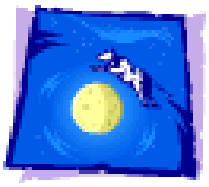
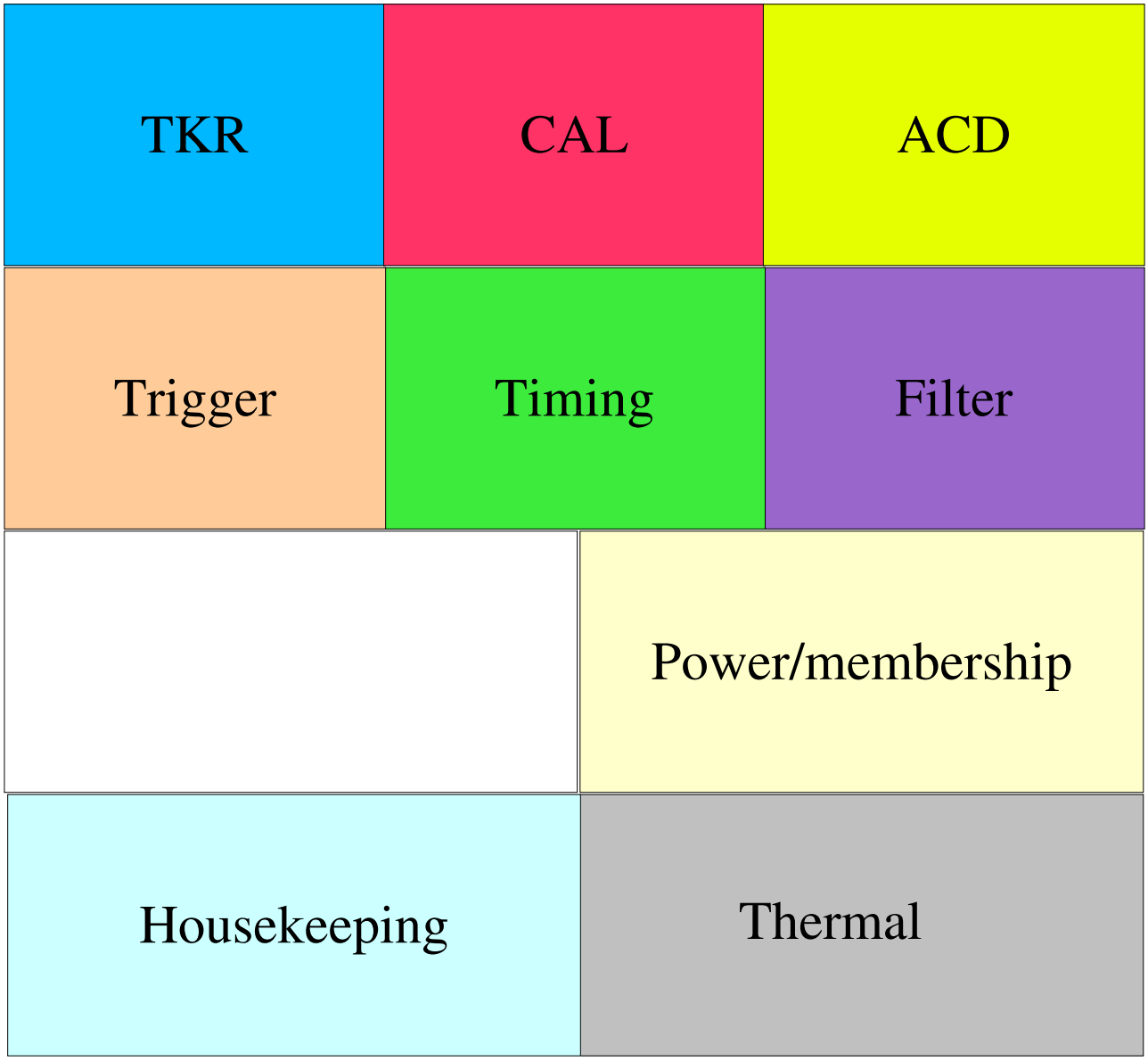
MOOT & Friends



