

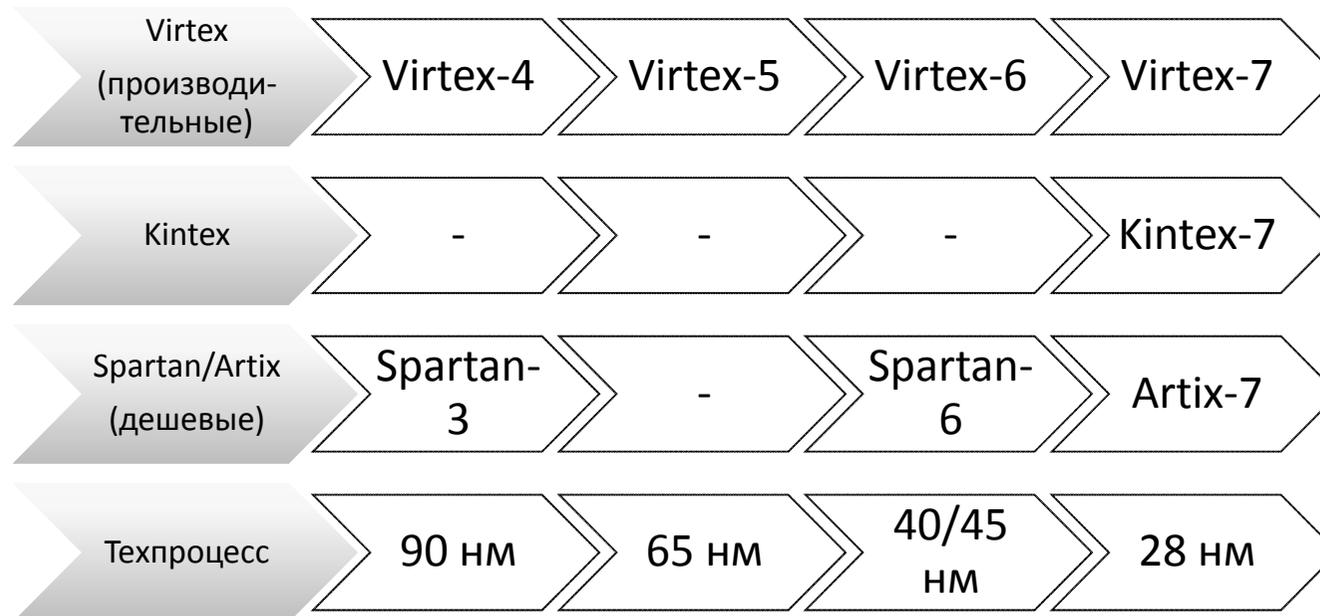
# Программная поддержка вычислительных систем в FPGA на основе конфигурируемых процессоров

Тарасов И.Е., д.т.н.

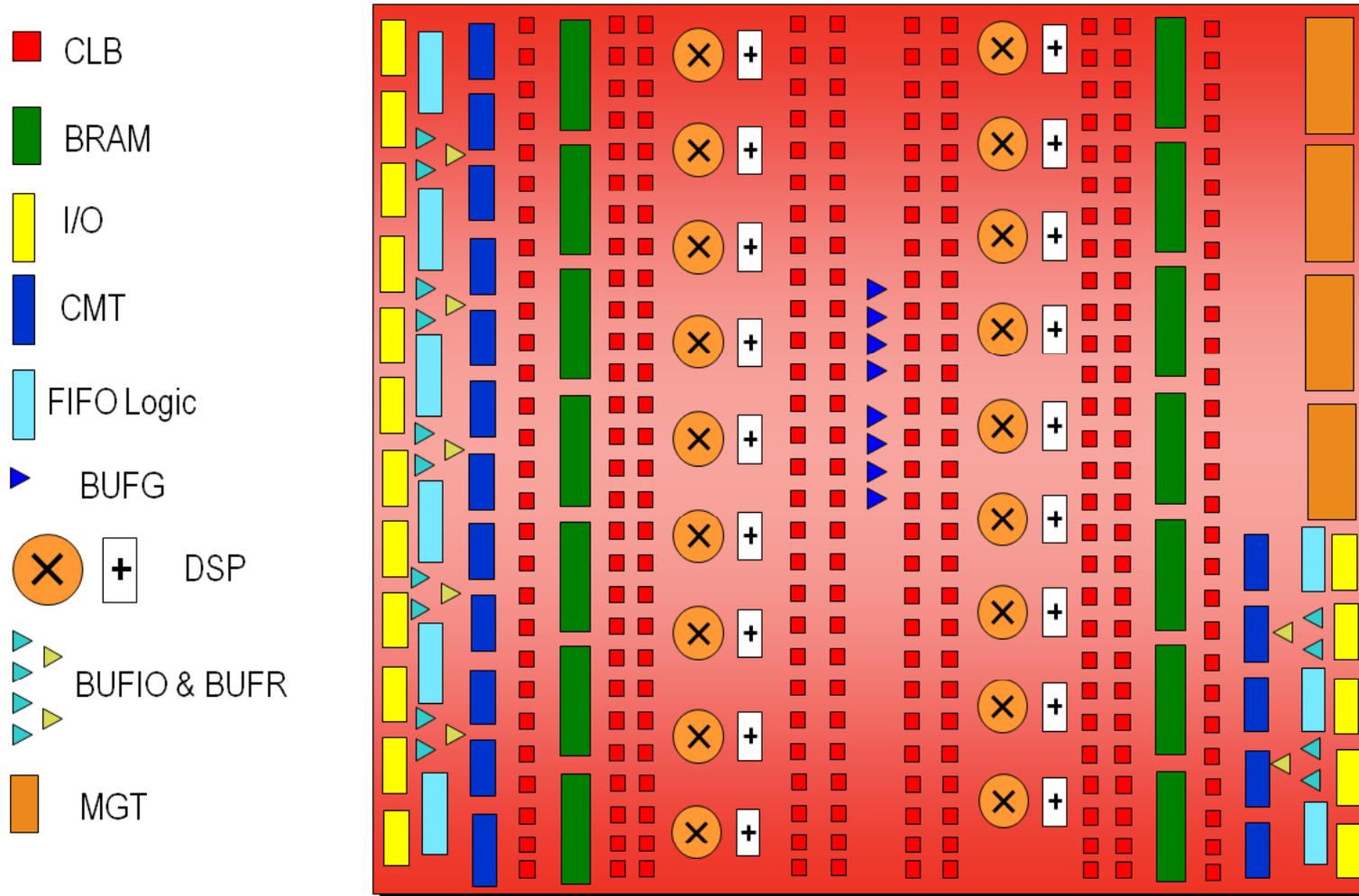
- Ковровская государственная технологическая академия им. В.А. Дегтярева
- КТЦ Инлайн групп

# FPGA Xilinx

- ~ x2 логический объем с каждым поколением
- ~ 20% рост системной тактовой частоты с каждым поколением



# Kintex-7



# Суперкомпьютеры: гибридные решения



**CUDA**



**X86**



**FPGA**

# Обработка видео и распознавание изображений в реальном времени

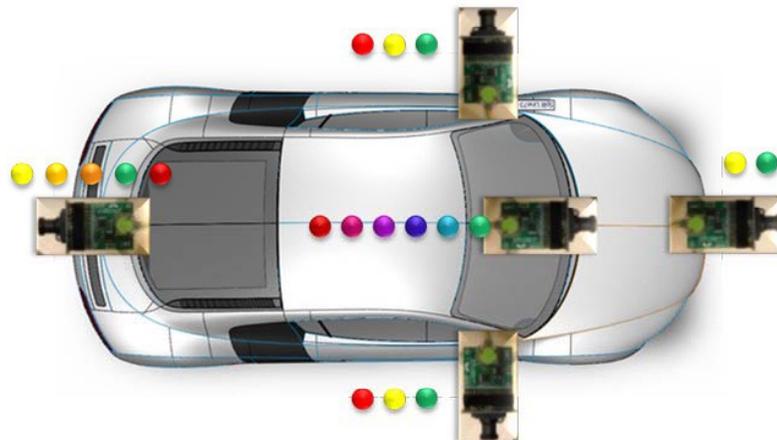


*logiADAK Automotive Driver Assistance Kit*

- Open All-in-One Platform
- Programmable, Scalable, Reconfigurable...

+ ROOM FOR YOUR INNOVATION!

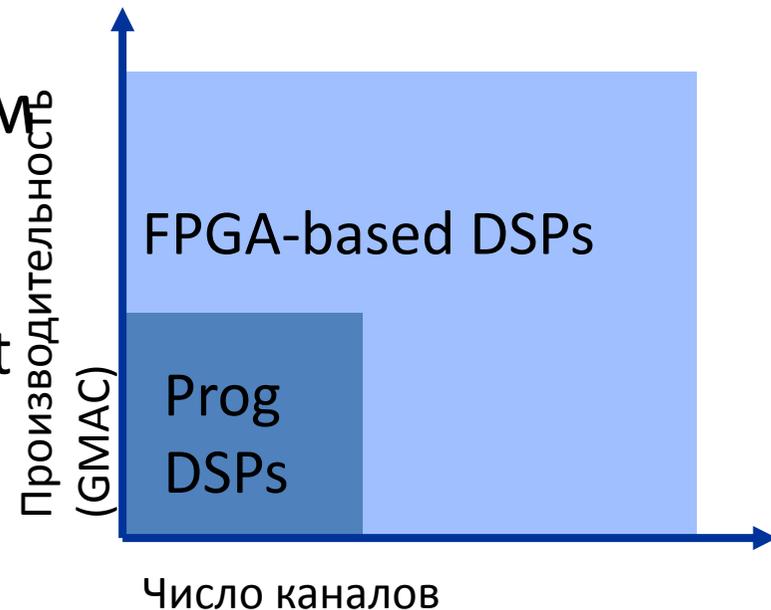
*Based on Xilinx® Zynq™ -7000 All Programmable SoC*



- 3D Surround View
- Rear View Camera
- Rear Cross Traffic
- Blind Spot Detection
- Lane Departure Warning
- Intelligent Headlamp Control
- Traffic Sign Recognition
- Forward Collision Warning
- Intelligent Speed Control
- Pedestrian Detection

