ALEPH 95-94
DATACQ 95-01
U. Becker et al.
28.7.1995

The Hitchhiker's Guide to the Level1 Universe

U. Becker, B. Rensch, J. Sommer, S. Werner
Institut für Hochenergiephysik, Universität Heidelberg, Germany

LEVEL1 ONLINE MANUAL

2nd edition
July 1995

Contents

Chapter 1

DAQ in a Nutshell

1.1 The Finite State Machine

The ALEPH DAQ with all its different subdetectors and subdetector tasks (≈ 350) is a highly complex system which needs much effort to be handled during data taking. To control the status of each task, one has adopted a "Finite State Machine" (FSM) protocol (see figure 1.1) controlled by the DAQ Run Controller. The different states (boxes in figure 1.1) of the FSM reflect at every instant the state of the whole DAQ

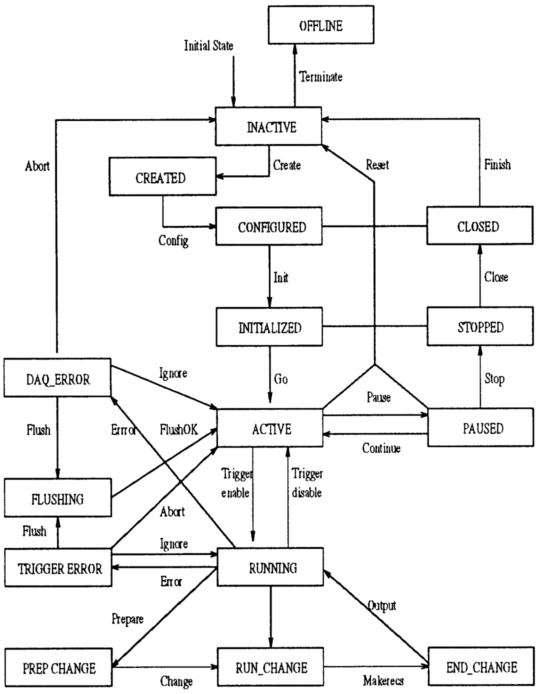


Figure 1.1: The Finite State Machine Diagram

system. Transitions between different states are only allowed along predefined paths (arrows in figure 1.1). The interface which translates the shift operator commands and which steers the readout system is called the DAQ Run Controller. The DAQ Run Controller creates the Subdetector Run Controllers on the VAX, and a control task on the readout processors (FICs). The Level1 Run Controller is called X1DAQ (X1SUPT in the mainstream), the Level1 control task on the FIC is called VDAQ_CONT. The control task finally creates, depending on the Run Contol Table, the producer, consumer and reformatter tasks required on the particular FIC. Commands given by the operator to the DAQ Run Controller instruct the program to bring all subsidiary tasks into a certain state. This is called a transition. When all subtasks have reached successfully the desired state, the DAQ Run Controller itself executes this transition and the FSM reaches the next state.

1.2 Starting a Run

Lets have a closer look on what is happening during the start of a run with special emphasis on the Level1 trigger: Starting a run is under the control of the DAQ Run Controller. It gives the command to all subdetector Run Controllers to change their state in the FSM diagram. In the first step, the global data base of the trigger, defining the parameters of the system, is prepared and downloaded into the X1 Global Section (see appendix C). From here the DAQ Run Controller selects the trigger type (e.g. CALI_RUN, TEST_RUN, RUN_DFLT ...) and Level3 receives the thresholds and calibration constants. How this data base is managed and how you can change the settings of the trigger is described in more detail in section 3. Then all data modules necessary on the FIC have to be prepared and sent down to the FIC. Now the system is set up, and some simple tests are performed and the start of run record (SOR), containing and archiving all information of the trigger setting of this specific run, is written. When everything was successful, the trigger has nothing left to do before going into the state ACTIVE.

1.3 The Readout

Each bunch crossing starts a trigger cycle unless the detector is busy. The trigger signals are distributed by the Main Trigger Supervisor (MTS) via the Fan In/Out (FIO) tree. Each readout processor in ALEPH has an associated trigger signal receiver (TSR) which sends SD specific signals to its electronics and handles the trigger protocol with the MTS. If the signals arriving from a particular SD at the Level1 hardware have forced a Level1 yes, the Level1 TSR will inform the MTS via a dedicated cable connection of its decision, and the MTS will start the Level2 cycle. In addition, the MTS receives the trigger mask on two cables from the TPR. If this cycle, too, results in a positive trigger decision, the readout will be started.

On the Level1 FIC the producer task X1PROD now starts to collect all relevant trigger

information from the hardware and writes it into the memory of the FVSBI. The next step belongs to the reformatter task X1FORM which reads from the event buffer, creates the BOS banks (see appendix E) and puts the event into the format expected by the offline software. Finally, the consumer task informs the next level of the readout tree that the event is ready. The administration of the memory is done by a Buffer Manager; it allocates the requested space, notifies the consumer task X1FORM if there is a new event, and frees the memory from events already seen by the consumer.
After all pieces of an event are collected in the Main Aleph Event Builder (MAEB), they are sent to the online VAX (AXADAQ). Here the LEVEL3 trigger decides whether the event will be accepted or not; accepted events are finally written to disk.

1.4 The Monitoring

A copy of the event is sent, via the so called spy channel, to AXADAQ where it is available for monitoring tasks. Some events are shown on the event display, an important monitoring tool. Level1 has provided three monitoring tasks: XT1MON, XT1VER and XT1ANL, which check the data coming from the Level1 trigger. While XT1VER checks whether the trigger logic works correctly, XT1MON provides the histograms for the PRESENTER where they can be investigated (for XT1MON and XT1VER see also below, PRESENTER and event display are desribed in detail in the SD Coordinator Manual and the Data Manager Manual). XT1ANL checks the histograms of the ITC, TPC, LCW, ECW and HCW segment bits from the registers (see below). The error messages and information provided by the trigger tasks can be derived by X1ERR, which is described further down in the text. XT1MON and XT1VER also deliver some statistics and run summaries at the end of each run. The control task X1WTCH on the FIC also performs a lot of monitoring and informs the Run Controller X1SUPT about all errors. Besides that, it delivers all kind of information to the DISPLAY task which allows you to follow up the trigger activities online. For details how to use DISPLAY see below.

1.5 The Run Quality

When the disks receiving the ALEPH data are full, the events are sent to FALCON which copies them onto tape and processes the data with JULIA. The processed data (POTs) are now used for the Run Quality checks. In this step, the Level1 trigger software produces some statistics and prepares the data files necessary for the determination of the thresholds in the trigger hardware. How to derive the threshold is described in the sections X1CALB and THRFIT. The Run Quality also produces a printout with statistics, reports of problems and other information. You can get it using X1RUNQ (See section 5.17). Note, that the downgrading of a run is **NOT** done by the Run Quality but has to be done by every SD coordinator by hand with the help of the CIA tool.

Chapter 2

AXAONL and FIC

2.1 AXAONL

AXAONL is the name of the Aleph Online Cluster which consists of two AXP servers (AXAONL and AXADAQ) and several stations. To log on, use preferably the AXALV1 and –if you don't have your own account– the userid TRIGGER; ask for the current password in time. Several trigger tasks running on the Online Cluster are described below. It is very useful, even mandatory, to have the following line in your AXAONL login.com:

```
$ @Disk$USER:[TRIGGER]X1symbols
```

The following commands are then defined in X1SYMBOLS:

Cmds	List Commands
X1mist	XLV1 Database Editor
X1runq	Print RunQ output for XLV1
X1err	Scan Logfiles X1supt, X1mon, X1ver
X1prb	XLV1 General Problem Loggerr
X1alert	Permanent Display of X1prb messages
X1calb	Calculate Effective Thresholds
X1xsum	Print a short RunSummary/Rates
ThrFit	Test Linearity of Thresholds
X1pres	Poor man's Presenter for XLV1 Histograms
X1chart	Poor man's Presenter for XLV1 TimeCharts
X1RGd	Display X1RG bank from ONLINE data
X1TDd	Display X1TD bank from ONLINE data

X1trgd	Print/Update Trigger-definitions
X1news	Post News for communication in XLV1-group
Aldisp	Permanent Display of ALEPH status
CIA	Report RunQuality problems
Hedt	Edit Histogram Database (X1mon, X1ver)
HsaveRef	Generate Reference Histos for X1mon, X1ver
Pres	Start the Presenter
ErrLog	Extract Messages sent to ErrorLogger
ErrHelp	List HelpInfo for ErrorLogger Messages
Coord	Scan/Update Coordinators
Schedb	Scheduler DataBase
Xprint	Print TEXT_File double_sided to LWACR$TEXT_132
PSprint	Print POST_File double_sided to LWACR$POST

2.2 FIC

Since 1994 all Fastbus readout processors are replaced by VME based 68040 processors, called FIC (Fast Intelligent! Controller). Each FIC is connected to a Fastbus crate via an interface, the FVSBI. The trigger FIC is named L1RP01 and is placed at the far right hand side in the barrack C2 in the HCAL VME maincrate (figure 2.1) . It is labelled 'L1RP01'. If you want to log on, you first have to connect to the Online cluster. If you are in the control room use the Falco terminal in the trigger corner. At the prompt *Local>* type *c axalv1*, then logon as *TRIGGER*.

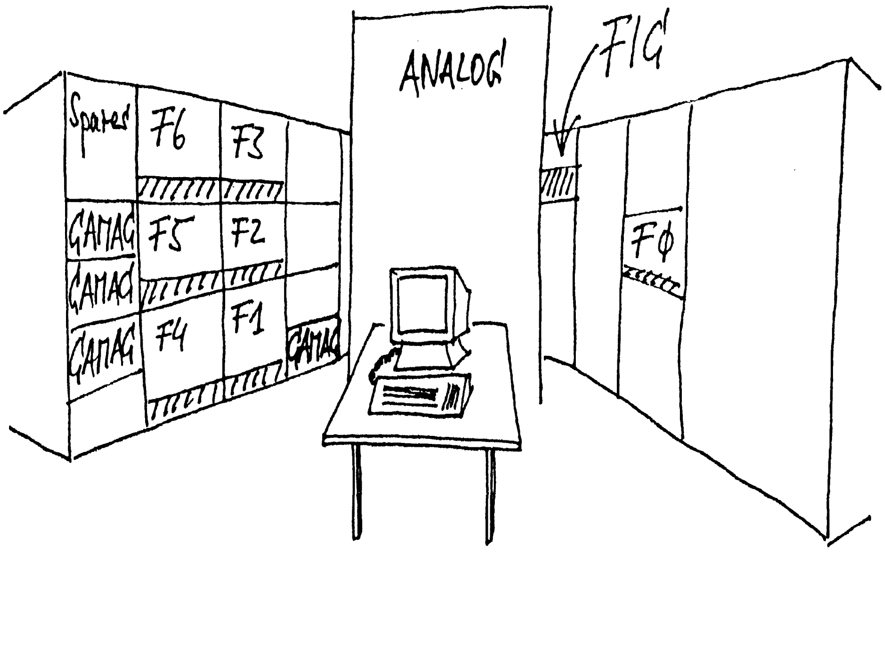


Figure 2.1: The C2 barrack

Now type *telnet l1rp01* and wait for the prompt *Enter username:* Write *level1 aleph* where *level1* is the username and *aleph* is the password. (Using the password *manager* instead, gives you write access to the area outside /os9/usr/level1. Be bloody careful!) This Falco terminal is normally used to monitor the trigger via the DISPLAY task. Type *display* to start it after logging on to the FIC. How to handle DISPLAY is described in section 7.1.

All important tasks running on the FIC are started or at least prepared at boot time when */os9/boot/l1rp01_start* is called automatically. *l1rp01_start* calls */os9/usr/level1/vme/scripts/level1_start* which loads all required Level1 tasks in the module directory and creates the Run Control Table from */os9/boot/data/l1rp_table.txt* . If changes in the boot procedure are necessary, they have to be made in *l1rp01_start.* An overview of the modules in the Fastbus crates is given in fig 2.2. Note that the F0 crates contains two CIs (Cluster Interconnect), one for the branch F1 to F3, the second for the branch F4 to F6.

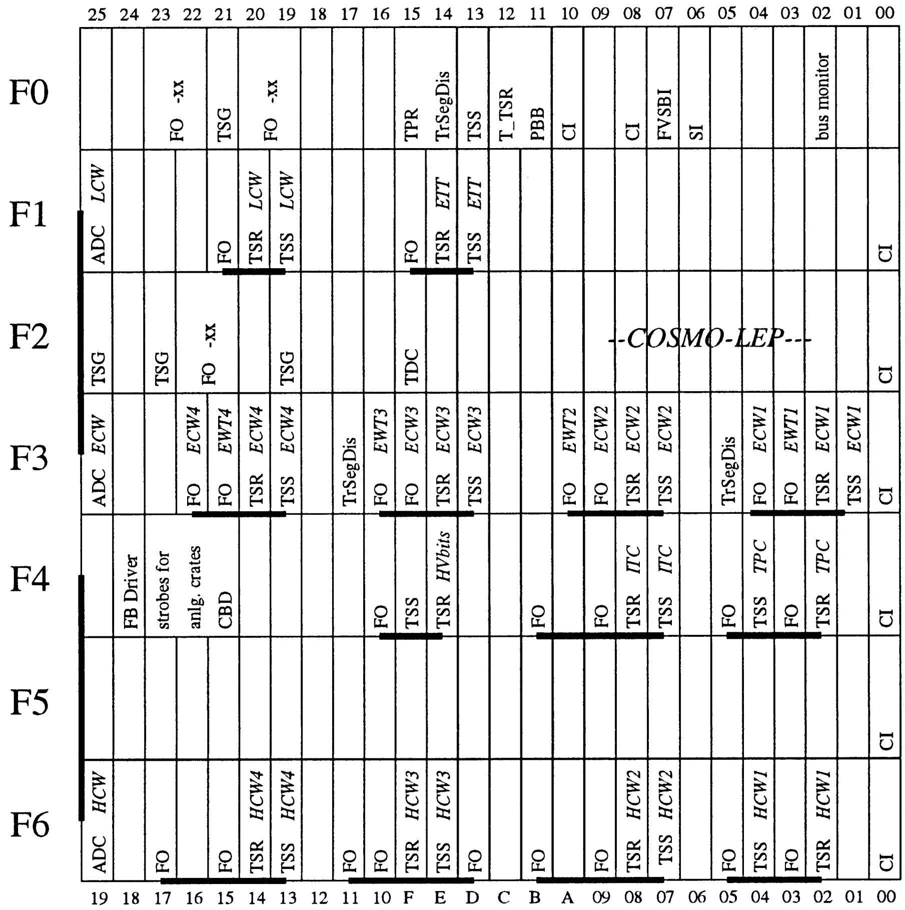


Figure 2.2: The modules in the Fastbus crates

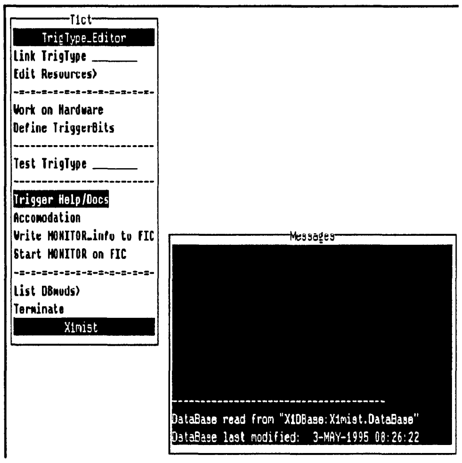
Chapter 3

X1MIST: The Level1 Data Base Editor

See [Trigger.Mist]x1mist.mms for the resources.

X1MIST is a powerful tool to manage the Level1 data base. This task allows you to change [1] the trigger settings and to test the hardware without much effort. See also the section 5.23 on the Fastbus data base. It is absolutely necessary for you as the Level1 coordinator to be familiar with this task.

To run X1MIST type *x1mist*. It may take a while until you can take any action in the UPI menu which should appear on your screen (figure 3.1).



```
DataBase read from "X1DBase:X1mist.DataBase"
DataBase last modified:  3-MAY-1995 08:26:22
```

Figure 3.1: X1MIST Main Menu

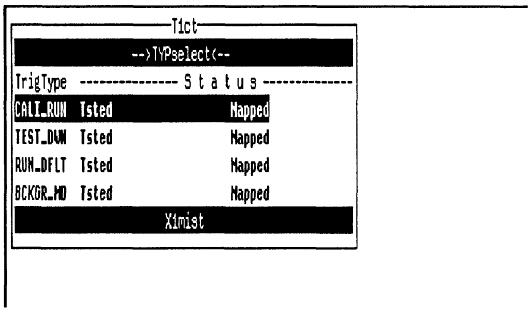
If a highlighted line is visible, you can start. Use the cursor keys to move up and down in the menu. Commands which can be executed are indicated in bold letters. Place

[1] X1MIST may crash when writing to the DB. This may be due to some unknown BOS timing (BOS checks the time left of a process). Logout, Logon, Retry.

the cursor on the appropriate line and press the return key. Some commands require parameters (indicated by a '__________'). If you don't know what to fill in, hit again the return key, and the program usually will show you a list of possible choices. To go up one level press 'F12'.

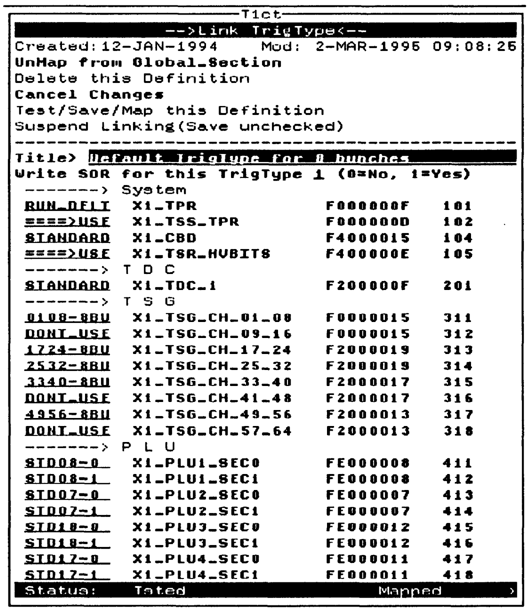
3.0.1 Link Trigtype

Here you can compile different trigger types out of the given hardware facilities (their use is defined under *Edit Resources*). You have to follow some conventions which are described in *trigger help/docs* under the keyword *Mist*. As a beginner you will use this facility only to inform yourself how the Trigger is set up. To get information, follow these instructions: go up to the line *link TrigType* and press enter (figure 3.2). Choose e.g. *RUN_DFLT* with the cursor keys. Press enter again. Now you see a long list of trigger hardware components which are linked to already defined options (figure 3.3). Choose the one on which you need information and press enter once more. Now all the necessary information about this component and how it is used in this special trigger type appears on the screen.



```
                  Tlct
              -->TYPselect<--
TrigType  -------------- S t a t u s --------------
CALI_RUN  Tsted                    Mapped
TEST_DWN  Tsted                    Mapped
RUN_DFLT  Tsted                    Mapped
BCKGR_MD  Tsted                    Mapped
                  X1mist
```

Figure 3.2: Trigger Type Menu



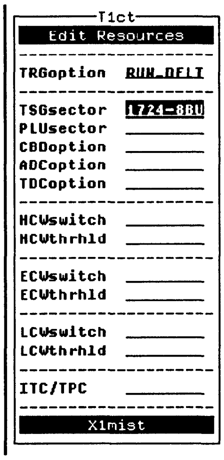
```
                         Tlct
                   -->Link TrigType<--
Created:12-JAN-1994      Mod:  2-MAR-1996 09:08:26
UnMap from Global_Section
Delete this Definition
Cancel Changes
Test/Save/Map this Definition
Suspend Linking(Save unchecked)
----------------------------------------------------
Title> Default TrigType for 8 bunches
Write SOR for this TrigType 1 (0=No, 1=Yes)
 -------> System
 RUN_DFLT  X1_TPR               F000000F   101
 ====>USE  X1_TSS_TPR           F000000D   102
 STANDARD  X1_CBD               F4000015   104
 ====>USE  X1_TSR_HVBITS        F400000E   105
 -------> T D C
 STANDARD  X1_TDC_1             F200000F   201
 -------> T S G
 0108-8BU  X1_TSG_CH_01_08      F0000015   311
 DONT_USE  X1_TSG_CH_09_16      F0000015   312
 1724-8BU  X1_TSG_CH_17_24      F2000019   313
 2532-8BU  X1_TSG_CH_25_32      F2000019   314
 3340-8BU  X1_TSG_CH_33_40      F2000017   315
 DONT_USE  X1_TSG_CH_41_48      F2000017   316
 4956-8BU  X1_TSG_CH_49_56      F2000013   317
 DONT_USE  X1_TSG_CH_57_64      F2000013   318
 -------> P L U
 STD08-0   X1_PLU1_SEC0         FE000008   411
 STD08-1   X1_PLU1_SEC1         FE000008   412
 STD07-0   X1_PLU2_SEC0         FE000007   413
 STD07-1   X1_PLU2_SEC1         FE000007   414
 STD18-0   X1_PLU3_SEC0         FE000012   415
 STD18-1   X1_PLU3_SEC1         FE000012   416
 STD17-0   X1_PLU4_SEC0         FE000011   417
 STD17-1   X1_PLU4_SEC1         FE000011   418
 Status:    Tsted                 Mapped          >
```

Figure 3.3: Hardware Components

To see all possible (defined) options, clear the line in front of the component and press enter. If you have, by accident, overwritten something, don't hesitate to use the *cancel changes* command. Note that *save/update this Definition* takes quite some time.

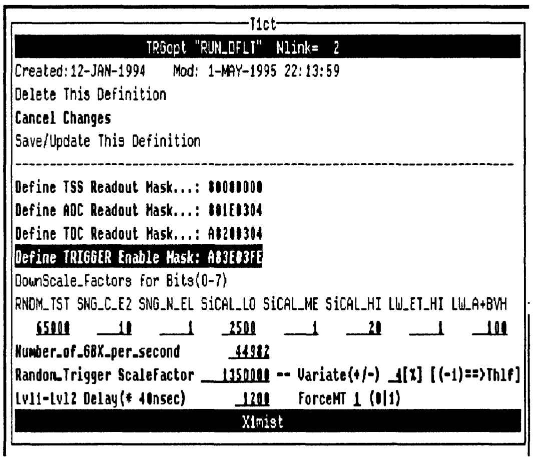
3.0.2 Edit Resources

Here you can define the components out of which a trigger type is to be composed (figure 3.4). You may also receive information about a trigger type from here (figure 3.5 and 3.6). The procedure is quite similiar to that described above. If you have to change the setting of the trigger (e.g. dis- or enable a trigger, change a downscaler, the timing, the PLU setting etc.), here is the right place to do it. For a detailed description how to perform the most common changes, see below. If you have, by accident, overwritten something, don't hesitate to use the *cancel changes* command.



