NeXus Instrument Definitions for ISIS muon Data

This paper presents Revision 4 of a new Muon NeXus Instrument Definition (Version 2).

Stephen Cottrell
30 October 2007

Table of Contents

Muon Instrument Definition: Version 2 – 'pulsedTD'

A revision of the Instrument Definition is proposed for the following reasons:

- To correct misunderstandings and misinterpretations of the NeXus format before the definition moves into wider use (PSI are considering adopting NeXus as their RAW file format, and NeXus is a candidate for an exchange file format for sharing muon data between facilities).
- To bring the muon definition closer to that proposed for the neutron instruments at ISIS (see http://www.isis.rl.ac.uk/computing/nexus), from which many of the ideas in this document are taken.
- Where possible, to incorporate ideas and standards set by the NeXus International Advisory Committee (NIAC) (see http://www.nexusformat.org/Main_Page for current discussion). The new definition, however, still requires class definitions not currently defined and/or ratified by the NIAC; these are indicated by the prefix 'IX', and listed in Appendix 1 in preparation for ratification where applicable.
- To provide a RAW file format suitable for storing data written by PC DAE2 and that, in particular, provides the facility for handling unlimited periods in an efficient fashion.
- To provide a definition that is better able to adapt to the wide range of specialist experiments that now run at RAL. The new definition has not attempted to define all entries required for all possible experiments (e.g. RF, laser, etc), it is anticipated that the basic definition (together with the necessary 'read' routines) will be extended to accommodate the additional information as required. Guidelines are provided to assist the extension of the basic definition in a compatible way.

The Instrument Definition described in this document is uniquely identified by the data entries 'IDF_version' ('2') and 'definition' ('pulsedTD') stored within the 'NXentry' group, and the attribute 'creator' ('ISIS') in the group 'NXroot'. It is specifically designed for the ISIS muon instruments (EC and RIKEN-RAL) for storing raw data files, although it should be equally applicable to other pulsed muon sources (with appropriate localisation). For continuous sources, modifications appear inevitable; however, it is hoped that the majority of the meta-data can be written in a compatible fashion. It is vital for reading programs that any modification to entries in the existing definition should be marked by new values written for the 'definition', 'version' and 'creator' entries. A simple extension of the definition need not yield a new version number, although reading programs would then need to use other methods to discover the additional data. The IXuif group provides a convenient group to subsume entries not consistent with the definition.

NXroot

RE	Name	Attribute	Type	Value	Description
	NXroot				top level class
1		**file_name**	NX_CHAR		file name of original NeXus file to assist identification if the external name has been changed
1		**file_time**	ISO 8601		date and time of file creation
0/1		file_format	NX_CHAR	'HDF(4)' | 'HDF(5)' | 'XML'	Format used when creating initial NeXus file
0/1		file_update_tim e	ISO 8601		date and time of last file update at close
0/1		nexus_version	NX_CHAR		version of nexus API used in writing the file
0/1		HDF_version	NX_CHAR		version of HDF library used by nexus to create file
0/1		HDF5_version	NX_CHAR		version of HDF5 library used by nexus to create file
0/1		XML_version	NX_CHAR		version of XML library used to create file
1		**creator**	NX_CHAR		facility or program where file originated
1+	***{run_data}***		NXentry		First entry in file will be the raw data; subsequent entries may be analysed data etc.

NXentry

RE	Name	Attribute	Type	Value	Description
	NXentry				name of entry
1	**IDF_version**		NX_INT32	'2'	version of IDF that NeXus file confirms to
1	**definition**		NX_CHAR	'pulsedTD'	specified the template (DTD name) on which the entry was based, e.g. 'pulsedTD' (pulsed muon, time differential)
0/1		version	NX_CHAR		DTD version number
0/1		URL	NX_CHAR		URL of XML DTD or schema appropriate for file
0/1	program_name		NX_CHAR	'SECI' | 'MCS' | 'CONVERT_NEXUS'	name of creating program
0/1		version	NX_CHAR		version of creating program
0/1		configuration	NX_CHAR		configuration of software e.g. SECI configuration
1	**run_number**		NX_INT32		run number
1	**title**		NX_CHAR		extended title for the entry, e.g. string containing sample, temperature and field
0/1	notes		NX_CHAR		log of useful stuff about the experiment, supplied by the user
1	**start_time**		ISO 8601		start time and date of measurement
1	**end_time**		ISO 8601		end time and date of measurement
0/1	duration		NX_INT32		duration of measurement
0/1		units	NX_CHAR	'second'	
1	**experiment_identifier**		NX_CHAR		experiment number, for ISIS, the RB number
0/1	run_cycle		NX_CHAR		ISIS cycle
1	**user01**		NXuser		details of representative user
0/1+	*{experiment team}*		NXuser		details of members of experiment team, numbered in sequence
1	**run**		IXrun		details of run acquisition parameters, specific to each facility. See Note 0
1	**sample**		NXsample		details of the sample under investigation
1	**instrument**		NXinstrument		details of the instrument used
1+	***{detector_*}***		NXdata		the data collected *(see Note 1)*

0/1+	*{characterization}*		NXcharacterization		contains a link calibration data, e.g. silver runs, spot size measurements.
0/1	uif		IXuif		group containing user supplied information contents of group not defined in definition