Delegates

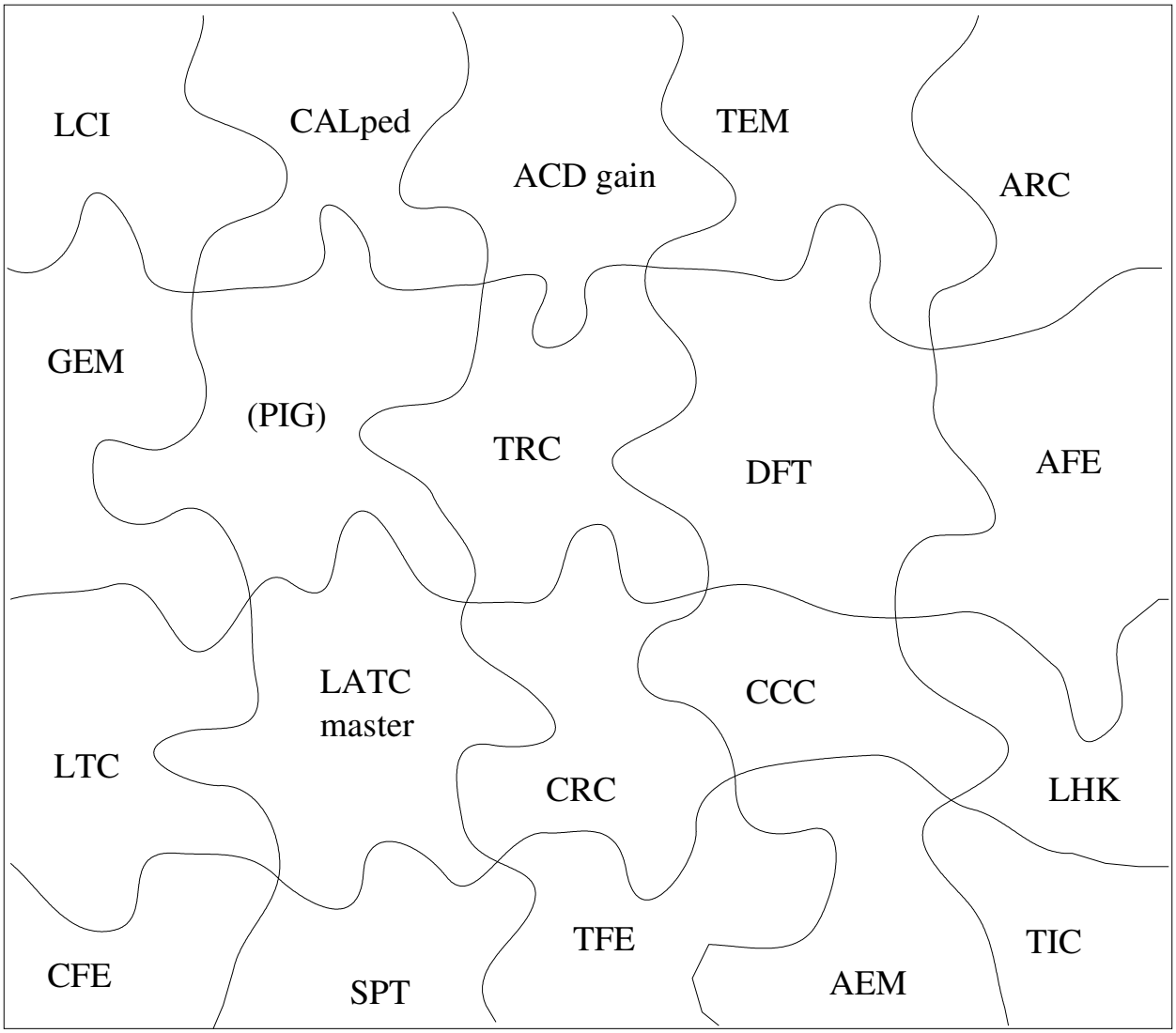


Delegates are meant to partition the configuration along boundaries natural to an operator; ideally information in one box would be managed by one person or group.