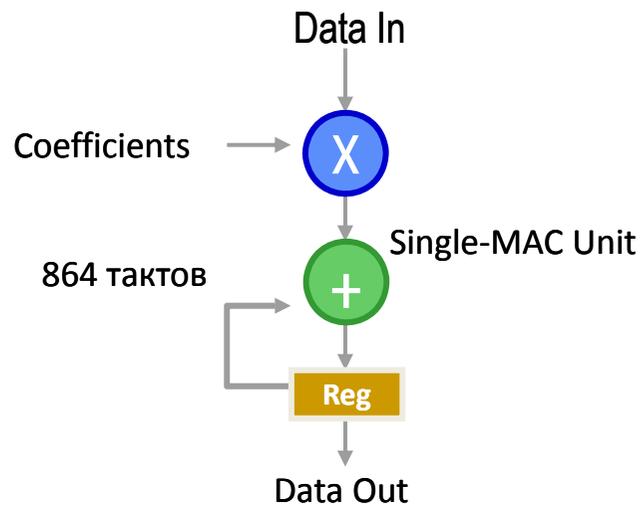
# FPGA для цифровой обработки сигналов

- FPGA – это альтернатива сигнальным процессорам для многоканальной обработки
  - “FPGAs deliver 25X the cost performance advantage of DSPs”
    - Jeff Bier, BDTI
  - Аппаратная платформа обеспечивает до 5000 GMACs в одном устройстве



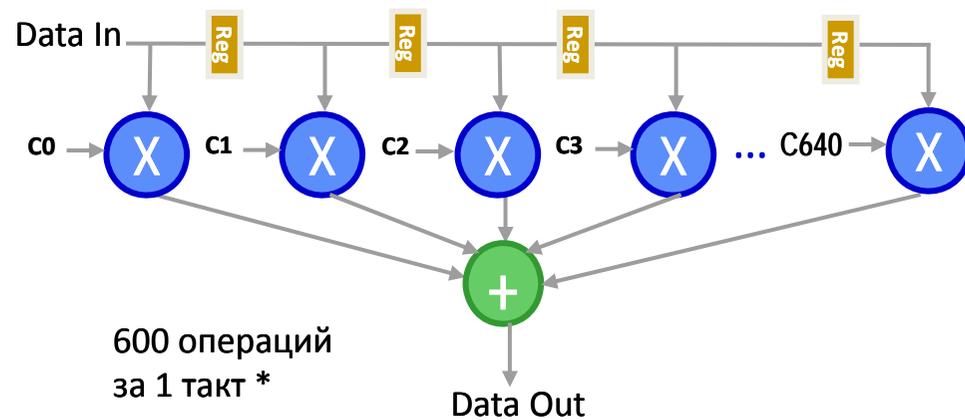
# Параллельная обработка в FPGA

Стандартный DSP процессор –  
последовательное исполнение



$$\frac{1.2 \text{ GHz}}{864 \text{ такта}} = 1.38 \text{ MSPS}$$

FPGA – полностью параллельное исполнение  
(Virtex-6 FPGA)

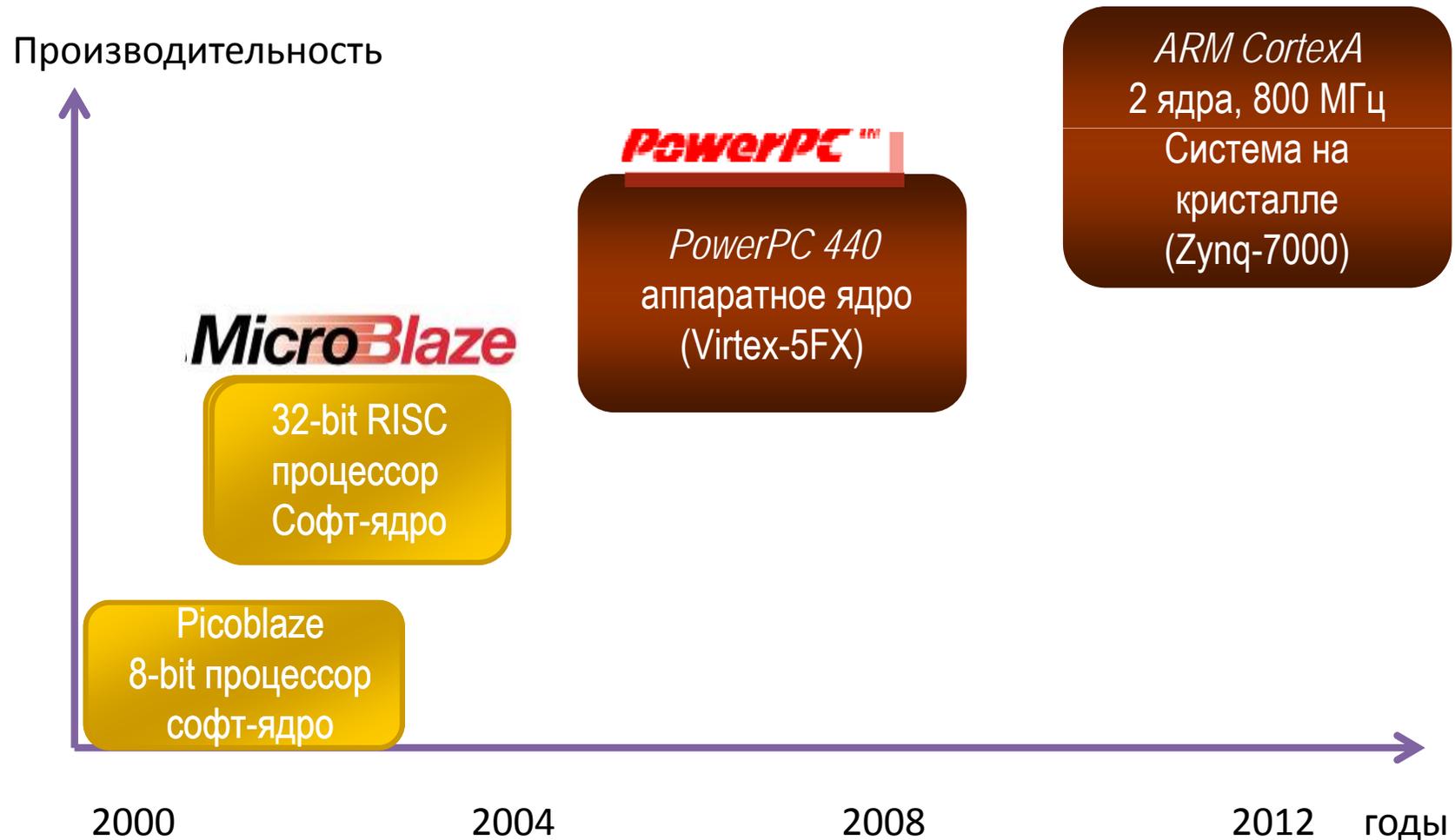


$$\frac{600 \text{ MHz}}{1 \text{ такт}} = 600 \text{ MSPS}$$

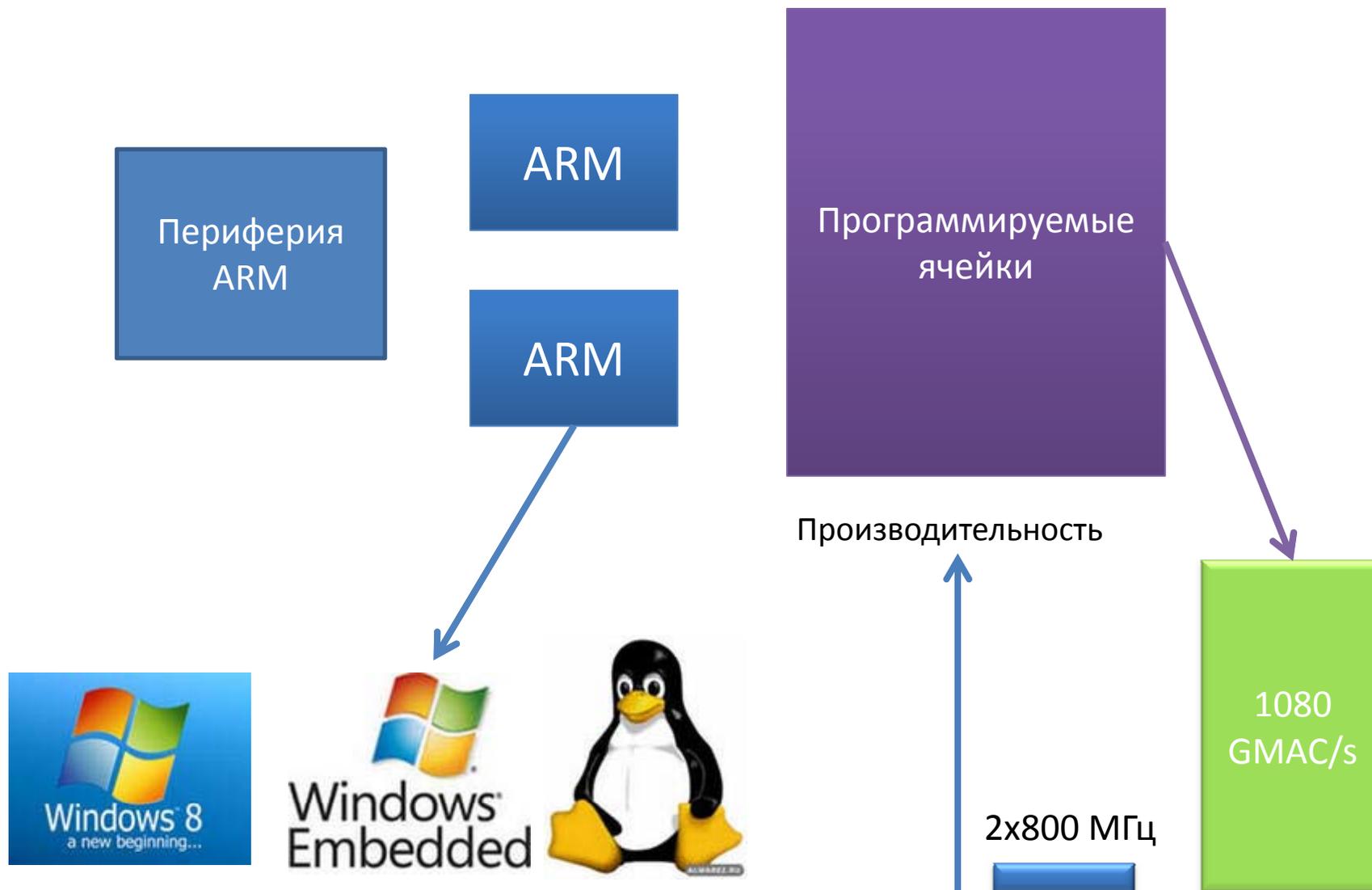
# Производительность FPGA в задачах цифровой обработки сигналов