NXuser

RE	Name	Attribute	Type	Value	Description
	NXuser				name of user
1	**name**		NX_CHAR		full name of user
0/1		role	NX_CHAR		role of user e.g. ‘PI’, ‘Contact’ etc, multiple roles are allowed.
0/1	affiliation		NX_CHAR		institute
0/1	address		NX_CHAR		address
0/1	telephone_number		NX_CHAR		telephone number
0/1	fax_number		NX_CHAR		FAX
0/1	email		NX_CHAR		email
0/1	facility_user_id		NX_CHAR		unique facility based identifier, at ISIS the User Number

IXrun

RE	Name	Attribute	Type	Value	Description
	IXrun				
1	**number_periods**		NX_INT32	'np'	number of periods used
1	**period_type**		NX_INT32[np]	'1' | '2'	function of period: '1' – DAQ, '2' - DWELL
1	**period_output**		NX_INT32[np]		output bit pattern on period card. If not known, write '0' … 'np'-1 into array
1	**frames_period_requested**		NX_INT32[np]		frames collected in each period before switching, '0' for unlimited frames
1		**frame_type**	NX_CHAR	'good' | 'raw'	type of frame for period switching
0/1	period_labels		NX_CHAR		list of period names, separated by character given as attribute. May use a 2D array of NX_CHAR - TBC
0/1		separator	NX_CHAR		separator character for label list
0/1	raw_frames_period		NX_INT32[np]		raw frames collected for each period
0/1		cycled	NX_INT32	'0' | '1'	'0' – no, '1' - yes
0/1	good_frames_period		NX_INT32[np]		good frames collected for each period
0/1		cycled	NX_INT32	'0' | '1'	'0' – no, '1' - yes
0/1	raw_total_frames		NX_INT32		total raw frames collected in all periods
1	**good_total_frames**		NX_INT32		total good frames collected in all periods
0/1	period_cycles		NX_INT32[np]		number of times data collection took place in each period
0/1	counts_period		NX_FLOAT32[np]		counts collected for each period for all detectors
0/1		cycled	NX_INT32	'0' | '1'	'0' – no, '1' - yes
1	**total_counts**		NX_FLOAT32		total counts collected in all periods
0/1	counts_log		NXlog		counts collected between accumulations as a log against time as typically found in the '.MACQLOG' files
0/1	count_rate		NXlog		count rate at each update

NXsample

RE	Name	Attribute	Type	Value	Description
	NXsample				sample
1	**name**		NX_CHAR		sample name
0/1	chemical_formula		NX_CHAR		element symbols to be arranged in 'Hill System' order: C, H, then other elements alphabetically
0/1	description		NX_CHAR		description of sample
0/1	type		NX_CHAR	'sample' | 'calibration sample' | 'normalisation sample' | 'simulated data' | 'none' | 'sample environment'	type of sample
0/1	situation		NX_CHAR	'atmosphere' | 'vacuum'	
0/1	preparation_date		ISO 8601		date of preparation of sample
0/1	sample_holder		NX_CHAR		description of sample holder
0/1	flypast		NX_INT32	'0' | '1'	flypast collection used '0' – No, '1' – Yes
0/1	geometry		NXgeometry		sample size
0/1	sample_component		NX_CHAR[n_comp]		name of each sample componenet
0/1	thickness		NX_FLOAT32[n_comp]		sample thickness, may be multiple components
0/1		units	NX_CHAR	'milli.metre'	
0/1	mass		NX_FLOAT32[n_comp]		sample mass, may be multiple components
0/1		units	NX_CHAR	'milli.gram'	
0/1	density		NX_FLOAT32[n_comp]		sample density, may be multiple components
0/1		units	NX_CHAR	'milli.gram.centimetre-3'	
0/1	temperature		NX_FLOAT32		linked to most representative sample temperature (to help cataloguing programs)
0/1	magnetic_field		NX_FLOAT32[]		linked to most representative magnetic field (to help cataloguing programs)
0/1	magnetic_field_state		NX_CHAR	'LF' | 'TF' | 'ZF'	current field operating mode
0/1+	temperature_*		NX_FLOAT32		temperature *(see Note 2)*
0/1+		units	NX_CHAR	'kelvin'	units for temperature
0/1+		role	NX_CHAR	'control' | 'sample'	function of temperature
0/1+		value	NX_CHAR	'nominal' | 'derived'	.eq.
0/1+	temperature_*_env		NXenvironment		details of associated hardware
0/1+	temperature_*_l		NXlog		temperature log

	og				
0/1+	magnetic_field_*		NX_FLOAT32[ ]	field setting	magnetic field, may be a vector
0/1+		units	NX_CHAR	‘gauss’	units for field
0/1+		role	NX_CHAR	‘active’ | ‘inactive’	field status
0/1+		direction	NX_CHAR	‘x’ | ‘y’ | ‘z’	field direction
0/1+	magnetic_field_*_env		NXenvironment		details of associated hardware
0/1+	magnetic_field_*_log		NXlog		log of field values – extension to NeXus to enable log of vector quantities

NXenvironment

RE	Name	Attribute	Type	Value	Description
	NXenvironment				
1	**name**		NX_CHAR		name of apparatus
0/1	short_name		NX_CHAR		name displayed on DAE software
0/1	type		NX_CHAR		short code
0/1	description		NX_CHAR		long description
0/1	program		NX_CHAR		version of driver used to collect data, e.g. VI name and version
0/1+	*{hardware log}*		NXlog		log of hardware parameter relating to apparatus, e.g. temperature controller parameters, etc
0/1+	*{sensor name}*		NXsensor		