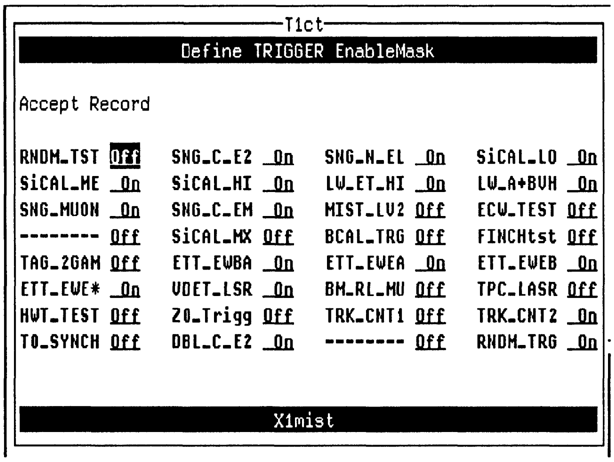
```
            T1ct
       Edit Resources
-----------------------
TRGoption   RUN_DFLT
-----------------------
TSGsector   1724-8BU
PLUsector   ________
CBDoption   ________
ADCoption   ________
TDCoption   ________
-----------------------
HCWswitch   ________
HCWthrhld   ________
-----------------------
ECWswitch   ________
ECWthrhld   ________
-----------------------
LCWswitch   ________
LCWthrhld   ________
-----------------------
ITC/TPC     ________
-----------------------
           X1mist
```

Figure 3.4: The Resources Menu



```
                                  T1ct
                   TRGopt "RUN_DFLT"  Nlink=  2
Created:12-JAN-1994   Mod: 1-MAY-1995 22:13:59
Delete This Definition
Cancel Changes
Save/Update This Definition
-----------------------------------------------------------------------
Define TSS Readout Mask...: 00000000
Define ADC Readout Mask...: 001E0304
Define TDC Readout Mask...: A0200304
Define TRIGGER Enable Mask: A03E03FE
DownScale_Factors for Bits(0-7)
RNDM_TST SNG_C_E2 SNG_N_EL SiCAL_LO SiCAL_ME SiCAL_HI LW_ET_HI LW_A+BVH
  65000      10        1     2500        1       20        1      100
Number_of_GBX_per_second        44982
Random_Trigger ScaleFactor    1350000 -- Variate(+/-)  4[%] [(-1)==>Thlf]
Lvl1-Lvl2 Delay(* 40nsec)        1200    ForceMT 1 (0|1)
                                 X1mist
```

Figure 3.5: The Trigger Options



```
                         T1ct
               Define TRIGGER EnableMask

Accept Record

RNDM_TST Off  SNG_C_E2 On   SNG_N_EL On   SiCAL_LO On
SiCAL_ME On   SiCAL_HI On   LW_ET_HI On   LW_A+BVH On
SNG_MUON On   SNG_C_EM On   MIST_LV2 Off  ECW_TEST Off
-------- Off  SiCAL_MX Off  BCAL_TRG Off  FINCHtst Off
TAG_2GAM Off  ETT_EWBA On   ETT_EWEA On   ETT_EWEB On
ETT_EWE* On   VDET_LSR On   BM_RL_MU Off  TPC_LASR Off
HWT_TEST Off  Z0_Trigg Off  TRK_CNT1 Off  TRK_CNT2 On
T0_SYNCH Off  DBL_C_E2 On   -------- Off  RNDM_TRG On

                         X1mist
```

Figure 3.6: The Enable Mask

3.0.3 Work on Hardware

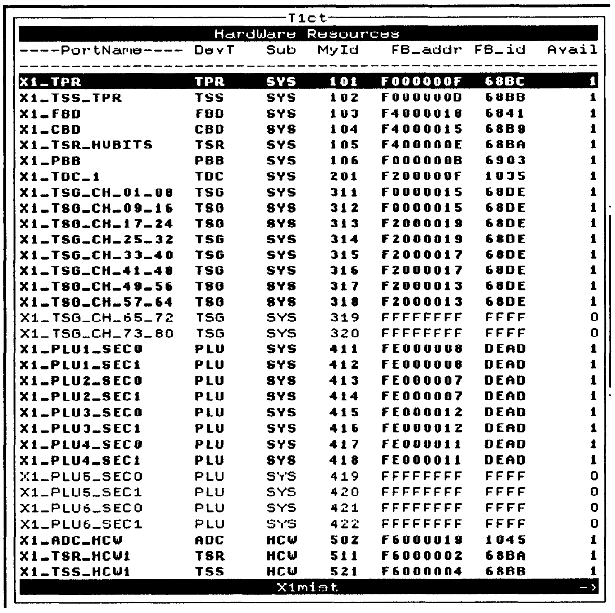
Here one has the possibility to check the trigger hardware (figure 3.7).

You shouldn't do this during data taking!

(In fact you will get a warning if the DAQ is running.) To perform tests on a particular module, place the cursor on the corresponding line and hit enter. If the module has been identified, a menu will be displayed, giving you access to all the internal registers and functions. Another, though less comfortable, way to play around with the hardware is at your disposal by running the task RTEST on the FIC.

Note:

Since X1MIST receives the information about the hardware from the Fastbus data base, all modules have to be declared there (see section 5.23).

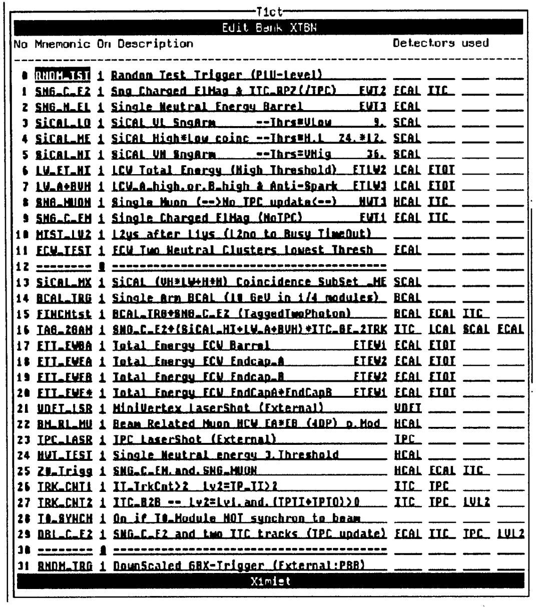


```
                          Tict
                   HardWare Resources
----PortName---- DevT   Sub  MyId   FB_addr  FB_id   Avail
-----------------------------------------------------------
X1_TPR            TPR    SYS   101  F000000F  68BC       1
X1_TSS_TPR        TSS    SYS   102  F000000D  68BB       1
X1_FBD            FBD    SYS   103  F4000018  6841       1
X1_CBD            CBD    SYS   104  F4000015  68B9       1
X1_TSR_HUBITS     TSR    SYS   105  F400000E  68BA       1
X1_PBB            PBB    SYS   106  F000000B  6903       1
X1_TDC_1          TDC    SYS   201  F200000F  1035       1
X1_TSG_CH_01_08   TSG    SYS   311  F0000015  68DE       1
X1_TSG_CH_09_16   TSG    SYS   312  F0000015  68DE       1
X1_TSG_CH_17_24   TSG    SYS   313  F2000019  68DE       1
X1_TSG_CH_25_32   TSG    SYS   314  F2000019  68DE       1
X1_TSG_CH_33_40   TSG    SYS   315  F2000017  68DE       1
X1_TSG_CH_41_48   TSG    SYS   316  F2000017  68DE       1
X1_TSG_CH_49_56   TSG    SYS   317  F2000013  68DE       1
X1_TSG_CH_57_64   TSG    SYS   318  F2000013  68DE       1
X1_TSG_CH_65_72   TSG    SYS   319  FFFFFFFF  FFFF       0
X1_TSG_CH_73_80   TSG    SYS   320  FFFFFFFF  FFFF       0
X1_PLU1_SEC0      PLU    SYS   411  FE000008  DEAD       1
X1_PLU1_SEC1      PLU    SYS   412  FE000008  DEAD       1
X1_PLU2_SEC0      PLU    SYS   413  FE000007  DEAD       1
X1_PLU2_SEC1      PLU    SYS   414  FE000007  DEAD       1
X1_PLU3_SEC0      PLU    SYS   415  FE000012  DEAD       1
X1_PLU3_SEC1      PLU    SYS   416  FE000012  DEAD       1
X1_PLU4_SEC0      PLU    SYS   417  FE000011  DEAD       1
X1_PLU4_SEC1      PLU    SYS   418  FE000011  DEAD       1
X1_PLU5_SEC0      PLU    SYS   419  FFFFFFFF  FFFF       0
X1_PLU5_SEC1      PLU    SYS   420  FFFFFFFF  FFFF       0
X1_PLU6_SEC0      PLU    SYS   421  FFFFFFFF  FFFF       0
X1_PLU6_SEC1      PLU    SYS   422  FFFFFFFF  FFFF       0
X1_ADC_HCW        ADC    HCW   502  F6000018  1045       1
X1_TSR_HCW1       TSR    HCW   511  F6000002  68BA       1
X1_TSS_HCW1       TSS    HCW   521  F6000004  68BB       1
                         X1mist                        ->
```

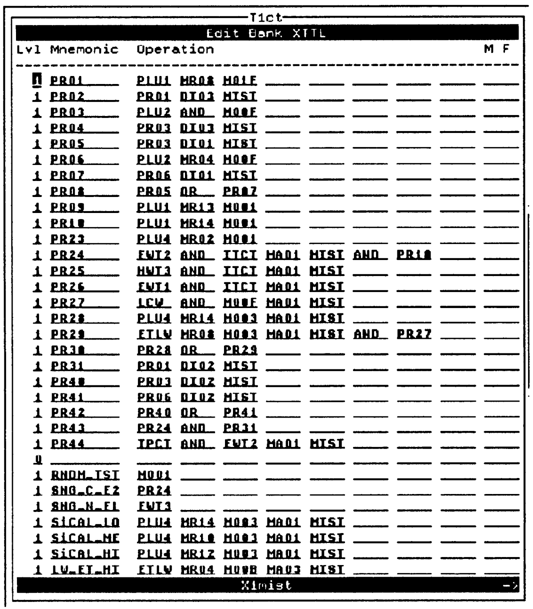
Figure 3.7: Work on Hardware Menu

3.0.4 Define Trigger Bits

Edit XTBN gives a short description of the trigger bits (figure 3.8). You can find here the information which subdetector is used for which trigger bit. It is also used to autodisable a trigger in X1SUPT if the corresponding SD is not in the partition. In *Edit XTTL* the recipe is given how to derive the trigger decision in software (see section 4.3 and appendix G) (figure 3.9). If you want to exchange two lines enter *M* (move) in the right column of the line to be moved and *F* (following) in the right column of the line after which the new line should appear. *Edit XTMS* defines the masks used in *Edit XTTL*.



```
                            T1ct
                      Edit Bank XTBN
No Mnemonic On Description                          Detectors used
-------------------------------------------------------------------
 0 RNDM_TST 1 Random Test Trigger (PlU-level)
 1 SNG_C_E2 1 Sng Charged ElMag & ITC_RPZ(/TPC)   EWT2 ECAL ITC
 2 SNG_N_EL 1 Single Neutral Energy Barrel        EWT3 ECAL
 3 SiCAL_LO 1 SiCAL VL SngArm      --Thrs=VLow     9.  SCAL
 4 SiCAL_ME 1 SiCAL High*Low coinc --Thrs=H.L  24.*12. SCAL
 5 SiCAL_HI 1 SiCAL VH SngArm      --Thrs=VHig    36.  SCAL
 6 LW_ET_HI 1 LCW Total Energy (High Threshold)  ETLW2 LCAL ETOT
 7 LW_A+BVH 1 LCW_A_high.or.B_high & Anti-Spark  ETLW3 LCAL ETOT
 8 SNG_MUON 1 Single Muon (-->No TPC update<--)   HWT3 HCAL ITC
 9 SNG_C_EM 1 Single Charged ElMag (NoTPC)        EWT1 ECAL ITC
10 MIST_LV2 1 L2us after L1us (L2no to Busy TimeOut)
11 ECW_TEST 1 ECW Two Neutral Clusters Lowest Thresh  ECAL
12 -------- 0 ----------------------------------------
13 SiCAL_MX 1 SiCAL (VH*LW+H*H) Coincidence SubSet _ME SCAL
14 BCAL_TRG 1 Single Arm BCAL (10 GeV in 1/4 modules) BCAL
15 FINCHtst 1 BCAL_TRG*SNG_C_EZ (TaggedTwoPhoton)      BCAL ECAL ITC
16 TAG_2GAM 1 SNG_C_E2*(SiCAL_HI+LW_A+BVH)*ITC_BE_2TRK ITC LCAL SCAL ECAL
17 ETT_EWBA 1 Total Energy ECW Barrel            ETEW1 ECAL ETOT
18 ETT_EWEA 1 Total Energy ECW Endcap_A          ETEW2 ECAL ETOT
19 ETT_EWEB 1 Total Energy ECW Endcap_B          ETEW2 ECAL ETOT
20 ETT_EWE* 1 Total Energy ECW EndCapA*EndCapB   ETEW1 ECAL ETOT
21 VDET_LSR 1 MiniVertex LaserShot (External)          VDET
22 BM_RL_MU 1 Beam Related Muon HCW EA*EB (4DP) p.Mod  HCAL
23 TPC_LASR 1 TPC LaserShot (External)                 TPC
24 NWT_TEST 1 Single Neutral energy 3.Threshold        HCAL
25 Z0_Trigg 1 SNG_C_EM.and.SNG_MUON                    HCAL ECAL ITC
26 TRK_CNT1 1 IT_TrkCnt>2  Lv2=TP_TI>2                 ITC  TPC
27 TRK_CNT2 1 ITC_B2B -- Lv2=Lv1.and.(TPTI+TPTO)>0     ITC  TPC  LVL2
28 T0_SYNCH 1 On if T0_Module NOT synchron to beam
29 DBL_C_E2 1 SNG_C_E2 and two ITC tracks (TPC update) ECAL ITC TPC LVL2
30 -------- 0 ----------------------------------------
31 RNDM_TRG 1 DownScaled GBX-Trigger (External:PBB)
                          X1mist
```



```
                            T1ct
                      Edit Bank XTTL
Lvl Mnemonic  Operation                                   M F
-------------------------------------------------------------
 0 PR01       PLU1 MR08 M01F
 1 PR02       PR01 DI03 MIST
 1 PR03       PLU2 AND  M00F
 1 PR04       PR03 DI03 MIST
 1 PR05       PR03 DI01 MIST
 1 PR06       PLU2 MR04 M00F
 1 PR07       PR06 DI01 MIST
 1 PR08       PR05 OR   PR07
 1 PR09       PLU1 MR13 M001
 1 PR10       PLU1 MR14 M001
 1 PR23       PLU4 MR02 M001
 1 PR24       EWT2 AND  ITCI MA01 MIST AND PR10
 1 PR25       HWT3 AND  ITCI MA01 MIST
 1 PR26       EWT1 AND  ITCI MA01 MIST
 1 PR27       LCW  AND  M00F MA01 MIST
 1 PR28       PLU4 MR14 M003 MA01 MIST
 1 PR29       ETLW MR08 M003 MA01 MIST AND PR27
 1 PR30       PR28 OR   PR29
 1 PR31       PR01 DI02 MIST
 1 PR40       PR03 DI02 MIST
 1 PR41       PR06 DI02 MIST
 1 PR42       PR40 OR   PR41
 1 PR43       PR24 AND  PR31
 1 PR44       TPCI AND  EWT2 MA01 MIST
 0
 1 RNDM_TST   M001
 1 SNG_C_E2   PR24
 1 SNG_N_EL   EWT3
 1 SiCAL_LO   PLU4 MR14 M003 MA01 MIST
 1 SiCAL_ME   PLU4 MR10 M003 MA01 MIST
 1 SiCAL_HI   PLU4 MR12 M003 MA01 MIST
 1 LW_ET_HI   ETLW MR04 M00B MA03 MIST
                          X1mist                          ->
```

Figure 3.8: Description of the Trigger Bits

Figure 3.9: Software Trigger Decision

3.0.5 Test TrigType

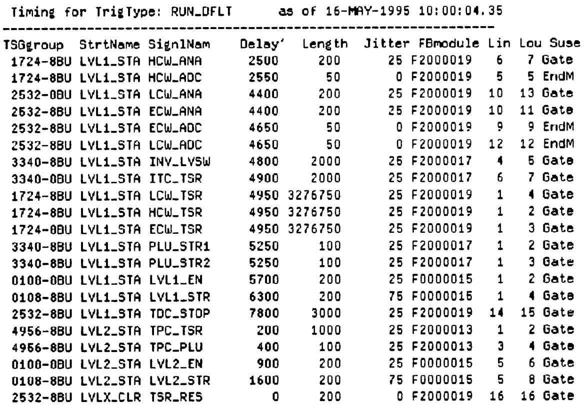
Having chosen the desired trigger type, you can now do what X1SUPT (X1DAQ) does while preparing a run: call XHWTEST, XSETUP... You will get a warning if you try so while the DAQ is active, because this might be destructive — in those cases the task will anyway refuse to follow your orders because the DAQ is not in a state allowing these tasks to take over. The results of the tasks will appear on your screen. In *Test TrgType* you have also the opportunity to review the trigger type and make some printouts of the actual trigger type, the corresponding timing (see figure 3.10) and PLU setting.

3.0.6 Trigger Help/Doc

To retrieve information, type the keyword, press enter and select *List DOCuments with above KeyWord(s)*, then *list*. To leave the file which now pops up, type 'Ctrl Z'.

3.0.7 Accommodation

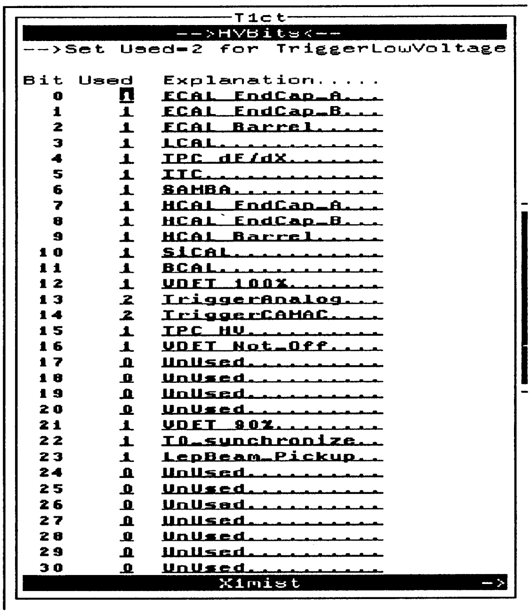
This is the one and only place to rename the FIC: with *FIC_processor* you may define a new name and also declare the names of the tasks and their timeouts. Furthermore, the printer and the X1MIST scratch file are defined with *Printer,Scrfile* as well as the high voltage bits under *HVBits Definitions* (see figure 3.11). With *X1moni Options* one can set thresholds for error messages due to trigger timeout or exceeding maximum



```
Timing for TrigType: RUN_DFLT      as of 16-MAY-1995 10:00:04.35
-------------------------------------------------------------------
TSGgroup  StrtName SignlNam   Delay'  Length  Jitter FBmodule Lin Lou Suse
 1724-8BU LVL1_STA HCW_ANA     2500      200      25 F2000019   6   7 Gate
 1724-8BU LVL1_STA HCW_ADC     2550       50       0 F2000019   5   5 EndM
 2532-0BU LVL1_STA LCW_ANA     4400      200      25 F2000019  10  13 Gate
 2532-8BU LVL1_STA ECW_ANA     4400      200      25 F2000019  10  11 Gate
 2532-8BU LVL1_STA ECW_ADC     4650       50       0 F2000019   9   9 EndM
 2532-8BU LVL1_STA LCW_ADC     4650       50       0 F2000019  12  12 EndM
 3340-8BU LVL1_STA INV_LVSW    4800     2000      26 F2000017   4   5 Gate
 3340-0BU LVL1_STA ITC_TSR     4900     2000      25 F2000017   6   7 Gate
 1724-8BU LVL1_STA LCW_TSR     4950  3276750      25 F2000019   1   4 Gate
 1724-8BU LVL1_STA HCW_TSR     4950  3276750      25 F2000019   1   2 Gate
 1724-0BU LVL1_STA ECW_TSR     4950  3276750      25 F2000019   1   3 Gate
 3340-8BU LVL1_STA PLU_STR1    5250      100      25 F2000017   1   2 Gate
 3340-8BU LVL1_STA PLU_STR2    5250      100      25 F2000017   1   3 Gate
 0108-0BU LVL1_STA LVL1_EN     5700      200      25 F0000015   1   2 Gate
 0108-8BU LVL1_STA LVL1_STR    6300      200      75 F0000016   1   4 Gate
 2532-8BU LVL1_STA TDC_STOP    7800     3000      25 F2000019  14  15 Gate
 4956-8BU LVL2_STA TPC_TSR      200     1000      25 F2000013   1   2 Gate
 4956-8BU LVL2_STA TPC_PLU      400      100      25 F2000013   3   4 Gate
 0108-0BU LVL2_STA LVL2_EN      900      200      25 F0000015   5   6 Gate
 0108-8BU LVL2_STA LVL2_STR    1600      200      75 F0000015   5   8 Gate
 2532-8BU LVLX_CLR TSR_RES        0      200       0 F2000019  16  16 Gate
```