Показатель	Virtex-4	Virtex-5	Virtex-6	Virtex-7
Тактовая частота, МГц	500	550	600	740
Количество блоков в ПЛИС наибольшего объема	512	1056	2016	3960
Максимальная производительность, GMAC/s (с симметричными коэффициентами)	256 (256)	580 (580)	1200 (2400)	2900 (5800)

# Процессоры для FPGA Xilinx

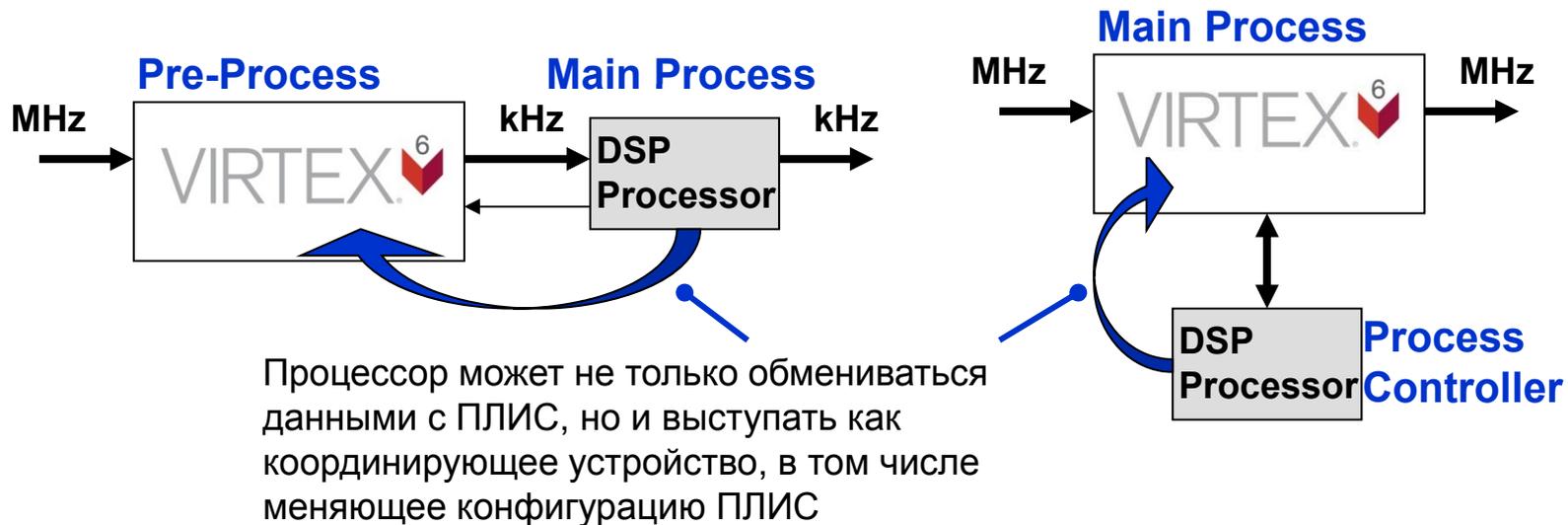


# Распределение задач между процессором и программируемой логикой

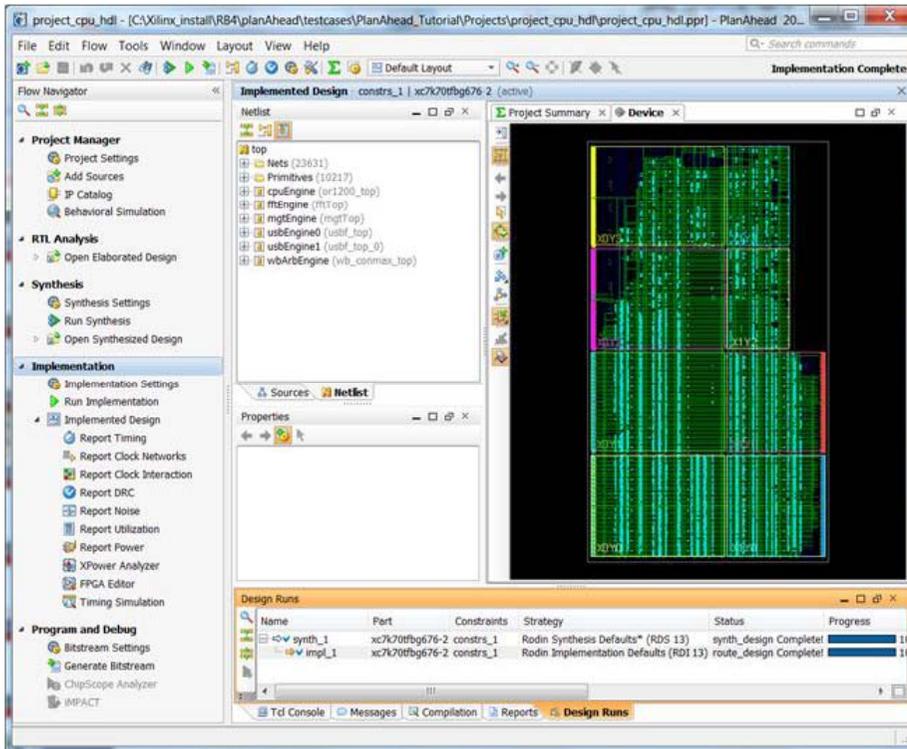


# FPGA как сопроцессор

- Сигнальные процессоры хорошо соответствуют задачам с низкой частотой входных данных, для обработки которых используются сложные алгоритмы
- Является ли FPGA сопроцессором?



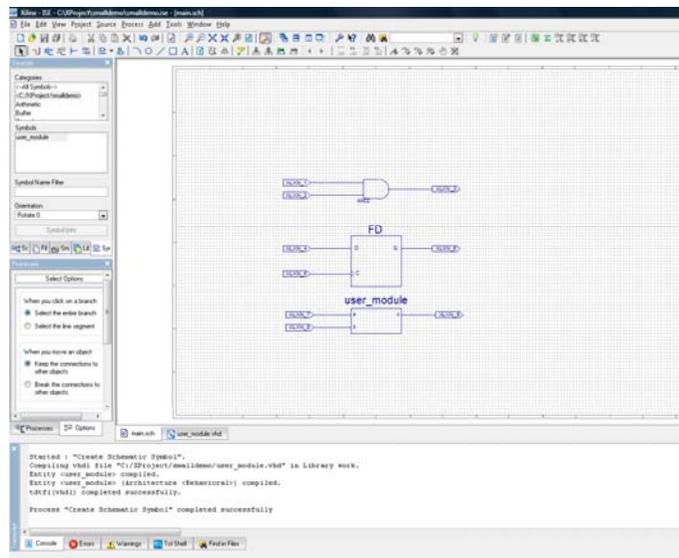
# САПР Vivado



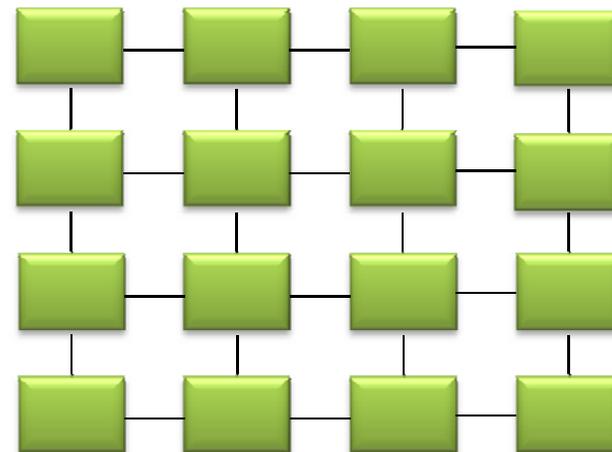
- САПР нового поколения, масштабируемая до объемов ~20 млн. логических ячеек (выпускаемый Virtex-7 – 2 млн. ячеек)
- Одна из решаемых задач – *уменьшение времени одной итерации до 18 часов*

# Подходы к разработке высокопроизводительных вычислительных систем

**HDL/schematic-ориентированное  
проектирование**



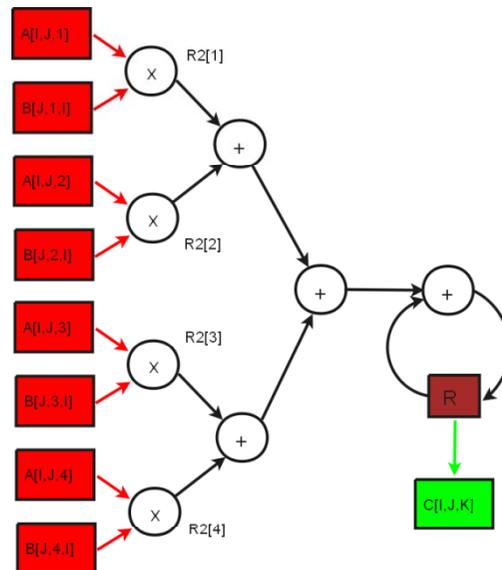
**Матрица процессоров**



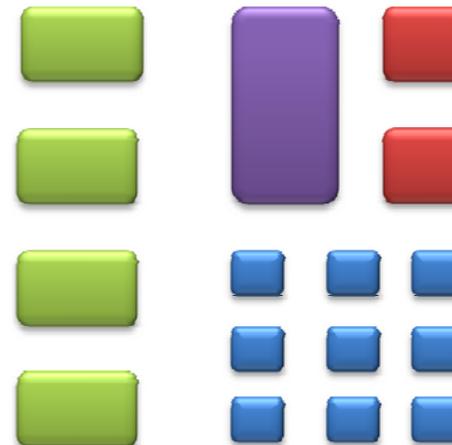
# Промежуточные подходы

## HDL → HLL

- **Var A,B,C: Array [2: Vector, 5: Stream**  
**CADR summa4;**  
**FOR i=1 TO 5 BEGIN**  
  **FOR j=1 TO 2 BEGIN**  
    **C[i, j]=A[i, j]+B[i, j];**  
  **END;**  
**END;**  
**ENDCadr;**

