Garmin Points of Interest (POI)

File Format

Herbert Oppmann

herby@memotech.franken.de

http://www.memotech.franken.de/FileFormats/

2020-06-15

Content

Garmin Points of Interest (POI) File Format

Filename extension *.gpi

This documentation is based on own research and the sources listed in the references section.

Basic data types

All values are serialized in little-endian byte order (least significant byte first).

Type	Length	Description
char	1	ASCII character (see [9])
byte	1	8 bit unsigned integer (range 0 .. 255)
ushort	2	16 bit unsigned integer (range 0 .. 65535)
int	4	32 bit signed integer (range -2147483648 .. 2147483647)
uint	4	32 bit unsigned integer (range 0 .. 4294967295)
coord24	3	A 24 bit integer which has to be multiplied by 360.0° and divided by 2^{24} to get a longitude or latitude coordinate.
coord32	4	A 32 bit integer which has to be multiplied by 360.0° and divided by 2^{32} to get a longitude or latitude coordinate.

Version:
A byte or ushort where the value is calculated as major version x 100 + minor Version.
E.g. 0x0064 = 100 = V1.0

GDate:
An uint, where the values 0 or 0xFFFFFFFF both mean “no value given”. All other values have to be interpreted as unix time (see [4]), but with a starting value (epoch) of 1989-12-31 instead of 1970-01-01.
Note: While the number counts seconds, unix time is not defined as “number of seconds since the epoch”, but “number of days since the epoch” x 24 x 60 x 60 + “number of seconds since midnight”. The difference between the two is that unix time ignores leap seconds. The difference between unix time and UTC is already more than 20 seconds.

PString:
Pascal-like string.

Type	Content
ushort	Length. This is the number of bytes following this length field. The string may be empty, in which case this length is 0.
byte[n]	The characters of the string, encoded according to the CodePage. The string is not zero-terminated.

LString:
Multi-language string.

Type	Content
uint	Length. This is the number of bytes following this length field. The list of localized strings may be empty, in which case this length is 0.

Type		Content
Repeat until length bytes consumed:		
	char[2]	Two ASCII chars locale according to [10]. Within one LString, the locales must be unique. Seen values: CS, DA, DE, EN, ES, FI, FR, IT, NL, NO, PL, PT, RU and SV
	PString	Localized string

VarInt:
An unsigned integer in a variable-sized encoding

Using 1 byte: (probably, but not seen yet)

	7	6	5	4	3	2	1	0
0	6	5	4	3	2	1	0	=1

Contains 7 bit. Used for numbers in the range 0 .. 0x7F.

Using 2 bytes:

	7	6	5	4	3	2	1	0
0	5	4	3	2	1	0	=1	=0

	7	6	5	4	3	2	1	0
1	13	12	11	10	9	8	7	6

Contains 14 bit. Used for numbers in the range 0x0080 .. 0x3FFF.

Using 3 bytes:

	7	6	5	4	3	2	1	0
0	4	3	2	1	0	=1	=0	=0

	7	6	5	4	3	2	1	0
1	12	11	10	9	8	7	6	5

	7	6	5	4	3	2	1	0
2	20	19	18	17	16	15	14	13

Contains 21 bit. Used for numbers in the range 0x004000 .. 0x1FFFFF.

Using 4 bytes: (probably, but not seen yet)

	7	6	5	4	3	2	1	0
0	4	3	2	1	0	=0	=0	=0

	7	6	5	4	3	2	1	0
1	12	11	10	9	8	7	6	5

	7	6	5	4	3	2	1	0
2	20	19	18	17	16	15	14	13

	7	6	5	4	3	2	1	0
3	28	27	26	25	24	23	22	21

Contains 29 bit. Used for numbers in the range 0x00200000 .. 0x1FFFFFFF.

Note: The encoding does not exclude smaller numbers to be represented in more bytes than is necessary. But this is normally not done and therefore indicates some problem.

General file structure

The file consists of a list of records immediately following each other. The list is terminated by an End record.