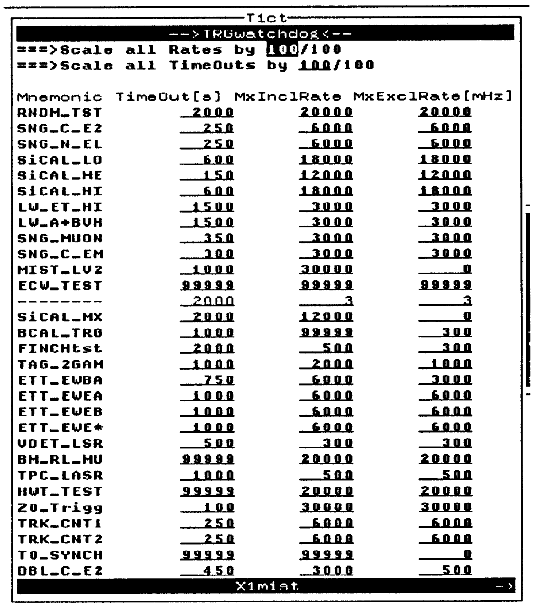
Figure 3.10: The Timing Signals

rate (see figure 3.12); these thresholds are used and compared to the actual values during the run by X1WTCH running on the FIC. *X1veri Options* allows you to switch off the TDC check, usually done in X1VER. The calibration constants for the Level3 tasks are set with *Calib_Constants.* Use *X1supt Options* to set the parameters for the calibration task X1CALB to dis- or enable the 'Autotrigger' which, if set, disables automatically all triggers whose defining detector components are switched off; this avoids unneccessary timeout warnings.



```
                  T1ct
             -->HVBits<--
-->Set Used=2 for TriggerLowVoltage

Bit  Used   Explanation.....
  0    0    ECAL EndCap_A...
  1    1    ECAL EndCap_B...
  2    1    ECAL Barrel.....
  3    1    LCAL............
  4    1    TPC dE/dX.......
  5    1    ITC.............
  6    1    SAMBA...........
  7    1    HCAL EndCap_A...
  8    1    HCAL`EndCap_B...
  9    1    HCAL Barrel.....
 10    1    SiCAL...........
 11    1    BCAL............
 12    1    VDET 100%.......
 13    2    TriggerAnalog...
 14    2    TriggerCAMAC....
 15    1    TPC HV..........
 16    1    VDET Not_Off....
 17    0    UnUsed..........
 18    0    UnUsed..........
 19    0    UnUsed..........
 20    0    UnUsed..........
 21    1    VDET 90%........
 22    1    T0_synchronize..
 23    1    LepBeam_Pickup..
 24    0    UnUsed..........
 25    0    UnUsed..........
 26    0    UnUsed..........
 27    0    UnUsed..........
 28    0    UnUsed..........
 29    0    UnUsed..........
 30    0    UnUsed..........
               X1mist                  ->
```

Figure 3.11: The HV Bits



```
                    T1ct
              -->TRGwatchdog<--
===>Scale all Rates by 100/100
===>Scale all TimeOuts by 100/100

Mnemonic  TimeOut[s] MxInclRate MxExclRate[mHz]
RNDM_TST        2000      20000      20000
SNG_C_E2         250       6000       6000
SNG_N_EL         250       6000       6000
SiCAL_LO         600      18000      18000
SiCAL_HE         150      12000      12000
SiCAL_HI         600      18000      18000
LW_ET_HI        1500       3000       3000
LW_A+BVH        1500       3000       3000
SNG_MUON         350       3000       3000
SNG_C_EM         300       3000       3000
MIST_LV2        1000      30000          0
ECW_TEST       99999      99999      99999
--------        2000          3          3
SiCAL_MX        2000      12000          0
BCAL_TRG        1000      99999        300
FINCHtst        2000        500        300
TAG_2GAM        1000       2000       1000
ETT_EWBA         750       6000       3000
ETT_EWEA        1000       6000       6000
ETT_EWEB        1000       6000       6000
ETT_EWE*        1000       6000       6000
VDET_LSR         500        300        300
BH_RL_MU       99999      20000      20000
TPC_LASR        1000        500        500
HWT_TEST       99999      20000      20000
Z0_Trigg         100      30000      30000
TRK_CNT1         250       6000       6000
TRK_CNT2         250       6000       6000
T0_SYNCH       99999      99999          0
DBL_C_E2         450       3000        500
                 X1mist                  ->
```

Figure 3.12: The Timeout Thresholds

If you have changed something here, do not forget to *Write MONITOR_Info to FIC* afterwards. (This is done automatically by X1SUPT during startup, the watchdog X1WTCH on the FIC checks every five minutes for a new data module.)

3.0.8 Write Monitor_Info to FIC

see **Accommodation**

3.0.9 Start Monitor on FIC

Usually X1SUPT takes care that the monitor task X1WTCH is running; if this is not the case and if it fails to launch the task, a warning will be sent to the error logger.

3.0.10 List DBmods

Lists the time when modifications have been made. Here you can give or deny access to X1MIST: **Only registered users are allowed to use X1MIST.**

3.1 How to do What with X1MIST

There may come the time when it is necessary to change the trigger settings, so make yourself familiar with the procedures in time:

3.1.1 Dis- or Enable a Trigger

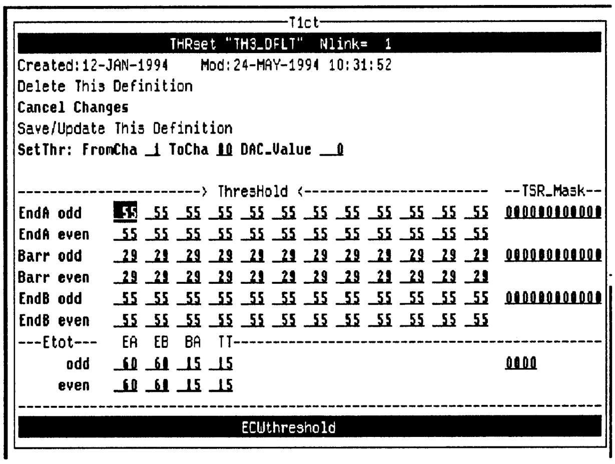
Level1 is using internally a selective readout. The basic information (TSR, PLU, TPR, PBB) is read on every event, while the ADCs, TDCs and TSSs are only read for some triggers. Especially the TSS are read only on random triggers.
To change the trigger mask select: *Edit Resources, TRGoption, TrigType* (most probably *RUN_DFLT*) and *Define Trigger Enable Mask* (see figures 3.5 and 3.6) and go to the appropriate trigger, press the dot '.' on the numerical keypad and select *ON* or *OFF*. Choose *Accept Record - Return* and *Save/Update this Definition.* After a change of the trigger enable mask, take care that the mask is also changed in a_eor$src:runchk_constants.inc. Otherwise this will produce error messages in the aleph run summary. Ask your friendly software manager.
The submasks for the selective readout of the TSSs, ADC and TDC can be changed after selecting the corresponding line *Define ... Readout Mask* from the menu shown in figure 3.5. However, **the TSS should only be read out on random triggers.**

3.1.2 Set Downscale Factors and Random Trigger Parameters

Same as above but stop at the menu shown in figure 3.5. Go to the appropriate downscaler (above the eight downscale values you can see the corresponding trigger) and write the desired value on the line - press *Return*, then *Save/Update this Definition.* The random trigger, which is usually taken each 2 700 000 GBX (30 sec) – 1 350 000 GBX in 4-bunch mode –, may be randomized using the option *variate(+/−)*, e.g. *variate* ±2% will choose the random triggers uniformly from a GBX interval with mean 2 700 000 (1 350 000) and width +/− 54 000 (27 000).
To meet a request of the DAQ, a random trigger scheme where the time difference between two subsequent triggers is exponentially distributed is possible:
This feature is invoked by setting *variate* $= -1$; the mean trigger rate is given by the ratio: $Number\_of\_GBX\_per\_second/Random\_TriggerScalefactor$ (see figure 3.5).

3.1.3 Set Thresholds

When you have determined the effective thresholds with X1CALB (see section 5.14) and their (hopefully) linear dependence with the DAC counts (see section 5.15), you might want to set certain thresholds to new values. Choose the *HCW-,ECW-* or *LCWthrhld* in *Edit Resources* (figure 3.13). Choose which DAC values you want to change, overwrite the value (one by one or use *SetThr: FromCha 1 ToCha 80 DAC_Value 0* instead; *Return*, *Save/Update this Definition.*
Note: If it is really necessary to set bits in the TSR mask – in order not to loose in trigger efficiency because of a module malfunction –, you should know that this mask is not known to the verification (X1VER) and will lead to error messages like 'junk trigger', guaranteed.



```
                              T1ct
                  THRset "TH3_DFLT"  Nlink=  1
Created:12-JAN-1994   Mod:24-MAY-1994 10:31:52
Delete This Definition
Cancel Changes
Save/Update This Definition
SetThr: FromCha _1 ToCha 80 DAC_Value __0

-----------------------> ThresHold <-----------------------  --TSR_Mask--
EndA odd    55  55  55  55  55  55  55  55  55  55  55  55  000000000000
EndA even   55  55  55  55  55  55  55  55  55  55  55  55
Barr odd    29  29  29  29  29  29  29  29  29  29  29  29  000000000000
Barr even   29  29  29  29  29  29  29  29  29  29  29  29
EndB odd    55  55  55  55  55  55  55  55  55  55  55  55  000000000000
EndB even   55  55  55  55  55  55  55  55  55  55  55  55
---Etot---  EA  EB  BA  TT-----------------------------------------------
      odd   60  60  15  15                                  0000
     even   60  60  15  15
-------------------------------------------------------------------------
                          ECWthreshold
```

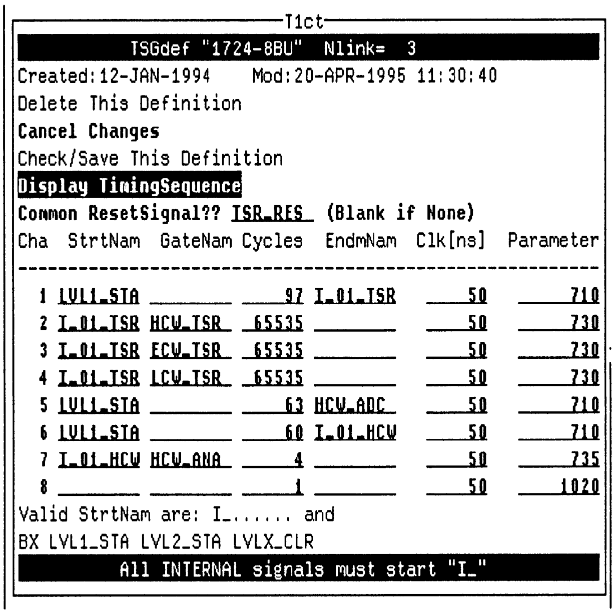
Figure 3.13: Trigger Thresholds

3.1.4 Disable an ADC, TDC or TSS from Readout

If an ADC, TDC or TSS hardware error occurs in XHWTEST during the CONFIG step and prevents ALEPH from running, it might be necessary to disable the component from the readout (if there is not enough time to investigate and fix the problem.) **Note that all other trigger components are absolutely essential for data taking and must not be disabled !**
Select: *Link Trigtype, Run_Dflt* and the component you want to disable (figure 3.3). Write *Dont_use* in the corresponding line, press *Return* and select *Save/Update this Definition* (be patient, very slow).
Do not forget to enable the component again when the problem is fixed. Clear the line of the component and choose, after pressing 'Return', the appropriate run mode from the popped up menu.

3.1.5 Change the Timing

If you have good reasons to believe that the trigger timing has to be adjusted, choose the appropriate TSG in *Edit Resources* and change the programming of the TSG by overwriting the setting (figure 3.14).

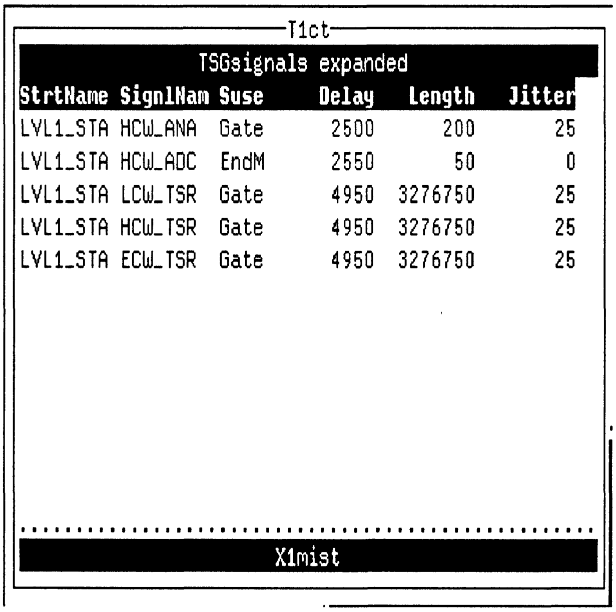


```
                          T1ct
           TSGdef "1724-8BU"  Nlink=  3
Created:12-JAN-1994   Mod:20-APR-1995 11:30:40
Delete This Definition
Cancel Changes
Check/Save This Definition
Display TimingSequence
Common ResetSignal?? ISR_RES  (Blank if None)
Cha  StrtNam  GateNam Cycles  EndmNam  Clk[ns]  Parameter
----------------------------------------------------------
  1 LVL1_STA ________ ______97 I_01_TSR ____50 _______710
  2 I_01_TSR HCW_TSR  _65535 ________ ____50 _______730
  3 I_01_TSR ECW_TSR  _65535 ________ ____50 _______730
  4 I_01_TSR LCW_TSR  _65535 ________ ____50 _______730
  5 LVL1_STA ________ ______63 HCW_ADC  ____50 _______710
  6 LVL1_STA ________ ______60 I_01_HCW ____50 _______710
  7 I_01_HCW HCW_ANA  _______4 ________ ____50 _______735
  8 ________ ________ _______1 ________ ____50 ______1020
Valid StrtNam are: I_...... and
BX LVL1_STA LVL2_STA LVLX_CLR
         All INTERNAL signals must start "I_"
```

Figure 3.14: The TSG 1724-8BU Signals

This example shows the "daisy chain" of some signals:
The starting signal is *LVL1_STA* (issued by the PBB/TSR), its endmarker is generated after 97 clock cycles (each clock cycle lasts 50 ns). This endmarker is used as an

internal signal named *I_01_TSR* which produces the gates *HCW_TSR, ECW_TSR* and *LCW_TSR* each at a length of 65535 cycles. — This large number is necessary in order to hold the signals on the TSR during the level1 and level2 cycle. Note that this TSG receives a hard reset: *TSR_RES* (see the line: *Common ResetSignal??*) —
In addition, the *LVL1_STA* signal generates after 63 cycles the endmarker *HCW_ADC* which is used to gate (length: 1 cycle) the HCW ADC. Finally, after 60 cycles the gate (length: 4 cycles) for the HCW analog crate is produced via the internal signal *I_01_HCW.*
Choosing *Display TimingSequence* from the menu displays the signals, their delays, lengths and jitter (figure 3.15). Note that the switching to the internal signals *I_...*



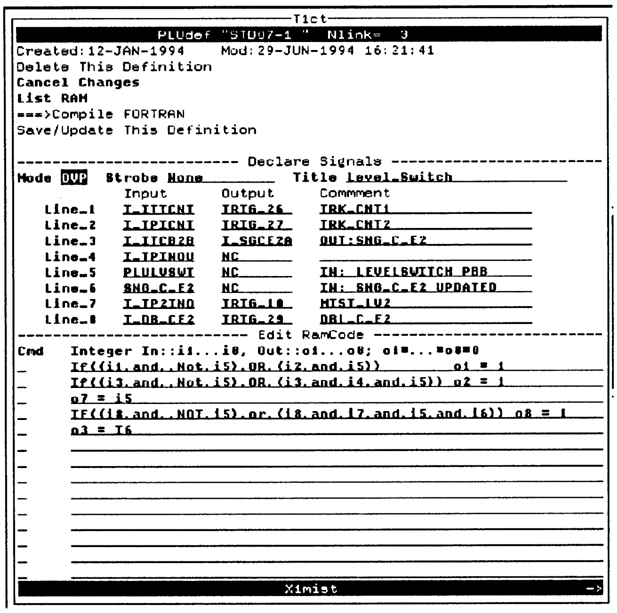
```
                         T1ct
                   TSGsignals expanded
StrtName SignlNam Suse    Delay  Length  Jitter
LVL1_STA HCW_ANA  Gate     2500     200      25
LVL1_STA HCW_ADC  EndM     2550      50       0
LVL1_STA LCW_TSR  Gate     4950 3276750      25
LVL1_STA HCW_TSR  Gate     4950 3276750      25
LVL1_STA ECW_TSR  Gate     4950 3276750      25

.................................................
                         X1mist
```

Figure 3.15: The Timing Sequence

takes 2 cycles.
The most common case is to change the delay of signal A by changing the number of cycles of the signal whose endmarker starts A. Take care that no other signals are affected. **You must not use different clock periods within a group of eight channels!** A printout of the whole trigger timing can be made in *Test Trigtype.* The main timing signals are measured with a TDC and checked by X1VER. You may also survey these TDC values online with X1TDD (see below).

3.1.6 Change the Trigger Logic

If you want to build a new or change an old trigger: Do the neccessary cabling in C2, then choose the affected PLUs in *Edit Resources* and do the programming (then *Compile* and *Save/Update this Definition*) (figure 3.16). Valid output signals (see



```
                              Tict
              PLUdef "STD07-1 "  Nlink=  3
Created:12-JAN-1994   Mod:29-JUN-1994 16:21:41
Delete This Definition
Cancel Changes
List RAM
===>Compile FORTRAN
Save/Update This Definition

------------------------- Declare Signals -----------------------
Mode OVP  Strobe None          Title Level_Switch
               Input      Output     Commment
    Line_1     I_ITTCNI   TRIG_26    TRK_CNT1
    Line_2     I_IPICNI   TRIG_27    TRK_CNT2
    Line_3     I_ITCB2B   I_SGCE2A   OUT:SNG_C_E2
    Line_4     I_IPINOU   NC
    Line_5     PLULVSWI   NC         IN: LEVELSWITCH PBB
    Line_6     SNG_C_E2   NC         IN: SNG_C_E2 UPDATED
    Line_7     I_IP2INO   TRIG_18    MTST_LV2
    Line_8     I_DB_CE2   TRIG_29    DBL_C_E2
-------------------------- Edit RamCode --------------------------
Cmd    Integer In::i1...i8, Out::o1...o8; o1=...=o8=0
_      If((i1.and..Not.i5).OR.(i2.and.i5))          o1 = 1
_      If((i3.and..Not.i5).OR.(i3.and.i4.and.i5)) o2 = 1
_      o7 = i5
_      IF((i8.and..NOT.i5).or.(i8.and.i7.and.i5.and.i6)) o8 = 1
_      o3 = I6
_
_
_
_
_
_
_
_
_
                              X1mist                          ->
```

Figure 3.16: PLU Setting

column *Output* in figure 3.16) are: *TRIG_xx* or an internal signal *I_...* or *NC* for "not connected". The internal signals may cross the boundaries of a single sector e.g. the output signal 'I_SGCE2A' (see figure 3.16) is defined in sector $STD07-1$ but it is used in a different sector. After a change it is wise to check with *Test TrigType* whether the whole set of definitons is consistent. **Note:**

Some PLUs are strobed (*LV1STROBE*, *LV2STROBE*), i.e. their input is looked at (and the output is evaluated) when the strobe signal is present; their hardware switch must be on "cnt" (continuous). The others are operated on the overlap ("ovp") mode: their input is looked at continuously. The corresponding hardware switch and the mode parameter must be on "ovp". The *mode* parameter in the PLU setting menu (figure 3.16) has no effect, it is there as a reminder.

Maybe you have to change the bank XTTL (for verification) and XTBN. It may also be wise to adjust the setting of the maximum rate and timeout in *X1MONI options* under *ACCOMODATION*. A description how to change XTTL can be found in appendix G.

Note:

The run has to be started from INACTIVE (execute RESET, PREPARE) to load the new setting into the global section. Write down in the Shift and Trigger logbooks which runs are affected by your changes and add a comment in X1PRB, where all your changes to the X1MIST data base are logged automatically.

Chapter 4

Permanent Tasks on the VAX

4.1 X1DAQ

See [Trigger.Mist]x1daq.mms for the resources.
X1DAQ runs as the Run Controller X1SUPT in the DAQ during data taking. It performs the transitions between the states of the FSM tree, reports errors and allows to change certain settings of the trigger also from the DAQ console (SABOTAGE option in the X1SUPT menu). Furthermore, X1DAQ writes the Start Of Run Record (SOR, see appendix D) and prepares a special SOR for X1VER and X1MON. Its error messages and explanation can be found in [trigger.doc]xlvl_x1daq.hlp or can be extracted by ERRHELP (see also appendix H). The logfile is written to a_s$log:xt1supt_0.log.

4.2 X1MON

See [Trigger.Ver]x1mon.mms for the resources.
X1MON (called XT1MON in the mainstream) monitors the running of the Level1 trigger and reports all the faults and changes of the system, e.g. it monitors the trigger mask, the low voltages of the trigger crates, the trigger rates etc. Last but not least it provides the histograms for the presenter. Its error messages and explanation can be found in [trigger.ver]xlvl_x1mon.hlp or can be extracted by ERRHELP. The error messages of X1MON can be found in appendix I, the logfile is written to a_s$log:xt1mon_0.log. The helpfile for the presenter histograms on the Level1 shiftpage is placed in [trigger.doc]xt1mon_0.hlp and a_pres$help:xt1mon_0.hlp (see appendix I).

4.3 X1VER

See [Trigger.Ver]x1ver.mms for the resources.
The verification task X1VER (XT1VER in the mainstream) serves the monitoring of the trigger, too. Its main purpose is to check whether the trigger logic is working correctly. The logic is simulated in software using the XTTL bank in the start of run record (see appendices D.7 and G and figure 3.9) . The result is compared with the hardware decision. X1VER's error messages and explanation can be found in [trigger.doc]xlvl_x1ver.hlp. The logfile of X1VER is written to a_s$log:xt1ver_0.log.

4.4 X1ANL

See [Trigger.Ver]x1anl.mms for the resources.
The task X1ANL (XT1ANL in the mainstream) looks at the histograms produced by X1MON for the ITC, TPC, LCW, ECW and HCW segment bits from the registers (TSRs). It looks for large deviations from the mean value, asymmetries in the distributions and other unpleasant features and reports them to ERRLOG and to X1PRB. The logfile is written to a_s$log:xt1anl_0.log.

4.5 X1WTCH

See [Trigger.doc]x1wtch.mms for the resources.
This permanent task performs an automatic logging of the LEP and run status and writes the information to the problem logger under the tag X1WATCH (see section 5.1). In addition X1WTCH cleans the directory [Trigger.Eordata]. All automatic routine checks should be added into this task. **Note:** There exists also a 'X1WTCH' task running on the FIC (see section 6.6) which, however, has nothing to do with this one, running on the VAX.