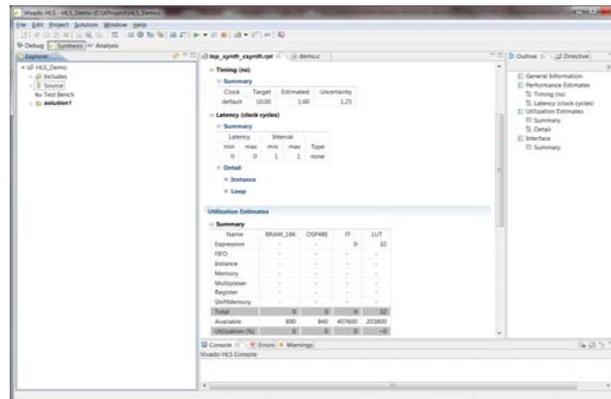
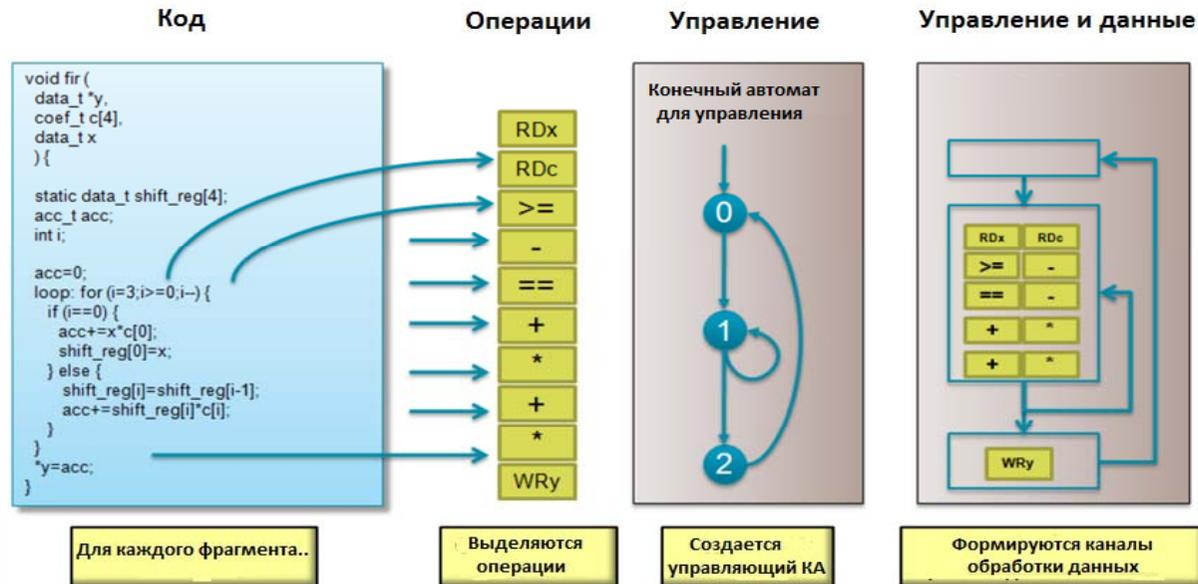
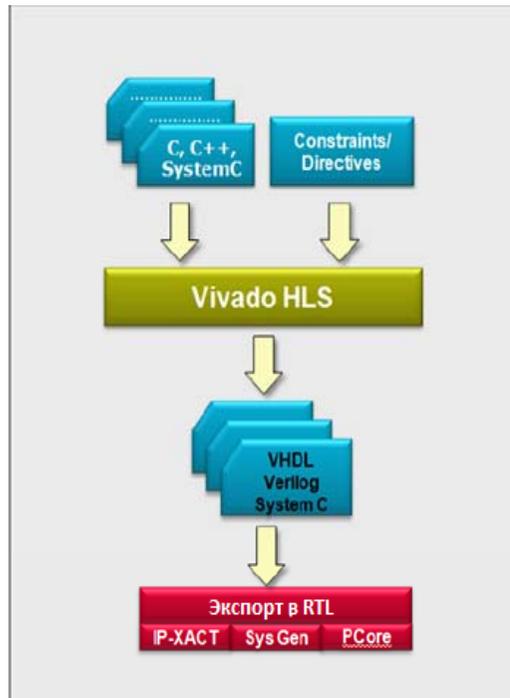


## Специализированные процессоры



Ряд публикаций посвящен сосуществованию ARM и MicroBlaze в платформе Xilinx Zynq7000

