EUDET-Memo-2010-27

JRA1 Telescope: NI Flex RIO DAQ

DAQ emulator software overview

Gilles Claus[1], Mathieu Goffe[1], Kimmo Jaaskelainen[1], Cayetano Santos[1], Matthieu Specht[1]

January 17, 2011

Abstract

The EUDET JRA1 Pixel Telescope is using a custom-made data acquisition system since a couple of years. In preparation for AIDA, the group decided to investigate different off the shelf I/O systems. The advantage of such a system is the easier support and the availability over the next years. The IPHC group selected the NI Flex Rio system and prepared LabView sources, which can rather easy be connected to the existing DAQ. In this memo describes the DAQ emulator software which can be used for DAQ development without the hardware.

[1] IPHC, Strasbourg, France

Table of Contents

1 Introduction

The DAQ emulator software is a windows application which emulates the data stream produced by a telescope equipped with six Mimosa 26, read by the Flex RIO board. It has been developed with Borland C++ Builder 6 IDE, therefore you need this software suite to compile the project.

Its main goal is to have a tool to test the Flex RIO lib (eudet_frio) without the need to run all the hardware : Mimosa 26, JTAG control, Flex RIO board and DAQ. Because; otherwise debugging will quickly become painful, due to the number of things we need to handle.

Moreover, our EUDET collaborators will have to implement interface from EUDET DAQ to Flex RIO DAQ via Ethernet in this library. It will be easier for them to work with an emulator rather than with the whole DAQ chain (HW & SW).

We should also keep in mind that the DAQ application is Labview and it uses the eudet_frio lib embedded in a DLL, therefore it will complicate debugging. Especially if the software crashed … Because we don’t have an integrated debugger for Labview and the DLL (one day I must take the time to have a close look to LabWindows CVi ;-).

The DAQ emulator also helps in this case, because it doesn’t need to see eudet_frio library as a DLL. Library source files can be included in application and therefore we can use the debugger : inspect variable, set breakpoints and so on.

But please don’t ask me how to use Borland debugger, I never use it … I have my own set of macros to log what happens in source file and it’s enough in case of problem.

What this DAQ emulator can do ?

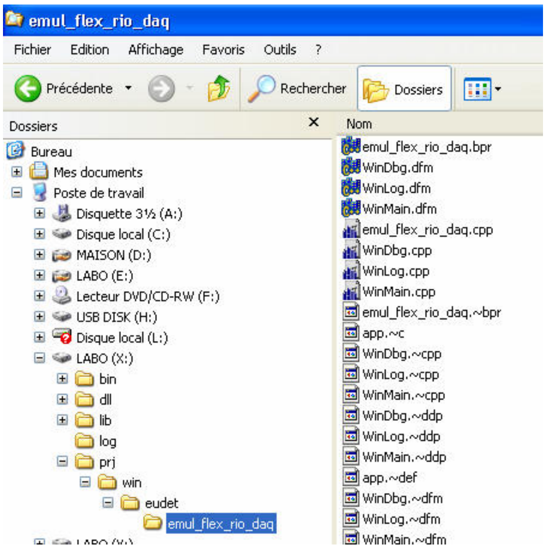
- It can emulate one or six Mimosa 26
- It has four modes of board " readout " emulation
  - IPHC system → Data format for compatibility with our previous DAQ
  - EUDET 1 mode → Acquire all frames & but doesn't store TLU triggers It stores only first three triggers → like in the IPHC
  - EUDET 2 → Acquire all frames & store TLU triggers (up to 288 / frame)
  - EUDET 3 → Acquire only frames with trigger & store TLU triggers

There is only one mode useful for EUDET collaboration → mode EUDET 3. The others modes had been developed to test the software step by step and to compare execution times. Because it was important to evaluate the execution time of frames with trigger extraction by software, and to do this we need to know execution time without this processing.

- The values of Mimosa 26 frame " relevant fields " are configurable from GUI
  - Header
  - Data length
  - Trailer
- The frame counter is incremented automatically, two modes are available
  - By default, incrementation starts at acquisition beginning → 0 .. 1799
  - It can be modified (source code) to increment from beginning of run
- The data size can be configured as
  - Fixed value hard coded in emulation function
  - Random value
  - Maximum value on first Mimosa 26, others hard coded in emulation function
- The data part of the frame contains 0, but user can modify the emulation function code to set any other value.
- Triggers (TLU & Flex RIO) are also configurable
  - **The number of trigger to emulate per frame or each N frames**
  - **The first three triggers + the last one are configurable from GUI**
  - **The others triggers are hard coded to 0 in emulation function**
- **Save run to disk, load run from disk, scan run & display frames " relevant fields "**

2 How to compile the software

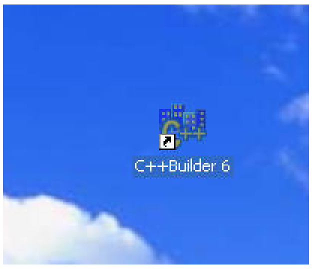
The application source code is in directory x:\prj\win\eudet\emul_flex_rio_daq.



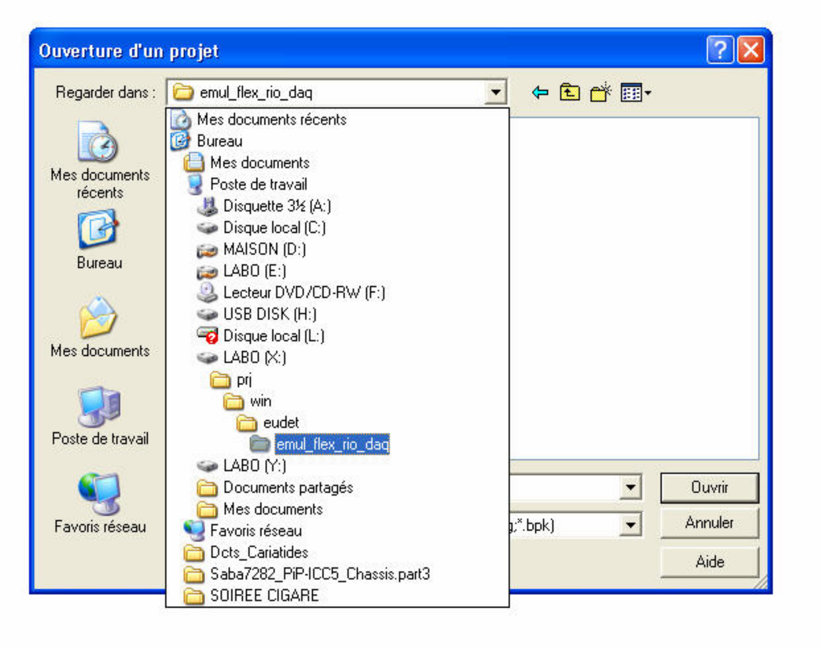
```
Bureau
  Mes documents
  Poste de travail
    Disquette 3½ (A:)
    Disque local (C:)
    MAISON (D:)
    LABO (E:)
    Lecteur DVD/CD-RW (F:)
    USB DISK (H:)
    Disque local (L:)
    LABO (X:)
      bin
      dll
      lib
      log
      prj
        win
          eudet
            emul_flex_rio_daq
```

```
emul_flex_rio_daq.bpr
WinDbg.dfm
WinLog.dfm
WinMain.dfm
emul_flex_rio_daq.cpp
WinDbg.cpp
WinLog.cpp
WinMain.cpp
emul_flex_rio_daq.~bpr
app.~c
WinDbg.~cpp
WinLog.~cpp
WinMain.~cpp
WinDbg.~ddp
WinLog.~ddp
WinMain.~ddp
app.~def
WinDbg.~dfm
WinLog.~dfm
WinMain.~dfm
```

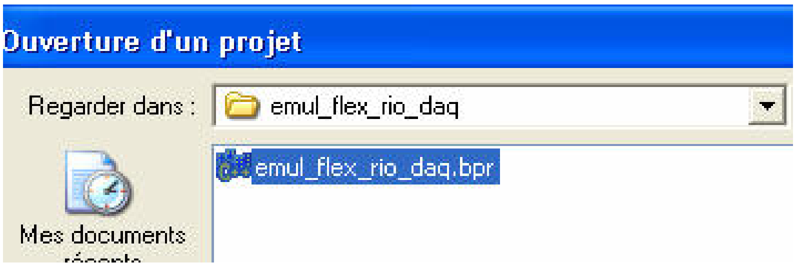
Launch C++ Builder



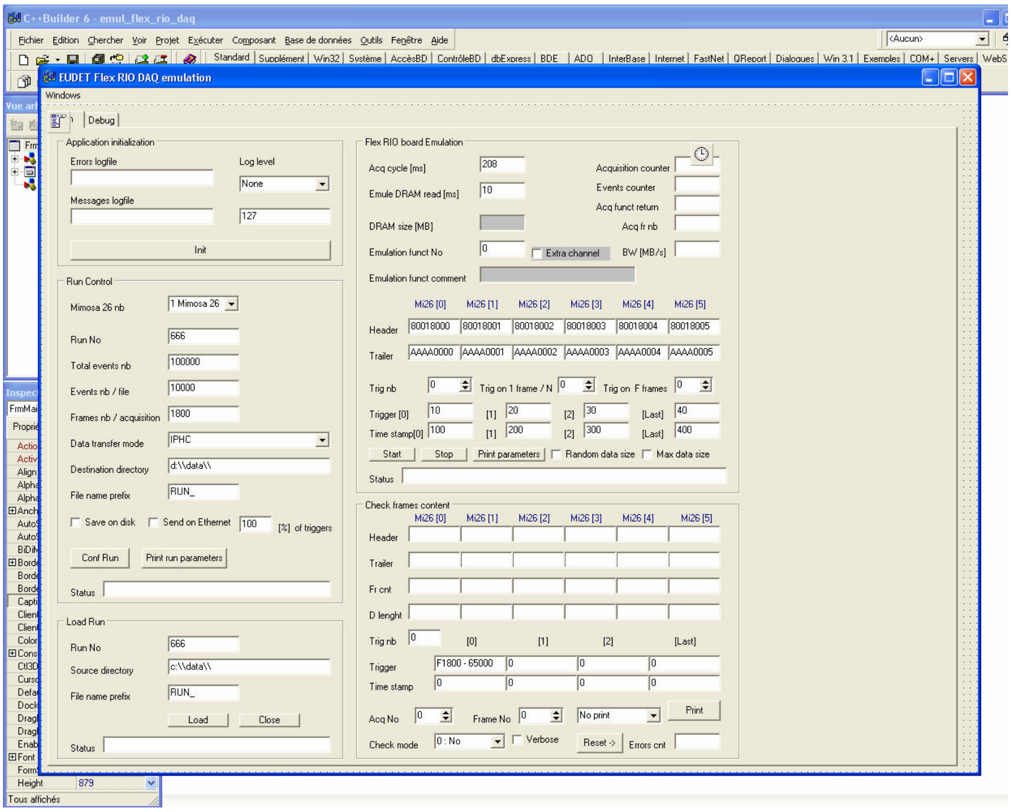
Open project



by selecting emul_flex_rio_daq.bpr file.



You should get the following window.



C++Builder 6 - emul_flex_rio_daq
Fichier Edition Chercher Voir Projet Exécuter Composant Base de données Outils Fenêtre Aide
Standard | Supplément | Win32 | Système | AccèsBD | ContrôleBD | dbExpress | BDE | ADO | InterBase | Internet | FastNet | QReport | Dialogues | Win 3.1 | Exemples | COM+ | Servers | WebS
<Aucun>
EUDET Flex RIO DAQ emulation
Windows
Debug
Init
Conf Run
Print run parameters
Load
Close
Start
Stop
Print parameters
Print
Reset ->
Tous affichés

Application initialization

- Errors logfile:
- Log level: None
- Messages logfile:
- 127

Run Control

- Mimosa 26 nb: 1 Mimosa 26
- Run No: 666
- Total events nb: 100000
- Events nb / file: 10000
- Frames nb / acquisition: 1800
- Data transfer mode: IPHC
- Destination directory: d:\\data\\
- File name prefix: RUN_
- □ Save on disk □ Send on Ethernet 100 [%] of triggers
- Status:

Load Run

- Run No: 666
- Source directory: c:\\data\\
- File name prefix: RUN_
- Status:

Flex RIO board Emulation

- Acq cycle [ms]: 208
- Emule DRAM read [ms]: 10
- DRAM size [MB]:
- Emulation funct No: 0 □ Extra channel
- Emulation funct comment:
- Acquisition counter:
- Events counter:
- Acq funct return:
- Acq fr nb:
- BW [MB/s]:

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

- Trig nb: 0 Trig on 1 frame / N: 0 Trig on F frames: 0

	[0]	[1]	[2]	[Last]
Trigger	10	20	30	40
Time stamp	100	200	300	400

- □ Random data size □ Max data size
- Status:

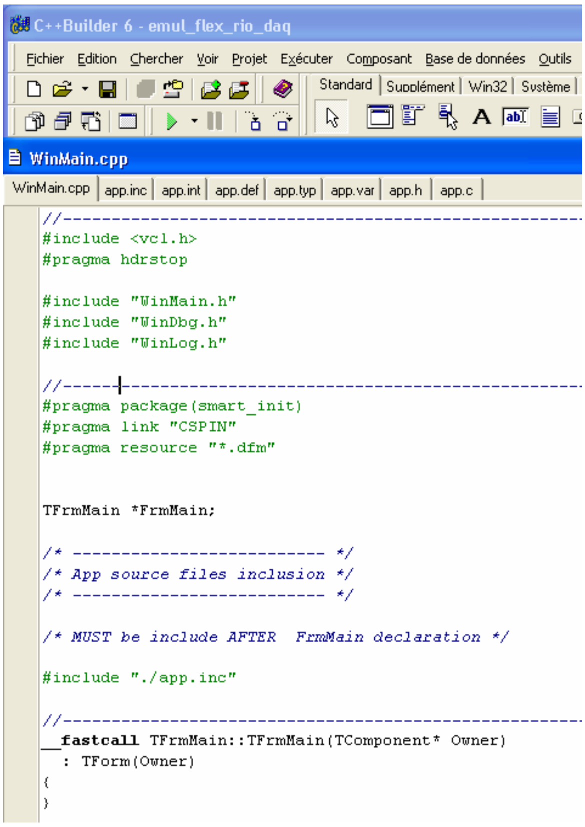
Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header						
Trailer						
Fr cnt						
D lenght						

- Trig nb: 0

	[0]	[1]	[2]	[Last]
Trigger	F1800 - 65000	0	0	0
Time stamp	0	0	0	0

- Acq No: 0 Frame No: 0 No print
- Check mode: 0 : No □ Verbose
- Errors cnt:



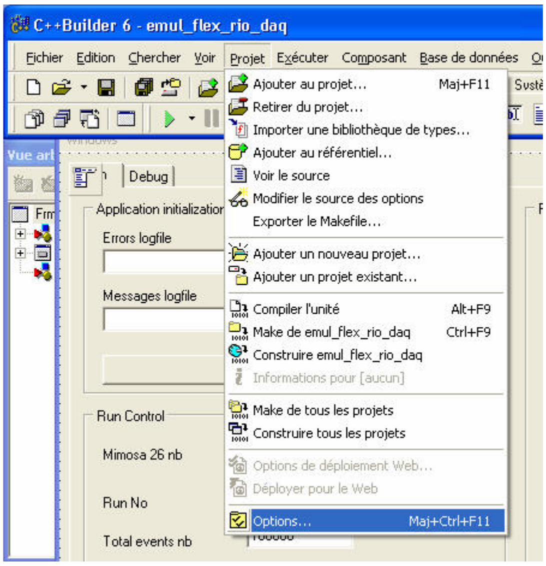
```
//-------------------------------------------------------
#include <vcl.h>
#pragma hdrstop

#include "WinMain.h"
#include "WinDbg.h"
#include "WinLog.h"

//-------------------------------------------------------
#pragma package(smart_init)
#pragma link "CSPIN"
#pragma resource "*.dfm"


TFrmMain *FrmMain;

/* -------------------------- */
/* App source files inclusion */
/* -------------------------- */

/* MUST be include AFTER  FrmMain declaration */

#include "./app.inc"

//-------------------------------------------------------
__fastcall TFrmMain::TFrmMain(TComponent* Owner)
  : TForm(Owner)
{
}
```

Lets have a look on the application source files list :

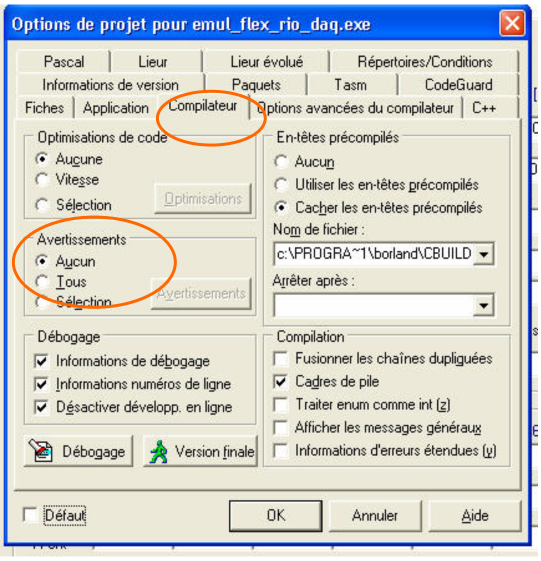
- WinMain.cpp → Main window source code
- app.inc → Includes all source files (“copy / paste” of source …)
- app.int → Includes all interface files (cst, types, etc definitions)
- app.typ → Types definitions
- app.var → Global variables definition
- app.h → Functions prototypes
- app.c → C source code

This organisation of files *.def, typ, var, h, c is not proposed or defined by Borland, it’s my way of programming, which in fact comes from Borland Pascal language.

Disable warnings for compilation …


C++Builder 6 - emul_flex_rio_daq
Fichier Edition Chercher Voir Projet Exécuter Composant Base de données
Ajouter au projet… Maj+F11
Retirer du projet…
Importer une bibliothèque de types…
Ajouter au référentiel…
Voir le source
Modifier le source des options
Exporter le Makefile…
Ajouter un nouveau projet…
Ajouter un projet existant…
Compiler l'unité Alt+F9
Make de emul_flex_rio_daq Ctrl+F9
Construire emul_flex_rio_daq
Informations pour [aucun]
Make de tous les projets
Construire tous les projets
Options de déploiement Web…
Déployer pour le Web
Options… Maj+Ctrl+F11
Debug
Application initialization
Errors logfile
Messages logfile
Run Control
Mimosa 26 nb
Run No
Total events nb

Compilator panel, subpanel " warnings "


Options de projet pour emul_flex_rio_daq.exe
Pascal | Lieur | Lieur évolué | Répertoires/Conditions
Informations de version | Paquets | Tasm | CodeGuard
Fiches | Application | Compilateur | Options avancées du compilateur | C++
Optimisations
Avertissements
Débogage
Version finale
Défaut
OK
Annuler
Aide

Optimisations de code
- ● Aucune
- ○ Vitesse
- ○ Sélection

Avertissements
- ● Aucun
- ○ Tous
- ○ Sélection

Débogage
- ☑ Informations de débogage
- ☑ Informations numéros de ligne
- ☑ Désactiver développ. en ligne

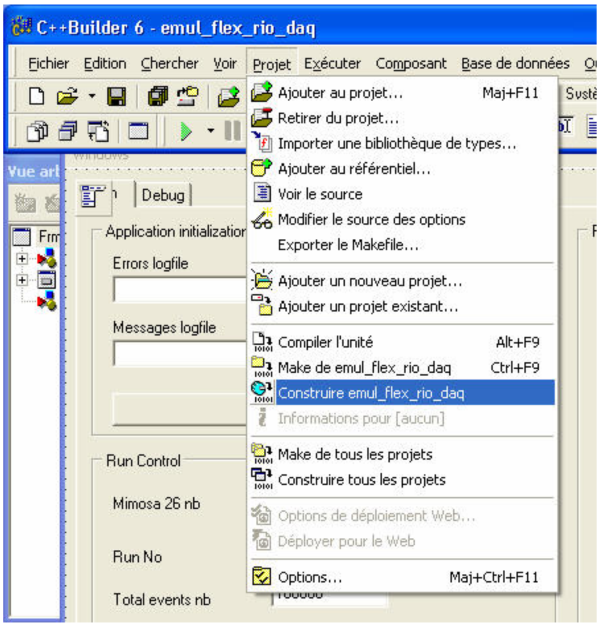
En-têtes précompilés
- ○ Aucun
- ○ Utiliser les en-têtes précompilés
- ● Cacher les en-têtes précompilés

Nom de fichier : c:\PROGRA~1\borland\CBUILD

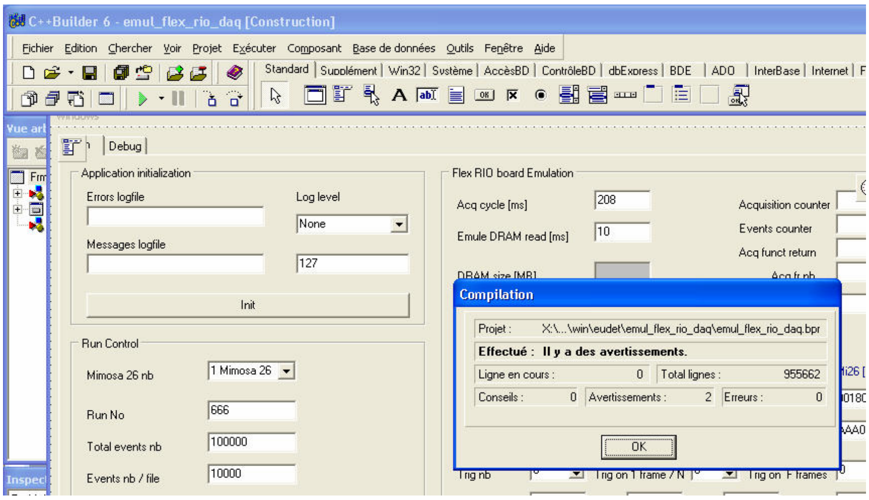
Arrêter après :

Compilation
- □ Fusionner les chaînes dupliquées
- ☑ Cadres de pile
- □ Traiter enum comme int (z)
- □ Afficher les messages généraux
- □ Informations d'erreurs étendues (y)

Compile the project


C++Builder 6 - emul_flex_rio_daq
Fichier
Edition
Chercher
Voir
Projet
Exécuter
Composant
Base de données
Ajouter au projet... Maj+F11
Retirer du projet...
Importer une bibliothèque de types...
Ajouter au référentiel...
Voir le source
Modifier le source des options
Exporter le Makefile...
Ajouter un nouveau projet...
Ajouter un projet existant...
Compiler l'unité Alt+F9
Make de emul_flex_rio_daq Ctrl+F9
Construire emul_flex_rio_daq
Informations pour [aucun]
Make de tous les projets
Construire tous les projets
Options de déploiement Web...
Déployer pour le Web
Options... Maj+Ctrl+F11
Debug
Application initialization
Errors logfile
Messages logfile
Run Control
Mimosa 26 nb
Run No
Total events nb

You should get " 0 errors "



Application initialization

Errors logfile

Messages logfile

Log level: None

127

Run Control

Mimosa 26 nb: 1 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]

Acquisition counter

Events counter

Acq funct return

Acq fr nb

Compilation

Projet : X:\...\win\eudet\emul_flex_rio_daq\emul_flex_rio_daq.bpr

Effectué : Il y a des avertissements.

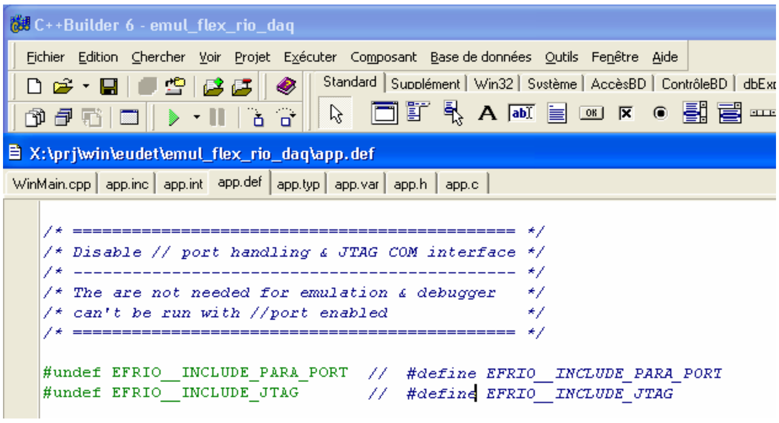
Ligne en cours : 0 Total lignes : 955662

Conseils : 0 Avertissements : 2 Erreurs : 0

OK

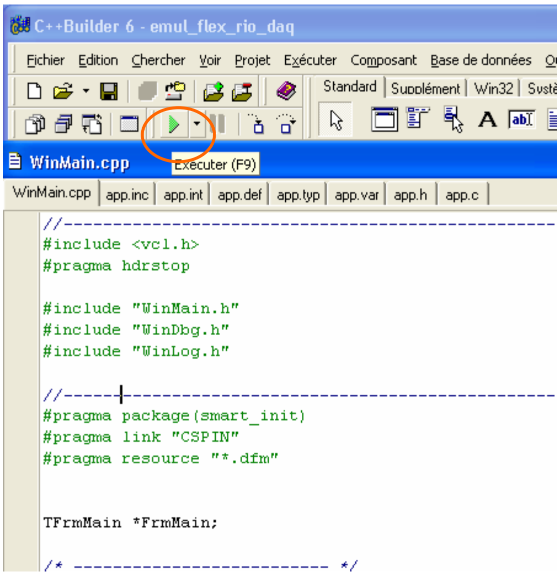
You can run it from the IDE, and of course can use debugger if needed to set breakpoints, inspect variables and so on. But first, you need to check that parallel port is disabled, otherwise you can't run the software from IDE.

The compilation directive " EFRIO__INCLUDE_PARA_PORT " must be disabled.


C++Builder 6 - emul_flex_rio_daq
Fichier Edition Chercher Voir Projet Exécuter Composant Base de données Outils Fenêtre Aide
Standard Supplément Win32 Système AccèsBD ContrôleBD dbExp
X:\prj\win\eudet\emul_flex_rio_daq\app.def
WinMain.cpp app.inc app.int app.def app.typ app.var app.h app.c

```
/* ============================================ */
/* Disable // port handling & JTAG COM interface */
/* -------------------------------------------- */
/* The are not needed for emulation & debugger   */
/* can't be run with //port enabled              */
/* ============================================ */

#undef EFRIO__INCLUDE_PARA_PORT  //  #define EFRIO__INCLUDE_PARA_PORT
#undef EFRIO__INCLUDE_JTAG       //  #define EFRIO__INCLUDE_JTAG
```

Now you can run it by a click on the green arrow.

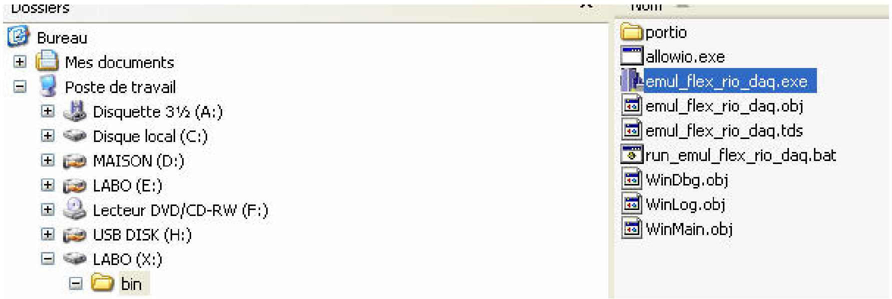

C++Builder 6 - emul_flex_rio_daq
Fichier Edition Chercher Voir Projet Exécuter Composant Base de données O
Standard Supplément Win32 Systè
WinMain.cpp
Exécuter (F9)
WinMain.cpp app.inc app.int app.def app.typ app.var app.h app.c

```
//--------------------------------------------------
#include <vcl.h>
#pragma hdrstop

#include "WinMain.h"
#include "WinDbg.h"
#include "WinLog.h"

//--------------------------------------------------
#pragma package(smart_init)
#pragma link "CSPIN"
#pragma resource "*.dfm"


TFrmMain *FrmMain;

/* ------------------------- */
```

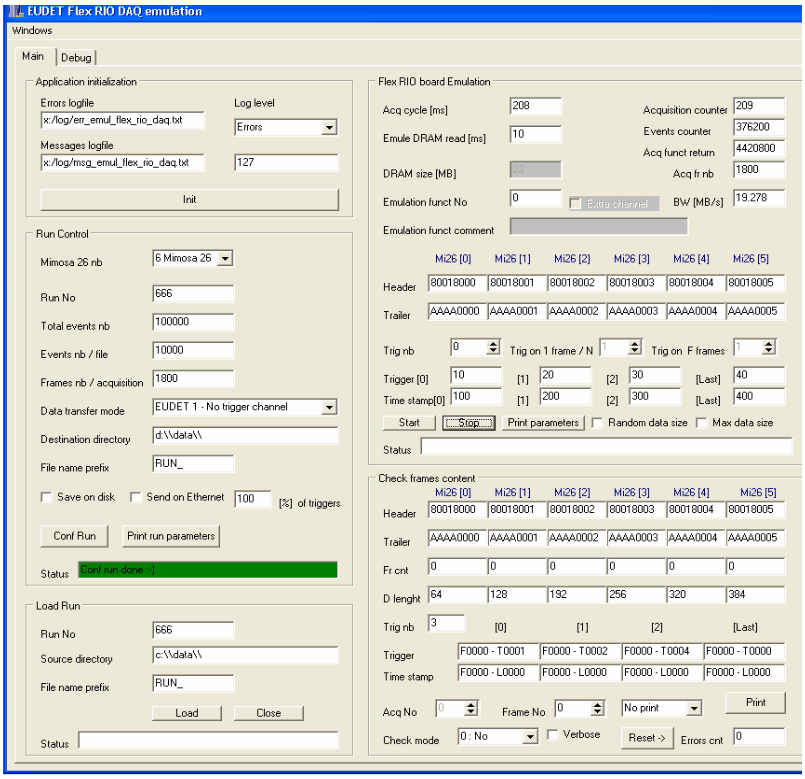
3 How to start the software

You can run it from IDE (see $ 2) if you need the debugger, or as a standalone application as explained in this chapter.

You can launch the executable file emul_flex_rio_daq.exe from x:\bin directory.



The following window should appear.



Application initialization

Errors logfile x:/log/err_emul_flex_rio_daq.txt

Log level Errors

Messages logfile x:/log/msg_emul_flex_rio_daq.txt

127

Run Control

Mimosa 26 nb 6 Mimosa 26

Run No 666

Total events nb 100000

Events nb / file 10000

Frames nb / acquisition 1800

Data transfer mode EUDET 1 - No trigger channel

Destination directory d:\\data\\

File name prefix RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Status Conf run done :-)

Load Run

Run No 666

Source directory c:\\data\\

File name prefix RUN_

Status

Flex RIO board Emulation

Acq cycle [ms] 208

Emule DRAM read [ms] 10

DRAM size [MB] 23

Emulation funct No 0 □ Extra channel

Emulation funct comment

Acquisition counter 209

Events counter 376200

Acq funct return 4420800

Acq fr nb 1800

BW [MB/s] 19.278

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb 0 Trig on 1 frame / N 1 Trig on F frames 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

□ Random data size □ Max data size

Status

Check frames content

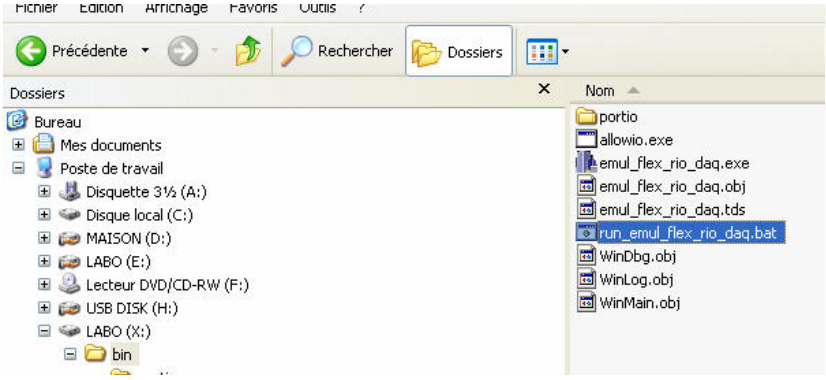
	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	64	128	192	256	320	384

Trig nb 3	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No 0 Frame No 0 No print

Check mode 0 : No □ Verbose Errors cnt 0

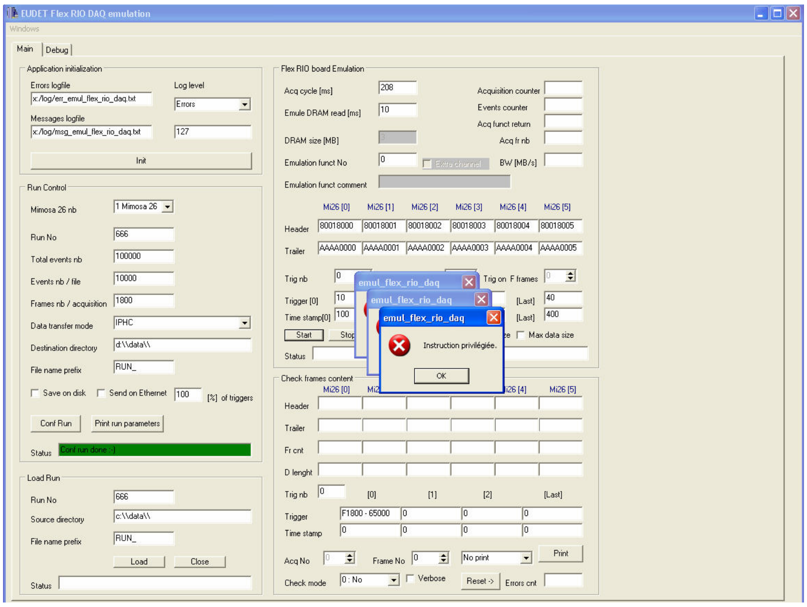
The emulator can also allow you to measure functions execution time by generating a pulse on PC parallel port during their execution. If you want to run the software in this mode, you need to compile it with parallel port enabled (the conditional compilation directive EFRIO__INCLUDE_PARA_PORT must be enabled in file app.def) and to run it via the a batch file names : run emul_flex_rio_daq.bat.



Bureau
Mes documents
Poste de travail
Disquette 3½ (A:)
Disque local (C:)
MAISON (D:)
LABO (E:)
Lecteur DVD/CD-RW (F:)
USB DISK (H:)
LABO (X:)
bin

portio
allowio.exe
emul_flex_rio_daq.exe
emul_flex_rio_daq.obj
emul_flex_rio_daq.tds
run_emul_flex_rio_daq.bat
WinDbg.obj
WinLog.obj
WinMain.obj

If, by mistake, you run the version compiled with PC parallel port handling directly via a call to executable file (not via the batch) you will get exception errors. Because access to parallel port is not allowed, please kill the program and start it via batch file.



Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt
Log level: Errors
Messages logfile: x:/log/msg_emul_flex_rio_daq.txt
127

Run Control

Mimosa 26 nb: 1 Mimosa 26
Run No: 666
Total events nb: 100000
Events nb / file: 10000
Frames nb / acquisition: 1800
Data transfer mode: IPHC
Destination directory: d:\\data\\
File name prefix: RUN_
□ Save on disk □ Send on Ethernet 100 [%] of triggers
Status: Conf run done :-)

Load Run

Run No: 666
Source directory: c:\\data\\
File name prefix: RUN_
Status:

Flex RIO board Emulation

Acq cycle [ms]: 208
Emule DRAM read [ms]: 10
DRAM size [MB]: 3
Emulation funct No: 0
□ Extra channel
Emulation funct comment:
Acquisition counter:
Events counter:
Acq funct return:
Acq fr nb:
BW [MB/s]:

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 0
Trigger [0]: 10
Time stamp[0]: 100
Trig on F frames: 0
[Last]: 40
[Last]: 400
□ Max data size
Status:

Instruction privilégiée.

Check frames content

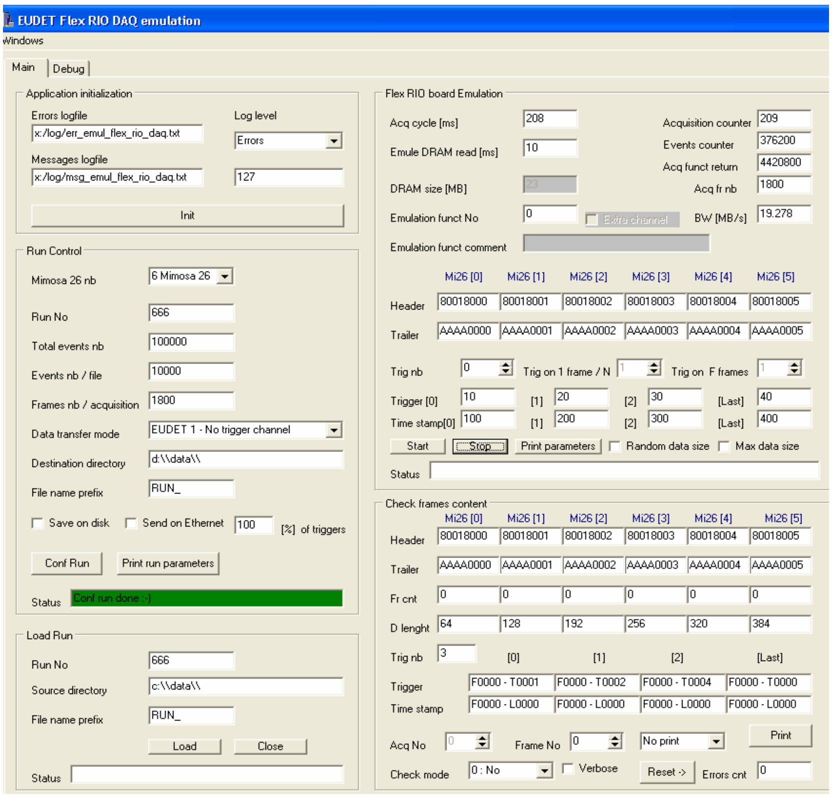
Trig nb: 0

	[0]	[1]	[2]	[Last]
Trigger	F1800 - 65000	0	0	0
Time stamp	0	0	0	0

Acq No: 0
Frame No: 0
No print
Check mode: 0 : No
□ Verbose
Errors cnt:

4 Software GUI overview

The software has two panels : Main and a Debug panel on which user can add GUI controls and indicators to test code. There is also a Window to log messages.



Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 1 - No trigger channel

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 23

Emulation funct No: 0

□ Extra channel

Emulation funct comment:

Acquisition counter: 209

Events counter: 376200

Acq funct return: 4420800

Acq fr nb: 1800

BW [MB/s]: 19.278

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 0 Trig on 1 frame / N: 1 Trig on F frames: 1

	[0]	[1]	[2]	[Last]
Trigger	10	20	30	40
Time stamp	100	200	300	400

□ Random data size □ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	64	128	192	256	320	384

Trig nb: 3

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No: 0 Frame No: 0 No print

Check mode: 0 : No □ Verbose

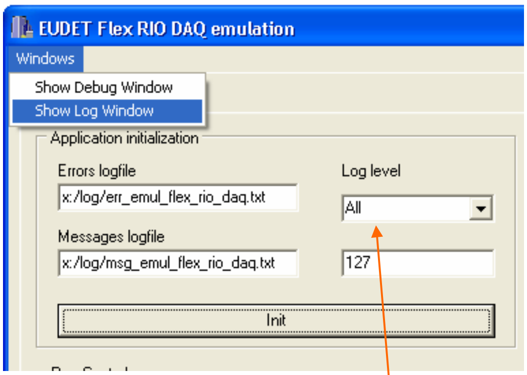
Errors cnt: 0

The main panel hast five sub panels :

- Application initialization
- Run control
- Load Run
- Run RIO board Emulation
- Check frames content

4.1 Window to log errors and general messages

Open this window via the menu " Windows ", submenu " Show Log Window ".



Errors logfile: x:/log/err_emul_flex_rio_daq.txt

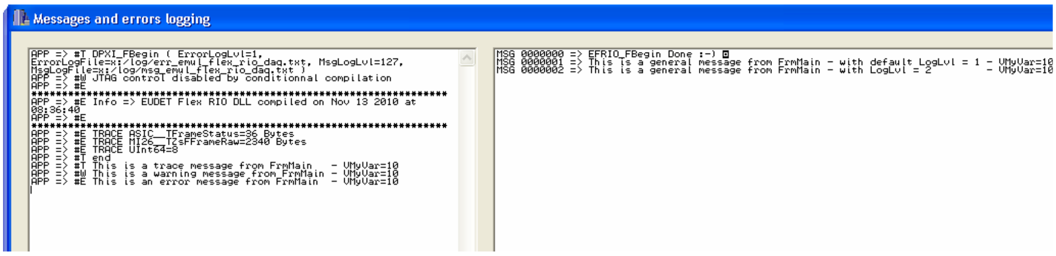
Log level: All

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

The log level specified is " All ", it means that all kind of messages will be logged.

The following window will appear

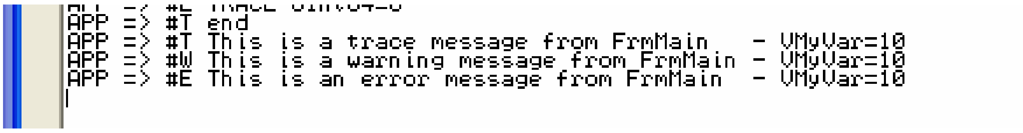


```
APP => #T DPXI_FBegin ( ErrorLogLvl=1,
ErrorLogFile=x:/log/err_emul_flex_rio_daq.txt, MsgLogLvl=127,
MsgLogFile=x:/log/msg_emul_flex_rio_daq.txt )
APP => #W JTAG control disabled by conditionnal compilation
APP => #E
********************************************************************
APP => #E Info => EUDET Flex RIO DLL compiled on Nov 13 2010 at
08:36:40
APP => #E
********************************************************************
APP => #E TRACE ASIC__TFrameStatus=36 Bytes
APP => #E TRACE MI26__TZsFFrameRaw=2340 Bytes
APP => #E TRACE UInt64=8
APP => #T end
APP => #T This is a trace message from FrmMain   - VMyVar=10
APP => #W This is a warning message from FrmMain - VMyVar=10
APP => #E This is an error message from FrmMain  - VMyVar=10
```

```
MSG 0000000 => EFRIO_FBegin Done :-) 
MSG 0000001 => This is a general message from FrmMain - with default LogLvl = 1 - VMyVar=10
MSG 0000002 => This is a general message from FrmMain - with LogLvl = 2         - VMyVar=10
```

Left part displays errors message and right part general messages. The messages are also written in log file specified in the " Application initialization " subpanel.

A closer look to error messages list shows three messages printed as demonstration of messages logging macros.

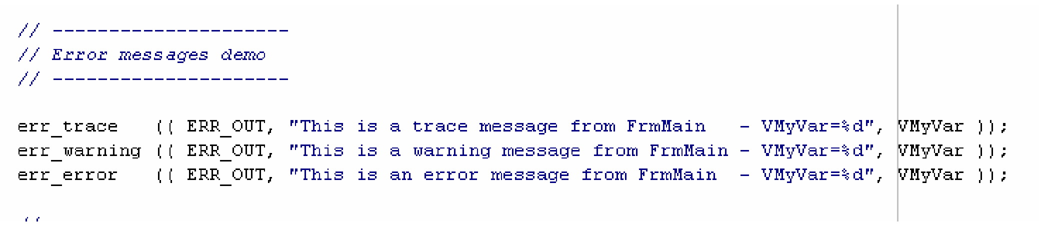


```
APP => #T end
APP => #T This is a trace message from FrmMain   - VMyVar=10
APP => #W This is a warning message from FrmMain - VMyVar=10
APP => #E This is an error message from FrmMain  - VMyVar=10
```

The letter indicates the level of error and the macro used to print it :

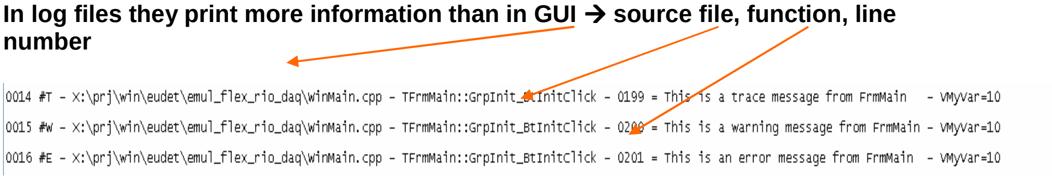
- **#T → Code tracing message - → Macro err_trace (…)**
- **#W → Warning messages → Macro err_warning (…)**
- **#E → Error message → Macro err_error (…)**

This is the source code which call theses macro



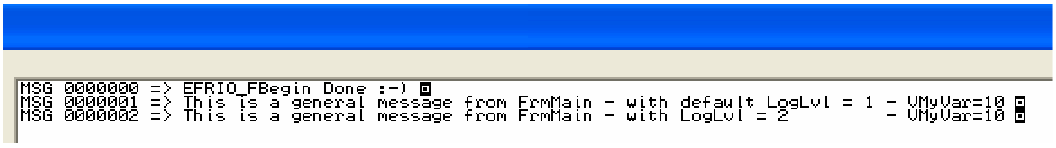
```
// ----------------------
// Error messages demo
// ----------------------

err_trace   (( ERR_OUT, "This is a trace message from FrmMain   - VMyVar=%d", VMyVar ));
err_warning (( ERR_OUT, "This is a warning message from FrmMain - VMyVar=%d", VMyVar ));
err_error   (( ERR_OUT, "This is an error message from FrmMain  - VMyVar=%d", VMyVar ));
```

These macros work like the " old " printf (…), accept the same syntax. In log files they print more information than in GUI → source file, function, line number

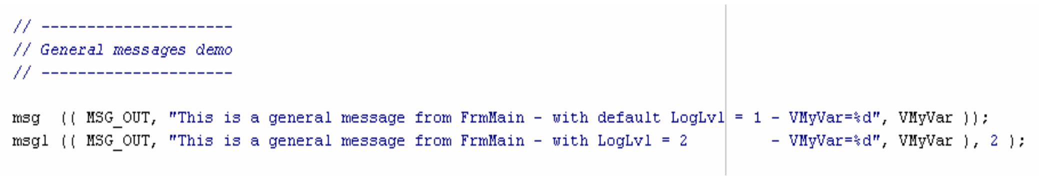


```
0014 #T - X:\prj\win\eudet\emul_flex_rio_daq\WinMain.cpp - TFrmMain::GrpInit_BtInitClick - 0199 = This is a trace message from FrmMain   - VMyVar=10
0015 #W - X:\prj\win\eudet\emul_flex_rio_daq\WinMain.cpp - TFrmMain::GrpInit_BtInitClick - 0200 = This is a warning message from FrmMain - VMyVar=10
0016 #E - X:\prj\win\eudet\emul_flex_rio_daq\WinMain.cpp - TFrmMain::GrpInit_BtInitClick - 0201 = This is an error message from FrmMain  - VMyVar=10
```

There is also macros to log general message (right panel)

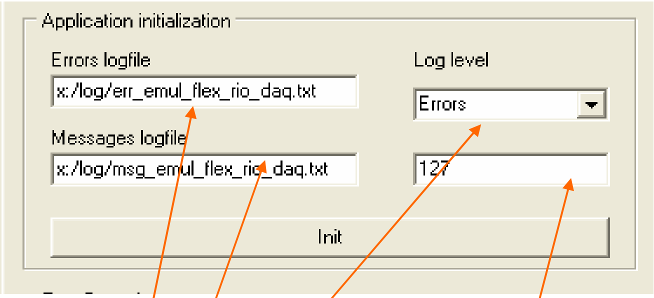


```
MSG 0000000 => EFRIO_FBegin Done :-)
MSG 0000001 => This is a general message from FrmMain - with default LogLvl = 1 - VMyVar=10
MSG 0000002 => This is a general message from FrmMain - with LogLvl = 2         - VMyVar=10
```



```
// ---------------------
// General messages demo
// ---------------------

msg  (( MSG_OUT, "This is a general message from FrmMain - with default LogLvl = 1 - VMyVar=%d", VMyVar ));
msgl (( MSG_OUT, "This is a general message from FrmMain - with LogLvl = 2         - VMyVar=%d", VMyVar ), 2 );
```

4.2 Application initialization sub panel



Application initialization

Errors logfile

x:/log/err_emul_flex_rio_daq.txt

Messages logfile

x:/log/msg_emul_flex_rio_daq.txt

Log level

Errors

127

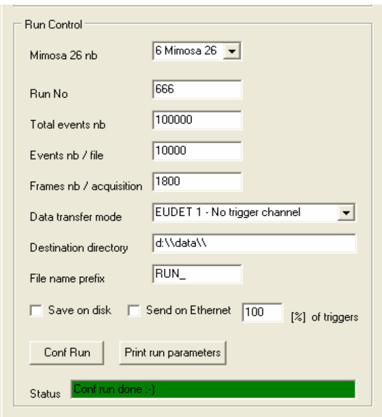
Via this panel you can define

- **The error logging level which can be**
  - **No**
  - **All**
  - **Warnings & Errors**
  - **Errors**
- **The general message logging level**
  - **127 to gel all messages**
  - **Other value → depend on the level convention used in macro call**
- **The log files used to store messages**
  - **Errors log file → for errors**
  - **Messages log file → for general messages**

Errors and general message are displays in " Errors & Messages" window (see $ 4.1) and printed in log files.

Once you have defined errors and messages log levels, you can click on " Init " button to initialize library and have a look to messages in the log window.

4.3 Run control sub panel



Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 1 - No trigger channel

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Status: Conf run done :-)

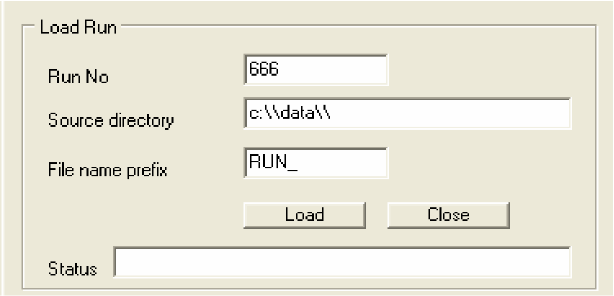
Via this panel you configure the run control :

- Number of Mimosa 26 → Only two options : 1 or 6
- A number to identify the run
- The total event number to store in the run
- The events number stored per run file (a run is split in many files)
- The frames number per acquisition
- The data transfer mode (IPHC, EUDET 1,2,3 → See Introduction)
- Destination directory for run file
- Run file name prefix (RUN_666 → RUN_ is the prefix)
- Selection to save or not data to disk
- Selection to send data or not on Ethernet + % of triggers / events sent

Some of theses parameters are not handled now, but they will be useful later.

Perform run configuration by clicking on button “ Conf RUN”. You can also print run configuration parameters record in log file via button “ Print run parameters ”.

4.4 Load Run sub panel



Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Status:

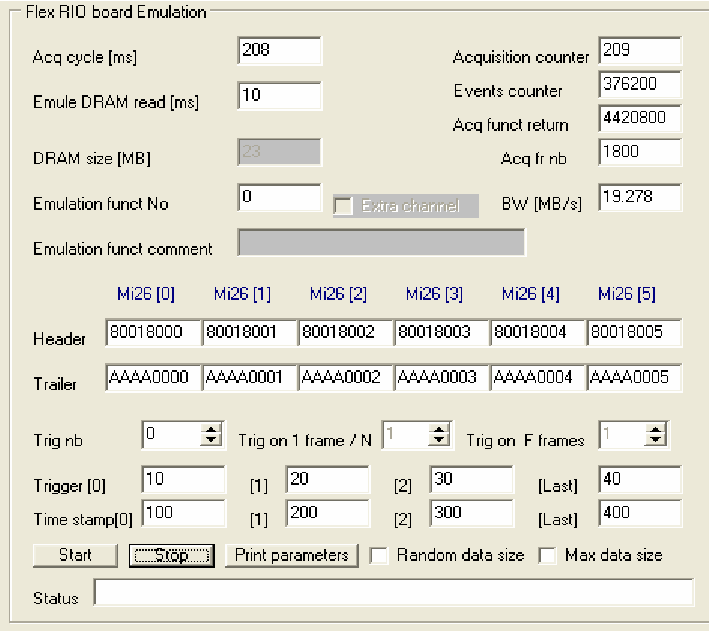
Via this panel you specify the run file you want to load :

- **The number which identifies the run**
- **The run directory**
- **The run file prefix**

Once parameters are set, click on “ Load ” to load a run file, the status field will indicated the result of operation.

Before loading another run or before closing the application, please click on
“ Close ” button.

4.5 Run RIO board Emulation sub panel



Flex RIO board Emulation

Acq cycle [ms] 208

Emule DRAM read [ms] 10

DRAM size [MB] 23

Emulation funct No 0

□ Extra channel

Emulation funct comment

Acquisition counter 209

Events counter 376200

Acq funct return 4420800

Acq fr nb 1800

BW [MB/s] 19.278

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb 0 Trig on 1 frame / N 1 Trig on F frames 1

		[1]	[2]	[Last]
Trigger [0]	10	20	30	40
Time stamp[0]	100	200	300	400

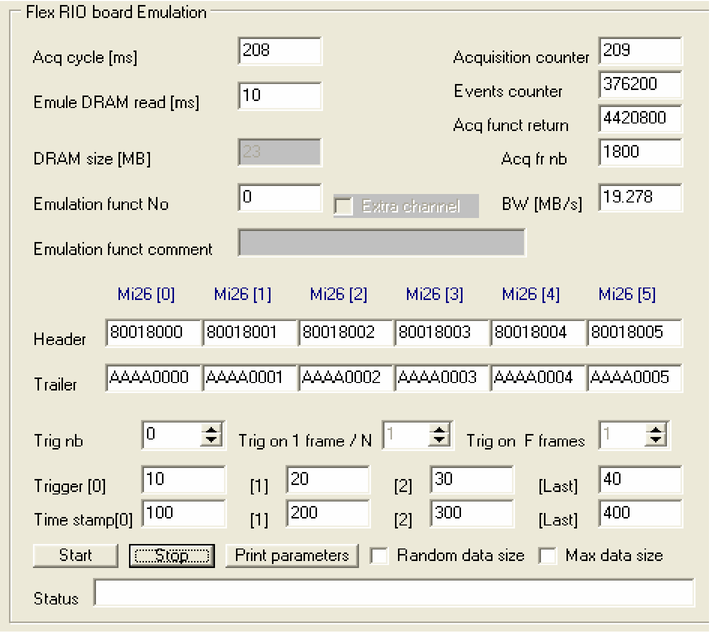
□ Random data size □ Max data size

Status

Via this panel you configure DAQ emulation parameters

- **Acq cycle [ms] → Period between two acquisitions**
- **Emule DRAM → A delay to emulate Flex RIO DRAM access (not very useful)**
- **Emulation function No → Select which emulation function to use it is not implemented now, there is only one emulation function.**
- **Header → Header of each Mimosa 26**
- **Trailer → Trailer of each Mimosa 26**
- **Trig nb → Number of triggers / frame**
  - **In mode IPHC, EUDET 1 → field ignored → Always 3 triggers**
  - **In mode IPHC, EUDET 1-2 " Trig nb " are emulated on each frame.**
  - **in mode IPHC, EUDET 1-2 " Trig nb " are emulated on each frame. In mode EUDET 3 it is possible to emulate " Trig nb " each N**

frames on F consecutive frames via the fields " Trig on 1 frame/n ", " Trig on F frames ".



Flex RIO board Emulation

Acq cycle [ms]	208	Acquisition counter	209
Emule DRAM read [ms]	10	Events counter	376200
		Acq funct return	4420800
DRAM size [MB]	23	Acq fr nb	1800
Emulation funct No	0	BW [MB/s]	19.278
Emulation funct comment			

□ Extra channel

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

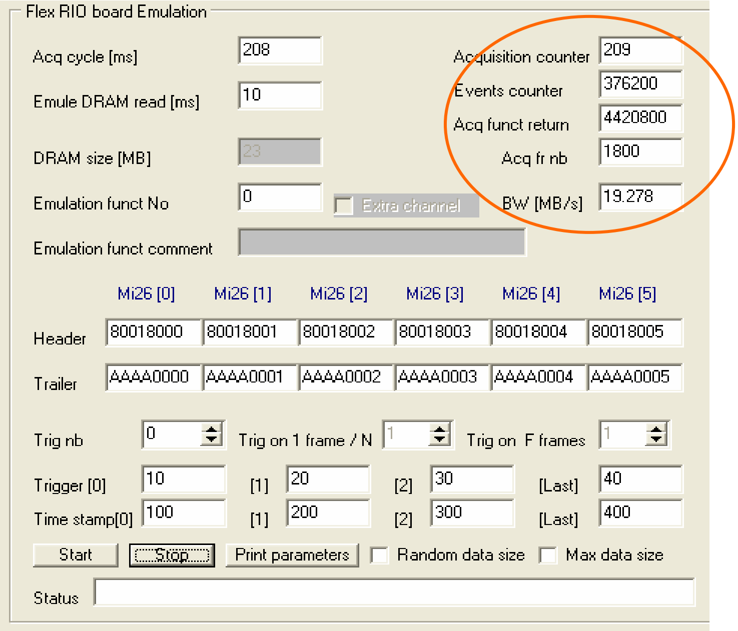
Trig nb 0 Trig on 1 frame / N 1 Trig on F frames 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

- Trigger [0], [1], [2], [last] = TLU triggers (up to 288 / frame)
  - In mode IPHC, EUDET 1 → They are ignored, three triggers are generated with values 16, 32, 64 for IPHC mode and 1, 2, 4 for EUDET 1 mode.
  - In modes EUDET2, EUDET 3 they allow to specify the first three triggers values [0], [1], [2] and the value of the last trigger [Trig nb – 1]. The triggers between [2] and [Trig nb – 1]. Have their value set to 0.
- Time stamp [0], [1], [2], [last] = Fex RIO triggers (up to 288 / frame)
  - In mode IPHC, EUDET 1 → They are ignored → no time stamp
  - In modes EUDET2, EUDET 3 they allow to specify the first three time stamps values [0], [1], [2] and the value of the last time stamp [Trig nb – 1]. The time stamp between [2] and [Trig nb – 1]. Have their value set to 0.
- “ Trig on 1 frame / N ” & “ Trig on F frames ” are only enabled in mode EUDET 3 and allow to generated “ Trig Nb ” triggers on F consecutive frames each N frames.

- **Random data size → Allows to generate random data size on each Mimosa 26, by default the data size if fixed and hard coded in the emulation function.**

- **Max data size → Set maximum data size on first Mimosa 26**



Flex RIO board Emulation

Acq cycle [ms] 208

Emule DRAM read [ms] 10

DRAM size [MB] 23

Emulation funct No 0 □ Extra channel

Emulation funct comment

Acquisition counter 209

Events counter 376200

Acq funct return 4420800

Acq fr nb 1800

BW [MB/s] 19.278

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb 0 Trig on 1 frame / N 1 Trig on F frames 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

Status

This panel has also displays

- **Acquisition counter → Counter of acquisitions**

- **Events counter → = Acquisition counter X Frame nb per acquisition**

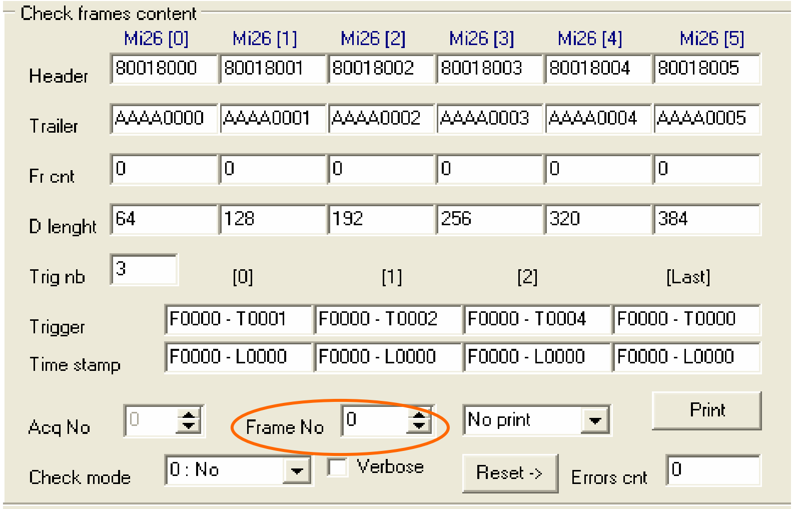
- **Acq funct return → Code returned by acquisition function = acq size**

- **Acq fr nb → Number of frames per acquisition**

- **BW [MB/s} → Evaluation of data rate produced by DAQ**

Click on " Start " button to start DAQ emulation, on " Stop " to stop it and on " Print parameters " to print DAQ emulation parameters record value in log window & file.

Check frames content subpanel



Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	64	128	192	256	320	384

Trig nb 3

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No 0 Frame No 0 No print

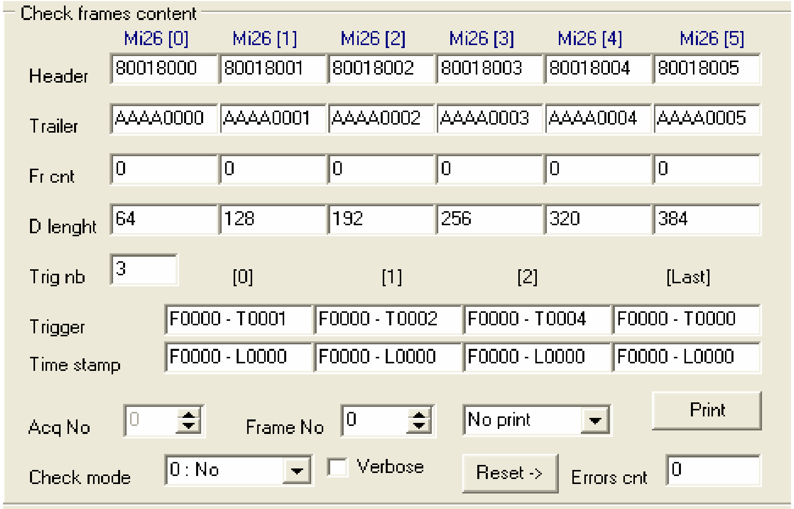
Check mode 0 : No ☐ Verbose Errors cnt 0

This panel shows on-line the values of Mimosa 26 data stream “ relevant fields ” of the frame selected by the control “ Frame No ”

- Header of each Mimosa 26
- Trailer of each Mimosa 26
- Frame counter of each Mimosa 26
- Data length [in bytes] of each Mimosa 26
- The triggers number
- The first three triggers (TLU) + last one
- The first three time stamps (Flex RIO) + last one

It’s also possible to display frames off-line when emulation has been stopped. The eudet_frio lib keep in a buffer all the frames of current acquisition. Therefore it’s possible to scan them off-line, specify the index of the frame in field “ Frame No ”, it will display content.

If you want to display frames content in text mode, select a print level via the control “ No print ”, move between frames and look in the errors and messages Window.



Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	64	128	192	256	320	384

Trig nb: 3	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No: 0   Frame No: 0   No print

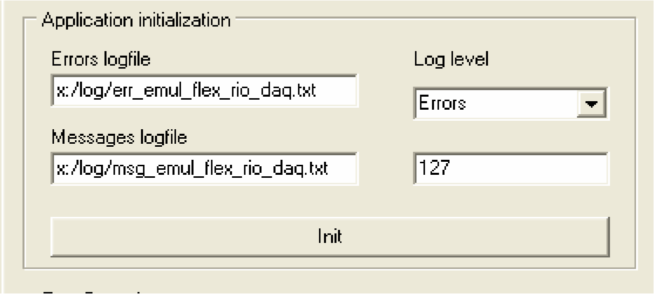
Check mode: 0 : No   □ Verbose   Errors cnt: 0

A verification of all frames of each acquisition can be done on-line. You can select the check level via the control " Check mode ", errors are count in the display " Errors cnt", which you can reset via button " Reset -→ ".

It's also possible to display frames loaded from a file. Load a file via the sub panel " Load run ", the " Acq No " control will be enabled and allows you to select the acquisition to scan via " Frame No " control. You must select the " Acq No " first and after you can display frames by selecting them via " Frame No ". If you forget to specify
" Acq No" bad results may be displayed.

5 Procedure to start emulation

Initialize the software, need to be done only one time at beginning.



Application initialization

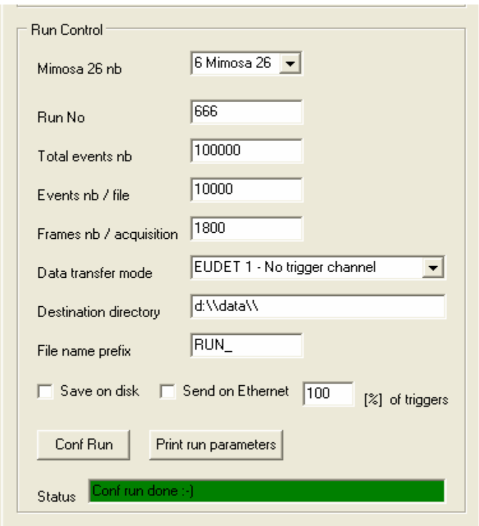
Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

Log level: Errors

127

Set run configuration, the. Main parameters are the Mimosa 26 number and the data transfer mode. Then click on “ Conf Run ”.



Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 1 - No trigger channel

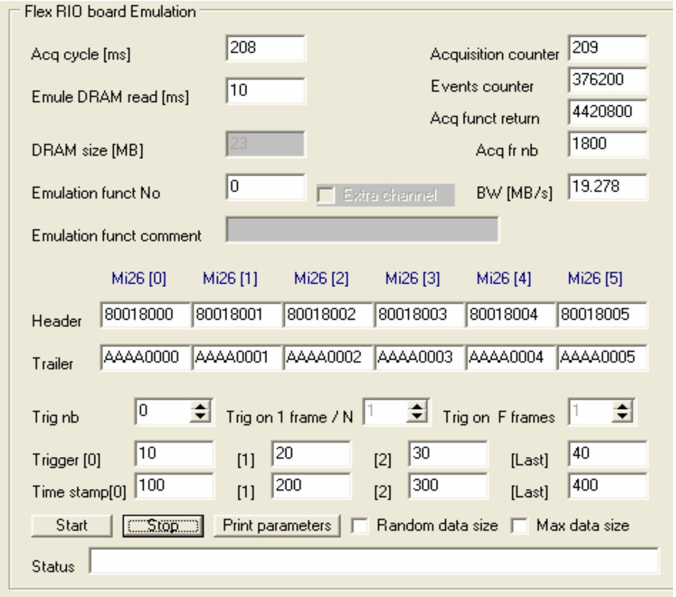
Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Status: Conf run done :-)

Set emulation configuration → header, trailer, triggers … Start emulation by a click on " Start " button.



Flex RIO board Emulation

Acq cycle [ms] 208

Emule DRAM read [ms] 10

DRAM size [MB] 23

Emulation funct No 0

□ Extra channel

Emulation funct comment

Acquisition counter 209

Events counter 376200

Acq funct return 4420800

Acq fr nb 1800

BW [MB/s] 19.278

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

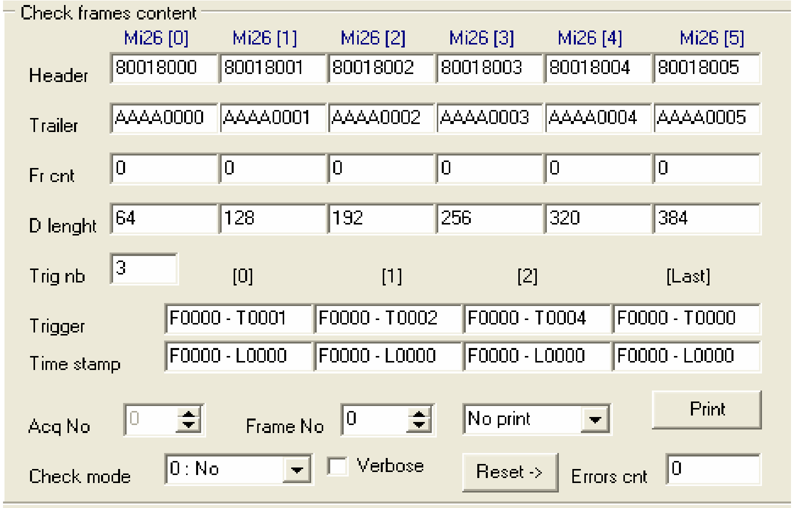
Trig nb 0   Trig on 1 frame / N 1   Trig on F frames 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

□ Random data size   □ Max data size

Status

Look at results in " Check frames content " sub panel.



Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	64	128	192	256	320	384

Trig nb 3

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No 0   Frame No 0   No print

Check mode 0 : No   □ Verbose

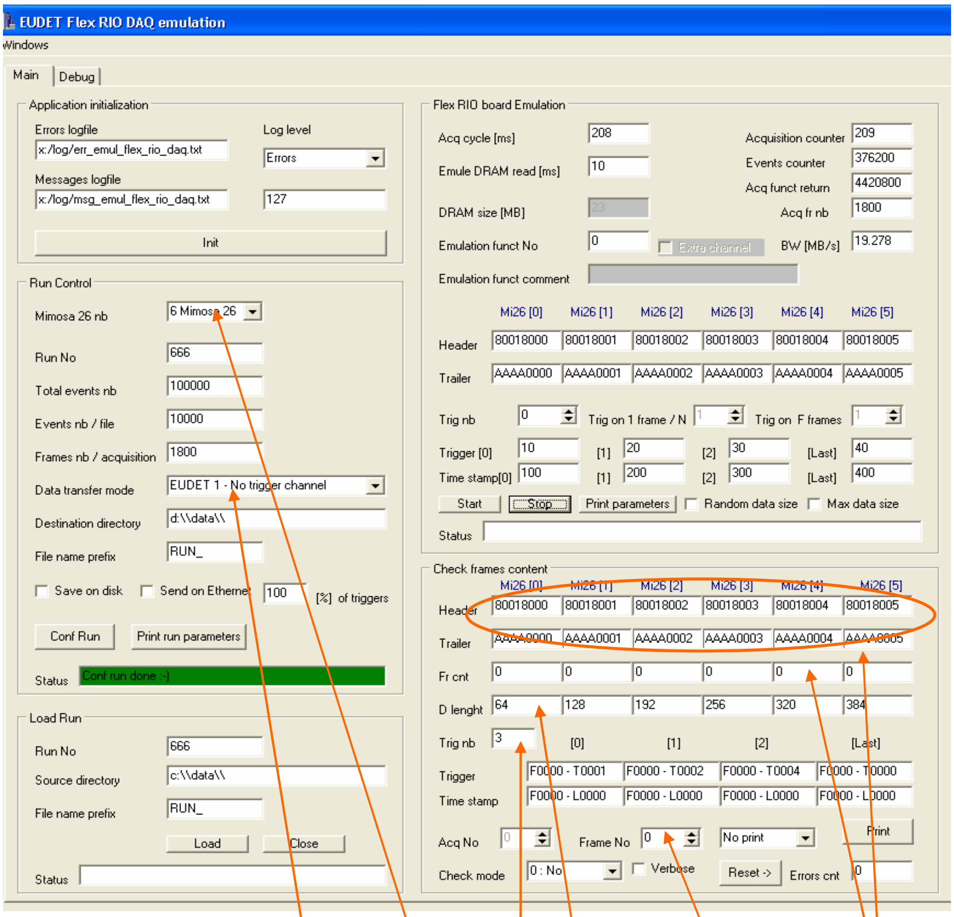
Errors cnt 0

You can stop emulation via " Stop" button and go to " Run control " panel to select others run parameters.

6 Playing with the DAQ emulator

6.1 Mode EUDET 1

6.1.1 Fixed frame size



EUDET Flex RIO DAQ emulation

Windows

Main | Debug

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Init

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 1 - No trigger channel

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Conf Run | Print run parameters

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Load | Close

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 23

Emulation funct No: 0 □ Extra channel

Emulation funct comment:

Acquisition counter: 209

Events counter: 376200

Acq funct return: 4420800

Acq fr nb: 1800

BW [MB/s]: 19.278

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 0 Trig on 1 frame / N: 1 Trig on F frames: 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

Start | Stop | Print parameters | □ Random data size □ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	64	128	192	256	320	384

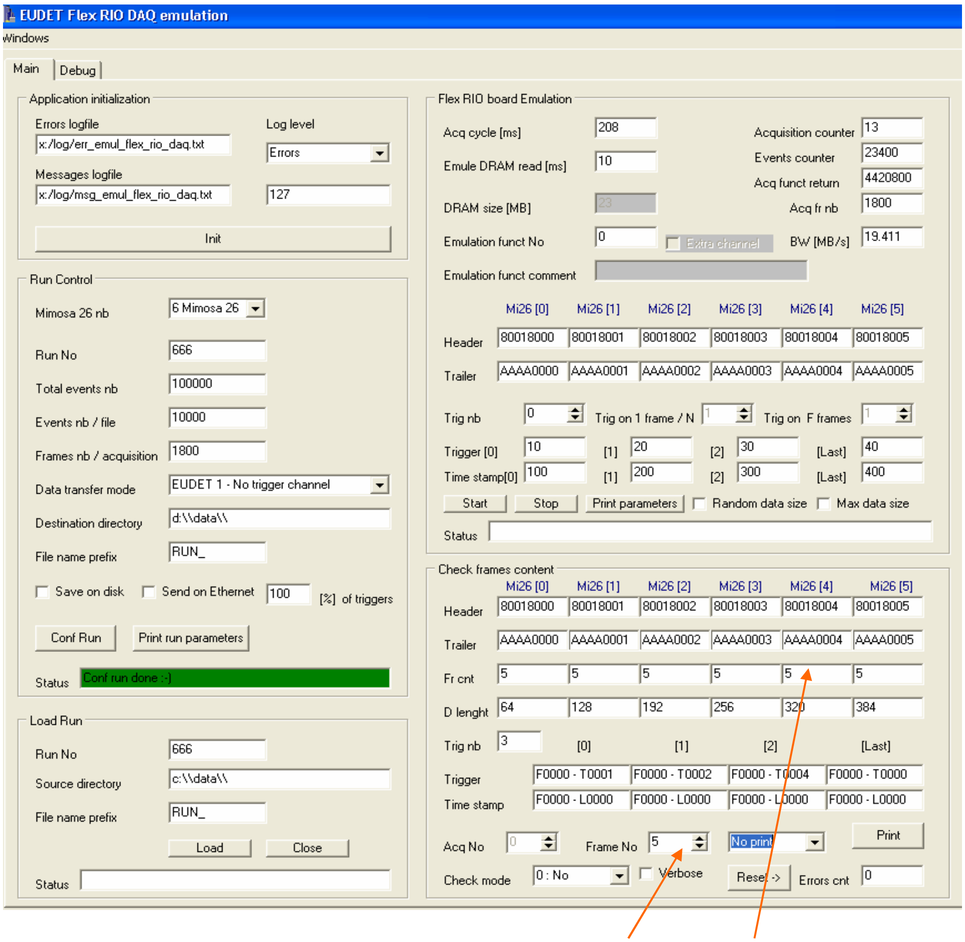
Trig nb: 3

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No: 0 Frame No: 0 No print | Print

Check mode: 0 : No □ Verbose Reset -> Errors cnt: 0

Emulator in mode EUDET 1, 6 Mimosa 26, default frame size, three trigger are generated and frame No 0 is displayed.



Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 1 - No trigger channel

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 23

Emulation funct No: 0 □ Extra channel

Emulation funct comment:

Acquisition counter: 13

Events counter: 23400

Acq funct return: 4420800

Acq fr nb: 1800

BW [MB/s]: 19.411

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 0 Trig on 1 frame / N: 1 Trig on F frames: 1

	[0]	[1]	[2]	[Last]
Trigger	10	20	30	40
Time stamp	100	200	300	400

□ Random data size □ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	5	5	5	5	5	5
D lenght	64	128	192	256	320	384

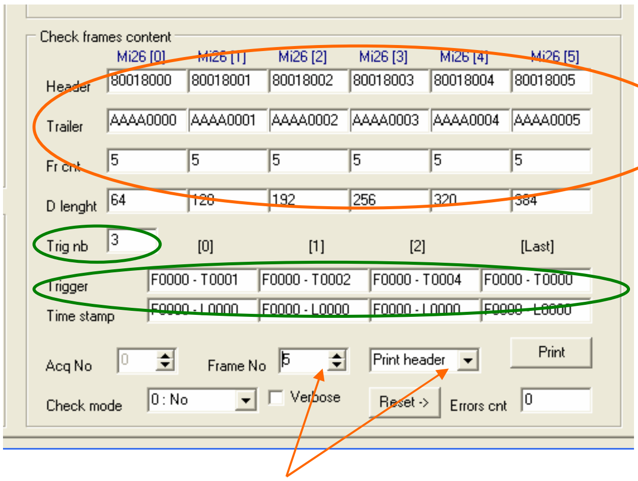
Trig nb: 3

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No: 0 Frame No: 5 No print

Check mode: 0 : No □ Verbose Errors cnt: 0

Now frame No 5 is displayed



Check frames content

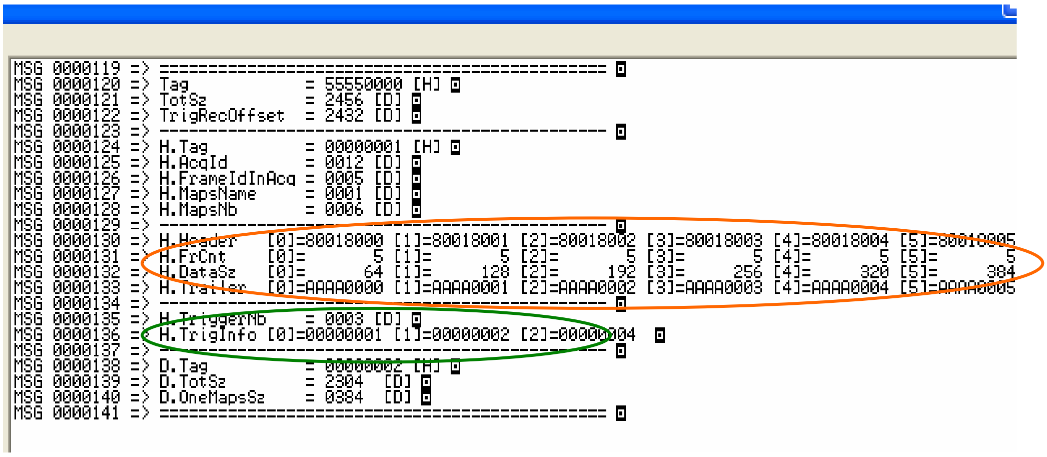
	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	5	5	5	5	5	5
D lenght	64	128	192	256	320	384

Trig nb 3	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No 0 Frame No 5 Print header

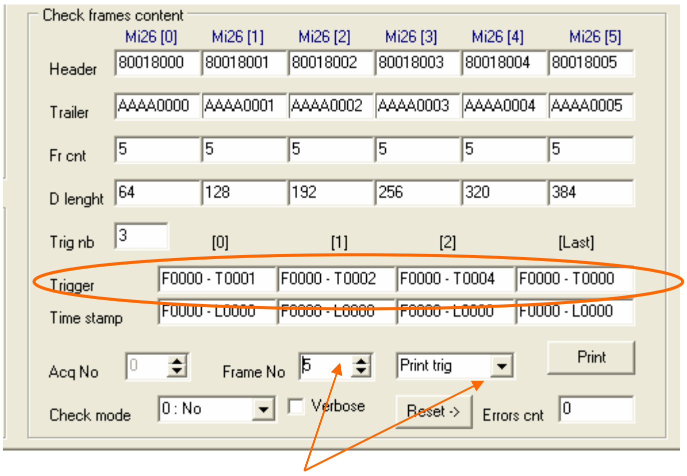
Check mode 0 : No ☐ Verbose Errors cnt 0

We print header of frame No 5 in log windows



```
MSG 0000119 => ============================================= ◘
MSG 0000120 => Tag              = 55550000 [H] ◘
MSG 0000121 => TotSz            = 2456 [D] ◘
MSG 0000122 => TrigRecOffset    = 2432 [D] ◘
MSG 0000123 => --------------------------------------------- ◘
MSG 0000124 => H.Tag            = 00000001 [H] ◘
MSG 0000125 => H.AcqId          = 0012 [D] ◘
MSG 0000126 => H.FrameIdInAcq   = 0005 [D] ◘
MSG 0000127 => H.MapsName       = 0001 [D] ◘
MSG 0000128 => H.MapsNb         = 0006 [D] ◘
MSG 0000129 => --------------------------------------------- ◘
MSG 0000130 => H.Header    [0]=80018000 [1]=80018001 [2]=80018002 [3]=80018003 [4]=80018004 [5]=80018005
MSG 0000131 => H.FrCnt     [0]=        5 [1]=        5 [2]=        5 [3]=        5 [4]=        5 [5]=        5
MSG 0000132 => H.DataSz    [0]=       64 [1]=      128 [2]=      192 [3]=      256 [4]=      320 [5]=      384
MSG 0000133 => H.Trailer   [0]=AAAA0000 [1]=AAAA0001 [2]=AAAA0002 [3]=AAAA0003 [4]=AAAA0004 [5]=AAAA0005
MSG 0000134 => --------------------------------------------- ◘
MSG 0000135 => H.TriggerNb      = 0003 [D] ◘
MSG 0000136 => H.TrigInfo [0]=00000001 [1]=00000002 [2]=00000004  ◘
MSG 0000137 => --------------------------------------------- ◘
MSG 0000138 => D.Tag            = 00000002 [H] ◘
MSG 0000139 => D.TotSz          = 2304   [D] ◘
MSG 0000140 => D.OneMapsSz      = 0384   [D] ◘
MSG 0000141 => ============================================= ◘
```

This is the print result



Check frames content

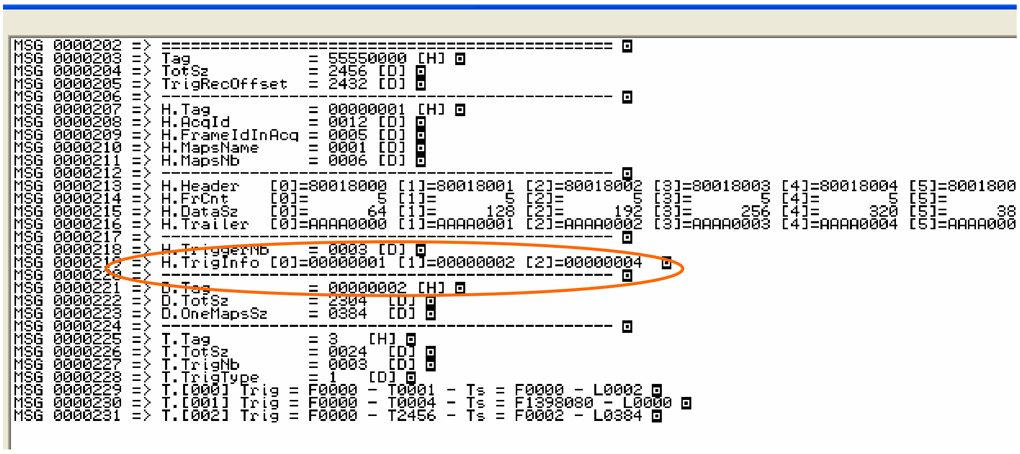
	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	5	5	5	5	5	5
D lenght	64	128	192	256	320	384

		[0]	[1]	[2]	[Last]
Trig nb	3				
Trigger		F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp		F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No 0 Frame No 5 Print trig Print

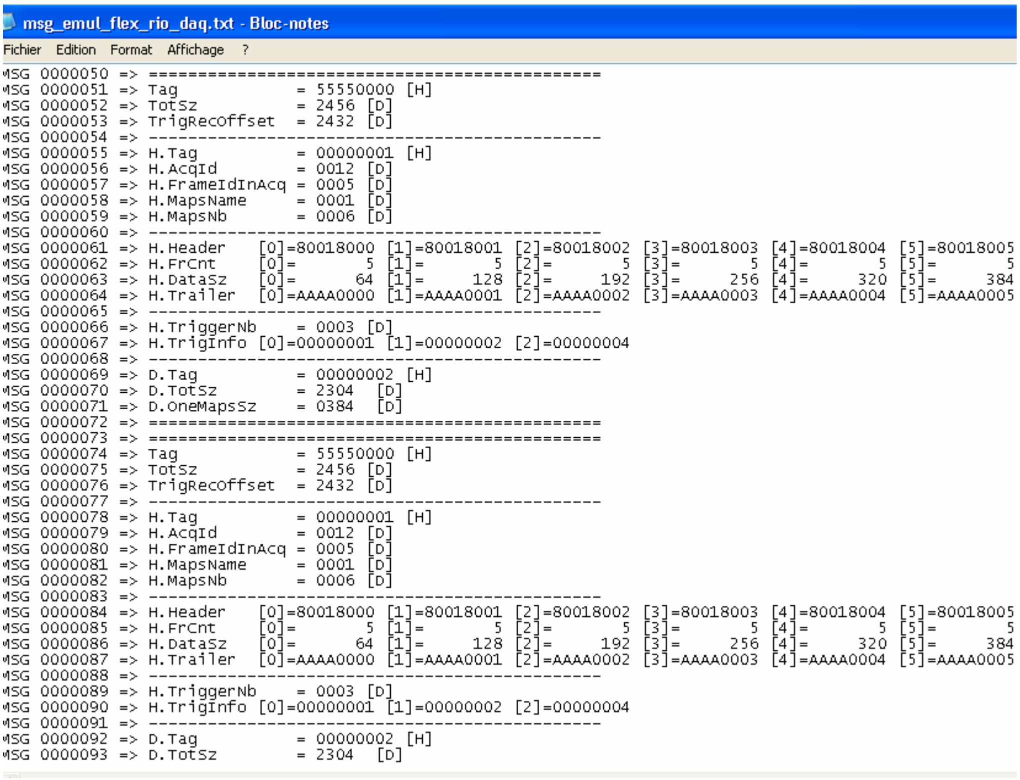
Check mode 0 : No □ Verbose Reset -> Errors cnt 0

We print header + triggers of frame No 5 in log windows



```
MSG 0000202 => ============================================== ◘
MSG 0000203 => Tag              = 55550000 [H] ◘
MSG 0000204 => TotSz            = 2456 [D] ◘
MSG 0000205 => TrigRecOffset    = 2432 [D] ◘
MSG 0000206 => ---------------------------------------------- ◘
MSG 0000207 => H.Tag            = 00000001 [H] ◘
MSG 0000208 => H.AcqId          = 0012 [D] ◘
MSG 0000209 => H.FrameIdInAcq   = 0005 [D] ◘
MSG 0000210 => H.MapsName       = 0001 [D] ◘
MSG 0000211 => H.MapsNb         = 0006 [D] ◘
MSG 0000212 => ---------------------------------------------- ◘
MSG 0000213 => H.Header    [0]=80018000 [1]=80018001 [2]=80018002 [3]=80018003 [4]=80018004 [5]=8001800
MSG 0000214 => H.FrCnt     [0]=       5 [1]=       5 [2]=       5 [3]=       5 [4]=       5 [5]=
MSG 0000215 => H.DataSz    [0]=      64 [1]=     128 [2]=     192 [3]=     256 [4]=     320 [5]=      38
MSG 0000216 => H.Trailer   [0]=AAAA0000 [1]=AAAA0001 [2]=AAAA0002 [3]=AAAA0003 [4]=AAAA0004 [5]=AAAA000
MSG 0000217 => ---------------------------------------------- ◘
MSG 0000218 => H.TriggerNb      = 0003 [D] ◘
MSG 0000219 => H.TrigInfo [0]=00000001 [1]=00000002 [2]=00000004  ◘
MSG 0000220 => ---------------------------------------------- ◘
MSG 0000221 => D.Tag            = 00000002 [H] ◘
MSG 0000222 => D.TotSz          = 2304  [D] ◘
MSG 0000223 => D.OneMapsSz      = 0384  [D] ◘
MSG 0000224 => ---------------------------------------------- ◘
MSG 0000225 => T.Tag            = 3    [H] ◘
MSG 0000226 => T.TotSz          = 0024  [D] ◘
MSG 0000227 => T.TrigNb         = 0003  [D] ◘
MSG 0000228 => T.TrigType       = 1    [D] ◘
MSG 0000229 => T.[000] Trig = F0000 - T0001 - Ts = F0000 - L0002 ◘
MSG 0000230 => T.[001] Trig = F0000 - T0004 - Ts = F1398080 - L0000 ◘
MSG 0000231 => T.[002] Trig = F0000 - T2456 - Ts = F0002 - L0384 ◘
```

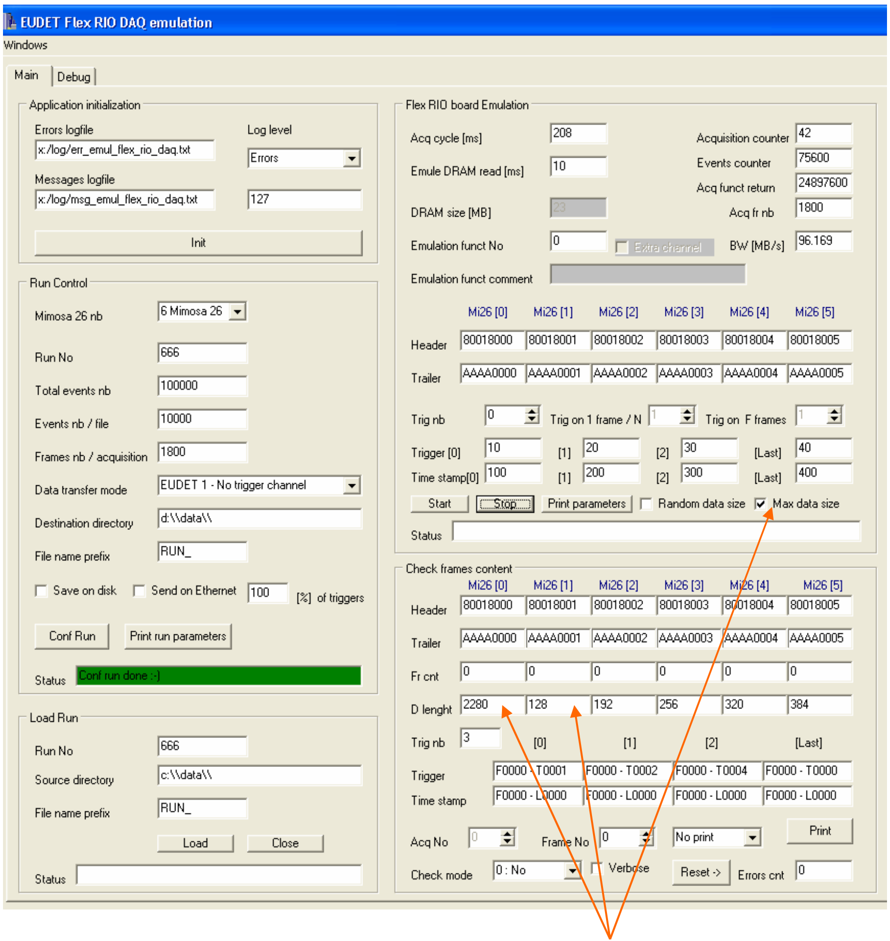
This is the print result


msg_emul_flex_rio_daq.txt - Bloc-notes
Fichier Edition Format Affichage ?

```
MSG 0000050 => ==============================================
MSG 0000051 => Tag             = 55550000 [H]
MSG 0000052 => TotSz           = 2456 [D]
MSG 0000053 => TrigRecOffset   = 2432 [D]
MSG 0000054 => ----------------------------------------------
MSG 0000055 => H.Tag           = 00000001 [H]
MSG 0000056 => H.AcqId         = 0012 [D]
MSG 0000057 => H.FrameIdInAcq  = 0005 [D]
MSG 0000058 => H.MapsName      = 0001 [D]
MSG 0000059 => H.MapsNb        = 0006 [D]
MSG 0000060 => ----------------------------------------------
MSG 0000061 => H.Header    [0]=80018000 [1]=80018001 [2]=80018002 [3]=80018003 [4]=80018004 [5]=80018005
MSG 0000062 => H.FrCnt     [0]=       5 [1]=       5 [2]=       5 [3]=       5 [4]=       5 [5]=       5
MSG 0000063 => H.DataSz    [0]=      64 [1]=     128 [2]=     192 [3]=     256 [4]=     320 [5]=     384
MSG 0000064 => H.Trailer   [0]=AAAA0000 [1]=AAAA0001 [2]=AAAA0002 [3]=AAAA0003 [4]=AAAA0004 [5]=AAAA0005
MSG 0000065 => ----------------------------------------------
MSG 0000066 => H.TriggerNb     = 0003 [D]
MSG 0000067 => H.TrigInfo [0]=00000001 [1]=00000002 [2]=00000004
MSG 0000068 => ----------------------------------------------
MSG 0000069 => D.Tag           = 00000002 [H]
MSG 0000070 => D.TotSz         = 2304  [D]
MSG 0000071 => D.OneMapsSz     = 0384  [D]
MSG 0000072 => ==============================================
MSG 0000073 => ==============================================
MSG 0000074 => Tag             = 55550000 [H]
MSG 0000075 => TotSz           = 2456 [D]
MSG 0000076 => TrigRecOffset   = 2432 [D]
MSG 0000077 => ----------------------------------------------
MSG 0000078 => H.Tag           = 00000001 [H]
MSG 0000079 => H.AcqId         = 0012 [D]
MSG 0000080 => H.FrameIdInAcq  = 0005 [D]
MSG 0000081 => H.MapsName      = 0001 [D]
MSG 0000082 => H.MapsNb        = 0006 [D]
MSG 0000083 => ----------------------------------------------
MSG 0000084 => H.Header    [0]=80018000 [1]=80018001 [2]=80018002 [3]=80018003 [4]=80018004 [5]=80018005
MSG 0000085 => H.FrCnt     [0]=       5 [1]=       5 [2]=       5 [3]=       5 [4]=       5 [5]=       5
MSG 0000086 => H.DataSz    [0]=      64 [1]=     128 [2]=     192 [3]=     256 [4]=     320 [5]=     384
MSG 0000087 => H.Trailer   [0]=AAAA0000 [1]=AAAA0001 [2]=AAAA0002 [3]=AAAA0003 [4]=AAAA0004 [5]=AAAA0005
MSG 0000088 => ----------------------------------------------
MSG 0000089 => H.TriggerNb     = 0003 [D]
MSG 0000090 => H.TrigInfo [0]=00000001 [1]=00000002 [2]=00000004
MSG 0000091 => ----------------------------------------------
MSG 0000092 => D.Tag           = 00000002 [H]
MSG 0000093 => D.TotSz         = 2304  [D]
```

The frame is also printed in the log file x:\log\msg_emul_flex_rio_daq.txt.

6.1.2 Maximum frame size



EUDET Flex RIO DAQ emulation

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 1 - No trigger channel

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 23

Emulation funct No: 0 □ Extra channel

Emulation funct comment:

Acquisition counter: 42

Events counter: 75600

Acq funct return: 24897600

Acq fr nb: 1800

BW [MB/s]: 96.169

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 0 Trig on 1 frame / N: 1 Trig on F frames: 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

□ Random data size ☑ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	2280	128	192	256	320	384

Trig nb: 3

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No: 0 Frame No: 0 No print

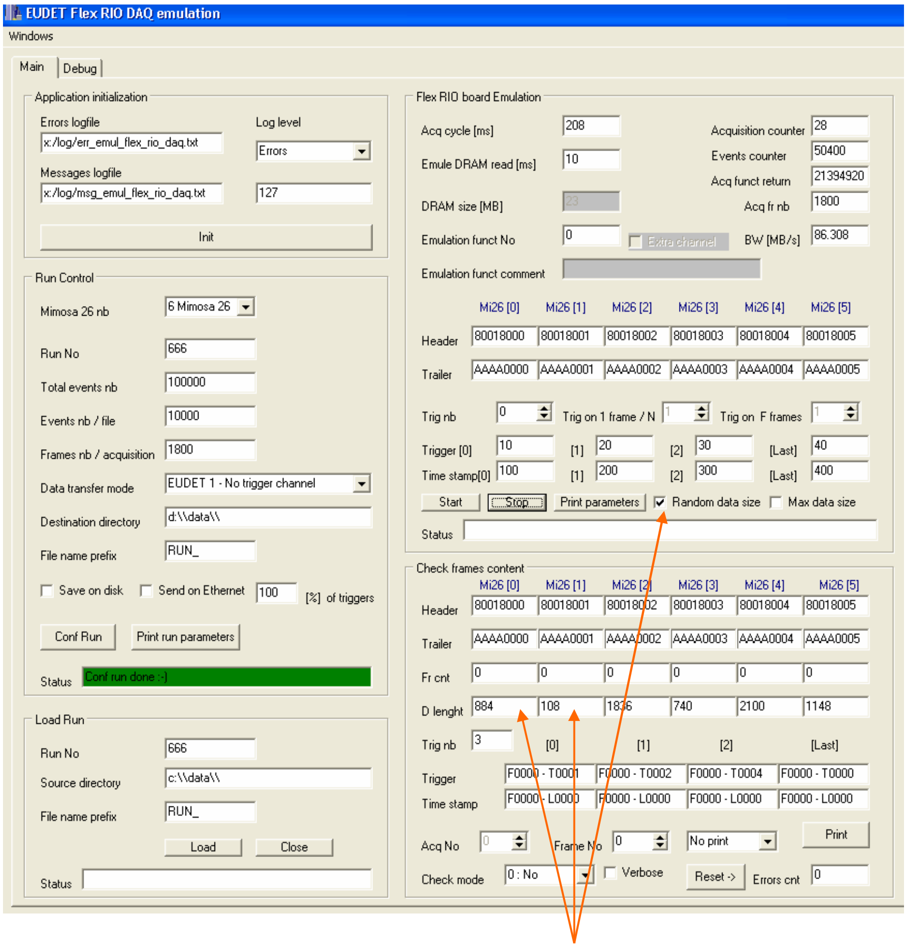
Check mode: 0 : No □ Verbose

Errors cnt: 0

Emulator in mode EUDET 1, 6 Mimosa 26, maximal frame size, three trigger are generated and frame No 0 is displayed.

The first Mimosa 26 has maximum data length (2280 bytes), others keep default data length.

6.1.3 Random frame size



EUDET Flex RIO DAQ emulation

Windows

Main | Debug

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Init

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 1 - No trigger channel

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Conf Run | Print run parameters

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Load | Close

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 23

Emulation funct No: 0 □ Extra channel

Emulation funct comment:

Acquisition counter: 28

Events counter: 50400

Acq funct return: 21394920

Acq fr nb: 1800

BW [MB/s]: 86.308

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 0 Trig on 1 frame / N: 1 Trig on F frames: 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

Start | Stop | Print parameters | ☑ Random data size | □ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	884	108	1836	740	2100	1148

Trig nb: 3

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0001	F0000 - T0002	F0000 - T0004	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No: 0 Frame No: 0 No print | Print

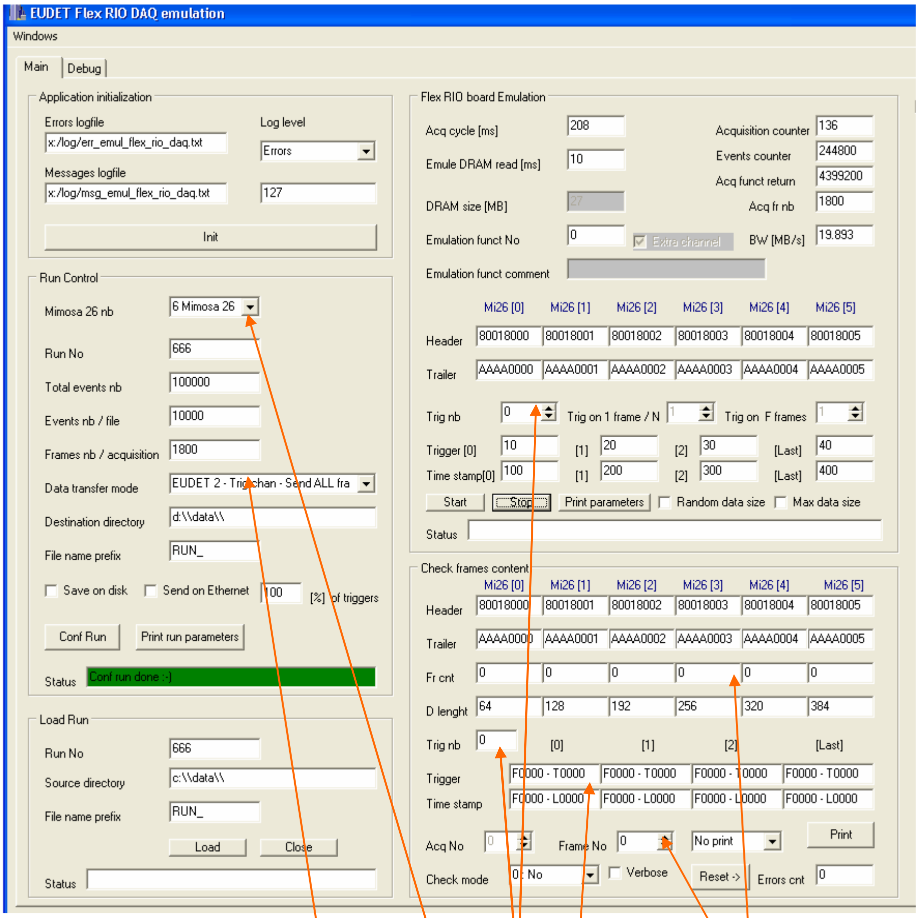
Check mode: 0 : No □ Verbose Reset -> Errors cnt: 0

Emulator in mode EUDET 1, 6 Mimosa 26, random frame size, three trigger are generated and frame No 0 is displayed.

The six Mimosa 26 have a random data length.

6.2 Mode EUDET 2

6.2.1 Default frame size & no trigger



Application initialization

Errors logfile x:/log/err_emul_flex_rio_daq.txt

Log level Errors

Messages logfile x:/log/msg_emul_flex_rio_daq.txt

127

Run Control

Mimosa 26 nb 6 Mimosa 26

Run No 666

Total events nb 100000

Events nb / file 10000

Frames nb / acquisition 1800

Data transfer mode EUDET 2 - Trig chan - Send ALL fra

Destination directory d:\\data\\

File name prefix RUN_

☐ Save on disk ☐ Send on Ethernet 100 [%] of triggers

Status Conf run done :-)

Load Run

Run No 666

Source directory c:\\data\\

File name prefix RUN_

Status

Flex RIO board Emulation

Acq cycle [ms] 208

Emule DRAM read [ms] 10

DRAM size [MB] 27

Emulation funct No 0

☑ Extra channel

Emulation funct comment

Acquisition counter 136

Events counter 244800

Acq funct return 4399200

Acq fr nb 1800

BW [MB/s] 19.893

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb 0 Trig on 1 frame / N 1 Trig on F frames 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

☐ Random data size ☐ Max data size

Status

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	64	128	192	256	320	384

Trig nb 0

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0000	F0000 - T0000	F0000 - T0000	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No 0 Frame No 0 No print

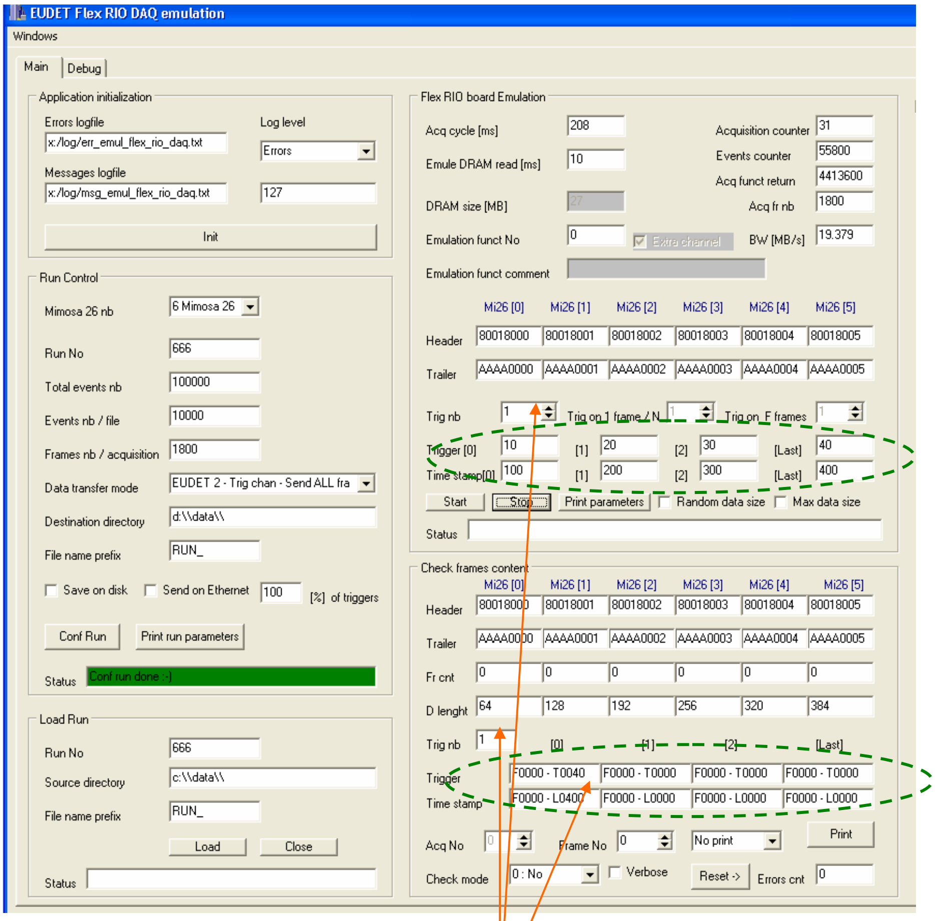
Check mode 0 : No ☐ Verbose Errors cnt 0

Emulator in mode EUDET 2, 6 Mimosa 26, default frame size, no trigger

and frame No 0 is displayed.

