Kaleido K2

Protocol specification
Serial to TCP/IP Dispatcher

Document Number 819-03M00-001

Miranda Technologies Inc.
Internal Document

1 Table of contents

2 Table of figures

3 Introduction

The purpose of this document is to present the Serial-to-TCP dispatcher protocol.

The Serial-to-TCP dispatcher is a piece of software designed to run on a separate PC in order to act as a bridge between a Serial device and the TCP/IP network. Figure 1 presents a functional model of the dispatcher. As it is shown here, the Dispatcher allows one Serial device to communicate with many TCP/IP devices (in this case represented by Kaleido equipments).

The Serial-to-TCP Dispatcher protocol must be implemented by the Serial device that wishes to use the Serial-to-TCP Dispatcher.

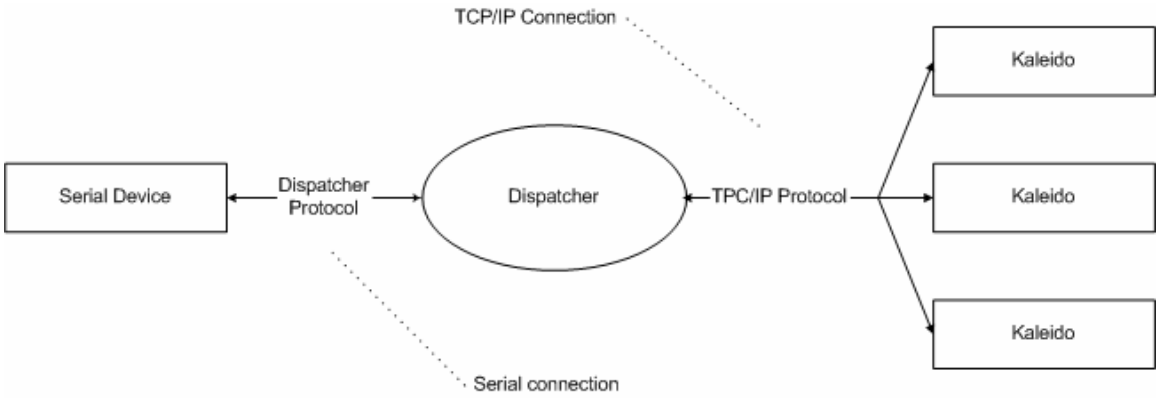

Figure 1 - Functional model

4 Working Assumptions

This section list external conditions on which this feature depends.

WA1 The Dispatcher software is expected to run on a Device that has one Serial port.

WA2 The Dispatcher software is expected to run on a Device that provides TCP/IP stack support over an Ethernet physical connection.

5 Description

5.1 Description of dataflow

FR001 The Dispatcher expects the data coming from the Serial port to respect the format defined in *Figure 4 - Dispatcher Protocol.*

FR002 In normal circumstances (see *Figure 3 – Dispatcher dataflow diagram*), the Dispatcher:

-extracts the *Message* section of the Data packet (shown in *Figure 4 - Dispatcher Protocol*) received on the Serial port,

-sends it to the IP address/TCP port number that matches the Link ID.

FR003 In normal circumstances (see *Figure 3 – Dispatcher dataflow diagram*), the Dispatcher:

-extracts the data contained in a TCP/IP segment it receives from an opened TCP/IP connection;

-encapsulates it to a Serial data packet (see *Figure 4 - Dispatcher Protocol*)

-forwards it onto the Serial connection.

5.2 Interaction diagram

5.2.1 Normal case.

This section presents the normal flow of data from a serial device to multiple TCP/IP devices.

