Tesla Vehicle Log Files

Tom Saxton • Updated June 10, 2016

Overview

The Tesla Roadster maintains a log file of events and activity that helps Tesla understand how the cars are being used and how they are performing. A copy of this log file can be downloaded using a USB keychain drive.

The log file contains two sections of records.

The first section contains a daily record with a small amount of data (e.g. the odometer reading) and records of any error conditions. With 2008 Roadsters, this section of the log is large and gets relatively small amounts of data, so it seems like it will last the lifetime of the vehicle. (We've had our Roadster for one year, there are about 450 daily records, and the first section is 0.8% full.) With 2010 Roadsters, the first section grows as needed, yielding smaller log files.

The second section contains much more detailed records, including a record for every second the car is on and a record for every minute of charging. The section overwrites the oldest records with new records and hold information for about the last 60 hours of driving, depending on how much time you spend charging relative to driving.

Downloading Log Files

To download a Roadster log file, obtain a USB flash drive with a capacity of no more than 2 GB with more than 12 MB of available free space. On this drive, create a top-level folder called VehicleLogs (no space between the two words).

Insert this drive into the USB port. On 2008 Roadsters, it is located under the center console on the vertical support just forward of the iPod cubby. (Where is it on 2010 Roadsters?)

A few seconds after the drive is inserted, if the car likes the drive, the touch screen will display a message that the log file is being written to the drive and telling you not to remove the drive until it is done. In the current version of the 2008 firmware, it takes about 14 minutes to write the file and the percent complete is displayed at the bottom of the screen in hexadecimal notation.

The Log File Components

The download is a single file with a name like *201005311610.tar*. The name indicates the date and time the log file was downloaded, in this case May 31, 2010 at 16:10 (4:10 pm). The file is written in the TAR format, a convenient way to pack a hierarchy of files and folders into a single file, originally created for backing up files to magnetic tape, hence "Tape ARchive."

To decode the file, just double-click on it in any reasonably current version of Windows or Mac OS X. On Linux, the tar utility will do the trick.

Once the tar file is unpacked, you should have a folder which contains an 8- to 12-MB file called vms_log. That's where most of the log data is.

There will also be a folder called *flash* that contains the following files:

File	**Notes**
ahr.log	Information about battery pack voltages and SOC.
chargeLocs2.cfg	A list of the all the locations where you've charged and limited the current. For each location, the file contains the GPS coordinates of that location, the current limit and a number that looks like the number of times you've charged there. **Note**: In spring of 2010, this file was replaced by an encrypted version called chargeLocs2.cfg.enc
firmware.rc	Versions of various firmware modules.
log_offsets	The offsets into the two log file sections (relative to the start of each section) where the next record should be written.
messages	Text messages for various events and error conditions with date and time for each.
messages.n	Older messages, there may be several of these files numbered starting with 0.
rms.log	Lots of numbers, significance unknown.
soh_debug.txt	More battery pack info.
soh_debug.txt.bak	An older version of the previous file.
soh_log.raw	A binary file, significance unknown.
user.cfg	Various state values including the number of range mode charges, whether the alarm is armed, if valet mode is active, the odometer, the cost per kWh hour entered into the touch screen by the owner, a hash of the user PIN, etc. **Note**: In spring of 2010, this file was replaced by an encrypted version called user.cfg.enc
vms.cfg	A text file with the VIN, factory ID, the car's "birthday", service ID, etc. **Note**: In spring of 2010, this file was replaced by an encrypted version called vms.cfg.enc
vt.dat	A text file with VIN, a timestamp, 99 voltages and 44 temperatures.
whpm.log	This appears to be data similar to what's shown in the Wh/mi screen. The first column appears to be hex values representing Wh (divide by 32.768 to get Wh) and the second column is the number of tenths of miles (decimal) in that drive segment. The file has forty segments of 1 mile each, except for the last segment which may be a partial mile. I need more data sets (the file plus a photo of the screen) to confirm this theory.

The Main Log File: vms_log

The binary file *vms_log* is where most of the log information resides. It is divided into two sections.