If there is no trigger " F0000 – T0000 " is displayed.,

6.2.2 Default frame size & one trigger



EUDET Flex RIO DAQ emulation

Windows

Main | Debug

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt
Log level: Errors
Messages logfile: x:/log/msg_emul_flex_rio_daq.txt
127

Init

Run Control

Mimosa 26 nb: 6 Mimosa 26
Run No: 666
Total events nb: 100000
Events nb / file: 10000
Frames nb / acquisition: 1800
Data transfer mode: EUDET 2 - Trig chan - Send ALL fra
Destination directory: d:\\data\\
File name prefix: RUN_
□ Save on disk □ Send on Ethernet 100 [%] of triggers

Conf Run | Print run parameters

Status: Conf run done :-)

Load Run

Run No: 666
Source directory: c:\\data\\
File name prefix: RUN_

Load | Close

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208
Emule DRAM read [ms]: 10
DRAM size [MB]: 27
Emulation funct No: 0
☑ Extra channel
Emulation funct comment:
Acquisition counter: 31
Events counter: 55800
Acq funct return: 4413600
Acq fr nb: 1800
BW [MB/s]: 19.379

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 1 Trig on 1 frame / N: 1 Trig on F frames: 1

	[0]	[1]	[2]	[Last]
Trigger	10	20	30	40
Time stamp	100	200	300	400

Start | Stop | Print parameters | □ Random data size | □ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	64	128	192	256	320	384

Trig nb: 1

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0040	F0000 - T0000	F0000 - T0000	F0000 - T0000
Time stamp	F0000 - L0400	F0000 - L0000	F0000 - L0000	F0000 - L0000

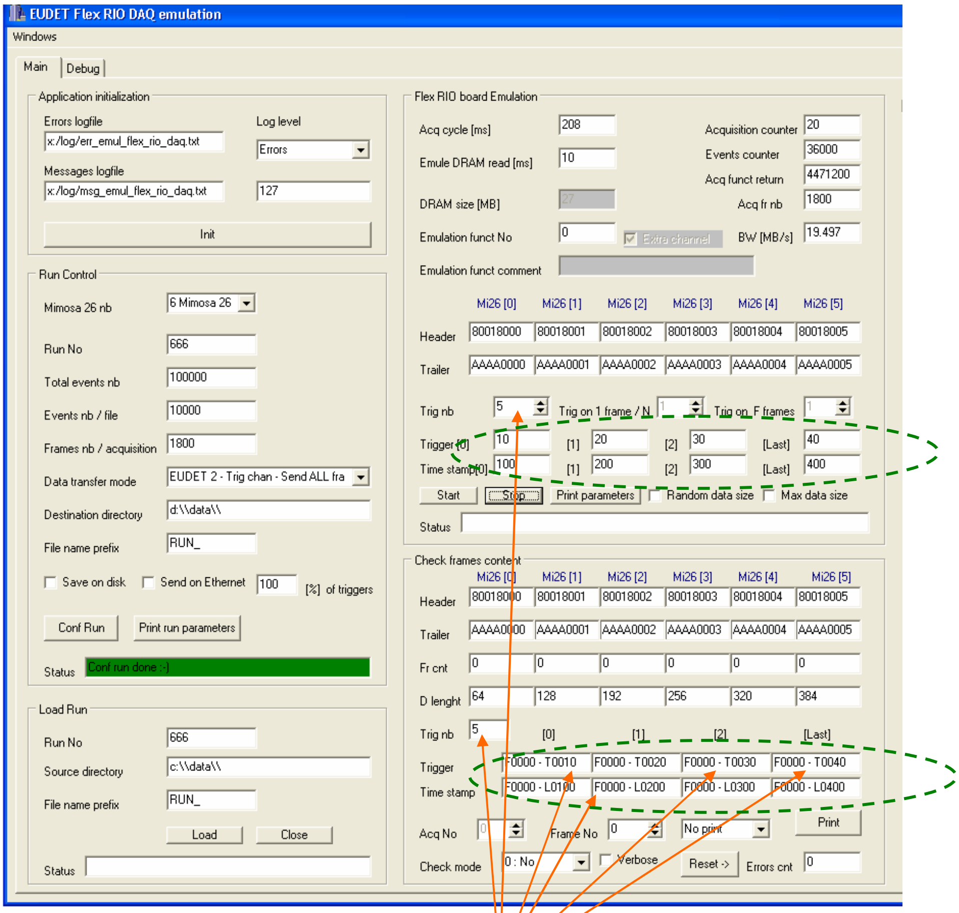
Acq No: 0 Frame No: 0 No print | Print

Check mode: 0 : No □ Verbose | Reset -> | Errors cnt: 0

Emulator in mode EUDET 2, 6 Mimosa 26, default frame size, one trigger generated and frame No 0 is displayed.

The triggers and time stamp values displayed in the bottom panel are the ones configured as emulation parameters in the top panel.

6.2.3 Default frame size & five triggers



Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 2 - Trig chan - Send ALL fra

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 27

Emulation funct No: 0

☑ Extra channel

Emulation funct comment:

Acquisition counter: 20

Events counter: 36000

Acq funct return: 4471200

Acq fr nb: 1800

BW [MB/s]: 19.497

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 5 Trig on 1 frame / N: 1 Trig on F frames: 1

	[0]	[1]	[2]	[Last]
Trigger	10	20	30	40
Time stamp	100	200	300	400

□ Random data size □ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	64	128	192	256	320	384

Trig nb: 5

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0010	F0000 - T0020	F0000 - T0030	F0000 - T0040
Time stamp	F0000 - L0100	F0000 - L0200	F0000 - L0300	F0000 - L0400

Acq No: 0 Frame No: 0 No print

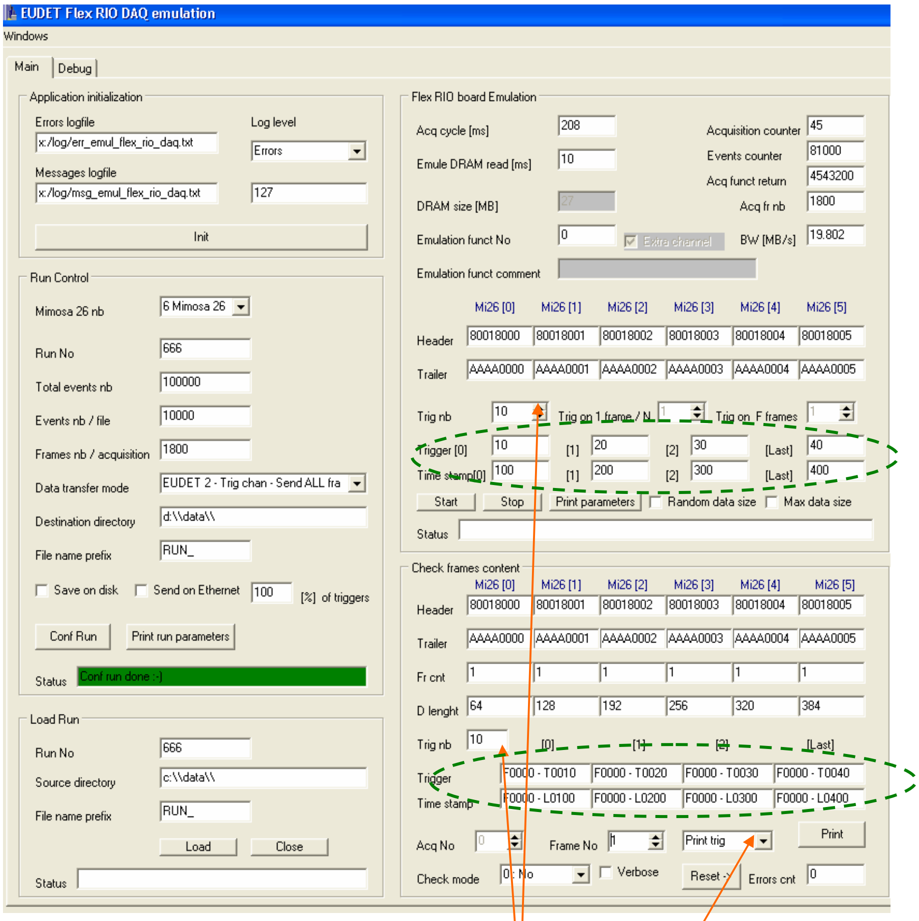
Check mode: 0 : No □ Verbose Errors cnt: 0

Emulator in mode EUDET 2, 6 Mimosa 26, default frame size, five triggers and frame No 0 is displayed.

The triggers and time stamp values displayed in the bottom panel are the ones configured as emulation parameters in the top panel.

The GUI displayed the first three triggers plus the last one of the five.

6.2.4 Default frame size & ten triggers



EUDET Flex RIO DAQ emulation

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 2 - Trig chan - Send ALL fra

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 27

Emulation funct No: 0

☑ Extra channel

Emulation funct comment:

Acquisition counter: 45

Events counter: 81000

Acq funct return: 4543200

Acq fr nb: 1800

BW [MB/s]: 19.802

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 10 Trig on 1 frame / N: 1 Trig on F frames: 1

	[0]	[1]	[2]	[Last]
Trigger	10	20	30	40
Time stamp	100	200	300	400

□ Random data size □ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	1	1	1	1	1	1
D lenght	64	128	192	256	320	384

Trig nb: 10

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0010	F0000 - T0020	F0000 - T0030	F0000 - T0040
Time stamp	F0000 - L0100	F0000 - L0200	F0000 - L0300	F0000 - L0400

Acq No: 0 Frame No: 1 Print trig

Check mode: 0 : No □ Verbose

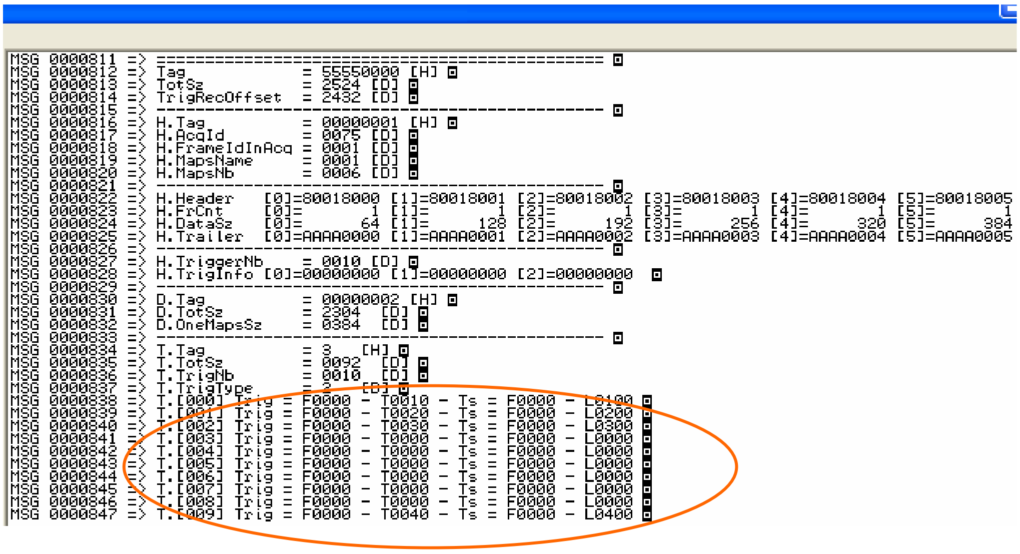
Errors cnt: 0

Emulator in mode EUDET 2, 6 Mimosa 26, default frame size, ten triggers and frame No 0 is displayed.

The triggers and time stamp values displayed in the bottom panel are the ones configured as emulation parameters in the top panel.

Printing of frame header and trigger list is selected.

6.2.5 Print result.

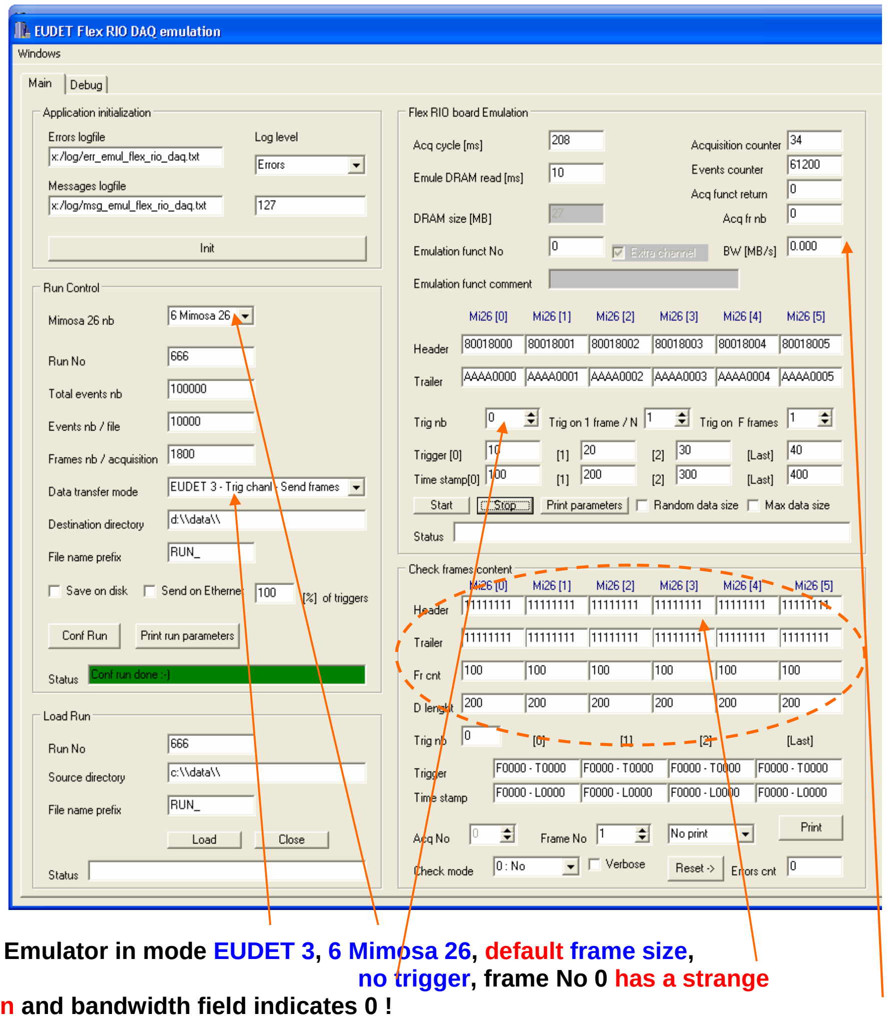


```
MSG 0000811 => ============================================== ▣
MSG 0000812 => Tag            = 55550000 [H] ▣
MSG 0000813 => TotSz          = 2524 [D] ▣
MSG 0000814 => TrigRecOffset  = 2432 [D] ▣
MSG 0000815 => ---------------------------------------------- ▣
MSG 0000816 => H.Tag          = 00000001 [H] ▣
MSG 0000817 => H.AcqId        = 0075 [D] ▣
MSG 0000818 => H.FrameIdInAcq = 0001 [D] ▣
MSG 0000819 => H.MapsName     = 0001 [D] ▣
MSG 0000820 => H.MapsNb       = 0006 [D] ▣
MSG 0000821 => ---------------------------------------------- ▣
MSG 0000822 => H.Header   [0]=80018000 [1]=80018001 [2]=80018002 [3]=80018003 [4]=80018004 [5]=80018005
MSG 0000823 => H.FrCnt    [0]=       1 [1]=       1 [2]=       1 [3]=       1 [4]=       1 [5]=       1
MSG 0000824 => H.DataSz   [0]=      64 [1]=     128 [2]=     192 [3]=     256 [4]=     320 [5]=     384
MSG 0000825 => H.Trailer  [0]=AAAA0000 [1]=AAAA0001 [2]=AAAA0002 [3]=AAAA0003 [4]=AAAA0004 [5]=AAAA0005
MSG 0000826 => ---------------------------------------------- ▣
MSG 0000827 => H.TriggerNb    = 0010 [D] ▣
MSG 0000828 => H.TrigInfo [0]=00000000 [1]=00000000 [2]=00000000  ▣
MSG 0000829 => ---------------------------------------------- ▣
MSG 0000830 => D.Tag          = 00000002 [H] ▣
MSG 0000831 => D.TotSz        = 2304  [D] ▣
MSG 0000832 => D.OneMapsSz    = 0384  [D] ▣
MSG 0000833 => ---------------------------------------------- ▣
MSG 0000834 => T.Tag          = 3    [H] ▣
MSG 0000835 => T.TotSz        = 0092  [D] ▣
MSG 0000836 => T.TrigNb       = 0010  [D] ▣
MSG 0000837 => T.TrigType     = 2    [D] ▣
MSG 0000838 => T.[000] Trig = F0000 - T0010 - Ts = F0000 - L0100 ▣
MSG 0000839 => T.[001] Trig = F0000 - T0020 - Ts = F0000 - L0200 ▣
MSG 0000840 => T.[002] Trig = F0000 - T0030 - Ts = F0000 - L0300 ▣
MSG 0000841 => T.[003] Trig = F0000 - T0000 - Ts = F0000 - L0000 ▣
MSG 0000842 => T.[004] Trig = F0000 - T0000 - Ts = F0000 - L0000 ▣
MSG 0000843 => T.[005] Trig = F0000 - T0000 - Ts = F0000 - L0000 ▣
MSG 0000844 => T.[006] Trig = F0000 - T0000 - Ts = F0000 - L0000 ▣
MSG 0000845 => T.[007] Trig = F0000 - T0000 - Ts = F0000 - L0000 ▣
MSG 0000846 => T.[008] Trig = F0000 - T0000 - Ts = F0000 - L0000 ▣
MSG 0000847 => T.[009] Trig = F0000 - T0040 - Ts = F0000 - L0400 ▣
```

We can see the trigger list, first three and last one are set with the values configured in GUI, others are set to0.

6.3 Mode EUDET 3

6.3.1 Default frame size & no trigger



EUDET Flex RIO DAQ emulation

Windows

Main | Debug

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Init

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 3 - Trig chanl - Send frames

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Conf Run | Print run parameters

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Load | Close

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 27

Emulation funct No: 0 ☑ Extra channel

Emulation funct comment:

Acquisition counter: 34

Events counter: 61200

Acq funct return: 0

Acq fr nb: 0

BW [MB/s]: 0.000

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 0 Trig on 1 frame / N: 1 Trig on F frames: 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

Start | Stop | Print parameters | □ Random data size □ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	11111111	11111111	11111111	11111111	11111111	11111111
Trailer	11111111	11111111	11111111	11111111	11111111	11111111
Fr cnt	100	100	100	100	100	100
D lenght	200	200	200	200	200	200

Trig nb: 0

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0000	F0000 - T0000	F0000 - T0000	F0000 - T0000
Time stamp	F0000 - L0000	F0000 - L0000	F0000 - L0000	F0000 - L0000

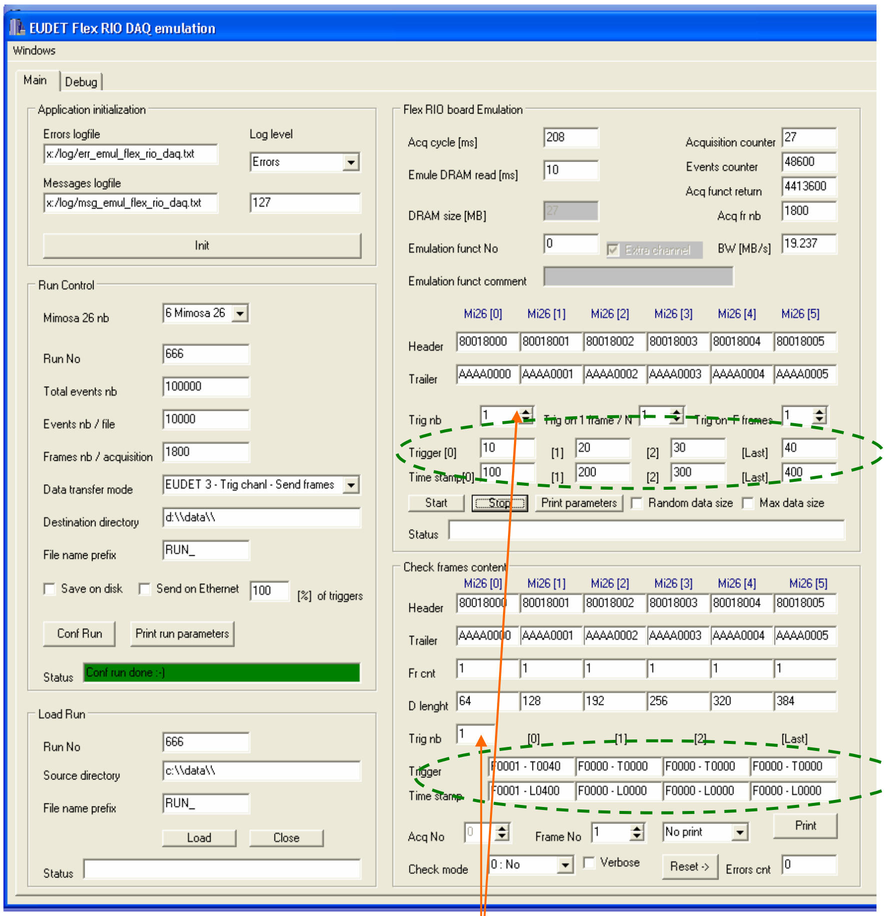
Acq No: 0 Frame No: 1 No print | Print

Check mode: 0 : No □ Verbose | Reset -> | Errors cnt: 0

Emulator in mode EUDET 3, 6 Mimosa 26, default frame size, no trigger, frame No 0 has a strange pattern and bandwidth field indicates 0 !

In EUDET mode 3, only frames with trigger are read. Therefore as there is no trigger, there are no frames, a default pattern is displayed for frame 0 and bandwidth is 0 because there is no data transfer.

6.3.2 Default frame size & one trigger



Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 3 - Trig chanl - Send frames

Destination directory: d:\\data\\

File name prefix: RUN_

☐ Save on disk ☐ Send on Ethernet 100 [%] of triggers

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 27

Emulation funct No: 0 ☑ Extra channel

Emulation funct comment:

Acquisition counter: 27

Events counter: 48600

Acq funct return: 4413600

Acq fr nb: 1800

BW [MB/s]: 19.237

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 1 Trig on 1 frame / N: 1 Trig on F frames: 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

☐ Random data size ☐ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	1	1	1	1	1	1
D lenght	64	128	192	256	320	384

Trig nb: 1

	[0]	[1]	[2]	[Last]
Trigger	F0001 - T0040	F0000 - T0000	F0000 - T0000	F0000 - T0000
Time stamp	F0001 - L0400	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No: 0 Frame No: 1 No print

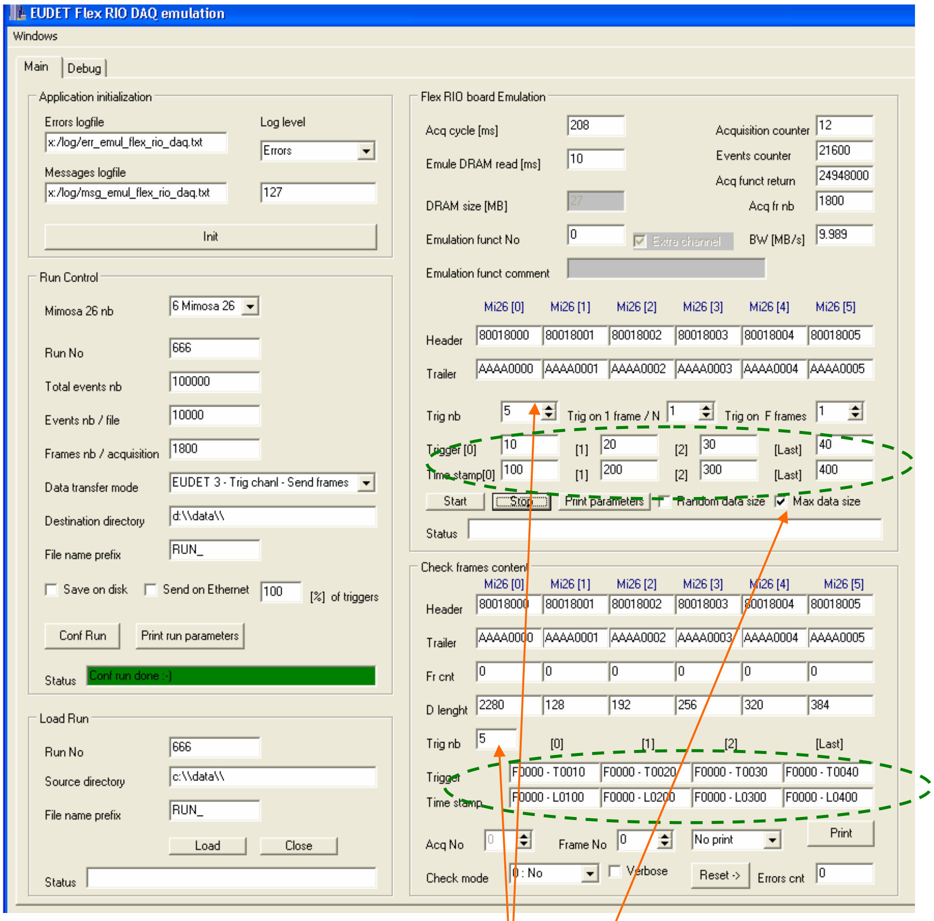
Check mode: 0 : No ☐ Verbose Errors cnt: 0

Emulator in mode EUDET 3, 6 Mimosa 26, default frame size, one trigger generated and frame No 0 is displayed.

Now there is one trigger, bandwidth is not zero and frame No is displayed.

The triggers and time stamp values displayed in the bottom panel are the ones configured as emulation parameters in the top panel.

6.3.3 Maximum frame size & five triggers



EUDET Flex RIO DAQ emulation

Windows

Main | Debug

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Init

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 3 - Trig chanl - Send frames

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Conf Run | Print run parameters

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Load | Close

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 27

Emulation funct No: 0

☑ Extra channel

Emulation funct comment:

Acquisition counter: 12

Events counter: 21600

Acq funct return: 24948000

Acq fr nb: 1800

BW [MB/s]: 9.989

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 5 | Trig on 1 frame / N: 1 | Trig on F frames: 1

		[1]	[2]	[Last]
Trigger [0]	10	20	30	40
Time stamp[0]	100	200	300	400

Start | Stop | Print parameters | □ Random data size | ☑ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	2280	128	192	256	320	384

Trig nb: 5

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0010	F0000 - T0020	F0000 - T0030	F0000 - T0040
Time stamp	F0000 - L0100	F0000 - L0200	F0000 - L0300	F0000 - L0400

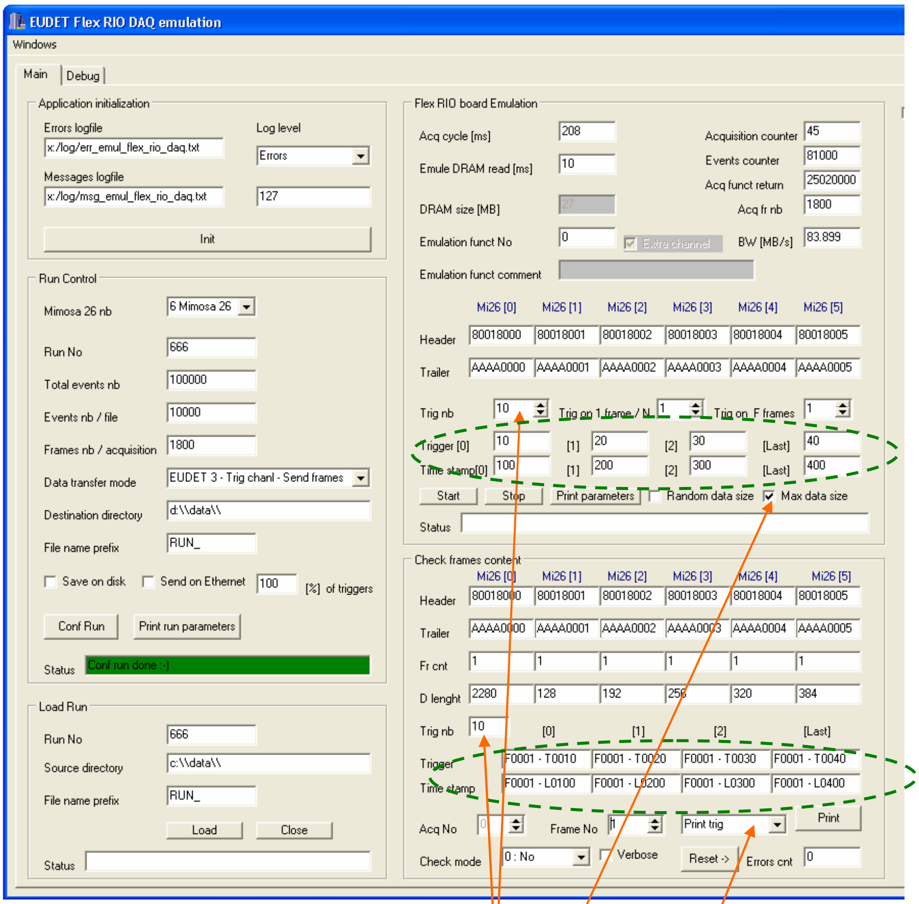
Acq No: 0 | Frame No: 0 | No print | Print

Check mode: 0 : No | □ Verbose | Reset -> | Errors cnt: 0

Emulator in mode EUDET 3, 6 Mimosa 26, maximum frame size, five triggers generated and frame No 0 is displayed.

The triggers and time stamp values displayed in the bottom panel are the ones configured as emulation parameters in the top panel.

6.3.4 Maximum frame size & ten triggers



Application initialization
Errors logfile: x:/log/err_emul_flex_rio_daq.txt
Log level: Errors
Messages logfile: x:/log/msg_emul_flex_rio_daq.txt
127
Init

Run Control
Mimosa 26 nb: 6 Mimosa 26
Run No: 666
Total events nb: 100000
Events nb / file: 10000
Frames nb / acquisition: 1800
Data transfer mode: EUDET 3 - Trig chanl - Send frames
Destination directory: d:\\data\\
File name prefix: RUN_
□ Save on disk □ Send on Ethernet 100 [%] of triggers
Conf Run | Print run parameters
Status: Conf run done :-)

Load Run
Run No: 666
Source directory: c:\\data\\
File name prefix: RUN_
Load | Close
Status:

Flex RIO board Emulation
Acq cycle [ms]: 208
Emule DRAM read [ms]: 10
DRAM size [MB]: 27
Emulation funct No: 0
☑ Extra channel
Emulation funct comment:
Acquisition counter: 45
Events counter: 81000
Acq funct return: 25020000
Acq fr nb: 1800
BW [MB/s]: 83.899

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 10 | Trig on 1 frame / N: 1 | Trig on F frames: 1

	[0]		[1]		[2]		[Last]
Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

Start | Stop | Print parameters | □ Random data size | ☑ Max data size
Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	1	1	1	1	1	1
D lenght	2280	128	192	256	320	384

Trig nb: 10

	[0]	[1]	[2]	[Last]
Trigger	F0001 - T0010	F0001 - T0020	F0001 - T0030	F0001 - T0040
Time stamp	F0001 - L0100	F0001 - L0200	F0001 - L0300	F0001 - L0400

Acq No: 0 | Frame No: 1 | Print trig | Print
Check mode: 0 : No | □ Verbose | Reset -> | Errors cnt: 0

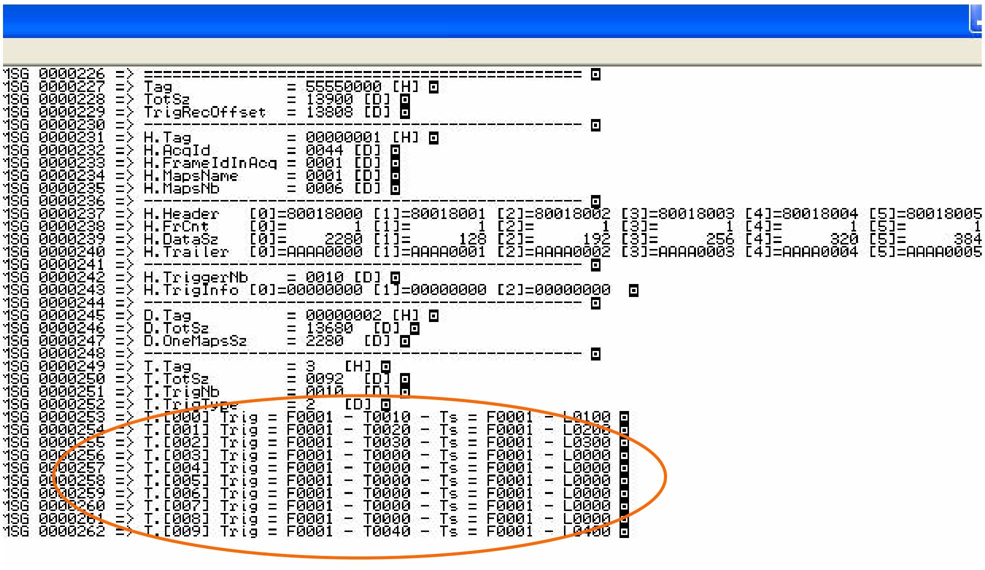
Emulator in mode EUDET 3, 6 Mimosa 26, maximum frame size, ten triggers generated and frame No 0 is displayed.

