

# Table of Contents

<b>mapping</b> .....	<b>1</b>
<b>Introduction</b> .....	<b>2</b>
<b>xml_diff</b> .....	<b>3</b>
<b>xml_diff_cosmic1to2</b> .....	<b>4</b>
<b>dumpCTP</b> .....	<b>5</b>
Specifics of the mapping.....	5
<b>ctpid_printer</b> .....	<b>6</b>
<b>Usage</b> .....	<b>7</b>
xml_diff.....	7
dumpCTP.....	7
ctpid_printer.....	7

# mapping

# Introduction

When changing from one TriggerMenu to another old ByteStream datafiles will no longer work properly, as CTPID's, TriggerNames and other attributes are changed. It is therefore desired to have a program that maps the changes from one TriggerMenu to another, thus allowing a transformation of the ByteStream file to mimic data taking with the new TriggerMenu.

The following describes scripts made to map ATLAS ByteStream files from one TriggerMenu to another TriggerMenu. The mapping will change CTPID and TriggerType in both the CTPFragment and the FullEventFragment accordingly.

It should be noted that the transformation is not perfect, as Triggers are omitted or merged from one TriggerMenu to another and as the mapping only changes the CTPID and TriggerType.

It consist of four different scripts:

- xml\_diff.py
- xml\_diff\_comic1to2.py
- dumpCTP\_0.95.py
- ctpid\_printer\_0.2.py

## xml\_diff

This script does a CTPID mapping, mapping the CTPID's from the one TriggerMenu to another TriggerMenu. The mapping is done using XML TriggerMenu files, finding the differences between the two menus and generating a python dictionary with the CTPID map in a python file named `ctpidmap.py`. Furthermore a dictionary for the TriggerType mapping is written in the same file. This has to be modified manually.

There are options to compare both different TriggerMenus from the same version (ie. 1.15 and 1.16) and to compare TriggerMenus from different TriggerMenuVersions (ie Cosmic\_2009\_v1 to Cosmic\_2009\_v2).

# xml\_diff\_cosmic1to2

The cosmic1to2 is a version specific for mapping between version 1 and version 2 of the cosmic TriggerMenus. The specific changes are mapping of:

- RD0\_FILLED -> RD0\_EMPTY
- L1\_MU0\* -> L1\_MU0\_EMPTY
- L1\_MU6\* -> L1\_MU6\_EMPTY

and for the mbts:

- L1\_MBTS\_A -> L1\_MBTS\_A\_EMPTY
- L1\_MBTS\_C -> L1\_MBTS\_C\_EMPTY

etc. ...

# dumpCTP

The `dumpCTP.py` changes the actual ByteStream files. As input it takes a ByteStream file and the `ctpmap.py` generated by `xml_diff.py` with the relevant TriggerMenus.

The program uses the `EventFormat`, `EventApps` and `EventStorage` (`libpyevent_storage`) in order to change the CTPIDs and TriggerTypes as stated in the `ctpidmap` generated with `xml_diff`.

## Specifics of the mapping

The mapping of CTPIDs and TriggerTypes will be done in the `FullEventFragment` and the `CTPFragment`.

In the `FullEventFragment` of the binary ByteStream file the CTPIDs are stored in the class `'lv11_trigger_info()` as a 24bit string(?) where the first 8 words are the CTPIDs before prescale, the next 8 are the CTPIDs after prescale, and the last 8 words are the CTPIDs after veto. The words are a binary representation of a list of decimal numbers - the actual CTPIDs.

- Example from a 8bit string; `[192,0,0,0,0,0,0,0]` gives the CTPIDs `[6,7]` ( $2^{**6}+2^{**7} = 192$ )

After having translated into decimal numbers, the mapping is done, and the CTPIDs is again transformed into the binary representation which is stored in the new datafile. This will be written after all events in the ByteStream file have been processed.

The same procedure is done for the TriggerTypes - taking the dictionary (`triggermap`) from the `ctpidmap` file and mapping it accordingly.

In the `CTPFragment` the mapping is done in the exact same way although CTPIDs are stored in the payload of the fragment, which also contains other information that are not important for the mapping. The `CTPFragment` is actually the really important place to do the mapping, as it is from here the level 2 triggers gets their information.

### Document links

- The ATLAS Event Format (v4.0): [EDMS](#)
- The CTPFragment header and the FullEventFragment header: [ISSCVS](#)

## ctpid\_printer

ctpid\_printer is a small program that takes a ByteStream datafile and prints the level 1 identifier, level 1 TriggerType and the CTPIDs of the triggers that fired in the events - both before and after prescale and after veto. If provided with a TriggerMenu file (XML), it will paste the trigger names defined in the TriggerMenu.

# Usage

## xml\_diff

The program is run by typing: `python xml_diff.py < arg > < OldTriggerMenuFile > < NewTriggerMenuFile >`. The arguments, `-a` and `-s` are for comparing TriggerMenus from completely different menus as described above and `-sv` is for same version.

- `-a`: "all" - gives errors and warning as both files and as output in terminal.
- `-s`: "simple" - only writes errors and warnings to a file without printing to terminal.
- `-sv`: "same version" - for files of same version, as this avoids some false errors that otherwise show up when comparing files from same version.

### The created files:

- `ctpidmap.py`:
  - ◆ A python file with a dictionary of the ctpid mapping and trigger names as comments and a map of trigger types.
- `errors.dat`:
  - ◆ Triggers from both files that have NOTHING in common
- `warnings.dat`:
  - ◆ Triggers where BGRP 0/1 was mapped to BGRP 0 and BPTX 0/1

## dumpCTP

The program is run by typing: `python dumpCTP.py < Input ByteStream file > ctpidmap < Output ByteStream file >`, where `ctpidmap` is the `ctpidmap.py` made by `xml_diff.py` **OBS**. If you need to change the TriggerType mapping do this manually in the `triggermap` dictionary in `ctpidmap.py` before running `dumpCTP` or in `xml_diff`.

## ctpid\_printer

The program is run with: `python ctpid_printer.py < Input ByteStream file > ( < menu-file.xml > )`  
The last argument ( `< menu-file.xml >` ) can be provided to return TriggerNames as well.

-- AskEmilJensen - 2009-08-28 -- MortenBadenseoe - 2009-08-28

---

This topic: [Sandbox > AskEmilJensen](#)

Topic revision: [r4 - 2009-08-31 - AskEmilJensen](#)



Copyright &© 2008-2021 by the contributing authors. All material on this collaboration platform is the property of the contributing authors.  
or Ideas, requests, problems regarding TWiki? use [Discourse](#) or [Send feedback](#)