on the CELL-DYN Ruby. The host computer must acknowledge that it has received the information via LIS1 level frame ACK’s.

3.1.4 Identification Requirements

The CELL-DYN Ruby identifies itself and the Specimen ID in each conversation (except the serial link test and Orders List Entry Request ALL message) in order to help the host computer track multiple instances of the conversations, or conversations with multiple CELL-DYN Ruby instruments. Similarly, the host computer ***must*** provide the Specimen ID in any LIS2 level message which it sends to the CELL-DYN Ruby instrument. The host computer does not have to identify itself, because the CELL-DYN Ruby can only communicate to one host computer at a time.

3.1.5 Character Sets

The CELL-DYN Ruby instrument communicates using text strings for the message content. The character set range is from 32-127 for 7-bit characters (ASCII encoding), and from 32-255 for 8-bit characters (ASCII/ISO8559 encoding) regardless of the National Language support installed or active on the CELL-DYN Ruby instrument.

3.2 Serial Communications Test (Link Test)

The Serial Communications Test is transacted at the low level interface (LIS1) and consists of the CELL-DYN Ruby issuing an <ENQ>, waiting for a response, and then replying to the response.

The host computer response should be either <ACK>, <NAK>, or <ENQ>.

If the host computer response is an <ACK>, the CELL-DYN Ruby issues an <EOT> to return the link to the neutral condition.

The Serial Loopback Device response should be <ENQ>.

If the CELL-DYN Ruby receives a response of <ACK>, <NAK>, or <ENQ> within the configured establishment timeout period (LIS1 paragraph 6.5.2.1), the link status is reported as up. Any other response, or a timeout, will result in the CELL-DYN Ruby reporting that the link status is down.

The Serial Communications Test shall be conducted based on its enabled/disabled state and the value of its configurable time period whenever a host computer is configured to be present, as follows.

- If enabled: The Serial Communications Test is initiated after the serial driver is successfully opened and periodically after, based on the time period
- If disabled: The Serial Communications Test is initiated once after the serial driver is successfully opened

The time period is configurable (in integer minutes) between 1 and 120.

The Serial Communications Test will also be initiated by the CELL-DYN Ruby at the CELL-DYN Ruby operator’s request.

The Serial Communications Test may be conducted with the host computer connected to the CELL-DYN Ruby Data Station or with the Serial Loopback Device connected to the CELL-DYN Ruby Data Station.

3.3 Submit Orders List Entries

3.3.1 The Submit Orders List Entry Message

If the CELL-DYN Ruby instrument's polling interval is non-zero and the link is up, it will periodically transmit an Orders List Entry Request message consisting of one (1) Header Record, followed by one (1) Request Information Record, followed by one (1) Message Terminator Record. This message may also be sent if host query by Specimen ID is enabled on the instrument.

If the host computer has no test orders pending to be sent to the CELL-DYN Ruby instrument which issued the request for information, the host computer should not respond to the inquiry (except to acknowledge the request as required by the LIS1 level protocol).

If the host computer has one or more test orders pending to be sent to the CELL-DYN Ruby instrument which issued the request, then the host computer should respond with an Orders List Entry Submission message consisting of one (1) Header Record, followed by one or more (1, n) Patient-Order Record Combinations, followed by one (1) Message Terminator Record. A Patient-Order Record Combination shall consist of one (1) Patient Information Record, followed by one (1) Test Order Record. There must be 1 and may be up to 3000 Patient-Order Record Combinations in a single Orders List Entry Submission message. Note: the CELL-DYN Ruby's multi-tasking capability allows it to process Orders List Entries out of the Orders List even as additional entries are being submitted.

The CELL-DYN Ruby shall parse the Orders List Entry Submission and respond with an Orders List Entry Rejection message for each Patient-Order Record Combination in the Orders List Entry Submission that was not accepted. If the following conditions are met, an Orders List Entry Rejection message shall consist of one (1) Header Record, followed by one (1) Patient Information Record, followed by one (1) Test Order Record, followed by one (1) Comment Record, followed by one (1) Message Terminator Record; otherwise, only the Header, Comment, and Terminator records are present.

- The Orders List Entry Submission message has 4 or more records
- The first record in the message is a Header Record (sequence number is not validated)
- The last record in the message is a Terminator Record (sequence number must be as specified)
- Records between the Header and Terminator must be in proper POPOPO... sequence (sequence numbers are not validated)

The Test Order Record's Specimen ID field (LIS2 paragraph 9.4.3) and Report Type field (LIS2 paragraph 9.4.26) will be used to encode the CELL-DYN Ruby's rejection of the Orders List Entry for the Patient-Order Record combination parsed.

When a given Patient-Order combination is not accepted, the host computer will be responsible for maintaining the test order in its list of pending orders and downloading it to the CELL-DYN Ruby later on.

No acknowledgement is sent for Orders List Entries that are accepted.

3.3.2 Orders List Entry Request (issued by CELL-DYN Ruby, received by the host computer)

3.3.2.1 Header Record

The Header Record shall conform to LIS2 section 7.0, with the fields used and mapped as shown in Table 1. Required fields shall be provided with non-null

entries. Fields not listed in the table are nulled or not produced by the CELL-DYN Ruby as per LIS2 paragraphs 6.4.8 and 6.4.9.

Table 1 — Header Record Field Usage for Orders List Entry Request Message

LIS2 ref	Item	Req.	LIS2/CELL-DYN Ruby Content
7.1.1	Record ID	x	H
7.1.2	Delimiters	x	|\^&
7.1.5	Sender ID	x	Analyzer Serial Number, CELL-DYN Ruby Model, CELL-DYN Ruby Software Version ID, CELL-DYN Ruby Host Computer I/F Version ID
7.1.12	Processing ID	x	P
7.1.13	CLSI Version No.	x	LIS2-A

The Record ID must be the single letter H in either upper or lower case.

The Delimiters field contents must conform to LIS2 with no additional restrictions.

3.3.2.1.1 Sender ID

The Sender ID field shall be composed of component fields as described in Table 2; the field order shall be ASN^M^SVID^HCVID. The first two components collectively describe the CELL-DYN Ruby's concept of Analyzer ID.

Table 2 — Sender ID components

Mnemonic	Definition
ASN	Analyzer Serial Number
M	CELL-DYN Ruby Model
SVID	CELL-DYN Ruby Software Version ID
HCVID	CELL-DYN Ruby Host Computer I/F Version ID

The Analyzer Serial Number shall be an ASCII character string of at least 1 and at most 9 characters. Mixed case, punctuation, alpha, and numeric characters are allowed. All characters must be in the ASCII range 32-255, inclusive. This string is used to uniquely identify the CELL-DYN Ruby (Analyzer) Instrument to its manufacturer.

The CELL-DYN Ruby Model shall be the ASCII character string, "CDRuby."

The CELL-DYN Ruby Software Version ID shall be an ASCII character string of at least 1 and at most 16 characters. Mixed case, punctuation, alpha, and numeric characters are allowed. All characters must be in the ASCII range 32-255, inclusive. The string is used to uniquely identify the version of the CELL-DYN Ruby application software resident on the CELL-DYN Ruby Instrument.

The CELL-DYN Ruby Host Computer Revision ID shall be the 3-character ASCII character string, "1.0". It uniquely identifies the

version of CELL-DYN Ruby Host Computer interface supported in this communication.

3.3.2.1.2 Processing ID

The Processing ID field must be the single letter P in either upper case or lower case.

3.3.2.1.3 CLSI Version No.

The CLSI Version No. field must be the string "LIS2-A."

3.3.2.1.4 Example

An example of a conforming header record is:
H|\^&|||40009603A^CDRuby^R5-4H^1.0|||||||P|LIS2-A<CR>

3.3.2.2 Request Information Record (RIR)

The Request Information Record shall conform to LIS2 section 12.0, with the fields used and mapped as shown in Table 3. Required fields shall be provided with non-null values. Fields not listed in the table are nulled or not produced by the CELL- DYN Ruby as per LIS2 paragraphs 6.4.8 and 6.4.9.

Table 3 — Request Information Record Field Usage for Orders List Entry Request Message

LIS2 ref	Item	Req.	LIS2/CELL-DYN Ruby Content
12.1.1	Record ID	x	Q
12.1.2	RIR Sequence Number	x	1
12.1.3	Starting Range ID Number	x	see below

3.3.2.2.1 Record ID

The Record ID must be the single letter Q in either upper or lower case.

3.3.2.2.2 Request Information Record (RIR) Sequence Number

The RIR Sequence Number shall be the single digit 1.

3.3.2.2.3 Starting Range ID Number

The Starting Range ID Number shall be a two-component field. The first component shall be null. The second component shall be the uppercase 3-character ASCII string “ALL” to request all Test Orders and shall be a Specimen ID (as described in the Test Order Record in the Orders List Entry Submission message) to request all Test Orders for that Specimen ID.

When requesting Test Orders for a particular Specimen ID, the CELL-DYN Ruby waits for the Host Query Timeout period (default = 5 seconds, LIS response time + 1 second) before using the CELL-DYN Ruby defaults to process the sample. The LIS must respond within 4 seconds. That part of the Host Query Timeout period within which the LIS must respond begins when the CELL-DYN Ruby transmits the LIS1 end of transmission (EOT) character at the end of the Orders List Entry Request Message and ends when the CELL-DYN Ruby receives the LIS1 EOT character at the end of the Orders List Entry Submission message.

To help ensure that the query leads to the successful transfer of information about one or more pending orders for the Specimen ID, the CELL-DYN Ruby prioritizes the transfer of the query message above pending results transmissions, and does not initiate any new results transmissions until after the Host Query Timeout period expires.

