# ИНФОРМАЦИОННО-УПРАВЛЯЮЩИЕ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ

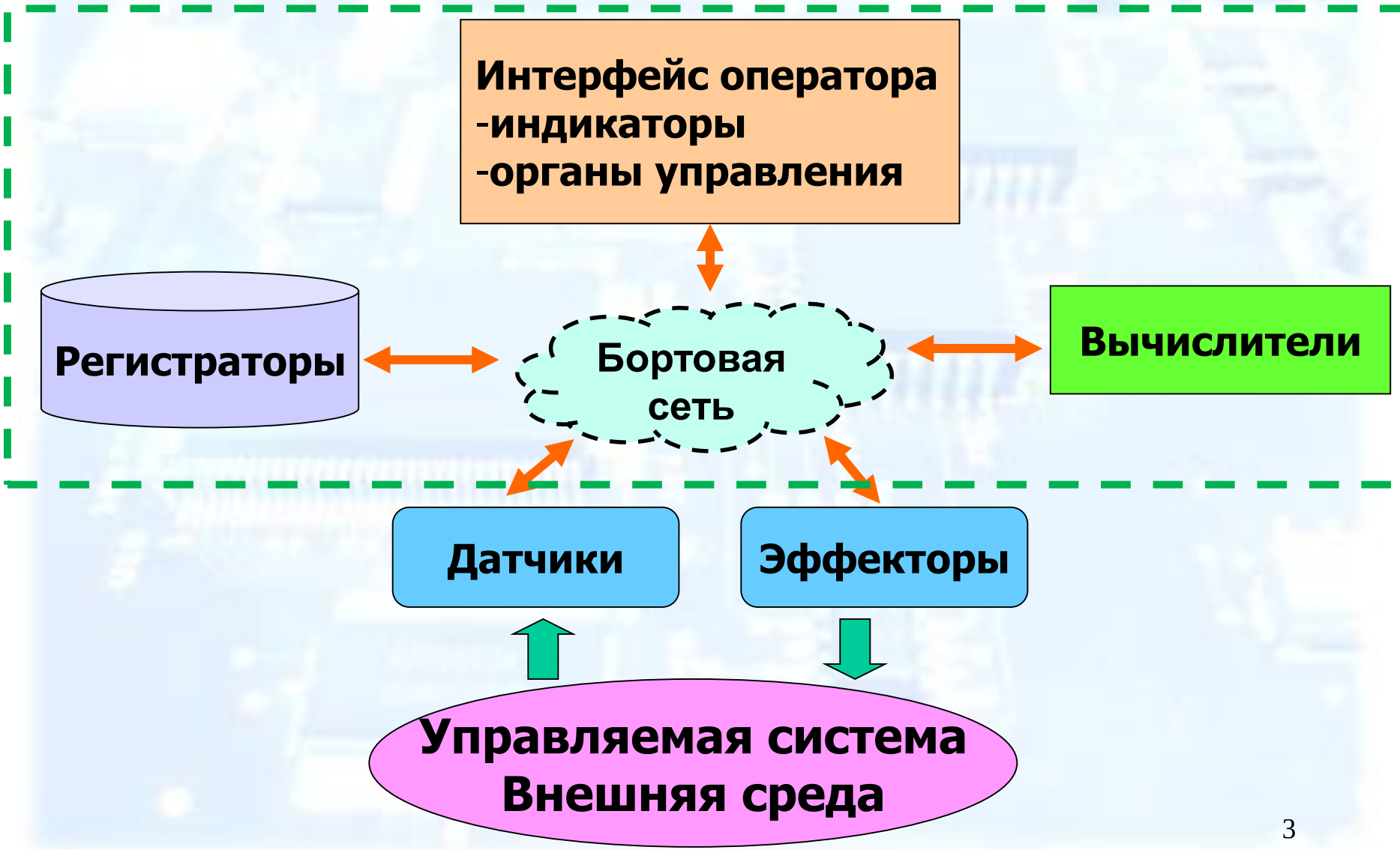
## Лекция 11:
## *Тестирование и интеграция ИУС РВ*

Кафедра АСВК,
Лаборатория Вычислительных Комплексов
Балашов В.В.

# ИУС РВ

- Информационно-управляющая система (ИУС) – вычислительная система верхнего уровня, обеспечивающая:
  - функциональную и информационную интеграцию составных частей управляемого объекта
  - взаимодействие между объектом и оператором

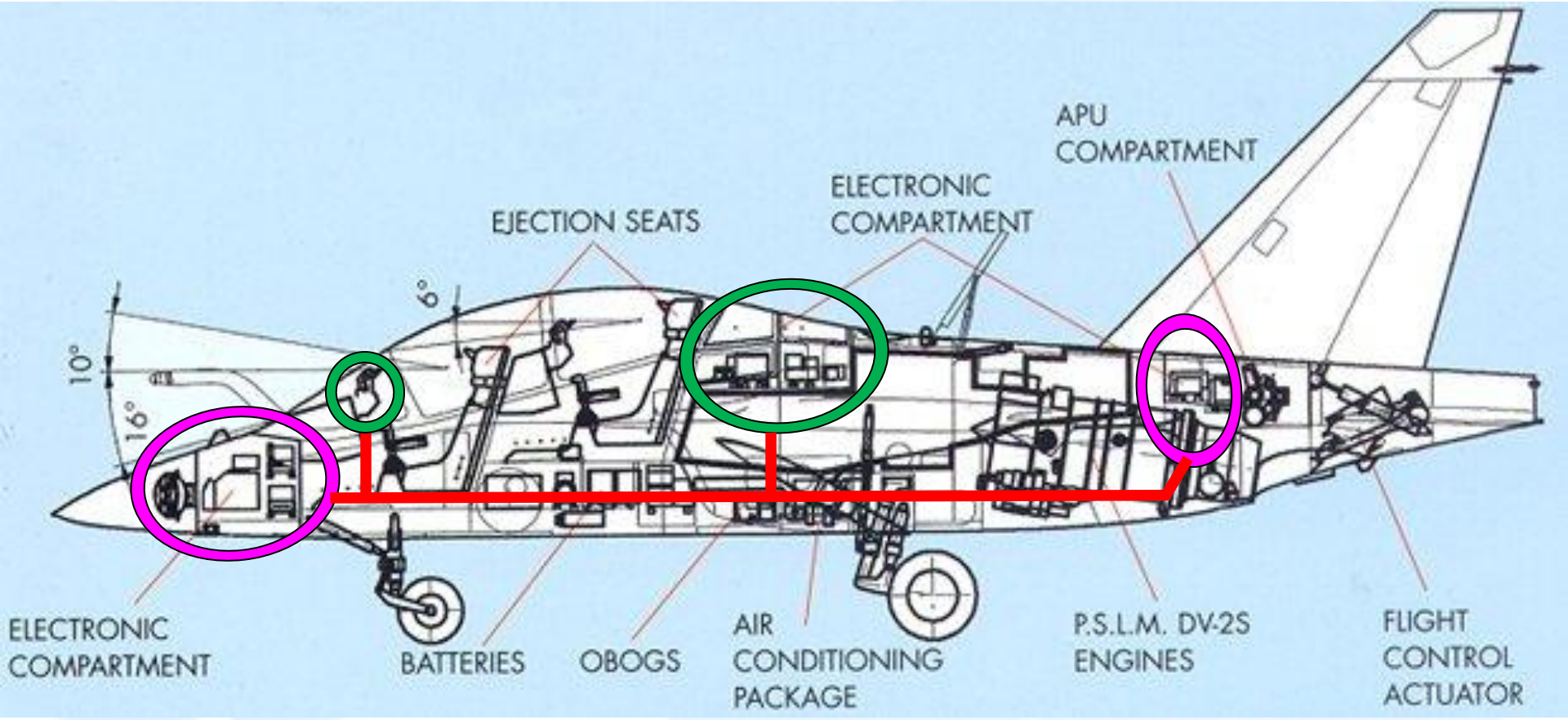
# Состав ИУС


Интерфейс оператора
-индикаторы
-органы управления
Регистраторы
Бортовая сеть
Вычислители
Датчики
Эффекторы
Управляемая система
Внешняя среда

# Функции ИУС

- Контроль состояния управляемого объекта
- Управление движением объекта или его частей
- Отслеживание положения объекта или его частей в пространстве
- Обмен данными с внешними системами
- Управление специализированными приборами (прикладной нагрузкой)
- Обмен данными с оператором
  - отображение данных
  - ввод данных

# ИУС в управляемой системе



# Устройства в составе ИУС

# Устройства вне ИУС (на примере самолета)

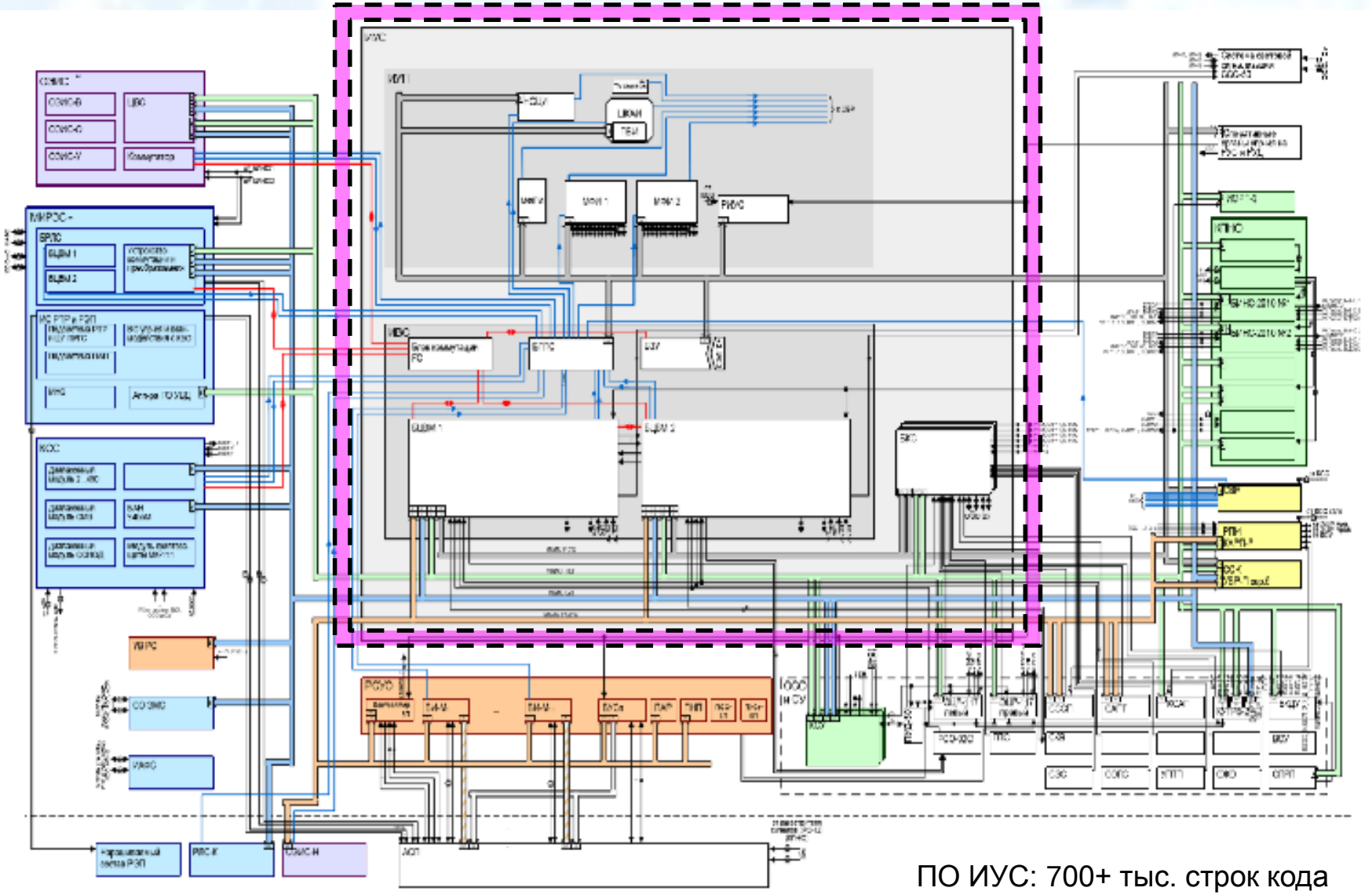
- Датчики:
  - Радиолокатор
  - Радиовысотомер
  - Топливный датчик
  - Тахометр (измеритель частоты вращения турбины)
  - Датчики спутниковой навигации
- Эффекторы
  - управление двигателем
  - управление рулями высоты
  - управление выпуском шасси
  - радиопередатчик

# Неоднородность ИУС

- Каналы: точка-точка, шина, коммутатор; 12 kbps, 1 Mbps, 1 Gbps
- Устройства: датчики, индикаторы, вычислители, органы управления, исполнительные устройства
- Данные: аналоговые, цифровые; числовые массивы, видеопотоки

