The CalSlt Cluster Algorithm.

S. de Jong, A. Kruse, H. vd Lugt, L. Wiggers
NIKHEF-H
P.O. Box 41882, NL-1009 DB Amsterdam

August 2001

1 SLSCAL - The CAL SLT Simulation

1.1 Overview

The simulation of the ZEUS Calorimeter Second Level Trigger is part of the ZGANA package. It is operating on raw data as well as on MC data. The simulation is " perfect " , i.e. all differences from the ONLINE implementation are due to rounding errors. The transputer network structure influences the results of the clustering algorithm. For this reason the structure of the code reflects the network structure.

SLSCAL runs on raw data banks CxTENE and -if existing- CxPECO and CxDCCN or the Caltru bank that is produced during the PHASE1 reconstruction.

1.2 Release Notes

1.2.1 Release 2.01/13

This version is made for running in 1996 and is still up to date. Shifted vertex runs are handled by changing datacards.

1.2.2 Release 2.01/10

This version from April 1995 provides BP-Et sums for HAC and EMC for the second ring of the FCAL-BP region and clustering algorithm has changed.

1.2.3 Release 2.01/09

Version made by L. Wiggers: no looping for combining of clusters too far away. Add neighbouring cells to cluster.

1.2.4 Release 2.01/05

This version contains a few bug fixes and small improvements to the previous version.

- Use of `STRUCTURE` and `POINTER`. This version makes use of new `FORTRAN` language elements to clarify parts of the code.
- Swap of array indices. In order to speed up the simulation array indices have been swapped for some critical arrays.
- New steering card `ZGANA-ZGSLCRAD` to toggle wether the simulation checks that the data of all digital cards is present. This card is by default set to `.FALSE.` so no checks are performed.
- The routine `CSCOCCB` which generates the OCCAM code for the FCAL beampipe energy calculations creates now more efficient code.

1.2.5 Release 2.01/04

This version contains a full simulation of the 1994 CAL SLT. Many new steering cards were added to allow a detailed study of the performance of the CAL SLT.

- Default setup. The default setup has been set such that the expected 1994 CAL SLT configuration is simulated. It is possible to go to the 1993 configuration by using the steering card ZGANA-ZGSLCS93 described below.
- New steering card ZGANA-ZGSLCTIA to set the time algorithm that is used to calculate the average FCAL, BCAL and RCAL time. New algorithms introduced allow
  - a straight average over all PM times when the energy of both PMs exceeds a threshold.
  - a straight average over all PM times when either the energy of both PMs exceeds a threshold or one of the channels is bad and then it is sufficient that the other one exceeds the threshold.
  - a simple energy weighting algorithm, the energy weight equals the energy up to a maximum value.
  - several time averages with different thresholds. The final average time is calculated by a simple weighting of those time averages.
- New steering cards ZGANA-ZGSLCFGR, ZGANA-ZGSLCBGR, ZGANA-ZGSLCRGR which allow a selection of different granularities for F,B and RCAL.
- New steering card ZGANA-ZGSLCFBG which allows to modify the granularity for the FCAL beampipe region only.
- New steering card ZGANA-ZGSLCHIS which sets a histogramming level in order to select `NTUPLE` output. Since the `NTUPLE` is created in the `/PAWC/` directory one can set an offset for the histogram identifier using the ZGANA-ZGSLCHID.
- New steering cards ZGANA-ZGSLCGL1, ZGANA-ZGSLCGL2, ZGANA-ZGSLCGL3 and ZGANA-ZGSLCGL4 which allow the modification of global constants through datacards.
- Output which is generated by ZGANA-ZGSLCCMA is extended by a file "slscaltemplate.cards" which contains the steering cards ZGANA-ZGSLCGL[1-4] with the current contents of the global constants.
- In order to speed up the ONLINE code a "brute force" optimization of the `OCCAM` is implemented: The transport of the energy sums from the digital cards to the trigger transputers was using as an additional step the private memory of the readout transputers. This was unnecessary and has been removed, resulting in a further speedup of this part of the operation.
- The `OCCAM` optimization has been fine tuned. From the disassembled transputer code one can see that it takes 8 ticks per 32 bit word transfer. The 'optimized' sequence takes $19 + 2 * n$ ticks per n32 bit words transfer. So it is only efficient to optimize the code if more than 2 cells are involved.
- For the ONLINE calculation of the FCAL beampipe energy sums an `OCCAM` code is generated which calculates those quantities, using the correct cell ordering.

- New steering card ZGANA-ZGSLC2PI which enables combining clusters close to $\phi = 0$ with clusters close $\phi = 2\pi$.
- New steering card ZGANA-ZGSLCS93 to force a simulation of the 1993 CAL-SLT.
- New steering card ZGANA-ZGSLCMUO to switch on muon identification.
- New steering card ZGANA-ZGSLCCFB to toggle clustering in the FCal beampipe region.
- Extended Sum bank **CCGSUM**: This bank has been extended by values for the total energy in F,B,RCal Emc, the average time of the 'up' and 'down' regions (see below), the number of photomultipliers in those regions and the total energy seen. It also contains the global time calculated from all photomultipliers in the entire calorimeter as well as various energy sums from the FCal beampipe region.
- New steering card ZGANA-ZGSLCTBA to steer wether the average time should be calculated with or without taking online reconstruction errors into account.
- Many many bug fixes. Most of them have only little or no effect on the simulation results.

