The new HADES Trigger Bus

Ingo Fröhlich
The HADES Trigger

- All black arrows are stand-alone protocols
  - Trigger bus (red)
  - MU bus (green)
Trigger Bus: Simplified View

- Communication done via 3 flat cables
Problems

• Too many individual components
  – All with their own debug tools etc...
• Thinking in „units“ (Detector Trigger Unit, Central Trigger Unit, Matching Unit, Image Processing Unit)
  – Very often „project“ of a PhD-Theses, optimized for algorithms, not for communication
• Too many individual links/busses (IPU-Link, Trigger Bus, TIP-concentrator, CTU-MU, DTU-Backplane, IPC-Link, RACE-PRC, etc...)
• Links have no debug feedback
• Need to login on several CPUs -> Already now a mess
The New Concept
(for the TRB Project)

- Merge at least MU- and trigger bus
- Use point-to-point connections (optical links)

Talk of M. Traxler
13.3.2006
New HADES Trigger Network

- Requirements
  - Latency*hub-levels adds to deadtime
    - Low-latency (150-250ns)
  - No high-level OSI-layers possible
    - No data loss on the network layer
    - Needs back-pressure
  - Trigger logic hidden in the network layer
  - Adaptable to small FPGAs and any medium bandwidth

Central Trigger System

LVL1 uplink

LVL2 downlink

LVL2 uplink

Detector Trigger and Readout System

Optical network
New HADES Trigger Network

- 3 simple layers

**Data**
- Source/target address
- Trigger tag & code
- Slow control

**Network**
- Locking & busy logic
- Data integrity
- Multiplexing

**Medium**
- Optical, 8B/10B
- LVDS-cable
- Connector for TRB-AddOns

Central Trigger System
- LVL1 uplink
- LVL2 downlink
- LVL2 uplink

Detector Trigger and Readout System

Optical network
Virtual Channels

- One channel is e.g. LVL1, LVL2, IPUxxx
- Channels and pathes have separated buffers
  - Avoiding deadlocks
Network: Basic Idea

- HADES busy-logic:
  - One channel is blocked until all receivers have released it
  - Old system: wired-or
  - No routing needed
- Realisation in the HUBs:
The Hub

- **IBUF**
  - A fifo with 2 logical buffers
  - Size not fixed
  - Transmitter has to know the size

- **OBUF:**
  - Pulling the data from the IBUFs
  - Size normally =0
  - May insert additional control words for handshake logic

The words are pushed into the IBUF, but pulled from the MEDIUM via the OBUF
Speed defined by MEDIUM!
Protocol

- Fixed 64 bit word size. Each word is a:
  - Header (HDR)
  - Data (DAT)
  - Termination (TRM)
  - End-of-Buffer (EOB)
  - Acknowledge (ACK)

- Words are forming buffers
  - Matching the *real* IBUF size
    - min 3 words
  - No data loss!
  - 2 buffers may be sent before ACK

- Buffers are forming packets
  - Packet size 1 up to infinity
Data Structure

- **P (Path)**
  - 0=INIT, 1=REPLY

- **CID (Channel ID)**
  - Which of the 16 Channels is used?
  - Allows easy de-multiplexing

### Table: Description

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR</td>
<td>0x1</td>
<td>Initial Transfer</td>
<td>Source adress</td>
<td>Target adress</td>
<td>Bit data</td>
</tr>
<tr>
<td>TRM</td>
<td>0x2</td>
<td>Initial Transfer Terminated</td>
<td>Source adress</td>
<td>Target adress</td>
<td>Error pattern</td>
</tr>
<tr>
<td>EOB</td>
<td>0x3</td>
<td>End of Buffer</td>
<td>0x0</td>
<td>CRC (optional)</td>
<td>res.</td>
</tr>
<tr>
<td>ACK</td>
<td>0x3</td>
<td>End of Buffer</td>
<td>0x1</td>
<td>Length of buffer</td>
<td>res.</td>
</tr>
<tr>
<td>DAT</td>
<td>0x4</td>
<td>Data word</td>
<td>Data</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Optional protection via (7,4) Hamming code

48 Bit data load

No routing, just for info
Data Application

- The user should not deal with internals
- Data (one event) may be messed up, but not the network
- IOPIPE: Listen to a fifo, and write to a fifo (only data)
- Additional register control for start the transfer, set error bits, see status etc...