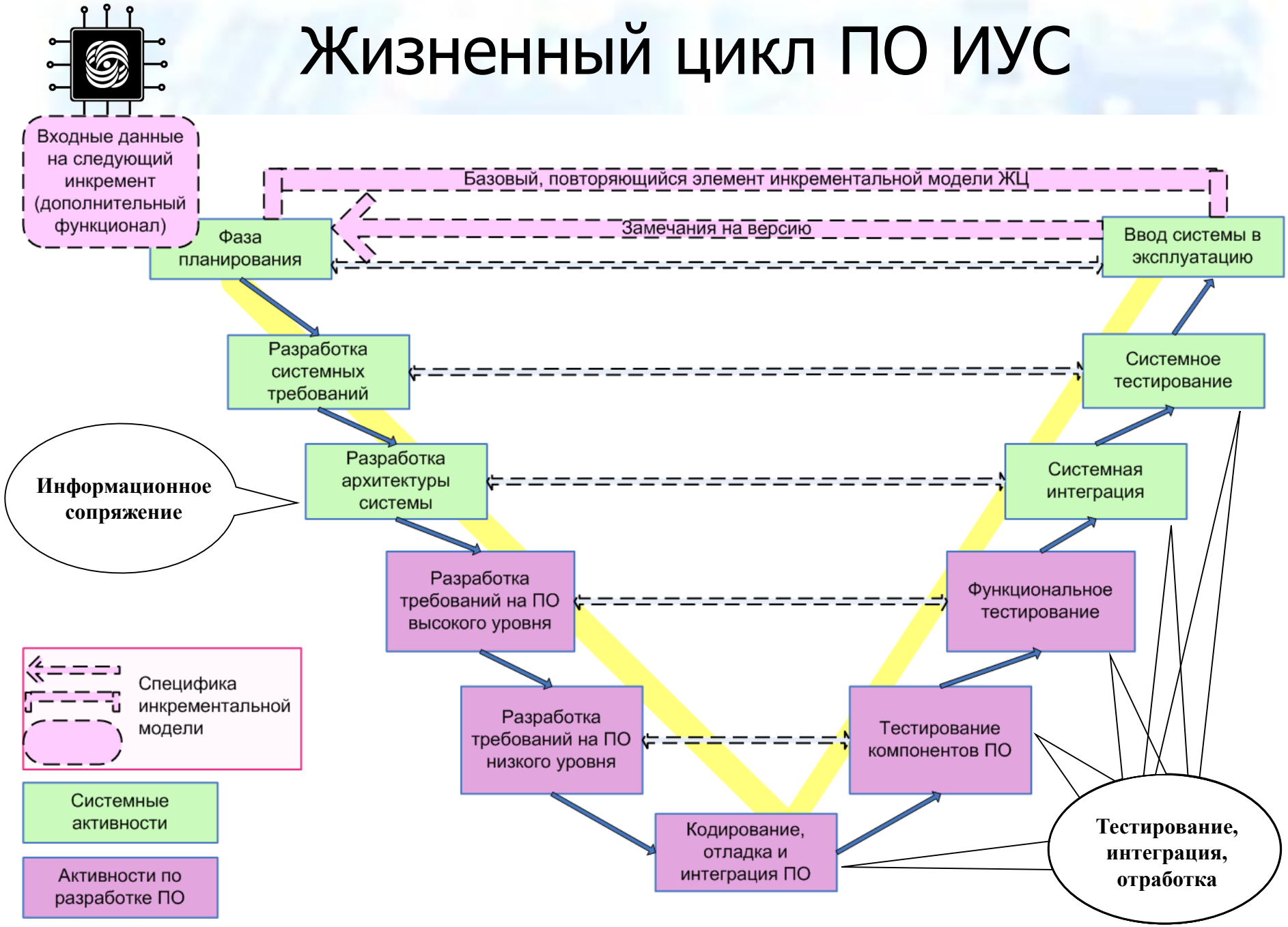
*Проблема унаследованных устройств*

# Уровень сложности комплекса бортового оборудования



ПО ИУС: 700+ тыс. строк кода

# Жизненный цикл ПО ИУС



# Требования к средствам тестирования ИУС

- Поддержка тестирования ПО на целевом вычислителе без инструментирования аппаратуры или ПО вычислителя
- Поддержка обмена данными через все типы каналов, используемых в ИУС
  - выдача в каналы тестовых данных и прием ответных данных для последующего анализа
  - мониторинг обмена по каналам бортовых интерфейсов и обеспечение доступа тестовых сценариев к результатам мониторинга
  - поддержка формирования сбойного трафика
- Поддержка тестирования временн*ы*х характеристик функционирования целевой системы
  - формирование и выдача тестовых данных в режиме реального времени
  - измерение задержек поступления ответных данных
- Поддержка многомашинных конфигураций
  - одновременный обмен по десяткам (сотням) каналов

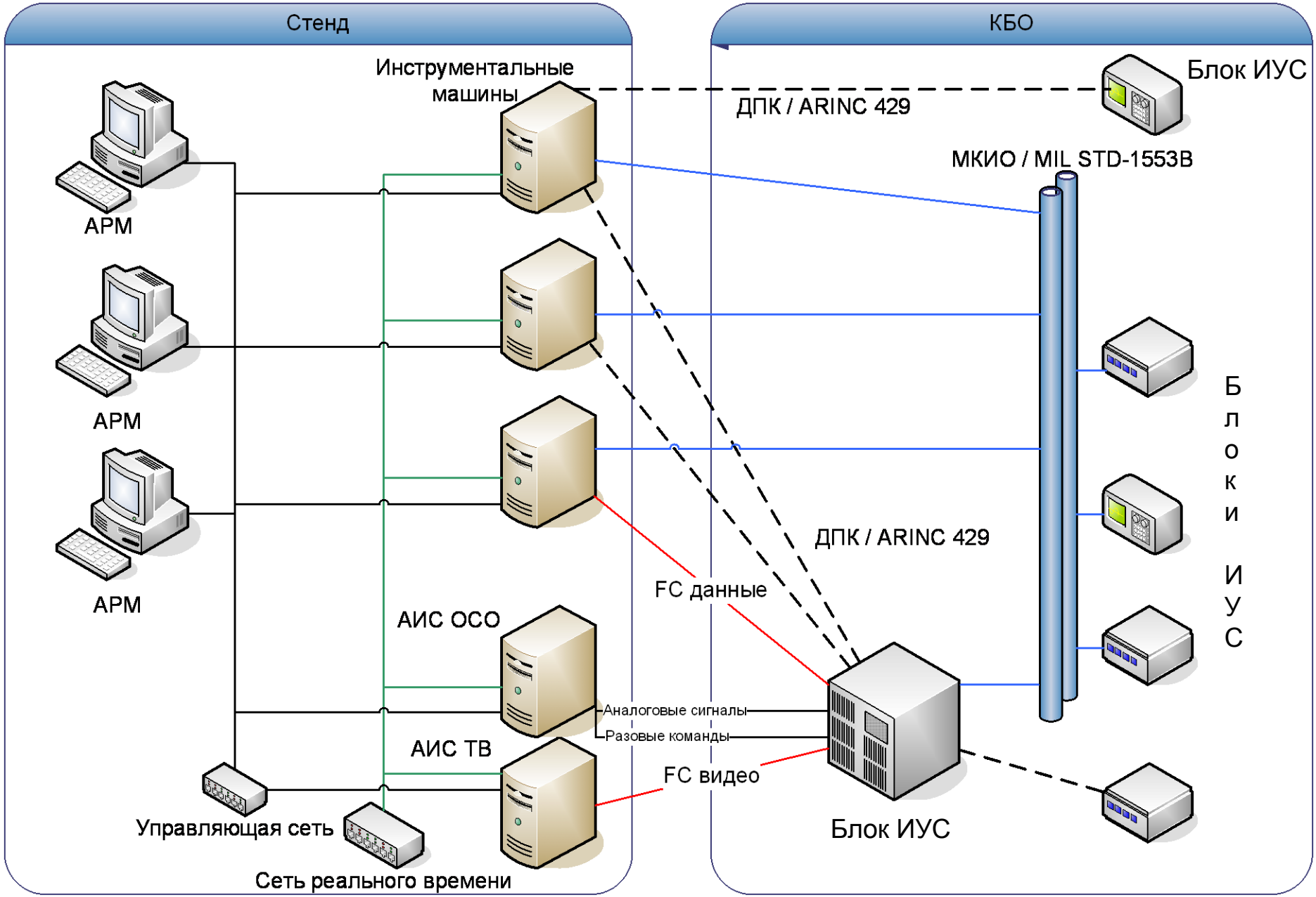
# Требования к средствам тестирования ИУС

- Поддержка автоматического и интерактивного тестирования
  - интерактивное: для проверки индикаторов и пультов управления
- Поддержка пакетного режима выполнения тестов
- Поддержка оперативного управления тестированием
  - выбор порядка выполнения тестов
  - задание значений тестовых данных
- Поддержка оперативного отображения хода тестирования
  - значения тестовых данных и ответных данных от тестируемой системы
  - протоколы тестирования
- Поддержка прослеживаемости требований и формирования отчётов по результатам тестирования
  - задание соответствия требований тестовым сценариям
  - формирование матрицы прослеживаемости требований
  - формирование отчёта о прохождении тестов и выполненности требований по результатам тестирования
- Интеграция со средствами поддержки разработки ПО ИУС
  - средства управления версиями (хранение ПО и тестов в едином репозитории)
  - средства управления требованиями
  - база данных бортовых интерфейсов (форматы информационных сообщений)
- Единый подход к тестированию для различных фаз жизненного цикла ИУС

# Комплекс средств тестирования ИУС

- Разработан в Лаборатории вычислительных комплексов ВМК МГУ
- Предназначен для тестирования устройств ИУС через каналы бортовых интерфейсов (КБИ)
- Функционирует на ПК под управлением ОС Linux, в состав которых входят адаптеры КБИ
- Поддерживает распределенное выполнение тестовых сценариев
- Удовлетворяет перечисленным выше требованиям
- Положен в основу семейства стендов тестирования, отработки и интеграции ИУС