Now there is one trigger, bandwidth is not zero and frame No is displayed.

The triggers and time stamp values displayed in the bottom panel are the ones configured as emulation parameters in the top panel.

Printing of frame header and trigger list is selected.

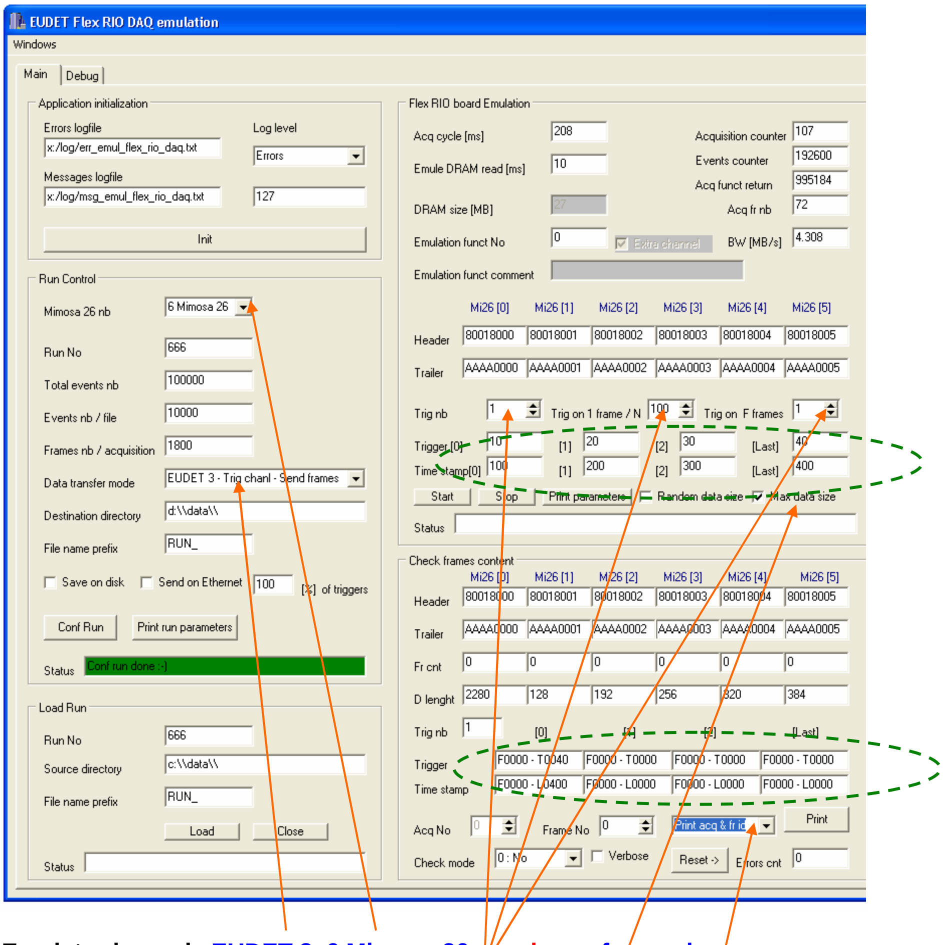
6.3.5 Print result.



```
MSG 0000226 => ============================================== ◘
MSG 0000227 => Tag              = 55550000 [H] ◘
MSG 0000228 => TotSz            = 13900 [D] ◘
MSG 0000229 => TrigRecOffset    = 13808 [D] ◘
MSG 0000230 => ---------------------------------------------- ◘
MSG 0000231 => H.Tag            = 00000001 [H] ◘
MSG 0000232 => H.AcqId          = 0044 [D] ◘
MSG 0000233 => H.FrameIdInAcq   = 0001 [D] ◘
MSG 0000234 => H.MapsName       = 0001 [D] ◘
MSG 0000235 => H.MapsNb         = 0006 [D] ◘
MSG 0000236 => ---------------------------------------------- ◘
MSG 0000237 => H.Header    [0]=80018000 [1]=80018001 [2]=80018002 [3]=80018003 [4]=80018004 [5]=80018005
MSG 0000238 => H.FrCnt     [0]=       1 [1]=       1 [2]=       1 [3]=       1 [4]=       1 [5]=       1
MSG 0000239 => H.DataSz    [0]=    2280 [1]=     128 [2]=     192 [3]=     256 [4]=     320 [5]=     384
MSG 0000240 => H.Trailer   [0]=AAAA0000 [1]=AAAA0001 [2]=AAAA0002 [3]=AAAA0003 [4]=AAAA0004 [5]=AAAA0005
MSG 0000241 => ---------------------------------------------- ◘
MSG 0000242 => H.TriggerNb      = 0010 [D] ◘
MSG 0000243 => H.TrigInfo  [0]=00000000 [1]=00000000 [2]=00000000  ◘
MSG 0000244 => ---------------------------------------------- ◘
MSG 0000245 => D.Tag            = 00000002 [H] ◘
MSG 0000246 => D.TotSz          = 13680  [D] ◘
MSG 0000247 => D.OneMapsSz      = 2280  [D] ◘
MSG 0000248 => ---------------------------------------------- ◘
MSG 0000249 => T.Tag            = 3    [H] ◘
MSG 0000250 => T.TotSz          = 0092  [D] ◘
MSG 0000251 => T.TrigNb         = 0010  [D] ◘
MSG 0000252 => T.TrigType       = 2    [D] ◘
MSG 0000253 => T.[000] Trig = F0001 - T0010 - Ts = F0001 - L0100 ◘
MSG 0000254 => T.[001] Trig = F0001 - T0020 - Ts = F0001 - L0200 ◘
MSG 0000255 => T.[002] Trig = F0001 - T0030 - Ts = F0001 - L0300 ◘
MSG 0000256 => T.[003] Trig = F0001 - T0000 - Ts = F0001 - L0000 ◘
MSG 0000257 => T.[004] Trig = F0001 - T0000 - Ts = F0001 - L0000 ◘
MSG 0000258 => T.[005] Trig = F0001 - T0000 - Ts = F0001 - L0000 ◘
MSG 0000259 => T.[006] Trig = F0001 - T0000 - Ts = F0001 - L0000 ◘
MSG 0000260 => T.[007] Trig = F0001 - T0000 - Ts = F0001 - L0000 ◘
MSG 0000261 => T.[008] Trig = F0001 - T0000 - Ts = F0001 - L0000 ◘
MSG 0000262 => T.[009] Trig = F0001 - T0040 - Ts = F0001 - L0400 ◘
```

We can see the trigger list, first three and last one are set with the values configured in GUI, others are set to0.

6.3.6 Maximum frame size & 1 trigger / 100 frames



EUDET Flex RIO DAQ emulation

Windows

Main | Debug

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Init

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 3 - Trig chanl - Send frames

Destination directory: d:\\data\\

File name prefix: RUN_

□ Save on disk □ Send on Ethernet 100 [%] of triggers

Conf Run | Print run parameters

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Load | Close

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 27

Emulation funct No: 0 ☑ Extra channel

Emulation funct comment:

Acquisition counter: 107

Events counter: 192600

Acq funct return: 995184

Acq fr nb: 72

BW [MB/s]: 4.308

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 1 Trig on 1 frame / N: 100 Trig on F frames: 1

	[0]	[1]	[2]	[Last]
Trigger	10	20	30	40
Time stamp	100	200	300	400

Start | Stop | Print parameters | □ Random data size ☑ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	2280	128	192	256	320	384

Trig nb: 1

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0040	F0000 - T0000	F0000 - T0000	F0000 - T0000
Time stamp	F0000 - L0400	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No: 0 Frame No: 0 Print acq & fr id | Print

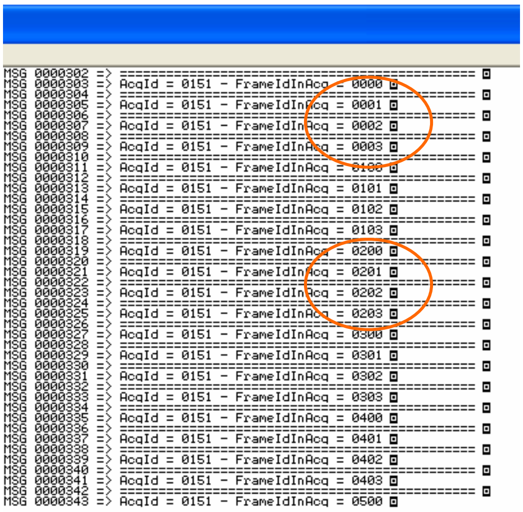
Check mode: 0 : No □ Verbose Reset -> Errors cnt: 0

Emulator in mode **EUDET 3**, **6 Mimosa 26**, **maximum frame size**, **1 trigger / 100 frames** generated and **frame No 0 is displayed**.

The **triggers and time stamp values displayed in the bottom panel are the ones configured as emulation parameters in the top panel.**

Print minimal information → acquisition and frame No.

6.3.7 Print result.



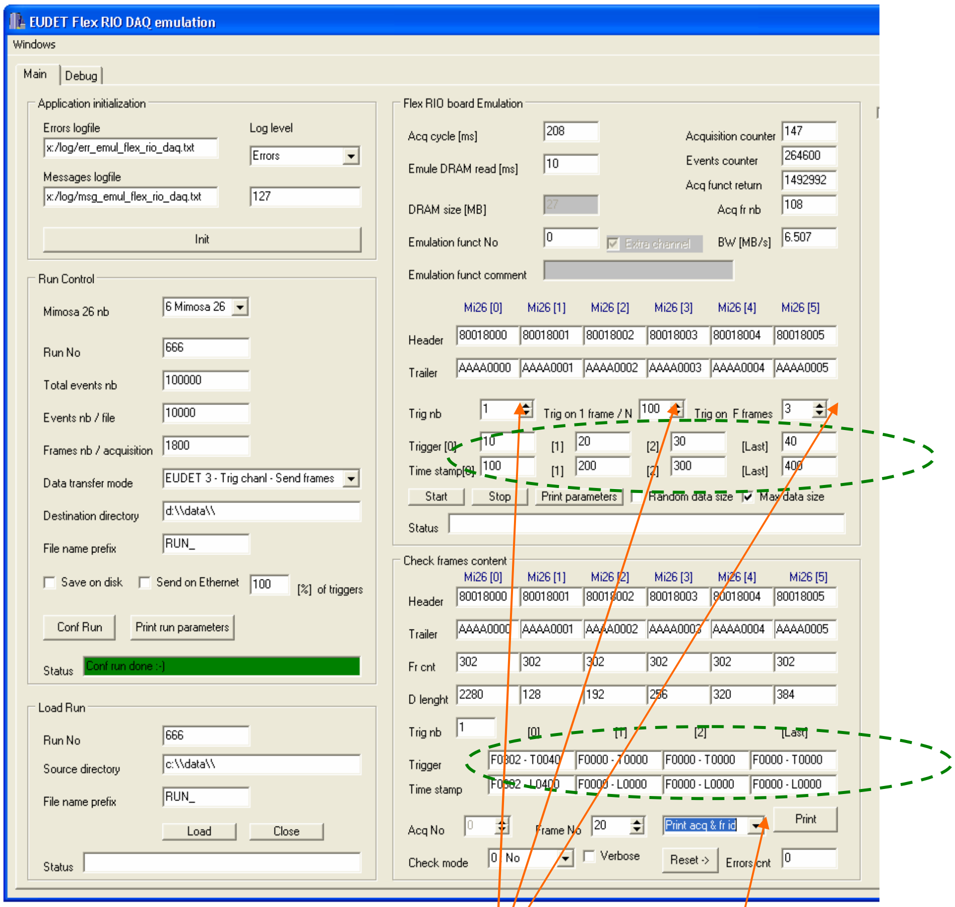
```
MSG 0000302 => ============================================== ▣
MSG 0000303 => AcqId = 0151 - FrameIdInAcq = 0000 ▣
MSG 0000304 => ============================================== ▣
MSG 0000305 => AcqId = 0151 - FrameIdInAcq = 0001 ▣
MSG 0000306 => ============================================== ▣
MSG 0000307 => AcqId = 0151 - FrameIdInAcq = 0002 ▣
MSG 0000308 => ============================================== ▣
MSG 0000309 => AcqId = 0151 - FrameIdInAcq = 0003 ▣
MSG 0000310 => ============================================== ▣
MSG 0000311 => AcqId = 0151 - FrameIdInAcq = 0100 ▣
MSG 0000312 => ============================================== ▣
MSG 0000313 => AcqId = 0151 - FrameIdInAcq = 0101 ▣
MSG 0000314 => ============================================== ▣
MSG 0000315 => AcqId = 0151 - FrameIdInAcq = 0102 ▣
MSG 0000316 => ============================================== ▣
MSG 0000317 => AcqId = 0151 - FrameIdInAcq = 0103 ▣
MSG 0000318 => ============================================== ▣
MSG 0000319 => AcqId = 0151 - FrameIdInAcq = 0200 ▣
MSG 0000320 => ============================================== ▣
MSG 0000321 => AcqId = 0151 - FrameIdInAcq = 0201 ▣
MSG 0000322 => ============================================== ▣
MSG 0000323 => AcqId = 0151 - FrameIdInAcq = 0202 ▣
MSG 0000324 => ============================================== ▣
MSG 0000325 => AcqId = 0151 - FrameIdInAcq = 0203 ▣
MSG 0000326 => ============================================== ▣
MSG 0000327 => AcqId = 0151 - FrameIdInAcq = 0300 ▣
MSG 0000328 => ============================================== ▣
MSG 0000329 => AcqId = 0151 - FrameIdInAcq = 0301 ▣
MSG 0000330 => ============================================== ▣
MSG 0000331 => AcqId = 0151 - FrameIdInAcq = 0302 ▣
MSG 0000332 => ============================================== ▣
MSG 0000333 => AcqId = 0151 - FrameIdInAcq = 0303 ▣
MSG 0000334 => ============================================== ▣
MSG 0000335 => AcqId = 0151 - FrameIdInAcq = 0400 ▣
MSG 0000336 => ============================================== ▣
MSG 0000337 => AcqId = 0151 - FrameIdInAcq = 0401 ▣
MSG 0000338 => ============================================== ▣
MSG 0000339 => AcqId = 0151 - FrameIdInAcq = 0402 ▣
MSG 0000340 => ============================================== ▣
MSG 0000341 => AcqId = 0151 - FrameIdInAcq = 0403 ▣
MSG 0000342 => ============================================== ▣
MSG 0000343 => AcqId = 0151 - FrameIdInAcq = 0500 ▣
```

The frames acquired are 0,1,2,3 – 100,101,102,103 – 200, 201, 202, 203 etc …

We configured emulator to generate one trigger each 100 frames, therefore we should get the frames 0 – 100 – 200 etc … it's the case ☺

But we also get three following frames, that's because we have configured the DAQ to acquire also the three frames following the trigger. This is done by setting the constant EFRIO__FRAME_NB_TO_READ_AFTER_TRIG to 3 in eudet_frio.def file.

6.3.8 Maximum frame size & 3 consecutive triggers / 100 frames



Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt
Log level: Errors
Messages logfile: x:/log/msg_emul_flex_rio_daq.txt
127
Init

Run Control

Mimosa 26 nb: 6 Mimosa 26
Run No: 666
Total events nb: 100000
Events nb / file: 10000
Frames nb / acquisition: 1800
Data transfer mode: EUDET 3 - Trig chanl - Send frames
Destination directory: d:\\data\\
File name prefix: RUN_
□ Save on disk □ Send on Ethernet 100 [%] of triggers
Conf Run | Print run parameters
Status: Conf run done :-)

Load Run

Run No: 666
Source directory: c:\\data\\
File name prefix: RUN_
Load | Close
Status:

Flex RIO board Emulation

Acq cycle [ms]: 208
Emule DRAM read [ms]: 10
DRAM size [MB]: 27
Emulation funct No: 0
☑ Extra channel
Emulation funct comment:
Acquisition counter: 147
Events counter: 264600
Acq funct return: 1492992
Acq fr nb: 108
BW [MB/s]: 6.507

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 1 Trig on 1 frame / N: 100 Trig on F frames: 3

	[0]	[1]	[2]	[Last]
Trigger	10	20	30	40
Time stamp	100	200	300	400

Start | Stop | Print parameters | □ Random data size ☑ Max data size
Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	302	302	302	302	302	302
D lenght	2280	128	192	256	320	384

Trig nb: 1

	[0]	[1]	[2]	[Last]
Trigger	F0302 - T0040	F0000 - T0000	F0000 - T0000	F0000 - T0000
Time stamp	F0302 - L0400	F0000 - L0000	F0000 - L0000	F0000 - L0000

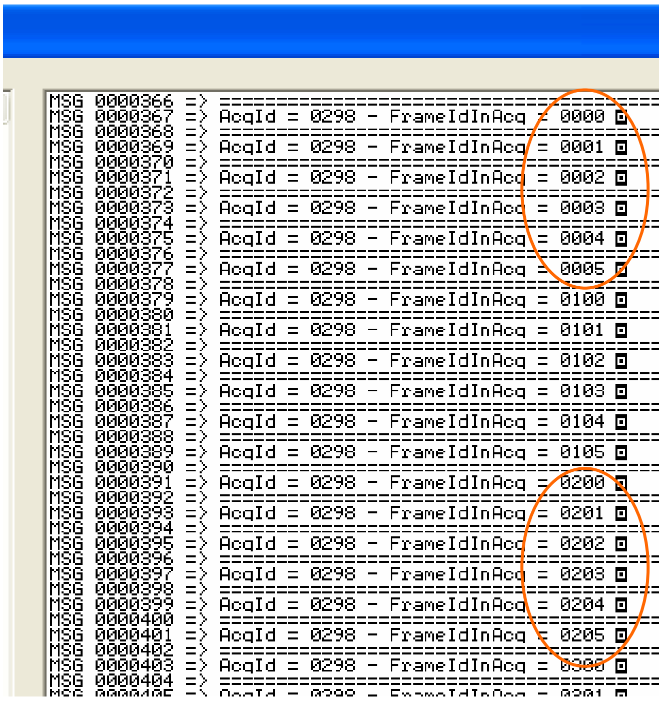
Acq No: 0 Frame No: 20 Print acq & fr id | Print
Check mode: 0 : No □ Verbose Reset -> Errors cnt: 0

Emulator in mode EUDET 3, 6 Mimosa 26, maximum frame size, 3 consecutive triggers / 100 frames generated and frame No 0 is displayed.

The triggers and time stamp values displayed in the bottom panel are the ones configured as emulation parameters in the top panel.

Print minimal information → acquisition and frame No.

6.3.9 Print result.



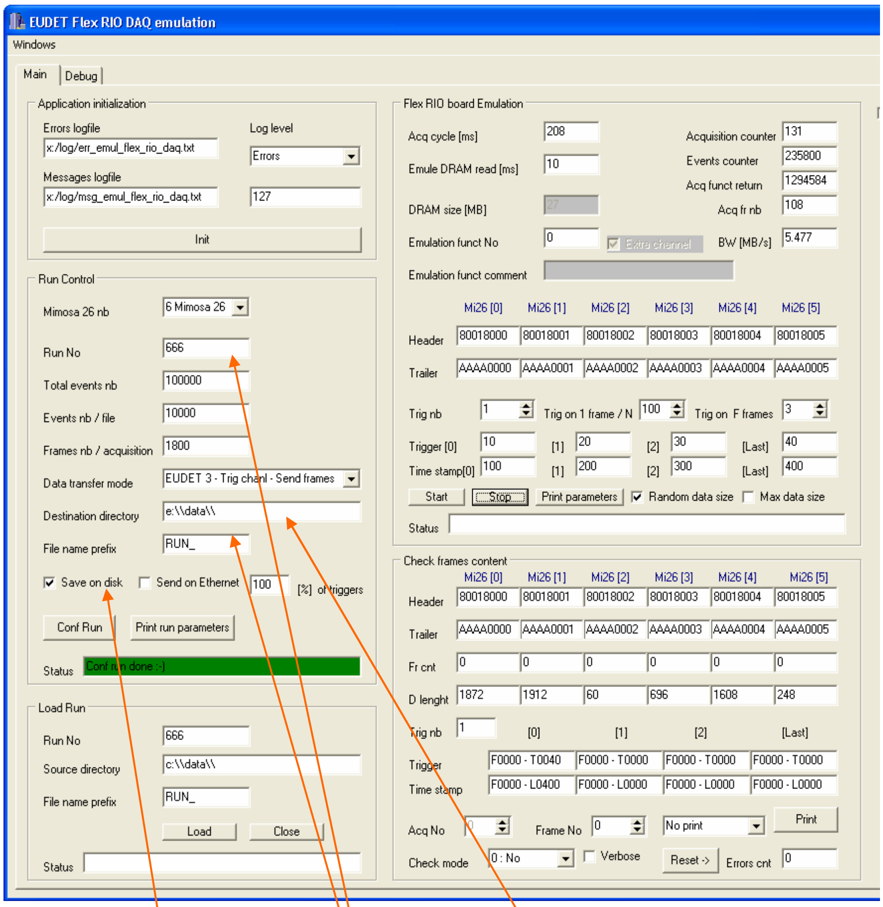
```
MSG 0000366 => ================================================
MSG 0000367 => AcqId = 0298 - FrameIdInAcq = 0000 ◘
MSG 0000368 => ================================================
MSG 0000369 => AcqId = 0298 - FrameIdInAcq = 0001 ◘
MSG 0000370 => ================================================
MSG 0000371 => AcqId = 0298 - FrameIdInAcq = 0002 ◘
MSG 0000372 => ================================================
MSG 0000373 => AcqId = 0298 - FrameIdInAcq = 0003 ◘
MSG 0000374 => ================================================
MSG 0000375 => AcqId = 0298 - FrameIdInAcq = 0004 ◘
MSG 0000376 => ================================================
MSG 0000377 => AcqId = 0298 - FrameIdInAcq = 0005 ◘
MSG 0000378 => ================================================
MSG 0000379 => AcqId = 0298 - FrameIdInAcq = 0100 ◘
MSG 0000380 => ================================================
MSG 0000381 => AcqId = 0298 - FrameIdInAcq = 0101 ◘
MSG 0000382 => ================================================
MSG 0000383 => AcqId = 0298 - FrameIdInAcq = 0102 ◘
MSG 0000384 => ================================================
MSG 0000385 => AcqId = 0298 - FrameIdInAcq = 0103 ◘
MSG 0000386 => ================================================
MSG 0000387 => AcqId = 0298 - FrameIdInAcq = 0104 ◘
MSG 0000388 => ================================================
MSG 0000389 => AcqId = 0298 - FrameIdInAcq = 0105 ◘
MSG 0000390 => ================================================
MSG 0000391 => AcqId = 0298 - FrameIdInAcq = 0200 ◘
MSG 0000392 => ================================================
MSG 0000393 => AcqId = 0298 - FrameIdInAcq = 0201 ◘
MSG 0000394 => ================================================
MSG 0000395 => AcqId = 0298 - FrameIdInAcq = 0202 ◘
MSG 0000396 => ================================================
MSG 0000397 => AcqId = 0298 - FrameIdInAcq = 0203 ◘
MSG 0000398 => ================================================
MSG 0000399 => AcqId = 0298 - FrameIdInAcq = 0204 ◘
MSG 0000400 => ================================================
MSG 0000401 => AcqId = 0298 - FrameIdInAcq = 0205 ◘
MSG 0000402 => ================================================
MSG 0000403 => AcqId = 0298 - FrameIdInAcq = 0300 ◘
MSG 0000404 => ================================================
MSG 0000405 => AcqId = 0298 - FrameIdInAcq = 0301 ◘
```

The frames acquired are 0,1,2,3,4,5 – 100,101,102,103,104,105 – 200, 201, 202, 203,204,205 etc …

We configured emulator to generate three triggers each 100 frames, therefore we should get the frames 0,1,2 – 100,101,102 – 200,201,202 etc … it's the case ☺

But we also get three following frames, that's because we have configured the DAQ to acquire also the three frames following the trigger. This is done by setting the constant EFRIO__FRAME_NB_TO_READ_AFTER_TRIG to 3 in eudet_frio.def file.

6.3.10 Random frame size & 3 consecutive triggers / 100 frames & save to disk



EUDET Flex RIO DAQ emulation

Windows

Main | Debug

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt

Log level: Errors

Messages logfile: x:/log/msg_emul_flex_rio_daq.txt

127

Init

Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

Frames nb / acquisition: 1800

Data transfer mode: EUDET 3 - Trig chanl - Send frames

Destination directory: e:\\data\\

File name prefix: RUN_

☑ Save on disk ☐ Send on Ethernet 100 [%] of triggers

Conf Run | Print run parameters

Status: Conf run done :-)

Load Run

Run No: 666

Source directory: c:\\data\\

File name prefix: RUN_

Load | Close

Status:

Flex RIO board Emulation

Acq cycle [ms]: 208

Emule DRAM read [ms]: 10

DRAM size [MB]: 27

Emulation funct No: 0 ☑ Extra channel

Emulation funct comment:

Acquisition counter: 131

Events counter: 235800

Acq funct return: 1294584

Acq fr nb: 108

BW [MB/s]: 5.477

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 1 Trig on 1 frame / N: 100 Trig on F frames: 3

	[0]	[1]	[2]	[Last]
Trigger	10	20	30	40
Time stamp	100	200	300	400

Start | Stop | Print parameters ☑ Random data size ☐ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	1872	1912	60	696	1608	248

Trig nb: 1

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0040	F0000 - T0000	F0000 - T0000	F0000 - T0000
Time stamp	F0000 - L0400	F0000 - L0000	F0000 - L0000	F0000 - L0000

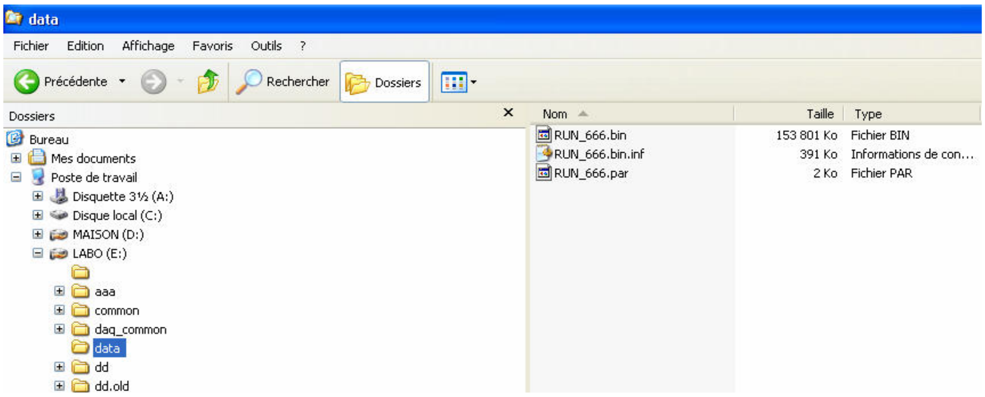
Acq No: 0 Frame No: 0 No print | Print

Check mode: 0 : No ☐ Verbose Reset -> Errors cnt: 0

Emulator in mode EUDET 3, 6 Mimosa 26, maximum frame size, 3 consecutive triggers / 100 frames generated and frame No 0 is displayed.

Saving data to run file RUN_666 in directory e:\data is enabled.

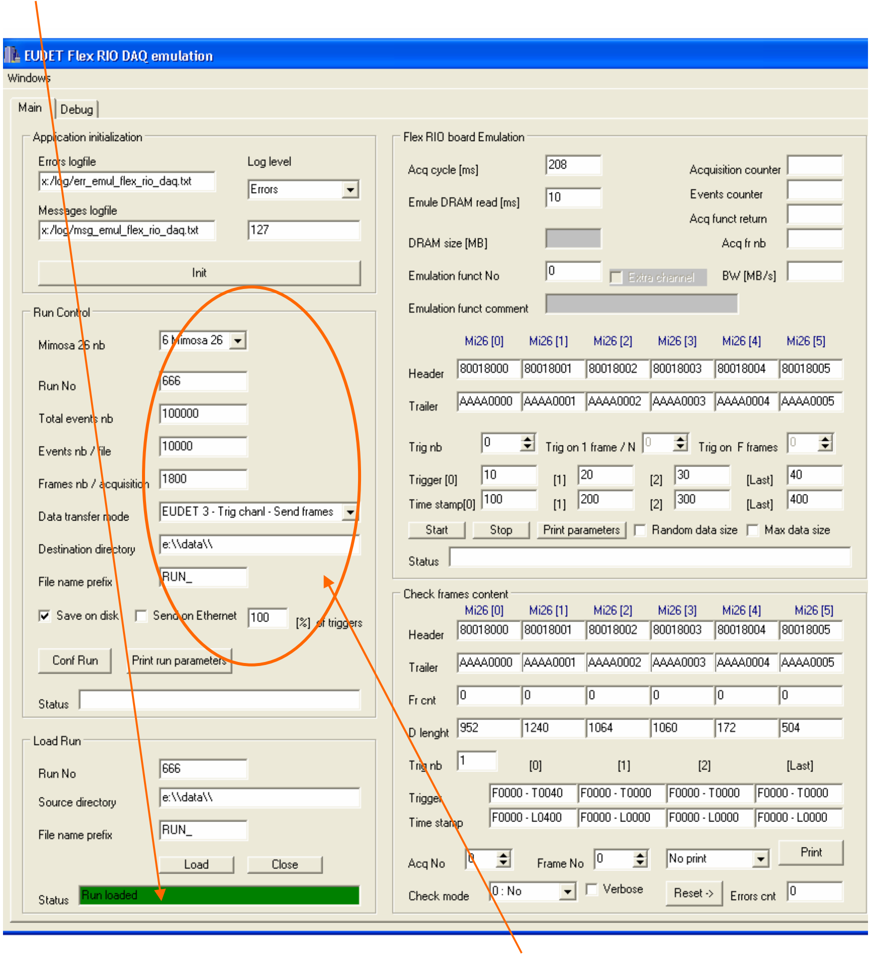
Run file RUN_666.bin created on disk in directory e:\data.



Nom	Taille	Type
RUN_666.bin	153 801 Ko	Fichier BIN
RUN_666.bin.inf	391 Ko	Informations de con…
RUN_666.par	2 Ko	Fichier PAR

6.3.11 Load a run from disk

Load the run file created in 6.3.7 via the panel " Load Run ", if loading is successful the status field switch to green, otherwise it will get red.



EUDET Flex RIO DAQ emulation

Windows

Main | Debug

Application initialization

Errors logfile: x:/log/err_emul_flex_rio_daq.txt
Log level: Errors
Messages logfile: x:/log/msg_emul_flex_rio_daq.txt
127

Init

Run Control

Mimosa 26 nb: 6 Mimosa 26
Run No: 666
Total events nb: 100000
Events nb / file: 10000
Frames nb / acquisition: 1800
Data transfer mode: EUDET 3 - Trig chanl - Send frames
Destination directory: e:\\data\\
File name prefix: RUN_
☑ Save on disk ☐ Send on Ethernet 100 [%] of triggers

Conf Run | Print run parameters

Status:

Load Run

Run No: 666
Source directory: e:\\data\\
File name prefix: RUN_

Load | Close

Status: Run loaded

Flex RIO board Emulation

Acq cycle [ms]: 208
Emule DRAM read [ms]: 10
DRAM size [MB]:
Emulation funct No: 0 ☐ Extra channel
Emulation funct comment:
Acquisition counter:
Events counter:
Acq funct return:
Acq fr nb:
BW [MB/s]:

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb: 0 Trig on 1 frame / N: 0 Trig on F frames: 0

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

Start | Stop | Print parameters | ☐ Random data size ☐ Max data size

Status:

Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	0	0	0	0	0	0
D lenght	952	1240	1064	1060	172	504

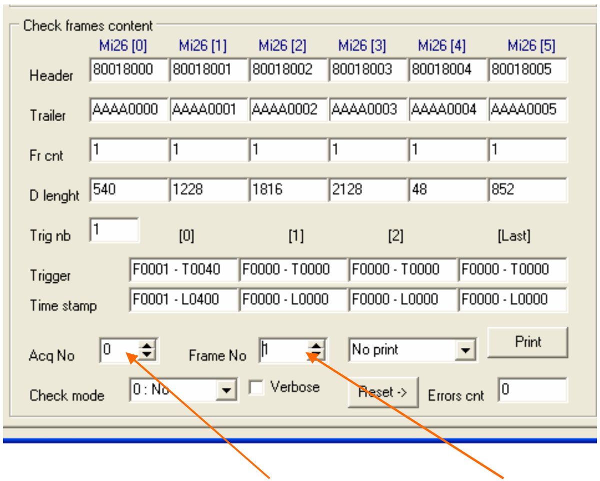
Trig nb: 1

	[0]	[1]	[2]	[Last]
Trigger	F0000 - T0040	F0000 - T0000	F0000 - T0000	F0000 - T0000
Time stamp	F0000 - L0400	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No: 0 Frame No: 0 No print | Print
Check mode: 0 : No ☐ Verbose | Reset -> | Errors cnt: 0

The run parameters, of the loaded file, are displayed in " Run Control " panel.