4.6 Update Error Messages

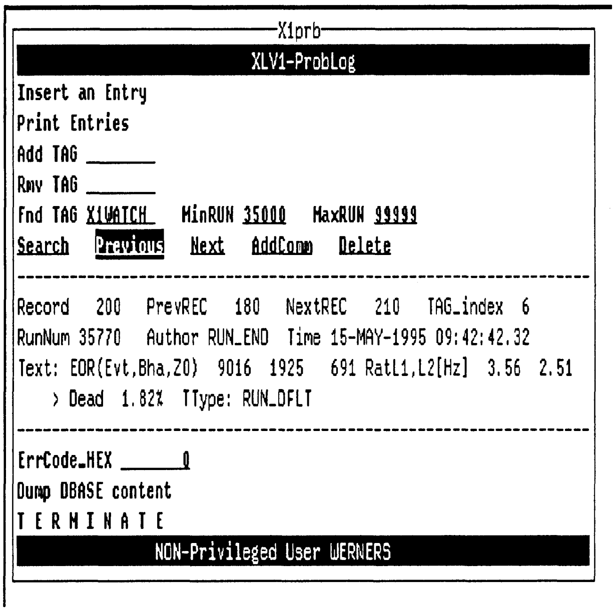
If you want to change or update the error helpfiles of X1MON, X1VER, X1DAQ you first have to edit the appropriate file ([trigger.doc]xlvl_***.hlp). Now copy the new version on a_err$dat:x1***.hlp (*** stands for MON,VER or DAQ) with: *mms/d=[Trigger.doc]x1help* or *@[Trigger]all_update.*

Chapter 5

Utilities on the VAX

5.1 X1PRB

See [Trigger.Doc]x1prb.mms for the resources.
This is the Level1 problem and incident logger. It administrates a data base which contains error messages submitted by the Level1 tasks, important changes in the trigger setup or any malfunction of the trigger and other relevant information.
There are two ways to add an entry: one is via the UPI menu (figure 5.1) which appears when you type *X1prb*; the alternative is realized in the X1*** tasks which write their error messages directly to the DB. All entries are tagged according to the topic or task to which they belong.
To search through the data base, specify the tag in the line *fnd tag* and the run range in the lines *MinRUN* and *MaxRUN*. If you don't know which tags are defined, leave the line in *fnd tag* void and press enter: this shows a menu of the up to date tags. Now choose *search* from the menu: the latest entry belonging to your tag will be displayed. With the help of the commands *Previous* or *Next* you may page through the DB.
Add an entry with *Insert an Entry* which asks you for the tag to which your entry belongs and the text you want to add. New tags may be defined with *add tag*, obsolete tags may be removed with *rmv tag* if they contain no entries. Note that the tags have to be defined **before** any task may write to the DB under the specific tag.
You may also add comments to already existing entries, print the entries or even dump the whole DB (meant for trouble shooting). Since X1MIST reports to X1PRB you are encouraged to comment your changes in the trigger DB if necessary.
To change an existing entry: select the entry, use *AddComment* to write the new entry, then delete the original entry with *Delete*.



```
X1prb
XLV1-ProbLog
Insert an Entry
Print Entries
Add TAG ______
Rmv TAG ______
Fnd TAG X1WATCH   MinRUN 35000   MaxRUN 99999
Search  Previous  Next  AddComm  Delete
-----------------------------------------------------------------
Record   200   PrevREC   180   NextREC   210   TAG_index  6
RunNum 35770   Author RUN_END  Time 15-MAY-1995 09:42:42.32
Text: EOR(Evt,Bha,ZO)  9016  1925   691 RatL1,L2[Hz]  3.56  2.51
    > Dead  1.82%  TType: RUN_DFLT
-----------------------------------------------------------------
ErrCode_HEX ______0
Dump DBASE content
TERMINATE
NON-Privileged User WERNERS
```

Figure 5.1: X1PRB Main menu

5.2 X1ALERT

The source is contained in [Trigger.Doc]x1alert.for .
Another Level1 'Blockwart'. X1ALERT observes the X1PRB data base and notifies you immediately if anybody makes an entry to X1PRB, and what its contents was.

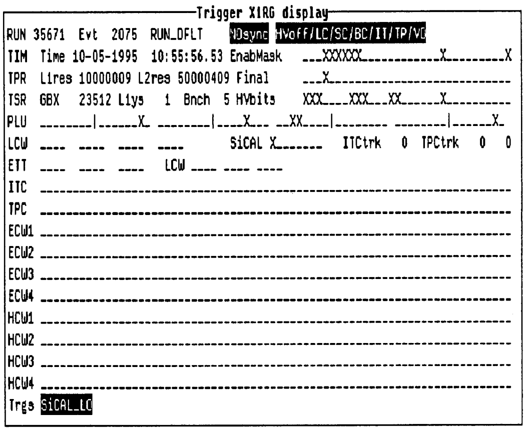
5.3 X1RGD

The source is contained in [Trigger.Ver]x1rgdisp.for .
X1RGD displays the contents of the X1RG bank online (figure 5.2). The commands to steer the program are displayed on the bottom line, e.g. type *F* to display the next event after hitting 'Enter' or type *A* to see a new event every 7 seconds.
Use 'Ctrl W' to refresh the screen, print the screen on the LWACR printer with 'Ctrl P', and leave the program selecting *S* or with 'Ctrl C' or 'Ctrl Z'.

5.4 X1TDD

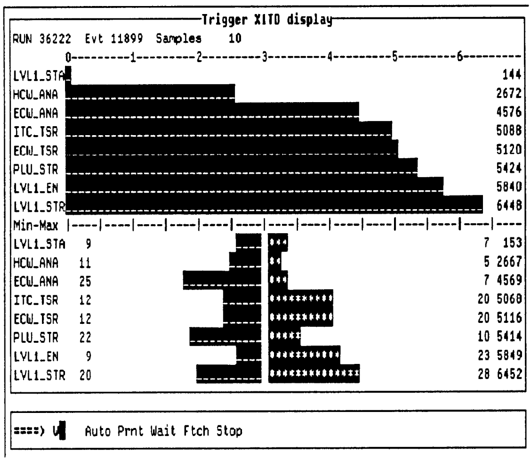
The source is contained in [Trigger.Doc]x1tddisp.for .
X1TDD is similiar to X1RGD: It displays and samples the main timing signals of the trigger together with their variations (figure 5.3). The commands to steer the program are displayed on the bottom line, e.g. type *F* to display the next event after hitting

'Enter' or type *A* to see a new event every 7 seconds.
Use 'Ctrl W' to refresh the screen, print the screen on the LWACR printer with 'Ctrl P', and leave the program selecting *S* or with 'Ctrl C' or 'Ctrl Z'.



```
                    Trigger X1RG display
RUN 35671  Evt  2075  RUN_DFLT    NDsync HVoff/LC/SC/BC/IT/TP/VD
TIM  Time 10-05-1995  10:55:56.53 EnabMask   ___XXXXXX__________X________X
TPR  L1res 10000009 L2res 50000409 Final     ___X__________________________
TSR  GBX   23512 L1ys   1  Bnch  5 HVbits    XXX___XXX__XX_____X_________
PLU  _______|______X_ _______|___X__ __XX___|_______ _______|_____X_
LCW  ____  ____  ____  ____       SiCAL X_______   ITCtrk   0  TPCtrk   0   0
ETT  ____  ____  ____     LCW ____ ____ ____
ITC  _________________________________________________________________
TPC  _________________________________________________________________
ECW1 _________________________________________________________________
ECW2 _________________________________________________________________
ECW3 _________________________________________________________________
ECW4 _________________________________________________________________
HCW1 _________________________________________________________________
HCW2 _________________________________________________________________
HCW3 _________________________________________________________________
HCW4 _________________________________________________________________
Trgs SiCAL_LC
```

Figure 5.2: The X1RGD Bank



```
                    Trigger X1TD display
RUN 36222  Evt 11899  Samples   10
         0---------1---------2---------3---------4---------5---------6---------
LVL1_STA                                                                    144
HCW_ANA                                                                    2672
ECW_ANA                                                                    4576
ITC_TSR                                                                    5088
ECW_TSR                                                                    5120
PLU_STR                                                                    5424
LVL1_EN                                                                    5840
LVL1_STR                                                                   6448
Min-Max |----|----|----|----|----|----|----|----|----|----|----|----|----|----
LVL1_STA   9                                                           7  153
HCW_ANA   11                                                           5 2667
ECW_ANA   25                                                           7 4569
ITC_TSR   12                                                          20 5068
ECW_TSR   12                                                          20 5116
PLU_STR   22                                                          10 5414
LVL1_EN    9                                                          23 5849
LVL1_STR  20                                                          28 6452

====) W   Auto Prnt Wait Ftch Stop
```

Figure 5.3: The X1TDD Display

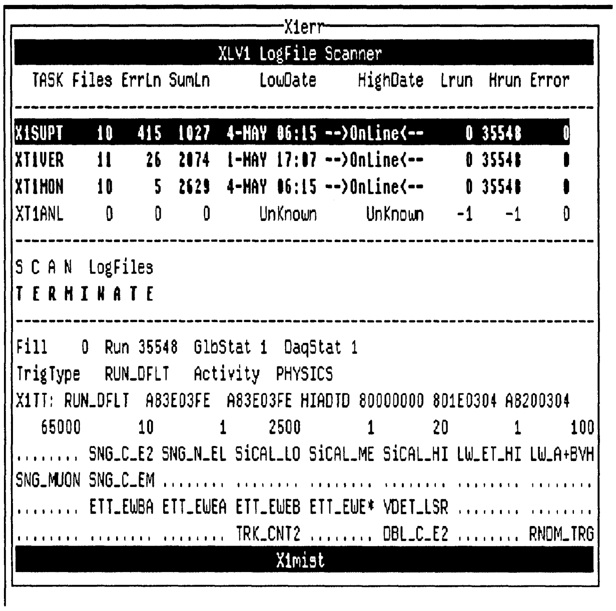
5.5 X1ERR

See [Trigger.Mist]x1err.mms for the resources.
With the help of this task one can conveniently scan the logfiles of X1SUPT, X1MON, X1VER and X1ANL for error messages. To make this program work properly you have to follow some convention in indicating an error message in X1mon.for or X1ver.for. (See [trigger.doc]x1err.rules.) To list all errors reported during a specific period choose *Scan Logfiles* from the menu. Wait until the logfiles are written in bold letters (figure 5.4). Now you can choose a particular task by moving the cursor to the line and pressing 'Return'. The menu will now show you the available logfiles. Select the one with the right date. If you press the '.' on the right side of your keyboard, X1ERR will offer you several options for looking at the logfiles. *FndERR*, showing only the error messages, is the default. **Note:** It is your duty to check all these logfiles before the nine o'clock meeting every morning.

5.6 ERRLOG

> *"It's not an error. It's a feature!"*
> Heard in the nine o'clock meeting

The ERRLOG facility is a more official tool for scanning logfiles from all subdetectors. To use it for trigger purpose select (after giving the desired time range and/or run /fill



```
                         Xlerr
                  XLV1 LogFile Scanner
  TASK Files ErrLn SumLn      LowDate      HighDate  Lrun  Hrun Error
----------------------------------------------------------------------
X1SUPT    10   415  1027  4-MAY 06:15 -->OnLine<--      0 35548     0
XT1VER    11    26  2074  1-MAY 17:07 -->OnLine<--      0 35548     0
XT1MON    10     5  2629  4-MAY 06:15 -->OnLine<--      0 35548     0
XT1ANL     0     0     0      UnKnown       UnKnown    -1    -1     0
----------------------------------------------------------------------
S C A N  LogFiles
T E R M I N A T E
----------------------------------------------------------------------
Fill    0  Run 35548  GlbStat 1  DaqStat 1
TrigType   RUN_DFLT   Activity  PHYSICS
X1TT: RUN_DFLT  A83E03FE  A83E03FE HIADTD 80000000 801E0304 A8200304
   65000      10         1     2500       1        20        1      100
........ SNG_C_E2 SNG_N_EL SiCAL_LO SiCAL_ME SiCAL_HI LW_ET_HI LW_A+BVH
SNG_MUON SNG_C_EM ........ ........ ........ ........ ........ ........
........ ETT_EWBA ETT_EWEA ETT_EWEB ETT_EWE* VDET_LSR ........ ........
........ ........ ........ TRK_CNT2 ........ DBL_C_E2 ........ RNDM_TRG
                          X1mist
```

Figure 5.4: X1ERR Main menu

number) the detector XLV1. The facility can be X1MON, X1VER or X1DAQ or all if you leave it blank. *Sorted* is a sensible choice to *Display the information.* Building the output file may take a while. The message window will tell you when it is ready.

5.7 ErrHelp

ErrHelp gives a short explanation for every trigger error message that may appear on the error logger. Use it e.g. in the following way:

Specify (part of) the library name : xlv1
*** Ambiguous file name ***
XLV1_X1DAQ.HLB
XLV1_X1MON.HLB
XLV1_X1VER.HLB
Specify (part of) the library name : x1daq
Information available:
Ficwtch GBXwait Modtrip Mskchng PBBdisa RObIk Sabotag
TPRerrs Trbit Trgwait TrigHvb
Topic?
GBXwait
!!Either there is a DAQ error, e.g. TSCON got stuck, or*
!!the DAQ-operator is fighting a DAQerr or*

!!the trigger is disabled unintentionally or*
!!Trigger-Level1 has a HardWare-problem (very improbable).*
!!Call TriggerExpert only, if you have evidence that the*
!!LEVEL1 hardware is faulty*
Press 'Return' several times to leave the program.

5.8 CIA

If you have to downgrade a run to MAYBE or DUCK you have to use the CIA facility. The downgrading is **NOT** done by the Run Quality. See Subdetector Coordinator Manual for further information.

5.9 PRESENTER

Start the PRESENTER with *Pres.*
However, this doesn't work on every workstation, see Subdetector Coordinator Manual for further information.

5.10 HEDT

With this tool you define the histograms and time charts which are generated by the monitoring tasks and which are to be shown in the PRESENTER. Take care that the savesets of the generating task are also attached to X1ANL. An overview of the existing Level1 histograms can be found in [Trigger.Doc]presenter.info .

5.11 X1PRES

See [Trigger.Ver]x1pres.mms for the resources.
This is the "Poor Man's Presenter" for XLV1 histos.
X1PRES makes it possible to look at the Level1 histograms when not sitting in front of an AXADAQ terminal. Choose the X1MON or X1VER histograms. Select the desired run or fill. Then *Show Hist* or *Plot Hist.* Press 'Return' if you don't know which number your histogram has, and use the appearing menu. Press 'Return' to see the contents of the histogram rows in numbers. Use 'F12' to go back to the main menu.

5.12 X1CHART

See [Trigger.Ver]x1chart.mms for the resources.
"Poor Man's Presenter" for XLV1 TimeCharts.
After giving a start time and a period you can choose between five timecharts (995 - 999) and scan or rather plot them, their plot output is written to [Trigger.data]xlvl_tc*.ps .

5.13 HSAVEREF

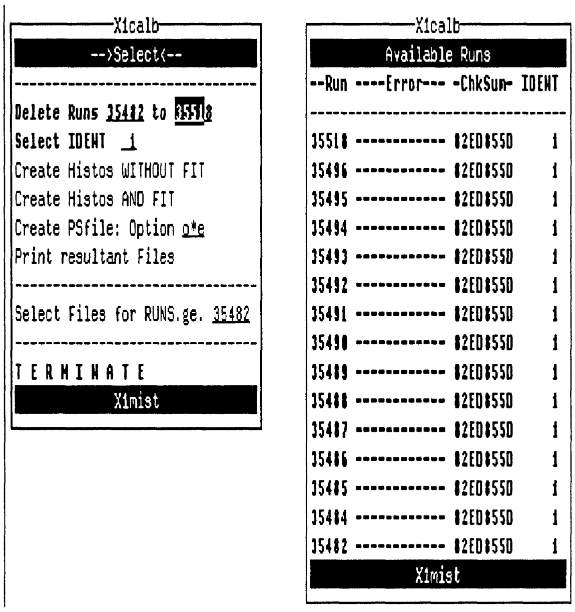
Generate Reference Histos for X1MON and X1VER.
You should select a good fill for the reference histograms. The required saveset name is XT1MON or XT1VER. See Subdetector Coordinator Manual for further information.

5.14 X1CALB

See [Trigger.Calb]x1calb.mms for the resources.
It is also your duty to control the trigger thresholds from time to time and to adjust them if necessary. (Report in the nine o' clock meeting before!) You can determine them with X1CALB which picks up histogrammed information in a run–by–run basis compiled during the Run Quality checks and calculates the threshold behaviour. A fit then calculates the effective thresholds. Type X1CALB and choose in the appearing UPI menu (figure 5.5) from which run onwards you wish to determine the thresholds (A maximum of 100 runs is kept on disk). *Select IDENT 1* selects the desired runs, choose *Create histos* and *fit.* X1CALB now shows you the results. Press 'F12' to go back to the main menu. For *Create PS file: Option*____choose *ALL* or *o*e* with the help of the '.' on the right keypad. Print the resultant files. The thresholds and the fit results are now plotted on the printer in the control room. Compare the thresholds fit to previous measurements. Watch out for inefficiencies! Use THRFIT to investigate the linearity of the effective thresholds with the discriminator counts, the slopes and the possible offsets.

5.15 THRFIT

See [Trigger.Calb]x1thrfit.mms for the resources.
Fill in the DAC values and corresponding thresholds derived by X1CALB in the appropriate rows. Choose *Perform Fit* and *Plot* the result. You must give a title first. It will help you to adjust any threshold properly if necessary. Don't forget to update the DB, i.e. use X1MIST (→ *ACCOMMODATION* → *CALIB constants*) to write the new slopes and offsets to the DB.



```
X1calb
-->Select<--
--------------------------------
Delete Runs 35482 to 35518
Select IDENT  1
Create Histos WITHOUT FIT
Create Histos AND FIT
Create PSfile: Option o*e
Print resultant Files
--------------------------------
Select Files for RUNS.ge. 35482
--------------------------------
T E R M I N A T E
X1mist
```

```
X1calb
Available Runs
--Run ----Error--- -ChkSum- IDENT
---------------------------------
35518 ------------ 02ED0550    1
35496 ------------ 02ED0550    1
35495 ------------ 02ED0550    1
35494 ------------ 02ED0550    1
35493 ------------ 02ED0550    1
35492 ------------ 02ED0550    1
35491 ------------ 02ED0550    1
35490 ------------ 02ED0550    1
35489 ------------ 02ED0550    1
35488 ------------ 02ED0550    1
35487 ------------ 02ED0550    1
35486 ------------ 02ED0550    1
35485 ------------ 02ED0550    1
35484 ------------ 02ED0550    1
35482 ------------ 02ED0550    1
X1mist
```

Figure 5.5: X1CALB Main menu

5.16 X1XSUM

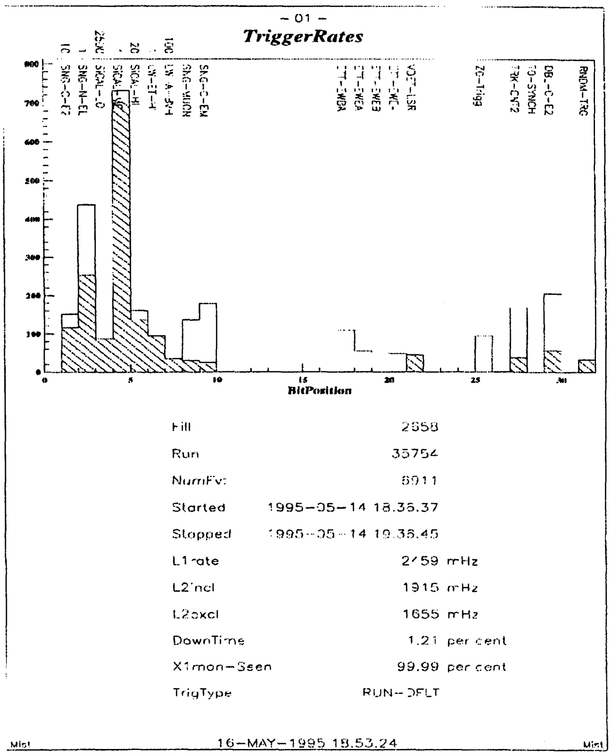
See [Trigger.Calb]x1xsum.mms for the resources.
X1XSUM prints a table with the inclusive and exclusive Rates of all enabled triggers of that run from data compiled by the Run Quality (figure 5.6).

5.17 X1RUNQ

See [Trigger.Calb]x1runq.for for the resources.
X1RUNQ prints the Run Quality output for XLV1 after prompting you for the run number. An example is given in (figure 5.7).

5.17.1 Update the Run Quality Code

The Run Quality resources for Level1 are kept in the directory [trigger.runq] and its subdirectories [.import] and [.export]. Your duties in maintaining the code are described in [...import]readme .
Go to the subdirectory [...export]. The Run Quality source is in xlv1rq.for and in xlrqcm.inc . There are two more include files (x1meor.inc and x1veor.inc) that will be automatically copied and included from [trigger.ver].



```
Fill                          2558
Run                          35754
NumEv:                        8911
Started       1995-05-14 18.36.37
Stopped       1995-05-14 19.36.45
L1rate                        2459 mHz
L2incl                        1915 mHz
L2excl                        1655 mHz
DownTime                      1.21 per cent
X1mon-Seen                   99.99 per cent
TrigType                  RUN-DFLT

Mist               16-MAY-1995 18.53.24               Mist
```

Figure 5.6: The Run Summary

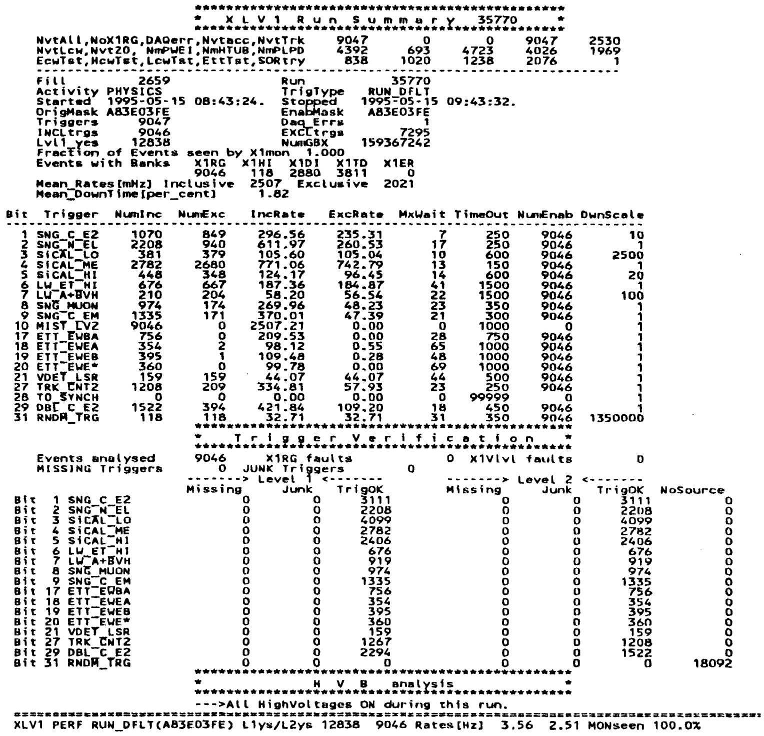
1. Edit and update the source files. If you have lost the source files, go to [trigger.runq] and do *mms/d=import.* This will fetch the latest version from disk$user:[runqmgr.export] and unpack the saveset into [trigger.runq.import] . From there you can now copy the Run Quality source files to [...export].