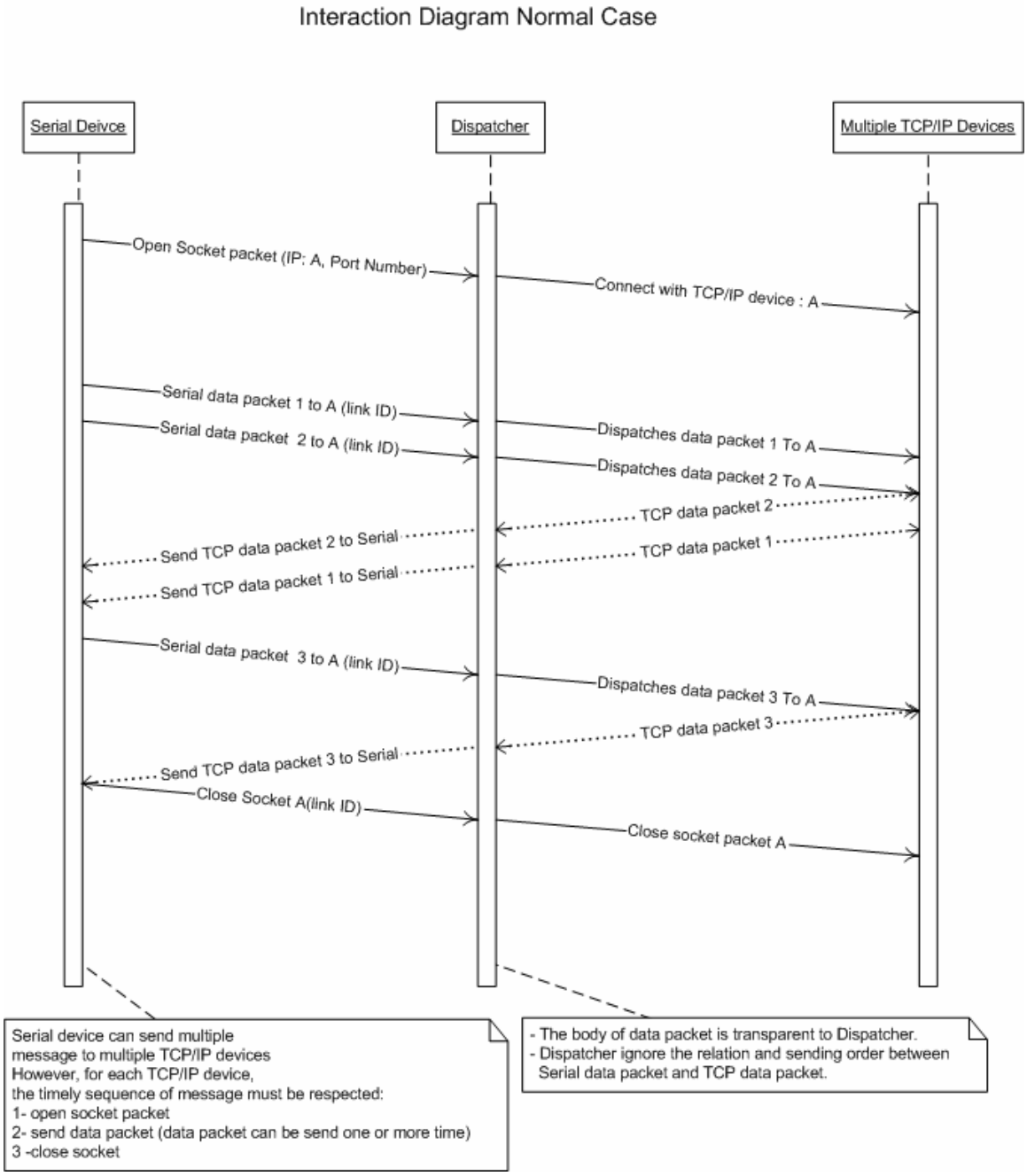

Figure 2 - Interaction diagram

5.2.2 Exception case

The Dispatcher protocol does not return any error message. Invalid packets are discarded silently. The Serial device that uses the Dispatcher must implements its own mechanism to ensure that data is successfully transmitted.

5.3 Dispatcher dataflow diagram

This section presents the Dispatcher dataflow diagram.

Figure 3 – Dispatcher dataflow diagram

6 Interface

This section presents the interface used to manage this new feature.

6.1 Serial Interface

Figure 4 shows the format of the packets received by the Dispatcher over its serial interface (i.e. Open socket packet, Data packet and Close socket packet) as well as the format of the packets sent by the Dispatcher onto its serial interface (i.e. Data packet from TCP/IP).

