

# ROD Test Stand Overview

<http://www-hep.fzu.cz/~tomasekl/RODTestStandOverview.ppt>

Lukáš Tomášek <tomasekl@fzu.cz>  
Jan Valenta <valenta@fzu.cz>

LBL  
The University of Wisconsin, Madison  
Institute of Physics AS CR, Prague

# Test Stand basics

## **Purpose**

- initial testing and debugging a standalone ROD,
- production testing and ROD maintenance.

## **Software**

- programming language ANSI C
- MS Windows 9X, NT, 2000, XP operating system
- NI LabWindows/CVI 5.5 development software

## **Hardware**

- PC with NI PCI-MXI-2 interface card,
- VME crate with NI VME-MXI-2 card

# Main TestStand tasks and features

- **ROD initialization and configuration:**
  - loading FPGA code to flash memories,
  - R/W FPGA command/status registers (program/reset manager),
  - loading program for DSPs (MDSP flash, slaves directly).
- **ROD control:**
  - by *PrimitiveLists*,
  - direct commands via *VmeCommandRegister* bits (DmaRequest, Pause ...),
  - more comprehensive tasks by *CommandLists*.
- **ROD monitoring:**
  - periodic checking of the ROD status registers (ControllerFPGA, DSP status),
  - readout text messages from the *TextBuffers*.
- **ROD utilities and supporting applications:**
  - file format converters (binary-hex, coff, ...),
  - binary data creator, binary file viewer (freeware Witched), etc.

TestStand supports one VME crate - up to 16 Rods can run simultaneously.
--

## Memory Panels



## Flash Panel and FPGA Program/Reset Manager

**FLASH**

MainPanel! SaveCurrentSettings! LoadSettings! ViewBinFile! Hide!

Slot# **18**

**LOCATION FILE**

c:\RodTS\Data\Fpga\locationFile\_v0.bin

Size[B]	Sects	Start addr	End addr
x000018	1	x E00000	x E00017

Read Write

**RRIF FILE**

c:\RodTS\Data\Fpga\revC\rrif\_v24c.bin

Size[B]	Sects	Start addr	End addr
x078E64	121	x E01000	x E79E63

Read Write

**FORMATTER FILE**

c:\RodTS\Data\Fpga\revC\fsct\_xp6.bin

Size[B]	Sects	Start addr	End addr
x052330	83	x E80000	x ED232F

Read Write

**EFB FILE**

c:\RodTS\Data\Fpga\revC\efb\_s13c.bin

Size[B]	Sects	Start addr	End addr
x052328	83	x ED3000	x F25327

Read Write

**ROUTER FILE**

c:\RodTS\Data\Fpga\revC\rrtr\_s16c.bin

Size[B]	Sects	Start addr	End addr
x0393D8	58	x F26000	x F5F3D7

Read Write

Start addr

Erase FLASH0 x E00000 Read FLASH0

Erase FLASH1 x E80000 Read FLASH1

Erase FLASH2 x F00000 Read FLASH2

**DSP PROGRAM**

c:\RodTS\Data\DSPPProgram\rodRun\rodRun\_FLASH.bin

Size[B]	Sects	Start addr	End addr
x04DD20	78	x 01400000	x 0144DD11

Write

Erase FLASH DSP

**FPGA STATUS/COMMAND REGS**

MainPanel! Hide!

ROD ID **AD** Code Version **5** Slot# **18** ROD S/N **5** ROD Revision **C** InMem Size **32k**

**STATUS**

FpgaCnfg	FpgaReset	VmeDspReset	Fpgalnit	Flash	FpgaHalt
S0 x 1F	S1 x 1F	S2 x 3E	S3 x 1F	S4 x 00	S5 x 00
Cnfg Ovrld 7					
Cnfg En 6					
Cnfg En 5		SDSP3 5			
Router Done 4	Router 4	SDSP2 4	Router Init 4		Router Halt 4
EFB Done 3	EFB 3	SDSP1 3	EFB Init 3		EFB Halt 3
FormB Done 2	Form B 2	SDSP0 2	FormB Init 2	WR Flash I 2	FormB Halt 2
FormA Done 1	Form A 1	MDSP 1	FormA Init 1	WR Flash 1	FormA Halt 1
RRIF Done 0	RRIF 0		RRIF Init 0	RD Flash 0	RRIF Halt 0

**CONTROL**

FpgaCnfg	FpgaReset	VmeDspReset	Flash	FlashAddr+WtData
C0 x 00	C1 x 00	C2 x 00	C3 x 00	C4 x 00E79E63
Cnfg Ovrld 7 OK	Hold FPGA 7 OK	Reset board 7 OK		FlashRdData S7 x 00
CnfgAll 6 OK	Reset all FPGA 6 OK	Reset SDSP3 6 OK		
CnfgRouter 5 OK	Reset Router 5 OK	Reset SDSP2 5 OK		
CnfgEFB 4 OK	Reset EFB 4 OK	Reset SDSP1 4 OK		
CnfgFormA&B 3 OK	Reset FormB 3 OK	Reset SDSP0 3 OK	WR Flash I 3	
CnfgFormA&B 2 OK	Reset FormA 2 OK	Reset MDSP 2 OK	WR Flash 2	
CnfgRRIF 1 OK	Reset RRIF 1 OK		RD Flash 1	
CnfgRRIF 0 OK			RD Flash 0	

**READ ALL REGISTERS**



## ROD Control/Status[n] and Main TestStand Control/Status Panels

The image displays two windows from the ROD TestStand software. The main window is titled "ROD slot#18" and contains various control and status panels. The "COMMAND LIST STATUS" panel shows a list of commands with indices and repetition counts. The "Router/DSP Trap Status" panel displays trap settings for different events. The "FIFO Status" panel shows the status of various FIFO buffers. The "TestGlobalReg" panel displays global register values. The "ERR MASK" panel shows error mask settings. The "RW REG FIELD" panel displays register field values. The "POLL REG FIELD" panel displays poll register field values. The "MEMORY TEST" panel shows memory test results. The "HISTOGRAM SETUP" panel displays histogram setup parameters. The "RCC MAIN WINDOW" is a separate window titled "RCC MAIN WINDOW" with a menu bar (LoadRod!, Params, ErrorFile!, ListEditor, MemoryAccess, ExitProgram!) and a status bar (v4/25/03). It contains a "CRATE STATUS/CONTROL" panel with a "ShowErrorWindow" checkbox, a "Config" dropdown (SCT), and a "CommLoop" status (RUNNING). It also has a "Status Refresh" section with "TestStand", "FPGA Monitor", and "DSP Monitor" status indicators. The "Thread Priority" section shows "UIR" (ABOVE\_NORMAL), "CommLoop" (BELOW\_NORMAL), and "CmdListExecution" (NORMAL). The "Timeout[sec]" section shows various timeout values. The "CmdListBusy RODslot#" section shows a list of busy slots. The "COMMAND LIST" section shows a list of commands with repetition counts. The "PRIMITIVE LIST" section shows a list of primitive commands with repetition counts.