3.3.2.2.4 Examples

Examples of CELL-DYN Ruby Request Information Records are:

Q|1|^ALL<CR>to request all Test Orders

Q|1|^M34566<CR> to request all Test Orders for M34566

3.3.2.3 Message Terminator Record (MTR)

The Message Terminator Record shall conform to LIS2 section 13.0, with the fields used and mapped as shown in Table 4. Required fields shall be provided with non-null values; optional fields may have null or non-null values. Fields not listed in the table are nulled or not produced by the CELL-DYN Ruby as per LIS2 paragraphs 6.4.8 and 6.4.9.

Table 4 — Message Terminator Record Field Usage for Orders List Entry Request Message

LIS2 ref	Item	Req.	LIS2/CELL-DYN Ruby Content
13.1.1	Record ID	x	L
13.1.2	MTR Seq Number	x	1
13.1.3	Termination Code	x	N, T, E

3.3.2.3.1 Record ID

The Record ID must be the single letter L in either upper or lower case.

3.3.2.3.2 Sequence Number

The Sequence Number shall be the single digit 1.

3.3.2.3.3 Termination Code

The Termination Code must be either N, T, or E. If the Termination Code is N, the message shall be processed normally. If the Termination Code is T or E, the entire message should/shall be discarded.

3.3.2.3.4 Example

A valid example of a Message Terminator Record is L|1|N<CR>

3.3.3 Orders List Entry Submission (issued by the host computer, received by CELL-DYN Ruby)

3.3.3.1 Header Record

The Header Record issued by the host computer as part of its Orders List Entry Submission must conform to LIS2 section 7.0. Table 5 specifies the fields of interest to the CELL-DYN Ruby. Required fields must contain non-null entries; optional fields may contain null or non-null values. Optional fields need not be included in the record unless needed for field count (see LIS2 paragraphs 6.4.8 and 6.4.9). Any additional fields produced by the host computer as part of the Header Record shall be ignored by the CELL-DYN Ruby. Extra fields, if properly delimited, shall not invalidate the record.

Table 5 — Header Record Field Usage for Orders List Entry Submission Message

LIS2 ref	Item	Req.
7.1.1	Record ID	x
7.1.2	Delimiters	x
7.1.12	Processing ID	
7.1.13	CLSI Version No.	

All Header Record field definitions are identical to the definitions provided in the Header Record for the Orders List Entry Request message.

3.3.3.2 Patient Information Record (PIR)

The Patient Information Record shall conform to LIS2 section 8.0, with the fields used as shown in Table 6.

Table 6 — Patient Information Record Field Usage for Orders List Entry Submission Message

LIS2 ref	Item	Req.	LIS2/CELL-DYN Ruby Content
8.1.1	Record ID	x	P
8.1.2	PIR Seq Number	x	1, 2, 3 ...
8.1.4	Laboratory Assigned Patient ID		Patient ID
8.1.6	Patient Name		Patient Name
8.1.8	Birthdate		date of birth
8.1.9	Patient Sex		sex
8.1.14	Attending Physician		doctor’s name
8.1.16	Special Field 2		User Field 1 User Field 2

3.3.3.2.1 Record ID

The Record ID must be the single letter P in upper or lower case.

3.3.3.2.2 PIR Sequence Number

The Sequence Number for the first Patient Information Record in the Orders List Entry Submission message must be the single digit 1; each successive Patient Information Record contained in

the same Orders List Entry Submission message shall have a Sequence Number one more than the Sequence Number in the previous record.

3.3.3.2.3 Laboratory Assigned Patient ID

If present, the Laboratory Assigned Patient ID field shall consist of a 16-character or less string. The characters must be in the ASCII range from 32-255, inclusive. The contents of the Laboratory Assigned Patient ID field shall be mapped into the Patient ID field on the CELL-DYN Ruby.

3.3.3.2.4 Patient Name

If present, the Patient Name field shall consist of a 28-character or less string. The characters must be in the ASCII range from 32-255, inclusive.The contents of the Patient Name field shall be mapped into the Patient Name field on the CELL-DYN Ruby. Note that this single component Patient Name field does not contain the multiple components specified in LIS2 paragraph 8.1.6.

3.3.3.2.5 Birthdate

If present, the Birthdate field shall consist of a date string (YYYYMMDD) per LIS2 paragraph 6.6.2. This field shall be mapped to the birthdate field on the CELL-DYN Ruby. The current year minus 150 is the oldest valid YYYY for this field. A Birthdate in the future is not valid.

3.3.3.2.6 Patient Sex

If present, the Patient Sex must be 1 character in length. The valid character choices for the representation of Patient Sex are “M” for male, “F” for female, “U” for unknown; either upper or lower case is acceptable. The contents of the Patient Sex field is mapped to the patient sex field on the CELL-DYN Ruby.

3.3.3.2.7 Attending Physician

If present, the Attending Physician field shall consist of a 22-character or less string. The characters must be in the ASCII range from 32-255, inclusive. The contents of this field are mapped to the Doctor’s Name field on the CELL-DYN Ruby. Note that this single component Attending Physician field does not contain the multiple components specified in LIS2 paragraph 8.1.14.

3.3.3.2.8 Special Field 2 - User-Defined Demographics

If present, Special Field 2 shall consist of 2 components, ordered as UF1^UF2. Each component shall be 16 characters or less in length. Any component may be null, as per the CLSI standard. All characters must be in the ASCII range 32-255, inclusive. The components shall be mapped to the User Defined fields of the CELL-DYN Ruby as shown in Table 7. Use of and interpretation of these fields should be defined by conventions established between the laboratory staff and host computer system. The CELL-DYN Ruby instrument does not interpret these fields.

Table 7 — Special Field 2 Component Mapping

Mnemonic	CELL-DYN Ruby Definition in Demographics
UF1	User Field 1
UF2	User Field 2

3.3.3.2.9 Example

An example of a conforming Patient Information Record is:
P|1||A56342||David Jones||19640315|M|||||Roberts||Phillips^sickle crisis<CR>

3.3.3.3 Test Order Record (TOR)

The Test Order Record shall conform to LIS2 section 9.0, with the fields used as shown in Table 8. Multiple test orders within a single Test Order Record are not supported.

Table 8 — Test Order Record Field Usage for Orders List Entry Submission Message

LIS2 ref	Item	Req.	LIS2/CELL-DYN Ruby Content
9.4.1	Record ID	x	O
9.4.2	TOR Seq Number	x	1
9.4.3	Specimen ID	x	Specimen ID
9.4.5	Universal Test ID		Test Selection, Parameter Set Selection, Limit Set Selection
9.4.8	Spec. Collection Date & Time		Draw date Draw time
9.4.16	Specimen Descriptor		Specimen Type Specimen Subtype
9.4.26	Report Type	x (see note below)	
Note: Required only for Orders List Entry Rejection Messages			

3.3.3.3.1 Record ID

The Record ID must be the single letter O in either upper or lower case.

3.3.3.3.2 TOR Sequence Number

The Sequence Number for the Test Order Record in the Orders List Entry Submission message shall be the single digit 1 as there shall be one and only one Test Order Record per Patient Information Record. There can be only one test selected per Test Order Record.

3.3.3.3.3 Specimen ID

The Specimen ID must be at least three (3) and not more than twenty (20) characters in length. The Specimen ID must not be blank; it must not have embedded blanks; it must not contain delimiters used in host computer messages or those specified for use in instrument messages (see LIS2 Conformance section of this document). The Specimen ID must not contain any asterisks (*), nor begin with the tilde character (~), nor contain the following reserved words: "Invalid_ID" (case insensitive), "No_ID" (case insensitive). The Specimen ID should match the barcode label on the specimen tube which contains the blood sample from the Patient identified in the associated Patient Information Record. Use of a check digit on the barcode or as part of the Specimen ID should be coordinated against the CELL-DYN Ruby's Analyzer Barcode Reader Check Digit Setup selections. The barcoded specimen tube has its barcode read by the CELL-DYN Ruby Analyzer, transferred to the CELL- DYN Ruby Data Station, and matched against the Specimen ID field of the Orders List Entries in order to identify which test is to be run on the sample which is being aspirated. The content of this field is critical to proper identification, analysis, and reporting of patient results.

3.3.3.3.4 Universal Test ID

If present, the Universal Test ID is composed of multiple components as described in Table 9 below; the field order shall be ^^^TS^PSS^LSS. The first 3 components of the Universal Test ID are reserved by LIS2 and thus are presented as null components in the Universal Test ID. Null values in any of the components 4 - 6 will cause the CELL-DYN Ruby instrument to supply an appropriate value.

Table 9 — Universal Test ID Component Field Usage for Test Order Record

Component Field (positional)	**Universal Test ID Mnemonic**	**Definition**
1	Universal Test ID	Reserved by LIS2
2	Universal Test ID Name	Reserved by LIS2
3	Universal Test ID Type	Reserved by LIS2
4	TS	CELL-DYN Ruby Test Selection
5	PSS	CELL-DYN Ruby Parameter Set Selection

Table 9 — Universal Test ID Component Field Usage for Test Order Record

6	LSS	CELL-DYN Ruby Patient Limit Set Selection

If present, the Test Selection component must be one of the Test Selection Strings shown in Table 10, which correspond to the Test Selections available on the CELL-DYN Ruby instrument. The Test Selection controls which test is run, analyzed, and reported. Note that support for the NON-WHOLE BLOOD* Test Selection is configurable on the instrument (enabled only after entry of an Abbott provided security code), and will be disabled by default. When disabled, the instrument will reject Orders List Entry Submission messages containing a NON-WHOLE BLOOD* Test Selection (as a Test Selection parse error).

