MIPSfpga

These materials produced in association with Imagination.
Join our University community for more resources.
community.imgtec.com/university

Обзор

- **Структура каталогов**
- **История архитектуры MIPS**
- **MIPSfpga**
  - Основы
  - Ядро и система
  - Интерфейсы
    - Системный интерфейс
    - Шина AHB-Lite
    - EJTAG

Структура каталогов

- **Каталог MIPSfpga_Fundamentals:**
  - Руководства по лабораторным работам находятся в подкаталоге **LabInstructions**
  - Дополнительные материалы находятся в каталогах **Lab*_***
  - Файлы Verilog MIPSfpga находятся в каталоге **rtl_up** архива **MIPSfpga Getting Started Guide**
  - Слайды Powerpoint находятся в каталоге **Slides**

Обзор Лабораторных работ

- **1:** Создание проекта Vivado для работы с MIPSfpga
- **2:** Программирование для MIPSfpga на C с использованием Codescape
- **3:** Программирование для MIPSfpga на ассемблере MIPS с использованием Codescape
- **4:** Программирование для MIPSfpga
- **5:** Периферийные устройства: семисегментный индикатор
- **6**: Периферийные устройства : миллисекундный таймер
- **7:** Периферийные устройства: зуммер
- **8:** Периферийные устройства : SPI LCD
- **9:** Перенос MIPSfpga на другие платы FPGA

Обзор

- Структура каталогов
- **История архитектуры MIPS**
- MIPSfpga
  - Основы
  - Ядро и система
  - Интерфейсы
    - Системный интерфейс
    - Шина AHB-Lite
    - EJTAG

История архитектуры MIPS

- Разработана Джоном Хеннесси и его коллегами в Стэнфордском университете в 1980 годах.
- Одна из первых коммерческих архитектур с **Сокращенным Набором Команд** (RISC)
- Хеннесси – соучредитель компании MIPS Computer Systems, позже переименованой в MIPS Technologies
- Используется во многих коммерческих системах, включая рабочие станции Silicon Graphics, приставки Nintendo, сервера Cisco
- Изучается во множестве университетов
- Продано более 5 миллиардов процессоров MIPS

История архитектуры MIPS

- Imagination Technologies приобрела MIPS Technologies в феврале 2013
  - Британская компания
  - Другие продукты: мобильные графические процессоры PowerVR, потребительская электроника и аудиотехника

Ядра MIPS

- **MIPS R3000, R4000, R10000**
  - 1980 и 1990 годы
  - Использовалось, например, в рабочих станциях Silicon Graphics
- **Для встроенных систем: M4K, M14K**
  - Популярная линия PIC32 микроконтроллеров фирмы Microchip основана на ядре M4K

Ядра MIPS

- **microAptiv**
  - высокоэффективное, компактное, встраиваемое ядро
  - основано на архитектуре M14K
- **interAptiv, proAptiv**
  - высокоэффективное
  - многопроцессорное, суперскалярное, многопотоковое
- **Warrior**
  - новейшая линия ядер MIPS компании Imagination
  - ряд ядер от высокопроизводительных до встраиваемых
  - P5600: T1 Байкал Электроникс

Ядра MIPS

- **microAptiv: MIPSfpga является ядром microAptiv**
  - высокоэффективное, компактное, встраиваемое ядро
  - основано на архитектуре M14K
- **interAptiv, proAptiv**
  - высокоэффективное
  - многопроцессорное, суперскалярное, многопотоковое
- **Warrior**
  - новейшая линия ядер MIPS компании Imagination
  - ряд ядер от высокопроизводительных до встраиваемых
  - P5600: T1 Байкал Электроникс

Обзор MIPSfpga

- **История архитектуры MIPS**
- **MIPSfpga**
  - Основы
  - Ядро и система
  - Интерфейсы
    - Системный интерфейс
    - Шина AHB-Lite
    - EJTAG
    - Платы FPGA

Основы MIPSfpga

Назначение

- Коммерческое процессорное ядро MIPS, реализуемое в FPGA (конфигурируемое soft-core)
- Открытое для университетов компанией Imagination Technologies

Условия использования MIPSfpga

- MIPSfpga предназначен исключительно для **академическое использование**
- **Каждый** преподаватель **должен зарегистрироваться** для получения доступа к MIPSfpga (не передавайте регистрацию другим преподавателям!)

  http://community.imgtec.com/university/university-registration
- Реализация в **кремнии запрещена**
- Во всех **публикациях** следует выражать благодарность за использование MIPSfpga
- **Следует присылать Imagination копию** вашей работы – нам интересно узнать, что вы сделали!

MIPSfpga: ядро microAptiv

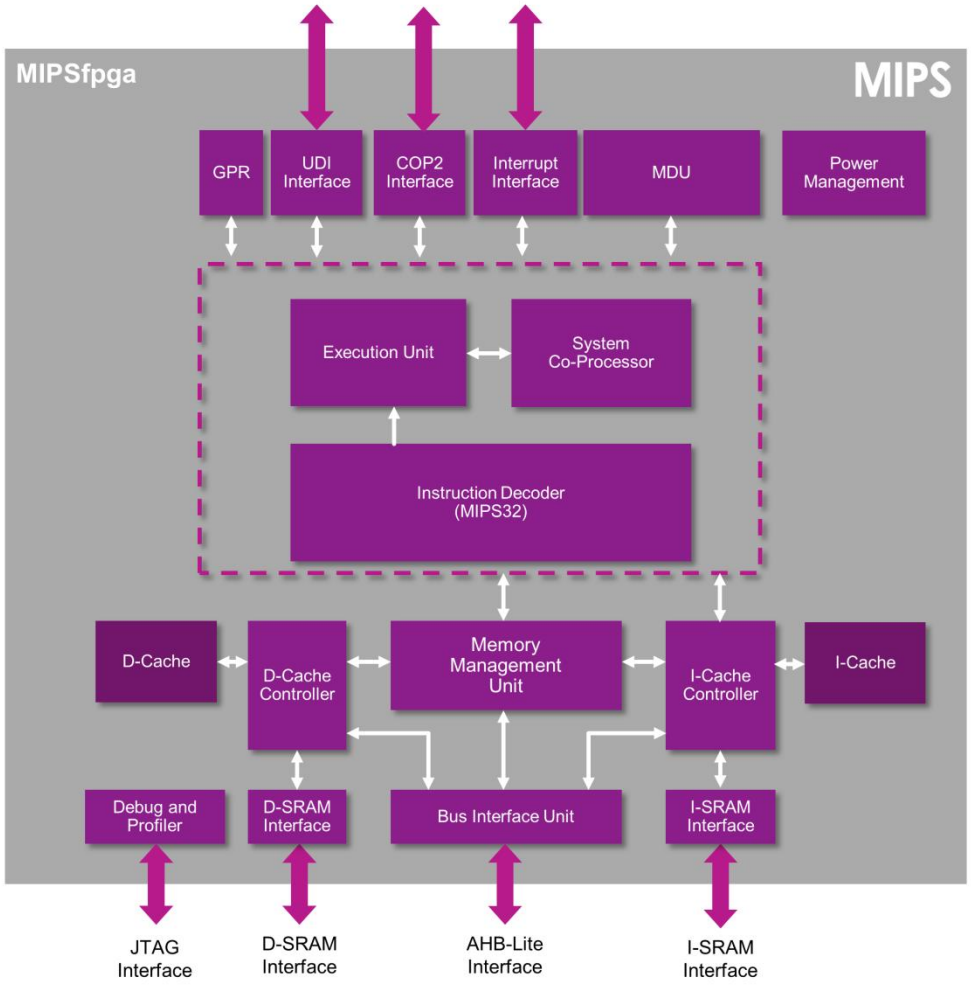
Коммерческое ядро microAptiv

- Пятистадийный конвейер
- Производительность 1.5 Dhrystone MIPS/MHz
- Блоки кэш-памяти команд и данных, двухсекционные ассоциативные, по 2KB
- MMU (блок управления памятью) с буфером ассоциативной трансляции (TLB) объемом 16 записей

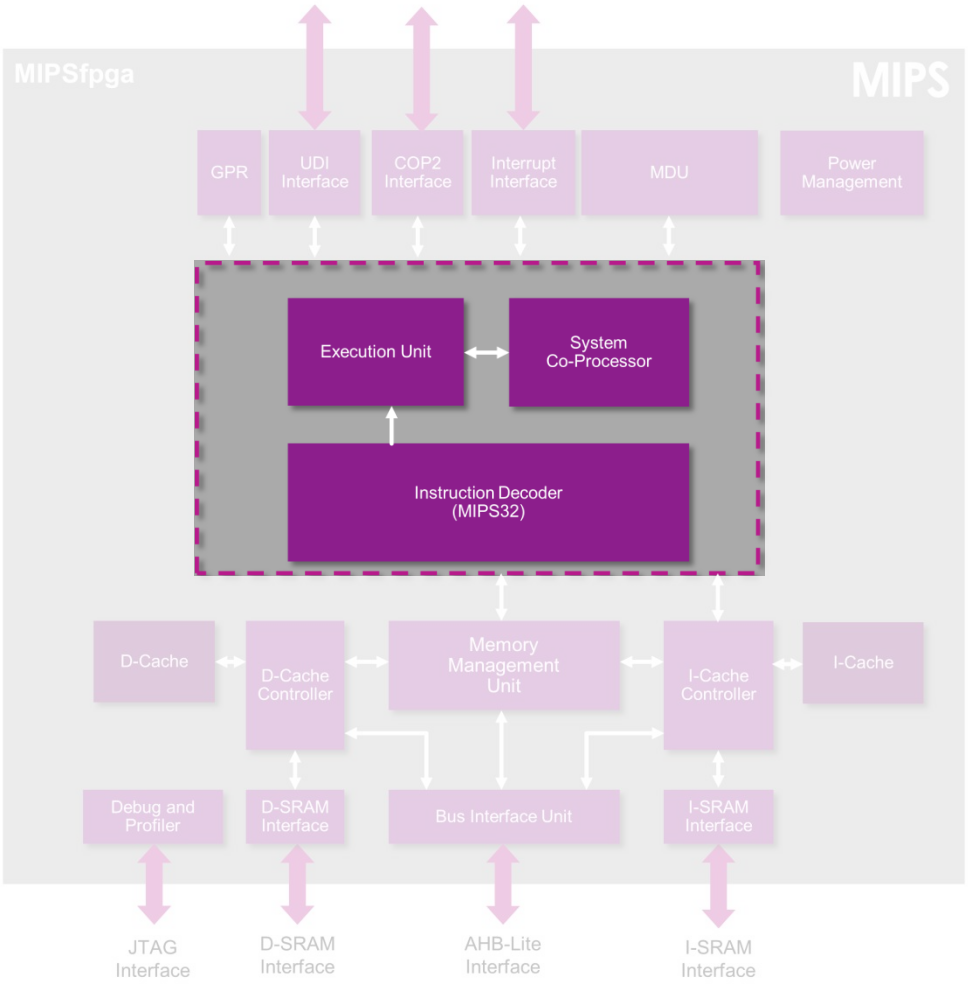
Обзор MIPSfpga

- История архитектуры MIPS
- **MIPSfpga**
  - Основы
  - **Ядро и система**
  - Интерфейсы
    - Системный интерфейс
    - Шина AHB-Lite
    - EJTAG
    - Платы FPGA

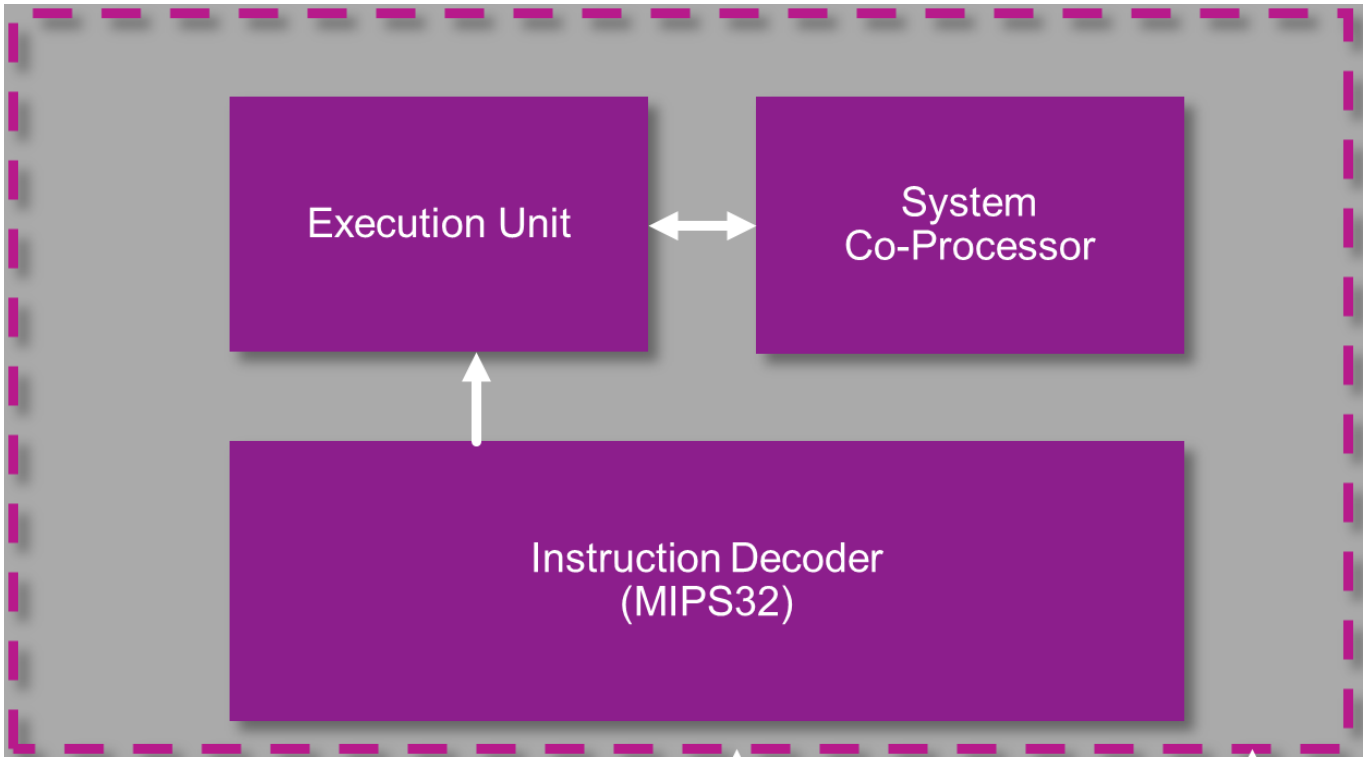
Ядро MIPSfpga



Ядро MIPSfpga

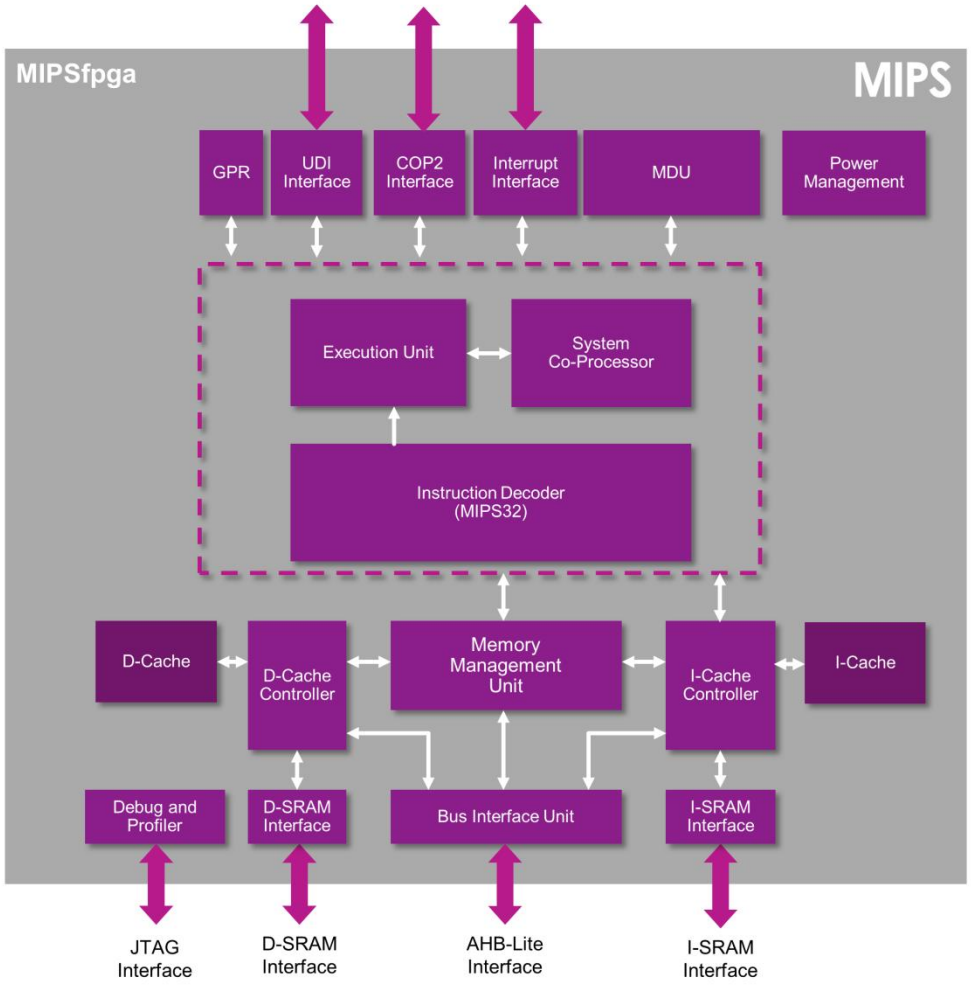


Ядро MIPSfpga



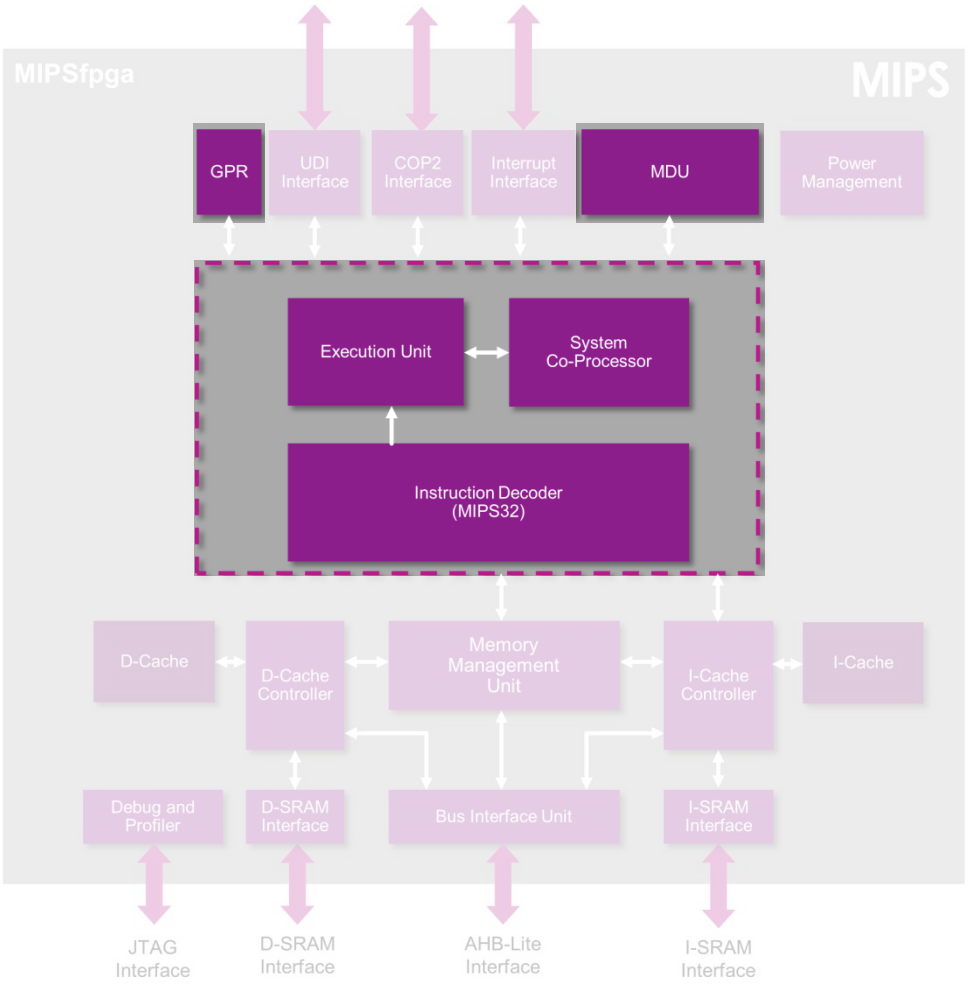
- **Исполнение операций**
- **Системные регистры и сброс (сопроцессор)**
- **Декодирование команд**

