

# **Kangaroo Packet Serial**

## **Reference Manual**

Copyright © 2013 Dimension Engineering LLC

## Table of Contents

<b>Simplified or Packet Serial?</b> .....	<b>3</b>
<b>Software Libraries</b> .....	<b>4</b>
<b>Packet Format</b> .....	<b>5</b>
<b>Example Code</b> .....	<b>6</b>
<b>Channel Names</b> .....	<b>8</b>
<b>Commands</b> .....	<b>9</b>
<b>Start</b> .....	<b>9</b>
<b>Units</b> .....	<b>9</b>
<b>Home</b> .....	<b>10</b>
<b>Move</b> .....	<b>10</b>
<b>Get</b> .....	<b>11</b>
Error Codes .....	12
<b>System</b> .....	<b>12</b>
Power Down .....	13
Power Down All .....	13
Set Baud Rate .....	13
Set Serial Timeout .....	13
Tuning .....	14
Enter Mode .....	14
Set Disabled Channels .....	14
Control Open Loop .....	14
Go .....	15
Abort .....	15
<b>On the Web</b> .....	<b>16</b>
<b>Revision History</b> .....	<b>17</b>
<b>Index</b> .....	<b>18</b>

# Simplified or Packet Serial?

Kangaroo supports two serial protocols, Simplified Serial and Packet Serial:

## **Simplified Serial**

An extremely easy way to interact with the Kangaroo from a microcontroller or PC.

For example, if you open up a terminal at 9600 baud (8-N-1), and have tuned channel 1 in Independent Mode, the following is actually a completely working sequence of commands:

```
1, START
1, HOME
1, GETMIN
1, GETMAX
1, P123
1, GETP
```

Mixed mode (tank-style) is similarly easy. If this interests you, see the manual's section on Simplified Serial.

## **Packet Serial**

Designed for high reliability even on noisy signal lines.

Among Packet Serial's advantages, it uses 14-bit CRCs, it is possible to make sure packets are received once and only once and to match up commands with replies, and it can share a S1 line with SyRen and Sabertooth motor drivers running in Packet Serial mode. On the other hand, Packet Serial is much more difficult than Simplified Serial to implement correctly.

If you decide to use Packet Serial, this documentation has all the necessary details. Alternatively, we've made [libraries](#) for several platforms.

# Software Libraries

We provide Packet Serial libraries for several platforms:

## **Arduino**

<http://www.dimensionengineering.com/arduino>

## **C#, VB.NET, and other .NET Framework languages**

<http://www.dimensionengineering.com/dotnet>

# Packet Format

## General Layout

Kangaroo packets have the following form:

**Address** (1 byte)

The Kangaroo address. By default, this is 128. Address bytes have the high bit set.

**Command** (1 byte)

The command number.

**Data Length** (1 byte)

The number of bytes  $n$  of data.

**Data** ( $n$  bytes)

The command data.

**CRC** (2 bytes)

See the [Example Code](#) section.

A 14-bit CRC computed from the address (excluding its high bit), command, data length, and data.

The lower 7 bits are written into the first byte, and the upper 7 bits into the second byte.

The polynomial is 0x21E8 in [Koopman](#) notation.

## Bit-Packed Numbers

See the [Example Code](#) section.

Many numbers sent to and from Kangaroo are larger than 7 bits. As a result they must be packed into multiple bytes. First, if the number is positive, it is doubled. If it is negative, the absolute value is taken, it is doubled, and then 1 is added. 6 bits are packed into each byte, starting with the lowest bits and moving up. The 7th bit is set in a byte if there are more bytes to come. Decoding is the reverse process. This means -64 to 64 is 1 byte, -4096 to 4096 is 2 bytes, and so on.

## Echo Code

Echo codes are 1 byte and can optionally be used in Get commands. The reply echoes back the same code sent with the initial command. You can use these to pair up commands with their replies.

## Sequence Code

Sequence codes are 1 byte and can optionally be used in most commands.

When used in any command except Get, they set the sequence code and do the action, unless the last received sequence code is identical. If it is identical, no action is taken. This prevents a command from executing more than once.