# Vivado HLS

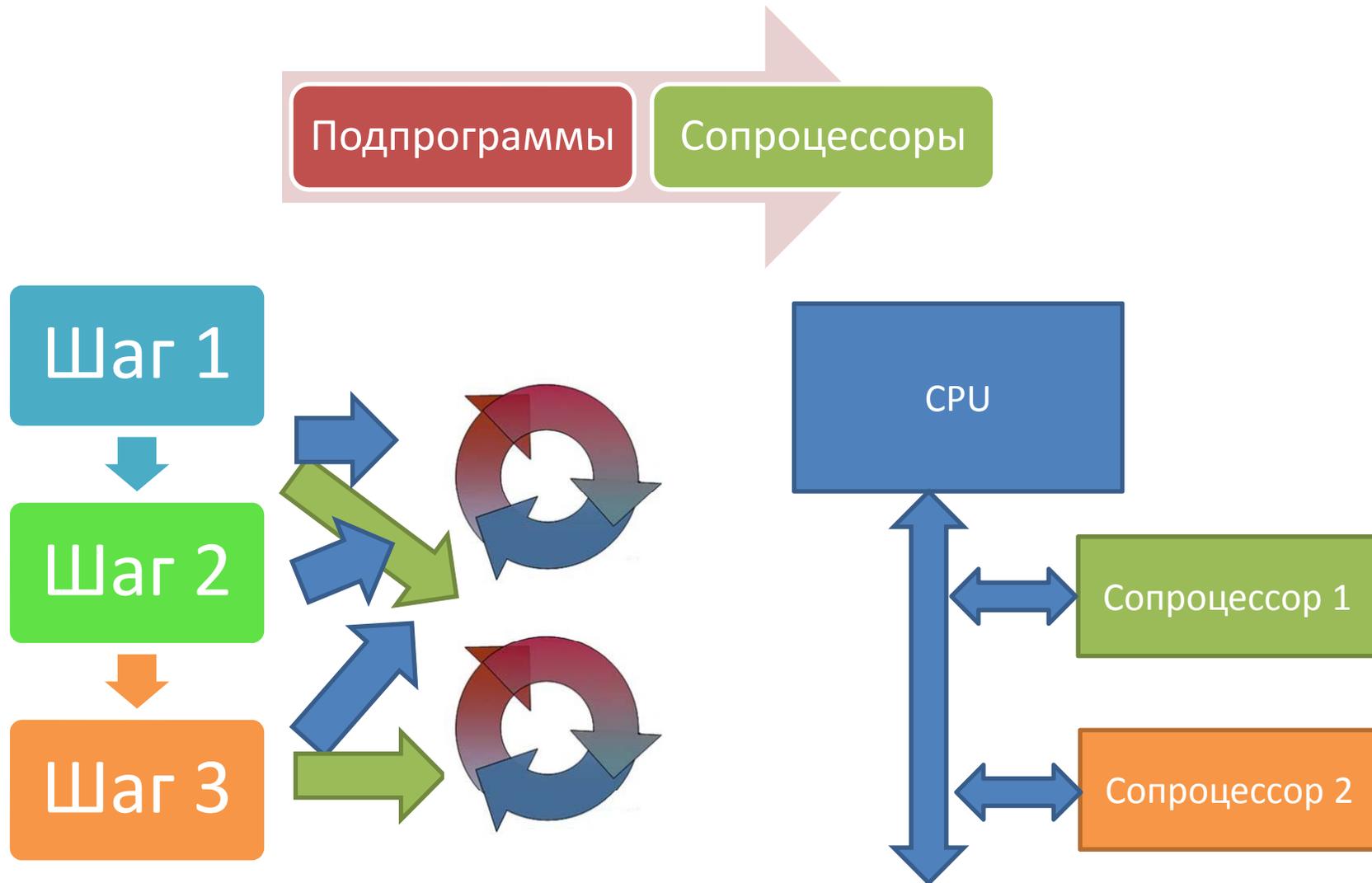


# 1. DSP-контроллер

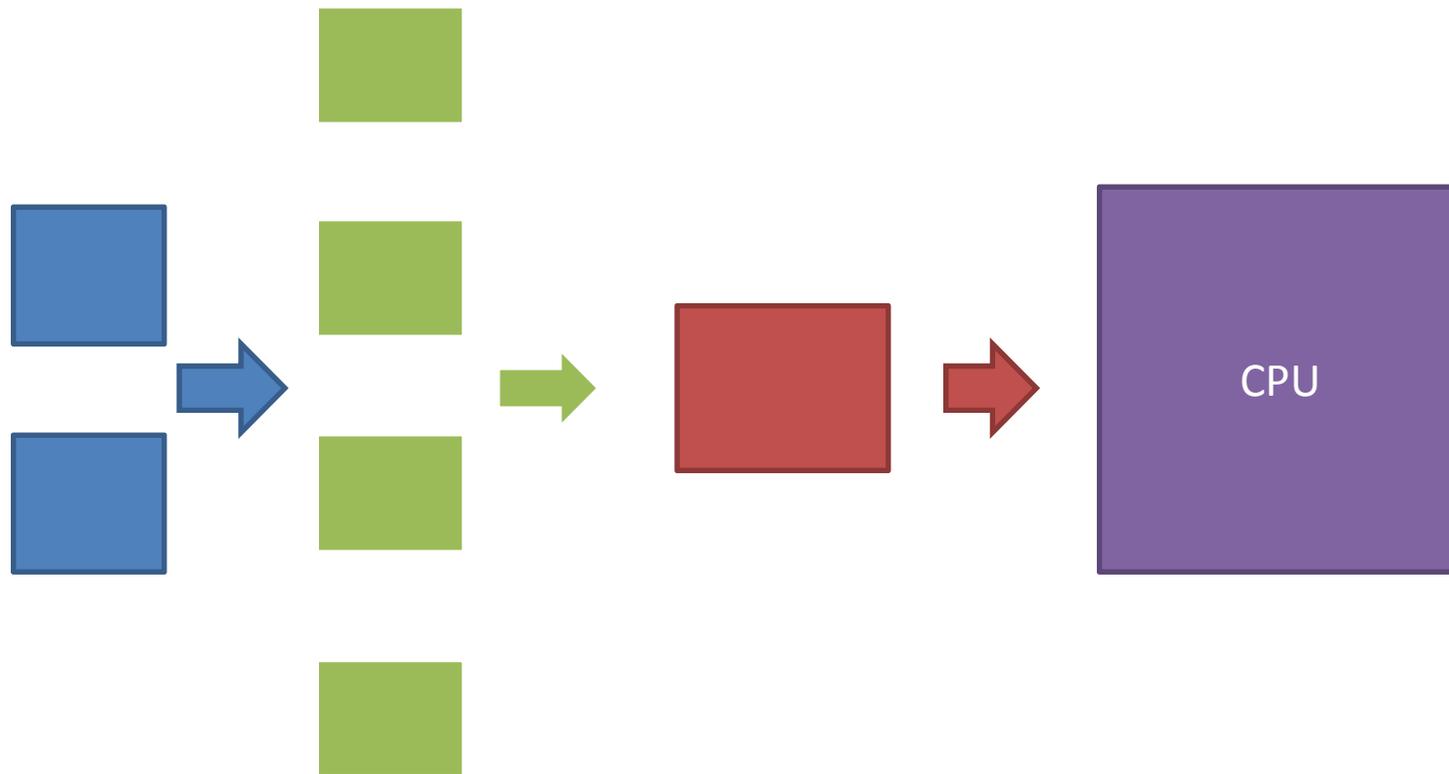
- Параллельная цифровая обработка обеспечивает большое число операций за такт
- Приоритет для процессора – управление многопоточной цифровой обработкой (возможно, SIMD)



## 2. Встраиваемый контроллер для системы реального времени



### 3. Аппаратный ускоритель вычислений для подкласса задач



На отдельных этапах вычислений требуется разное количество вычислительных узлов для обеспечения равномерной загрузки

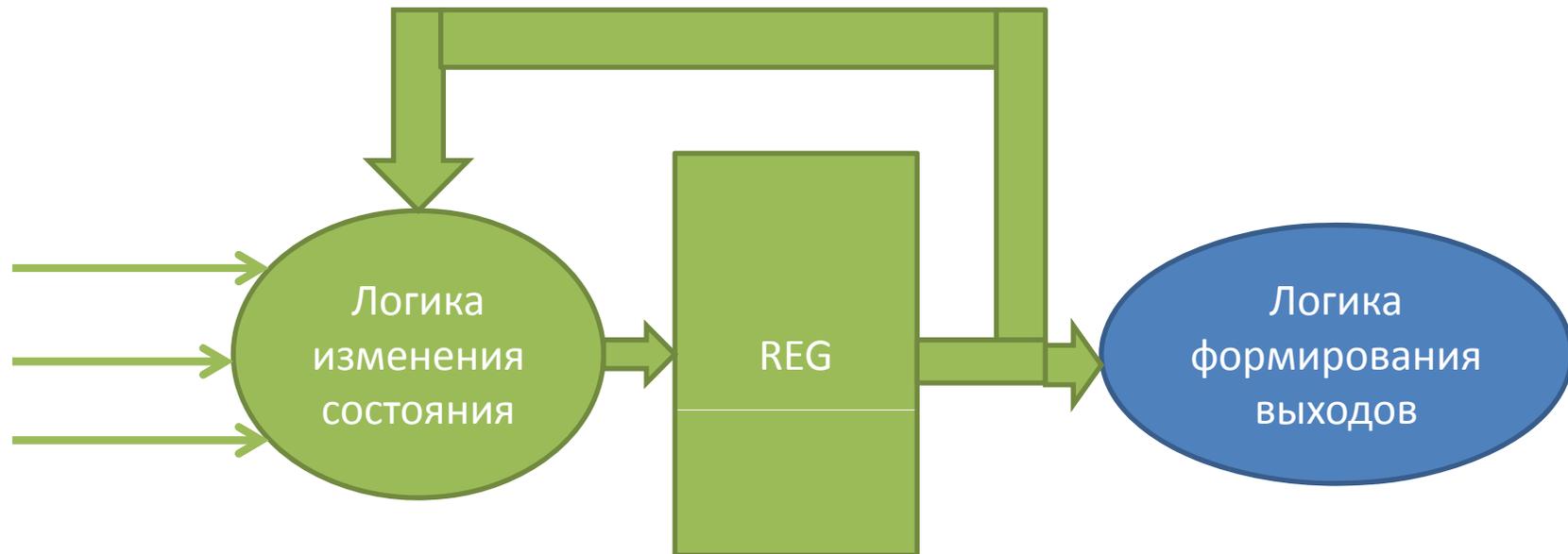
# Промежуточные выводы

- В системах на базе FPGA обеспечение производительности, функциональных возможностей и реакции в реальном времени происходит за счет специализированных вычислительных узлов
- Процессор:
  - Производительность 
  - Совместная оптимизация с ПО 
  - Сокращение цикла проектирования 
- **Проблемы**
  - Получение оптимального расположения компонентов и трассировки связей (Fmax)
  - Инструментальная поддержка разработанных процессоров

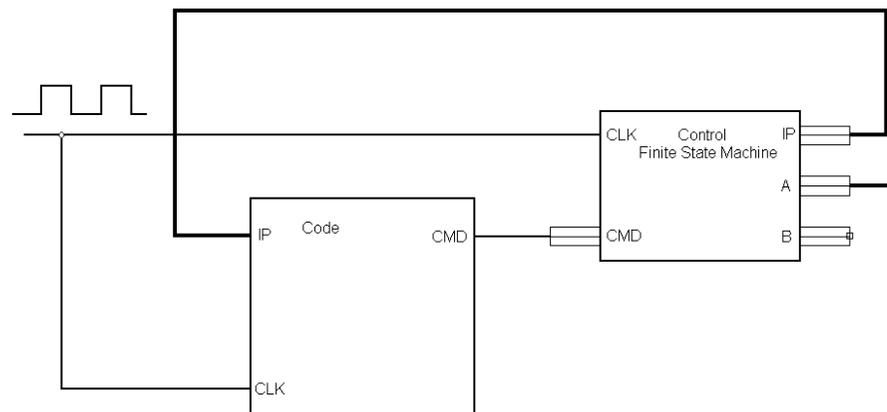
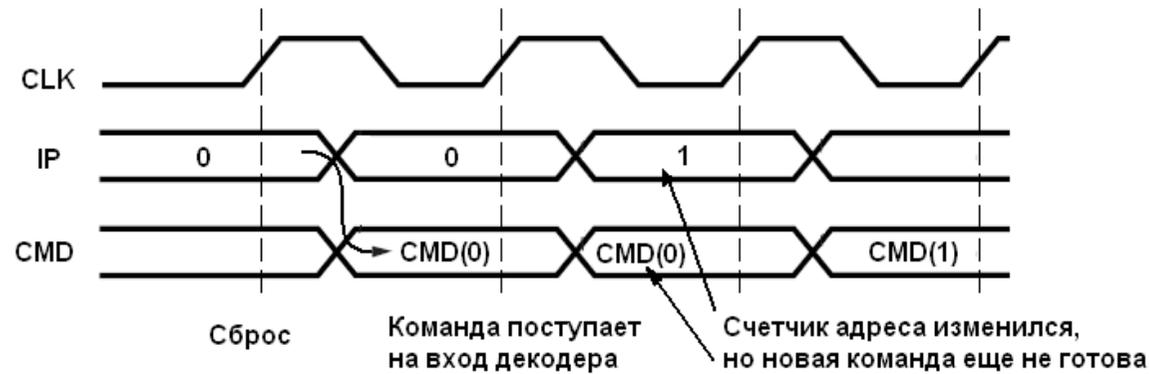
# Достижение $F_{max}$