2. Test the code. Still you are in [trigger.runq.export]. Now do *mms/d=rqtest.* This will fetch two '.inc' files if neccessary and will also create the new saveset after successful linking.

3. Do rqtest (Use the .com file, since some logicals have to be defined). You are queried for the full pathname of an epio–file. The output goes to [trigger.runq].

4. If everything looks fine, you can export the saveset. Do *copy [trigger.runq.export]x1lv1rq.bck axaonl"runqusr password":: [runqusr]*.** . Get the password from the Run Quality guys and send a mail to AXAONL::RUNQMGR to notify them that you have submitted new code, along with a brief description of the changes.

The Run Quality program running in FALCON will, from 1995 on, copy the LEVEL1 summary for each run to [trigger.runsum]x1runq_xxxxx.out . The data for calibration will be written to [trigger.eordata]xlvlthr_xxxxx.native from where they can be picked up by X1CALB for the extraction of the nominal thresholds.



```
                         ************************************************
                         *   X L V 1   R u n   S u m m a r y   35770    *
                         ************************************************
  NvtAll,NoX1RG,DAQerr,Nvtacc,NvtTrk     9047       0       0    9047    2530
  NvtLcw,Nvt20, NmPWEI,NmHTUB,NmPLPD     4392     693    4723    4026    1969
  EcwTst,HcwTst,LcwTst,EttTst,SORtry      838    1020    1238    2076       1
  ----------------------------------------------------------------------------
  Fill         2659                Run            35770
  Activity PHYSICS                 TrigType    RUN_DFLT
  Started  1995-05-15 08:43:24.    Stopped    1995-05-15 09:43:32.
  OrigMask A83E03FE                EnabMask    A83E03FE
  Triggers      9047               Daq_Errs           1
  INCLtrgs      9046               EXCLtrgs        7295
  Lvl1_yes     12838               NumGBX     159367242
  Fraction of Events seen by X1mon  1.000
  Events with Banks     X1RG   X1HI   X1DI   X1TD   X1ER
                        9046    118   2880   3811      0
  Mean_Rates[mHz] Inclusive  2507  Exclusive  2021
  Mean_DownTime[per_cent]       1.82

Bit  Trigger  NumInc  NumExc   IncRate   ExcRate  MxWait TimeOut NumEnab DwnScale
-------------------------------------------------------------------------------
  1 SNG_C_E2    1070     849    296.56    235.31      7     250    9046       10
  2 SNG_N_EL    2208     940    611.97    260.53     17     250    9046        1
  3 SiCAL_LO     381     379    105.60    105.04     10     600    9046     2500
  4 SiCAL_ME    2782    2680    771.06    742.79     13     150    9046        1
  5 SiCAL_HI     448     348    124.17     96.45     14     600    9046       20
  6 LW_ET_HI     676     667    187.36    184.87     41    1500    9046        1
  7 LW_A+BVH     210     204     58.20     56.54     22    1500    9046      100
  8 SNG_MUON     974     174    269.96     48.23     23     350    9046        1
  9 SNG_C_EM    1335     171    370.01     47.39     21     300    9046        1
 10 MIST_LV2    9046       0   2507.21      0.00      0    1000       0        1
 17 ETT_EWBA     756       0    209.53      0.00     28     750    9046        1
 18 ETT_EWEA     354       2     98.12      0.55     65    1000    9046        1
 19 ETT_EWEB     395       1    109.48      0.28     48    1000    9046        1
 20 ETT_EWE*     360       0     99.78      0.00     69    1000    9046        1
 21 VDET_LSR     159     159     44.07     44.07     44     500    9046        1
 27 TRK_CNT2    1208     209    334.81     57.93     23     250    9046        1
 28 TO_SYNCH       0       0      0.00      0.00      0   99999       0        1
 29 DBL_C_E2    1522     394    421.84    109.20     18     450    9046        1
 31 RNDM_TRG     118     118     32.71     32.71     31     350    9046  1350000
                         ************************************************
                         *   T r i g g e r   V e r i f i c a t i o n    *
                         ************************************************
  Events analysed      9046     X1RG faults          0  X1Vlvl faults        0
  MISSING Triggers        0  JUNK Triggers      0
                      -------> Level 1 <-------     -------> Level 2 <-------
                      Missing      Junk   TrigOK     Missing      Junk   TrigOK  NoSource
   Bit  1 SNG_C_E2          0         0     3111           0         0     3111         0
   Bit  2 SNG_N_EL          0         0     2208           0         0     2208         0
   Bit  3 SiCAL_LO          0         0     4099           0         0     4099         0
   Bit  4 SiCAL_ME          0         0     2782           0         0     2782         0
   Bit  5 SiCAL_HI          0         0     2406           0         0     2406         0
   Bit  6 LW_ET_HI          0         0      676           0         0      676         0
   Bit  7 LW_A+BVH          0         0      919           0         0      919         0
   Bit  8 SNG_MUON          0         0      974           0         0      974         0
   Bit  9 SNG_C_EM          0         0     1335           0         0     1335         0
   Bit 17 ETT_EWBA          0         0      756           0         0      756         0
   Bit 18 ETT_EWEA          0         0      354           0         0      354         0
   Bit 19 ETT_EWEB          0         0      395           0         0      395         0
   Bit 20 ETT_EWE*          0         0      360           0         0      360         0
   Bit 21 VDET_LSR          0         0      159           0         0      159         0
   Bit 27 TRK_CNT2          0         0     1267           0         0     1208         0
   Bit 29 DBL_C_E2          0         0     2294           0         0     1522         0
   Bit 31 RNDM_TRG          0         0        0           0         0        0     18092
                         ************************************************
                         *             H  V  B   analysis               *
                         ************************************************
                         --->All HighVoltages ON during this run.
================================================================================
 XLV1 PERF RUN_DFLT(A83E03FE) L1ys/L2ys 12838  9046 Rates[Hz]  3.56  2.51 MONseen 100.0%
```

Figure 5.7: The Run Quality output

5.18 X1TRGD

See [Trigger.Doc]x1trgd.mms for the resources.

X1TRGD administrates a list of the currently defined triggers and their parameters, like downscaling factors, thresholds in GeV and DAC counts, the signals from which the triggers are derived and the position within the 32 trigger bits; enabled triggers are marked with an asterisk (figure 5.8). However, this program is not linked to any DB, it is just an independent bookkeeper.

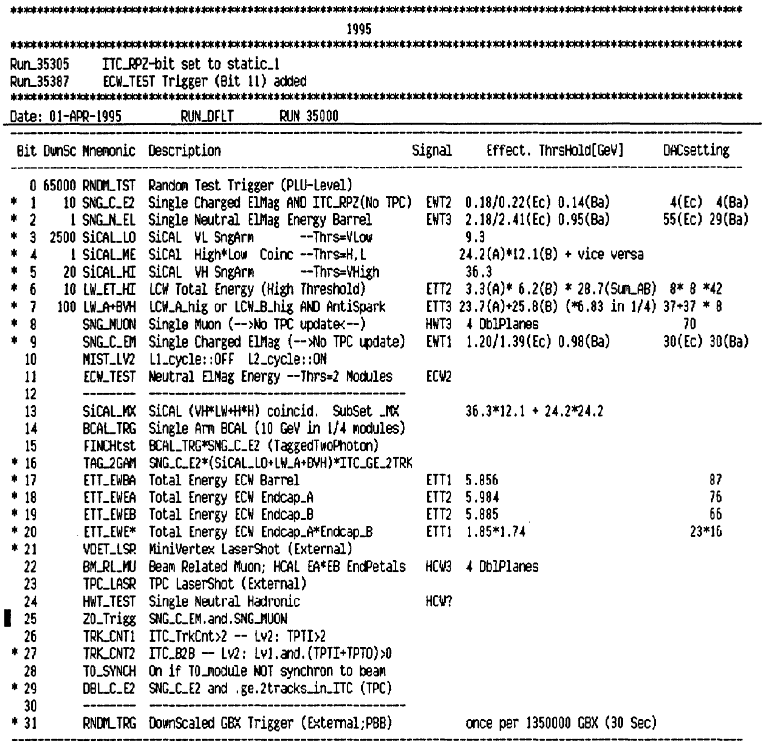
Note:

After any change in one of these items, e.g. setting of new thresholds or downscaling factors, dis- or enabling of triggers, this list has to be updated and posted above the Level1 console; put one copy into the appropriate trigger folder. Don't forget to note the date and the run number of this change!

5.19 X1NEWS

See [Trigger.Doc]x1news.mms for the resources.

A pinboard for the online group to announce trigger related messages, informations or



```
***************************************************************************************************
                                                1995
***************************************************************************************************
Run_35305     ITC_RPZ-bit set to static_1
Run_35387     ECW_TEST Trigger (Bit 11) added
***************************************************************************************************
Date: 01-APR-1995        RUN_DFLT      RUN 35000
---------------------------------------------------------------------------------------------------
 Bit DwnSc Mnemonic  Description                           Signal    Effect. ThrsHold[GeV]      DACsetting
---------------------------------------------------------------------------------------------------
   0 65000 RNDM_TST  Random Test Trigger (PLU-Level)
*  1    10 SNG_C_E2  Single Charged ElMag AND ITC_RPZ(No TPC)  EWT2  0.18/0.22(Ec) 0.14(Ba)         4(Ec)  4(Ba)
*  2     1 SNG_N_EL  Single Neutral ElMag Energy Barrel        EWT3  2.18/2.41(Ec) 0.95(Ba)        55(Ec) 29(Ba)
*  3  2500 SiCAL_LO  SiCAL  VL SngArm       --Thrs=VLow              9.3
*  4     1 SiCAL_ME  SiCAl  High*Low  Coinc --Thrs=H,L              24.2(A)*12.1(B) + vice versa
*  5    20 SiCAL_HI  SiCAL  VH SngArm       --Thrs=VHigh             36.3
*  6    10 LW_ET_HI  LCW Total Energy (High Threshold)         ETT2  3.3(A)* 6.2(B) * 28.7(Sum_AB)  8* 8 *42
*  7   100 LW_A+BVH  LCW_A_hig or LCW_B_hig AND AntiSpark      ETT3 23.7(A)+25.8(B) (*6.83 in 1/4) 37+37 * 8
*  8       SNG_MUON  Single Muon (-->No TPC update<--)         HWT3  4 DblPlanes                     70
*  9       SNG_C_EM  Single Charged ElMag (-->No TPC update)   EWT1  1.20/1.39(Ec) 0.98(Ba)        30(Ec) 30(Ba)
  10       MIST_LV2  L1_cycle::OFF  L2_cycle::ON
  11       ECW_TEST  Neutral ElMag Energy --Thrs=2 Modules     ECW2
  12       --------  ------------------------------------
  13       SiCAL_MX  SiCAL (VH*LW+H*H) coincid.  SubSet _MX        36.3*12.1 + 24.2*24.2
  14       BCAL_TRG  Single Arm BCAL (10 GeV in 1/4 modules)
  15       FINCHtst  BCAL_TRG*SNG_C_E2 (TaggedTwoPhoton)
* 16       TAG_2GAM  SNG_C_E2*(SiCAL_LO+LW_A+BVH)*ITC_GE_2TRK
* 17       ETT_EWBA  Total Energy ECW Barrel                   ETT1  5.856                              87
* 18       ETT_EWEA  Total Energy ECW Endcap_A                 ETT2  5.984                              76
* 19       ETT_EWEB  Total Energy ECW Endcap_B                 ETT2  5.885                              66
* 20       ETT_EWE*  Total Energy ECW Endcap_A*Endcap_B        ETT1  1.85*1.74                       23*16
* 21       VDET_LSR  MiniVertex LaserShot (External)
  22       BM_RL_MU  Beam Related Muon; HCAL EA*EB EndPetals   HCW3  4 DblPlanes
  23       TPC_LASR  TPC LaserShot (External)
  24       HWT_TEST  Single Neutral Hadronic                   HCW?
█ 25       ZO_Trigg  SNG_C_EM.and.SNG_MUON
  26       TRK_CNT1  ITC_TrkCnt>2 -- Lv2: TPTI>2
* 27       TRK_CNT2  ITC_B2B -- Lv2: Lv1.and.(TPTI+TPTO)>0
  28       TO_SYNCH  On if TO_module NOT synchron to beam
* 29       DBL_C_E2  SNG_C_E2 and .ge.2tracks_in_ITC (TPC)
  30       --------  ------------------------------------
* 31       RNDM_TRG  DownScaled GBX Trigger (External;PBB)           once per 1350000 GBX (30 Sec)
---------------------------------------------------------------------------------------------------
```

Figure 5.8: Triggerbit Synopsis

news. This is a quick and dirty tool and allows only for single-user access. In principle the locking/unlocking of records can easily be done to allow for multi-user-access; Maybe, YOU would like to improve it...

5.20 XPRINT

Print TextFile double_sided to LWACR.
Usage: *xprint filename.filetype*. The file is printed on the printer in the ALEPH control room.

5.21 PSPRINT

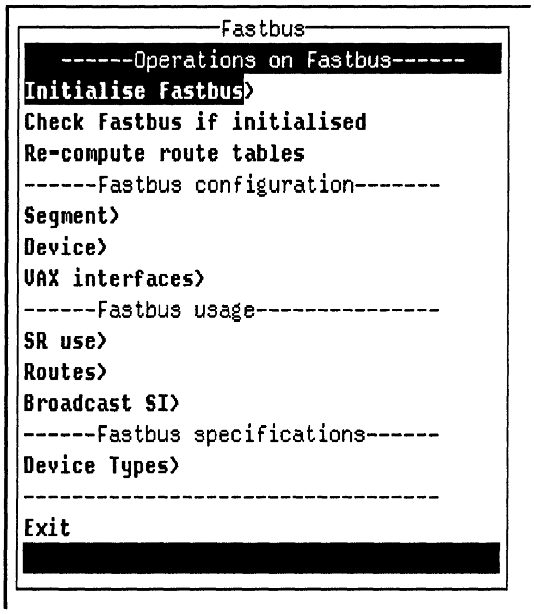
Print POST_File double_sided to LWACR$POST.

5.22 All_UPDATE

The command file [Trigger]all_update.com invokes an update of the above mentioned X1*** programs and calls in turn (and in the right order) all mms–files; x1help files are updates as well. Works only on the TRIGGER account; simply do: @[Trigger]all_update .

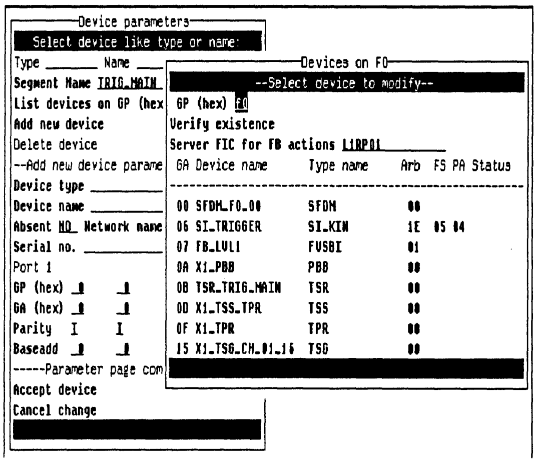
5.23 FBDB

FBDB administrates the Fastbus Data Base (see figure 5.9). All Fastbus modules must be declared here, since X1MIST and X1DAQ receive their information (names, addresses, ID's etc.) from this data base. (Note that the PBB is overwritten in the access routine for X1MIST/X1DAQ because the PBB has to be declared as a TSR in the Fastbus Data Base) Choosing *segments* from the main menu gives you an overview over the Fastbus segment you are interested in:
Enter *TRIG_MAIN* in the *Select names like*_____ option and e.g. *F0* in the *Show devices on the GP* line to get the modules of the F0 crate (figure 5.10). New modules can be added to the data base with the option *Device* followed by *Add new device*. If you introduce or replace a Fastbus module, you have to obey the naming convention of the device:



```
Fastbus
------Operations on Fastbus------
Initialise Fastbus>
Check Fastbus if initialised
Re-compute route tables
------Fastbus configuration-------
Segment>
Device>
VAX interfaces>
------Fastbus usage---------------
SR use>
Routes>
Broadcast SI>
------Fastbus specifications------
Device Types>
---------------------------------
Exit
```

Figure 5.9: The FBDB Main Menu



```
Device parameters
Select device like type or name:
Type _______ Name ____
Segment Name TRIG_MAIN
List devices on GP (hex
Add new device
Delete device
--Add new device parame
Device type ___________
Device name ___________
Absent NO  Network name
Serial no. ___________
Port 1
GP (hex) _0     _0
GA (hex) _0     _0
Parity   I      I
Baseadd  _0     _0
-----Parameter page com
Accept device
Cancel change

Devices on F0
--Select device to modify--
GP (hex) F0
Verify existence
Server FIC for FB actions L1RP01
GA Device name        Type name     Arb  FS PA Status
-------------------------------------------------------
00 SFDM_F0_00         SFDM          00
06 SI_TRIGGER         SI_KIN        1E   05 04
07 FB_LVL1            FVSBI         01
0A X1_PBB             PBB           00
0B TSR_TRIG_MAIN      TSR           00
0D X1_TSS_TPR         TSS           00
0F X1_TPR             TPR           00
15 X1_TSG_CH_01_16    TSG           00
```

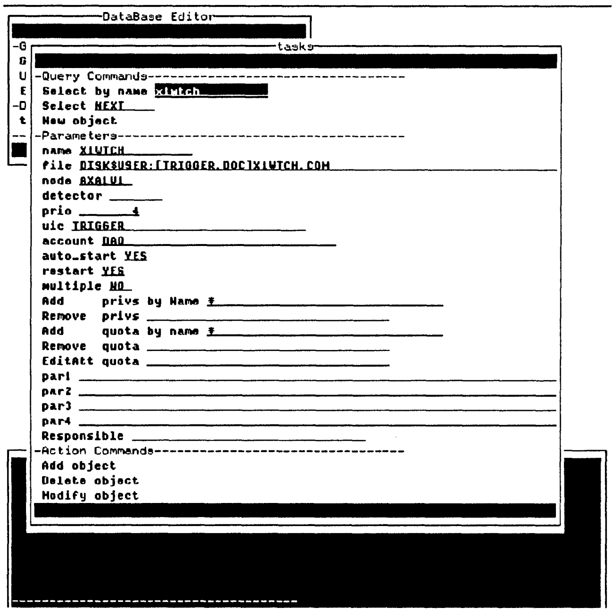
Figure 5.10: The Modules in F0

The name has the structure *X1_nnn[rest]* where *nnn* is the device type, i.e. *ADC, TSR, TSS, TDC, CBD, FBD, TPR, TSG, PLU#,...* and *rest* is the abbreviation of the corresponding SD together with the threshold. For the PLUs, the *'#'*is the number

of the PLU followed by the section: *sec0* or *sec1*.
You must follow this convention, or the module will not be recognized.
The command *FBDB/act [/display=NAME/from=GP/node=FIC_NAME]* initializes the Fastbus tree; a display of the tree is produced, and faults in the initialization are marked in bold or blinking display. The option */display=NAME* shows the SD names for each segment. Using */ver* instead of */act* only checks the status. The */help* qualifier displays all options.

5.24 SCHEDB

SCHEDB allows you to search through the Scheduler DB. It contains about 350 tasks and their parameters such as the name of the file to run as batch job, on which node to run or whether the task starts automatically (see figure 5.11).