MIPSfpga: Регистры, MDU



MIPSfpga: Регистры, MDU

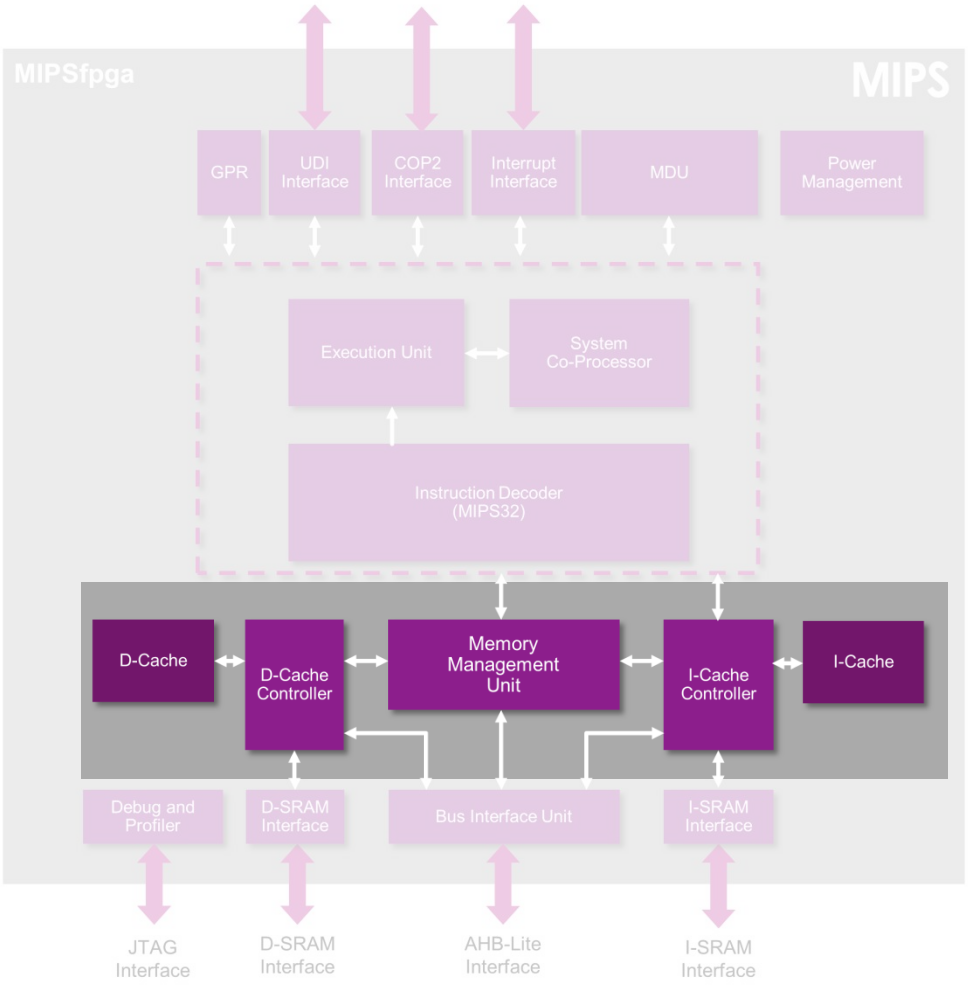
GPR: Регистры общего назначения



MDU: Блок умножения/ деления

MIPSfpga: MMU, Кэш-память

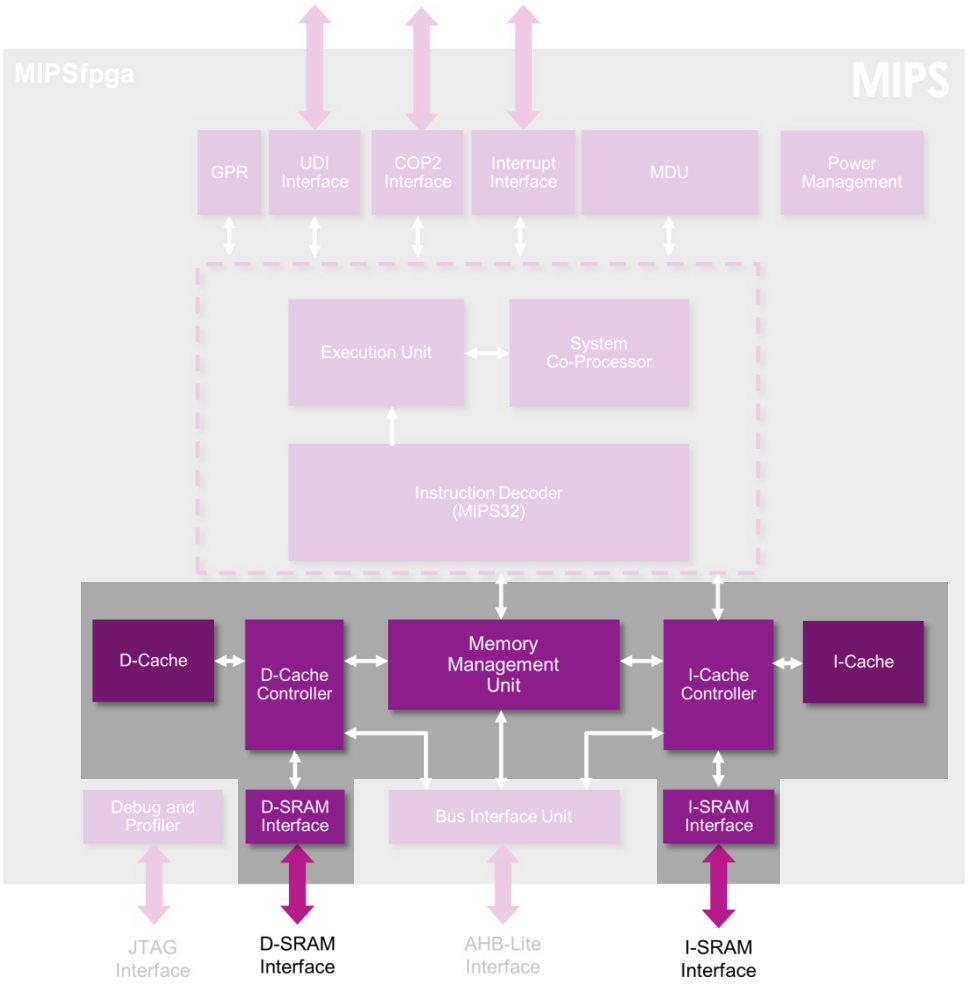
MMU:
Блок управления памятью



Кэш-память:
Команды и данные раздельно

Контроллеры кэш-памяти:
Управление кэш-памятью

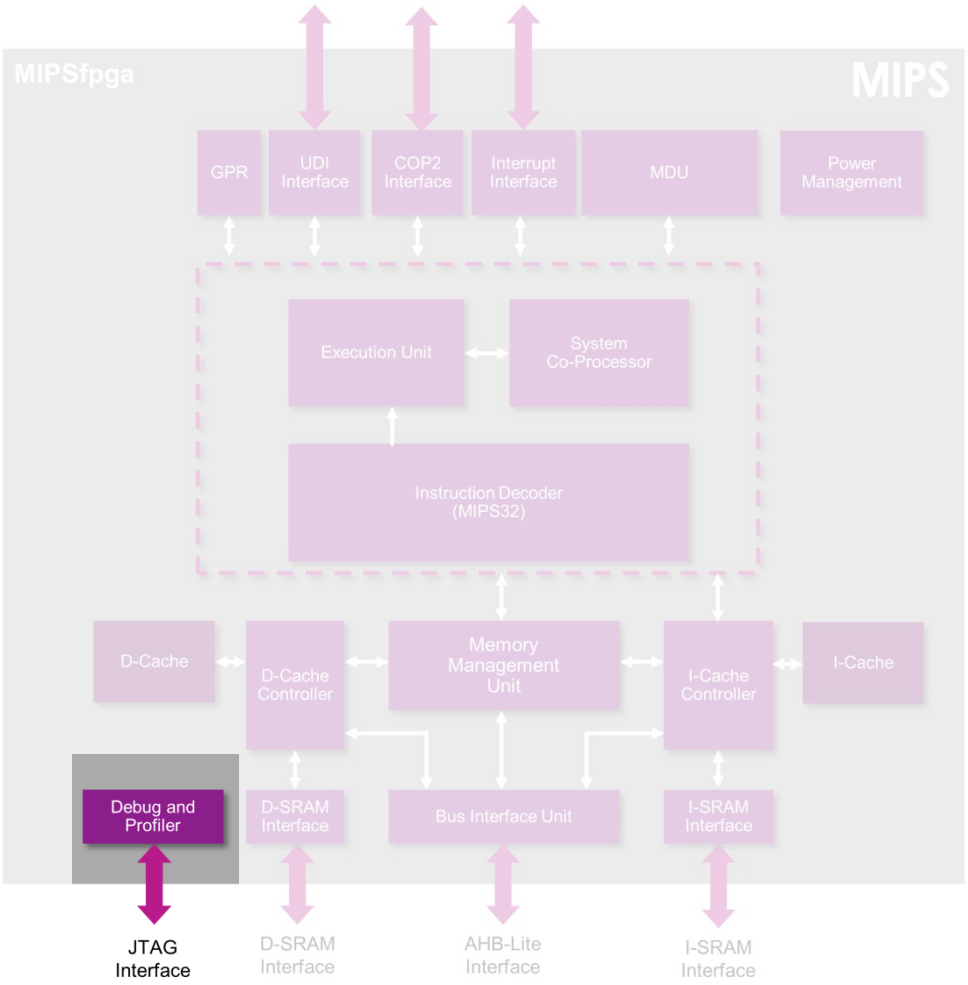
MIPSfpga: Котроллеры кэш-памяти



Контроллеры кэш-памяти: Обеспечивают взаимодейст-вие блоков кэш-памяти команд и данных со внешней памятью (сверх-оперативное ОЗУ , scratchpad RAM, SRAM)

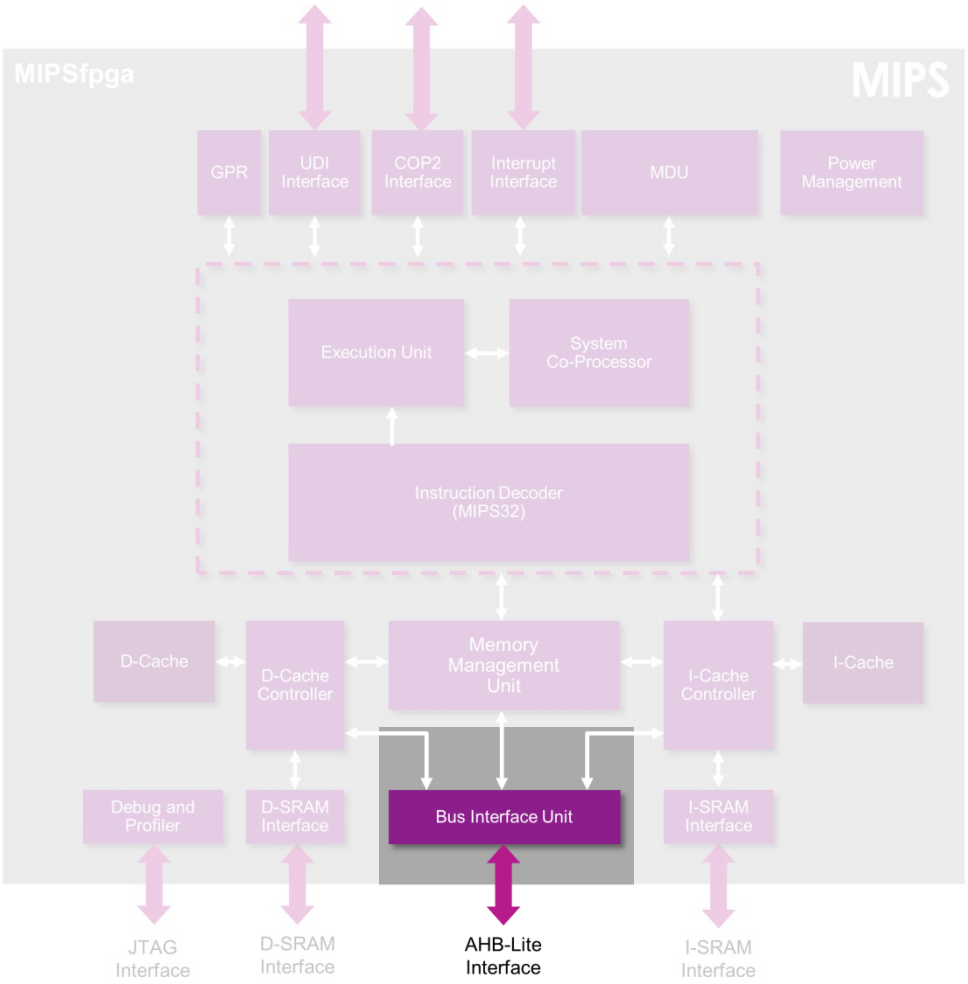
MIPSfpga: Интерфейс (E)JTAG

JTAG (EJTAG): Используется для программиро-вания и отладки ядра в реальных условиях



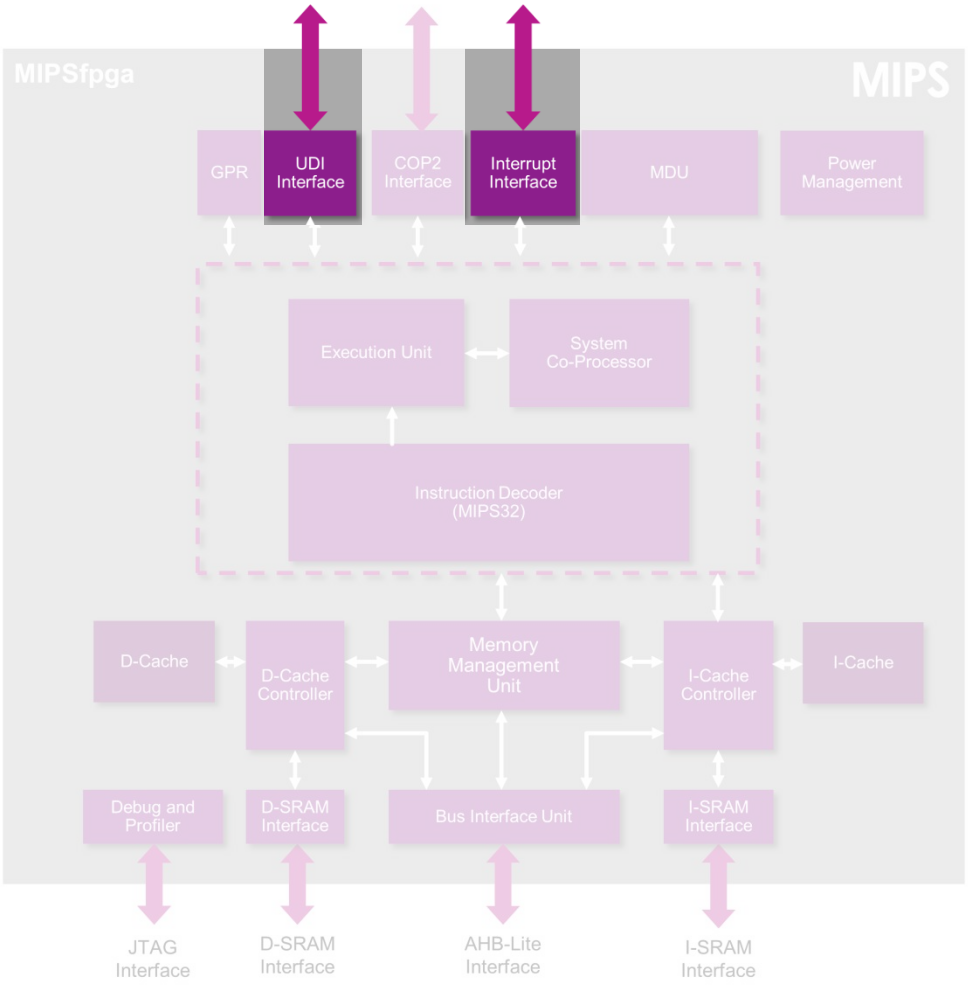
MIPSfpga: Шина AHB-Lite

Шина AHB-Lite : Используется для организации взаимодейст-вия с памятью и периферией



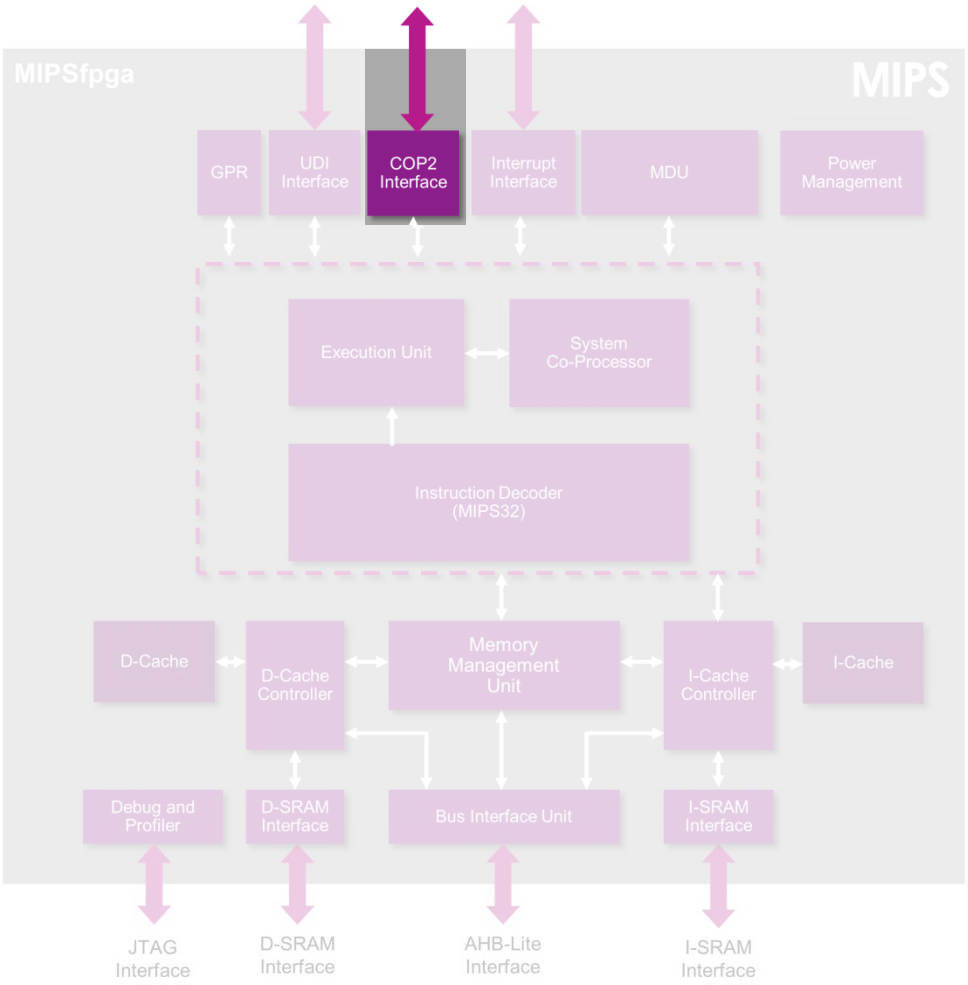
MIPSfpga: Интерфейс UDI,прерывания

UDI (User-Defined Interface Unit, блок интерфейса пользователь-ских команд): Позволяет использовать пользователь-ские команды



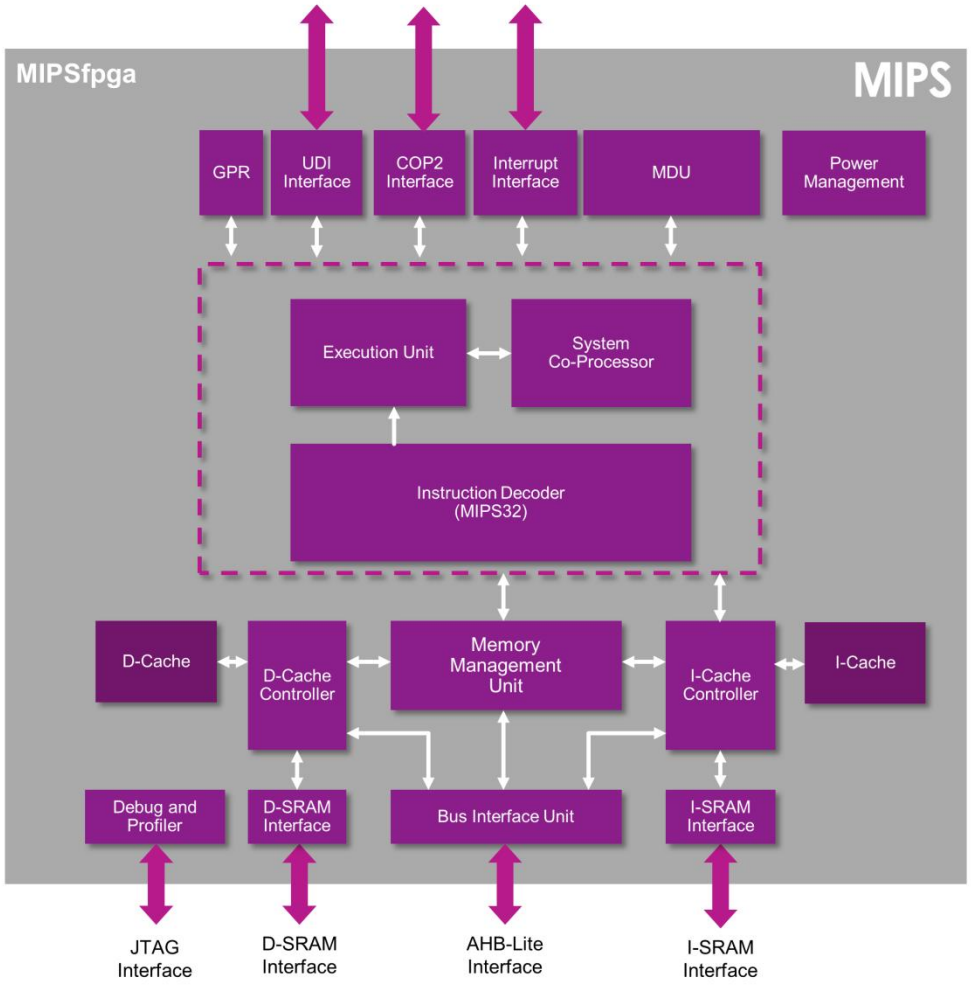
Интерфейс прерываний: используется для аппаратных прерываний