The collection of delegates above is not complete, though it's close.

Filter is a representative of FSW software tasks needing configuration.

FSW Files

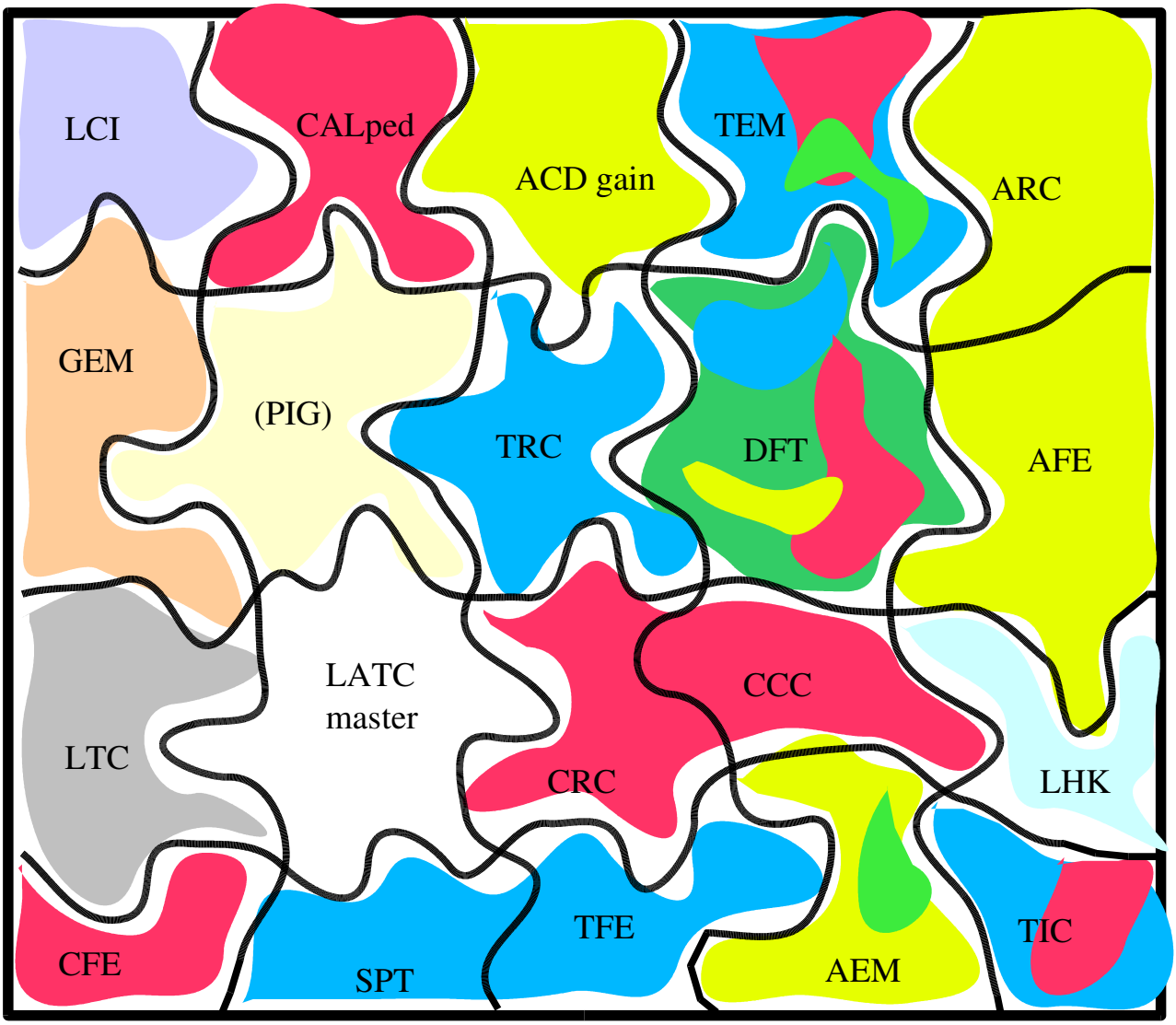


Many binary files are needed to completely determine the configuration (even excluding certain categories, such as binaries for the code). It is a delicate business to ensure that everything that should be specified *is* specified, and only once!