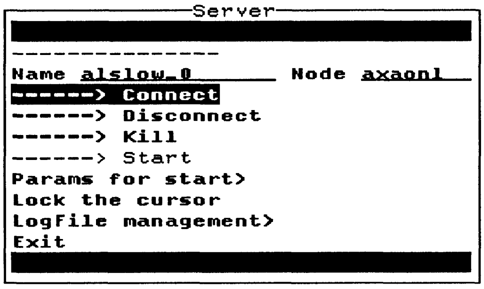


```
DataBase Editor
tasks
-Query Commands-----------------------------------
 Select by name xiwtch
 Select NEXT
 New object
-Parameters---------------------------------------
 name XIWTCH
 file DISK$USER:[TRIGGER.DOC]XIWTCH.COM
 node AXALUI
 detector ______
 prio _______4
 uic TRIGGER
 account DAQ
 auto_start YES
 restart YES
 multiple NO
 Add     privs by Name *
 Remove  privs
 Add     quota by name *
 Remove  quota
 EditAtt quota
 par1
 par2
 par3
 par4
 Responsible
-Action Commands----------------------------------
 Add object
 Delete object
 Modify object
```

Figure 5.11: The Scheduler Data Base Administrator

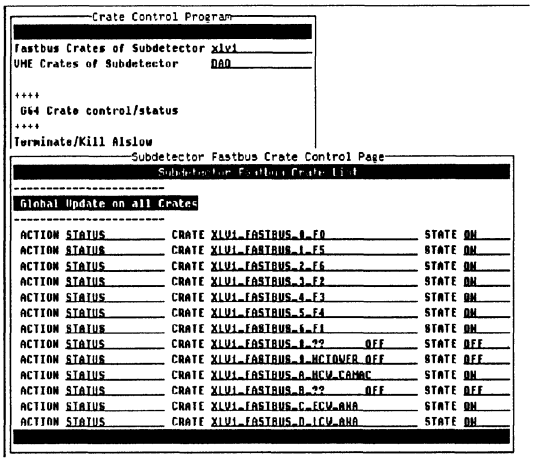
5.25 SERV

SERV allows you to connect to a task and and to give it commands and to receive messages from it. To connect to an already running task, enter the task name (including the version number) and the node on which it is running.
See figure 5.12 where a connection to the task ALSLOW_0 is prepared. After executing *connect*, the UPI menu with the header *Crate Control Program* appears. If you write *xlv1* in the parameter line *Fastbus Crates of Subdetector* ____ (*VME Crates of Subdetector* ____), you will get an overview over the Level1 Fastbus (VME) crates and the status of their power supplies (right column, see figure 5.13). Maybe the *State* row says *unknown*; in this case use the command *Global Update on all Crates.* Now you may switch off or on the power supplies for the particular Fastbus (VME) crates by typing *off* or *on* in the right column of the crate you have in mind. Contrary to the FB crates, the power supplies for the CAMAC crate housing the PLUs cannot be cycled via ALSLOW.
Note that the Level1 FIC (L1RP01) sits in the HCAL_MAIN VME crate.
Before you leave the server, you have to disconnect from the task. Otherwise you will kill the task you are connected to.



```
Server
---------------
Name alslow_0          Node axaonl
------> Connect
------> Disconnect
------> Kill
------> Start
Params for start>
Lock the cursor
LogFile management>
Exit
```

Figure 5.12: The Server Main Menu



```
Crate Control Program
Fastbus Crates of Subdetector xlv1
VME Crates of Subdetector     DAQ

++++
 G64 Crate control/status
++++
Terminate/Kill Alslow

Subdetector Fastbus Crate Control Page
Subdetector Fastbus Crate List
-----------------------
Global Update on all Crates
-----------------------
ACTION STATUS    CRATE XLV1_FASTBUS_0_F0                 STATE ON
ACTION STATUS    CRATE XLV1_FASTBUS_1_F5                 STATE ON
ACTION STATUS    CRATE XLV1_FASTBUS_2_F6                 STATE ON
ACTION STATUS    CRATE XLV1_FASTBUS_3_F2                 STATE ON
ACTION STATUS    CRATE XLV1_FASTBUS_4_F3                 STATE ON
ACTION STATUS    CRATE XLV1_FASTBUS_5_F4                 STATE ON
ACTION STATUS    CRATE XLV1_FASTBUS_6_F1                 STATE ON
ACTION STATUS    CRATE XLV1_FASTBUS_8_??        OFF      STATE OFF
ACTION STATUS    CRATE XLV1_FASTBUS_9_HCTOWER OFF        STATE OFF
ACTION STATUS    CRATE XLV1_FASTBUS_A_HCV_CAMAC          STATE ON
ACTION STATUS    CRATE XLV1_FASTBUS_B_??        OFF      STATE OFF
ACTION STATUS    CRATE XLV1_FASTBUS_C_ECV_ANA            STATE ON
ACTION STATUS    CRATE XLV1_FASTBUS_D_ICV_ANA            STATE ON
```

Figure 5.13: The FB power supplies

Chapter 6

Permanent Tasks on the FIC

"...just boot his bloody fucking FIC !"
heard in the control room

6.1 XSETUP

The source is contained in /os9/usr/levell/vme/source/xsetup.c .
XSETUP is used to download the trigger setup into the hardware (thresholds, trigger mask, timings, etc). It is called in the CONFIG step and has to be successful to proceed in the FSM. Its error messages can be found in the X1SUPT logfile. XSETUP can also be called from X1MIST or directly on the FIC. The required data module *setup.data* is built automatically when called by X1MIST or X1SUPT. When called from X1SUPT, the data module is automatically deleted after the task terminates, opposed to X1MIST. Type *xsetup -?* for the list of options.

6.2 XRODBA

The source is contained in /os9/usr/levell/vme/source/xrodba.c .
XRODBA prepares the read out control module containing the the recipes to format BOS banks from the raw data as well as the hardware addresses used by X1PROD, X1FORM and X1WTCH. It is called in the CONFIG step and has to be successful before proceeding. XRODBA can also be called from X1MIST or directly on the FIC. Its error message can be found in the X1SUPT logfile. The required data module *rodba.data* is built automatically when called by X1MIST or X1SUPT. When called from X1SUPT, the data module is automatically deleted after the task terminates, opposed to X1MIST. Type *xrodba -?* for the list of options.

6.3 XHWTEST

The source is contained in /os9/usr/level1/vme/source/xhwtest.c .
XHWTEST is called by X1SUPT during the CONFIG step when a run is prepared. It performs several tests on the hardware: e.g. it looks whether the modules are accessible and reads the CSR0. If XHWTEST fails, ALEPH can not proceed in the FSM diagramm. The error messages of this task can be found in the X1SUPT logfile. XHWTEST can also be started in X1MIST or directly on the FIC. It needs the data module *hwtst.data* wich is prepared automatically when XHWTEST is called from X1MIST or X1SUPT. When called from X1SUPT, the data module is automatically deleted after the task terminates, as opposed to X1MIST. Type *xhwtest -?* for the list of options.

6.4 X1PROD

The source is contained in /os9/usr/level1/vme/source/X1prod.c .
The producer X1PROD reads out the trigger hardware modules in case of a level2 "yes" and writes the data to the raw event buffer on the FVSBI. This task, as well as X1FORM and the consumer task, are forked automatically (depending on the activity of the system and following the prescription in /os9/boot/tables/l1rp01_table.txt).

6.5 X1FORM

The source is contained in /os9/usr/level1/vme/source/X1form.c .
The formatter X1FORM reads the data from the raw event buffer and builds the trigger BOS banks. These banks are finally send to the MAEB.

6.6 X1WTCH

The source is contained in /os9/usr/level1/vme/source/X1wtch.c .
X1WTCH is another monitoring task and should run permanently. It reports the status of the DAQ and the trigger to the DISPLAY task, monitors the read out etc. and communicates extensively with X1SUPT on the VAX. X1WTCH is automatically forked at boot time.

6.7 XCBCHAIN

The source is contained in /os9/usr/level1/vme/source/xcbchain.c .
XCBCHAIN is called during a calibration run (see X1MIST → *ACCOMODATION* → *X1SUPT* to enable/disable and to set the number of cycles and the timeout). It performs a test on the discriminators and mixers in the analog crates as well as on the registers and on the scalers in the Fastbus crates. The strategy is as follows:

- Set the FET switches on the mixer cards to DAC input.

- Load the DACs and calculate the expectation.

- Now set the threshold of ALL lines successively in steps of 1 starting from 1 to 255 and read the registers on every sweep. (Also increment the s/w counters for the scalers – if enabled.)

- For the registers:
  Note the first and the last answer for each channel. There must be no gaps (see column 'IfGap' in table 6.1). If there is no answer at all for a particular channel, increment 'UnPlug' and note the minimum expected threshold for all sweeps in these cases: 'UnPlugMn'.
  If there are no 'UnPlug' and no 'IfGap' for a particular sweep, note the maximum difference of $|expected\_threshold - last\_answer|$ and sum up these differences to get the mean 'MeanDel'.
  If 'UnPlug' > 0 and 'UnPlugMn' ≤ 'MaxDel' ignore 'UnPlug': this may happen when the expected threshold is small ($< 3 - 5$) and may be hidden by offsets.

- There must be NO GAPS and NO UNPLUG. 'MaxDel' should not be greater than 2 for a single register and not be greater than 4 for the total registers. There must be no FB ERRORS or MISSING ACKNOWLEDGES.

```
→ HCW Thrs=3 Ntry=20
          Include TSS at f600000e
          UnPlug  UnPlugMn  IfGap  MaxDel  MeanDel
Reg( 6)     0         0       0       3        1  Single
Reg( 7)     0         0       0       3        1  Single
Reg(13)     0         0       0       3        2  Single
Reg(16)     0         0       0       3        1  Single
Reg(54)     0         0       0       3        2  Single
Reg(57)     0         0       0       3        2  Single
Reg(58)     0         0       0       3        2  Single
```

Table 6.1: Example of a CBCHAIN output

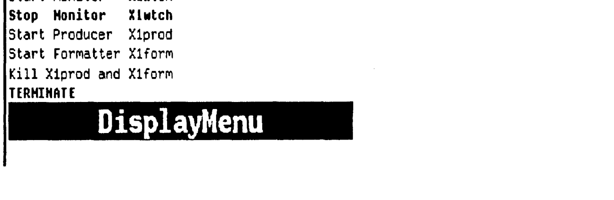
This procedure should detect broken or unplugged cables. XCBCHAIN needs a data module which is downloaded only in the CONFIG step of a physics run (**NOT** in the CALI-run). It can also be started from X1MIST or directly from the FIC. Its logfile is written to a_s$log:xlv1_calib.report and is also sent to the printer. The printout should be placed in the appropriate folder.

Chapter 7

Utilities on the FIC

7.1 DISPLAY

The source is contained in /os9/usr/level1/vme/source/display.c . DISPLAY should always be running on the FALCO terminal in the trigger corner of the control room. It provides a large menu to observe the trigger activities. When you are logged on the FIC just type *display* to start the task. Now the display menu appears on your screen (figure 7.1). Choose your page with the cursor and hit 'Return'. Available commands are written in bold rendition. To go back to the main menu hit any key.

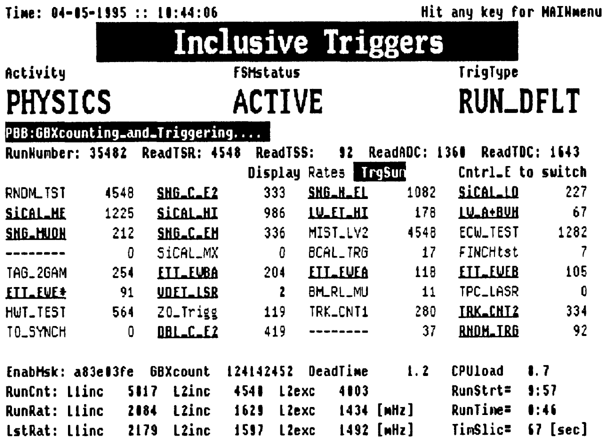


```
Time: 04-05-1995 :: 10:46:18          PBB:GBXcounting_and_Triggering....
              DisplayMenu

Display Global Trigger Status
TriggerActivity INclusive Triggers
TriggerActivity EXclusive Triggers
Display Messages,Warnings and Alarms
Display Readout DataBase
Display RCT (Readout Control Table)
Display SlaveReport (Report of SlaveTasks)
Display HVBits
Trigger TSReceiver ServiceRequest (Think Twice)
Fetch and Display Next Event
Start Monitor   X1wtch
Stop  Monitor   X1wtch
Start Producer  X1prod
Start Formatter X1form
Kill X1prod and X1form
TERMINATE
              DisplayMenu
```

Figure 7.1: The DISPLAY Main Menu

- Display Global Trigger Status
  Shows the global trigger status (self explaning)

- TriggerActivity Inclusive Triggers
  Besides some other useful information the page shows you how often a particular trigger has fired (figure 7.2). The enabled triggers are in bold letters; for disabled triggers the parasitic counts are displayed. With *Ctrl E* you can switch to the corresponding trigger rates. Wait to do so until the cursor is not moving.



```
Time: 04-05-1995 :: 10:44:06                        Hit any key for MAINmenu
                        Inclusive Triggers
Activity                   FSMstatus                  TrigType
PHYSICS                    ACTIVE                     RUN_DFLT
PBB:GBXcounting_and_Triggering....
RunNumber: 35482  ReadTSR: 4548  ReadTSS:   92  ReadADC: 1360  ReadTDC: 1643
                              Display Rates  TrgSum      Cntrl_E to switch
RNDM_TST      4548   SNG_C_E2      333   SNG_N_EL     1082   SiCAL_LO      227
SiCAL_ME      1225   SiCAL_HI      986   LW_EI_HI      178   LW_A+BVH       67
SNG_MUON       212   SNG_C_EM      336   MIST_LV2     4548   ECW_TEST     1282
--------         0   SiCAL_MX        0   BCAL_TRG       17   FINCHtst        7
TAG_2GAM       254   EIT_EWBA      204   EIT_EWEA      118   EIT_EWEB      105
EIT_EWE*        91   VDET_LSR        2   BM_RL_MU       11   TPC_LASR        0
HWT_TEST       564   ZO_Trigg      119   TRK_CNT1      280   TRK_CNT2      334
TO_SYNCH         0   DBL_C_E2      419   --------       37   RNDM_TRG       92

EnabMsk: a83e03fe  GBXcount  124142452  DeadTime    1.2  CPUload   0.7
RunCnt: L1inc   5017  L2inc   4540  L2exc   4003         RunStrt=  9:57
RunRat: L1inc   2084  L2inc   1629  L2exc   1434 [mHz]   RunTime=  0:46
LstRat: L1inc   2179  L2inc   1597  L2exc   1492 [mHz]   TimSlic=  67 [sec]
```

Figure 7.2: The Inclusive Triggers

- Trigger Activity Exclusive Triggers
  The same as above but for exclusive triggers.

- Display Messages,Warnings And Alarms
  Self explaining

- Display Readout Database
  Shows the enabled hardware components with their fastbus addresses and some further information. R/O errors are indicated in reverse video.

- Display RCT (Readout Control Table)
  Shows the content of the Run Control Table

- Display SlaveReport (Report of SlaveTasks)
  Shows the messages for the last active task (XHWTEST,XSETUP,XRODBA, XCBCHAIN).

- Display HVBits
  Self explaining. You'll find the definition of these bits in X1MIST (→ *ACCOMODATION* → *HVBits*)

- Trigger TSReceiver Service Request (Think Twice)
  This feature allows you to perform a standalone readout test. The system should be fully setup: (X1MIST → *TestTrig*). Then you have to make sure that the

Readout Control Table is present. If not, say *Load rct_0* outside DISPLAY. Enter *display* again. Start the formatter as well as the producer and then trigger a TS receiver service request. Do *Fetch And Display the Event* (see next item). **Don't use it during normal data taking !** (unless you want to make yourself famous..)

- Fetch And Display Next Event
  Shows you all the Trigger banks of the next event.

- Start/stop Monitor X1WTCH
  Self explaining! The request will be refused during data taking since X1WTCH and X1DAQ are talking to each other.

- Start Producer X1PROD
  Self explaining. This option is only enabled if the DAQ is OFFLINE

- Start Formatter X1FORM
  Self explaining. This option is only enabled if the DAQ is OFFLINE

- Kill X1PROD And X1FORM
  Self explaining. This option is only enabled if the DAQ is OFFLINE

- Terminate

7.2 RTEST

The source is contained in /os9/usr/level1/vme/source/Rtest.c .
RTEST is actually not a monitoring task but very helpful for testing the hardware. It gives you the opportunity to read or write to every control or data space register of every Fastbus module reachable from our main crate (F1-F6 via CI, others via SI). To run this program, you have to log on L1RP01 (the FIC) and type *rtest.* To see if you can reach a particular module you can use *scan* < *crate* > where < *crate* > stands for the address of the crate housing the desired module (F0 e.g. is the maincrate). RTEST then shows you the modules accessible in that crate. Now you have to specify the module you want to examine. Type *prim* < *crate* > < *slot* >. E.g. for the TSR you have to say *prim f0 b*, where the slotnumber 11 is given in hexadecimal notation. Now you can read (write to) any known CSR or DSR with *frc* (*fwc*) respectively *frd* (*fwd*). To write three times a 5 (in hex. numbers) into CSR0: *fwc 0 3 5* . Rtest can be very helpful during the investigation of a malfunction. **Note:** There is no interlock mechanism. Make sure you do not interfere with the DAQ; be careful when you are writing.

7.3 What to do after a Power Cut

Check the trigger power supplies of all analog, CAMAC and Fastbus crates and switch them on again if necessary. They are all marked with a green sticker. Sometimes the CAMAC crates housing the trigger logic can only be switched on with the circuit breaker on the rear side of the rack; don't forget the power supply in the rack which is in the middle of C2, opposite of the F0 crate, at the bottom. The last FB crate to be powered, should be F0. Initialize FB with *FBDB/act.* Boot the FIC. Run HWTEST, SETUP, RODBA and CBCHAIN (via X1MIST → *Test TrigTyp*).

Chapter 8

Trouble Shooting

"When it works, it works nicely."
Heard in the nine o'clock meeting

8.1 General hints

In the following section some hints for handling failures are given. Of course they are far from being complete and can only give some ideas and impressions how to act in case of a malfunction.

There isn't any strict procedure to follow when there is a failure in the trigger system but the first rule is:

DON'T PANIC!

The most important step to solve the problem is now to collect as much information as possible. Try to figure out if anything unusual or special has happened just before the error occurred. Have YOU touched anything in the trigger system before? Have there been any problems with the DAQ? Try to extract as much information as possible from the shift crew about the precise circumstances. Consult the logbook if necessary. Look out for hints in the presenter histograms and in the logfiles of your tasks. Think about which signals, cables and tasks may be related with the malfunction. Especially if more than one failure happens at the same time try to find out what they may have in common. Think also about the (not unlikely) possibility that there is no failure in Level1 but somewhere else, and insist that the corresponding subdetector coordinator is called if you believe that Level1 is working properly. If the malfunction prevents Aleph from running **a quick solution is desired**. Exchange a faulty card quickly and try to diagnose the precise problem later when there is more time. Do not hesitate to profit from the rich expertise present in Echenevex.

8.2 Follow up signals

8.2.1 Follow up signals

During an investigation of a malfunction it is often necessary to have all kind of signals (strobes etc.) available e.g. if you have to check the timing. If the DAQ is not running there is nevertheless the possibility to create all signals in the Level1 system. To generate all timing signals you need a start signal. Normally this is the *Lvl1_strt* given by the PBB after obtaining the GBX and RUN signal from the TS (*Trigger Superviser*). You can substitute this by the BX signal from the T0 Module which is always available. The BX arrives in the ECL Fanout in the Maincrate slot 23 on the first channel. Switch the corresponding little levers to use the BX as a Lvl1_strt. Don't forget that the system must be set up to make sure that the TSGs are programmed. Run SETUP in X1MIST if necessary. Now all timing signals and strobes are available. Do not forget to switch the BX away after you have finished your work. Otherwise HWTEST cannot run and will end with *'..error in reading pedestals'* because the TPR and the ADC's cannot be read while they are strobed.

8.2.2 Using RTEST

There is also the possibility to create all signals in software with Rtest writing to all the CSRs and DSR's e.g. of the PBB. Make yourself familiar with all relevant addresses on the single cards (see sections 7.2 and 3.0.3 for further information).

8.3 Fastbus Errors

8.3.1 Fastbus not operational

Look at the Fastbus display above the Main crate. With some experience the LEDs will show you the last action taken by the system. If the Fastbus system is stuck there might be a broken Fastbus coupler or fuse somewhere in the system. The problem is to find the responsible card.

The first thing to do now is to isolate the main crate. Therefore pull out the two CIs in F0 and use *crate=0* instead of *crate=F0* in RTEST. Power the crate, boot if necessary the FIC and check with RTEST if the Main crate alone is in an operational state. If not pull out one card after the other and check the status of the crate after each card. Remember that you cannot reach Fastbus from the FIC with a broken or without FVSBI.

If the Main crate is ok, you can check now the other crates. Connect the first branch

again. In every crate you will find a Cluster Interconnect card (CI), and on every CI there is a LED which indicates if the CI is the end of a Fastbus branch or an intermediate piece. Remove the crates now one after the other from the Fastbus system by setting the CI from the previous crate to END by switching over the little switch on the card just behind the LED (on the bottom right of the card). If you have found the faulty crate you have to identify the broken card. Connect the crate again to the Fastbus tree and pull out one card after the other until Fastbus can work again. Exchange the Fastbus coupler on the faulty card, or if this doesn't help exchange the complete card. Don't forget that also the CI itself can be faulty. Connect all crates again after you have finished.
Another device which may cause very unpleasant Fastbus problems is the Ancillary logic. There is only one in the Trigger system placed on the backplane of the Main crate. The Ancillary Logic controls the execution of the Arbitration Cycles, flags Geographical Addresses (on the remote crates this is done by the CIs), generates System handshakes for Broadcast operations etc. The Fastbus Display will give you hints if the Ancillary Logic is broken.
Last but not least it is necessary that the backplanes of all Fastbus crates are terminated properly; otherwise there may be strange signals on the bus.

8.3.2 XHWTEST

The task XHWTEST does some general checks in the configure step of the Finite State Machine. If it fails Aleph is prevented from running. So almost every problem reported by XHWTEST needs an action. Normally XHWTEST will give you the address and the name of the faulty card. Then you can check if this card is properly fixed in the crate, high voltage is on etc. Exchange the whole card if you think it is necessary.
If XHWTEST gives a very high return code (e.g. 992) it was not even possible to start the task. One reason may be that too many tasks are already running on the FIC. But most probably it is an AMS/Ethernet failure. Boot the FIC and hope the best.
Error messages concerning the PBB are often more complicated. It may happen that an error in the TS leaves the PBB with some signals on the FIO connector. These signals may spoil the answers XHWTEST expects from the PBB (*...unexpected number in CSR...*). Disconnect the connector and try again or check the connector directly with the scope. If you have reasons to think that this has caused your problem contact the DAQ coordinator, he/she has to reset the TS module. Otherwise test the PBB with RTEST or with PBBTST, a task specially written for this purpose, and exchange it if necessary.
If the problem is caused by an ADC or TSS and cannot be solved immediatly exclude the card from the read out and investigate later (see section 3.1.4).
If this all doesn't help and the data quality would still be acceptable, you have to change the return code in the corresponding setup tasks (XSETUP,XHWTEST) from *return(Qsev)* to *return(Qwrn)*. Do not forget to recompile the tasks.

8.3.3 High Voltage

An failure in the High Voltage is a very serious malfunction, because data taking is definitly corrupted. Check which crates are affected and switch them back on. If this is not possible check the fuses and the connectors of the power supply. Exchange the supply if necessary. There are enough spares available.
If the power supplies look okay the malfunction can also happen in the monitoring. Check the monitoring cables and follow them through the CAMAC crate below the HCAL analog crate to the HVbit TSR. The CAMAC signals and the Analog signals are 'anded'.

8.4 Missing or Junk Trigger

There may be several reasons for a malfunction in the trigger logic: false description of a trigger bit in the software, wrong cabling, broken cables or bad connectors in the CAMAC crates, broken cards (PLU, TPR ..) or maybe a timing problem.
Especially missing triggers are a serious problem because the trigger efficiency is affected. Junk triggers are annoying but not as serious if they don't create measurable downtime by forcing useless readouts. But be aware: the classification of junk or missing trigger is biased by the assumption that the software is always right. As usual the first thing to do is to get all necessary information to track down the problem.