NXinstrument

RE	Name	Attribute	Type	Value	Description
	NXinstrument				name
1	**name**		NX_CHAR		instrument name
1	**source**		NXsource		details of the muon source used
1	**beamline**		IXbeamline		beamline description
1+	***{detector_*}***		NXdetector		details of detector used
0/1	dae		IXdae		Details of the DAE used. See note 0.
0/1+	***{aperture}***		NXlink		link to aperture in NXbeamline (naming must be identical) apertures used to collimate beam (only link those that the user is interested in)

NXdata

RE	Name	Attribute	Type	Value	Description
	NXdata				
1	**counts**		NX_INT32[ns] | NX_INT32[ns][ntc] | NX_INT32[np][ns][ntc]		linked to detector counts in NXdetector
1		**signal**	NX_INT32	‘1’	
1		**axes**	NX_CHAR	‘[period_index, spectrum_index, raw_time]’	axes definitions
1		**long_name**	NX_CHAR	‘positron counts’ | ‘electron counts’	
0/1	raw_time		NX_FLOAT32[ntc+1]		linked to ‘raw_time’ in NXdetector
0/1		units	NX_CHAR	‘micro.second’	
0/1		long_name	NX_CHAR	‘time’	
0/1	spectrum_index		NX_INT32[ns]		linked to ‘spectrum_index’ in NXdetector *(see Note 3)*
0/1		long_name	NX_CHAR	‘spectrum number’	
0/1	spectrum_labels		NX_CHAR		linked to semicolon separated list of spectrum names
0/1	period_index		NX_INT32[np]		link to ‘period_index’ in NXdetector
0/1		long_name	NX_CHAR	‘period number’	
0/1	period_output		NX_INT32[np]		linked to ‘period_output’ in IXrun - usually integer representing output bit pattern on period card
0/1	period_labels		NX_CHAR		linked to semicolon separated list of period names

NXcharacterization

RE	Name	Attribute	Type	Value	Description
	NXcharacterizat ion			*{type of characterization}*	name of group should reflect type of characterization
0/1	source		NX_CHAR		source file, if missing then use current file
1	location		NX_CHAR		path to data in file
0/1	mime_type		NX_CHAR		if missing, the source file is NAPI readablee
0/1	definition		NX_CHAR		definition data conforms to
0/1		version	NX_CHAR		version of definition

NXsensor

RE	Name	Attribute	Type	Value	Description
	NXsensor				
0/1	model		NX_CHAR		sensor model number, e.g. 'RhFe 11074'
1	**name**		NX_CHAR		name
0/1	short_name		NX_CHAR		name displayed on DAE software, e.g. SECI block name
0/1	attached_to		NX_CHAR		where sensor is attached
0/1	measurement		NX_CHAR	'temperature | pH | magnetic_field | electric field | conductivity | resistance | voltage | pressure | flow | stress | strain | shear | surface_pressure'	type of measurement
0/1	type		NX_CHAR	e.g. 'Rh/Fe | Cernox | Hall | NMR'	type of probe making the measurement
0/1	run_control		NX_BOOLEAN		is data collection controlled by sensor value
0/1	high_trip_value		NX_FLOAT32		if using run control, upper bound
0/1		units	NX_CHAR		
0/1	low_trip_value		NX_FLOAT32		if using run control, lower bound
0/1		units	NX_CHAR		
0/1	value		NX_FLOAT32[ ]		sensor value
0/1		units	NX_CHAR		
0/1	value_log		NXlog		log of sensor value

NXsource

RE	Name	Attribute	Type	Value	Description
	NXsource				name of source
1	**name**		NX_CHAR		facility name
1	**type**		NX_CHAR	'pulsed muon source' | 'low energy muon source'	
1	**probe**		NX_CHAR	'positive muons' | 'negative muons'	
0/1	source_frequency		NX_FLOAT32		accelerator frequency, note that some frames may be 'missing' at target
0/1		units	NX_CHAR	'hertz'	
0/1	source_frequency_log		NXlog		log of source frequency during run
0/1	source_frame_pattern		NX_INT32[rep_len]		frame pattern: '1' frame to target, '0' frame missing at 'frequency', e.g. ISIS target 1 with TS2: '1,1,1,1,0', with a 'rep_len' of '5' and 'period' 100ms.
0/1		rep_len	NX_INT32		repetition length of frame pattern in terms of frames to target
0/1		period	NX_FLOAT32		period of repetition of frame pattern, e.g. 100ms at ISIS target 1, with TS2
0/1		units	NX_CHAR	'milli.second'	units of period
0/1		pulses_per_frame	NX_FLOAT32		number of pulses for each accelerator frame, e.g. '2' at ISIS
0/1	source_energy		NX_FLOAT32		source energy at target
0/1		units	NX_CHAR	'mega.electronvolt'	
0/1	source_current		NX_FLOAT32 | NXlog		source current - this could be an average source current for the run, or logged values
0/1		units	NX_CHAR	'micro.amp'	
0/1	source_current_log		NXlog		log of source current during run
0/1	source_pulse_width		NX_FLOAT32		source pulse width
0/1		units	NX_CHAR		
0/1	target_material		NX_CHAR		e.g. 'carbon'
0/1	target_thickness		NX_FLOAT32		thickness of target
0/1		units	NX_CHAR	'milli.metre'	
0/1	pion_momentum		NX_FLOAT32		pion momentum
0/1		units	NX_CHAR	'mega.electronvolt.c^-1'	
0/1	muon_energy		NX_FLOAT32		muon energy
0/1		units	NX_CHAR	'electronvolt'	
0/1	muon_momentu		NX_FLOAT32		muon momentum