# Архитектура стенда тестирования ИУС

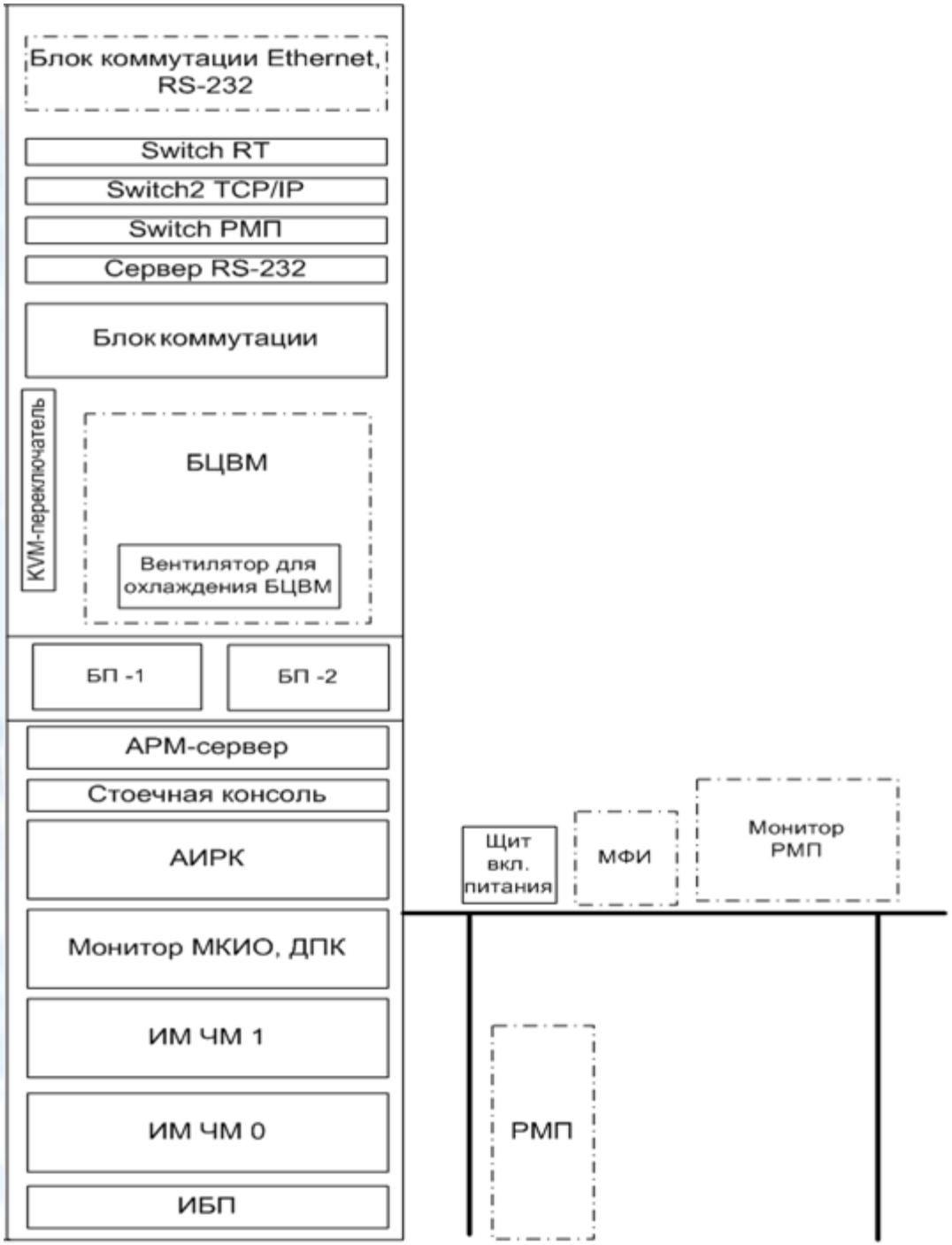


# Задачи, требующие работы с натурными устройствами ИУС

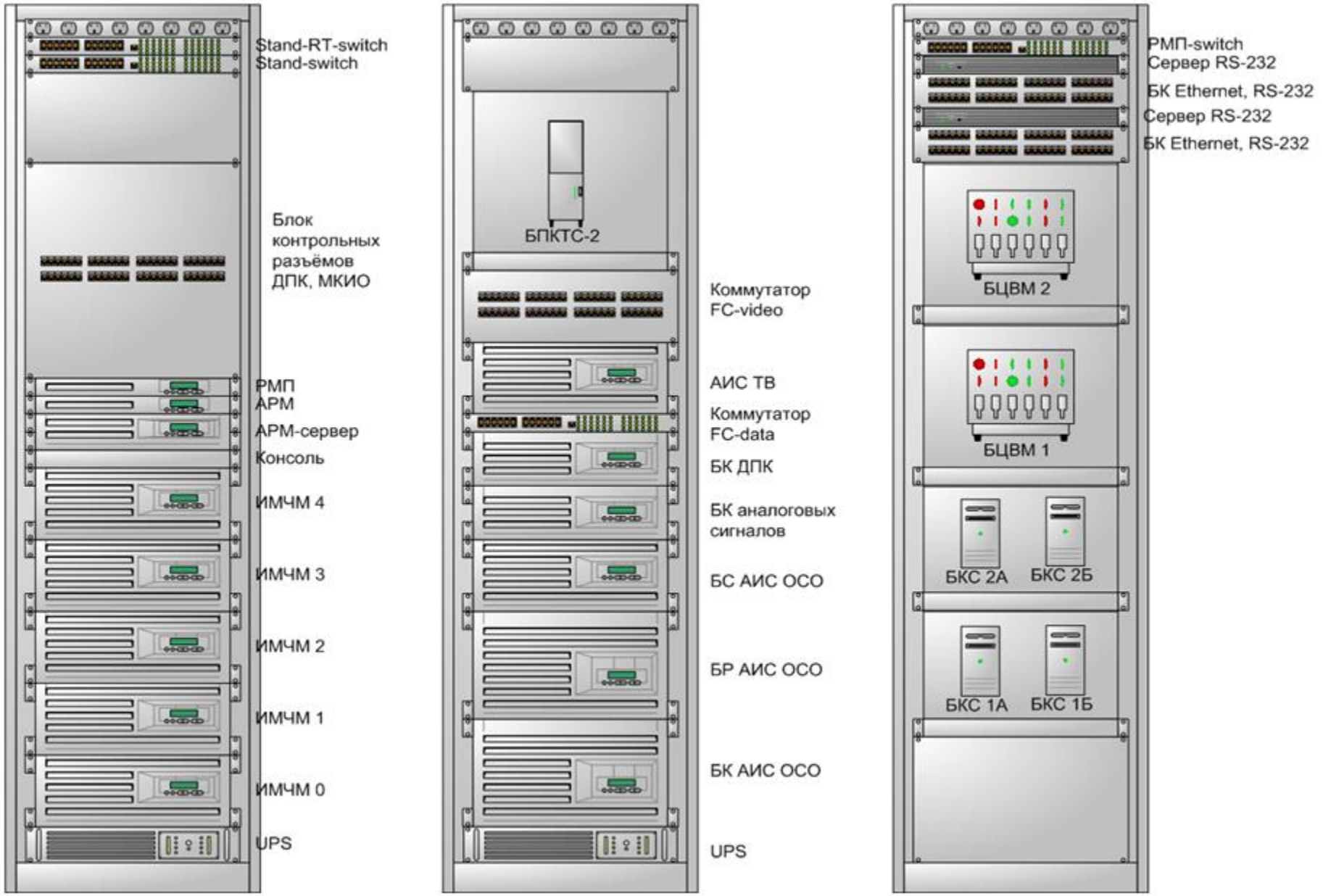
- интеграция аппаратуры и ПО, отладка ПО ИУС на целевой платформе
- интеграция компонентов ПО ИУС, в т.ч. компонентов, поступающих от предприятий-соразработчиков
- интеграция подсистем ИУС, а также ИУС в целом как многокомпонентной аппаратно-программной системы
- функциональное и квалификационное тестирование ПО ИУС
- приемосдаточные испытания серийно выпускаемых комплектов ИУС
- диагностика блоков ИУС, по которым поступили рекламации
- диагностика блоков ИУС в составе объекта

# Стенд отработки ПО БЦВМ

# Стенд отработки БЦВМ+МФИ



# Стенд тестирования серийных комплектов ИУС



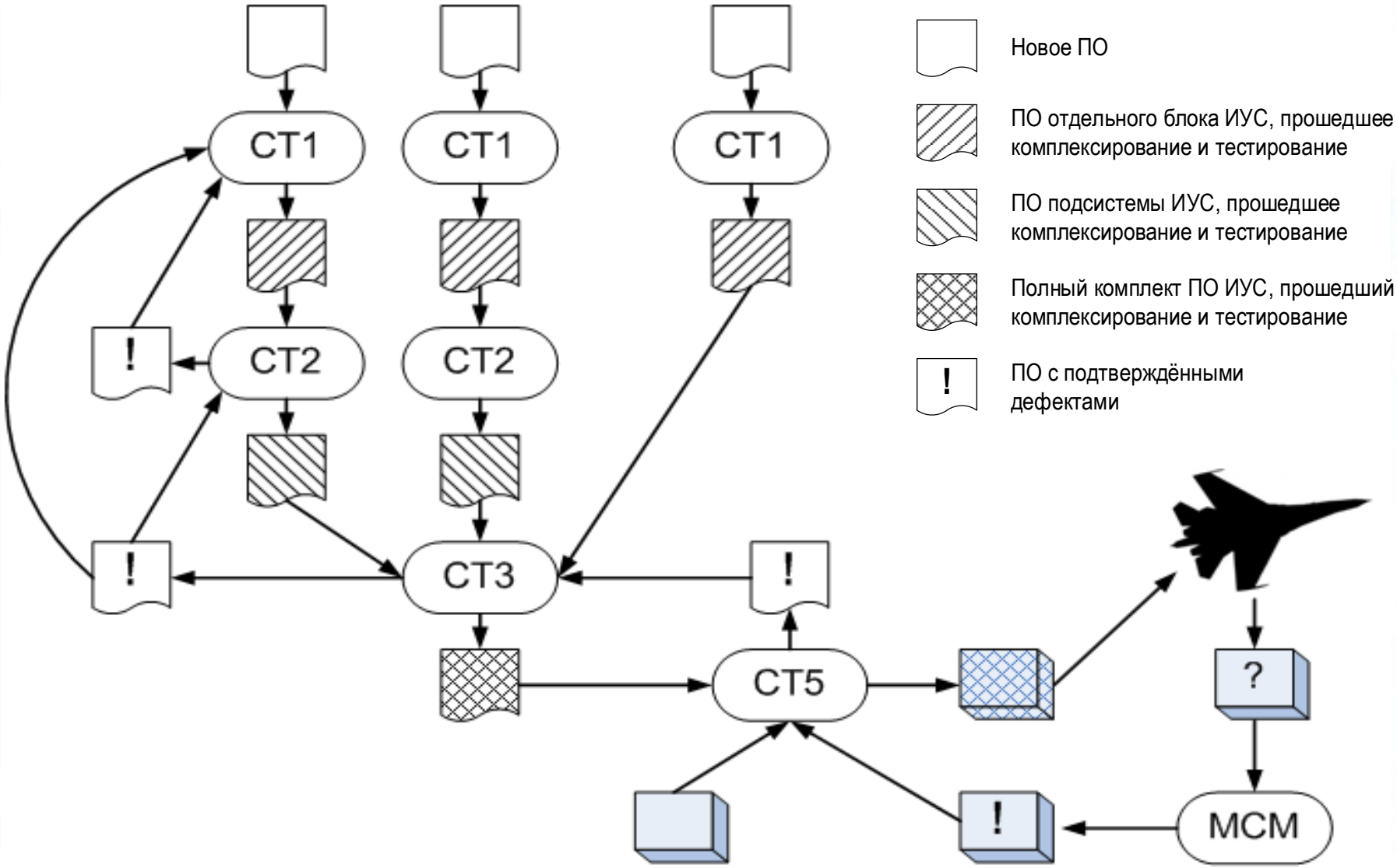
# Мобильная рабочая станция мониторинга и анализа бортовых интерфейсов

Промышленный ноутбук

Мобильный коммутатор Fibre Channel

Адаптеры бортовых интерфейсов (MIL STD-1553B, Fibre Channel и др.)

# Технологический цикл применения семейства стендов отработки ИУС



# Технологический цикл применения семейства стендов отработки ИУС

## Обозначения

### Программное обеспечение (ПО)

Новое ПО

ПО отдельного блока ИУС, прошедшее комплексирование и тестирование

ПО подсистемы ИУС, прошедшее комплексирование и тестирование

Полный комплект ПО ИУС, прошедший комплексирование и тестирование

ПО с подтверждёнными дефектами

### Стенды

СТ1 — Стенд тестирования и отладки ПО отдельного блока ИУС

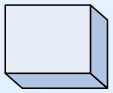
СТ2 — Стенд тестирования и отладки ПО нескольких сопряжённых блоков ИУС

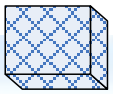
СТ3 — Стенд комплексирования и приёмосдаточных испытаний ИУС

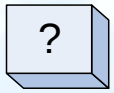
СТ5 — Стенд приёмосдаточных испытаний серийных комплектов ИУС

МСМ — Мобильная рабочая станция мониторинга и тестирования блоков ИУС

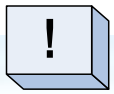
### Аппаратура


Новые серийные образцы блоков ИУС


Комплект аппаратуры и ПО ИУС, прошедший комплексирование и приёмосдаточные испытания

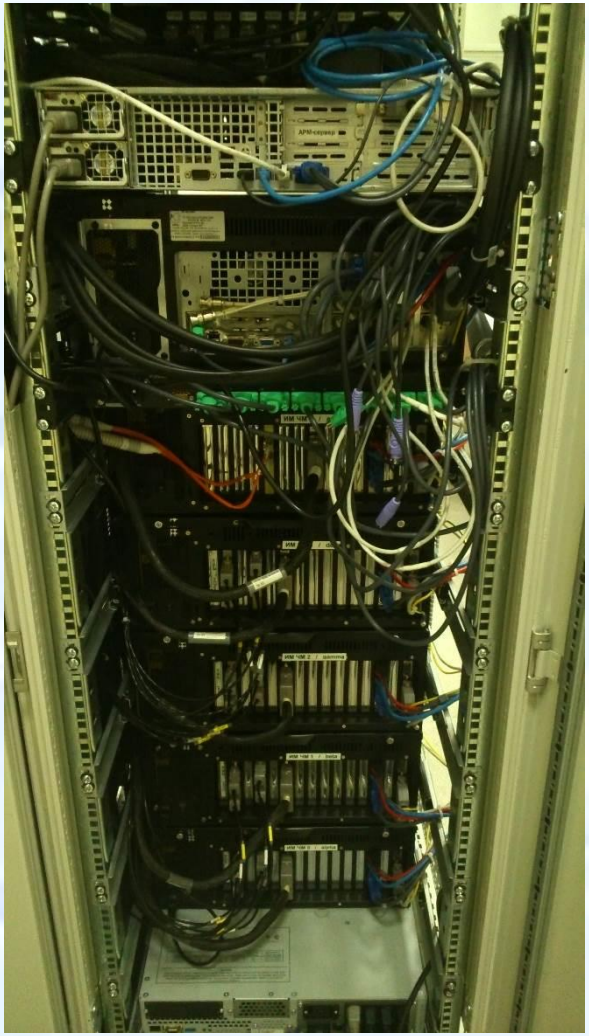


Комплект ИУС, предположительно имеющий дефекты



Блок ИУС с выявленными дефектами

# Стенд тестирования серийных комплектов ИУС Су-35



# Стенд интеграции ИУС МС-21

ADVANTECH

- ACIB – Avionics Core Integration Bench

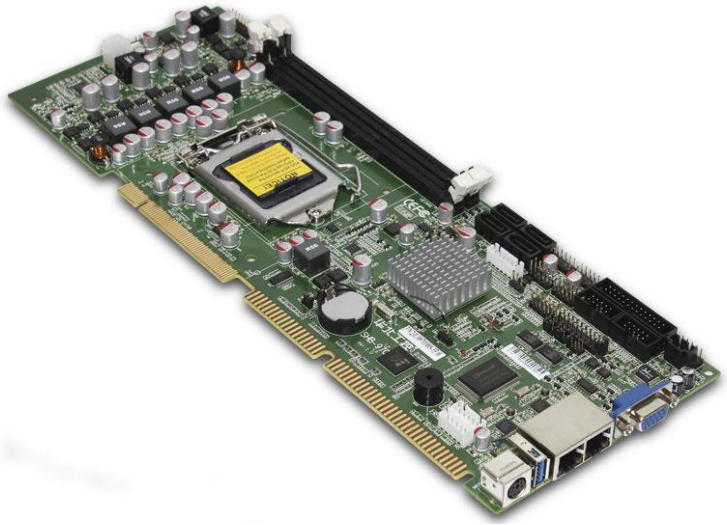
# Стенд «Электронная птица»