**ROD slot#18**

Main! StatusControl! ListEditor TextBuffers CommandListStatus! DSPMonitor! Data! Reset ExitRod!

Slot# 18 **ENABLED** ROD S/N 5 ROD Revision C SCT 32k

SlvDSP 0 1 2 3 ON

**Status0** 00000011 **Command0** 00000000

DspRunning 0 3 Abort OFF

Busy 1 2 Resume OFF

Executing 2 2 Pause OFF

Paused 3 1 InListRdy

OutListRdy 5 0 InListRdy

DspAck 5 0 InListRdy

DmaAck A A DmaReq OFF

DmaErr B

ErrorBufNE 8 4 RdRq DIS

InfoBufNE 7 5 RdRq EN

DiagBufNE 8 6 RdRq DIS

XferBufNE 9 7 RdRq DIS

**Status1** 0000003C **List Index** 0000000C

READ STATUS

Repetitions Counter 0 0

Stop EXEC BUILD

dataPathFiles\readEventData\_allSlv.pl

EDIT SEND PRIM LIST

**Router/DSP Trap Status**

En	Trap	On EF	FF	LD	Format
0	NONE				S-link/Event
1	NONE				S-link/Event
2	NONE				S-link/Event
3	NONE				S-link/Event

**FIFO Status**

Debug FIFO I/O	EventMem
InMem En	A EF FF
DebugMem En	B EF FF
Int TIM En	FMB/EFBDM En
Header/Trailer Limit	CorrTrig En
Formatter FIFO Full	

**TestGlobalReg**

ListIndex	Time
00000000	
TestResult	00000000

**ERR MASK**

ListIndex	Time
00000000	
ErrmsgMask	00000000

**RW REG FIELD**

ListIndex	Time	ObjectID	Offset
00000000		00000000	00000000
Width	DataOut	00000000	00000000

**POLL REG FIELD**

ListIndex	Time	ObjectID	Offset
00000000		00000000	00000000
Width	DesiredValue	00000000	00000000

**MEMORY TEST**

ListIndex	Time	StartAddr	MemorySize	ReturnCode
00000000		00000000	00000000	0

**HISTOGRAM SETUP**

Slv	ctrlBase	pulseBase
0	00000000	00000000
1	00000000	00000000
2	00000000	00000000
3	00000000	00000000

**COMMAND LIST STATUS**

ClearStatusWindow! OpenStatusFile! CmdListEditor! Hide!

Busy Stop Reset Resume

nCmds Index Command

27 26 Send PrimList

Stop Rep 0 Cnt 0

ListRep Counter If Error

0 0 DEFAULT

EDIT SEND COMMAND LIST

\*\*\*\*\* Date:05-01-2003, Time:12:52:11 \*\*\*\*\*

C:\Rods\Rcc\DefaultLists\rodInit.cl

CLIST START (NumCmds:2; ListRep:0)

CLIST END (CmdIndex:2; ListRepCount:1)

\*\*\*\*\* Date:05-01-2003, Time:12:52:47 \*\*\*\*\*

c:\RodTS\Data\Testing\loadSlaveCode\load&startAllSlv.cl

CLIST START (NumCmds:27; ListRep:0)

CLIST END (CmdIndex:27; ListRepCount:1)

\*\*\*\*\* Date:05-01-2003, Time:12:52:51 \*\*\*\*\*

**INFO BUFFER**

ClearWindow! OpenFile! HideWindow!

BufferMode LINEAR Overwrite Overflow

MASTER DSP: resetrod.c 89

**RCC MAIN WINDOW**

LoadRod! Params ErrorFile! ListEditor MemoryAccess ExitProgram!

Date:05-01-2003, Time:12:52:51

CRATE STATUS/CONTROL v4/25/03

ShowErrorWindow Config SCT

RESMAN Rod Count 2

RESET RODs ActiveSlot# 19

WME byte order DefaultSlot# 20

INTEL

**Status Refresh**

TestStand FPGA Monitor DSP Monitor

Off On Runtime Period 1.0 s

**Thread Priority**

UIR CommLoop CmdListExecution

ABOVE\_NORMAL BELOW\_NORMAL NORMAL RST ALL

**Timeout[sec]**

rodInit textBuff dspAckClear dspAckSet

005.000 005.000 005.000 010.000

On Off

**CmdListBusy RODslot#**

5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21

Slot# ALL ExtList CMD

**COMMAND LIST**

Repetitions 0

If Error DEFAULT

c:\RodTS\Data\Testing\loadSlaveCode\load&startAllSlv.cl

ADD REMOVE EDIT SEND CMD LIST

**PRIMITIVE LIST**

Repetitions 0 EXEC BUILD

SaveLists OFF

ChecksumMaster ON

ChecksumSlave ON

c:\RodTS\Data\Testing\dataPathFiles\data.pl

ADD REMOVE EDIT SEND PRIM LIST

5/12/03

ROD TestStand Overview

# Primitive List

- List of *primitives* executed by master/slave DSP; prepared and sent by Host.
- The primitives, input and output data formats, communication registers and communication scheme are described in “*Communication Protocol*”:  
<http://sctpixel.home.cern.ch/sctpixel/MessageProtocol/MessageProtocolV25.pdf>
- The TestStand user can create different PrimitiveLists in the **PrimitiveList Editor** and save them in files for the next use or directly execute. The PrimitiveListEditor uses more complex version of the PrimitiveList (*PrimitiveListTable*), since it has to contain information not only about the input data but also variable parameters for the reply data (save to file, print on screen etc.).

