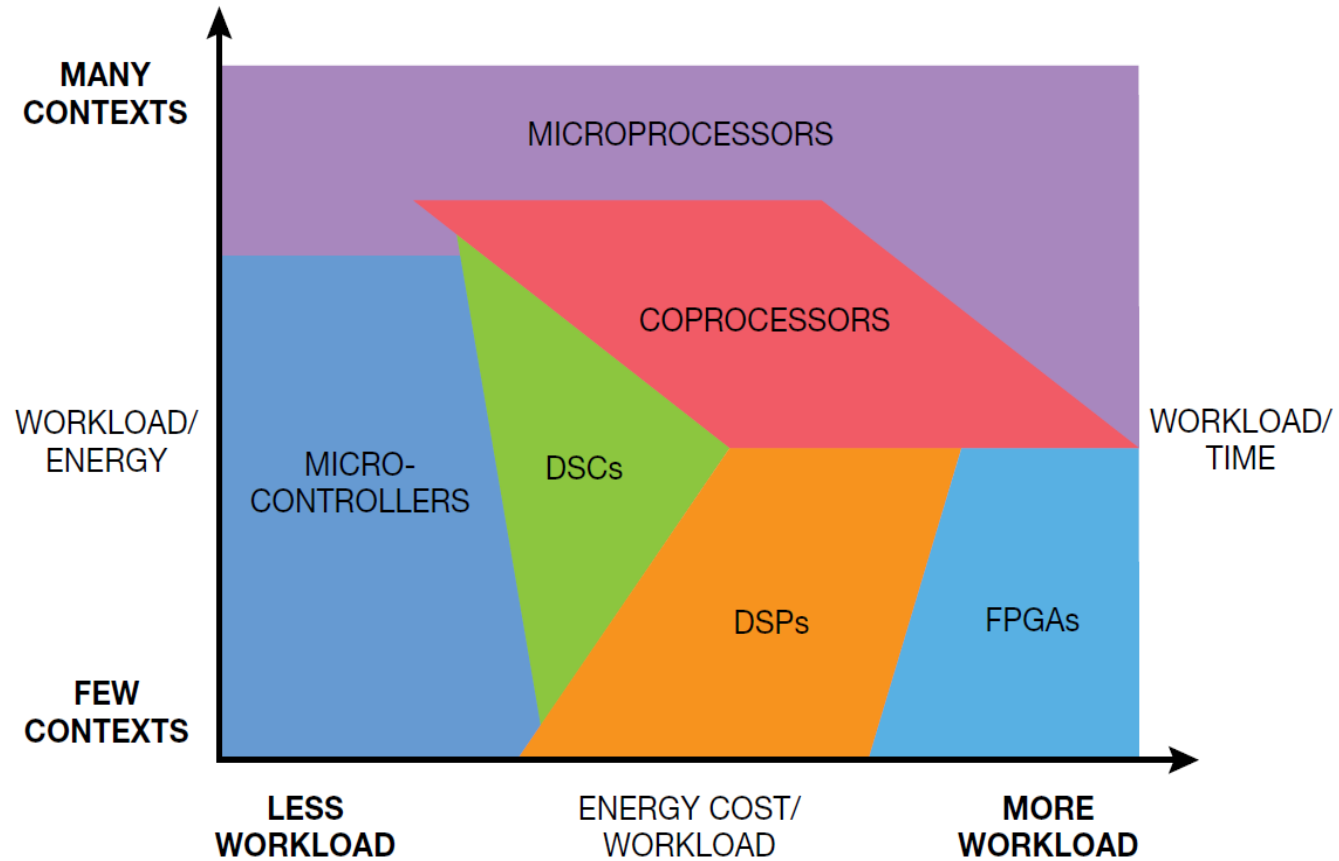


Обзор процессоров



- GPU

- SoC

Цифровой сигнальный процессор (ЦСП, DSP, ЦПОС, ПЦОС) — это микропроцессор, архитектура которого ориентирована на эффективную реализацию алгоритмов цифровой обработки сигналов в реальном масштабе времени

- 1. Быстрое выполнение операции умножения с накоплением.**
- 2. Архитектура памяти с множественным доступом.**
- 3. Специальные режимы адресации.**
- 4. Специальные возможности программного автомата.**
- 5. Специализированная периферия и интерфейсы.**

- 1. малое энергопотребление;**
- 2. малые габариты микросхемы;**
- 3. низкая стоимость**

Основная задача ЦСП — реализация алгоритмов цифровой обработки сигналов в реальном масштабе времени

Обработка сигнала в реальном масштабе времени — это обработка отсчетов входного сигнала в темпе их поступления (*пример с конвейером*)

Что же влияет на возможность работы процессора в реальном времени?