- SIB – System Integration Bench
- Компьютерное моделирование бортового электромеханического оборудования

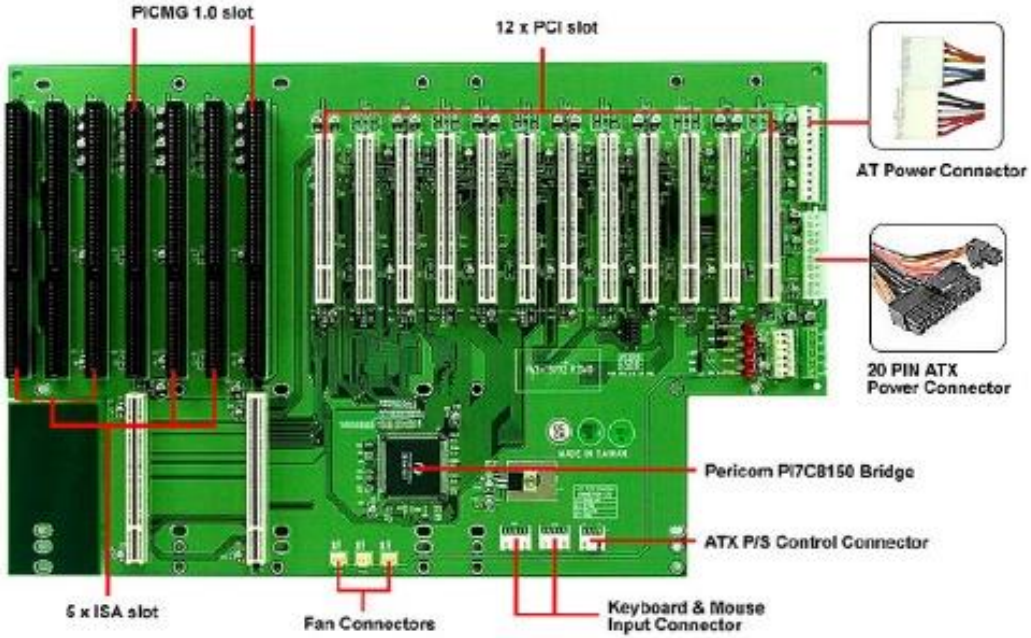
# Стенд «Железная птица»

- Установлена б**о**льшая часть электромеханического оборудования
- «Самолёт на полках», дальше – прототип самолёта в сборе

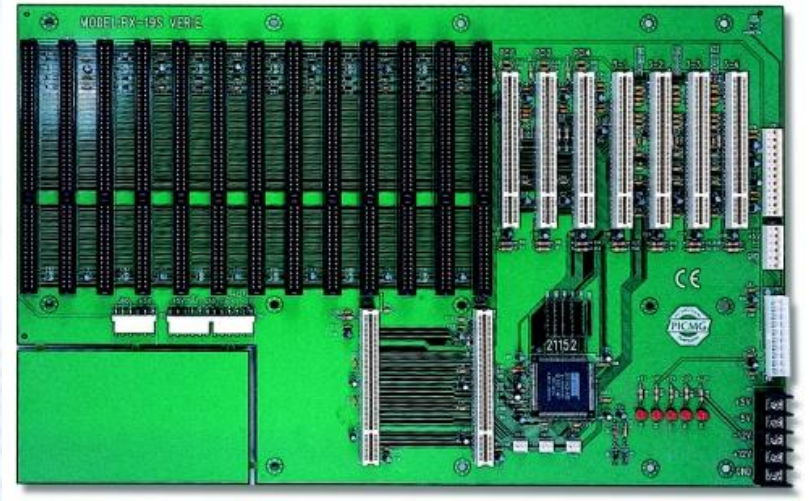
# Аппаратная база стенда

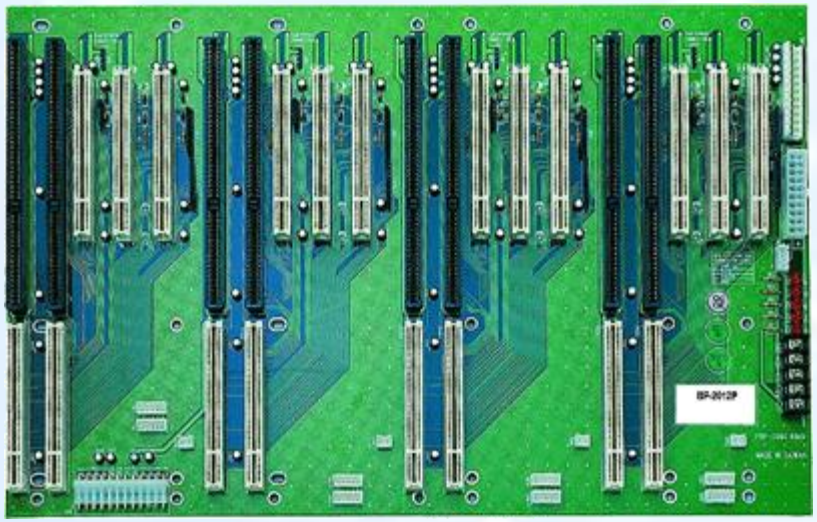


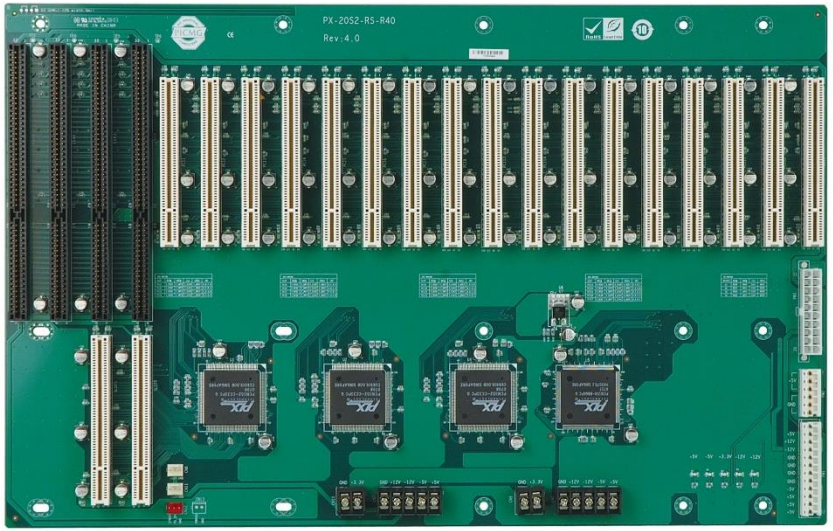
- Промышленные компьютеры (PICMG)
- Платы-расширители шин
  - ISA
  - PCI
  - PCI Express
- Процессорные платы
- Адаптеры бортовых интерфейсов



