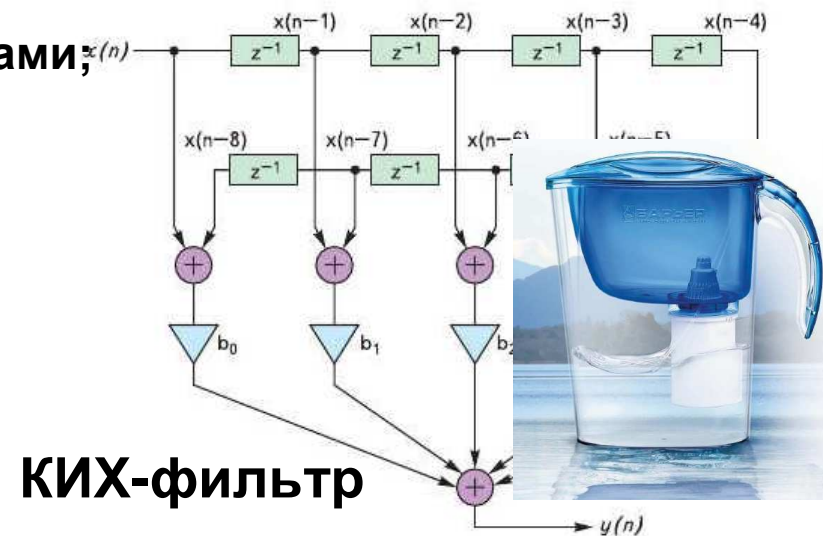


## Цели при реализации задачи 2 – генератора эха:

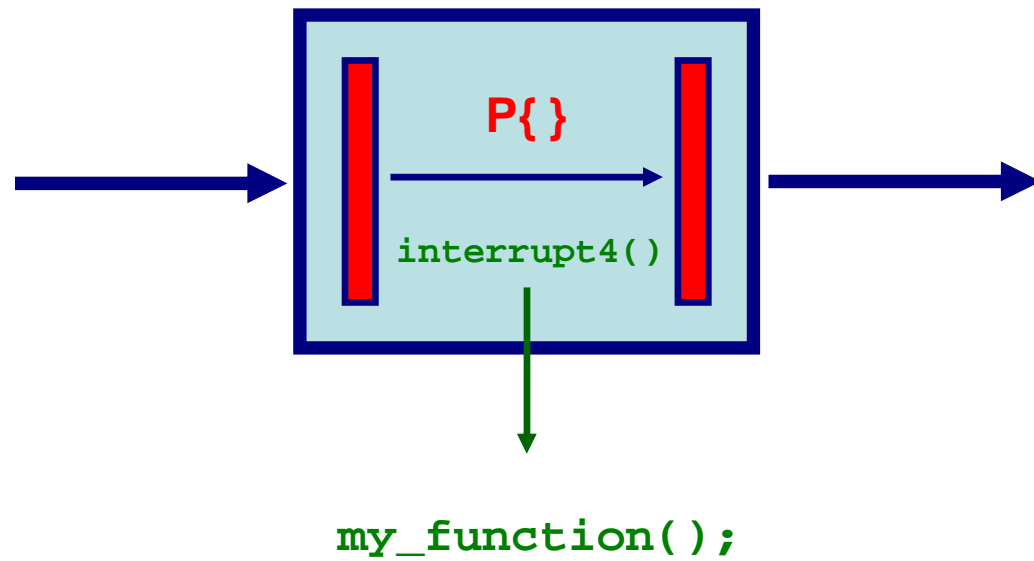
- знакомство со средой программирования;
- знакомство с ассемблером, вычислительными блоками и памятью;
- оценка затрат времени и памяти;
- оптимизация и работа в реальном масштабе времени.

