

# DataFormats

---

## Points of Interest. (GPI extension)

The specifications used for reading and writing the GPI file format can be found here.

[https://www.memotech.franken.de/FileFormats/Garmin\\_GPI\\_Format.pdf](https://www.memotech.franken.de/FileFormats/Garmin_GPI_Format.pdf)

Herby Franken has more file formats documented.

<https://www.memotech.franken.de/FileFormats/>

Note: For all people that can read Pascal, have a look at unit 'UnitGPI.pas', that implements this format.

## Subclass field

The subclass field can be found in a GPX file created from Basecamp, or Mapsource. But also in the trip files in field **mUdbDataHndl**.

Data that can found in the subclass field.

- MapSegment
- RoadId
- Begin/Shaping point/Via point
- Lat/Lon values

See: Subclass for RoutePoints in Garmin GPX.pdf

## Trip File format

Notes:

- For all people that can read Pascal, have a look at unit 'UnitTripObjects.pas', that implements this format.
- '0x' is used for Hexadecimal notation. E.G. 0x0A = 10

The file starts with a header

Field	Size (in Bytes)	Sample Value	Description
ID	4	'TRPL'	Identifier of the format. Always 'TRPL'
SubLength	4	0x0007 9A03 498179	Size of the sub elements. Filesize minus 8. (ID + SubLength?)
HeaderLength	1	0x0A 10	Previously, when 0x09 was considered a terminator, this field was considered to be the size of the header. But that no longer seems likely. It could also be Datatype 0x0a. Always 10.
TotalItems	4	0x0000 0014 20	Nr of items in this trip file

0x000000	5452 504C 0007 9A03 0A00 0000 1409 0000	TRPL.. .....
----------	---	--------------

Structure of an item

Field	Size (in Bytes)	Sample Value	Description
Initiator	1	0x09	Marks the start of an item. Note: Previously this field was considered to be a terminator of the preceding field. But programming proved it was more likely an initiator.
NameLen	4	0x0000 0009	Length of the name field, immediately following.
Name	Variable	'mImported'	Name of this item. Contains only ANSI characters (1 byte per character). E.G. no international characters, as can be found in string types, that take up 4 bytes for every character.
ValueLen	4	0x0000 0002	Length of the value, including the datatype
Datatype	1	0x07	See the list of datatypes. (0x07 is a Boolean)
Value	Variable	0x00	The size varies according to the datatype. For a Boolean field it is always 1 byte.

0x00130	0000 0005 0300 0049 2009 0000 0009 6D49	.....I .....mI
0x00140	6D70 6F72 7465 6400 0000 0207 0009 0000	mported.....

## Known datatypes:

Note: This list might not be complete. Not all datatypes have been decoded completely.

ID	Type	Size in bytes	Description
0x01	Byte	1	Data type for integer values 0-255. Mostly used for enums. Fields containing only some discrete values. Examples: mDayNumber mRoutePreference mTransportationMode
0x03	Cardinal	4	A cardinal is, in Pascal terms, a 4 byte unsigned integer. Range: 0-4294967295 Examples: mTotalTripTime mDuration mShapingRadius mAttr (Only values seen are 0 and 1, so this could also have been a Byte Field!) A special use case of the type is a Date/Time Field. It defines the nr. of seconds starting from: 1989,12,31,00,00,00,00. It looks like a Unix date/time with a different base date/time. mTripDate mArrival
0x04	Single	4	A float field. (Can hold decimal values) It supports approximately 7 digits of precision in a range from $1.18 \times 10^{-38}$ to $3.4 \times 10^{38}$ . Examples mTotalTripDistance
0x07	Boolean	1	True or False. Examples: mPreserveTrackToRoute mIsDisplayable mAvoidancesChanged mImported etc.
0x08	Version	8	Defines the version. Consist of 2 cardinals Values seen: <b>Major:</b> 0x0000 0004 <b>Minor:</b> 0x0700 0000 for XT and 0x1000 0000 for XT2
0x08	ScPosn	17	Defines the position, in GPS coordinates of a location (Via, or Shaping) Consists of 4 cardinals: <b>Length:</b> Always 0x0000 000C = 12 <b>Unknown:</b> values seen: 0x022EECB4, 0x02F0FF04. When creating a trip file this field can be set to zeroes. Recalculation on the XT(2) will set the correct value. <b>Lat:</b> <b>Lon:</b> See below how to read these numbers.
0x0a	Prefix	Varies	The lists mLocations and mAllroutes have a 'prefix' block. For more info see mLocations and mAllroutes.