MIPSfpga: Интерфейс UDI,прерывания



Интерфейс COP2:
Интерфейс сопроцессора 2

Ядро MIPSfpga



Ядро MIPSfpga: резюме

- **Коммерческое ядро microAptive**
  - пятистадийный конвейер
  - раздельные Блоки кэш-памяти команд и данных, двухсекционные ассоциативные, по 2KB
  - MMU (блок управления памятью) с буфером ассоциативной трансляции (TLB) объемом 16 записей
  - Счетчики производительности, входные синхронизаторы
  - Отсутствуют DSP, сопроцессор 2, теневые регистры
  - Интерфейсы:
    - Шина AHB-Lite
    - EJTAG программатор/отладчик
    - CorExtend для пользовательских команд

MIPSfpga: Пятистадийный конвейер

#	Стадия	Название	Описание
1	I	Команда	Выборка команды
2	E	Выполнение	Операнды считываются из RF, ALU выполняет операцию
3	M	Память	Доступ к памяти
4	A	Выравнивание	Выравнивание данных по границе слов
5	W	Обратная запись	Запись результатов в RF

Режимы работы MIPSfpga

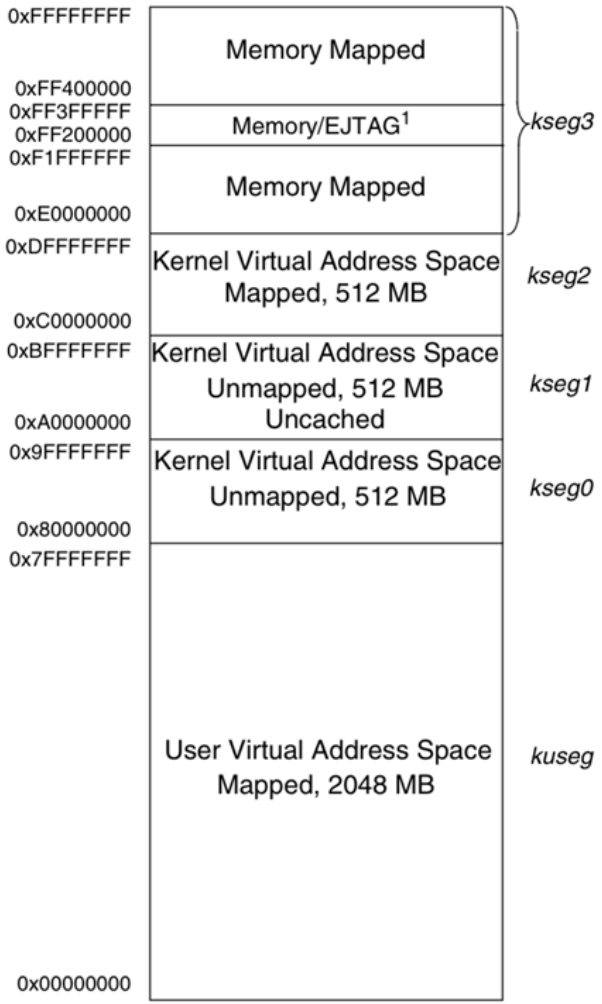
- Супервизора
- Пользователя
- Отладки

Режимы работы MIPSfpga

- Супервизора
- Пользователя
- Отладки

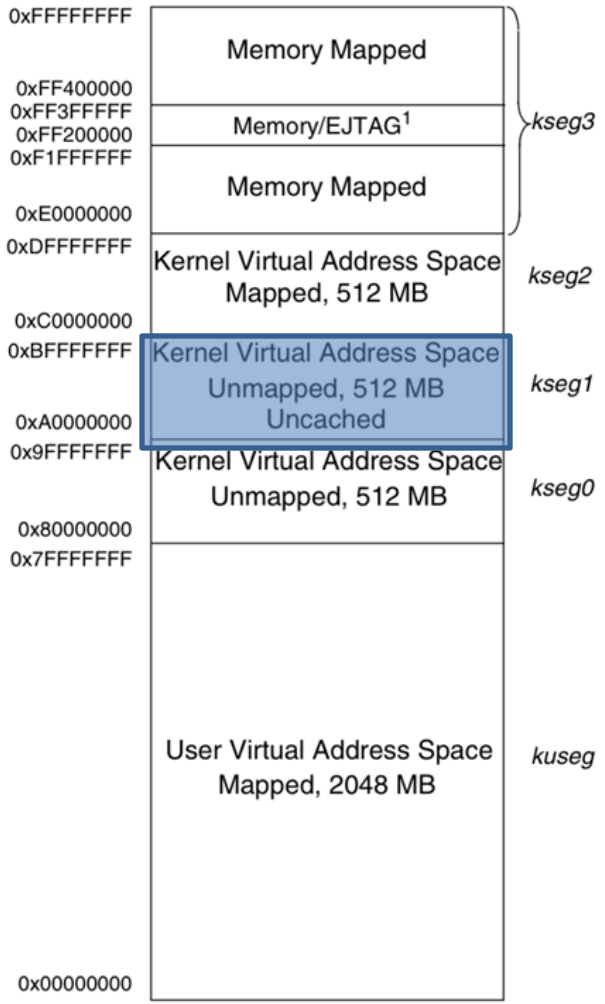
После сброса процессор начинает работать в режиме супервизора и осуществляет переход к команде по адресу 0xbfc00000.

Карта памяти MIPSfpga



- **32-разрядное** виртуальное адресное пространство (0x00000000 – 0xFFFFFFFF)
- Разделено на насколько сегментов
- Оба сегмента **kseg0** и **kseg1** отображаются на физический сегмент по адресу 0x0, т.е.:
  - **0xA0000000** отображается на физический адрес **0x00000000**
  - **0xBFC00000** => **0x1FC00000**
  - **0x80000000** => **0x00000000**

Карта памяти MIPSfpga

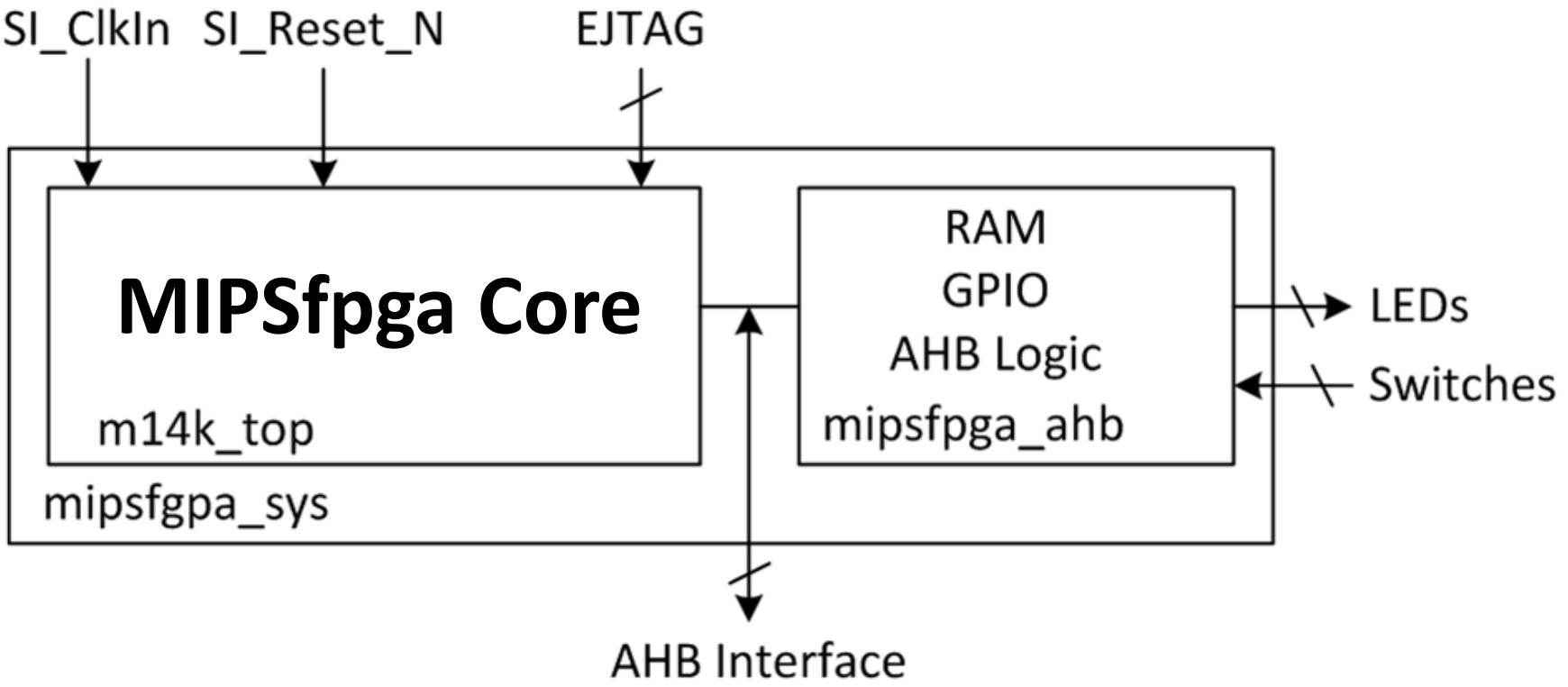


- После сброса процессор начинает работать в режиме супервизора и осуществляет переход к команде по адресу **0xBFC00000**
- Сегмент **kseg1:** не отображается TLB и не кэшируется (эти блоки после сброса не инициализированы)
  - Все команды **выбираются из внешней памяти** (а не из кэш-памяти)
  - Адрес **0xBFC00000** отображается на физический адрес **0x1FC00000**

Обзор MIPSfpga

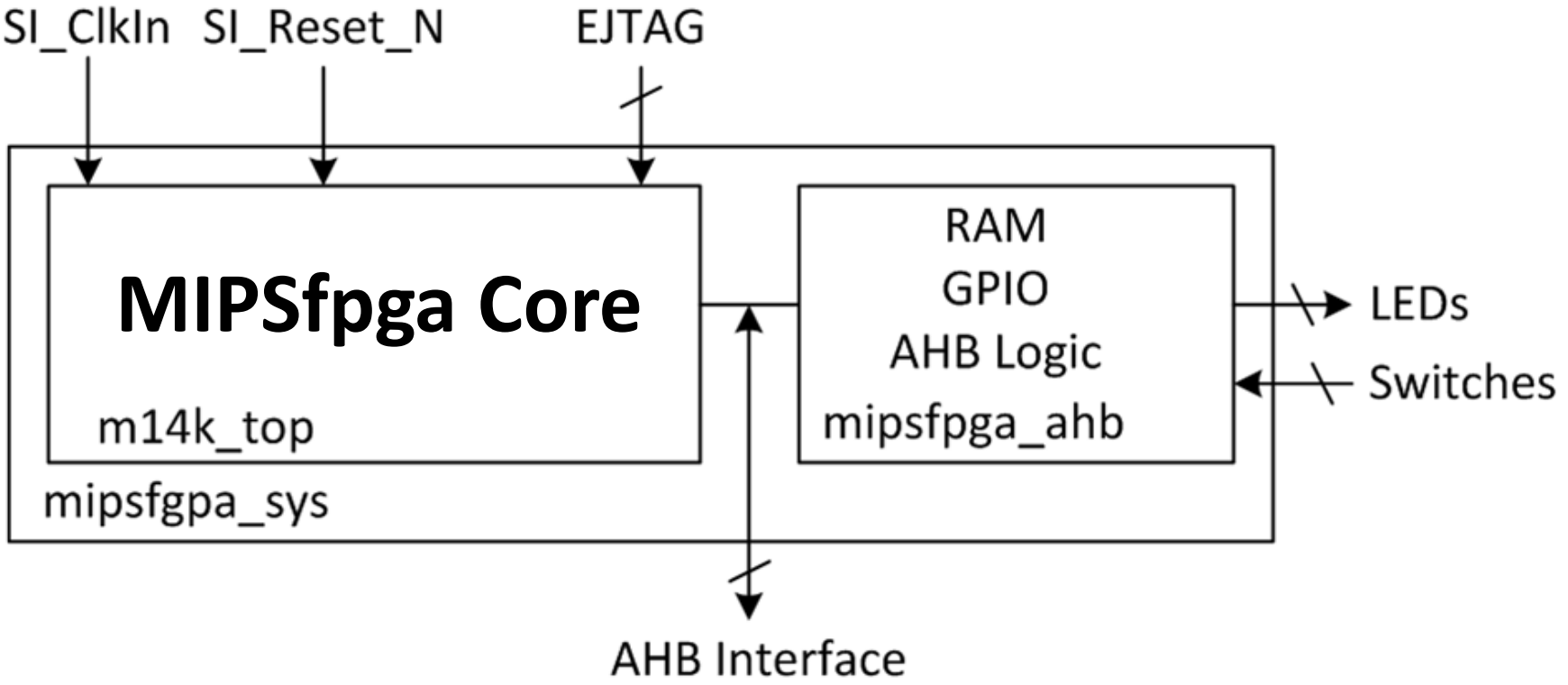
- **История архитектуры MIPS**
- **MIPSfpga**
  - Основы
  - Ядро и **система**
  - Интерфейсы
    - Системный интерфейс
    - Шина AHB-Lite
    - EJTAG

Система MIPSfpga

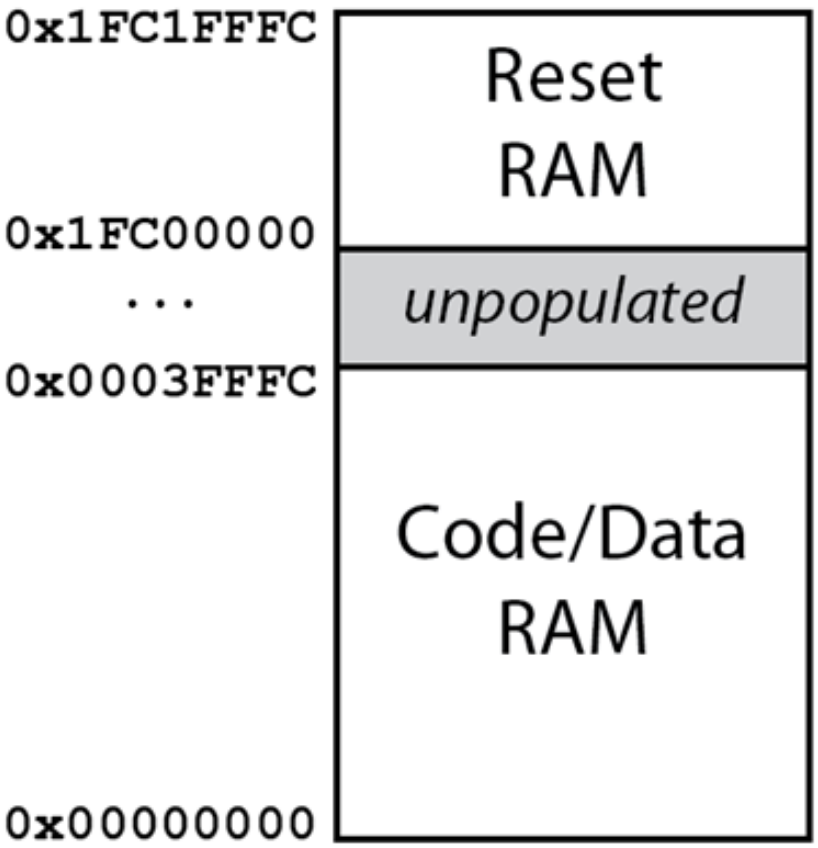


Система MIPSfpga

ОЗУ: **128 KB** (загрузочный код)
256 KB (пользовательский код)



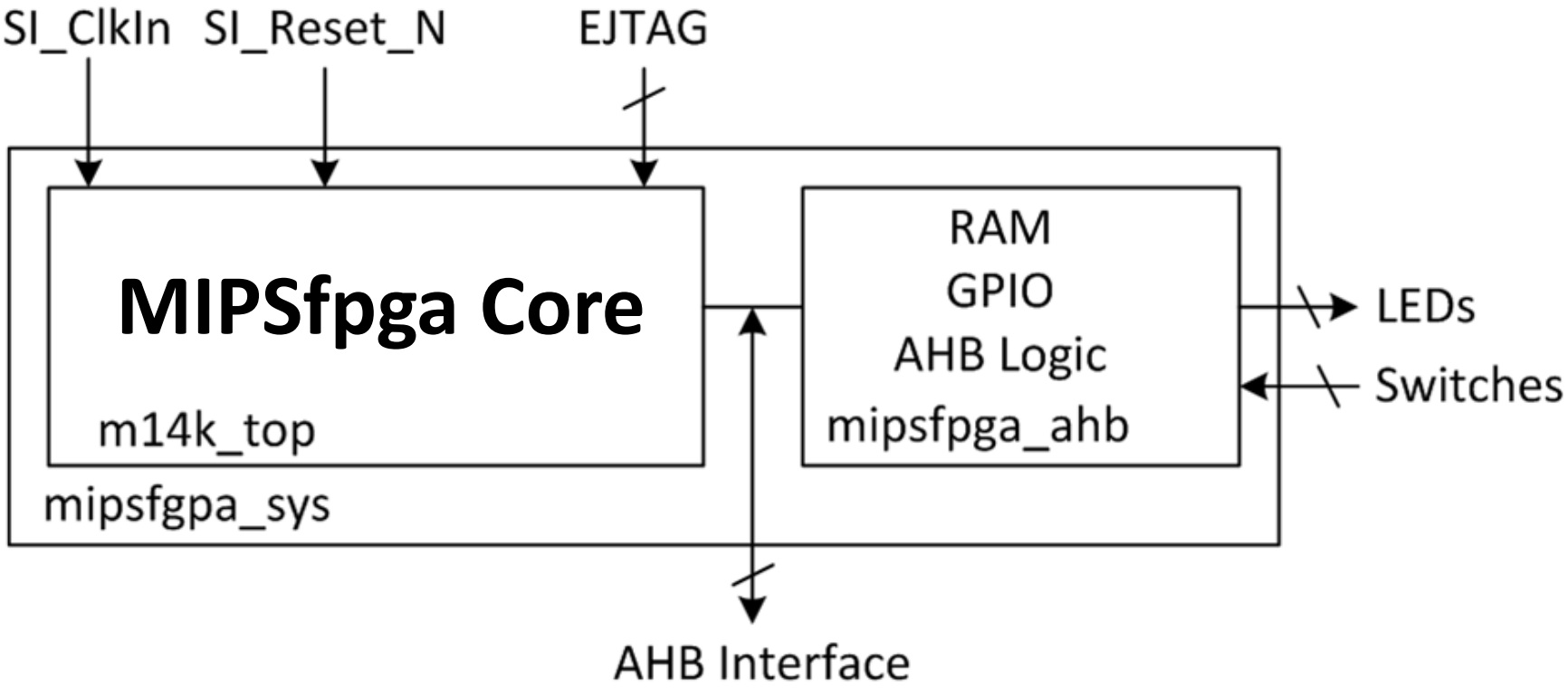
Система MIPSfpga: Физическая память



Загрузочный код:
Выполняется после включения (128 KB)

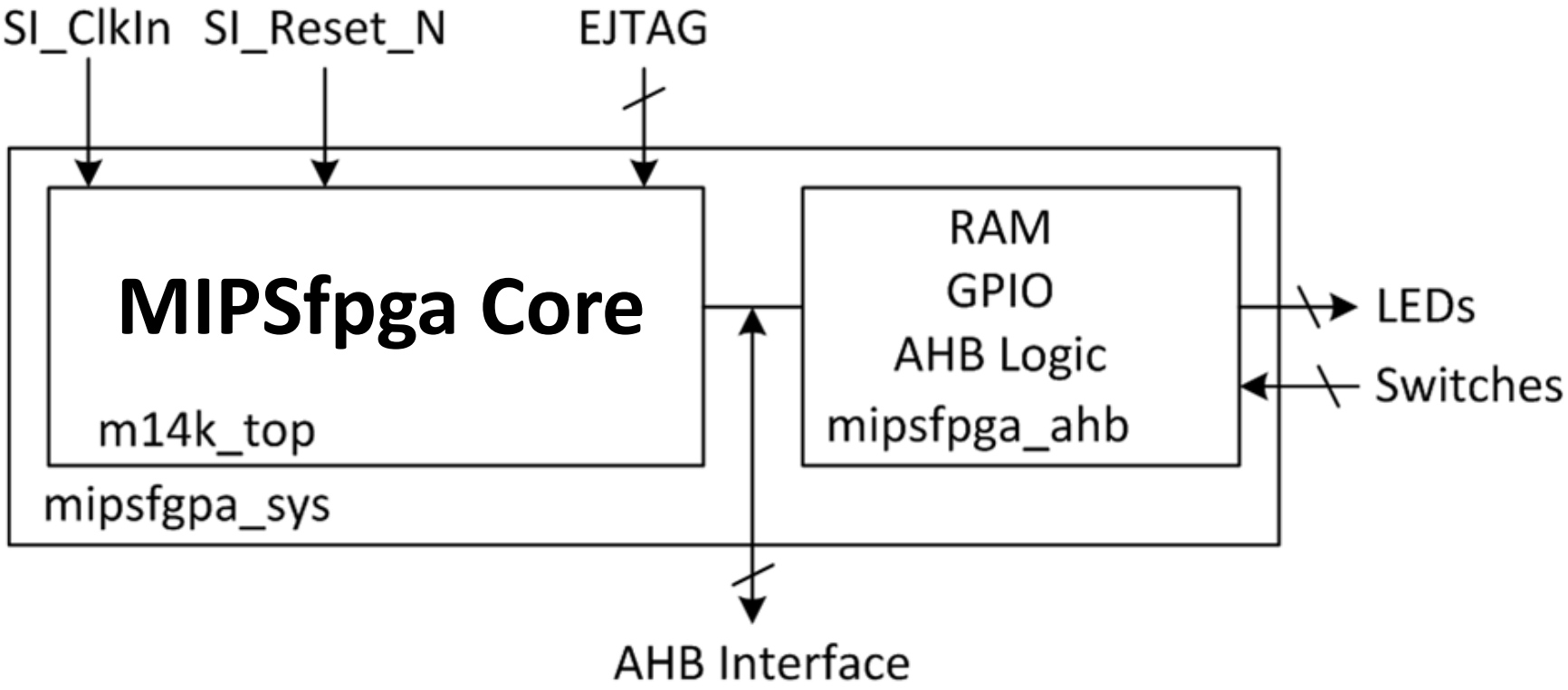
Пользовательский код/данные
(256 KB)