IBUF

OBUF

FIFO

IOPIPE interface

User

Pull

Push

Dummy

„Termination“
HADES trigger bus: One early TRM is enough
- Bit data for trigger tag, code
- Error pattern for... well, errors
- Only broadcasts

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR</td>
<td>0x1</td>
<td>Initial Transfer</td>
<td>Source address</td>
<td>Target address</td>
<td>Bit data</td>
</tr>
<tr>
<td>TRM</td>
<td>0x2</td>
<td>Initial Transfer Terminated</td>
<td>Source address</td>
<td>Target address</td>
<td>Error pattern</td>
</tr>
<tr>
<td>EOB</td>
<td>0x3</td>
<td>End of Buffer</td>
<td>0x0</td>
<td>CRC (optional)</td>
<td>res.</td>
</tr>
<tr>
<td>ACK</td>
<td>0x3</td>
<td>End of Buffer</td>
<td>0x1</td>
<td>Length of buffer</td>
<td>res.</td>
</tr>
<tr>
<td>DAT</td>
<td>0x4</td>
<td>Data word</td>
<td>Data</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
IPU Bus Application

- One Addressing cycle (broadcast)
  - All IPUs are sending data at the same time
- Data merging in the hubs
  - Needs HDR retransmit
  - In other words: Buffers may be

---

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR</td>
<td>0x1</td>
<td>Initial Transfer</td>
<td>Source address</td>
<td>Target address</td>
<td>Bit data</td>
</tr>
<tr>
<td>TRM</td>
<td>0x2</td>
<td>Initial Transfer Terminated</td>
<td>Source address</td>
<td>Target address</td>
<td>Error pattern</td>
</tr>
<tr>
<td>EOB</td>
<td>0x3</td>
<td>End of Buffer</td>
<td>0x0</td>
<td>CRC (optional)</td>
<td>res.</td>
</tr>
<tr>
<td>ACK</td>
<td>0x3</td>
<td>End of Buffer</td>
<td>0x1</td>
<td>Length of buffer</td>
<td>res.</td>
</tr>
<tr>
<td>DAT</td>
<td>0x4</td>
<td>Data word</td>
<td>Data</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Media Layer Interface

- Hides the current implementation of the medium from the NPL

<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT_DATAREADY_OUT</td>
<td>Data word is reconstructed and ready to be read out by the Internal logic</td>
</tr>
<tr>
<td>INT_DATA_OUT[0...55]</td>
<td>Data word</td>
</tr>
<tr>
<td>INT_READ_IN</td>
<td>Internal logic is reading from the FIFO</td>
</tr>
<tr>
<td>INT_ERROR_OUT[0..2]</td>
<td>Status bits</td>
</tr>
<tr>
<td>MED_DATAREADY_IN</td>
<td>Data word is offered by the internal logic or FIFO</td>
</tr>
<tr>
<td>MED_DATA_IN[0...55]</td>
<td>Data word</td>
</tr>
<tr>
<td>MED_READ_OUT</td>
<td>Media interface reads a word</td>
</tr>
<tr>
<td>MED_ERROR_IN[0..2]</td>
<td>Status bits</td>
</tr>
</tbody>
</table>
More Features

- Adresses: 12 bits are enough for the TRB system
  - 4 Bits could be used as a minor number
  - E.g. add-on board, virtual components...
- Also protocol for the TRB<->AddOn communication
- Channel 15 should emulate a remote bus (part of the Data Layer)
  - the way hubs are programmed
  - basic state machine states should be readable
    - Common monitoring from a central place
- A lot of spare word types
  - Open for future extensions