0x0b	UdbHandle	Varies	Block describing UdbDirs. See mAllroutes.
0x0e	String	Varies	<p>Type used for strings. Strings are stored as UCS4 chars. 4 bytes for every character.</p> <p>From sample below:</p> <p>0x09: initiator</p> <p>0x 0000 0009: Length of 'mTripName'</p> <p>'mTripName'</p> <p>0x0000 0037: Length of Value + Datatype.</p> <p>0x0e: Datatype</p> <p>0x0034: Nr of bytes for this string.(Divide by 4 to get #chars)</p> <p>0x3200 0000 = '2'</p> <p>0x3000 0000 = '0'</p> <p>Etc.</p> <p>More info:</p> <p><a href="https://docwiki.embarcadero.com/Libraries/Athens/en/System.UCS4Char">https://docwiki.embarcadero.com/Libraries/Athens/en/System.UCS4Char</a></p>

0x7D1A0	0000 0000 0000 0000 0000 0900 0000 096D	.....m
0x7D1B0	5472 6970 4E61 6D65 0000 0037 0E00 3432	TripName...7..42
0x7D1C0	0000 0030 0000 0032 0000 0033 0000 0020	..0...2...3...
0x7D1D0	0000 0053 0000 0055 0000 0044 0000 0020	...S...U...D...
0x7D1E0	0000 0056 0000 0032 0000 002D 0000 0032	...V...2...-...2
0x7D1F0	0000 00	...

ID	Type	Size in bytes	Description
0x80	List	Varies	A list is a type that can contain sub items. A good example is 'mLocations'.

Key	Value
*** Begin mLocations	-----
Size	Size: 18157
DataType	128
LocationCount	47
*** Begin location header	Begin 2023 SUD V2-2
ID	LCTN
Size	364
DataType	10
Count	9
--- End location header	-----
mAttr	Via point
mIsDFSPoint	False
mDuration	-1
mArrival	24-8-2023 09:00:00
mScPosn	Unknown: 0x02EECB4, Lat: 48.29037, Lon:
mAddress	48,29037 8,21907
mIsTravelapseDestination	False
mShapingRadius	0x80000000
mName	Begin 2023 SUD V2-2
--- End location	-----
*** Begin location header	2023 SUD V2-2_005 Km
ID	LCTN
Size	368
DataType	10
Count	9
--- End location header	-----
mAttr	Shaping point
mIsDFSPoint	False
mDuration	-1
mArrival	31-12-1989 01:00:00

The locations are preceded by a block describing the total block. (\*\*\*) Begin mLocations)

Then for every location a block describing 1 location (\*\*\*) Begin location header)

The marked portions are all basic data type items.

mAttr = Cardinal

mIsDFSPoint = Boolean

...

mName = String

## How to read the Lat/Lon values.

The Lat Lon values can be found in mScPosn and UdbDir (Sub item of mUdbDataHndl) fields.

Sample:

Key	Value
mArrival	31-12-1989 01:00:00
mScPosn	Unknown: 0x022EECB4, Lat: 48.09557, Lon: 8.20745
mAddress	2023 SUD V2-2_044 Km
mIsTravelapseDestination	False
mShapingRadius	0x80000000
mName	2023 SUD V2-2_044 Km
--- End location	-----

Address	Value	Comment
0x00B80	0009 0000 0009 6D44 7572 6174 696F 6E00	.....mDuration.
0x00B90	0000 0503 FFFF FFFF 0900 0000 086D 4172	....ÿÿÿÿ.....mAr
0x00BA0	7269 7661 6C00 0000 0503 0000 0000 0900	rival.....
0x00BB0	0000 076D 5363 506F 736E 0000 0011 0800	...mScPosn.....
0x00BC0	0000 0CB4 EC2E 0205 8833 22FC 1ED6 0509	...i... 3"ü.ö..
0x00BD0	0000 0008 6D41 6464 7265 7373 0000 0053	...mAddress...S
0x00BE0	0E00 5032 0000 0030 0000 0032 0000 0033	..P2...0...2...3
0x00BF0	0000 0020 0000 0053 0000 0055 0000 0044	... ..S...U...D
0x00C00	0000 0020 0000 0056 0000 0032 0000 002D	... ..V...2...-
0x00C10	0000 0032 0000 005F 0000 0030 0000 0034	...2..._...0...4