Система MIPSfpga



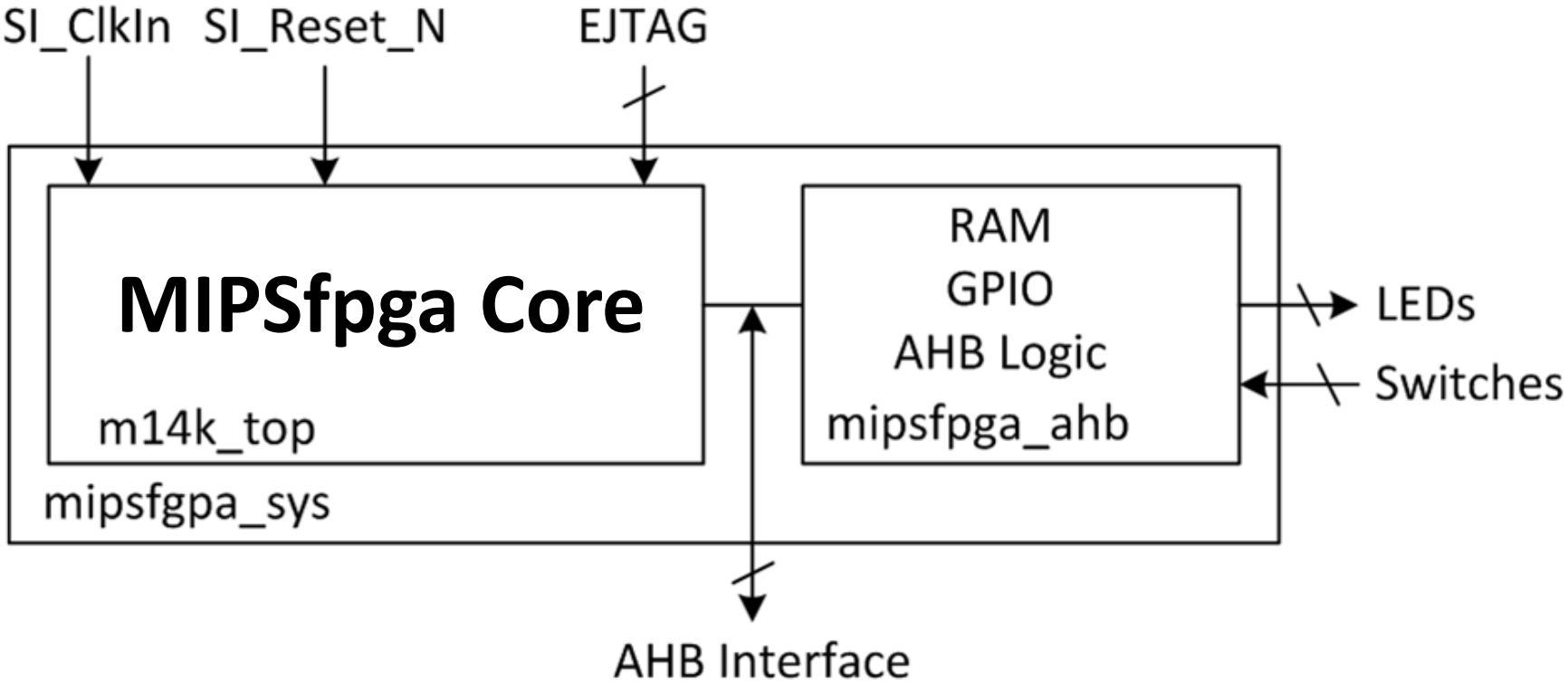
Система MIPSfpga

Система

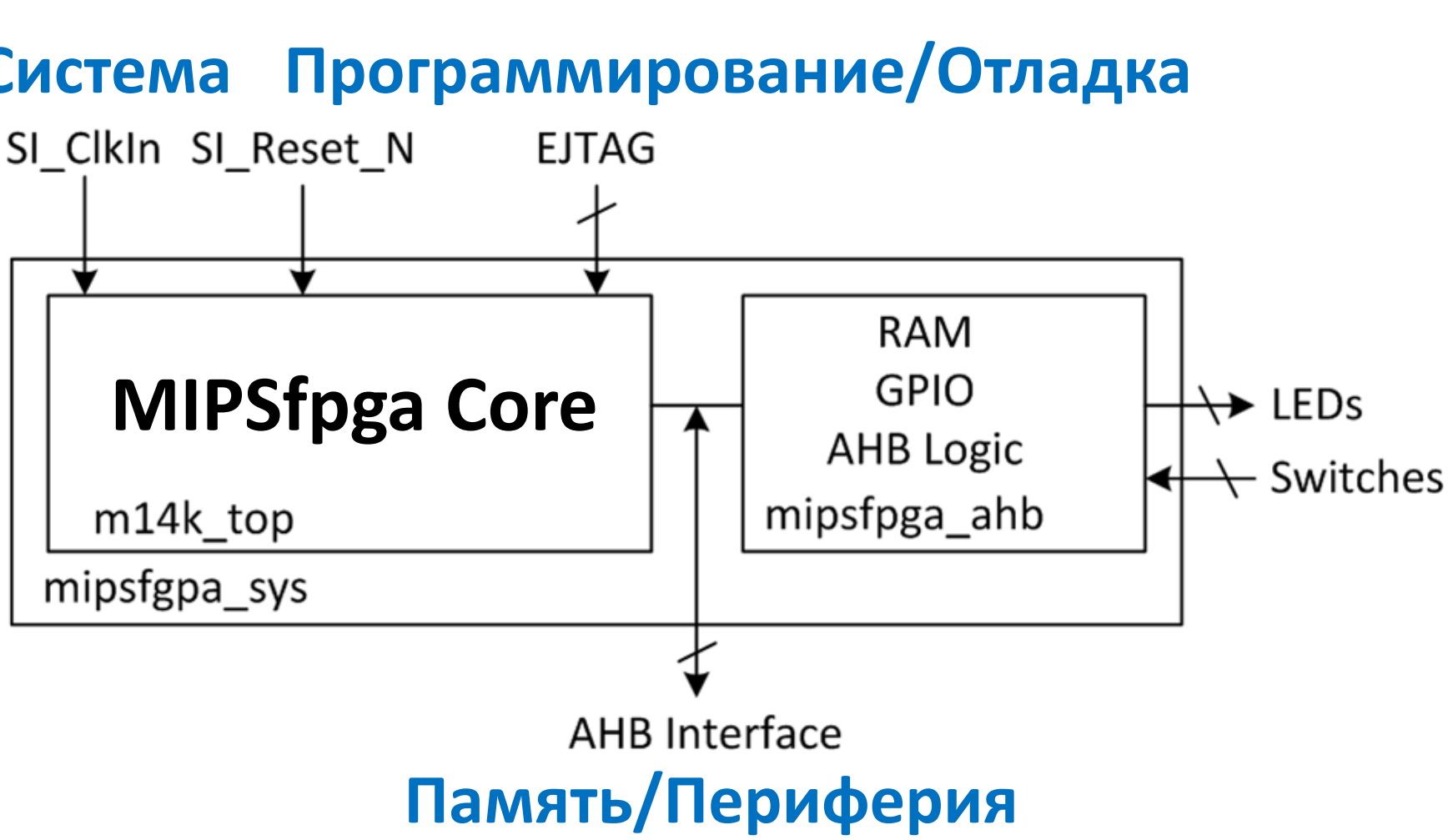


Система MIPSfpga

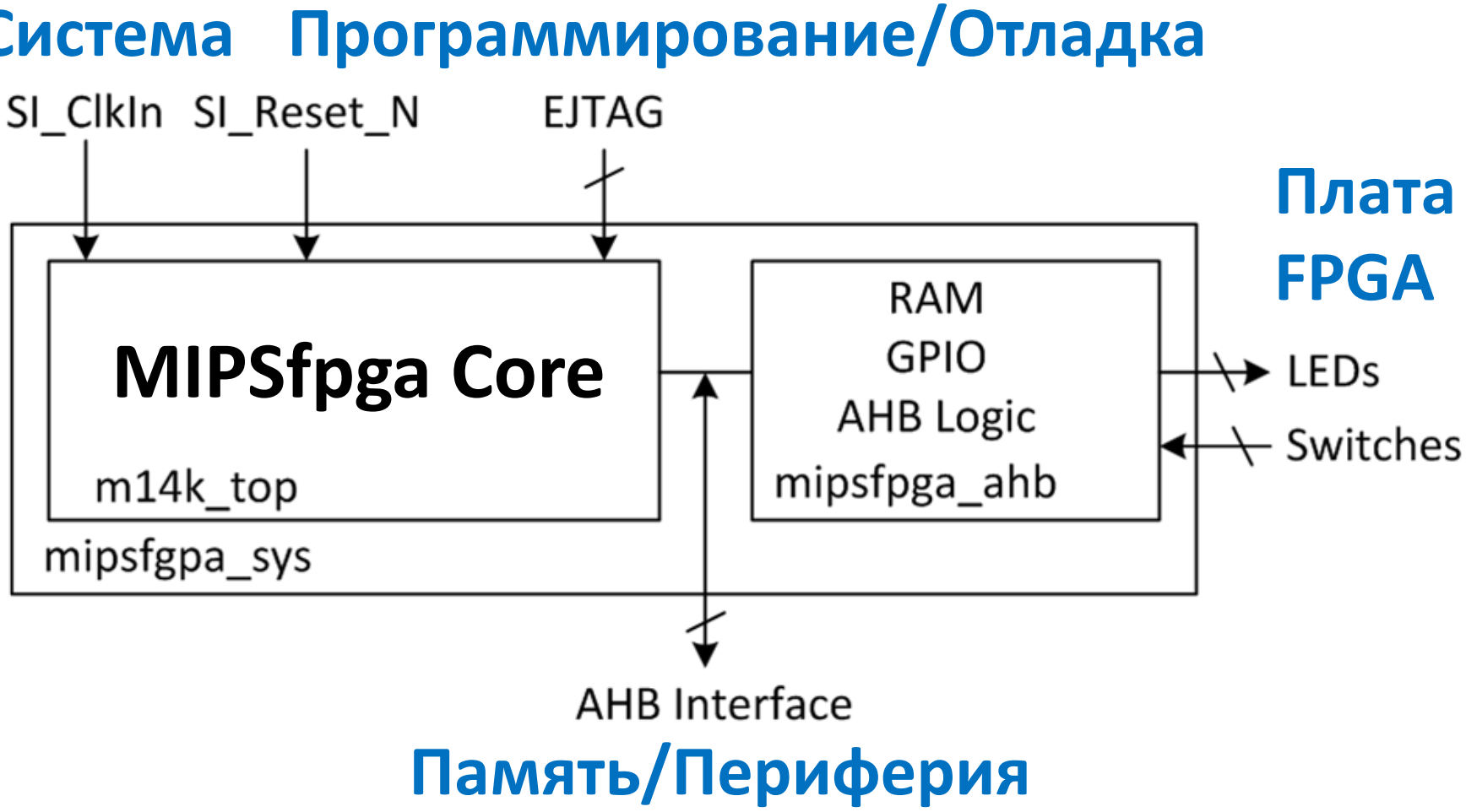
Система Программирование/Отладка



Система MIPSfpga



Система MIPSfpga



Обзор MIPSfpga

- История архитектуры MIPS
- **MIPSfpga**
  - Основы
  - Ядро и система
  - **Интерфейсы**
    - **Системный интерфейс**
    - **Шина AHB-Lite**
    - **EJTAG**

Обзор MIPSfpga

- История архитектуры MIPS
- **MIPSfpga**
  - Основы
  - Ядро и система
  - **Интерфейсы**
    - **Системный интерфейс**
    - **Шина AHB-Lite**
    - **EJTAG**

Системные интерфейсы MIPSfpga

Сигнал	Описание	Плата Nexys4 DDR
SI_Reset_N	Сброса процессора (активный низкий уровень)	Кнопка сброса CPU
SI_ClkIn	Тактовый сигнал	62 MHz (генерируется из 100 MHz сигнала платы)

Префикс **SI** используется в Verilog файлах для сигналов системного интерфейса

Обзор MIPSfpga

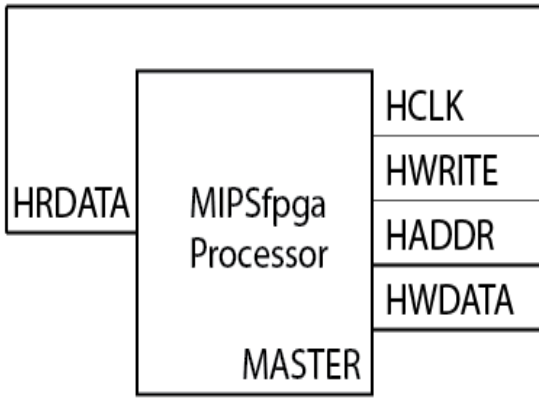
- История архитектуры MIPS
- **MIPSfpga**
  - **Основы**
  - Ядро и система
  - **Интерфейсы**
    - **Системный интерфейс**
    - **AHB-Lite Bus**
    - **EJTAG**

Интерфейсы MIPSfpga: AHB-Lite

Наименование сигнала	Описание
HADDR[31:0]	Шина адреса
HRDATA[31:0]	Шина чтения данных
HWDATA[31:0]	Шина записи данных
HWRITE	Разрешение записи
HCLK	Тактовый сигнал

Префикс **H** используется в Verilog файлах для сигналов шины A**H**B-Lite

Интерфейсы MIPSfpga: AHB-Lite



Шина AHB-Lite: Память/Периферия

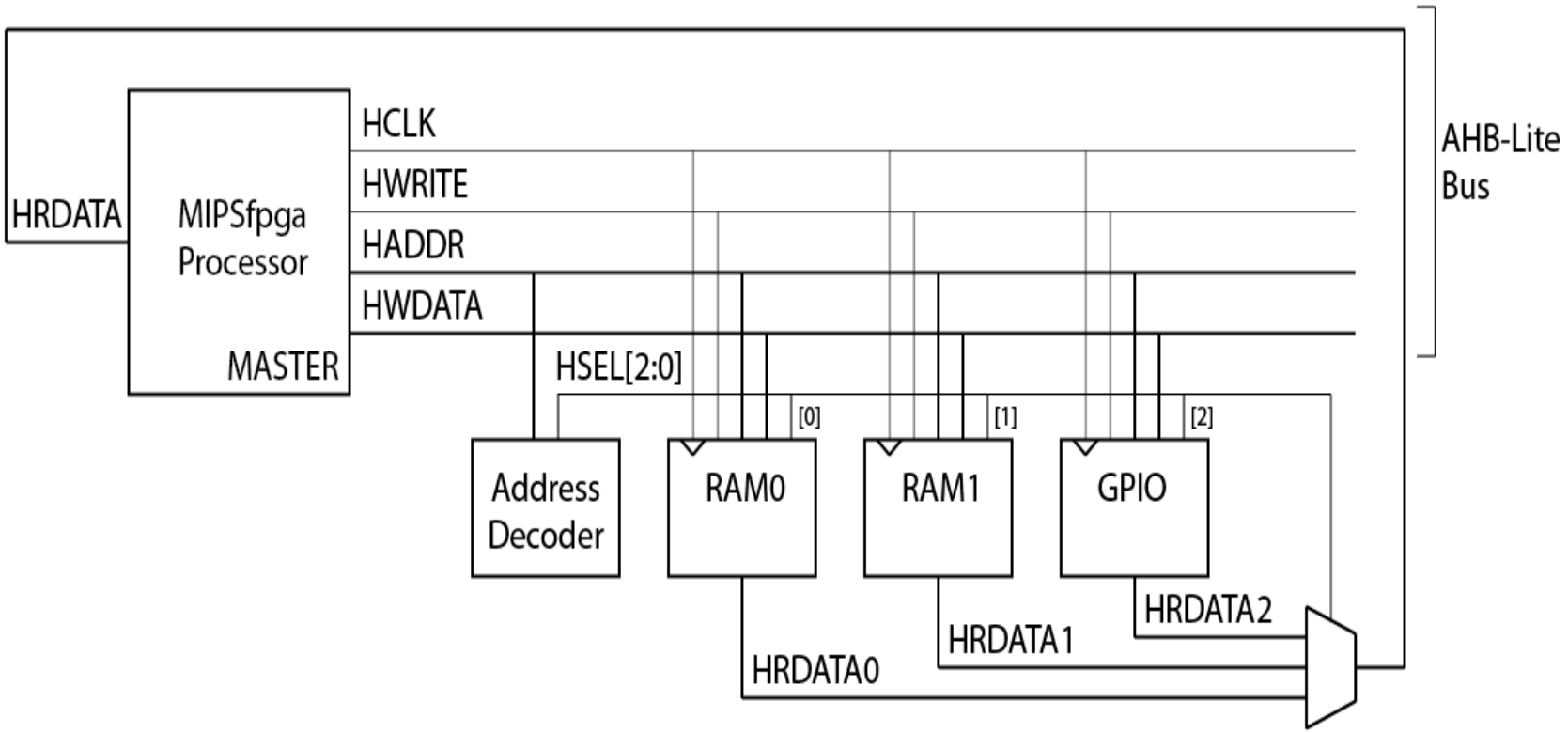
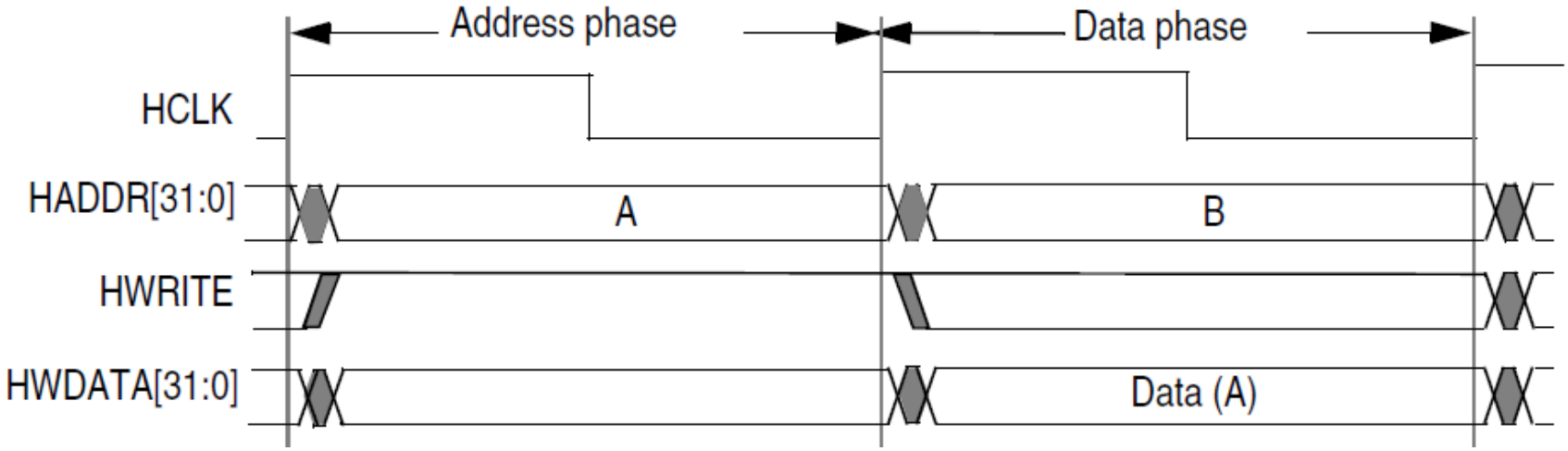


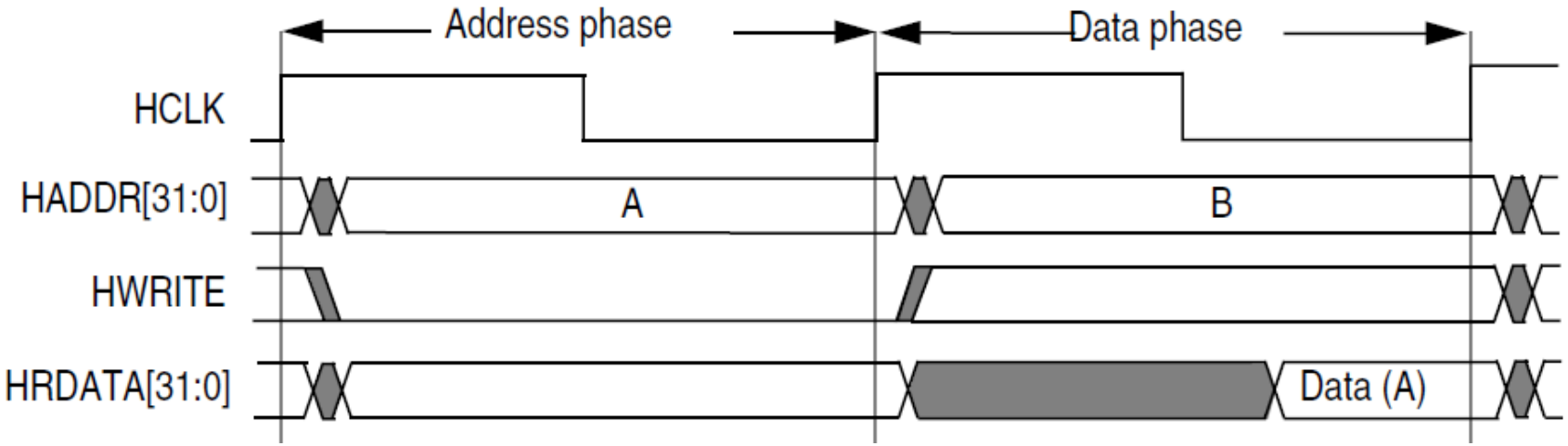
Диаграмма записи AHB-Lite



Цикл 1: Адрес и разрешение записи (HADDR & HWRITE)

