Bunch Pattern Server Manual

Release 0.0

Lars Fröhlich

Dec 17, 2019

CONTENTS

1 Introduction 1

2 Pulse Patterns & The Pattern Builder 3

3 Pattern Sequences 5

4 Beam Limits 7

5 Dark Current Suppression (DCS) 9

6 CONTROL Location 11

7 PATTERN_BUILDER Location 15

8 Technical Information 17

Index 19

CHAPTER

ONE

INTRODUCTION

The bunch pattern server controls the timing pattern of the machine. It allows the construction of complex bunch patterns and allows clients to reduce the number of bunches when they intend to perform potentially harmful actions like magnet cycling. It also offers a dark current suppression mechanism that can stop the transport of gun dark current into the linac. More details are found under *Beam Limits* and *Dark Current Suppression (DCS)*. The properties of the main server location are explained under *CONTROL Location*.

Other options

If you want to dig deeper, have a look at the *Technical Information* section.

CHAPTER

TWO

PULSE PATTERNS & THE PATTERN BUILDER

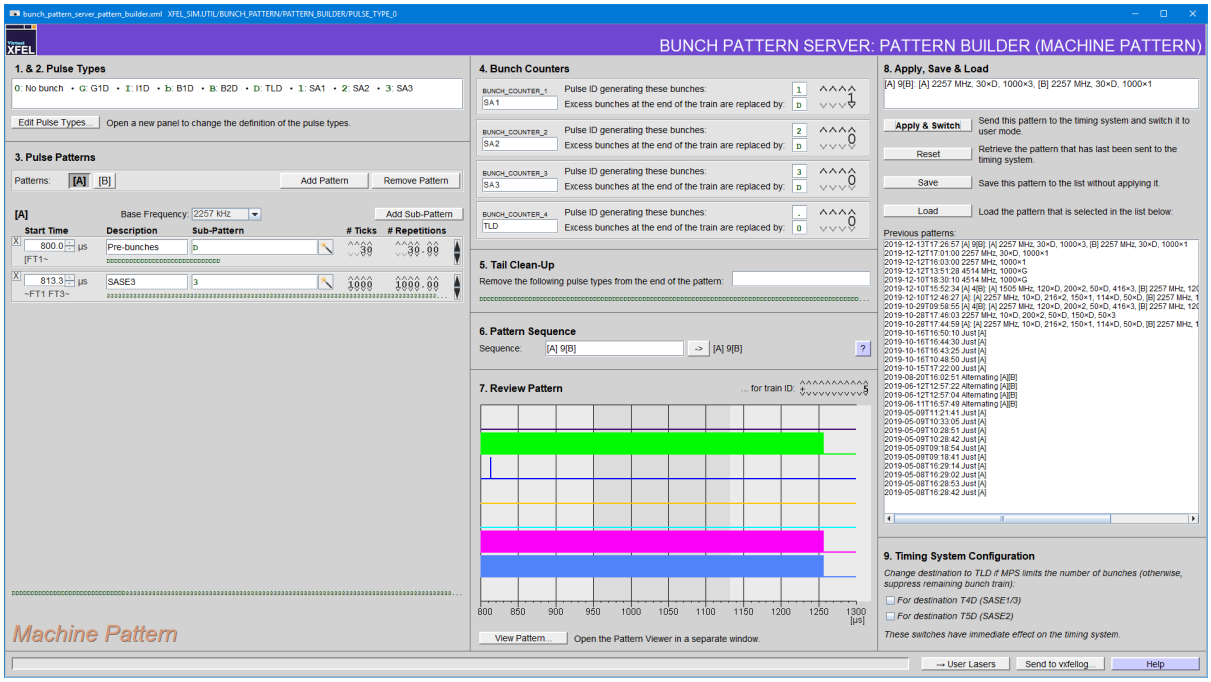

Fig. 2.1: Screenshot of the “pattern builder” jddd panel

A pulse pattern describes all the bunches and triggers within a macropulse or bunch train. In this server, it is represented by a series of characters like `DD11112`. Each of the characters stands for a specific pulse type, so that the pattern `DD11112` means a macropulse with 2 pulses of type `D`, then four of type `1`, followed by a single pulse of type `2`. The distance between the pulses can be chosen with the *base frequency* setting – a base frequency of 1 MHz means that the base clock for the pattern performs 1 *tick* per microsecond, hence *tick* is used as a unit representing “one character in the pattern”.