- быстродействие процессора
- сложность алгоритма
- скорость поступления данных

Условие работы в реальном времени:

Решение задач

Начинаем работать с ЦСП

Программное обеспечение

Языки ассемблер и Си

Средства разработки

Симуляторы

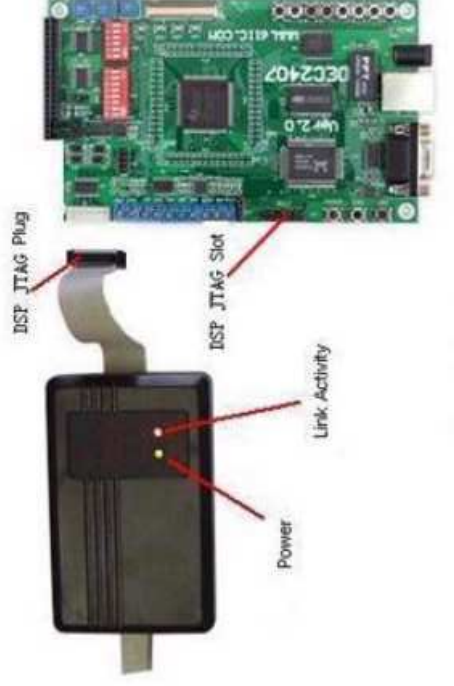
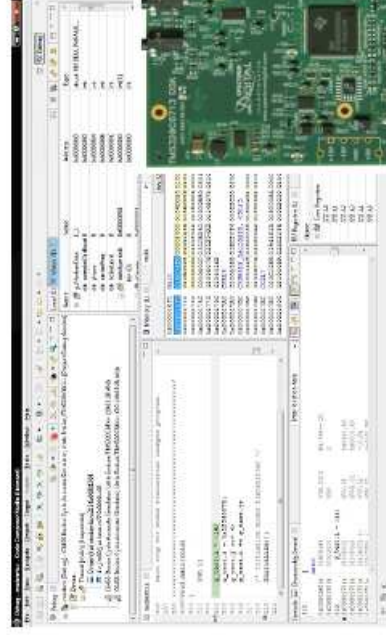
Стартовые наборы разработчика (Starter Developer Kit - DSK)

Отладочные модули (Evaluation Module — EVM)

Эмуляторы

Среды разработки (Code Composer Studio)

Открытые аппаратно-программные средства



Задача 2 – Генератор эха

1. Постановка задачи – добавление эха к звуковому сигналу в реальном времени

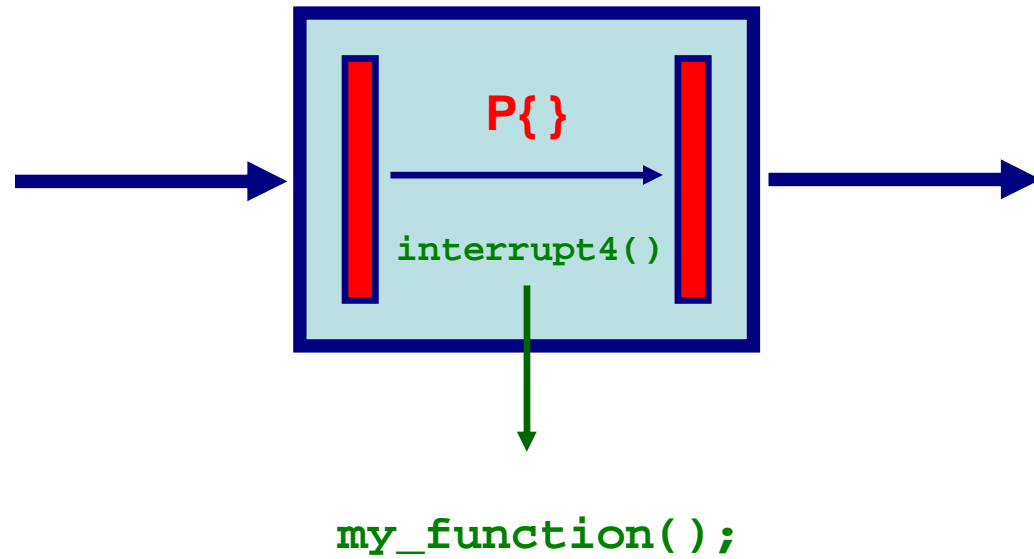
2. Математическое описание алгоритма:

$$r(n) = s(n) + s(n-N_0)$$

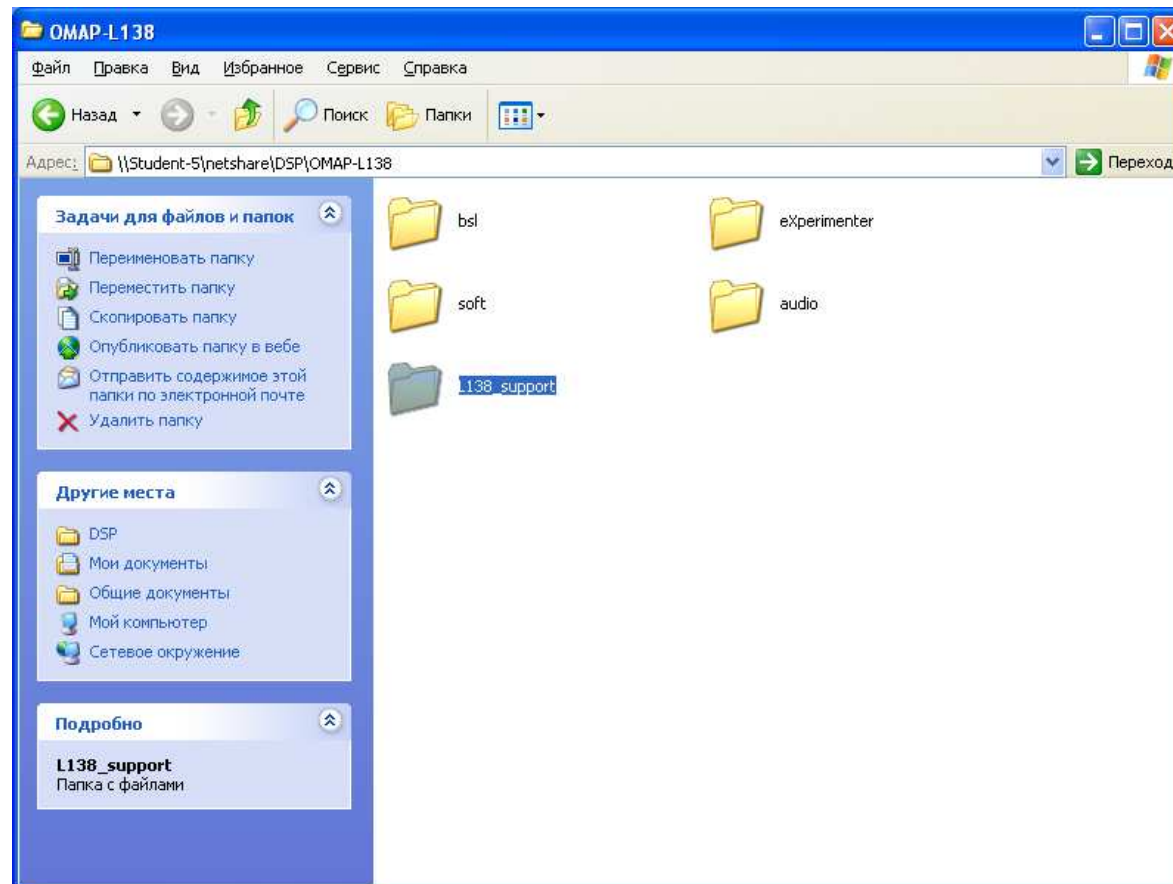
3. Блок-схема алгоритма и программа

Блок-схема алгоритма и программа