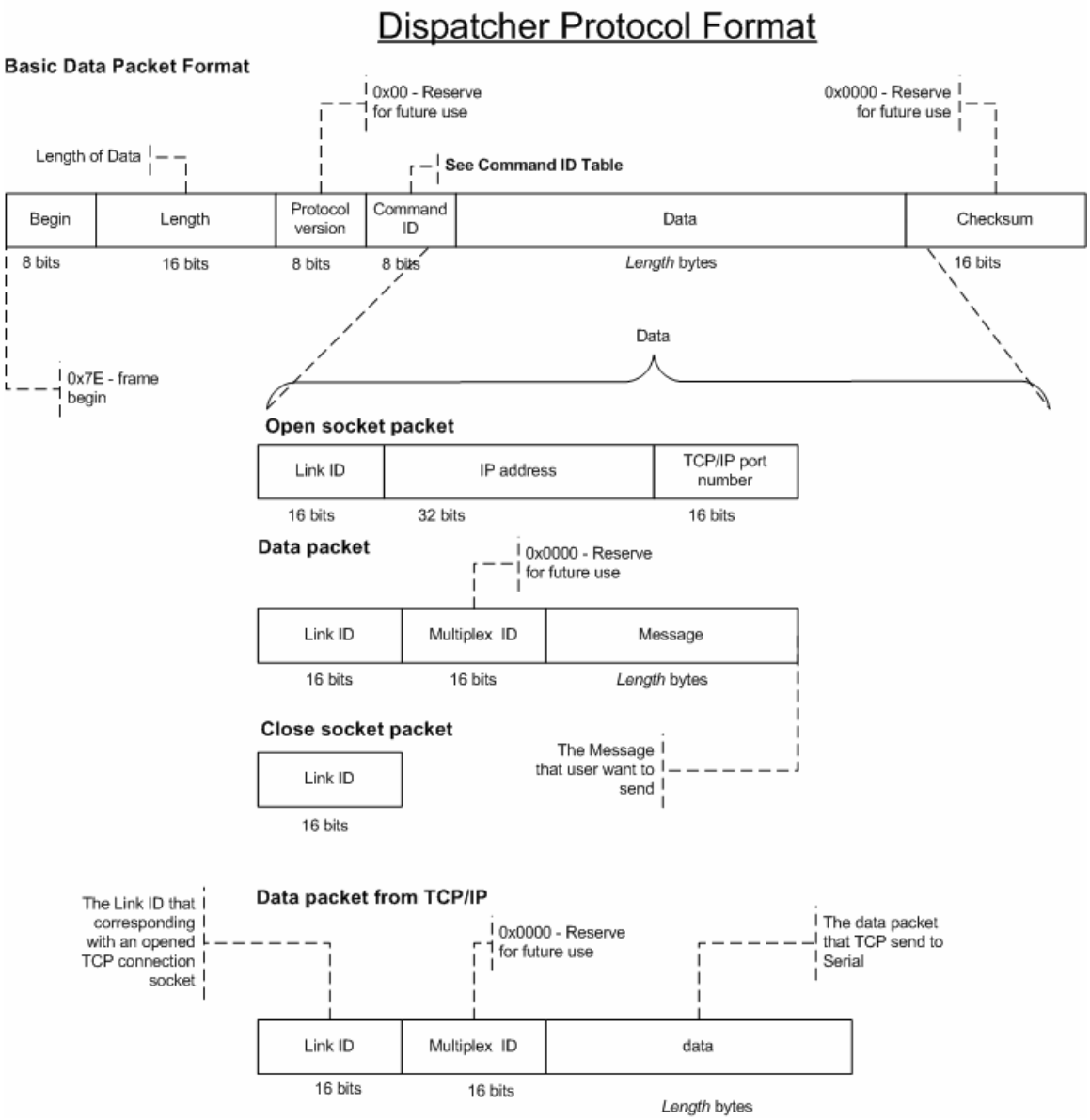

Figure 4 - Dispatcher Protocol

6.1.1 Command ID Value table

This section presents the valid values for the Command ID field presented earlier.

Command ID value	Description
0x01	Open Socket Command *
0x02	Send Data Command *
0x03	Close Socket Command *
0x04	Data from TCP device

* data from Serial device.

Figure 5 – Command ID values

6.1.2 Serialization convention

The Dispatcher expects the data coming from the Serial port respect the Big Endian bits ordering and network byte order.

7 Annex B - Use case example

This section presents a Serial-To-TCP Dispatcher protocol use case example.

7.1 Ross Interaction Use Case Diagram

The following sections present a use case in which the Dispatcher is used with a Ross Production switcher in order to send XML commands from to a Miranda Kaleido Multi-image system. Each subsection focuses on a specific Dispatcher packet.