# Аппаратная база стенда

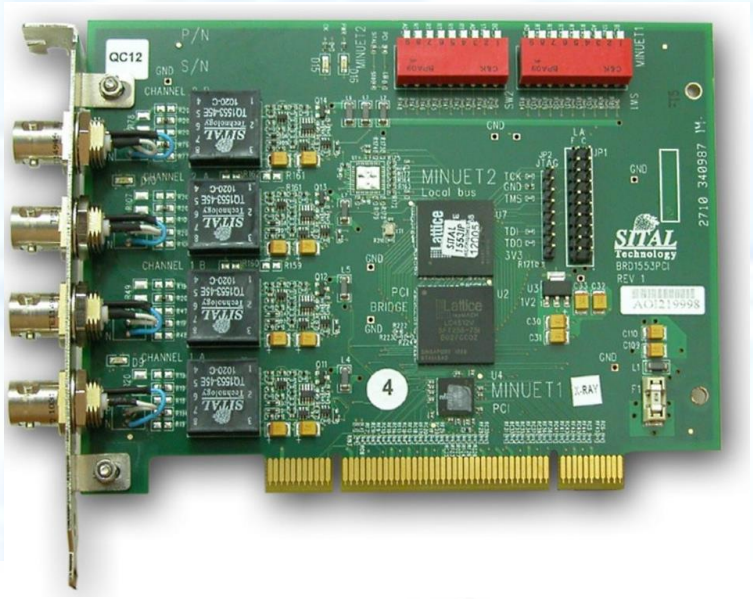


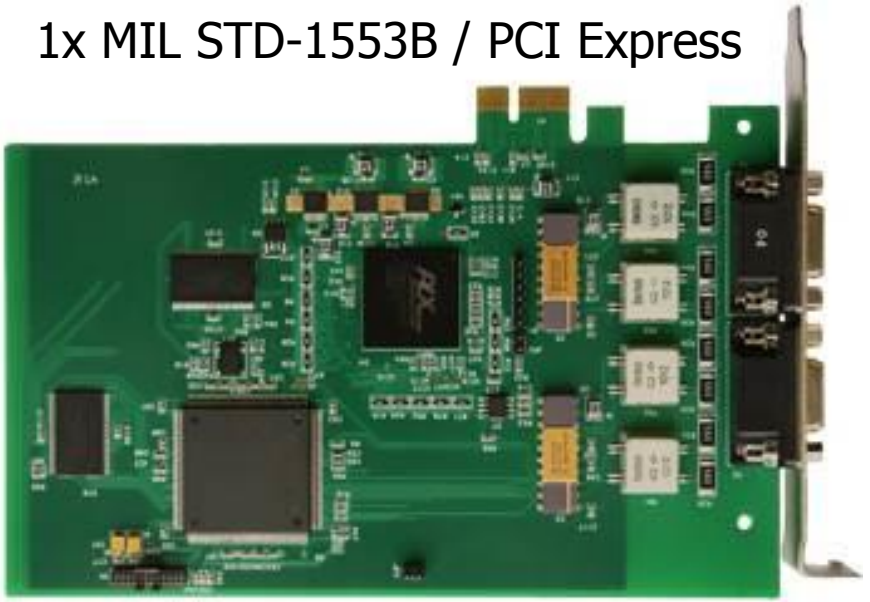


# Аппаратная база стенда

2x MIL STD-1553B / PCI

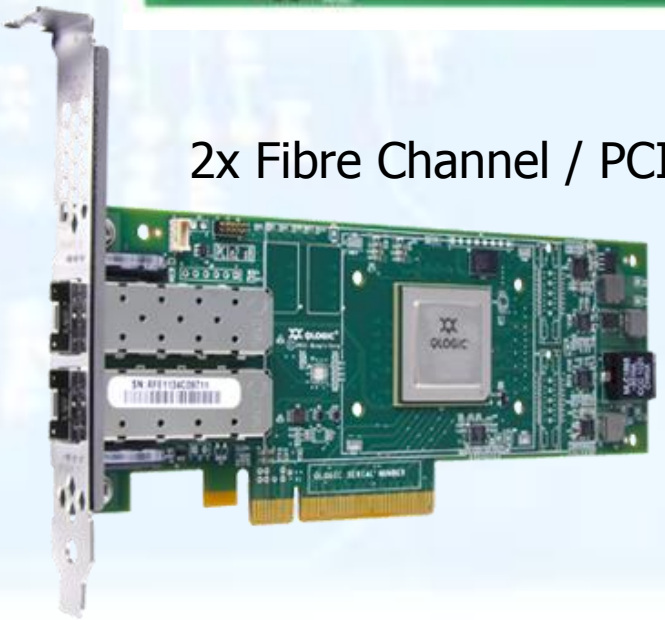



1x MIL STD-1553B / PCI Express

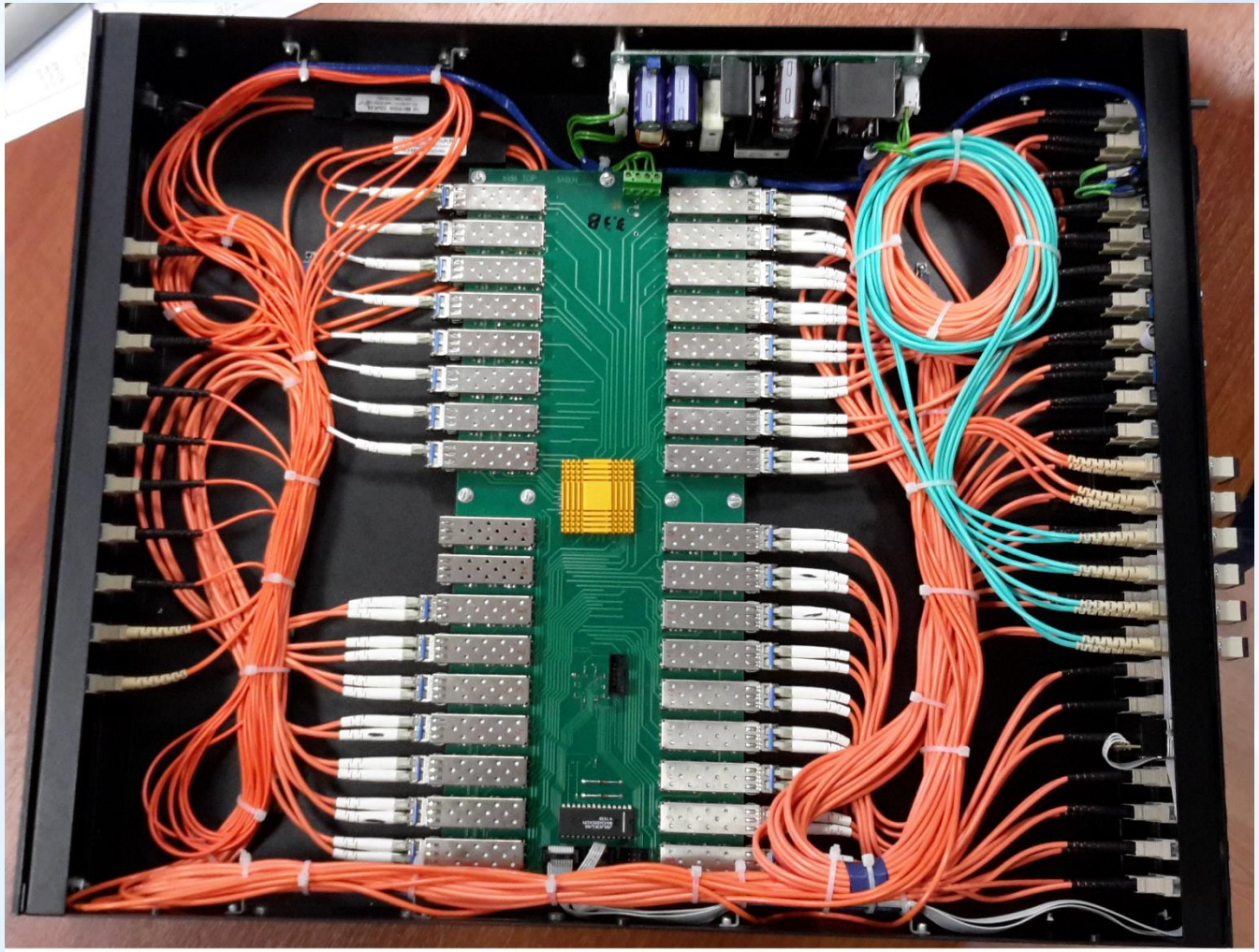


16T16R
ARINC 429 / PCI

2x Fibre Channel / PCI Express



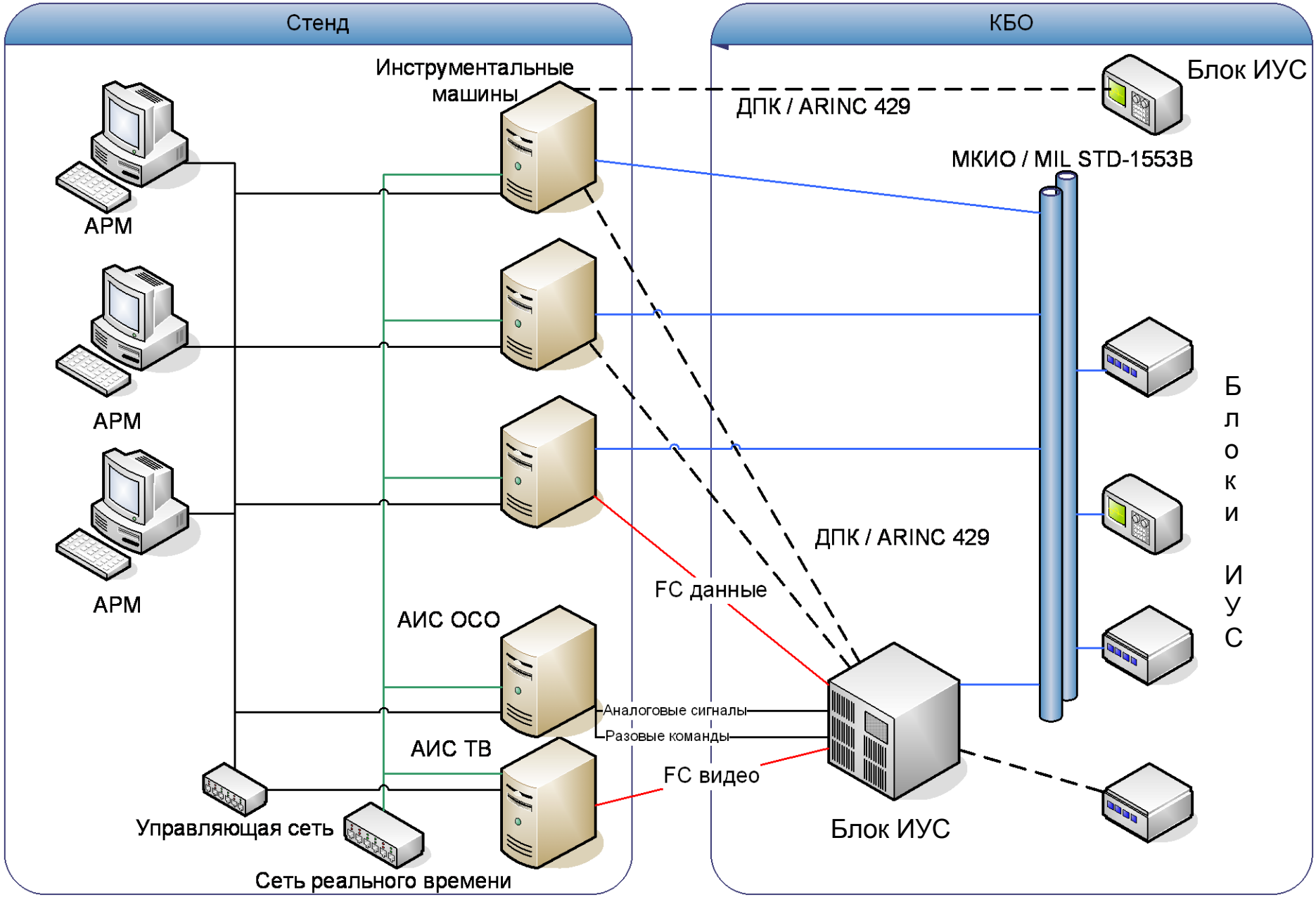
# Аппаратная база стенда



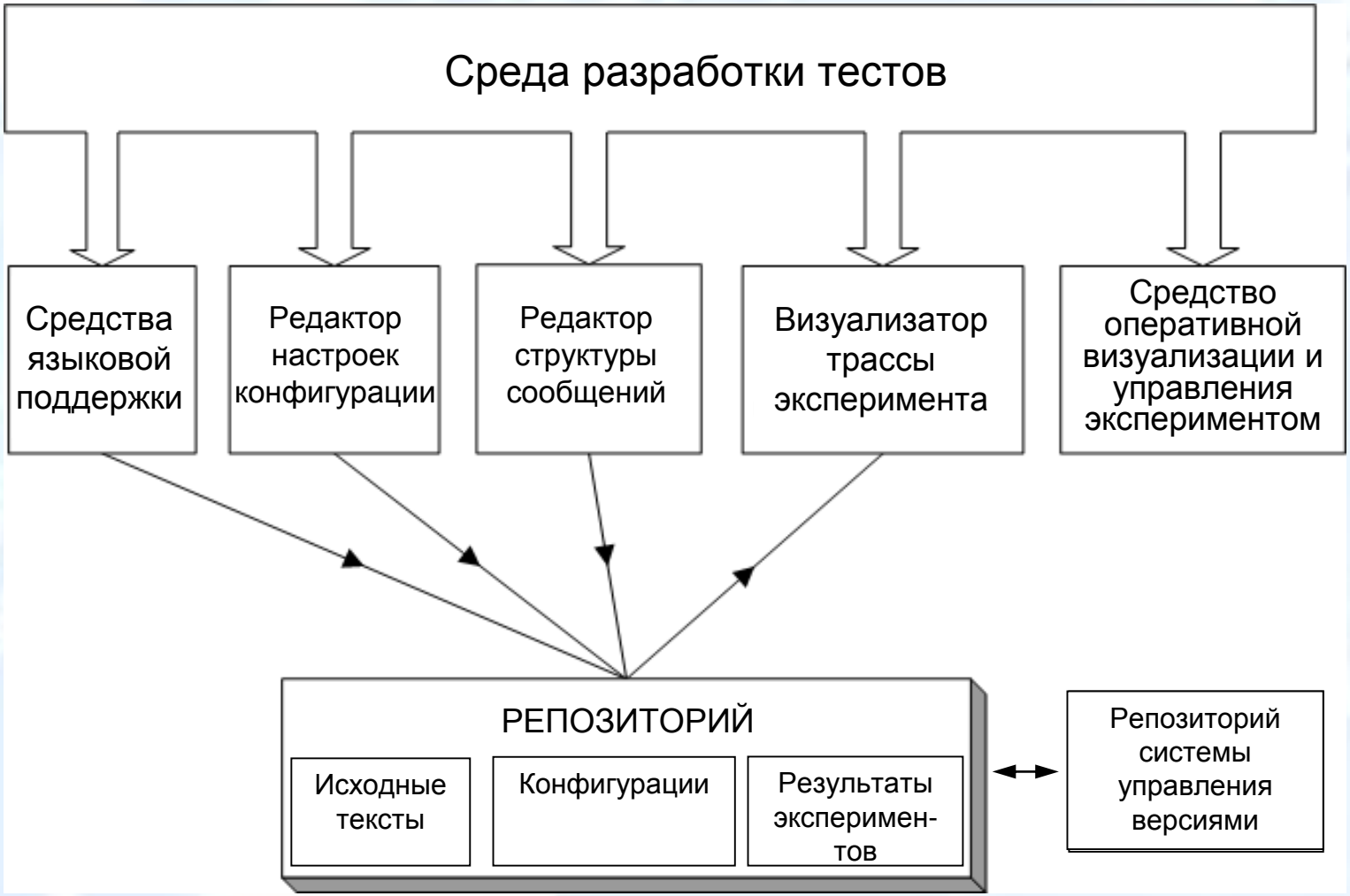
- Программируемый коммутатор Fibre Channel
- Оптический сигнал преобразуется в электрический
- Коммутация электрического сигнала при помощи ПЛИС

