

BmnRoot DIGI Format

Reference guide to the detector digits
of the BM@N experiment

Document	Description of the DIGI (detector digits) format, version 1.0
Date	9 October 2026
Experiment	BM@N (Baryonic Matter at Nuclotron), NICA facility, JINR
Software	BmnRoot, dev branch (commit 027633c4)
Applies to	Simulated and experimental DIGI files

Contents

1 Introduction.....	3
1.1 Purpose and scope	3
1.2 Two processing chains	3
1.3 Producers of the DIGI branches	4
1.4 Conventions	5
2 File structure	5
2.1 Objects stored in the file	5
2.2 Event tree of experimental files	6
2.3 Event tree of simulated files	7
2.4 Branch names in simulated and experimental files	8
2.5 Links between objects	9
3 Run-level information	9
3.1 DigiRunHeader	10
3.2 Run metadata: RawRunMetadata and the spill tree	10
3.3 Simulation parameters and geometry	10
4 Event headers	11
4.1 BmnEventHeader. (experimental files)	11
4.2 MCEventHeader. (simulated files)	12
5 Digits of the strip detectors	12
5.1 Common fields: BmnStripDigit	12
5.2 Detector specifics	13
5.3 MC matching of the strip digits (simulated files)	13
6 TOF and Drift Chamber digits (experimental files)	13
6.1 TOF-400 and TOF-700: BmnTof1Digit	14
6.2 Drift Chambers: BmnDchDigit	14
7 Digits of the forward detectors	14
7.1 Common fields: BmnAbstractDigit	14
7.2 Waveform information (experimental files)	15
8 Trigger and beam detectors	16
8.1 TQDC channels: BmnTrigWaveDigit (experimental files)	16
8.2 TDC channels: BmnTrigDigit	16
9 Monte Carlo information (simulated files)	17
9.1 MC tracks: CbmMCTrack	17
9.2 MC points	17
9.3 GeoTracks	18
10 Working with DIGI files	18
10.1 Environment	18
10.2 Recommended: FairRoot tasks	18
10.3 Alternative: reading a DIGI file directly	20
10.4 Quick look with TTree::Draw	21
10.5 Further examples	21
Appendix A. Summary of the DIGI content	22
Appendix B. Detector identifiers (DetectorId)	25
Appendix C. Differences between simulated and experimental DIGI files	25
Appendix D. Fields that are not filled or need care	26
Appendix E. Data volume per event	27
E.1 Simulated data	27
E.2 Experimental data	28

1 Introduction

1.1 Purpose and scope

The DIGI files contain the detector digits: the responses of the individual detector channels (strips, cells, scintillators, TDC and ADC channels) in each event. They are the output of the first processing step and the input of the reconstruction macro `run_reco_bmn.C`, which writes the DST (described in [BmnRoot DST Format](#)). This document describes the DIGI format: the layout of the ROOT files, every branch of the event tree, the stored data classes and their fields, the links between the stored objects, and the differences between DIGI files of simulated and experimental events.

Simulated and experimental DIGI files are written by different macros: the simulated digits by the digitizers of `run_sim_bmn.C`, the experimental digits by the raw data decoder of `run_raw_bmn.C`. Both files store the events in a tree named `bmndata`, and the strip detectors use the same digit classes, but the formats differ in several respects:

- the same detector data are stored in branches with **different names**, for example `BmnGemStripDigit` in simulated and `GEM` in experimental files (Section 2.4);
- the digits of the forward detectors are stored in experimental files with derived classes that also keep the results of the waveform processing, for example `BmnFHCaLDigi` instead of `BmnFHCaLDigit`;
- simulated files contain the Monte Carlo truth: MC tracks, MC points and the MC matching of the strip digits;
- experimental files contain the event header of the data acquisition, the waveforms and times of the beam counters and trigger channels, the TOF digits, and the run metadata.

Where the content of an object differs, this is marked in place. Appendix A gives a short overview of all objects of DIGI files with their main methods, Appendix C summarises all differences, and Appendix D lists the fields that are currently not filled or need care.

The description corresponds to the BmnRoot `dev` branch as of October 2026.

1.2 Two processing chains

DIGI files are the intermediate files of the two processing chains that end with the DST:

Stage	Simulated data	Experimental data
Input	Output of an event generator: the particles produced in the collision with their momenta	Raw data files written by the BM@N data acquisition system
First step	<code>run_sim_bmn.C</code> : macro simulates passing of the generated particles through the BM@N setup with Geant4 transport package, producing Monte Carlo tracks, their points and detector responses (digits)	<code>run_raw_bmn.C</code> macro converts the raw data to the ROOT format and decodes to digits
DIGI file	Simulation file (default name <code>bmnsim.root</code>): MC tracks, MC points, digits and their MC matching	Digit file (default name <code>bm_n_run<run>_digi.root</code>): event headers, digits, run metadata
Reconstruction	<code>run_reco_bmn.C</code> macro reconstructs event data: hits, tracks, vertices, particle identification information, data of the forward detectors, centrality information	<code>run_reco_bmn.C</code> – the same macro with the similar output format
Output	DST file with the reconstructed data of the simulated events	DST file with the reconstructed data of the experimental events

`run_reco_bmn.C` recognises the type of its input file automatically: a file is simulated if it contains the FairRoot list `BranchList` with the branch `MCTrack` (function `BmnFunctionSet::isSimulationFile`). Simulated files are read with `FairFileSource`, experimental files with `BmnFileSource`, which takes the period and run numbers from `DigiRunHeader` (Section 3.1). The reconstruction tasks then read the branches of the given file type (Section 2.4).

Note: The conversion step of `run_raw_bmn.C` also writes an intermediate RAW ROOT file with the data of the electronics modules (ADC, TDC, synchronisation). It is kept by default (`doHoldRawRoot = kTRUE`), but it is not used in the analysis and is not described here.

1.3 Producers of the DIGI branches

In `run_sim_bmn.C` the MC tracks and MC points are written during the Geant4 transport, the MC points by the detector classes of the setup, whose geometry is loaded from the Geometry Database. After the transport of each event the digitizer tasks convert the MC points into digits. The table lists the digitizers in the order in which they are executed in `run_sim_bmn.C`:

Task (class)	Branch(es)	Content
Transport (FairRunSim, FairStack)	MCTrack, MCEventHeader., GeoTracks	MC tracks, generator information of the event, trajectories for the event display (GeoTracks only with <code>saveGeoInfo = kTRUE</code>)
Detector classes: BmnSiBT, BmnBd, BmnFD, BmnSilicon, CbmSts (GEM), BmnCSC, BmnTOF1, BmnTOF701, BmnScWall, BmnHodo, BmnNdet, BmnFHCa1, BmnSiProf, BmnVSP	<Det>Point, for example SiliconPoint, StsPoint (GEM)	MC points of the detectors (Section 9.2)
BmnSiBTDigitizer	BmnSiBTDigit, BmnSiBTDigitMatch	Silicon Beam Tracker strip digits and their MC matching
BmnSiProfDigitizer	BmnSiProfDigit, BmnSiProfDigitMatch	Silicon Beam Profilometer strip digits and their MC matching
BmnBdDigitizer	BdDigit	Barrel Detector digits
BmnSiMDDigitizer	SiMDDigit	Silicon Multiplicity Detector digits (only if the detector is in the setup)
BmnVSPDigitizer	BmnVSPDigit, BmnVSPDigitMatch	Vertex Silicon Plane strip digits and their MC matching
BmnSiliconDigitizer	BmnSiliconDigit, BmnSiliconDigitMatch	Forward Silicon Detector strip digits and their MC matching
BmnGemStripDigitizer	BmnGemStripDigit, BmnGemStripDigitMatch	GEM strip digits and their MC matching
BmnCSCDigitizer	BmnCSCDigit, BmnCSCDigitMatch	Cathode Strip Chamber strip digits and their MC matching
BmnFHCa1Digitizer	FHCa1Digit	Forward Hadron Calorimeter digits
BmnScWallDigitizer	ScWallDigit	Scintillation Wall digits
BmnHodoDigitizer	HodoDigit	Forward Hodoscope digits
BmnNdetDigitizer	NdetDigit	Neutron Detector digits

There are no digitizers for the TOF-400 and TOF-700: in simulated files these detectors are represented by their MC points, from which the reconstruction makes the TOF hits directly. The Forward Detector (FD) has MC points only as well. A digitizer whose detector is not in the setup is deactivated and writes no branch.

In `run_raw_bmn.C` the decoder (`BmnDecoder`) creates a mapper for each detector that is switched on in the detector setup of the macro (`SetDetectorSetup`). The mapper converts the data of the electronics modules into the digits of the detector according to the channel mapping of the run period. The table lists the mappers in the order of their creation:

Mapper (class)	Branch(es)	Content
BmnDecoder	BmnEventHeader.	Event header: run and event numbers, time, event type, trigger inputs
BmnTrigRaw2Digit	TQDC_<channel>, BD, FHCAL: names from the trigger mapping file <code>Trig_map_Run<period>.txt</code>	Waveforms and TDC times of the beam counters and trigger channels; TDC digits of the Barrel Detector and of other TDC channels
BmnSiliconRaw2Digit	SILICON	Forward Silicon Detector strip digits
BmnSiBTRaw2Digit	SiBT	Silicon Beam Tracker strip digits
BmnGemRaw2Digit	GEM	GEM strip digits
BmnVspRaw2Digit	VSP	Vertex Silicon Plane strip digits

Mapper (class)	Branch(es)	Content
BmnTof1Raw2Digit	TOF400	TOF-400 digits
BmnTof701Raw2Digit	TOF701	TOF-700 digits
BmnDchRaw2Digit	DCH	Drift Chamber digits
BmnScWallRaw2Digit	ScWallDigi	Scintillation Wall digits
BmnFHCALRaw2Digit	FHCALDigi	Forward Hadron Calorimeter digits
BmnHodoRaw2Digit	HodoDigi	Forward Hodoscope digits
BmnNdetRaw2Digit	NdetDigi	Neutron Detector digits
BmnHgndRaw2Digit	HgndDigi	Full-scale Neutron Detector (HGND) digits
BmnCscRaw2Digit	CSC	Cathode Strip Chamber strip digits
BmnDecoder (end of the run)	DigiRunHeader, RawRunMetadata, spill tree	Run information and run metadata (Section 3)

A detector that is switched on in the setup but has no data in a run gives an empty branch. Detectors of earlier run periods (MWPC, ZDC, ECAL) have their own branches (*MWPC*, *ZdcDigit*, *EcalDigit*), which are not covered here.