The puzzle is slightly bigger than shown; some pieces have been omitted. However all significant categories of pieces are represented.

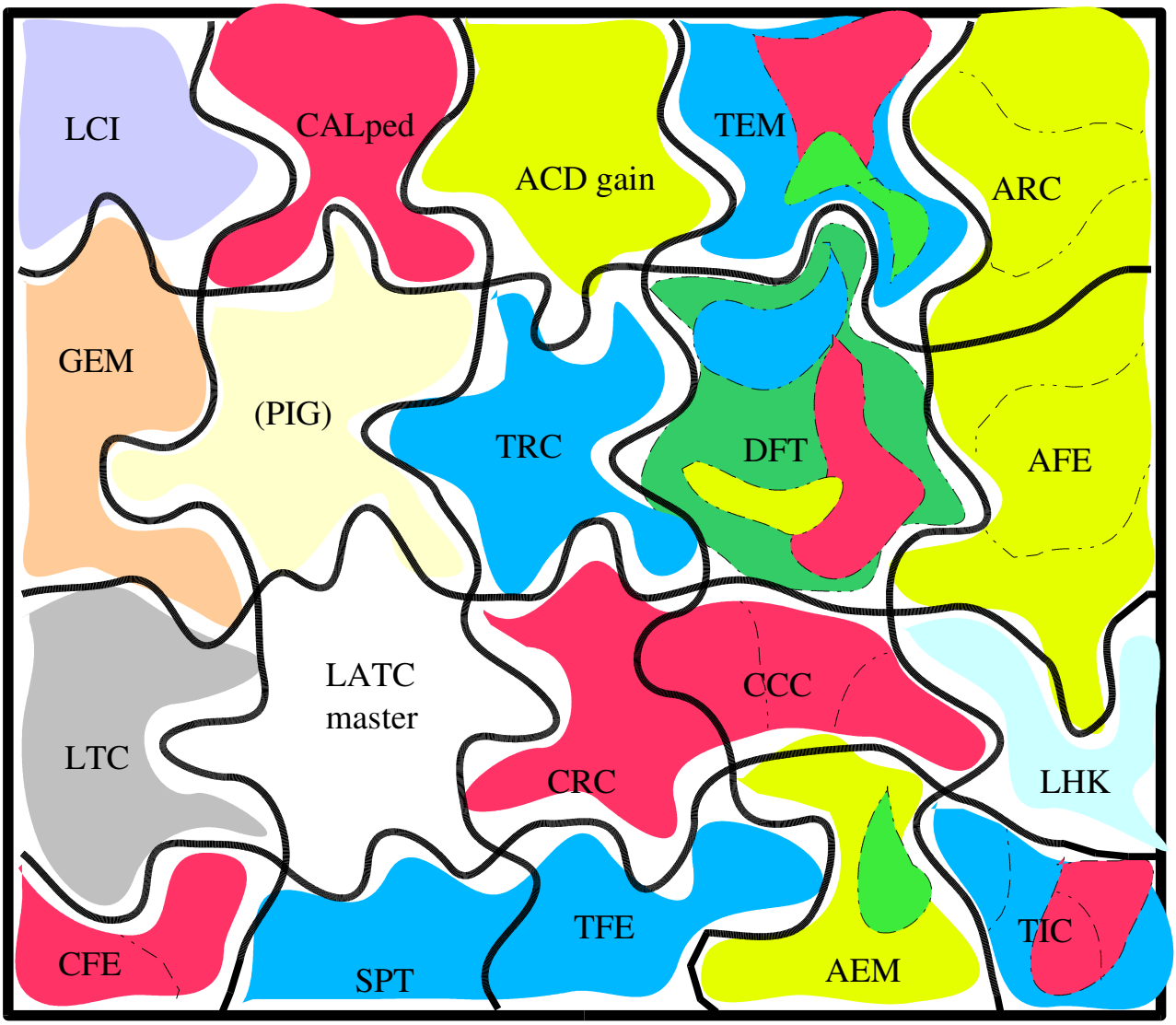
No meaning should be attached to the relative locations of pieces above.