	m				
0/1		units	NX_CHAR	'mega.electronvolt.c^-1'	
0/1	muon_pulse_pattern		NX_FLOAT32[rep_len]		pulse pattern – 'n' number of pulses to instrument each frame, e.g. ISIS target 1 with TS2: '2,2,2,2,0', with a 'rep_len' of '5' and 'period' 100ms, assuming no muon kicker.
0/1		rep_len	NX_INT32		repetition length of pulse pattern in terms of frames to target
0/1		period	NX_FLOAT32		period of repetition of pulse pattern, e.g. 100ms at ISIS target 1, with TS2
0/1		units	NX_CHAR	'milli.second'	units of period
0/1	muon_pulse_width		NX_FLOAT32[]		pulse width for each pulse in frame, e.g. 80ns at ISIS
0/1		units	NX_CHAR	'nano.second'	
0/1	muon_pulse_separation		NX_FLOAT32[]		separation of consecutive pulses in frame, e.g. 300ns at ISIS
0/1		units	NX_CHAR	'nano.second'	
0/1	notes		NX_CHAR		source related messages or announcements, e.g. MCR messages

IXbeamline

RE	Name	Attribute	Type	Value	Description
	IXbeamline				
1	**beamline**		NX_CHAR		name
0/1	diagnostics		IXdiagnostics		container for any beamline diagnostic information.
0/1+	*{beamline component}*		NXbending_magnet | IXquadrupole | IXsolenoid | IXkicker | IXseparator | NXaperture | IXseptum | IXsteering_magnet		beamline components

IXdiagnostics

RE	Name	Attribute	Type	Value	Description
	IXdiagnostics				diagnostics
1+	type		NX_CHAR		type of diagnostic

NXbending_magnet

RE	Name	Attribute	Type	Value	Description
	NXbending_magnet				component name
0/1	set_current		NX_INT32 | NX_FLOAT32		set value on PS
0/1		units	NX_CHAR		
0/1	read_current		NX_INT32 | NX_FLOAT32 | NXlog		value read from PS (may be logged)
0/1		units	NX_CHAR		

IXquadrupole

RE	Name	Attribute	Type	Value	Description
	IXquadrupole				component name
0/1	set_current		NX_INT32 | NX_FLOAT32		set value on PS
0/1		units	NX_CHAR		
0/1	read_current		NX_INT32 | NX_FLOAT32 | NXlog		value read from PS (may be logged)
0/1		units	NX_CHAR		

IXseptum

RE	Name	Attribute	Type	Value	Description
	IXseptum				component name
0/1	set_current		NX_INT32 | NX_FLOAT32		set value on PS
0/1		units	NX_CHAR		
0/1	read_current		NX_INT32 | NX_FLOAT32 | NXlog		value read from PS (may be logged)
0/1		units	NX_CHAR		

IXsolenoid

RE	Name	Attribute	Type	Value	Description
	IXsolenoid				component name
0/1	set_current		NX_INT32 | NX_FLOAT32		set value on PS
0/1		units	NX_CHAR		
0/1	read_current		NX_INT32 | NX_FLOAT32 | NXlog		value read from PS (may be logged)
0/1		units	NX_CHAR		

IXseparator

RE	Name	Attribute	Type	Value	Description
	IXseparator				component name
0/1	set_current		NX_INT32 | NX_FLOAT32		set value on PS
0/1		units	NX_CHAR		
0/1	read_current		NX_INT32 | NX_FLOAT32 | NXlog		value read from PS
0/1		units	NX_CHAR		
0/1	set_volts		NX_INT32 | NX_FLOAT32		set value on PS
0/1		units	NX_CHAR		
0/1	read_volts		NX_INT32 | NX_FLOAT32 | NXlog		value read from PS (may be logged)
0/1		units	NX_CHAR		

IXkicker

RE	Name	Attribute	Type	Value	Description
	IXkicker				component name
0/1	timing		NX_FLOAT32		kicker timing, as defined by 'description' attribute
0/1		units	NX_CHAR	'nano.second'	
0/1		description	NX_CHAR		
0/1	set_value		NX_INT32 | NX_FLOAT32		set value on PS
0/1		units	NX_CHAR		
0/1	read_value		NX_INT32 | NX_FLOAT32 | NXlog		value read from PS (may be logged)
0/1		units	NX_CHAR		

NXaperture

RE	Name	Attribute	Type	Value	Description
	NXaperture				
0/1	description		NX_CHAR		
0/1	shape		NXgeometry		location and shape of aperture
0/1	material		NX_CHAR		material from which aperture is fabricated

IXdae

RE	Name	Attribute	Type	Value	Description
	IXdae				
1	**type**		NX_CHAR	‘MCS’ | ‘DAE2’ | ‘MACS-EXP’	description of DAE hardware
0/1	interface		NX_CHAR	‘CAMAC’ | ‘VME’	
0/1	ICPevent		NXlog		log of ICP events
0/1	ICPmon		NXlog		log of ICP certain data, e.g. beam current
0/1	Status		NXlog		log of run status throughout run