1.4 Conventions

- Units: coordinates and lengths in cm, momenta in GeV/c, energies in GeV, times in ns, unless stated otherwise. Signals of the strip detectors are given in ADC counts.
- Events: one entry of the tree *bmndata* is one event; all objects of an event are in the same entry.
- Index links: integer links are positions (starting from 0) in the *TClonesArray* of another branch of the same event. The value -1 means “no link”.
- Branch names: names ending with a dot (for example *BmnEventHeader.*) are single objects; names without a dot are *TClonesArray* collections.
- Numbering: stations, modules, strip layers and strips of the strip detectors are numbered from 0.
- Getters and members: the tables give the getter to be used and, where useful, the data member name (prefix f). The branches are split, so member names can be used directly in *TTree::Draw*, for example *bmndata->Draw("GEM.fStripSignal")*.
- Availability column: Both, Sim (simulated files only) or Exp (experimental files only). Both means that the information exists in both file types, possibly in branches with different names (Section 2.4). Remarks such as “Sim: ...; Exp: ...” describe fields that are filled differently.

2 File structure

2.1 Objects stored in the file

Besides the event tree, a DIGI file contains objects written once per file:

Key	Class	Content	Avail.
bmndata	TTree	Event tree: one entry per event (Sections 2.2 and 2.3)	Both
DigiRunHeader	DigiRunHeader	Run information of the decoder: period, run, times of the first and the last event, trigger configuration, channel maps (Section 3.1)	Exp
RawRunMetadata	BmnMetadataRaw	Spill and scaler metadata of the run (Section 3.2)	Exp
spill	TTree	Spill tree: one entry per spill record of the raw data (Section 3.2)	Exp
FairGeoParSet	FairGeoParSet	Geometry of the simulation (<i>TGeoManager</i>); the description holds the geometry tag (Section 3.3)	Sim
FairBaseParSet	FairBaseParSet	Simulation run parameters: beam energy, description “beam;target” (Section 3.3)	Sim
BmnFieldPar	BmnFieldPar	Magnetic field parameters of the simulation	Sim
FairGeoPassivePar	FairGeoPassivePar	Passive geometry parameters of the simulation	Sim

Key	Class	Content	Avail.
BmnFHCalGeoPar, BmnNdetGeoPar	same as the key	Geometry parameters (sensitive and passive nodes) of the FHCal and of the Neutron Detector	Sim
run ID	FairRtdbRun	Runtime database bookkeeping; the key name is the run ID of the simulation, which is the period number (e.g. 9)	Sim
FileHeader	FairFileHeader	FairRoot file header	Sim
BranchList, TimeBasedBranchList	TList	FairRoot bookkeeping: names of all branches; <code>MCTrack</code> in <code>BranchList</code> marks a simulated file	Sim
bmnroot	TFolder	FairRoot folder structure: which branch belongs to which detector folder	Sim
ProcessID0	TProcessID	ROOT bookkeeping of object references	Sim

Experimental files are written by the decoder without FairRoot (the title of the tree is `bmndigit`), so they contain neither FairRoot bookkeeping objects nor parameter containers.

Note: A file can contain several cycles of the tree key (for example `bmndata;2` and `bmndata;3`). ROOT automatically uses the latest cycle when the tree is read by name.

2.2 Event tree of experimental files

The table lists the branches of `bmndata` in experimental DIGI files with the class of the object (for `TClonesArray` branches, the class of the elements). The set of `TQDC_<channel>` branches and their names are defined by the trigger mapping file of the run period.

Branch	Class	Content
BmnEventHeader.	BmnEventHeader	Event header (Section 4.1)
TQDC_<channel>	BmnTrigWaveDigit	One branch per channel of the beam counters and trigger detectors: waveform and TDC times (Section 8.1)
BD, FHCal	BmnTrigDigit	TDC digits of the Barrel Detector and of other TDC channels (Section 8.2)
SILICON	BmnSiliconDigit	Forward Silicon Detector strip digits (Section 5)
SiBT	BmnSiBTDigit	Silicon Beam Tracker strip digits
GEM	BmnGemStripDigit	GEM strip digits
VSP	BmnVSPDigit	Vertex Silicon Plane strip digits
CSC	BmnCSCDigit	Cathode Strip Chamber strip digits
TOF400	BmnTof1Digit	TOF-400 digits (Section 6.1)
TOF701	BmnTof701Digit	TOF-700 digits (Section 6.1)
DCH	BmnDchDigit	Drift Chamber digits (Section 6.2)
FHCalDigi	BmnFHCalDigi	Forward Hadron Calorimeter digits (Section 7)
ScWallDigi	BmnScWallDigi	Scintillation Wall digits
HodoDigi	BmnHodoDigi	Forward Hodoscope digits
NdetDigi	BmnNdetDigi	Neutron Detector digits
HgndDigi	BmnHgndDigi	Full-scale Neutron Detector digits

2.3 Event tree of simulated files

The table lists the branches of `bmndata` in simulated DIGI files with the class of the object and the FairRoot folder of the branch.

Branch	Class	Folder	Content
<code>MCEventHeader.</code>	<code>FairMCEventHeader</code>	Event	Generator information of the event (Section 4.2)
<code>MCTrack</code>	<code>CbmMCTrack</code>	Stack	MC tracks (Section 9.1)
<code>BmnVSPDigit,</code> <code>BmnVSPDigitMatch</code>	<code>BmnVSPDigit, BmnMatch</code>	VSP	VSP strip digits and their MC matching (Section 5)
<code>BmnSiliconDigit,</code> <code>BmnSiliconDigitMatch</code>	<code>BmnSiliconDigit,</code> <code>BmnMatch</code>	SILICON	Forward Silicon Detector strip digits and their MC matching
<code>BmnGemStripDigit,</code> <code>BmnGemStripDigitMatch</code>	<code>BmnGemStripDigit,</code> <code>BmnMatch</code>	GEM_DIGIT	GEM strip digits and their MC matching
<code>BmnCSCDigit,</code> <code>BmnCSCDigitMatch</code>	<code>BmnCSCDigit, BmnMatch</code>	CSC_DIGIT	CSC strip digits and their MC matching
<code>BmnSiBTDigit,</code> <code>BmnSiBTDigitMatch</code>	<code>BmnSiBTDigit, BmnMatch</code>	SiBT_DIGIT	SiBT strip digits and their MC matching
<code>BmnSiProfDigit,</code> <code>BmnSiProfDigitMatch</code>	<code>BmnSiProfDigit,</code> <code>BmnMatch</code>	SiProf_DIGIT	Silicon Beam Profilometer strip digits and their MC matching
<code>BdDigit</code>	<code>BmnTrigDigit</code>	BD	Barrel Detector digits (Section 8.2)
<code>FHCalDigit</code>	<code>BmnFHCalDigit</code>	FHCal	Forward Hadron Calorimeter digits (Section 7)
<code>ScWallDigit</code>	<code>BmnScWallDigit</code>	ScWall	Scintillation Wall digits
<code>HodoDigit</code>	<code>BmnHodoDigit</code>	Hodo	Forward Hodoscope digits
<code>NdetDigit</code>	<code>BmnNdetDigit</code>	Ndet	Neutron Detector digits
<code>VSPPoint</code>	<code>BmnVSPPoint</code>	VSP	MC points of the VSP (Section 9.2)
<code>SiliconPoint</code>	<code>BmnSiliconPoint</code>	Silicon	MC points of the Forward Silicon Detector
<code>StsPoint</code>	<code>CbmStsPoint</code>	GEM	MC points of the GEM detectors
<code>CSCPoint</code>	<code>BmnCSCPoint</code>	CSC	MC points of the CSC
<code>SiBTPoint</code>	<code>BmnSiBTPoint</code>	SiBT	MC points of the Silicon Beam Tracker
<code>SiProfPoint</code>	<code>BmnSiProfPoint</code>	SiProf	MC points of the Silicon Beam Profilometer
<code>BdPoint</code>	<code>BmnBdPoint</code>	BD	MC points of the Barrel Detector
<code>FDPoint</code>	<code>BmnFDPoint</code>	FD	MC points of the Forward Detector
<code>TOF400Point</code>	<code>BmnTOF1Point</code>	TOF400	MC points of the TOF-400
<code>TOF700Point</code>	<code>BmnTOF1Point</code>	TOF700	MC points of the TOF-700
<code>FHCalPoint</code>	<code>BmnFHCalPoint</code>	FHCal	MC points of the FHCal
<code>ScWallPoint</code>	<code>BmnScWallPoint</code>	ScWall	MC points of the ScWall
<code>HodoPoint</code>	<code>BmnHodoPoint</code>	Hodo	MC points of the Forward Hodoscope
<code>NdetPoint, NdetSurfacePoint</code>	<code>BmnNdetPoint,</code> <code>BmnNdetSurfacePoint</code>	Ndet	MC points in the cells and on the surfaces of the Neutron Detector
<code>GeoTracks</code>	<code>TGeoTrack</code>	MCGeoTrack	Trajectories for the event display, only with <code>saveGeoInfo = kTRUE</code> (Section 9.3)

Tip: The MC points and especially `GeoTracks` make up most of the data volume of a simulated file (Appendix E). `GeoTracks` are needed only for the event display: keep `saveGeoInfo = kFALSE` (the default of `run_sim_bmn.C`) for mass production.

2.4 Branch names in simulated and experimental files

The table maps the branches that hold the same detector data in the two file types:

Data	Simulated file	Experimental file	Class
Event header	MCEventHeader.	BmnEventHeader.	Sim: FairMCEventHeader; Exp: BmnEventHeader (different content, Section 4)
VSP strip digits	BmnVSPDigit	VSP	BmnVSPDigit
Silicon strip digits	BmnSiliconDigit	SILICON	BmnSiliconDigit
GEM strip digits	BmnGemStripDigit	GEM	BmnGemStripDigit
CSC strip digits	BmnCSCDigit	CSC	BmnCSCDigit
SiBT strip digits	BmnSiBTDigit	SiBT	BmnSiBTDigit
SiProf strip digits	BmnSiProfDigit	–	BmnSiProfDigit
Barrel Detector	BdDigit	BD	BmnTrigDigit
FHCal	FHCalDigit	FHCalDigi	Sim: BmnFHCalDigit; Exp: BmnFHCalDigi, derived from BmnFHCalDigit
ScWall	ScWallDigit	ScWallDigi	Sim: BmnScWallDigit; Exp: BmnScWallDigi, derived from BmnScWallDigit
Forward Hodoscope	HodoDigit	HodoDigi	Sim: BmnHodoDigit; Exp: BmnHodoDigi, derived from BmnHodoDigit
Neutron Detector	NdetDigit	NdetDigi	Sim: BmnNdetDigit; Exp: BmnNdetDigi, derived from BmnNdetDigit
HGND	–	HgndDigi	BmnHgndDigi
TOF-400	TOF400Point (MC points, no digits)	TOF400	Sim: BmnTOF1Point; Exp: BmnTof1Digit
TOF-700	TOF700Point (MC points, no digits)	TOF701	Sim: BmnTOF1Point; Exp: BmnTof701Digit
Forward Detector (FD)	FDPoint (MC points, no digits)	TQDC_FD*	Sim: BmnFDPoint; Exp: BmnTrigWaveDigit
Drift Chambers	–	DCH	BmnDchDigit
Beam counters, veto counter, trigger inputs	–	TQDC_<channel>	BmnTrigWaveDigit
MC truth	MCTrack, <Det>Point, <Det>DigitMatch, GeoTracks	–	Section 9

The reconstruction tasks select the branch by the type of the input file: for example, `BmnGemStripHitMaker` reads `BmnGemStripDigit` from simulated and `GEM` from experimental files, and `BmnFHCalReconstructor` reads `FHCalDigit` or `FHCalDigi`. Code that reads the digits of both file types has to do the same (Sections 10.2 and 10.3). Since the experimental classes of the forward detectors derive from the simulated ones, code that uses only the address, time and signal of these digits works with both file types through the base class, e.g. `BmnFHCalDigit`.