Table 10 — CELL-DYN Ruby Test Selections

Test Selection Strings
CBC
CBC+NOC
CBC+RRBC
RETIC
NON-WHOLE BLOOD*

If present, the Parameter Set Selection component must be a number between 1 and 8. The Parameter Set Selection is used by the CELL-DYN Ruby to control local presentation of the CELL-DYN Ruby's test results; it does not affect the information reported to the host computer.

If present, the Limit Set Selection component should be 0, to indicate that the instrument should select the Limit Set based on the age/sex of the patient. Note: since the instrument ensures a value of 0 is used for the selection, a non-null value other than 0 for this field does not cause the associated Orders List Entry Submission Message to be rejected.

The Limit Set Selection is used by the CELL-DYN Ruby to control local presentation of the CELL-DYN Ruby's test results; it does not affect the information reported to the host computer.

3.3.3.3.5 Specimen Collection Date and Time

If present, the Specimen Collection Date and Time field must conform to LIS2 paragraph 6.6.2 (YYYYMMDDHHMMSS). It shall be stored by the CELL-DYN Ruby in the Draw Date and Draw Time fields.

The current year minus 150 is the oldest valid YYYY for the field. A Collection Date in the future is not valid.

* In Development

3.3.3.3.6 Specimen Descriptor

If present, the Specimen Descriptor shall be composed of 2 component fields as described in Table 11; the field order shall be ST^SST.

Table 11 — Specimen Descriptor Component Definitions

Mnemonic	Definition
ST	Specimen Type
SST	Specimen Subtype

If present, the Specimen Type shall be the following case sensitive ASCII character string: “Patient.”

If the Specimen Type is "Patient," then the Specimen Subtype is the case sensitive ASCII string “Human.”

If the Specimen Type is null, then the Specimen Subtype must be null.

3.3.3.3.7 Example

An example of a conforming Test Order Record is:
O|1|19345||^^^CBC^5^0|||19960810153028|||||||Patient^Human <CR>

3.3.3.4 Message Terminator Record (MTR)

The Message Terminator Record for a Orders List Entry Submission message shall be as described earlier for the Orders List Entry request message.

3.3.4 Orders List Entry Rejection (issued by CELL-DYN Ruby, received by the host computer)

3.3.4.1 Header Record

The Header Record for a Orders List Entry Rejection message shall be as described earlier for the Orders List Entry Request message.

3.3.4.2 Patient Information Record (PIR)

The contents of the Patient Information Record shall be an exact copy of the Patient Information Record of the Patient-Order Combination of the Orders List Entry Submission message that is being rejected with the exception of the Patient Information Record Sequence Number shall be the single digit 1. See the PIR described earlier for the Orders List Entry Submission message. A Patient Information Record is not included in the rejection message if the records in the Orders List Entry Submission message did not comply with the record requirements of the message (see The Submits Orders List Entry Message paragraph 3.3.1).

3.3.4.3 Test Order Record (TOR)

The contents of the Test Order Record shall be an exact copy of the Test Order Record of the Submit Orders List Entry message that is being rejected, with the exception that there shall be a non-null entry in the Report Types field (LIS2 paragraph 9.4.26). Note that the Specimen ID field is available to ensure correct association of the particular Orders List Entry Rejection with the host computer Patient-Order being rejected. See the Test Order Record in the Orders List Entry Submission message. A Test Order Record is not included in the rejection message if the records in the Orders List Entry Submission message did not comply with the record requirements of the message (see The Submits Orders List Entry Message paragraph 3.3.1).

The Report Type field value shall be “X” to indicate that no Orders List Entry has been made in the CELL-DYN Ruby Orders List for the associated Patient-Test Order. This is essentially a high level (LIS2) NAK. The reason for the CELL-DYN Ruby’s rejection of the Patient-Order combination shall be provided in a Comment Record which shall immediately follow the Test Order Record.

The host computer should disposition the rejected Patient-Order combinations as appropriate.

3.3.4.4 Comment Record (CR)

The Comment Record shall conform to LIS2 section 11.0, making use of the fields listed in Table 12. The Comment Record shall specify the reason the Patient-Order combination of the Orders List Entry message was rejected.

Table 12 — Comment Record Field Usage for Orders List Entry Rejection Message

LIS2 ref	Item	Req.	LIS2/CELL-DYN Ruby Content
11.1.1	Record ID	x	C
11.1.2	CR Sequence Number	x	1
11.1.3	Comment Source	x	I
11.1.4	Comment Text	x	see below
11.1.5	Comment Type	x	G

3.3.4.4.1 Record ID

The Record ID shall be the single character C in either upper or lower case.

3.3.4.4.2 CR Sequence Number

The Comment Record Sequence Number shall be the single digit 1.

3.3.4.4.3 Comment Source

The Comment Source shall be the single character I in either upper or lower case.

3.3.4.4.4 Comment Text

The Comment Text shall consist of two strings separated by the component delimiter. The first string is one of the strings shown in the leftmost column of Table 13 and shall reflect the main reason for rejecting the Patient-Test Order combination. The second string is one of the following (shown with a short explanation of the reason):

- None <used as the second string when the first string is other than Parsing Error>
- MessageType <record sequence does not conform to a recognized message. Note: this message may override another error (for example an invalid Record ID) if the other error results in an unrecognizable record sequence>
- RecordID <invalid Record ID>

- SequenceNo <invalid Sequence Number>
- Header <invalid Header record>
- Terminator <invalid Terminator record>
- DOB <invalid date>
- Sex <invalid sex field>
- SpecimenID <invalid Specimen ID field>
- TestSelect <invalid Test Selection field>
- ParamSelect <invalid Parameter Set Selection field>
- CollectionDateTime <invalid date/time>
- SpecimenType <invalid Specimen Type>
- SpecimenSubtype <invalid Specimen Subtype>
- Patient Name <contains invalid ASCII character(s)>
- Patient ID <contains invalid ASCII character(s)>
- Doctor <contains invalid ASCII character(s)>
- Demographics <contains invalid ASCII character(s)>

Table 13 — Orders List Entry Rejection Indications

Host Computer Interface String	**Associated Rejection Criterion**
Orders List Full	The CELL-DYN Ruby instrument Orders List is full; an Orders List Entry for this Patient-Order combination could not be made at this time.
Order List Add Failed	There was an error during an attempt to create an order in the CELL-DYN Ruby instrument Orders List; an Orders List Entry for this Patient-Order combination was not made.
Duplicate Order	There is already an Order in the CELL-DYN Ruby instrument Orders List that matches this Order, based on the Specimen ID and the Test Selection.
RRTT Orders List	The CELL-DYN Ruby instrument Orders List is configured to match on rack number and tube position (RRTT), and is not accepting any orders from the host computer.
ID Conflict in the Order	The Specimen ID in the Order matches the Specimen ID in an existing CELL-DYN Ruby instrument Order, but the Patient IDs in the two orders do not match. Note that a null Patient ID in one order and a non-null Patient ID in the other is considered a mismatch.
Specimen ID in the Order is not unique	The Specimen ID in the Order matches the Specimen ID in an existing CELL-DYN Ruby instrument Order, and the Test Selections in the two Orders are for a whole blood specimen and a non-whole blood specimen; or the Specimen ID in the Order matches the Specimen ID in a CELL-DYN Ruby instrument non-Orders List based facility, for example the identifier for a QCID.
Parsing Error	A parsing error occurred or the value associated with one or more of the fields was incorrect in content.

3.3.4.4.5 Comment Type

The Comment Type shall be the single character G in either upper or lower case.

3.3.4.4.6 Example

The following is an example of an Orders List Entry Rejection Comment Record detailing the rejection as an Orders List full condition.

C|1|I|Orders List Full^None|G<CR>

3.3.4.5 Message Terminator Record (MTR)

The Message Terminator Record for a Orders List Entry Rejection message shall be as described earlier for the Orders List Entry Request message, and shall have a Termination Code of "E."

3.4 Transfer of Test Results

3.4.1 Transfer Test Results Conversation

The Transfer Test Results conversation is transacted in high level (LIS2) messages. The conversation results in the transfer of the set of Test Results for one (1) aspiration on the CELL-DYN Ruby Instrument (i.e., one Sample Record).

The CELL-DYN Ruby must begin the conversation with a Test Result Transfer message consisting of one (1) Header Record, followed by one (1) Patient Information Record, followed by one (1) Test Order Record, followed by (i) Result Records containing numerical results, optionally followed by (j) Result Records containing flagging results, optionally followed by (k) Manufacturer Information Records containing graphical results, followed by one (1) Message Terminator Record.

The Result Records each itemize a single quantitative or flagging result produced by the CELL-DYN Ruby instrument. The status of each “numerical result” as determined by the CELL-DYN Ruby instrument is also reported in the pertinent Result Record.

Each Manufacturer Information Record specifies either a single histogram or a set of scattergrams, including the name, axis labels, discriminants, and data points.

The CELL-DYN Ruby-host computer link must be up in order for the CELL-DYN Ruby to begin a Transfer Test Results conversation. The CELL-DYN Ruby shall begin a Transfer Test Results conversation when directed to do so by the CELL-DYN Ruby operator. In addition, if auto-transmit is enabled, the CELL-DYN Ruby shall begin the Transfer Test Results conversation whenever a Sample Record which qualifies for auto-transmission becomes available at the conclusion of sample processing of the associated aspiration. The ability to and qualification criteria for auto-transmission of a Sample Record is configured by the CELL-DYN Ruby operator. Note that the CELL-DYN Ruby may transmit Test Results to the host computer which are associated with specimens other than the ones submitted to the CELL-DYN Ruby by the host computer.