In this sample:

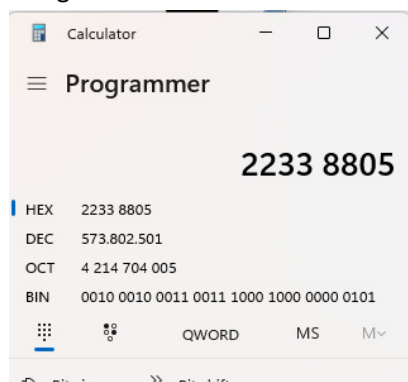
0x0588 3322 corresponds to Lat: 48.09557. Read the Hex value from Right to Left. (Little-Endian)

- 1) Convert the Hexadecimal value to Decimal = 573802501 (Using Windows calculator in Programmer mode)
- 2) Divide by 4294967296 ( $2^{32}$ ) = 0.13359880563803
- 3) Multiply by 360 = 48.0955700296909
- 4) Round to 6 digits = 48.09557

0xFC1E D605 corresponds to Lon: 8.20745

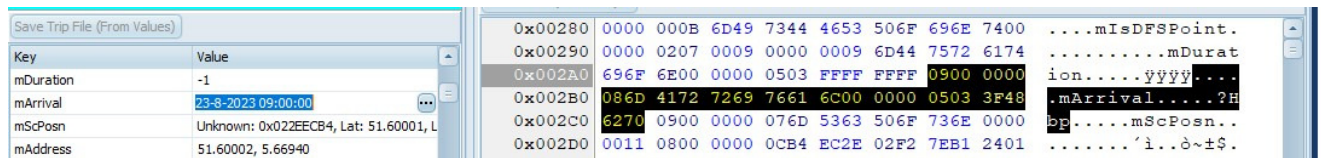
- 1) Convert the Hexadecimal value to Decimal = 97918716
- 2) Divide by 4294967296 ( $2^{32}$ ) = 0,0227984776720405
- 3) Multiply by 360 = 8.20745196193457
- 4) Round to 6 digits = 8.20745

Using Windows calculator. Remember the hex values have to be read from Right to Left!



## How to read the DateTime values.

The date time format as used in mArrival can be decoded like this.



The Hex value 0x3F48 6270 corresponds to August 23, 2023 09:00:00 Basically this value defines the nr. of seconds since 1989/12/31 00:00:00

- |    |                         |   |
|----|-------------------------|---|
| 1) | Convert hex to integer. | Again use Windows Calculator.   |
|    | 1061708400              |   |
| 2) | Add the constant        | This is the nr. of seconds on 1989/12/31 00:00:00 since 1970/01/01 00:00:00 |
|    | 631065600               |   |
| 3) | Resulting in            | Nr of seconds since 1970/01/01 00:00:00                                     |
|    | 1692774000              | August 23, 2023 09:00:00 = Unix Date Time                                   |

For converting seconds to date time you can use the website: <https://www.epochconverter.com/>

## Convert epoch to human-readable date and vice versa

1692774000    **Timestamp to Human date**    [\[batch convert\]](#)

Supports Unix timestamps in seconds, milliseconds, microseconds and nanoseconds.

Assuming that this timestamp is in **seconds**:

**GMT:** Wednesday 23 August 2023 07:00:00

**Your time zone:** woensdag 23 augustus 2023 09:00:00 GMT+02:00 DST

**Relative:** A year ago

Note: You may notice a difference (usually an hour) that is likely caused by daylight savings.

