The MilesTag 2 protocol

Christopher Malton

April 2011

This document aims to provide a complete reference to the MilesTag 2 protocol, as the documentation on the LaserTagParts website is sketchy at best. It covers all of the protocol in common use and supported in the stock MilesTag firmware as of 26th April 2011.

Contents

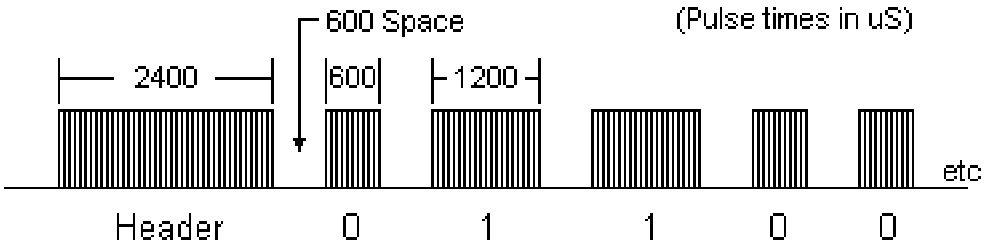

Figure 1: An example waveform for the MilesTag 2 protocol

1 The infrared protocol

The MilesTag 2 protocol is modulated onto a pulse-width-modulated (PWM) waveform. This PWM waveform either runs at 38, 40 of 56 kHz.

The MilesTag 2 protocol is a series of pulses of this carrier wave beginning with a 2400 microsecond (µS) long pulse. This is followed by a series of pulses either 1200µS or 600µS in length, representing 1 and 0 respectively. Each of the pulses is separated with a gap of 600µS of no carrier wave. The above is best explained by figure 1.

2 The data protocol

The data protocol is layered on top of the simple infrared protocol described in section 1. I will use the term "packet" to describe a collection of data "bytes" grouped together and sent following a single header. I use the term "bytes" loosely here, as in certain types of message, only six of eight bits are sent. There are two standard types of data packet: shot and message.

2.1 The shot packet

The shot packet is 14 bits in length, consisting of one bit for packet type, 7 bits for player ID, and 6 bits for team ID and damage.

This can be better represented as follows:

```
[Header] - [0ppppppp] - [ttdddd]
```

The 0 signifies a shot packet, ppppppp is replaced with the 7-bit player ID, tt is replaced with the team ID, and dddd is replaced with the damage value from the lookup table (see Appendix A).

2.2 The message packet

The message packet is a considerably more complex beast. It consists of one bit for packet type, 7 bits for the message ID, 8 bits for the message data, followed by 8 bits of message terminator.

This can be represented as follows:

```
[Header] - [1mmmmmmm] - [dddddddd] - [0xE8]
```

The 1 signifying a message, followed by the message ID, its data, and then the message termination literal. It is generally accepted that the message ID includes the leading 1, so all message IDs referred to in this document will either be referred to by name, or in hexadecimal form including the leading bit. Message IDs will therefore all be equal to or higher than 0x80.

At the time of writing, there are 13 registered messages in the protocol table, and the data bytes are message specific. Each of the messages are described in the following sections.

2.2.1 0x80 - Add Health

The message ID 0x80 corresponds to "Add Health", and its data byte is a value between 1 and 100, which will be added to the target player's health.

2.2.2 0x81 - Add Rounds

"Add rounds" adds the number of rounds specified in the data byte (between 1 and 100) to the tagger's current ammo count.

2.2.3 0x82 - Reserved

This message ID is reserved for future expansion.