FormatVersion '01' only: The presence of the index record indicates, that there is an optional index table immediately following the End record, up to the end of the file. All entries in the index table are uints. The first entry is the length of the remaining part of the table in byte. That means the value of the first entry is 4 times the number of following entries. The second entry is Unknown75 (seen value 2), and the third is the total number of Waypoints no matter how they are distributed amongst the POI Groups. The remaining entries are file offsets (from the beginning of the file) to each of the Waypoints, but not necessarily in order.

Obfuscation

Obfuscation is optional, and seen in FormatVersion '01' only. It starts after the Header1 and Header2 records. Starting with the first byte of the first obfuscated record, consecutive groups of 4 bytes are processed. It does not matter whether the starting offset of the record is on a 4-byte-boundary or not. The groups are not restarted on the start of following records. The grouping just runs until the end of the file. This also means throughout the optional index table after the End record.

Each byte within such a 4-byte group is processed individually. The first byte is processed with constant 0x48, the second with 0x06, the third with 0xB3, and the last with 0x00. A byte consists of two nibbles (groups of four bits). The operation here is a subtraction which is done individually for each nibble. One can also say, a byte-wide subtraction without half-carry, i.e. without overflow from the low-nibble to the high-nibble.

Example: End of the Header1 record (or its subordinated record in its extra data) and start of the first record which is obfuscated (in red), which is here the Index record.

In file:	61	73	6F	6E	21	20	5D	06	B3	00	72	06	B3	00	70	06
							−	48	06	B3	00	48	06	B3	00	48	06
Clear text:	61	73	6F	6E	21	20	15	00	00	00	3A	00	00	00	38	00

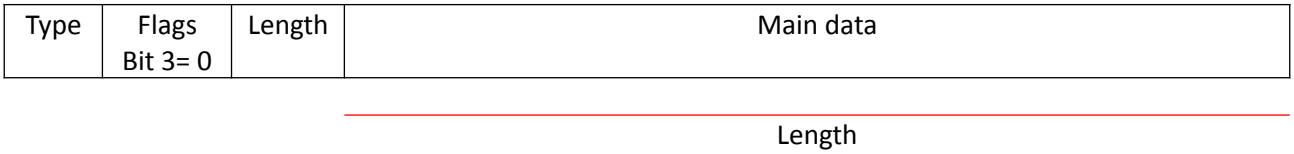
Here 0x15 0x00 is the record type, 0x00 0x00 is the record flags, 0x3A 0x00 0x00 0x00 is the record length, and so on.

The bytes with the orange background are where a normal byte-wide subtraction would have a different result than this nibble-wise subtraction. Instead of subtracting, it is also possible to add (nibble-wise, without half-carry) the two's-complement of the constant byte values, which are 0xB8 0xFA 0x4D 0x00.

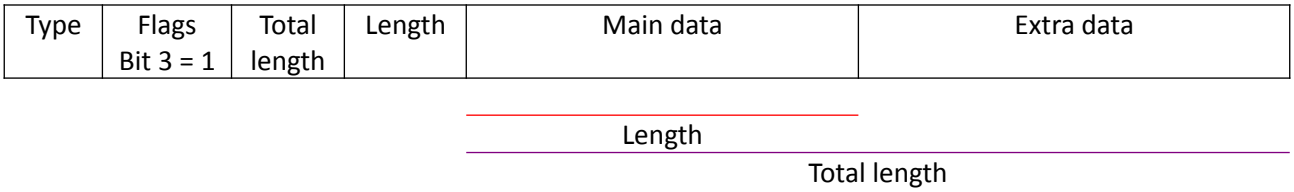
General record structure

Type	Content
ushort	Type Identifies the record content. See table below.
ushort	Flags, seen values: 0, 8 and 0x18 Bit 3: Extra data is present Bit 4: Only seen in FileFormat '01' on Header2 and POI Group records. Seems to indicate that record 23 and record 24 are present.
uint	If extra data present: Total data length in byte. This is the length of the main data plus the extra data.
uint	Length of main data in byte. Must be <= Total data length.
byte[Main data length]	Main data
byte[Total data length - main data length]	If extra data present: Extra data

Layout without extra data:



Layout with extra data:



Records contain main data and may optionally contain extra data. Main data is usually used for a structure (a sequence of fields with specific data type and meaning), but sometimes for a list of subordinated records. Extra data is usually used for a list of subordinated records, but sometimes for embedded media. Lists of subordinated records stop at the end of the main or extra data area. They are not terminated by the end record.