When used in a Get command, the last received sequence code is sent with the reply. The sequence code is not modified. You can use this to verify that a command was received.

At startup, the sequence code is 0 on all channels.

## Example Code

```
#include <stdint.h>

/*! Bit-packs a number.
\param buffer The buffer to write into.
\param number The number to bit-pack. Should be between  $-(2^{29}-1)$  and  $2^{29}-1$ .
\return How many bytes were written (1 to 5). */
size_t bitpackNumber(uint8_t* buffer, int32_t number)
{
    size_t i = 0;

    if (number < 0) { number = -number; number <<= 1; number |= 1; }
    else { number <<= 1; }

    while (i < 5)
    {
        buffer[i++] = (number & 0x3f) | (number >= 0x40 ? 0x40 : 0x00);
        number >>= 6; if (!number) { break; }
    }

    return i;
}

/*! Computes a 14-bit CRC.
\param data The buffer to compute the CRC of.
\param length The length of the data.
\return The CRC. */
uint16_t crc14(const uint8_t* data, size_t length)
{
    uint16_t crc = 0x3fff; size_t i, bit;

    for (i = 0; i < length; i++)
    {
        crc ^= data[i] & 0x7f;
        for (bit = 0; bit < 7; bit++)
        {
            if (crc & 1) { crc >>= 1; crc ^= 0x22f0; }
            else { crc >>= 1; }
        }
    }

    return crc ^ 0x3fff;
}

/*! Writes a Packet Serial command into a buffer.
\param address The address of the Kangaroo. By default, this is 128.
\param command The command number.
```

```

\param data      The command data.
\param length    The number of bytes of data.
\param buffer    The buffer to write into.
\return         How many bytes were written. This always equals 4 + length. */
size_t writeKangarooCommand(uint8_t address, uint8_t command,
                            const uint8_t* data, uint8_t length,
                            uint8_t* buffer)
{
    size_t i; uint16_t crc;

    buffer[0] = address; buffer[1] = command;
    for (i = 0; i < length; i++) { buffer[2 + i] = data[i]; }

    crc = crc14(data, 2 + length);
    buffer[2 + length] =  crc      & 0x7f;
    buffer[3 + length] = (crc >> 7) & 0x7f;
    return 4 + length;
}

/*! Writes a Move command for Position into a buffer.
   This could have many, many more options, but I've kept it basic
   to make the example easier to read.
\param address    The address of the Kangaroo. By default, this is 128.
\param channel    The channel name.
                 By default, for Independent Mode, these are '1' and '2'.
                 For Mixed Mode, these are 'D' and 'T'.
\param position   The position to go to.
\param speedLimit The speed limit to use. Negative numbers use the default.
\param buffer     The buffer to write into.
\return          How many bytes were written (at most 18). */
size_t writeKangarooPositionCommand(uint8_t address, char channel,
                                    int32_t position, int32_t speedLimit,
                                    uint8_t* buffer)
{
    uint8_t data[14]; size_t length = 0;
    data[length++] = (uint8_t)channel;
    data[length++] = 0; // move flags

    data[length++] = 1; // Position
    length += bitpackNumber(&data[length], position);

    if (speedLimit >= 0)
    {
        data[length++] = 2; // Speed (Limit if combined with Position)
        length += bitpackNumber(&data[length], speedLimit);
    }

    return writeKangarooCommand(address, 36, data, length, buffer);
}

```

## Channel Names

Each channel of the Kangaroo has a single-character name. These can be changed with [DEscribe](#).

By default, in Independent Mode, the channel names are '1' and '2'.  
In Mixed Mode, they are 'D' (Drive) and 'T' (Turn).



# Commands

## Closed-Loop

Kangaroo supports a variety of commands, detailed in the various pages here.

## Open-Loop

Sabertooth-style open-loop commands are also supported. These can be used if you need to temporarily run a channel open-loop. See the Sabertooth documentation for open-loop packet format and command information.

# Start

Starts a channel. Also, the Kangaroo LED will shine brightly for a third of a second.

You must call this before [Units](#), [Home](#), or [Move](#) commands will work.