The host computer must parse the Test Results Transfer message as described in Test Results Acknowledgement Message paragraph 3.4.3. If the Test Results Transfer message is successfully transmitted, no further attempts to transmit this Sample Record will be made unless the CELL-DYN Ruby instrument is explicitly directed to do so by the operator. If the Test Results Transfer message is not successfully transmitted, the CELL-DYN Ruby remembers this so that it can be resent.

The Sample Records in the CELL-DYN Ruby's "to be transmitted" queue are not cleared as a result of the loss of the CELL- DYN Ruby-host computer link, and are also not cleared when power to the CELL-DYN Ruby is cycled. The "to be transmitted" queue may be purged of all pending Sample Records by direct instruction from the CELL-DYN Ruby operator to the CELL-DYN Ruby instrument. Qualifying Sample Records continue to be added to the "to be transmitted" queue while the link is down; if transmission from the queue has been enabled by the operator (normal condition), transmission of queued Sample Records begins automatically when the link is restored.

3.4.2 Test Results Message (issued by CELL-DYN Ruby, received by the host computer)

3.4.2.1 Header Record

The Header Record for a Test Results Transfer message shall be as described earlier for the Orders List Entry Request message.

3.4.2.2 Patient Information Record (PIR)

There will be one and only one Patient Information Record per Test Results Transfer message. The Patient Information Record Sequence Number shall be the single digit 1. The Patient Information Record shall conform to LIS2 section 8.0, with content taken from the following result record fields from Table 6 based on the Specimen Type.

Patient Specimen Type:

- Laboratory Assigned Patient ID, Patient Name, Birthdate, Patient Sex, Attending Physician, Special Field 2

All other Specimen Types:

- None

3.4.2.3 Test Order Record (TOR)

The Test Order Record of a Test Results Transfer message shall conform to LIS2 section 9.0 with the field usage and mapping shown below in Table 14.

Table 14 — Test Order Record Field Usage for Test Results Transfer Message

LIS2 ref	Item	Req.	LIS2/CELL-DYN Ruby Content
9.4.1	Record ID	x	O
9.4.2	TOR Seq Number	x	1
9.4.3	Specimen ID	x (see note 1 below)	Specimen Tube ID, Rack Number, Tube Position
9.4.4	Instrument Specimen ID	x	Sequence Number
9.4.5	Universal Test ID	x	Test Selection, Parameter Set Selection, Limit Set Selection
9.4.8	Spec. Collection Date & Time		draw date draw time
9.4.16	Specimen Descriptor	x	Specimen Type Specimen Subtype
9.4.26	Report Types	x	F
Note 1: The Specimen Tube ID component is only required for records with a Specimen Type of Patient.			

3.4.2.3.1 Record ID

The Record ID shall be the single character O, in either upper or lower case.

3.4.2.3.2 TOR Sequence Number

The Sequence Number shall be the single digit 1.

3.4.2.3.3 Specimen ID

When required, the Specimen ID field shall consist of 3 components as noted below:

- Specimen Tube ID: an ASCII string of 1 to 20 characters in length. It shall be the actual Specimen ID value recorded on the CELL-DYN Ruby instrument. As such, it may or may not have been received from the host computer in a previous Orders List Entry Submission message.
- Rack Number: an ASCII string 2 characters in length. If the specimen was run in closed mode, it shall be the actual Rack Number recorded on the CELL-DYN Ruby instrument. This component is null for specimens run in open mode.
- Tube Position: an ASCII string 2 characters in length. If the specimen was run in closed mode, it shall be the actual Tube Position recorded on the CELL-DYN Ruby instrument. This component is null for specimens run in open mode.

3.4.2.3.4 Instrument Specimen ID

The Instrument Specimen ID shall be a 4-character ASCII string of digits in the range 0000 to 9999, inclusive. The Instrument Specimen ID represents the CELL-DYN Ruby instrument's Datalog sequence number which is used, along with the date and time of the test, to uniquely identify a given aspiration on the CELL-DYN Ruby instrument. As the Datalog sequence number rolls over at 9999, it is not a unique identifier in and of itself.

3.4.2.3.5 Universal Test ID

The Universal Test ID field of the Test Order record shall be as defined earlier in the Orders List Entry Submission message with the exception that component fields 4, 5, and 6 shall be as specified below based on the Specimen Descriptor (Specimen Type, Specimen Subtype).

3.4.2.3.6 Specimen Collection Date & Time

The Specimen Collection Date and Time shall be as described earlier for the Orders List Entry Submission message.

3.4.2.3.7 Specimen Descriptor

The Specimen Type shall be the ASCII string "Patient," the Specimen Subtype shall be null, and component fields 4, 5, and 6 of the Universal Test ID field shall be non-null and as described earlier in the Orders List Entry Submission message for runs with an instrument Specimen Type of Patient.

The Specimen Type shall be the ASCII string "QC," the Specimen Subtype shall be the ASCII string "Whole Blood," and component fields 4, 5, and 6 of the Universal Test ID field shall be as specified below for runs with an instrument Specimen Type/ Subtype of QC/Whole Blood:

- Component field 4 (Test Selection), one of the following: All Test Selections in Table 10 except NON-WHOLE BLOOD*
- Component field 5 (Parameter Set selection): Per Universal Test ID paragraph 3.3.3.3.4
- Component field 6 (Limit Set selection): null

The Specimen Type shall be the ASCII string "QC," the Specimen Subtype shall be the ASCII string "Commercial," and component fields 4, 5, and 6 of the Universal Test ID field shall be as specified below for runs with an instrument Specimen Type/ Subtype of QC/Commercial:

- Component field 4 (Test Selection), one of the following: "CBC+NOC," “RETIC"
- Component field 5 (Parameter Set selection): Per Universal Test ID paragraph 3.3.3.3.4
- Component field 6 (Limit Set selection): null

The Specimen Type shall be the ASCII string "QC," the Specimen Subtype shall be the ASCII string "Background CBC," and component fields 4, 5, and 6 of the Universal Test ID field shall be as specified below for runs with an instrument Specimen Type/Subtype of QC/Background CBC:

- Component field 4: “CBC+NOC”
- Component field 5: null
- Component field 6: null

The Specimen Type shall be the ASCII string "QC," the Specimen Subtype shall be the ASCII string "Background Retic", and component fields 4, 5, and 6 of the Universal Test ID field shall be as specified below for runs with an instrument Specimen Type/Subtype of QC/Background Retic:

- Component field 4: “RETIC”
- Component field 5: null
- Component field 6: null

The Specimen Type shall be the ASCII string "Autobackground," the Specimen Subtype shall be null, and component fields 4, 5, and 6 of the Universal Test ID field shall be as specified below for runs with an instrument Specimen Type of Autobackground:

- Component field 4: “CBC+NOC”
- Component field 5: null
- Component field 6: null

* In Development

The Specimen Type shall be the ASCII string "Calibrator Whole Blood," the Specimen Subtype shall be null, and component fields 4, 5, and 6 of the Universal Test ID field shall be as specified below for runs with an instrument Specimen Type of Calibrator Whole Blood:

- Component field 4: "CBC+NOC"
- Component field 5: null
- Component field 6: null

The Specimen Type shall be the ASCII string "Calibrator Commercial," the Specimen Subtype shall be null, and component fields 4, 5, and 6 of the Universal Test ID field shall be as specified below for runs with an instrument Specimen Type of Calibrator Commercial:

- Component field 4: "CBC+NOC"
- Component field 5: null
- Component field 6: null

The Specimen Type field in the Test Results record shall be the ASCII character string "Standard Reference Particle," the Specimen Subtype shall be null, and component fields 4, 5, and 6 of the Universal Test ID shall be as specified below for runs with an instrument Specimen Type of Standard Reference Particle:

- Component field 4: "CBC"
- Component field 5: null
- Component field 6: null

3.4.2.3.8 Report Types

The Report Type of the Test Order record shall be the single character F.

3.4.2.4 Results Record(s)

Result Records shall conform to LIS2 section 10.0, with field usage as shown in Table 15. Result Records are used to transfer information about numerical test results and flags. Each numerical result is reported in USA units of measure (see exception below for RBC in results from NON-WHOLE BLOOD* test selections). Each numerical test result and flag is reported in a separate Results Record, distinguished by the Result Value component of the Universal Test ID field. The host computer should map test results and flags based on the Result Value.

Results from NON-WHOLE BLOOD Test Selection*

-The RBC numerical result is reported in the following units: cells/µL

* In Development

Table 15 — Results Record Field Usage for Test Results Transfer Message

LIS2 ref	Item	Req.	LIS2/CELL-DYN Ruby Content
10.1.1	Record Type	x	R
10.1.2	RR Seq Num	x	1, 2, 3...
10.1.3	Universal Test ID	x	Test Selection, Parameter Set Selection, Limit Set Selection, Result Label
10.1.4	Data or Meas. Value	x	Result Value
10.1.9	Result Status	x (see note below)	Result Status
10.1.11	Operator ID		Operator ID
10.1.12	Date/Time Test Started	x	Aspiration Date & Time
10.1.14	Instrument Identification	x	Serial # of instrument that generated the result
Note: Result Status is required only for numerical Test Result Records, not Flag Result Records			

3.4.2.4.1 Record Type

The Record Type of a Results Record shall be the single character R in either upper or lower case.

3.4.2.4.2 RR Sequence Number

The Result Record Sequence Number shall be in the range 1 to n, inclusive, based on the Test Selection and the number of flags, and shall be incremented by 1 for each successive record.

3.4.2.4.3 Universal Test ID

The Universal Test ID shall conform to the component definitions shown in Table 9 of this document. The Universal Test ID shall be present in a Results Record.