NXdetector

RE	Name	Attribute	Type	Value	Description
	NXdetector			detector name	
1	**description**		NX_CHAR		description
0/1	orientation		NX_CHAR	‘Longitudinal’ | ‘Transverse’ | ‘L’ | ‘T’	general detector arrangement -:legacy entry for MuSR spectrometer at ISIS
1	**counts**		NX_INT32[ns] | NX_INT32[ns][ntc] | NX_INT32[np][ns][ntc]		1, 2 or 3D array of counts, [*period*, spectrum, time bin]
1		**signal**	NX_INT32	‘1’	
1		**axes**	NX_CHAR	‘[period_index, spectrum_index, raw_time]’	axes definitions
1		**long_name**	NX_CHAR	‘positron counts’ | ‘electron counts’	
0/1	raw_time		NX_FLOAT32[ntc+1]		linked to ‘raw_time’ in NXdetector
0/1		units	NX_CHAR	‘micro.second’	
0/1		long_name	NX_CHAR	‘time’	
0/1	histogram_resolution		NX_FLOAT32		histogram resolution in this NXdata group, time differential data must have one of ‘histogram_resolution’ or ‘raw_time’
0/1		units	NX_CHAR	‘nano.second’	
1	**spectrum_index**		NX_INT32[ns]		list of global spectra
0/1		long_name	NX_CHAR	‘spectrum number’	
0/1	spectrum_labels		NX_CHAR		list of labels for spectra, separated by character given as attribute. May use a 2D array of NX_CHAR - TBC
0/1		separator	NX_CHAR		separator character for label list
0/1	period_index		NX_INT32[np]		list of period numbers
0/1		long_name	NX_CHAR	‘period index	
0/1	detector_index		NX_INT32[ns]		‘dectector_index[j]’ is the location of first detector in ‘spectrum_index[j]’ in the array ‘detector_list’ *(see Note 3)*
0/1	detector_count		NX_INT32[ns]		‘detector_count[j]’ is the total number of detectors forming spectrum in ‘spectrum_index[j]’ *(see Note 3)*
0/1	detector_list		NX_INT32[nd]		sorted list of detector numbers *(see Note 3)*

0/1	crate		NX_INT32[nd]		crate number (DAE2 ISIS parameter)
0/1	slot		NX_INT32[nd]		slot number (DAE2 ISIS parameter)
0/1	input		NX_INT32[nd]		input number (DAE2 ISIS parameter)
0/1	type		NX_CHAR[nd]	'scintillator'	type of detector
0/1	voltage		NX_FLOAT32[nd]		PMT voltage
0/1		units	NX_CHAR	'volt'	
0/1	discriminator		NX_CHAR[nd]	'LE' | 'CF'	type of disciminator
0/1	threshold		NX_FLOAT32[nd]		thresholds
0/1		units	NX_CHAR	'milli.volt'	
0/1	geometry		NXgeometry		defining detector positions *(see Note 4)*
0/1	solid_angle		NX_FLOAT32[nd]		solid angle subtended by the detector at the sample
0/1		units	NX_CHAR	'steradian'	
0/1	calibrated_angle s		NX_FLOAT32[nd]		calibrated detector angles
0/1		units	NX_CHAR	'degree'	
0/1		calibration_date	ISO8601		date when angles were determined
0/1		calibration_run	NX_CHAR		run used to determine angles
0/1	dead_time		NX_FLOAT32[np][nd]		array of detector deadtime values *(see Note 5)*
0/1		units	NX_CHAR	'micro.second'	
0/1		calibration_date	ISO8601		date when deadtime file was determined
0/1		calibration_run	NX_CHAR		run used to generate deadtime data
0/1+	grouping_*		NX_INT32[np][ns]		preferred spectrum grouping - 1D array *(see Note 6)*
0/1		primary	NX_INT32		Define preferred grouping, if multiple entries
0/1+		number_groups	NX_INT32	'ng'	number of groups
0/1+	alpha_*		NX_FLOAT32[ng/2][3]		alpha, for pairs of groups defined in 'grouping' - 2D array *(see Note 7)*
0/1+		number_alpha	NX_INT32		number of alpha values defined
0/1	time_zero		NX_FLOAT32[np][ns]		zero time for histograms - reference 'raw_time'
0/1		units	NX_CHAR	'micro.second'	
0/1	first_good_time		NX_FLOAT32[np][ns]		time for first good data - reference 'raw_time'
0/1		units	NX_CHAR	'micro.second'	
0/1	last_good_time		NX_FLOAT32[np][ns]		time for last good data - reference 'raw_time'
		units	NX_CHAR	'micro.second'	

NXlog

RE	Name	Attribute	Type	Value	Description
	NXlog				name of log
0/1	description		NX_CHAR		description of log
0/1	displayname		NX_CHAR		name displayed on DAE software, e.g. SECI block name
0/1	hardware		NX_CHAR		Rig, e.g. 'CCR'
0/1	software		NX_CHAR		version of driver used to collect data - typically VI name and version
0/1	raw_value		NX_FLOAT32 | NX_INT32		log of raw values, e.g. voltage, etc
0/1		units			
1	**value**		NX_FLOAT32[ ] | NX_INT32[]		log of values, e.g. 'temperature values obtained from the 'TLOG' file
1		**units**	NX_CHAR		
0/1	time		NX_FLOAT32[ ]		time stamp of values as offsets from start time
0/1		start	ISO 8601		time of run start
0/1		units	NX_CHAR		units of logged values
0/1	average_value		NX_FLOAT32		average of logged values
0/1		units	NX_CHAR		
0/1	average_value_error		NX_FLOAT32		standard deviation of logged values
0/1		units	NX_CHAR		
0/1	minimum_value		NX_FLOAT32		minimum of logged values
0/1		units	NX_CHAR		
0/1	maximum_value		NX_FLOAT32		maximum of logged values
0/1		units	NX_CHAR		