2.2.4 0x83 - Command

There are numerous single-byte commands that can be sent, the following table lists the data byte followed by the details of what that command does:

Data byte	Command name	Actions taken
0x00	Admin Kill	Removes the remainder of a player's health
0x01	Pause/Unpause	Pauses/unpauses the timer and all in-play actions
0x02	Start Game	Starts a new game with the start delay active
0x03	Restore defaults	Resets a tagger to its default (factory) settings.
0x04	Respawn	Respawn a dead player
0x05	Immediate new game	Start a new game, ignoring the start delay
0x06	Full ammo	Restore a player's ammo counter to its initial value
0x07	End game	Immediately end the current game
0x08	Reset clock	Reset a player's clock to 00:00
0x09	Reserved	Reserved for future expansion
0x0a	Initialize player	Resets the tagger
0x0b	Explode player	Causes the explosion sound effect to be played. Sets health to 0
0x0c	New game (ready)	Prepares taggers for the 0x02 start game command.
0x0d	Full health	Reset the player's heath to its initial value
0x0e	Reserved	Reserved for future expansion
0x0f	Full armour	Reset the player's armour to its initial value
0x10	Reserved	Reserved for future expansion
0x11	Reserved	Reserved for future expansion
0x12	Reserved	Reserved for future expansion
0x13	Reserved	Reserved for future expansion
0x14	Clear scores	Resets all scoring data on the tagger
0x15	Test sensors	Flash the sensor LEDs
0x16	Stun player	Prevent the player doing anything for a few moments
0x17	Disarm player	Set the player's ammo counter to 0.

2.2.5 0x84, 0x85, 0x86 - Reserved

These message IDs are reserved for future expansion.

2.2.6 0x87 - System Data

There are multiple types of system data. The table below outlines the currently in-use data bytes. All of these packets will exceed the 3-byte length prescribed in section 2.2. The extra bytes for each system data type are explained in the relevant sections.

Data byte	Meaning	Extra info
0x00	Reserved	
0x01	Cloning Data	See Section 2.3
0x02	Reserved	
0x03	Score Data (pt 1)	See Section 2.4
0x04	Score Data (pt 2)	See Section 2.4
0x05	Score Data (pt 3)	See Section 2.4

2.2.7 0x88, 0x89 - Reserved

These message IDs are reserved for future expansion.

2.2.8 0x8A - Clips pickup

Picked up clips from the ammo box ID given the in data field (0-15)

2.2.9 0x8B - Health pickup

Picked up heath from the medic box ID given the in data field (0-15)

2.2.10 0x8C - Flag pickup

Picked up flag ID given the in data field (0-15)

2.3 Cloning data

For cloning, much more data is required to be sent. A further 36 bytes is sent after the 0xE8 byte to mark the end of the message.

This table starts from byte 4, which is the first byte sent after the 0xE8.

Byte number	Description of data
4	Reserved
5	Reserved
6	Reserved
7	Team ID - See section 2.3.1
8	Reserved
9	Clips added by picking up an ammo box
10	Health added by picking up a medic box
11	Reserved (0x03 for 5.41)
12	Hit LED timeout in seconds
13	Sound set - See section 2.3.2
14	Overheat limit in rounds/min
15	Reserved
16	Reserved
17	Damage per shot - See section 2.3.3
18	Clip size - 0xFF is unlimited
19	Number of clips - 0xCA is unlimited
20	Fire selector - See 2.3.4
21	Number of rounds for burst mode
22	Cyclic RPM - See 2.3.5
23	Reload delay in seconds
24	IR power - See 2.3.6
25	IR range - See 2.3.7
26	Tagger on/off settings - See 2.3.8
27	Respawn health - See 2.3.9
28	Reserved
29	Respawn delay in tens of seconds
30	Armour value
31	Game on/off settings 1/2 - See 2.3.10
32	Game on/off settings 2/2 - See 2.3.11
33	Hit delay - See 2.3.12
34	Start delay in seconds
35	Death delay in seconds
36	Time limit in minutes
37	Maximum respawns
38	Reserved (0xFF in 5.41)
39	Checksum - See 2.3.13

2.3.1 Byte 7 - Team ID

Value	Team name
0x00	Red
0x01	Blue
0x02	Yellow
0x03	Green

2.3.2 Byte 13 - Sound set

Value	Sound set name
0x00	Mil-sim
0x01	Sci-fi
0x02	Silenced (Rev H only)

