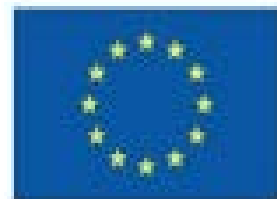




Simulate the DAQ for the VMM-FE-ASICs

A. Stylianidis [Undergraduate Student of Computer Science UoPeloponnese]

N. Benekos [NTUA]



European Union
European Social Fund



MINISTRY OF EDUCATION, LIFELONG LEARNING AND RELIGIOUS AFFAIRS
MANAGING AUTHORITY



EUROPEAN SOCIAL FUND

Co- financed by Greece and the European Union



Outline of Presentation



- ▶ Introduction
- ▶ Motivation
- ▶ Semantics
- ▶ Program Description and Features
- ▶ Future Steps
- ▶ Summary



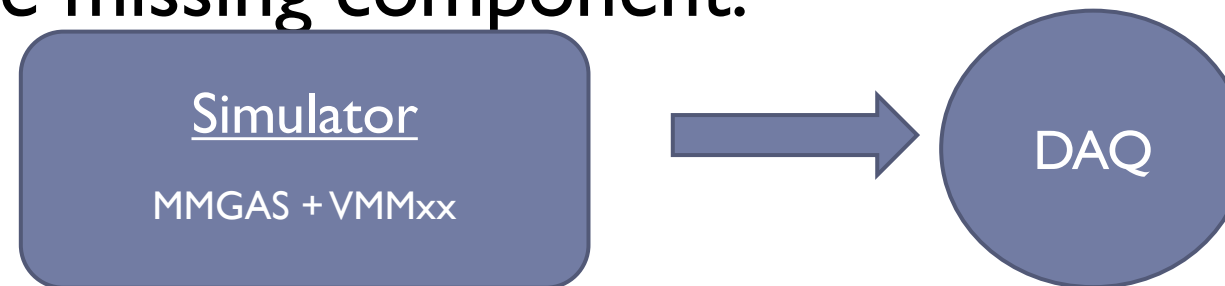
Introduction - Motivation



- ▶ In the framework of NSW Project update for ATLAS detector there is a need to test the DAQ system for the VMMxx electronic chip.



- ▶ There is ultimate need to simulate the response of VMMxx cards, since it is the missing component.





Introduction - Motivation



- ▶ The DAQ software system of the previous version of VMMxx is available.
- ▶ A computer simulator written in C++ using the QT framework was implemented, with the following features :
 - ▶ **1. Dynamic**
 - ▶ Very wide flexibility in the simulated UDP packages created
 - ▶ **2. Ability to load real datasets and simulate the behavior for the real case.**
 - ▶ **3. User friendly GUI for easy use by the physics users not only at CERN**
- ▶ In the following slides the code characteristics and the program itself are presented in detail.



Preview of the simulator



Words

...

Restrictions

Electronic Cards

Max Words per Packet

Words Len (bits)

Trigger Frequency (Hz)

How many events

Electronic Cards First IP

DAQ IP

DAQ Port

☒ Dummy Data ☐ Real Data

Config File: no file
Data File: no file

Words

frameCounter

Restrictions

Electronic Cards

Max Words per Packet

Words Len (bits)

Trigger Frequency (Hz)

How many events

Electronic Cards First IP

DAQ IP

DAQ Port

☒ Dummy Data ☐ Real Data

Config File: /home/angelos/Documents/temp/qtGUI/myGui/final2/config2.json
Data File: no file