NXgeometry

RE	Name	Attribute	Type	Value	Description
	NXgeometry				name of component
0/1	{*shape*}		NXshape		shape/size information of component
0/1	{*translation*}		NXtranslation		translation of component
0/1	{*orientation*}		NXorientation		orientation of componenet
0/1	description		NX_CHAR		description
0/1	component_index		NX_INT32		position of the component along the beam path - sample is at 0, components upstream have negative index, components downstream have positive index

NXshape

RE	Name	Attribute	Type	Value	Description
	NXshape				
0/1	shape		NX_CHAR		defining shape
0/1	size		NX_FLOAT32[numobj,nshapepar]		extent of object along local axes – defined by ‘nshapepar’ parameters
0/1		units	NX_CHAR		

NXtranslation

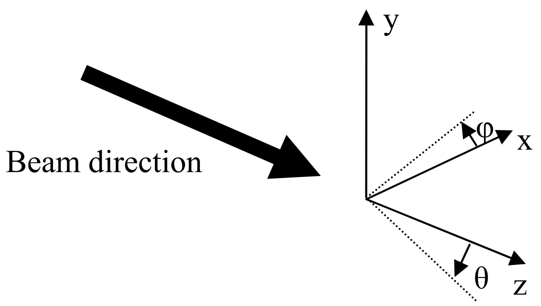
RE	Name	Attribute	Type	Value	Description
	NXtranslation				positions
0/1	distances		NX_FLOAT32[numobj,3]		positions of components
0/1		units	NX_CHAR	‘m’	

NXorientation

RE	Name	Attribute	Type	Value	Description
	NXorientation				orientations
0/1	value		NX_FLOAT32[numobj,6]		orientation information stored as directional cosines

Notes:

Coordinate systems: The coordinate system has been revised since Version 1. Coordinate axes are taken with the z-axis pointing along the incident beam direction, the y-axis defined as upwards perpendicular to the beam in the vertical plane and the x-axis perpendicular to the beam in the horizontal plane pointing left as seen from the source. The origin is taken to be at the centre of the spectrometer, usually about the sample position. Cartesian, cylindrical and spherical polar coordinate systems may be used with angles ϕ (rotation about the z-axis) and θ (rotation about the x-axis), increasing for clockwise rotations viewed from the origin of the coordinate system.



Conventions used in the document: The conventions used in this document are as defined in the paper 'The ISIS NeXus RAW Data File Format' by F.A. Akeroyd. In brief, to help understand this document:

- Variables in bold are mandatory, names in italics are examples and any suitable name may be substituted.
- The column headed 'RE' indicates how often an item name may occur: 0/1 – optional with a maximum of one instance, 1 – a single instance is mandatory, 0/1+ - not mandatory, but may occur more than once.
- Special variables: np – number of periods, ntc – number of time channels in spectra, ns – number of spectra (histograms), nd – number of detectors and ng – number of groups.
- For strings where the variable should only take certain values, this is indicated by 'string 1' | 'string 2'.

Note 0: The groups 'IXrun' and 'IXdae' are defined to store data acquisition parameters. These are facility specific and not defined by a general NeXus group, and are therefore included in a group with the prefix 'IX'. It is anticipated that each facility will define their own version of this group, suing an appropriate prefix, e.g

IXrun – ISIS
RXrun – RIKEN/RAL
PXrun – PSI
JXrun – J-PARC
TXrun – TRIUMF

Note 1: In line with the latest advice from NIAC, raw data is now moved to the NXdetector group, i.e. this contains both the parameters describing the detector and the resulting data. The NXdata group contains links to plottable data. Division of data into multiple NXdetector groups may be made according to convenience; strategies might include having an NXdetector group for each physical detector e.g. 'UP', 'DOWN', etc, or one for each bank of detectors, e.g. the 'Forward/Backward' array – the latter strategy is adopted at ISIS. It is suggested that both the NXdetector and NXdata groups are named according to the convention 'detector_*{function}*'. The situation where a single physical detector is read using multiple data acquisition electronics should be handled by writing multiple NXdetector groups, where an underscore and instance number is appended to the functional name. For example, for a transverse detector array used to collect data using DAE2 and the DASH2 systems, there would be two NXdetector groups (called 'detector_transverse_1' and detector_transverse_2) and two NXdata groups (also called 'detector_transverse_1' and 'detector_transverse_2').

Note 2: Similar naming schemes exist for defining sample environment (temperature_*) and logged (temperature_*_log) variables, immediately showing the dependence to the variable. Sample environment variables should begin with a prefix that uniquely defines the class of sample environment they describe.

Note 3: Data is collected as a series of spectra (histograms). Usually there is a one to one mapping between the spectra and detectors, but occasionally signals from a number of detectors can be combined to form a single spectrum (muon DAE2 may have this facility in the future), and therefore the number of spectra (ns) can be less than the number of detectors (nd). The proposed detector <-> spectrum indexing scheme is suggested, that follows that proposed for neutron data.

> The detectors {i} are numbered i=[1,nd], and their output is mapped onto the spectra {j}. The global spectrum number must be unique, and this number is provided by spectrum_index[j]. The array detector_list contains a list of detector numbers {i}, but they are arranged such that detectors which map to the same spectrum appear sequentially with the start of this sequence given by detector_index[j]. The spectrum spectrum_index[j] will thus have detector_count[j] detectors mapped into it, the actually detector numbers being given by detector_list[k ... k+detector_count[j]-1], where k=detector_index[j].