A pattern is composed from one or multiple sub-patterns. Each sub-pattern is defined by a series of characters and by how often it should be repeated. For example, a sub-pattern `12` with 3 repetitions would expand into `121212`. A sub-pattern can also be repeated a fractional number of times; in these cases it is often easier to refer to the total number of *ticks* or characters in the total pattern: A sub-pattern `ABCD` repeated 2.5 times would yield `ABCDABCDAB`, which is a total of 10 characters or *ticks* of the base clock.

The expansions of all sub-patterns are concatenated to form the final pulse pattern.

CHAPTER

THREE

PATTERN SEQUENCES

A pattern sequence describes how the bunch pattern should change from macropulse to macropulse. It can be written as a short string whose main building blocks are the *tags* of the defined patterns like `[A]` or `[B]`, complete with their surrounding square brackets. The server cycles through this list of patterns periodically. For example:

`[A]` This is the default pattern sequence. It instructs the server to use pattern [A] for all macropulses.

`[A][B][C]` This pattern sequence makes the server alternate between three patterns; macropulse 0 uses pattern [A], macropulse 1 pattern [B], macropulse 2 pattern [C], macropulse 3 again pattern [A], and so on.

In addition, numbers can be inserted in a pattern sequence as if they were coefficients for a pattern entry, i.e. before a pattern that should be repeated. For instance:

`9[A][B]` This pattern sequence alternates between two patterns; macropulse 0-8 use pattern [A], macropulse 9 uses pattern [B], macropulse 10-18 again pattern [A], and so on.

CHAPTER

FOUR

BEAM LIMITS

A beam limit is like a speed limit — it says something like “at this location, so many bunches or less are allowed”. The bunch pattern server keeps a list of these beam limits and enforces them — it makes sure that no more than the allowed number of bunches can reach the location. The limits themselves are requested by client applications. For example, before the magnet middle layer server cycles a magnet, it sends a limit request to the bunch pattern server, stating “0 bunches allowed at the position of magnet XYZ”. This limit will stay in place until the magnet server cancels it (when the cycling is finished).

Beam limits are always temporary. Therefore, every beam limit is associated with a timeout period of few seconds or minutes. When the limit has been active for the specified time, it is automatically removed. For operators, there is also the possibility to clear all limits.

Dark current suppression (DCS) is a topic of its own, see *Dark Current Suppression (DCS)*. It can be requested through a beam limit with “-1” as the maximum number of bunches.

Beam limits are controlled through the *CONTROL Location*.

CHAPTER

FIVE

DARK CURRENT SUPPRESSION (DCS)

The server can suppress the transport of dark current from the gun into the linac by shifting the RF pulses of the gun and of the first acceleration module(s) out of temporal overlap. Dark current suppression is configured through the *CONTROL Location*.

CHAPTER

SIX

CONTROL LOCATION

The main controls for the timing pattern are found under the CONTROL location. Using this location, beam can be allowed or inhibited, the number of bunches can be changed, beam limits can be set or removed, and dark current suppression can be enabled or disabled.

The location has the following properties:

ALLOW_USER_BUNCH_CONTROL_n Allow or disallow user control over the number of bunches. If this property is nonzero, the number of bunches of type n can be modified via the USER_NUM_BUNCHES_REQUESTED_n property. Otherwise, an attempt to use that user property generates an error message.

BEAM_ALLOWED Determines whether injector lasers may be fired or not. If BEAM_ALLOWED = 0, the injector laser bits of all timing pattern entries are cleared along with the destination and charge limit bits. In legacy mode, all numbers of bunches are set to zero.

BUNCH_COUNTER_*_DESC A user-defined name for the bunches controlled by a certain bunch counter (1 to 4). Typical values are “SA1”, “SA2”, or “Pre-bunches”.