FSW Files/**Delegates** Overlay



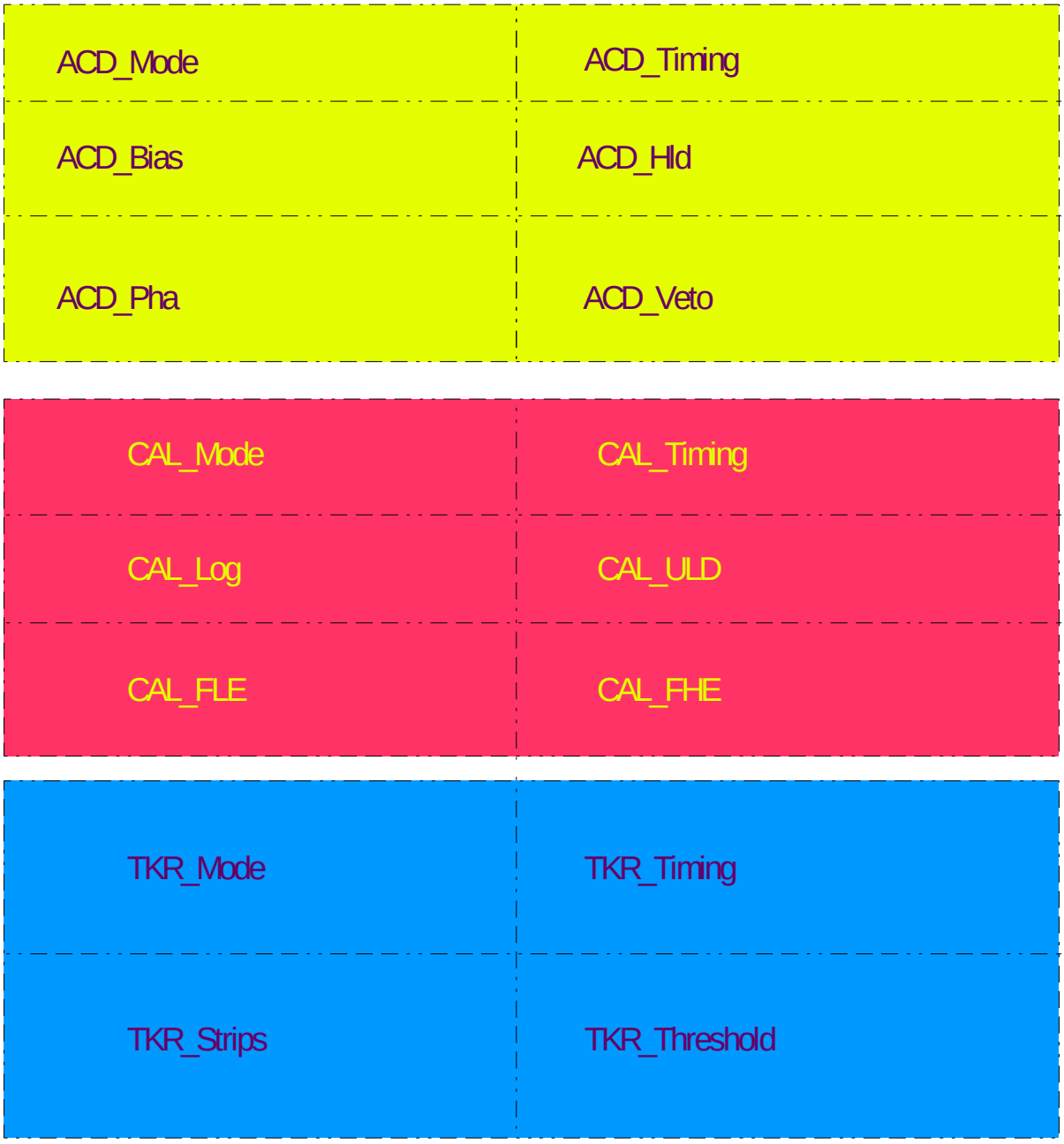
An attempt to indicate which delegates have an interest in the FSW entities. Some carry over nicely; e.g. **Trigger** to GEM, **Thermal** to LTC. But the subsystem delegates and **Timing** pop up in several. To add to the confusion, some FSW files have input from more than one delegate.