## Primitive List Editor:

**PRIM LIST EDITOR**

Main | CommandListEditor | DataFileCreator | RevisionUpdate | HideWindow

**PRIMITIVE**

MASTER ▾

RW\_FIFO

ECHO  
SET\_ERRMSG\_MASK  
PAUSE\_LIST  
EVENT\_TRAP\_SETUP  
SET\_MEMORY  
COPY\_MEMORY  
MEMORY\_TEST  
SET\_LED  
FLASH\_LED  
SEND\_DATA  
MODULE\_MASK  
SET\_TRIGGER  
START\_TASK  
TASK\_OPERATION  
TEST  
WRITE\_BUFFER  
RW\_SLAVE\_MEMORY  
TRANS\_SERIAL\_DATA  
START\_SLAVE\_EXECUTING  
CONFIG\_SLAVE  
RW\_REG\_FIELD  
POLL\_REG\_FIELD  
✓RW\_FIFO  
  >>SEND\_SLAVE\_LIST  
  START\_SLAVE\_LIST  
  SLAVE\_LIST\_OP  
  BUILD\_STREAM  
  SEND\_STREAM  
  RW\_MODULE\_DATA  
  SEND\_CONFIG  
  DSP\_RESET  
  SET\_ROD\_MODE  
  RW\_MODULE\_VARIABLE  
  RW\_BOC\_DATA

**Slot#**

ALL ▾

Repetitions  
0

SEND TO ROD

Ctrl+  
ADD BEFORE>

REPLACE>>=

ADD AFTER>>=

ViewPrim ↑  
↓

Ctrl+  
<<EDIT PRIM

Ctrl+Del  
DELETE PRIM

LOAD LIST

INSERT LIST

SAVE LIST

DELETE LIST

Clear

**MASTER LIST**

ListLength PrimCount AllPrimCount

PrimList x 1821 3 3

RepList x 1815 3

LastFile

c:\Rods\Data\Testing\fifoTesting\writeReadInpFifoA.pl

ID	PRIMITIVE	PrimLn	RepLn
1	POLL_REG_FIELD	9	5
2	RW_FIFO	1809	5
3	RW_FIFO	9	1805

Write and read back random data to Input FIFO A.

5/12/03



## Primitive List Internal Structure (PrimListTable):

primListTableDefinitions.h

```
/*----- LIST TABLE DEFINITIONS -----*/
#define MAXPRIMS_IN_LIST 100
#define COMMENT_LENGTH 600

struct PRIM_TABLE{
    char comment[COMMENT_LENGTH]; /* primitive comment */
    struct MSG_HEAD inPrimHeader; /* primitive header */
    UINT32 replyLength; /* predicted length of reply data */
    union PRIM_PARAMS_UNION params; /* primitive parameters */
};

struct LIST_TABLE{
    char comment[COMMENT_LENGTH]; /* list comment */
    UINT32 allPrimCount; /* number of all primitives(master+slave) in list */
    UINT32 masterToTablePrimIndex[MAXPRIMS_IN_LIST];
        /* the array contains the real index in PrimTable for each master primitive;
           array index=master index */
    struct MSG_LIST_HEAD inListHeader; /* primitive list header */
    UINT32 replyLength; /* predicted length of reply list */
    UINT32 replyCount; /* number of prims with reply data */
    struct PRIM_TABLE_UNION primTable[MAXPRIMS_IN_LIST];
};

-----

primFunc_RWFifo.h
/* RW_FIFO primitive params example - included in PRIM_PARAMS_UNION */

struct PRIM_RWFIFO_PARAMS{ /* params for host */
    struct RW_FIFO_IN inputParams;
    char dataFileName[PRIM_PATHNAME_LENGTH];
};
```

**Note:** The max. number of *primitives* in the *PrimListTable* is limited for simplicity and to avoid possible problems with dynamic memory allocation (current limit is 100). However this number can be easily changed if necessary. The same applies to the comment length field “artificially” limited to 600 characters.

## Primitive List example:

**PRIM LIST EDITOR**  
Main! CommandListEditor! DataFileCreator! RevisionUpdate! HideWindow!

**PRIMITIVE**  
MASTER

**POLL\_REG\_FIELD**

**INPUT DATA**

Base ID: 0, RegID: 227

IDE\_MEM\_STAT

```

**** RRIF Cmd/Status Reg ****
**** TestBench I/O Status ****

/* IDE_MEM_STAT */

INP_MEM_A_DONE_0      0 /* IN MEM COUNTER,
INP_MEM_B_DONE_0      1 /* IN MEM COUNTER,
DBG_MEM_A_DONE_0      2 /* IN MEM COUNTER,
DBG_MEM_B_DONE_0      3 /* IN MEM COUNTER,
EVT_MEM_A_DONE_0      4 /* IN MEM COUNTER,
EVT_MEM_B_DONE_0      5 /* IN MEM COUNTER,
OPERATION_DONE_0      6 /* TEST BENCH OPER
MEM_OP_DONE_W         2 /* 00 AND 11 IS 0
INMEM_A_EMP_0         8
INMEM_A_FULL_0        9
INMEM_B_EMP_0        10
  
```

Offset: 8 bits, Width: 2 bits, DesiredValue: 00000001

Timeout[μSec]: 00100000

**OUTPUT DATA**

Last reply in ROD Status Window.

Poll IDE\_MEM\_STATUS register for INMEM\_A\_EMP=1 and INMEM\_A\_FULL=0.

**Slot#**  
ALL  
Repetitions: 0

SEND TO ROD  
ADD BEFORE  
REPLACE  
ADD AFTER  
ViewPrim  
EDIT PRIM  
DELETE PRIM  
LOAD LIST  
INSERT LIST  
SAVE LIST  
DELETE LIST

**MASTER LIST**

ID	PRIMITIVE	PrimLn	RepLn
1	POLL_REG_FIELD	9	5
2	RW_FIFO	1809	5
3	RW_FIFO	9	1805

Write and read back random data to Input FIFO A.