1.2.6 Release 2.01/03

This release contains the following modifications:

- Read compressed data banks CxPECO.

1.2.7 Release 2.01/02

This release contains the following modifications:

- New steering card ZGANA-ZGSLCDFL. Using this steering card SLSCAL can be forced to use the Caltru table instead of CxTENE, CxPECO and CxDCCN.
- New steering card ZGANA-ZGSLCLUN to set the logical unit number that is used to access files. If this steering card is missing or unit number 0 is used SLSCAL will use `GETLUN` to get a free logical unit number.
- RCAL moved back by 10 mm. The lookup tables are corrected for the moved RCAL position.

1.2.8 Release 2.00/02

With the current SLSCAL release 2.00/02 a completely new simulation is introduced. It simulates exactly the behaviour of the online code and can thus be used as a convenient and safe tool to study possible SLT cuts.

A ZEUS note that describes in detail the differences to the previous implementation is in preparation.

1.3 Interfacing to SLSCAL

There are basically only two FORTRAN calls necessary:

- *Initialisation:* This has to be be done once (in `ZUINIT`). Here all constants tables are read in and lookup tables are calculated.

  `call SLSCALI( ErrorFlag, First )`

  `Integer ErrorFlag`

  `Logical First`

  `First` has to be set to `.TRUE.` for the initialisation call. The user never has to use `SLSCALI` twice.

  If `ErrorFlag` is not equal to zero a problem has occured.

- *Simulation call:* This has to be done for each event (in `ZUANAL`). The data is read in from the ADAMO-tables and put into common blocks. Then the simulation is started and the results are either put into common blocks or online banks.

  `call SLSCAL( ErrorFlag )`

  `Integer ErrorFlag`

  If `ErrorFlag` is not equal to zero a problem has occured.

  There are two special user calls that are used by the GSLT simulation:

- *Was the Simulation executed?* In order to be able to determine wether the simulation was called for a particular event, the following call is used:

  `call SLSCALT( SimulationHasRunOnThisEvent )`

  `Logical SimulationHasRunOnThisEvent`

  The variable `SimulationHasRunOnThisEvent` is set to `.TRUE.` if the CAL-SLT simulation was executed on the current event. Note: This routine is not able to determine wether the ONLINE banks were originally produced by the transputer network or by the simulation in a previous ZGANA run. It only checks if the data in the common blocks originates from a simulation call.

- *Put ONLINE banks into common blocks*: The next routine is also used by the GSLT. Calling this routine puts the current ONLINE bank contents into common blocks. By this the data that was generated on the transputer network can be put into the common blocks.

  `call SLSCALO( ErrorFlag )`

  `Integer ErrorFlag`

  `ErrorFlag` is set to a non-zero value if the **CCGSUM** bank is missing.

1.4 The Log File Output of SLSCAL

SLSCAL will work quietly without any log file output, unless the debug level has been modified with the ZGANA-ZGSLCDEB steering card or with a call to CSSTDEB.

SLSCALI however will print out the setting of all steering cards, except for ZGANA-ZGSLCGL[1-3]. This information can be used to determine the version and setup of the simulation.

1.5 Steering Cards

1.5.1 Steering the Algorithms

In order to be able to study the effect of various modifications to the CAL-SLT output a bunch of steering cards are available. Some of the steering cards modify the lookup tables which are used during the simulation, others only modify the algorithm itself.

1.5.2 Steering Cards which modify the Lookup Tables

Note that a call from your EAZE job to a FORTRAN routine which modifies the settings of one of the following steering cards will result in the recalculation of all lookup tables. This may slow down the program considerably.

Card Name	Type	Default	Contents
`ZGANA-ZGSLCVTX`	1 `Integer`	0	Vertex offset in cm. This will be used inside the calculation of the lookup tables. The distance of the layer2 regions in F/RCAL from the interaction point are corrected accordingly.
`ZGANA-ZGSLCR12`	`Logical`	`.TRUE.`	Lowered RCAL12 Used in calculation of lookup tables. True from 1995 onwards.
`ZGANA-ZGSLCSOR`	1 `Integer`	3	scanning order of the layer1 regions 0 = use the cell order given by the cable files 1 = ascending cell numbers 2 = fall92 algorithm 3 = symetric algorithm
`ZGANA-ZGSLCGRA`	2 `Integers`	1 1	Granularity used inside the algorithms First word: Granularity in Emc 1 = finest granularity 2 = in FCAL/BCAL add EMC1+2 and EMC3+EMC4 Second word: Granularity in Hac 1 = finest granularity 2 = add HAC0/HAC1/HAC2
`ZGANA-ZGSLCFGR`	2 `Integers`	1 1	Granularity for FCAL EMC and HAC
`ZGANA-ZGSLCBGR`	2 `Integers`	1 1	Granularity for BCAL EMC and HAC
`ZGANA-ZGSLCRGR`	2 `Integers`	1 1	Granularity for RCAL EMC and HAC
`ZGANA-ZGSLCBGR`	2 `Integers`	1 1	Granularity for the FCAL beampipe regions.