7.1.1 Ross: Open socket Message

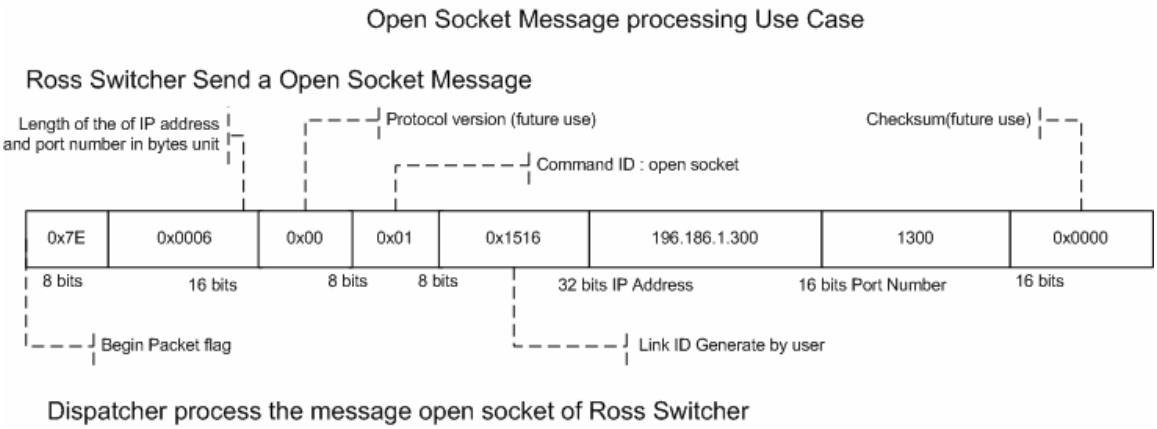

Figure 6- Ross Switcher Open socket example

7.1.2 Ross: Send data packet Message

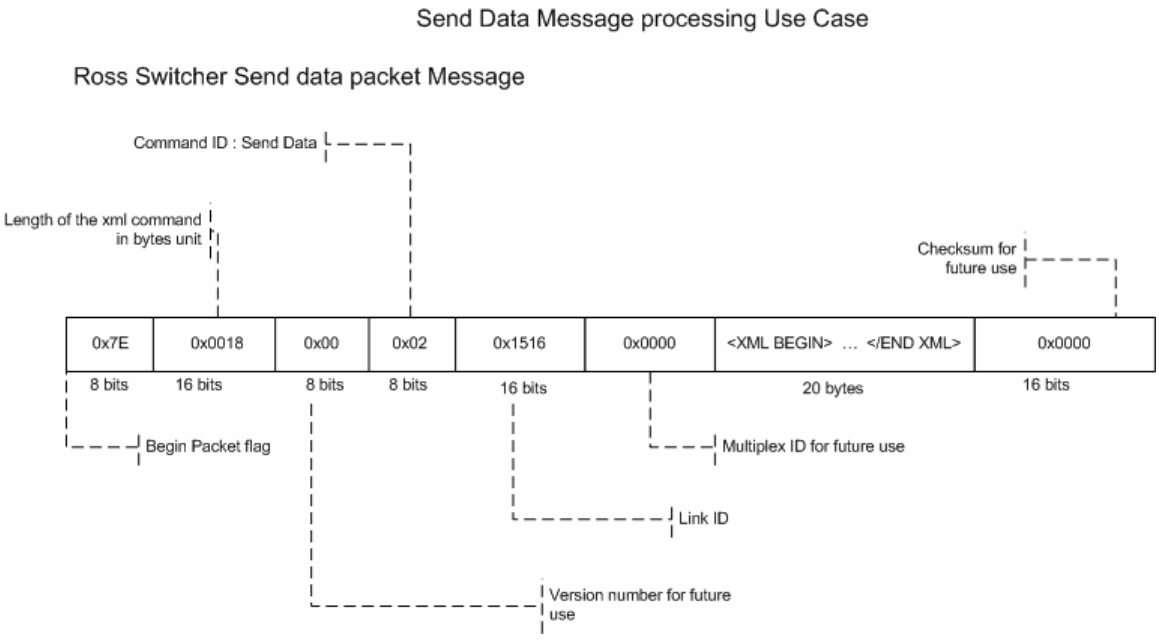

Figure 7 – Ross Example Send Data packet to TCP