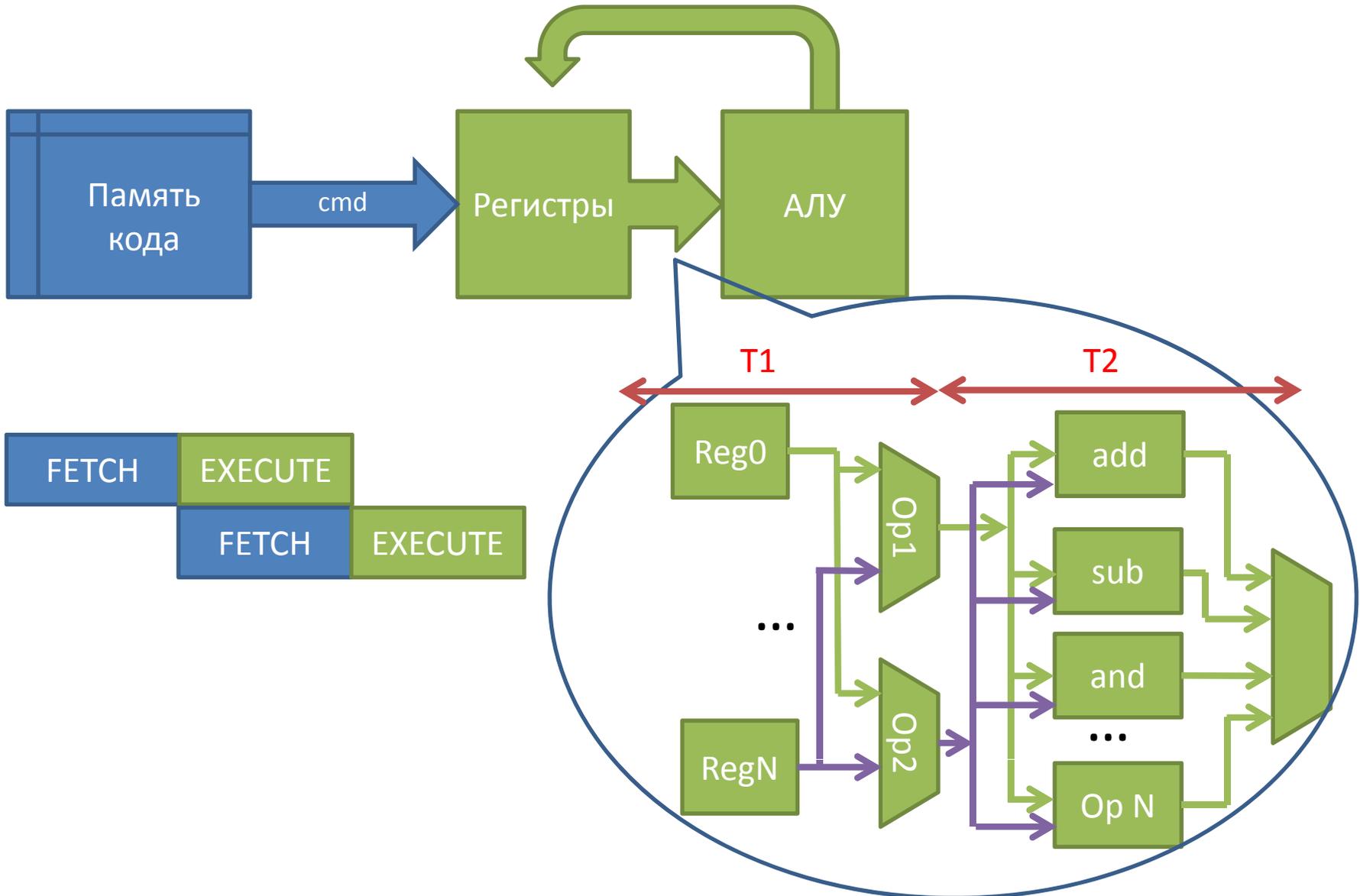
# Конечный автомат



# Временные диаграммы и схема простейшего синхронного процессора



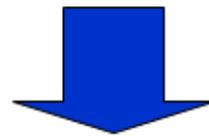
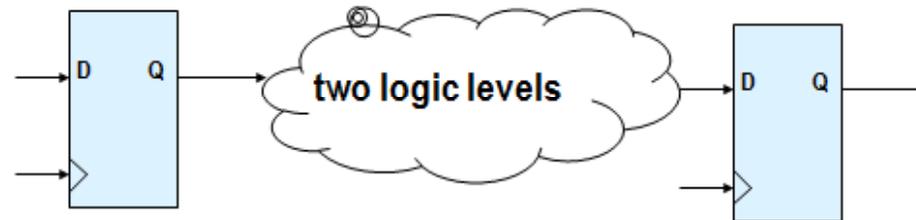
# Двухступенчатый конвейер



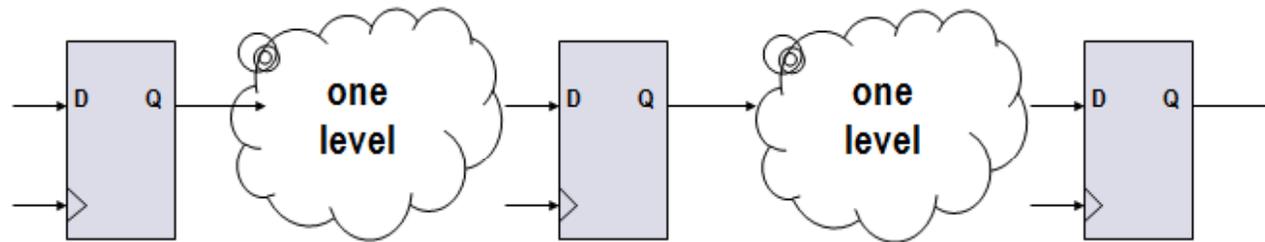
# Конвейеризация

- Конвейеризация уменьшает время распространения сигнала между синхронными элементами

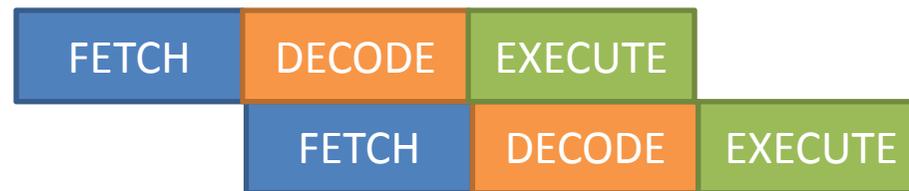
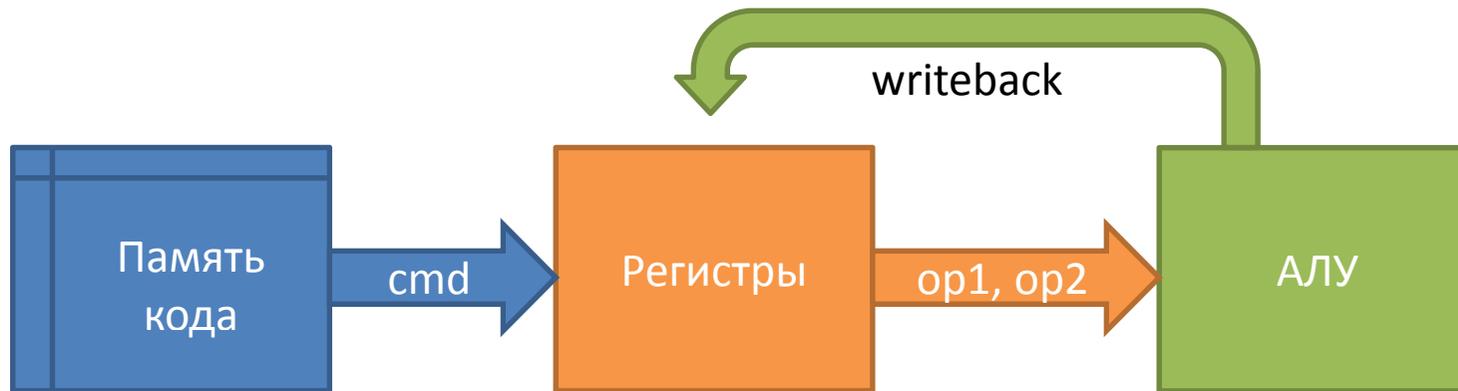
$$f_{MAX} = n \text{ MHz}$$



$$f_{MAX} \approx 2n \text{ MHz}$$



# Трехступенчатый конвейер



# Увеличение числа ступеней конвейера

- 5-ступенчатый конвейер предусматривает отдельные стадии для работы с памятью

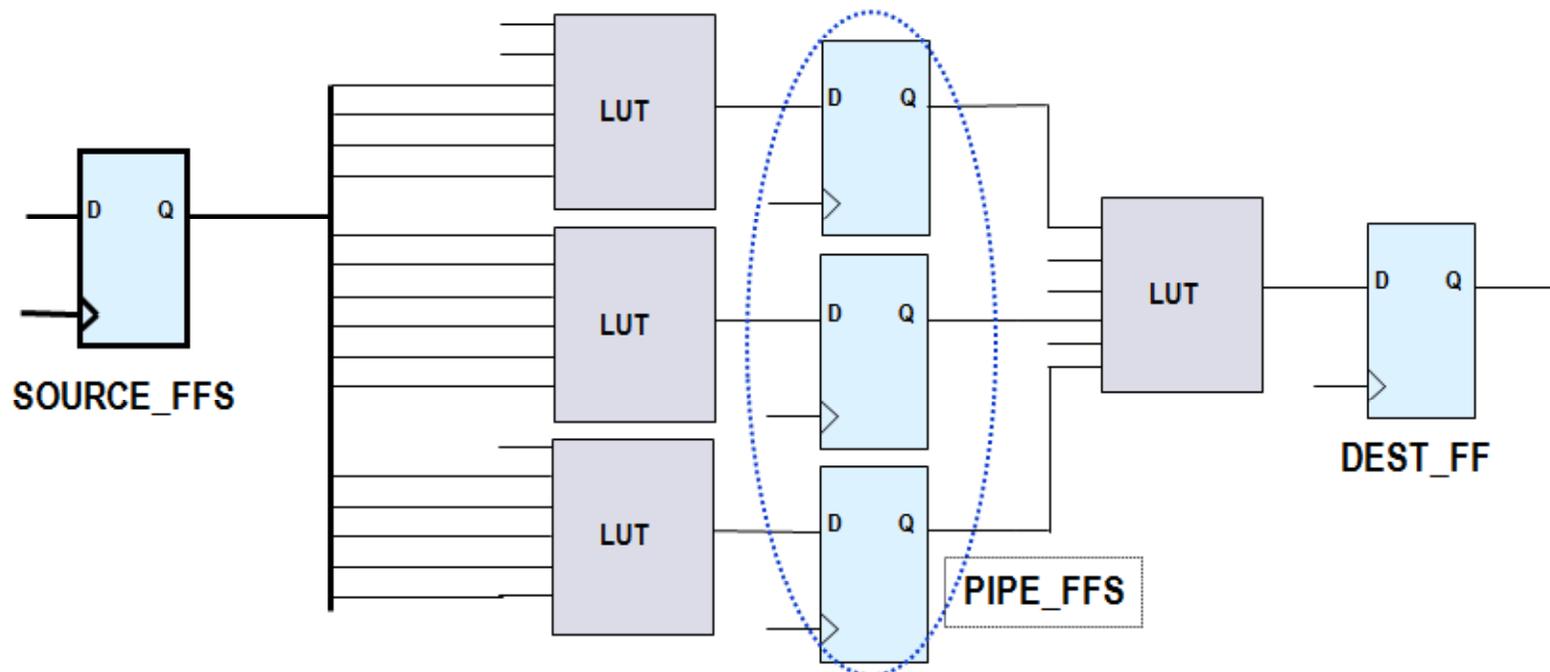


- Дальнейшее увеличение числа ступеней конвейера предусматривает конвейеризацию внутри отдельных блоков



# Конвейеризация в FPGA

- Гранулярность FPGA не позволяет уменьшать задержку после достижения *logic level = 1*



# Конвейеризация в FPGA

- Увеличение числа стадий конвейера приводит к появлению «пузырей» из-за конфликтов по данным и архитектурных конфликтов
- С увеличением числа стадий конвейера растет сложность управляющей схемы
- Нет возможности изменить задержку в логических блоках FPGA
- Процессоры в FPGA не получают существенных преимуществ при увеличении числа стадий конвейера

# Формирование системы команд

Трехадресная команда (PowerPC, Microblaze, ARM)