Some record types appear only as subordinated records within other records. But independent of where a record appears, the record type always has the same semantic. So the record content can be described with a single list with the record type as key.

Records seen in all FormatVersions:

Value	Meaning
0	Header1
1	Header2
2	Waypoint
3	Alert
4	Bitmap reference
5	Bitmap
6	Category reference
7	Category

Value	Meaning
8	Area
9	POI Group
10	Comment
11	Address
12	Contact
13	Image
14	Description
0xFFFF	End

Records seen in FormatVersion '01' only:

Value	Meaning
15	
16	
17	Copyright
18	Media
19	Speed camera
21	Index
23	
24	
26	
27	

Sequence of records on top level

FormatVersion '00':
Header1 Header2 POIGroup* End

FormatVersion '01':
Header1 Header2 [Index] POI Group* End [Index table]

[] = optional, * = zero or more occurrences

Record type 0: Header1

Main data: Length is at least 16.

Type	Content
char[6]	'GRMREC'
char[2]	FormatVersion: Two decimal digits. Values seen: '00', '01'
GDate	TimeStamp
byte	Unknown1. Seen values: 0, 1 in 3D_Bogenschiessen.gpi, 3 in Angelgeschaefte.gpi and 5 in AustriaGermanyCyclopsSample.gpi Maybe flag bits. So maybe a single field ushort Flags instead of two byte fields. Whether record 15 follows is independent of the bits here. Is always 0 in FormatVersion '00'
byte	Flags2. Seen values: 0, 3 and 5 Bit 0: File is obfuscated after Header1 and Header2 records

Type	Content
	Bit 1: ? Bit 2: ? Is always 0 in FormatVersion '00'
PString	Name Several examples from [15] have three bytes 0x00 between the string length and the actual string bytes, which are not counted by the string length?!? When this happens, the first unknown byte has a value of 3, but not vice-versa.

Extra data is sequence of records.

For FormatVersion '01':
[15]

Record type 1: Header2

Main data: Length is 12.

Type	Content
string	ASCII 'POI', 0-terminated
ushort	Unknown2. =0
char[2]	FormatVersion: Two decimal digits. Same as in record Header1
ushort	CodePage, (see [5] and [6]) Seen values: 0x036A Thai (Windows) 0x03B6 Chinese Traditional (Big5) 0x04E2 Central European (Windows) 0x04E3 Cyrillic (Windows) 0x04E4 Western European (Windows) 0xFDE9 Unicode (UTF-8)
ushort	Copyright record in extra data 0 = no 17 = yes (the copyright record has record type 17!)

Extra data is sequence of records.

For FormatVersion '01':
[Copyright]

Record type 2: Waypoint

Main data: Length is at least 15.

Type	Content
coord32	Latitude
coord32	Longitude
ushort	Unknown4. =1
byte	Unknown5. Seen values 0, 1, 2 (3D_Bogenschiessen.gpi), 3 (dorognoe_radio.gpi), 0x12 (bashneft_hi_res.gpi)

Type	Content
	Could also be flags. Bit 0 could be “Alert record in extra data” Bit 1 ? Bit 4 ?
LString	Shortname

Extra data is sequence of records.

For FormatVersion '00':
[Category reference] [Bitmap reference] [Alert] [Comment] [Address] [Contact] Image* [Description]

For FormatVersion '01':
[Category reference] [Bitmap reference] [Alert] [Comment] [Address] [Contact] Image* [Description] [26]

Record type 3: Alert

Speed, proximity and other stuff.

Main data: Length is 12.

