



Updated definition of a unique Transit Identifier for the MDB

prepared by: J. Portell, U. Bastian, C. Fabricius, O. Martínez,
J. Castañeda, P. Vallès
approved by: DPAC
reference: GAIA-C3-TN-UB-JP-011-05
issue: 5
revision: 0
date: 2018-08-17
status: Issued

Abstract

In this note we define the concept of “TransitId”, the unique identifier of a Gaia observation used in the MainDB by several DPAC processes. We describe here its definition, which is based on the OBMT time scale. The present issue 5 describes the “human-readable” format of TransitIds.

Document History

Issue	Revision	Date	Author	Comment
5	0	2018-08-17	JP	Added “human-readable” format of TransitIds in Section 3. Added table of contents. Changed ‘Cycle’ by ‘Development Cycle’ in some places, to avoid confusion with Data Reduction Cycles. Moved old sections 3 and 4 as Appendixes.
4	1	2010-12-09	HS+ CF+ JP	“CU2-like” TransitIds named as “TrueValuesTransitIds”, to avoid confusion. Added a note at the end mentioning that the OBMT-OBT offset just needs to be multiple of 204.8 μ s during development (not operations).
4	0	2010-11-25	JP+ PV	Added a new section indicating the considerations on the validation of IDT (and other) data. More specifically, on the OBMT-OBT offset issue, and the TransitId conversion between “non-CU2” and CU2. Also, some correction in the text (OBT was said in some places, instead of OBMT). References to BAS-030 added. Some clarifications added. Some references corrected (JC-014 and CF-011).
3	0	2009-01-15	JP+ BAS	Actions closed after confirmations from Astrium and JdB (by indicating us the adequate documents and sections). OBMT better defined, and OBMT50 introduced (in order to avoid confusions). Added formulae for OBMT50.
2	0	2008-03-06	JC+ JP	Correction to the TransitId formula of page 8. There were ‘and’ operations while they had to be ‘or’. Also, addition of a second formula explaining how the TransitId could be calculated directly from the OBT data included in the TM plus some adequate offset considering OBT resets.
1	0	2008-02-06	BAS+ JP	Minor corrections to Discussion of proposal and Actions. Document code corrected. First version.
D	1	2008-02-05	JP	Updates and corrections after comments from UB, CF and JC. Corrections only to the description of coordinates and references, not to the definition of TransitId in itself.
D	0	2008-01-28	JP	First draft

This document is available in SVN here: http://gaia.esac.esa.int/dpacsvn/DPAC/CU3/docs/General/JP-011_NewTransitId/

Contents

1	Introduction	4
1.1	Old TransitId definition	4
2	Redefinition of the TransitId	5
2.1	Differences between TM models	5
2.2	Considerations and definitions	8
2.3	The proposal	9
2.4	Discussion of the proposal	11
3	Human-readable format of TransitID	12
4	References and acronyms	13
4.1	References	13
4.2	Definitions, acronyms, and abbreviations	14
App. A: Validation using CU2 reference data: Conversion of TransitIds (OBT/OBMT)		16
App. B: Status of actions in this issue		18

1 Introduction

A unique identifier (or label) for all the Gaia observations is required by most DPAC processes. Such identifier, called “TransitId”, is required for the cross-match of different measurements of a same observation (that is, astrometric, photometric and spectroscopic), and is also required in the cross-match process that links observations to sources, just to mention a couple of applications.

1.1 Old TransitId definition

The previous definition of TransitId (BAS-017) was based on the reference systems and conventions described in BAS-003, Issue 6, Revision 1 (2007-07-05). The previous contents of the TransitId included the Along-scan Pixel Coordinate (APC), with one-TDI resolution (982.8 μ s):

- APC (40 bits): Along-scan Pixel Coordinate of the object detection in the SM, in TDI periods since a given reference epoch, following GAIA-UL-009 and BAS-003.
- The FOV index (1 bit), indicating a detection in SM1 (FOV1) or SM2 (FOV2).
- The CCD Row (3 bits, 1...7) where the measurement has been acquired.
- The across-scan (AC) pixel coordinate of the SM window (11 bits, 1...1966), following BAS-003.

This definition led to a unique TransitId of 55 bits, thus fitting in 64-bit types (and thus being of handy use in the DPAC processes). It has been used in the past in several processes like CU2-GASS, CU3-IDT and CU3-IDU (including XM).

In Development Cycle 4, CU2-GASS and CU3-IDT moved to the new telemetry model according to the flight-realistic definition given by Astrium (GAIA.ASF.ICD.SAT.00011). This new TM Model still includes in each observation packet the several data fields required for the TransitId definition, that is, an observation time, the FOV, the CCD Row and an AC coordinate. The problem is that both the observation time and the AC coordinate use a different reference system than the one currently assumed by the TransitId (defined in BAS-003). Additionally, such coordinates are referred to the AF1 readout window, instead of the SM transmitted window. Hence, a formal redefinition of the TransitId was imperative in order to avoid confusion in the DPAC processes.