1

continued on next page

**PRIM LIST EDITOR**  
Main! CommandListEditor! DataFileCreator! RevisionUpdate! HideWindow!

**PRIMITIVE**  
MASTER

**RW\_FIFO**

**INPUT DATA**

READ WRITE

**FifoD&Bank**  
INPUT\_MEM - A

**FifoSize[elems]**  
Input x 8000 48b  
Debug x 1000 48b  
EventAB x 4000 48b  
EventC x 00FF 32b  
Tim x 2000 8b

**NumElements**  
x 00001000

**INPUT FILE**  
c:\RodTS\Data\Testing\fifoTesting\4Krandom1.bin

**OUTPUT DATA**

**OUTPUT FILE** ☐ SAVE OUTPUT DATA TO FILE?

Write random data from file to Input FIFO A.

Clear Clear

**Slot#**  
ALL  
Repetitions 0

SEND TO ROD

ADD BEFORE> Ctrl+  
REPLACE>>  
ADD AFTER>>  
<<EDIT PRIM Ctrl+  
DELETE PRIM Ctrl+Del

LOAD LIST  
INSERT LIST  
SAVE LIST  
DELETE LIST

**MASTER LIST**  
ListLength PrimCount AllPrimCount  
PrimList x 1821 3 3  
ReplList x 1815 3 LastFile  
c:\Rods\Data\Testing\fifoTesting\writeReadInpFifoA.pl

ID	PRIMITIVE	PrimLn	ReplLn
1	POLL_REG_FIELD	9	5
2	RW_FIFO	1809	5
3	RW_FIFO	9	1805

2

3

**PRIM LIST EDITOR**  
Main! CommandListEditor! DataFileCreator! RevisionUpdate! HideWindow!

**PRIMITIVE**  
MASTER

**RW\_FIFO**

**INPUT DATA**

READ WRITE

**FifoD&Bank**  
INPUT\_MEM - A

**FifoSize[elems]**  
Input x 8000 48b  
Debug x 1000 48b  
EventAB x 4000 48b  
EventC x 00FF 32b  
Tim x 2000 8b

**NumElements**  
x 00001000

**INPUT FILE**

**OUTPUT DATA**

**OUTPUT FILE** ☒ SAVE OUTPUT DATA TO FILE?  
inmemA.bin

Read data from file to Input FIFO A and store it to file.

Clear Clear

**Slot#**  
ALL  
Repetitions 0

SEND TO ROD

ADD BEFORE> Ctrl+  
REPLACE>>  
ADD AFTER>>  
<<EDIT PRIM Ctrl+  
DELETE PRIM Ctrl+Del

LOAD LIST  
INSERT LIST  
SAVE LIST  
DELETE LIST

**MASTER LIST**  
ListLength PrimCount AllPrimCount  
PrimList x 1821 3 3  
ReplList x 1815 3 LastFile  
c:\Rods\Data\Testing\fifoTesting\writeReadInpFifoA.pl

ID	PRIMITIVE	PrimLn	ReplLn
1	POLL_REG_FIELD	9	5
2	RW_FIFO	1809	5
3	RW_FIFO	9	1805

Write and read back random data to Input FIFO A.

5/12/03

RGD TestStand Overview

11



## Primitive List File (\*.pl) example: (created and saved using *PrimitiveListEditor*)

writeReadInpFifoA.pl

	Write and read back random data to Input FIFO A. ----- length: 0x1821, numMsgs= 0x3, primListRevision= 111 allPrimCount: 3, replyLength: 0x1815, replyCount: 0x3 =====	← <i>PrimList comment</i>  - <i>PrimListTable header</i>
1st primitive	Poll IDE_MEM_STATUS register for INMEM_A_EMP=1 and INMEM_A_FULL=0. ----- index: 0, id: 0x2005, primRevision: 104, length: 0x9, replyLength: 0x5 POLL_REG_FIELD registerID= 0x227, offset= 8, width= 2, desiredValue= 0x1, timeoutInUsec= 0x100000 baseId= 0x227, x= 0, y= 0 =====	← <i>primitive comment</i>  - <i>primitive header</i> - <i>primitive name</i> - <i>primitive params</i>
2nd primitive	Write random data from file to Input FIFO A. ----- index: 1, id: 0x2006, primRevision: 104, length: 0x1809, replyLength: 0x5 RW_FIFO fifoId= 0, bank= 0, readNotWrite= 0, numElements= 0x1000, *dataBaseAdr= 0xFFFFFFFF dataFileName[]= c:\RodTS\Data\Testing\fifoTesting\4Krandom1.bin =====	
3rd primitive	Read data from file to Input FIFO A and store it to file. ----- index: 2, id: 0x2006, primRevision: 104, length: 0x9, replyLength: 0x1805 RW_FIFO fifoId= 0, bank= 0, readNotWrite= 1, numElements= 0x1000, *dataBaseAdr= 0xFFFFFFFF dataFileName[]= inmemA.bin	

**Note:** The comments can be edited also outside the CommandList Editor using any text editor.  
Number of comment lines is not explicitly limited.

# Command List

In general any Rod task is defined in the TestStand environment as a sequence of the basic commands executed by the host - **CommandList**, which can be edited and saved in the interactive *CommandListEditor*. The implementation and the structure of the *CommandList* is very similar to the *PrimitiveList* (CommandList is the host parallel to the Rod PrimitiveList). => TestStand as a “list processor” is ready to run most of possible user defined tasks **without recompiling the code**.

