15th May 2002

Concepts in the Software Overview Document

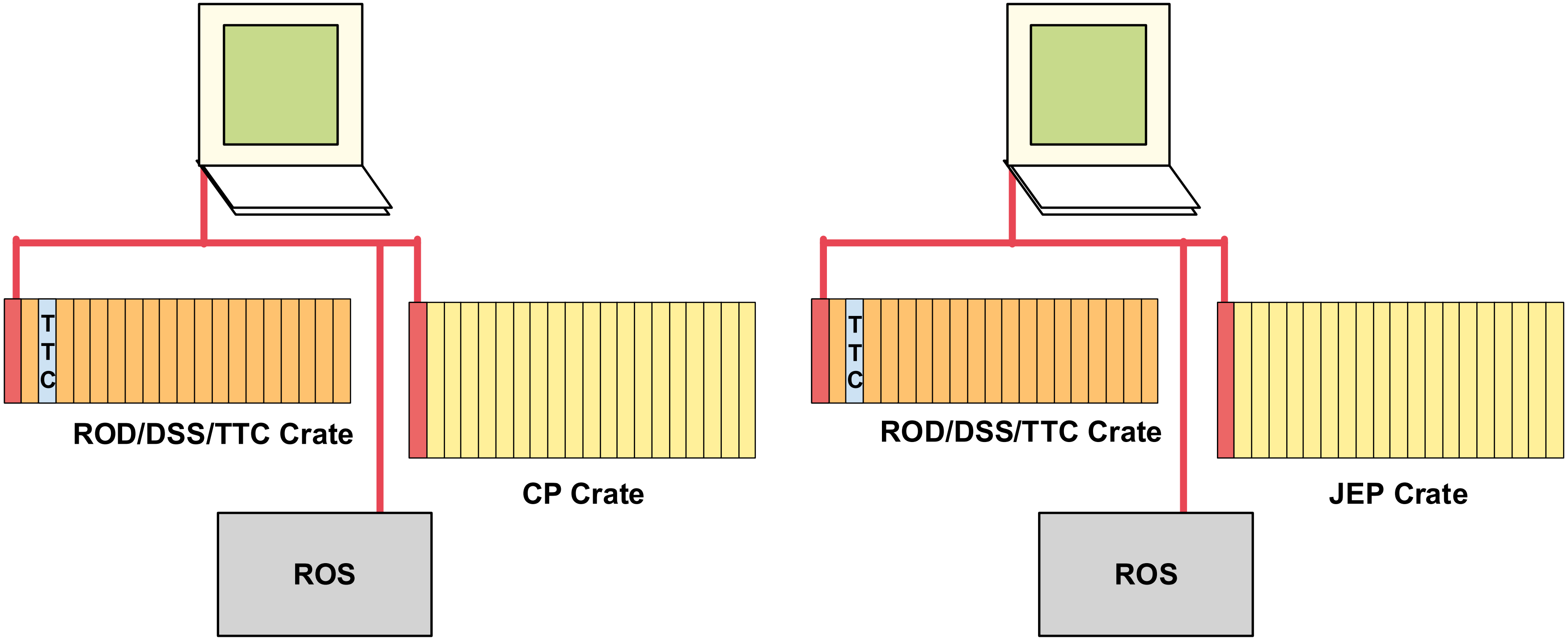
C .N .P .Gee
Rutherford Appleton Laboratory

Terminology

- **ROD-crate DAQ will be used with and in all our trigger processor and readout crates during testing and final ATLAS.**
- **Some terminology:**
  - RCC (ROD Crate Controller): the single-board computer in each crate
  - RCW (ROD Crate Workstation): A workstation connected to RCCs by Ethernet and providing an operator interface.