Цикл 2: Запись данных(HWDATA)

Диаграмма чтения AHB-Lite



Цикл 1: **Адрес**(HADDR) (также, HWRITE = 0)

Цикл2: **Чтение данных** (HRDATA)

Обзор MIPSfpga

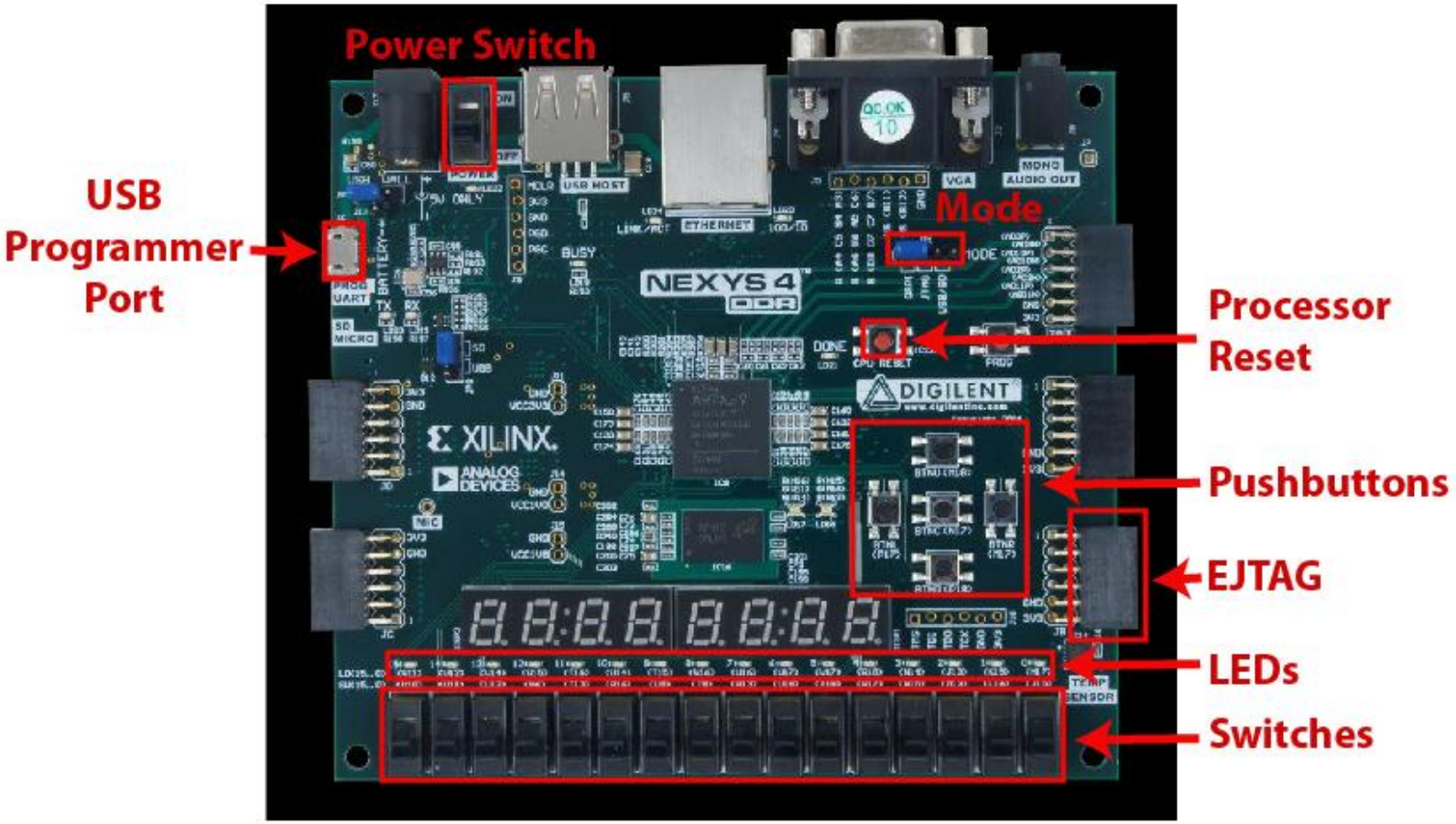
- История архитектуры MIPS
- **MIPSfpga**
  - **Основы**
  - Ядро и система
  - **Интерфейсы**
    - **Системный интерфейс**
    - **Шина AHB-Lite: Ввод/вывод платы FPGA**
    - **EJTAG**

Интерфейсы MIPSfpga: плата Nexys4

Наименование сигнала	Описание
IO_LEDR[15:0]	Светодиоды
IO_Switch[15:0]	Переключатели
IO_PB[4:0]	Кнопки (BTNU, D, L, R, C)

Префикс **IO** используется в Verilog файлах для сигналов ввода/вывода FPGA платы

FPGA плата Nexys4 DDR

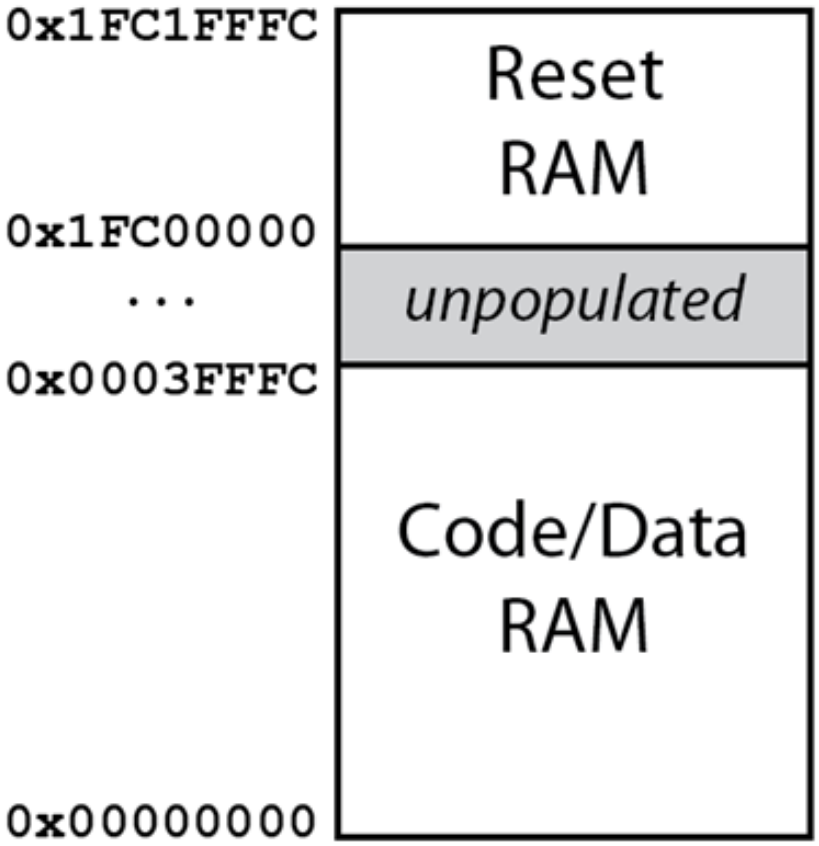


Ввод/вывод с отображением в память

Наименование сигнала	Виртуальный адрес	Физический адрес
IO_LEDR[15:0]	0xbf800000	0x1f800000
IO_Switch[15:0]	0xbf800008	0x1f800008
IO_PB[4:0]	0xbf80000c	0x1f80000c

- Запись по адресу 0xbf800000 **включает/выключает светодиоды**
- Чтение по адресу 0xbf800008 **позволяет определить положение переключателей**

Система MIPSfpga: Физическая память



По адресам, на которые отображаются устройства ввода/вывода, физическая память отсутсвует (0x1f800000+)

Ввод/вывод с отображением в память

Наименование сигнала	Виртуальный адрес	Физический адрес
IO_LEDR[15:0]	0xbf800000	0x1f800000
IO_Switch[15:0]	0xbf800008	0x1f800008
IO_PB[4:0]	0xbf80000c	0x1f80000c

- Запись по адресу 0xbf800000 **включает/выключает светодиоды**

Ввод/вывод с отображением в память

Наименование сигнала	Виртуальный адрес	Физический адрес
IO_LEDR[15:0]	0xbf800000	0x1f800000
IO_Switch[15:0]	0xbf800008	0x1f800008
IO_PB[4:0]	0xbf80000c	0x1f80000c

- Запись по адресу 0xbf800000 **включает/выключает светодиоды**

```
// Write 0x543 to LEDs
addiu $7, $0, 0x543 # $7 = 0x543
lui   $5, 0xbf80    # $5 = 0xbf800000 (LED address)
sw    $7, 0($5)     # LEDs = 0x543
```

Ввод/вывод с отображением в память

Наименование сигнала	Виртуальный адрес	Физический адрес
IO_LEDR[15:0]	0xbf800000	0x1f800000
IO_Switch[15:0]	0xbf800008	0x1f800008
IO_PB[4:0]	0xbf80000c	0x1f80000c

- Запись по адресу 0xbf800000 **включает/выключает светодиоды**
- Чтение по адресу 0xbf800008 **позволяет определить положение переключателей**

Ввод/вывод с отображением в память

Наименование сигнала	Виртуальный адрес	Физический адрес
IO_LEDR[15:0]	0xbf800000	0x1f800000
IO_Switch[15:0]	0xbf800008	0x1f800008
IO_PB[4:0]	0xbf80000c	0x1f80000c

- Запись по адресу 0xbf800000 **включает/выключает светодиоды**
- Чтение по адресу 0xbf800008 **позволяет определить положение переключателей**

```
// Read value of switches into $10
lui   $5,  0xbf80    # $5 = 0xbf800000
lw    $10, 8($5)     # $10 = value of switches
```

Пример программы для MIPS

Код на языке C

```
unsigned int val = 1;
volatile unsigned int* dest;
dest = 0xbf800000;

while (1) {
  *dest = val;
  val++;
}
```

Пример программы для MIPS

Код на языке C

```
unsigned int val = 1;
volatile unsigned int* dest;
dest = 0xbf800000;

while (1) {
  *dest = val;
  val++;
}
```

Код на языке ассемблер MIPS

```
# $9=val, $8=0xbf800000
    addiu $9, $0, 1    # val=1
    lui   $8, 0xbf80   # address

L1: sw    $9, 0($8)    # write to addr
    addiu $9, $9, 1    # val++
    beqz  $0, L1       # loop
    nop                # branch delay
                       # slot
```

Пример программы для MIPS

Код на языке C

```
unsigned int val = 1;
volatile unsigned int* dest;
dest = 0xbf800000;

while (1) {
  *dest = val;
  val++;
}
```

Код на языке ассемблер MIPS

```
# $9=val, $8=0xbf800000
    addiu $9, $0, 1    # val=1
    lui   $8, 0xbf80   # address

L1: sw    $9, 0($8)    # write to addr
    addiu $9, $9, 1    # val++
    beqz  $0, L1       # loop
    nop                # branch delay
                       # slot
```

Записывает увеличивающееся значение в слово по адресу 0xbf800000 (светодиоды)

Как запустить программу на MIPSfpga?

- **В среде моделирования**
- **В FPGA:**
  - **Загрузить программу в память при синтезе**
  - **Загрузить программу в память через интерфейс EJTAG**

Как запустить программу на MIPSfpga?

- **В среде моделирования**
- **В FPGA:**
  - **Загрузить программу в память при синтезе**
  - **Загрузить программу в память через интерфейс EJTAG**

Обзор MIPSfpga

- История архитектуры MIPS
- **MIPSfpga**
  - **Основы**
  - Ядро и система
  - **Интерфейсы**
    - **Системный интерфейс**
    - **Шина AHB-Lite**
    - **EJTAG**

Интерфейсы MIPSfpga: EJTAG

- Используется для программирования и отладки ядра MIPSfpga
- Наименования сигналов и функции заимствованы у протокола JTAG

Интерфейсы MIPSfpga: EJTAG

Название сигнала	Описание
EJ_TDI	Вход Тестовых Данных
EJ_TDO	Выход Тестовых Данных
EJ_TMS	Выбор Режима Тестирования
EJ_TCK	Тактовый сигнал тестирования
EJ_DINT	Запрос прерывания отладки
SI_ColdReset_N	Сброс процессора
EJ_TRST_N_probe	Сброс контроллера EJTAG

Префикс **EJ** используется в Verilog файлах для интерфейсных сигналов EJTAG

Как запустить программу на MIPSfpga?

- **В среде моделирования**
- **В FPGA:**
  - **Загрузить программу в память при синтезе**
  - **Загрузить программу в память через интерфейс EJTAG**

Пример программы для MIPS

Код на языке C

```
unsigned int val = 1;
volatile unsigned int* dest;
dest = 0xbf800000;

while (1) {
  *dest = val;
  val++;
}
```

Код на языке ассемблер MIPS

```
# $9=val, $8=0xbf800000
    addiu $9, $0, 1    # val=1
    lui   $8, 0xbf80   # address

L1: sw    $9, 0($8)    # write to addr
    addiu $9, $9, 1    # val++
    beqz  $0, L1       # loop
    nop                # branch delay
                       # slot
```

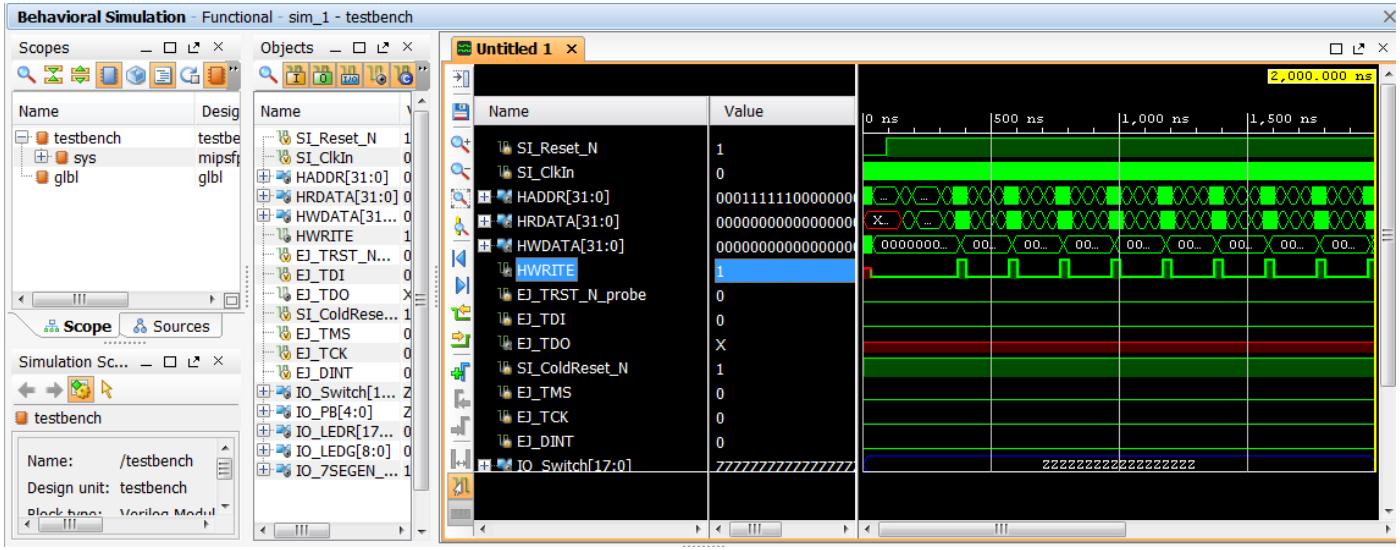
Записывает увеличивающееся значение в слово по адресу 0xbf800000 (светодиоды)

Машинный код

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	// bfc00008:	L1:	sw $9, 0($8)
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop

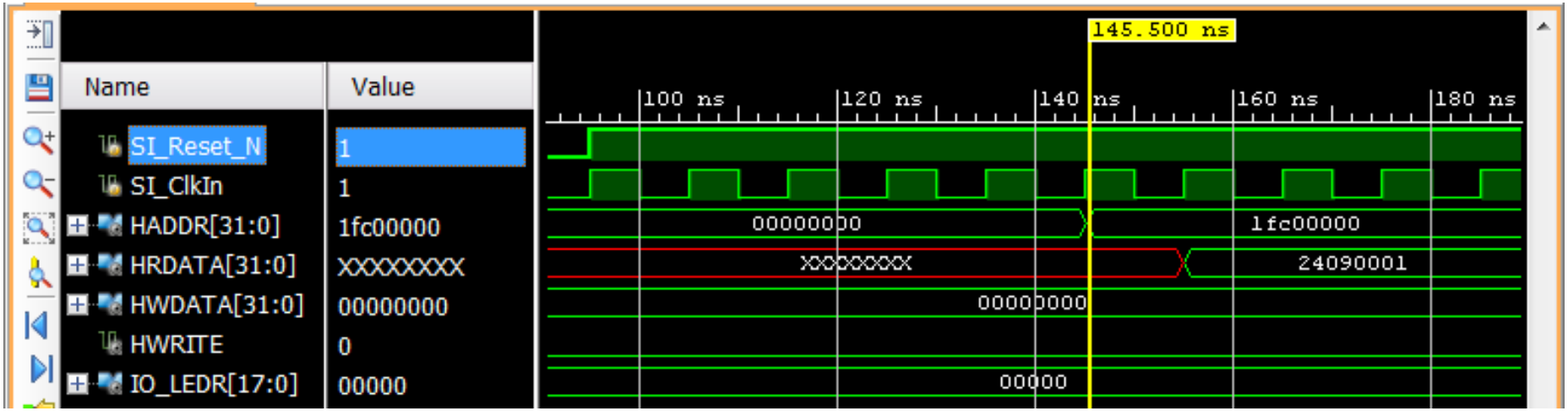
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	// bfc00008:	L1:	sw $9, 0($8)
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



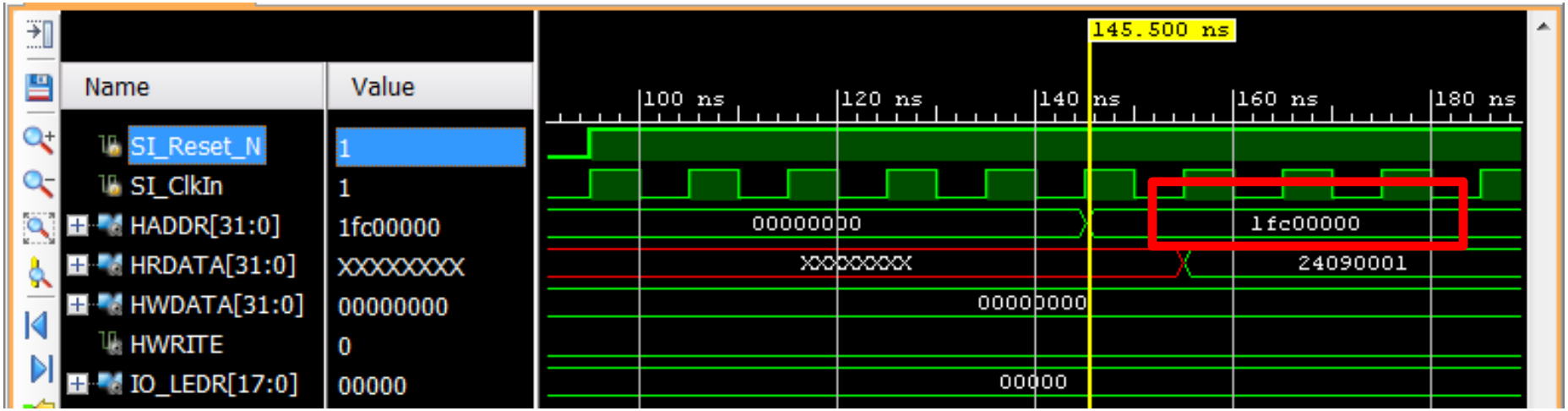
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	// bfc00008:	L1:	sw $9, 0($8)
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



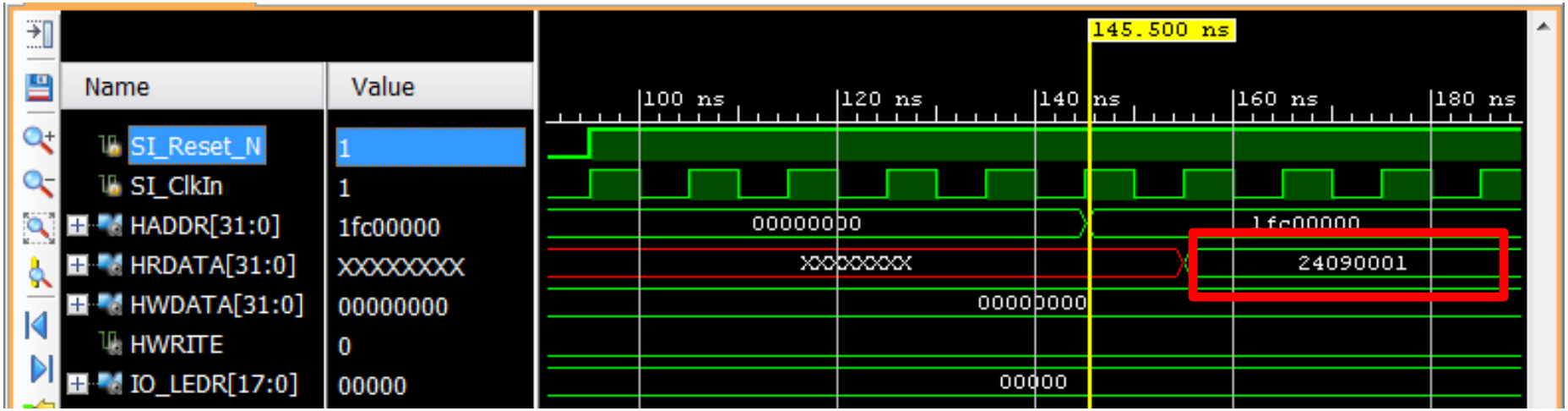
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	// bfc00008:	L1:	sw $9, 0($8)
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