Type	Content
ushort	Proximity in meter
ushort	Speed in 100 x meters / second, 0 = none
ushort	Unknown6. Seen values 0 (dorognoe_radio.gpi) and 0x100
ushort	Unknown7. Seen values 0 (dorognoe_radio.gpi) and 0x100
byte	Alert (0=Off, 1=On) (see [7]) Seen value 1
byte	Type of alert: 0=proximity (360° alert), 1=along road, 2=TourGuide (see [7]) Seen values 1 and 2 (in 2011_DOC_Camping_Sites_Update.gpi)
byte	Sound number (see [7]) In case Audio alert below is 0x00: seen values 4..105 could this be a symbol number? else if Audio alert below is "predefined audio": 0=beep, 1=tone, 2=3 beep, 3=silence, 4=plung, 5=double plung Seen values: 4 and 5 Else (custom audio): 1..n
byte	Audio alert 0x00, 0x01, 0x02, 0x03, 0x06, 0x07, 0x09, 0x0A, 0x0B, 0x0C, 0x27, 0x28, 0x29, 0x2A, 0x2B, 0x2C, 0x2D, 0x2E, 0x3A, 0x3B, 0x3E, 0x76, 0x77, 0x78, 0x79, 0x7A, 0x10 for predefined audio, 0x20 for custom audio (in media record)

Extra data is a sequence of records.

For FormatVersion '01':
[16] [27]

Record type 4: Bitmap reference

Main data: Length is 2 or 4.

Type	Content
ushort	Bitmap ID Points to the Bitmap record with this ID
ushort	Unknown8. Optional, =2

No extra data.

Record type 5: Bitmap

Main data: Length is 36.

Type	Content
ushort	Bitmap ID, starting with 0
ushort	Height
ushort	Width
ushort	Line size
ushort	Bits per pixel
ushort	Unknown9. =0
uint	Image size (= Line size x Height)
uint	Unknown10. Seen value 44
uint	Number of color palette entries Seen values: 0 (no palette), 16 (then bits per pixel must be 4) and 256 (then bits per pixel must be 8)
uint	Transparent color Seen values 0 and 0xFF00FF = Magenta
uint	Flags2, seen value 0, 1 and 0x0100 (D0743030F.gpi) Bit 1 (mask 0x0001): enable transparent mode? According to [2] Bit 8 (mask 0x0100): ?
uint	Unknown11. Seen values: 0 and Image size + 44
Image size x byte	Pixel
Number of color palette entries x uint	Optional: Color palette for 4 or 8 bit per pixel
n x byte	=0 Flags2.Bit1 is set when this is present, but not the other way round n is up to Image size, but sometimes less

No extra data.

Record type 6: Category reference

Main data: Length is 2.

Type	Content
ushort	Category ID Points to the Category record with this ID

No extra data.

Record type 7: Category

Main data: Length is at least 6.

Type	Content
ushort	Category ID, starting with 0
LString	Category

Extra data is sequence of records.

For FormatVersion '01':
[Bitmap reference]

Record type 8: Area

Main data: Length is 23.

Type	Content
coord32	Max. latitude
coord32	Max. longitude
coord32	Min. latitude
coord32	Min. longitude
uint	Unknown12. =0
ushort	Unknown13. =1
byte	Unknown14. Seen values 0, 1, 2 (3D_Bogenschiessen.gpi), 3 (dorognoe_radio.gpi), 5 (2011_DOC_Camping_Sites_Update.gpi), 8 and 18 (spb_metro_norm.gpi)

Extra data is sequence of records.

For FormatVersion '00':
Area* Waypoint*

For FormatVersion '01':
Area* (Waypoint* | Speed camera*)

Record type 9: POI Group

Main data: Length is at least 4.

Type	Content
LString	Data source
n x Record	List of area records (type 8).

Extra data is sequence of records.

For FormatVersion '00':
Category* Bitmap* Media*

For FormatVersion '01':
Category* Bitmap* Media* [23 24]

Record type 10: Comment

Main data: Length is at least 4.

Type	Content
LString	Comment

No extra data.

Record type 11: Address

Main data:

If FileVersion == '00': Length is 2.

If FileVersion == '01': Length is at least 2.

Type	Content
ushort	Flags: Bit 0: City, bit 1: Country, bit 2: State, bit 3: Postal code, bit 4: Street, bit 5: Street address

The following fields are only present when the respective bits in the Flags are set. In FileVersion '00', the strings are in the extra data, while in FileVersion '01', the strings are in the main data and nothing is in the extra data.

Type	Content
LString	City
LString	Country
LString	State
PString	Postal code
LString	Street
PString	House number

Record type 12: Contact

Main data:

If FileVersion == '00': Length is 2.