## Цели при реализации задачи 3 – КИХ-фильтр:

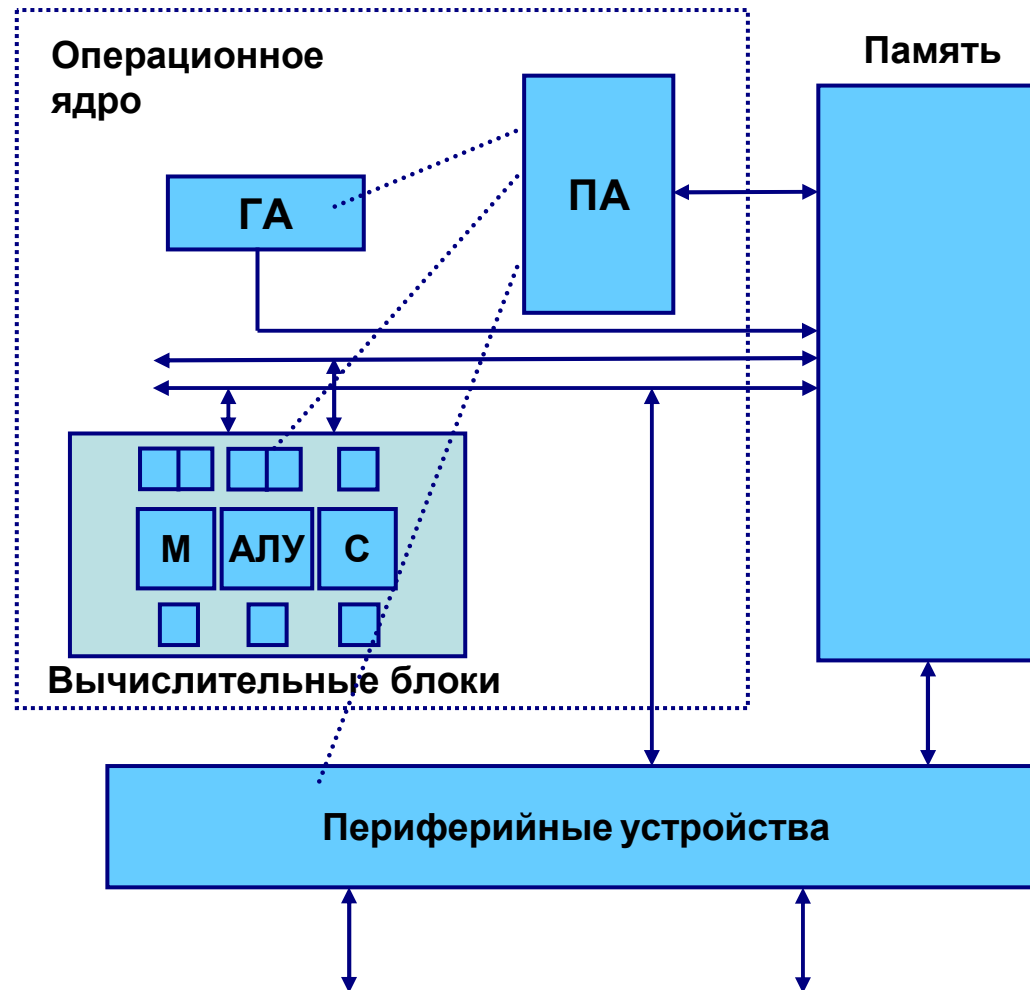
- изучение ассемблера и архитектуры;
- построение графиков в среде CCS;
- командный конвейер;
- приемы оптимизации циклов на ассемблере и Си;
- циклическая адресация;
- работа с периферийными устройствами;
- прерывания.