2 Redefinition of the TransitId

2.1 Differences between TM models

The following are the main differences between the previous TM model and the new one:

- Instead of an APC of 40 bits in TDI periods ($982.8\mu\text{s}$) referred to the SM transmitted window, we now have a ‘ref_acquisition_time’ of 64 bits in OBT (On-Board Time¹), that is, in periods of the on-board master clock (50 nanoseconds), furthermore referred to a given reference pixel of the AF1 acquired window. Note that the the AF1 acquired window may differ from the transmitted window.
- Instead of a single bit of FOV, indicating ‘FOV1’ or ‘FOV2’, we now have a 2-bit FOV data field that can also indicate that a source has been propagated in the focal plane “with NULL motion” (GAIA.ASF.ICD.SAT.00011), either detected by SM1 or SM2. Note that such “null motion” is used for the Virtual Objects (VO) acquired by the VPU, used e.g. for background determination.
- Instead of an across-scan pixel coordinate of the SM window, we now have an ‘ac’ coordinate of a given reference pixel of the AF1 window. Additionally, this data field is now 12 bits instead of 11 (it seems that Astrium gives the possibility of having an AF1 window within the “post-scan pixels” section, which reaches the pixel coordinate 2111).

Fig. 1, Fig. 2 and Fig. 3 illustrate the differences in the conventions used up to Development Cycle 3 (according to BAS-003) and those to be used for the new TransitId (according to Astrium). We must note that the window classes indicated here are those indicated by Astrium (for a summary of the sampling scheme see GAIA.ASF.TCN.PLM.00137, section 8). In Fig. 1 we can see the Class 0 window of 18×6 samples (1×2 pixels each) acquired for AF1, also indicating the symmetry axes and the AC and AL axes. Regarding AL references, we can see that the current conventions use a reference pixel in the first line of pixels read, while Astrium uses the “latest” of the two central lines read. Regarding AC references, BAS-003 uses the “lowest” pixel of the “highest” sample, while Astrium uses one of the two central pixels depending on the AC parity (as we will explain hereafter).

The case of Class 0 is somehow simpler, since all of the samples read for AF1 are transmitted. In Class 1 and Class 2, on the other hand, the VPU reads more samples than those actually transmitted. Fig. 2 illustrates the Class 1 case, where we can see the darker area that indicates the samples actually transmitted. The VPU runs a simple centroiding algorithm that determines the best fitting of such samples (so that the transmitted window is well centred in the PSF).

¹Please note that we will use OBMT, not OBT, for the definition of the TransitId — see the explanation hereafter

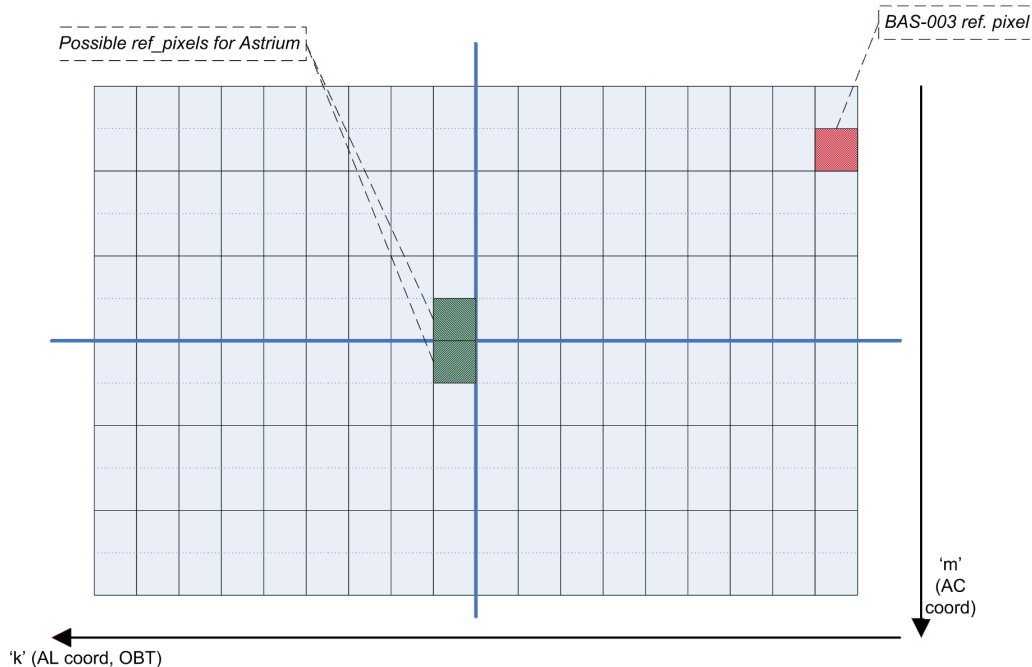


FIGURE 1: AF1 window acquired (identical to the transmitted one) in Class 0, with the reference pixels used in of both conventions (BAS-003 and Astrium).