You can scan frames in the run



Check frames content

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005
Fr cnt	1	1	1	1	1	1
D lenght	540	1228	1816	2128	48	852

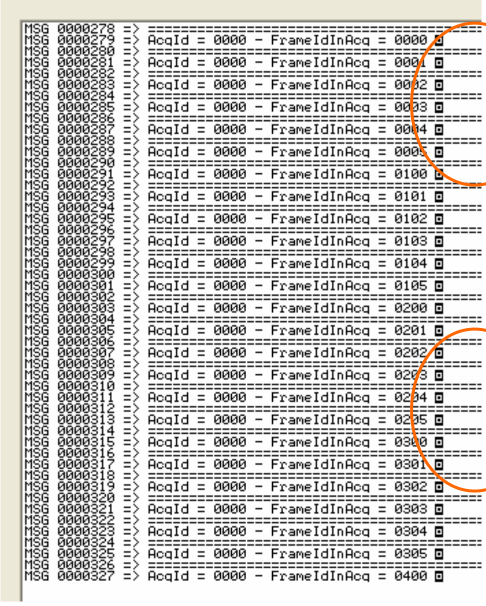
Trig nb 1	[0]	[1]	[2]	[Last]
Trigger	F0001 - T0040	F0000 - T0000	F0000 - T0000	F0000 - T0000
Time stamp	F0001 - L0400	F0000 - L0000	F0000 - L0000	F0000 - L0000

Acq No 0 Frame No 1 No print Print

Check mode 0 : No □ Verbose Reset -> Errors cnt 0

first specify the acquisition No and after the frame No to display

We can also print frames in text mode



```
MSG 0000278 => ========================================
MSG 0000279 => AcqId = 0000 - FrameIdInAcq = 0000 ▣
MSG 0000280 => ========================================
MSG 0000281 => AcqId = 0000 - FrameIdInAcq = 0001 ▣
MSG 0000282 => ========================================
MSG 0000283 => AcqId = 0000 - FrameIdInAcq = 0002 ▣
MSG 0000284 => ========================================
MSG 0000285 => AcqId = 0000 - FrameIdInAcq = 0003 ▣
MSG 0000286 => ========================================
MSG 0000287 => AcqId = 0000 - FrameIdInAcq = 0004 ▣
MSG 0000288 => ========================================
MSG 0000289 => AcqId = 0000 - FrameIdInAcq = 0005 ▣
MSG 0000290 => ========================================
MSG 0000291 => AcqId = 0000 - FrameIdInAcq = 0100 ▣
MSG 0000292 => ========================================
MSG 0000293 => AcqId = 0000 - FrameIdInAcq = 0101 ▣
MSG 0000294 => ========================================
MSG 0000295 => AcqId = 0000 - FrameIdInAcq = 0102 ▣
MSG 0000296 => ========================================
MSG 0000297 => AcqId = 0000 - FrameIdInAcq = 0103 ▣
MSG 0000298 => ========================================
MSG 0000299 => AcqId = 0000 - FrameIdInAcq = 0104 ▣
MSG 0000300 => ========================================
MSG 0000301 => AcqId = 0000 - FrameIdInAcq = 0105 ▣
MSG 0000302 => ========================================
MSG 0000303 => AcqId = 0000 - FrameIdInAcq = 0200 ▣
MSG 0000304 => ========================================
MSG 0000305 => AcqId = 0000 - FrameIdInAcq = 0201 ▣
MSG 0000306 => ========================================
MSG 0000307 => AcqId = 0000 - FrameIdInAcq = 0202 ▣
MSG 0000308 => ========================================
MSG 0000309 => AcqId = 0000 - FrameIdInAcq = 0203 ▣
MSG 0000310 => ========================================
MSG 0000311 => AcqId = 0000 - FrameIdInAcq = 0204 ▣
MSG 0000312 => ========================================
MSG 0000313 => AcqId = 0000 - FrameIdInAcq = 0205 ▣
MSG 0000314 => ========================================
MSG 0000315 => AcqId = 0000 - FrameIdInAcq = 0300 ▣
MSG 0000316 => ========================================
MSG 0000317 => AcqId = 0000 - FrameIdInAcq = 0301 ▣
MSG 0000318 => ========================================
MSG 0000319 => AcqId = 0000 - FrameIdInAcq = 0302 ▣
MSG 0000320 => ========================================
MSG 0000321 => AcqId = 0000 - FrameIdInAcq = 0303 ▣
MSG 0000322 => ========================================
MSG 0000323 => AcqId = 0000 - FrameIdInAcq = 0304 ▣
MSG 0000324 => ========================================
MSG 0000325 => AcqId = 0000 - FrameIdInAcq = 0305 ▣
MSG 0000326 => ========================================
MSG 0000327 => AcqId = 0000 - FrameIdInAcq = 0400 ▣
```

We get the same sequence of frames – 0,1,2,3,4,5 – 100,101,102,103,104,105 etc .. - as the one in 6.3.7 when we took the rum

7 How to interface emulator to EUDET DAQ ?

7.1 Introduction

I never work with EUDET DAQ software, and I never find the time to read all the documentation about it. Therefore I will propose a sketch of interfacing and we will adjust it “ on-line” if it doesn’t fit well or is not applicable …

We can imagine a sequence in four steps :

- **EUDET DAQ send a request to emulator to configure run parameters**
- **EUDET DAQ send a request to start emulation**
- **EUDET DAQ wait for data from emulator**
- **EUDET DAQ send a stop request to stop emulation**

The emulator may have more parameters than EUDET can provide, it would not be a problem → they can be hard coded on emulator side.

In the current version of the software the emulator get his parameters from GUI controls, copy them in global variables named “ context records ” and call eudet_frio library functions to execute actions.

We can add a remote control option, selected via check box on GUI, if it’s enabled all GUI controls no more act as controls but as indicators. They receive theirs values from EUDET DAQ via Ethernet. The GUI “ Run Control ” and “ Flex RIO board emulation ” panels will be simple display of request send by EUDET DAQ.

I think that this approach is close to what we need for the real DAQ control.

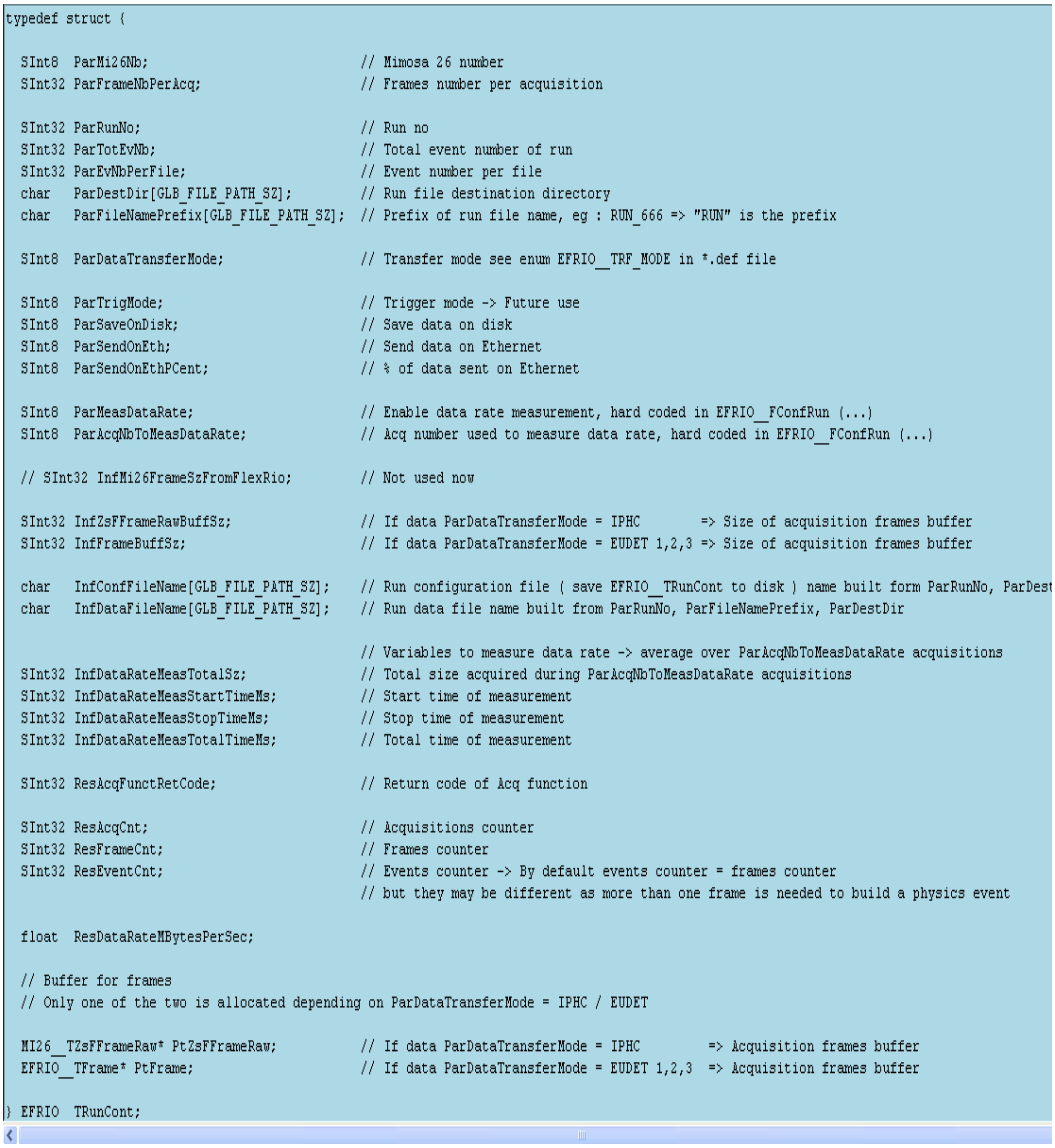
On emulator software point of view we have two directions

- **The input → run control & emulation request**
- **The output → telescope data stream**

7.2 The input side → Run control & Start emulation

7.2.1 Run control context record and configuration function

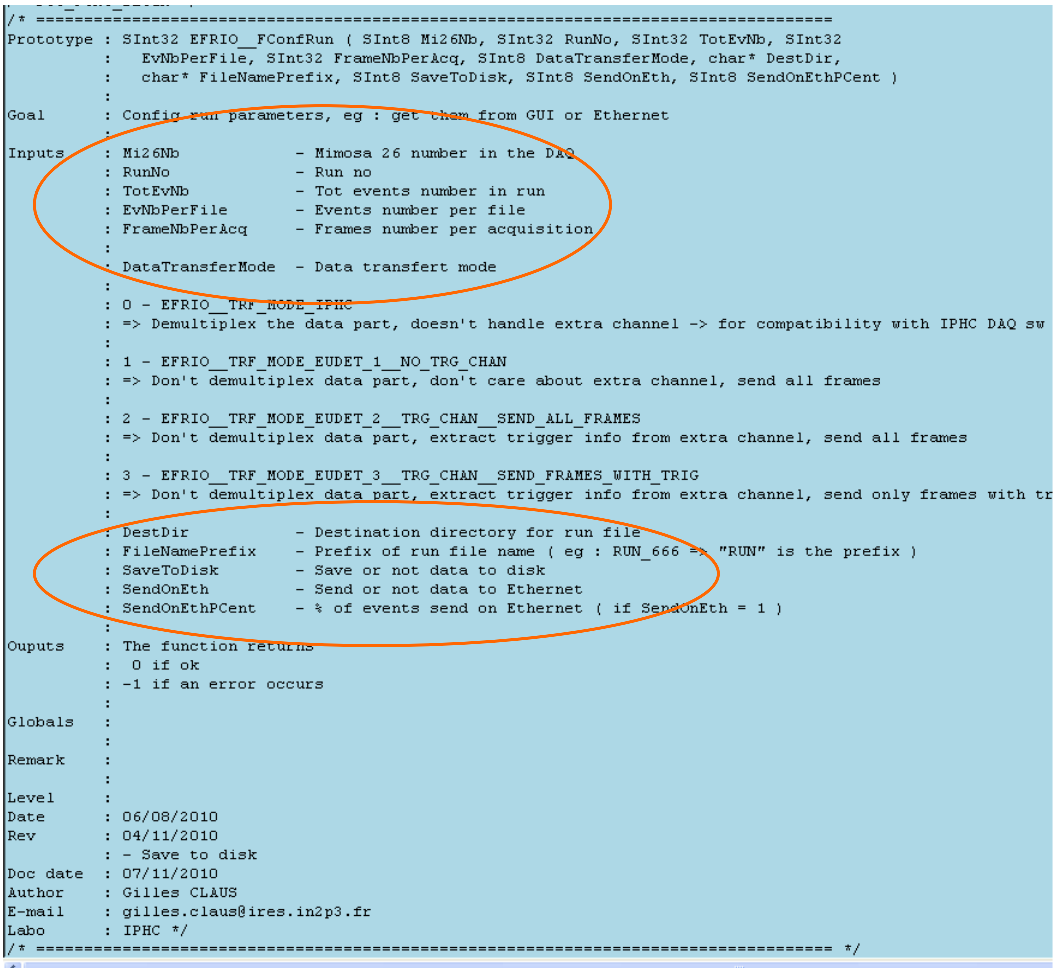
Context record EFRIO__TRunCont → eudet_frio.typ



```
typedef struct {

  SInt8  ParMi26Nb;                           // Mimosa 26 number
  SInt32 ParFrameNbPerAcq;                    // Frames number per acquisition

  SInt32 ParRunNo;                            // Run no
  SInt32 ParTotEvNb;                          // Total event number of run
  SInt32 ParEvNbPerFile;                      // Event number per file
  char   ParDestDir[GLB_FILE_PATH_SZ];        // Run file destination directory
  char   ParFileNamePrefix[GLB_FILE_PATH_SZ]; // Prefix of run file name, eg : RUN_666 => "RUN" is the prefix

  SInt8  ParDataTransferMode;                 // Transfer mode see enum EFRIO__TRF_MODE in *.def file

  SInt8  ParTrigMode;                         // Trigger mode -> Future use
  SInt8  ParSaveOnDisk;                       // Save data on disk
  SInt8  ParSendOnEth;                        // Send data on Ethernet
  SInt8  ParSendOnEthPCent;                   // % of data sent on Ethernet

  SInt8  ParMeasDataRate;                     // Enable data rate measurement, hard coded in EFRIO__FConfRun (...)
  SInt8  ParAcqNbToMeasDataRate;              // Acq number used to measure data rate, hard coded in EFRIO__FConfRun (...)

  // SInt32 InfMi26FrameSzFromFlexRio;        // Not used now

  SInt32 InfZsFFrameRawBuffSz;                // If data ParDataTransferMode = IPHC        => Size of acquisition frames buffer
  SInt32 InfFrameBuffSz;                      // If data ParDataTransferMode = EUDET 1,2,3 => Size of acquisition frames buffer

  char   InfConfFileName[GLB_FILE_PATH_SZ];   // Run configuration file ( save EFRIO__TRunCont to disk ) name built form ParRunNo, ParDest
  char   InfDataFileName[GLB_FILE_PATH_SZ];   // Run data file name built from ParRunNo, ParFileNamePrefix, ParDestDir

                                              // Variables to measure data rate -> average over ParAcqNbToMeasDataRate acquisitions
  SInt32 InfDataRateMeasTotalSz;              // Total size acquired during ParAcqNbToMeasDataRate acquisitions
  SInt32 InfDataRateMeasStartTimeMs;          // Start time of measurement
  SInt32 InfDataRateMeasStopTimeMs;           // Stop time of measurement
  SInt32 InfDataRateMeasTotalTimeMs;          // Total time of measurement

  SInt32 ResAcqFunctRetCode;                  // Return code of Acq function

  SInt32 ResAcqCnt;                           // Acquisitions counter
  SInt32 ResFrameCnt;                         // Frames counter
  SInt32 ResEventCnt;                         // Events counter -> By default events counter = frames counter
                                              // but they may be different as more than one frame is needed to build a physics event

  float  ResDataRateMBytesPerSec;

  // Buffer for frames
  // Only one of the two is allocated depending on ParDataTransferMode = IPHC / EUDET

  MI26__TZsFFrameRaw* PtZsFFrameRaw;          // If data ParDataTransferMode = IPHC         => Acquisition frames buffer
  EFRIO__TFrame* PtFrame;                     // If data ParDataTransferMode = EUDET 1,2,3  => Acquisition frames buffer

} EFRIO  TRunCont;
```

Configuration function EFRIO_FConfRun (…) -→ eudet_frio.c

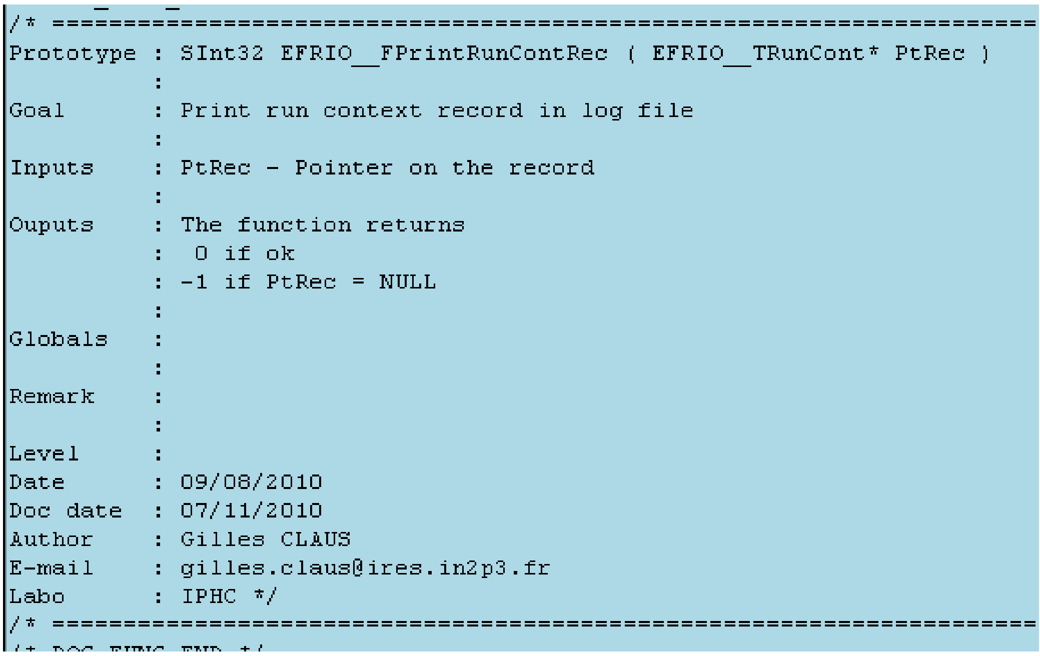
Less parameters than fields on EFRIO__TRunCont ;-)



```
/* ===================================================================================
Prototype : SInt32 EFRIO__FConfRun ( SInt8 Mi26Nb, SInt32 RunNo, SInt32 TotEvNb, SInt32
          :   EvNbPerFile, SInt32 FrameNbPerAcq, SInt8 DataTransferMode, char* DestDir,
          :   char* FileNamePrefix, SInt8 SaveToDisk, SInt8 SendOnEth, SInt8 SendOnEthPCent )
          :
Goal      : Config run parameters, eg : get them from GUI or Ethernet
          :
Inputs    : Mi26Nb            - Mimosa 26 number in the DAQ
          : RunNo             - Run no
          : TotEvNb           - Tot events number in run
          : EvNbPerFile       - Events number per file
          : FrameNbPerAcq     - Frames number per acquisition
          :
          : DataTransferMode  - Data transfert mode
          :
          : 0 - EFRIO__TRF_MODE_IPHC
          : => Demultiplex the data part, doesn't handle extra channel -> for compatibility with IPHC DAQ sw
          :
          : 1 - EFRIO__TRF_MODE_EUDET_1__NO_TRG_CHAN
          : => Don't demultiplex data part, don't care about extra channel, send all frames
          :
          : 2 - EFRIO__TRF_MODE_EUDET_2__TRG_CHAN__SEND_ALL_FRAMES
          : => Don't demultiplex data part, extract trigger info from extra channel, send all frames
          :
          : 3 - EFRIO__TRF_MODE_EUDET_3__TRG_CHAN__SEND_FRAMES_WITH_TRIG
          : => Don't demultiplex data part, extract trigger info from extra channel, send only frames with tr
          :
          : DestDir           - Destination directory for run file
          : FileNamePrefix    - Prefix of run file name ( eg : RUN_666 => "RUN" is the prefix )
          : SaveToDisk        - Save or not data to disk
          : SendOnEth         - Send or not data to Ethernet
          : SendOnEthPCent    - % of events send on Ethernet ( if SendOnEth = 1 )
          :
Ouputs    : The function returns
          :  0 if ok
          : -1 if an error occurs
          :
Globals   :
          :
Remark    :
          :
Level     :
Date      : 06/08/2010
Rev       : 04/11/2010
          : - Save to disk
Doc date  : 07/11/2010
Author    : Gilles CLAUS
E-mail    : gilles.claus@ires.in2p3.fr
Labo      : IPHC */
/* =================================================================================== */
```

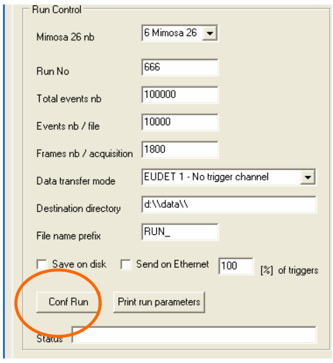
Context printing function EFRIO_FPrintRunContRec (…) -→ eudet_frio_print.c

This function prints run context record in messages window and log file.



```
/* =====================================================================
Prototype : SInt32 EFRIO__FPrintRunContRec ( EFRIO__TRunCont* PtRec )
          :
Goal      : Print run context record in log file
          :
Inputs    : PtRec - Pointer on the record
          :
Ouputs    : The function returns
          :  0 if ok
          : -1 if PtRec = NULL
          :
Globals   :
          :
Remark    :
          :
Level     :
Date      : 09/08/2010
Doc date  : 07/11/2010
Author    : Gilles CLAUS
E-mail    : gilles.claus@ires.in2p3.fr
Labo      : IPHC */
/* =====================================================================
/* DOC_FUNC_END */
```

Part of code called by a click on button “ Conf Run ”



Run Control

Mimosa 26 nb: 6 Mimosa 26

Run No: 666

Total events nb: 100000

Events nb / file: 10000

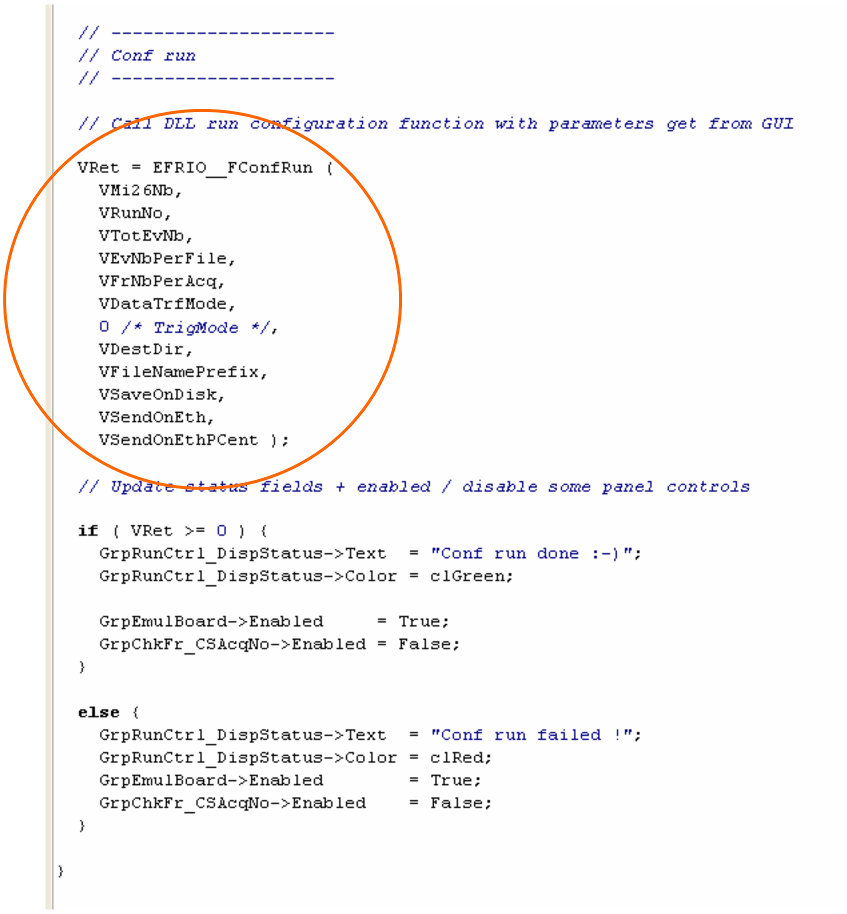
Frames nb / acquisition: 1800

Data transfer mode: EUDET 1 - No trigger channel

Destination directory: d:\\data\\

File name prefix: RUN_

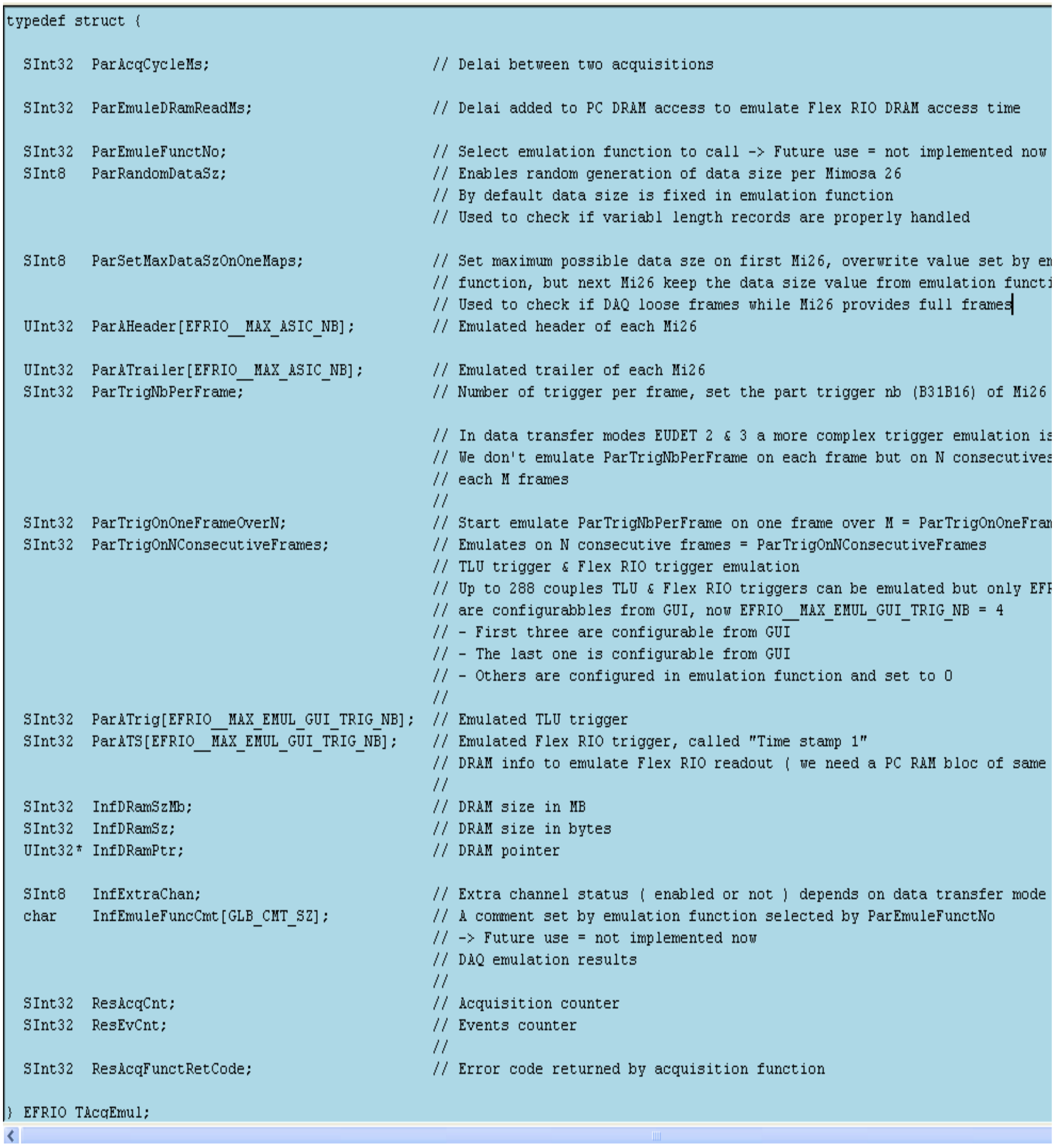
□ Save on disk □ Send on Ethernet 100 [%] of triggers



```
  // ----------------------
  // Conf run
  // ----------------------

  // Call DLL run configuration function with parameters get from GUI

  VRet = EFRIO__FConfRun (
    VMi26Nb,
    VRunNo,
    VTotEvNb,
    VEvNbPerFile,
    VFrNbPerAcq,
    VDataTrfMode,
    0 /* TrigMode */,
    VDestDir,
    VFileNamePrefix,
    VSaveOnDisk,
    VSendOnEth,
    VSendOnEthPCent );

  // Update status fields + enabled / disable some panel controls

  if ( VRet >= 0 ) {
    GrpRunCtrl_DispStatus->Text  = "Conf run done :-)";
    GrpRunCtrl_DispStatus->Color = clGreen;

    GrpEmulBoard->Enabled     = True;
    GrpChkFr_CSAcqNo->Enabled = False;
  }

  else {
    GrpRunCtrl_DispStatus->Text  = "Conf run failed !";
    GrpRunCtrl_DispStatus->Color = clRed;
    GrpEmulBoard->Enabled        = True;
    GrpChkFr_CSAcqNo->Enabled    = False;
  }

}
```

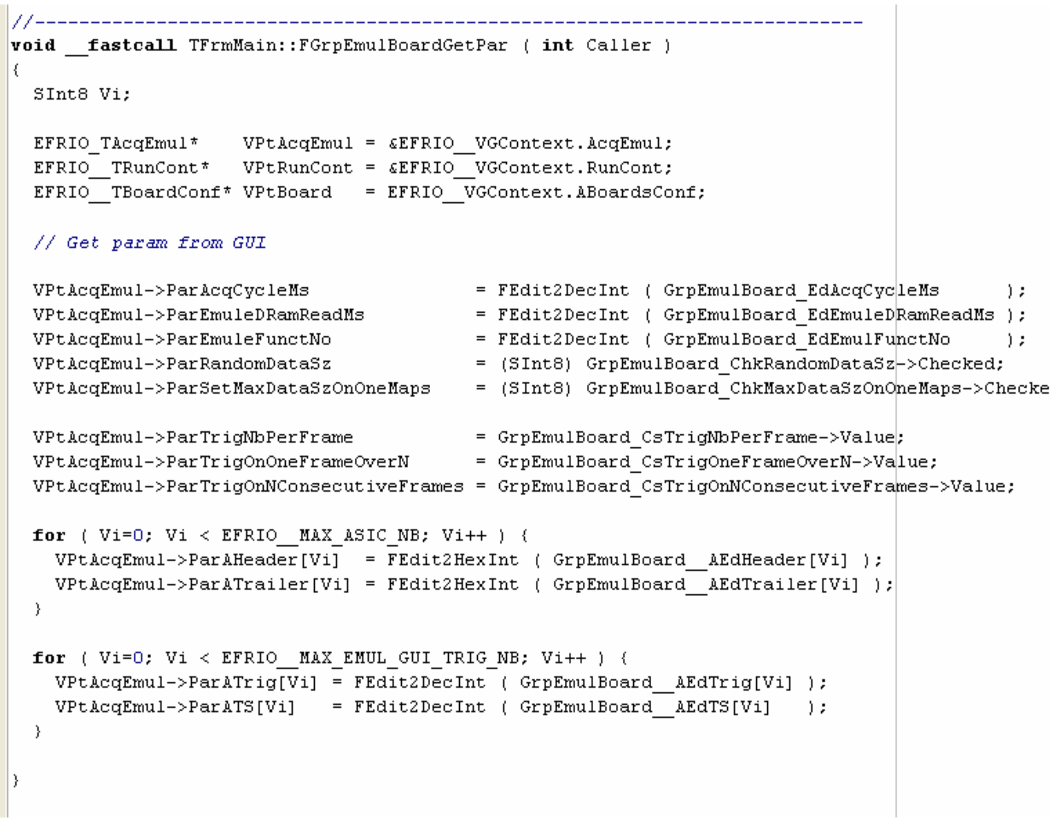
7.2.2 Emulation context record and configuration function

Context record EFRIO__TAcqEmul → eudet_frio.typ



```
typedef struct {

  SInt32  ParAcqCycleMs;                         // Delai between two acquisitions

  SInt32  ParEmuleDRamReadMs;                    // Delai added to PC DRAM access to emulate Flex RIO DRAM access time

  SInt32  ParEmuleFunctNo;                       // Select emulation function to call -> Future use = not implemented now
  SInt8   ParRandomDataSz;                       // Enables random generation of data size per Mimosa 26
                                                 // By default data size is fixed in emulation function
                                                 // Used to check if variabl length records are properly handled

  SInt8   ParSetMaxDataSzOnOneMaps;              // Set maximum possible data sze on first Mi26, overwrite value set by em
                                                 // function, but next Mi26 keep the data size value from emulation functi
                                                 // Used to check if DAQ loose frames while Mi26 provides full frames
  UInt32  ParAHeader[EFRIO__MAX_ASIC_NB];        // Emulated header of each Mi26

  UInt32  ParATrailer[EFRIO__MAX_ASIC_NB];       // Emulated trailer of each Mi26
  SInt32  ParTrigNbPerFrame;                     // Number of trigger per frame, set the part trigger nb (B31B16) of Mi26

                                                 // In data transfer modes EUDET 2 & 3 a more complex trigger emulation is
                                                 // We don't emulate ParTrigNbPerFrame on each frame but on N consecutives
                                                 // each M frames
                                                 //
  SInt32  ParTrigOnOneFrameOverN;                // Start emulate ParTrigNbPerFrame on one frame over M = ParTrigOnOneFram
  SInt32  ParTrigOnNConsecutiveFrames;           // Emulates on N consecutive frames = ParTrigOnNConsecutiveFrames
                                                 // TLU trigger & Flex RIO trigger emulation
                                                 // Up to 288 couples TLU & Flex RIO triggers can be emulated but only EFR
                                                 // are configurabbles from GUI, now EFRIO__MAX_EMUL_GUI_TRIG_NB = 4
                                                 // - First three are configurable from GUI
                                                 // - The last one is configurable from GUI
                                                 // - Others are configured in emulation function and set to 0
                                                 //
  SInt32  ParATrig[EFRIO__MAX_EMUL_GUI_TRIG_NB]; // Emulated TLU trigger
  SInt32  ParATS[EFRIO__MAX_EMUL_GUI_TRIG_NB];   // Emulated Flex RIO trigger, called "Time stamp 1"
                                                 // DRAM info to emulate Flex RIO readout ( we need a PC RAM bloc of same
                                                 //
  SInt32  InfDRamSzMb;                           // DRAM size in MB
  SInt32  InfDRamSz;                             // DRAM size in bytes
  UInt32* InfDRamPtr;                            // DRAM pointer

  SInt8   InfExtraChan;                          // Extra channel status ( enabled or not ) depends on data transfer mode
  char    InfEmuleFuncCmt[GLB_CMT_SZ];           // A comment set by emulation function selected by ParEmuleFunctNo
                                                 // -> Future use = not implemented now
                                                 // DAQ emulation results
                                                 //
  SInt32  ResAcqCnt;                             // Acquisition counter
  SInt32  ResEvCnt;                              // Events counter
                                                 //
  SInt32  ResAcqFunctRetCode;                    // Error code returned by acquisition function

} EFRIO TAcqEmul;
```