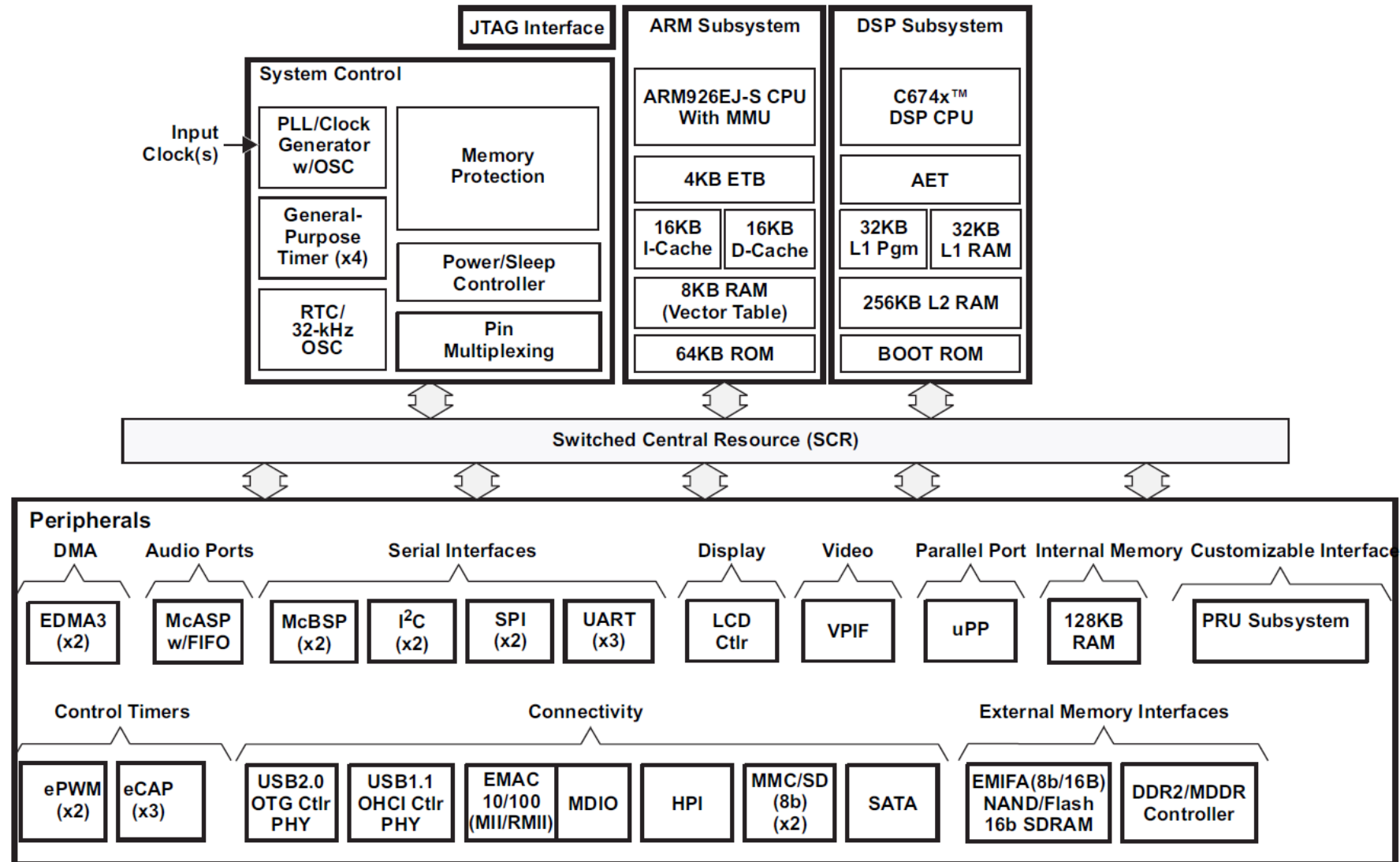
## Базовый (шаблонный) проект



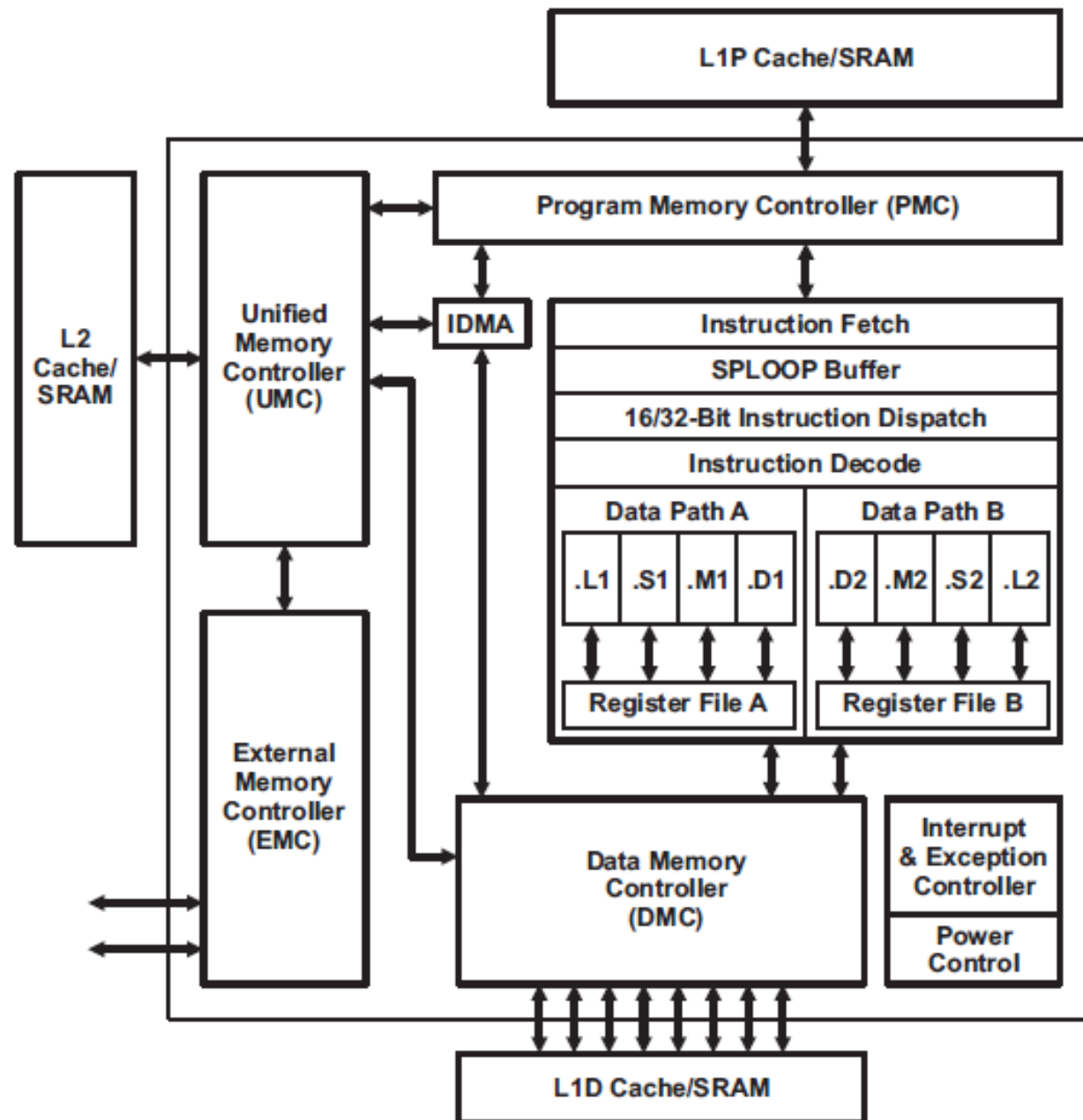
$$A = A + B \times C$$



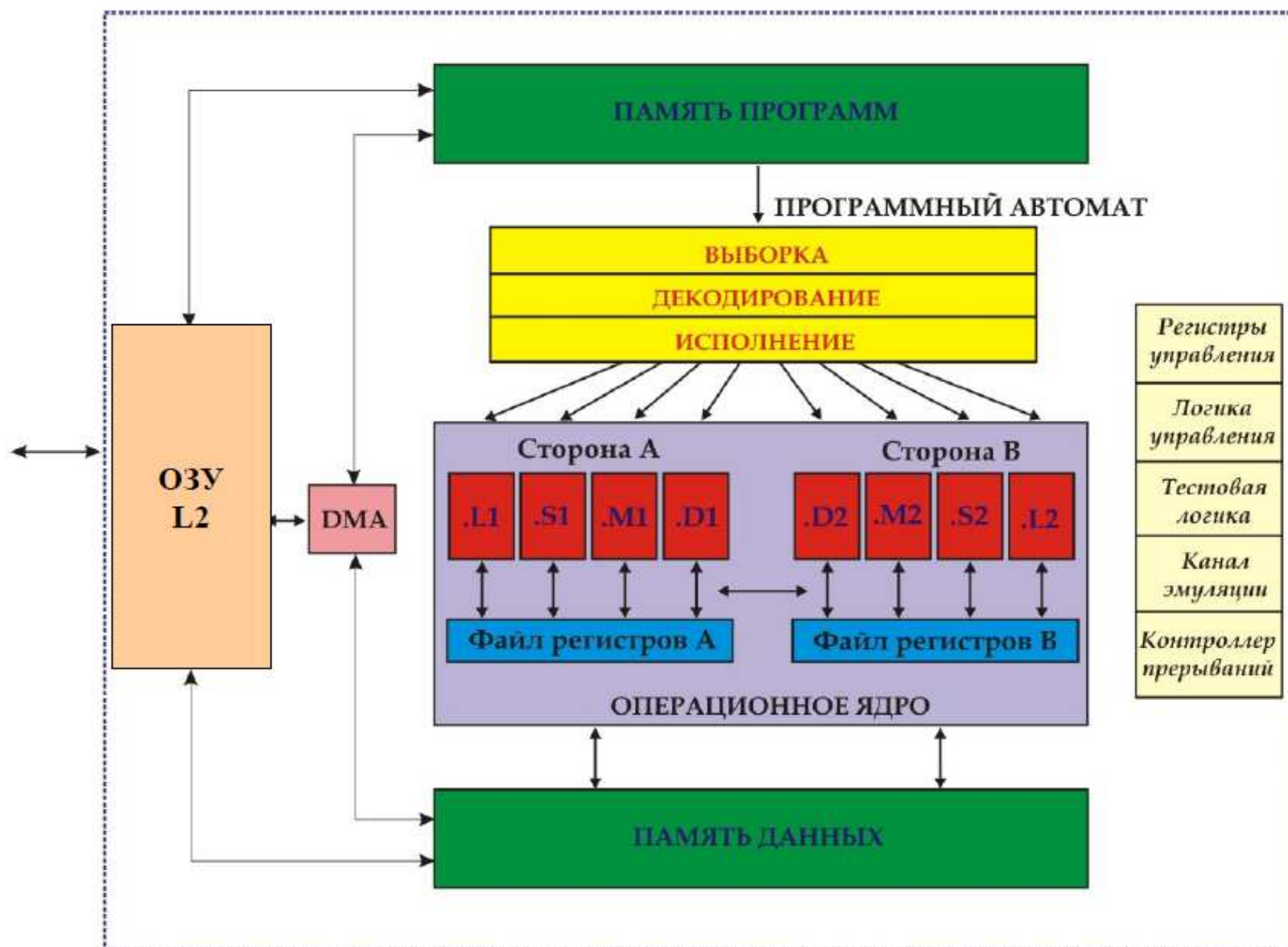
# Архитектура процессора OMAP-L138



## Подсистема DSP



## Подсистема DSP



# Задача 3 – КИХ-фильтр

## 1. Постановка задачи – частотная селекция сигнала:

пусть имеем сумму двух гармонических сигналов с частотами  $f_1 = 50$  Гц и  $f_2 = 1000$  Гц. Требуется разработать устройство, осуществляющее подавление одного из этих сигналов и передачу другого на выход без изменений.

## 2. Математическое описание алгоритма:

$$y[n] = \sum_{k=0}^{N-1} x[n-k] \cdot h[k]$$