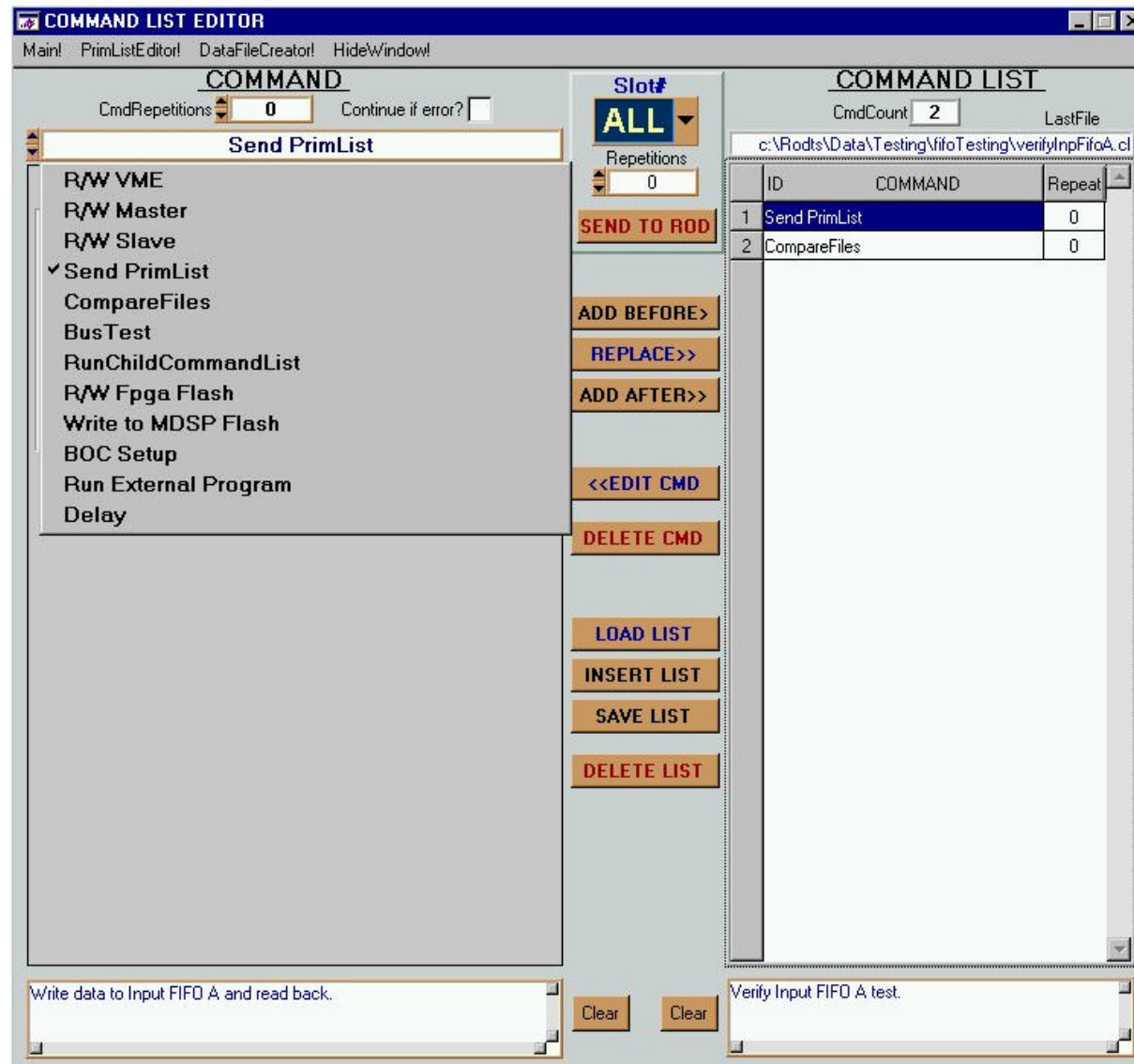
## Currently implemented commands:

- R/W Master DSP memory
- R/W Slave DSP memory
- **Send PrimitiveList**
- Compare Files
- Bus Test
- Run Child Command List (the parent command list can execute other command lists)
- R/W FPGA Flashes
- Write to Master DSP Flash
- BOC Setup (used temporarily, PrimitiveList should be used instead)
- Run External Program - user can (create and) call any executable (so far used mainly for data processing and conversion)
- Delay.

**New commands can be easily added** in the TestStand if the current set is not sufficient for some tests (for example if some variable primitive lists must be built “on the fly”) or if it’s useful to move more complicated and often repeated tasks into separate commands.

TestStand is also able to execute *CommandLists* and *PrimitiveLists* (in “raw” or *PrimListTable* format) sent from another application.

## Command List Editor:





## Command List Internal Structure:

commandListDefinitions.h

```
/*----- COMMAND LIST DEFINITIONS -----*/
#define MAXCOMMANDS_IN_LIST 100
#define CMD_COMMENT_LENGTH 600

struct COMMAND{
    char comment[CMD_COMMENT_LENGTH]; /* command comment */
    int id; /* command id */
    UINT32 numRepetitions; /* number of command repetitions; 0 means execute once, don't repeat */
    int continueIfError; /* if set, continue to the next command in the list even
                           if this command returned error */
    union COMMAND_PARAMS_UNION params; /* command parameters */
};

struct COMMAND_LIST{
    char comment[CMD_COMMENT_LENGTH]; /* cmd list comment */
    int commandCount; /* number of commands in the list */
    struct COMMAND command[MAXCOMMANDS_IN_LIST]; /* array of commands */
};

-----

commandFunc_sendPrimList.h
/* SEND PRIM LIST command params example - included in COMMAND_PARAMS_UNION */
struct SEND_PRIM_LIST_PARAMS{
    int repeatExecNotBuild; /* repeat only execution of the list not building */
    char fileName[PATHNAME_LENGTH]; /* primitive list file */
};
```

**Note:** Again, the number of *Commands* in the *CommadList* is limited (current max is 100), but can be easily redefined if necessary, as well as This number can be redefined if necessary as well as the comment array size (now max 600characters).

Command List example:

1

COMMAND LIST EDITOR

Main! PrimListEditor! DataFileCreator! HideWindow!

COMMAND

CmdRepetitions 0 Continue if error? ☐

Send PrimList

INPUT DATA

Repeat

EXECUTION BUILDING

PrimList

c:\Rods\Data\Testing\VifoTesting\writeReadInpFifoA.pl

Edit PrimList

Slot# ALL

Repetitions 0

SEND TO ROD

ADD BEFORE>

REPLACE>>

ADD AFTER>>

<<EDIT CMD

DELETE CMD

LOAD LIST

INSERT LIST

SAVE LIST

DELETE LIST

COMMAND LIST

CmdCount 2 LastFile c:\Rods\Data\Testing\VifoTesting\verifyInpFifoA.cl

ID	COMMAND	Repeat
1	Send PrimList	0
2	CompareFiles	0

Write data to Input FIFO A and read back.

Clear

Clear

Verify Input

2

COMMAND LIST EDITOR

Main! PrimListEditor! DataFileCreator! HideWindow!

