Interface to the JRA1 trigger

Overview

The central Trigger Logic Unit provides the trigger and an event number to the subdetectors, it is desirable to get the event number into the data stream as early as possible in the readouttree. In addition must our system provide a busy to the TLU. In our case do we need also to distribute the sampling clock for the front end electronics. The ALICE RCU receives trigger and clock through the LHC TTC system, in our case it must receive the trigger and clock from our own source. A modification to the RCU is said to exist to run in a mode without the TTC.

The event number is read by the subdetectors by clocking it from the TLU. In our system will there be 4 RCU's. Since we will get only one line to the TLU do we have to build a module, in this document called distributor, that:

- Receives the trigger
- Sets the busy
- Reads the event number
- Send the trigger to the RCUs as Level1. The RCU needs a level2 trigger.
    - **Q**: how is level2 trigger sent? Internally generated in RCU or must it be generated from outside?
- Distribute the sampling clock to the RCUs
- Deliver the event number to the destination
- Receive busy from the destination and reset the TLU busy

There are several options discussed for the destination, each of them is described below. One could consider to make a combination of these options, e.g. PCI I/O card and one of the others, to have a fallback solution.

Destination = PCI I/O card in PC DAQ

Fig 1. shows an overview of this mode. The distributor reads the event number, and output it to the PCI card, the simplest could be one line per bit of the event number, and one line for the reset of the busy. In this case is it the DAQ software that reads the I/O card and puts the data into the datastream, and resets the busy. The advantage is that this system requires no modifications to the frontend hardware. It requires extra software for the readout. The disadvantage is that this do not put the event number into each of the RCUs datastreams, i.e. one might loose the synchronization inbetween the RCUs. In the data from the RCU is a header, where one 24 bit field contain an incremental, unsigned number, which is put to zero at a reset, to my knowledge the only event id provided by the front end. I once looked around for PCI I/O cards, but did not find any with a linux driver. **Q**: Any proposal?