Card Name	Type	Default	Contents
ZGANA-ZGSLCFH0	1 Integer	0	Distance from the beampipe, measured in units of tower width, from which on Hac0 cells are used as Emc cells. If the distance is greater or equal than this number the cell is Hac. 0-9 = all Hac0 cells are Hac cells 10 = the first ring of Hac0 cells is called 'EMC'. 11 = the next ring 12 = all Hac0 cells are called Emc.
ZGANA-ZGSLCRH0	1 Integer	0	see above ZGANA-ZGSLCFH0 this is the value for RCAL

1.5.3 Steering Cards which modify the Algorithm

The following steering cards modify the behaviour of the algorithms.

Card Name	Type	Default	Contents
ZGANA-ZGSLCTIA	1 Integer	0	Selection of the algorithm used to calculate the time average 0 = if the energy recorded by a PM exceeds the threshold set in ZGANA-ZGSLCTIC then the time of this PM will contribute to the time average. 1 = if both PMs of a cell exceed the threshold set in ZGANA-ZGSLCTIC then the time of both PMs contributes to the time average. 2 = like (1) but for bad channels in a cell it is sufficient that the energy recorded by the good PM exceeds the threshold. 3 = simple energy weighting algorithm, the energy recorded by both PMs have to be above threshold set in ZGANA-ZGSLCTIC. The average time is a weigthed mean of all times where the weight is equal to the energy up to the maximum value set in ZGANA-ZGSLCTWM and remains constant for all values exceeding the maximum. 4 = calculate weighted average with several thresholds and weights, number of thresholds set in ZGANA-ZGSLCTNx, weights and thresholds set in ZGANA-ZGSLCTLx. 5 = like (4) but require that both PMs in a cell record more energy than the minimum threshold energy. 6 = like (5) but require that both PMs in a cell record more than the threshold energy of the weight they are added to.

Card Name	Type	Default	Contents
ZGANA-ZGSLCTIC	3 Integers	500 200 500	cuts for calculating the FCAL, BCAL and RCAL time in MeV, for algorithm 0 through 3.
ZGANA-ZGSLCTWM	3 Integers	2000 2000 2000	maximum energy used for timing algorithm 3.
ZGANA-ZGSLCTNF	1 Integer	2	number of thresholds used for timing algorithm 4. the thresholds and weights are set in ZGANA-ZGSLCTLF.
ZGANA-ZGSLCTLF	3 Integer	1 200 1 2 2000 1	Threshold number, energy cut, weight used for timing algorithm 4, This card has to occur ZGANA-ZGSLCTNF
ZGANA-ZGSLCTNB	1 Integer	2	analog to ZGANA-ZGSLCTNF, value for BCAL
ZGANA-ZGSLCTLB	3 Integer	1 200 1 2 2000 1	analog to ZGANA-ZGSLCTLF, value for BCAL
ZGANA-ZGSLCTNR	1 Integer	2	analog to ZGANA-ZGSLCTNF, value for RCAL
ZGANA-ZGSLCTLR	3 Integer	1 200 1 2 2000 1	analog to ZGANA-ZGSLCTLF, value for RCAL
ZGANA-ZGSLC2PI	1 Logical	.FALSE.	If set to .TRUE. then SLSCAL will try to combine clusters in layer 3 which are close to $\phi = 0$ with clusters close $\phi = 2\pi$
ZGANA-ZGSLCMUO	1 Logical	.FALSE.	If set to .TRUE. then SLSCAL will try to identify muons looking at the energy deposited in the EMC and HAC as well as the number of cells.

1.5.4 Force 1992 Simulation

The following steering cards are used to mimic the 1992 behaviour of the CAL-SLT. Note that this is only true for the global energy sums and the F/RCal timing words. The vector energy sums, the BCal time and the are clusters are NOT simulated.

Card Name	Type	Default	Contents
`ZGANA-ZGSLCF92`	1 `Logical`	`.FALSE.`	Do the 'fall92' CxDC bank correction such that if PM 24 on a digital card is bad and one of the two PMs from the first cell is bad, this cells energy is set to zero. ⇒ hole
`ZGANA-ZGSLCS92`	1 `Logical`	`.FALSE.`	If set to `.TRUE.` all steering is done such that the 1992 code is simulated. This is only true for the global energy sums and timing information in CCGSUM. Note: Setting this parameter to `.TRUE.` overrides all other parameters.

1.5.5 Force 1993 Simulation

The following steering cards set several other steering cards so that SLSCAL simulates the 1993 CAL-SLT behaviour. All values, including clusters are simulated.

Card Name	Type	Default	Contents
`ZGANA-ZGSLCS93`	1 `Logical`	`.FALSE.`	If set to `.TRUE.` all steering cards are set at once so that the 1993 CAL-SLT is simulated.