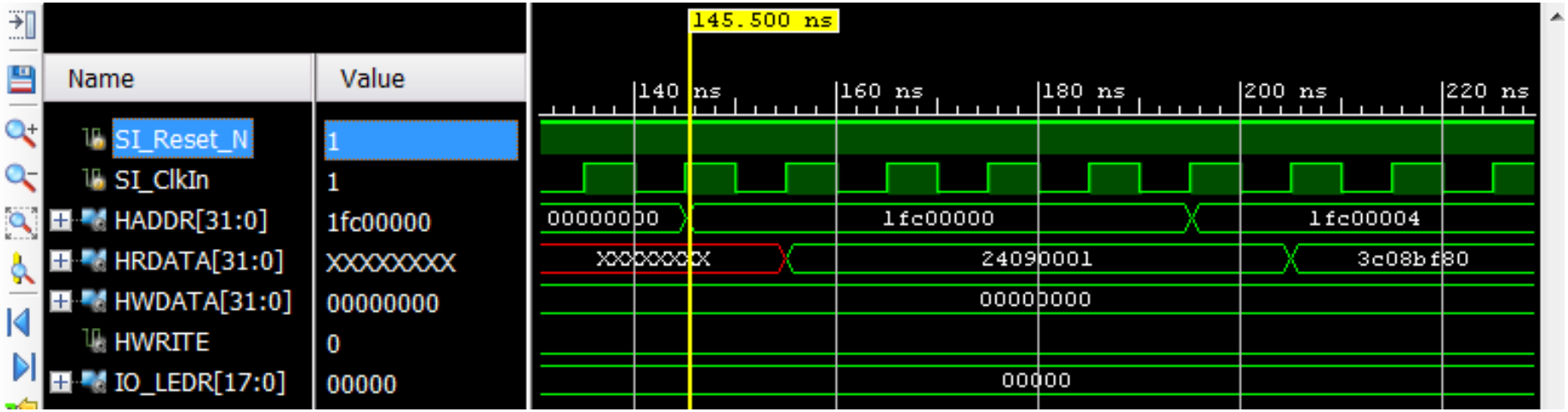
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	// bfc00008:	L1:	sw $9, 0($8)
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



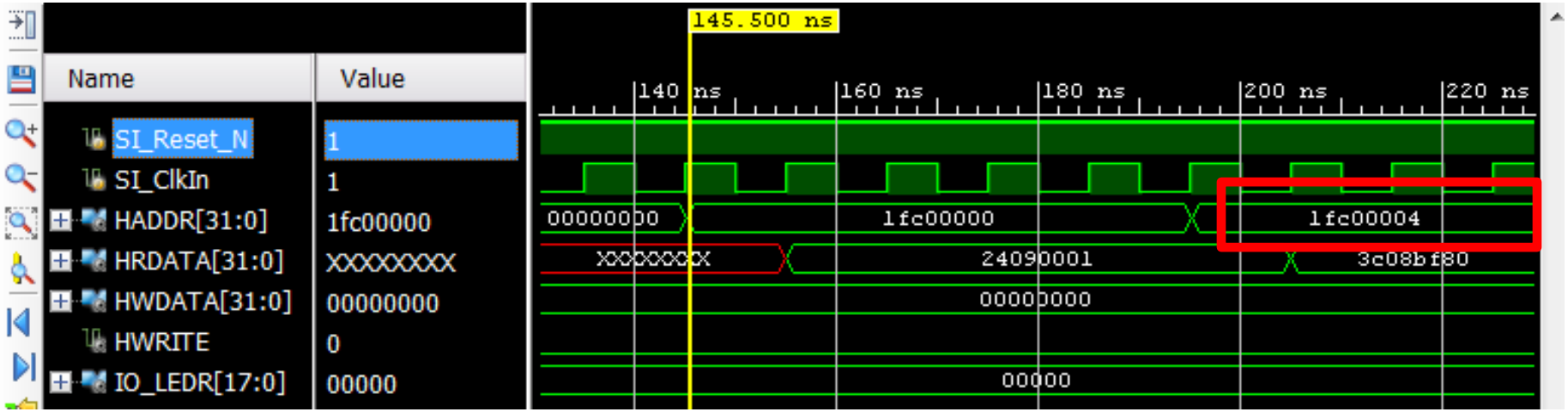
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	// bfc00008:	L1:	sw $9, 0($8)
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



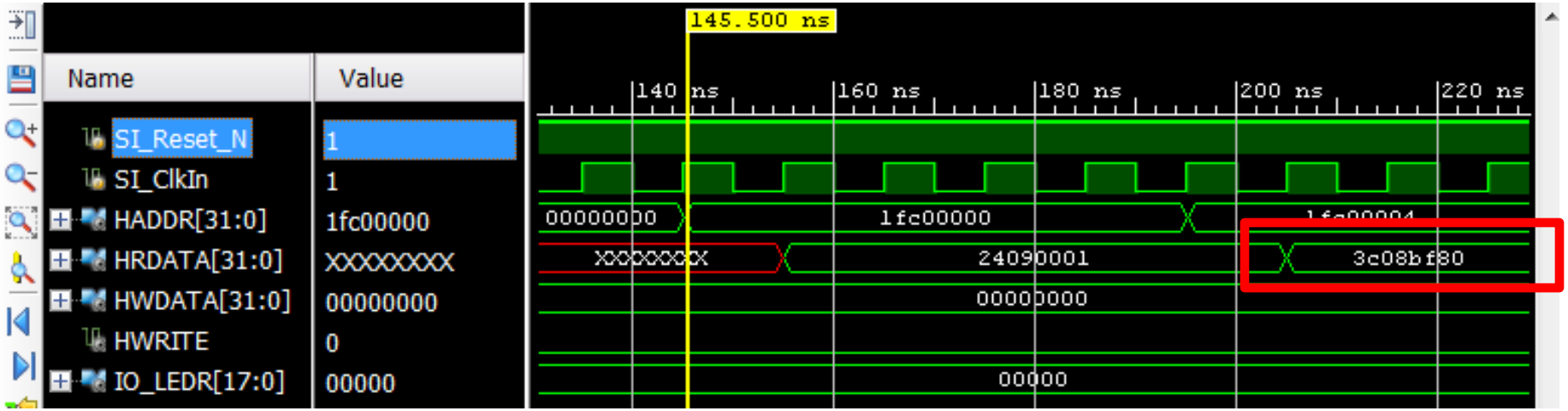
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	// bfc00008:	L1:	sw $9, 0($8)
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



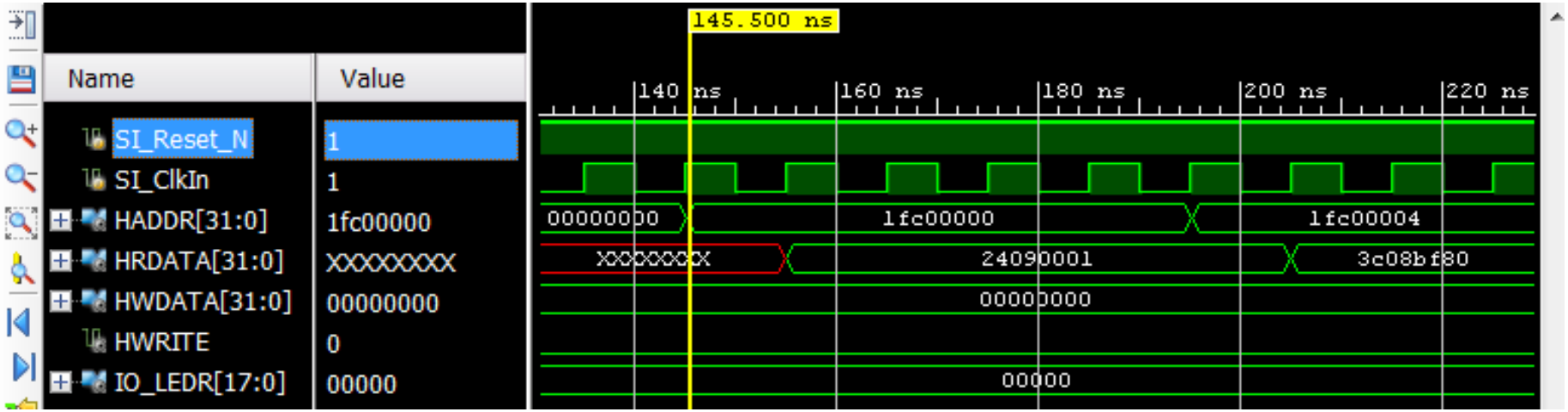
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	// bfc00008:	L1:	sw $9, 0($8)
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



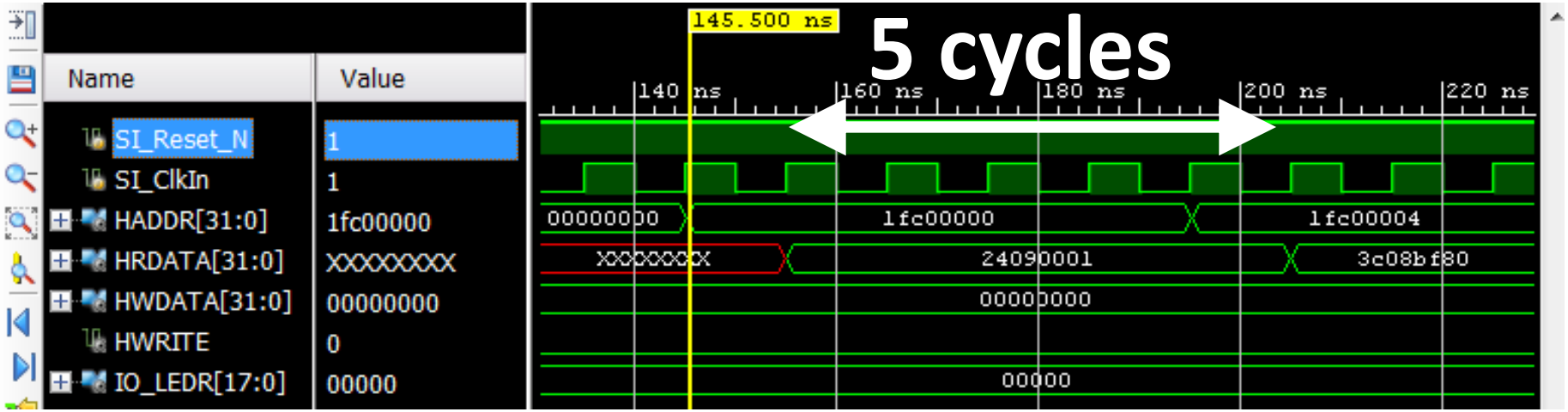
Моделирование MIPSfpga

До инициализации кэш-памяти каждая команда занимает 5 циклов (а не 1).



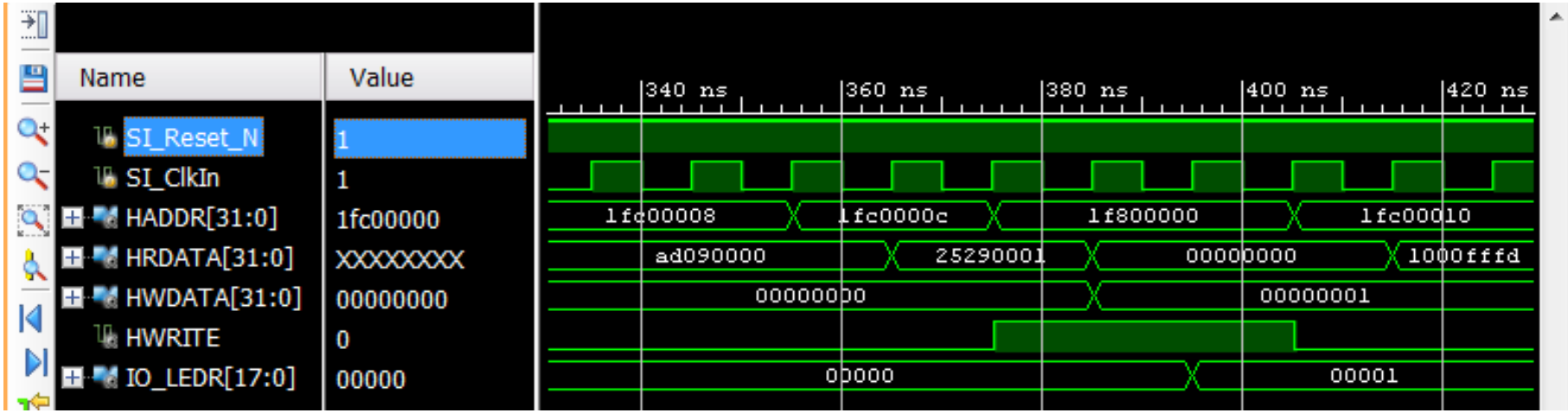
Моделирование MIPSfpga

До инициализации кэш-памяти каждая команда занимает 5 циклов (а не 1).



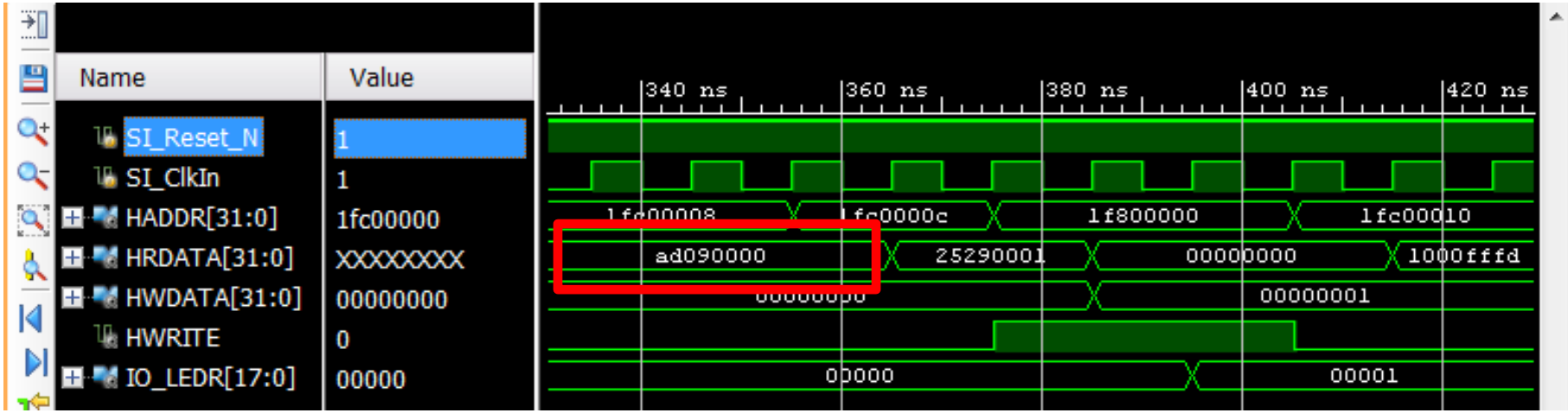
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	**// bfc00008:**	**L1:**	**sw $9, 0($8)**
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



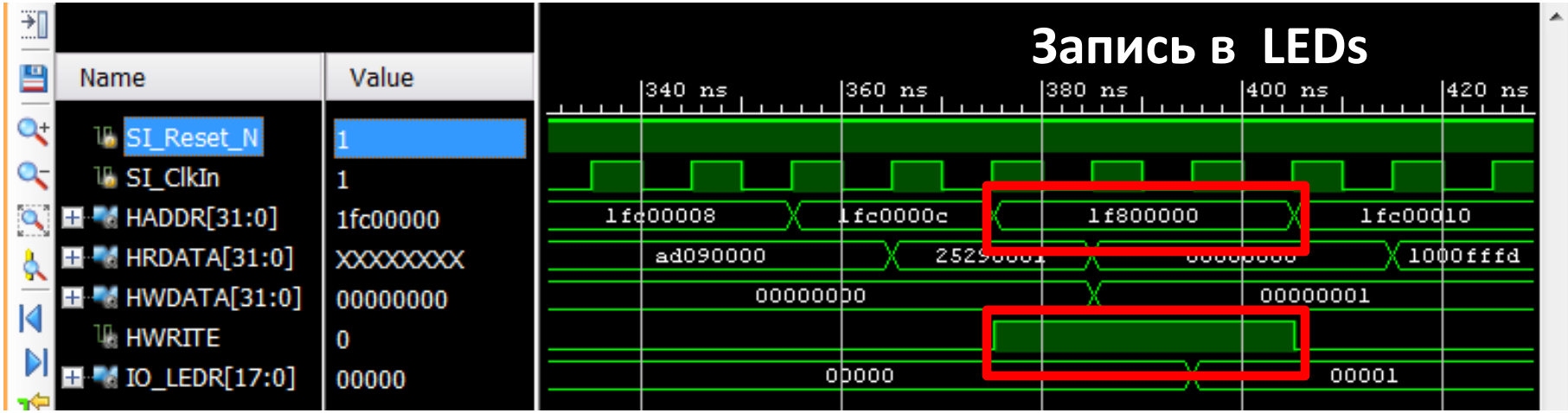
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	**// bfc00008:**	**L1:**	**sw $9, 0($8)**
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



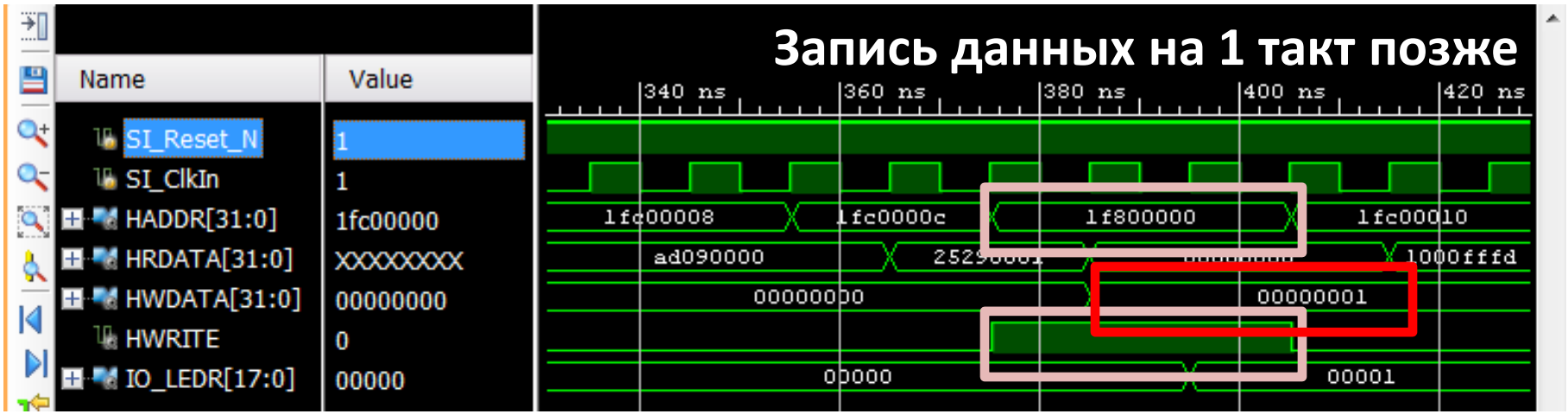
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	**// bfc00008:**	**L1:**	**sw $9, 0($8)**
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



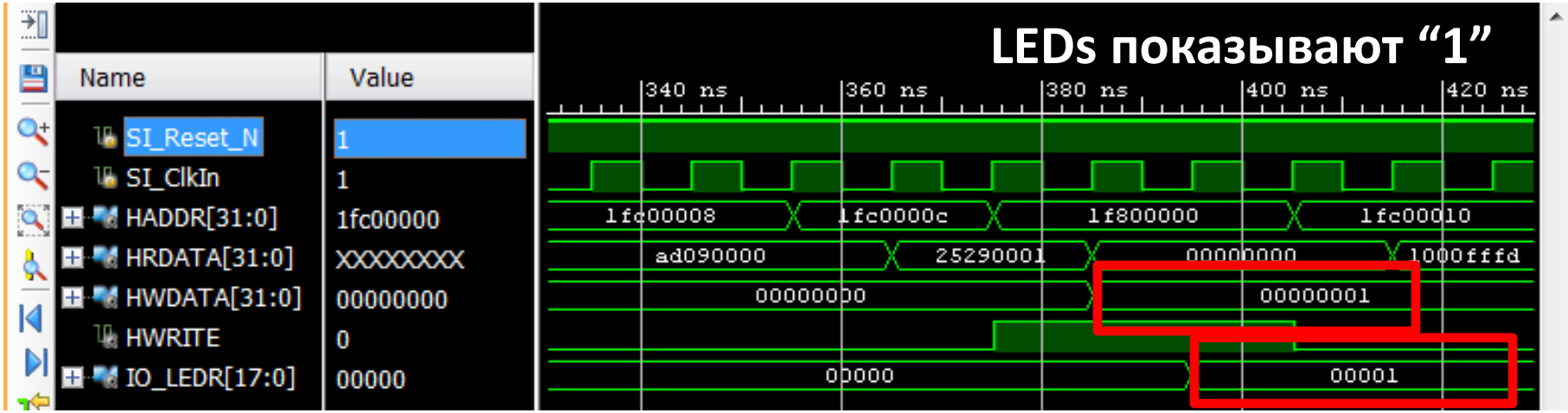
Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	**// bfc00008:**	**L1:**	**sw $9, 0($8)**
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



Моделирование MIPSfpga

Машинный код	Адрес команды		Ассемблер MIPS
24090001	// bfc00000:		addiu $9, $0, 1
3c08bf80	// bfc00004:		lui $8, 0xbf80
ad090000	**// bfc00008:**	**L1:**	**sw $9, 0($8)**
25290001	// bfc0000c:		addiu $9, $9, 1
1000fffd	// bfc00010:		beqz $0, L1
00000000	// bfc00014:		nop



Как запустить программу на MIPSfpga?