- **Calo trig subsystems will be controlled during testing from RCWs. There may be several separate RCWs each controlling separate subsystems doing independent testing.**

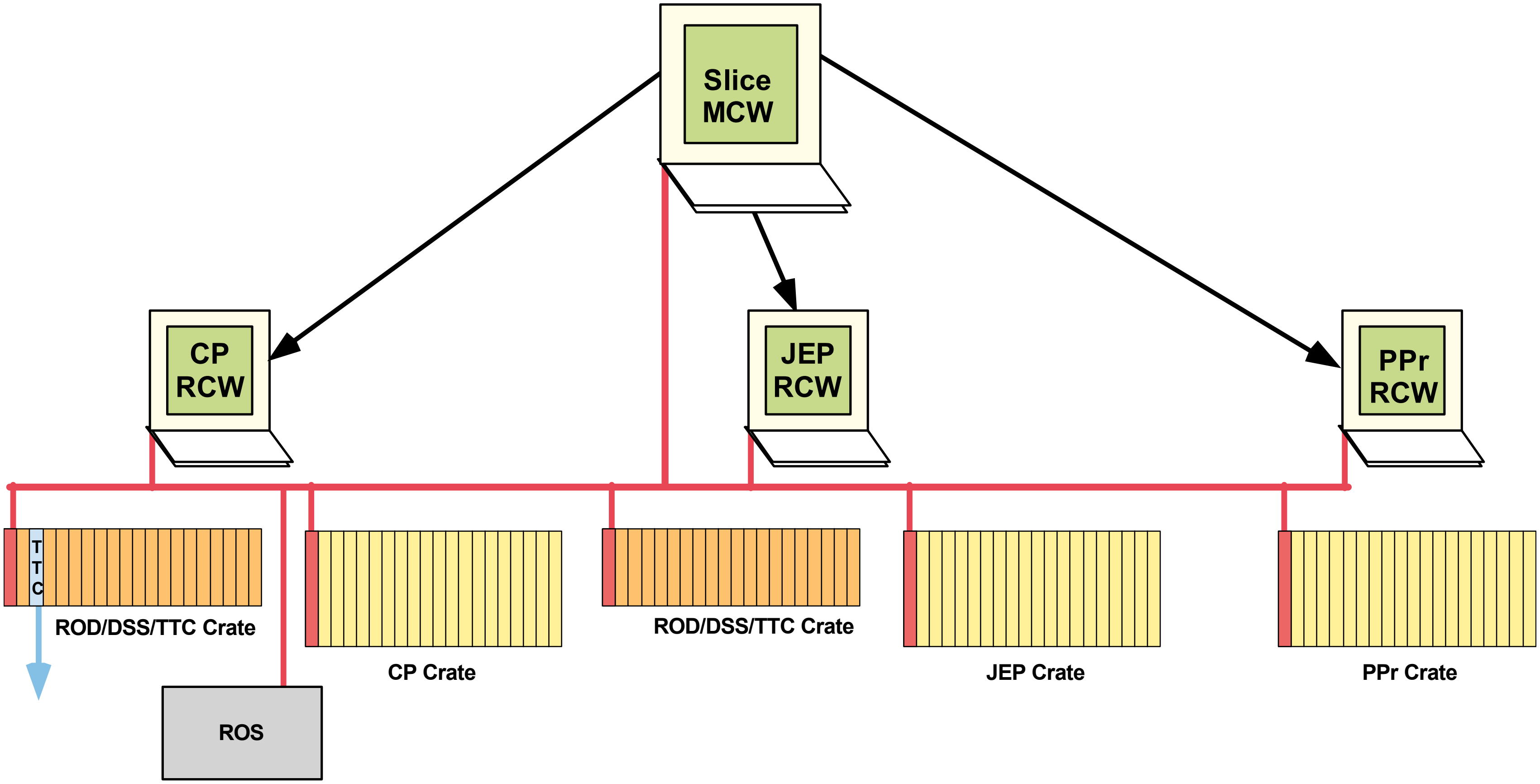
Parallel Tests



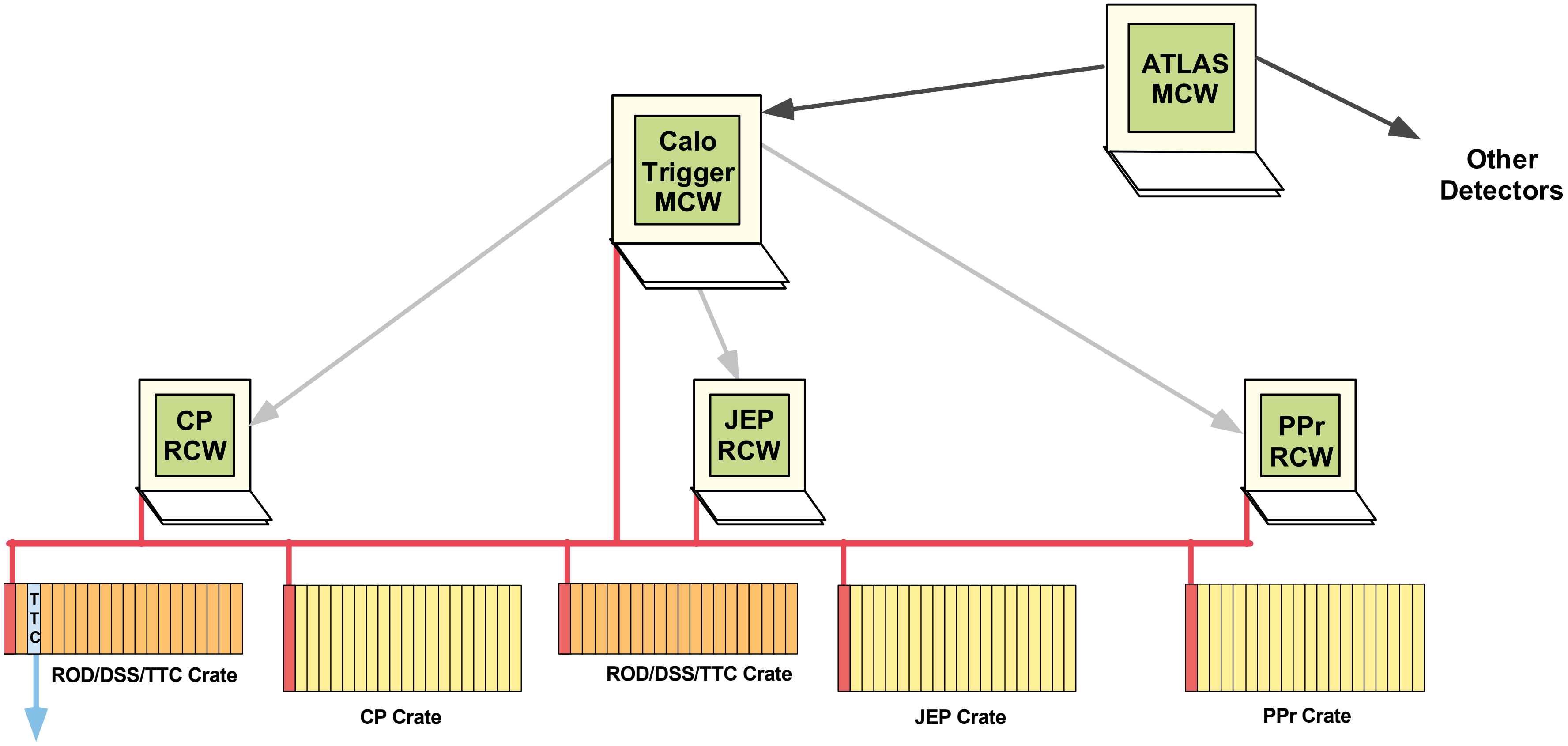
Terminology (2)