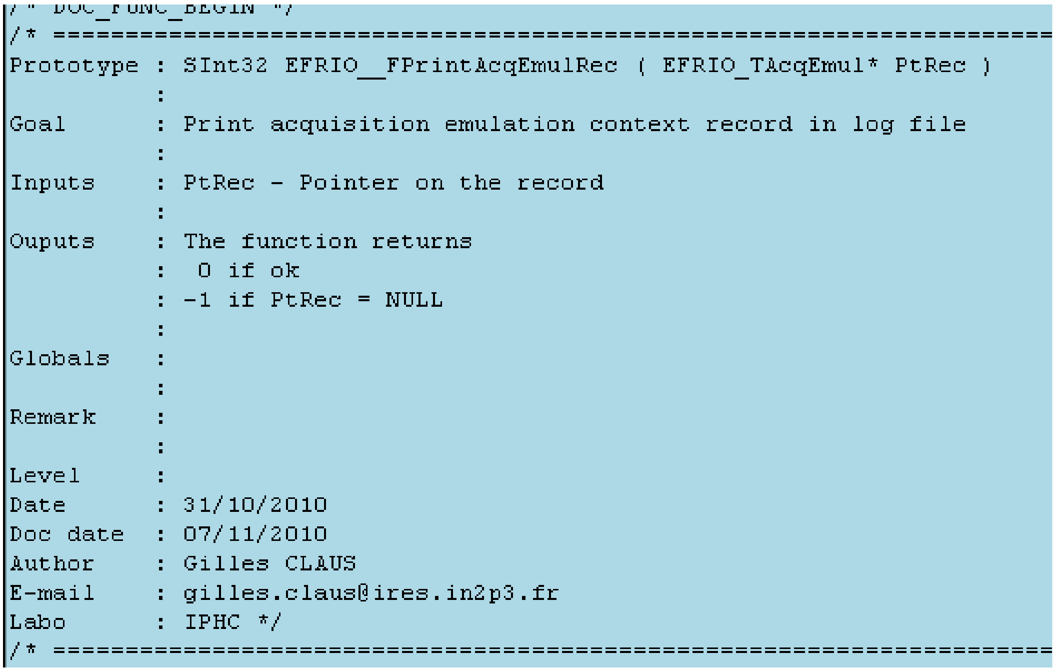
Function which fill EFRIO__TAcqEmul with parameters from gui → WinMain.cpp



```
//--------------------------------------------------------------------------
void __fastcall TFrmMain::FGrpEmulBoardGetPar ( int Caller )
{
  SInt8 Vi;

  EFRIO_TAcqEmul*    VPtAcqEmul = &EFRIO__VGContext.AcqEmul;
  EFRIO__TRunCont*   VPtRunCont = &EFRIO__VGContext.RunCont;
  EFRIO__TBoardConf* VPtBoard   = EFRIO__VGContext.ABoardsConf;

  // Get param from GUI

  VPtAcqEmul->ParAcqCycleMs                = FEdit2DecInt ( GrpEmulBoard_EdAcqCycleMs      );
  VPtAcqEmul->ParEmuleDRamReadMs           = FEdit2DecInt ( GrpEmulBoard_EdEmuleDRamReadMs );
  VPtAcqEmul->ParEmuleFunctNo              = FEdit2DecInt ( GrpEmulBoard_EdEmulFunctNo     );
  VPtAcqEmul->ParRandomDataSz              = (SInt8) GrpEmulBoard_ChkRandomDataSz->Checked;
  VPtAcqEmul->ParSetMaxDataSzOnOneMaps     = (SInt8) GrpEmulBoard_ChkMaxDataSzOnOneMaps->Checke

  VPtAcqEmul->ParTrigNbPerFrame            = GrpEmulBoard_CsTrigNbPerFrame->Value;
  VPtAcqEmul->ParTrigOnOneFrameOverN       = GrpEmulBoard_CsTrigOneFrameOverN->Value;
  VPtAcqEmul->ParTrigOnNConsecutiveFrames = GrpEmulBoard_CsTrigOnNConsecutiveFrames->Value;

  for ( Vi=0; Vi < EFRIO__MAX_ASIC_NB; Vi++ ) {
    VPtAcqEmul->ParAHeader[Vi]  = FEdit2HexInt ( GrpEmulBoard__AEdHeader[Vi] );
    VPtAcqEmul->ParATrailer[Vi] = FEdit2HexInt ( GrpEmulBoard__AEdTrailer[Vi] );
  }

  for ( Vi=0; Vi < EFRIO__MAX_EMUL_GUI_TRIG_NB; Vi++ ) {
    VPtAcqEmul->ParATrig[Vi] = FEdit2DecInt ( GrpEmulBoard__AEdTrig[Vi] );
    VPtAcqEmul->ParATS[Vi]   = FEdit2DecInt ( GrpEmulBoard__AEdTS[Vi]   );
  }

}
```

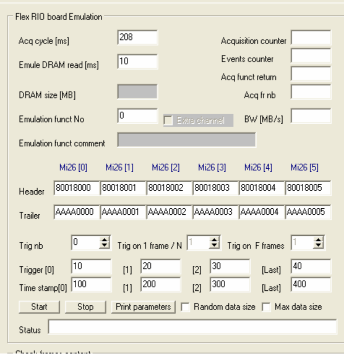
Context printing function EFRIO_FPrintAcqEmulRec (…) → eudet_frio_print.c

This function prints run context record in messages window and log file.



```
/* DOC_FUNC_BEGIN */
/* =====================================================================
Prototype : SInt32 EFRIO__FPrintAcqEmulRec ( EFRIO_TAcqEmul* PtRec )
          :
Goal      : Print acquisition emulation context record in log file
          :
Inputs    : PtRec - Pointer on the record
          :
Ouputs    : The function returns
          :  0 if ok
          : -1 if PtRec = NULL
          :
Globals   :
          :
Remark    :
          :
Level     :
Date      : 31/10/2010
Doc date  : 07/11/2010
Author    : Gilles CLAUS
E-mail    : gilles.claus@ires.in2p3.fr
Labo      : IPHC */
/* =====================================================================
```

Part of code called by a click on button “ Start ”



Acq cycle [ms] 208
Emule DRAM read [ms] 10
DRAM size [MB]
Emulation funct No 0
Emulation funct comment

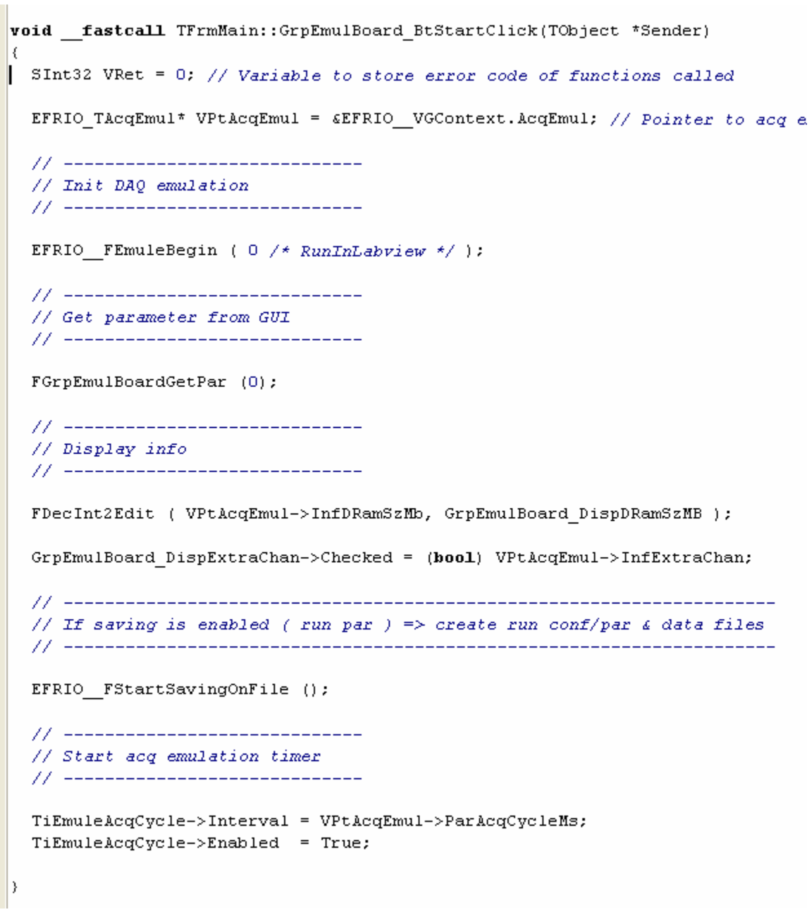
Acquisition counter
Events counter
Acq funct return
Acq fr nb
BW [MB/s]

	Mi26 [0]	Mi26 [1]	Mi26 [2]	Mi26 [3]	Mi26 [4]	Mi26 [5]
Header	80018000	80018001	80018002	80018003	80018004	80018005
Trailer	AAAA0000	AAAA0001	AAAA0002	AAAA0003	AAAA0004	AAAA0005

Trig nb 0 Trig on 1 frame / N 1 Trig on F frames 1

Trigger [0]	10	[1]	20	[2]	30	[Last]	40
Time stamp[0]	100	[1]	200	[2]	300	[Last]	400

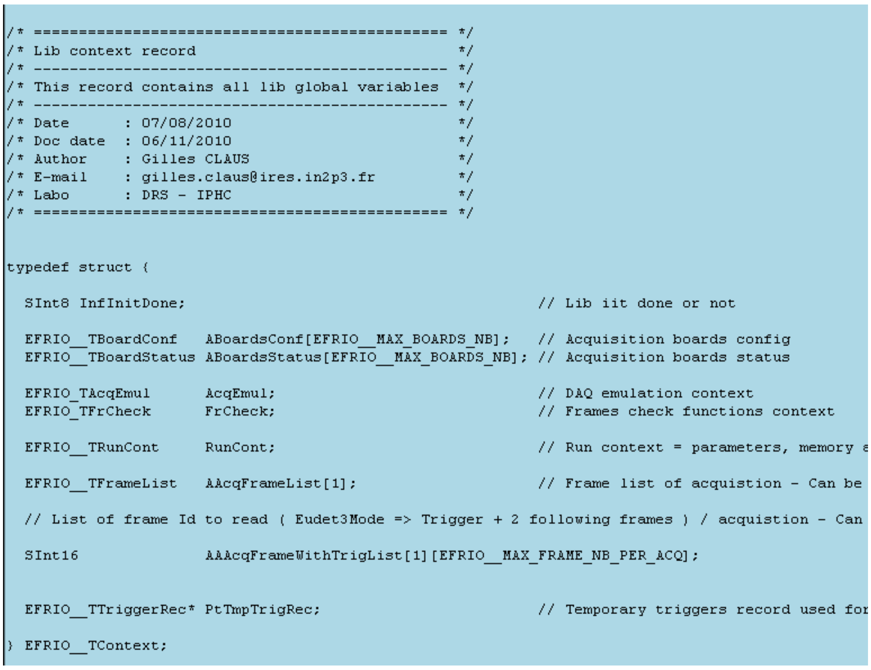
Status



```
void __fastcall TFrmMain::GrpEmulBoard_BtStartClick(TObject *Sender)
{
  SInt32 VRet = 0; // Variable to store error code of functions called

  EFRIO_TAcqEmul* VPtAcqEmul = &EFRIO__VGContext.AcqEmul; // Pointer to acq e

  // -----------------------------
  // Init DAQ emulation
  // -----------------------------

  EFRIO__FEmuleBegin ( 0 /* RunInLabview */ );

  // -----------------------------
  // Get parameter from GUI
  // -----------------------------

  FGrpEmulBoardGetPar (0);

  // -----------------------------
  // Display info
  // -----------------------------

  FDecInt2Edit ( VPtAcqEmul->InfDRamSzMb, GrpEmulBoard_DispDRamSzMB );

  GrpEmulBoard_DispExtraChan->Checked = (bool) VPtAcqEmul->InfExtraChan;

  // ------------------------------------------------------------------
  // If saving is enabled ( run par ) => create run conf/par & data files
  // ------------------------------------------------------------------

  EFRIO__FStartSavingOnFile ();

  // -----------------------------
  // Start acq emulation timer
  // -----------------------------

  TiEmuleAcqCycle->Interval = VPtAcqEmul->ParAcqCycleMs;
  TiEmuleAcqCycle->Enabled  = True;

}
```

7.2.3 How to access to context records → which variables ?

If the code is written in eudet_frio library we can access via the global variable EFRIO__VGContext which contains all variables of library.



```
/* ============================================== */
/* Lib context record                             */
/* ---------------------------------------------- */
/* This record contains all lib global variables  */
/* ---------------------------------------------- */
/* Date       : 07/08/2010                        */
/* Doc date   : 06/11/2010                        */
/* Author     : Gilles CLAUS                      */
/* E-mail     : gilles.claus@ires.in2p3.fr        */
/* Labo       : DRS - IPHC                        */
/* ============================================== */


typedef struct {

  SInt8 InfInitDone;                                   // Lib iit done or not

  EFRIO__TBoardConf   ABoardsConf[EFRIO__MAX_BOARDS_NB];   // Acquisition boards config
  EFRIO__TBoardStatus ABoardsStatus[EFRIO__MAX_BOARDS_NB]; // Acquisition boards status

  EFRIO_TAcqEmul      AcqEmul;                             // DAQ emulation context
  EFRIO_TFrCheck      FrCheck;                             // Frames check functions context

  EFRIO__TRunCont     RunCont;                             // Run context = parameters, memory a

  EFRIO__TFrameList   AAcqFrameList[1];                    // Frame list of acquistion - Can be

  // List of frame Id to read ( Eudet3Mode => Trigger + 2 following frames ) / acquistion - Can

  SInt16              AAAcqFrameWithTrigList[1][EFRIO__MAX_FRAME_NB_PER_ACQ];


  EFRIO__TTriggerRec* PtTmpTrigRec;                        // Temporary triggers record used for

} EFRIO__TContext;
```

You can use the following fields

- **RunCont to access run context record → EFRIO__TRunCont**
- **AcqEmul to access acquisition emulation context → EFRIO_TAcqEmul**

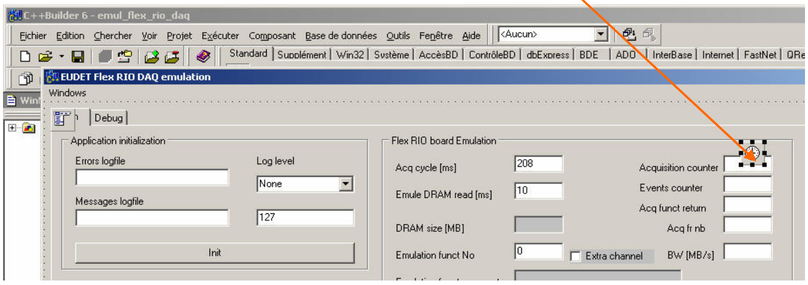
If the code is written outside eudet_frio library … you will find yourself the way … If you are afraid about global variables malediction … you can write a function which return a pointer to EFRIO_VGContext ;-) or encapsulate it in a class with a method to access to each field …

7.3 The output side → Telescope data stream

7.3.1 How the Flex RIO board is read ?

The flex RIO board acquires bunches of consecutive frames, then the software read the board. One of this bunches is called " an acquisition ", the default number of consecutives frames stored in one acquisition is 1800 (for historical reasons). As long as the software "runs fast enough" there is no missing frames from one acquisition to the next one. The period between acquisitions is ~ 207 ms for 1800 frames / acquisition.

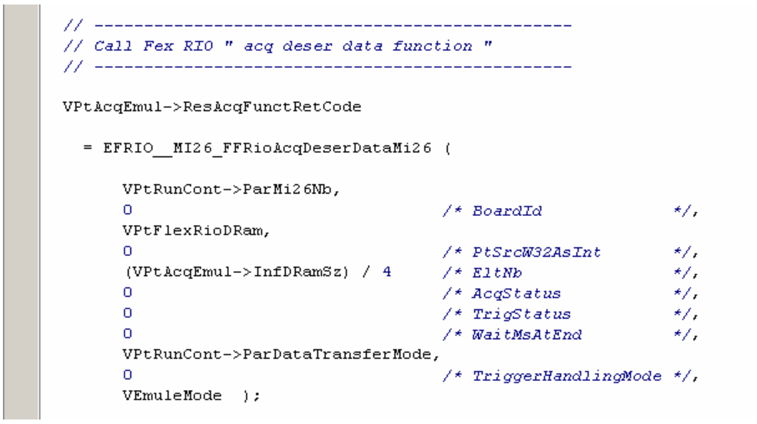
As the time between two acquisitions is ~ 207 ms, we can use a timer to call the board readout function. It's done like this, with a timer in DAQ emulator, in Labview DAQ application it's done with an endless loop because we want to minimize the risk to loose frames.


C++Builder 6 - emul_flex_rio_daq
Fichier Edition Chercher Voir Projet Exécuter Composant Base de données Outils Fenêtre Aide
<Aucun>
Standard Supplément Win32 Système AccèsBD ContrôleBD dbExpress BDE ADO InterBase Internet FastNet QRe
EUDET Flex RIO DAQ emulation
Windows
Debug
Application initialization
Flex RIO board Emulation
Init
☐ Extra channel

Errors logfile:
Log level: None
Messages logfile:
127
Acq cycle [ms]: 208
Emule DRAM read [ms]: 10
DRAM size [MB]:
Emulation funct No: 0
Acquisition counter:
Events counter:
Acq funct return:
Acq fr nb:
BW [MB/s]:

This is a part of the timer callback function which call the eudet_frio library functions which process data (frame with trigger selection, etc …)

→ EFRIO__MI26_FFRioAcqDeserDataMi26 (…)



```
// -----------------------------------------------
// Call Fex RIO " acq deser data function "
// -----------------------------------------------

VPtAcqEmul->ResAcqFunctRetCode

  = EFRIO__MI26_FFRioAcqDeserDataMi26 (

      VPtRunCont->ParMi26Nb,
      0                                /* BoardId              */,
      VPtFlexRioDRam,
      0                                /* PtSrcW32AsInt        */,
      (VPtAcqEmul->InfDRamSz) / 4      /* EltNb                */,
      0                                /* AcqStatus            */,
      0                                /* TrigStatus           */,
      0                                /* WaitMsAtEnd          */,
      VPtRunCont->ParDataTransferMode,
      0                                /* TriggerHandlingMode */,
      VEmuleMode   );
```

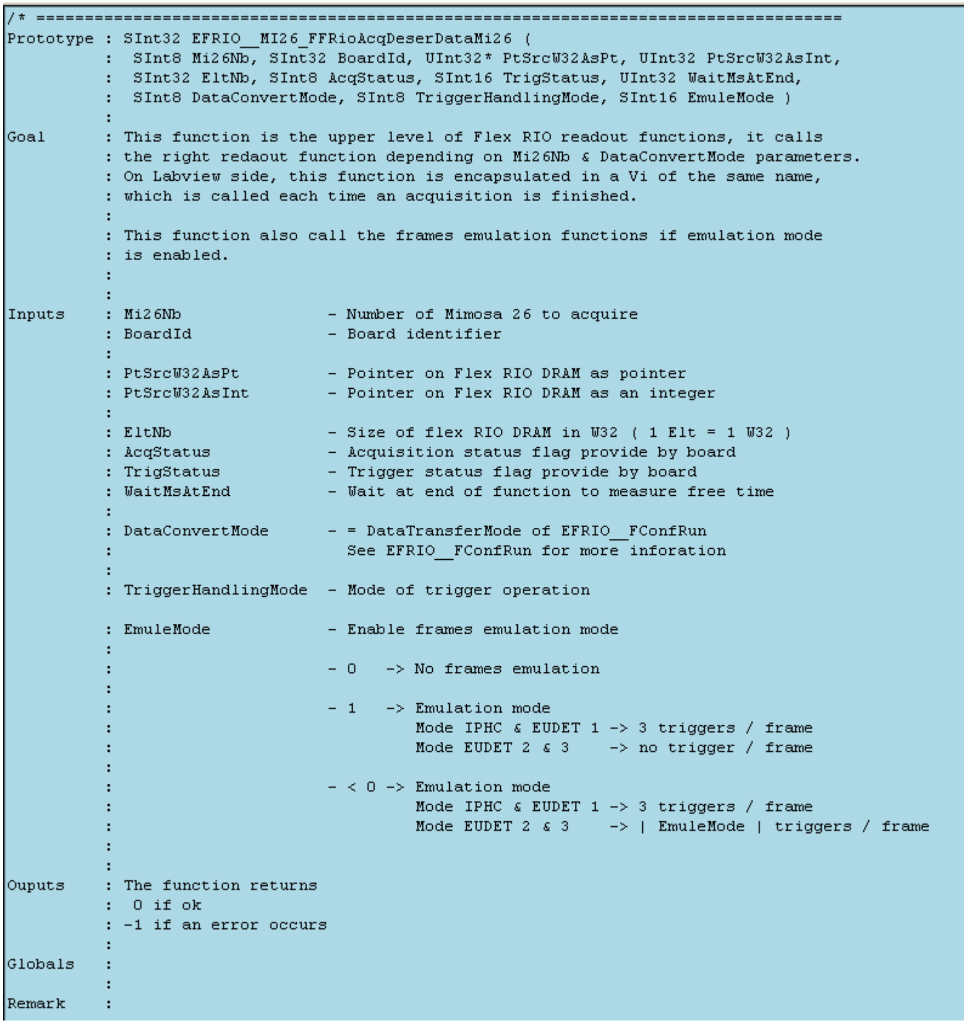
The EFRIO__MI26_FFRioAcqDeserDataMi26 (…) :

- **Get access to Flex RIO data via parameter PtSrcW32AsPt or PtSrcW32AsInt**

- **Call the emulation function if needed**

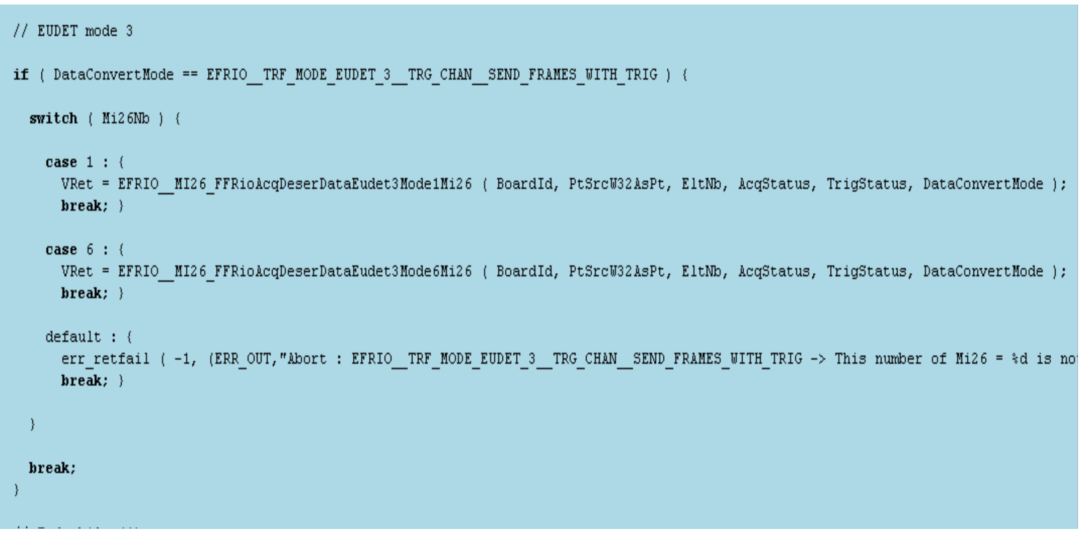
- **Call the add-hoc function to process data depending of "DataTransferMode" and Mimosa 26 number.**

This is the function comment header in eudet_frio.c file.



```
/* ===================================================================================
Prototype : SInt32 EFRIO__MI26_FFRioAcqDeserDataMi26 (
          :  SInt8 Mi26Nb, SInt32 BoardId, UInt32* PtSrcW32AsPt, UInt32 PtSrcW32AsInt,
          :  SInt32 EltNb, SInt8 AcqStatus, SInt16 TrigStatus, UInt32 WaitMsAtEnd,
          :  SInt8 DataConvertMode, SInt8 TriggerHandlingMode, SInt16 EmuleMode )
          :
Goal      : This function is the upper level of Flex RIO readout functions, it calls
          : the right redaout function depending on Mi26Nb & DataConvertMode parameters.
          : On Labview side, this function is encapsulated in a Vi of the same name,
          : which is called each time an acquisition is finished.
          :
          : This function also call the frames emulation functions if emulation mode
          : is enabled.
          :
          :
Inputs    : Mi26Nb               - Number of Mimosa 26 to acquire
          : BoardId              - Board identifier
          :
          : PtSrcW32AsPt         - Pointer on Flex RIO DRAM as pointer
          : PtSrcW32AsInt        - Pointer on Flex RIO DRAM as an integer
          :
          : EltNb                - Size of flex RIO DRAM in W32 ( 1 Elt = 1 W32 )
          : AcqStatus            - Acquisition status flag provide by board
          : TrigStatus           - Trigger status flag provide by board
          : WaitMsAtEnd          - Wait at end of function to measure free time
          :
          : DataConvertMode      - = DataTransferMode of EFRIO__FConfRun
          :                        See EFRIO__FConfRun for more inforation
          :
          : TriggerHandlingMode  - Mode of trigger operation

          : EmuleMode            - Enable frames emulation mode
          :
          :                      - 0   -> No frames emulation
          :
          :                      - 1   -> Emulation mode
          :                               Mode IPHC & EUDET 1 -> 3 triggers / frame
          :                               Mode EUDET 2 & 3    -> no trigger / frame
          :
          :                      - < 0 -> Emulation mode
          :                               Mode IPHC & EUDET 1 -> 3 triggers / frame
          :                               Mode EUDET 2 & 3    -> | EmuleMode | triggers / frame
          :
          :
Ouputs    : The function returns
          :  0 if ok
          : -1 if an error occurs
          :
Globals   :
          :
Remark    :
```

This is the part of EFRIO__MI26_FFRioAcqDeserDataMi26 (…) which calls the add-hoc data processing function in mode EUDET 3.



```
// EUDET mode 3

if ( DataConvertMode == EFRIO__TRF_MODE_EUDET_3__TRG_CHAN__SEND_FRAMES_WITH_TRIG ) {

  switch ( Mi26Nb ) {

    case 1 : {
      VRet = EFRIO__MI26_FFRioAcqDeserDataEudet3Mode1Mi26 ( BoardId, PtSrcW32AsPt, EltNb, AcqStatus, TrigStatus, DataConvertMode );
      break; }

    case 6 : {
      VRet = EFRIO__MI26_FFRioAcqDeserDataEudet3Mode6Mi26 ( BoardId, PtSrcW32AsPt, EltNb, AcqStatus, TrigStatus, DataConvertMode );
      break; }

    default : {
      err_retfail ( -1, (ERR_OUT,"Abort : EFRIO__TRF_MODE_EUDET_3__TRG_CHAN__SEND_FRAMES_WITH_TRIG -> This number of Mi26 = %d is no
      break; }

  }

  break;
}
```

7.3.2 How the data stream is organized ?

The data stream is organized “ per acquisition ”, if the number of frames per acquisition is set to 1800, the period between two acquisition will be 1800 x 115,2 µs (Mimosa 26 frame duration) = ~ 207 ms. It means that each 207 ms an acquisition will be ready in library memory, the number of frames will depend of the number of triggers therefore it can be less than 1800.

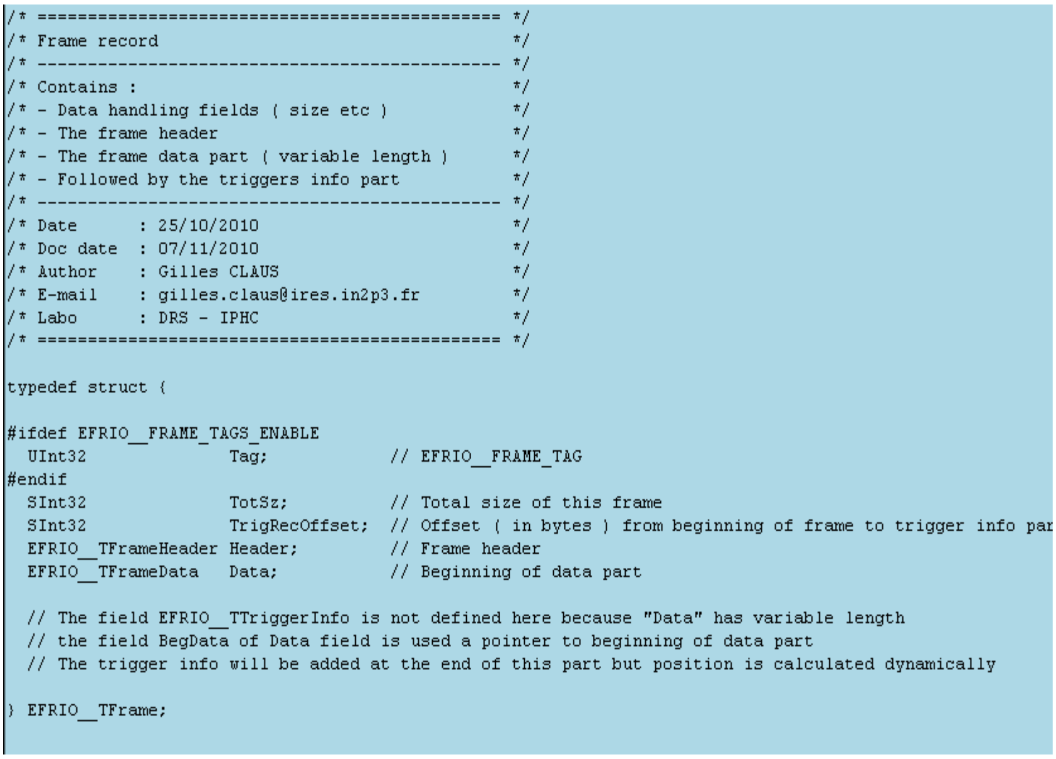
The eudet_frio library allocates a buffer large enough to contain an acquisition of the specified frames nb per acquisition value set in run control. This will be a single bloc of RAM on which you can have access in two ways :

- **Pointer on the bloc beginning → PtFrame**
- **An array of pointers on each frame → AFramePtr[i]**

It means that you can send on Ethernet the whole acquisition defined by PtFrame and it’s size or frame by frame by scanning the array AFramePtr[i]. It’s up to you to decide which method is the best.

7.3.3 Organization of one frame → EFRIO__TFrame

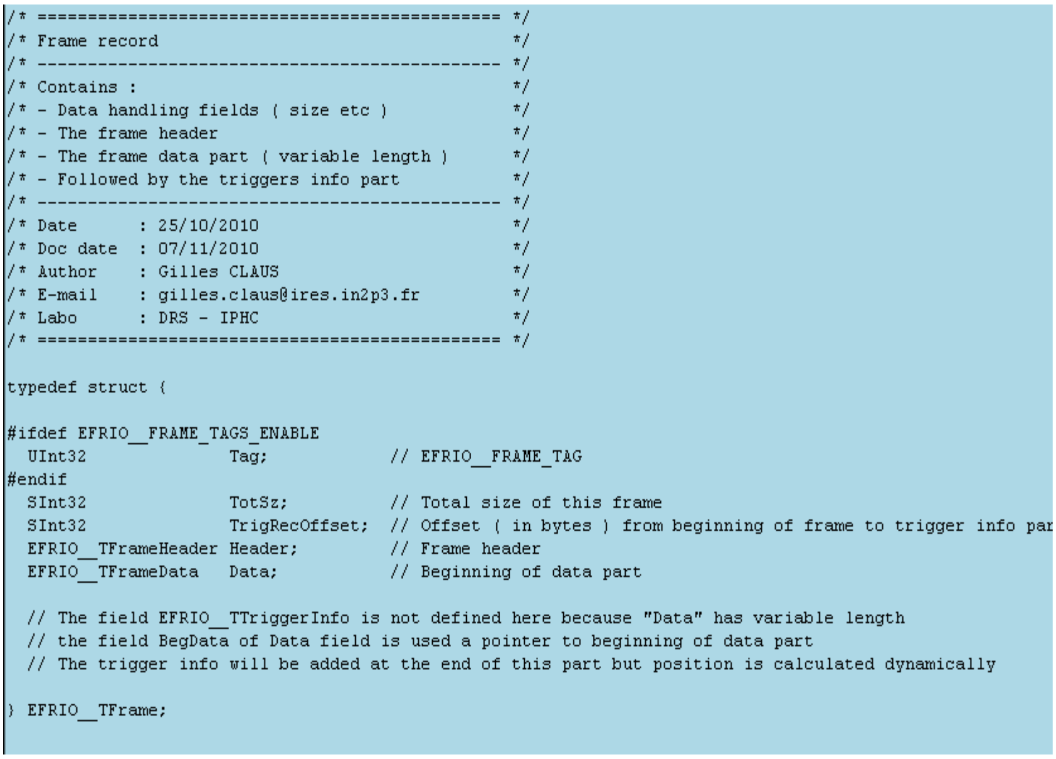
The type which define the frame is EFRIO__TFrame → eudet_frio.typ



```
/* ============================================== */
/* Frame record                                   */
/* ---------------------------------------------- */
/* Contains :                                     */
/* - Data handling fields ( size etc )            */
/* - The frame header                             */
/* - The frame data part ( variable length )      */
/* - Followed by the triggers info part           */
/* ---------------------------------------------- */
/* Date      : 25/10/2010                         */
/* Doc date  : 07/11/2010                         */
/* Author    : Gilles CLAUS                       */
/* E-mail    : gilles.claus@ires.in2p3.fr         */
/* Labo      : DRS - IPHC                         */
/* ============================================== */

typedef struct {

#ifdef EFRIO__FRAME_TAGS_ENABLE
  UInt32              Tag;            // EFRIO__FRAME_TAG
#endif
  SInt32              TotSz;          // Total size of this frame
  SInt32              TrigRecOffset;  // Offset ( in bytes ) from beginning of frame to trigger info par
  EFRIO__TFrameHeader Header;         // Frame header
  EFRIO__TFrameData   Data;           // Beginning of data part

  // The field EFRIO__TTriggerInfo is not defined here because "Data" has variable length
  // the field BegData of Data field is used a pointer to beginning of data part
  // The trigger info will be added at the end of this part but position is calculated dynamically

} EFRIO__TFrame;
```

The frame will contains the following items :

- **Size handling fields**
- **A header of fixed size → Header**
- **A data part of variable size → Data**
- **After the data part a list of triggers of variable size**



```
/* ============================================== */
/* Frame record                                   */
/* ---------------------------------------------- */
/* Contains :                                     */
/* - Data handling fields ( size etc )            */
/* - The frame header                             */
/* - The frame data part ( variable length )      */
/* - Followed by the triggers info part           */
/* ---------------------------------------------- */
/* Date      : 25/10/2010                         */
/* Doc date  : 07/11/2010                         */
/* Author    : Gilles CLAUS                       */
/* E-mail    : gilles.claus@ires.in2p3.fr         */
/* Labo      : DRS - IPHC                         */
/* ============================================== */

typedef struct {

#ifdef EFRIO__FRAME_TAGS_ENABLE
  UInt32              Tag;            // EFRIO__FRAME_TAG
#endif
  SInt32              TotSz;          // Total size of this frame
  SInt32              TrigRecOffset;  // Offset ( in bytes ) from beginning of frame to trigger info par
  EFRIO__TFrameHeader Header;         // Frame header
  EFRIO__TFrameData   Data;           // Beginning of data part

  // The field EFRIO__TTriggerInfo is not defined here because "Data" has variable length
  // the field BegData of Data field is used a pointer to beginning of data part
  // The trigger info will be added at the end of this part but position is calculated dynamically

} EFRIO__TFrame;
```

The first field “ Tag ” is used to tag beginning of frame in data stream, it can be helpful if someone need to deal with binary data. It’s value is set by constant EFRIO__FRAME_TAG define in eudet_frio.h. This field can be removed by conditional compilation.

The second field “ ToSz ” indicates the total size of the current frame. If you want to go o next frames, set a byte pointer on current one, add “ TotSz ”, cast the byte pointer to EFRIO__TFrame*, and it’s done.