2.5 Links between objects

The objects of an event are connected through indices and addresses:

From	Accessor	To	Avail.
Strip digits (BmnStripDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber()	Strip of the detector as defined in the XML configuration of the detector (Section 5.1)	Both
Forward detector digits (BmnAbstractDigit)	GetAddress()	Channel of the detector, decoded with the address class of the detector (Section 7.1)	Both
BdDigit / BD (BmnTrigDigit)	GetMod()	Module of the Barrel Detector	Both
TQDC_<channel> (BmnTrigWaveDigit)	GetMod()	Channel within the branch, if a branch has several channels (Section 8.1)	Exp
TOF digits (BmnTof1Digit)	GetPlane(), GetStrip(), GetSide()	Strip end of the TOF detector; the digits of both ends of a strip form a TOF hit in the reconstruction	Exp
<Det>DigitMatch[i]	same index i	<Det>Digit[i]	Sim
<Det>DigitMatch (BmnMatch)	GetLink(j).GetIndex()	MC point in the point branch of the detector: StsPoint (GEM), SiliconPoint, CSCPoint, VSPPoint, SiBTPoint, SiProfPoint (Section 5.3)	Sim
MC points (FairMCPoint)	GetTrackID()	MCTrack	Sim
CbmMCTrack	GetMotherId()	Mother MC track in the same array (-1 for primary particles)	Sim
FHCalPoint	BmnFHCalAddress::GetPhysicalAddress(GetAddress())	FHCalDigit with this address	Sim

3 Run-level information

Experimental files keep the run information of the decoder ([DigiRunHeader](#)) and the run metadata ([RawRunMetadata](#), [spill](#) tree). Simulated files keep the FairRoot parameter containers of the simulation.

3.1 DigiRunHeader

`DigiRunHeader` is written once per experimental file by the decoder. `BmnFileSource` takes the period and run numbers from it, and `run_reco_bmn.C` uses them to obtain the geometry from the Geometry Database and the magnetic field, beam and target – from the Condition Database (UniConDa).

Getter (member)	Type	Description
<code>GetPeriodId()</code> (<code>fPeriodId</code>)	<code>uint32_t</code>	Run period (session) number, e.g. 9
<code>GetRunId()</code> (<code>fRunId</code>)	<code>uint32_t</code>	Run number
<code>GetRunStartTime()</code> , <code>GetRunEndTime()</code> (<code>fRunStartTime</code> , <code>fRunEndTime</code>)	<code>TTimeStamp</code>	Times of the first and the last event of the file in UTC. <code>GetRunStartTP()</code> and <code>GetRunEndTP()</code> return them as <code>std::chrono</code> time points
<code>T0Map()</code> , <code>GetTrcBitMap()</code> , <code>GetScalerMap()</code>	maps	Channel maps: channels of the T0 module, input bits of the trigger module TRC (name → bit), scaler channels (name → board serial number and channel)
<code>GetTrigConfig()</code> (<code>trig_config</code>)	<code>BmnTrigConfig</code>	Trigger configuration of the data acquisition: <code>TrigChannelConfig()</code> per trigger channel, match window and trigger delay [ns], channel masks
<code>GetNEvents()</code> (<code>fEventCnt</code>)	<code>uint64_t</code>	Number of events; not filled by <code>run_raw_bmn.C</code> (Appendix D)
<code>GetBT()</code> , <code>GetBTnBusy()</code> , <code>GetBTAccepted()</code> , <code>GetAccepted()</code> , <code>GetProtection()</code> , <code>GetL0()</code>	<code>uint64_t</code> , <code>Double_t</code>	Beam trigger and DAQ counters of the run: beam triggers, beam triggers while the DAQ is not busy (also normalised by the live time), accepted triggers, trigger protection, L0 triggers. Filled by the metadata collection (Section 3.2), not by <code>run_raw_bmn.C</code>
<code>GetSpillMap()</code> , <code>GetScaler(name)</code> , <code>GetRawT0Counter(name, ch)</code> , <code>GetRawMSC()</code>	map, <code>uint64_t</code>	Spill information by the spill start time, run sums of the scaler (MSC) and T0 counters. Filled by the metadata collection

3.2 Run metadata: RawRunMetadata and the spill tree

`RawRunMetadata` (class `BmnMetadataRaw`) keeps the metadata that the data acquisition records besides the events: the status records of the spills (`SpillStatusVec()`: start and stop times, spill number, counters of the spill), the spill map (`SpillMap()`), the time distributions of the scalers and of the external conditions (`Scalers()`, `ExtConditions()`), and the channel maps of the T0, trigger and scaler modules (`RawT0Map()`, `TrcMap()`, `ScalersMap()`).

The spill tree has one entry per spill record of the raw data with the branches `BmnSpillHeader`. (`BmnSpillHeader`: first and last event numbers, number of events, spill number, start and stop times, scaler and trigger counters of the spill), `MSC` (scaler digits) and `T0Raw` (`BmnT0Raw`: counters of the T0 module). The decoder fills `BmnSpillHeader` only for run periods before 8. For the later periods the spill and run statistics are collected from all DIGI files of a run by the metadata collection (`macro/raw/BmnMetadataCollect.C`, class `BmnMetadataDecoder`), which writes a separate file with `DigiRunHeader` and the spill tree filled for the whole run.

Note: The spill tree and the run counters of a single DIGI file can be empty. Take the run statistics (beam flux, numbers of triggers) from the output of the metadata collection.

3.3 Simulation parameters and geometry

Simulated files contain the FairRoot parameter containers written by `run_sim_bmn.C`. The geometry of the simulation is stored in `FairGeoParSet`; its description holds the geometry tag of the Geometry Database (e.g. `run9_26.09.0`), followed by the path of the XML customization file after ‘;’ if the setup was customised. `FairBaseParSet` holds the beam energy per nucleon in its beam momentum field and “beam;target” (e.g. “Xe;Sn”) in its description; both are taken from the name of the generator file. `BmnFieldPar` holds the type, map and scale of the magnetic field used in the transport. The key name of the `FairRtdbRun` object is the run ID of the simulation, which is set to the period number of the geometry tag (e.g. 9); `run_reco_bmn.C` takes the period of a simulated file from it (`BmnFunctionSet::GetPeriodNumber`).

When a simulated file is processed with `FairRunAna` and the file is also set as the parameter input (Section 10.2), the geometry is loaded automatically as `gGeoManager` during `Init()`. Experimental files do not contain the geometry: it is loaded from the Geometry Database by the geometry tag of the run period, as in `run_reco_bmn.C`.

4 Event headers

4.1 BmnEventHeader. (experimental files)

`BmnEventHeader` (derived from `FairEventHeader`) is filled by the decoder for every event from the event data of the data acquisition and of the trigger module.

Getter (member)	Type	Description
<code>GetRunId()</code> (fRunId)	<code>UInt_t</code>	Run number
<code>GetPeriodId()</code> (fPeriodId)	<code>UInt_t</code>	Run period
<code>GetEventId()</code> (fEventId)	<code>UInt_t</code>	Event number of the data acquisition in the run. The events of a run are distributed over several raw data files, so the numbers in one DIGI file are not consecutive
<code>GetEventTimeTS()</code> (fEventTimeTS)	<code>TTimeStamp</code>	Absolute (astronomical) time of the event in UTC with nanosecond precision: <code>GetSec()</code> , <code>GetNanoSec()</code> . <code>GetEventTimeTP()</code> returns it as a <code>std::chrono</code> time point
<code>GetEventTime()</code> (fEventTime)	<code>Double_t</code>	The same moment as UNIX time: seconds since 1970-01-01, equal to <code>GetEventTimeTS().AsDouble()</code>
<code>GetVspTS()</code> (fEventVspTS)	<code>TTimeStamp</code>	Trigger time of the event in the clock of the VSP readout; not an absolute time
<code>GetEventType()</code> (fEventType)	<code>BmnEventType</code>	1 – physics event (<code>kBMNPAYLOAD</code>), 0 – pedestal event (<code>kBMNPEDESTAL</code>); 2–4 are service records (end of spill, statistics, empty)
<code>GetSpillStart()</code> (fSpillStart)	<code>Bool_t</code>	True for the first event of a new spill
<code>GetInputSignalsAR()</code> , <code>GetInputSignalsBR()</code> (fInputsAR, fInputsBR)	<code>UInt_t</code>	Input bits of the trigger module TRC after and before the reduction (prescaling); the bit of a trigger is given by <code>DigiRunHeader::GetTrcBitMap()</code>
<code>GetInputSignalsVector()</code> (fInputsVec)	<code>vector<uint32_t></code>	States of the trigger inputs in successive TRC clock time slices around the trigger
<code>GetTimeShift()</code> (fTimeShift)	<code>unordered_map<UInt_t, Long64_t></code>	Time shift of each electronics board (key: board serial number) relative to the T0 board, from the White Rabbit synchronisation [ns]. The decoder adds it to the TDC times of the board
<code>GetStartSignalTime()</code> , <code>GetStartSignalWidth()</code>	<code>Double_t</code>	Time and width of the start (T0) signal [ns]; not filled for the current period (Appendix D)
<code>GetTrigInfo()</code> (fTrigInfo)	<code>BmnTrigInfo</code>	Counters of the trigger module: <code>GetTrigCand()</code> , <code>GetTrigAccepted()</code> , <code>GetTrigBefo()</code> , <code>GetTrigAfter()</code> , <code>GetTrigRjct()</code> , <code>GetTrigAll()</code> , <code>GetTrigAvail()</code> (Appendix D)
<code>GetTripWord()</code> (fTripWord)	<code>Bool_t</code>	Flag of tripped GEM modules; not filled (false)
<code>GetInputFileId()</code> , <code>GetMCEntryNumber()</code>	<code>Int_t</code>	Fields of <code>FairEventHeader</code> , not used (0)

Note: In the DST the event information is stored in `DstEventHeader.`, which is filled from `BmnEventHeader.` (Exp) or `MCEventHeader.` (Sim), and the processed trigger information in `BmnTrigInfo..`

4.2 MCEventHeader. (simulated files)

FairMCEventHeader keeps the information of the event generator and of the primary vertex.