## Command Format

The command number for Start is 32. Its data is:

**Channel Name** (1 byte)

**Flags** (1 byte)

0 for no options.

64 if a sequence code is used.

**Sequence Code** (optional, 1 byte)

# Units

Sets custom units for a channel. If you do not set custom units, the units you have set up in [DEscribe](#) will be used. If you haven't set any, machine units will be used.

This command may be called after you start the channel but before you home it.

## Command Format

The command number for Units is 33. Its data is:

**Channel Name** (1 byte)

**Flags** (1 byte)

0 for no options.

64 if a sequence code is used.

**Sequence Code** (optional, 1 byte)

**Desired Units** (bit-packed number)

The amount in your units that correspond to the specified amount of machine units.

**Machine Units** (bit-packed number)

The amount of machine units (millivolts or lines) corresponding to the specified amount in your units.

## Home

Homes a channel.

Channels that require homing will ignore [Move](#) commands until they have homed.

### Command Format

The command number for Home is 34. Its data is:

**Channel Name** (1 byte)

**Flags** (1 byte)

0 for no options.

64 if a sequence code is used.

**Sequence Code** (optional, 1 byte)

## Move

Tells the controller to do a motion.

### Command Format

The command number for Move is 36. Its data is:

**Channel Name** (1 byte)

**Flags** (1 byte)

0 for no options.

8 to not apply the speed limit and speed ramping source settings. By default, the speed limit comes from Kangaroo's potentiometers.

32 to use raw units. For analog, raw units are millivolts. For quadrature, 4 raw units equal 1 line.

64 if a sequence code is used.

Add these to combine options.

**Sequence Code** (optional, 1 byte)

**Motion Parameter(s)**

Use one or multiple motion parameters.

**Type** (1 byte)

1 for Position.

2 for Speed, if no Position parameter is included. If a Position parameter is included, it becomes a Speed Limit.

3 for Speed Ramping. This does not affect Position Control.

Add 64 to make the command incremental (relative to the current position and speed).

**Value** (bit-packed number)  
The value for the parameter.

## Get

Gets the current value of a parameter from the controller.

### Command Format

The command number for Get is 35. Its data is:

**Channel Name** (1 byte)

**Flags** (1 byte)

0 for no options.

16 if an echo code is used. It will be sent with the reply.

32 to use raw units. For analog, raw units are millivolts. For quadrature, 4 raw units equal 1 line.

64 for a sequence code to be sent with the reply.

Add these to combine options.

**Echo Code** (optional, 1 byte)

**Parameter** (1 byte)

1 to get the current position.

2 to get the current speed.

Add 64 to get an incremental position or speed (relative to the last [Move](#) you commanded).

8 to get the minimum position.

9 to get the maximum position.

### Reply Format

The reply command number is 67. Its data is:

**Channel Name** (1 byte)

**Flags** (1 byte)

1 if the reply is an error. If so, the value is the [error code](#).

2 if the value corresponds to a pending state. If the controller is in motion, the motion is not finished.

For an error, the error (such as "not homed") should self-clear.

16 if the reply has an echo code.

32 if the value is in raw units.

64 if the reply has a sequence code.

These are added together. Use bitwise AND (for example,  $(x \ \& \ 1) \ != \ 0$ ) to test individual flags.

**Echo Code** (conditional, 1 byte)

**Sequence Code** (conditional, 1 byte)

**Parameter** (1 byte)

**Value** (bit-packed number)  
The current value of the parameter.

## Error Codes

[Get](#) can return a number of errors:

**0**

No error occurred.

**1**

The channel is not started. Command [Start](#) on the channel.

**2**

The channel needs to be homed. Command [Home](#) on the channel.

**3**

A control error has occurred. Command [Start](#) on the channel to clear the control error.

**4**

The controller is in the wrong mode. For example, the DIP switches may be in mixed mode while the tune was done in independent mode.

**5**

The parameter is unknown.

**6**

A serial timeout occurred, or the TX line was disconnected. Command [Start](#) on the channel to clear the serial timeout.

## System

See the particular system command's documentation.