The samples transmitted are indicated by the “af1_centrimg” flag, included in the telemetry packet headers. The figure reveals that BAS-003 uses a reference pixel based on the transmitted window, while Astrium keeps referring to the central pixels of the acquired window.

Finally, Fig. 3 illustrates the case for Class 2, which is the same as for Class 1 except for a smaller amount of samples acquired (and transmitted) in the AL direction. This example represents an AF1 window that has been predicted not very well in the AL direction (e.g., due to a moving object), and hence the AF1 Centring corrects it. Here we also illustrate the clarification for the AC coordinates. The AF1 samples (1×2 pixels each) are “phased” in the CCD in the AC direction, that is, they are always acquired starting at an even m coordinate, as indicated in the figure. Then, the parity of the ac coordinate indicated in the telemetry packet will indicate if the AC reference pixel is the “higher” or “lower” pixel of the centre of the window. Following the example of the figure, $ac = 27$ will indicate that the reference pixel is the “upper” central pixel, while $ac = 28$ would indicate the “lower” central pixel. In other words, odd ac values will indicate an “upper” (lower m coordinate) reference, while even ac values will indicate a “lower” (higher m coordinate) reference.

We must note that all these clarifications regarding the reference pixels are not strictly required for the definition of the TransitId in itself. We just include them here for the sake of completeness, and in order to make clear which coordinates are actually indicated by the TransitId components.

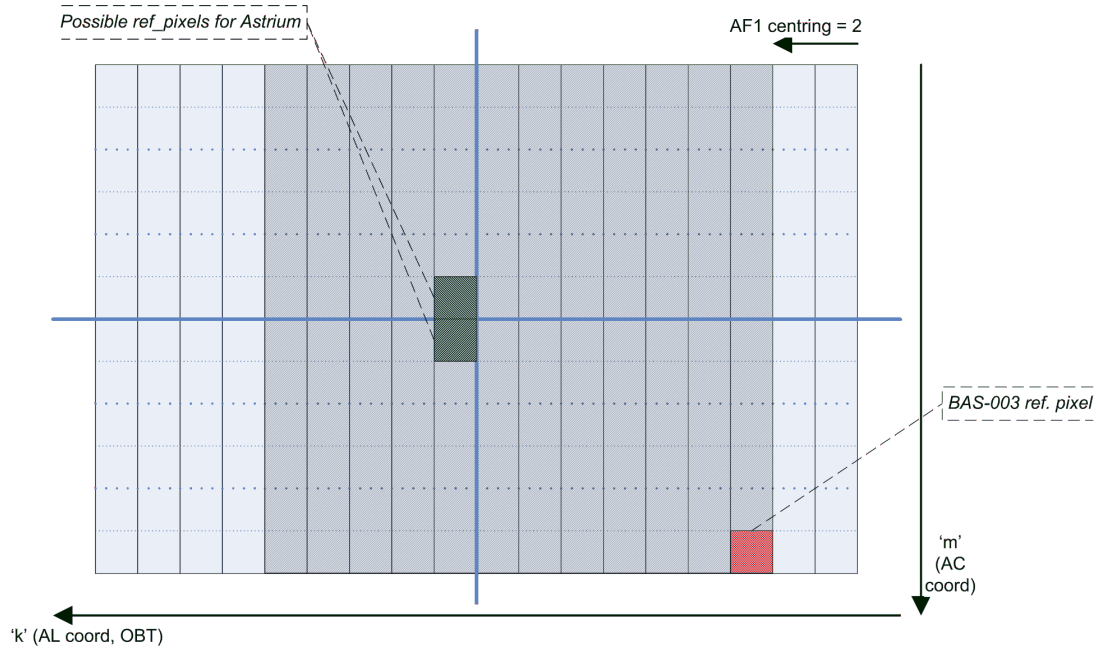


FIGURE 2: AF1 windows acquired (background) and transmitted (gray) in Class 1.

