artdaq Utilities - Feature #21328

A set of subconfigurations which realistically model an experiment should be added to simple_test_config

11/07/2018 01:52 PM - John Freeman

Status:	Resolved	**Start date:**	11/07/2018
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	80%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour
Experiment:	-	**Co-Assignees:**	

Description

Now that subconfigurations are supported by DAQInterface (Issue #21226), Kurt's suggested that we add some to simple_test_config which look realistically like those an experiment might use, and which could be used as a jumping-off point for actual experiments designing their own subconfigurations. After discussing this with him, ideas we have are:

- That different types of artdaq process each have their own subconfiguration, except for boardreaders, which will be further subdivided into fragment generator type. E.g., something like "toysimulator_push toysimulator_pull eventbuilders_nodiskwriting routingmaster_enabled dataloggers_diskwriting dispatchers", or "toysimulator_push toysimulator_pull eventbuilders_diskwriting routingmaster_disabled".
- That we make functionality swappable- e.g., in the above example, one could imagine the eventbuilders #include a file called "routingmaster_for_eventbuilder.fcl" which would contain routing FHiCL in the case of the routingmaster_enabled subconfiguration and just be an empty file in the routingmaster_disabled subconfiguration
- That after an initial set of subconfigurations, we remain open to changing their structure as we gain experience and come up with new ideas (such as further subdividing subconfigurations, merging subconfigurations, thinking of new #includes, etc.)

Associated revisions

Revision 357e555c - 11/07/2018 04:46 PM - John Freeman

JCF: added subconfigs subdirectory to simple_test_config

The subconfigurations inside the subconfigs directory are based on mediumsystem_with_routing_master. Still to-do:

-Numerous locations where FHiCL can be cleaned up (indentation, repeated code which should be replaced with #includes, etc.)

-Interchangeable parts would be desirable (e.g., routingmaster_enabled.fcl and routingmaster_disabled.fcl, as described in Issue #21328)

History

#1 - 11/15/2018 05:22 PM - John Freeman

- Status changed from New to Work in progress

- % Done changed from 0 to 80

With the develop branch's commit 0e65280f1c1fad140ca4347fc1e55a04652e2918, there's a family of subconfigurations which can be assembled to perform a run. Most of them you have to use, but some are interchangeable. E.g., if you use the configuration

```
subconfigs/component_pull subconfigs/component_push subconfigs/dataloggers subconfigs/dispatchers subconfigs/e
ventbuilders subconfigs/metrics subconfigs/routingmaster
```

...then you'll be able to perform a run with a boardreader in push mode (component01, from component_push) and up to 9 boardreaders in pull mode (component02 through 09), writing metrics to ASCII files and using a routingmaster between the boardreaders and eventbuilders. However, if you swap out subconfigs/routingmaster_disabled for subconfigs/routingmaster, you'll remove the routingmaster (note you'd also want to remove the routingmaster from the boot file), and if you swap out subconfigs/metrics for subconfigs/metrics_disabled, you'll remove the metrics.

I have yet to thoroughly comment the variables.

As far as the structure of the FHiCL documents, one change I haven't made but am considering is to have an aggregator document whose variables are inside a PROLOG, which can be #included by dispatcher and datalogger FHiCL documents where variable overrides then define whether we have a dispatcher or datalogger.

#2 - 11/17/2018 08:16 PM - John Freeman

- Status changed from Work in progress to Resolved

With commit 902dd0e93b10a769665037854bd89d53290b6fe4, I've added comments to the FHiCL variables in the set of subconfigs. In general, I just used the comments for the variables as described in the relevant artdaq header files, occasionally performing light edits; I also left some variables whose names are self-explanatory uncommented. I'm marking this Issue as "resolved", although I anticipate the the subconfigs will be further refined as we gain experience with using them and come up with new ideas.

#3 - 11/21/2018 07:14 AM - Kurt Biery

Thanks John, this is sounding good. I'll try to take a look at the simple_test_config soon. A quick question in the meantime: would the new functionality in DAQInterface "just work" if it were configured to use configurations from the artdaqDB rather than disk files?
Thanks,
Kurt

#4 - 11/21/2018 12:06 PM - John Freeman

Yes, that's correct. The **only** difference between using the disk files and artdaqDB for configurations is that in the former case, DAQInterface will look for a subdirectory called "common_code" at the same level as the subdirectories denoting the subconfigurations and add it if it finds it - i.e., effectively adding a subconfiguration called "common_code" if available - while if you're using the database, this feature doesn't apply.