For a one-to-one mapping between spectra and detectors (usual for μSR experiments), the entries 'detector_index', 'detector_count' and 'detector_list' need not be written.

Note 4: 'geometry' (given in group NXgeometry), 'shape', 'size' and 'translation' uniquely define the instrument detector array. These values will usually be 'engineering' values, with the calibrated azimuthal angles given in 'calibrated_angles'.

Note 5: The deadtime values for each spectrum. The values, read from the default deadtime file 'DT(E)PAR.DAT', are stored as a 1D array where the index represents the detector number.

Note 6: The grouping is represented as a 1D array with the index representing the spectrum number and the value corresponding to the group into which it should be placed. The number of groups is defined as an attribute. Multiple groupings for a given detector bank may be written, with the preferred grouping indicated by the 'primary' attribute. The grouping entry should only be written if the values can be defined. Possible sources of the grouping information may be a default file or grouping tables derived from analysis programme, e.g. 'long.uda' for longitudinal data and 'trans.uda' for transverse data from UDA.

Note 7: The balance parameter, alpha, is given for pairs of groups defined in 'grouping'. Association with a particular grouping entry is by name. Values are stored in a 2D array together with the numbers for the forward and backward groups.

General comments:

- The NeXus Instrument Definition has entries that define how a wide range of information should be stored in the file. Obviously most entries will be of no use to most people, and therefore their inclusion is optional. What will be required is a method of allowing users to add the details they need, without the data acquisition program embarking on a long question and answer session at the start of every run!

- A number of additional quantities were logged by the MuSR instrument and written to the top level of the Version 1 NeXus file. These entries were not defined by the Instrument Definition; Appendix 2 discusses how these have been incorporated into the Version 2 definition.

- Special experiments (such as RF, illumination, E-field) will require additional entries to fully describe experiment parameters. It is recommended that additional entries are placed into the NeXus file structure as follows:

    a) An addendum to the IDF should be defined to show how new entries are written within the muon IDF structure, and clarify their exact meaning. The spirit of the existing IDF should be followed wherever possible when defining new entries.

    b) In general, all new entries taken from SECI blocks will be written to the NXsample group (see note below).

    c) New entries should be named using a unique prefix. Where possible conventions suggested by the NIAC should be followed, otherwise e.g. 'rf_' for RF experiments, 'laser_' for laser experiments, 'current_' for currents. A numbered suffix could be used to allow multiple numbers of a specific device. For example, a simple IDF extension for an RF experiment might include 'rf_frequency', 'rf_power', 'rf_delay', 'rf_pulselength' and 'rf_field', and the definition might be made as follows:

RE	Name	Attribute	Type	Value	Description
	uif				
	rf_frequency		NX_FLOAT32	rf frquency	
		units	NX_CHAR	'MHz'	
		hardware	NX_CHAR	'marconi'	
		display_name	NX_CHAR	name displayed on DAE software	at ISIS, typically SECI block name
		software	NX_CHAR	version of driver used to collect data	at ISIS, typically VI name and version
	rf_delay		NX_FLOAT32	rf delay	relative to extract
		units	NX_CHAR	'us'	
		hardware	NX_CHAR	'stanford'	
		display_name	NX_CHAR	name displayed on DAE software	at ISIS, typically SECI block name
		software	NX_CHAR	version of driver used to collect data	at ISIS, typically VI name and version
	rf_pulselength		NX_FLOAT32	rf pulse length	
		units	NX_CHAR	'us'	
		hardware	NX_CHAR	'CAEN Dual Timer'	
		display_name	NX_CHAR	name displayed on DAE software	at ISIS, typically SECI block name
		software	NX_CHAR	version of driver used to collect data	at ISIS, typically VI name and version
	rf_power		NX_FLOAT32	desired RF power	
		units	NX_CHAR	'W'	
		hardware	NX_CHAR	'AMT'	
		display_name	NX_CHAR	name displayed on DAE software	at ISIS, typically SECI block name
		software	NX_CHAR	version of driver used to collect data	at ISIS, typically VI name and version
	rf_power_log		NXlog	power log	
	rf_field		NX_FLOAT32	rf field strength	
		units	NX_CHAR	'G'	
		rf_field_vector	NX_FLOAT[3]	vector describing orientation of RF field	
		display_name	NX_CHAR	name displayed on DAE software	at ISIS, typically SECI block name
		software	NX_CHAR	version of driver used to collect data	at ISIS, typically VI name and version

- Various schemes for defining the location of the entry and mapping the SECI block name to a suitable NeXus name have been considered. It was decided that at the SECI 'user' level to provide a single option for storing block information within the 'IXuif' group, with the block name written unchanged. At the SECI 'operator' level a further option will be presented, allowing data to the written to the NXsample group with appropriate (and definable) mapping of the block name. In general, non-specific information about the run should be added to IXuif, while information that adds to the description of the sample and its environment should be included in NXsample.

Appendix 1: Class definitions unique to 'muonTD'

The following class names are used in the definition of 'muonTD', but have not been defined and ratified by the NIAC. Except where indicated, the intention is to seek ratification; in one case an extension of the agreed definition is required.