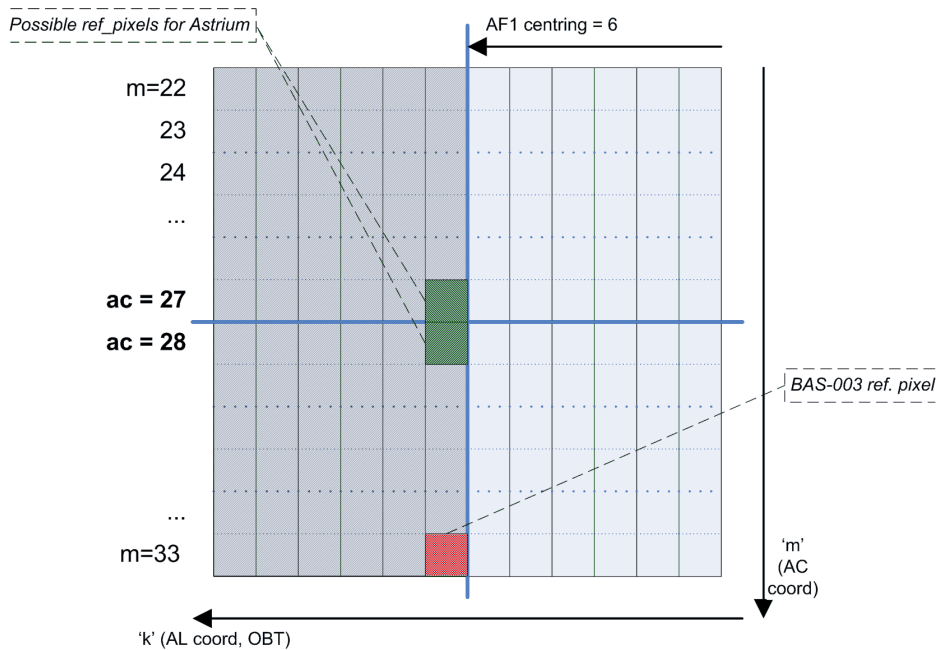


FIGURE 3: AF1 windows acquired (background) and transmitted (gray) in Class 2.

2.2 Considerations and definitions

The objective is to define a new TransitId as similar as possible to the previous definition (BAS-017). Also, it is obviously interesting that this identifier can fit in 64 bits, so that we can store it in a single Java variable (namely, a ‘long int’). In order to make it possible, we should discard some bits of the largest data field implied, that is, the OBMT. Actually, we cannot discard any bit of the rest of fields.

For this, we assume that the ‘ref_acquisition_time’ contained in the telemetry packets is, actually, a simple counter of the OBT periods (of 50ns each) since a given reference epoch (or since a given reset of the master clock). It has been confirmed by Astrium. It means that we can discard the 12 least-significant-bits (LSB) of ‘ref_acquisition_time’ and it will still have a resolution better than 1 TDI — more specifically, it will have a resolution of $204.8\mu\text{s}$.

Additionally, and continuing with the fact that ‘ref_acquisition_time’ is a simple OBT counter, even with an extended mission of 28 years (actually much more than any possible extension of the mission), the 10 most-significant-bits (MSB) of OBT will always be zero, and hence we can also discard them. It has also been confirmed by Astrium.

Finally, we must take into account that OBT can be reset during the mission, as indicated in GAIA.ASF.SP.PLM.00034. It means that the contents of ‘ref_acquisition_time’ received on ground will repeat sometime, which could lead to duplicated TransitId values if such resets are not corrected. We assume (and it has also been confirmed) that such resets will be known, either because they will be requested by an operator on-ground, or because the on-board systems will indicate it in some way through the downlink. Additionally, the OBT instant when a reset has occurred will be known, and thus it will make possible to add an adequate offset to the value of ‘ref_acquisition_time’ in the first on-ground data processing stages.

These considerations lead us to the concept of **On-Board Mission Timeline (OBMT)**, which is a non-physical timescale based on the on-board clock, including corrections due to the clock resets so that it takes monotonically increasing values. The conversion between OBMT and other physical timescales (such as TCB) is not trivial and will be addressed in other documents. The concept of “reference epoch” is not directly applicable to OBMT neither. More details can be found in BAS-030, but here we summarize the major points.

The OBMT will be determined by the MOC-SOC Interface Task (MIT), and that all IDT outputs corresponding to observational data will use the OBMT time scale (not OBT). It is worth noting that the corrections to the on-board clock resets (that is, the OBT-OBMT offset determined by MIT) will be integer multiples of 50 ns^2 . The resolution of OBMT data fields is 1 nanosecond unless explicitly indicated, that is, the least-significant bit of an OBMT integer data field will

²Actually, the offset should be an integer multiple of $204.8\mu\text{s}$, for the reasons mentioned hereafter regarding TransitId conversion for validations.

correspond to 1ns. This is adequate for values that require a high precision, such as observation times obtained by a centroiding algorithm. On the other hand, some data fields (mainly those corresponding to on-board time measurements) may not require such a high time resolution. This is the case of the window acquisition times, the time labelling of on-board events (such as charge injections), or the time component of the TransitId. In these cases a reduced-resolution version of the OBMT can be used, using the same resolution than the on-board clock (50ns). This reduced-resolution version of OBMT will be called **OBMT50** (thus explicitly indicating the 50ns resolution).