## Блок-схема алгоритма и программа

*Расчет коэффициентов !*

**Построение базового проекта; тестирование**

**Модификация проекта: подключение новой функции**

**Построение, отладка нового проекта, отображение и анализ результатов**

**Обзор архитектуры процессора: память, регистры, вычислительные блоки, командный конвейер**

## Обзор архитектуры процессора: память, регистры, вычислительные блоки, командный конвейер

1. Найти в ассемблерном коде цикл свертки (адреса памяти)
2. Определить регистр, на котором происходит накопление
3. Найти регистр-счетчик итераций цикла
4. Найти адрес массива отсчетов и адрес ИХ
5. Определить время одной итерации цикла. За сколько тактов происходит одно умножение-накопление?
6. Причина – многотактное исполнение команд
7. Понятие командного конвейера

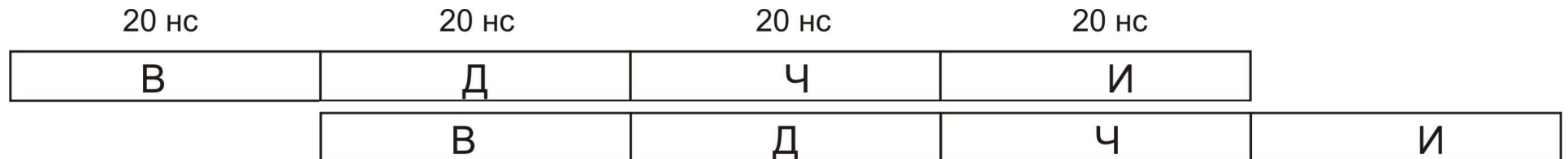
# Командный конвейер

MPY R1, 0x7540, R0

80 нс



- выборка команды из памяти;
- декодирование команды;
- чтение входных операндов;
- исполнение



# Командный конвейер

20 нс	20 нс	20 нс	20 нс
<b>В</b>	<b>Д</b>	<b>Ч</b>	<b>И</b>
LDW	X	X	X
MPY	LDW	X	X
ADD	MPY	LDW	X
NOP	ADD	MPY	LDW
SUB	NOP	ADD	MPY
MPY	SUB	NOP	ADD
NOP	MPY	SUB	NOP
STW	NOP	MPY	SUB