8.4.1 Errors in the Trigger Verification

If XT1VER gives an alarm most of all information can be found in the XT1VER logfile. Here you can find all the read out data of the trigger system, especially the input signals of the PLU's and the TPR. Find out which trigger bits are affected and which detector signals are used for them. If you have done some cabling before or invented or modified a trigger there is an exellent chance that this action has caused the alarm. If the affected trigger bit is a new one check the logic in XTTL and the programming of the PLU's in X1mist *Edit resources* (see section 3.0.4 and G). If there haven't been any changes on the software side, study the XT1VER banks. Follow up the signals from the TSRs through the PLU system to the TPR and watch out for the point where the signal is lost or created unjustified. Maybe this is enough to find a broken cable, a bad connector or wrong polarity. There may also be a broken CAMAC card. Check if other signals on the same card are affected. Exchange the card if you think it is necessary. If everything looks allright in the PLUs there is still the possibility of a faulty TPR. Check the card with Rtest (you can always get some ECL signals from the Fanouts in F0).
There might also be a problem with the timing. Check if there is enough time between strobing the TSRs, the PLUs, the TPR and the PBB, especially if the transit time of any signal has been increased for any reason. Check if the necessary timing signals

arrive properly. The TDC statistics in X1TDD (see section 5.4) might be helpful here. If a signal on a PLU in overlap mode is not there in Level1 but in Level2 too tight a timing is very likely. Keep in mind that the difference between enabling the TPR and strobing the PPB must *at least* be 300 ns.

If none of these explanations apply, there is still the possibility of an error in the readout.

8.4.2 Timeout or Maximum Rate Exceeded

If there are no additional alarms from the verification task it is probably not a trigger problem but nevertheless you should react. Ask the shift crew about the beam conditions (high or low luminosity or background). Check if your error boundaries defined in X1MIST *Accomodations* are sensible.

Some trigger signals arrive directly on the PLUs (e.g. SICAL). Check with the help of the presenter histogram if they have arrived, and maybe have a look with the scope directly on the incoming cable. Insist that the corresponding subdetector coordinator will be called if you think that there is no problem on the trigger side.

If one of the first eight trigger bits is affected check if the downscaling is OK. Unfortunately it is not possible to read back the setting of the downscalers from the TPR so if you have reason to think that the problems are caused by a broken downscaler you have to check the TPR with RTEST.

If there are no triggers at all except of the Random trigger there may also something be wrong between TPR and PBB. Check that the PBB really strobes the TPR, and gets the trigger decision back. Check also the Enable mask from the TPR (read it back with RTEST).

8.4.3 Inefficiencies

An inefficiency will normally show up as a hole in one or more of the trigger histograms. Check e.g. if this hole appears in all thresholds and also in the TSS *and* ADCs. If this is true the problem might be due to a broken Mixer card but also might be not on the trigger side. Contact the corresponding subdetector coordinator if you think the trigger is working correctly.

Keep in mind that a TSS sees the same signals as its corresponding TSR because they share a Minibus on the backplane. If it seems to be a trigger problem you should check if the same holes appear in the TSS histograms. If this is not the case there may exist a readout problem for the TSR. If both histograms show the same shape check the cables to the TSRs. Are they properly connected? If one histogram is completely empty you should check if the TSR is strobed at the right time. If everything looks okay you might think of a broken discriminator, a broken Mixer card or a broken cable between the analog crate and the TSR. Run XCBCHAIN to find out or just put a signal on the line and read the TSR with RTEST. If only one threshhold is affected it can also be useful to swap the card with another to find out if the inefficiencies moves, too.

8.5 Where to Get Information if Something Went Wrong

- The PRESENTER
  The PRESENTER provides you with several histograms monitoring the running of Aleph and the trigger. It is an important tool to find malfunctions of the detector including the Level1 trigger system. It belongs to your daily duties to check your subdetector's (expert) histograms. The histograms for the shift crew are shown in figure 8.1.

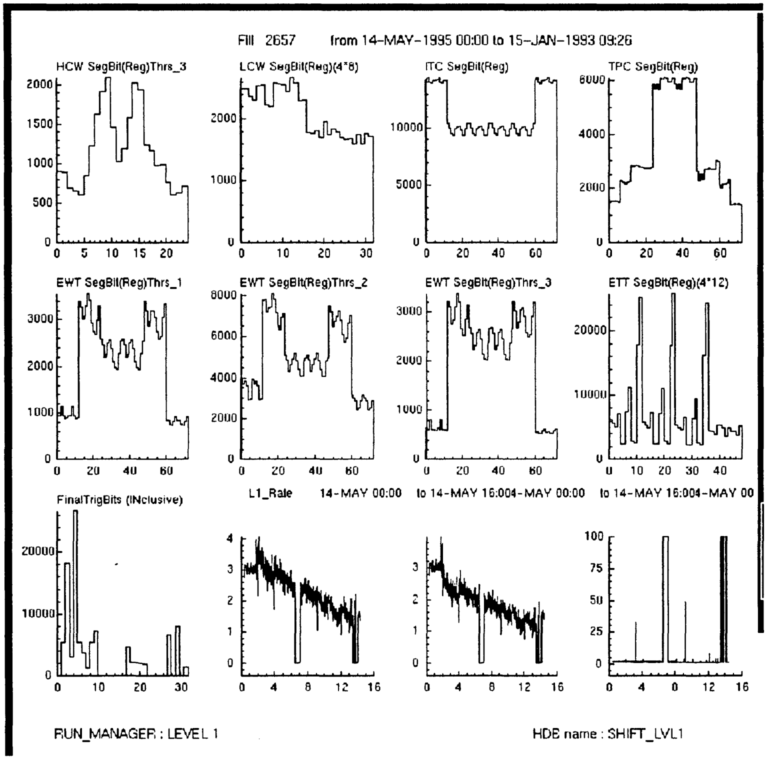


Figure 8.1: The LEVEL1 Histograms

A list of all histograms/timecharts is maintained in [Trigger.doc]presenter.info . You will find all necessary information about handling the presenter in the Subdetector Coordinator Manual or in the Data Manager Manual.

- The Error Logger
  The Error Logger saves all the error messages from the trigger tasks. Use ERRLOG or X1ERR to scan them.

- The General logbook
  What happend before the trigger error occured? Could it have had any effect?

8.6 OS9 for Pedestrians

Listed below are some OS9 commands you might need on the FIC:

cd *dir*	change to the directory *dir*
dir [-e]	print the contents of directory *dir*
go *dir*	go to directory *dir*
down *subdir*	go down to the subdirectory *subdir*
up	go up in the directory tree
mdir [-e]	list the module directory
list *file*	print the contents of a file on the screen
del *file*	delete *file*
procs [-e]	display all running processes
e *file*	edit *file*
	(press the 'Help' key for a list of editor commands)
'GOLD' q	quit the file
'GOLD' f	save and quit the file
ftp	to import files from the VAX

8.7 Spare Modules

Spare modules can be found in the spare crate above the CAMAC crates in barrack C2 and in the barrack No 6389 outside the pit. When changing a Fastbus card, note that the subprints can differ from card to card (TSR, ECL Fanout). Take care that the new card has the correct subprint. If you have powered the main crate, you have to reinitialize Fastbus. On the Vax type *FBDB/ACT*.

Chapter 9

Miscellaneous

9.1 Telephone Numbers

Some useful telephon numbers are:

C2:	7491
Shift:	8425 or 7461 or 8427
Trigger console:	7459

9.2 Useful Manuals

-The SD Coordinator Manual
-The Data Manager Manual
-The Online Manual

9.3 Trigger Banks

If you want to change a trigger BOS bank change the description in alws::aleph$general:[doc.bankdoc_new]bankx1.ddl and contact Françoise LOVERRE (JACOTF@VXCERN.CERN.CH).

Appendix A

The Directory Structure on the FIC

The default directory is */os9/usr/level1* with the subdirectories */eas* (Cosmolep) and */vme*. In .../*vme* you find the subdirectories: */cmds, /cmds_dbg, /data, /dev, /rels, /rels_dbg, /scripts, /source.* This branch, except of */dev*, contains the **pro** version of the resources, the **dev** versions can be found in */dev.* To compile and link a new version of a program, use the *mms 'file'* command in the directory */vme* for the **pro** version or */vme/dev* for the **dev** version. Using the *mms* command without a filename compiles and links all programs referenced in the file *makefile.* To load the new program, first use the command *unlink 'file'* and then *load os9/usr/level1/vme/cmds* (or .../*vme/dev/cmds*); alternatively you may boot the FIC.
To switch from the **pro** to the **dev** version for the data taking, edit */os9/usr/level1/vme/scripts/level1_start*, change the line

VERSION = "/os9/usr/level1/vme/cmds/"

to

VERSION = "/os9/usr/level1/vme/dev/cmds/"

and boot the FIC. There three possibilities to boot the FIC:

- Press the button.
- Type *reset* when you are logged on the FIC.
- Type *reset fic=l1rp01* on any other FIC connected via the VIC tree to the HCAL main crate (e.g. mats01 or maeb01).

Appendix B

The Directory Structure on the VAX

disk$user:

[Trigger.Mist]

pluaux..., x1daq..., x1library..., x1err..., x1mist..., xapack..., xauxil..., xbpack..., xdpack..., xhpack..., xrpack..., xspack..., xs_bos..., xs_mist..., xtpack...

[Trigger.Doc]

aldisp..., x1alert..., x1news..., x1prb..., x1trgd..., x1wtch..., presenter.info, xlerr.rules, xlv1_x1mon.hlp, xlv1_x1ver.hlp, xlv1_x1daq.hlp, xt1mon_0.hlp

[Trigger.Ver]

trgver..., x1anl..., x1chart..., x1mon..., x1pres..., x1rgdisp..., x1tddisp..., x1ver...

[Trigger.Dbase]

x1mist.db, x1news.db, x1prb.db

[Trigger.Calb]

thrfit..., x1calb..., x1runq..., x1sum...

[Trigger.Runq]

[Trigger.Runq.Export]

xlv1rq..., export.note

[Trigger.Runq.Import]

xlvlrq.bck

[Trigger.Data]

[Trigger.Eordata]

xlvlthr_xxxxx.native

[Trigger.Runsum]

xlrunq_....out

[Trigger.Runsumtest]

hetest..., rqtest..., rq_ecw..., rq_hcw..., rq_lcw..., xlvlrqtest...

[Triger.Vme]

xlform.c, xlprod.c, xlwtch.c, xhwtest.c, xrodba.c, xsetup.c, pbbtest.c, display.c, rtest.c

a_xldb$root:

a_xldb$root:[ddl]

xlglb..., xlglb_decl..., xlglb_struct...

a_xldb$root:[nodeb]

tg_get_ttlist.mms

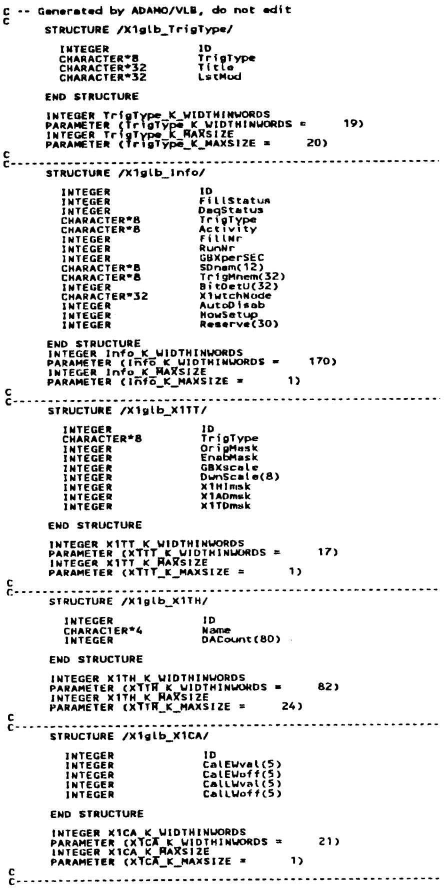
a_xldb$root:[source]

create_xl_global.for, read_hvb_reg.for, read_xlvl_data.for, tg_get_ttlist.for, xlodef.inc

The programs read_hvb_reg.for and read_xlvl_data.for are used by ALMON, whereas tg_get_ttlist.for supplies the DAQ with a list of possible trigger types.

Appendix C

The X1 Global Section



```
C -- Generated by ADAMO/VLB, do not edit
C
      STRUCTURE /X1glb_TrigType/

        INTEGER               ID
        CHARACTER*8           TrigType
        CHARACTER*32          Title
        CHARACTER*32          LstMod

      END STRUCTURE

      INTEGER TrigType_K_WIDTHINWORDS
      PARAMETER (TrigType_K_WIDTHINWORDS =      19)
      INTEGER TrigType_K_MAXSIZE
      PARAMETER (TrigType_K_MAXSIZE =      20)
C
C-----------------------------------------------------------------
      STRUCTURE /X1glb_Info/

        INTEGER               ID
        INTEGER               FillStatus
        INTEGER               DaqStatus
        CHARACTER*8           TrigType
        CHARACTER*8           Activity
        INTEGER               FillNr
        INTEGER               RunNr
        INTEGER               GBXperSEC
        CHARACTER*8           SDnam(12)
        CHARACTER*8           TrigMnem(32)
        INTEGER               BitDetU(32)
        CHARACTER*32          X1wtchNode
        INTEGER               AutoDisab
        INTEGER               HowSetup
        INTEGER               Reserve(30)

      END STRUCTURE
      INTEGER Info_K_WIDTHINWORDS
      PARAMETER (Info_K_WIDTHINWORDS =     170)
      INTEGER Info_K_MAXSIZE
      PARAMETER (Info_K_MAXSIZE =       1)
C
C-----------------------------------------------------------------
      STRUCTURE /X1glb_X1TT/

        INTEGER               ID
        CHARACTER*8           TrigType
        INTEGER               OrigMask
        INTEGER               EnabMask
        INTEGER               GBXscale
        INTEGER               DwnScale(8)
        INTEGER               X1HImsk
        INTEGER               X1ADmsk
        INTEGER               X1TDmsk

      END STRUCTURE

      INTEGER X1TT_K_WIDTHINWORDS
      PARAMETER (X1TT_K_WIDTHINWORDS =      17)
      INTEGER X1TT_K_MAXSIZE
      PARAMETER (X1TT_K_MAXSIZE =       1)
C
C-----------------------------------------------------------------
      STRUCTURE /X1glb_X1TH/

        INTEGER               ID
        CHARACTER*4           Name
        INTEGER               DACount(80)

      END STRUCTURE

      INTEGER X1TH_K_WIDTHINWORDS
      PARAMETER (X1TH_K_WIDTHINWORDS =      82)
      INTEGER X1TH_K_MAXSIZE
      PARAMETER (X1TH_K_MAXSIZE =      24)
C
C-----------------------------------------------------------------
      STRUCTURE /X1glb_X1CA/

        INTEGER               ID
        INTEGER               CalEWval(5)
        INTEGER               CalEWoff(5)
        INTEGER               CalLWval(5)
        INTEGER               CalLWoff(5)

      END STRUCTURE

      INTEGER X1CA_K_WIDTHINWORDS
      PARAMETER (X1CA_K_WIDTHINWORDS =      21)
      INTEGER X1CA_K_MAXSIZE
      PARAMETER (X1CA_K_MAXSIZE =       1)
C
C-----------------------------------------------------------------
```

Figure C.1: The X1 Global Section: a_x1db$root:[ddl]x1glb_struct.vlb

Appendix D

SOR Banks

D.1 X1CB

XLV1 SOR calibration constans. DACount → Energy=DAC*val+off.

1		# of words per row (=20)
2		# of rows=1
1-5	CE	CalEWval [MeV/cnt]: Segment,EA,EB,BA,TOT
6-10	AE	CalEWoff [MeV]: Segment,EA,EB,BA,TOT
11-15	LW	CalLWval [MeV/cnt]: Segment,EA,EB,TOT
16-20	WO	CalLWoff [MeV]: Segment,EA,EB,TH,TOT

D.2 X1DA

XLV1 SOR This TrigType threshold setting in DAC_counts.

1		# of words per row (=81)
2		# of rows
1	NA	Name Name HCWn, ECWn, LCW /n=1..4
2-81	DA	DACount [0,255] DACsetting for one Thrs. HCW 72 values. ECW 72+2*4 (Seg+Etot) LCW 4*(4+4+4) all 4 Thrs

D.3 X1TH

Trigger level1 thresholds.

1	# of words per row
2	# of rows
1 ID	ID
2-3 VR	ValRng [0,99999999] Validity range

4 TT A	ThrType
	Type of threshold
	ECTR = EcTowerEnergy
	ECWI = EcWireEnergy
	HCTR = HcTowerEnergy
	HCWI = HcWire
	LCTR = LcTowerEnergy
	ECTT = EcTotalTowerEnergy
	ECTW = EcTotalWireEnergy
	HCTT = HcTotalTowerEnergy

5-8	ThrVal [0,99999] Threshold Values

D.4 X1TT

XLV1 SOR. Current TrigType expanded.

1		# of words per row (=16)
2		# of rows
1-2	TT	TrigType current TrigType
3	OM	OrigMask Original TrigEnableMask
4	EM	EnablMask Current TrigEnableMask
5	GB	GBXscale DwnScale for RandomTrigger
6-13	DS	DwnScale DwnScale for first 8 Trigs
14	1H	x1HImsk ReadoutMask for TSS Scalers
15	1H	x1ADmsk ReadoutMask for ADC
16	1T	x1TDmsk ReadoutMask for TDC

D.5 XTBN

Trigger Bit Names of all 32 trigger bits.

1		# of words per row
2		# of rows
1	ID	ID
2-3	VR	ValRng [0,99999999] Validity range
4	TN	TrgdefiN [0,32] Trigger defiNed flag, if 0 the bit is undefined and not used for triggering.
5	TB	TrgBitnr [0,31] Number of trigger bit in the level 1/2 trigger bit mask.
6-7	BM	BitMnem Mnemonic name of this trigger bit
8-17	BN	BitName Full name of this trigger bit
18-22	AD	ActDet [HCAL,HCAL]... Name of active subdetectors needed to realize this trigger. This information is deduced from the XTLF bank.
23	L1	Lvl1trg Links a trigger bit with corresponding logical function
24	L2	Lvl2trg Links a trigger bit with corresponding logical function for the level 2 decision of this trigger

D.6 XTMS

Trigger Mask strings used in logical operations defined in bank XTTL.

1		# of words per row
2		# of rows
1	ID	ID
2-3	VR	ValRng [0,99999999] Validity range
4	MI	MaskId [0,30] Mask Identification, if 0 this row is undefined.
5	MN	MaskNam Bit Mask Name
6-8	MS	MaskStrng [-999999999,999999999] Mask bit String of 72bits

D.7 XTTL

Trigger Logical functions. Each row codes logical function which defines operations on a 72 bit string corresponding to the 72 bit wide pattern entering the ECLine modules of the trigger logic. The bit strings are either data from the fastbus register or predefined bit masks out of table XTMS. The result of a logic function is again a 72 bit string.

1	# of words per row
2	# of rows
1 ID	ID
2-3 VR	ValRng [0,99999999] Validity range
4 FI	FunlvlInd [0,2] Function level Index, if 0 this row is undefined, 1 for level one definition, 2 for trigger level two definition.
5-6 FN	FunNam Name of logical function, either PRxx if the result is a pretrigger or the name of a trigger the function is related to.
7 NO I	NumOper [0,5] Number of bit string operands
8 B1 A	Bitstrl Name of the bit string used in the logical function. Names starting with M are mask bit pattern contained in table XTMS, Names starting with PR are results from pretrigger logical functions, all other names refer to input signals.
9 L1 A	Logoperl Logic operator acting on the enclosing bit strings AND : A bitwise AND OR : A bitwise OR MAxx: A bit majority function is performed. If the number of bits in the left string is less than xx, the right hand side operand is inverted. ROxx: A bit rotation function is performed. The number of bits indicated by 2*xx is rotated by xx bit positions. Sxxx: A bit shift function is performed. The number of bits indicated by xxx is shifted xxx positions. If xxx is negative the shift is in right direction. DIxx: A bit discrimination function is performed. The righthand operator is interpreted as an integer value. If the value less than xx, the right hand side operand is inverted.
10 B2 A	Bitstr2 Bit string name (see Bitstrl).
11 L2 A	Logoper2 Logic Operator
12 B3 A	Bitstr3 Bit string name (see Bitstrl).
13 L3 A	Logoper3 Logic Operator
14 B4 A	Bitstr4 Bit string name (see Bitstrl).
15 L3 A	Logoper4 Logic Operator
16 B5 A	Bitstr5 Bit string name (see Bitstrl).

Appendix E

Readout Banks

E.1 X1DI

XLV1 ReadOut ADCs. Selective R/O.

1	# of words per row (=37)
2	# of rows
1 AN	AdcName Name HCW, ECW, LCW
2-37 CO	Cont 2 values(12 bits) in 1 word

E.2 X1ER

XLV1 ReadOut Hardware. ReadOut errors.

1	# of words per row (=4)
2	# of rows
1-3 PN	PortName 12 Bytes DeviceName
4 EC	ErrCode Normally FBrc: -1 if R/O permanently disabled 1,2,3 for PLU only: (No X-rsp,No Q-rsp, NoneAtAll)

E.3 X1HI

XLV1 ReadOut TSS_Scalers. History bank: selective R/O, usually only for RNDM trigger.

1	# of words per row (=31)
2	# of rows
1 NA	Name Name: see X1RG w/o PLU,TIM,TSR
2-31 CO	Cont 2 scalers packed in one word

E.4 X1RG

XLV1 ReadOut. Registers and related stuff.

1	# of words per row (=4)
2	# of rows
1 NA	Name Name of entry TIM 2*Time+TPR_EnabMask TSR GBXcount,{L1Ys,BunchNum},HV_bits TPR L1rslt,L2rslt,EvtMask PLU One Byte/sector; Four sect/word ITC Bits {24-35}->{36-47} TPC Bits {24-35}->{36-47} LCW squeezed; 4*{2*4} Bits ETT 3*12 Bits {ECW_o,ECW_e,LCW} HCWn no manipulation wrt TSR ECWn no manipulation wrt TSR
2-4 CO	Cont Data

E.5 X1TD

XLV1 ReadOut TDC-info formatted. Selective R/O (see figure 5.3).

1	# of words per row (=1)
2	# of rows
1 CO	Cont 7 bits channel [0,95] 9 bits StrtTime [0,511] 9 bits StopTime [0,511] 2 bits ClkSelec [0,3]=[2,16]ns 5 bits ErrCode [0,2] 0=ok, Matching Srt/Stop 1=STRTcha w/o StopCha 2=STOPcha w/o StrtCha

Appendix F

EOR Banks

F.1 X1SM

XLV1 EOR. Summary from X1MON_Online_Monitoring.