Двухадресная команда (x86)



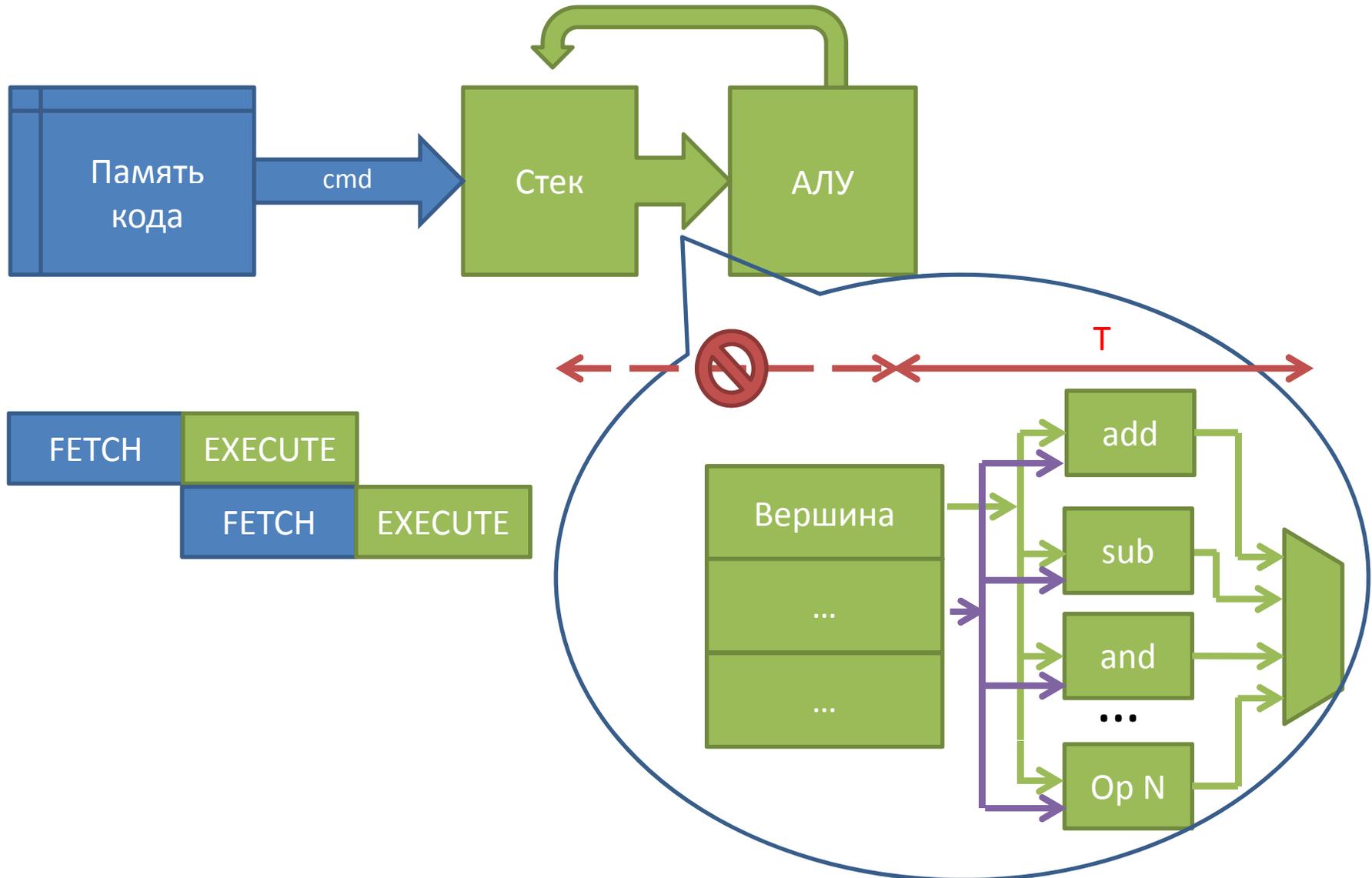
Одноадресная команда (TI TMS320)



Безадресная команда (транспьютер, стековые процессоры)



# Двухступенчатый конвейер стекового процессора



# Архитектура ядра

Адресность	2-ступенчатый конвейер	3-ступенчатый конвейер	5-ступенчатый конвейер
0-адресный	+		
1-адресный		+	
2-адресный		+	+
3-адресный		+	+

- Не существует универсальной архитектуры, подходящей для решения любых задач
- FPGA делают привлекательным выбор архитектуры ядра для конкретной задачи, основываясь на регулярных подходах к проектированию

# Совместная разработка аппаратного и программного обеспечения

- Выделяются ключевые особенности задачи (операции, формулы, вычисления, протоколы и др. характерные особенности, прямо влияющие на работоспособность и эффективность)
- Если существует явное выражение для часто используемой операции, эта операция может быть оформлена как команда процессора
  - *Пример:  $Result \leq (a + b) * k - (c \text{ and } d);$*
- Если протоколы аппаратных взаимодействий сложны или могут измениться, рекомендуется реализовать операцию программным путем

# Когда использовать софт-процессор?

- ПЛИС медленнее и дороже сопоставимого по характеристикам микроконтроллера
- Организационные причины использования софт-процессора
  - Наличие собственного продукта
  - Минимизация списка компонентов
- Технико-экономические причины
  - Параллельная обработка
  - Создание уникальной архитектуры и системы команд
  - Многопроцессорная система на кристалле
  - Минимизация числа корпусов

## «...совместная разработка аппаратного и программного обеспечения»

- Выбор архитектуры комплекса базируется на возможностях инструментального ПО
  - Для компилятора, разрешающего конфликты по данным, оптимален процессор с симметричной 3-адресной системой команд
  - Для компилятора, использующего стековую вычислительную модель, оптимален процессор со стековой архитектурой
- Для решения вопроса используется моделирование, основанное на *ранней интеграции аппаратного и программного обеспечения*

Пример

# Язык программирования Forth

- **Задачи проектирования**
  - Автоматизированная генерация описаний на HDL
  - Быстрое прототипирование кросс-ассемблеров для разрабатываемых специализированных процессоров
- **Преимущества Forth**
  - Быстрая разработка транслятора, доступная специалисту в смежной с CS области
  - Реализация кросс-ассемблеров простым расширением базового словаря
  - Хорошо проработанная структура транслятора и вычислительная модель



# Forth в КГТА



- 1998 г.
- MS-DOS protected mode, 32 bit
- SP-Forth

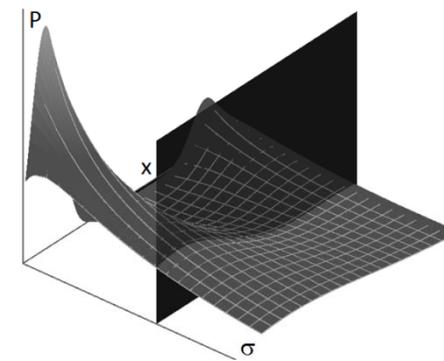
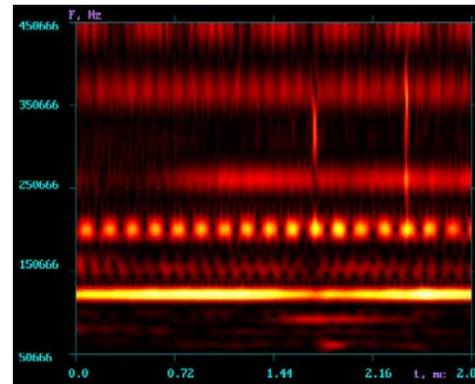
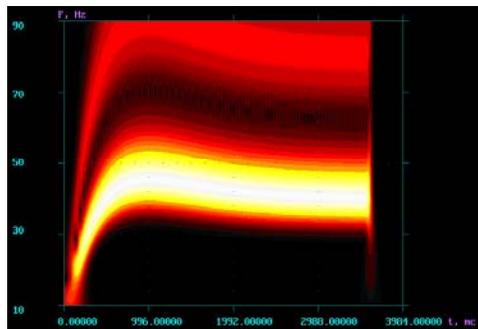
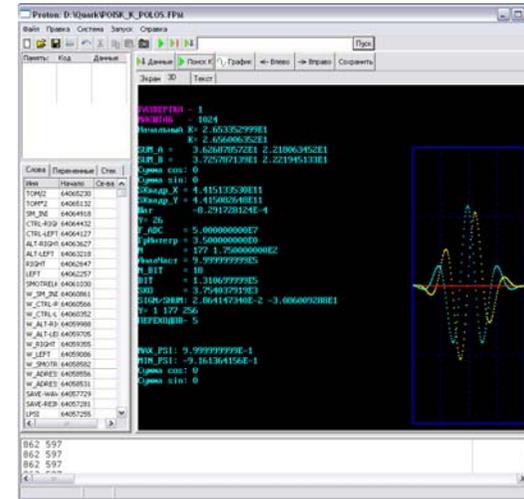
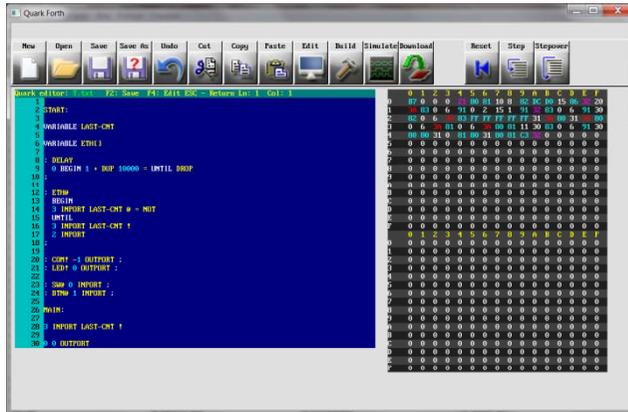