2.3 The proposal

Given the previously indicated considerations and definitions, the following is the format for the new TransitId. It will be composed by the concatenation of the following data fields:

- OBMT (On-Board Mission Timeline, 42 bits) with a resolution of $204.8\mu\text{s}$. This is, the result of discarding the 10 MSB and the 12 LSB of OBMT50. It leads to a constantly increasing OBMT counter with a step of $204.8\mu\text{s}$ and a range of up to 28.5 years. Such time is referred to the Astrium ‘ref_sample’ of the window in AF1.
- FOV (Field Of View, 2 bits), as received from the TM. That is, ‘0’ means ‘preceding’ and ‘1’ means ‘following’. Values ‘2’ and ‘3’ indicate a ‘null’ AC propagation for objects detected in SM1 or SM2, respectively — corresponding to Virtual Objects (VO).
- CCD Row (3 bits), as received from the TM. That is, ranging from 1 to 7. As a clarification, the four rows containing the RVS CCDs are numbered from 4 to 7.
- Ac (Across-scan coordinate, 12 bits), as received from the TM. That is, ranging from 0 to 2112, referring to the AF1 AC window position (associated to Astrium ‘ref_sample’). We remind that Astrium includes the “post-scan” section in the *ac* range, and this is the reason why we need one more bit than before.

This definition leads to a TransitId of 59 bits, thus fitting in a 64-bit type and leaving the 5 MSB as ‘spare bits’ — which may be used in the future if necessary. For example, some HEALPix index could be assigned to such bits, thus having time and space information for any observation in a single value. Nevertheless, we should be careful about their usage: any data field assigned to them would be used as the main sorting rule in a TransitId list.

Fig. 4 illustrates this definition (bottom panel), together with the previous TransitId definition of BAS-017 (central panel) and the 64-bit range (top panel).

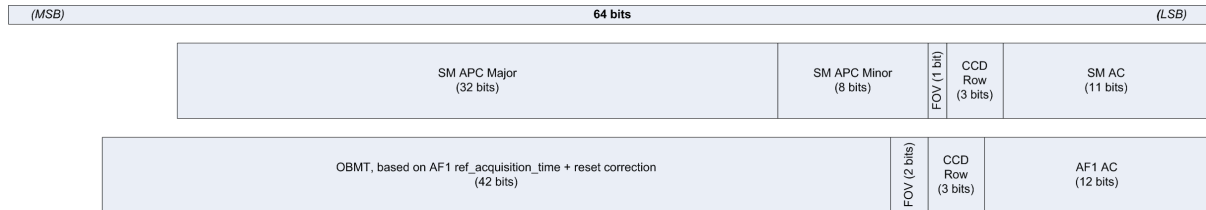


FIGURE 4: Contents of the old (central panel) and new (bottom panel) Transit Identifiers.

The several features of a detection can be easily retrieved from a TransitId value using bit masks and shifts:

- $OBMT(204.8\mu s) = (long)(TransitId \gg 17) \& (0x0000\ 03FF\ FFFF\ FFFF)$
- $OBMT50 = (long)(TransitId \gg 5) \& (0x003F\ FFFF\ FFFF\ F000)$
- $FOV = (byte)(TransitId \gg 15) \& 0x03$
- $CCDRow = (byte)(TransitId \gg 12) \& 0x07$
- $AC = (short)(TransitId) \& 0x0FFF$

Additionally, a TransitId can be calculated from these data fields as following (assuming that each data field is guaranteed to be within the accepted ranges):

$$\begin{aligned}
 TransitId &= ((long)OBMT(204.8\mu s) \ll 17) \\
 &|((long)FOV \ll 15) \\
 &|((long)CCDrow \ll 12) \\
 &|((long)AC)
 \end{aligned}$$

Or, alternatively:

$$\begin{aligned}
 TransitId &= ((long)(OBMT50 \& 0x003F\ FFFF\ FFFF\ F000) \ll 5) \\
 &|((long)FOV \ll 15) \\
 &|((long)CCDrow \ll 12) \\
 &|((long)AC)
 \end{aligned}$$

Finally, the following formula illustrates how the TransitId could be calculated directly from the on-board time plus the adequate OBT-OBMT offset determined by MIT. That is, implicitly calculating OBMT50 within the formula. The offset is supposed to be coded with the same resolution as OBT (that is, 50ns):

$$\text{TransitId} = ((\text{long})(AF1_ref_acquisition_time + \text{ObtObmtOffset50}) \gg 12) \ll 17) \\ |((\text{long})FOV \ll 15)|((\text{long})CCDrow \ll 12)|((\text{long})AC)$$

2.4 Discussion of the proposal

The following are the key advantages of this proposal:

- It keeps in line with the existing approach, so minimal changes should be required in the many pieces of DPAC code using it (bit masks, etc).
- It still gives an indication of the observation time, so sorting a list of observations according to their TransitId will be equivalent to sorting them according to their observation time.
- It still fits a 64-bit long integer, so with such a single Java value we will have all these relevant features of any observation.
- Considering that the OBT resets will be corrected (with the adequate offset) at the MIT (prior to IDT), the uniqueness of the TransitId is guaranteed. Even in case two detections (one in SM1 and the other in SM2) lead to the same propagated pixel (both in AL and AC) in AF1, the value of FOV will be different, and hence the TransitIds will also differ.
- Astrium “guarantees” that such an identifier is really unique — except for possible OBT resets, which will be counteracted DPAC-internally, see above — and implicitly included in SP1, SP2 and SP3 (although in those cases they use the full 64-bit OBT). It is indicated in GAIA.ASF.TCN.PLM.00137, section 6.4.
- This proposal leaves the whole BAS-003 concepts of window datation and of the along-scan pixel coordinate (APC) untouched. Just a simple formula shall be applied in order to retrieve the APC from what is in the TM.

3 Human-readable format of TransitID

At the CU3 Plenary Meeting #7, held in Vilanova de Gaia (13-15 June 2012), C. Fabricius proposed an alternative representation of the TransitIDs which is more “human-friendly”.

It is based on the concept of *OBMT Revolutions*, which is the OBMT value in units of 6 hours (OBMT divided by $21600E9$ nanoseconds). That is, approximately, one revolution of Gaia around its spin axis. Such “human-readable” format, thus, shows the number of revolutions as an integer value, followed by the *remainder* of the revolutions (in steps of $204.8\mu s$), and finally followed by the rest of information contained in the TransitID.

Thus, the format is the following:

RRRRR–MMMMMMMM–F–C–AAAA

where:

- RRRRR (5 digits) indicates the Revolutions, from 0 to 41699 (28.5 years).
- MMMMMMMM (9 digits) indicates the Remainder of the revolution, from 0 to 105468749 (21599.9997952 seconds, or 6 hours minus $204.8\mu s$).
- F (1 digit) is the FOV, from 1 to 4, with values 3 and 4 indicating “no AC motion” (for VOs).
- C (1 digit) is the CCD Row, from 1 to 7.
- AAAA (4 digits) is the AC Coordinate, from 0014 to 1979 pixels.

Thus, for example, TransitID 41001999429850823 becomes 2966-000117720-2-5-1735.

This format is useful to quickly see the OBMT revolution, the FOV (and whether a TransitID belongs to a VO or not), the CCD Row and the AC Coordinate.

GaiaTools provides the `TransitIdParser.toString()` method³ to generate this format from a given TransitID value. Additionally, <https://gaia.esac.esa.int/decoder/> provides TransitID decoding tools.

³This is in the `gaia.cul.tools.satellite.telemetry` package.

4 References and acronyms

4.1 References

- [GAIA.ASF.ICD.SAT.00011], Airbus DS, 2015, *VPU to PDHU Telemetry Interface Control Document*,
GAIA.ASF.ICD.SAT.00011,
URL <http://www.rssd.esa.int/cs/livelihood/open/2777572>
- [BAS-017], Bastian, U., 2006, *On-board AF/RVS cross-match indicator; transit label in MDB*,
GAIA-C3-TN-ARI-BAS-017,
URL <http://www.rssd.esa.int/cs/livelihood/open/2795465>
- [BAS-003], Bastian, U., 2007, *Reference systems, conventions and notations for Gaia*,
GAIA-CA-SP-ARI-BAS-003,
URL <http://www.rssd.esa.int/cs/livelihood/open/358698>
- [BAS-030], Bastian, U., 2013, *OBMT, the technical time coordinate of DPAC*,
GAIA-C1-TN-ARI-BAS-030,
URL <http://www.rssd.esa.int/cs/livelihood/open/2873574>
- [JC-014], Castaneda, J., Martinez, O., Fabricius, C., et al., 2009, *TmTools Implementation of Astrium TM Model for Cycle 6*,
GAIA-C1-TN-UB-JC-014,
URL <http://www.rssd.esa.int/cs/livelihood/open/2894737>
- [GAIA.ASF.SP.PLM.00034], EADS Astrium, 2007, *Clock Distribution Unit (CDU) Requirement Specification*,
GAIA.ASF.SP.PLM.00034,
URL <http://www.rssd.esa.int/cs/livelihood/open/2722559>
- [GAIA.ASF.TCN.PLM.00137], EADS Astrium, 2007, *Video processing unit (VPU) algorithms main topics description*,
GAIA.ASF.TCN.PLM.00137,
URL <http://www.rssd.esa.int/cs/livelihood/open/2770198>
- [GAIA.ASU.SP.PLM.00024], EADS Astrium, 2013, *Video processing unit (VPU) specification*,
GAIA.ASU.SP.PLM.00024,
URL <http://www.rssd.esa.int/cs/livelihood/open/2733886>
- [CF-011], Fabricius, C., de Bruijne, J., Biermann, M., et al., 2015, *On ground reconstruction of windows*,
GAIA-C3-TN-UB-CF-011,
URL <http://www.rssd.esa.int/cs/livelihood/open/2824566>

