Open source проекты как инженерные решения на примере компании «Инструментальные Системы»

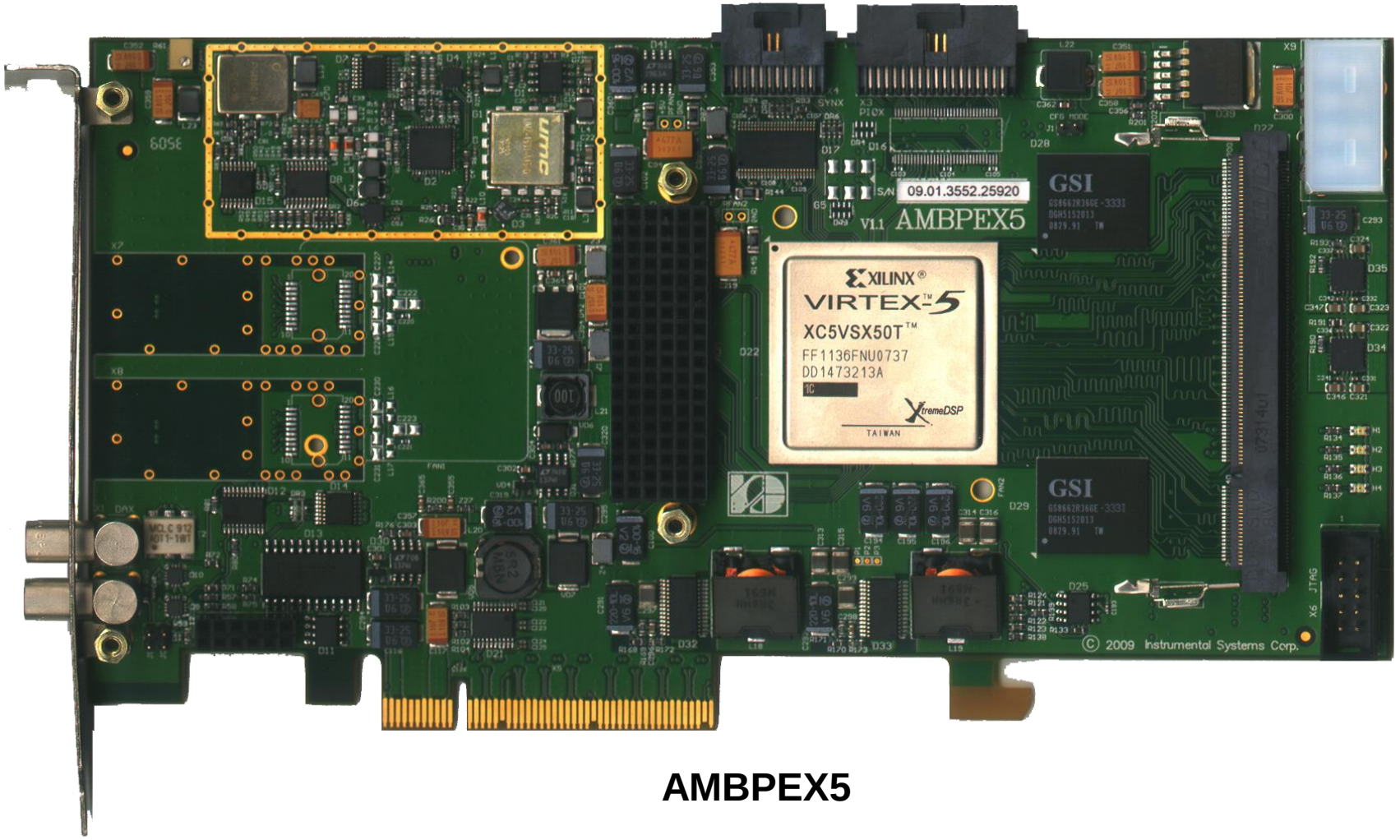
Дмитрий Смехов

Высшая школа экономики

ADM классической реализации

ADP60PCI - пять процессоров ADSP-21062
1996 год

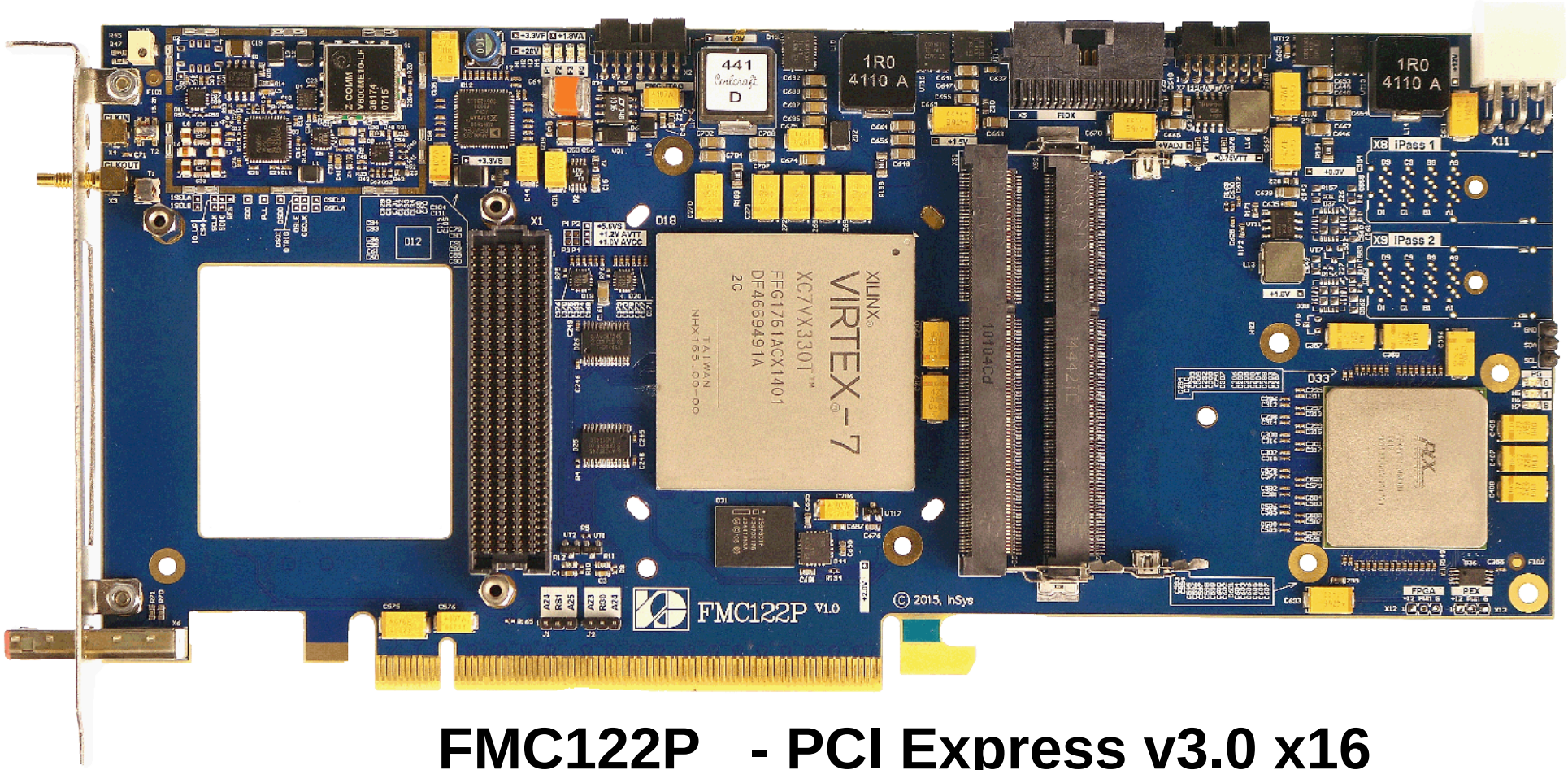
ADM прогрессивной реализации



AMBPEX5

2009 год

FMC – несущий модуль



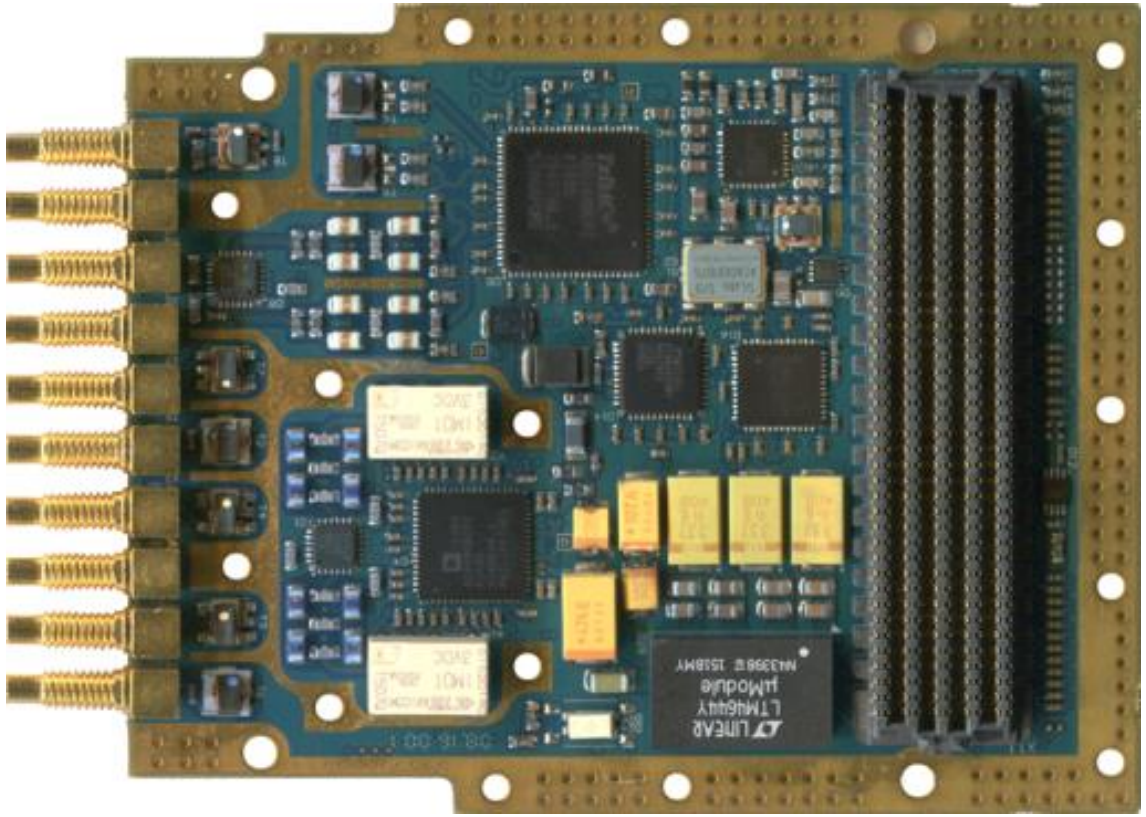
FMC122P - PCI Express v3.0 x16

2016 год 13000 Мбайт/с

FMC – субмодуль

FM214x1GTRF

- АЦП 1 ГГц
- ЦАП 2,8 ГГц
- Квадратурный модулятор
- Квадратурный демодулятор
- DDC



PCIe_DS_DMA

Контроллер шины PCI Express для ПЛИС Virtex 5, Virtex 6, Spartan 6, Artix 7

http://opencores.org/project,pcie_ds_dma

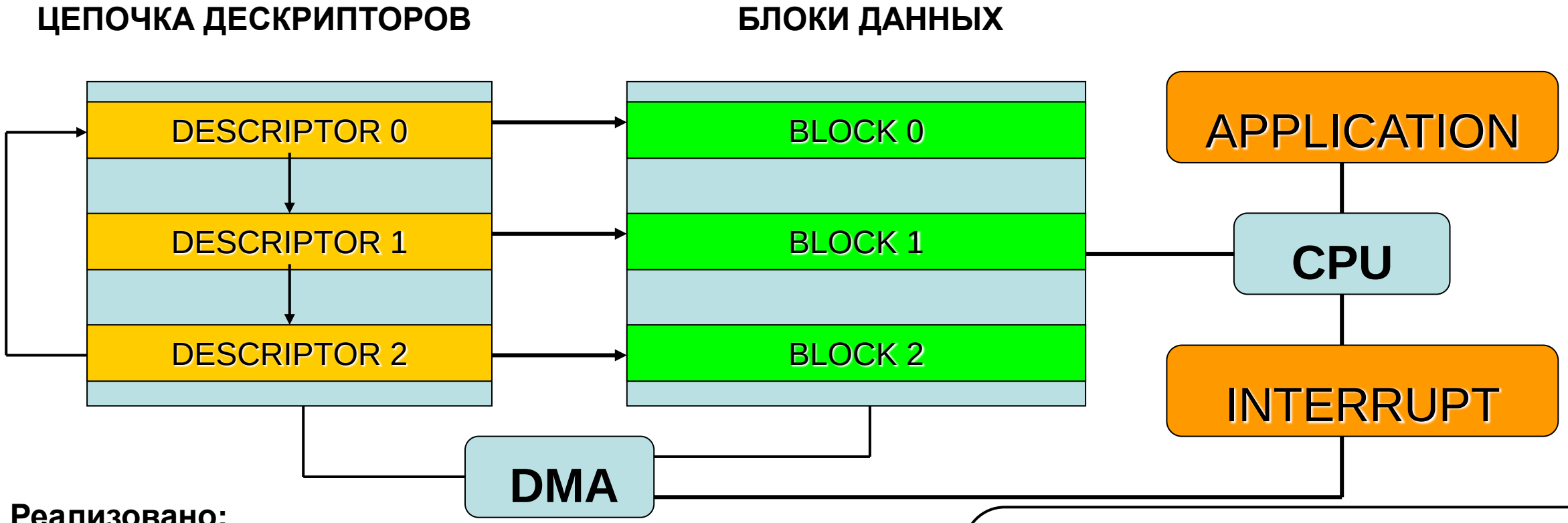
Разработчик: Дмитрий Смехов

Область применения:

- Обмен по PCI Express

Категория: передача данных

SCATTER-GATHER MODE

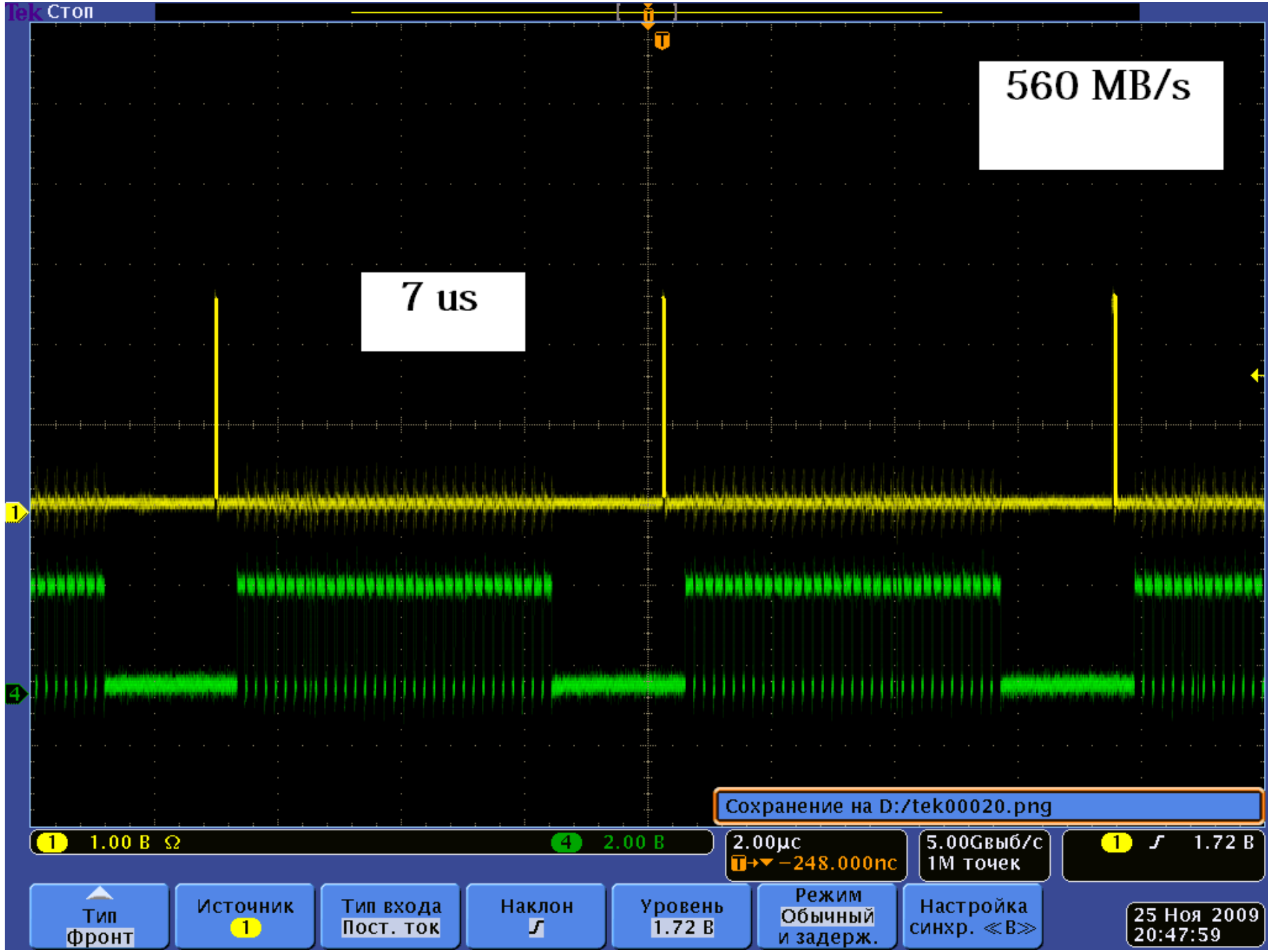


Реализовано:

1. **PCI9056, PEX8311 : PLX Technology**
2. **EZDMA : PLD Application**
3. **IP Core from Nortwest Logic**

ВНИМАНИЕ !!!

БЛОКИ ДАННЫХ ДОЛЖНЫ БЫТЬ НЕПРЕРЫВНЫМИ ПО ФИЗИЧЕСКИМ АДРЕСАМ


Tek Стоп
560 MB/s
7 us
Сохранение на D:/tek00020.png
1 1.00 В Ω
4 2.00 В
2.00µс
T→▼ −248.000nc
5.00Gвыб/с
1M точек
1 ƒ 1.72 В
Тип
Фронт
Источник
1
Тип входа
Пост. ток
Наклон
Уровень
1.72 В
Режим
Обычный
и задерж.
Настройка
синхр. ≪В≫
25 Ноя 2009
20:47:59

Блок дескрипторов

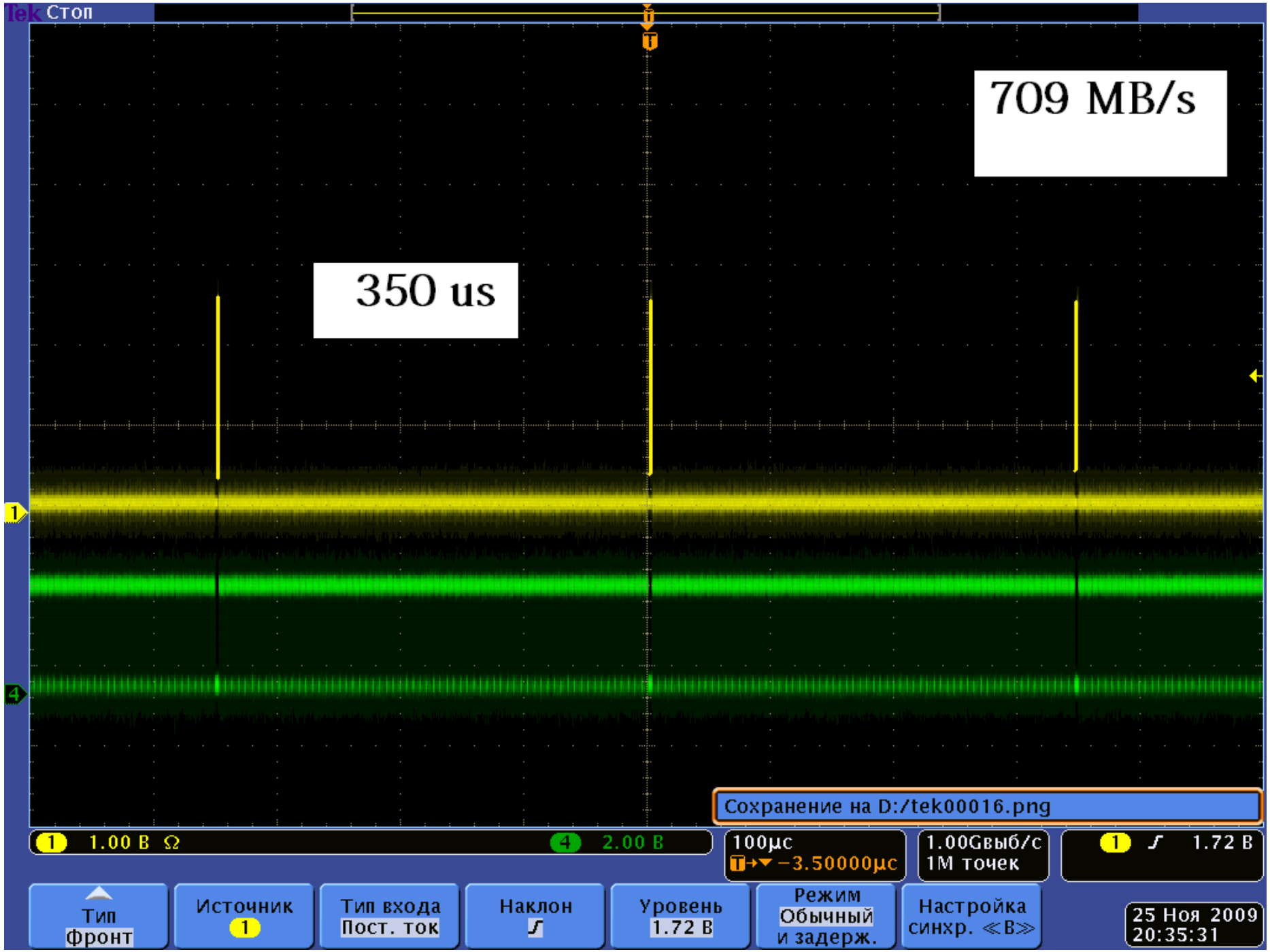
Отдельные дескрипторы объединяются в блок дескрипторов

0	SIZE0	ADR0
1	SIZE1	ADR1
2	SIZE2	ADR2
…		
62	SIZE62	ADR62
63	CRC / SIG	NEXT

Размер блока 512 байт

CRC и SIG защищают систему от ошибок

Размер блока данных кратен 4К


Tek Стоп
709 MB/s
350 us
Сохранение на D:/tek00016.png
1 1.00 В Ω
4 2.00 В
100µс
T→▼ −3.50000µс
1.00Gвыб/с
1M точек
1 ⎍ 1.72 В
Тип
Фронт
Источник
1
Тип входа
Пост. ток
Наклон
Уровень
1.72 В
Режим
Обычный
и задерж.
Настройка
синхр. ≪В≫
25 Ноя 2009
20:35:31

Результаты

Год	Модуль	ПЛИС	PCIe	Скорость Мбайт/с
2008	AMBPEX8	Virtex 4	v1.1 x4	800
2009	ADP201x1	Virtex 5	v1.1 x8	1600
2012	FMC106P	Virtex 6	v2.0 x8	3200
2017	FMC126P	KU	v3.0 x8	6200
2016	FMC122P	Virtex 7	v3.0 x16	13000

PROTEQ

Протокол связи по оптической линии для ПЛИС Virtex 6, Kintex 7

http://ds-dev.ru/projects/proteq

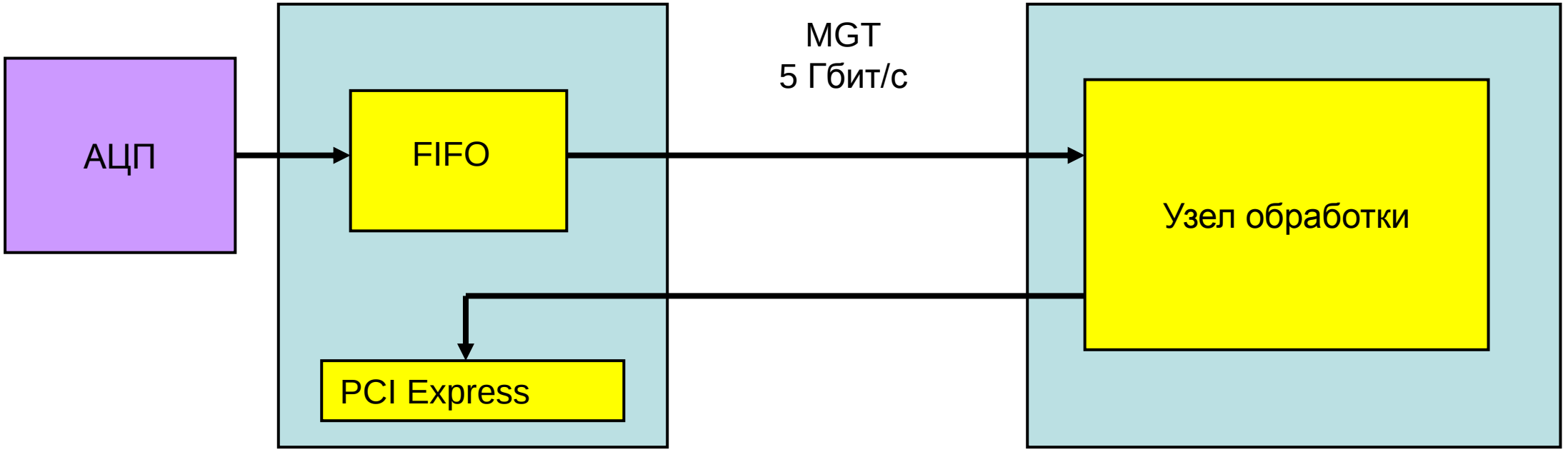
Разработчик: Дмитрий Смехов

Область применения:

- Обмен между ПЛИС внутри модуля
- Обмен между модулями

Категория: передача данных

Типовая задача



Задача – передача данных между двумя ПЛИС

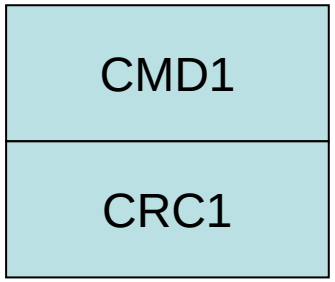
Особенности: **Источник данных АЦП – не может приостанавливаться**

