The sector correction (also sometimes referred to as the "Aras cabling scheme")

NB! This article does not refer to the so called "repeater board reordering", which you don't have to worry about.

It was discovered in September 2006 that the channel reordering in the Tell1 did not seem work properly. The reordering on the TELL1 would scramble the four readout sectors of a module (each read out with their own kapton cable) in such a way that the reordering treated sector 0 as sector 3, sector 1 as sector 2 and vice-versa for sectors 3 and 2.

The origin of this problem was that the cables between the repeater board and the TELL1 boards were scrambled. In the configuration used at the time, the cables were connected like this:

Old ("wrong") cabling

Repeater board out	Tell1 in
J17	PP0
J18	PP1
J19	PP2
J20	PP3

and the NZS bank decoding (VeloTELL1Decoding/DecodeVeloFullRawBuffer.cpp) was assuming this configuration.
IF YOU WANT TO USE THE TELL1 REORDERING, DON'T USE THIS CABLING SCHEME!

The right way to do it

The solution to the problem is to change the order of the cables into this

Correct ("Aras") cabling

Repeater board out	Tell1 in
J17	PP3
J18	PP2
J19	PP1
J20	PP0

When the cables are connected like this, the reordering in the TELL1 will work. A new problem then appears, namely that the NZS bank is decoded with the sectors scrambled. To cure this we implemented a sector unscrambling in the NZS bank decoding, which takes all NZS channels from sector 0 and puts them in sector 3 etc etc. Note that within the individual sectors the order of the channels is correct.
This unscrambling step is the default behaviour of the decoding but can be overridden with the job option `DecodeVeloFullRawBuffer.SectorCorrection = false` (default is true).

What are the consequences for assembly, testbeam and operation?

This gives us 2 possible ways of reading out the sensors and reorder properly:
1. **THE TESTBEAM AND OPERATION SCENARIO**
Use the correct ("Aras") cabling configuration and make sure that `DecodeVeloFullRawBuffer.SectorCorrection = true` (this is the default behaviour). In this scenario the reordering on the TELL1 will work properly (given that you have configured the TELL1 properly with respect to the sensor type).

2. **THE ASSEMBLY LAB SETUP**
Use the old cabling configuration and set `DecodeVeloFullRawBuffer.SectorCorrection = false`. This can

only be used if you don't plan to use the ZS bank of the TELL1 but do the reordering offline. This is the case in the assembly lab where the channel reordering is done using text files located in the Vetra/macros/. An alternative way of doing this would be to rely on the reordering methods of the DeVeloSensor class, but this requires the Tell1 --> Sensor mapping to be correct in the conditions database. One cas also use the reodering in the emulation, with the same result.

If you are only going to to study NZS data these two scenarios are completely identical meaning that the a snapshot plot produced in the assembly lab will agree with a snapshot plot produced in the commissioned VELO. The only thing you have to remember is to make sure that `DecodeVeloFullRawBuffer.SectorCorrection` has the correct setting.

-- Main.arasp - 25 Apr 2007

This topic: LHCb > SectorCorrection
Topic revision: r2 - 2007-04-26 - ArasPapadelis