# Программные средства функционального тестирования в составе стенда

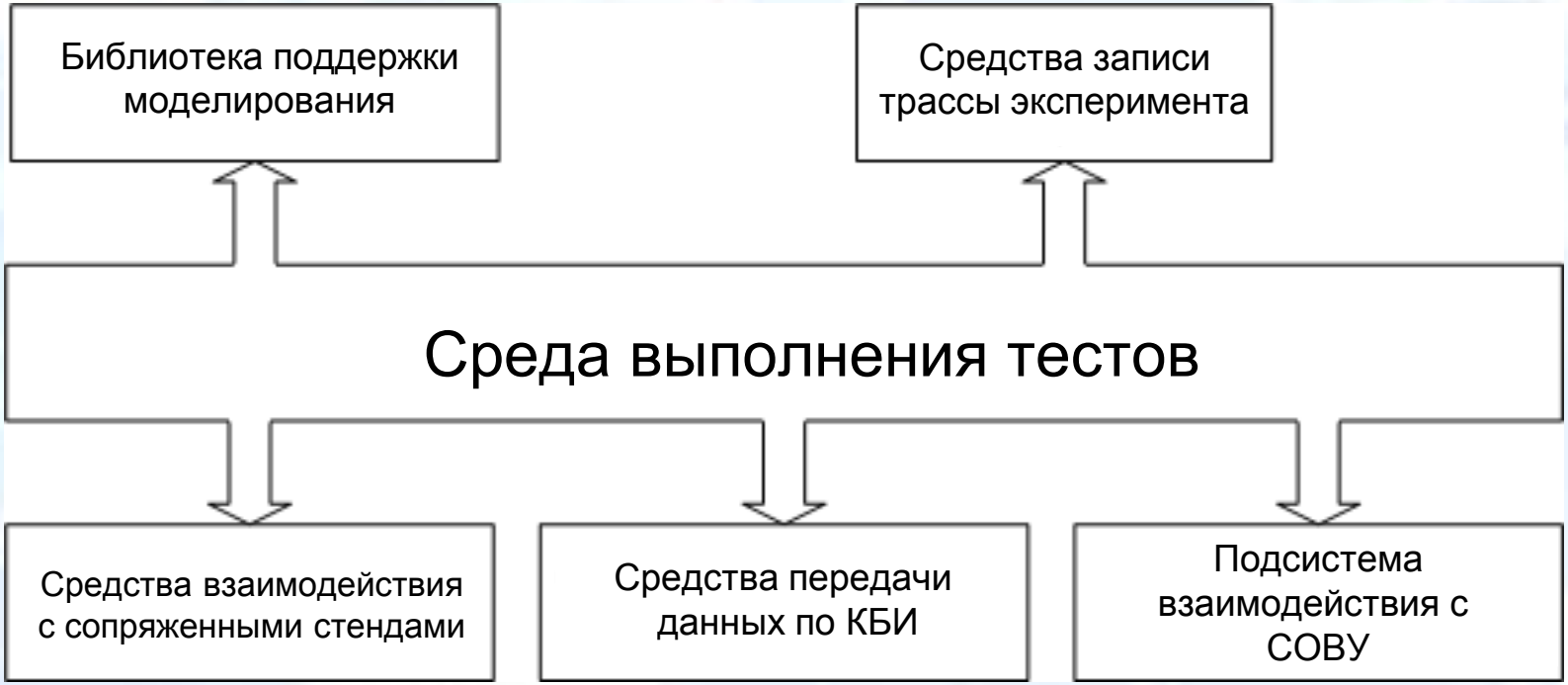
# Архитектура стенда тестирования ИУС



# Средства АРМ



# Средства ИМ


Библиотека поддержки моделирования
Средства записи трассы эксперимента
Среда выполнения тестов
Средства взаимодействия с сопряженными стендами
Средства передачи данных по КБИ
Подсистема взаимодействия с СОВУ

# Язык описания тестов (ЯОТ)

- Поддерживает тестирование ИУС через каналы бортовых интерфейсов (MIL-STD-1553B, ARINC 429, FC-AE-ASM, CAN, AFDX и т.п.) с использование следующих основных элементов:
  - Сообщения и слова КБИ
  - Битовые поля сообщений/слов
- Предоставляет операторы для:
  - Определения структуры слов/сообщений (состав битовых полей, их имена и расположение)
  - Задания и обработки значений битовых полей
  - Автоматической проверки тестируемых условий
  - Взаимодействия с пользователем
  - Задания структуры тестового сценария

# Основные понятия ЯОТ

- Тестовые компоненты (ТСК)
- Интерфейсы
  - Обеспечивают подключение к каналам БИ
- Сообщения бортовых интерфейсов
- Битовые поля сообщений
- Тестовые случаи (test case)
- Тестовые шаги (test step)

# Тестовые компоненты

*Тестовый компонент (ТСК)* – это средство компоновки объектов ФТ в составе стенда. ТСК определяет состав и последовательность выполнения тестов.

С каждым ТСК связаны:

- *имя (определяет тип ТСК)*
- *описание сообщений КБИ с указанием их структуры*
- *описание тестов*

# Структура ТСК

*Заголовок ТСК* определяет состав интерфейсов, состав и структуру сообщений КБИ, состав тестов и их привязку к требованиям

*Тело ТСК* определяет состав и логику выполнения тестовых случаев и тестовых шагов

# Интерфейсы ТСК

*Интерфейс* – это объект средств ФТ, предоставляющий программный интерфейс для взаимодействия ТСК с натурными блоками ИУС, а также друг с другом

- Интерфейсы обеспечивают доступ ТСК к адаптерам КБИ
- Через интерфейсы обеспечивается передача и прием сообщений

# Сообщение MIL STD-1553B

*Сообщение MIL STD-1553B* – это объект средств ФТ, соответствующий сообщению бортового интерфейса MIL STD-1553B.

С каждым сообщением связаны:

- *имя*
- *интерфейс MILS*
- *адрес*
- *подадрес*

С отдельными словами в составе сообщения можно работать как с элементами массива.

(сообщения для других типов интерфейсов устроены аналогично)

# Сообщения MIL STD-1553B

**RTS_OUTPUT_MESSAGE** (<имя>,
<интерфейс MILS>,
<адрес>,
<подадрес>,
<кол-во слов в сообщении>,
<период в мс>)

**RTS_INPUT_MESSAGE** (<имя>,
<интерфейс MILS>,
<адрес>,
<подадрес>,
<кол-во слов в сообщении>,
<период в мс>)

**RTS_OUTPUT_ / RTS_INPUT_TERM_MESSAGE** (<имя>,
<интерфейс MILS>,
<подадрес>,
<кол-во слов в сообщении>)

# Битовые поля и Биты

*Битовое поле* – это объект средств ФТ, соответствующий группе последовательных битов в составе слова сообщения.

*Бит* – это битовое поле, соответствующее 1 биту.

С каждым битовым полем связаны:

- *имя*
- *позиция в сообщении (номер первого слова, номер начального бита, кол-во битов)*
- *признак использования знакового или беззнакового поля*
- *коэффициент масштабирования (ЦМР)*

С битовым полем можно работать как с числовой переменной.

# Битовые поля и биты

**RTS_BITFIELD** (<имя поля>,
<имя сообщения>[<номер слова>],
<начальный бит>,
<кол-во битов>
)

**RTS_BOOLEAN** (<имя поля>,
<имя сообщения>[<номер слова>],
<позиция бита>)

# Задание иерархического идентификатора теста

*Идентификатор теста задается в формате:*

**<имя 1>.<имя 2>....<имя n>**

*Например:* «**Су35.СОЛС.Наземный_контроль.17**»

**RTS_TESTNAME** (“<имя теста>” )

**RTS_TESTNUM** (<номер теста>)

**RTS_TEST_SHOW** ()

**RTS_TEST_PASSED** ()

**RTS_TEST_FAILED** ()

# Задание соответствия требований идентификаторам тестов

**RTS_DOCUMENT**(“<requirement specification document name>”,

**RTS_REQUIREMENT**(<requirement identifier>, “<requirement name>”),

**RTS_REQUIREMENT**(<requirement identifier>, “<requirement name>”),

…

)

**RTS_TESTSPEC** (“<test name (or name template)>“,

<requirement identifier>, <requirement identifier>, ...)

# Задание соответствия требований идентификаторам тестов

Пример:

Заголовок

**RTS_DOCUMENT**( “Board computer system initialization”,

**RTS_REQUIREMENT**(BCS.INIT.START, “Starting up MIL STD-1553B subscribers”),

**RTS_REQUIREMENT**(BCS.INIT.WORK, “Starting BCS operation main loop”),

)

**RTS_TESTSPEC** (“3.2.{1-4}“, BCS.INIT.START)

**RTS_TESTSPEC** (“3.2.{1-4}“, BCS.INIT.WORK, BCS.SKV.WORK)

Тело

**RTS_TESTNAME**(3.2.2)

// логика тестового шага

# Временная задержка, отправка данных

// Задержать выполнение теста на заданное время

**RTS_DELAY** (<duration (ms)>)

// Выдать сообщение в канал (для активных абонентов)

**RTS_EXECUTE** (<Message/word name>)

# Взаимодействие с пользователем

// Отобразить значение объекта
// и занести это значение в протокол
**RTS_DISPLAY** (<object name>)

// Отобразить и записать в протокол текстовое сообщение
**RTS_COMMENT (<message text>)**

// Приостановить выполнение теста
// и ждать команды пользователя на продолжение
**RTS_PAUSE** ()

// Запросить у пользователя заключение (тест пройден / не пройден)
// и завершить тест в соответствии с полученным заключением
**RTS_CONFIRM** ()

// Запросить у пользователя заключение и занести его в протокол
**RTS_REQUEST** ()

// Запросить у пользователя текстовый комментарий и занести его в протокол
**RTS_EXPLAIN** ()

Пользователь выполняет все действия через интерфейс СОВУ

# Тестовый шаг

*Тестовый шаг* – фрагмент теста, проверяющий элементарную функцию тестируемого устройства.

Тестовый шаг может проверять правильность приема/передачи отдельного сообщения или слова, или отдельного битового поля в составе сообщения.

Типичная структура тестового шага:

- *Установка тестовых условий (входных значений)*
- *Ожидание результата*
- *Проверка ожидаемого результата (+ нумерация)*
  - Операторы **if** и **RTS_WHEN**

Тело:

RTS_TESTNAME (“<name>”),

RTS_TESTNUM (<multi-level number>)  // автоинкремент

# Оператор RTS_WHEN

```
RTS_WHEN ( “W46A185_13 == 0” , 15) {  // ожидание 15 мс
  RTS_THEN():
        RTS_COMMENT (
            “ERROR Circuit- Command Port (W46A185_13 = 0)”
          );
        RTS_TEST_FAILED();
    break;
  RTS_OTHERWIZE():
        RTS_TEST_PASSED();
    break;
}
```

# Пример тестового шага

```
// StateValue – битовое поле в RTS_OUTPUT_TERM_MESSAGE()
// ControlWord – битовое поле в RTS_INPUT_TERM_MESSAGE ()
//  ---------------------------------------------------------------------------
RTS_TESTNUM( 2.1 );

// Установка значения отправляемых данных
StateValue = 0.0;
// Ожидание выдачи сообщения
RTS_DELAY( 0.5 );
// Проверка ответа от тестируемого устройства
RTS_WHEN ( “ControlWord == 5”) {
    //……
}
```

# Тестовый случай

*Тестовый случай* – последовательность связанных по смыслу тестовых шагов, соответствующих отдельным разделам или отдельным требованиям *системной спецификации* на тестируемое устройство.

“Минималистический” тестовый случай может состоять из одного тестового шага.

При выполнении тестового случая увеличивается нумерация тестов на соответствующем уровне.

# Протокол тестирования

- Информация о смене (назначении) номеров тестов
- Информация о смене выполняемых тестов
- Результаты тестирования: «Тест прошел» / «Тест не прошел» с указанием номера теста
- Результаты всех команд взаимодействия с пользователем
- Комментарии с описаниями ожидаемого поведения теста и другая дополнительная информация, выдаваемая командой RTS_COMMENT
- Информация о текущих значениях объектов ФТ (в т.ч. битовых полей и битов), выдаваемая командой RTS_DISPLAY

# Работа с протоколами тестирования

Заголовок

RTS_LOG_FILE(LOG1, “log1.log”)

// Файл «log1.log» будет помещен в подкаталог

// «ПротоколыТестирования» каталога

// результатов эксперимента

Тело

RTS_LOG(LOG1)

RTS_CLEAR_LOG(LOG1)

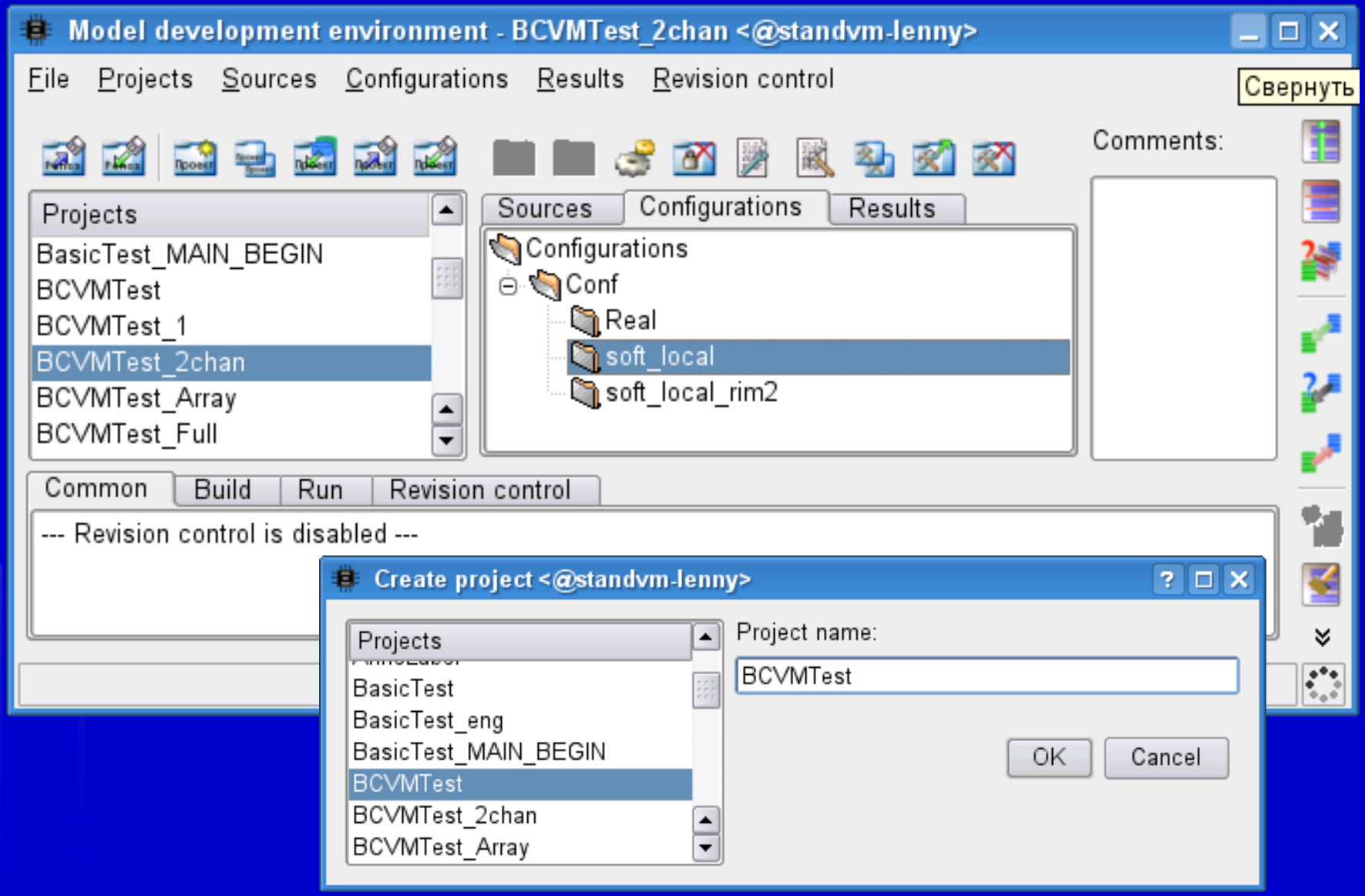
Протоколы тестирования служат основой для формирования отчетов по тестированию.