COMMAND

CmdRepetitions 0 Continue if error? ☐

CompareFiles

INPUT DATA

WordSize 48 bits

DataMask 0000FFFF FFFFFFFF

COMMON DATA FILE

c:\RodTS\Data\Testing\VifoTesting\32Krandom1.bin

ROD DATA FILE (in RodData directory)

inmemA.bin

Slot# ALL

Repetitions 0

SEND TO ROD

ADD BEFORE>

REPLACE>>

ADD AFTER>>

<<EDIT CMD

DELETE CMD

LOAD LIST

INSERT LIST

SAVE LIST

DELETE LIST

COMMAND LIST

CmdCount 2 LastFile c:\Rods\Data\Testing\VifoTesting\verifyInpFifoA.cl

ID	COMMAND	Repeat
1	Send PrimList	0
2	CompareFiles	0

Compare data readout from Input FIFO A with the written original.

Clear

Clear

Verify Input FIFO A test.

## Command List File (\*.cl) example: (created and saved using *CommandListEditor*)

```
verifyInpFifoA.cl  
  
Verify Input FIFO A test.  
-----  
commandCount: 2  
=====
```

**1st command** {  
Write data to Input FIFO A and read back.  
-----  
index: 0, id: 3, numRepetitions: 0, continueOnError: 0  
SEND\_PRIM\_LIST\_COMMAND\_ID  
repeatExecNotBuild= 1  
fileName[]= c:\Rodts\Data\Testing\fifoTesting\writeReadInpFifoA.pl  
=====

**2nd command** {  
Compare data readout from Input FIFO A with the written original.  
-----  
index: 1, id: 4, numRepetitions: 0, continueOnError: 0  
COMPARE\_FILES\_COMMAND\_ID  
wordSizeInBytes= 6, dataMask[]={ 0xFFFFFFFF, 0xFFFF }  
commonFile[]= c:\RodTS\Data\Testing\fifoTesting\32Krandom1.bin  
rodFile[]= inmemA.bin

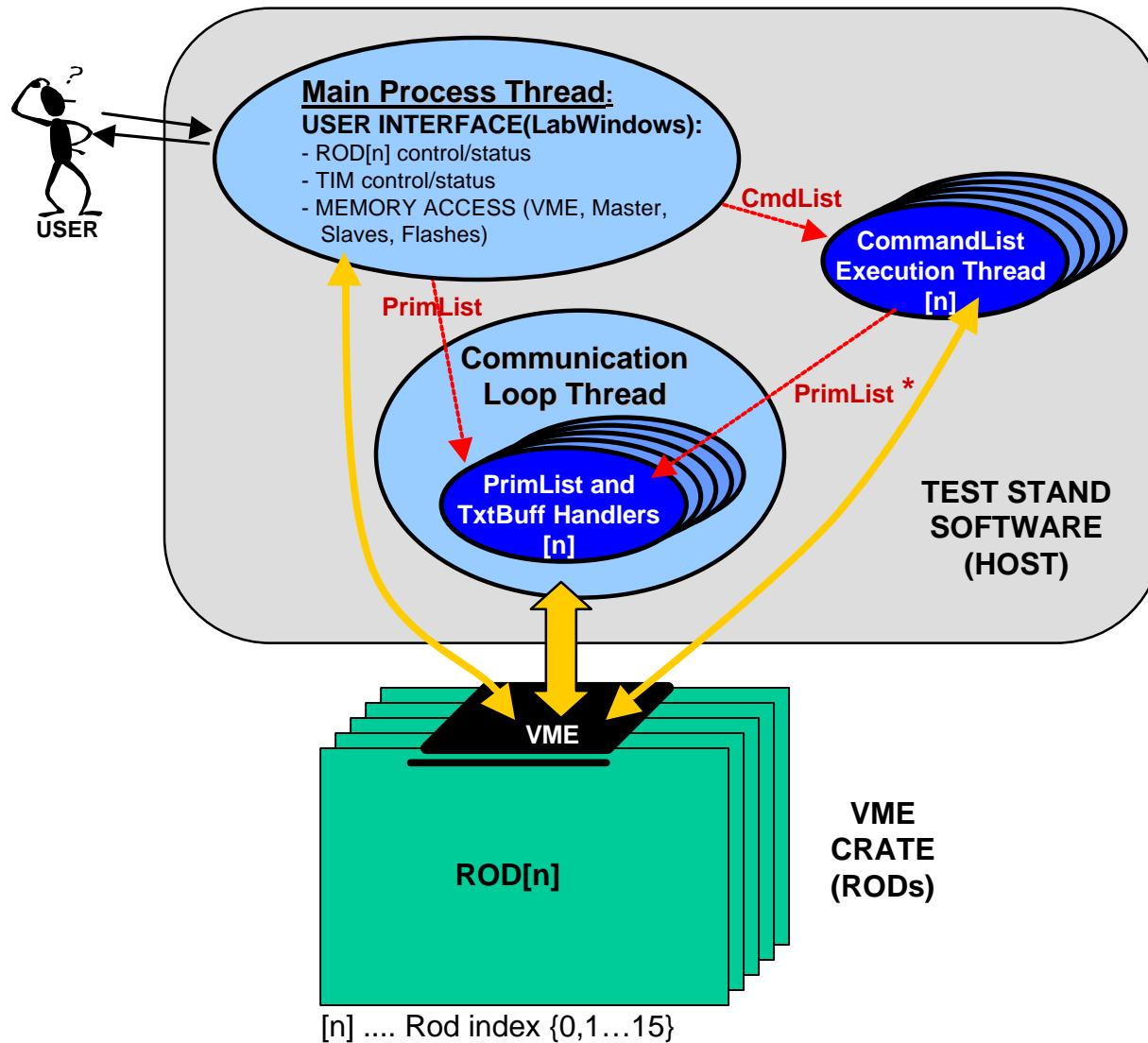
← *CommandList comment*  
- *CommandList header*  
← *command comment*  
- *command header*  
- *command name*  
- *command params*

**Note:** The comments can be edited also outside the CommandList Editor using any text editor.  
Number of comment lines is not explicitly limited.



# Test Stand Structure

- *Single process multithreaded* application:



\* Commands with VME access are synchronized/executed via CommunicationLoop.