Word Specs

minAppearance From Bit

maxAppearance To Bit

changeable

setOnlyOnce

trigger

position

minValue

maxValue

increase

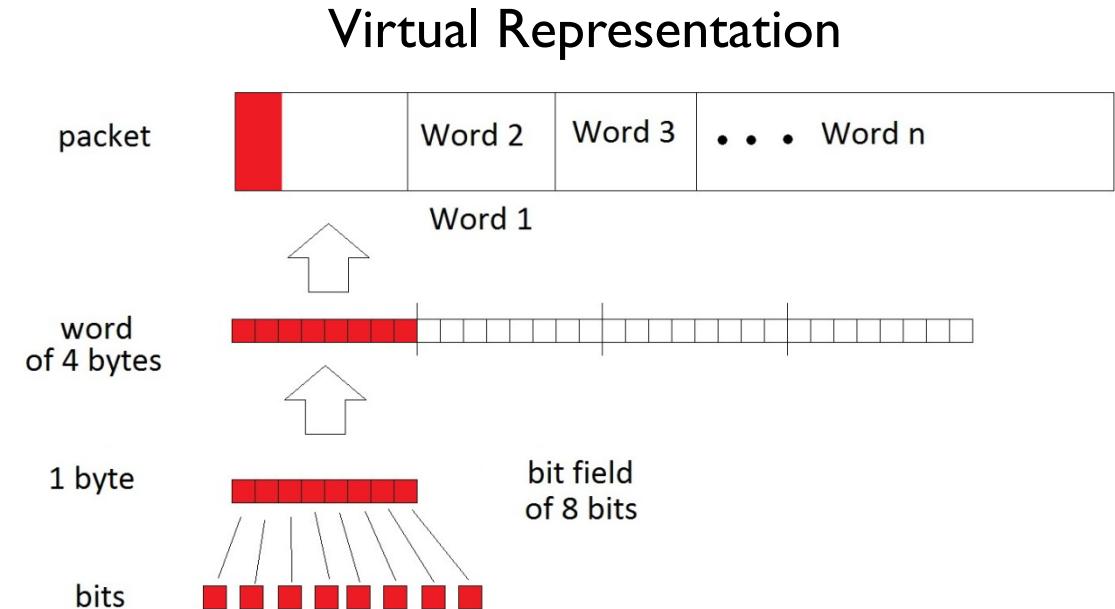
Program screen shot before any input

The window with a loaded file, with initialization information



Semantics

- ▶ Special terminology that will be used in the rest of the presentation :
 - ▶ **“BIT FIELD”** → A series of bits that can be treated together.
 - ▶ **“WORD”** → It's a bit sequence that is recognized by the current architecture as an instruction.
 - ▶ **“PACKET”** → It's equivalent of a post-office envelope. Consists of the headers and the main information(data).



- ▶ Normally the UDP header is taken from the system, but in our case we build a custom UDP header using raw data. This is implemented to allow for the user to define custom IP address for them VMMxx for the proposes of his simulation.



Code Structure



- ▶ Main purpose of the program is **to create** and **transmit** custom packages, with characteristics (“restrictions”) defined by the user.
- ▶ Each “part”(=bit field) of the “word” is treated according to the restriction referring to that bit field.
- ▶ Each restriction set by the user affects the corresponding bit field



Restrictions Panel



- ▶ Each restriction defines:
 - ▶ The bit field which is referring at:
 - ▶ From Bit $n_1 \rightarrow$ To Bit n_2
 - ▶ The minimum and maximum allowed values of this bit-field
 - ▶ The way in which the bit-field handles its value.
 - ▶ Either remain static
 - ▶ Change Randomly
 - ▶ Changing by step +1

From	Bit	<input type="text" value="0"/>	<input type="text" value="24"/>
To	Bit	<input type="text" value="23"/>	<input type="text" value="31"/>
minValue		<input type="text" value="0x0"/>	<input type="text" value="0x0"/>
maxValue		<input type="text" value="max"/>	<input type="text" value="NaN"/>
increase		<input type="text" value="+1"/>	<input type="text" value="Static"/>
		<input type="text" value="-"/>	<input type="text" value="-"/>



Words' Panel

Each “word” obeys also to a set of (user defined) restrictions :

▶ **Number of occurrences**

▶ **“Changeable” and/or “Send only one”**

▶ This is to gain CPU power.

▶ There are two case scenario where we can know the bit field value before we need it.

- ☐ The case where we know from the beginning that when we set the bits they will never change.

→ **STATIC WORD**

- ☐ And the case where a word is never going to change after the first time that the user (electronic cheap VMMxx) sets it.

→ **STATIC & CHANGEABLE WORD**

minApparence	<input type="text" value="1"/>
maxApparence	<input type="text" value="1"/>
changeable	<input type="text" value="true"/>
setOnlyOnce	<input type="text" value="false"/>
trigger	<input type="text" value="true"/>
position	<input type="text" value="0"/>

▶ **Trigger**

▶ Trigger refers to a word that changes only when the trigger hits. When a user(VMMxx) needs it, it just takes it with the current value.

▶ The only way for the value of that word to be changed is by Object of class Trigger.

▶ **Position**

▶ The position of the word in packet (1st 2nd ...)



Package Panel



- ▶ From this panel new words and restrictions can be created, as well as various parameters of the simulation can be regulated.
- ▶ **Electronic Cards:**
 - ▶ number of VMMxx to be used
- ▶ **How many events:**
 - ▶ the user can choose how many events he will run, either with dummy or real data
- ▶ **Electronic Cards First IP:**
 - ▶ The user can change the sender's IP. The IP's are generated one-by-one starting from the first-one.