Может быть разное число линий: 1 – 8

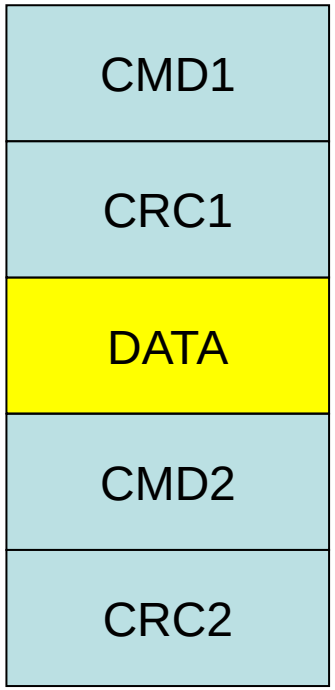
В линии возможны ошибки

Формат пакета

Служебный пакет



Пакет с данными



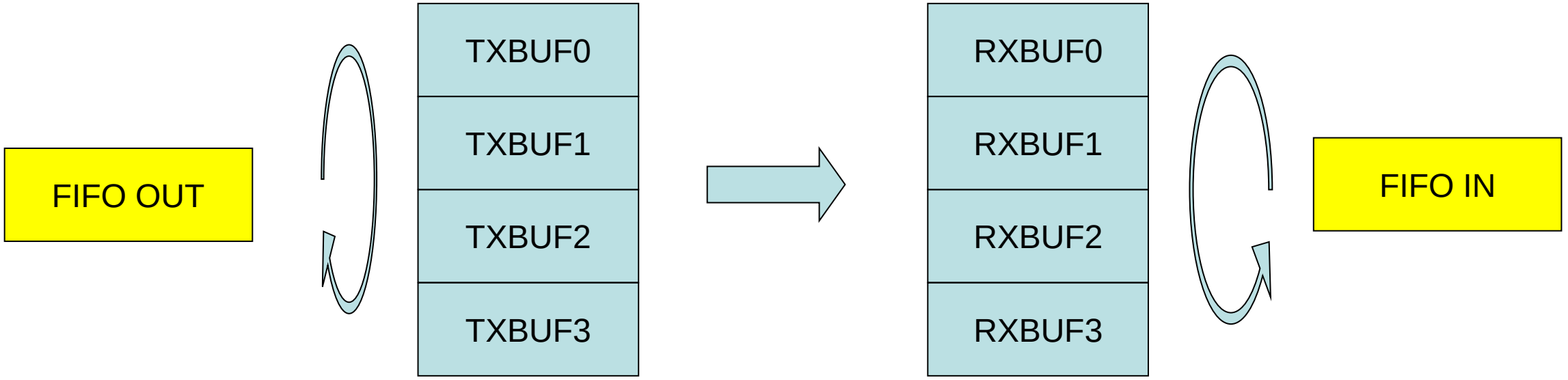
Длина пакета: 256x32

Эффективность:
256/260 = 98%

С учётом кодировки 64/67
(64/67)*(256/260) = 94 %

Главная особенность:
две контрольные суммы

Организация линка



250 МГц **156.25 МГц** **156.25 МГц** **250 МГц**

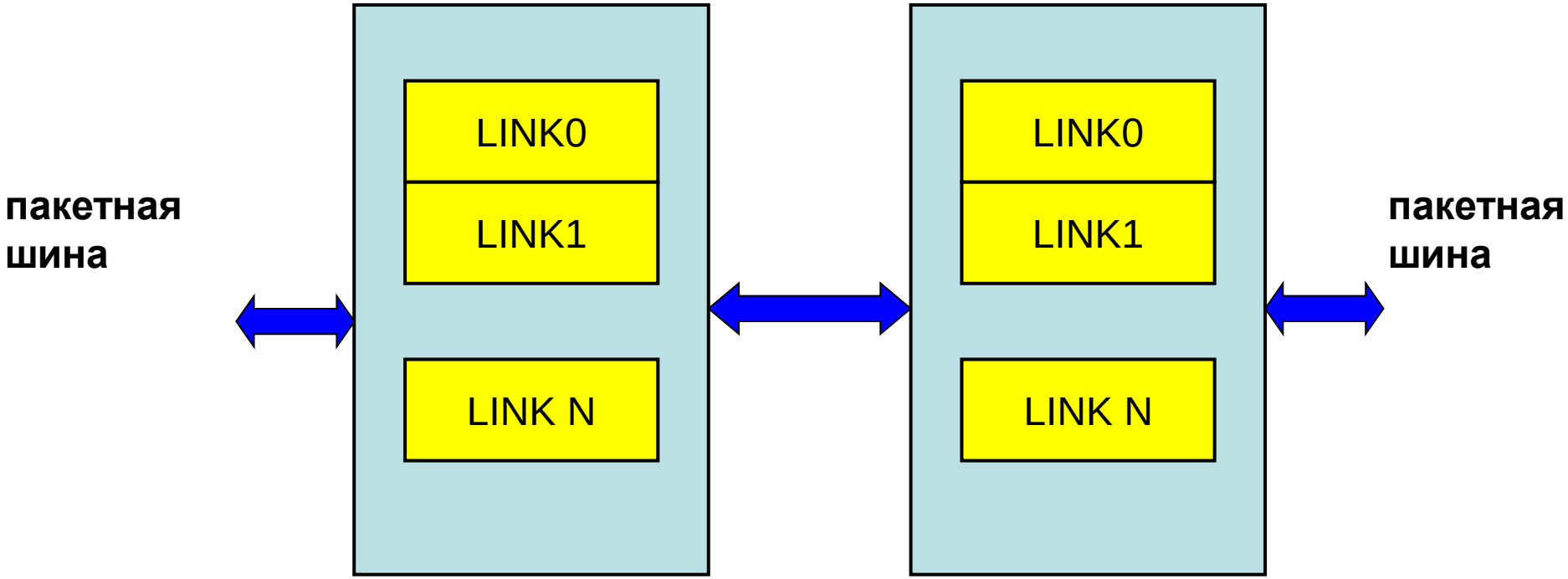
Информация о состоянии RXBUF

Запись производится по порядку

Передача происходит из занятого буфера

Передача в FIFO происходит в порядке очерёдности

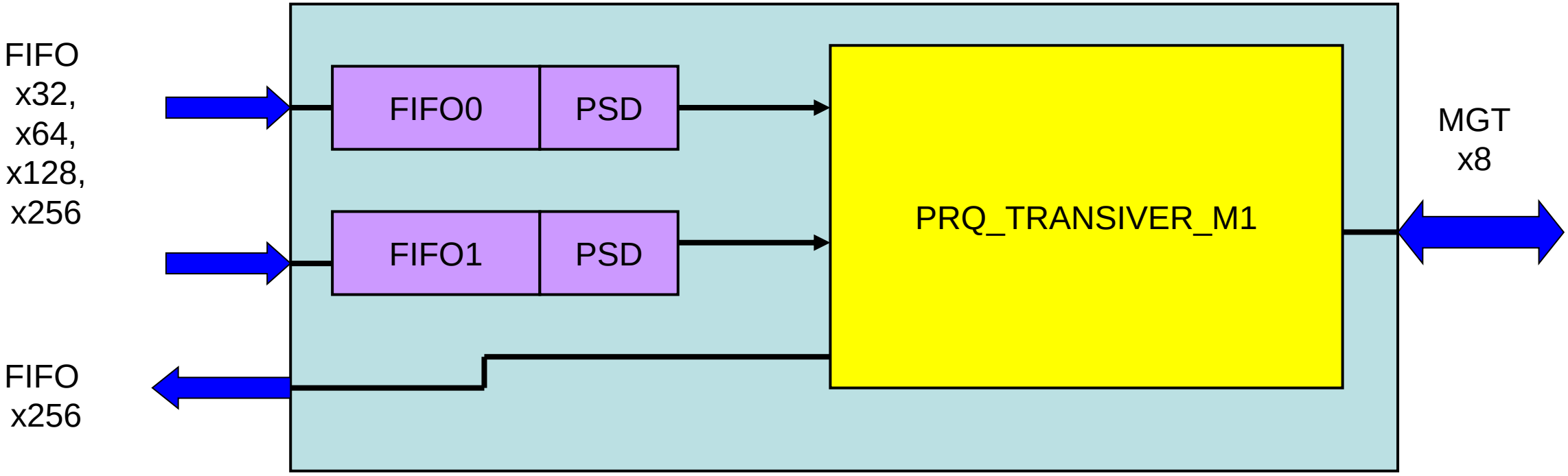
PRQ_TRANSIVER_M1



Обмен происходит
когда все линки готовы

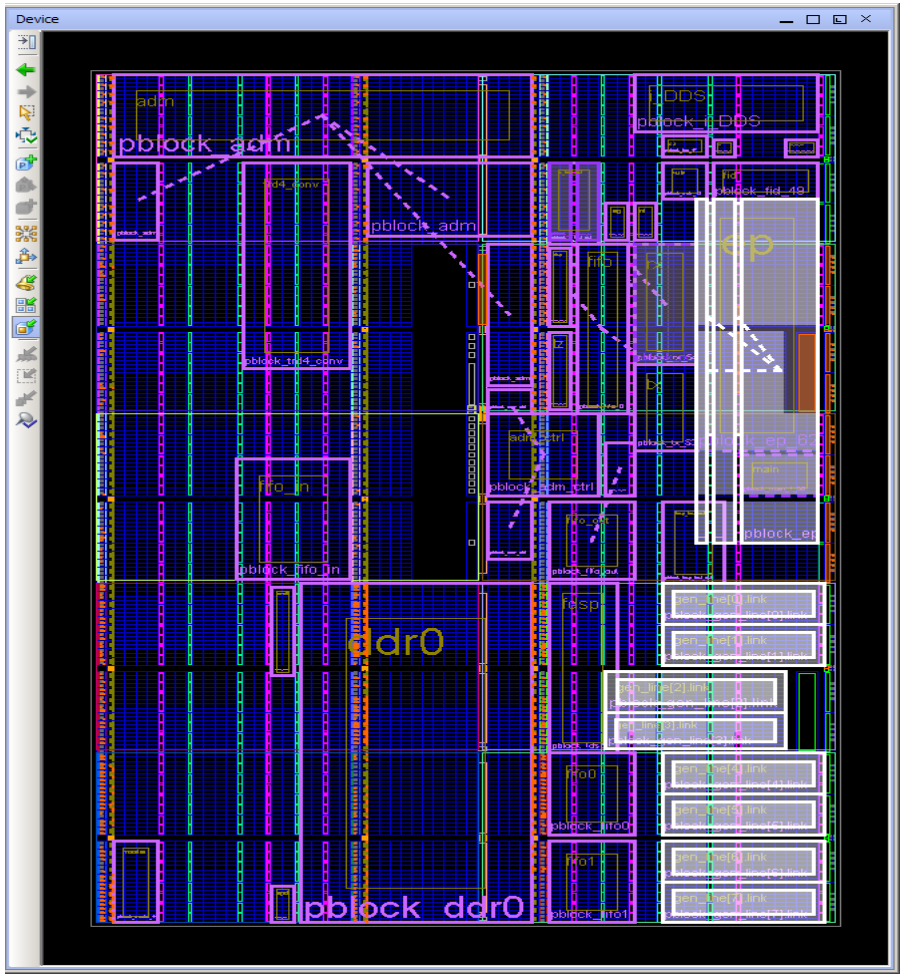
Ширина внутренней шины:
x1 – 32 разряда
x2 – 64 разряда
....
x8 – 256 разрядов

PRQ_CONNECT_M1



Готовый компонент для обмена через восемь линий MGT
Содержит два входных FIFO с переменным размером и шириной входной шины.
Содержит узел формирования 256-ти разрядной тестовой последовательности.

FMC106P – размещение на ПЛИС



Virtex 6: LX130T-2

PCI Express x8
заполнение 43%

PROTEQ x8
заполнение 88%

Размещение компонентов внутри ПЛИС в программе PlanAhead

Результаты

Протокол	Скорость	Эффективность
PROTEQ x8 5 Гбит/с	4484 Мбайт/с	94%
PCI Express x8 5 Гбит/с	3200 Мбайт/с	67%
FOTR x1 6.5 Гбит/с	566 Мбайт/с	73%

FP23FFTK

Вычисление БПФ и ОБПФ на ПЛИС в специальном формате числа с плавающей точкой

https://github.com/capitanov/fp23fftk

Разработчик: Александр Капитанов

Область применения:

- Вычисления с плавающей точкой на ПЛИС
- БПФ и ОБПФ размером 16К – 256К

Категория: вычисления

IP Core - Xilinx

Стандартные решения от Xilinx/Altera:

- Длина преобразования N = 8-64K,
- Конвейерная и последовательная схемы обработки,
- Целочисленный формат и формат float IEEE-754,
- Большой входной буфер,
- **Входные данные в натуральном порядке!**
- **Выходные данные в двоично-инверсном порядке!**

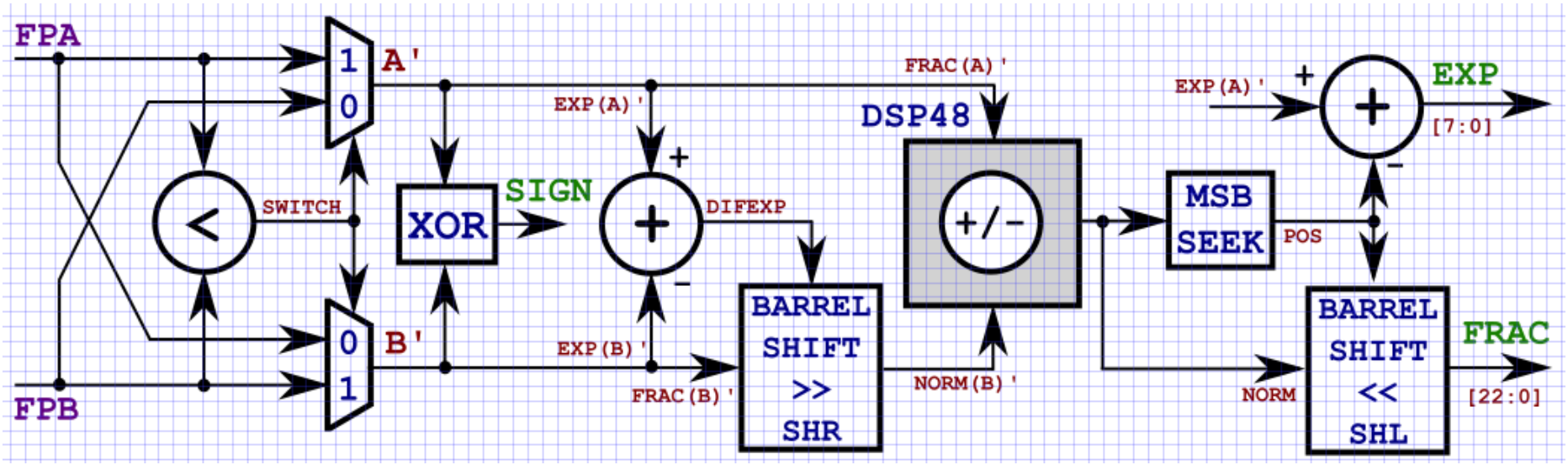
IP Core – FP23FFTK

- Длина преобразования N = 8-256K
- Конвейерная схема обработки
- Оптимизированный формат **floating point,** размер числа 23 бита
- **Для ОБПФ инверсный порядок!**

IP Core	*NFFT*	*1K*	*2K*	*4K*	*8K*	*16K*	*32K*	*64K*
FP23FFTK	*DSP48E*	*40*	*44*	*48*	*55*	*62*	*69*	*76*
	RAMB18E	*2*	*9*	*18*	*33*	*58*	*107*	*202*
Xilinx IP Core	*DSP48E*	*94*	*114*	*118*	*138*	*142*	*162*	*166*
	RAMB18E (1)	*12*	*22*	*36*	*66*	*123*	*242*	*478*
	RAMB18E (2)	*17*	*31*	*54*	*103*	*198*	*396*	*794*

В ~2.5 раза лучше!

