

## Bit Counting in the TDB User Interface

J. Linnemann Feb 18, 2006

Revised July 6, 2006

I have been asked to specify L2 and L3 bit counting updates to the report interface. Right now we have `tl_index` counting (input triplet line counting), and L1 bit counting. The index counting was a good approximation to L3 bit counting before L2 branching became possible.

Now, the reporting interface should mimic the `xml/xmlgen` in understanding how the bits are assigned. For Trigger Lists of version 14 or earlier (no L2 branching, no Oring), the existing numbering is adequate.

For Trigger Lists of V15 or higher come in 2 forms, based on Or Markers, or based on L2Groups. This proposal would leave the Index number to stand in approximately for the L3 bit count. For Or Marker V15 triggers the index number will change faster than the L3 bit count, since the index increments for lines which generate no L3 bit assignments—those lines with L2Or markers). However, the index matches L3 bit count (as accurately as it does for V14 and lower lists) for V15 triggers written in terms of L2Groups without Or Markers.

The proposal is to replace the present pair of numbers in the first column

Index (L1bit)

with 3 numbers:

Index (L1bit, last\_L2bit)

Where

**Index** remains one per line, no matter what

**L1bit** count remains as now, and

increments whenever a new bit is intended to be assigned (i.e. knows about L2 branching)

**last\_L2bit** count:

increments whenever a non-null L2 script exists: i.e. whenever a new L2 bit is assigned

If a line consumes more than one L2 bit (for nontrivial L2 Groups), the count increments by the number used; if none is defined (for example for a L2\_OR\_MARKER), the L2 bit count does not increment.

Somewhere on the Report page there should be a note indicating the meaning of the 2 numbers in the parentheses.

For an **Or maker example** (ignoring the TriggerName)

26 (7,10)	...	...	...
27 (8, 11)	L1	L2a	*Or
28 (8, 12)	L1	L2b	*Or
29 (8, 12)	L1	*Or	L3

Notice that the Index increments 3 times although only one L3 bit is consumed

For the **equivalent L2Group example**, where L2Ga contains scripts {L2a, L2b}

26 (7,10)	...	...	...
27 (8, 12)	L1	L2Ga	L3

Here it is clear that L2Ga consumes two L2 bits; the Index increments only once, and (more obviously than in the previous example) only one L3 bit is consumed.