FSW Files/Delegates/Parameters Overlay



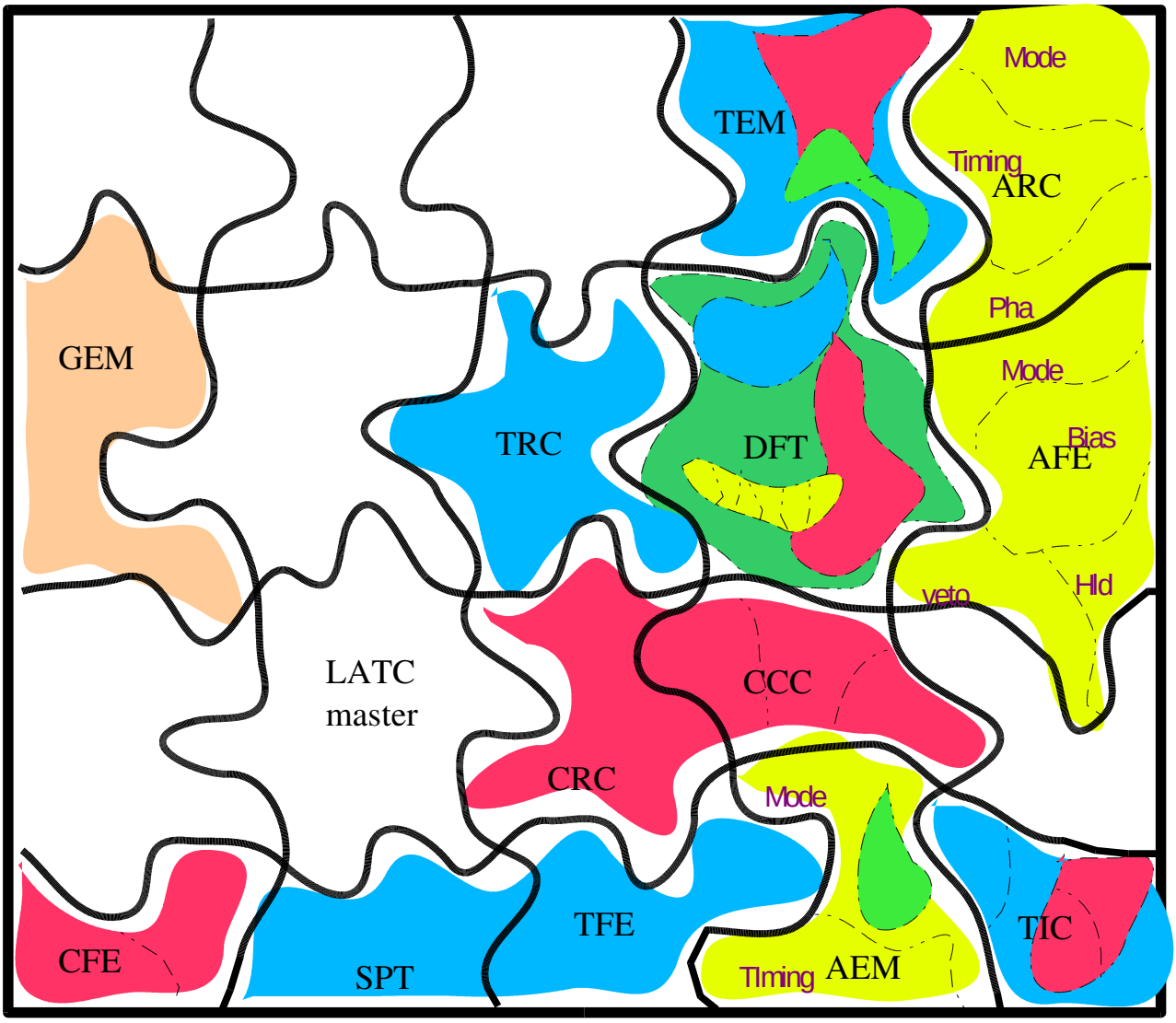
Extra dotted lines indicate how information from different parameters may be merged into a single FSW file. Note that different colors (indicating different delegates) are always separated by lines.