- 2006 г.
- Windows XP/Vista/7/8
- FASM

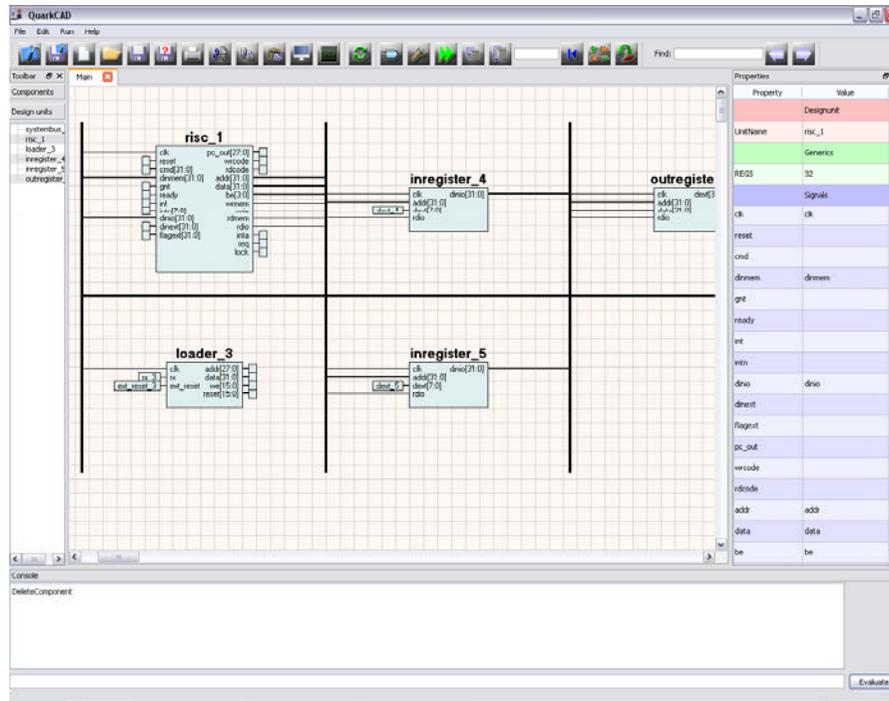


- 2009 г.
- Windows/Linux
- C++

# Кросс-компиляция, моделирование, визуализация результатов измерений



## САПР системного уровня (Qt + Quark VM)

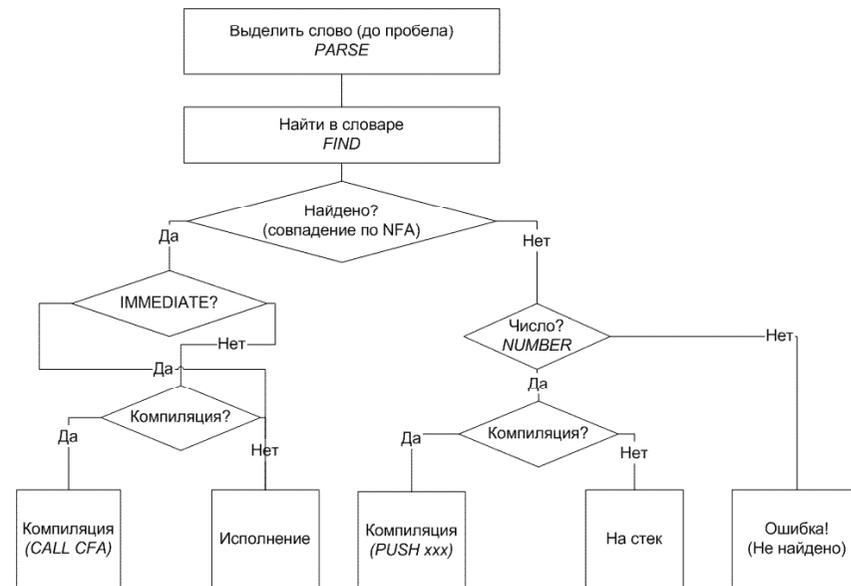
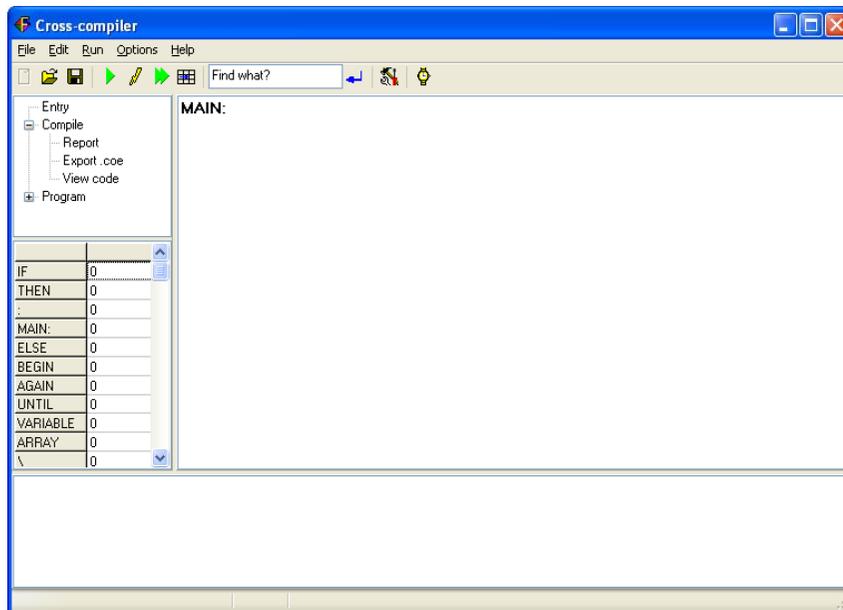


- ОКР «Разработка архитектуры и основных компонентов унифицированной параметризованной платформы для высокопроизводительных «систем-на-кристалле» шифр 2007-9-2.7-00-01-003 (основание для проведения ОКР - решение Конкурсной комиссии Роснауки № 24 протокол № 6 от 24 августа 2007 г
- Реализованы IP-ядра процессоров
  - RISC
  - Векторно-скалярный
  - Матрица процессорных элементов

Учтены существующие тенденции:

- Ранее связывание программного и аппаратного обеспечения
- Моделирование на уровне транзакций

# Программная поддержка – нуль-операндные ассемблеры на базе языка Форт



## Процессорное ядро kf – стековый процессор с высокой плотностью кода

	Kf116	Kf216	kf332	kf432	kf500
Данные	16 бит	16 бит	32 бита	16/32/64 бит	8-128* (параметр)
Аппаратные стеки	Данных Возврат ОВ	Данных Возвратов	Данных Возвратов Циклов	Данных Возвратов Циклов float	Данных Возвратов Циклов
Команда	16 бит	16 бит	18 бит	8 бит	6 бит
Тактовая частота и платформа	Spartan 10 MHz	Spartan-2E 10 MHz	Spartan-3 40 MHz Virtex-5 80 MHz	-	Spartan-6 100 MHz Kintex-7 160 MHz
Число проектов	-	<b>1</b>	<b>&gt;10</b>	-	<b>&gt;10</b>

# Фрагмент программы, настраивающий целевой компилятор на систему команд софт-процессора

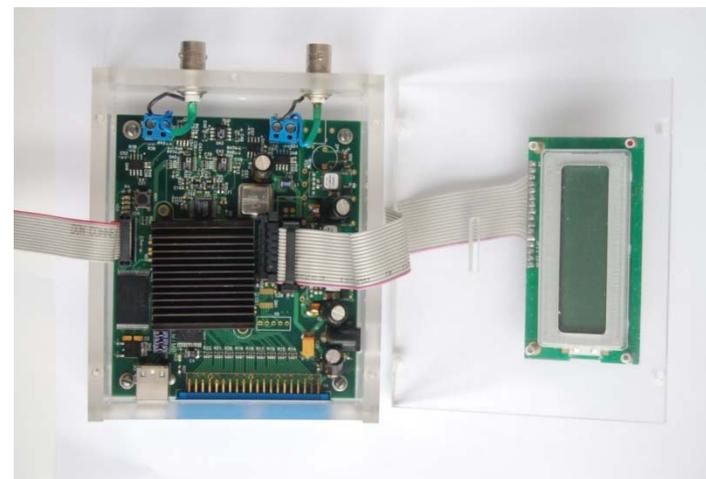
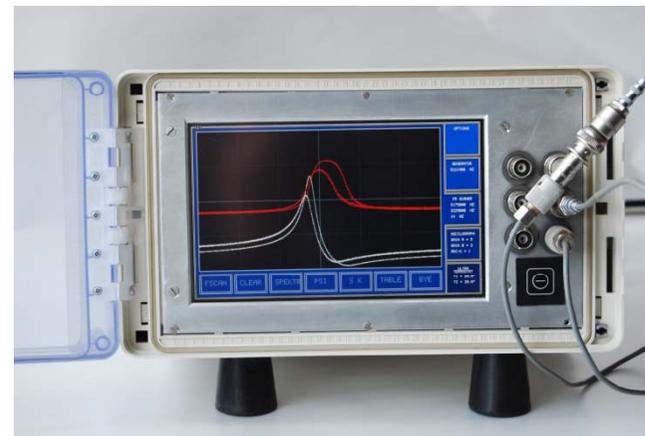
: CMD CREATE , DOES> @ TC-C, ;

...

KF532 DEFINITIONS

0 CMD NOP	: LOOP 11 ZC, 0 ZC, ;	22 CMD DROP
1 CMD NOT	12 CMD SYSREG@	23 CMD cmdJMP
2 CMD2 @	13 CMD +	24 CMD cmdCALL
3 CMD SHL	14 CMD -	25 CMD cmdRJMP
4 CMD SHR	15 CMD AND	26 CMD >R
5 CMD SHRA	16 CMD OR	27 CMD !
6 CMD2 INPORT	17 CMD XOR	28 CMD DO
7 CMD SWAP	18 CMD =	29 CMD cmdRIF
8 CMD DUP	19 CMD <	30 CMD cmdUNTIL
9 CMD OVER	20 CMD >	31 CMD RET
10 CMD R>	21 CMD2 *	

# Разработки на базе ПЛИС под управлением форт-процессоров



# Постановка задачи обеспечения программной поддержки вновь разрабатываемых процессорных устройств

## Текущая ситуация:

- Сроки разработки процессорных ядер и вычислительных систем существенно сокращаются с применением FPGA
- Отсутствие инструментов разработки программного обеспечения является главным сдерживающим фактором
- Перенацеливаемые (retargetable) компиляторы (gcc, llvm) имеют относительно высокую трудоемкость адаптации к новым процессорным ядрам

Постановка задачи обеспечения программной поддержки вновь разрабатываемых процессорных устройств

- Требуются решения по адаптации ЯВУ к гибко настраиваемым процессорным ядрам, с возможностью:
  - Выбора типа конвейера
  - Выбора регистровой модели
  - Гибкого управления системой команд