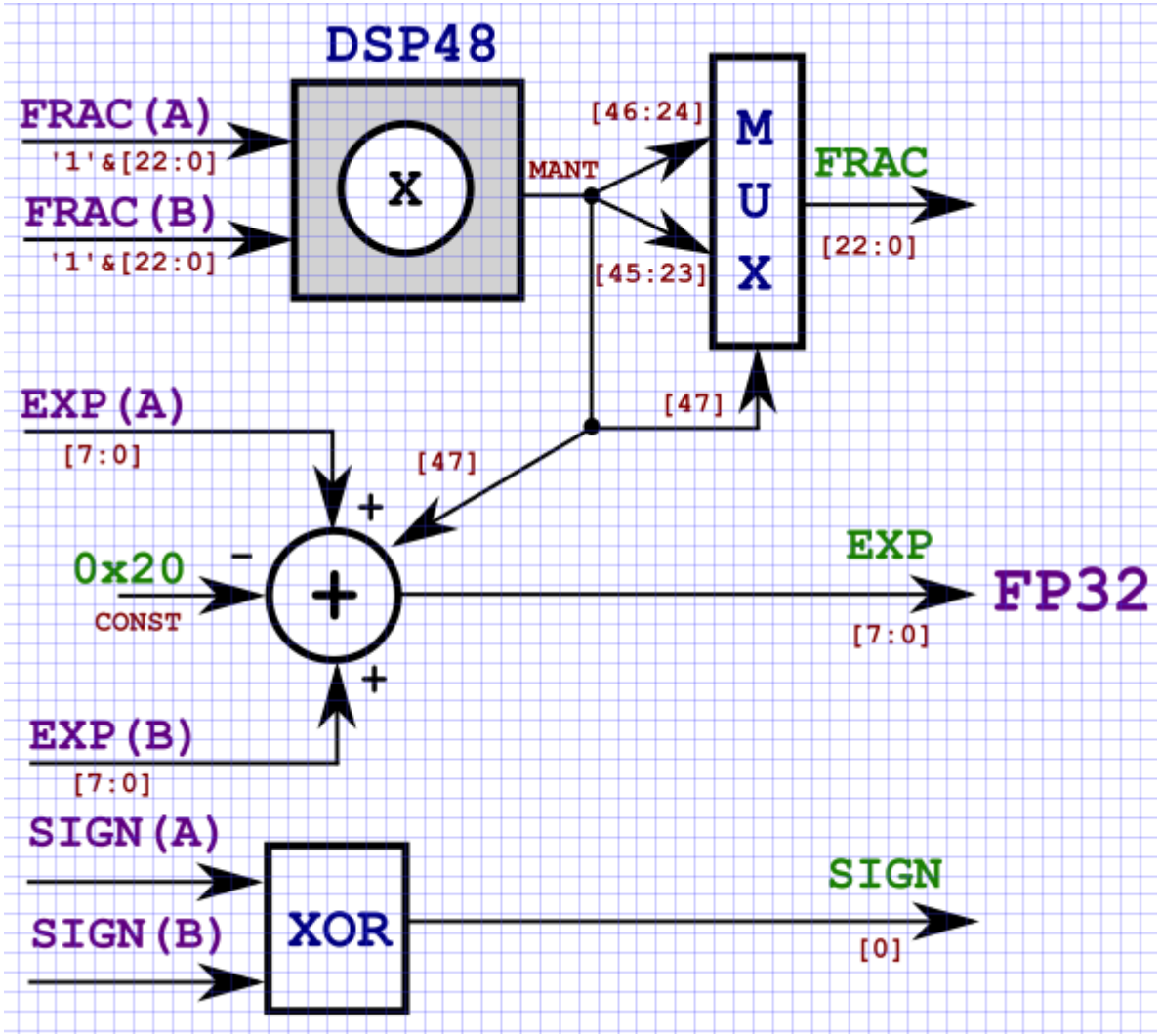
Сумматор



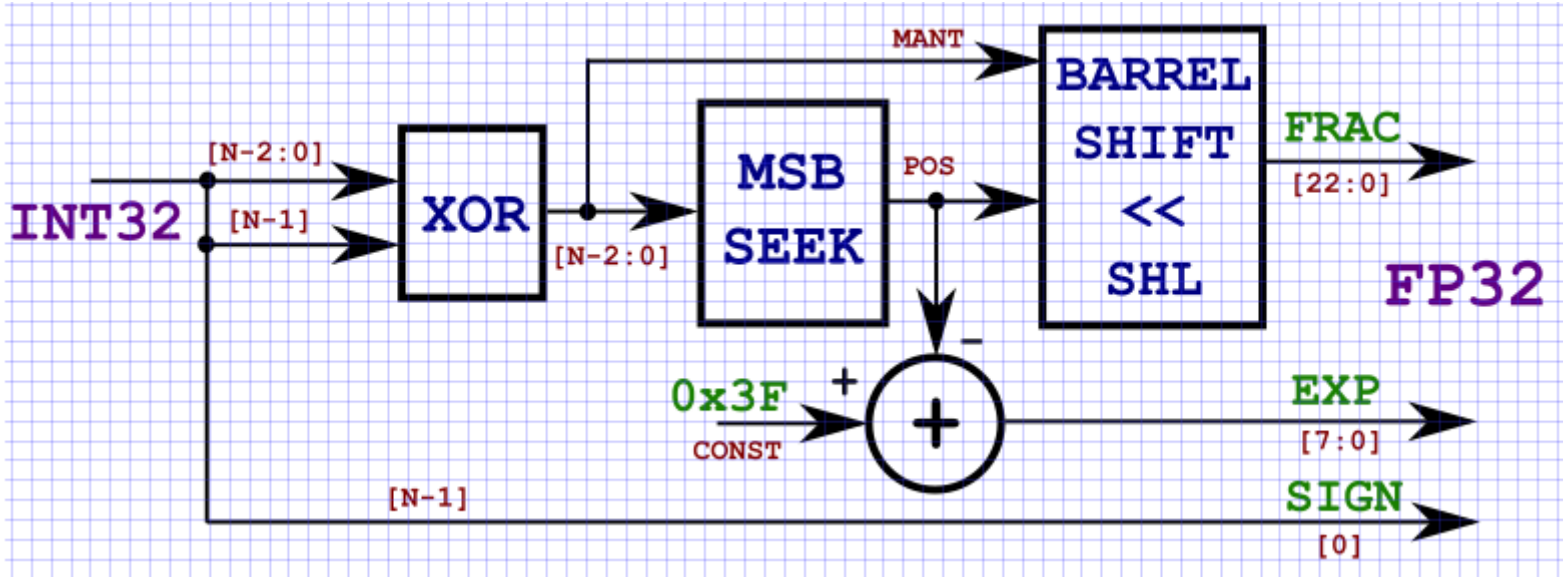
Latency	LUTs	SRL16E	FDRE	DSP48	CARRY8
9	314	36	340	1	11

Умножитель

LATENCY	5
LUTs	47
SRL16E	19
FDRE	92
DSP48	2
CARRY8	1



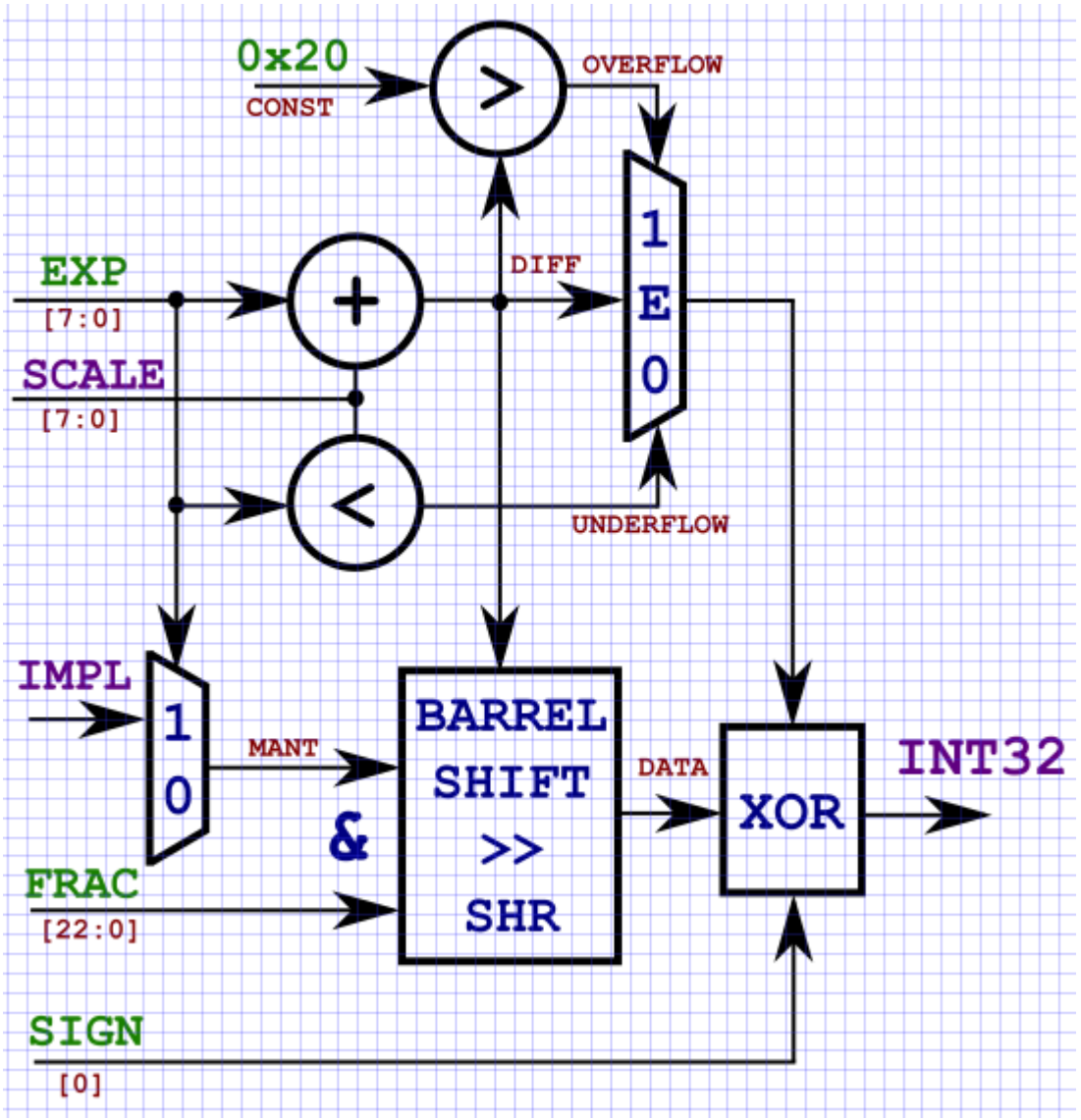
INT32 to FLOAT



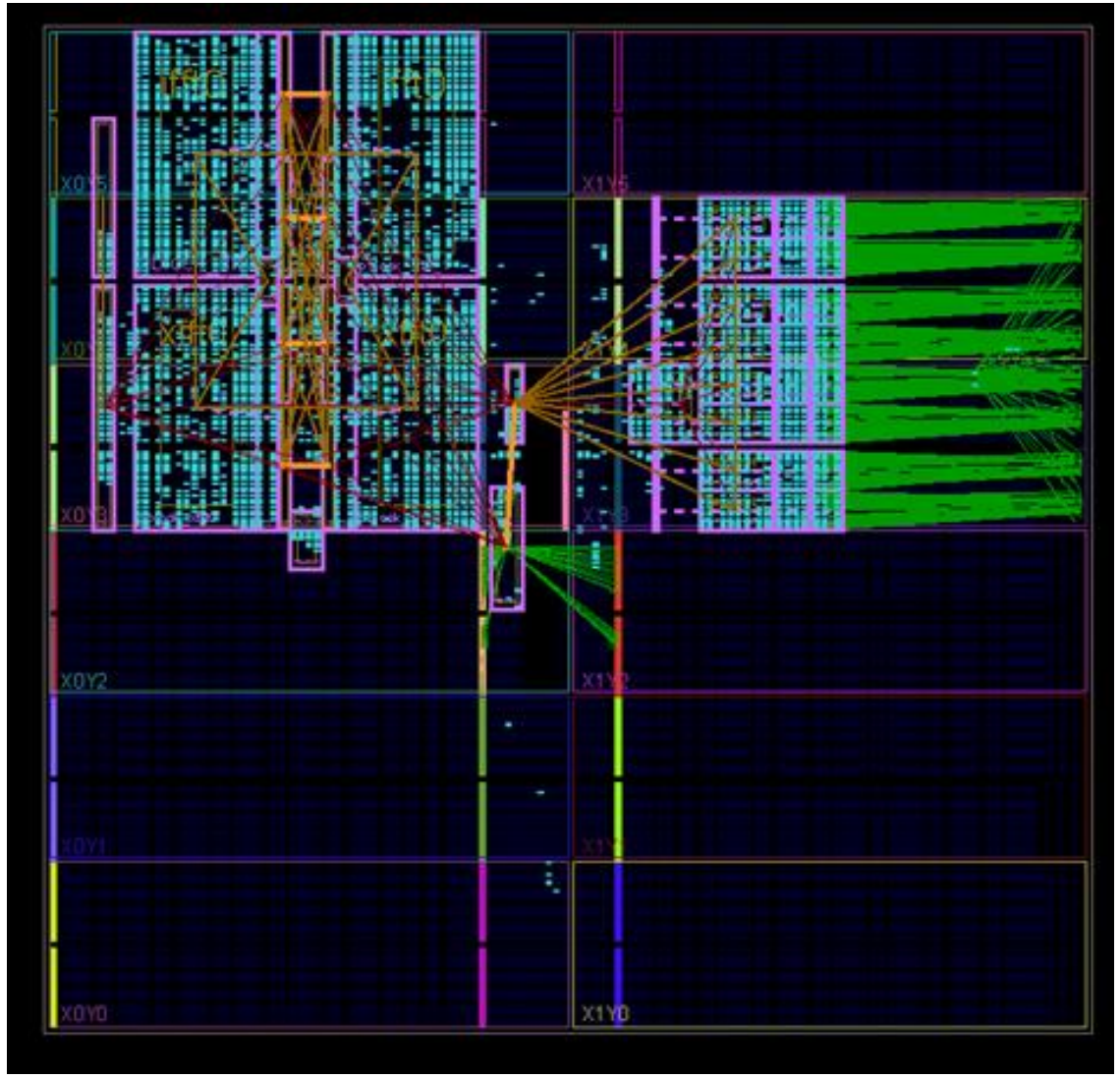
Latency	LUTs	SRL16E	FDRE	DSP48	CARRY8
6	183	1	196	0	4

FLOAT to INT32

LATENCY	4
LUTs	182
SRL16E	1
FDRE	145
DSP48	0
CARRY8	6



Быстрая свёртка



ПЛИС: Virtex-6 SX315T
(~1300 RAMB, ~1400 DSP48)

- 4x **FP23FFTK**, NFFT = 16K

или

- 4x FFT Xilinx NFFT = 8K

- До x16 **FP23FFTK**, N = 16K (!!!)
  ~ 8 каналов фильтров сжатия

SDAccel examples

Пример разработки проекта для ПЛИС Xilinx с использованием OpenCL.

https://github.com/dsmv/sdaccel

Разработчик: Дмитрий Смехов

Область применения:

- OpenCL
- Разработка проектов ПЛИС на языках Си / C++

Категория: разработка проектов ПЛИС

Модель вычислителя

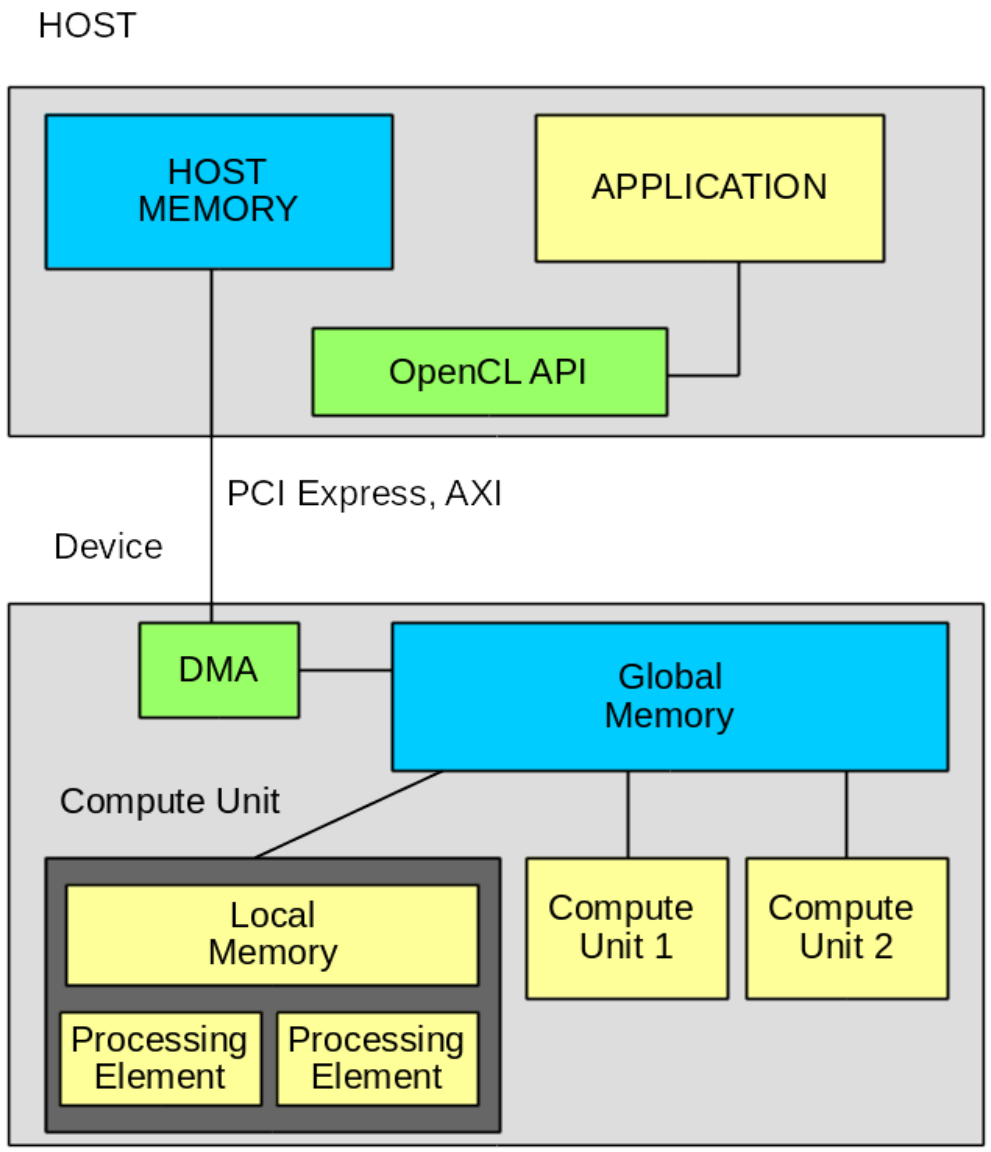
- **Структура памяти:**

  - Глобальная память
    DDR3, DDR4, SODIMM
    Доступ с обеих сторон

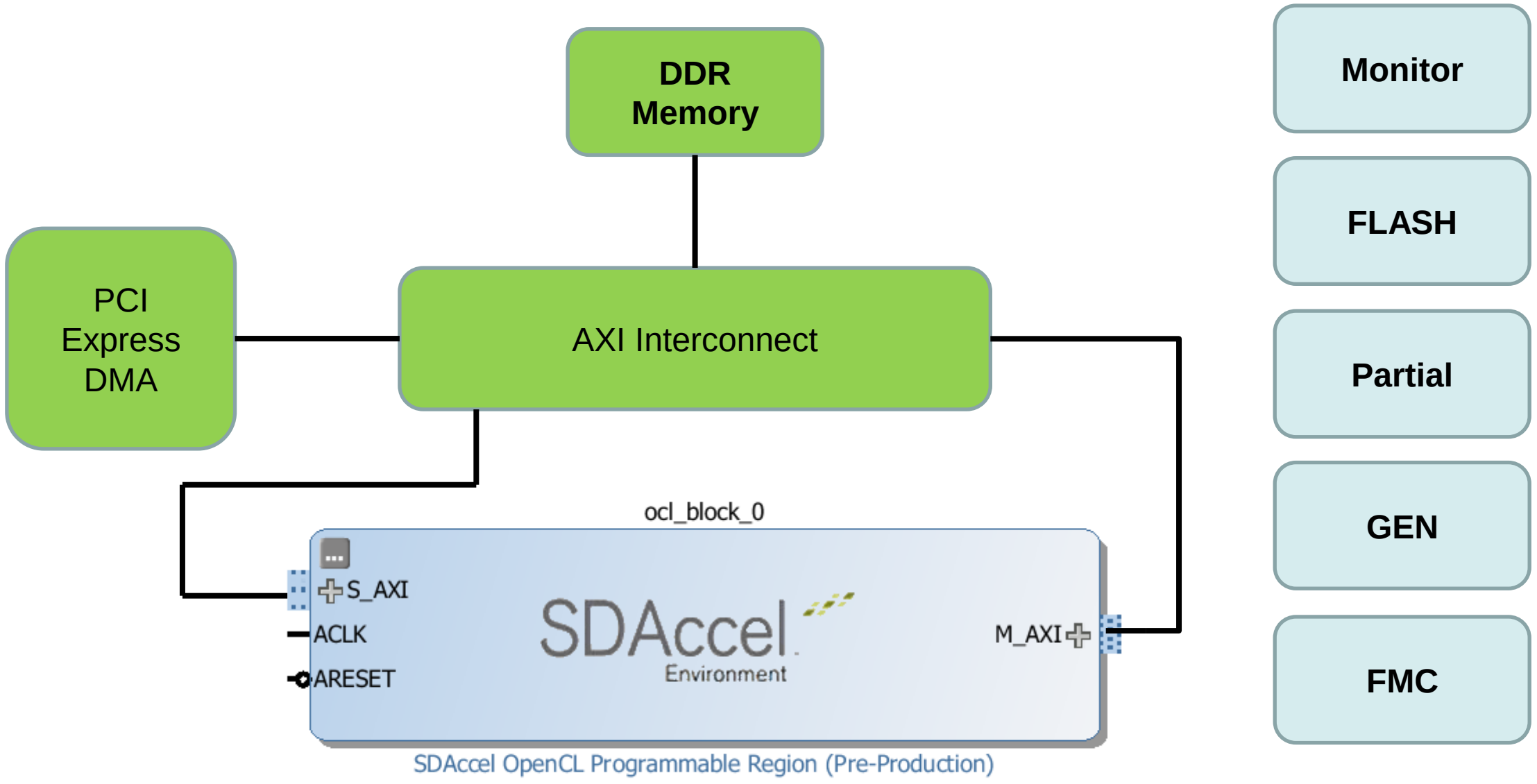
  - Локальная память -
    внутренняя память Compute Unit

  - Регистровая память
    внутренняя память Processing Element

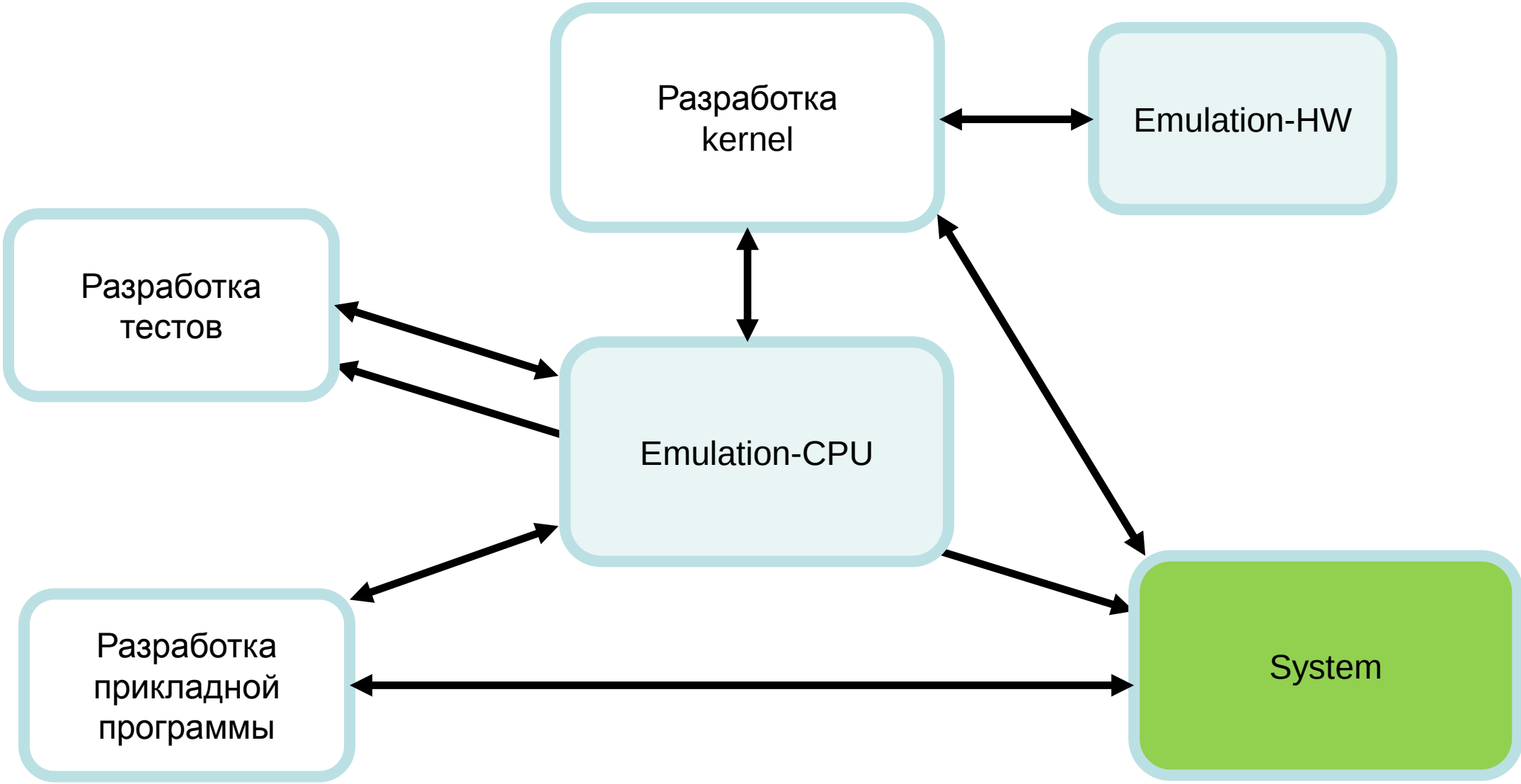
- Дополнительно OpenCL 2.0 вводит понятие PIPE (FIFO)



Структура проекта ПЛИС



Цикл разработки



Программа CheckTransfer

mc [user52@home]:/xprj/sdaccel/check_transfer/bin

File Edit View Search Terminal Help

```
[user52@home bin]$ ./system_run
```

TIME	BLOCK_WR	BLOCK_RD	BLOCK_OK	BLOCK_ERROR	SPD_CURRENT	SPD_AVR
OUT: 30.0	159	156	156	0	1280.0	1344.0
IN: 30.0	193	192	191	0	1664.0	1627.4

```
platform Name: Xilinx
Vendor Name : Xilinx
Found Platform
Device: xilinx:adm-pcie-ku3:2ddr-xpr:4.0
INFO: Importing ../sdx_wsp/check_transfer/System/binary_container_1.xclbin
Loading: '../sdx_wsp/check_transfer/System/binary_container_1.xclbin'

TF_CheckTransferOut::PrepareInThread

Alloc host memory 2 x 268435456  - Ok
Alloc device memory 2 x 268435456 (DDR0 & DDR1)  - Ok
```

Два независимых теста – на приём и на передачу

Фрагмент кода: gen_cnt

- Объявление переменных

  ulong8 temp1;

  ulong8 addConst;

- Тип **ulong8** это восемь элементов типа **ulong**

  ulong - беззнаковое целое 64 бита.

  ulong8 - беззнаковое целое, 512 бит

- Обращения к элементам возможно по именам или по индексам

  addConst.s0 или **addConst.s[0]**

- Основной цикл:

```
for( int ii = 0; ii < size; ii++ )  {
    pOut[ii] = temp1;
    temp1.s0 += addConst.s0;
    temp1.s1 += addConst.s1;
    temp1.s2 += addConst.s2;
    temp1.s3 += addConst.s3;
    temp1.s4 += addConst.s4;
    temp1.s5 += addConst.s5;
    temp1.s6 += addConst.s6;
    temp1.s7 += addConst.s7;
}
```

Фрагмент кода: check_cnt

- Объявление FIFO

```
pipe ulong8 pipe_input __attribute__((xcl_reqd_pipe_depth(512)));
```

- Запись в FIFO

```
write_pipe_block( pipe_input,  &in[ii] );
```

- Чтение и проверка

```
read_pipe_block( pipe_input, &temp0 );
```

…

```
flag0 |= check_data64( temp0.s0, temp1.s0 );
flag1 |= check_data64( temp0.s1, temp1.s1 );
flag2 |= check_data64( temp0.s2, temp1.s2 );
```

SDAccel в облаке

ALVEO U200

ALVEO U250

ALVEO U280

Облачные серверы:

- Amazon FC2
- Nimbix

- Дешёвый доступ к мощным ПЛИС
- Магазин приложений

GPUDircet RDMA example

Пример прямой передачи данных в память GPU NVidia

https://github.com/karakozov/gpudma

Разработчики: Владимир Каракозов

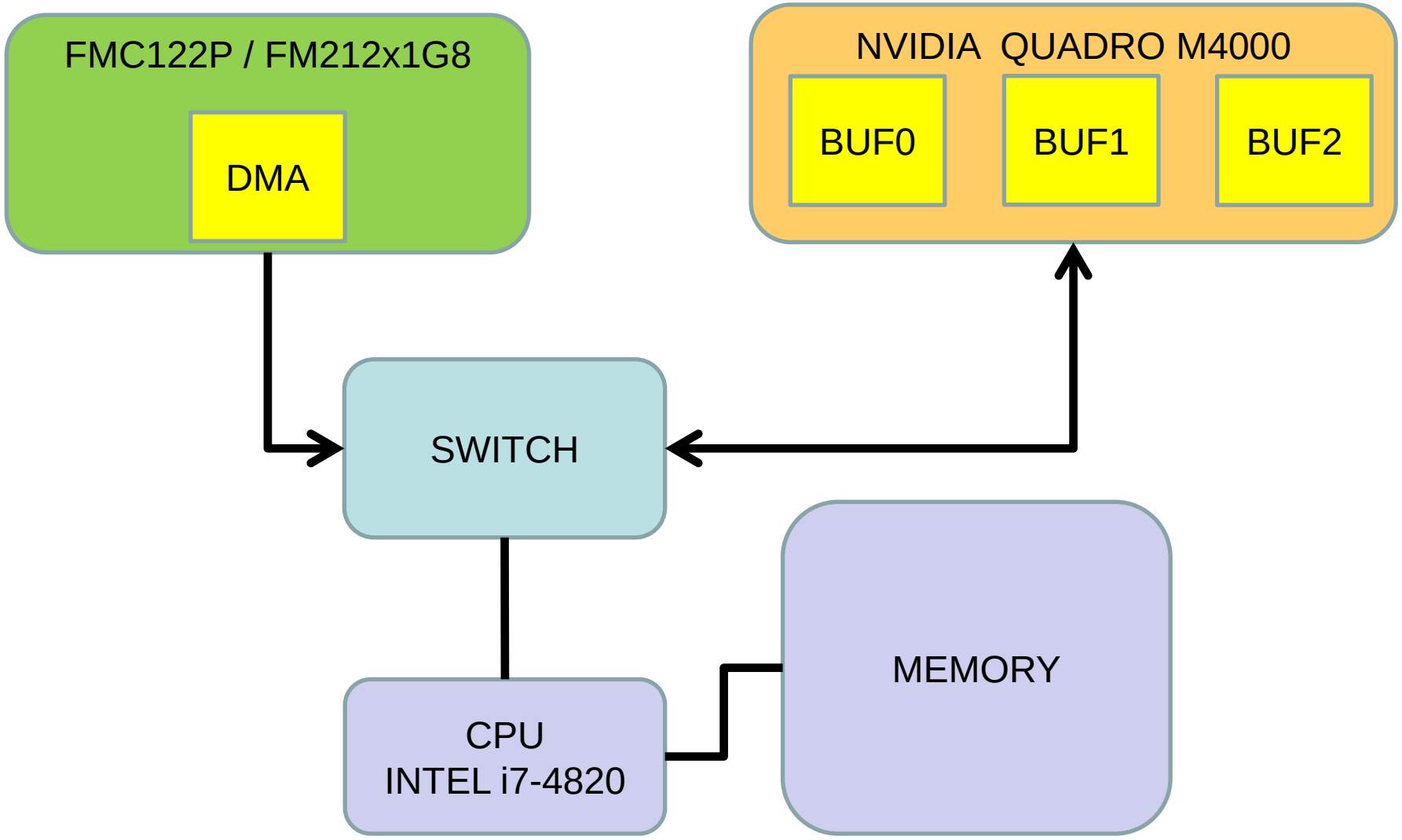
Дмитрий Смехов

Область применения:

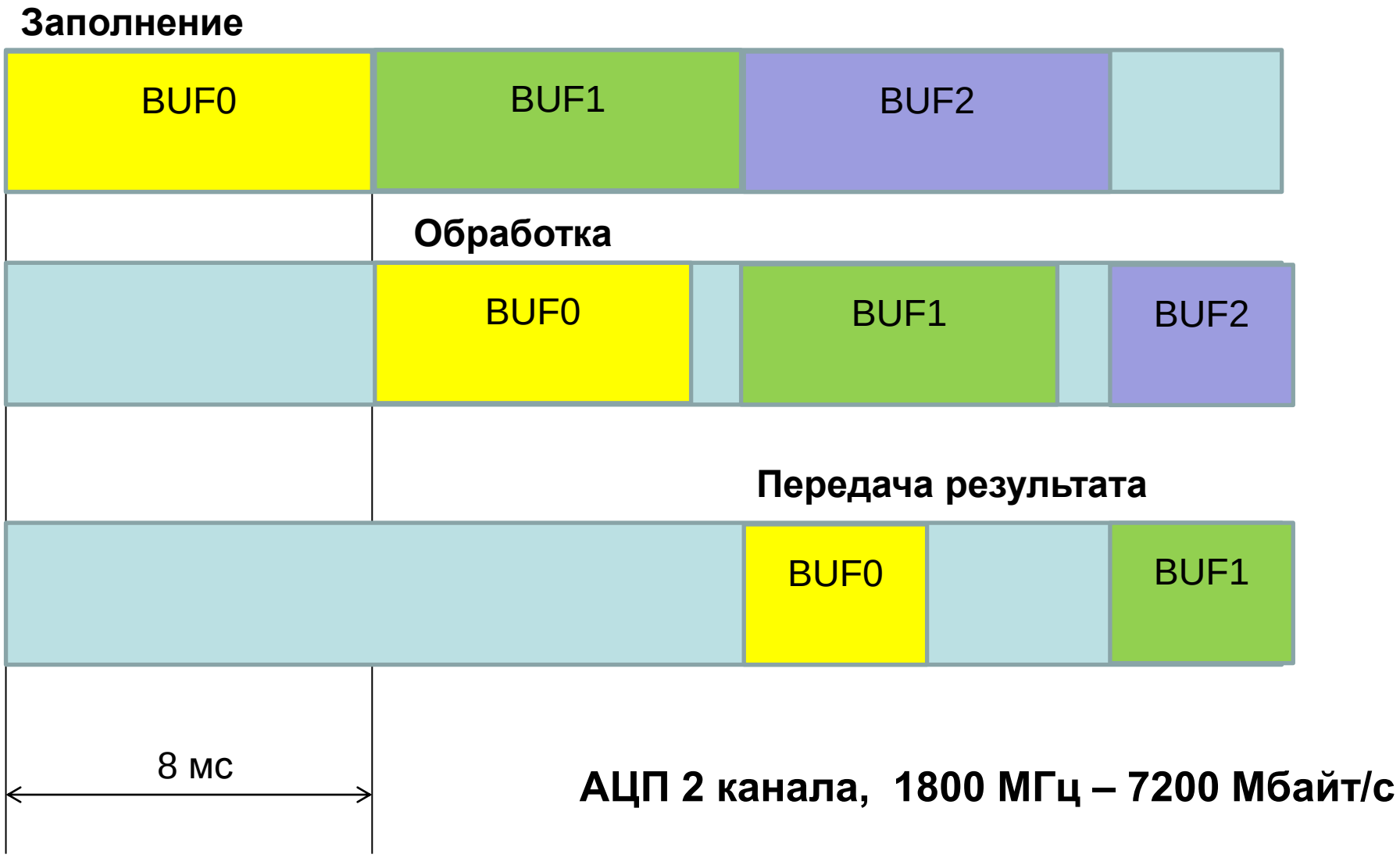
- вычисления на GPU

Категория: передача данных

Структура системы

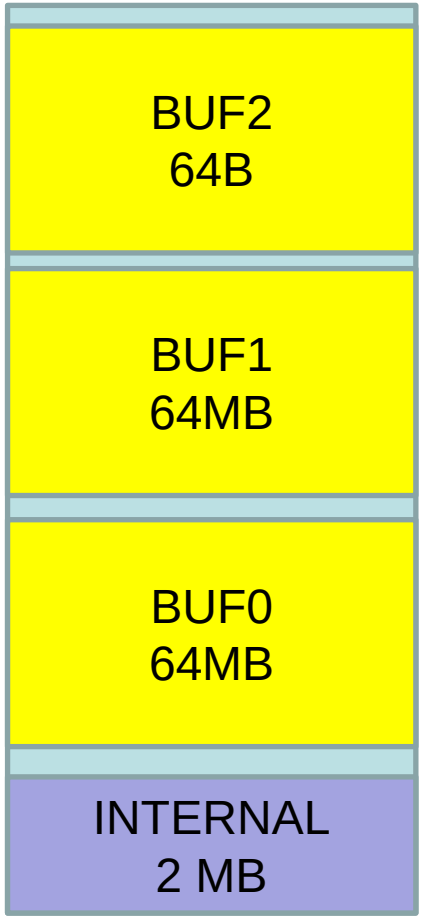


Временная диаграмма



АЦП 2 канала, 1800 МГц – 7200 Мбайт/с

Структура BAR1



- Выделяется три буфера внутри GPU
- Каждый буфер отображается внутри BAR1
- **nvidia_p2p_get_pages()** – получение таблицы дескрипторов для буфера внутри BAR1
- Функция вызывается из драйвера нулевого кольца
- Таблица используется для программирования DMA канала
- Требуется разработать отдельный драйвер

Приложение APP_TEMPLATE

- Выделение трёх буферов в памяти GPU
- Заполнение буферов счётчиком
- Проверка счетчика на GPU
- Передача прореженного потока в HOST

Важно: программа GPU работает в режиме сигнального процессора.

SIMULINK_SM

Пример взаимодействия между программами сбора данных, генерации сигналов и SIMULINK

https://github.com/dsmv/simulink_sm

Разработчик: Дмитрий Смехов

Область применения:

- взаимодействие с MATLAB/SIMULINK

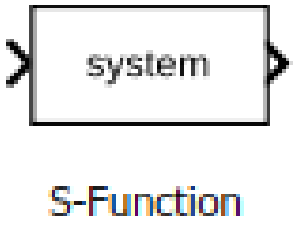
Категория: передача данных

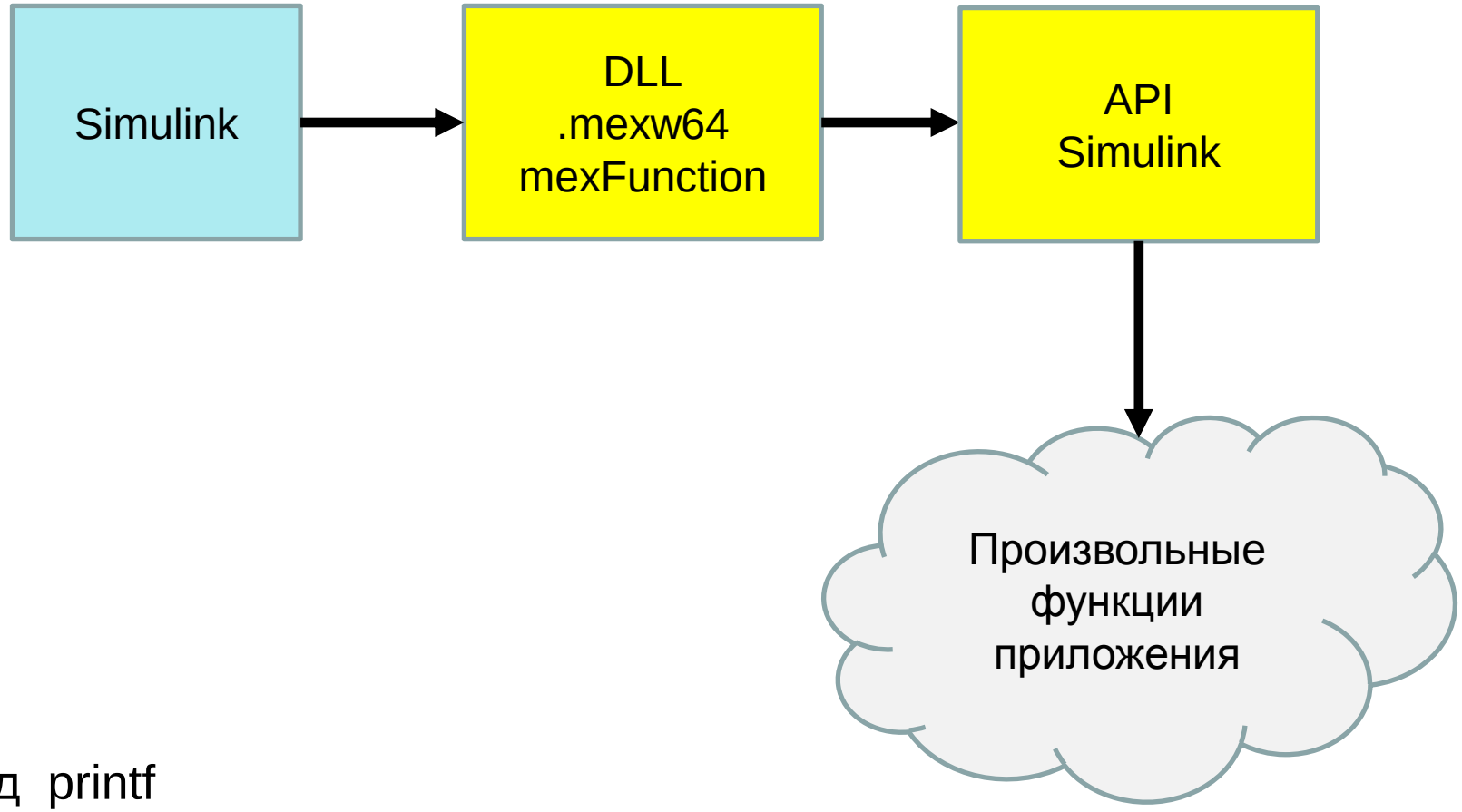
Подключение к Simulink

Simulink

User-Defined Functions

S-Function

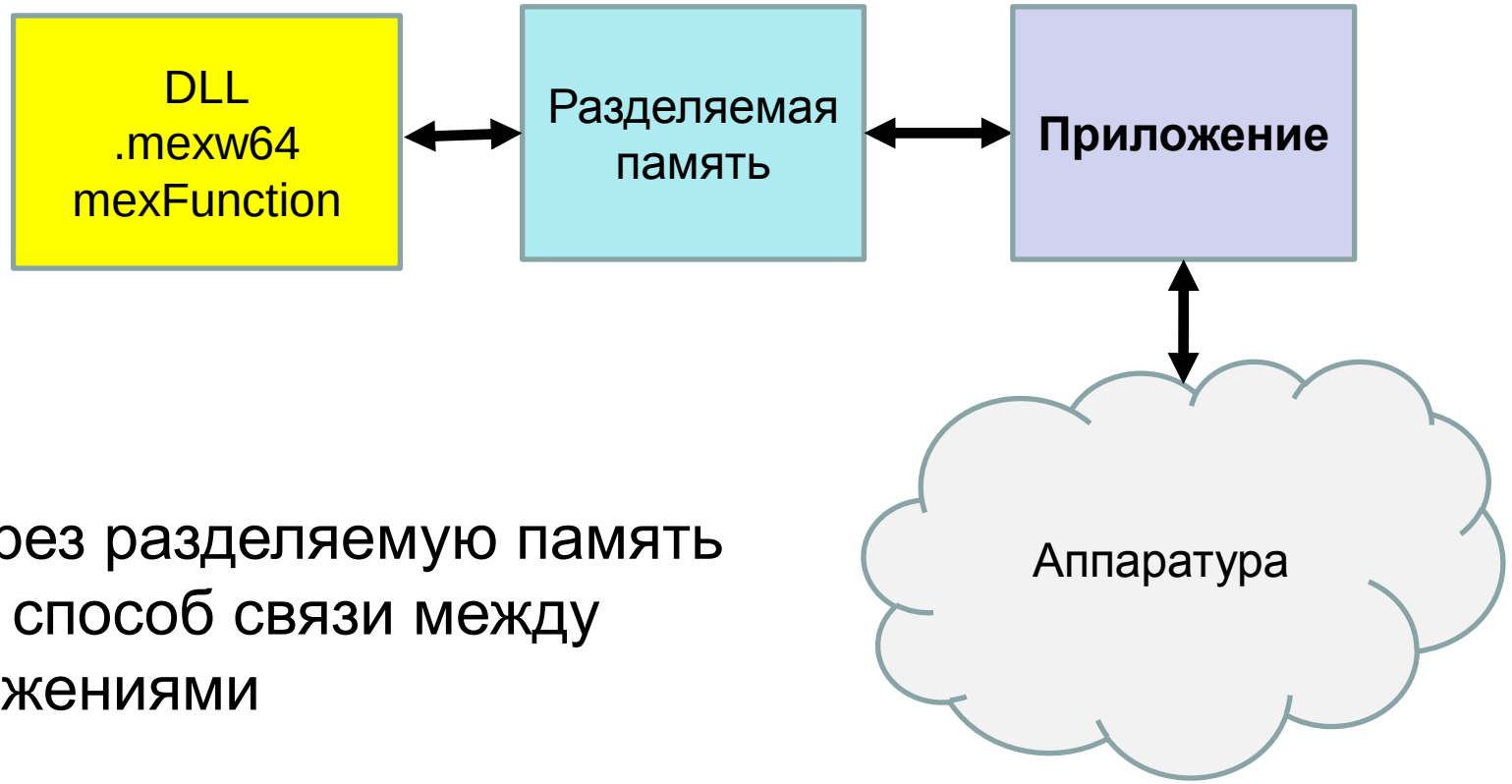




Проблемы:

- Затруднена отладка DLL
- Неудобно смотреть вывод printf
- Перекомпиляция DLL требует выхода из MATLAB

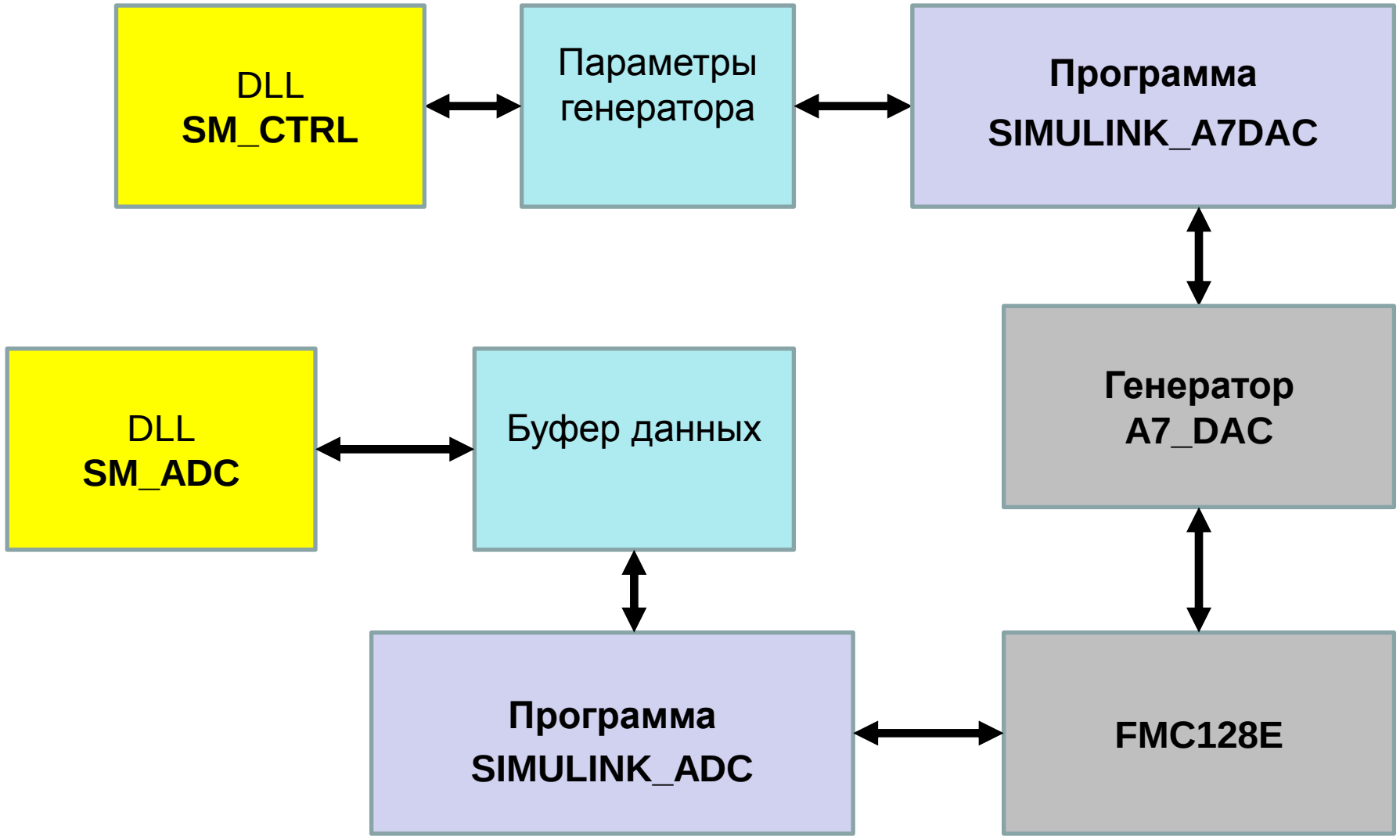
Связь через разделяемую память



Подключение через разделяемую память это стандартный способ связи между сложными приложениями

ВАЖНО: не требуется перезапускать приложение

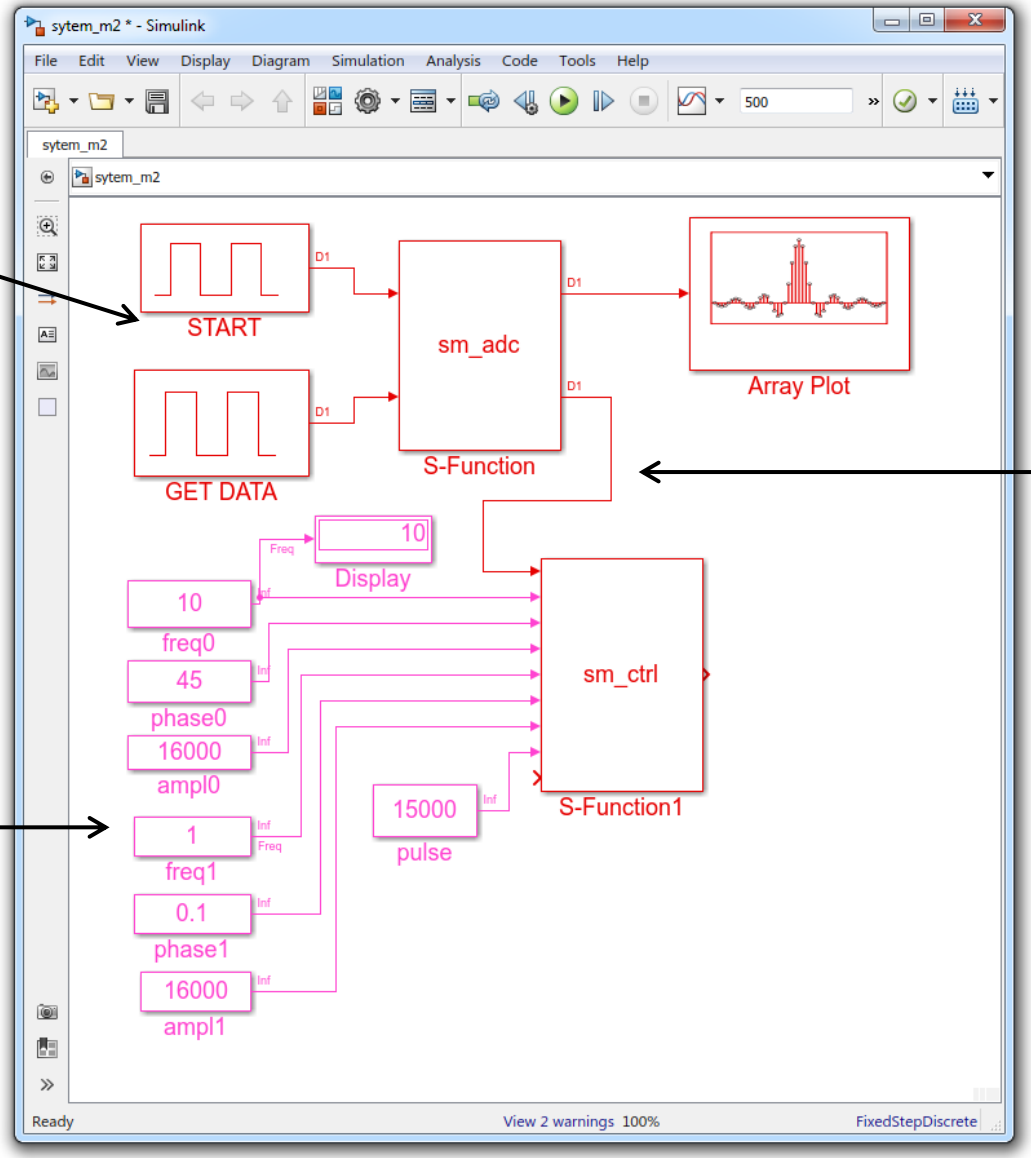
Система сбора



Simulink

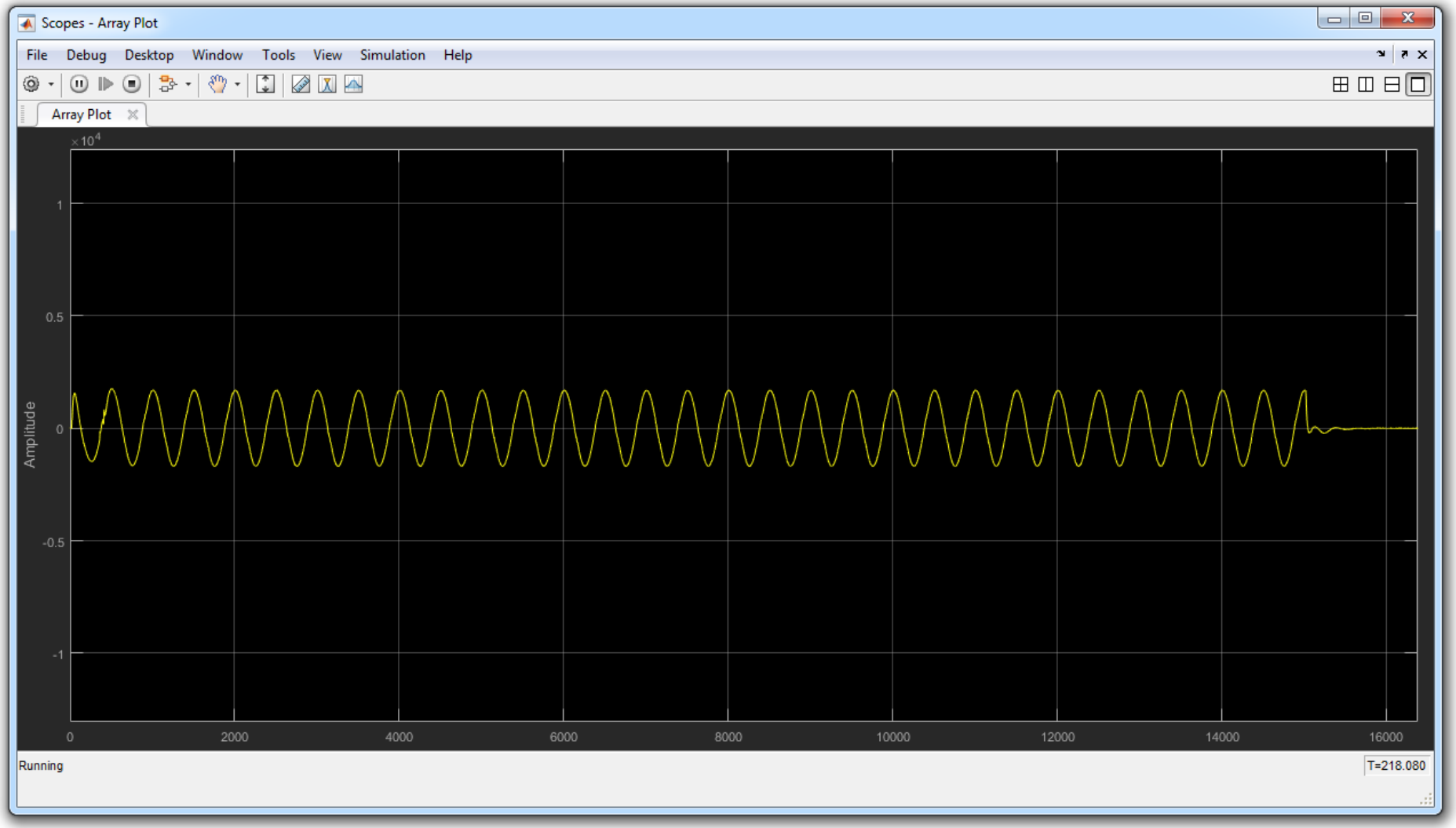
Два задающих генератора

Параметры A7_DAC

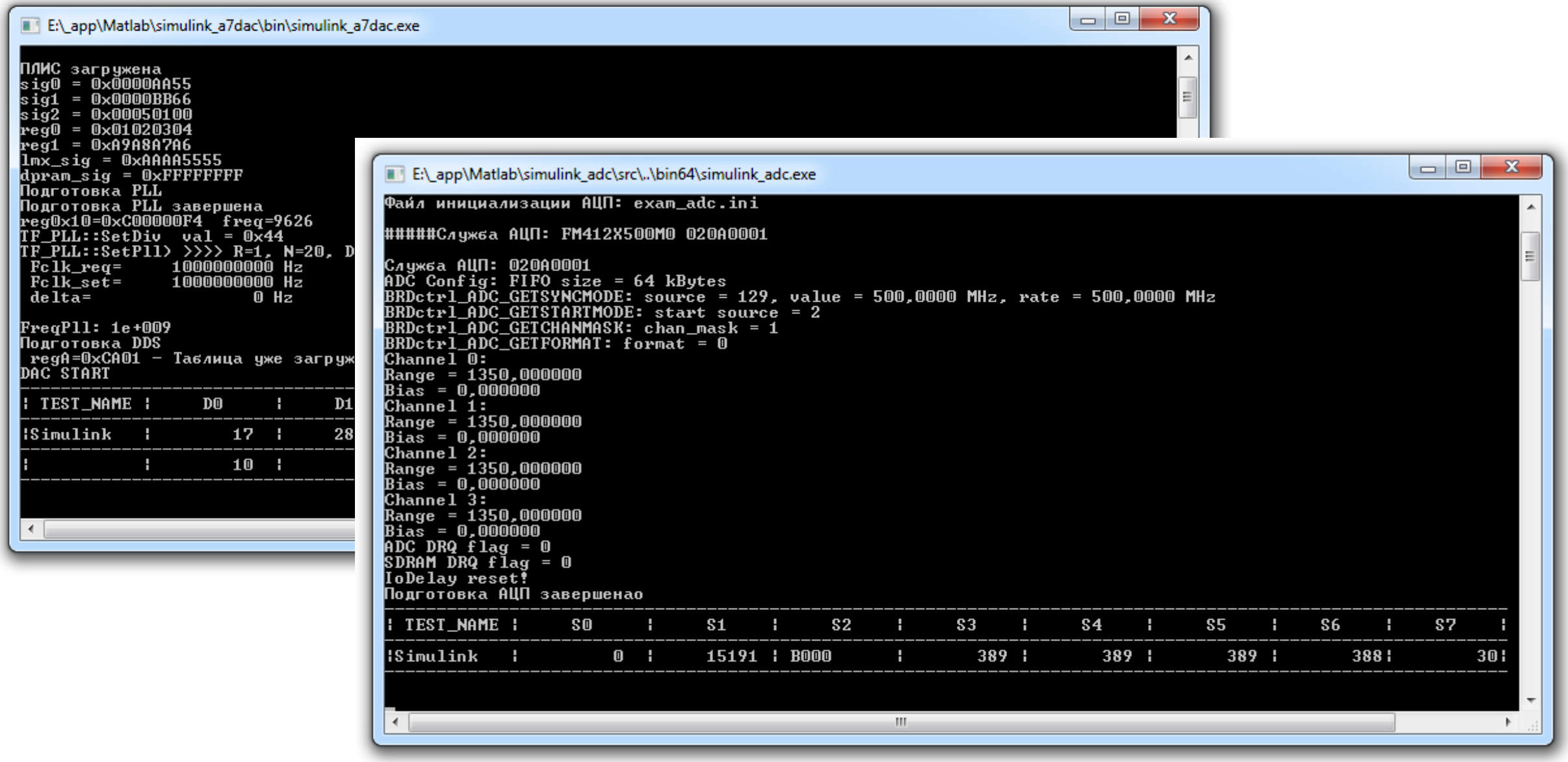


Связь сигнала старта

Результат


Scopes - Array Plot
File Debug Desktop Window Tools View Simulation Help
Array Plot
×10^4
1
0.5
0
-0.5
-1
Amplitude
0 2000 4000 6000 8000 10000 12000 14000 16000
Running
T=218.080