- *IXrun*
  parameters specific to data acquisition at ISIS – <u>not for ratification</u>
- *IXdae*
  information about the data acquisition electronics – <u>not for ratification</u>
- *IXuif*
  additional data/information, not defined by the IDF – <u>not for ratification</u>
- *IXbeamline*
  information on the beamline between target and instrument
- *IXdiagnostics*
  placeholder for diagnostic information on beamline function
- *NXbending_magnet*
  (aka dipole) defined by the NIAC, but definition requires extension
- *IXquadrupole*
  quadrupole magnet
- *IXsolenoid*
  solenoid magnet
- *IXkicker*
  beam kicker
- *IXseparator*
  particle separator
- *IXseptum*
  septum magnet

Appendix 2: Incorporating existing SECI logged values

A number of additional values were logged by the SECI configurations and, independent of the existing IDF, written to the top level of the Version 1 NeXus file. These entries were not defined by the Instrument Definition; this section discusses how these have been incorporated into the Version 2 definition.

- *beamlog_current*
  ISIS beam current
  Map to: '/entry/instrument/source/source_current_log'
- *beamlog_freq*
  ISIS running frequency
  Map to: '/entry/instrument/source/source_frequency_log'
- *CountRate*
  count rate at each update (typically 10s)
  Map to: '/entry/run/count_rate'
- *Field_Danfysik*
  field values of Danfysik (main) magnet
  Map to NXsample, see following example
- *Field_T20*
  field values of TF20 (transverse) magnet
  Map to NXsample, see following example
- *Field_ZF_Magnitude*
  magnitude of measured field by zero field correction probe
  Map to NXsample, see following example

Example: Writing magnetic field data for MuSR (assume 300G longitudinal field):

NXsample (section)		
magnetic_field_state		'LF'
magnetic_field_1		0
	units	'Gauss'
	magnetic_field_direction	'x'
magnetic_field_1_env		details of hardware generating field (see following table)
magnetic_field_1_log		values currently written to 'Field_T20'
magnetic_field_2		300
	units	'Gauss'
	magnetic_field_direction	'z'
magnetic_field_2_env		details of hardware generating field (see following table)
magnetic_field_2_log		Values currently written to 'Field_Danfysik'
magnetic_field_3		0

	units	‘Gauss’
magnetic_field_3_env		details of hardware generating field (see following table)
magnetic_field_3_log		Values currently written to ‘Field_ZF_Magnitude’
NXenvironment, magnetic_field_1_env		
name		‘Gossen TF’
type		‘TF’
short_name		‘Field_T20’
program		‘<VI name and version>’
NXenvironment, magnetic_field_2_env		
name		‘Danfysik’
type		‘LF’
short_name		‘Field_Danfysik’
program		‘<VI name and version>’
NXenvironment, magnetic_field_3_env		
name		‘ZF’
type		‘Zero Field’
short_name		‘Field_ZF_Magnitude’
program		‘<VI name and version>’

- *ICPevent*
  Map to ‘/entry/instrument/dae/ICPevent’
- *ICPmon*
  Map to ‘/entry/instrument/dae/ICPmon’
- *Status*
  Map to ‘/entry/instrument/dae/Status’
- *Steer_HSM*
  current set for horizontal steering magnet
  Map to: ‘/entry/instrument/beamline/Steer_HSM/read_current’, where ‘Steer_HSM’ is defined as type ‘NXbending_magnet’ and ‘read_current’ is type ‘NXlog’
- *Steer_VSM*
  current set for vertical steering magnet
  Map to: ‘/entry/instrument/beamline/Steer_HSM/read_current’, where ‘Steer_VSM’ is defined as type ‘NXbending_magnet’ and ‘read_current’ is type ‘NXlog’
- *Temp_Cryostat*
  temperature recorded by control sensor
  Map to NXsample, see following example
- *Temp_D*
  derivative value set on the controller
  Map to NXsample/NXenvironment of controlling temperature, see following example

- *Temp_I*
  integral value set on the controller
  Map to NXsample/NXenvironment of controlling temperature, see following example

- *Temp_P*
  proportional value set on the controller
  Map to NXsample/NXenvironment of controlling temperature, see following example

- *Temp_MaxPower*
  Maximum heater power set on the controller
  Map to NXsample/NXenvironment of controlling temperature, see following example

- *Temp_Power*
  Power delivered by controller
  Map to NXsample/NXenvironment of controlling temperature, see following example

- *Temp_SP*
  temperature recorded by controlling sensor, usually the 'Cryostat'
  This may be logged to NXsample as an additional sensor or as an equivalent of the Cryostat temperature (as in following example); however, we should consider whether it might be better (or possible) to make a link to the actual controlling sensor ('alias' suggested in discussions with FAA).

Example: Writing temperature data for MuSR (assume 30K in the CCR):

NXsample (section)		
temperature_1		30
	units	'Kelvin'
	role	'control'
temperature_1_env		environment information
temperature_1_log		values currently written to 'Temp_SP'
temperature_2		30
	units	'Kelvin'
	role	'sample'
temperature_2_env		environment information
temperature_2_log		values currently written to 'temperature_log_1'
NXenvironment, temperature_1_env		
name		'CCR'
type		'CCR'
short_name		'Temp_SP'
program		'<VI name and version>'
Temp_D		logged values previously written to '/Temp_D'
Temp_I		logged values previously written to '/Temp_I'
Temp_P		logged values previously written to '/Temp_P'
Temp_MaxPower		logged values previously written to '/Temp_MaxPower'

Temp_Power		logged values previously written to '/Temp_Power'
NXenvironment, temperature_2_env		
name		'CCR'
type		'CCR'
short_name		'Temp_Cryostat'
program		'<VI name and version>'