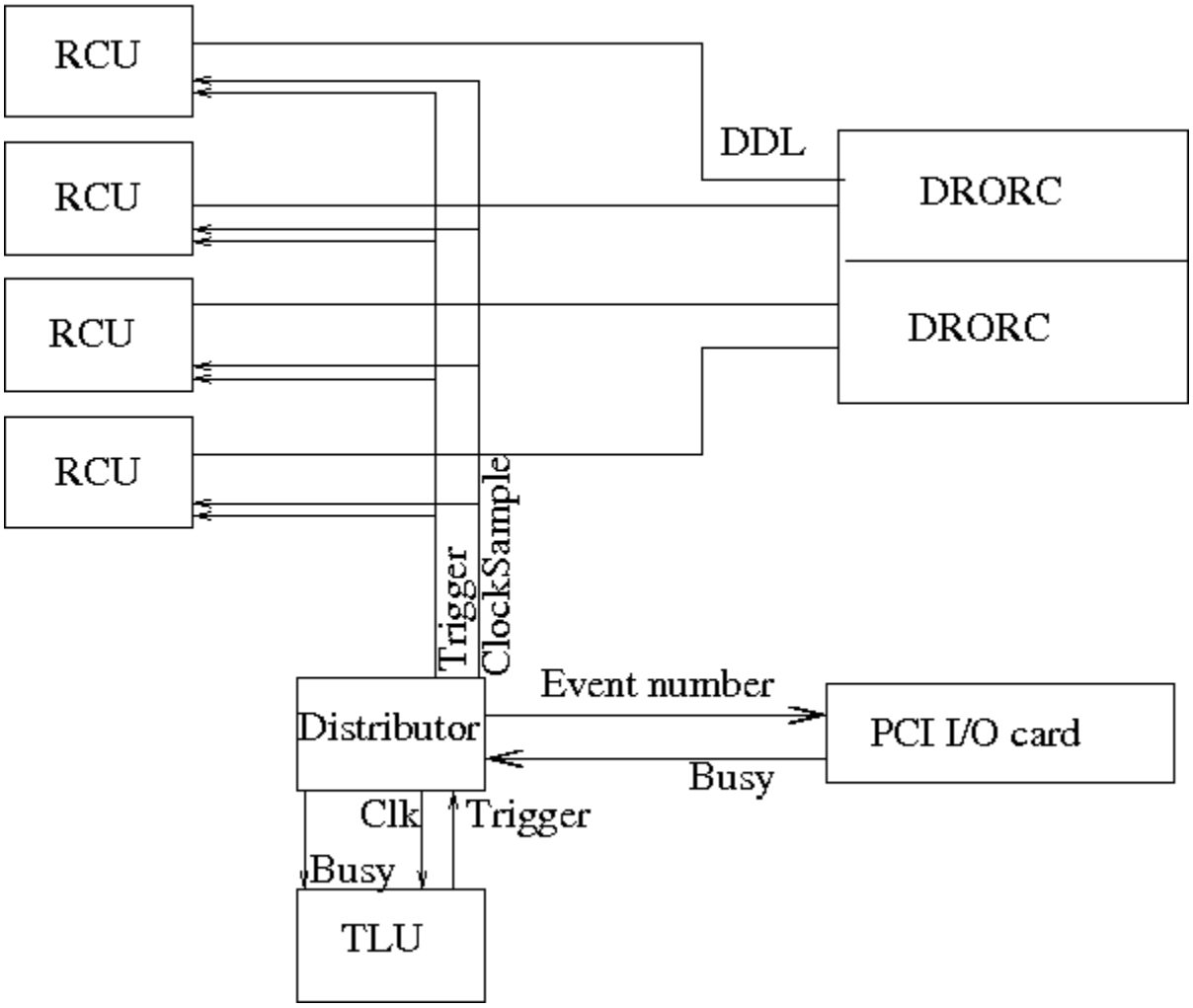

Fig. 1: Event number read and busy reset from the PC DAQ using a PCI I/O card.

Destination = RCU

Fig 2. shows an overview of this mode. The distributor reads the event number, and clocks it to each RCU. The RCU then puts the event into the event header, e.g. inserts it into the place of one the Events Ids normally taken from the TTC. The RCU must when the readout is finished reset the busy. The advantage is that the event number is part of the data from each RCU. The disadvantage is that the RCU must be modified, hardware and firrmware. Unknown if possible and by who!? Since I do not have any manual/documentation on the RCU is it difficult to know how it can be made.

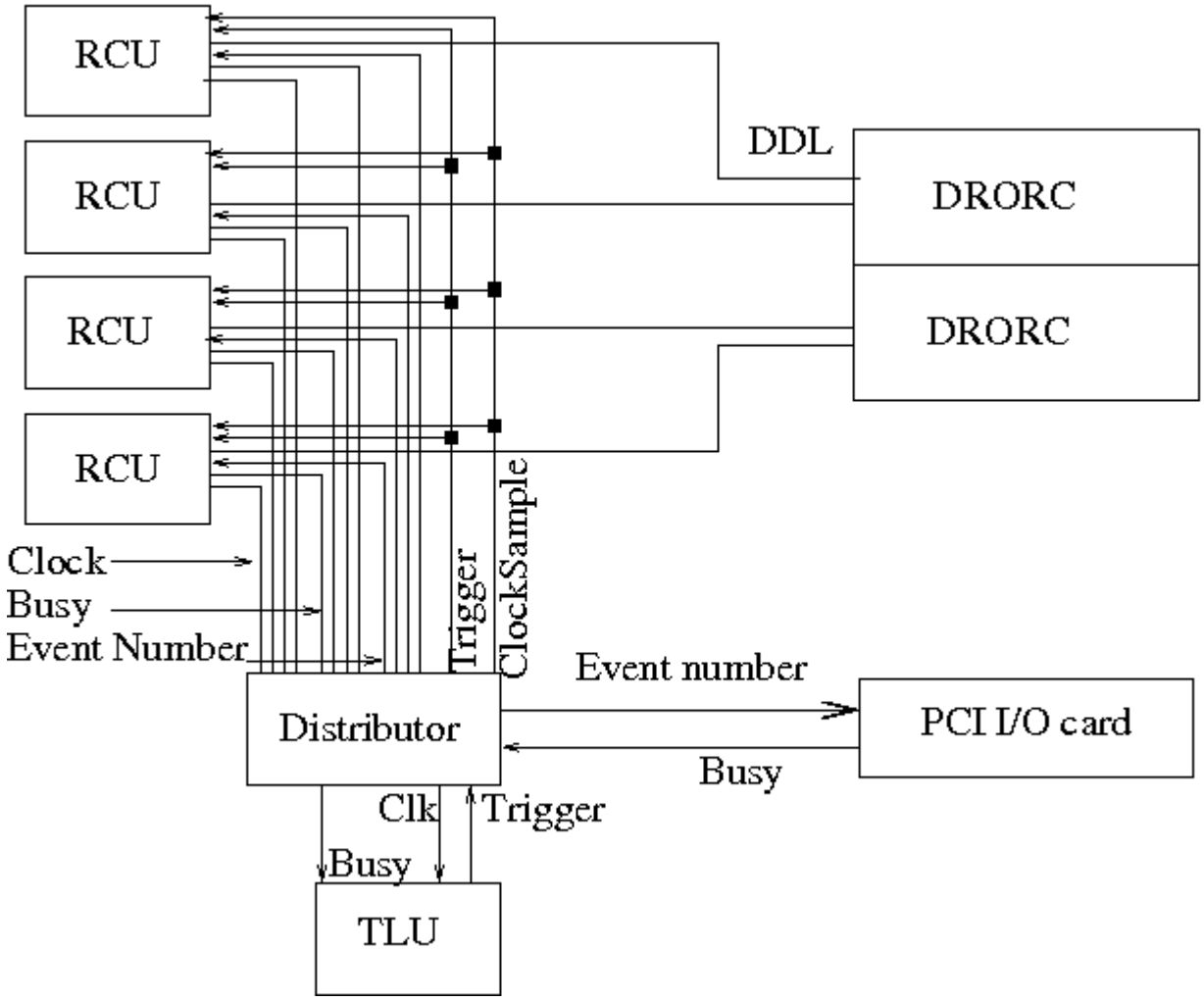

Fig. 2: Event number read and busy reset from the RCU. The PCI alternative is also shown.