Delegates and Precincts



A delegate may be partitioned into precincts. A precinct controls a subset of the delegate's registers which are derived by a common procedure, from a common set of inputs. For example, ACD has 6 precincts: ACD_mode, ACD_timing, ACD_bias, ACD_PHA, ACD_Veto and ACD_Hld.

LATC Files/Delegates/Parameters/Precincts Overlay



A precinct like ACD_Mode may "own" registers in several LATC components
A procedure associated with the precinct takes certain inputs and produces
LATC source files (at least one per represented LATC component).

In order to get this going, for a particular precinct must specify

* procedure method
* input files (e.g., subsystem calibrations) if appropriate
* fixed register settings, if appropriate
* parameters to processing, if appropriate

Intent for a Precinct (Vote)

```
<!-- Sample ACD_Hld intent -->

<ACD_Hld>
  <method>CALIB</method>

  <ACD_Hld-registers>
      <AFE_hld_dac>
          <broadcast>4</broadcast>
          <except garc="0" gafe="1">3</except>
        </AFE_hld_dac>
  </ACD_Hld-registers>
  <ACD_Hld-constants/>

  <ACD_Hld-anc>
   <ACD_HldCalibration

nickname="hld">nominal</ACD_HldCalibration>
  </ACD_Hld-anc>

  <ACD_Hld-param>
   <!-- no stub specified; use default -->
   <latc_AFE_ACD_Hld/>
   <latc_DFT_ACD_Hld/>
  </ACD_Hld-param>
  <ACD_Hld-offline/>
</ACD_Hld>
```

Use

Create inputs from intent:

- Write vote (intent) files via operator gui; register vote file with MOOT.
- When a new vote file is written or new calibrations are available, may “run” the vote file, producing new LATC source.
- Register new source files with MOOT.

Information saved in the db includes links to vote file and to ancillary files used to generate the source file.

Build a configuration:

- From intent files and queries to db, determine collection of parameter files and whether they are “up-to-date” w.r.t. latest calibrations.
- Invoke MOOT configuration build function. Creates new binaries if necessary, makes db updates and returns a config key.

Use (cont'd)

Determine intent of a config:

- Use MOOT queries to recover parameter (e.g., LATC source) files
- From there use other MOOT queries to backtrack to ancillary files and vote files

There's more...

Ancillary file aliases:

In the vote file example the word “nominal” appears as data for the <ACD_HldCalibration> element. All ancillary files appearing in vote files are identified by an alias name like this. MOOT will keep track of the actual file associated with an alias name at any given time.

Vote file aliases:

When building a configuration one typically wants the union of several vote files (one per precinct); all the “little intents” embodied in those files are to be bound together in a single grand intent. It is convenient to implement the grand intent as a file which refers to all the subsidiary vote files by alias name. MOOT will track these associations also.

Calibrations and Intent

Things are still more complex when LCI is involved. The procedure of interest for subsystem expert or operator may involve a sequence of LATC configurations as well as one or more LCI source files. Eric has a draft XML description for it; I haven't yet worked out all the implications for MOOT, which likes to consider only one configuration description at a time.

To Do*

MOOT:

To support this realization of intent MOOT has to do more tracking of more objects and some new kinds of verification. This translates into more tables, more fields in some old tables, and more services to fill and query the tables. See also

http://www.slac.stanford.edu/exp/glast/ground/notes/moot/intent.shtml

Intent files:

Eric has drafts for all precincts involved in physics running. So far only the ones for the ACD precincts have been formalized in an XML schema; the others should follow similar patterns. There may be more to do for the files describing calibrations.

LATC (in)completeness:

The LATC input generated by Eric's software from the intent files is not quite what is needed to get the desired effect. Sometimes more registers need to be explicitly set. David D. is working on a utility to fill the gap.

* jrb-centric