If FileVersion == '01': Length is at least 2.

Type	Content
ushort	Flags: Bit 0: Phone, bit 1: Phone2, bit 2: Fax, bit 3: Email, bit 4: Link

The following fields are only present when the respective bits in the Flags are set. In FileVersion '00', the strings are in the extra data, while in FileVersion '01', the strings are in the main data and nothing is in the extra data.

Type	Content
PString	Phone
PString	Phone2

Type	Content
PString	Fax
PString	Email
PString	Link

Record type 13: Image

Main data: Length is at least 5.

Type	Content
byte	Unknown15. =0
uint	Length of image
byte[Length of image]	Image Content (Bitmap or JFIF = JPEG Interchange File Format)

No extra data.

Record type 14: Description

Main data: Length is at least 5.

Type	Content
byte	Unknown16. Seen values: 1, 5, 50
LString	Description

No extra data.

Record type 18: Media

E.g. sound in MP3 format, pictures in JFIF (JPEG interchange file format).

Main data: Length is 3.

Type	Content
ushort	Media ID
byte	Media format: 0 = WAV, 1 = MP3 (see [8])

Extra data:

Type		Content
uint		Total length (including this length field!)
Repeated until total length reached:		
	char[2]	Locale
	uint	Length of media for this locale
	byte[Length of media]	Media Content

Record type 0xFFFF: End

Record flags is 0.

Main data: Length is 0, i.e. this record has no content.

No extra data.

The following records are only useable with FormatVersion '01'

Record type 15: ?

Main data: Length is 5 or 6.

Type	Content
ushort	FID (Family ID a.k.a. MapId) 0xFFFF = none
byte	PID (Product ID) 0xFF = none
byte	RgnID (Region ID) 0 = none 1 = United Kingdom + Ireland 2 = Netherlands 3 = France 4 = Belgium + Luxemburg 5 = Australia + New Zealand 6 = Spain + Portugal 7 = Italy + Slovenia 8 = Austria + Germany 10 = Nordics 14 = Eastern Europe 26 = Greece 27 = USA + Canada 28 = Russia 29 = South Africa 32 = Middle East 0xFF = none
byte	VenID (Vendor ID) Seen values: 0 (none), 2, 3, 11, 27, 28, 30 and 43 At least for speed cameras this is used like a release number
byte	Unknown17. Optional, seen values: 0xE9 and 0xEE

No extra data.

Record type 16: ?

See opencam_hi_res.gpi.

Main data: Length is 2 + n * 12.

Type		Content
ushort		Number of structures following (>0)
Repeat:		
	coord32	Latitude
	coord32	Longitude
	uint	Unknown19. Seen values: 8, 20, 22, 24, 26, 29, 37, 71, 73, 88, 106, 523, 9578, 11660 and many more

No extra data.

Record type 17: Copyright

Main data: Length is at least 24.

Type	Content
ushort	Flags1. Seen values: 0x0004, 0x0014, 0x0402, 0x0404, 0x0480 and 0x0481 Bit 0 (mask 0x0001): ? Bit 1 (mask 0x0002): Contains records 23/24 Bit 2 (mask 0x0004): ? Bit 4 (mask 0x0010): Optional uint is present in this record Bit 7 (mask 0x0080): Contains Speed camera records Bit 10 (mask 0x0400): Optional Device model string is present in this record
ushort	Flasg2. Seen values: 0x0000, 0x0080, 0x00A0 and 0x01A0 Bit 5 (mask 0x0020): Contains a waypoint index and an index record Bit 7 (mask 0x0080): ? Bit 8 (mask 0x0100): Optional Pictures and LString present in this record
ushort	Unknown24. Seen values: 0 and 25
ushort	Unknown25. Seen values: 0, 2, 3, 5, 9 and 11
LString	Data source
LString	Copyright

Optional, present if Flags1 bit 10 (mask 0x0400) is set:

Type	Content
PString	Device model

Optional, present if Flags2 bit 8 (mask 0x0100) is set:

Type			Content
12 Pictures, each consisting of a picture header and the picture itself:			
	Only present on 1st and 3rd picture:		
		VarInt	Some length On first picture: Could be Length of complete optional block On third picture: Could be length of remaining 10 pictures
		byte	Unknown26. Seen values: On first picture: 23 – ? On third picture: 10 – could be number of remaining pictures
		byte	Unknown27. =0
	VarInt		Length of picture entry in byte, i.e. length of remaining header bytes plus the picture

Type		Content
		itself
	byte	Unknown28. =0
	ushort	Width in pixel
	ushort	Height in pixel
	VarInt	Length of picture in byte
	byte[]	Picture in JFIF (JPEG interchange file format)
LString		Unknown29

Optional, present if Flags1 bit 4 (mask 0x0010) is set:

Type	Content
uint	Unknown30. Seen values: 08 0f de 07 (in bashneft_hi_res.gpi) – looks more like 2 ushorts or even 4 bytes with flags 07 0f de 07 (in bp_Hi_Res.gpi) 01 01 df 07 (in dorogne_radio.gpi) 04 15 e5 07 (in garmin.gpi) 04 0f e4 07 (in garminRU.gpi) 06 01 e2 07 (in gazpromneft_Hi_Res.gpi) 09 01 e1 07 (in kirishi.gpi) 05 01 e2 07 (in lukoil_hi_res.gpi) 04 11 e4 07 (in mapcamRU.gpi) 02 0f e4 07 (in msk_metro_hi_res.gpi) 06 01 e1 07 (in neste.gpi) 03 01 e6 07 (in rosneft.gpi) 0a 1d e1 07 (in shell_hi_res.gpi) 06 19 de 07 (in tnk_hi_res.gpi)

No extra data.

Record type 19: Speed camera

Main data: Length is at least 26.

Type	Content
coord24	Max. latitude
coord24	Max. longitude
coord24	Min. latitude
coord24	Min. longitude
byte	Unknown31. Defines the structure of the following bytes. Seen values: • 0x00, 0x03, 0x0f, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x17, 0x18, 0x19, 0x1a, 0x1b, 0x1c, 0x1d, 0x1f, 0x20, 0x21, 0x22, 0x23, 0x24, 0x27, 0x28, 0x29, 0x2a, 0x2b, 0x2c, 0x2f, 0x30, 0x31, 0x32, 0x33, 0x37, 0x38, 0x39, 0x3a, 0x3b • 0x81 • 0x80, 0x82, 0x83, 0x85, 0x8b, 0x8c, 0x8d, 0x8f, 0x90, 0x91, 0x92, 0x93, 0x94, 0x95, 0x97, 0x98, 0x99, 0x9a, 0x9b, 0x9c, 0x9d, 0x9f, 0xa0, 0xa1, 0xa2, 0xa3, 0xa4, 0xa5, 0xa7, 0xa8, 0xa9, 0xaa, 0xab, 0xac, 0xaf, 0xb0, 0xb1, 0xb2, 0xb3, 0xb4, 0xb6, 0xb7,

Type	Content
	0xb8, 0xb9, 0xba, 0xbb, 0xbc, 0xc0, 0xc1, 0xc2, 0xc3, 0xc8, 0xc9, 0xca, 0xcb, 0xd0, 0xd1, 0xd2, 0xd3, 0xd8, 0xd9, 0xda, 0xdb, 0xe0, 0xe1, 0xe2, 0xe3, 0xe8, 0xe9, 0xea, 0xeb, 0xf0, 0xf3 Seems to be: Bit 7-3: Type (or maybe only Bit 7-4: Type and bit 3 is also a flag) Bit 2-0: Flags?
byte[11]	Optional, if Unknown31 == 0x81 First byte seen: 2, 3, 10, 11 Remaining bytes: 00 00 00 55 55 15 c7 71 dc 01 (Notice that a three-nibble-value repeats: 0xc71 0xc71) Instead of first byte being 0x81, the condition could also be: 1) If bit 7 of first byte is set, a second byte follows 2) If second byte is 2, 3, 10 or 11 (or if bit 1 of the second byte is set), then this sequence follows, otherwise not.
byte	Optional, if Unknown31 is 0x80 or is > 0x81 Seen values 1, 8, 9
byte	Number of 10-bit values following. 0 means one value following, 1 means two values following ...
byte	Optional, if Unknown31 == 0x81 = 10 (bit-size of each value)
byte[]	List of 10-bit values (compacted). The least significant bits of each 10 bit value come first. Unused bits are in the most significant bits of the last byte and are 0. Example: 21 85 24 32 In bits: 0010 0001 1000 0101 0010 0100 ~~00~~11 0010 The three 10-bit-values are: 0100100001 0100100001 1100100010
coord24	Latitude
coord24	Longitude
byte[n]	Unknown area. n >= 3, up to 175 seen

Observed patterns for unknown area: Not yet known

No extra data.