# Обзор архитектуры процессора: память, регистры, вычислительные блоки, командный конвейер

1. Найти в ассемблерном коде цикл свертки (адреса памяти)
2. Определить регистр, на котором происходит накопление
3. Найти регистр-счетчик итераций цикла
4. Найти адрес массива отсчетов и адрес ИХ
5. Определить время одной итерации цикла. За сколько тактов происходит одно умножение-накопление?
6. Причина – многотактное исполнение команд
7. Понятие командного конвейера
8. Проблема: разное время выполнения команд и зависимости между командами. Выход?
9. Программная конвейеризация – Software Pipelining
10. Необходимо интеллектуально подходить к компиляции – оптимизирующие компиляторы и автоматическая оптимизация

Cycle	Loop							
	1	2	3	4	5	6	7	8
1	LDW							
2	NOP	LDW						
3	NOP	NOP	LDW					
4	NOP	NOP	NOP	LDW				
5	NOP	NOP	NOP	NOP	LDW			
6	MV	NOP	NOP	NOP	NOP	LDW		
7	STW	MV	NOP	NOP	NOP	NOP	LDW	
8		STW	MV	NOP	NOP	NOP	NOP	LDW
9			STW	MV	NOP	NOP	NOP	NOP
10				STW	MV	NOP	NOP	NOP
11					STW	MV	NOP	NOP
12						STW	MV	NOP
13							STW	MV
14								STW

**Оптимизация**

**Файл ассемблера без оптимизации**

**Оценка времени обработки и затрат памяти**

**Включение автоматической оптимизации**

**Анализ файла ассемблера с оптимизацией - обратная связь компилятора**

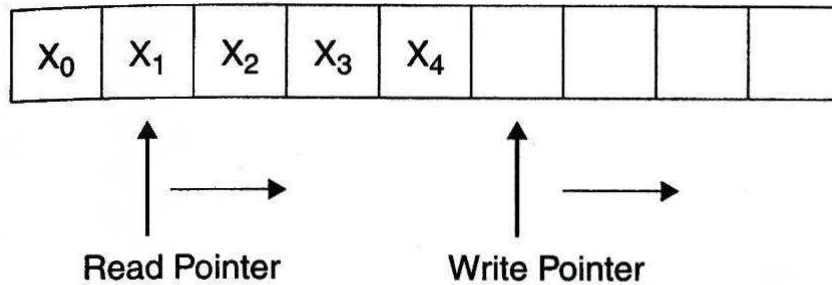
**Оценка времени обработки и затрат памяти после оптимизации**

**Пути дальнейшей оптимизации ?**

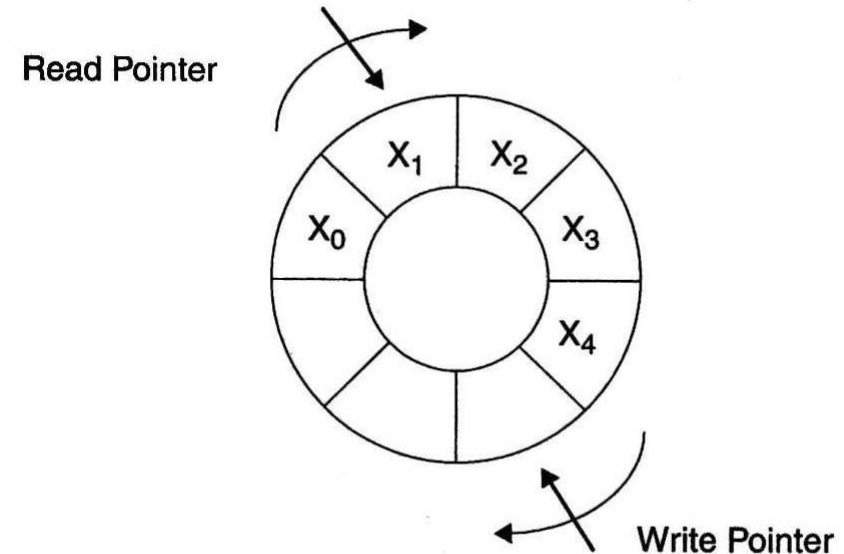
**Включение циклической адресации**

## Циклическая адресация в ЦСП

**Циклическая адресация** – при достижении конца массива данных, расположенного в памяти процессора, позволяет автоматически переводить указатель текущего элемента на начало массива.



**LDW .D1 \*A7++,A0**



**Построение, отладка нового проекта**

**Анализ кода и обратной связи компилятора**

**Оценка времени обработки и затрат памяти**

**Выводы о результатах оптимизации**

**Теоретический минимум времени обработки**

**Пути дальнейшей оптимизации**