[GAIA-UL-009], Lammers, U., 2005, *Representation of timing data in Gaia science telemetry*, GAIA-UL-009,
 URL <http://www.rssd.esa.int/cs/livelihood/open/456287>

4.2 Definitions, acronyms, and abbreviations

The following table has been generated from the on-line Gaia acronym list:

Acronym	Description
AC	ACross scan (direction)
AL	ALong scan (direction)
AOCS	Attitude and Orbit Control Sub-system
APC	Along-scan Pixel Coordinate
CCD	Charge-Coupled Device
CDMU	Command and Data Management Unit
CDU	Clock Distribution Unit
DPAC	Data Processing and Analysis Consortium
DPACE	Data Processing and Analysis Consortium Executive
E2ES1	End-to-End Stage 1
FOV	Field of View (also denoted FOV)
GASS	GAia System-level Simulator
GIBIS	Gaia Instrument and Basic Image Simulator
HEALPix	Hierarchical Equal-Area iso-Latitude Pixelisation
HW	Hardware
IDT	Initial Data Treatment
IDV	Intermediate Data Validation
LSB	Least Significant Bit (CCD)
MC	Master Clock
MDB	Main Database
MIT	MOC Interface Task
MSB	Most Significant Bit
MainDB	MAIN DataBase
OBMT	On-Board Mission Timeline
OBT	On-Board Time (realised in CDU)
PDHU	Payload Data Handling Unit
PSF	Point Spread Function
RVS	Radial Velocity Spectrometer
SM	Sky Mapper
SVN	SubVersion

SW	Software (also denoted S/W)
TCB	Barycentric Coordinate Time
TDI	Time-Delayed Integration (CCD)
TM	TeleMetry
UB	University of Barcelona (Spain)
UTC	Coordinated Universal Time
VO	Virtual Object
VPU	Video Processing Unit
XM	Cross-Matching

Appendix A: Validation using CU2 reference data: Conversion of TransitIds (OBT/OBMT)

As previously indicated, the TransitId is defined using the OBMT time scale, which is determined by the MIT — either during operations or during development and testing. Thus, in principle, the OBMT is actually unknown prior to the MIT — and thus the CU2 simulators (such as GASS or GIBIS) do not know the OBMT of a given observation. In practice, it means that the simulated data generated by CU2 can only use the OBT time scale for observational (transit) data.

The validation of the several software products of DPAC (such as IDT, for example) often makes use of the so-called “True Values” given by CU2, which indicate the “ideal” features of the observations. For example, they indicate the “ideal” (or “true”) Observation Times of a given transit in each CCD, or its Field Angles, or its sky coordinates or G magnitude at the epoch of observation.

In order to make possible the validation, we need some kind of “reference” for each observation, so that we can compare the results obtained from our system with the expected values indicated by CU2. Thus, the TrueValues also include a TransitId field — which we will call *TrueValues-TransitId* from now on to avoid confusion, following the same format described in the present document. The only difference is that it is generated from the OBT (instead of OBMT), for the reasons explained in the first paragraph of this section.

As of Development Cycle 9, and thus also for the pre-launch End-to-End test campaign, it was agreed to introduce a **non-zero offset in OBMT**. That is, the MIT determines OBMT as OBT plus a given offset. This offset was agreed to be **80 days** (see BAS-030). It means that if we look for the TrueValues of a given observation (i.e. transit) processed by our favourite system (such as IDT), we will *not* find it, because its TrueValuesTransitId will not coincide with the TransitId that we have e.g. in the AstroElementary or AstroObservation. This is caused by this 80-days difference between the time reference used by CU2 and the time reference used in our “post-MIT” system.

Fortunately, there is a simple solution: one of the identifiers must be “corrected” to the time scale of the other identifier.

We can either use the TransitId (of our processed data) or the TrueValuesTransitId (of the True-Values) as the reference. For example, if we use the OBMT as the reference time scale, the procedure would be as follows:

1. For each observation (from our favourite system) to validate, take its TransitId.
2. Decode the TransitId, obtaining the OBMT of the AF1 reference acquisition pixel

(as well as the rest of TransitId fields).

3. Convert this AF1 OBMT to the OBT scale, using the adequate offset (80 days as of Dev. Cycle 9, and most probably Dev. Cycle 10 as well).
4. Once we have the AF1 in OBT, recode again and obtain the TrueValuesTransitId.
5. Look for this TrueValuesTransitId in the TrueValues.
6. Now we can make the comparison and get the residuals.