Getter (member)	Description
GetRunID() (fRunId)	Run ID of the simulation: the period number (Section 3.3)
GetEventID() (fEventId)	MC event number, starting from 1
GetX(), GetY(), GetZ()	Primary vertex [cm], including the smearing of the beam position and of the target
GetT() (fT)	Event time of the generator; 0 if the generator does not set it
GetB() (fB)	Impact parameter [fm]
GetNPrim() (fNPrim)	Number of primary particles from the generator
GetRotX(), GetRotY()	Rotation around the x and y axes (beam tilt) [rad]
GetRotZ()	Rotation around the z axis (reaction plane angle) [rad]
IsSet() (fIsSet)	True if the header is filled

5 Digits of the strip detectors

The strip detectors – VSP, Forward Silicon Detector, GEM, CSC, SiBT and, in simulated files, SiProf – store one digit per strip with a signal. All of them use classes derived from **BmnStripDigit**; the same classes are used in simulated and experimental files (branch names in Section 2.4).

5.1 Common fields: BmnStripDigit

Getter (member)	Type	Description	Sim / Exp
GetStation() (fStation)	Int_t	Station of the detector	Both
GetModule() (fModule)	Int_t	Module within the station	Both
GetStripLayer() (fStripLayer)	Int_t	Strip layer: index of the layer in the module description of the XML configuration of the detector (<code>parameters/<det>/XMLConfigs</code>). Each layer is of the type “lower” or “upper”; a module with several zones has a pair of layers per zone, e.g. GEM: 0, 1 – large zone, 2, 3 – hot zone around the beam	Both
GetStripNumber() (fStripNumber)	Int_t	Strip number within the layer	Both
GetStripSignal() (fStripSignal)	Double_t	Signal of the strip [ADC counts]. Exp: ADC amplitude after the pedestal subtraction; only strips above the threshold are stored. Sim: simulated signal, for some detectors on a different scale (Appendix D)	Both
GetStripNoise() (fStripNoise)	Double_t	Noise of the strip (RMS of the pedestal) [ADC counts]. Exp: filled for GEM, Silicon, CSC and SiBT; VSP: -1. Sim: not filled (-1)	Both
IsGoodDigit() (fIsGoodDigit)	Bool_t	Always true: only good digits are stored	Both

5.2 Detector specifics

Detector	Class	Specifics
VSP	BmnVSPDigit	Additional field GetTimeSinceBegin_ns() (fTimeSinceBegin_ns): Exp – time of the strip signal relative to the trigger time of the VSP readout [ns]; Sim – not set. Exp: the signal is the ADC value of the VSP readout chips (1–31); the digits of the upper-layer strips that are also connected to the third layer of a module are stored twice, with layer 1 and with layer 2
Silicon	BmnSiliconDigit	No additional fields
GEM	BmnGemStripDigit	No additional fields. Sim: the MC points of the GEM are stored in StsPoint
CSC	BmnCSCDigit	No additional fields
SiBT	BmnSiBTDigit	Silicon Beam Tracker upstream of the target; its digits are reconstructed only in experimental data (beam tracks)
SiProf	BmnSiProfDigit	Silicon Beam Profilometer; simulated files only

5.3 MC matching of the strip digits (simulated files)

For each strip digit branch of a simulated file there is a branch `<Det>DigitMatch` with the same number of [BmnMatch](#) objects: the match with index *i* belongs to the digit with index *i*. A match lists the MC points that contributed to the signal of the strip:

Getter (member)	Description
GetNofLinks() , GetLink(j) , GetLinks() (fLinks)	Links (BmnLink) to the contributing MC points; no links for strips fired by noise only
BmnLink::GetIndex() (fIndex)	Index of the MC point in the point branch of the detector: StsPoint (GEM), SiliconPoint , CSCPoint , VSPPoint , SiBTPoint , SiProfPoint
BmnLink::GetWeight() (fWeight)	Contribution of the MC point to the strip signal; for GEM and CSC the fraction of the signal of the point that is collected on this strip
BmnLink::GetFile() , GetEntry()	Not used (–1): the points belong to the same event
GetMatchedLink() , GetMatchedIndex() (fMatchedIndex)	Link with the largest weight and its position in the list of links (–1 without links)
GetTotalWeight() (fTotalWeight)	Sum of the weights of all links

The MC track of a point is `MCTrack[point->GetTrackID()]` (Section 9). The reconstruction carries the matching over to the hits and tracks of the DST.

6 TOF and Drift Chamber digits (experimental files)

In experimental files the TOF-400 and TOF-700 store one digit per signal of a strip end, the Drift Chambers one digit per wire signal. Simulated files contain no digits of these detectors: the reconstruction creates the TOF hits directly from the MC points [TOF400Point](#) and [TOF700Point](#) (Section 9.2).

6.1 TOF-400 and TOF-700: BmnTof1Digit

The branches TOF400 (class `BmnTof1Digit`) and TOF701 (class `BmnTof701Digit`, derived from `BmnTof1Digit` without additional fields) have the same fields. The name TOF701 refers to the decoding of the TOF-700 in the same way as the TOF-400 (DetectorId `kTOF701`).

Getter (member)	Type	Description
<code>GetPlane()</code> (fPlane)	<code>Short_t</code>	Detector plane (chamber)
<code>GetStrip()</code> (fStrip)	<code>Short_t</code>	Strip within the plane
<code>GetSide()</code> (fSide)	<code>Short_t</code>	Strip end (0 or 1); the reconstruction combines the digits of both ends of a strip into a hit
<code>GetTime()</code> (fTime)	<code>Float_t</code>	Leading edge time of the signal in the TDC [ns], corrected for the time shift of the TDC board (Section 4.1)
<code>GetAmplitude()</code> (fAmplitude)	<code>Float_t</code>	Width of the signal: trailing minus leading edge time (time over threshold) [ns], a measure of the amplitude

The time of flight is obtained in the reconstruction from the difference of the strip time and the start (T0) time, with the calibration and slewing corrections of the TOF.

6.2 Drift Chambers: BmnDchDigit

The Drift Chambers belong to the setup of earlier run periods; in the current setup the branch DCH is empty.

Getter (member)	Type	Description
<code>GetPlane()</code> (fPlane)	<code>UChar_t</code>	Plane of the chambers
<code>GetWireNumber()</code> (fWire)	<code>Short_t</code>	Wire number
<code>GetTime()</code> (fTime)	<code>Double_t</code>	Time of the wire signal [ns]
<code>GetRefId()</code> (fRef)	<code>Int_t</code>	Not used (0)

7 Digits of the forward detectors

The forward detectors – FHCaI, Scintillation Wall (ScWall), Forward Hodoscope (Hodo), Neutron Detector (Ndet) and, in experimental files, the full-scale Neutron Detector (HGND) – store one digit per channel with a signal. All classes derive from `BmnAbstractDigit`; the channel is identified by a 32-bit address.

7.1 Common fields: BmnAbstractDigit

Getter (member)	Type	Description	Sim / Exp
<code>GetAddress()</code> (fAddress)	<code>uint32_t</code>	Address of the channel, decoded with the address class of the detector or with the getters of the digit class (table below)	Both
<code>GetSignal()</code> (fSignal)	<code>double</code>	Signal of the channel. Exp: signal calibrated with the calibration of the run. Sim: energy deposition converted with the scale of the digitizer configuration (<code>parameters/<det>/<DET>_digitizer_period<N>.json</code>); the scales differ between the detectors (Appendix D)	Both
<code>GetTime()</code> (fTime)	<code>double</code>	Time of the signal [ns]. Sim: time of the MC point with which the signal crossed the threshold (FHCaI, ScWall, Ndet) or mean time of the MC points (Hodo). Exp: filled for the Forward Hodoscope only (Appendix D)	Both

The address contains the system identifier (DetectorId, Appendix B) and the channel fields:

Detector	Address class	Fields of the address (getters of the digit class)
FHCal	BmnFHCalAddress	GetModuleType(), GetModuleId(), GetSectionId(): module type, module and longitudinal section; the MC points also have the scintillator (GetScintillatorId())
ScWall	BmnScWallAddress	GetCellId(): cell
Hodo	BmnHodoAddress	GetStripId(), GetStripSide(), GetGain(): strip, strip end, gain channel
Ndet, HGND	BmnNdetAddress	GetArmId(), GetRowId(), GetColumnId(), GetLayerId()
all	–	GetSystemId(): DetectorId of the detector

BmnHodoDigit also has GetIsSaturated() (fIsSaturated): true if the signal is saturated (Sim: always false). BmnHgndDigi (HGND) derives from BmnNdetDigit and adds the times GetToa() (fToa, time of arrival) and GetTot() (fTot, time over threshold).

7.2 Waveform information (experimental files)

The experimental classes BmnFHCalDigi, BmnScWallDigi, BmnHodoDigi and BmnNdetDigi derive from the simulated classes and from BmnDigiContainerTemplate, which keeps the result of the waveform processing of the channel in public members:

Member	Type	Description
fAmp1	int	Amplitude of the waveform [ADC counts]
fZL	int	Zero level (baseline) of the waveform [ADC counts]
fIntegral	int	Integral of the waveform [ADC counts]
fTimeMax	int	Time of the maximum [ADC samples]
fToT	int	Time over threshold [ADC samples]; not filled (0)
fFitAmp1, fFitZL, fFitIntegral, fFitTimeMax	float	The same values from the fit of the waveform; 0 if the fit is not done
fFitR2 (GetFitR2())	float	Quality of the fit, good near 0; 2 (default) if the fit is not done
fWfm, fFitWfm (GetWfm())	vector<float>	Waveform and fitted waveform; empty, not stored by default

BmnHodoDigi adds fSignalIntegral (signal from the waveform integral, calibrated), fNpeaks (number of pulses found in the waveform of the channel; with pile-up every pulse gives a separate digit), fCrosstalk (minimum of the waveform in the signal gate, a measure of the crosstalk) and fR2history (fit quality of the pulses, filled only with the fit).