1.5.6 Bad Channel Correction

In order to be able to study the effect of bad channels and DC reconstruction errors on the CAL-SLT results the following steering cards are used:

Card Name	Type	Default	Contents
ZGANA-ZGSLCDCF	1 `Logical`	`.TRUE.`	toggles wether the simulation will read in the CxDC bank to determine the bad channels and check for the trigger type.
ZGANA-ZGSLCBAD	1 `Logical`	`.TRUE.`	Do Badchannel-Correction?
ZGANA-ZGSLCBPM	10 `Integer`	0	Patchable bad channel list 1st `Integer`: Number of bad channels 2nd-10th `Integer`: PMNo of bad channel This steering card might occur up to 10 times.
ZGANA-ZGSLCTBA	1 `Logical`	`.TRUE.`	toggles wether channels which are marked as having a reconstruction error are used to calculate the time average.

1.5.7 Energy Scale

If the simulation has to run on old MC data or events taken with a different energy scale the energy scale for can be modified. Note that these are conversion factors to [MeV], not [GeV].

Card Name	Type	Default	Contents
ZGANA-ZGSLCESC	3 Real	1. 1. 1.	Energy scale for FCal BCal RCal i.e. conversion factor to [MeV]

Conversion factors for older data are as listed below for CxTENE to MeVs:

$$E(MeV) = factor * E(CxTENE)$$

Data type	FCAL	RCAL	BCAL
Now Real data	MTOM	MTOM	MTOM
Now MC data	MTOM	MTOM	MTOM
Old MC data	FRCONVER	FRCONVER	BCONVER
Summer MC data	FRCONVER	FRCONVER	BCONVER
Summer Real data	FCONVER2	BRCONVER2	BRCONVER2

where:

$$\begin{array}{lcl}
\text{FRCONVER} & = & 1000/(200*9.51) & = & 0.5258 \\
\text{BCONVER} & = & 1000/(200*10.9) & = & 0.4587 \\
\text{FCONVER2} & = & 1000/(200*7.30) & = & 0.6849 \\
\text{BRCONVER2} & = & 1000/(200*10.6) & = & 0.4717 \\
\text{MTOM} & = & 1000/1000.0 & = & 1.0
\end{array}$$

and

Old	=	before summer running period
Summer	=	until Sept 1. 1992 (Time in ZREVT)
Now	=	after Sept 1. 1992 (Time in ZREVT)

1.5.8 Source of the Constants

The CAL-SLT simulation is self-contained. All constants are hardwired into the code and read automatically at initialization. However it is possible to force reading of the constants from disk (UNIX only).

Card Name	Type	Default	Contents
ZGANA-ZGSLCSRC	1 Integer	0	Source of the constants 0 = from data statements 1 = from files 2 = from data statements for 1992 trigger. 3 = from data statements for 1993 trigger. 4 = from data statements for 1994 trigger. 5 = from data statements for 1995 trigger.
ZGANA-ZGSLCDIR	1 Char*64		Directory in which the constants files can be found. This is only used when ZGANA-ZGSLCSRC = 1.
ZGANA-ZGSLCGL1 ZGANA-ZGSLCGL2 ZGANA-ZGSLCGL3	9 Integers		Global Constants Part 1 - 3 the global constants can be modified. It is always necessary to have all three steering card entries per Region. More Information in Section **??**lobalconst

1.5.9 Processing Information

If no steering card is given the simulation operates quiet. That is, the simulation only runs on the data and puts the results into private common blocks.

Card Name	Type	Default	Contents
ZGANA-ZGSLCDEB	1 Integer	0	debug level 0 = no debug messages are printed 1 = print message whenever a routine is called 2 = print in additon information about all interemediate steps of the simulation. 3 = print extensive information about cell energies, times, bad channels etc.
ZGANA-ZGSLCPRR	1 Logical	.FALSE.	If set to .TRUE. all simulation-results are printed.
ZGANA-ZGSLCEXT	1 Logical	.FALSE.	Measure and print execution time for the simulation (UNIX only!)

1.5.10 NTUPLE Information

With the following steering cards it is possible to create NTUPLE data which can be used to analyse each processing step of the algorithm and thus enable tuning of global constants. The "histogramming level" defines which information should be put into the NTUPLE file. With the "histogram identifier offset" an offset will be added to every NTUPLE identifier to avoid clashes with identifiers which may be used by the user.

Card Name	Type	Default	Contents
ZGANA-ZGSLCHIS	1 Integer	0	Histogramming level 0 = no histogramming 1 = store the final results only 2 = store intermediate results of the layer3 simulation 3 = store intermediate results of the layer2 simulation 4 = store intermediate results of the layer1 simulation
ZGANA-ZGSLCHID	1 Integer	0	Hisotgram identifier offset This value will be added to the histogram identifier used to store the NTUPLE information.

The following NTUPLEs are produced during the simulation:

- *NTGSUM*: This NTUPLE contains all the simulation results: Global sums and clusters.