- **В среде моделирования**
- **В FPGA:**
  - **Загрузить программу в память при синтезе**
  - **Загрузить программу в память через интерфейс EJTAG**

Загрузка программы при синтезе

Модуль памяти

```
module ram_reset_dual_port
   initial begin
      $readmemh("ram_reset_init.txt",
ram);
   end
```

Файл инициализации памяти

ram_reset_init.txt:

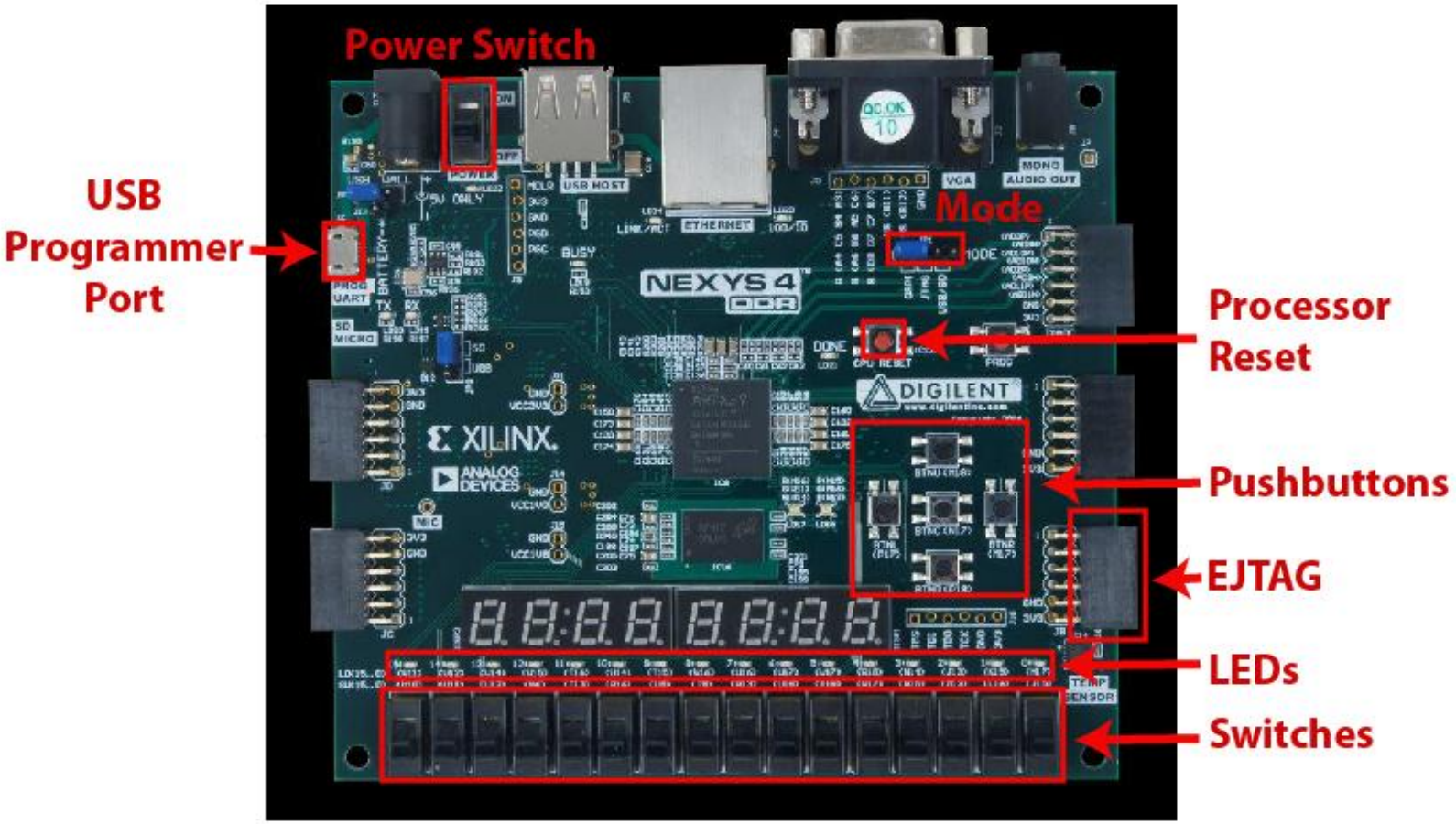
Машинный код	Адрес команды	Ассемблер MIPS
24090001	// bfc00000:	addiu $9, $0, 1
3c08bf80	// bfc00004:	lui $8, 0xbf80
ad090000	// bfc00008: L1:	sw $9, 0($8)
25290001	// bfc0000c:	addiu $9, $9, 1
3c050026	**// bfc00010: delay:**	**lui $5, 0x026**
34a525a0	**// bfc00014:**	**ori $5, $5, 0x25a0**
00003020	**// bfc00018:**	**add $6, $0, $0**
00a63822	**// bfc0001c: L2:**	**sub $7, $5, $6**
20c60001	**// bfc00020:**	**addi $6, $6, 1**
1ce0fffd	**// bfc00024:**	**bgtz $7, L2**
00000000	**// bfc00028:**	**nop**
1000fff6	**// bfc0002c:**	**beq $0, $0, L1**
00000000	**// bfc00030:**	**nop**

Добавлена задержка переключения светодиодов

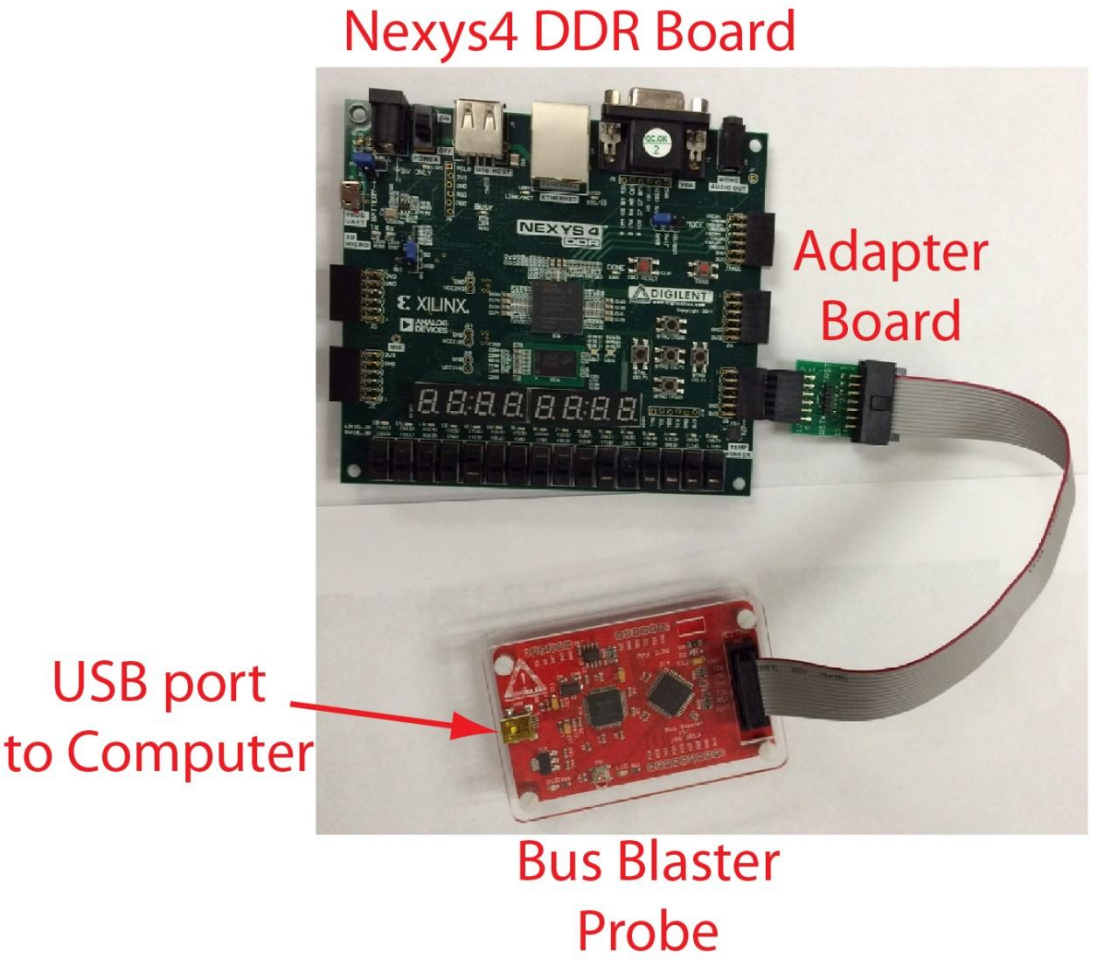
Как запустить программу на MIPSfpga?

- **В среде моделирования**
- **В FPGA:**
  - **Загрузить программу в память при синтезе**
  - **Загрузить программу в память через интерфейс EJTAG**

MIPSfpga на плате Nexys4 DDR



Соединение Nexys4 DDR / Bus Blaster



Загрузка кода в MIPSfpga

1. **Компилировать код – исправить ошибки**
2. **Подсоединить Bus Blaster к плате и компьютеру**
3. **Загрузить систему MIPSfpga на плату Nexys4 DDR**
4. **Загрузить код с помощью скрипта:**

   loadMIPSfpga.bat

Загрузка кода в MIPSfpga

1. **Компилировать код – исправить ошибки**
2. **Подсоединить Bus Blaster к плате и компьютеру**
   - с помощью zadig.exe установить драйвера
3. **Загрузить систему MIPSfpga на плату Nexys4 DDR**
   - при необходимости установить драйвера
4. **Загрузить код с помощью скрипта:**

   loadMIPSfpga.bat

Загрузка кода с помощью скрипта

1. **Открыть командною оболочку (cmd.exe)**
2. **Перейти в каталог Scripts:**

```
MIPSfpga\Codescape\ExamplePrograms\Scripts\Nexys4_DDR
```

3. **В оболочке ввести:**

```
loadMIPSfpga.bat
C:\MIPSfpga_Fundamentals\Lab02_C\ReadSwitches
```

Загрузка кода с помощью скрипта

1. **Открыть командною оболочку (cmd.exe)**
2. **Перейти в каталог Scripts:**

```
MIPSfpga\Codescape\ExamplePrograms\Scripts\Nexys4_DDR
```

3. **В оболочке ввести:**

```
loadMIPSfpga.bat
C:\MIPSfpga_Fundamentals\Lab02_C\ReadSwitches
```

Замечание: Параметром скрипта может быть любой каталог

Работы 2 & 3: Программирование

Программирование для MIPSfpga на языках C и ассемблер MIPS

- **Описание:**
  MIPSfpga_Fundamentals\LabInstructions
- **Дополнительные файлы:**
  MIPSfpga_Fundamentals\Lab02_C
  MIPSfpga_Fundamentals\Lab03_Assembly

Работа 4: Семисегментный индикатор

Цель: Добавить к системе MIPSfpga новое устройство ввода/вывода с отображением в память - семисегментный индикатор

Работа 4: Семисегментный индикатор

Цель: Добавить к системе MIPSfpga новое устройство ввода/вывода с отображением в память - семисегментный индикатор

Шаги:

1. Аппаратное подключение индикатора
2. Отображение на память разрядов и сигналов разрешения
3. Изменение интерфейса MIPSfpga для управления сегментами и сигналами разрешения

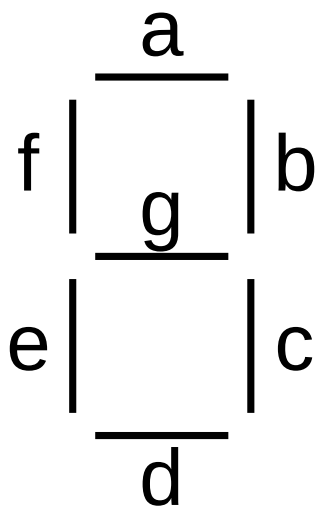
Работа 4: Семисегментный индикатор

Цель: Добавить к системе MIPSfpga новое устройство ввода/вывода с отображением в память - семисегментный индикатор

Шаги:

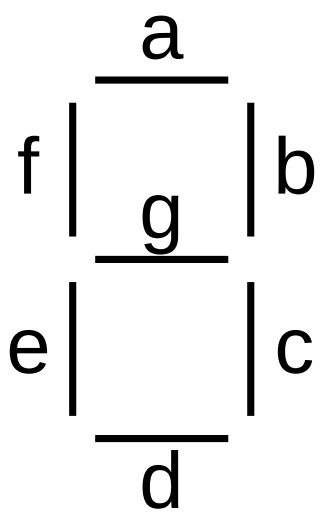
1. **Аппаратное подключение индикатора**
2. Отображение на память разрядов и сигналов разрешения
3. Изменение интерфейса MIPSfpga для управления сегментами и сигналами разрешения

Семисегментный индикатор



Выбор светящихся сегментов определяет выводимую цифру

Семисегментный индикатор



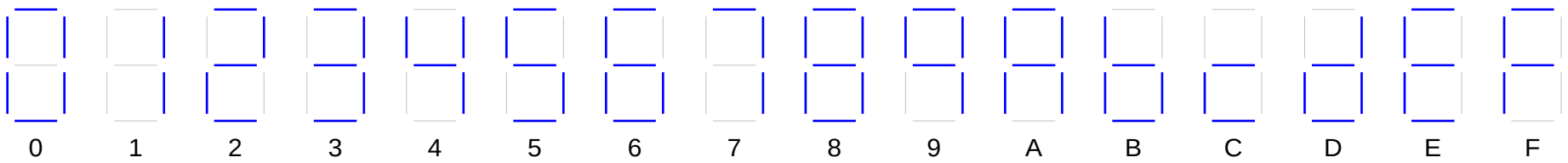
Например,

0 - сегменты: a,b,c,d,e,f

1 - сегменты: b,c

2 - сегменты: a,b,d,e,g

и так дплее



Семисегментный индикатор

```
module mipsfpga_ahb_sevensegdec(input      [3:0] data,
                                output reg [6:0] segments);
  always @(*)
    case(data)          // abc_defg
      4'h0: segments = 7'b000_0001;
      4'h1: segments = 7'b100_1111;
      4'h2: segments = 7'b001_0010;
      4'h3: segments = 7'b000_0110;
      4'h4: segments = 7'b100_1100;
      4'h5: segments = 7'b010_0100;
      4'h6: segments = 7'b010_0000;
      4'h7: segments = 7'b000_1111;
      4'h8: segments = 7'b000_0000;
      4'h9: segments = 7'b000_1100;
      4'ha: segments = 7'b000_1000;
      4'hb: segments = 7'b110_0000;
      4'hc: segments = 7'b111_0010;
      4'hd: segments = 7'b100_0010;
      4'he: segments = 7'b011_0000;
      4'hf: segments = 7'b011_1000;
      default:
            segments = 7'b111_1111;
    endcase
```

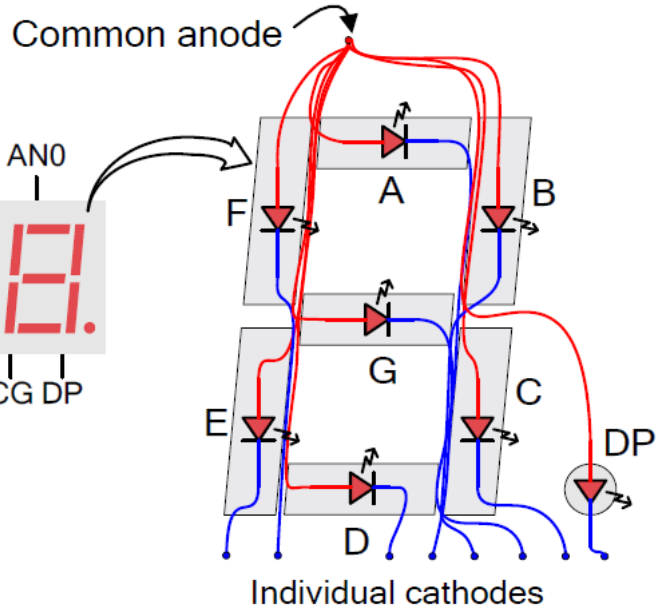
Сегменты активируются низким уровнем

Семисегментный индикатор Nexys4 DDR

- 8 семисегментных разрядов
- Все разряды соединены с общими входами сегментов (CA-CG)
- Сигналы разрешения (AN[7:0]) определяют какой разряд выводится
- AN[7:0] имеют активный низкий уровень

Eight-digit Seven Segment Display

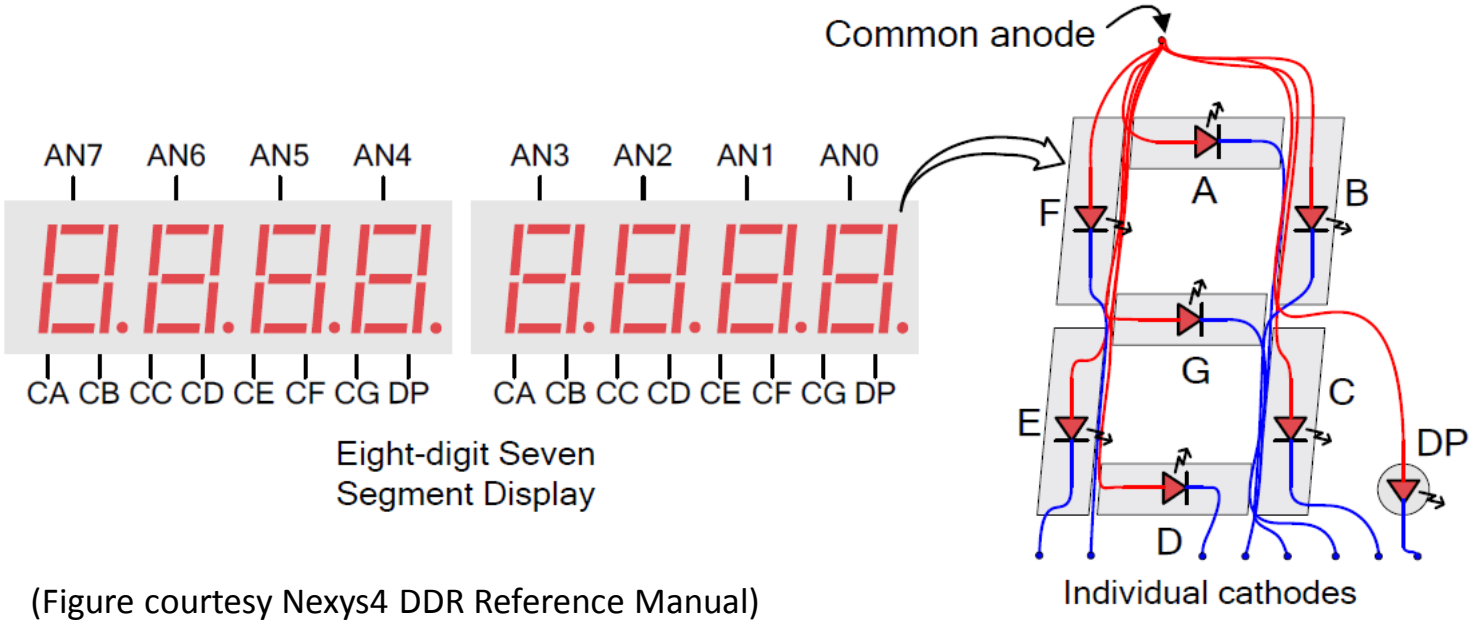
(Figure courtesy Nexys4 DDR Reference Manual)



Семисегментный индикатор Nexys4 DDR

Пример:

- AN[7:0] = 11111110_2 (только правый разряд светится)
- Выводимое значение определяется CA-CG

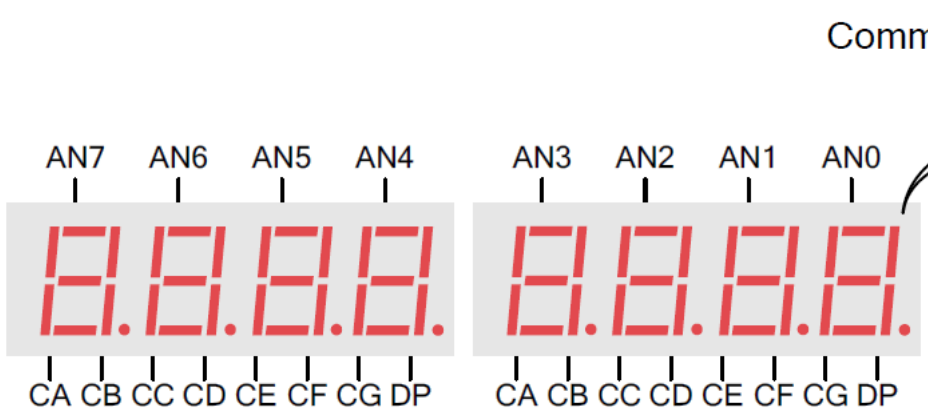


(Figure courtesy Nexys4 DDR Reference Manual)

