# Infrared Protocol for the Recoil Gun

|  |  |  |  |
| --- | --- | --- | --- |
| ***Version 1.0*** | ***Carl Muller*** | ***6 February 2016*** | ***Initial Version*** |
| ***Version 1.1*** | ***Carl Muller*** | ***19 May 2017*** | ***Specify grenade logic and states*** |
| ***Version 1.2*** | ***Carl Muller*** | ***24 May 2017*** | ***Change logic for priming*** |
| ***Version 1.3*** | ***Carl Muller*** | ***21 June 2017*** | ***Change protocol format to attempt to increase grenade range.*** |

This is a proposal for the Infrared (IR) protocol for the Recoil Gun project.

### Hardware Details

The transmission LED is a Vishay TSAL6100 (<http://www.vishay.com/docs/81009/tsal6100.pdf> ) which has a peak wavelength of 940nm. Or a cheaper Chinese one.

The receiver is a Vishay TSOP53338 (<http://www.vishay.com/docs/82745/tsop531.pdf> ) this has the statistics:

* 38kHz carrier (13.16µs on, 13.16µs off)
* 6 cycle minimum burst length (158µs)
* 10 cycle minimum gap (263µs) after a burst of 6 to 35 cycles

Alternative receivers: depending on stock availability we might also use these receivers

* Vishay TSOP4838
* Y-lin YL20170309-366Y

### Protocol options

Two defines independently control the format of the IR protocol being used.

On the grenade we no longer support SUPPORT\_FAST\_TICK which was

* 0 for 560µs (TV remote control standard) (~21 pulses)
* 1 for 320µs (~12 pulses)
* 2 for 420µs (16 pulses)

But instead now we supported **400 µs** / 800 µs marks and spaces (~15 or ~30 pulses), since the carrier is 38kHz and we are using Manchester (Bi-Phase) coding.

**But now (21 June) we support 600 µs marks and spaces to try to increase range.**

This tick controls the timing of the output pulses (mark and space). When polling, the interrupt takes place 8 times faster (so reading a 400µS pulse requires a 50µS interrupt, i.e. 20kHz).

***SUPPORT\_PROTOCOL***

* PROTOCOL\_NEC32 (TV remote control standard) NEC protocol with 32-bit payload (16-bit id) and full length header. Other protocols use shorter header pulses.
* PROTOCOL\_NEC20 NEC protocol with 20-bit payload
* PROTOCOL\_MAN20 Manchester protocol with 20-bit payload (13-bit id)
* PROTOCOL\_MAN16 Manchester protocol with 16-bit payload (pure serial number)
* PROTOCOL\_NEC9 NEC protocol with 9-bit payload (4-bit id)
* PROTOCOL\_MAN9 Manchester protocol with 9-bit payload (4-bit id)
* **PROTOCOL\_MAN20A** Manchester protocol with 20-bit payload (16-bit id). This is the one we use for the gun.
* PROTOCOL\_NEC12 Shortened NEC protocol with 12-bit payload.
* **PROTOCOL\_NEC4** Shortened NEC protocol with 4-bit payload. This is the one we use for the grenade.

### Low-level format

The transmission code method supports two methods of coding bits: NEC and Manchester.

NEC codes each payload bit 0 as “MS” over the air and bit 1 as “MSSS”, where “M” (Mark) is a burst of pulses using a 38kHz carrier, and “S” (Space) is an absence of carrier.

Manchester codes 0 as “SM” and 1 as “MS”

Both formats begin with a long header pulse. For the original NEC32 protocol this is “MMMMMMMMMMMMMMMMSSSSSSSS” (16 marks, 8 spaces, so at the 560µs tick speed this would be a 9ms pulse and a 4.5ms gap) but in the other protocols it is “MMMMMMMMSSSS” (8 marks, 4 spaces).

### Packet format

The MAN20A packet format has 16 bits of payload and 4 bits of CRC.

The packet format includes the id of the shooter (6 bits to let us expand to 64 players), the weapon type (4 Bits with 0..11 being normal weapons and 12..15 reserved for grenades) and the shot counter (6 bits)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bits** | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| **Type** | Shooter ID | | | | | | Weapon ID | | | | Shot counter | | | | | |
| **Grenade** | G | G | G | G | G | G | 1 | 1 | J | J | J | J | S | S | S | S |
| **Gun** | A | A | A | A | A | A | W | W | W | W | R | R | R | C | C | C |

* A = the id of the gun as set by the application (1..16) (default grenade == 50)
* W = Weapon ID the shot comes from (0-11 for the gun, 12-15 for the grenade)
* C = counter of shots fired (counts up from 0 and wraps)
* R = rounds in a plasma shot, according to the following formula: rounds = (RRR+1) \* 4.   
  e.g.: RRR=0 ->rounds=4   
   RRR=7->rounds=32
* G = grenade id (6 bit hash of 16 bit serial number)
* J = grenade random (4 bit random counter) for distinguishing grenade ids.
* S = grenade state (mostly acts as a countdown timer)

### Rounds in the shot

Used to convey the “damage dealt” by a plasma mode shot. Could also be used to artificially increase the in-game rate of fire without increasing the actual data rate (10 packets per second each containing 2 rounds would result in a simulated in-game RoF of 20Hz)

### Packet spacing

Packets should be paced so that the infrared receiver is not saturated.