The Test Selection shall be present in a Results Record and shall be an exact copy of the Test Selection in the associated Test Order Record.

The Parameter Set Selection of a Results Record shall be null.

The Limit Set Selection of a Results Record shall be null.

The Result Label component of the Universal Test ID shall be:

- one of the 2- 4 character case sensitive ASCII Numerical Result Labels shown below (Table 16) for numerical results. Table 16 also specifies the set of numerical results based on the Test Selection and Specimen Type.
- one of the case sensitive ASCII Flag Labels shown below (Tables 17- 20) for flags

Table 16 — Numerical Result (NR) Labels and Results for Test Selections and Specimen Types

NR Labels	Pat/ CBC	Pat/ RETIC	Auto-Bkgrnd	QC/ CBC	QC/ RETIC	QC/ CBC Bkgrnd	QC/ RETIC Bkgrnd	Calib	SRP	Non-Whole Blood*
WBC	x		x	x		x		x	x	x
WOC*	x***		x	x***		x		x		x
NOC*	x***		x	x***		x		x		x
NEU	x			x						x
LYM	x			x						x
MONO	x			x						x
EOS	x			x						x
BASO	x			x						x
%N	x			x						x
%L	x			x						x
%M	x			x						x
%E	x			x						x
%B	x			x						x
RBC	x		x	x		x		x	x	x
HGB	x		x	x		x		x	x	
MCV	x			x				x		x
MCH	x			x						
MCHC	x			x						
HCT	x			x						
RDW	x			x				x****		
PLT	x		x	x		x		x	x	x
MPV	x			x				x		
PDW*	x			x						
PCT*	x			x						
%R		x			x					
RETC		x **								
RBGD							x			

*Clinical significance has not been established for these parameters. Not reportable in the U.S. Results for the Non-Whole Blood Test Selection are not reportable in the U.S.
**RET C may or may not be present.
***Result is present if Test Selection specifies a NOC.
****RDW is present if result is from a Quick Precision run.

Table 17 — Flag Labels for Suspect Parameter Flags

Labels for Suspect Parameter Flags
WBC
DFLT(N)
DFLT(L)
DFLT(M)
DFLT(E)
DFLT(B)
MCHC
LRI
URI
Fragile RBC
Too Few Events
ERL

Table 18 — Flag Labels for Suspect Population Flags

Labels for Suspect Population Flags
BAND
IG
BLAST
VAR LYM
FWBC
NWBC
NRBC
RRBC
RBC MORPH
ATYP DEP

Table 19 — Labels for System Fault Flags

Labels for System Faults
SAMPLING ERROR
WOC FLOW ERROR
NOC FLOW ERROR
RBC FLOW ERROR
RETC FLOW ERROR
WOC HEATER ERROR
HGB HEATER ERROR

Table 20 — Labels for Informational Messages

Labels for Informational Messages
(WOC)
(NOC)

3.4.2.4.4 Data or Measurement Value

The Data or Measurement Value field for a Test Result shall be one of the 4 or 5 character ASCII strings shown in Table 21, depending on a Ruby configuration setting (5 character ASCII strings shall be the default setting). For numerical results, the actual string shall depend upon the actual quantitative Numerical Result determined by the CELL-DYN Ruby instrument.

Table 21 — Data or Measurement Value Representations for Test Results

Tie	**Data or Meas. Value String** (Example below is for 5 character ASCII string)	Description	Interpretation
1	-----	4-5 dashes	Data was recorded, but result was not calculated
2	>>>>>	4 - 5 greater than symbols	Result is higher than instrument's reportable range
3	<<<<<	4 - 5 less than symbols	HGB result when WBC result is greater than the WBC linear range
4	0.000 - 9999.	4 - 5 character string of decimal digits with floating decimal point	Actual numerical/ quantitative result; decimal point floats with magnitude
N/A	FLAG	4-character string	Result is a flag

3.4.2.4.5 Result Status

The Result Status shall be a single character as indicated in Table 22. Result Status is tied to the Data or Measurement Value Representations as indicated in the "Tie/Tied to" column of Tables 21 and 22. The Result Status for a flag result is null.

Table 22 — Result Status for Numerical Test Results

Result Status	Interpretation	Tied to
F	Final; quantitative result is reportable although it may be associated with a suspect population flag	4
W	Warning; quantitative result is suspect (there is an associated suspect parameter flag or system fault) or out of range; should be reviewed per lab protocol before being reported	2,3,4
X	The quantitative result was not computed	1

3.4.2.4.6 Operator ID

If present, the Operator ID shall be a 1 - 6 character ASCII string (32-255, inclusive). It is mapped from the Operator ID field of the CELL-DYN Ruby. This field may be null.

3.4.2.4.7 Date/Time Test Started

The Date/Time Test Started shall be a 14-character date/time string, YYYYMMDDHHMMSS, as per LIS2 paragraph 6.6.2. The value shall be the date and time of aspiration on the CELL-DYN Ruby instrument of the specimen which produced these results.

3.4.2.4.8 Instrument Identification

Instrument Identification (Analyzer Serial Number) shall be the same as the Analyzer Serial Number specified in Sender ID paragraph 3.3.2.1.1.

3.4.2.4.9 Example of a Numerical Test Results Record

An example of a conforming numerical test results record is as follows:
R|1|^^^CBC^^^MCHC|33.21|||||F|||19960810082234||40009603A<CR>

3.4.2.4.10 Example of a Flag Results Record

An example of a conforming flag results record is as follows:

R|23|^^^CBC^^^RBC MORPH|FLAG|||||||19960810082234||40009603A<CR>

3.4.2.5 Histogram Data Records

Histogram Data Records are structured as Manufacturer Information Records (MIR) and shall conform to LIS2 section 15.0, with field usage as shown in Table 23. Each Histogram Data Record contains information about a single histogram.

The set of Histogram Data Records in a Test Results Message depends on the Test Selection associated with the results, as noted below (graph description shown in parentheses):

CBC:

- WB1 (N-L-M)
- WB2 (M-P)
- RBC (RBC Volume)
- PLT (PLT)

CBC+NOC, CBC+RRBC:

- WB1
- WB2
- RBC
- PLT
- NOC (NOC)

RETIC:

- RT1 (RETC 90/0 Angle)
- RT2 (RETC 90/10 Angle)

NON-WHOLE BLOOD:*

- None

Table 23 — Histogram Data Record Field Usage for Test Results Transfer Message

LIS2 ref	Item	Content
15.1.1	Record Type	M
15.1.2	MIR Sequence Number	1, 2, 3...
N/A	Data Type	H
N/A	Graph Type	One of the following: WB1, WB2, RBC, PLT, NOC, RT1, RT2
N/A	Measurement Labels	Null
N/A	Discriminants	See below.
N/A	Cell Data	See below.

3.4.2.5.1 Record Type

The Record Type of a Histogram Data Record shall be the single character M in either upper or lower case.

3.4.2.5.2 MIR Sequence Number

The MIR Sequence Number shall be in the range 1 to n, inclusive, based on the Test Selection, and shall be incremented by 1 for each successive record (histogram or scattergram).

3.4.2.5.3 Data Type

The Data Type of a Histogram Data Record shall be the single character H in either upper or lower case.

3.4.2.5.4 Graph Type

The Graph Type of a Histogram Data Record shall be one of the following strings in either upper or lower case: WB1, WB2, RBC, PLT, NOC, RT1, RT2

3.4.2.5.5 Measurement Labels

The Measurement Labels field of a Histogram Data Record shall be null.

* In Development

3.4.2.5.6 Discriminants

The Discriminants field of a Histogram Data Record shall contain from 0 to n components, each of which is an ASCII string from 1 to 3 characters in length representing integers from 0 - 255. Multiple components are separated by the repeat delimiter. Each component is to be interpreted as the X-axis value of a vertical line drawn from the X-axis to the top of the graph frame.

3.4.2.5.7 Cell Data

The Cell Data field of a Histogram Data Record shall contain a set of characters created by converting up to 256 16-bit values (512 bytes) to Base64. The Cell Data is Base64 encoded to avoid both transmission control characters and LIS2 default delimiters. Each of the 16-bit values represents the Y-value of a point on a histogram with an associated X-axis value from 0 - 255. After decoding, any odd byte at the end of the set is to be ignored prior to combining the bytes back into 16-bit values (first and odd numbered bytes are the low byte, second and even numbered bytes are the high byte).

3.4.2.5.8 Example of a Histogram Data Record

An example of a conforming Histogram Data record is as follows:

M|1|H|WB1||40\120|asfvOkIhF...<CR>

3.4.2.6 Scattergram Data Records

Scattergram Data Records are structured as Manufacturer Information Records (MIR) and shall conform to LIS2 section 15.0, with field usage as shown in Table 24. Each Scattergram Data Record contains information about a set of scattergrams.

The set of Scattergram Data Records in a Test Results Message depends on the Test Selection associated with the results, as noted below (data channels shown in parentheses):

CBC, CBC+NOC, CBC+RRBC:

- WBC (0, 10, 90, 90D)
- RBC/PLT (0, 10, 90) RETIC:
- RTC (0, 10, 90, 90D)

Note: At this time, there are no instrument graphs that make use of the RTC 90D data.

NON-WHOLE BLOOD:*

- None

* In Development

Table 24 — Scattergram Data Record Field Usage for Test Results Transfer Message

LIS2 ref	Item	Content
15.1.1	Record Type	M
15.1.2	MIR Sequence Number	1, 2, 3...
N/A	Data Type	S
N/A	Graph Type	One of the following: WBC, RBC/PLT, RTC
N/A	Measurement Labels	See below.
N/A	Discriminants	See below.
N/A	Cell Data	See below.