Семисегментный индикатор Nexys4 DDR

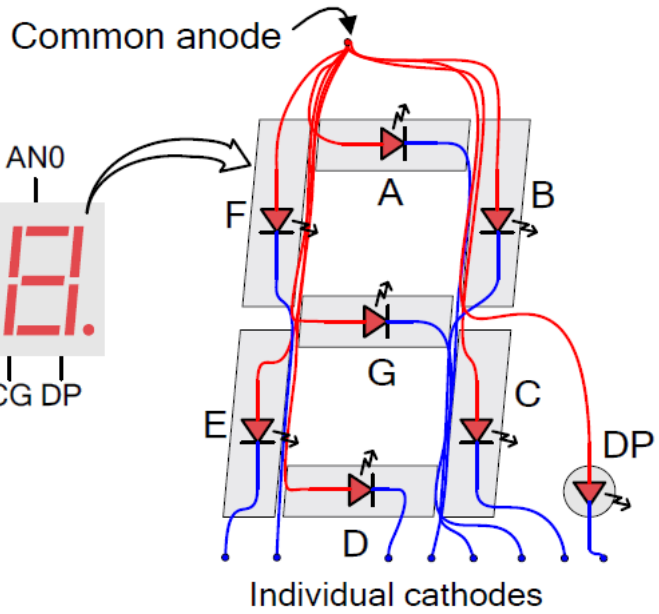
Для управления несколькими разрядами:

- За раз выводится один разряд
- Чтобы мерцание не было заметно, вывод разных разрядов происходит достаточно часто



Eight-digit Seven Segment Display

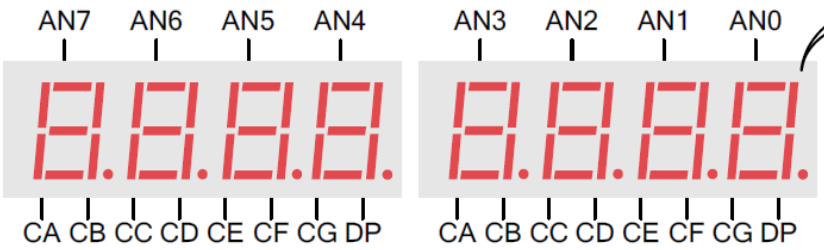
(Figure courtesy Nexys4 DDR Reference Manual)



Семисегментный индикатор Nexys4 DDR

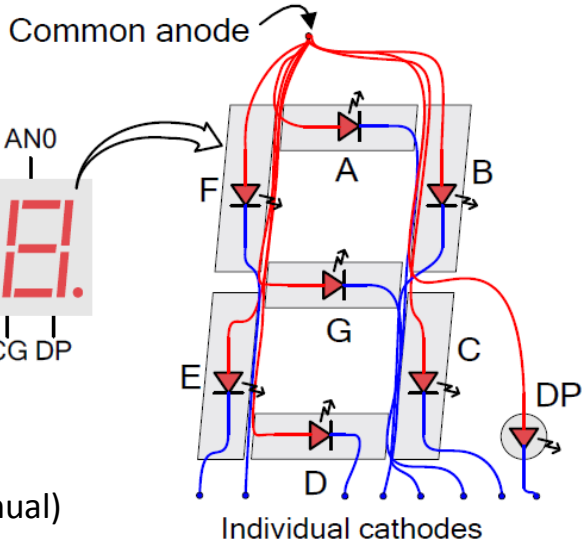
Каждые ~2 мс выводится следующий разряд:

- В t=0, AN[7:0] = 11111110, CA-CG для разряда 0
- В t=2ms, AN[7:0] = 11111101, CA-CG для разряда 1
- В t=4ms, AN[7:0] = 11111011, CA-CG для разряда 2
- И так далее…



Eight-digit Seven Segment Display

(Figure courtesy Nexys4 DDR Reference Manual)



Подключение индикатора Nexys4 DDR

Нужно создать 9 отображаемых в память регистров:

- 1 для хранения разрешения вывода разряда (SEGEN_N[7:0])
- 8 для хранения выводимого значения каждого разряда (SEG0_N[3:0], SEG1_N[3:0], … SEG7_N[3:0])

Подключение индикатора Nexys4 DDR

Нужно создать 9 отображаемых в память регистров:

- 1 для хранения разрешения вывода разряда (SEGEN_N[7:0])
- 8 для хранения выводимого значения каждого разряда (SEG0_N[3:0], SEG1_N[3:0], … SEG7_N[3:0])

Трехразрядный счетчик (работающий на частоте ~500 Гц, т.е. период = 2мс) последовательно выбирает все разряды и выводит их, если установлено разрешение вывода

Работа 4: Семисегментный индикатор

Цель: Добавить к системе MIPSfpga новое устройство ввода/вывода с отображением в память – семисегментный индикатор

Шаги:

1. Аппаратное подключение индикатора
2. **Отображение на память разрядов и сигналов разрешения**
3. Изменение интерфейса MIPSfpga для управления сегментами и сигналами разрешения

Отображение на память сигналов и разрядов

Регистр	Описание	Адрес памяти
SEGEN_N[7:0]	Разрешения	0xbf800010
SEG0_N[3:0]	Значение разряда 0	0xbf800014
SEG1_N[3:0]	Значение разряда 1	0xbf800018
SEG2_N[3:0]	Значение разряда 2	0xbf80001c
SEG3_N[3:0]	Значение разряда 3	0xbf800020
SEG4_N[3:0]	Значение разряда 4	0xbf800024
SEG5_N[3:0]	Значение разряда 5	0xbf800028
SEG6_N[3:0]	Значение разряда 6	0xbf80002c
SEG7_N[3:0]	Значение разряда 7	0xbf800030

Работа 4: Семисегментный индикатор

Цель: Добавить к системе MIPSfpga новое устройство ввода/вывода с отображением в память - семисегментный индикатор

Шаги:

1. Аппаратное подключение индикатора
2. Отображение на память разрядов и сигналов разрешения
3. **Изменение интерфейса MIPSfpga для управления сегментами и сигналами разрешения**

Интерфейс MIPSfpga

Выходы модуля GPIO и системы MIPSfpga:

```
...
output      [  7: 0] IO_7SEGEN_N,
output      [  6: 0] IO_7SEG_N
```

Интерфейс MIPSfpga

Выходы модуля GPIO и системы MIPSfpga:

```
    ...
    output      [  7: 0] IO_7SEGEN_N,
    output      [  6: 0] IO_7SEG_N
```

Модуль-оболочка Nexys4 DDR:

```
module mipsfpga_nexys4_ddr( ...
      output [ 7:0] AN,
      output        CA, CB, CC, CD, CE, CF, CG);
  ...
  mipsfpga_sys mipsfpga_sys(
                    ...
                    .IO_7SEGEN_N(AN),
                    .IO_7SEG_N({CA,CB,CC,CD,CE,CF,CG})
```

Интерфейс MIPSfpga: выводы Nexys4 DDR

MIPSfpga_Nexys4DDR.xdc:

```
set_property -dict { PACKAGE_PIN T10   IOSTANDARD
LVCMOS33 } [get_ports { CA }];
set_property -dict { PACKAGE_PIN R10   IOSTANDARD
LVCMOS33 } [get_ports { CB }];
...
set_property -dict { PACKAGE_PIN J17   IOSTANDARD
LVCMOS33 } [get_ports { AN[0] }];
set_property -dict { PACKAGE_PIN J18   IOSTANDARD
LVCMOS33 } [get_ports { AN[1] }];
...
```

Работа 9: Перенос на другие платы

Цель: Перенести MIPSfpga на другие FPGA платы

Зачем переносить на другие платы?

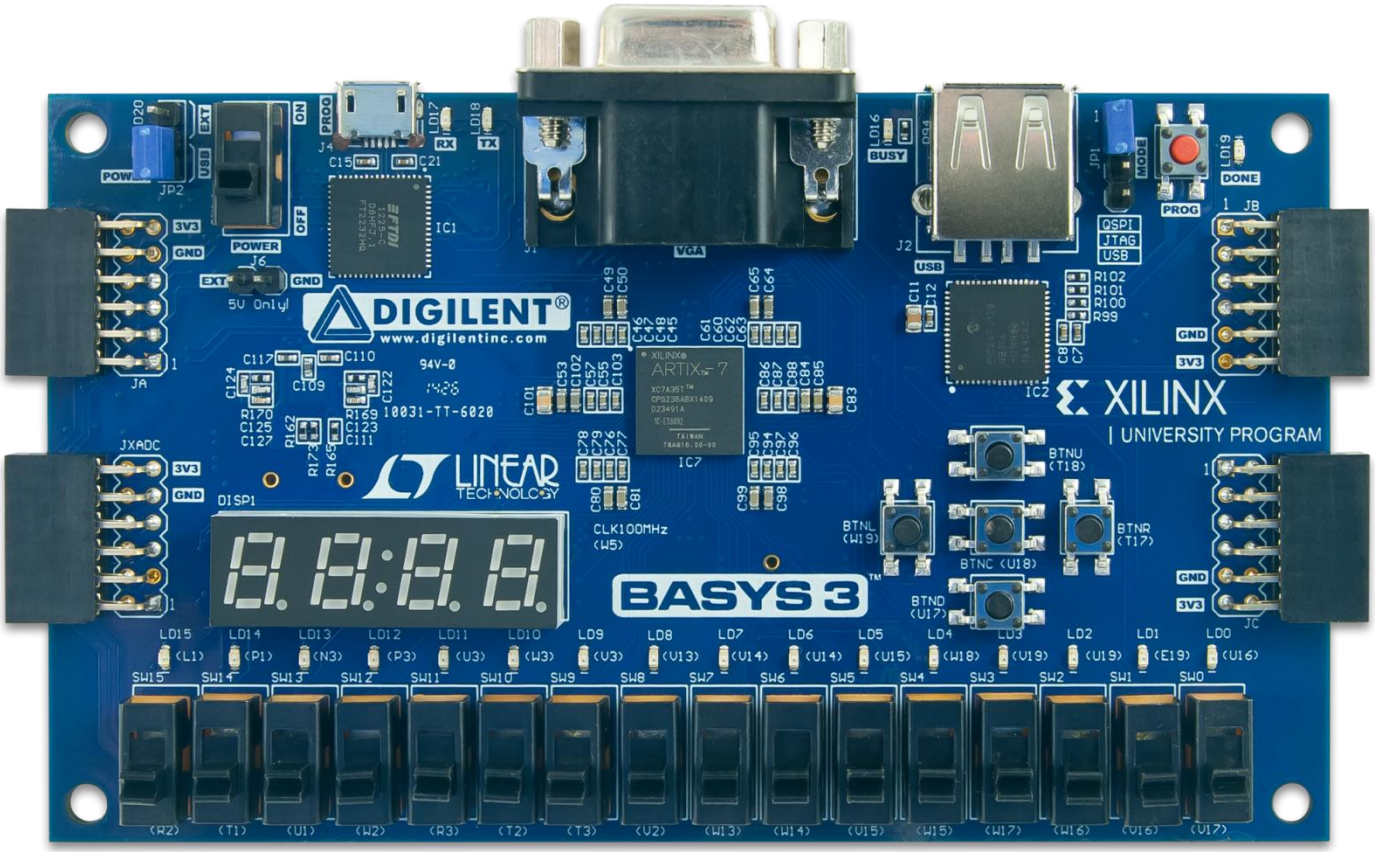
- Доступность
- Меньшая цена

Работа 9: Перенос на другие платы

Необходимые изменения:

- Модуль-оболочка
- Файл ограничений проекта Xilinx (.xdc): отображает сигналы ввода/вывода верхнего уровня на выводы FPGA
- Изменение объема памяти (возможно)

Пример: плата Basys3



Работа 9: Перенос на другие платы

Необходимые изменения:

- Модуль-оболочка
- Файл ограничений проекта Xilinx (.xdc): отображает сигналы ввода/вывода верхнего уровня на выводы FPGA
- Изменение объема памяти (возможно)

Сравнение плат Basys3 и Nexys4 DDR

	Basys3	Nexys4 DDR
Цена	$79 (для университетов), $149	$159 (для университетов), $320
FPGA	Artix-7 (XC7A35T-CPG236C)	Artix-7 (XC7A100T-1CSG324C)
Память (блоковое ОЗУ)	225 KB	607 KB
Логических элементов	33k	101k
Семисегментных индикаторов	4	8

Basys3: Модуль-оболочка

```
module mipsfpga_basys3(
   input          clk,
   input          btnU, btnD, btnL, btnR, btnC,
   input  [15:0] sw,
   output [15:0] led,
   inout  [ 5:0] JB,
   output [ 3:0] an,
   output [ 0:6] seg
);
```

Nexys4 DDR: Модуль-оболочка

```
module mipsfpga_nexys4_ddr(
   input          CLK100MHZ,
   input          CPU_RESETN,
   input          BTNU, BTND, BTNL, BTNR, BTNC,
   input  [15:0] SW,
   output [15:0] LED,
   inout  [ 8:1] JB,
   output [ 7:0] AN,
   output         CA, CB, CC, CD, CE, CF, CG
);
```

Модули-оболочки

```
module mipsfpga_nexys4_ddr(
   input          CLK100MHZ,
   input          CPU_RESETN,
   input          BTNU, BTND, BTNL, BTNR, BTNC,
   input  [15:0] SW,
   output [15:0] LED,
   inout  [ 8:1] JB,
   output [ 7:0] AN,
   output         CA, CB, CC, CD, CE, CF, CG );

module mipsfpga_basys3(
   input          clk,
   input          btnU, btnD, btnL, btnR, btnC,
   input  [15:0] sw,
   output [15:0] led,
   inout  [ 5:0] JB,
   output [ 3:0] an,
   output [ 0:6] seg );
```

Работа 9: Перенос на другие платы

Необходимые изменения:

- Модуль-оболочка
- Файл ограничений проекта Xilinx (.xdc): отображает сигналы ввода/вывода верхнего уровня на выводы FPGA
- Изменение объема памяти (возможно)

Файл ограничений

mipsfpga_basys3.xdc:

```
set_property PACKAGE_PIN V17 [get_ports {sw[0]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[0]}]
set_property PACKAGE_PIN V16 [get_ports {sw[1]}]
set_property IOSTANDARD LVCMOS33 [get_ports {sw[1]}]
set_property PACKAGE_PIN W16 [get_ports {sw[2]}]
...
```

Работа 9: Перенос на другие платы

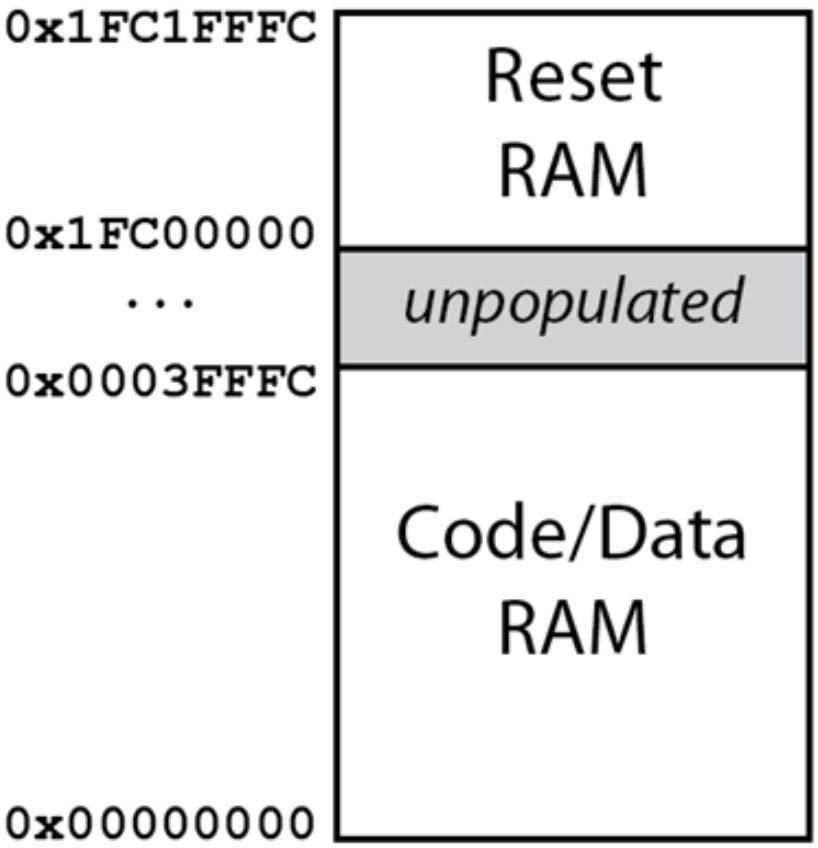
Необходимые изменения:

- Модуль-оболочка
- Файл ограничений проекта Xilinx (.xdc): отображает сигналы ввода/вывода верхнего уровня на выводы FPGA
- Изменение объема памяти (возможно)

Сравнение плат Basys3 и Nexys4

	Basys3	Nexys4 DDR
Цена	$79 (для университетов), $149	$159 (для университетов), $320
FPGA	Artix-7 (XC7A35T-CPG236C)	Artix-7 (XC7A100T-1CSG324C)
Память (блоковое ОЗУ)	225 KB	607 KB
Логических элементов	33k	101k
Семисегментных индикаторов	4	8

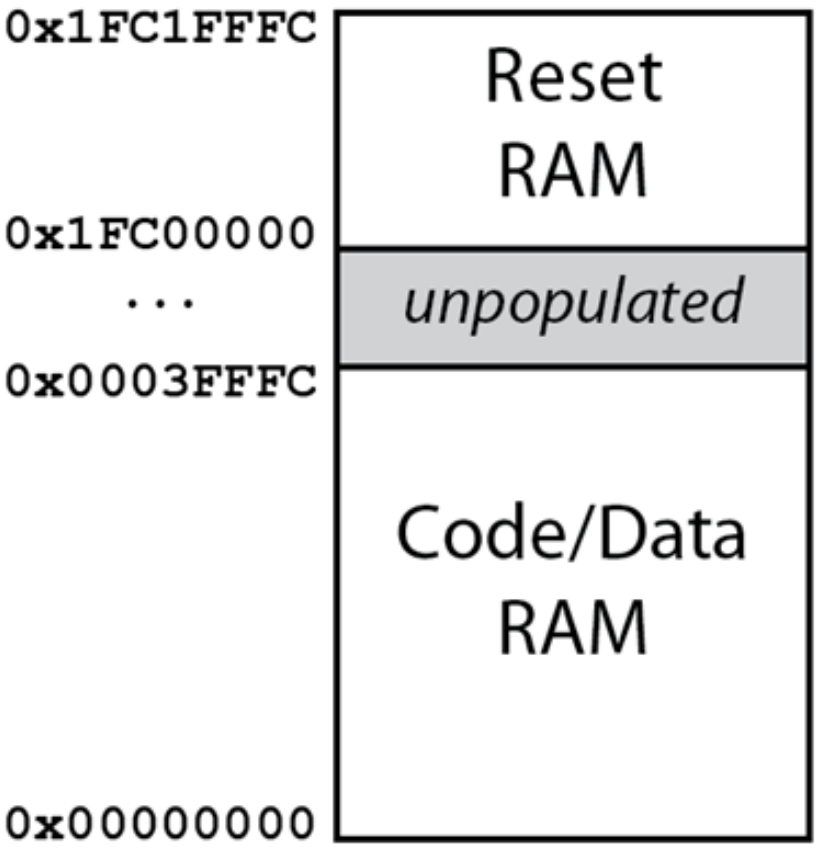
Система MIPSfpga: Физическая память



Загрузочный код:
Выполняется после включения (128 KB)

Пользовательский код/данные
(256 KB)

Система MIPSfpga: Физическая память



Загрузочный код: Выполняется после включения (128 KB)

Пользовательский код/данные (256 KB)

Не помещается на Basys3: объем памяти всего 225 KB

Уменьшение размера памяти

Загрузочный код: 32 KB (2^{15} байт= 2^{13} слов)

Пользовательский код: 64 KB (2^{16} байт = 2^{14} слов)

Уменьшение размера памяти

Загрузочный код: 32 KB (2^{15} байт= 2^{13} слов)

Пользовательский код: 64 KB (2^{16} байт = 2^{14} слов)

mipsfpga_ahb_const.vh:

```
`define H_RAM_RESET_ADDR_WIDTH(13)
`define H_RAM_ADDR_WIDTH      (14)
```

Использование MIPSfpga

- **Отличное средство для курсов по:**
  - Цифровой схемотехнике
  - Архитектуре компьютеров
  - Встроенным системам
  - СБИС
  - Проектированию систем на кристалле
- **Также может применяться в научных исследованиях**

Учебные материалы MIPSfpga

Доступны на сайте Университетской программы компании Imagination в разделе Teaching Materials:

http://community.imgtec.com/university/university-registration

Поддержка

Форум MIPSfpga(техническая поддержка):

http://community.imgtec.com/forums/cat/mips-insider/mipsfpga/

Другие форумы

- **Техническая поддержка MIPS (общие вопросы):**

  http://community.imgtec.com/forums/cat/mips-insider/

- **Университетская программа Imagination (обсуждения учебных планов, вопросов о IUP):**

  http://community.imgtec.com/forums/cat/university/

Благодарности

- Robert Owen
- Sarah Harris
- David Money Harris
- Yuri Panchul
- Bruce Ableidinger
- Kent Brinkley
- Chuck Swartley
- Sean Raby
- Rick Leatherman
- Matthew Fortune
- Munir Hasan
- Sachin Sundar
- Michael Alexander
- Sam Bobrowicz
- Larissa Swanland
- Clint Cole
- Students and faculty at UCL
- Ian Oliver
- Steve Kromer
- Daniel Martinez
- Parimal Patel and Jason Wong

These materials produced in association with Imagination.
Join our University community for more resources.
community.imgtec.com/university