CLEAR_ALL_LIMITS Remove all beam limits at once. Not to be used by applications, scripts, whatever.

CLEAR_LIMIT Clear a beam limit. The comand expects the unique ID of a previously established beam limit as an integer parameter. There is no feedback on whether the operation succeeded or no.

DCS_ACTIVE Controls the *Dark Current Suppression (DCS)* feature: If set to 1, the timing delay of an RF station is shifted with respect to the pulse of the gun so that no more dark current is transported. If set to 0, the delay between the stations is zero. If USE_DCS is on, the dark current suppression is activated or deactivated by the server itself depending on beam limits, but DCS_ACTIVE can still be read from.

INJ_LASER_n_CHARGE The desired charge for injector laser n. This value is only effective in user-defined pattern mode and it controls the “maximum charge” bits in the timing pattern. The values set here are also copied to the legacy charge properties on the timing server for compatibility with the charge feedback.

INJ_LASER_n_DESC A user-defined name for injector laser n (e.g. “short pulse laser”).

INJ_LASER_n_ID The integer ID for injector laser n. In other words, n.

LIST_OF_LIMITS A text-only list of the currently active beam limits, separated by newlines.

NUM_BUNCHES_CURRENT.(bunch_type) This read-only property contains the current number of bunches of the corresponding bunch type. If the timing system is in legacy mode, this is equivalent to the number of bunches set in the timing server. If the system is in user-defined mode, the server analyzes the timing pattern that it writes to the timing system: Each bunch type has an associated "active" pulse type that produces a certain entry (32-bit word) in the pattern. The server simply counts the number of occurrences of this word. Note that the low-level timing server may add entries through features such as "special bunches". These bunches are not counted by the bunch pattern server.

NUM_BUNCHES_REQUESTED.(bunch_type) The requested number of bunches of the corresponding bunch type. The actual number of bunches that is produced can be lower because of many reasons (e.g. BEAM_ALLOWED = 0, beam limits).

NUM_LIMITS The number of active beam limits. This number does not give any information on whether any of the beam limits actually applies to the current operation mode and timing pattern of the machine.

PATTERN_SEQUENCE The currently active pattern sequence, e.g. `[A][B]`. See *Pattern Sequences* for more information.

PATTERN_SERIALIZATION A string containing machine-readable information about the current pattern setup. For internal use only.

PATTERN_VIEW_n A "time-domain spectrum" that display a certain aspect of the current pattern in an oscilloscope-like manner, e.g. bunches from injector laser 1 or bunches belonging to a certain subtrain. The corresponding .COMMENT property describes what is shown. The various PATTERN_VIEW_n properties are vertically offset so that they can be plotted in a common plot.

PATTERN_VIEW_n.COMMENT A short description of what the corresponding PATTERN_VIEW_n property shows, e.g. bunches created by "Injector laser 1".

REM_CONTROL.APINAME The remote address of an x2timer's REM_CONTROL property. This property is used to switch between legacy and user-defined timing modes. For example: "XFEL_SIM.DIAG/TIMER.CENTRAL/MASTER/REM_CONTROL"

REM_TABLE.APINAME The remote address of an x2timer's REM_TABLE property. This property is used to send timing patterns to the timer server. For example: "XFEL_SIM.DIAG/TIMER.CENTRAL/MASTER/REM_TABLE"

RENEW_LIMIT Renew the lifetime of a beam limit (restart the timeout period). This command expects the unique ID of a previously established beam limit as an integer parameter.

REQUEST_LIMIT Set up a new beam limit. The command expects a string parameter in the format "component|num_bunches|timeout|cause" and returns a unique ID for the limit as an integer.

component Name of a beamline component as listed in the component database (e.g. "BK.24.I1"). The location of this component defines which bunches are influenced by the limit, and if the limit has an effect at all — if it is in a section of the machine that cannot get beam anyhow, no action is taken.

num_bunches The maximum number of bunches that are allowed at the position of the component. The value -1 has a special meaning: It allows no bunches and enables the dark current suppression mechanism, see *Dark Current Suppression (DCS)*.

timeout The desired timeout period in seconds. The beam limit is automatically canceled after this period (if it has not been renewed in the meantime). Note that the server can modify the timeout to reasonable values. Five minutes (300 s) is a reasonable maximum.

cause A string describing the cause of (or reason for) the beam limit. The first letter of the string should be uppercase, the remainder lowercase, e.g. "Magnet cycling". The name of the component should not be included in this description.