2.3.3 Byte 17 - Damage per shot

Value	Damage dealt
0x00	1
0x01	2
0x02	4
0x03	5
0x04	7
0x05	10
0x06	15
0x07	17
0x08	20
0x09	25
0x0A	30
0x0B	35
0x0C	40
0x0D	50
0x0E	75
0x0F	100

2.3.4 Byte 20 - Fire selector

Value	Fire mode
0x00	Semi Auto
0x01	Burst
0x02	Full Auto

2.3.5 Byte 22 - Cyclic RPM

Value	RPM
0x00	250
0x01	300
0x02	350
0x03	400
0x04	450
0x05	500
0x06	550
0x07	600
0x08	650
0x09	700
0x0a	750
0x0b	800

2.3.6 Byte 24 - IR power

Value	IR power
0x00	Indoor
0x01	Outdoor

2.3.7 Byte 25 - IR range

Value	Range
0x01	Minimum
0x02	10%
0x04	20%
0x07	40%
0x0A	60%
0x0E	80%
0x12	Maximum

2.3.8 Byte 26 - Tagger settings

These values are ORed together to get the value of this field.

Value	Meaning
0x01	Muzzle flash on
0x02	Overheat feature on
0x04	
0x08	
0x10	
0x20	
0x40	
0x80	

2.3.9 Byte 27 - Respawn health

Value	Health	Value	Health	Value	Health	Value	Health
0x01	1	0x14	20	0x27	115	0x39	250
0x02	2	0x15	25	0x28	120	0x3a	300
0x03	3	0x16	30	0x29	125	0x3b	350
0x04	4	0x17	35	0x2a	130	0x3c	400
0x05	5	0x18	40	0x2b	135	0x3d	450
0x06	6	0x19	45	0x2c	140	0x3e	500
0x07	7	0x1a	50	0x2d	145	0x3f	550
0x08	8	0x1b	55	0x2e	150	0x40	600
0x09	9	0x1c	60	0x2f	155	0x41	650
0x0a	10	0x1d	65	0x30	160	0x42	700
0x0b	11	0x1e	70	0x31	165	0x43	750
0x0c	12	0x1f	75	0x32	170	0x44	800
0x0d	13	0x20	80	0x33	175	0x45	850
0x0e	14	0x21	85	0x34	180	0x46	900
0x0f	15	0x22	90	0x35	185	0x47	950
0x10	16	0x23	95	0x36	190	0x48	999
0x11	17	0x24	100	0x37	195		
0x12	18	0x25	105	0x37	195		
0x13	19	0x26	110	0x38	200		

2.3.10 Byte 31 - Game on/off settings (1/2)

These values are ORed together to get the value of this field.

Value	Meaning
0x01	
0x02	Hit LED enabled
0x04	Friendly fire enabled
0x08	Unlimited clips enabled
0x10	Zombie mode enabled
0x20	Medics enabled
0x40	Game boxes reset on respawn
0x80	Game boxes are not used up

2.3.11 Byte 32 - Game on/off settings (2/2)

These values are ORed together to get the value of this field.

Value	Meaning
0x01	
0x02	
0x04	Capture-the-flag display enabled
0x08	Respawn enabled
0x10	Tagger nicknames enabled
0x20	Old IR level field
0x40	Full ammo reset on respawn
0x80	Enable game mode

2.3.12 Byte 33 - Hit delay

Value	Delay (seconds)	Value	Delay (seconds)
0x00	0.00	0x0c	9
0x01	0.25	0x0d	10
0x02	0.5	0x0e	11
0x03	0.75	0x0f	12
0x04	1	0x10	13
0x05	2	0x11	14
0x06	3	0x12	15
0x07	4	0x13	16
0x08	5	0x14	17
0x09	6	0x15	18
0x0a	7	0x16	19
0x0b	8	0x17	20

2.3.13 Byte 39 - Checksum

The checksum is the sum of bytes 4 through 38, modulo 0x100.