Varable Name	Type	Contents
RunNo	I*4	Run number
EventNo	I*4	Event number
EvCount	I*4	Event counter, set to zero for each new event, is increased for consecutive calls to SLSCAL.
ETotEmc	R*4	total energy in EMC section [GeV]
ETotHac	R*4	total energy in HAC section [GeV]
CellsEmc	R*4	number of cells above threshold in EMC [GeV]
CellsHac	R*4	number of cells above threshold in HAC [GeV]
ETEmc	R*4	total transverse energy sum in EMC [GeV]
PxEmc	R*4	total vector energy sum in X-direction in EMC [GeV]
PyEmc	R*4	total vector energy sum in Y-direction in EMC [GeV]
PzEmc	R*4	total vector energy sum in Z-direction in EMC [GeV]
FBP2EtEmc	R*4	total transverse energy minus that in the two inner rings of the FCAL [GeV]

Varable Name	Type	Contents
ETHac	R*4	total transverse energy sum in HAC [GeV]
PxHac	R*4	total vector energy sum in X-direction in HAC [GeV]
PyHac	R*4	total vector energy sum in Y-direction in HAC [GeV]
PzHac	R*4	total vector energy sum in Z-direction in HAC [GeV]
FBP2EtHac	R*4	total HAC transverse energy minus that in the two inner rings of the FCAL [GeV]
FCalTime	R*4	average FCal time [ns]
BCalTime	R*4	average BCal time [ns]
RCalTime	R*4	average RCal time [ns]
TimeFlag	I*4	not used
NoOfClus	I*4	packed information about the number of clusters found, byte 0: ELEC, byte 1: EMCL, byte 2: HADR byte 3: CAMU
SltError	I*4	bitmask showing wether an error occured during the simulation
ETotFCal	R*4	total energy in FCal [GeV]
ETotBCal	R*4	total energy in BCal [GeV]
ETotRCal	R*4	total energy in RCal [GeV]
NoPMsFTi	I*4	number of photomultipliers used for the FCal time
NoPMsBTi	I*4	number of photomultipliers used for the BCal time
NoPMsRTi	I*4	number of photomultipliers used for the RCal time
ETotFEmc	R*4	total energy in FCal EMC [GeV]
ETotBEmc	R*4	total energy in BCal EMC [GeV]
ETotREmc	R*4	total energy in RCal EMC [GeV]
UpTime	R*4	average time of the "upper" regions [ns]
DownTime	R*4	average time of the "lower" regions [ns]
UpTiNPMs	I*4	number of photomultipliers used for "UpTime"
DnTiNPMs	I*4	number of photomultipliers used for "DownTime"
UpETot	R*4	total energy in the "upper" regions [GeV]
DnETot	R*4	total energy in the "lower" regions [GeV]
FBPEEmc	R*4	energy in FCAL beampipe region EMC [GeV]
FBPEHac	R*4	energy in FCAL beampipe region HAC [GeV]
FBPPxEmc	R*4	Px energy in FCAL beampipe region EMC [GeV]
FBPPyEmc	R*4	Py energy in FCAL beampipe region EMC [GeV]
FBPPxHac	R*4	Px energy in FCAL beampipe region HAC [GeV]
FBPPyHac	R*4	Py energy in FCAL beampipe region HAC [GeV]
FBPEtEmc	R*4	Et energy in FCAL beampipe region EMC [GeV]
FBPEtHac	R*4	Et energy in FCAL beampipe region HAC [GeV]
Reserv	R*4	not used

Varable Name	Type	Contents
ELECNCl	`I*4`	number of **ELEC** clusters
ELECEE	`R*4`	array: energy of cluster in EMC section [GeV]
ELECETh	`R*4`	array: position θ of the cluster in the EMC section
ELECEPh	`R*4`	array: position ϕ of the cluster in the EMC section
ELECENC	`I*4`	array: number of cells of the cluster in the EMC section
ELECER	`R*4`	array: radius of the cluster in the EMC section
ELECHE	`R*4`	array: energy of cluster in HAC section [GeV]
ELECHTh	`R*4`	array: position θ of the cluster in the HAC section
ELECHPh	`R*4`	array: position ϕ of the cluster in the HAC section
ELECHNC	`I*4`	array: number of cells of the cluster in the HAC section
ELECHR	`R*4`	array: radius of the cluster in the HAC section
EMCLxxx		**EMCL** cluster data, analog to **ELEC**
HADRxxx		**HADR** cluster data, analog to **ELEC**
CAMUxxx		**CAMU** cluster data, analog to **ELEC**

- *NTL1*: Clusters in layer 1: For every processing step on layer 1 this NTUPLE contains all clusters which were present at that moment. The (X, Y) position of the cluster is equal to (x, y) in ZEUS coordinates for FCAL and RCAL only. For BCAL $(X, Y) = (r * \phi, z)$.

