# LEGO Power Functions RC

**Version 1.20**

# Introduction

The purpose of this document is to describe the RC protocol supported by the LEGO Power Functions RC Receiver.

## Attention!



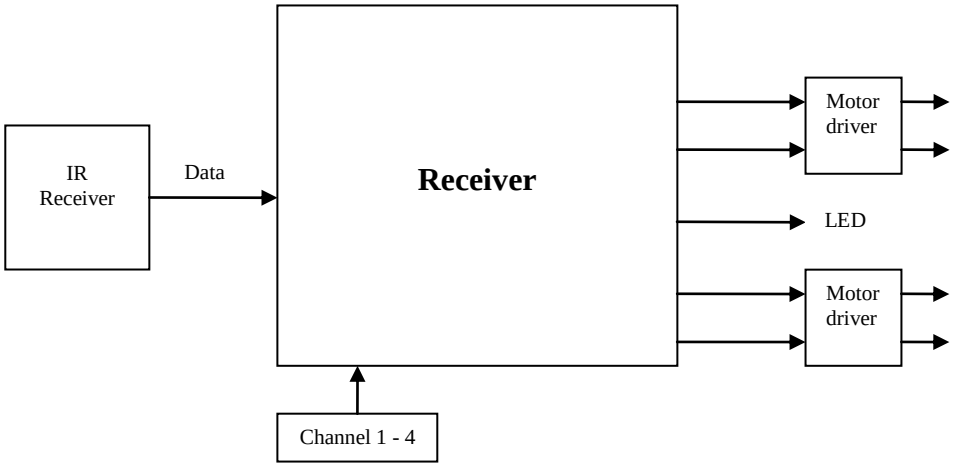
The LEGO Group 02/2010

# Table of content

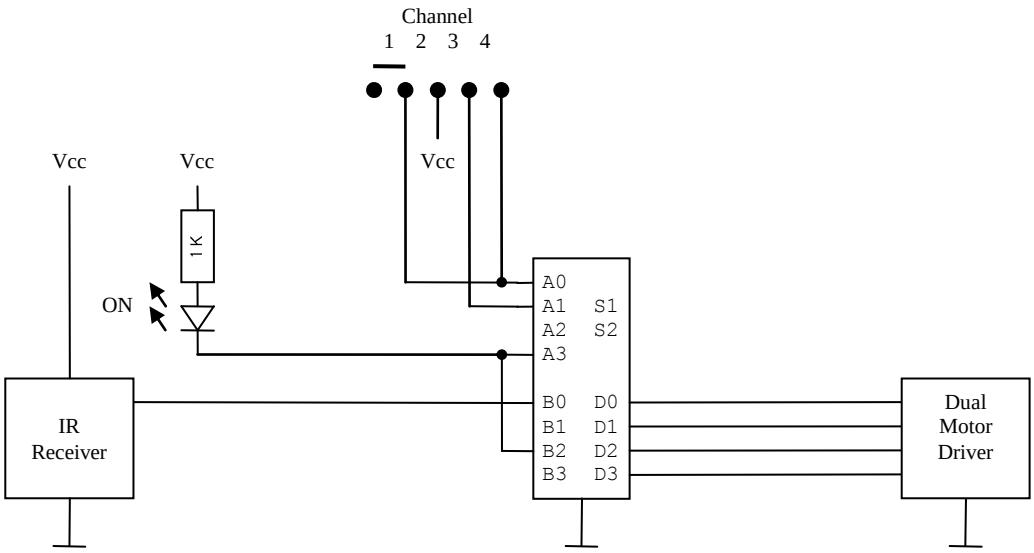
# LEGO Power Functions RC

## *LEGO Power Functions RC Receiver*

The receiver has input for IR data and channel switch and output for two LPF plugs and one LED.




## Application Schematics



## Description

This receiver firmware is capable of executing all commands in the “LPF RC Protocol” – acting in a variety of RC modes. Each mode implements a certain type of RC functionality.

When applying supply voltage the LED will give a short blink and then light up - the receiver is now ready. If a legal valid command of the right channel is received the LED will shortly turn off and indicate that the command is executed. The effect you will see is the LED blinking when messages are received.

The outputs of the RC Receiver are generic Power Functions outputs – in the following we will use motors as examples to describe the functionality of the control.

Depending on command the four output port pins will turn into two motor controls or individually controlled outputs. The motor outputs will either be forward, float, brake, backward – ON/OFF or PWM controlled. Some commands are timed out after 1.2 second when not receiving IR others are not. Default behavior is floating outputs.

The receiver does not power down and can only be turned off by removing its supply voltage.