# Threads

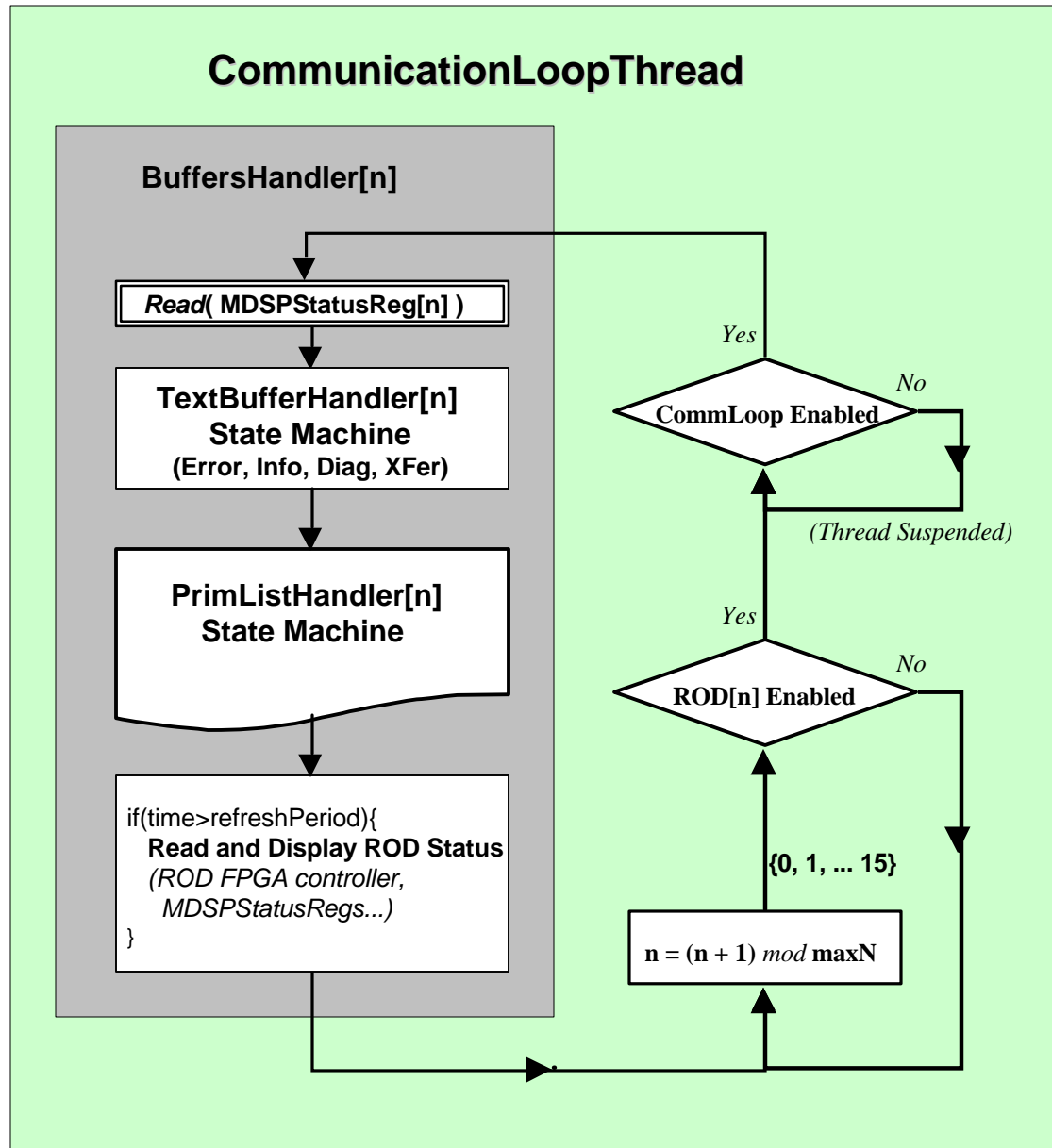
- **Main Thread**

- **interaction with the User** (LabWindows GUI),
  - Main Panel (program options, thread priorities...),
  - RW Memory Panels(VME, MDSP, SDSPs, Flashes, FPGA Control/Status),
  - PrimList Editor, CommandList Editor,
  - ROD Status/Control Panel;

- **CmdListExecution Thread[n]** - *CommandList* execution.