3.4.2.6.1 Record Type

The Record Type of a Scattergram Data Record shall be the single character M in either upper or lower case.

3.4.2.6.2 MIR Sequence Number

The MIR Sequence Number shall be in the range 1 to n, inclusive, based on the Test Selection, and shall be incremented by 1 for each successive record (histogram or scattergram).

3.4.2.6.3 Data Type

The Data Type of a Scattergram Data Record shall be the single character S in either upper or lower case.

3.4.2.6.4 Graph Type

The Graph Type of a Scattergram Data Record shall be one of the following strings in either upper or lower case: WBC, RBC/PLT, RTC

3.4.2.6.5 Measurement Labels

The Measurement Labels field of a Scattergram Data Record depends on the content of the Graph Type field, as noted below:

- When Graph Type is *WBC* or *RTC,* Measurement Labels are: *0, 10, 90, 90D, CT*
- When Graph Type is *RBC/PLT,* Measurement Labels are: *0, 10, 90, CT*

3.4.2.6.6 Discriminants

The Discriminants field of a Scattergram Data Record shall contain from 0 to n elements each containing multiple components described below. Multiple elements are separated by the repeat delimiter. The X and Y axis Measurement Label components are used to identify on which scattergram the discriminant is to be drawn. For example, if the X and Y axis Measurement Label components are 0 and 10, then the discriminant is only drawn on a scattergram created by plotting the 0 degree Cell Data (X-axis) against the 10 degree Cell Data (Y-axis).

Components of a Discriminant Element:

- X-axis Measurement Label: One of the Measurement Labels in Measurement Labels paragraph 3.4.2.6.5
- Y-axis Measurement Label: One of the Measurement Labels in Measurement Labels paragraph 3.4.2.6.5
- 2 - n X-Y point descriptors: X, Y are ASCII strings from 1 to 3 characters in length representing integers from 0 - 255. Each point descriptor is to be interpreted as one of a series of points which are connected with a straight line to create a discriminant (first point to second point, second point to third point, etc. The last point is not connected back to the first point.

3.4.2.6.7 Cell Data

The Cell Data field of a Scattergram Data Record depends on the content of the Graph Type field, as noted below:

- When Graph Type is *WBC* or *RTC,* Cell Data contains up to 2000 five-byte vectors: *0, 10, 90, 90D, CT*
- When Graph Type is *RBC/PLT,* Cell Data contains up to 2000 four byte vectors: *0, 10, 90, CT*

The vector content is as follows:

0: 0 - 255 data value for the 0 degree channel

10: 0 - 255 data value for the 10 degree channel

90: 0 - 255 data value for the 90 degree channel

90D: 0 - 255 data value for the 90 degree depolarized channel
CT: 0 - 255 data value for the cell type

The Cell Data field shall contain n characters created by converting up to 2000 four (up to 8000 bytes) or five (up to 10000 bytes) byte vectors to Base64. The Cell Data is Base64 encoded to avoid both transmission control characters and LIS2 default delimiters. After decoding, any partial vector at the end is ignored.

The cell type vector byte (*CT*) is mapped to the cell type as noted below: The RGB color designations used on the instrument displays and printouts are also indicated for each cell type.

Cell Type (CT)	**Display Color (R, G, B)**	**Print Color (R, G, B)**
UNKNOWN = 0	160, 160, 160)	160, 160, 160)
LYMPHOCYTE = 1	(0, 255, 255)	(0, 204, 206)
MONOCYTE = 2	(250, 0, 250)	(250, 0, 250)
BASOPHIL = 3	(255, 255, 255)	(0, 0, 0)
NEUTROPHIL = 4	(255, 255, 0)	(250, 146, 6)
EOSINOPHIL = 5	(0, 255, 0)	(0, 220, 21)
MONONUCLEAR = 6	(160, 160, 160)	(160, 160, 160)
STROMA = 7	(250, 175, 160)	(250, 175, 160)
NOISE2 = 8	(250, 175, 160)	(250, 175, 160)
RBC_CELL = 9	(255, 80, 80)	(255, 0, 0)
PLT_CELL = 10	(255, 255, 0)	(250, 146, 6)
WBC_CELL = 11	(0, 255, 255)	(0, 204, 206)
COIN_CELL = 12	(250, 0, 250)	(250, 0, 250)
AGG_CELL = 13	(255, 255, 255)	(0, 0, 0)
RTC_RBC_CELL = 14	(255, 80, 80)	(255, 0, 0)
RTC_PLT_CELL = 15	(255, 255, 0)	(250, 146, 6)
RETIC_CELL = 16	(102, 102, 255)	(0, 0, 255)
IRF_CELL = 17	(0, 255, 255)	(0, 204, 206)
BORDER_CELL = 18	(255, 255, 255)	(0, 0, 0)
IRF_COINC_CELL = 19	(160, 160, 160)	(160, 160, 160)
RETIC_COINC_CELL = 20	(160, 160, 160)	(160, 160, 160)
RBC_COINC_CELL = 21	(160, 160, 160)	(160, 160, 160)

3.4.2.6.8 Example of a Scattergram Data Record

An example of a conforming Scattergram Data record is as follows:

M|1|S|WBC|0^10^90^90D^CT|90^10^0^0^63^151\90^90D^13^0^13^255\0^10^0^17^18^17^

68^37^158^37|a14J0sfvOkIh F...<CR>

3.4.2.7 Message Terminator Record (MTR)

The Message Terminator Record of a Test Results Transfer message shall be as described earlier for the Orders List Entry Request message.

3.4.3 Test Results Acknowledgement Message (issued by the host computer, received by CELL- DYN Ruby)

The host computer must issue LIS1-level ACK's for each frame of the Test Results Transfer Message which it properly receives and parses. If each and every frame is ACK'd, the CELL-DYN Ruby shall interpret the outcome as a successful transfer of the Test Results Message to the host computer.

4.0 ERROR HANDLING

The CELL-DYN Ruby Instrument shall truncate optional fields which exceed the specified field lengths (e.g., physician names longer than 22 characters); and shall ignore the contents of fields not specified in this document.

The CELL-DYN Ruby Instrument shall provide values for Orders List entry data items associated with optional fields in an accepted Orders List Entry Submission message when such fields are blank, and shall reject Orders List Entry Submission messages when optional fields are present but not interpretable (includes being out of range).

5.0 APPENDIX A - EXAMPLES

5.1 Example of the CELL-DYN Ruby sending Orders List Entry Request to the host computer (Request Record)

The following message, issued by the CELL-DYN Ruby, would request that the host computer send all pending Orders List Entries for the CELL-DYN Ruby System identified as having Analyzer Serial Number 40009603A:

H|\^&|||40009603A^CDRuby^2.0ML^1.0|||||||P|LIS2-A<CR>

Q|1|^ALL<CR>

L|1|N<CR>

5.2 Examples of Orders List Entry Submission Messages (sent by the host computer to CELL-DYN Ruby in response to query)

Ex 1 - the minimum Orders List Entry Submission Message (1 Orders List Entry, required elements only):

H|\^&<CR>

P|1<CR>

O|1|18465<CR>

L|1|N<CR>

This message will cause the CELL-DYN Ruby to put 1 Orders List Entry in the Orders List for a human patient identified by Specimen ID = 18465. The Test Selection, Parameter Set and Limit Set will be provided by the CELL-DYN Ruby. The Orders List Entry will be activated (selected and run) when a specimen tube labeled with barcode = 18465 is presented to the aspiration probe by the Autoloader.

Ex 2 - a Orders List Entry Submission using all required and optional elements of significance to the CELL-DYN Ruby (records terminated after last used field):

H|\^&|||||||||P|LIS2-A<CR>

P|1||A56342||David Jones||19640315|M|||||Roberts||ICU^327-56-9567<CR>

O|1|19345||^^^CBC^4^0|||19960810153028|||||||Patient^Human<CR>

L|1|N<CR>

This message will cause the CELL-DYN Ruby to put 1 Orders List Entry in the Orders List for a human patient identified by Specimen ID = 19345 and Patient ID = A56342. The test to be run against this specimen will be a CBC test. The chartable report will use Parameter Set #4, and the results will be evaluated against a Limit Set based on the age and sex of the patient. The demographic information recorded on the CELL-DYN Ruby about this specimen includes the Patient Name (David Jones), Birthdate (Mar 15, 1964), Sex (Male). The Attending Physician is Dr. Roberts. Assuming the Laboratory and host computer have agreed upon definitions of the “user defined demographic fields” of the CELL-DYN Ruby to be: 1st = current patient location,

and 2nd = patient Social Security Number; the additional information recorded in the CELL-DYN Ruby demographics area will be: patient location = ICU; patient Social Security # = 327-56-9567. The Orders List Entry will be activated (selected and run) when a specimen tube labeled with barcode = 19345 is presented to the aspiration probe by the Autoloader.

Ex.3 Orders List Entry Submission using multiple Orders List Entries, packaged together in response to the CELL- DYN Ruby's query:

H|\^&<CR>

P|1||A56342||David Jones||19640315|M|||||Roberts||ICU^327-56-9567<CR>

O|1|19345||^^^CBC^4^0|||19960810153028|||||||Patient^Human<CR>

P|2<CR>

O|1|18465<CR>

L|1|N<CR>

The message in Ex. 3 will cause two Orders List Entries to be put in the Orders List:

The first entry is for the human patient, David Jones, who is identified by Specimen ID = 19345. See Example 2. The Orders List entry will be activated (selected and run) when a specimen tube with barcode = 19345 is presented to the aspiration probe by the Autoloader.

