Московский физико-технический институт
(государственный университет)
Факультет радиотехники и кибернетики
Кафедра информатики и вычислительной техники
Магистерскся диссертация

Портирование стандартной библиотеки языка C uclibc-ng в защищённый режим архитектуры Эльбрус

Научный руководитель: д.т.н. Семенихин С. В.
Научный консультант: Кравцунов Е.М.
Студент: Алёхин А. И., ФРТК 213 группа

Москва 2018

Защищённый режим

Защищённый режим архитектуры Эльбрус — специальный режим исполнения программ, который обеспечивает защиту контекста программных модулей согласно правилам языка программирования

Основные термины:

- Тэги — специальные биты, добавляемые к данным и определяющие тип этих данных.

Два вида тэгов:

Внешние тэги — хранятся отдельно от данных по 2 бита тэгов для каждого слова (32 бита)

Внутренние тэги — хранятся внутри данных, служат для типизации данных, имеющих одинаковые внешние тэги

Защищённый режим

Дескриптор памяти – специальный тип указателя, использующийся для обращения к области памяти в защищённом режиме:

tag.hi (4 b)	size (32 b)			curptr (32 b)
tag.lo (4 b)	itag (2 b)	rw (2 b)	psl (28 b)	base (32 b)

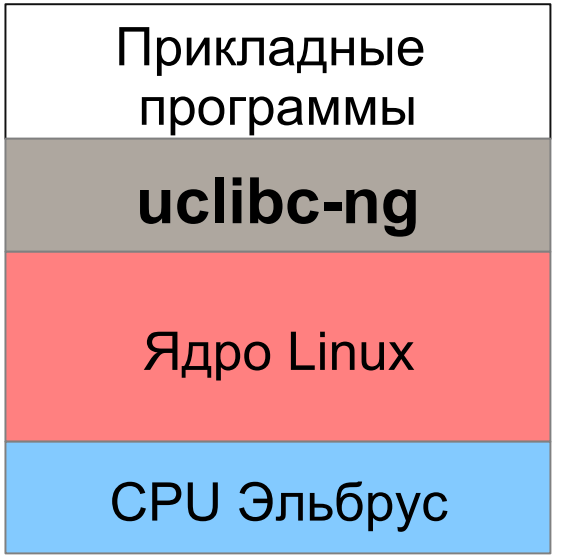
tag.hi, tag.lo — внешние тэги
base — адрес начала участка выделенной памяти
size — размер участка выделенной памяти
curptr — текущая позиции указателя на участок памяти
psl — уровень процедуры в стеке
itag — внутренние тэги
rw — права доступа

Дескриптор функции — специальный тип указателя, использующийся для вызова функции в защищённом режиме:

tag (4 b)	itag (1 b)	xx	address (60 b)

tag — внешние тэги
address — адрес функции
itag — внутренние тэги

Библиотека uclibc-ng



Uclibc-ng — стандартная библиотека языка C, обеспечивающая взаимодействие прикладных программ с ядром ОС, включает 3 основных модуля:

- libc.so (основная библиотека)
- ld.so (динамический загрузчик)
- libpthread.so (библиотека многопоточности)

Основные термины:

- Статическая библиотека — библиотека, код которой включается в программу на этапе компиляции — **статическое связывание**
- Вход в ядро — специальная функция в ядре, через которую обращаются к ядру все системные вызовы, принимает и обрабатывает параметры для системных вызовов
- Динамическая библиотека — библиотека, подключаемая к программе в момент её исполнения — **динамическое связывание**
- Динамический загрузчик — специальная программа, входящая в состав uclibc-ng и осуществляющая загрузку динамических библиотек и динамическое связывание

Цель работы

Портировать в защищённый режим архитектуры Эльбрус стандартную библиотеку языка C uclibc-ng

Задачи:

- Поддержка статического связывания
- Реализация входа в ядро для защищённого режима
- Поддержка динамического связывания
- Поддержка многопоточности
- Обеспечение работы оптимизированной функции malloc
- Тестирование и отладка
- Сборка пользовательских программ в защищённом режиме

Поддержка статического связывания

- Реализация библиотечной функции **_start** на языке ассемблера

  Функция **_start** служит точкой входа в программу, отвечает за вызов основной функции **main** и передачу ей аргументов командной строки
- Реализация новых макросов **INLINE_SYSCALL** на языке ассемблера

  Макросы **INLINE_SYSCALL** обеспечивают обращение к системным вызовам из библиотеки и передачу параметров системным вызовам
- Реализация нового входа в ядро для обработки параметров системных вызовов