Record type 21: Index

The presence of this record indicates, that an index table is following after the end record, up to the end of file.

Main data: Length is 58.

Type	Content
ushort	Length until end of this record, i.e. it is the record length minus 2
uint	Offset of the index table (from beginning of file)
byte[32]	=0
uint	Offset of the index table (from beginning of file)

Type	Content
uint	Total length of the index table in byte (the first entry in the index table is the length in byte of the following entries, so this value here is the same, but +4 for the first entry itself).
uint	Unknown36. =1
uint	Unknown37. =0
uint	Unknown38. =0

No extra data.

Record type 23: ?

Main data: Length is at least 10.

Type	Content
ushort	Unknown39. =1
ushort	Unknown40. =6
ushort	Unknown41. Seen values 2 and 3
LString	Unknown42. ?

Extra data is sequence of records.

Bitmap+ (or *?)

Record type 24: ?

Main data: Length is 2.

Type	Content
ushort	Unknown44. =1

No extra data.

Record type 26: ?

Main data: Length is 5 or 6 (in TRCameronHlands.gpi).

Type	Content
byte	Unknown45. Seen values 1 and 0x41. Seems to be flags and bit 6 (mask 0x40) seems to indicate the optional byte
uint	Unknown46. ? Some kind of length/offset or checksum?
byte	Unknown47. Optional, =0

No extra data.

Record type 27: ?

Main data: Length is at least 4 and even.

Type		Content
ushort		Number of structures following (>0)
Repeat:		
	byte	? These values are often (but not always) a multiple of 5 or 10
	byte	?

No extra data.

According to [8] “extending the alert records”

References

Used sources of information

[1] https://www.gpsbabel.org/htmldoc-development/fmt_garmin_gpi.html

[2] https://github.com/gpsbabel/gpsbabel/blob/master/garmin_gpi.cc

[3] https://www.pinns.co.uk/osm/gpi.html

[4] https://en.wikipedia.org/wiki/Unix_time

[5] Code Pages http://msdn.microsoft.com/en-us/goglobal/bb964653.aspx

[6] Code Pages http://en.wikipedia.org/wiki/Code_page

[7] https://www.gpspower.net/software-tools/196870-garmin-content-toolkit-v-4-0-a-4.html

[8] http://www.poi-factory.com/node/31703

Standards and specifications

[9] ISO/IEC 646:1991, Information technology – ISO 7-bit coded character set for information interchange

[10] ISO 639-1:2002 – Codes for the representation of names of languages – Part 1: Alpha-2 code

Sources of sample files

[11] https://www.pinns.co.uk/osm/gpi.html

[12] https://www.bordatlas.de/overlays.php

[13] http://www.gmaptool.eu/en/content/gpi-device

[14] https://www.garmin.com.sg/richpoi/

[15] https://www.garmin.com/de/extras/poi2/

[16] https://www.garmin.ru/download/extras/poi.php

[17] http://gps.maroufi.net/gpi-pois.shtml

[18] https://www.tourenfahrer.de/reise/pois-fuers-motorradnavi/motorradhotels-als-poi/tf-partnerhaeuser-als-pois-fuer-garmin/

[19] https://www.promobil.de/gps-daten/stellplaetze-gps-daten-navi/

[20] https://bikeaway.info/gps-navigation/poi-gps-navigation/bmw-motorradhaendler-poi/

[21] https://www.paesse.info/pois-points-of-interest-paesse-schweiz-und-alpen-fuer-garmin-und-tomtom/

[22] https://github.com/gpsbabel/gpsbabel/tree/master/reference

[23] https://sites.google.com/a/smail.nchu.edu.tw/chemistry-education/gps/garmin/speed-camera