Varable Name	Type	Contents
RunNo	I*4	Run number
EventNo	I*4	Event number
EvCount	I*4	Event counter, set to zero for each new event, is increased for consecutive calls to SLSCAL.
Region	I*4	Region number.
Step	I*4	Processing step number: 1 = Clusters after scanning the HAC cells 2 = Clusters after recombining HAC preclusters 3 = Clusters after energy cut of HAC preclusters 4 = Clusters after scanning the EMC cells 5 = Clusters after recombining EMC preclusters 6 = Clusters after energy cut of EMC preclusters 7 = Clusters after merging HAC and EMC preclusters 8 = Clusters after energy cut on combined clusters
NoClus	I*4	number of clusters
ClusEE	I*4	cluster energy in EMC section [MeV]
ClusEX	I*4	cluster position X in EMC section [mm]
ClusEY	I*4	cluster position Y in EMC section [mm]
ClusENC	I*4	number of cells in EMC section
ClusER	I*4	cluster radius in EMC section [mm] (obsolete)
ClusType	I*4	cluster type: 0 = hadronic -1 = electromagnetic -2 = muon candidate
ClusHE	I*4	cluster energy in HAC section [MeV]
ClusHX	I*4	cluster position X in HAC section [mm]
ClusHY	I*4	cluster position Y in HAC section [mm]
ClusHNC	I*4	number of cells in HAC section
ClusHR	I*4	cluster radius in HAC section [mm] (obsolete)
ClusE	I*4	total cluster energy [MeV]
ClusX	I*4	energy weighted cluster position X [mm]
ClusY	I*4	energy weighted cluster position Y [mm]

- *NTL2*: Clusters in layer 2: For every processing step on layer 2 this NTUPLE contains all clusters which were present at that moment. For processing steps < 4 the (X, Y) position of the cluster is equal to (x, y) in ZEUS coordinates for FCAL and RCAL and $(X, Y) = (r * \phi, z)$ for BCAL. After processing step 4 the coordinates are the global coordinates used in layer 3: $(X, Y) = (r * \phi * sin(\theta), r * \theta)$.

Varable Name	Type	Contents
RunNo	`I*4`	Run number
EventNo	`I*4`	Event number
EvCount	`I*4`	Event counter, set to zero for each new event, is increased for consecutive calls to SLSCAL.
Region	`I*4`	Region number.
Step	`I*4`	Processing step number: 1 = Clusters received from layer1 (no processing) 2 = Clusters after merging 3 = Clusters after energy cut 4 = Clusters after coordinate transformation (see above)
NoClus	`I*4`	number of clusters
ClusEE	`I*4`	cluster energy in EMC section [MeV]
ClusEX	`I*4`	cluster position X in EMC section [mm]
ClusEY	`I*4`	cluster position Y in EMC section [mm]
ClusENC	`I*4`	number of cells in EMC section
ClusER	`I*4`	cluster radius in EMC section [mm] (obsolete)
ClusType	`I*4`	cluster type: 0 = hadronic -1 = electromagnetic -2 = muon candidate
ClusHE	`I*4`	cluster energy in HAC section [MeV]
ClusHX	`I*4`	cluster position X in HAC section [mm]
ClusHY	`I*4`	cluster position Y in HAC section [mm]
ClusHNC	`I*4`	number of cells in HAC section
ClusHR	`I*4`	cluster radius in HAC section [mm] (obsolete)
ClusE	`I*4`	total cluster energy [MeV]
ClusX	`I*4`	energy weighted cluster position X [mm]
ClusY	`I*4`	energy weighted cluster position Y [mm]

- *NTL3*: Clusters in layer 3: For every processing step on layer 2 this NTUPLE contains all clusters which were present at that moment. For processing steps < 4 the cluster coordinates are the global coordinates: $(X, Y) = (r{*}\phi{*}sin(\theta), r{*}\theta)$. After processing step 4 the coordinates are the ZEUS spherical coordinates: $(X, Y) = (\theta * 1000.0, \phi * 1000.0)$.

Varable Name	Type	Contents
RunNo	I*4	Run number
EventNo	I*4	Event number
EvCount	I*4	Event counter, set to zero for each new event, is increased for consecutive calls to SLSCAL.
Step	I*4	Processing step number: 1 = Clusters received from layer2 (no processing) 2 = Clusters after merging 3 = Clusters after energy cut 4 = Clusters after coordinate transformation (see above) into ZEUS coordinates
NoClus	I*4	number of clusters
ClusEE	I*4	cluster energy in EMC section [MeV]
ClusEX	I*4	cluster position X in EMC section [mm]
ClusEY	I*4	cluster position Y in EMC section [mm]
ClusENC	I*4	number of cells in EMC section
ClusER	I*4	cluster radius in EMC section [mm] (obsolete)
ClusType	I*4	cluster type: 0 = hadronic -1 = electromagnetic -2 = muon candidate
ClusHE	I*4	cluster energy in HAC section [MeV]
ClusHX	I*4	cluster position X in HAC section [mm]
ClusHY	I*4	cluster position Y in HAC section [mm]
ClusHNC	I*4	number of cells in HAC section
ClusHR	I*4	cluster radius in HAC section [mm] (obsolete)
ClusE	I*4	total cluster energy [MeV]
ClusX	I*4	energy weighted cluster position X [mm]
ClusY	I*4	energy weighted cluster position Y [mm]

1.5.11 Force ONLINE banks

Card Name	Type	Default	Contents
ZGANA-ZGSLCONB	1 Logical	.FALSE.	forces the simulation to overwrite already existing online banks CCGSUM, CCELEC, CCEMCL, CCHADR, CCCAMU

1.5.12 Generate Debug Files

In order to be able to check the lookup tables they can be written to disk. It is also possible to generate transputer code in OCCAM that is used to read the data from the digital card and put it into the layer1 input buffers.