CU3-IDV is already implementing this correction successfully (assessed with E2E-S1 tests). The code is available in `gaia.cu3.idv.utils.Utils`. There are two routines:

- `unapplyObtToObmtOffset ()` returns the TrueValuesTransitId from a TransitId
- `applyObtToObmtOffset ()` does the other way around.

Such routines may be moved to GaiaTools in the future. It is worth mentioning that both routines (in their implementation as of Dev. Cycle 9) make use of an OBT-OBMT offset defined in the IDV or IDT properties. In case no specific offset is defined, the default one is used — that is, the 80 days, defined in `gaia.cu1.tools.time.Obmt` as `INITIAL_OFFSET`.

It is worth mentioning that the current OBMT implementation by CU1 (as of Dev. Cycle 9) is very simple, that is, just applying this fixed 80-day offset to OBT. Also, no reset is applied to OBT (although TmTools 9.x is prepared to simulate it), thus leading to an OBT with a well-defined reference epoch (more specifically, J2010.0). Future implementations will be more elaborated, including the effect of on-board clock resets and the simulation of UTC (see BAS-030).

Finally, we must note that the OBT-OBMT offset determined by MIT must always be an integer multiple of the time resolution available in the TransitId, that is, $204.8 \mu\text{s}$. If, for example, the offset is always multiple of 1 hour, it will be fulfilled. This requirement will make possible the one-to-one conversion between TrueValuesTransitIds and TransitIds. We must note that this is obviously just a concern for simulations (i.e. development); once in operations, the offset may be any integer multiple of 50 ns (not necessarily of $204.8 \mu\text{s}$).

Appendix B: Status of actions in this issue

Below we list the several actions that were opened in Issue 02 of this document, for the adaptation to this new TransitId format, and we also include their resolution:

1. Verify with Astrium the format of ‘ref_acquisition_time’. More specifically, verify that:
 - (a) “ref_acquisition_time is actually a counter of the master clock, with a resolution of 50ns.”

It is confirmed in section 7.2 of GAIA.TSE.DD.CDU.0026 (“CDU Technical Description”):
The CDU maintains a 64-bits counter, called On-Board Time counter (OBT). This counter is incremented by the 20 MHz Master Clock. The counter resolution is 50 ns.

Also section 4.1 of GAIA.ASU.SP.PLM.00024 confirms this:
The OBS signal from the CDU contains an OBT value which is synchronized with the TDII period and transmitted every TDII period (see Section 6.4.3 “Discrete Signals Interface” for more details). The OBT value is incremented by the CDU at each Master Clock (MC) period. The OBT value is extracted by the VPU HW at each TDII period and made available in a sliding buffer over N consecutive TDII cycles (typically 256 or 512 cycles). The VPU SW uses the OBT value for time tagging such as the time tagging of PDHU packets and the time tagging of AOCs speed measurements for the CDMU.
 - (b) “ref_acquisition_time will never contain any information in the 10 MSB.”

It was directly confirmed by Astrium, in a set of answers to doubts raised by the preparation of CF-011:
Those bits are not explicitly set to 0, but the OBT is not supposed to exceed 52 bits over the mission, so those 10 MSB will not change and stay null.
 - (c) “Check which is the epoch (or ‘zero point’) of such counter.”

The initial epoch of the OBT (before any reset) is to be determined, but it will probably be some date close to the launch of Gaia. When its exact value will be known it will be applied to the OBMT determination and, at the same time, to the several timescale conversions (such as OBMT to TCB). Regarding to the clock resets, Astrium stated that:
Reset is either commanded by ground or by CDU reconfiguration. If it is a “ground reset”, it is up to the ground to record the last OBT. If it is a CDU reconfiguration, the TM will tell that the reconfiguration occurred, and then the OBT starts from 0.
2. “Provide an easy transformation between the APC and window datation (based on the SM and BAS-003) and the primary telemetry content (based on AF1 and the

Astrium reference system). Within the frame of the new TM Tools package of CU2 (in charge of adapting GASS to this new TM Model) we are already defining such formulae and validating them with simulated data.”

Indeed, TM Tools already includes such transformations, as indicated in JC-014. The on-ground reconstruction of window coordinates is also described in detail in CF-011.

3. “Update BAS-003 in order to include:
 - (a) References to the documents with the definition of the Astrium reference system and with the transformation previously mentioned.
 - (b) A definition of the APC from OBT (including OBT reset corrections).”

This action is not strictly required anymore, since BAS-030, CF-011 and JC-014 provide enough information.

4. “Update the TransitId generation and usage in CU2-GASS, CU3-IDT and any other DPAC process. It should be done by including in GaiaTools the necessary routines for such calculations and interpretations.”

This last action has also been fulfilled. The necessary routines are available in GaiaTools.