On the XT you may see 08:00, while in Basecamp/Tripmanager it is 09:00

## Items seen in trip files on the XT

Name	DataType	Description
mPreserveTrackToRoute	0x07	Set to true for trips created from a track.
mParentTripId	0x03	See mParentTripName
mDayNumber	0x01	?? Always 0xFF 255. May be used for sorting?
mTripDate	0x03	?? Always 0xFFFF FFFF
mIsDisplayable	0x07	If set to False deletes the trip from the device at next start.
mAvoidancesChanged	0x07	?? Always False. May be changes after modifying avoidances?
mIsRoundTrip	0x07	?? Always False
mParentTripName	0x0E	Can be used together with mParentTripId to group the trips.
mOptimized	0x07	?? Always False
mTotalTripTime	0x03	Contains Time in seconds of trip. After recalculation by XT
mImported	0x07	Set to False to fix the RUT behaviour.
mRoutePreference	0x01	0x00 = Faster Time 0x01 = Shorter Distance 0x04 = Direct 0x07 = Curvy Roads
mTransportationMode	0x01	0x01 = Automotive 0x09 = MotorCycling 0x0A = Off Road
mTotalTripDistance	0x04	Contains Distance in meters of trip. After recalculation by XT.
mFileName	0x0E	File name on device. 0:/.System/Trips/<1234567890>.trip If a file is saved by importing. It is the timestamp of saving. See 'How to read the DateTime values'.
mLocations	0x80	A list of Begin, Via / Shaping points and End point. See 'mLocations'
mPartOfSplitRoute	0x07	?? Always False
mVersionNumber	0x08	Apparently the version nbr. Only value seen for XT is: 0x04000000 / 0x00000007
mAllRoutes	0x80	Is the result of the calculation. It contains a list of mUdbDataHandle and UdbDir. See 'mAllRoutes'
mTripName	0x0E	TripName. This is the name of the route in BaseCamp.

## Items seen in trip files on the XT2

Name	DataType	Default value	Description
mGreatRidesInfoMap	0x0C	0x0000 0000	Note 2
mAvoidancesChangedTimeAtSave	0x03	0x0000 0000	Note 3
mTrackToRouteInfoMap	0x0C	0x0000 0000	Note 2
mlsDisplayable	0x07		If set to False deletes the trip from the device at next start.
mlsDeviceRoute	0x07	False	??
mDayNumber	0x01		?? Always 0xFF 255. May be used for sorting?
mTripDate	0x03		?? Always 0xFFFF FFFF
mOptimized	0x07		?? Always False
mTotalTripTime	0x03		Contains Time in seconds of trip. After recalculation by XT
mTripName	0x0E		TripName. This is the name of the route in BaseCamp.
mVehicleProfileGuid	0x0E	'00000000-0000-0000-0000-000000000000'	Note 4
mParentTripId	0x03		See mParentTripName
mlsRoundTrip	0x07		?? Always False
mVehicleProfileName	0x0E	'zūmo Motorcycle'	Note 4
mAvoidancesChanged	0x07		?? Always False. May be changes after modifying avoidances?
mParentTripName	0x0E		Can be used together with mParentTripId to group the trips.
mVehicleProfileTruckType	0x01	0x07	Note 4
mVehicleProfileHash	0x03	0x0000 0000	Note 4
mRoutePreferences	0x80	Loc cnt * 0001	Note 5
mlmported	0x07		Set to False to fix the RUT behaviour.
mFileName	0x0E		File name on device. 0:/System/Trips/<1234567890>.trip If a file is saved by importing. It is the timestamp of saving. See 'How to read the DateTime values'.
mExploreUuid	0x0E	'00000000-0000-0000-0000-000000000000'	Note 6
mVersionNumber	0x08		Apparently the version nbr. Only value seen for XT2 is: 0x04000000 / 0x00000010
mRoutePreferencesAdventurousHillsAndCurves	0x80	Loc cnt * 0x0001	Note 5
mTotalTripDistance	0x04		Contains Distance in meters of trip. After recalculation by XT.
mVehicleId	0x03	0x01	Note 4
mRoutePreferencesAdventurousScenicRoads	0x80	Loc cnt * 0001	Note 5
mAllRoutes	0x80		Is the result of the calculation. It contains a list of mUdbDataHndle and UdbDir. See 'mAllRoutes'
mRoutePreferencesAdventurousPopularPaths	0x80	Loc cnt * 0001	Note 5
mPartOfSplitRoute	0x07		?? Always False
mRoutePreference	0x01		0x00 = Faster Time 0x01 = Shorter Distance 0x04 = Direct 0x07 = Curvy Roads
mShowLastStopAsShapingPoint	0x07	False	??
mRoutePreferencesAdventurousMode	0x80	Loc cnt * 0001	Note 5
mTransportationMode	0x01		0x01 = Automotive 0x09 = MotorCycling 0x0A = Off Road
mLocations	0x80		A list of Begin, Via / Shaping points and End point. See 'mLocations'

The default value is used when creating a trip file.

