

# COOR Communication Conventions

scott snyder

July 22, 1997

This note discusses the general conventions which COOR uses when talking to the ‘trigger’ subsystems. For purposes of this note, ‘trigger’ also includes the interface to the token ring downloads and the data logger. The intention is to try to make these conventions uniform across all these systems. System-specific commands are discussed separately.

- D0IP is used for the underlying transport. One command or acknowledgment is sent per D0IP message.
- Commands are always sent one way: from COOR to the trigger. If a trigger needs to make asynchronous requests, it must explicitly open an additional command channel to COOR.
- Every command should result in an acknowledgment from the trigger back to COOR. In some cases, the order of acknowledgment may not be the same as the order in which the commands were issued. In order to keep straight the correspondence between commands and acknowledgments, each command has a command id which is sent with the command and returned with the acknowledgment. Triggers should not assume anything about the format of this id, other than that it consists of printable characters, contains no whitespace, and is no longer than 32 characters.
- Commands come in two types: *immediate* or *batched*.
  - For an immediate command, COOR will expect a reply from the trigger before sending another command.

- COOR may send multiple batched commands without an acknowledgment. A batch is implicitly started by the first batched command received. It is ended by the special command ‘configure’. When ‘configure’ is received, the batched commands should describe a consistent configuration. Once the ‘configure’ command is sent, no additional commands will be sent until every command in the batch (including the ‘configure’ command) has been acknowledged.

The trigger can start processing commands at any time. It can process them as they are received, or it can queue them up and process them all once the ‘configure’ command has been received. Acknowledgments can be sent before the ‘configure’ command arrives. Except for the ‘configure’ command (which must be acknowledged last) and for commands within a block, commands may be acknowledged in any order.

- General command format.
  - Commands should be case-insensitive.
  - COMMAND-ID COMMAND [ARGS] ...
- Acknowledgment format.
  - COMMAND-ID ack STATUS [TEXT]
  - COMMAND-ID is copied from the corresponding command.
  - STATUS is either ‘ok’, ‘bad’, or ‘more’.
  - TEXT is either status information being returned from the command (if STATUS is ‘ok’) or an error message (if STATUS is ‘bad’). If TEXT would take more than one line, each line should be sent separately, in order. For all except the last line, STATUS should be ‘more’. For the last line, STATUS should be the final value.
  - If STATUS is ‘ok’ and the command was not requesting any information, then TEXT may be blank. If STATUS is ‘bad’, TEXT should contain a brief error message.
- Some common command names which should be recognized by all systems. (They might not have to do anything for some of them, but they should be able to recognize and acknowledge them.)

- configure: As discussed above.
- begin\_block
- end\_block: These are not really commands, per se, and should not be acknowledged. (For uniformity, they will still have a COMMAND-ID, but it may be a dummy.) Commands which occur between begin\_block and end\_block must be processed in the order in which they were sent.  
 Only batched commands may appear within a block, and ‘configure’ may not appear within a block.  
 Not all order dependencies will be protected within a block. In general, if it doesn’t make sense to reorder the commands they won’t be put into a block. (This will probably only be used for tkr downloads.)
- init: This is an immediate command. The trigger should immediately end all ongoing DAQ, release resources, and restore all programming to the default state.  
 It should not be necessary to reboot nodes, reload FPGAs, etc. in response to this command. The assumption being made is that the targeted system is still sane, but is in an unknown state.  
 This is sent by COOR on startup, and whenever it reinitializes.
- start\_run RUNNO SPECTRIGS
- stop\_run RUNNO: Start or stop run number RUNNO. SPECTRIGS is a list of level-1 (and level-2) specific triggers participating in the run. (This would probably be a space-separated list, possibly with the notation FIRST:LAST to specify a range.) Some systems won’t have to actually do anything for these commands.
- pause
- resume: Pause and resume processing events. Systems not in the readout chain won’t have to do anything for these commands.
- begin\_store STORENUM
- end\_store STORENUM: Note that a store is beginning or ending.