White Rabbit and KM3NeT

Peter Jansweijer, on behalf of KM3NeT
Outline

1. Intro:
   - A multi-km$^3$ neutrino telescope in the deep-sea

2. KM3NeT daq from a technical perspective
Potential neutrino sources

- Supernova Remnants
- Pulsar Wind Nebula
- Gamma-Ray Burst
- Dark Matter
- Active Galactic Nuclei
- Cosmogenic neutrinos
- Micro Quasars

KM3NeT
Opens a new window on our universe
Detection method

Neutrino-induced muons in the deep sea

Strings with optical sensor modules

Cherenkov light

muon

neutrino
Deployment

640 string with optical sensors in the deep sea at 3-5 km depth
KM3NeT Artist Impression

12,800 DOMs in the deep sea at 3-5 km depth
with point-to-point connection to shore

Volume: \( \sim 5 \text{ km}^3 \)

640 strings
20 DOM/string
12800 DOMs

\( \sim 860\) m

100 m
Digital Optical Module (DOM)

Upper Hemisphere
12 PMTs

Lower Hemisphere
19 PMTs

Central Logic Board (CLB)

PMT Base:
High Voltage Supply
Analog Front-End
Central Logic Board (CLB)

- Readout 31 TDC’s with 1 ns resolution
- “Knowledge” of absolute time (1 ns resolution)
- Data pushed from PMTs to Shore Station
- I2C: PMT-HV, Threshold, Compass, Tilt
- Other IO: Temp, Nano beacon, Acoustics
- Firmware must be reconfigurable
- Low Power
- Low Cost
- Part of a scalable system (with respect to the complete detector)
- Highly reliable
White Rabbit is going deep-sea!
Central Logic Board (CLB)

Xilinx Kintex-7

10 PMTs

UTC time & Clock (PPS, 125 MHz)

Data

Control

Point to Point interconnection

Wishbone shared bus (32 bits)
Shore Station

1. Each DOM synchronizes to the absolute time (using White Rabbit).
2. Each DOM receives a look up table with IP addresses while configuring the detector.
3. All DOMs start at an absolute point in time which was communicated via a command over the White Rabbit network.
4. All DOMs start their first time-slice at exactly the same time.
5. All data is IP/UDP formatted and passed to the IP number corresponding to the time slice.
6. After ‘n’ time slices, first PC is again selected to process the data.

<table>
<thead>
<tr>
<th>Time Slice</th>
<th>IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>192.16.x.1</td>
</tr>
<tr>
<td>2</td>
<td>192.16.x.2</td>
</tr>
<tr>
<td>:</td>
<td>:</td>
</tr>
<tr>
<td>n</td>
<td>192.16.y.z</td>
</tr>
</tbody>
</table>

Servers may be used for storage and Pre-processing.
Work to be done (with respect to White Rabbit)

- There is no PCIe in the deep sea…
  - Debug via UART
  - Firmware reconfigurable; (new) software loaded together with (new) FPGA configuration (Block Memory Mapping file)
- Explore the Kintex-7 deterministic latency GTX
  - Future Artix-7 may be even more cost effective in terms of money and power
- Implement Oscillators, DACs (used by Soft-PLL) and SFP on FMC card which is to be plugged onto Xilinx KC705 Evaluation board (Mesfin Gebyehu)
- 1st goal: Replace the (slave) SPEC in the SPEC<-> SPEC test setup with the KC705 implementation for validation.
- Study Shore Station (e.g. White Rabbit Switch v3) broadcast
To be studied: Network Broadcast

Can this be done with the WR Switch-V3? Need Firmware/Software change?

Main Electrical Optical Cable (MEOC) May be 100 Km long…
Thank you!
(White Rabbit community)
Backup Slides
LM32, three boot types

1. Generate ROM image before synthesis (used for functional simulation debug)
   - Describe a generic RAM using “init” file
   - Useful for functional simulation
   - Incorporates “boot.elf” in the block-rams

2. Download “elf” via an external interface in a running system (used for software debug)
   - Useful for debugging purposes (the SPEC uses PCIe or JTAG)
   - No BMM=Block Memory Mapping file needed

3. After Place&Route (will be used in final CLB)
   - Merge “FPGA.bit” file and “boot.elf” file (data2mem)
   - This needs BMM=Block Memory Mapping file
   - The “bit” file which is outputted by the merge can be used as updated configuration file
   - This avoids synthesis each time software is updated
BMM (Block Memory Mapping) File

Fpga.bit

ROM
RAMB36 0x00
RAMB36 0x00

RAM
RAMB36 0x00
RAMB36 0x00

Fpga_elf.bit

ROM
RAMB36 0x12
RAMB36 0x34

RAM
RAMB36 0x56
RAMB36 0x78

hello.elf

ADDRESS_SPACE lm32_memory RAMB36 [0x00000000:0x00011FFF]
BUS_BLOCK
ul_u0/U0/xst_blk_mem_generator/gnativebmg.native_blk_mem_gen/valid.cstr/ramloop[15].ram.r/v6_init.ram/NO_BMM_INFO.SP.SIMP
LE_PRIM36.ram [31:30];
END_BUS_BLOCK;
END_ADDRESS_SPACE;

ADDRESS_SPACE lm32_data_memory RAMB36 [0x00000000:0x00011FFF]
BUS_BLOCK
u6_u0/U0/xst_blk_mem_generator/gnativebmg.native_blk_mem_gen/valid.cstr/ramloop[15].ram.r/v6_init.ram/NO_BMM_INFO.SP.SIMP
LE_PRIM36.ram [31:30];
END_BUS_BLOCK;
END_ADDRESS_SPACE;

Data2Mem

fpga.bmm

hello.elf

ADDRESS_SPACE lm32_memory RAMB36 [0x00000000:0x00011FFF]
Optical Network (Simplified)

- Broadcast
- Shore station
- Junction Box
- DU-container
- Slow Control $\lambda_1$
- Multiple $\lambda$
- Single $\lambda$