# Пример протокола тестирования

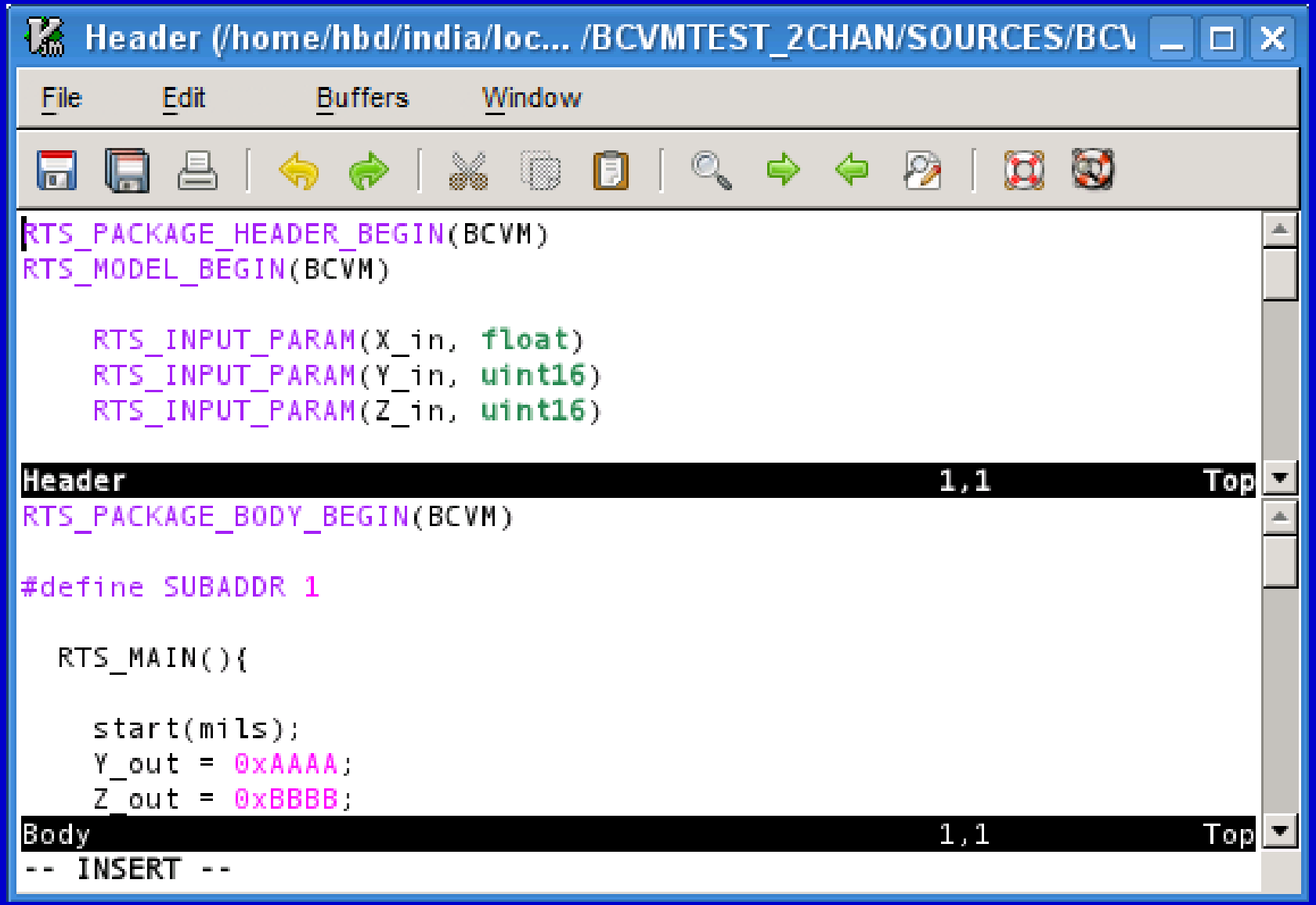
```
************************************************
* Протокол функционального тестирования LOG1
* Собран компонентом tsk (Stand.tsk)
* Тип компонента: TCK_Stand::TCK::TCK
* Имя эксперимента: T1
* Дата эксперимента: Втр Май 30 09:46:25 MSD 2006
* Продолжительность эксперимента: 30с 000мс
* Эксперимент запущен пользователем: mike
************************************************

>>>> [0мс] от начала эксперимента
-------------------------------------------------
| Начало функционального тестирования БЦВМ из проекта DemoLang
-------------------------------------------------
Тест номер 3.2.1

>>>> [9с 000мс] от начала эксперимента
Тест номер 3.2.1 прошёл

Тест номер 3.2.2

…
```