One method we used on a previous project was several packets at 30ms intervals then a 100ms gap.

### Sensor independence

There are 4 independent IR sensors on the gun, so these need to be processed independently to make the signal reception more robust.

## Grenade Timing

The firmware cycles through the connected LED pairs outputting the same codes to each LED sequentially. The codes include the timing of the explosion (in seconds); 0 during the explosion, 10 to 1 in the countdown to the explosion.

### Grenade States

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | state\_unarmed | 8 | state\_6 |
| 1 | state\_cancelled | 9 | state\_5 |
| 2 | state\_priming | 10 | state\_4 |
| 3 | state\_primed | 11 | state\_3 |
| 4 | state\_10 | 12 | state\_2 |
| 5 | state\_9 | 13 | state\_1 |
| 6 | state\_8 | 14 | state\_explode |
| 7 | state\_7 | 15 | state\_waiting |

### Grenade State Logic

Initial state: If button is held down after 0.25 seconds then set state to “waiting” (for named kills if button held for 1 second, else “10”), else set state to “10” (anonymous kill).

End state name: “off”

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Button Up** | **Time out** | **LED** | **App** |
| 0 | Unarmed | X | X | X | (Does not see this state) |
| 1 | Cancelled | X | Off | Off | Could tell the user “that was close!” |
| 2 | Priming | Primed | X | Fast | Could give sfx to user |
| 3 | Primed | X | 10 | Solid | Must remember for awarding kills.  Could give sfx to user. |
| 4 | 10 | Cancelled | 9 | Slow | Could give ticking sfx to user |
| 5 | 9 | Cancelled | 8 | Slow | Could give ticking sfx to user |
| 6 | 8 | Cancelled | 7 | Slow | Could give ticking sfx to user |
| 7 | 7 | Cancelled | 6 | Slow | Could give ticking sfx to user |
| 8 | 6 | Cancelled | 5 | Medium | Could give ticking sfx to user |
| 9 | 5 | Cancelled | 4 | Medium | Could give ticking sfx to user |
| 10 | 4 | Cancelled | 3 | Medium | Could give ticking sfx to user |
| 11 | 3 | Cancelled | 2 | Fast | Could give ticking sfx to user |
| 12 | 2 | Cancelled | 1 | Fast | Could give ticking sfx to user |
| 13 | 1 | Cancelled | Explode | Fast | Could give ticking sfx to user |
| 14 | Explode | X | Off | Solid | Give damage to user |
| 15 | Waiting | 10 | Priming | Solid | (Does not see this state) |

## Alternative grenade protocol – NEC12

This protocol is used by the grenade to provide for greater range when using Y-lin receivers.

The test gun firmware just detects the header pulse.

The packet format is: (header) (12\*data bits) (stop bit).

Header: 2400us mark followed by 2400us space: “MMMMSSSS”

0 bit: 600us mark followed by 600us space: “MS”

1 bit: 600us mark followed by 1200us space: “MSS” (shorter than standard NEC codes)

Grenade protocol has 8 bits of payload and 4 bits of CRC

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Bits** | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| **Grenade** | P | P | P | P | J | J | J | J | S | S | S | S |

* P = parity/CRC for the packet (the sum of the lower two nybbles of a CRC).
* J = grenade random (4-bit random counter) for distinguishing grenade ids.
* S = grenade state (mostly acts as a countdown timer)

## Alternative grenade protocol – NEC4

This protocol is used by the grenade to provide for greater range when using Y-lin receivers.

The packet format is: (header) (4\*data bits) (stop bit).

Header: 2400us mark followed by 2400us space: “MMMMSSSS”

0 bit: 600us mark followed by 600us space: “MS”

1 bit: 600us mark followed by 1200us space: “MSS” (shorter than standard NEC codes)

Stop bit: 600us mark: “MSSSSSSS”

Grenade protocol has 4 bits of payload

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Bits** | 3 | 2 | 1 | 0 |
| **Grenade** | S | S | S | S |

* S = grenade state (mostly acts as a countdown timer)