8 Trigger and beam detectors

8.1 TQDC channels: BmnTrigWaveDigit (experimental files)

The beam counters, the veto counter, the Forward Detector, the trigger signals of the FHCAL and of the Neutron Detectors, and the input signals of the trigger module are read out with TQDC modules, which record the waveform and the TDC times of a channel. The decoder writes one branch `TQDC_<channel>` per channel of the trigger mapping file `input/Trig_map_Run<period>.txt` (section `TDC-CONFIGURATION`), so the set of branches depends on the run period:

Branches	Channels
<code>TQDC_BC0*</code> , <code>TQDC_BC1*</code> , <code>TQDC_BC2*</code>	Beam counters
<code>TQDC_VC*</code>	Veto counter
<code>TQDC_FD*</code>	Forward Detector
<code>TQDC_FHCAL*</code>	Trigger signals of the FHCAL
<code>TQDC_NDS*</code> , <code>TQDC_NDF*</code> , <code>TQDC_ND*V</code> , <code>TQDC_FND*</code>	Trigger signals of the Neutron Detectors
<code>TQDC_TR_<input></code>	Input signals of the trigger module; the names are those of the trigger inputs in the section <code>INPUTS</code> of the mapping file (e.g. <code>BT</code> , <code>MBT</code> , <code>CCT1</code>)

Getter (member)	Type	Description
<code>GetMod()</code> (<code>fMod</code>)	<code>uint16_t</code>	Channel number within the branch (0 if the branch has one channel)
<code>GetValue()</code> (<code>fValue</code>)	<code>vector<int16_t></code>	Waveform: ADC samples (8 ns) after the baseline subtraction; the sign of negative signals is inverted, so the pulses are positive
<code>GetNSamples()</code>	<code>size_t</code>	Number of samples of the waveform
<code>GetIntegral(start, stop)</code> , <code>GetPeak(start, stop)</code>	<code>int32_t</code> , <code>int16_t</code>	Sum and maximum of the samples in the range [start, stop]
<code>TdcVector()</code> (<code>fTdcTimes</code>)	<code>vector<Double_t></code>	TDC times of the signals of the channel in the event [ns], corrected for the time shift of the board (Section 4.1)
<code>GetTime()</code> , <code>GetTrigTimestamp()</code> , <code>GetAdcTimestamp()</code>	<code>Double_t</code>	Time and time stamps of the trigger and of the waveform; not filled (Appendix D)
<code>fNsmpl</code> , <code>fValueI</code>	–	Fields of earlier versions of the class, kept to read old files

Tip: The DST keeps the processed information of the main channels (amplitudes and times of the beam counters, veto counter and FD, start time) in `BmnTrigInfo.`; the waveforms are available only in the DIGI files.

8.2 TDC channels: BmnTrigDigit

Channels read out with TDCs only, such as the 40 modules of the Barrel Detector, are stored as `BmnTrigDigit`, one digit per signal: in the branch `BD` (and other TDC channels of the trigger mapping) of experimental files and in the branch `BdDigit` of simulated files. The simulated digits are made by `BmnBdDigitizer` from the energy deposition of the MC points of a module, with noise and a threshold.

Getter (member)	Type	Description	Sim / Exp
<code>GetMod()</code> (<code>fMod</code>)	<code>Short_t</code>	Module (channel) number, e.g. BD module 0–39	Both
<code>GetTime()</code> (<code>fTime</code>)	<code>Double_t</code>	Leading edge time [ns]. Exp: TDC time corrected for the time shift of the board; Sim: time of the earliest MC point of the module	Both
<code>GetAmp()</code> (<code>fAmp</code>)	<code>Double_t</code>	Width of the signal (time over threshold) [ns]. Sim: computed from the energy deposition	Both
<code>GetTimestamp()</code> (<code>fTimestamp</code>)	<code>Double_t</code>	Not used (–1)	Both

9 Monte Carlo information (simulated files)

9.1 MC tracks: CbmMCTrack

MCTrack contains the primary particles of the generator and the secondary particles kept by the stack of the simulation. The DST contains the same array in BmnMCInfo..

Getter (member)	Description
GetPdgCode()	PDG code; nuclei have codes 100ZZZAAAI (e.g. 1000010020 for the deuteron)
GetMotherId()	Index of the mother track in the same array; -1 for primary particles
GetPx(), GetPy(), GetPz(), GetMomentum(TVector3&)	Momentum at the start vertex [GeV/c]
GetStartX(), GetStartY(), GetStartZ(), GetStartT()	Start vertex [cm] and time [ns]
GetP(), GetPt(), GetRapidity(), GetEnergy(), GetMass()	Derived kinematics; the mass is taken from TDatabasePDG
Get4Momentum(TLorentzVector&)	Four-momentum
GetNPoints(DetectorId)	Number of MC points of the track in a detector, bit-packed in the member fNPoints with a few bits per detector (e.g. at most 7 for the GEM)
GetPolar(i)	Polarisation components, i = 0, 1, 2

9.2 MC points

The MC points are written by the detector classes during the transport: a point describes the passage of a particle through a sensitive volume. All point classes derive from FairMCPoint, which has the following functions to get the information on an MC point:

Getter (member)	Description
GetTrackID() (fTrackID)	Index of the MC track in MCTrack
GetX(), GetY(), GetZ()	Position [cm]; for most detectors the entry point into the volume
GetPx(), GetPy(), GetPz()	Momentum at this position [GeV/c]
GetTime() (fTime)	Time since the start of the event [ns]
GetLength() (fLength)	Track length since the creation of the particle [cm]
GetEnergyLoss() (fELoss)	Energy deposition in the volume [GeV]
GetDetectorID() (fDetectorID)	Detector identifier; its meaning depends on the detector

The point classes of the detectors add their own information to that of `FairMCPPoint`. The table lists the branches with the MC points, their classes and the additional information of each class:

Branch	Class	Additional information
StsPoint (GEM)	CbmStsPoint	Exit point and momentum (<code>GetXOut()</code> , ..., <code>GetPxOut()</code> , ...), station and module (<code>GetStation()</code> , <code>GetModule()</code>)
SiliconPoint, VSPPoint, CSCPoint, SiBTPoint, SiProfPoint	BmnSiliconPoint, BmnVSPPoint, BmnCSCPoint, BmnSiBTPoint, BmnSiProfPoint	Exit point and momentum, station and module, charge, PDG code and primary flag of the particle (<code>GetCharge()</code> , <code>GetPdgId()</code> , <code>GetIsPrimary()</code>); CSC, SiBT and SiProf also the point in the middle plane of the sensitive volume (members <code>fX_center</code> , <code>fY_center</code> , <code>fZ_center</code>)
BdPoint, FDPPoint	BmnBdPoint, BmnFDPPoint	Exit point and momentum, copy number of the module (<code>GetCopy()</code>), charge, PDG code, primary flag, light yield (<code>GetLightYield()</code>), entry and exit times
TOF400Point, TOF700Point	BmnTOF1Point	Strip identifier (<code>GetVolumeUID()</code>), decoded with <code>GetModule()</code> (plane), <code>GetStrip()</code> and <code>GetRegion()</code>
ScWallPoint, HodoPoint	BmnScWallPoint, BmnHodoPoint	Copy numbers of the volume and of its mother volume (<code>GetCopy()</code> , <code>GetCopyMother()</code>): ScWall – cell in the mother volume, Hodo – strip in the mother volume
FHCalPoint	BmnFHCalPoint	Address of the scintillator (<code>GetAddress()</code> , <code>BmnFHCalAddress</code>)
NdetPoint	BmnNdetPoint	Address of the cell (<code>GetAddress()</code> , <code>BmnNdetAddress</code>)
NdetSurfacePoint	BmnNdetSurfacePoint	Particles crossing the surfaces of the Neutron Detector: PDG code (<code>GetPdg()</code>), arm, plane and direction (<code>GetArmID()</code> , <code>GetPlaneID()</code> , <code>GetDirectionID()</code>)

9.3 GeoTracks

`GeoTracks` contains the trajectories of the particles (TGeoTrack: PDG code `GetPDG()`, number of points `GetNpoints()`, points `GetPoint(i, x, y, z, t)`) for the event display. The branch is written only if `run_sim_bmn.C` is run with `saveGeoInfo = kTRUE` parameter, which also creates the `full_geometry.root` file with the geometry of the setup. The trajectories are stored for particles with a momentum above 50 MeV/c and within the vertex region set in the macro (`FairTrajFilter`). They dominate the size of the file (Appendix E).

10 Working with DIGI files

DIGI files are mainly the input of the reconstruction. Studies on the digit level – detector performance, calibration, noise, comparison of simulation and data – can be done as FairRoot tasks (Section 10.2) or by reading the tree directly in a ROOT macro (Section 10.3).

10.1 Environment

The digit classes are BmnRoot and FairRoot classes, so the BmnRoot environment must be loaded before CERN ROOT is started; the class dictionaries are then found automatically:

```
source <bmnroot build directory>/config.sh
```

10.2 Recommended: FairRoot tasks

A FairRoot task gets the digit branches from `FairRootManager` and can run in its own macro or in the reconstruction macro `macro/run/run_reco_bmn.C` together with the reconstruction, where it can also use the objects of the tasks executed before it (hits, tracks). As the reconstruction tasks of this macro do, the task has to read the branch of the given file type; the flag `isExp` is passed to its constructor. The example task fills the strip signal spectra of the GEM stations (for brevity the methods are defined in the class body):

```
// BmnGemDigitQa: example task for DIGI files - strip signal spectra of the GEM stations
```

```

class BmnGemDigitQa : public FairTask
{
public:
    BmnGemDigitQa(Bool_t isExp = kFALSE)
        : FairTask("BmnGemDigitQa")
        , fIsExp(isExp)
    {}

    InitStatus Init() override
    {
        // the GEM digits have different branch names in experimental and simulated files
        FairRootManager* ioman = FairRootManager::Instance();
        fDigits = (TClonesArray*)ioman->GetObject(fIsExp ? "GEM" : "BmnGemStripDigit");
        if (!fDigits) {
            LOG(error) << "BmnGemDigitQa::Init(): GEM digit branch not found, the task is deactivated";
            return kERROR;
        }
        fHistSignal = new TH2D("hGemSignal", "GEM strip signal;station;signal, ADC counts", 8, 0, 8, 100, 0,
1000);
        return kSUCCESS;
    }

    void Exec(Option_t* /*option*/) override
    {
        for (Int_t i = 0; i < fDigits->GetEntriesFast(); i++) {
            BmnStripDigit* digit = (BmnStripDigit*)fDigits->UncheckedAt(i);
            fHistSignal->Fill(digit->GetStation(), digit->GetStripSignal());
        }
    }

    void Finish() override
    {
        // write the results to the output file of the macro
        FairRootManager::Instance()->GetSink()->WriteObject(fHistSignal, "hGemSignal");
    }

private:
    Bool_t fIsExp;
    TClonesArray* fDigits = nullptr;
    TH2D* fHistSignal = nullptr;

    ClassDefOverride(BmnGemDigitQa, 1);
};

```

The macro below runs the task on a DIGI file with the same choice of the input source as `run_reco_bmn.C`. For simulated files the file is also the parameter input, so the geometry of the simulation is available in the tasks:

```

// run_digi_task.C: runs analysis tasks on a simulated or experimental DIGI file
#include "BmnGemDigitQa.h"

void run_digi_task(TString digiFileName = "bmnsim.root", TString outFileName = "digi_qa.root", Int_t nEvents = 0)
{
    FairRunAna* fRunAna = new FairRunAna();

    // the same choice of the input source as in run_reco_bmn.C
    Bool_t isExp = (BmnFunctionSet::isSimulationFile(digiFileName) == 0);
    if (isExp) {
        Int_t run_period = -1, run_number = -1; // filled from DigiRunHeader
        fRunAna->SetSource(new BmnFileSource(digiFileName, run_period, run_number));
    } else {
        fRunAna->SetSource(new FairFileSource(digiFileName));
        // parameter containers of the simulation file: geometry, magnetic field, run parameters
        FairParRootFileIo* parIo = new FairParRootFileIo();
        parIo->open(digiFileName.Data());
        fRunAna->GetRuntimeDb()->setFirstInput(parIo);
    }
    fRunAna->SetSink(new FairRootFileSink(outFileName));

    fRunAna->AddTask(new BmnGemDigitQa(isExp));

    fRunAna->Init();
    fRunAna->Run(0, nEvents);
}

```

```
root -b -q 'run_digi_task.C("bmnr10446_digi.root", "digi_qa.root")'
```

To use the task in `run_reco_bmn.C`, add it with `fRunAna->AddTask(new BmnGemDigitQa(isExp));`. A task that is used regularly is added to a BmnRoot library (to its `CMakeLists.txt` and `*LinkDef.h`).