## *LEGO Power Functions RC Protocol*

The payload is: 1 toggle bit, 1 escape bit, 2 bits for channel switch, 1 bit for address, 3 bits for mode and 4 bits for various data depending on mode.
The address bit is intended for enabling an extra set of 4 channels for future use. The current PF RC Receiver expects by default the address bit to be 0.
A message consists of: A special length synchronisation start bit, payload and "Longitudinal Redundancy Check" to validate the entire message before executing the command and at last a stop bit to terminate the message.

Binary representation:

| | Nibble 1 | | | | Nibble 2 | | | | Nibble 3 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ***start*** | ***T*** | ***E*** | ***C*** | ***C*** | ***a*** | ***M*** | ***M*** | ***M*** | ***D*** | ***D*** | ***D*** | ***D*** | ***L*** | ***L*** | ***L*** | ***L*** | ***stop*** |
| Start | Toggle | Escape | Channel | | Address | Mode | | | Data | | | | LRC | | | | Stop |

| | | | |
|---|---|---|---|
| **Start** | ***start*** | Special synchronisation start bit (see description under "Encoding") | |
| **Toggle** | ***T*** | 0-1 | Toggling for every new command |
| **Escape** | ***E*** | 0 | Use "Mode" to select the modes listed below |
| | | 1 | Combo PWM mode |
| **Channel** | ***CC*** | 0-3 | Channel switch 1 - 4 |
| **Address** | ***a*** | 0 | Default address space (from power up) |
| | | 1 | Extra address space |
| **Mode** | ***MMM*** | 000 | Extended mode |
| | | 001 | Combo direct mode |
| | | 01x | Reserved |
| | | 1xx | Single output mode |
| **Data** | ***DDDD*** | 0-15 | Data: different meaning depending on "Mode" |
| **LRC** | ***LLLL*** | xxxx = 0xF xor Nibble 1 xor Nibble 2 xor Nibble 3 | |
| **Stop** | ***stop*** | Same as Start | |

## Extended mode

This mode is able to control:
Brake, increment and decrement PWM in 7 steps on Output A and toggle Forward/Float on Output B. Toggle bit is verified on receiver. No timeout for lost IR.
From power up the address bit is always expected to be 0 (default address space). If the "Toggle Address bit" command is received (with a = 0) the extra address space is used and commands are from now expected to have the address bit set to 1. A new "Toggle Address bit" command (now with a = 1) will toggle back to default address space.
The "Align toggle bit" command has no action and is used to make sure the next command send is in sync.

Binary representation:

| | Nibble 1 | | | | Nibble 2 | | | | Nibble 3 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ***start*** | ***T*** | ***0*** | ***C*** | ***C*** | ***a*** | ***0*** | ***0*** | ***0*** | ***F*** | ***F*** | ***F*** | ***F*** | ***L*** | ***L*** | ***L*** | ***L*** | ***stop*** |
| Start | Toggle | Escape | Channel | | Address | Mode | | | Data | | | | LRC | | | | Stop |

| **Function** | ***FFFF*** | | |
|---|---|---|---|
| | | 0000 | Brake **then float** output A |
| | | 0001 | Increment speed on output A |
| | | 0010 | Decrement speed on output A |
| | | 0011 | Not used |
| | | 0100 | Toggle forward/float on output B |
| | | 0101 | Not used |
| | | 0110 | **Toggle Address bit** |
| | | 0111 | **Align toggle bit (get in sync)** |
| | | 1000 | **Reserved** |

## Combo direct mode

This mode is able to control: Two outputs float/forward/backward/brake.
This is a combo command controlling the state of both output A and B at the same time.
Toggle bit is not verified on receiver.
This mode has timeout for lost IR.

Binary representation:

| | Nibble 1 | | | | Nibble 2 | | | | Nibble 3 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ***start*** | ***T*** | ***0*** | ***C*** | ***C*** | ***a*** | ***0*** | ***0*** | ***1*** | ***B*** | ***B*** | ***A*** | ***A*** | ***L*** | ***L*** | ***L*** | ***L*** | ***stop*** |
| Start | Toggle | Escape | Channel | | Address | Mode | | | Data | | | | LRC | | | | Stop |

| | | | |
|---|---|---|---|
| **B output** | ***BB*** | 00xx | Float output B |
| | | 01xx | Forward on output B |
| | | 10xx | Backward on output B |
| | | 11xx | Brake **then float** output B |
| **A output** | ***AA*** | xx00 | Float output A |
| | | xx01 | Forward on output A |
| | | xx10 | Backward on output A |
| | | xx11 | Brake **then float** output A |

## Single output mode

This mode is able to control: One output at a time with PWM or clear/set/toggle control pins.
Toggle bit is verified on receiver if increment/decrement/toggle command is received.
This mode has no timeout for lost IR on all commands except “full forward” and “full backward”.

