

# Data interface between Pixel Converter and Pixel Router

---

F. Formenti, A. Kluge

Working Document Version 1, 9 January 2001

**This is a working document and this is subject to changes and updates. At the moment this note is very crude. It will be updated and completed in the future.**

## 1. Interface description

---

The router accesses the event data in a memory mapped manner. Event data are presented as 32 bit words. Each out of 6 pixel converter daughter board contains one event memory storing up to eight events of one half stave. Data of one event are contained in a half stave frame (as described in the note 'raw data format of one SPD sector as seen by the data acquisition'). The advantage of a memory mapped data access is that the router can access the same data one or more times. This allows for a future upgrade where the router might process algorithms using the entire event data. Two control words, a flush event register and a test/run register control the data transfer.

The read process is controlled using a 19 bit (bits 18 downto 0) address bus. Bits 17 and 16 are used to select either the event memory, control word 0, control word 1 or the flush event register. Bit 18 is always 0 except when the test/run register is accessed. The address space of the event memory is 16 bits wide.

Control word 0 contains event related information. It is available for up to eight stored events. The address is bit 16 asserted to 1, bit 17 to 0. Bits 2 downto 0 indicate the control word of which event is to be read with event 0 (bits 2..0 = 0)

being the oldest event. Some of the status information in control word 0 cannot be attached to a single event. As result these status bits are copied to all eight control words. An example is the link ready signal. The entities of control word 0 are described:

- event ready (last word received): declares that at least one full event is located in the event buffer. The router polls this bit in order to know the arrival of new data. (bit 0). Additionally a single additional interrupt line for the event ready signal which is active all the time and not depending on the address bus is provided.
- event number. (bit 9..1)
- parity error: is active if one or more parity errors have been detected in the entire event. (bit 10)
- link down error: is active if the link was down at least once during the transmission of the corresponding event. (bit 11)
- format error: is active if the link receiver has detected a format error in the transmitted data. (bit 12)
- single event upset: is active if a single event upset which has not been able to recover has been detected (reset action is required). (bit 13)
- pixel control error: is active if the pixel control link is out of synchronisation. (bit 14)
- temperature high: is active when the preset temperature has been exceeded. (bit 15)
- link ready: is active once the link is ready. (bit 16)

Control word 1 contains start and end address of the corresponding event. It is available for up to eight stored events. The address bit 17 is asserted to 1, bit 16 asserted to 0. Bits 2 down to 0 indicate the control word of event to be read with event 0 (bits 2..0 = 0) being the oldest event.

- Start address of corresponding event. (bits 31 down to 16)
- End address of corresponding event. (bits 15 down to 0)

As the data access for the router is not being done with a FIFO the pixel converter does not know when data of an event is not needed any more. As a consequence one so called 'flush event' register is provided per pixel converter. When the pixel router accesses this register (write access) the pixel converter will free the corresponding event memory for further data and advances the positions of the data in the control word registers. The address for the flush registers is bit 17 and bit 16 asserted to 1.

Depending on implementation of the router the address space of the pixel converter can be increased by three bits in order to decode the corresponding half stave (see table \*). This would be the case when the event memories of all six half staves would be shared.

**TABLE 1.**

register	bits 21..19 *)	bit 18	bit 17	bit 16	bit 15..0
event data	half stave	0	0	0	data line
control 0	half stave	0	0	1	event
control 1	half stave	0	1	0	event

TABLE 1.

register	bits 21..19 *)	bit 18	bit 17	bit 16	bit 15..0
flush	half stave	0	1	1	-
test/ $\overline{\text{run}}$	half stave	1	0	0	-

---

## 2. Access sequence

---

**The access sequence the router has to comply with is described.**

In idle mode the router polls the content of the event ready bit in control word 0. If it is asserted at least one event is ready to be read out in the corresponding pixel converter daughter board. In order to read control word 0 the corresponding address has to be provided on the address bus. In the next step the router reads both the start and end address of event 0. Using these addresses the event data can be read out. Once the read process is terminated the router must access the flush event register in order to notify the pixel converter that it can delete the corresponding event data. Now the sequence starts over again.

All data transmissions are synchronous to the router system clock. A strobe signal provided by the router validates the address bus on the rising edge of the system clock. A write/read signal indicates the data transmission direction.

For error checking the router also can write data into the event memory. The event memory is a dual port memory. Any write access to the memory either from the pixel converter itself or the router is done via the so called top port. Read access for the router is done via the so called bottom port. As a result the test write access to the memory is performed via the top port as well and thus has a different address. However, the address room for the two ports are identical. The address decoding is done on the pixel converter boards using the write/read signal. This means the router writes and reads to the same address, although internally different ports are being used. In order to write to the memory the test/run register has to be set accordingly.

---

## 3. Integration of system

---

As the pixel system is a highly distributed system with several design teams involved the integration must be done with utmost care. It must be made clear that interface between design blocks are clearly understood and well defined. At the very moment this has not been accomplished to a satisfactory extent. As a consequence it must be made certain that different design blocks work together before the prototype building phase starts. This can be performed by system wide digital simulations. It is necessary to simulate the entire data stream using hardware description languages. This has been performed for a part of the data chain, namely: from pixel chip, to pilot chip, to GOL (Glink transmitter), to link receiver. The missing parts are pixel converter and router. Up to now the hardware description language used is VERILOG. However, implementing VHDL in the simulation is not a problem, whereas AHDL can

**not be used for system wide simulations. Designers of sub blocks are required to provide behavioural description code in either VHDL or VERILOG.**

**Preliminary**