Note: For experimental files FairRuntimeDb reports that FairGeoParSet and FairBaseParSet are not initialised, because the digit files contain no parameter containers. Tasks that do not need the geometry can ignore the message; otherwise load the geometry from the Geometry Database as in `run_reco_bmn.C`.

10.3 Alternative: reading a DIGI file directly

For quick checks the tree can also be read directly in a ROOT macro. The macro below works with simulated and experimental files: it chooses the branch names by the type of the file and prints for the first events the event header, the number of GEM digits per station, the FHCAL signal and, for simulated files, the MC particle of the first GEM digit.

```
// read_digi_example.C: loop over the events of a DIGI file and print the event header, the GEM digits per
// station,
// the FHCAL signal and, for simulated files, the MC particle of the first GEM digit
// works with simulated and experimental DIGI files: the branch names are chosen by the type of the file
void read_digi_example(TString digiFileName = "bmnsim.root", Int_t nPrint = 3)
{
    TFile* file = TFile::Open(digiFileName);
    TTree* tree = (TTree*)file->Get("bmndata");
    Bool_t isSimulation = (tree->GetBranch("MCTrack") != nullptr);

    // run information is stored only in experimental files
    DigiRunHeader* runHeader = (DigiRunHeader*)file->Get("DigiRunHeader");
    if (runHeader)
        cout << "Period " << runHeader->GetPeriodId() << ", run " << runHeader->GetRunId() << endl;

    // the same detector data are stored in branches with different names in simulated and experimental files
    TString gemName = isSimulation ? "BmnGemStripDigit" : "GEM";
    TString fhcalName = isSimulation ? "FHCALDigit" : "FHCALDigit";
    vector<TString> branches = {gemName, fhcalName};
    if (isSimulation)
        branches.insert(branches.end(), {"MCEventHeader.", "BmnGemStripDigitMatch", "StsPoint", "MCTrack"});
    else
        branches.push_back("BmnEventHeader.");
    tree->SetBranchStatus("", 0); // read only the needed branches
    for (auto& name : branches)
        tree->SetBranchStatus(name + "*", 1);

    TClonesArray *gemDigits = nullptr, *fhcalDigits = nullptr;
    tree->SetBranchAddress(gemName, &gemDigits);
    tree->SetBranchAddress(fhcalName, &fhcalDigits);
    BmnEventHeader* eventHeader = nullptr; // experimental files
    FairMCEventHeader* mcHeader = nullptr; // simulated files
    TClonesArray *gemMatches = nullptr, *gemPoints = nullptr, *mcTracks = nullptr;
    if (isSimulation) {
        tree->SetBranchAddress("MCEventHeader.", &mcHeader);
        tree->SetBranchAddress("BmnGemStripDigitMatch", &gemMatches);
        tree->SetBranchAddress("StsPoint", &gemPoints); // MC points of the GEM detectors
        tree->SetBranchAddress("MCTrack", &mcTracks);
    } else
        tree->SetBranchAddress("BmnEventHeader.", &eventHeader);

    for (Long64_t iEvent = 0; iEvent < TMath::Min(tree->GetEntries(), (Long64_t)nPrint); iEvent++) {
        tree->GetEntry(iEvent);
        if (isSimulation)
            printf("Event %u (b = %.1f fm):", mcHeader->GetEventID(), mcHeader->GetB());
        else
            printf("Event %u (%s):", eventHeader->GetEventId(), eventHeader->GetEventTimeTS().AsString("s"));

        // GEM: number of strip digits per station
        map<Int_t, Int_t> nGemDigits;
        for (Int_t i = 0; i < gemDigits->GetEntriesFast(); i++)
            nGemDigits[((BmnStripDigit*)gemDigits->UncheckedAt(i))>GetStation()]++;
        printf(" GEM digits per station:");
        for (auto& [station, n] : nGemDigits)
            printf(" %d", n);

        // FHCAL: the digit classes of both file types derive from BmnFHCALDigit
        Double_t fhcalSignal = 0;
    }
}
```

```

    for (Int_t i = 0; i < fhcalDigits->GetEntriesFast(); i++)
        fhcalSignal += ((BmnFHCALDigit*)fhcalDigits->UncheckedAt(i))->GetSignal();
    printf("; FHCAL: %d digits, total signal %.1f", fhcalDigits->GetEntriesFast(), fhcalSignal);

    // simulated files: MC particle with the largest contribution to the first GEM digit
    if (isSimulation && gemMatches->GetEntriesFast() > 0) {
        BmnMatch* match = (BmnMatch*)gemMatches->UncheckedAt(0);
        if (match->GetNofLinks() > 0) {
            FairMCPoint* point = (FairMCPoint*)gemPoints->UncheckedAt(match->GetMatchedLink().GetIndex());
            CbmMCTrack* track = (CbmMCTrack*)mcTracks->UncheckedAt(point->GetTrackID());
            printf("; first GEM digit: PDG %d", track->GetPdgCode());
        }
    }
    printf("\n");
}
}

```

```
root -l 'read_digi_example.C("bmnsim.root")'
```

As in the macro above, switch off the branches that are not needed before the event loop: the events are then read much faster. Most of the data volume of simulated files is in [GeoTracks](#), the MC tracks and the MC points, of experimental files in the strip digits and the TQDC waveforms (Appendix E):

```

tree->SetBranchStatus("*", 0);           // switch off all branches
tree->SetBranchStatus("GEM*", 1);        // switch on the needed ones together with their sub-
branches
tree->SetBranchStatus("BmnEventHeader*", 1);

```

10.4 Quick look with TTree::Draw

Member names can be used directly in `TTree::Draw` expressions:

```

TFile* f = TFile::Open("bmnsim.root");           // experimental file
TTree* t = (TTree*)f->Get("bmndata");
t->Draw("GEM.fStripSignal", "GEM.fStation == 0"); // GEM signals of station 0
t->Draw("Length$(SILICON)");                     // Silicon digits per event
t->Draw("Sum$(FHCALDigi.fSignal)");               // total FHCAL signal per event
t->Draw("TQDC_BC1.fTdcTimes");                   // TDC times of the BC1 counter
t->Draw("TOF400.fAmplitude", "TOF400.fPlane == 10"); // signal widths in a TOF-400
plane
t->Draw("(BmnEventHeader.fInputsAR & 4) != 0");   // trigger input bit 2

TFile* s = TFile::Open("bmnsim.root");           // simulated file
TTree* ts = (TTree*)s->Get("bmndata");
ts->Draw("BmnGemStripDigit.fStripSignal", "BmnGemStripDigit.fStation == 0");
ts->Draw("MCTrack.fPdgCode", "MCTrack.fMotherId == -1"); // primary particles
ts->Draw("BmnGemStripDigitMatch.fLinks.fWeight");   // weights of the MC links

```

10.5 Further examples

- [macro/howTo/readSim.C](#) – reading a simulated file: MC event header, MC tracks, GEM MC points and Silicon digits.
- [macro/howTo/readHeaders.C](#) – [DigiRunHeader](#) and [BmnEventHeader](#) of an experimental file.
- [macro/howTo/readScalers.C](#) – trigger input counters from the event headers.

Appendix A. Summary of the DIGI content

The table lists all objects of simulated and experimental DIGI files with a short description and the main methods of their classes. The objects stored once per file come first, followed by the branches of the event tree `bmndata`, written as `bmndata.<branch>`. For `TClonesArray` branches the methods are those of the array elements, which are accessed with `GetEntriesFast()` and `UncheckedAt(i)`. The notation `GetX/Y/Z()` stands for `GetX()`, `GetY()` and `GetZ()`. Objects marked Sim only or Exp only exist in one type of file only; the details are given in Sections 2–9.