The second entry is for the human patient identified by Specimen ID = 18465. See Example 1. The Orders List Entry will be activated (selected and run) when a specimen tube labeled with barcode = 18465 is presented to the aspiration probe by the Autoloader.

5.3 Examples of a Orders List Entry Rejection message sent by the CELL-DYN Ruby to the host computer

This example consists of the single Orders List Entry Rejection message sent in response to the Patient Order combinations of Ex. 3. In this example, the first Patient-Order combination is rejected due to a Orders List Full condition; the second is accepted. Note: The multitasking nature of the CELL-DYN Ruby Instrument software permits this type of behavior:

H|\^&|||40009603A^CDRuby^2.0ML^1.0|||||||P|LIS2-A<CR>

P|1||A56342||David Jones||19640315|M|||||Roberts||ICU^327-56-9567<CR>

O|1|19345||^^^CBC^4^0|||19960810153028|||||||Patient^Human|||||||||X<CR>

C|1|I|Orders List Full^None|G<CR>

L|1|E<CR>

<no message in response to second (correctly received) Patient-Order combination>

5.4 Examples of Test Result Transfer Messages (sent by CELL-DYN Ruby to the host computer)

The minimum message associated with a CBC Test Selection, with the CELL-DYN Ruby configured to only send numerical results (assumes anything that can be blank or null on the CELL-DYN Ruby is nulled in this transmission):

H|\^&|||400096030^CDRuby^2.0ML^1.0|||||||P|LIS2-A<CR>

P|1<CR>

O|1|18465^23^09|0098|^^^CBC^2^1||||||||||Patient|||||||||F<CR>

R|1|^^^CBC^^^WBC|72.51|||||F|||19960810082334||400096030<CR>

R|2|^^^CBC^^^NEU|48.92|||||F|||19960810082234||400096030<CR>

R|3|^^^CBC^^^LYM|14.73|||||F|||19960810082234||400096030<CR>

R|4|^^^CBC^^^MONO|8.923|||||F|||19960810082234||400096030<CR>

R|5|^^^CBC^^^EOS|0.000|||||F|||19960810082234||400096030<CR>
R|6|^^^CBC^^^BASO|0.000|||||F|||19960810082234||400096030<CR>
R|7|^^^CBC^^^%N|67.45|||||F|||19960810082234||400096030<CR>
R|8|^^^CBC^^^%L|20.38|||||F|||19960810082234||400096030<CR>
R|9|^^^CBC^^^%M|12.37|||||F|||19960810082234||400096030<CR>
R|10|^^^CBC^^^%E|0.013|||||F|||19960810082234||400096030<CR>
R|11|^^^CBC^^^%B|0.000|||||F|||19960810082234||400096030<CR>
R|12|^^^CBC^^^RBC|5.832|||||F|||19960810082234||400096030<CR>
R|13|^^^CBC^^^HGB|16.14|||||F|||19960810082234||400096030<CR>
R|14|^^^CBC^^^MCV|83.42|||||F|||19960810082234||400096030<CR>
R|15|^^^CBC^^^MCH|27.77|||||F|||19960810082234||400096030<CR>
R|16|^^^CBC^^^MCHC|33.23|||||F|||19960810082234||400096030<CR>
R|17|^^^CBC^^^HCT|48.61|||||F|||19960810082234||400096030<CR>
R|18|^^^CBC^^^RDW|11.64|||||F|||19960810082234||400096030<CR>
R|19|^^^CBC^^^PLT|>>>>>|||||W|||19960810082234||400096030<CR>
R|20|^^^CBC^^^MPV|5.682|||||W|||19960810082234||400096030<CR>
R|21|^^^CBC^^^PDW|-----|||||X|||19960810082234||400096030<CR>
R|22|^^^CBC^^^PCT|-----||||X|||19960810082234||400096030<CR>
L|1|N<CR>

A message for results from a CBC+NOC Test Selection that had flags, with the CELL-DYN Ruby configured to send histogram results:

H|\^&|||400096030^CDRuby^2.0ML^1.0|||||||P|LIS2-A<CR>
P|1||A56342||David Jones||19640315|M|||||Roberts||ICU^327-56-9567<CR>
O|1|18465^23^09|0098|^^^CBC+NOC^2^1|||19960801073254|||||||Patient|||||||||F<CR>
R|1|^^^CBC+NOC^^^WBC|72.51|||||F||jbws|19960810082334||400096030<CR>
R|2|^^^CBC+NOC^^^%N|67.14|||||F||jbws|19960810082234||400096030<CR>
R|3|^^^CBC+NOC^^^%L|20.33|||||F||jbws|19960810082234||400096030<CR>
R|4|^^^CBC+NOC^^^%M|12.38|||||F||jbws|19960810082234||400096030<CR>
R|5|^^^CBC+NOC^^^%E|0.019|||||F||jbws|19960810082234||400096030<CR>
R|6|^^^CBC+NOC^^^%B|0.000|||||F||jbws|19960810082234||400096030<CR>
R|7|^^^CBC+NOC^^^NEU|48.97|||||F||jbws|19960810082234||400096030<CR>
R|8|^^^CBC+NOC^^^LYM|14.75|||||F||jbws|19960810082234||400096030<CR>
R|9|^^^CBC+NOC^^^MONO|8.923|||||F||jbws|19960810082234||400096030<CR>
R|10|^^^CBC+NOC^^^EOS|0.000|||||F||jbws|19960810082234||400096030<CR>
R|11|^^^CBC+NOC^^^BASO|0.000|||||F||jbws|19960810082234||400096030<CR>
R|12|^^^CBC+NOC^^^RBC|5.832|||||F||jbws|19960810082234||400096030<CR>
R|13|^^^CBC+NOC^^^HGB|16.13|||||F||jbws|19960810082234||400096030<CR>
R|14|^^^CBC+NOC^^^MCV|83.14|||||F||jbws|19960810082234||400096030<CR>
R|15|^^^CBC+NOC^^^MCH|27.72|||||F||jbws|19960810082234||400096030<CR>
R|16|^^^CBC+NOC^^^MCHC|33.42|||||F||jbws|19960810082234||400096030<CR>
R|17|^^^CBC+NOC^^^HCT|48.46|||||F||jbws|19960810082234||400096030<CR>
R|18|^^^CBC+NOC^^^RDW|11.65|||||F||jbws|19960810082234||400096030<CR>

R|19|^^^CBC+NOC^^^PLT|>>>>>|||||W||jbws|19960810082234||400096030<CR>

R|20|^^^CBC+NOC^^^MPV|5.872|||||W||jbws|19960810082234||400096030<CR>

R|21|^^^CBC+NOC^^^PDW|-----|||||X||jbws|19960810082234||400096030<CR>

R|22|^^^CBC+NOC^^^PCT|-----|||||X||jbws|19960810082234||400096030<CR>

R|23|^^^CBC+NOC^^^NOC|72.51|||||F||jbws|19960810082234||400096030<CR>

R|24|^^^CBC+NOC^^^WOC|71.43|||||F||jbws|19960810082334||400096030<CR>

R|25|^^^CBC+NOC^^^RRBC|FLAG|||||||jbws|19960810082334||400096030<CR>

M|1|H|WB1||40\120|asfvOkIhF...<CR>

M|2|H|RBC||10\170\30|bhLaZiw4v...<CR>

M|3|H|WB2|||dfrtGvB789sfhj...<CR>

M|4|H|NOC||50|kv84GeHY4...<CR>

M|5|H|PLT||0\5|SG0jU54Dk...<CR>

L|1|N<CR>

A message for results from a RETIC Test Selection that had flags, with the CELL-DYN Ruby configured to send scattergram results. The operator provided a value for RBCr, the RBC value used in the calculation of the RETC result.

H|\^&|||400096030^CDRuby^2.0ML^1.0|||||||P|LIS2-A<CR>

P|1||A56342||Frank James||19640315|M|||||Roberts||ICU^317-90-9163<CR>

O|1|18465^23^09|0098|^^^RETIC^2^1|||19960801073254|||||||Patient|||||||||F<CR>

R|1|^^^RETIC^^^RETC|44.45|||||F||jbws|19960810082334||400096030<CR>

R|2|^^^RETIC^^^%R|1.312|||||F||jbws|19960810082234||400096030<CR>

R|3|^^^RETC^^^ERL|FLAG|||||||jbws|19960810082334||400096030<CR>

M|1|S|RTC|0^10^90^90D^CT|90^10^0^0^63^151\90^90D^13^0^13^255\0^10^0^17^18^17^68^37^158^37|a14J0 sfvjelNY...<CR>

L|1|N<CR>

A message for results from an autobackground run, with the CELL-DYN Ruby configured to not send graphical results.