First, a section of small daily records and error reports dating back to the car's birth.

Second, an 8 MB section of much more verbose records that wraps around, overwriting the oldest records with new records. This section holds very detailed information about the recent history of the car, approximately the last 60 hours of driving. This sections contains a record for every second the car is on and for every minute the car is charging, plus other less frequent records.

For every second the car is on: speed, amperage, accelerator pedal position, whether the friction brake is pressed, whether regen is happening, and whether traction control is enabled.

Once every ten minutes when the car is on: the odometer (to the tenth mile) and state of charge (range mode percent).

Every minute during charging: the line voltage and amperage.

Every thirty minutes of charging: SOC and the available amperage.

Once per hour when the car isn't on or charging: the SOC.

Once per day: the odometer reading to the tenth mile.

According to Tesla Motors, the only GPS coordinates stored are the positions of current-limited charging locations, they say they do not include GPS coordinates in the driving records.

This is just a small fraction of the recorded, with many portions not currently decoded. The details of the known fields in the log records are listed below.

vms_log Record Format

The vms_log file is a sequence of variable-length records packed together into two sections. The second section is written as a circular buffer in which each time a new record crosses a 128 KB boundary, the old records in that 128 KB section are cleared out (all bits set) to make room for the new records. Presumably, the first section is also treated this way, although it seems that it will take over 100 years of use to cause that section to wrap.

Each record begins with two start bytes, a length byte and a seven-byte header. For example, in hexadecimal:

```
    03 02 29 01 ff 03 87 b4 d8 4b ...
```

Field	Sample	Notes
start bytes	`03 02`	Every record starts with 03 02.
length	`29`	The number of bytes in the record, including the length byte, but not counting the start bytes. In the sample, 0x29 = 41 bytes of data, plus the two start bytes. So this record has 41-8 = 33 bytes of data following the header.

Field	Sample	Notes
record type	`01`	The record type, in this case the vehicle information record.
t2	`ff`	The next byte presumably further specifies the type of record, which so far has not yielded any definitive information.
t3	`03`	This byte appears to be a group of flags, although it clearly has different meanings for different record types.
timestamp	`4BD8B487`	This is a 32-bit integer, least significant byte first, shown reversed into the standard base-16 order, representing the number of seconds since the first moment of Jan 1, 1970 UTC. This sample value corresponds to 04/28/2010 15:19:51 PST.

Following the header is data specific to that that record type whose contents and size may vary even among instances of a given record type. The last byte of this record data is a checksum byte. After this last byte there will typically be another 03 02 record start pair, unless the record is the most recent record and followed by clear space.

Checksum Calculation

If you add up all of the bytes in a record, starting with the length byte and including the checksum byte at the end, the result should be a multiple of 256. (If you compute the sum with an 8-bit integer, the result should be zero.) If the sum doesn't work out as expected, the record is invalid and should not be considered reliable. Personally, I've only seen this happen when there's a partial record near one of the 128k section boundaries in the transient section.

The Sections

The log uses vehicle information records (see below) to break the sections into smaller segments. This is especially important in the transient section where the data is written as a circular buffer with the vehicle info records marking the start of each new segment and new data replaces old data.

In the permanent section, all vehicle information records have the high bit of t3 set. In the transient section, the high bit of t3 is always cleared.

The Records

The following sections list the observed record types and the data fields that have been decoded. The record numbers are Tesla's number as they appear in the *vms_log* file, the names and abbreviations are my names for the records based on my interpretation of the data. The known data fields listed are a small fraction of what's in the log records, there is much more information that hasn't yet been decoded.

Type	Abbrev.	Data Bytes	Notes
1	VINF	32	Vehicle Information.
3	XX03	4	Not much is known.
4	XX04	4	Not much is known.
5	XX05	varies	Not much is know.
6	DAY	32	Daily record in long-term section.
7	IDLE	20	Status every 60 minutes when car is not on or charging.
8	C1MA	40	Info every minute when charging (V1.5 Roadsters)
9	C30M	12	Info every 30 minutes when charging.
10	DR1S	24	Info every second when car is on.
11	DR1M	26	Info every minute when driving.
12	D10M	14	Info every 10 minutes when driving.
23	ERR	varies	Error message records in long-term section.
24	C1MB	36	Info every minute when charging (V2.0 Roadsters).