- **In the slice tests, commands will be issued from a Master Control Workstation (MCW) –**
  - a link is established between the MCW and RCW run controllers, and each RCW run control functions as an MCW slave.
  - commands are sent hierarchically from MCW to RCWs and on to slave run controllers in the RCCs.
- **In final ATLAS, an Overall Master Control Workstation will send commands to the detector MCWs.**

Slice Test


Slice MCW
CP RCW
JEP RCW
PPr RCW
TTC
ROD/DSS/TTC Crate
CP Crate
ROD/DSS/TTC Crate
JEP Crate
PPr Crate
ROS

Full System in ATLAS



Crate and Workstation Startup (for any of these configurations)

- **Start with all components are powered down, but all modules are in place in the required crates, computers are connected by Ethernet, and all inputs, high-speed links and readout links, TTC connections and all other necessary cabling are in place.**
- **Then turn on crates and computers in any order.**
  - Trigger processing modules enter their power-up reset state.
  - All computers boot into Linux
  - Diskless single-board computers boot over Ethernet from a workstation acting as boot server.
  - Start-up scripts initialise online system infrastructure

Operator Partition Selection

- **The operator logs into a control workstation**
  - He runs a script to initialise a partition and start the run control.
    - ***The partition selects the modules to be used.***
- **The main run control starts and displays its operator panel**
    - ***There are discussions about dynamically adding/removing modules without stopping and restarting the complete run control. You can't do this at the moment.***

Starting a Run

- **The operator selects the required run type from a run control menu, and sets any additional parameters.**
- **He then requests system state transitions from "Initial" to "Loaded" to "Configured" to "Active".**
- **State transition commands are sent hierarchically to to slave run controllers on the RCCs in each crate. These interact with modules through the module services.**
  - For each module, the RCC creates an appropriate module services object.
  - Using the run type and parameters the RCC creates a database **view**, and passes this to all module service objects.
  - The module service retrieves register and table values from the database view and sets the module hardware as needed.
- **The actual settings used are recorded in the "conditions database".**

Enabling Events

- When the system completes the transition to "Configured", all modules are executing the real-time trigger processing algorithms.
- L1A generation is inhibited by Busy signals.
  - Each partition has a Detector Central Trigger Processor Interface (DCTPI) module which manages the busy signals.
- On the transition to active, all error counters are cleared, ROD Busy signals are removed, and finally the DCTPI Busy is cleared.
- L1As can now be generated.

While the System is Running…

- **Module services poll error counters in modules and issue alarms if error rates exceed thresholds**
- **DCS values (temperatures, voltages) are checked by our CANbus controllers. Alarms are sent to online software if out-of-range values are found.**
- **Monitoring programs obtain events, perform analysis, create and fill histograms & tables.**
  – The histograms & tables are published, and can be selected, displayed, and cleared from workstations in the experiment.
- **A status repository is updated and displayed. The operator can display status information on one or many screens, and can change the number and content of status screens during any run.**

Ending Runs

- **Using the run control panels, the operator requests a run end.**
- **The command is passed to the modules services for the DCTPI, where Busy is asserted.**
  - L1As stop when Busy reaches the trigger source.
- **The last event flows through system (how do we know?)**
- **Monitoring programs complete their analysis and save histograms and statistics (where?)**
- **Run statistics are saved in the book-keeping database.**
- **The system completes the state transition back to “Configured”**

Using Test Data

- The user decides on a grouping of modules for a test.
- He creates an English-like Test Descriptor
  - contains a test name, specification of the data patterns (e.g. “ramp”)
  - plus all module settings (e.g. thresholds) needed for the test.
  - This is saved to the database.
- The system is then set up for a run as described above.
- Modules services interrogate the database view for simulation data.
- Now (or earlier) the simulation package is run.
  - It reads the Test Descriptor and computes the binary input data
  - and the expected output data produced by the algorithms.
- The system is run and outputs compared to expected output.
- The pattern of L1As is specified with the Test Descriptor and generated by the DSS.

End

The End