The screenshot shows a configuration window titled 'Package Panel'. It contains several settings:

- Words:** A dropdown menu showing 'frameCounter' with '+' and '-' buttons.
- Restrictions:** A '+' button.
- Electronic Cards:** A text box with the value '16'.
- Max Words per Packet:** A text box with the value '10'.
- Words Len (bits):** A text box with the value '32'.
- Trigger Frequency (Hz):** A text box with the value '200'.
- How many events:** A text box with the value '3'.
- Electronic Cards First IP:** A text box with the value '192.168.2.1'.
- DAQ IP:** A text box with the value '127.0.0.1'.
- DAQ Port:** A text box with the value '6006'.
- Data Type:** Two radio buttons: 'Dummy Data' (selected) and 'Real Data'.
- Buttons:** 'Save', 'Load Config', and 'Load Data' at the bottom.



Test Run

- ▶ In the following example real data are used.
 - ▶ Set-up the description of data model
 - ▶ Load the root ntuple with the real data
 - ▶ Run the simulation and...
 - ▶ Capture the communicated packages with tcpdump sniffer.
 - ▶ `tcpdump -i lo 'udp port 6006' -c 10 -e -XX`

Data Model					
		byte 3	byte 2	byte 1	byte 0
frame counter		0x0	trig cnt1	trig cnt2	trig cnt3
srs data header		0x0	0x4C	0x4E	0x42
srs header info		board id	0x0	0x0	user def
data		bits 31-19 amp		bits 18-6 time	bits 5-0 ad



