SCT/Atlas ROD Test Stand

Wisconsin SCT/Pixel ROD Group:

K. Dao
D. Fasching
R. Jared
J. Joseph
M. Nagel
L. Stromburg
L. Tomasek

December 17, 1999

SCT/Atlas ROD Test Stand

The **SCT/Atlas ROD (Readout Driver) Test Stand** is intended to provide the functionality to test and debug the standalone ROD.
The test will be done with a VME crate, PC with National Instruments interface (VME-MXI-2 hardware + LabWindows software).

Later phases of the testing will require the **Back of Crate (BOC)** card and ***Timing Interface Module (TIM)***.

Reference: ROD diagrammatic model(debug path)and ROD Controller – see Mark’s talk.

Testing procedures of the standalone ROD

Main tests (internal connectivity):

- write to input then read from output FIFO of the ROD Controller
- write to an object then read the same object
  (ie. ROD input memories, debug. memories, event fragment builder FIFO, DSP ext. memories)
- write to an object then read from the downstream object
  (function test of the decoder, gatherer, data director …)

Tests of the external signals:

- **control links**

  - loopback cable to ROD input memories

- **data links**

  - input links to ROD ->loopback cable
    (ie. test of the backplane interface connector, one memory plays while the other records)

  - output links from ROD (input to S-link, BOC) ->loopback cable

Later tests of the other components connected with ROD:

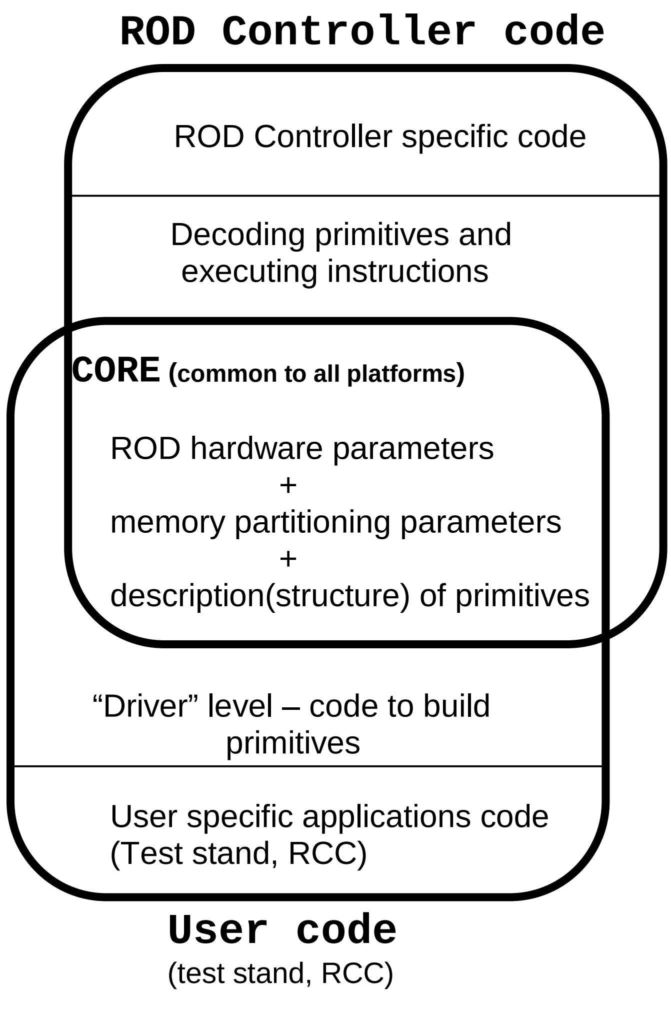
- **BOC**(Back of Crate)

    – check that we can read/write to config. registers

- **TIM** (Timing Interface Module)

    - communicates to ROD over special backplane
    - a sophisticated stage of the test stand will be needed to test TIM-ROD. (test of the fast commands L1A, BCR, ECR, CALStrobe etc.)

ROD Software Architecture



The software will consist of ROD Controller software primitives, a shell of primitives that communicate over VME to the ROD and higher level software.

The shell is intended to provide a communication protocol that will hide the details of the interface to the ROD from the VME crate. It means that ROD actions will be driven by lists of commands and ROD will look like a list processing engine. The higher level software resides on the PC (test stand) or crate processor (ROD Crate Controller – **RCC**) .

The desire is to make the software developed in the PC or crate processor largely interchangeable.

A. Development of software for use in all environments

- **Core code**

    - common to ROD Controller and any user platform (crate processor, PC..)

        * Rod hardware description
            - internal memory register addresses
            - memory spaces/partitioning

        * Primitives description
            - names and parameter spaces

    - common to crate processor/PC but not to ROD Controller (in the test stand software should be included all basic RCC functions necessary for testing procedures)

- **Code to build primitives** ("driver" level code)

B. Development of an application on the top of the Core code - User specific code

- **Test of ROD hardware**

    - commissioning in short term
    - fixing boards in longer term

- **R/W memories allow for testing isolated locations on the board**

    - software for the testing and initial running of the ROD.

Software team and responsibilities

John Hill (Cambridge University)

- is responsible for the RCC(ROD Crate Controller) software

- design of the software layers, finding and descriptions of the appropriate set of primitives and coding rules (together with Lukas Tomasek and Damon Fasching)

Lukas Tomasek

- first version of the “Core” and all the software for Test Stand (PC) plus “primitive decoding” software for ROD Controller

Damon Fasching

- software for ROD Controller (and back end DSPs)

Iowa - ?

Time plan

The basic software for "standalone" ROD testing should be ready till March 2000.

`