Record Type 1: Vehicle Information (VINF)

The vehicle information record contains the VIN and firmware version. It's also used as a division header in the transient section. A new vehicle information record occurs in both sections when the firmware is updated, and at the start of each 128 KB division in the transient section (perhaps also in the long-term section when it gets that large, about five years in).

Here's a sample record from the transient section:

```
header: 03 02 29 01 FF 03 4BD8B487
```

```
data: 00 01 11 04 03 04 33 08 0F 00 00 00 35 59 4A 52 45 31
31 42 30 38 31 30 30 30 30 30 30 00 00 00 39
```

The rt value of 01 in the header tells us this is a vehicle information record, the length byte of 29 (hex) = 41 (decimal), tells us there are 41 - 8 - 1= 32 bytes of record data, plus the presumed checksum byte at the end.

The known data bytes are:

Offset	Sample	Meaning
2	`11`	third byte of firmware version (11 hex is 17 decimal)
3	`04`	second byte of firmware version
4	`03`	first bytes of firmware version
8	`0F`	the hardware package version (0F hex is 15 decimal)
12-26	`35 59 4A 52 45 31 31 42 30 38 31 30 30 30 30 30 30`	ASCII encoding of the VIN

In this sample, the VIN would be 5YJRE11B081000000, a 2008 US Roadster. A 2010 US Roadster will have a VIN that looks like 5YJRE1A14A1000000. In both cases, the last few zeros will be replaced with the VIN sequence number.

The touch screen would display 3.4.17 15 as the version string, indicating a version 1.5 2008 Roadster. A version 2 2010 Roadster would have a version string like 4.1.33 23.

Record Type 3: Input Events? (XX03)

This record appears in the log in response to various actions taken on the vehicle, such as opening a door, the trunk, or the charge port door; and inserting, turning or removing the key. The records may represent those specific events, or more likely internal state changes caused by those events.

Record Type 4: Operating State? (XX04)

The record appears to indicate the car's operating state, but no specific data values have been determined.

Record Type 5: Charging Related? (XX05)

This record occurs during charging and when the car wakes up. Not much is known about what it encodes.

Record Type 6: Daily Status (DAY)

This record occurs in the long-term section, once per day.

A sample record:

```
header: 03 02 29 06 FF 87 4C02E5D7
```

```
data: 62 0C 02 00 47 85 03 00 D9 A6 28 07 B0 17 EB 0B 96 14
9F 13 48 60 01 00 D9 0A C0 0A 0E 45 0D 13 72
```

Offset	Sample	Meaning
20-23	48 60 01 00	32-bit integer, little endian, represents odometer in tenths of a mile. (Non US models still log the archaic English units still used in the US.) In this case 0x00016048 = 90184 or 9,018.4 miles or 14,510.6 km.
26-27	C0 0A	Divide this 16-bit integer by 17.5039 to get a number that matches CAC, but is lower resolution than provided on the CAN bus. In this case 0x0AC0 = 2752, which scaled is 157.22 Ah with a resolution of ±0.5/17.5039 = 0.028. Note that Scott451 called this the minimum brick Ah with a slightly different scale factor.
29	45	Range mode state of charge in percent, always reported in range mode regardless of the car's current charge mode. In this case 0x45 = 69, so 69%.

It looks like there are other strictly increasing 32-bit integers in the record, bytes 0-3, 4-7, 8-11, etc. They could be total energy in/out of the battery pack, etc.

Record Type 7: Hourly Status (IDLE)

This record occurs every 60 minutes when the car is not on or charging.

Offset	Sample	Meaning
6	57	The state of charge in range mode percent. In this case 57 (hex) = 87 (decimal) indicates 87% range mode charge, which is about 100% in standard mode.