# Подготовка и выполнение тестирования

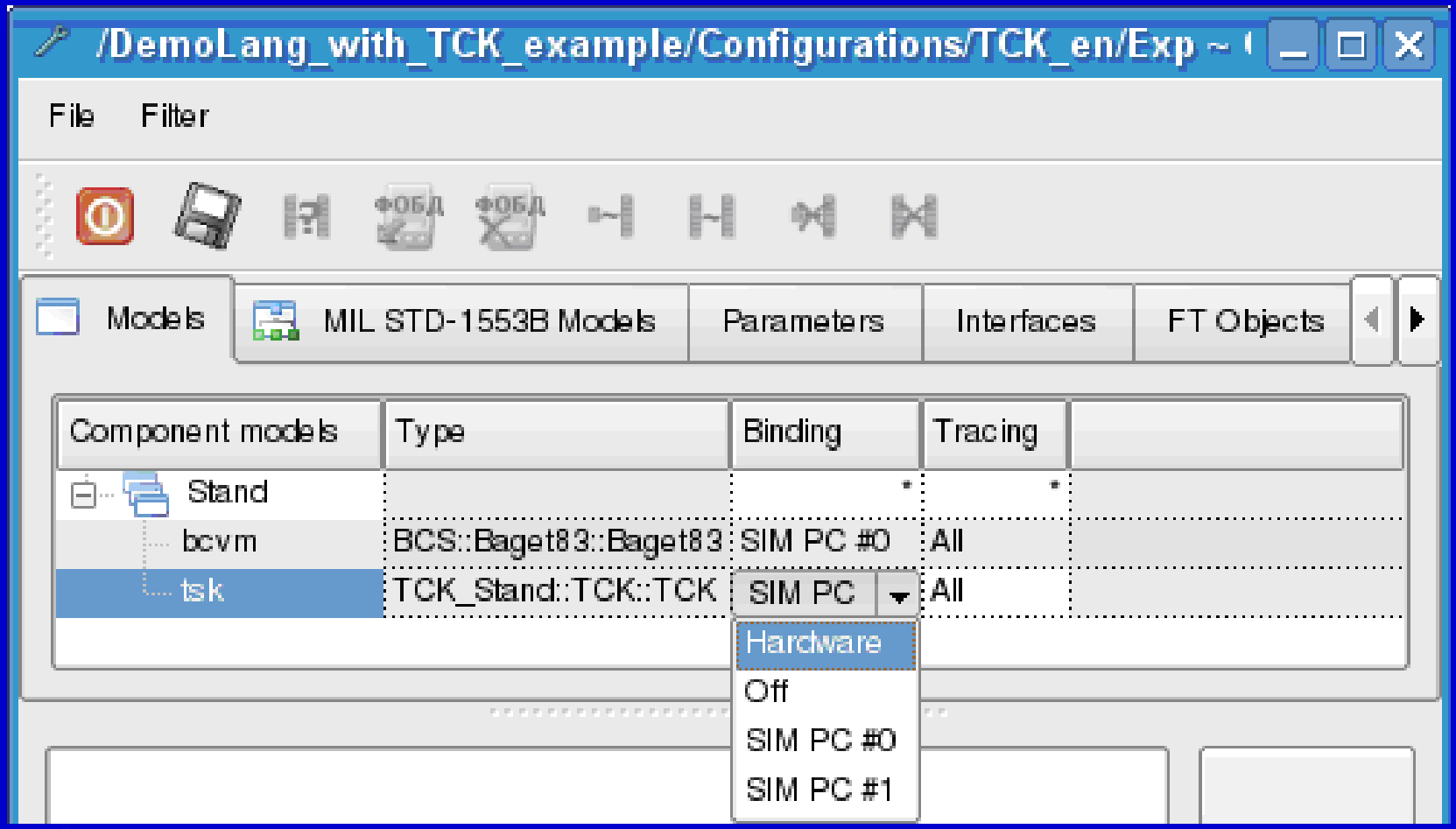
# Создание проекта



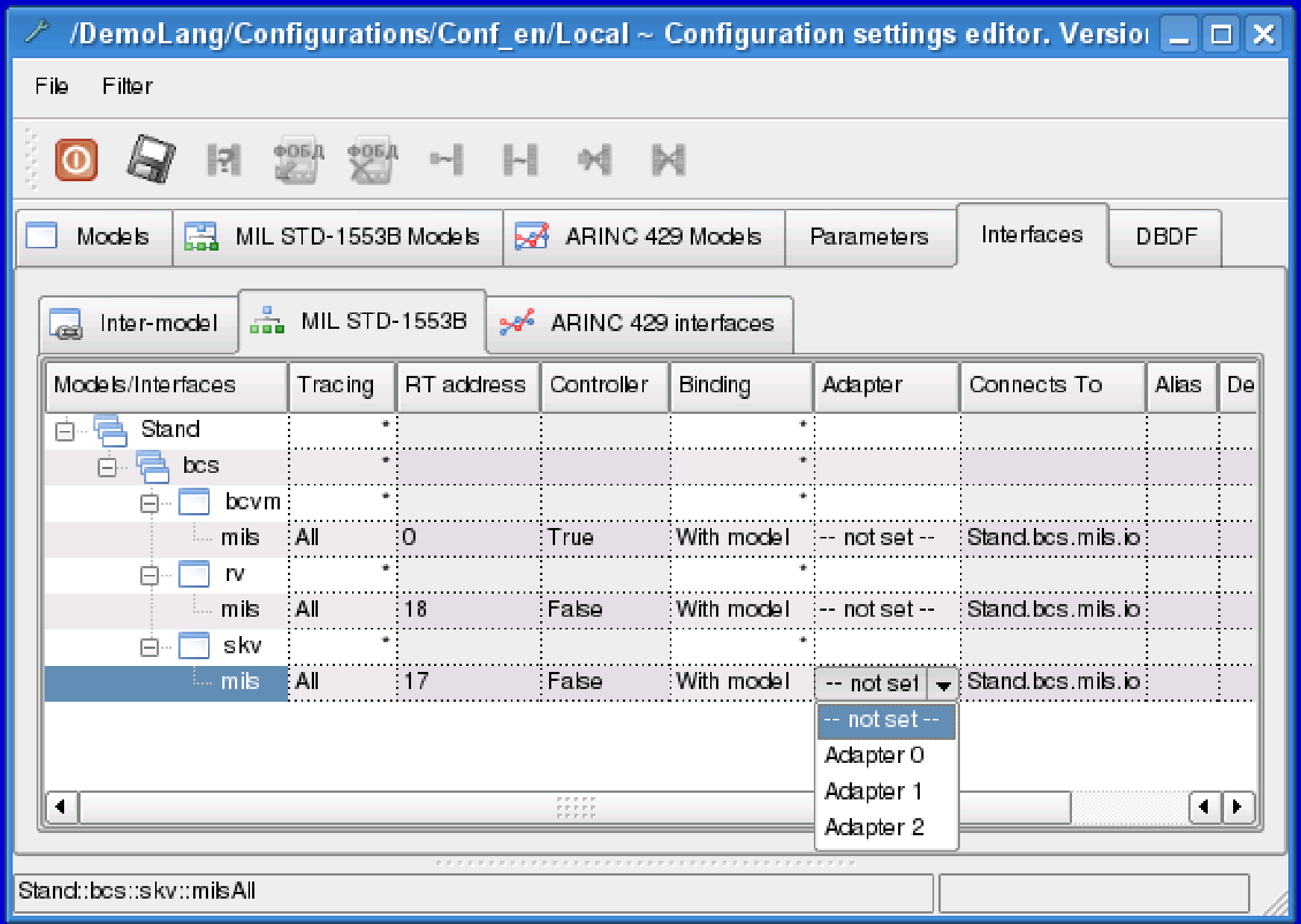
# Редактирование кода ТСК



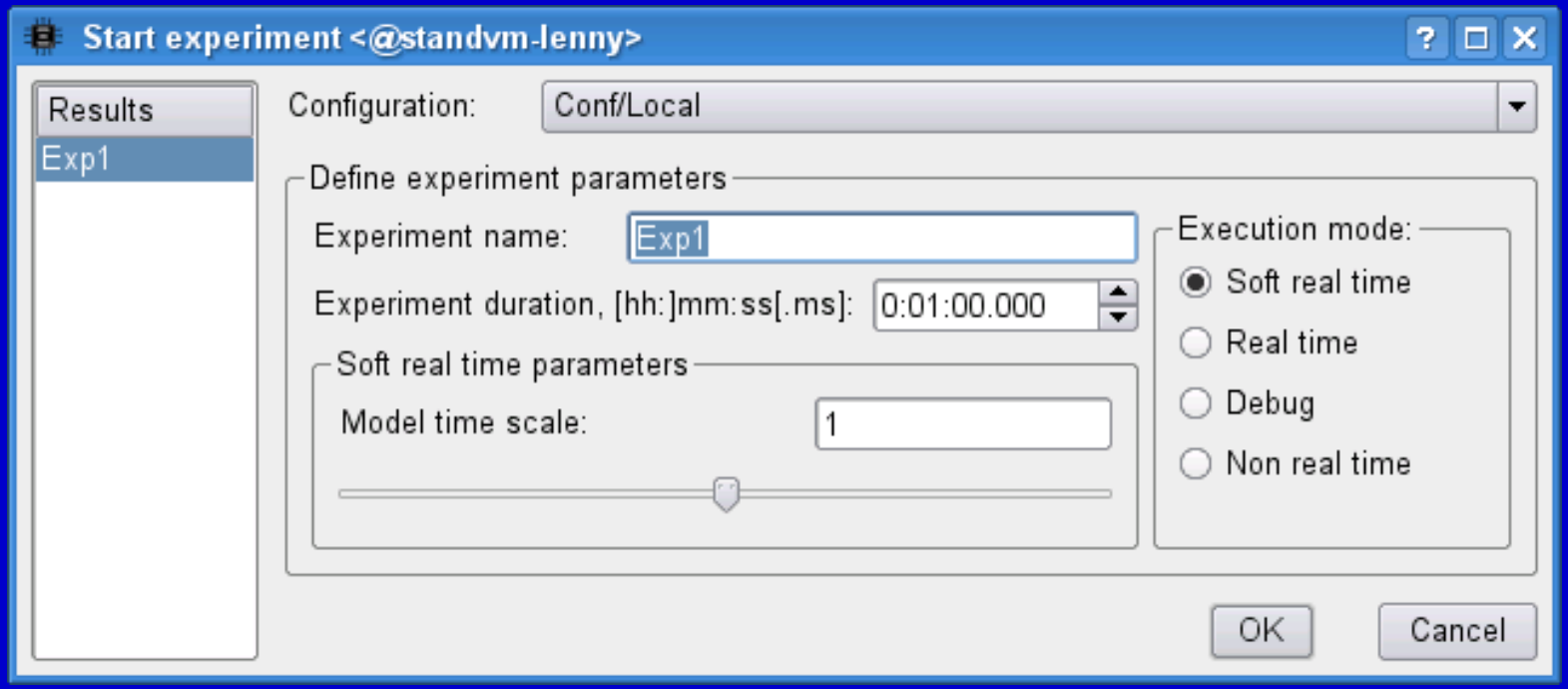
# Привязка ТСК к инструментальным машинам



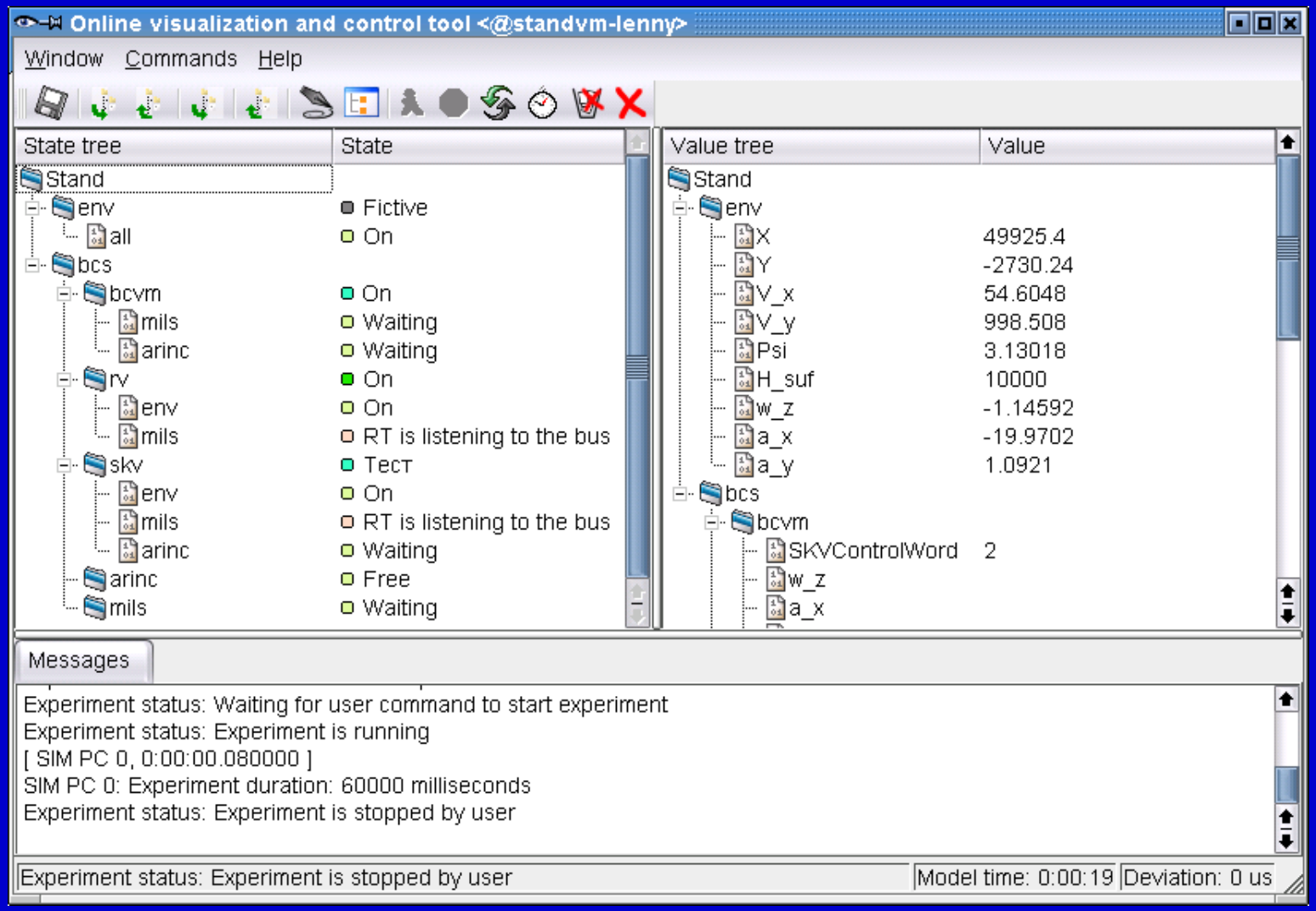
# Привязка интерфейсов к адаптерам КБИ



# Параметры эксперимента

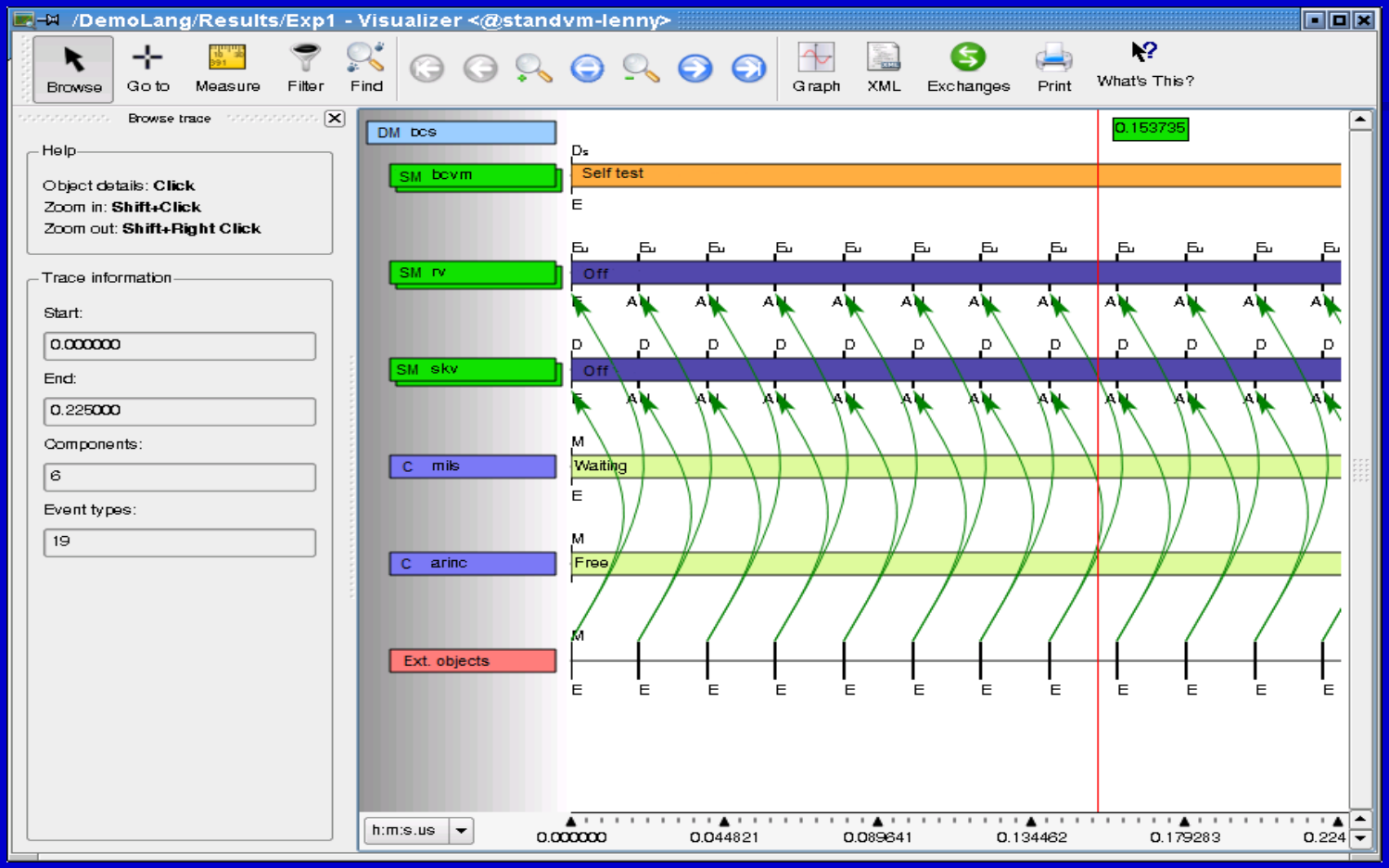


# Оперативное управление экспериментом



# Инструменты оперативного управления тестированием

# Обработка результатов эксперимента




- Формирование отчётов по результатам тестирования
- Анализ временной диаграммы событий эксперимента

# Перспективы

- Автоматизация проверки выполнения расписаний обмена
- Автоматизация тестирования индикаторов
  - интеграция с «блоком нажатия кнопок»
  - перехват изображения
  - сравнение с эталоном
- Автоматизация генерации тестов по описанию требований
- Интеграция тестов с отладочными средствами БЦВМ
- Новые внедрения
  - существующие: Су-35, Т-50/ПМИ, МС-21

# Спасибо за внимание!