Object in the DIGI file	Content	Main methods
File objects: FairRoot bookkeeping		
<code>bmnrout</code>	Sim only: FairRoot folder structure: which branch belongs to which detector folder (<code>TFolder</code>)	<code>FindObjectAny(name)</code> , <code>GetListOfFolders()</code>
<code>BranchList</code>	Sim only: names of all branches (<code>TList</code> of <code>TObjString</code>); <code>MCTrack</code> in the list marks a simulated file	<code>GetEntries()</code> , <code>At(i)->GetName()</code> , <code>FindObject(name)</code>
<code>FileHeader</code>	Sim only: FairRoot file header (<code>FairFileHeader</code>)	<code>GetRunId()</code> , <code>GetListOfTasks()</code>
<code>ProcessID0</code>	Sim only: ROOT bookkeeping of object references (<code>TProcessID</code>)	–
<code>TimeBasedBranchList</code>	Sim only: FairRoot list of the time-based branches (<code>TList</code>), not used in BmnRoot	<code>GetEntries()</code>
File objects: parameter containers		
<code>BmnFieldPar</code>	Sim only: magnetic field parameters of the simulation	<code>GetType()</code> , <code>GetScale()</code> , <code>MapName(name)</code> , <code>GetPositionX/Y/Z()</code> , <code>IsFieldOff()</code>
<code>FairBaseParSet</code>	Sim only: simulation run parameters; the beam momentum field holds the beam energy per nucleon, the description holds “beam;target”	<code>GetBeamMom()</code> , <code>getDescription()</code> , <code>GetRndSeed()</code>
<code>FairGeoParSet</code>	Sim only: geometry of the simulation (<code>TGeoManager</code>); the description holds the geometry tag	<code>GetGeometry()</code> , <code>getDescription()</code> , <code>GetGeoNodes()</code>
<code>FairGeoPassivePar</code>	Sim only: passive geometry parameters of the simulation	<code>GetGeoPassiveNodes()</code> , <code>GetGeoSensitiveNodes()</code>
<code>BmnFHCALGeoPar</code>	Sim only: geometry parameters of the FHCAL: sensitive and passive nodes	<code>GetGeoSensitiveNodes()</code> , <code>GetGeoPassiveNodes()</code>
<code>BmnNdetGeoPar</code>	Sim only: geometry parameters of the Neutron Detector: sensitive and passive nodes	<code>GetGeoSensitiveNodes()</code> , <code>GetGeoPassiveNodes()</code>
<code><run ID></code>	Sim only: runtime database bookkeeping (<code>FairRtdbRun</code>); the key name is the run ID, the period number of the simulation	<code>getRunId()</code> , <code>getParVersions()</code>
File objects: run information and metadata		
<code>DigiRunHeader</code>	Exp only: run information of the decoder: period, run, times of the first and last events, channel maps, trigger configuration, run counters (<code>DigiRunHeader</code>)	<code>GetPeriodId()</code> , <code>GetRunId()</code> , <code>GetRunStartTime()</code> , <code>GetRunEndTime()</code> , <code>GetTrcBitMap()</code> , <code>GetScalerMap()</code> , <code>T0Map()</code> , <code>GetTrigConfig()</code> , <code>GetBT()</code> , <code>GetAccepted()</code>
<code>RawRunMetadata</code>	Exp only: spill status records, spill map, scaler time distributions, channel maps (<code>BmnMetadataRaw</code>)	<code>SpillStatusVec()</code> , <code>SpillMap()</code> , <code>Scalers()</code> , <code>ExtConditions()</code> , <code>TrcMap()</code> , <code>ScalersMap()</code>
<code>spill</code>	Exp only: spill tree (<code>TTree</code>) with the branches <code>BmnSpillHeader.</code> , <code>MSC</code> and <code>T0Raw</code> , one entry per spill record	<code>BmnSpillHeader: GetSpillId()</code> , <code>GetFirstEventId()</code> , <code>GetLastEventId()</code> , <code>GetNEvents()</code> , <code>Scalers()</code>
Event tree bmndata: event headers		
<code>bmndata.</code> <code>BmnEventHeader.</code>	Exp only: event header of the data acquisition: run and event numbers, period, event time, event type, trigger inputs (<code>BmnEventHeader</code>)	<code>GetRunId()</code> , <code>GetEventId()</code> , <code>GetPeriodId()</code> , <code>GetEventTimeTS()</code> , <code>GetEventTime()</code> , <code>GetEventType()</code> , <code>GetSpillStart()</code> , <code>GetInputSignalsAR()</code> , <code>GetInputSignalsBR()</code> , <code>GetTimeShift()</code>
<code>bmndata.</code> <code>MCEventHeader.</code>	Sim only: generator information of the event: run ID, event number, primary vertex, impact parameter, reaction plane (<code>FairMCEventHeader</code>)	<code>GetRunID()</code> , <code>GetEventID()</code> , <code>GetX/Y/Z()</code> , <code>GetB()</code> , <code>GetNPrim()</code> , <code>GetRotZ()</code>

Object in the DIGI file	Content	Main methods
Event tree bmndata: strip digits		
bmndata.BmnCSCDigit	Sim only: CSC strip digits (BmnCSCDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.BmnGemStripDigit	Sim only: GEM strip digits (BmnGemStripDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.BmnSiBTDigit	Sim only: SiBT strip digits (BmnSiBTDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.BmnSiliconDigit	Sim only: Forward Silicon Detector strip digits (BmnSiliconDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.BmnSiProfDigit	Sim only: Silicon Beam Profilometer strip digits (BmnSiProfDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.BmnVSPDigit	Sim only: VSP strip digits (BmnVSPDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.CSC	Exp only: CSC strip digits (BmnCSCDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.GEM	Exp only: GEM strip digits (BmnGemStripDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.SiBT	Exp only: SiBT strip digits (BmnSiBTDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.SILICON	Exp only: Forward Silicon Detector strip digits (BmnSiliconDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal()
bmndata.VSP	Exp only: VSP strip digits (BmnVSPDigit)	GetStation(), GetModule(), GetStripLayer(), GetStripNumber(), GetStripSignal(), GetTimeSinceBegin_ns()
Event tree bmndata: MC matching of the strip digits		
bmndata.BmnCSCDigitMatch	Sim only: MC points of the CSC digits (BmnMatch, same index as the digit)	GetNofLinks(), GetLink(j), GetMatchedLink(); BmnLink::GetIndex(), GetWeight()
bmndata.BmnGemStripDigitMatch	Sim only: MC points of the GEM digits (BmnMatch)	GetNofLinks(), GetLink(j), GetMatchedLink(); BmnLink::GetIndex(), GetWeight()
bmndata.BmnSiBTDigitMatch	Sim only: MC points of the SiBT digits (BmnMatch)	GetNofLinks(), GetLink(j), GetMatchedLink(); BmnLink::GetIndex(), GetWeight()
bmndata.BmnSiliconDigitMatch	Sim only: MC points of the Silicon digits (BmnMatch)	GetNofLinks(), GetLink(j), GetMatchedLink(); BmnLink::GetIndex(), GetWeight()
bmndata.BmnSiProfDigitMatch	Sim only: MC points of the SiProf digits (BmnMatch)	GetNofLinks(), GetLink(j), GetMatchedLink(); BmnLink::GetIndex(), GetWeight()
bmndata.BmnVSPDigitMatch	Sim only: MC points of the VSP digits (BmnMatch)	GetNofLinks(), GetLink(j), GetMatchedLink(); BmnLink::GetIndex(), GetWeight()
Event tree bmndata: TOF and Drift Chamber digits		
bmndata.DCH	Exp only: Drift Chamber digits (BmnDchDigit)	GetPlane(), GetWireNumber(), GetTime()
bmndata.TOF400	Exp only: TOF-400 digits (BmnTof1Digit)	GetPlane(), GetStrip(), GetSide(), GetTime(), GetAmplitude()
bmndata.TOF701	Exp only: TOF-700 digits (BmnTof701Digit)	GetPlane(), GetStrip(), GetSide(), GetTime(), GetAmplitude()
Event tree bmndata: forward detector digits		
bmndata.FHCalDigi	Exp only: FHCal digits with waveform information (BmnFHCalDigi)	GetModuleId(), GetSectionId(), GetSignal(); members fAmpl, fIntegral, fTimeMax, fZL
bmndata.FHCalDigit	Sim only: FHCal digits (BmnFHCalDigit)	GetModuleId(), GetSectionId(), GetSignal(), GetTime()
bmndata.HgndDigi	Exp only: HGND digits (BmnHgndDigi)	GetArmId(), GetRowId(), GetColumnId(), GetLayerId(), GetSignal(), GetToa(), GetTot()
bmndata.HodoDigi	Exp only: Forward Hodoscope digits with waveform information (BmnHodoDigi)	GetStripId(), GetStripSide(), GetGain(), GetSignal(), GetTime(), GetIsSaturated(); members fAmpl, fIntegral, fTimeMax, fZL
bmndata.HodoDigit	Sim only: Forward Hodoscope digits (BmnHodoDigit)	GetStripId(), GetStripSide(), GetSignal(), GetTime()

Object in the DIGI file	Content	Main methods
bmndata.NdetDigi	Exp only: Neutron Detector digits with waveform information (BmnNdetDigi)	GetArmId(), GetRowId(), GetColumnId(), GetLayerId(), GetSignal(); members fAmpI, fIntegral, fTimeMax, fZL
bmndata.NdetDigit	Sim only: Neutron Detector digits (BmnNdetDigit)	GetArmId(), GetRowId(), GetColumnId(), GetLayerId(), GetSignal(), GetTime()
bmndata.ScWallDigi	Exp only: ScWall digits with waveform information (BmnScWallDigi)	GetCellId(), GetSignal(); members fAmpI, fIntegral, fTimeMax, fZL
bmndata.ScWallDigit	Sim only: ScWall digits (BmnScWallDigit)	GetCellId(), GetSignal(), GetTime()
Event tree bmndata: trigger and beam detectors		
bmndata.BD	Exp only: Barrel Detector TDC digits (BmnTrigDigit)	GetMod(), GetTime(), GetAmp()
bmndata.BdDigit	Sim only: Barrel Detector digits (BmnTrigDigit)	GetMod(), GetTime(), GetAmp()
bmndata.FHCAL	Exp only: TDC digits of the FHCAL trigger channel (BmnTrigDigit)	GetMod(), GetTime(), GetAmp()
bmndata.TQDC_<channel>	Exp only: waveform and TDC times of a beam counter or trigger channel (BmnTrigWaveDigit)	GetMod(), GetValue(), GetNSamples(), GetIntegral(start, stop), GetPeak(start, stop), TdcVector()
Event tree bmndata: MC tracks and trajectories		
bmndata.MCTrack	Sim only: MC tracks (CbmMCTrack)	GetPdgCode(), GetMotherId(), GetPx/Py/Pz(), GetStartX/Y/Z(), GetP(), GetPt(), GetNPoints(detId)
bmndata.GeoTracks	Sim only: particle trajectories for the event display (TGeoTrack), only with saveGeoInfo = kTRUE	GetPDG(), GetNpoints(), GetPoint(i, x, y, z, t)
Event tree bmndata: MC points		
bmndata.BdPoint	Sim only: MC points of the Barrel Detector (BmnBdPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetCopy()
bmndata.CSCPoint	Sim only: MC points of the CSC (BmnCSCPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetStation(), GetModule()
bmndata.FDPoint	Sim only: MC points of the Forward Detector (BmnFDPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetCopy()
bmndata.FHCALPoint	Sim only: MC points of the FHCAL (BmnFHCALPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetAddress()
bmndata.HodoPoint	Sim only: MC points of the Forward Hodoscope (BmnHodoPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetCopy(), GetCopyMother()
bmndata.NdetPoint	Sim only: MC points of the Neutron Detector (BmnNdetPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetAddress()
bmndata.NdetSurfacePoint	Sim only: particles crossing the surfaces of the Neutron Detector (BmnNdetSurfacePoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetPdg(), GetArmID(), GetPlaneID(), GetDirectionID()
bmndata.ScWallPoint	Sim only: MC points of the ScWall (BmnScWallPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetCopy(), GetCopyMother()
bmndata.SiBTPoint	Sim only: MC points of the SiBT (BmnSiBTPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetStation(), GetModule()
bmndata.SiliconPoint	Sim only: MC points of the Forward Silicon Detector (BmnSiliconPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetStation(), GetModule()
bmndata.SiProfPoint	Sim only: MC points of the Silicon Beam Profilometer (BmnSiProfPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetStation(), GetModule()
bmndata.StsPoint	Sim only: MC points of the GEM (CbmStsPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetStation(), GetModule()
bmndata.TOF400Point	Sim only: MC points of the TOF-400 (BmnTOF1Point)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetVolumeUID(), GetModule(), GetStrip()
bmndata.TOF700Point	Sim only: MC points of the TOF-700 (BmnTOF1Point)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetVolumeUID(), GetModule(), GetStrip()
bmndata.VSPPoint	Sim only: MC points of the VSP (BmnVSPPoint)	GetTrackID(), GetX/Y/Z(), GetPx/Py/Pz(), GetTime(), GetEnergyLoss(), GetStation(), GetModule()

Appendix B. Detector identifiers (DetectorId)

Values of the enumeration `DetectorId` (`base/BmnDetectorList.h`) used in the MC tracks (`GetNPoints()`), in the MC points and as the system identifier in the addresses of the forward detectors:

Value	Name	Detector	Value	Name	Detector
0	kREF	Reference plane	14	kSiMD	Silicon Multiplicity Detector
1	kBD	Barrel Detector	15	kSiBT	Silicon Beam Tracker
2	kGEM	GEM detectors	16	kSiProf	Silicon Beam Profilometer
3	kTOF1	TOF-400	17	kARMTRIG	SRC arm triggers
4	kDCH	Drift Chambers	18	kBC	Beam counters
5	kTOF	TOF-700 (old code)	19	kSCWALL	Scintillation Wall
6	kTOF701	TOF-700 (since 2025)	20	kHODO	Forward Hodoscope
7	kZDC	Zero Degree Calorimeter	21	kFHCAL	Forward Hadron Calorimeter
8	kVSP	Vertex Silicon Plane	22	kNDET	Neutron Detector
9	kMWPC	MWPC	23	kHGND	Full-scale Neutron Detector
10	kECAL	Electromagnetic calorimeter	24	kSSD	–
11	kCSC	Cathode Strip Chambers	25	kVACWALL	Vacuum wall
12	kSILICON	Silicon detectors (FSD)	26	kMSC	MSC data
13	kFD	Fragment Detector			

Appendix C. Differences between simulated and experimental DIGI files

The table summarises what differs between the two types of DIGI files.