Поддержка статического связывания

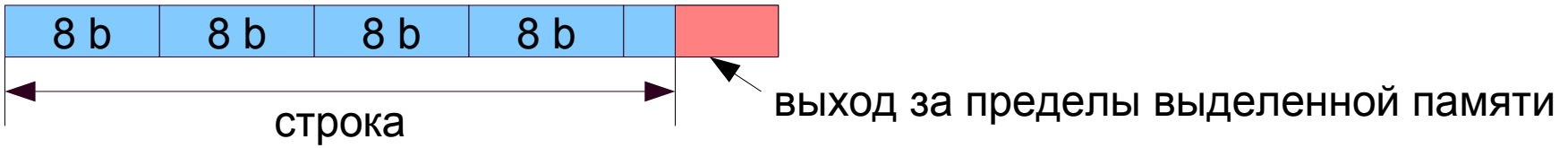
Замена недопустимых в защищённом режиме операций

- Использование целочисленных типов данных для хранения дескрипторов

unsigned int pointer → void* pointer

- Чтение за пределами выделенной памяти

Оптимизированная функции strlen, читает по 8 байт → Читать по 1 байту



- Копирование дескрипторов по частям (не по 16 байт)

В функции memcpy:

```
unsigned int* ptr_src, ptr_dst;
……
*ptr_dst = *ptr_src;
```

выровнены на 16 →

```
void** ptr_s = (void **) ptr_src;
void** ptr_d = (void **) ptr_dst;
*ptr_d = *ptr_s;
```

- Размещение дескрипторов по невыровненным на 16 байт адресам

```
union {
    char arr[SIZE];
    long int align; }
```

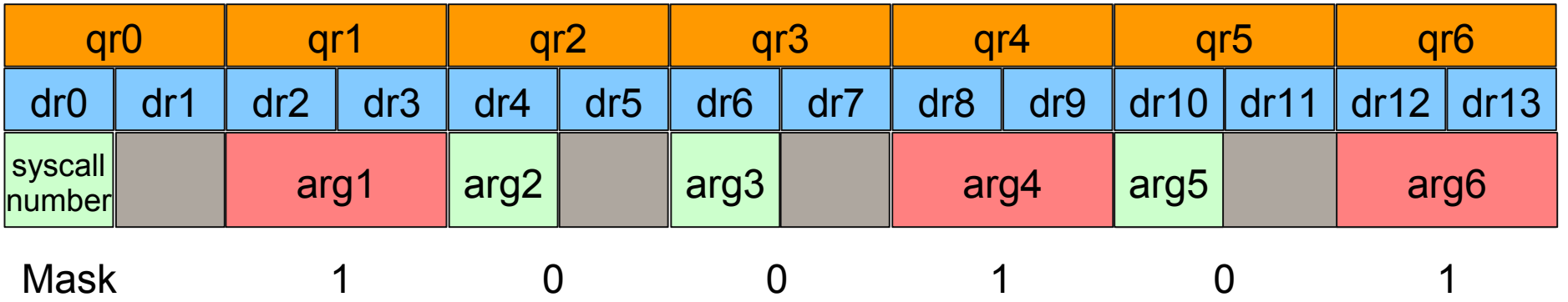
→

```
union {
    char arr[SIZE];
    void* align; }
```

Реализация входа в ядро

- **Со стороны uclibc-ng** (в макросах **INLINE_SYSCALL**):
  - Размещение каждого параметра на отдельном квадрорегистре (qr), чтобы не использовать различные макросы для разных наборов параметров
  - Расширение регистрового окна передачи параметров до 14 двойных регистров (dr)
  - Общие макросы для обращения к большинству системных вызовов
- **Со стороны ядра:**
  - Общий механизм приёма параметров для большинства системных вызовов
  - Таблица масок, обозначающая порядок и тип параметров системных вызовов

Размещение параметров на регистрах:



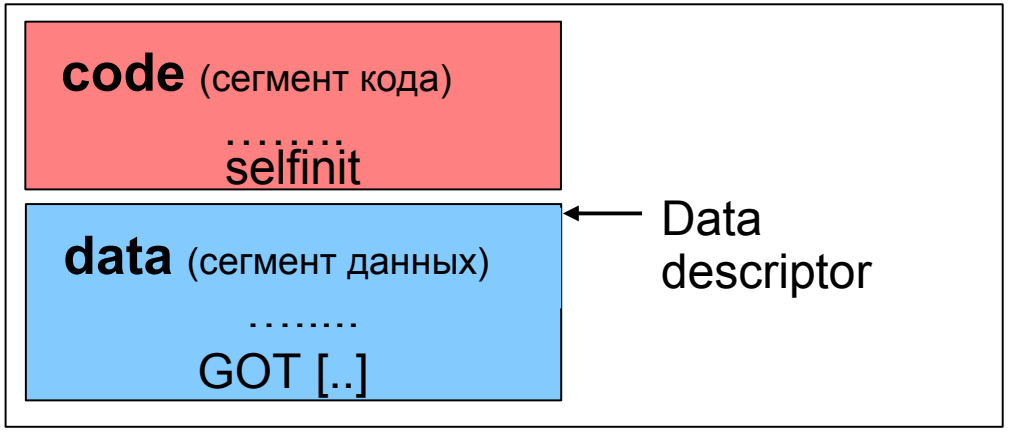
- Отдельные макросы **INLINE_SYSCALL** в uclibc-ng и обработка параметров в ядре необходима только системным вызовам, принимающим дескрипторы внутри структур и системным вызовам, возвращающим указатели

Поддержка динамического связывания загрузка

Загрузка модулей

- Обеспечение работы динамического загрузчика, выполняющего загрузку модулей в память и динамическое связывание

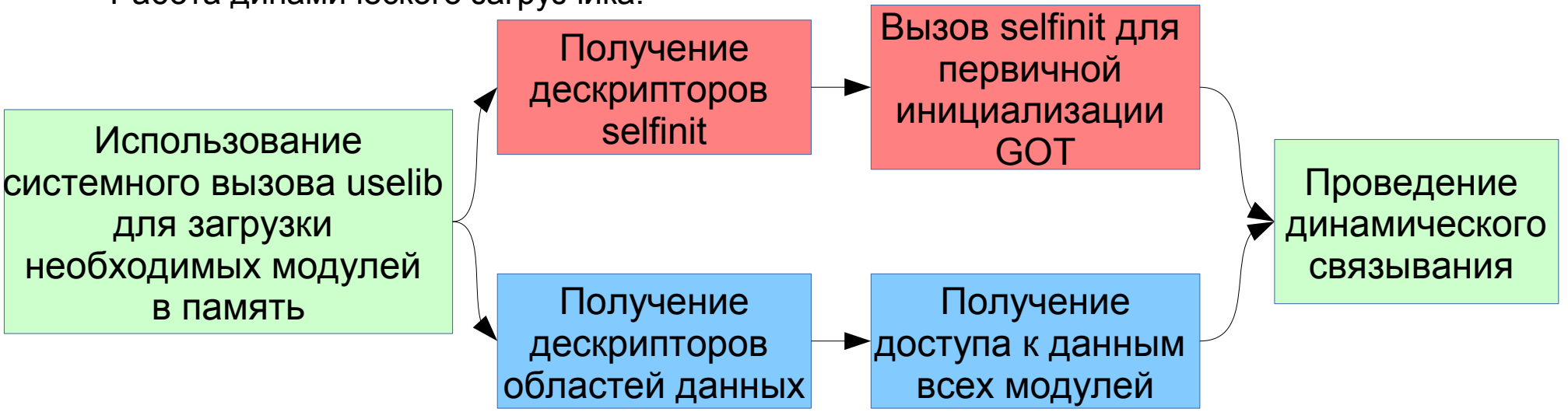
Структура загруженного в память модуля



GOT — глобальная таблица смещений, содержит дескрипторы функций и глобальных переменных

selfinit — специальная функция для инициализации GOT

Работа динамического загрузчика:



Динамическое связывание, пользовательский код

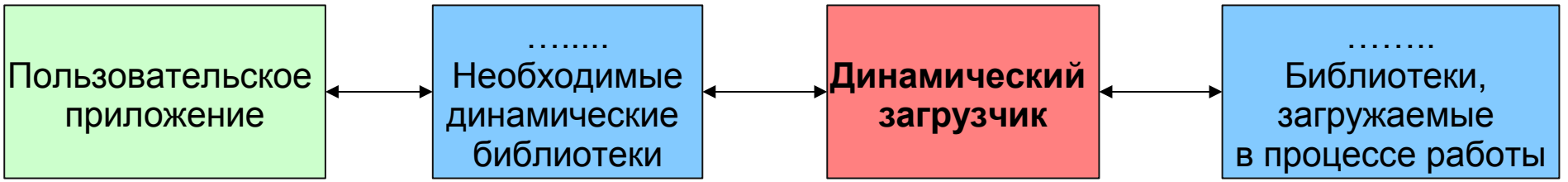
Пользовательский код

- Обеспечение работы библиотечных функций: dlopen, dlsym, dlclose
- dlopen, dlsym, dlclose требуют доступа к функциям динамического загрузчика из других модулей uclibc-ng