Examples for the request string:

"BK.24.I1|0|120|Magnet cycling" Prohibit (0) the production of bunches that could reach BK.24.I1 for 120 seconds.

"BK.24.I1|-1|0|Magnet cycling" Prohibit the production of bunches that could reach BK.24.I1 and (-1) move the RF pulse of the first modules to suppress dark current transport. A default timeout is used.

"OTRA.426.B2|3|10000|Scintillator screen inserted" Limit the number of bunches that could reach OTRA.426.B2 to 3. The timeout is not set to 10000 s, but it is limited by the server.

STAND_ALONE Configure the server for stand-alone mode. If this value is nonzero, the server does not manage any beam limits and it does not provide dark-current suppression. It can still be used to configure timing patterns, however.

TRIM_ALL_SUBTRAINS When an integer is written to this property, the number of bunches for each bunch type is reduced to no more than the specified value. Upon reading, this property returns zero. Note that this property does not establish a beam limit - it simply changes the desired number of bunches.

For example, assume that the requested number of bunches for types 1, 2, and 3 is 100, 50, and 0 bunches, respectively. Writing 20 to TRIM_ALL_SUBTRAINS will then reduce the number of bunches to 20, 20, and 0.

ZMQ_REM_TABLE The remote address of an x2timer's REM_TABLE property. The bunch pattern server subscribes to this address via ZeroMQ to receive its periodic triggers and to check the current timing pattern. For example: "XFEL_SIM.DIAG/TIMER.CENTRAL/MASTER/REM_TABLE"

CHAPTER

SEVEN

PATTERN_BUILDER LOCATION

The pattern builder is a sandbox in which timing patterns for the machine can be constructed and reviewed interactively. The changes made in the pattern builder are only sent to the machine on request via the APPLY property.

APPLY Send the currently defined pattern to the timing system and switch the timing system into user-defined mode. This command can return an error if the pattern is invalid.

CHAPTER

EIGHT

TECHNICAL INFORMATION

Author: Lars Fröhlich

Server type: DOOCS middle layer

Debian package name: doocs-bunch-pattern-server

Installed on:

VXFEL — xfelcpusimtime1:/export/doocs/server/bunch_pattern_server
XFEL — xfelcputime1:/export/doocs/server/bunch_pattern_server
FLASH — flashcputime1:/export/doocs/server/bunch_pattern_server
TEST.XFEL — exflutca3:/export/doocs/server/bunch_pattern_server

Source code: GIT repository

Source code documentation: Doxygen pages

INDEX

A

ALLOW_USER_BUNCH_CONTROL_n, 11
APPLY, 15

B

BEAM_ALLOWED, 11
BUNCH_COUNTER_*_DESC, 11

C

CLEAR_ALL_LIMITS, 11
CLEAR_LIMIT, 11

D

DCS_ACTIVE, 11

I

INJ_LASER_n_CHARGE, 11
INJ_LASER_n_DESC, 11
INJ_LASER_n_ID, 11

L

LIST_OF_LIMITS, 12

N

NUM_BUNCHES_CURRENT.(bunch_type), 12
NUM_BUNCHES_REQUESTED.(bunch_type), 12
NUM_LIMITS, 12

P

PATTERN_SEQUENCE, 12
PATTERN_SERIALIZATION, 12
PATTERN_VIEW_n, 12
PATTERN_VIEW_n.COMMENT, 12

R

REM_CONTROL.APINAME, 12
REM_TABLE.APINAME, 12
RENEW_LIMIT, 12
REQUEST_LIMIT, 12

S

STAND_ALONE, 13

T

TRIM_ALL_SUBTRAINS, 13

Z

ZMQ_REM_TABLE, 13