Record Type 8: Charging info once per minute (C1MA)

In version 1.5 (2008 model year) Roadsters, this record occurs once per minute while charging.

Offset	Sample	Meaning
8-9	03 29	16-bit integer represents scaled input current. The scale factor appears to be 327.68. In this case 2903 hex = 10499 decimal which scales to 32.04A.
10-11	AD 7A	16-bit integer represents scaled input line voltage. The scale factor appears to be 131.072. In this case 7AAD (hex) = 31405 which scales to 240.58 volts.

Record Type 9: Charging info once every 30 minutes (C30M)

This record appears every 30 minutes while charging. It has a different format for V1.5 and V2.0 Roadsters.

Offset	Sample	Meaning
6 (V1.5) 4 (V2.0)	46	Amperage available from the charge connector, which doesn't reflect any charge limit set via the touch screen. In this case 46 (hex) = 70 decimal indicates 70A is available from the charge connector.

Offset	Sample	Meaning
11 (V1.5) 7 (V2.0)	57	The state of charge expressed as range mode percent. In this case 57 (hex) = 87 (decimal) indicates 87% in range more, which is about 100% in standard mode.

Record Type 10: Drive info once per second (DR1S)

This record occurs every second when the car is on.

Offset	Sample	Meaning
0-1	20 03	16-bit integer indicates ESS voltage in half volts. 0302 (hex) = 800 (decimal) indicates 400V. (Thanks to Scott451.)
2-3	00 4B	16-bit integer indicates ESS amps multiplied by 32, negative values indicate taking current out of the ESS. 4B00 (hex) = 19200 (decimal) indicates 600A. (Thanks to Scott451.)
4-5	F1 01	16-bit integer indicates speed in 0.1 mph units. In this case, 01F1 (hex) = 497 (decimal) represents 49.7 mph or 80.0 kph.
10-11	F7 F4	16-bit integer indicates torque. Divide by 163.84 to get N-m then multiply by 0.737562 to get fl-lbs. F4F7 (hex) = 62711 (decimal) represents 282 ft-lbs. (Thanks to Scott451 for the field position, my scale factor unconfirmed.)

Offset	Sample	Meaning
12	43	The accelerator pedal position from 0 (not pressed) to 255 (pressed to the floor). In this case 43 (hex) = 67 (decimal), 67/255 is about 26.2% to the floor.
14	25	This byte contains several bit flags: 04 friction brake applied, 08 regen on, 40 traction control off.

Record Type 11: Info every minute when driving. (DR1M)

Not much is known about this record.

Record Type 12: Drive info once every 10 minutes (D10M)

This record occurs every ten minutes when the car is on.

Offset	Sample	Meaning
5	57	The state of charge expressed as range mode percent. In this case 57 (hex) = 87 (decimal) indicates 87% in range more, which is about 100% in standard mode.
8-11	48 60 01 00	32-bit integer, little endian, represents odometer in tenths of a mile. In this case 0x00016048 = 90184 or 9018.4 miles or 14,510.6 km.

Record Type 23: Error messages in long-term section (ERR)

Offset	Sample	Meaning
4-5	35 00	16-bit error code. In this case 0035 (hex) = 53 decimal. This is the error code that corresponds to the “Internal GPS Failure” error message.

Offset	Sample	Meaning
6	00	Length of error-specific data inserted after this length byte. In the case of error 53, there are no inserted bytes of data.

Record Type 24: Charging info once per minute (C1MB)

In V 2.0 (2010 model year) Roadsters, this record occurs once per minute while charging.

Offset	Sample	Meaning
4-5	03 29	16-bit integer represents scaled input current. The scale factor appears to be 327.68. In this case 2903 hex = 10499 decimal which scales to 32.04A.
6-7	AD 7A	16-bit integer represents scaled input line voltage. The scale factor appears to be 131.072. In this case 7AAD (hex) = 31405 which scales to 240.58 volts.

Revision History

09/28/2010 first release

06/20/2011 add “The Sections”

07/06/2011 add checksum info, fix VIN byte range

06/10/2016 add CAC and SOC in the Daily Status record