Binary representation:

| | Nibble 1 | | | | Nibble 2 | | | | Nibble 3 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ***start*** | ***T*** | ***0*** | ***C*** | ***C*** | ***a*** | ***1*** | ***M*** | ***O*** | ***D*** | ***D*** | ***D*** | ***D*** | ***L*** | ***L*** | ***L*** | ***L*** | ***stop*** |
| Start | Toggle | Escape | Channel | | Address | | Mode | | Data | | | | LRC | | | | Stop |

| | | | |
|---|---|---|---|
| **Mode** | ***M*** | 0 | PWM |
| | | 1 | Clear/Set/Toggle/Inc/Dec |
| **Output** | ***O*** | 0 | Output A |
| | | 1 | Output B |

**Mode = PWM**

| | | | |
|---|---|---|---|
| **Data** | ***DDDD*** | 0000 | Float |
| | | 0001 | PWM forward step 1 |
| | | 0010 | PWM forward step 2 |
| | | 0011 | PWM forward step 3 |
| | | 0100 | PWM forward step 4 |
| | | 0101 | PWM forward step 5 |
| | | 0110 | PWM forward step 6 |
| | | 0111 | PWM forward step 7 |
| | | 1000 | Brake **then float** |
| | | 1001 | PWM backward step 7 |
| | | 1010 | PWM backward step 6 |
| | | 1011 | PWM backward step 5 |
| | | 1100 | PWM backward step 4 |
| | | 1101 | PWM backward step 3 |
| | | 1110 | PWM backward step 2 |
| | | 1111 | PWM backward step 1 |

**Mode = Clear/Set/Toggle/Inc/Dec**

| | | | |
|---|---|---|---|
| **Data** | ***DDDD*** | 0000 | **Toggle full forward (Stop → Fw, Fw → Stop, Bw → Fw)** |
| | | 0001 | **Toggle direction** |
| | | 0010 | **Increment numerical PWM** |
| | | 0011 | **Decrement numerical PWM** |
| | | 0100 | Increment PWM |
| | | 0101 | Decrement PWM |
| | | 0110 | Full forward (timeout) |
| | | 0111 | Full backward (timeout) |
| | | 1000 | Toggle full forward/backward (default forward) |
| | | 1001 | **Clear C1 (negative logic – C1 high)** |
| | | 1010 | **Set C1 (negative logic – C1 low)** |
| | | 1011 | **Toggle C1** |
| | | 1100 | **Clear C2 (negative logic – C2 high)** |
| | | 1101 | **Set C2 (negative logic – C2 low)** |
| | | 1110 | **Toggle C2** |
| | | 1111 | **Toggle full backward (Stop → Bw, Bw → Stop, Fwd → Bw)** |

## Combo PWM mode

This mode is able to control: Two outputs with PWM in 7 steps forward and backward.
This is a combo command controlling the state of both output A and B at the same time.
Toggle bit is not verified on receiver.
This mode has timeout for lost IR.

Binary representation:

| | Nibble 1 | | | | Nibble 2 | | | | Nibble 3 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ***start*** | ***a*** | ***1*** | ***C*** | ***C*** | ***B*** | ***B*** | ***B*** | ***B*** | ***A*** | ***A*** | ***A*** | ***A*** | ***L*** | ***L*** | ***L*** | ***L*** | ***stop*** |
| Start | Address | Escape | Channel | | Output B | | | | Output A | | | | LRC | | | | Stop |

**Output B** ***BBBB***

| | |
|---|---|
| 0000 | Float |
| 0001 | PWM forward step 1 |
| 0010 | PWM forward step 2 |
| 0011 | PWM forward step 3 |
| 0100 | PWM forward step 4 |
| 0101 | PWM forward step 5 |
| 0110 | PWM forward step 6 |
| 0111 | PWM forward step 7 |
| 1000 | Brake **then float** |
| 1001 | PWM backward step 7 |
| 1010 | PWM backward step 6 |
| 1011 | PWM backward step 5 |
| 1100 | PWM backward step 4 |
| 1101 | PWM backward step 3 |
| 1110 | PWM backward step 2 |
| 1111 | PWM backward step 1 |