Item	Simulated file	Experimental file
Producer	<code>run_sim_bmn.C</code> : Geant4 transport and digitizer tasks (FairRoot)	<code>run_raw_bmn.C</code> : raw data decoder <code>BmnDecoder</code>
Objects in the file	FairRoot bookkeeping; parameter containers with the geometry, magnetic field, beam and target	<code>DigiRunHeader</code> , <code>RawRunMetadata</code> , <code>spill</code> tree
Period and run	Period: key name of <code>FairRtdbRun</code> ; no run number	<code>DigiRunHeader</code> (Section 3.1)
Event header	<code>MCEventHeader</code> . (<code>FairMCEventHeader</code>): MC event number, generator information	<code>BmnEventHeader</code> .: DAQ run and event numbers, absolute time (UTC), event type, trigger inputs
Strip digit branches	<code>Bmn<Det>Digit</code> , e.g. <code>BmnGemStripDigit</code>	Detector names, e.g. GEM
MC matching of strip digits	<code>Bmn<Det>DigitMatch</code>	–
Strip signals	Simulated; for CSC, SiBT and VSP on a scale different from the experimental one	ADC amplitude after the pedestal subtraction, above the threshold
VSP digits	Layers 0 and 1; time not set	Layer-2 copies of layer-1 digits; time relative to the VSP trigger time
SiProf digits	Present	–
Forward detector digits	<code>BmnFHCa1Digit</code> , <code>BmnScWallDigit</code> , <code>BmnHodoDigit</code> , <code>BmnNdetDigit</code> in <code>...Digit</code> branches	Derived <code>...Digi</code> classes with waveform information in <code>...Digi</code> branches; HGND in addition
Forward detector signal and time	Energy deposition times the digitizer scale; time of the MC points	Calibrated signal; time for the Forward Hodoscope only
TOF-400, TOF-700	MC points: <code>TOF400Point</code> , <code>TOF700Point</code>	Digits: <code>TOF400</code> , <code>TOF701</code>
Drift Chambers	–	DCH
Beam counters and trigger	<code>BdDigit</code> ; MC points of the FD	<code>TQDC_<channel></code> waveforms and TDC times; <code>BD</code> TDC digits
MC information	<code>MCTrack</code> , MC points, <code>GeoTracks</code> (optional)	–

Appendix D. Fields that are not filled or need care

The following fields exist in the format but are not filled by the current software, contain undefined values, or need special care.

Field	Status	Recommendation
<code>BmnStripDigit::GetStripNoise()</code>	Not filled (–1) in simulated files and for experimental VSP digits	Use for experimental GEM, Silicon, CSC and SiBT digits only
<code>BmnStripDigit::IsGoodDigit()</code>	Always true	–
Strip signals of simulated CSC and SiBT digits	Not converted to ADC counts; the digitizers add no noise and apply no threshold	Do not compare with experimental signals on the digit level
Strip signals of VSP digits	Sim: scale of the Forward Silicon Detector; Exp: ADC values of the VSP readout chips (1–31)	Do not compare simulated and experimental signals directly
<code>BmnVSPDigit::GetTimeSinceBegin_ns()</code>	Sim: not set (undefined value)	Use for experimental data only
VSP digits of layer 2 (Exp)	Copies of layer-1 digits for the strips connected to the third layer	Do not count them twice when summing the signals of a module
<code>GetTime()</code> of <code>FHCalDigi</code> , <code>ScWallDigi</code>	Exp: 0 (not filled)	–
<code>GetTime()</code> of <code>NdetDigi</code>	Exp: undefined values	Do not use
<code>GetTime()</code> of <code>ScWallDigit</code> , <code>HodoDigit</code>	Sim: about 0 (undefined) for a part of the digits	Check the time before using it
<code>GetSignal()</code> of the simulated forward detector digits	Scales of the digitizer configuration: FHCal in MIP units, Forward Hodoscope in Z^2 , ScWall and Ndet energy deposition in GeV	Check the scales before comparing simulated and experimental signals
Waveform members of <code>...Digi</code> classes	<code>fWfm</code> and <code>fFitWfm</code> empty, <code>fToT</code> = 0; fit values 0 and <code>fFitR2</code> = 2 if the fit is not done; <code>fR2history</code> empty	Use <code>fAmp1</code> , <code>fIntegral</code> , <code>fTimeMax</code>
<code>BmnTrigWaveDigit::GetTime()</code> , <code>GetTrigTimestamp()</code> , <code>GetAdcTimestamp()</code>	0	Use the TDC times <code>TdcVector()</code>
<code>BmnTrigDigit::GetTimestamp()</code>	Not used (–1)	–
Branch <code>TQDC_TR_BT`</code> (Exp)	The name contains the character ` (channel BT' of the trigger mapping file)	Quote the name carefully in code and scripts
<code>BmnEventHeader::GetStartSignalTime()</code> , <code>GetStartSignalWidth()</code>	0 for period 9: the decoder takes the start signal from a TDC branch (BC2) that is not in the trigger mapping of the period	Use the TDC times of the <code>TQDC_BC*</code> branches; the DST provides the start time in <code>BmnTrigInfo</code> .
<code>BmnEventHeader::GetTrigInfo()</code>	Counters are 0 in the period 9 data	Use the run statistics of the metadata collection
<code>BmnEventHeader::GetTripWord()</code>	Always false	–
<code>DigiRunHeader::GetNEvents()</code> and run counters (<code>GetBT()</code> , <code>GetBTnBusy()</code> , <code>GetBTAccepted()</code> , <code>GetAccepted()</code> , <code>GetProtection()</code> , <code>GetL0()</code>)	Not filled by <code>run_raw_bmn.C: GetNEvents()</code> is 0, the counters have undefined values; the spill map and the scaler sums are empty	Use the output of the metadata collection (<code>macro/raw/BmnMetadataCollect.C</code>)
spill tree, <code>BmnSpillHeader.</code>	<code>BmnSpillHeader.</code> is filled by the decoder only for periods before 8; the tree can be empty	Use the output of the metadata collection
<code>FairMCEventHeader::GetT()</code>	0 if the generator does not set the event time	–
<code>BmnLink::GetFile()</code> , <code>GetEntry()</code>	–1	–
GeoTracks	Most of the size of a simulated file	Simulate with <code>saveGeoInfo = kFALSE</code> unless needed for the event display

Appendix E. Data volume per event

The tables give the size of the branches per event as stored in the file (compressed), sorted by the size. The values are typical for the Run 9 setup (Xe + Sn at 3 AGeV); they depend on the collision system, the beam energy, the detector configuration and the running conditions.

E.1 Simulated data

The table gives the default output of `run_sim_bmn.C` (`saveGeoInfo = kFALSE`). With `saveGeoInfo = kTRUE` the branch `GeoTracks` adds about 1830 kB per event, about 10 times the size of all other branches.

Branch	Size [kB/event]	Share
MCTrack	39.0	20.6 %
FHCalPoint	35.7	18.9 %
BmnGemStripDigitMatch	22.9	12.1 %
BmnGemStripDigit	21.5	11.3 %
StsPoint	17.1	9.1 %
SiliconPoint	10.2	5.4 %
CSCPoint	7.3	3.9 %
BdPoint	4.5	2.4 %
BmnCSCDigitMatch	4.4	2.3 %
BmnSiliconDigitMatch	4.1	2.2 %
BmnCSCDigit	4.0	2.1 %
BmnSiliconDigit	3.8	2.0 %
FHCalDigit	3.4	1.8 %
VSPPoint	2.2	1.2 %
ScWallPoint	2.0	1.1 %
TOF700Point	1.4	0.7 %
BmnVSPDigitMatch	1.3	0.7 %
BmnVSPDigit	1.2	0.6 %
TOF400Point	0.9	0.5 %
HodoPoint	0.7	0.4 %
BdDigit	0.3	0.2 %
FDPoint	0.2	0.1 %
SiProfPoint	0.2	0.1 %
SiBTPoint	0.2	0.1 %
HodoDigit	0.1	0.1 %
NdetSurfacePoint	0.1	0.1 %
MCEventHeader.	0.1	0.1 %
NdetPoint	0.1	0.1 %
BmnSiProfDigit	0.1	< 0.1 %
BmnSiBTDigit	0.1	< 0.1 %
ScWallDigit	0.1	< 0.1 %
BmnSiProfDigitMatch	< 0.1	< 0.1 %
NdetDigit	< 0.1	< 0.1 %
BmnSiBTDigitMatch	< 0.1	< 0.1 %
Total	189.4	100 %

E.2 Experimental data

Branch	Size [kB/event]	Share
GEM	29.7	39.6 %
TQDC_<channel> (all channels)	21.8	29.1 %
CSC	9.9	13.2 %
SILICON	8.0	10.6 %
FHCalDigi	1.7	2.2 %
TOF701	1.4	1.9 %
ScWallDigi	0.8	1.1 %
VSP	0.6	0.8 %
TOF400	0.5	0.6 %
HodoDigi	0.4	0.5 %
BmnEventHeader.	0.2	0.2 %
SiBT	0.1	0.1 %
NdetDigi	< 0.1	0.1 %
HgndDigi	< 0.1	< 0.1 %
FHCAL	< 0.1	< 0.1 %
BD	< 0.1	< 0.1 %
DCH	< 0.1	< 0.1 %
Total	74.9	100 %