Destination = modified Front End Card

Fig 3. shows an overview of this mode. The distributor reads the event number, and clocks it to a special modified Front End Card.This card is from the RCU point of view identical to a true FEC. The event number would then appear as data from an ALTRO. The advantage is that the RCU is not needed to be modified. The modified FEC may also be used to read other things, e.g. timing of trigger PM signals if one want to know the time between trigger and sampling clock phase. The disadvantage is that this card may be not so easy to make!?

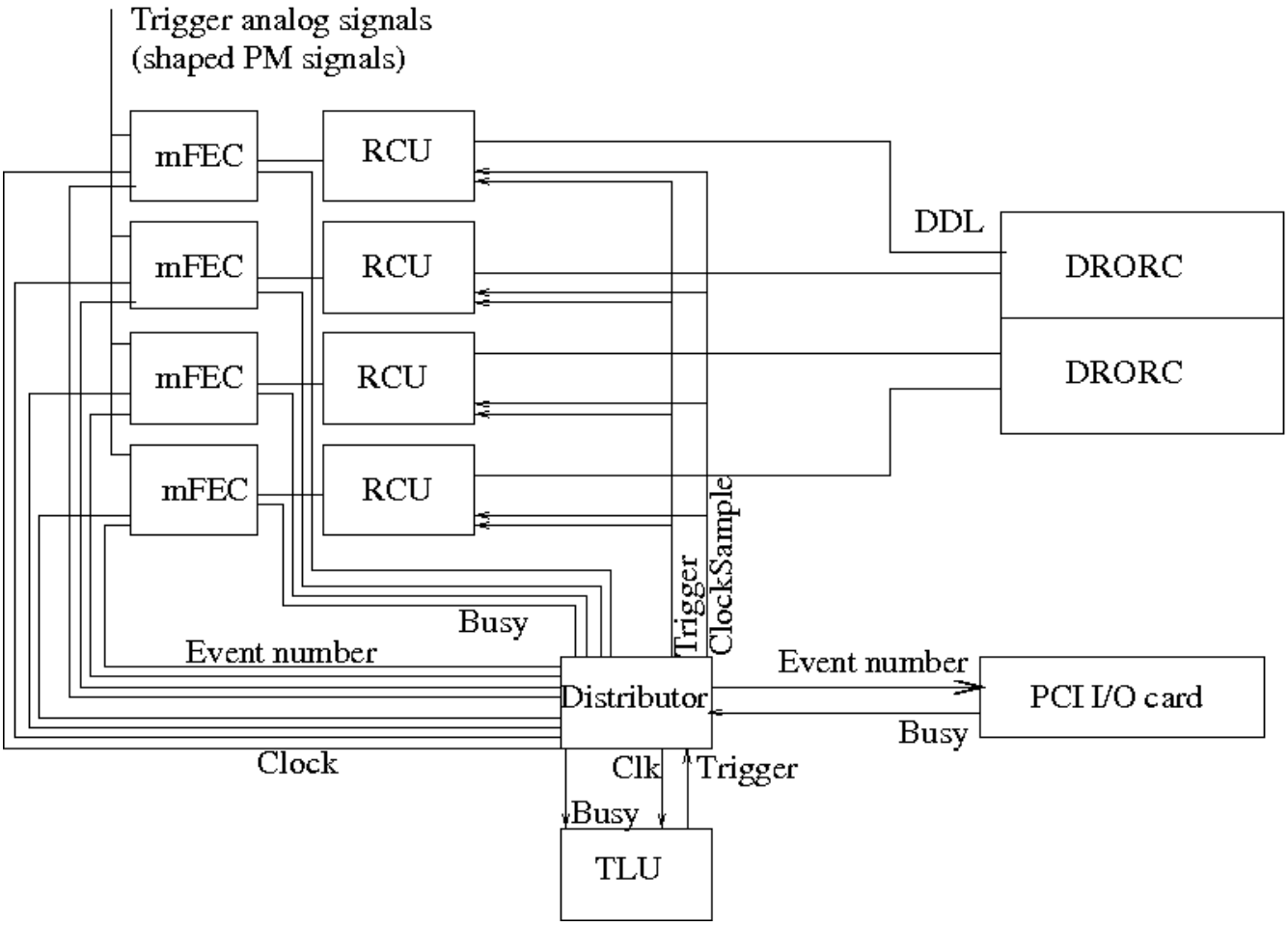

Fig. 3: Event number read and busy reset from a special modified FEC. The PCI alternative is also shown.