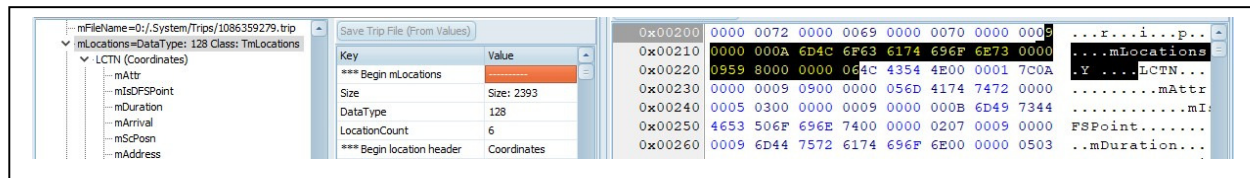


## Notes on the XT2 items

1. **mPreserveTrackToRoute** does not appear on the XT2
2. **mGreatRidesInfoMap** and **mTrackToRouteInfoMap** introduce 0x0c as datatype. The only value seen is 0x0000 0000.  
This value could also have been represented by datatype 0x03 (Cardinal) So I suspect that these items can have more data. Maybe only activated when using Tread?
3. **mAvoidancesChangedTimeAtSave** looks like a date time.
4. **mVehicleProfileGuid**, **mVehicleProfileName**, **mVehicleProfileTruckType**, **mVehicleProfileHash** and **mVehicleId** look related and appear to define the vehicle in use. I have no idea if the chosen defaults make sense.
5. **mRoutePreferences**, **mRoutePreferencesAdventurousHillsAndCurves**, **mRoutePreferencesAdventurousScenicRoads**, **mRoutePreferencesAdventurousPopularPaths** and **mRoutePreferencesAdventurousMode** have datatype 0x80, suggesting that they are lists. The first 4 bytes define the nr of items, followed by 2 bytes per route section. It appears to be related to the nr of **mUdbHandles**. So if you have **4 mUdbHandles**, the list is like this:  
**0x0000 0004 0001 0001 0001 0001**  
0001 = Faster time, 0002 = Shorter distance etc.?
6. **mExploreUuid**. An Id of the explore DB? I hope a value of all zeroes is accepted.

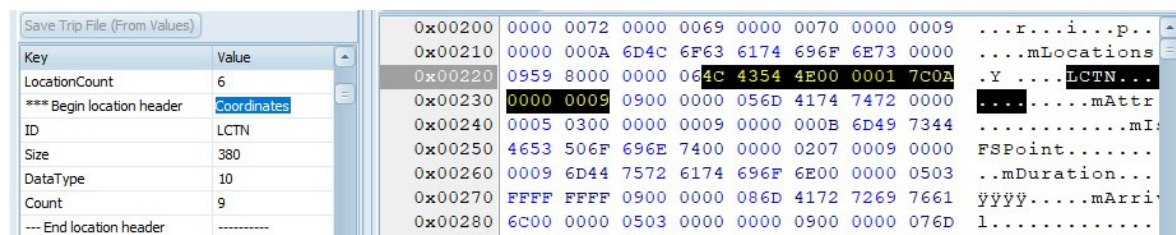
## mLocations

mLocations itself only contains a Size and the location count.



Sample values: **Size:** 0x0959 = 2393, **DataType** 0x80, **LocationCount:** 0x06

Every Location starts with a Location Header



Sample values: ID: 'LCTN', Size: 0x0000 017C = 380 DataType: 0x0A, Count: 0x0000 0009

Next a list of 'Count = 9' items: (For XT2 the count is usually 10)

Item	DataType	Description
mShapingCenter	0x08	XT2 specific. Only value seen: 0x0000 0008 0000 0080 0000 0080
mAttr	0x03	0x0000 0000 = Via Point (Also used for Begin and End point) 0x0000 0001 = Shaping Point
mIsDFSPoint	0x07	?? Always False
mDuration	0x03	?? Always 0xFFFF FFFF
mArrival	0x03	Used as 'Departure Date/Time'. The first Location determines the sort order within the Trip Planner See: 'How to read Date Time values'
mScPosn	0x08	Specifies the LAT/LON values. See 'Known datatypes 0x08'
mAddress	0x0E	A description of the point that the XT uses for displaying. A known issue is that Via/Shaping sometimes get renamed on the XT. Possible fixes: - Setting the subclass in the GPX to 0x00ff - Setting a value for this item.
mIsTravelapseDestination	0x07	?? Always False
mName	0x0E	The name of the Location

## mAllroutes

This list contains the result of the calculation. This data has only partly been decoded, but it is likely that it also contains references to Map data. Even if it would be possible to decode it completely then it still would not be very useful, because the logic needed to calculate the routes is not available.