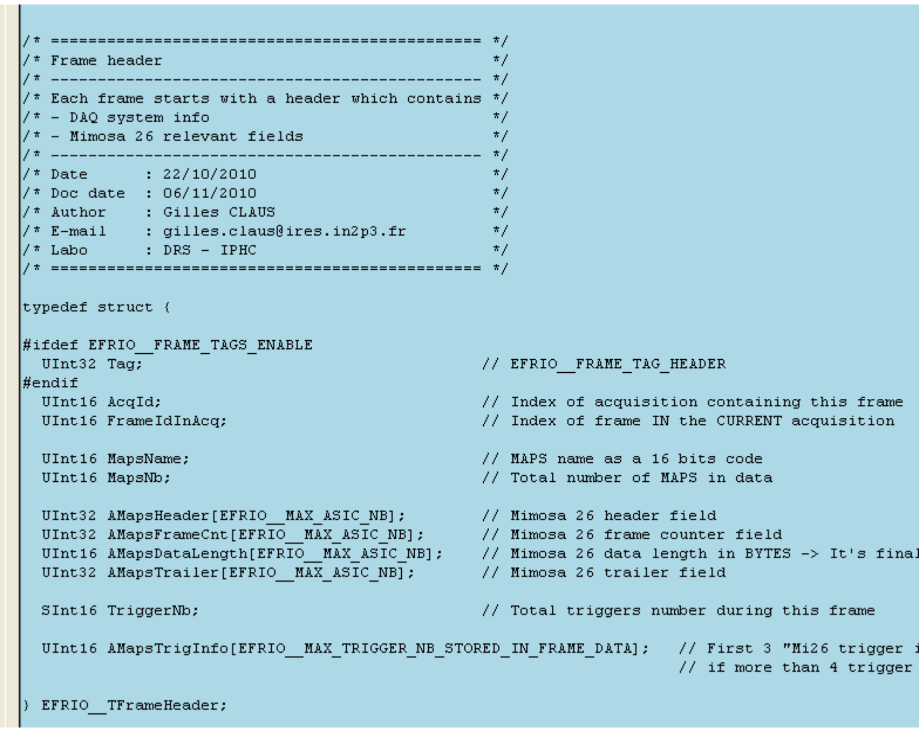
The third field “ TrigRecOffset ” indicates the position of the trigger record in the frame, it follows data part which has a variable size. To go to trigger record, set a byte pointer on beginning of frame, add “TrigRecOffset ”, cast the byte pointer to EFRIO__TTriggerRec*, and it’s done.

The fourth field “ Header ” is the frame header, it has a fixed size.

The fifth field “ Data ” is the beginning of data part

There is no field in EFRIO_TFrame for the triggers record because it’s not possible as the Data field has a variable length, that’s why we need the field “TrigRecOffset ”.

7.3.4 The frame header record → EFRIO_TFrameHeader



```
/* ============================================== */
/* Frame header                                   */
/* ---------------------------------------------- */
/* Each frame starts with a header which contains */
/* - DAQ system info                              */
/* - Mimosa 26 relevant fields                    */
/* ---------------------------------------------- */
/* Date      : 22/10/2010                         */
/* Doc date  : 06/11/2010                         */
/* Author    : Gilles CLAUS                       */
/* E-mail    : gilles.claus@ires.in2p3.fr         */
/* Labo      : DRS - IPHC                         */
/* ============================================== */

typedef struct {

#ifdef EFRIO__FRAME_TAGS_ENABLE
  UInt32 Tag;                                    // EFRIO__FRAME_TAG_HEADER
#endif
  UInt16 AcqId;                                  // Index of acquisition containing this frame
  UInt16 FrameIdInAcq;                           // Index of frame IN the CURRENT acquisition

  UInt16 MapsName;                               // MAPS name as a 16 bits code
  UInt16 MapsNb;                                 // Total number of MAPS in data

  UInt32 AMapsHeader[EFRIO__MAX_ASIC_NB];        // Mimosa 26 header field
  UInt32 AMapsFrameCnt[EFRIO__MAX_ASIC_NB];      // Mimosa 26 frame counter field
  UInt16 AMapsDataLength[EFRIO__MAX_ASIC_NB];    // Mimosa 26 data length in BYTES -> It's final
  UInt32 AMapsTrailer[EFRIO__MAX_ASIC_NB];       // Mimosa 26 trailer field

  SInt16 TriggerNb;                              // Total triggers number during this frame

  UInt16 AMapsTrigInfo[EFRIO__MAX_TRIGGER_NB_STORED_IN_FRAME_DATA];   // First 3 "Mi26 trigger i
                                                                      // if more than 4 trigger

} EFRIO__TFrameHeader;
```

The first field " Tag " acts like EFRIO_TFrame Tag,
it is set to EFRIO__FRAME_TAG_HEADER

AcqId and FrameIdInAcq indicates the index of the acquisition which contains this frame and the index of the frame (0..1799) in this acquisition.

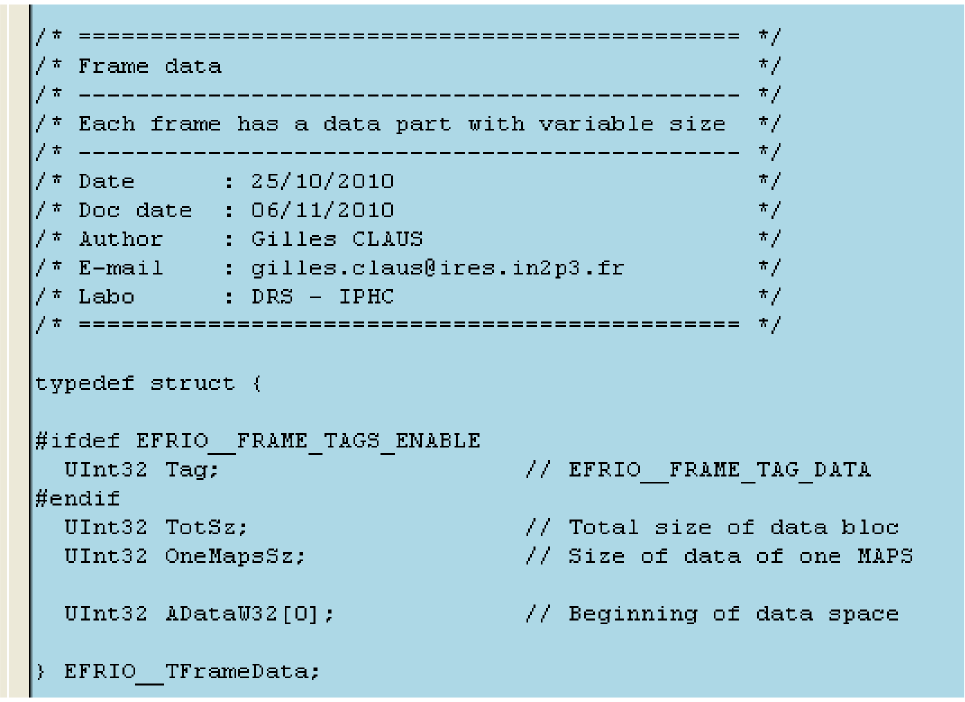
MapsName is a code to identify the MAPS, MapsNb the number of MAPS in DAQ.

The fields AMaps… are arrays containing Mimosa 26 frame header, … trailer

TriggerNb contains the number of trigger during the current frame

AMapsTrigInfo contains the first three triggers keep for compatibility with our previous DAQ -→ not useful for EUDET

7.3.5 The data part→ EFRIO_TFrameData



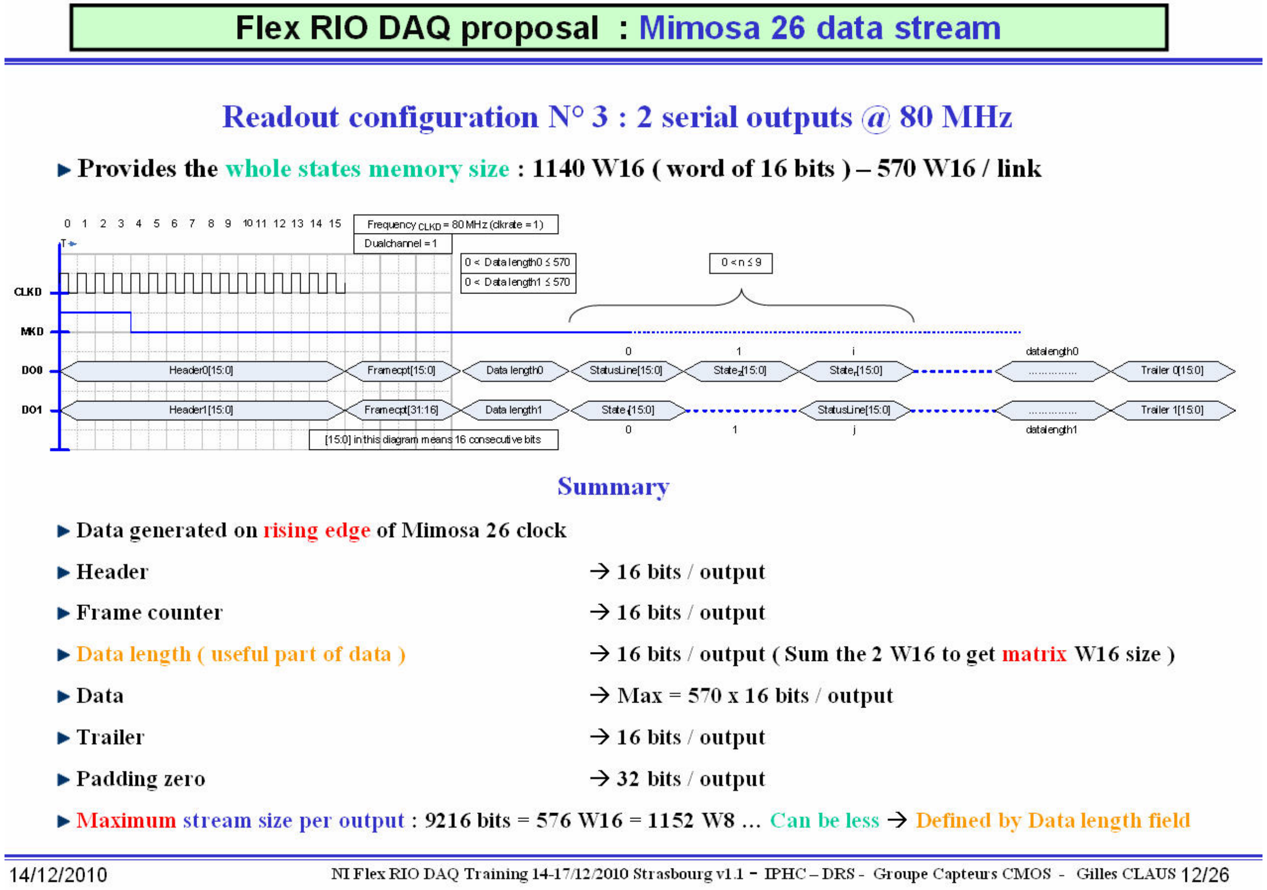
```
/* ============================================== */
/* Frame data                                     */
/* ---------------------------------------------- */
/* Each frame has a data part with variable size  */
/* ---------------------------------------------- */
/* Date       : 25/10/2010                        */
/* Doc date   : 06/11/2010                        */
/* Author     : Gilles CLAUS                      */
/* E-mail     : gilles.claus@ires.in2p3.fr        */
/* Labo       : DRS - IPHC                        */
/* ============================================== */

typedef struct {

#ifdef EFRIO__FRAME_TAGS_ENABLE
  UInt32 Tag;                         // EFRIO__FRAME_TAG_DATA
#endif
  UInt32 TotSz;                       // Total size of data bloc
  UInt32 OneMapsSz;                   // Size of data of one MAPS

  UInt32 ADataW32[0];                 // Beginning of data space

} EFRIO__TFrameData;
```

The first field “ Tag ” acts like EFRIO_TFrame Tag,
it is set to EFRIO__FRAME_TAG_DATA

The second field “ TotSz ” indicates the total size of data bloc

The third field “ OneMapSz ” indicates the data size for one MAPS, must be multiplied by the number of MAPS to get the size [in bytes] of ADataW32 array.

The fourth field “ ADataW32” is a pointer to the first W32 of data.



Flex RIO DAQ proposal : Mimosa 26 data stream

Readout configuration N° 3 : 2 serial outputs @ 80 MHz

- Provides the whole states memory size : 1140 W16 (word of 16 bits) – 570 W16 / link

Summary

- Data generated on rising edge of Mimosa 26 clock

▸ Header	→ 16 bits / output
▸ Frame counter	→ 16 bits / output
▸ Data length (useful part of data)	→ 16 bits / output (Sum the 2 W16 to get matrix W16 size)
▸ Data	→ Max = 570 x 16 bits / output
▸ Trailer	→ 16 bits / output
▸ Padding zero	→ 32 bits / output

- Maximum stream size per output : 9216 bits = 576 W16 = 1152 W8 … Can be less → Defined by Data length field

WARNING !

The Mimosa 26 data stream is multiplexed on two data links D00 and D01, as explained on previous page. The Flex Rio firmware has a 16 bits deserializer connected to each data link and it doesn’t demultiplex data after deserialization. Therefore this multiplexed data structure is still present in the “ ADataW32 ” array of the “ EFRIO__TFrameData ”.

The data stream has been demultiplexed to fill the EFRIO__TFrameHeader fields (header, frame counter, data length trailer) but not for the data part. This data demultiplexing has not been implemented because this processing cost execution time and when the first version of code has been written I didn’t know if it would be better to make this processing on NI CPU side or on EUDET DAQ side.

As we have 100 ms free CPU time, it can be implemented on NI CPU side, but it is not done yet, the function EFRIO__MI26_FFRioAcqDeserDataEudet3Mode6Mi26 must be modified to implement it.

Organization of data part in case of one Mimosa 26 is read by DAQ :

W n = Word of 32 bits from array “ ADataW32 ”

Data n link d0 = Word number n of 16 bits on data link d0

Data n link d1 = Word number n of 16 bits on data link d1

	D31D16	D15D00
W 0 =	Data 0 link d1	Data 0 link d0
W 1 =	Data 1 link d1	Data 1 link d0
W 2 =	Data 2 link d1	Data 2 link d0
…….		
…….		
…….		
…….		
W n =	Data n link d1	Data n link d0

<u>Organization of data part in case of six Mimosa 26 are read by DAQ :</u>

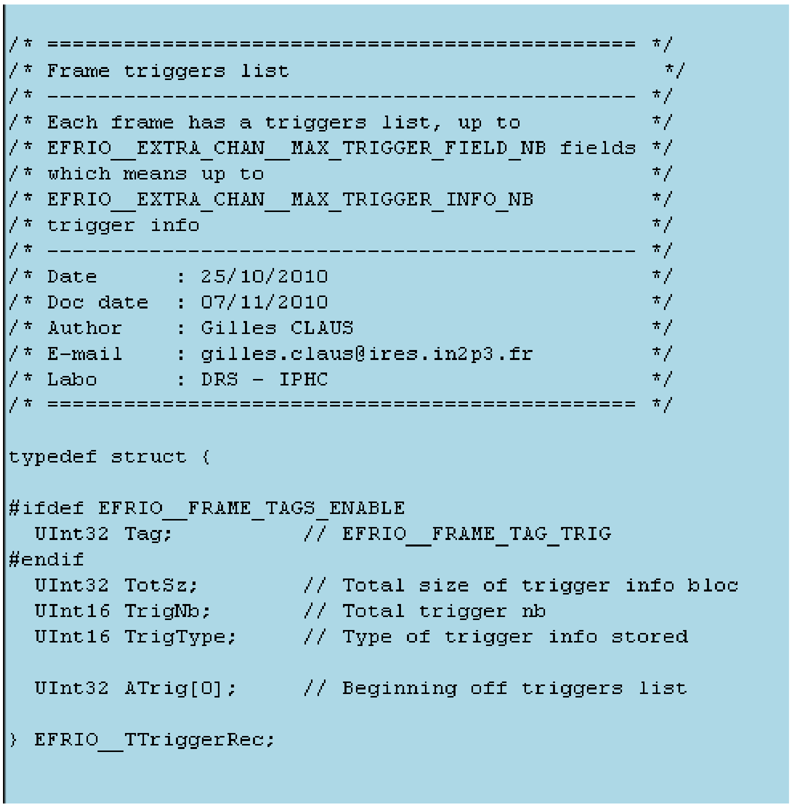
W n = Word of 32 bits from array “ ADataW32 ”

Data n link d0 chip x = Word number n of 16 bits on data link d0 of chip N°x

Data n link d1 chip x = Word number n of 16 bits on data link d1 of chip N°x

	D31D16	**D15D00**
W 0 =	**Data 0 link d1 chip 0**	**Data 0 link d0 chip 0**
W 1 =	**Data 0 link d1 chip 1**	**Data 1 link d0 chip 1**
W 2 =	**Data 0 link d1 chip 2**	**Data 2 link d0 chip 2**
W 3 =	**Data 0 link d1 chip 3**	**Data 2 link d0 chip 3**
W 4 =	**Data 0 link d1 chip 4**	**Data 2 link d0 chip 4**
W 5 =	**Data 0 link d1 chip 5**	**Data 2 link d0 chip 5**
W 6 =	**Data 1 link d1 chip 0**	**Data 1 link d0 chip 0**
W 7 =	**Data 1 link d1 chip 1**	**Data 1 link d0 chip 1**
W 8 =	**Data 1 link d1 chip 2**	**Data 1 link d0 chip 2**
W 9 =	**Data 1 link d1 chip 3**	**Data 1 link d0 chip 3**
W 10 =	**Data 1 link d1 chip 4**	**Data 1 link d0 chip 4**
W 11 =	**Data 1 link d1 chip 5**	**Data 1 link d0 chip 5**
…….		
…….		
…….		
…….		
W n =	**Data n link d1 chip 5**	**Data n link d0 chip 5**

7.3.6 The trigger record → EFRIO_TFrameData



```
/* ============================================== */
/* Frame triggers list                             */
/* ---------------------------------------------- */
/* Each frame has a triggers list, up to          */
/* EFRIO__EXTRA_CHAN__MAX_TRIGGER_FIELD_NB fields */
/* which means up to                              */
/* EFRIO__EXTRA_CHAN__MAX_TRIGGER_INFO_NB         */
/* trigger info                                   */
/* ---------------------------------------------- */
/* Date      : 25/10/2010                         */
/* Doc date  : 07/11/2010                         */
/* Author    : Gilles CLAUS                       */
/* E-mail    : gilles.claus@ires.in2p3.fr         */
/* Labo      : DRS - IPHC                         */
/* ============================================== */

typedef struct {

#ifdef EFRIO__FRAME_TAGS_ENABLE
  UInt32 Tag;          // EFRIO__FRAME_TAG_TRIG
#endif
  UInt32 TotSz;        // Total size of trigger info bloc
  UInt16 TrigNb;       // Total trigger nb
  UInt16 TrigType;     // Type of trigger info stored

  UInt32 ATrig[0];     // Beginning off triggers list

} EFRIO__TTriggerRec;
```

The first field " Tag " acts like EFRIO_TFrame Tag, it is set to EFRIO__FRAME_TAG_TRIG

The second field " TotSz " indicates the total size of trigger record

The third field " TrigNb " indicates the number of triggers

The fourth field " TrigType " indicates the type of trigger → reserved for future use

The fifth field " ATrig" is a pointer on triggers

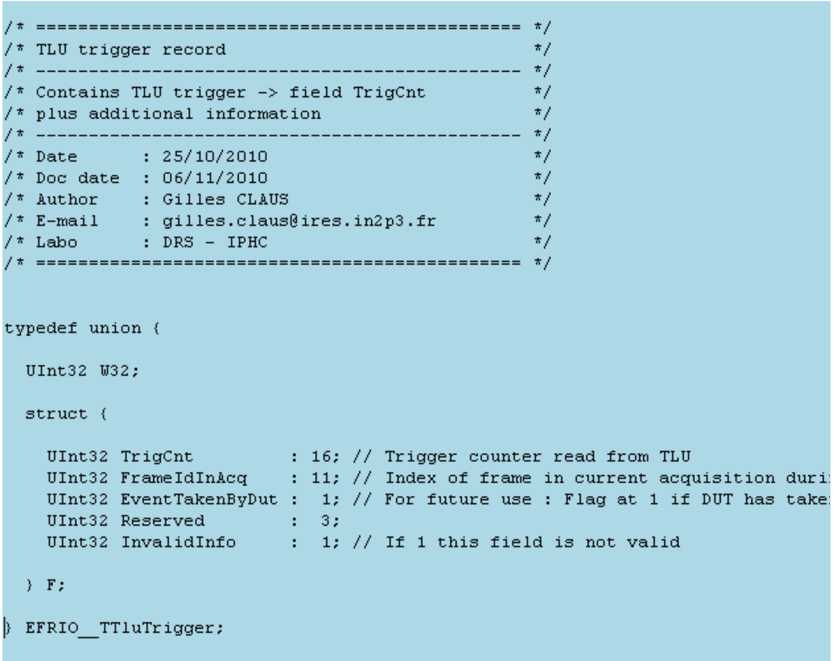
7.3.7 The trigger record items

For each trigger eudet_frio library stores two triggers fields : first from TLU and second one from Flex RIO. It means that the array ATrig[] will contain TrigNb X 2 items.

The TLU trigger info is written first, followed by the Flex RIO trigger / time stamp. It means that array organization will be as followed :

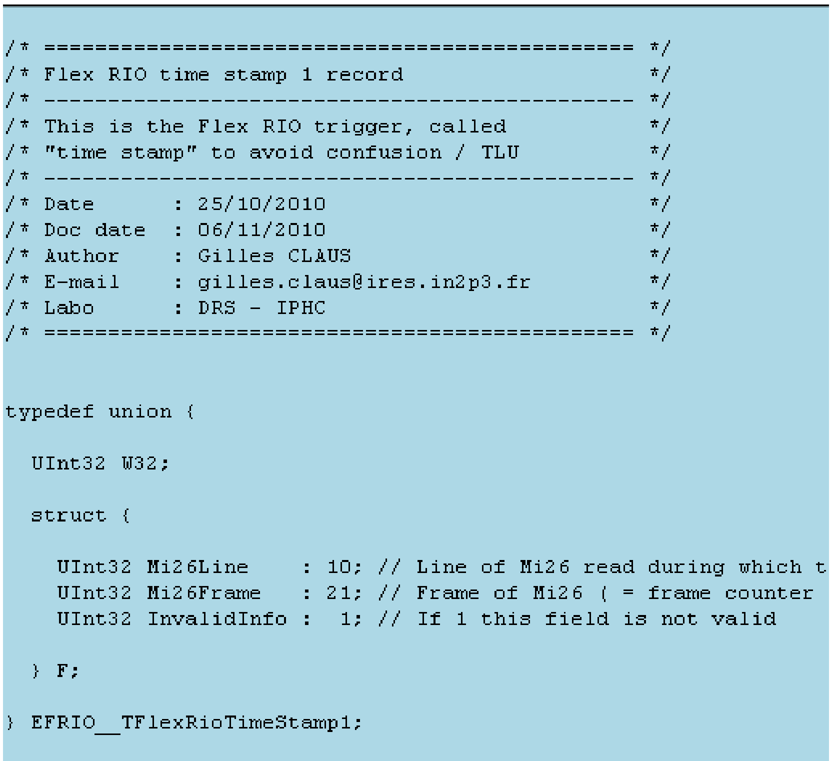
- **Trigger [0] TLU → ATrig[0]**
- **Trigger [0] Flex RIO → ATrig[1]**
- **Trigger [1] TLU → ATrig[2]**
- **Trigger [1] Flex RIO → ATrig[3]**
- **…**
- **…**
- **…**
- **Trigger [TrigNb-1] TLU → ATrig[(TrigNb X 2)]**
- **Trigger [TrigNb-1] Flex RIO → ATrig[[(TrigNb X 2) + 1]**

TLU trigger record → EFRIO__TTluTrigger -= W32



```
/* ============================================== */
/* TLU trigger record                             */
/* ---------------------------------------------- */
/* Contains TLU trigger -> field TrigCnt          */
/* plus additional information                    */
/* ---------------------------------------------- */
/* Date      : 25/10/2010                         */
/* Doc date  : 06/11/2010                         */
/* Author    : Gilles CLAUS                       */
/* E-mail    : gilles.claus@ires.in2p3.fr         */
/* Labo      : DRS - IPHC                         */
/* ============================================== */


typedef union {

  UInt32 W32;

  struct {

    UInt32 TrigCnt          : 16; // Trigger counter read from TLU
    UInt32 FrameIdInAcq     : 11; // Index of frame in current acquisition duri
    UInt32 EventTakenByDut  :  1; // For future use : Flag at 1 if DUT has take
    UInt32 Reserved         :  3;
    UInt32 InvalidInfo      :  1; // If 1 this field is not valid

  } F;

} EFRIO__TTluTrigger;
```

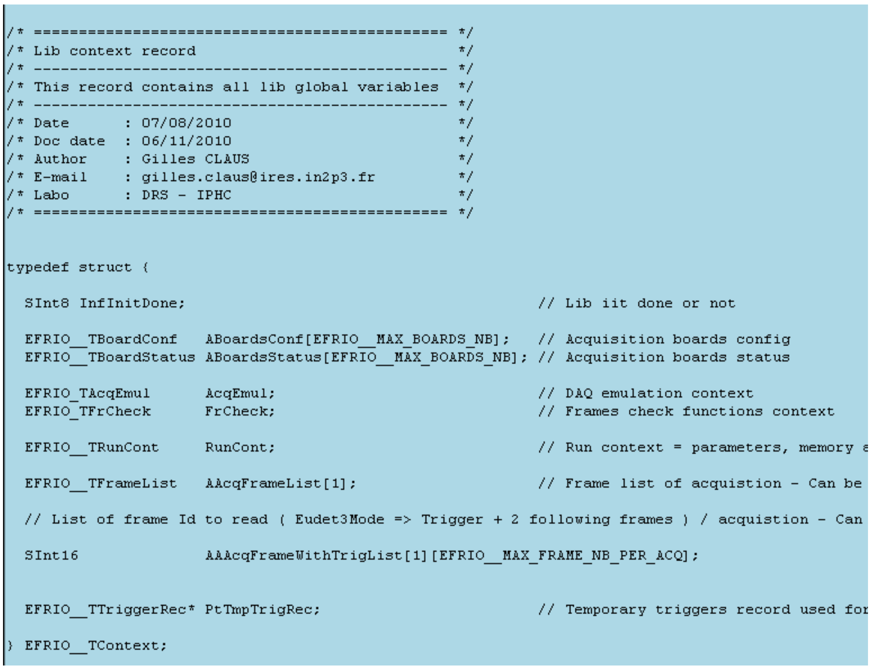
Flex RIO trigger / time stamp record → EFRIO__TFlexRioTimeStamp1 = W32



```
/* ============================================= */
/* Flex RIO time stamp 1 record                  */
/* --------------------------------------------- */
/* This is the Flex RIO trigger, called          */
/* "time stamp" to avoid confusion / TLU         */
/* --------------------------------------------- */
/* Date      : 25/10/2010                        */
/* Doc date  : 06/11/2010                        */
/* Author    : Gilles CLAUS                      */
/* E-mail    : gilles.claus@ires.in2p3.fr        */
/* Labo      : DRS - IPHC                        */
/* ============================================= */


typedef union {

  UInt32 W32;

  struct {

    UInt32 Mi26Line    : 10; // Line of Mi26 read during which t
    UInt32 Mi26Frame   : 21; // Frame of Mi26 ( = frame counter
    UInt32 InvalidInfo :  1; // If 1 this field is not valid

  } F;

} EFRIO__TFlexRioTimeStamp1;
```

7.3.8 How to access to frames data → which variables ?

If the code is written in eudet_frio library we can access via the global variable EFRIO__VGContext which contains all variables of library.



```
/* ============================================== */
/* Lib context record                             */
/* ---------------------------------------------- */
/* This record contains all lib global variables  */
/* ---------------------------------------------- */
/* Date      : 07/08/2010                         */
/* Doc date  : 06/11/2010                         */
/* Author    : Gilles CLAUS                       */
/* E-mail    : gilles.claus@ires.in2p3.fr         */
/* Labo      : DRS - IPHC                         */
/* ============================================== */


typedef struct {

  SInt8 InfInitDone;                                    // Lib iit done or not

  EFRIO__TBoardConf   ABoardsConf[EFRIO__MAX_BOARDS_NB];   // Acquisition boards config
  EFRIO__TBoardStatus ABoardsStatus[EFRIO__MAX_BOARDS_NB]; // Acquisition boards status

  EFRIO_TAcqEmul      AcqEmul;                             // DAQ emulation context
  EFRIO_TFrCheck      FrCheck;                             // Frames check functions context

  EFRIO__TRunCont     RunCont;                             // Run context = parameters, memory a

  EFRIO__TFrameList   AAcqFrameList[1];                    // Frame list of acquistion - Can be

  // List of frame Id to read ( Eudet3Mode => Trigger + 2 following frames ) / acquistion - Can

  SInt16              AAAcqFrameWithTrigList[1][EFRIO__MAX_FRAME_NB_PER_ACQ];


  EFRIO__TTriggerRec* PtTmpTrigRec;                        // Temporary triggers record used for

} EFRIO__TContext;
```

You can use the following fields

- **RunCont.PtFrame → EFRIO__TFrame* → Access to full bloc**
- **AAcqFrameList[0]. AFramePtr[FrameIndex] → Access frame by frame**

7.4 How / where to write the code ?

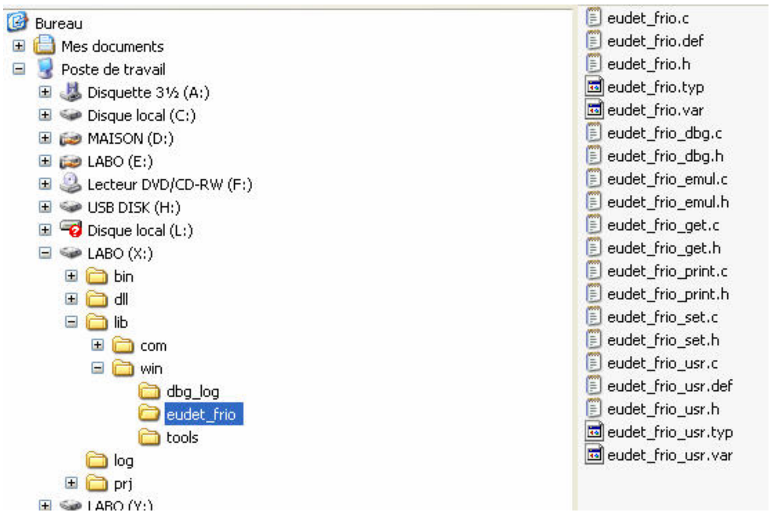
7.4.1 The eudet_frio library and DLL

It can be written in the eudet_frio library which is compiled as a DLL. The code can be C or C++. But for the interface to Labview, as far as I know, it must be simple C function, there is no easy way to interface a class to Labview. May be by encapsulation in ActiveX or .NET object ? I believe we don't such " funny things ", please use C and if a class is needed make a kind of wrapper via some C functions call. We want performances and reliability, we don't need state of the art in software development

7.4.2 Run control context record and configuration function

A set of files had been set in eudet_frio lib for user code implementation

- **Eudet_frio_usr.def → Macros and constants**
- **Eudet_frio_usr.typ → Types and classes definition**
- **Eudet_frio_usr.var → Global variables**
- **Eudet_frio_usr.h → Functions header**
- **Eudet_frio_usr.c → C or C++ code**



Bureau
⊞ Mes documents
⊟ Poste de travail
⊞ Disquette 3½ (A:)
⊞ Disque local (C:)
⊞ MAISON (D:)
⊞ LABO (E:)
⊞ Lecteur DVD/CD-RW (F:)
⊞ USB DISK (H:)
⊞ Disque local (L:)
⊟ LABO (X:)
⊞ bin
⊞ dll
⊟ lib
⊞ com
⊟ win
dbg_log
eudet_frio
tools
log
⊞ prj
⊞ LABO (Y:)

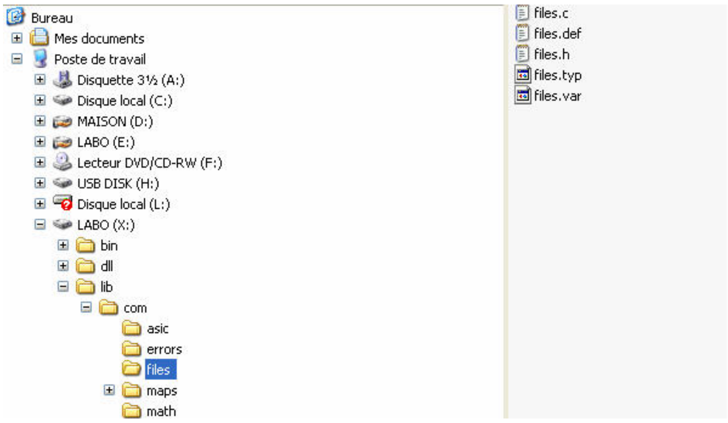
eudet_frio.c
eudet_frio.def
eudet_frio.h
eudet_frio.typ
eudet_frio.var
eudet_frio_dbg.c
eudet_frio_dbg.h
eudet_frio_emul.c
eudet_frio_emul.h
eudet_frio_get.c
eudet_frio_get.h
eudet_frio_print.c
eudet_frio_print.h
eudet_frio_set.c
eudet_frio_set.h
eudet_frio_usr.c
eudet_frio_usr.def
eudet_frio_usr.h
eudet_frio_usr.typ
eudet_frio_usr.var

They are empty, fill free to use them for your own source code. Therefore we can easily provide a library upgrade without impacting your own source code.

7.5 Warning about files library

This library handles files I/O, It implements classes TCBinFile and TCStreamFile used by EUDET Flex RIO library (eudet_frio).

This library is in directory x:\lib\com\files



```
Bureau
  Mes documents
  Poste de travail
    Disquette 3½ (A:)
    Disque local (C:)
    MAISON (D:)
    LABO (E:)
    Lecteur DVD/CD-RW (F:)
    USB DISK (H:)
    Disque local (L:)
    LABO (X:)
      bin
      dll
      lib
        com
          asic
          errors
          files
          maps
          math
```

```
files.c
files.def
files.h
files.typ
files.var
```

Warning about TCStreamFile class !

This class speed up disk access by

- **Making direct disk access = non buffered**
- **Having it's own thread to write data to disk, therefore saving is always done in background, it's not stopped while board is busy.**

But this class had been quickly designed to test the Flex RIO system hardware, therefore it has limitations and it had not been intensively tested. For example it creates a single file, the run is no split in different files … Therefore, if you decide to use it please do it carefully, test your code, report us bugs if needed.

Acknowledgement

This work is supported by the Commission of the European Communities under the 6th Framework Programme "Structuring the European Research Area", contract number RII3-026126.