H|\^&|||400096030^CDRuby^2.0ML^1.0|||||||P|LIS2-A<CR>

P|1<CR>

O|1|^23^09|0098|^^^CBC+NOC^^|||19960801073254|||||||Autobackground|||||||||F<CR>

R|1|^^^CBC+NOC^^^WOC|0.000|||||F||jbws|19960810082334||400096030<CR>

R|2|^^^CBC+NOC^^^WBC|0.000|||||F||jbws|19960810082234||400096030<CR>

R|3|^^^CBC+NOC^^^NOC|0.000|||||F||jbws|19960810082234||400096030<CR>

R|4|^^^CBC+NOC^^^PLT|0.103|||||F||jbws|19960810082334||400096030<CR>

R|5|^^^CBC+NOC^^^HGB|0.059|||||F||jbws|19960810082234||400096030<CR>

R|6|^^^CBC+NOC^^^RBC|0.022|||||F||jbws|19960810082234||400096030<CR>

L|1|N<CR>

6.0 APPENDIX B – BASE 64 ALPHABET AND ENCODING/DECODING DESCRIPTION

6.1 Base64 Alphabet

Value Encoding	Value Encoding	Value Encoding	Value Encoding
0 A	17 R	34 i	51z
1 B	18 S	35 j	52 0
2 C	19 T	36 k	53 1
3 D	20 U	37 l	54 2
4 E	21 V	38 m	55 3
5 F	22 W	39 n	56 4
6 G	23 X	40 o	57 5
7 H	24 Y	41 p	58 6
8 I	25 Z	42 q	59 7
9 J	26 a	43 r	60 8
10 K	27 b	44 s	61 9
11 L	28 c	45 t	62 +
12 M	29 d	46 u	63 /
13 N	30 e	47	v
14 O	31 f	48 w	(pad) =
15 P	32 g	49	x
16 Q	33 h	50	y

6.2 Base64 Encoding/Decoding Description

The instrument uses base64 encoding that complies with RFC 2045, without the option for 76 character lines and with padding. The encoding process represents 24-bit groups of input bits as output strings of four encoded characters. Proceeding from left to right, a 24-bit input group is formed by concatenating three 8-bit input groups.These 24 bits are then treated as four concatenated 6-bit groups, each of which is translated into a single digit in the base64 alphabet. When encoding a bit stream via the base64 encoding, the bit stream must be presumed to be ordered with the most- significant-bit first. That is, the first bit in the stream will be the high-order bit in the first 8-bit byte, and the eighth bit will be the low-order bit in the first 8-bit byte, and so on. Each 6-bit group is used as an index into an array of 64 printable characters. The character referenced by the index is placed in the output string. These characters, identified in the base64 alphabet above, are selected so as to be universally representable.

All line breaks or other characters not found in the base64 alphabet must be ignored by the decoding software. In base64 data, characters other than those in the base64 alphabet, line breaks, and other white space probably indicate a transmission error, about which a warning message or even a message rejection might be appropriate under some circumstances. Special processing is performed if fewer than 24 bits are available at the end of the data being encoded. A full encoding quantum is always completed at the end of a body. When fewer than 24 input bits are available in an input group, zero bits are added (on the right) to form an integral number of 6-bit groups. Padding at the end of the data is performed using the "=" character.

Since all base64 input is an integral number of octets, only the following cases can arise: (1) the final quantum of encoding input is an integral multiple of 24 bits; here, the final unit of

encoded output will be an integral multiple of 4 characters with no "=" padding, (2) the final quantum of encoding input is exactly 8 bits; here, the final unit of encoded output will be two characters followed by two "=" padding characters, or (3) the final quantum of encoding input is exactly 16 bits; here, the final unit of encoded output will be three characters followed by one "=" padding character. Because it is used only for padding at the end of the data, the occurrence of any "=" characters may be taken as evidence that the end of the data has been reached (without truncation in transit). No such assurance is possible, however, when the number of octets transmitted was a multiple of three and no "=" characters are present.

7.0 APPENDIX C – BAUD RATE CONSIDERATIONS WHEN TRANSMITTING GRAPHICAL INFORMATION IN A HOST QUERY ENVIRONMENT

In order to minimize the potential for host query timeouts, the transmission baud rate and host query timeout period should be set based on the types of information being transmitted. When a host query timeout occurs, the CELL-DYN Ruby's default test selection is used rather than the test selection in the host computer order that did not arrive in time to be used. The next table contains estimates for how long it takes to transmit results based on the baud rate and the types of information in the results.

Table 25 — Results Transmission Times (in seconds)

Record Types	Bytes	4800 Baud	9600 Baud	19,200 Baud	38,400 Baud	56,000 Baud	115,200 Baud
CBC Numerical Results	3760	7.83	3.92	1.96	0.98	0.67	0.33
CBC Numerical Results and Scatter Graphs	7180	14.96	7.48	3.74	1.87	1.28	0.62
CBC Numerical Results and Scatter Graphs	27764	57.84	28.92	14.46	7.23	4.96	2.41
CBC Numerical Results and Histograms and Scatter Graphs	31184	64.97	32.48	16.24	8.12	5.57	2.71
Retic Numerical Results	512	1.07	0.53	0.27	0.13	0.09	0.04
Retic Numerical Results and Histograms	1880	3.92	1.96	0.98	0.49	0.34	0.16
Retic Numerical Results and Scatter Graphs	13848	28.85	14.43	7.21	3.61	2.47	1.20
Retic Numerical Results and Histograms and Scatter Graphs	15216	31.70	15.85	7.93	3.96	2.72	1.32

The host query timeout period can be set to be a number from 1 to 20 seconds (inclusive). The default value is 5 seconds. If the CELL-DYN Ruby is busy transmitting a result record and a host query message is put on the transmit queue, the query will not be sent until the transmission of the results record is completed. If the results record transmission takes too long, the host query will fail. The actual transmission time may vary from the estimates in the table, since the host computer may ask for a retransmission or may be slow to respond.

The green shaded table entries are those most likely to work with the default host query timeout period of 5 seconds. Setting the host query timeout period to a value greater than 5 seconds will allow operation outside the annotated region. In general, add 2.5 seconds to the value in the table to generate a value for the host query timeout period. For example, if CBC Numerical Results and Histograms are to be transmitted over a 9600 baud connection, set the host query timeout period to 10 seconds (7.48 + 2.5). Note that increasing the host query timeout period will decrease the throughput of the CELL-DYN Ruby.

The next table contains background information on how the results message lengths are calculated. 10bits/byte is used to estimate transmission time over a serial RS232 connection (e.g., 56K baud is approximately 5.6K bytes/sec.) Approximate maximum values are used for CBC and Retic Result Data.

Table 26 — Message Lengths

Record Types	**Raw Bytes**	**Bytes in Transmitted Record (Base64)**
CBC Numerical Results	**3760**	N/A
Retic Numerical Results	**512**	N/A
Histogram (H)	256 * 2 = **512**	(512 / 3 = 171) * 4 = **684**
4 Channel Scatter Graph (S4*θ*)	5*2000 = **10000**	(10000 / 3 = 3334) * 4 = **13336**
3 Channel Scatter Graph (S3*θ*)	4*2000 = **8000**	(8000 / 3 = 2667) * 4 = **10668**
CBC = 5H + 1S4*θ* + 1S3*θ*	5*512 + 1*10000 + 1*8000 = **20560**	5*684 + 1*13336 + 1*10668 = **27424**
RTC = 2H + 1S4*θ*	2*512 * 1*10000 = **12560**	2*684 + 1*13336 = **14704**

8.0 APPENDIX D - Interface Differences by Software Release

8.1 Differences Between the CELL-DYN Ruby Host Interface Version 1.0 ML and the CELL-DYN Ruby Host Interface for Software Version 2.0 ML

Note: Not all differences in the record and message examples are explicitly indicated in the table. Changes are noted as either an interface change or a change only to the documentation.

Paragraph (1.0ML)	CELL-DYN Ruby Host I/F Spec. (1.0 ML)	Paragraph (2.0 ML)	CELL-DYN Ruby Host I/F Spec. (2.0 ML)	Type of Change
3.3.3.2.5	A Birthdate in the future is valid. Note: while the oldest valid YYYY was not stated in the spec, it was the current year minus 150.	3.3.3.2.5	A Birthdate in the future is not valid. The current year minus 150 is the oldest valid YYYY for this field.	Interface (Birthdate in the future); Documentation only (current year minus 150 is oldest valid YYYY)
3.3.3.3.5	A Collection Date in the future is valid. Note: while the oldest valid YYYY was not stated in the spec, it was the current year minus 150.	3.3.3.3.5	A Collection Date in the future is not valid. The current year minus 150 is the oldest valid YYYY for this field.	Interface (Collection Date in the future); Documentation only (current year minus 150 is oldest valid YYYY)
Table 16	-WOC, NOC shown as FDA reportable. -WOC not shown as included in results for Pat/CBC runs when Test Selection specifies a NOC, but is included in results for Pat/CBC runs when Test Selection specifies a NOC. -PCT, PDW not shown as included in results for QC/CBC runs, but are included in results for QC/CBC runs.	Table 16	-WOC, NOC shown as not FDA reportable. -WOC shown as included in results for Pat/CBC runs when Test Selection specifies a NOC, and is included in results for Pat/CBC runs when Test Selection specifies a NOC. -PCT, PDW shown as included in results for QC/CBC runs and are included in results for QC/CBC runs.	Documentation only
3.4.2.4.4	The Data or Measurement Value field for a Test Result shall be one of the 4 character ASCII strings in Table 21. Note: this should have referred to a 5 character ASCII string.	3.4.2.4.4	The Data or Measurement Value field for a Test Result shall be one of the 4 or 5 character ASCII strings shown in Table 21, depending on a Ruby configuration setting (5 character ASCII strings shall be the default setting).	Interface
Table 21	Data or Measurement Value field content is fixed at 5 characters, except when the content is the string FLAG.	Table 21	Data or Measurement Value field content can be either 4 characters or 5 characters, except when the content is the string FLAG.	Interface
3.4.2.4.6	The Operator ID shall be a 4 character ASCII string (32-255, inclusive).	3.4.2.4.6	The Operator ID shall be a 1-6 character ASCII string (32-255, inclusive).	Documentation only
5.4 Example #2	Example of Results Message does not include Results Records for WOC and NOC, and the Test Order record has CBC as the Test Selection.	5.4 Example #2	Example of Results Message does not include Results Records for WOC and NOC, and the Test Order record has CBC as the Test Selection.	Documentation only