**Output A** ***AAAA***

| | |
|---|---|
| 0000 | Float |
| 0001 | PWM forward step 1 |
| 0010 | PWM forward step 2 |
| 0011 | PWM forward step 3 |
| 0100 | PWM forward step 4 |
| 0101 | PWM forward step 5 |
| 0110 | PWM forward step 6 |
| 0111 | PWM forward step 7 |
| 1000 | Brake **then float** |
| 1001 | PWM backward step 7 |
| 1010 | PWM backward step 6 |
| 1011 | PWM backward step 5 |
| 1100 | PWM backward step 4 |
| 1101 | PWM backward step 3 |
| 1110 | PWM backward step 2 |
| 1111 | PWM backward step 1 |

## *LEGO Power Functions RC Encoding*

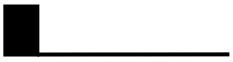
To ensure correct detection of IR messages six 38 kHz cycles are transmitted as mark. Low bit consists of 6 cycles of IR and 10 “cycles” of pause, high bit of 6 cycles IR and 21 “cycles” of pause and start bit of 6 cycles IR and 39 “cycles” of pause.

Graphically drawn:

IR Mark 

Low bit (data = 0) 

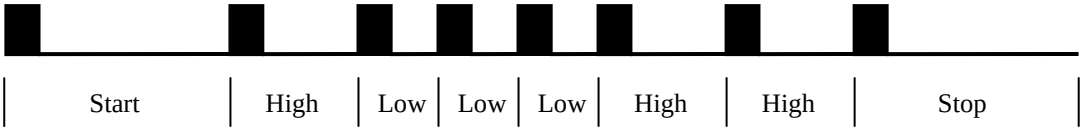
High bit (data = 1) 

Start/Stop bit 

The high pulse illustrates six 38 kHz cycles.

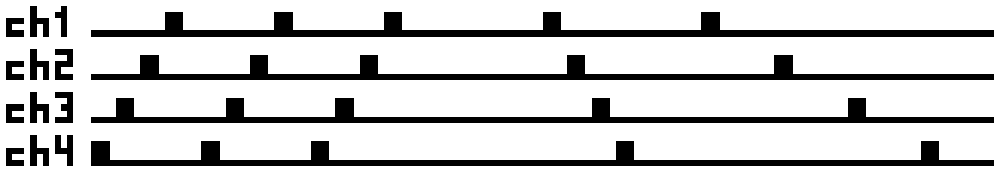
Low bit length = 16 x 1/38K = 421 us
High bit length = 27 x 1/38K = 711 us
Start bit length = 45 x 1/38K = 1184 us
Stop bit length = 45 x 1/38K = 1184 us

This example shows start bit, 6 bits and stop bit (not really the actual protocol).



## Transmitting Messages

When a button is pressed or released on the transmitter the message is sent. Five exactly matching messages (if no other buttons are pressed or released) are sent accordingly in time intervals depending on the channel selected. This ensures that other transmitters are not interfering with all the messages.




When a button is held down and the protocol needs update to prevent timeout the message is send continuously with a time interval as between message 4 and 5. First after all buttons are released and this is transmitted the transmitter will shut down.

If $t_m$ is the maximum message length (16ms) and Ch is the channel number, then

The delay before transmitting the first message is: $(4 - Ch)*t_m$

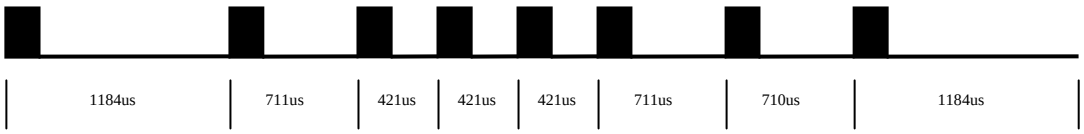
The time from start to start for the next 2 messages is: $5*t_m$

The time from start to start for the following messages is: $(6 + 2*Ch)*t_m$

## *LEGO Power Functions RC Decoding*

Decoding of message bits is done by measuring time from start of IR detection to next start of IR detection. Using only one, the active edge, stabilize the measured time nearly without influence of the automatic gain control in the IR receiver.

The example from above:




When the stop bits pause is reached the message is processed.

## Receiving Messages

The receiving firmware looks for a start bit and when this is detected it samples 16 data bits, calculates and compares the LRC. If any of the sampled bits are too long the sampling is terminated immediately and a new start bit is searched for.
When a bit time is sampled (measured) its time is hold against some limits.

| | |
|---|---|
| Low bit range | 316 - 526 us |
| High bit range | 526 – 947 us |
| Start/stop bit range | 947 – 1579 us |

Depending on the bit time a low or high bit is rotated into the receive buffer.