7.1.3 Ross: Data from Kaleido

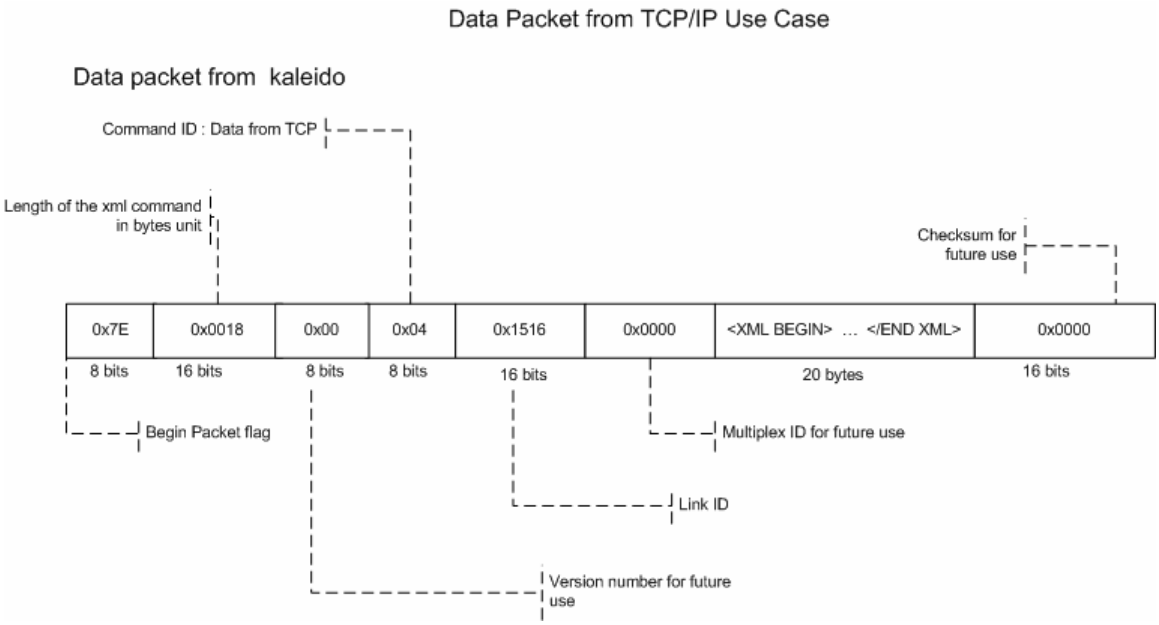

Figure 8 - Ross Example Data packet from TCP

7.1.4 Ross Close socket Message

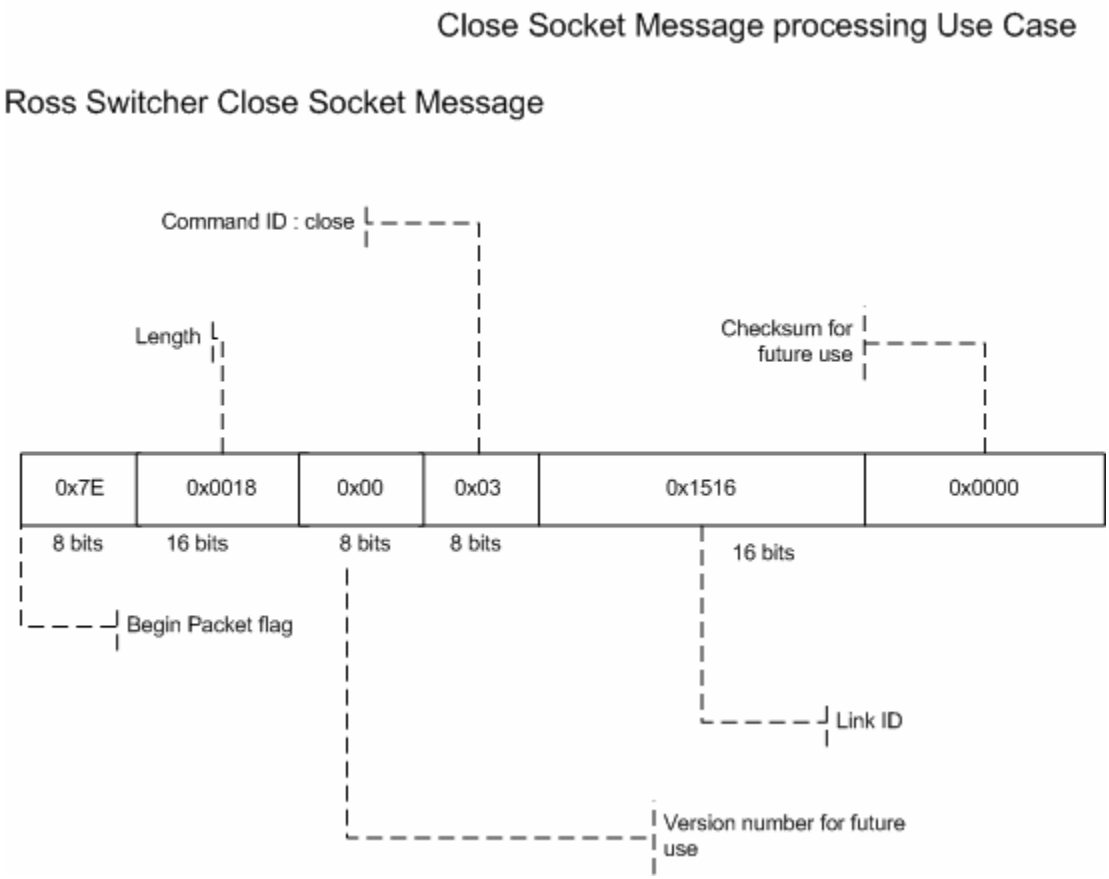

Figure 9 - Ross Example Close Message