Example:

```
Sum of 0xA9 + 0xAA + 0x00 + .... + 0x00 = 0x153
Checksum is therefore 0x53
```

2.4 Scoring data

The scoring data comes in three chunks, and they always come in the triplet part 1, part 2, part 3. The first contains some basic data about the player's game. The second and third are the list of times the player was hit by other players, and other players on their team.

Be aware that MilesTag out of the box supports scoring for up to 105 players..... but that those players don't all have names in the list.

2.4.1 Scoring data (part 1)

Byte number	Description of data
4	Player ID
5	Team ID (see 2.3.1)
6	Rounds fired (lower byte)
7	Rounds fired (upper byte)
8	Total hits (lower byte)
9	Total hits (upper byte)
10	Game time (minutes)
11	Game time (seconds)
12	Respawns
13	Tagged out counter
14	Flag counter
15	Checksum (See 2.3.13)

All of the above should be self explanatory, however for the avoidance of any doubt, byte 7 should be multiplied by 256, and added to byte 6 to get the total rounds fired. The same applies to bytes 9 and 8 for total hits.

The checksum is computed as in 2.3.13, but using bytes 4 through 14.

2.4.2 Scoring data (part 2)

This scoring data provided in this packet shows all hits that were recieved by the player.

Byte number	Description of data
4	Reserved
5	Hits by 000 Eagle (lower)
6	Hits by 000 Eagle (upper)
7	Hits by 001 Joker (lower)
8	Hits by 002 Joker (upper)
...	...
...	...
215	Hits by 105 (lower)
216	Hits by 105 (upper)
217	Checksum (See 2.3.13)

Byte 6 must be multiplied by 256, and added to byte 5, and this will give the number of times that the player this data came from was shot by the player referenced by the byte ID.

The checksum is computed as in 2.3.13, but using bytes 4 through 216.

2.4.3 Scoring data (part 3)

This scoring data provided in this packet shows hits from the same colour team (traitor shots) that were recieved by the player.

Byte number	Description of data
4	Reserved
5	Hits by 000 Eagle (lower)
6	Hits by 000 Eagle (upper)
7	Hits by 001 Joker (lower)
8	Hits by 002 Joker (upper)
...	...
...	...
215	Hits by 105 (lower)
216	Hits by 105 (upper)
217	Checksum (See 2.3.13)

Byte 6 must be multiplied by 256, and added to byte 5, and this will give the number of times that the player this data came from was shot by the player referenced by the byte ID.

The checksum is computed as in 2.3.13, but using bytes 4 through 216.

A Player ID reference

ID	Display name	ID	Display name	ID	Display name
0x00	Eagle	0x19	Rambo	0x32	Micro
0x01	Joker	0x1a	Snake	0x33	LgtMG
0x02	Raven	0x1b	Audie	0x34	HvyMG
0x03	Sarge	0x1c	Sting	0x35	ZOOKA
0x04	Angel	0x1d	Zeena	0x36	ROCKT
0x05	Cosmo	0x1e	Bugsy	0x37	GRNDE
0x06	Gecko	0x1f	Viper	0x38	CLYMR
0x07	Blaze	0x20	Jewel	0x39	MINE
0x08	Camo	0x21	Genie	0x3a	BOMB
0x09	Fury	0x22	Logan	0x3b	NUKE
0x0a	Flash	0x23	Razor		
0x0b	Gizmo	0x24	Slick		
0x0c	Homer	0x25	Venom		
0x0d	Storm	0x26	Rocky		
0x0e	Habit	0x27	Saber		
0x0f	Click	0x28	Crush		
0x10	Ronin	0x29	Titan		
0x11	Lucky	0x2a	Orbit		
0x12	Radar	0x2b	Vixen		
0x13	Blade	0x2c	Tank		
0x14	Ninja	0x2d	Rogue		
0x15	Magic	0x2e	Sheik		
0x16	Gonzo	0x2f	Gizmo		
0x17	Cobra	0x30	Siren		
0x18	Pappy	0x31	Dozer		