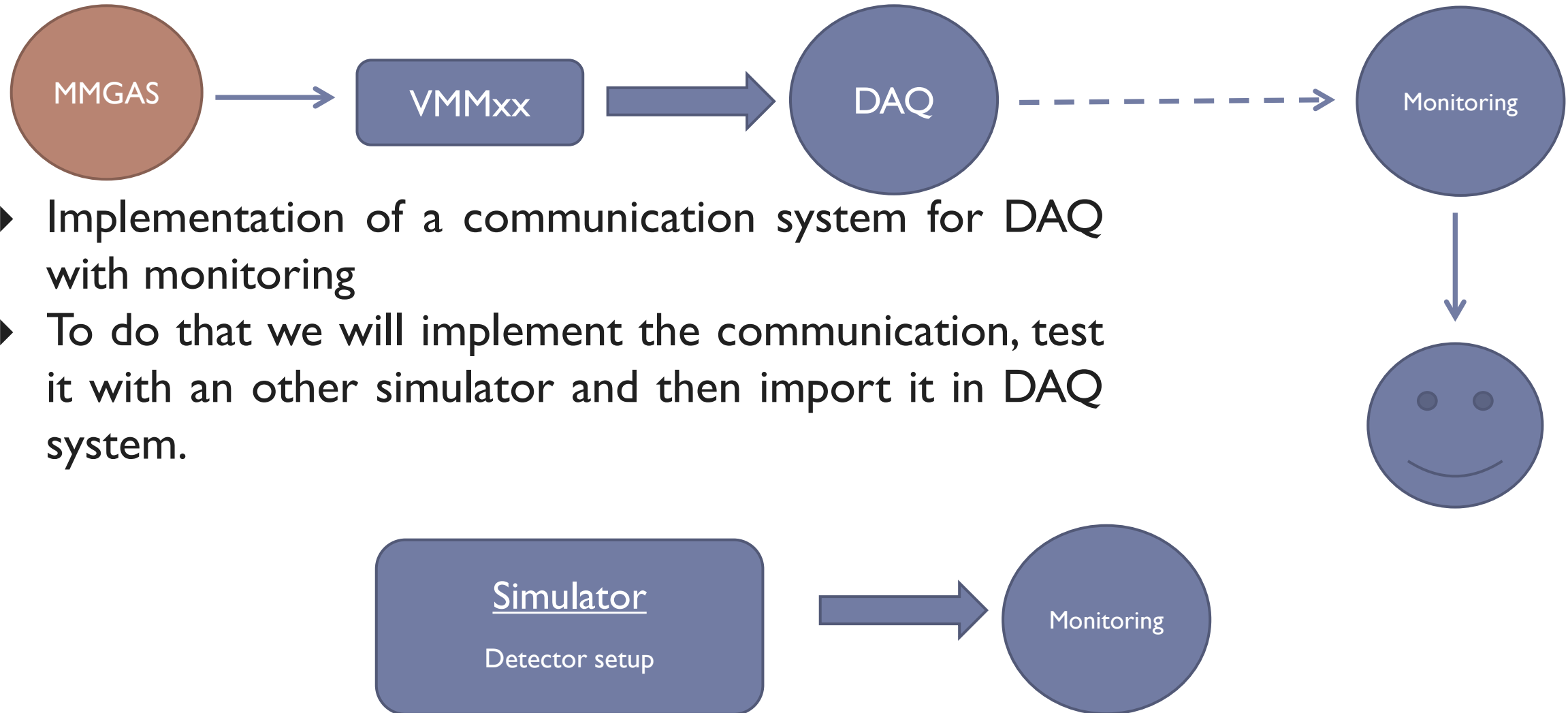
Read data with tcpdump

```
final2 : sudo - Konsole
File Edit View Bookmarks Settings Help
16:27:57.811974 00:00:00:00:00:00 (oui Ethernet) > 00:00:00:00:00:00 (oui Ethernet), ethertype IPv4 (0x0800), length
330: 192.168.2.19.6666 > localhost.x11-6: UDP, length 288
0x0000: 0000 0000 0000 0000 0000 0000 0800 4500 .....E.
0x0010: 013c b209 0000 ff11 c6ea c0a8 0213 7f00 .<.....
0x0020: 0001 1a0a 1776 0128 e81c 0000 0001 004c .....v.(.....L
0x0030: 4e42 1300 0000 20a9 c576 20c1 863a 20c0 NB.....v.....
0x0040: e634 20c1 4638 20c1 e63d 20c2 463e 20c1 .4..F8...=..F>..
0x0050: 8639 20c1 8637 20c1 6637 20c1 6638 20c1 .9...7...f7...f8..
0x0060: c639 20c2 463e 20c2 063c 20c1 2635 20c0 .9..F>...<..&5..
0x0070: e634 20c1 a63a 20c2 463e 20c1 e63b 20c1 .4.....F>...;..
0x0080: 8637 20c1 8639 20c1 8639 20c1 8638 20c1 .7...9...9...8..
0x0090: 8639 20c1 c63c 20c1 6638 20c1 0634 20c1 .9...<..f8...4..
0x00a0: 8638 20c1 e63d 20c1 e63b 20c1 8637 20c1 .8...=...;...7..
0x00b0: 4636 20c1 2638 20c1 6637 20c1 a63a 20c2 F6..&8..f7.....
0x00c0: 063e 20c1 863a 20c1 4637 20c1 4637 20c1 .>.....F7..F7..
0x00d0: 663a 20c1 c63c 20c1 c63b 20c1 8639 20c1 f:...<...;...9..
0x00e0: 4637 20c1 4637 20c1 a639 20c1 e63c 20c1 F7..F7...9...<..
0x00f0: 863a 20c1 4636 20c1 6637 20c1 8639 20c1 ....F6..f7...9..
0x0100: 8639 20c1 c63b 20c1 e63a 20c1 6638 20c1 .9...;...:..f8..
0x0110: 0634 20c1 6638 20c1 e63d 20c1 a63b 20c1 .4..f8...=...;..
0x0120: 6637 20c1 6636 20c1 863a 20c1 8639 20c1 f7..f6.....9..
0x0130: a639 20c1 e63a 20c1 6639 20c0 e634 20c1 .9.....f9...4..
0x0140: 4636 20c1 c63a 20c2 263f F6.....&?
```

- ▶ UDP Header
- ▶ frame counter
- ▶ srsDataHeader
- ▶ srsHeaderInfo
- ▶ Data
(no line)



Future Steps



- ▶ Implementation of a communication system for DAQ with monitoring
- ▶ To do that we will implement the communication, test it with an other simulator and then import it in DAQ system.



Thank you

I would like to thank the following people who gave me the opportunity to be here and also help me during my time here.

- Prof. Theodora Papadopoulou
 - For giving me the opportunity to be here and running all the background procedures
- Rector Prof. Konstantinos Masselos
 - For giving me the opportunity to be here
- Dr. Nectarios Benekos
 - For being my supervisor and for teaching me all the interesting things at test-beam and for being so patient.
- Prof. Theo Alexopoulos and all NTUA-HEP members
 - For giving me the opportunity to join the NTUA group and giving me the chance to work with them