- If a calculated route from Basecamp is sent to the XT(2) then this list contains the unmodified route. Provided a few conditions are met, like the same map, same transportation mode etc.
- If a route is sent without all the Ghost Points, Subclasses etc. the XT will recalculate the route upon importing, and this list will contain the result of the calculation. For example routes created by MRA.
- If, for some reason, the XT(2) recalculates the route, the result will likely not be the same as initially created in BaseCamp/MRA.

It is therefore desirable to be able to compare the mAllroutes with the original route. What has been decoded is sufficient to achieve that.

Start of mAllRoutes:

Save Trip File (From Values)		0x00A40	7369	6F6E	4E75	6D62	6572	0000	0009	0800	sionNumber.....
Key	Value	0x00A50	0000	0407	0000	0009	0000	000A	6D41	6C6C	.....mAll
*** Begin AllRoutes		0x00A60	526F	7574	6573	0002	5D11	8000	0000	0400	Routes..]. ....
Size	154897	0x00A70	0000	0000	0016	630A	0000	0001	0900	0000	.....c.....
DataType	128	0x00A80	0C6D	5564	6244	6174	6148	6E64	6C00	0016	.mUdbDataHndl...
UdbHandleCount	4	0x00A90	490B	0000	1644	FFFE	3805	0000	0000	0000	I....Dyp8.....
*** Begin UdbPrefix		0x00AA0	0000	0000	0000	0000	0000	0000	0000	0000	.....

**Size:** 0x0002 5D11 = 154897

**DataType:** 0x80 = 128

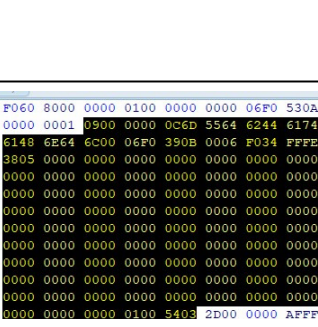
**UdbHandleCount:** 0x0000 0004 = 4

Note: For every trip section an UdbHandle is needed. So if you only have a Begin and End, there is only 1 UdbHandle. If you add 1 via/shaping point, there are 2 UdbHandles, for 2 via/shaping points there are 3 Udbhandles. Etc.

Each UdbHandle starts with a prefix:

Save Trip File (From Values)		0x00A40	7369	6F6E	4E75	6D62	6572	0000	0009	0800	sionNumber.....
Key	Value	0x00A50	0000	0407	0000	0009	0000	000A	6D41	6C6C	.....mAll
*** Begin UdbPrefix		0x00A60	526F	7574	6573	0002	5D11	8000	0000	0400	Routes..]. ....
Unknown	0x00000000	0x00A70	0000	0000	0016	630A	0000	0001	0900	0000	.....c.....
PrefixSize	5731	0x00A80	0C6D	5564	6244	6174	6148	6E64	6C00	0016	.mUdbDataHndl...
DataType	10	0x00A90	490B	0000	1644	FFFE	3805	0000	0000	0000	I....Dyp8.....
HandleId	1	0x00AA0	0000	0000	0000	0000	0000	0000	0000	0000	.....
*** Begin UdbHandle		0x00AB0	0000	0000	0000	0000	0000	0000	0000	0000	.....

Sample values: **Unknown:** 0x0000 0000, **PrefixSize:** 0x0016 630A = 5731, **DataType:** 0x0A, **HandleId:** 0x0000 0001 (Only value seen)

0x0b, **UdbHandleSize**: 0x00

the size of Unknown3 = 1288 b

the size of Unknown3 = 1448

ut the correct size for Unknown

- $$05D8 - 0x538 = 0xA0/160, 14$$

l in the GPX. The first 2 byte

s in Garmin GPX.pdf'

n values'

n values'

oDir. Contains road names.

the SubClass.

0x053B0	0000	0000	0000	0000	0000	0
0x053C0	0000	8F67	AB01	8592	9F00	1
0x053D0	0C00	00B3	5622	00C4	D705	5
0x053E0	0000	0000	0000	0000	0000	0
0x053F0	0000	4200	0000	3200	0000	3
0x05400	0000	2000	0000	4800	0000	6
0x05410	0000	6500	0000	6800	0000	6
0x05420	0000	6300	0000	6800	0000	7
0x05430	0000	7200	0000	6100	0000	7
0x05440	0000	6500	0000	0000	0000	0
0x05450	0000	0000	0000	0000	0000	0