1	# of words per row (=1)
2	# of rows
1 MD	MonDat Data

F.2 X1SV

XLV1 EOR. Summary from X1VER_Online_Monitoring. Picked up by the Run Quality.

1	# of words per row (=1)
2	# of rows
1 VD	VerDat Data

Appendix G

XTTL

G.1 Signal Names

HCWn	24	bits {6+12+6} projected from TSR readout
HWTn	72	bits {Segments after Mixer} from TSR readout
ECWn	36	bits {ANDed odd/even pairs}
EWTn	72	bits {ECW mapped onto TriggerSegments}
LCW	16	bits {4 thresholds * 4 signals}
PLUn	16	bits each; 2*8 bits {sector_0/sector_1} PLU1-PLU6 PLU1=geo_8; PLU2=geo_7; PLU3=geo_18; PLU4=geo_17
ITCT	72	bits ITC segments
TPCT	72	bits TPC segments
ETEW	12	bits {3 thresholds * 4 signals [EA,EB,BA,Sum]}
ETLW	12	bits {3 thresholds * 4 signals [A,B,Sum1,Sum2]}

G.2 XTTL Rules

All operators are binary ones; a unary operator has to be supplemented by a dummy 2nd operand; Equation:

$$operand\ OPERATOR\ operand\ OPERATOR\ operand...$$

is evaluated as

$$\{\{\{operand\ OPERATOR\ operand\}\ OPERATOR\ operand\}\ ...\}\ .$$

Operands are the SIGNALS defined above; they are internally expanded to 72 bits; MASKS (defined in XTMS) are also recognized as Operands if they start with 'M' as first character.
OPERATORS are:

AND	binary AND	binary
OR	binary OR	binary
EXOR	binary EXCLUSIVE OR	unary
MAnn	MAJORITY (Sum[72bits].ge.nn)	unary
DInn	DISCRIM first 8 bits interpreted as integer m (m.ge.nn)	unary
MLnn	MoveLeft Multiply by 2**nn	unary
MRnn	MoveRight Divide by 2**nn	unary

Example definition in XTTL:

1 SiCAL_LO PLU4 MR14 M003 MA01 MIST

The first row indicates the Trigger_Level for which this trigger has be be evaluated; If there is only a definition with 1, but not 2, then it is implicitly assumed, that the Level_1 and Level_2 definition are identical;
The second row is the name of the trigger as defined in XTBN (this name is case-sensitive).
The next rows define the trigger-bit and how to evaluate its value. Take the contents of the fourth PLU, MoveRight 14 bits and mask the first two bits:

PLU4= ..xxxxxnniiiiiiiiiiiiii

after MR14 we have

..xxxxxnn

after M003 we have:

0000000nn .

The next operation is:

0000000nn MA01 MIST .

We count the bits in 000000nn → m=0,1,2 and test if it is .ge.1; if YES the result is 000000000001, if NO the result is 000000000000 (expanded to 72 bits) to ease the description, on can define PreTriggers: PRnn where nn=[00,99]; these PreTriggers can then be used as Operands. One may even use PreTriggers to define other PreTriggers but here only backward references are allowed. If a trigger can/shall not be tested (e.g. the random trigger) insert *????* instead of the signal name: this signal will not be checked.

Appendix H

Xlv1_X1DAQ.hlp

- TRIGHVB
  At least one of the Trigger_LowVoltageBits is off (CAMAC or ANALOG). THESE BITS MUST BE ON ALWAYS! Checked by FIC.X1wtch on every event or directly from the HVBreg.
  !*!Serious problem. During DataTaking Data is definitely corrupted.
  !*!Call TriggerExpert. During DataTaking: Try to fix the problem
  !*!immediately by switching on the crates in C2 and restart the run.

- TPRERRS
  An error was detected in the data read from the TPR (Trigger_Pattern_Register). Checked by FIC.X1wtch on every event.
  -TPR_fault; Empty Mask <L1mask> <L2mask> <TrigMask> one of the three masks was found to be zero. <TrigMask>==0 may happen if the EnableMask was cleared. Would normally never result in a Trigger; But usually the RANDOM_TRIGGER is forced in the PBB (Trigger_TSR_Piggy_Back_Board).
  -TPR_fault; DwnScalers <L1mask> <L2mask> the first eight bits of the two masks are different which indicates an hardware error.
  -TPR_fault; MaskingErr <L2mask> <EnabMask> <TrigMask>
  Iand(L2mask,EnabMask)==TrigMask was not fulfilled.
  !*!During DataTaking call TriggerExpert.

- MSKCHNG
  The Trigger_Enable_Mask (read from the Trigger_TPR module) has changed and is different from the value loaded into the hardware during setup. This might happen if the TrigType was SABOTAGEd. Checked by FIC.X1wtch on every event.
  !*!This must not happen during DataTaking. ->Call TriggerExpert.

- TRGWAIT
  PBB (Trigger_TSR_Piggy_Back_Board) is counting GBX but there are no triggers for at least 30 seconds or so. Checked by FIC.X1wtch by directly polling the PBB.
  !*!Indicates a serious problem during DataTaking.

!*!Call TriggerExpert unless there is an obvious explanation
!*!like someone is fighting DAQ_errors.

- GBXWAIT
  PBB (Trigger_TSR_Piggy_Back_Board) did not see GBX for quite some time, although the DAQ is ready and all relevant HVBits are on. The principal symptom is NO TRIGGERS.
  !*!Either there is a DAQ error, e.g. TSCON got stuck, or
  !*!the DAQ-operator is fighting a DAQerr or
  !*!the trigger is disabled unintentionally or
  !*!Trigger-Level1 has a HardWare-problem (very improbable).
  !*!Call TriggerExpert only, if you have evidence that the
  !*!Level1 hardware is faulty.

- PBBDISA
  PBB (Trigger_TSR_Piggy_Back_Board) is DISABLED although the Trigger is ready to accept GBX. Checked by FIC.X1wtch by directly polling the PBB.
  !*!Serious problem. Call TriggerExpert. In the meantime reset the
  !*!run and restart. This might cure the problem.

- MODTRTP
  This is an informal message indicating that the selected TrigType was automatically modified by the TriggerController X1daq. This may happen if the appropriate option (EnAutoDisab in X1mist) has been enabled AND A SUBDETECTOR EXPECTED TO PROVIDE SIGNALS FOR A PARTICULAR TRIGGER IS OUT OF THE PARTITION -> This particular TriggerBit will then be disabled.
  !*!Call TriggerExpert if this explanation does not apply.

- SABOTAG
  This is an informal message indicating that the TrigType currently in use was modified by the DAQ_operator.
  !*!This is acceptable only if we are not taking data.

- FICWTCH
  The TriggerController X1daq cannot communicate with the Watchdog FIC.X1wtch running on the micro. X1daq was not able to fork it. Try to fork it with X1mist.
  !*!NOT a serious problem; BUT FICmessages cannot be forwarded to
  !*!the ErrorLogger; Call TriggerExpert during DayTime or DataTaking.

- TRBIT
  TriggerBit nn ('TRBITnn') is in ALARM: the reasons may be –Timeout: this trigger was not seen for too long a period –EXclusive TriggerRate too high – INclusive TriggerRate too high This message originates from FIC.X1wtch running on the micro. Maybe the thresholds are not well adjusted. If so change them with X1mist–>Accomodation–>X1mon_option and send the new Options to the

micro.
!*!Potentially serious problem: During DataTaking call TriggerExpert
!*!unless there is an obvious explanation.

- ROBLK
  Permanent ReadOut_Error detected for some Trigger_H/W_module. Readout of that particular module is automatically inhibited until the H/W is reinitialized (Config,Init). Message sent by FIC.X1prod via FIC.X1wtch; The nn in 'ROBLKnn' is a sequential number and has no meaning.
  !*!During DataTaking this is a serious problem IF THE MODULE BELONGS
  !*!TO ONE OF THE CLASSES: PBB TPR TSR PLU ->Call TriggerExpert.
  !*!If the module belongs to: ADC TDC TSS -> NOT terribly serious,
  !*!ExpertCall only during DayTime.

Appendix I

Xlv1_X1MON.hlp

- BNK_X1RG
  The Bank X1RG (Register_Masks) is missing. This is the only TriggerBank that must be present for every event.
  !*!If the error persists Call TriggerExpert.

- MSK_CHNG
  The TriggerEnableMask as read from the TriggerPatternRegister(TPR) has changed during a run. This may happen if this run was SABOTAGEd (should not happen during DataTaking) or if the hardware fails (mask may have been cleared/overwritten).
  !*!If you suspect a H/W problem Call TriggerExpert.

- LOW_VOLT
  The data read from the HighVoltageBit register indicates that one or both of the LowVoltageBits of the Trigger (CAMAC or ANALOG) is OFF. This means that the data taken are useless.
  !*!Call TriggerExpert.

- MODDISAB
  The ReadOut of at least one module in the TriggerSystem has been disabled due to five Fastbus errors w/o an intermediate success.
  !*!Call TriggerExpert.

- MASKERR
  This warning indicates a fault of the TriggerPatternRegister(TPR). The accompanying text says: Mask .... TPR .... where Mask is the EnableMask read from the TPR and the three words after TPR are Lvl1result, Lvl2result and the FinalTriggerMask
  !*!If the error persists Call TriggerExpert.

- MISMTCH
  It was found that the TriggerMask sent from the Trigger_TPR to the TriggerSupervisor via a cable do not compare. The message says "TrigMismatch:

Sent/Rcv ...."
!*!If the error persists DURING DATATAKING Call TriggerExpert.

- LVL_SWI
The (usually disabled) TriggerBit MIST_LV2 (Bit_10) is used to check the switching from Levell to Level2 in the TriggerSystem. This has failed and indicates a H/W and/or R/O problem, most probably to be located in the TPR (F0.F) module.
!*!If the error persists DURING DATATAKING Call TriggerExpert.
!*!If the error shows up spuriously Call the TriggerExpert during daytime (we can live with this). In any case make sure that this error is reported to the TriggerExpert.

Appendix J

Xlv1_X1VER.hlp

- MISS_TRIG_
  The Trigger_Verification_Program found a particular TrigBit ON in SoftWare but OFF in HardWare. This is a cabling problem or a faulty description of the TriggerBit. The message will be reset after 300 events without errors. Look at Presenter/XLV1_ExpertPage/"TDC and TrgVer" to find out if the error is a permanent one.
  !*!This could seriously affect the TriggerEfficiency.
  !*!During DataTaking call TriggerExpert immediately
  !*! IF THIS ERROR IS A PERMANENT ONE or during DayTime.

- JUNK_TRIG_
  The Trigger_Verification_Program found a particular TrigBit OFF in SoftWare but ON in HardWare. This is a cabling problem or a faulty description of the TriggerBit. The message will be reset after 150 events without errors. Look at Presenter/XLV1_ExpertPage/"TDC and TrgVer" to find out if the error is a permanent one.
  !*!Not terribly serious; Inform TriggerExpert during DayTime.

- X1V_NOSOR
  The Trigger_Verification_Program has to read some data from a SOR_file. This was not possible for any reason. No TrigVer will be done. One should try to write this file with X1mist->Test_TrigType and then reread the SOR from the X1ver MainMenuPage.
  !*!During DataTaking call TriggerExpert.

- X1RG_MISSING
  An event was found without the bank X1RG: This bank is essential for the TriggerVerification and also for XLUMOK.
  !*!If this is a permanent error call TriggerExpert.

- TDC_MISS_
  A TimingSignal was expected but was missing in bank X1TD. Could be a serious

problem or a faulty descriptor:
!*!During DataTaking call TriggerExpert if the error is permanent.

- TDC_NODF_
  A TimingSignal was seen in the bank X1TD which is not described. Obviously the SignalDescriptor is faulty.
  !*!Not serious; Inform TriggerExpert during DayTime.

- TDC_NEND_
  An imcomplete TimingSignal was found in bank X1TD. (Start/No_End) A GatingProblem or a HardWare problem or a FormattingProblem.
  !*!Not serious; Inform TriggerExpert during DayTime.

- TDC_NSTR_
  An imcomplete TimingSignal was found in bank X1TD. (No_Start/End) A GatingProblem or a HardWare problem or a FormattingProblem.
  !*!Not serious; Inform TriggerExpert during DayTime.

- TDC_SORT_
  The TimingSignals are measured relative to a StartingSignal(GBX). Now it appears that one signal was earlier than GBX.
  !*!Could be serious; During DataTaking call TriggerExpert.

- TDC_WIDE_
  Analysis showed that a particular TimingSignal has too large a jitter in the delay w.r.t GBX. This may spoil the calibration.
  !*!Could be serious; During DataTaking call TriggerExpert.

- TDC_LONG_
  A TimingSignal was found that exhibits too large a variation in the delay w.r.t the StartingSignal(GBX). Should be investigated.
  !*!Not terribly serious; During DataTaking call TriggerExpert.

- TDC_LSIG_
  A TimingSignal was found with too large a variation in the SignalLength. Should be investigated.
  !*!Not terribly serious; During DataTaking call TriggerExpert.

Appendix K

Xlv1_X1ANL.hlp

- ITC_HILO
  High or Low channel found in distribution of ITC-Segment_bits. The deviation is more than 5 sigma w.r.t. mean.
  !*!Check the histogram(s) with the PRESENTER.
  !*!Find out yourself if this needs an intervention ITC/LEVEL1.
  !*!ALMOST CERTAINLY NOT A LEVEL1-PROBLEM. Just a hint.

- ITC_TRAN
  The ITC-Histo does not exhibit the expected repetitive structure. Cable or Readout-problem may be suspected.
  !*!THE PROBLEM IS MOST PROBABLY WITH THE LEVEL1-HARDWARE.

- TPC_HILO
  High or Low channel found in distribution of TPC-Segment_bits. The deviation is more than 5 sigma w.r.t. mean.
  !*!Check the histogram(s) with the PRESENTER.
  !*!Find out yourself if this needs an intervention TPC/LEVEL1.
  !*!ALMOST CERTAINLY NOT A LEVEL1-PROBLEM. Just a hint.

- TPC_TRAN
  The TPC-Histo does not exhibit the expected repetitive structure. Cable or Readout-problem may be suspected.
  !*!THE PROBLEM IS MOST PROBABLY WITH THE LEVEL1-HARDWARE.

- TPC_JUMP
  The TPC-Histo has a marked discontinuity.
  !*!Check the histogram(s) with the PRESENTER.
  !*!Find out yourself if this needs an intervention TPC/LEVEL1.
  !*!ALMOST CERTAINLY NOT A LEVEL1-PROBLEM. Just a hint.

- LCW_HILO
  High or Low channel found in distribution of LCW-Segment_bits. Might be

explained by dirty/asymmetric beam conditions.
!*!Check the histogram(s) with the PRESENTER.
!*!Find out yourself if this needs an intervention LCAL/LEVEL1.
!*!ALMOST CERTAINLY NOT A LEVEL1-PROBLEM. Just a hint.

- LCW_ASYM
  A large odd-even-asymmetry has been found. Might happen at times of high synchrotron-radiation.
  !*!Check the histogram(s) with the PRESENTER.
  !*!(the series is odd-even-odd-even...)
  !*!ALMOST CERTAINLY NOT A LEVEL1-PROBLEM. Just a hint.

- ECW_HILO
  High or Low channel found in distribution of ECW-Segment_bits. Might be explained by dirty/asymmetric beam conditions.
  !*!Check the histogram(s) with the PRESENTER.
  !*!Find out yourself if this needs an intervention ECAL/LEVEL1.
  !*!ALMOST CERTAINLY NOT A LEVEL1-PROBLEM. Just a hint.

- ECW_ASYM
  A large odd-even-asymmetry has been found. Might happen at times of high synchrotron-radiation.
  !*!Check the histogram(s) with the PRESENTER.
  !*!(the series is odd-even-odd-even...)
  !*!ALMOST CERTAINLY NOT A LEVEL1-PROBLEM. Just a hint.

- HCW_HILO
  High or Low channel found in distribution of HCW-Segment_bits. Keep in mind that HCAL behaviour is not predictable in general. Contact HCAL if there are channels with very low content.
  !*!Check the histogram(s) with the PRESENTER.
  !*!Find out yourself if this needs an intervention HCAL/LEVEL1.
  !*!ALMOST CERTAINLY NOT A LEVEL1-PROBLEM. Just a hint.

Appendix L

Help for Level1 Histograms

- LVL1_HCW3

  Histogram shows response of the 24 HCAL Wires Modules after applying the third threshold (at least four double-planes fired) which is used for the SNG_MUON trigger.

A)	channel	0- 5	EndCap_A
B)	channel	6-17	Barrel
C)	channel	18-23	EndCap_B

  The signals used to form the coincidence with the ITC is LVL1_HWT (see LVL1 expert pages) which requires some mixing (24–>72)

1)	channel	0- 5	outer EndCap_A	unmixed	A)
2)	channel	6-11	inner EndCap_A	unmixed	A)
3)	channel	12-23	Overlap_A	mixed	A)+B)
4)	channel	24-35	Barrel_A	unmixed	B)
5)	channel	36-47	Barrel_B	unmixed	B)
6)	channel	48-59	Overlap_B	mixed	B)+C)
7)	channel	60-65	inner EndCap_B	unmixed	C)
8)	channel	66-71	outer EndCap_B	unmixed	C)

- LVL1_LCW

  Histogram shows response of LCAL Wires signals after applying four different thresholds.

1)	channel	0- 7	Thrs_1
2)	channel	8-15	Thrs_2
3)	channel	16-23	Thrs_3
4)	channel	24-31	Thrs_4

  Each group of eigth is subdivided as follows:

1) ODD WireSum HalfModule_SideA_outside
2) EVEN WireSum HalfModule_SideA_outside
3) ODD WireSum HalfModule_SideA_inside(=closer to LEP_centre)
4) EVEN WireSum HalfModule_SideA_inside
5) ODD WireSum HalfModule_SideB_inside
6) EVEN WireSum HalfModule_SideB_inside
7) ODD WireSum HalfModule_SideB_outside
8) EVEN WireSum HalfModule_SideB_outside

Every two adjacent odd/even signals are ANDED; we then have four signals per threshold. The OR of the four signals of the first threshold is used as a veto for the LW_A+BVH (AntiSpark)

- LVL1_ITC
  Histogram shows distribution of 60 SegmentBits, derived in the Level1 processor of the ITC and sent to the Trigger. The 60 Segments are expanded to 72 to match those of other SubDetectors (channels 36-47 are a copy of channels 24-35).

  1) channels 0-11) TWO Phi_rings of 6 (EndCap_A)
  2) channels 12-59) FOUR Phi_rings of 12 (Barrel)
  3) channels 60-71) TWO Phi_rings of 6 (EndCap_B)

  1) and 3) should have the same height, while 2) is significantly lower. Depending on Beam_conditions you might notice a more or less pronounced sinusdoial structure overlaid onto the flat parts. Since the histogram is only filled for Level2_triggers it may be biased by ECAL or HCAL (if coincidences are required).

- LVL1_TPC
  Histogram shows distribution of 60 SegmentBits, derived in the Level2 processor of the TPC and sent to the Trigger. The 60 Segments are expanded to 72 to match those of other SubDetectors (channels 36-47 are a copy of channels 24-35).

  1) channels 0-11) TWO Phi_rings of 6 (TPC inner zone)
  2) channels 12-23) ONE Phi_rings of 12 (TPC inner zone)
  3) channels 24-47) TWO Phi_rings of 12 (TPC outer zone)
  4) channels 48-59) ONE Phi_rings of 12 (TPC inner zone)
  5) channels 60-71) TWO Phi_rings of 6 (TPC inner zone)

  3) is significantly higher than 1),2) and 4),5). The central part should be flat while the outer parts resemble a slowly falling ramp (low_theta tracks cross less TriggerPads on the inner TPC_sectors). Depending on Beam_conditions you might notice a more or less pronounced sinusdoial structure overlaid onto the three parts. Since the histogram is only filled for Level2_triggers it may be biased by ITC (only very slightly, since the TPC_Level2_update is currently only used for trigger TRK_CNT2)

- LVL1_EWT
  There are 72 signals from the ECAL Wires modules, two for each of the 36 H/W modules (the ODD wire-sum and the EVEN wire-sum). These sets of signals are called ECW and can be found on LVL1 expert pages. After ANDING the

odd/even signals to suppress incoherent noise we are left with 36 signals that have to be mapped digitally onto 72 TriggerSegments (to e.g. form coincidences with the ITC). These sets of signals are called EWT (ECAL wire mapped onto TOWER).

Threshold_1 used for: SNG_C_EM
Threshold_2 used for: SNG_C_E2
Threshold_3 used for: SNG_N_EL

- LVL1_ETT
  Total Energy; The histogram shows four groups of twelve signals each.

  1) ECAL Wires ODD sums
  the twelve signals are subdivided into three groups (different thresholds) of four signals each which correspond to 1.00) EndCap_A
  1.01) EndCap_B
  1.02) Barrel
  1.03) Total Sum
  1.04)-1.07) as before, for 2nd threshold
  1.08)-1.11) as before, for 3rd threshold

  2) ECAL Wires EVEN sums
  2.00) Endcap_A
  ...and so on and so forth ad nauseam

  3) ECAL Wires after ODD/EVEN coincidence
  3.00) AND of 1.00) and 2.00)
  ...and so on...
  It is these ANDED odd/even coincidences which are used in the total energy triggers.

  4) Twelve LCAL wires signal; Three groups(thresholds) of four signals each
  0) Sum over Side_A
  1) Sum over Side_B
  2) Sum over both sides (half gain)
  3) Sum over both sides (full gain)
  Unlike the ECAL signals the LCAL signals for the total energy are not treated separately in the odd/even parts; they are simply added before the discrimination.

  Used for:
  ETT_EWBA Thrs_1 channel_26
  ETT_EWEA Thrs_3 channel_28
  ETT_EWEB Thrs_3 channel_29
  ETT_EWE* Thrs_1 channel_24 and _25
  LW_ET_HI Thrs_2 channel_40 and _41 and _43
  LW_A+BVH Thrs_3 channel_44 or _45 (vetoed by AntiSpark)

- LVL1_TRGBITS
  Histogram shows the distribution of the INCLUSIVE triggers after the TriggerEnableMask has been applied. Bits 00-07 are scaled down.

- LVL1_RATE
  If this TimeChart reminds you of some demon, beep the Pope of Rome or write to the 'Journal for Irreproducible Results'.

- LVL1_DOWN
  If this TimeChart reminds you of some demon, beep the Pope of Rome or write to the 'Journal for Irreproducible Results'. Note: DownTime is defined (here) as REALtime minus GBXtime.