Решение:

- В динамическом загрузчике получить с помощью инструкции gdtoap дескриптор на область данных динамического загрузчика
- Включить динамический загрузчик в глобальный список модулей с использованием полученного дескриптора и произвести связывание

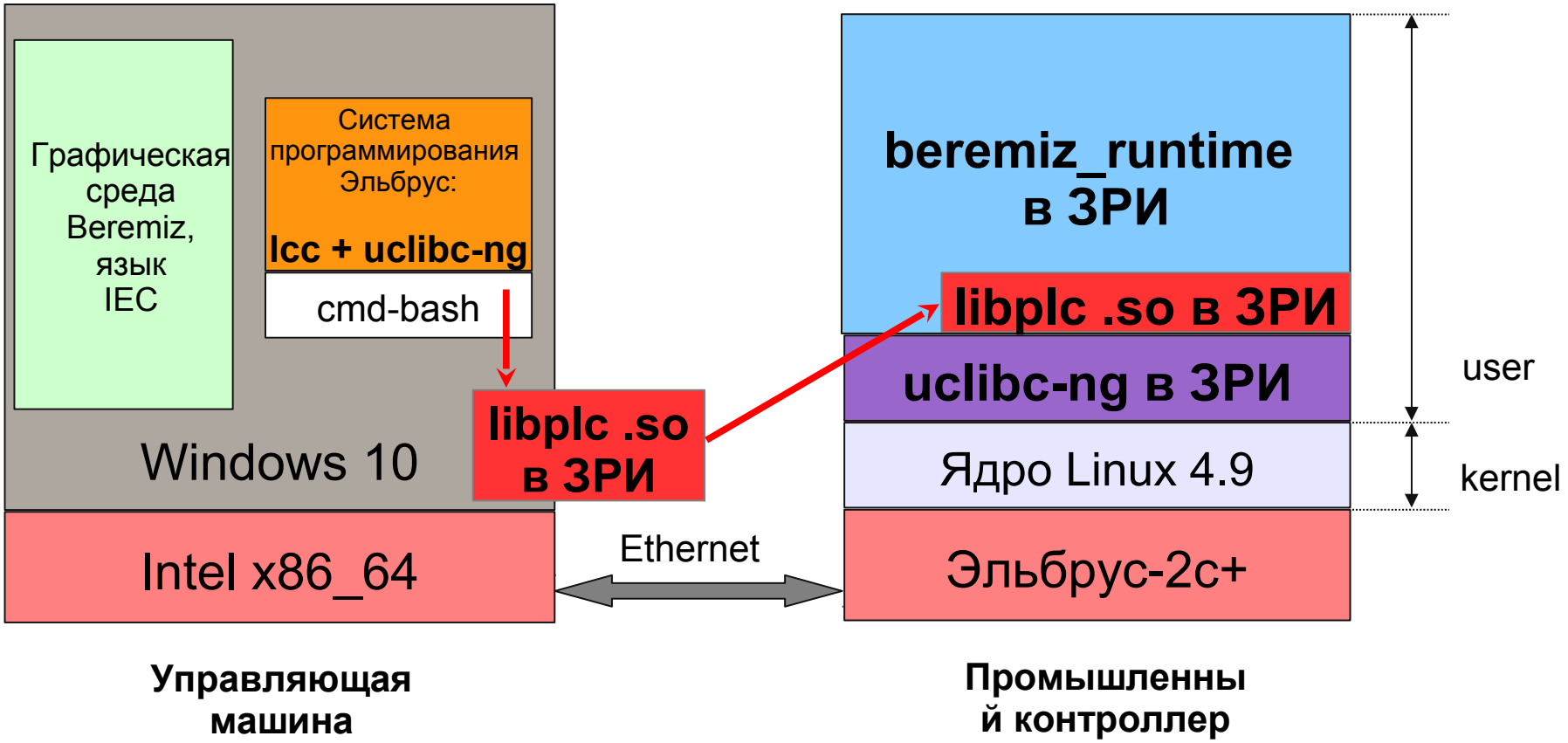
Глобальный список модулей — структура для хранения информации о загруженных модулях:



Тестирование

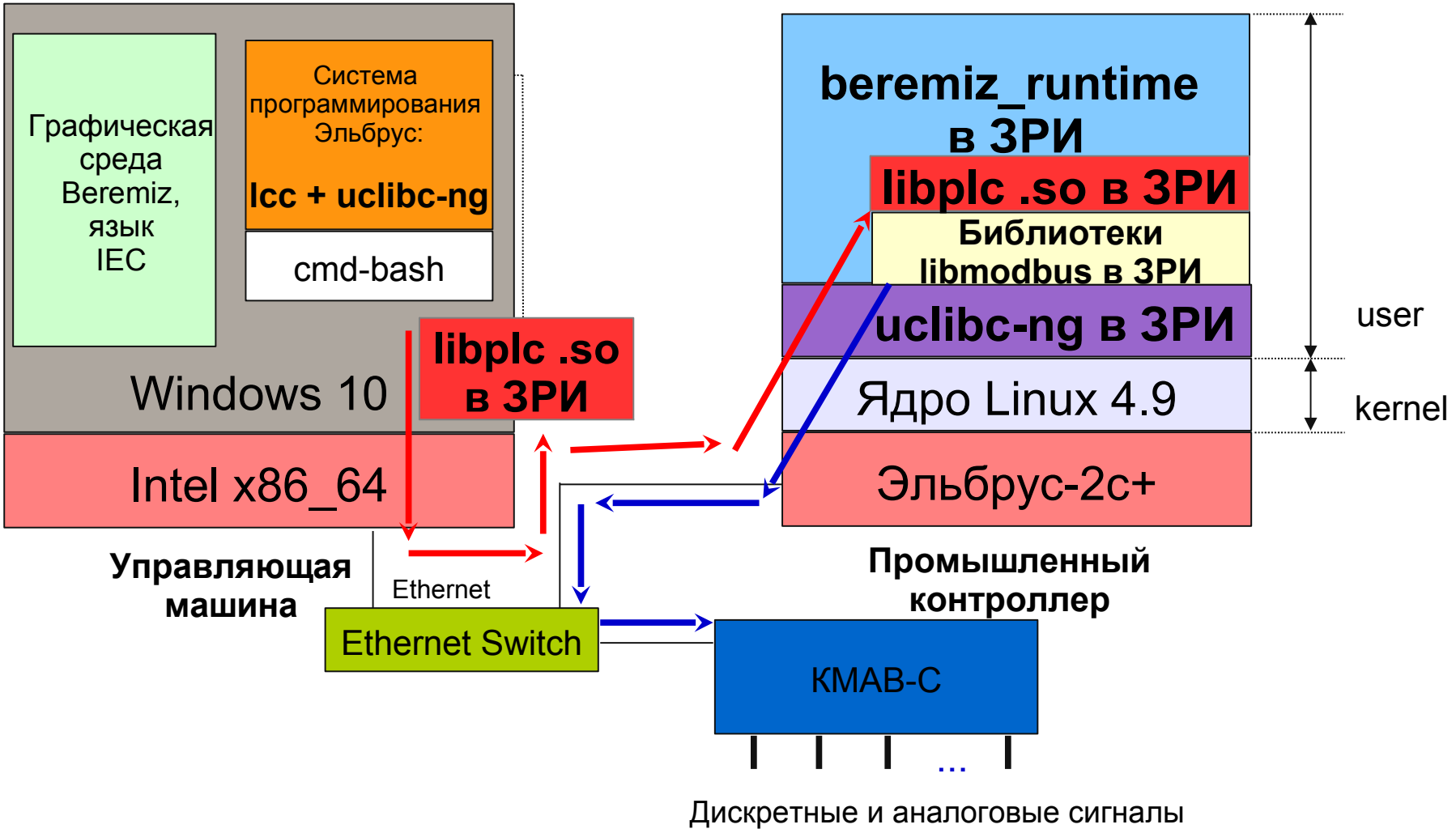
- Использование набора из 500 встроенных в uclibc-ng тестов
- Организация системы тестирования для автоматического запуска тестов после внесения правок в uclibc-gn

Пользовательские задачи в защищённом режиме



- Beremiz – графическая среда для программирования промышленных контроллеров на языке IEC и управления процессом отладки
- beremiz_runtime — специальная программа для управления промышленным контроллером

Пользовательские задачи в защищённом режиме



- Запуск проекта, использующего протокол Modbus для управления промышленными системами с помощью дискретных и аналоговых сигналов

Результаты работы

- Библиотека uclibc-ng портирована в защищённый режим архитектуры Эльбрус, реализованы:
  - поддержка статического связывания
  - новый вход в ядро для системных вызовов
  - поддержка динамического связывания
- Организована система тестирования uclibc-ng на наборе из встроенных тестов
- С uclibc-ng в защищённом режиме были собраны реальные программы для промышленной автоматизации