Программы управления



GIPCY

Кроссплатформенная библиотека межпроцессного взаимодействия

https://github.com/insys-projects/gipcy

Разработчики: Андрей Дорохин

Владимир Каракозов

Андрей Козлов

Область применения:

- разработка кроссплатформенных приложений

Категория: разработка ПО

BARDY

Системный драйвер
(кольцо 0)

Системные DLL
- базовый драйвер
- сайддрайвер

Приложения

Операционные системы:

- Windows x32
- Windows x64
- Linux для x86
- Linux для ARM
- Linux для TMS
- Linux для Zynq
- QNX

Основные свойства

- Определения типов
- Выделение памяти
- Семафоры
- Events
- Потоки
- Таймеры
- Набор системных функций

  (время, задержки, консоль)

FileHunter

Программа сравнения каталогов

https://github.com/insys-projects/FileHunter

Разработчик: Андрей Козлов

Область применения:

- управление файлами и каталогами

Категория: разработка ПО

Размещение исходных файлов

Вариант 1:

- Общая структура каталогов для нескольких проектов
- Каждый экземпляр файла входит в несколько проектов

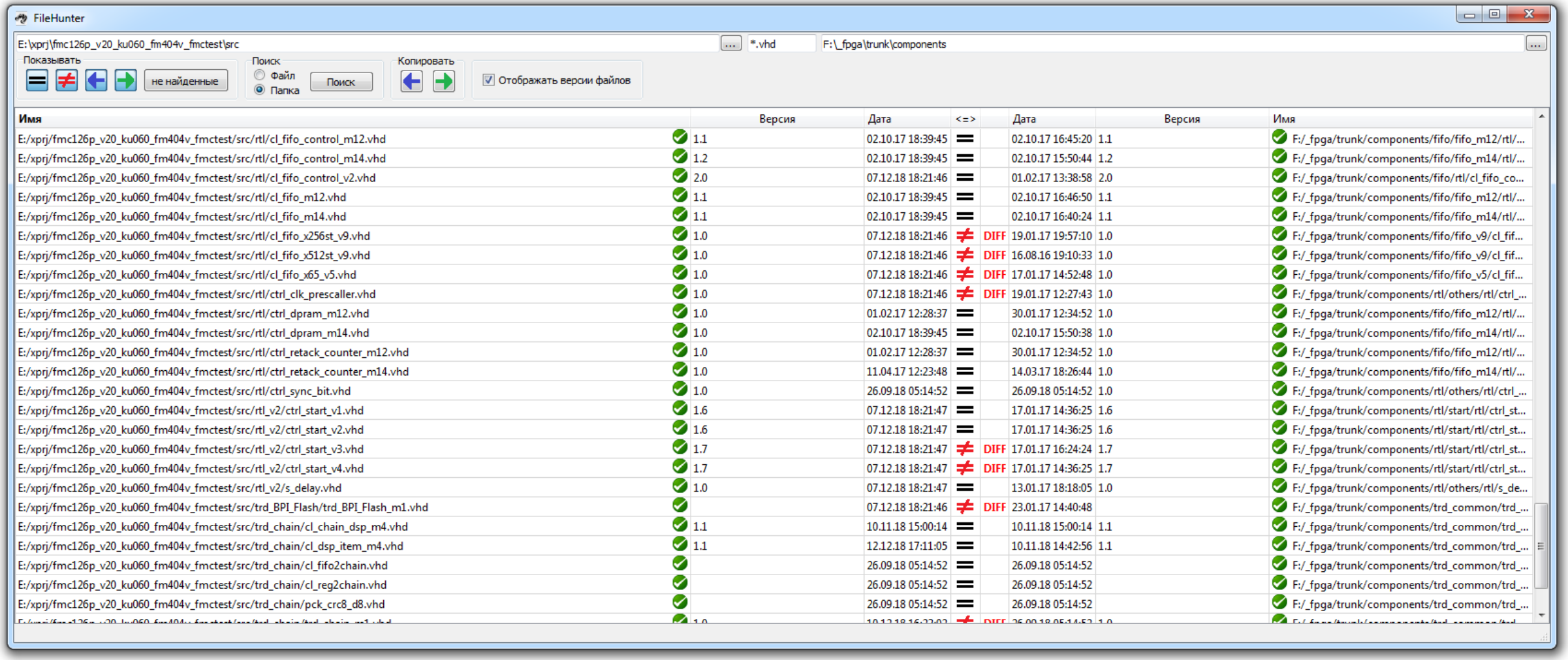
(ориентация на работу с git/subversion при помощи ветвлений)

Вариант 2:

- Каждый проект имеет свою структуру каталогов
- В разные проекты входят одинаковые копии файлов

(удобно проводить архивацию отдельного проекта)

FileHunter



Имя	Версия	Дата	<=>		Дата	Версия	Имя
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/cl_fifo_control_m12.vhd	1.1	02.10.17 18:39:45	=		02.10.17 16:45:20	1.1	F:/_fpga/trunk/components/fifo/fifo_m12/rtl/...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/cl_fifo_control_m14.vhd	1.2	02.10.17 18:39:45	=		02.10.17 15:50:44	1.2	F:/_fpga/trunk/components/fifo/fifo_m14/rtl/...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/cl_fifo_control_v2.vhd	2.0	07.12.18 18:21:46	=		01.02.17 13:38:58	2.0	F:/_fpga/trunk/components/fifo/rtl/cl_fifo_co...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/cl_fifo_m12.vhd	1.1	02.10.17 18:39:45	=		02.10.17 16:46:50	1.1	F:/_fpga/trunk/components/fifo/fifo_m12/rtl/...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/cl_fifo_m14.vhd	1.1	02.10.17 18:39:45	=		02.10.17 16:40:24	1.1	F:/_fpga/trunk/components/fifo/fifo_m14/rtl/...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/cl_fifo_x256st_v9.vhd	1.0	07.12.18 18:21:46	≠	DIFF	19.01.17 19:57:10	1.0	F:/_fpga/trunk/components/fifo/fifo_v9/cl_fif...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/cl_fifo_x512st_v9.vhd	1.0	07.12.18 18:21:46	≠	DIFF	16.08.16 19:10:33	1.0	F:/_fpga/trunk/components/fifo/fifo_v9/cl_fif...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/cl_fifo_x65_v5.vhd	1.0	07.12.18 18:21:46	≠	DIFF	17.01.17 14:52:48	1.0	F:/_fpga/trunk/components/fifo/fifo_v5/cl_fif...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/ctrl_clk_prescaller.vhd	1.0	07.12.18 18:21:46	≠	DIFF	19.01.17 12:27:43	1.0	F:/_fpga/trunk/components/rtl/others/rtl/ctrl_...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/ctrl_dpram_m12.vhd	1.0	01.02.17 12:28:37	=		30.01.17 12:34:52	1.0	F:/_fpga/trunk/components/fifo/fifo_m12/rtl/...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/ctrl_dpram_m14.vhd	1.0	02.10.17 18:39:45	=		02.10.17 15:50:38	1.0	F:/_fpga/trunk/components/fifo/fifo_m14/rtl/...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/ctrl_retack_counter_m12.vhd	1.0	01.02.17 12:28:37	=		30.01.17 12:34:52	1.0	F:/_fpga/trunk/components/fifo/fifo_m12/rtl/...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/ctrl_retack_counter_m14.vhd	1.0	11.04.17 12:23:48	=		14.03.17 18:26:44	1.0	F:/_fpga/trunk/components/fifo/fifo_m14/rtl/...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl/ctrl_sync_bit.vhd	1.0	26.09.18 05:14:52	=		26.09.18 05:14:52	1.0	F:/_fpga/trunk/components/rtl/others/rtl/ctrl_...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl_v2/ctrl_start_v1.vhd	1.6	07.12.18 18:21:47	=		17.01.17 14:36:25	1.6	F:/_fpga/trunk/components/rtl/start/rtl/ctrl_st...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl_v2/ctrl_start_v2.vhd	1.6	07.12.18 18:21:47	=		17.01.17 14:36:25	1.6	F:/_fpga/trunk/components/rtl/start/rtl/ctrl_st...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl_v2/ctrl_start_v3.vhd	1.7	07.12.18 18:21:47	≠	DIFF	17.01.17 16:24:24	1.7	F:/_fpga/trunk/components/rtl/start/rtl/ctrl_st...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl_v2/ctrl_start_v4.vhd	1.7	07.12.18 18:21:47	≠	DIFF	17.01.17 14:36:25	1.7	F:/_fpga/trunk/components/rtl/start/rtl/ctrl_st...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/rtl_v2/s_delay.vhd	1.0	07.12.18 18:21:47	=		13.01.17 18:18:05	1.0	F:/_fpga/trunk/components/rtl/others/rtl/s_de...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/trd_BPI_Flash/trd_BPI_Flash_m1.vhd		07.12.18 18:21:46	≠	DIFF	23.01.17 14:40:48		F:/_fpga/trunk/components/trd_common/trd_...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/trd_chain/cl_chain_dsp_m4.vhd	1.1	10.11.18 15:00:14	=		10.11.18 15:00:14	1.1	F:/_fpga/trunk/components/trd_common/trd_...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/trd_chain/cl_dsp_item_m4.vhd	1.1	12.12.18 17:11:05	=		10.11.18 14:42:56	1.1	F:/_fpga/trunk/components/trd_common/trd_...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/trd_chain/cl_fifo2chain.vhd		26.09.18 05:14:52	=		26.09.18 05:14:52		F:/_fpga/trunk/components/trd_common/trd_...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/trd_chain/cl_reg2chain.vhd		26.09.18 05:14:52	=		26.09.18 05:14:52		F:/_fpga/trunk/components/trd_common/trd_...
E:/xprj/fmc126p_v20_ku060_fm404v_fmctest/src/trd_chain/pck_crc8_d8.vhd		26.09.18 05:14:52	=		26.09.18 05:14:52		F:/_fpga/trunk/components/trd_common/trd_...

ISVI

Программа анализа сигналов

http://insys.ru/downloads/common

Разработчики: Андрей Козлов

Александр Чурзин

Возможности:

- отображение аналогового сигнала и спектра
- вычисление параметров сигнала

Категория: анализ сигналов

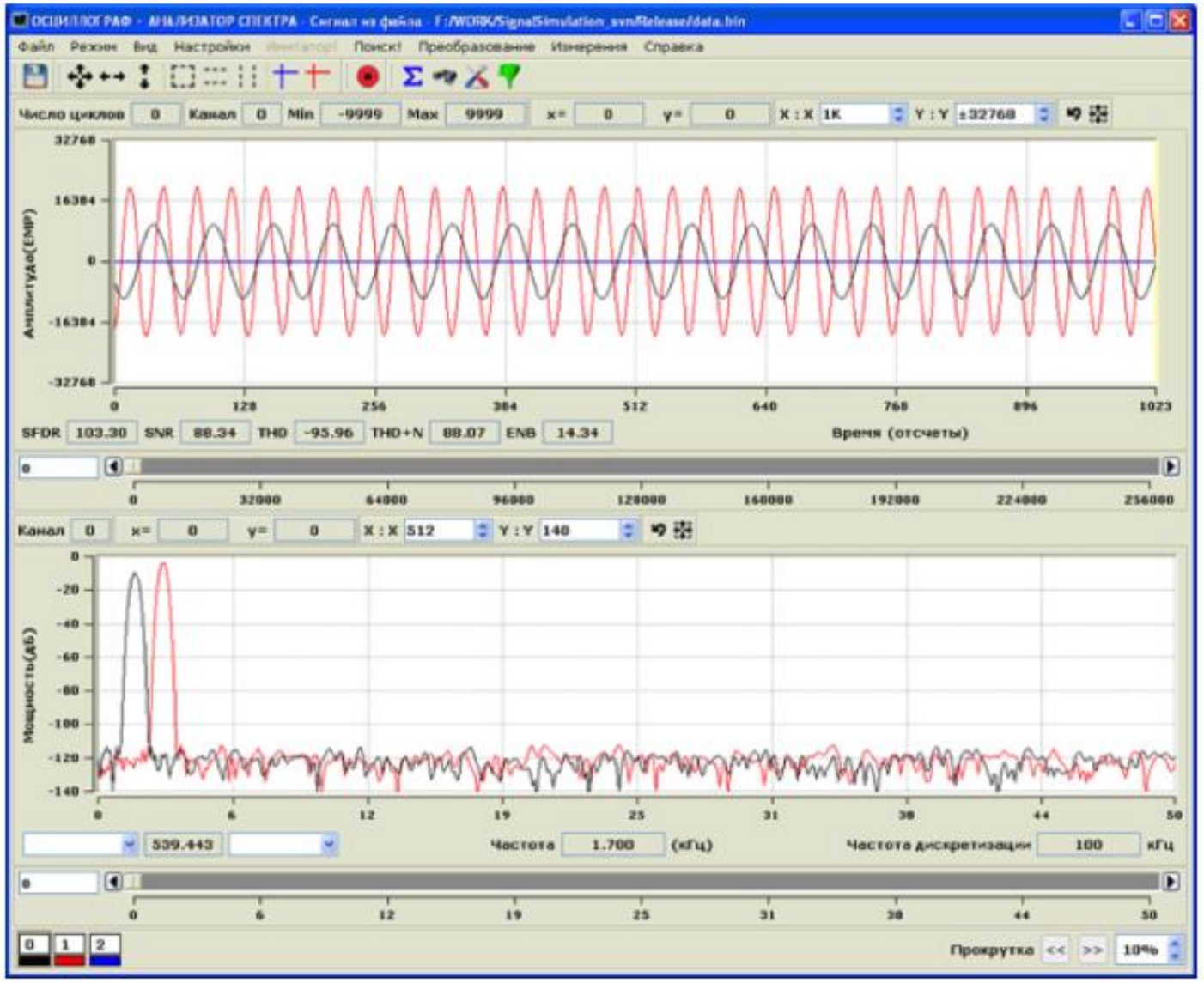
ISVI

Режимы отображения:

- Осциллограф
- Анализатор спектра
- Водопад

Основные измеряемые параметры:

- SNR
- THD
- THD+N
- ENB



Контакты

«Инструментальные Системы»

Москва

Дмитрий Смехов

WWW: http://www.insys.ru

E-mail: dsmv@insys.ru

Спасибо за внимание