### Command Format

The command number for System is 37. Its data is:

**Channel Name** (1 byte)

**Flags** (1 byte)

0 for no options.

64 if a sequence code is used.

**NOTE:**

[Tuning](#) commands may have unusual effects on sequence code.

*It is inadvisable to rely on sequence code consistency before and after a tuning command.*

These effects are not necessarily limited to the channel being commanded.

**Sequence Code** (optional, 1 byte)

**System Command Number** (1 byte)

**Parameter(s)** (bit-packed numbers)

System commands use zero or more bit-packed numbers for all of their parameters.

## Power Down

Powers down the channel.

### System Command Format

The system command number is 0. It has no parameters.

## Power Down All

Powers down all channels of the controller that receives the command.

### System Command Format

The system command number is 1. It has no parameters.

## Set Baud Rate

Sets the baud rate.

This affects all channels of the controller that receives the command.

### System Command Format

The system command number is 32. It has one parameter:

#### Baud Rate

The possible values are 0, 1, 2, and 3, corresponding to 9600, 19200, 38400, and 115200 baud.

## Set Serial Timeout

Sets the serial timeout.

This affects all channels of the controller that receives the command.

### System Command Format

The system command number is 33. It has one parameter:

#### Timeout

The timeout, in 1/16ths of a second. 0 uses the [DEscribe](#) setting. -1 disables the serial timeout.

# Tuning

The Packet Serial tuning commands can be used to initiate a tune without touching the Autotune button. This is useful for systems where the Kangaroo is physically inaccessible.

[DEscribe](#) uses these same commands, and so can be used to initiate a tune remotely without any programming.

## Enter Mode

Enters the desired tune mode. This corresponds to pressing the tune button multiple times to get into the mode you want.

### System Command Format

The system command number is 3. It has one parameter:

#### Tune Mode

Mode 1 is Teach, 2 is Limit Switches, and 3 is Mechanical Stops.

## Set Disabled Channels

Initially all channels are disabled for safety reasons after entering a tune mode.

You must send this bitmask before beginning the tune.

### System Command Format

The system command number is 8. It has one parameter:

#### Disabled Channel Bitmask

0 enables all channels.

## Control Open Loop

Sets the open loop power. This can be used to position for a Teach tune.

### System Command Format

The system command number is 6. It has one parameter:

#### Power

The range is  $-(2^{28}-1)$  to  $2^{28}-1$ .

## Go

Begins the tune.

This corresponds to pressing the tune button after you are in the desired mode.

Tuning has an automatic serial timeout for safety reasons.

You must continually send packets or it will abort. [Get](#) commands in a loop will do the job.

### **System Command Format**

The system command number is 4. It has no parameters.

## Abort

Aborts the tune.

### **System Command Format**

The system command number is 5. It has no parameters.

## On the Web

Click the following link to visit our main website:

[Dimension Engineering](#)

If you have questions, we can be reached via either of the following:

[Help Desk](#)

[Contact Us](#)

Also, here are some software links you may find useful:

[DEscribe](#)

[Libraries for Arduino](#)

[Libraries for C# and .NET](#)



## Revision History

**March 29, 2013:**

Initial public version of the Packet Serial Reference.

**September 10, 2013:**

Corrected the buffer length in the C example code.

# Index

## A

Arduino *4, 16*

## B

baud rate *2, 13*

bit-packed numbers *5, 13*

## C

C# *4, 16*

channel names *2, 8*

CRC *3, 5, 6, 7*

## E

echo code *5, 11*

error codes *2, 12*

example code *2, 5, 6, 17*

## G

Get command *5*

## H

Home command *10*

## L

libraries *2, 3, 4, 16*

## M

Move command *7*

## P

packet format *2, 5, 9*

Power Down All command *13*

Power Down command *13*

## S

sequence code *5, 9, 10, 11, 12, 13*

serial timeout *2, 12, 13, 15*

Set Baud Rate command *13*

Set Serial Timeout command *13*

Simplified Serial *3*

Start command *9*

system commands *13*

## T

tuning remotely *14*

## U

Units command *9*

## V

VB.NET 4