- **CommunicationLoop**

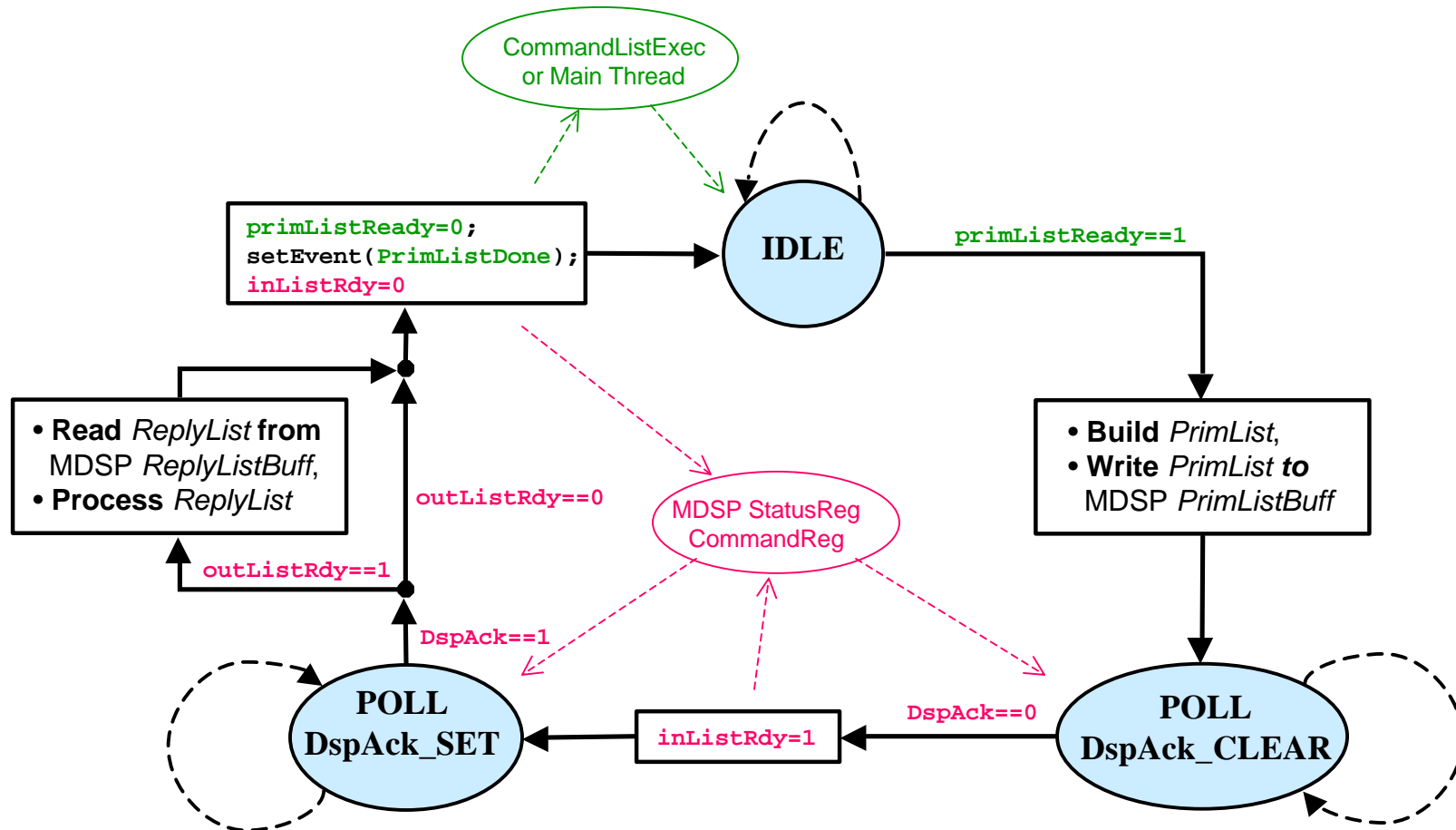
- common to all RODs,
- distributes an access to the shared VME bus between all RODs in the crate on *“round robin”* basis,
- executes PrimitiveLists and processes TextBuffer messages;
- periodically monitors the status of all Rods in the crate.





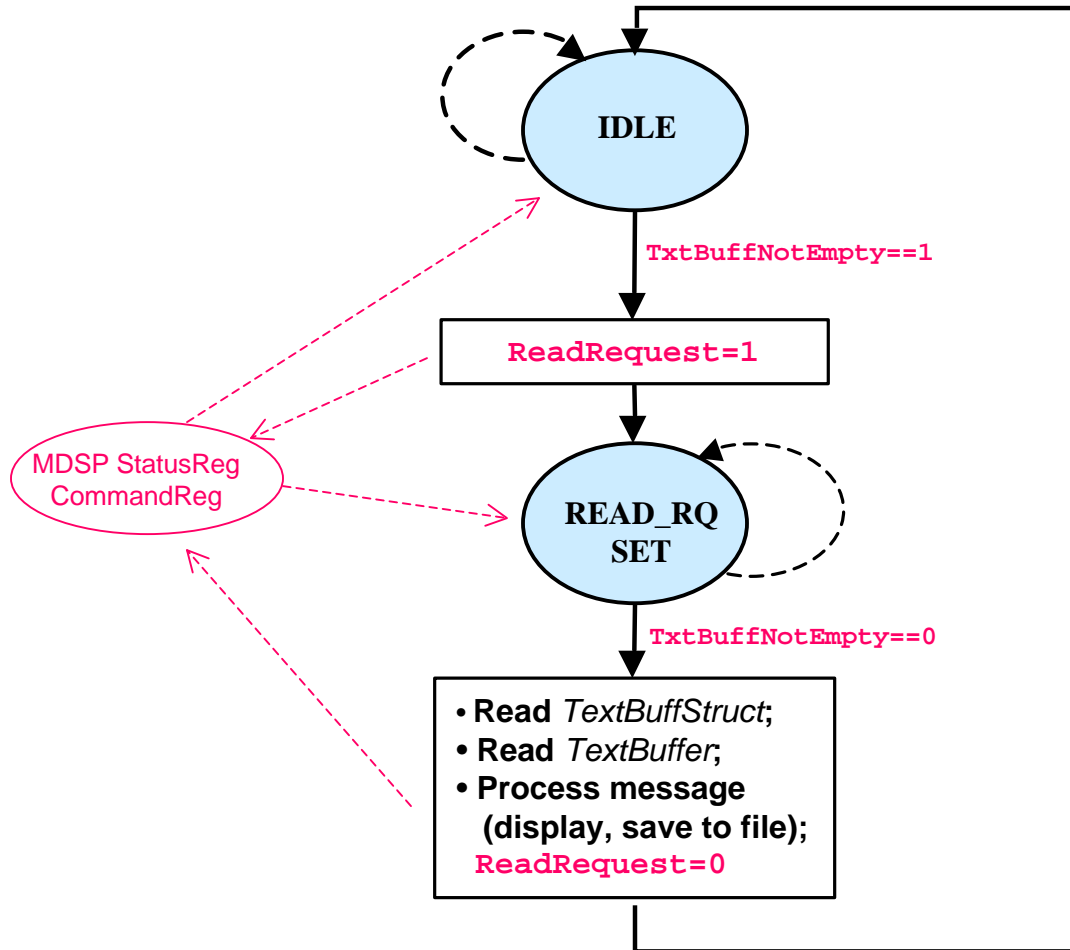
## PrimListHandler[n] State Machine

- executes *PrimitiveListTable* (builds *PrimitiveList*, send it to Rod, processes *ReplyData*).



## TextBufferHandler[n] State Machine

- reads and processes Text Messages from MasterDSP (*Error, Info, Diagnostic, XFer*).



# Thread priorities

- *MainProcessThread(UserInterface) > the other threads*

- the user interface must be responsive at any moment!!

- *CommunicationThread <= CmdListExecutonThreads*

When CommunicationLoop is not suspended, the processor is always 100% busy (no additional delays included).

The *CmdListExecutonThreads* are active only for a limited period of time when executing a *CommandList*, so they can have a higher priority than the *CommunicationLoop*.