Card Name	Type	Default	Contents
ZGANA-ZGSLCCMA	1 Logical	.FALSE.	Tells the simulation to put all the constants and lookup tables into *.map files.
ZGANA-ZGSLCOC	1 Logical	.FALSE.	Tells the simulation to generate the OCCAM code that reorders+sums the DSP output into the format used by the CAL-SLT.
ZGANA-ZGSLCOO	1 Logical	.FALSE.	Tells the simulation to optimize the OCCAM code for speed.
ZGANA-ZGSLCOR	1 Logical	.FALSE.	Reduces the size of the OCCAM source code by finding and eliminating regions which look identical.

1.5.13 Running on reconstructed data

SLSCAL is able to run on the Caltru bank which is produced during the PHASE1 reconstruction. The results are influenced by this because during in Caltru the energy calibration using the DU current is used. Also if the granularity is not maximal the results will differ.

Card Name	Type	Default	Contents
ZGANA-ZGSLCDFL	1 Integer	0	0 = the online banks CxTENE, CxPECO and CxDCCN are used 1 = the Caltru bank is used.

1.5.14 Miscellaneous Cards

The following steering cards were meant for debugging purposes:

Card Name	Type	Default	Contents
ZGANA-ZGSLCTTR	1 Logical	.FALSE.	tells the simulation to run on testtriggers too. By default SLSCAL checks the trigger type and does not run on testtriggers. ATTENTION: The simulation will crash if it is forced to run on a standard QINJ testtrigger!
ZGANA-ZGSLCRAC	1 Logical	.FALSE.	Require All Cells to be there? (This has to be set to .FALSE. when MC data is used, because there it is likely that not all cells are there.) If set to true then a error message is printed for each missing cell.
ZGANA-ZGSLCRAD	1 Logical	.FALSE.	Require All Digital Cards data to be there ?
ZGANA-ZGSLCLUN	1 Integer	0	Logial unit number used when reading or writing files If set to 0 (default) the ADAMO subroutine GETLUN is used to get an unused logical unit number.

1.6 Region Numbers

Every CAL SLT region has an identifier number. This number is used when the global constants are set via datacards. To find out which region number corresponds to which region it is usefull to use the steering card ZGANA-ZGSLCCMA to print out the region mapping for the different calorimeters. The number in the files 'slscall1region.maps' and 'slscall2region.maps' has to be increased by 'offset' in the following list:

Region No. Offset	Range	Location
4111	4112 - 4127	FCAL layer1
4159	4160 - 4167	RCAL layer1
4127	4128 - 4143	BCAL layer1
4623	4624 - 4628	FCAL layer2
4639	4640 - 4644	BCAL layer2
4671	4672 - 4673	RCAL layer2

1.7 Region Tags

For every CAL SLT region a bit pattern can toggle the calculation of certain quantities. Here a list of the bits and their meaning:

Bit	Meaning
1	This region is an 'UP' region, it's time will be used to calculate the average 'UP' time.
2	This region is a 'DOWN' region.
3	This region belongs to the FCAL beampipe. This bit is set automatically when the constants are calculated. It is used ONLINE to identify a FCAL beampipe region.

1.8 Modifying the Global Constants with Datacards

It is possible to override the values of the global constants using the steering cards `ZGANA-ZGSLCGL1`, `ZGANA-ZGSLCGL2` and `ZGANA-ZGSLCGL3`. For every steering card 9 `Integer` values have to follow. The first value gives the region number, then the global constants follow. To get a template cards file with the current global constants, set the `ZGANA-ZGSLCCMA` steering card to `.TRUE.`. This will create a file which can be used for further EAZE runs. It also creates a file 'slscalglobalconstants.maps' which contains the current setting of all global constants; this can be used to check if all constants have been set correctly.

Global constants in datacard `ZGANA-ZGSLCGL1`:

Nr.	Unit	Description
1		Region number (see chapter 1.6)
2		Scale factor: Gives the conversion factor from [cm] into the scale used in the simulation and ONLINE. The default value is 10 so the simulation uses [mm] as the scale.
3	[MeV]	EmShowerCut: Minimum energy for an electromagnetic cluster, used when doing processing step 'localcut'
4	[MeV]	JetCut: Minimum energy for a hadronic cluster, used when doing processing step 'localcut'
5	[MeV]	EmCellCut: Minimum energy for an EMC cell to enter the algorithms. Used to reject noise.
6	[MeV]	HacCellCut: Minimum energy for a HAC cell to enter the algorithms. Used to reject noise.
7	[MeV]	EmPreclCut: Minimum energy for an electromagnetic precluster, used when doing step 'precluster cut'
8	[MeV]	HacPreclCut: Minimum energy for a hadronic cluster, used when doing step 'precluster cut'
9	[cm/MeV]	RJetFactor: used when calculating the radius of a hadronic cluster

Global Constants in steering card ZGANA-ZGSLCGL2:

Nr.	Unit	Description
1		Region number (see chapter 1.6)
2	[cm]	RJetOffset: used when calculating the radius of a hadronic cluster
3	[cm]	REmShower: radius of an electromagnetic cluster
4	[MeV]	EMuEmcLCut: minimum energy in EMC required for a muon cluster
5	[MeV]	EMuEmcHCut: maximum energy in EMC required for a muon cluster
6	[MeV]	EMuHacLCut: minimum energy in HAC required for a muon cluster
7	[MeV]	EMuHacHCut: maximum energy in HAC required for a muon cluster
8	[MeV]	RMuLCut: minimum radius of a muon cluster (unused)
9	[MeV]	RMuHCut: maximum radius of a muon cluster (unused)

Global Constants in steering card ZGANA-ZGSLCGL3:

Nr.	Unit	Description
1		Region number (see chapter 1.6)
2		NMuLowCut: minimum number of cells in a muon cluster
3		NMuHighCut: maximum number of cells in a muon cluster
4		RegionTags: bit pattern used to toggle the calculation of certain quantities, (see chapter 1.7
5	[cm]	FrontDistance: Gives the distance of the front of this region to the nominal interaction point. This value is recalculated when the lookup tables are generated.
6	[cm]	ROld: Only used on layer2, contains 'FrontDistance' from the connected layer1 regions. Used to convert from local coordinates into global coordinates. This value is recalculated when the lookup tables are generated.
7	[cm]	RSphere: Radius of the sphere onto which the clusters are projected. Used in layer2 and layer3'.
8	[%]	ElecRatio Emc/Hac: Gives the EMC/HAC ratio of the energy of an electromagnetic cluster.
9	[cm]	Gives the radius of a muon cluster, used when trying to combine clusters to muon clusters.

Global Constants in steering card ZGANA-ZGSLCGL4:

Nr.	Unit	Description
1		Region number (see chapter 1.6)
2	[MeV]	EmcQCut: EMC cell energy cut used to define a 'quiet' region.
3	[MeV]	HacQCut: HAC cell energy cut used to define a 'quiet' region.
4		NoCells/EmcC: number of physical cells in one EMC Cell, this value is set automatically according to the granularity
5		NoCells/HacC: number of physical cells in one HAC Cell.
6		spare
7		spare
8		spare
9		spare

1.9 Steering the Steering Cards

This feature allows the modification of the steering card values using FORTRAN calls. By this it is possible to run the simulation several times on the same event with different constants values.

If SLSCAL is called twice for the same event the data is NOT read in from the ADAMO-tables again. This improves the speed.

Also SLSCAL recalculates only the constants that are affected by a previous steering card call.

Suppose a study has to be made about the effect of various thresholds for the timing calculation in FCAL and RCAL:

```
do i = 1,5
   TimingCut = 500 + i * 100
   call CSSTTIC( TimingCut, 200, TimingCut )
   call SLSCAL( ErrorFlag )
   call FETTAB( CCGSUM, ID, 1 )
   FCalTime ( i ) = CCGSUM_FCalTime
   RCalTime ( i ) = CCGSUM_RCalTime
enddo
```

The user has to take care of returning back to the original setting of the steering variables. This is done in the above example by calling CSSTTIC before calling SLSCAL.

The naming convention is simple: All steering cards differ only in the last three characters. In order to change the value of steering card ZGANA-ZGSLCXXX the FORTRAN routine CSSTXXX is used.

1.10 The Default Values

The default steering card values have been chosen so that the expected 1994 trigger setup is selected.

1.11 Running SLSCAL on MOZART data

SLSCAL accepts both RAW and MOZART data. In both cases the ONLINE banks CCGSUM, CCELEC, CCEMCL, CCHADR and CCCAMU are by default not generated. In order to do that one has to set ZGANA-ZGSLCONB to .TRUE.. It may also be necessary to select the configuration for a particular year with the steering cards ZGANA-ZGSLCF92 or ZGANA-ZGSLCS93.

1.12 Additional Subroutines

The SLSCAL library contains some subroutines which make use of the CAL SLT data and other routines of this library but which are not called by `SLSCAL(IErr)`.

1.12.1 `CSMKDCD`: Make data for the "eventplayer"

This routine returns for a given calorimeter, crate and digital card the image data which is seen by the readout transputer. This can be used to generate input data for the "eventplayer".

```
Integer Calorimeter, Crate, DC
Integer Page, FLTNo
Integer Data(64)
call CSMKDCD( Calorimeter, Crate, DC, Page, FLTNo, Data )
```

1.12.2 `CSMKDAT`: Make `CAR` file for lookup tables

This routine reads constants from disk files in the current directory and creates `CAR` files which can be included in the ZGANA code.

1.12.3 `CSMKOCC`: Make `OCCAM` code

This routine reads constants from disk files in the current directory and makes the necessary calls to create `OCCAM` code to be used ONLINE.

1.12.4 `CSMKSLT`: Make downloadable constants file

This routine reads constants from disk files in the current directory and makes the necessary subroutine calls to create a `ZEBRA` file which holds the constants. This `ZEBRA` file will then be loaded at SETUP time to the different transputers in the CAL SLT network.