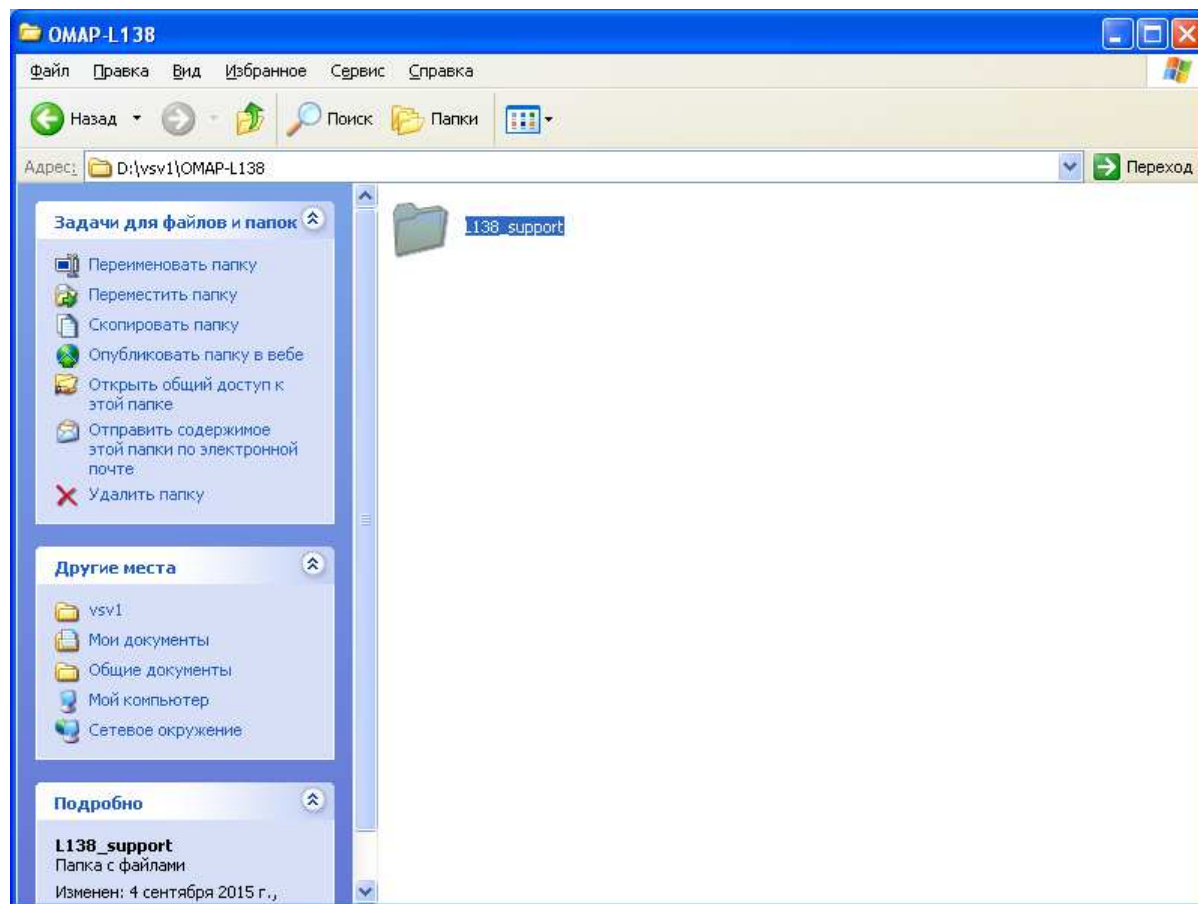
Знакомство с базовым проектом



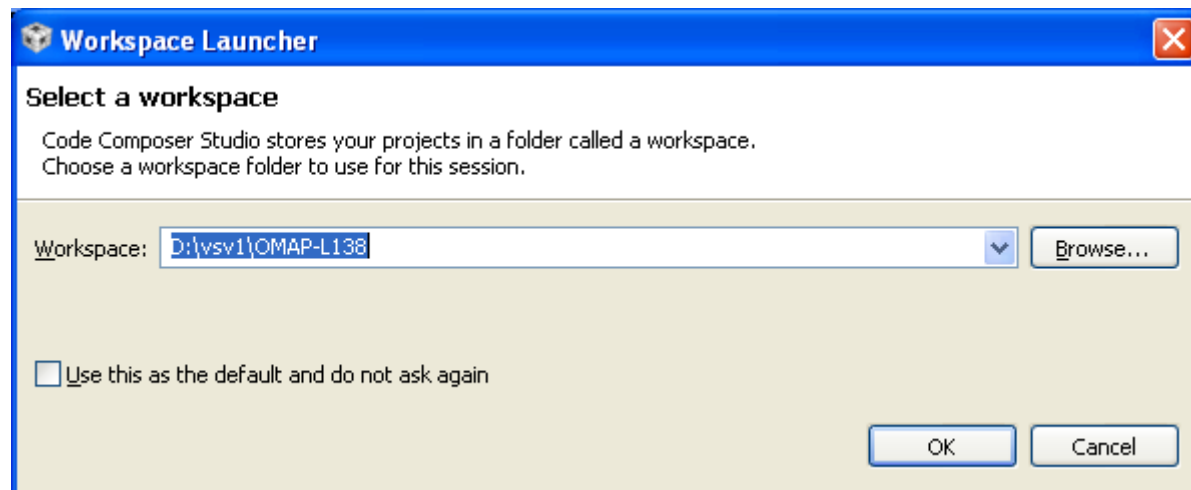
Копирование шаблонного проекта



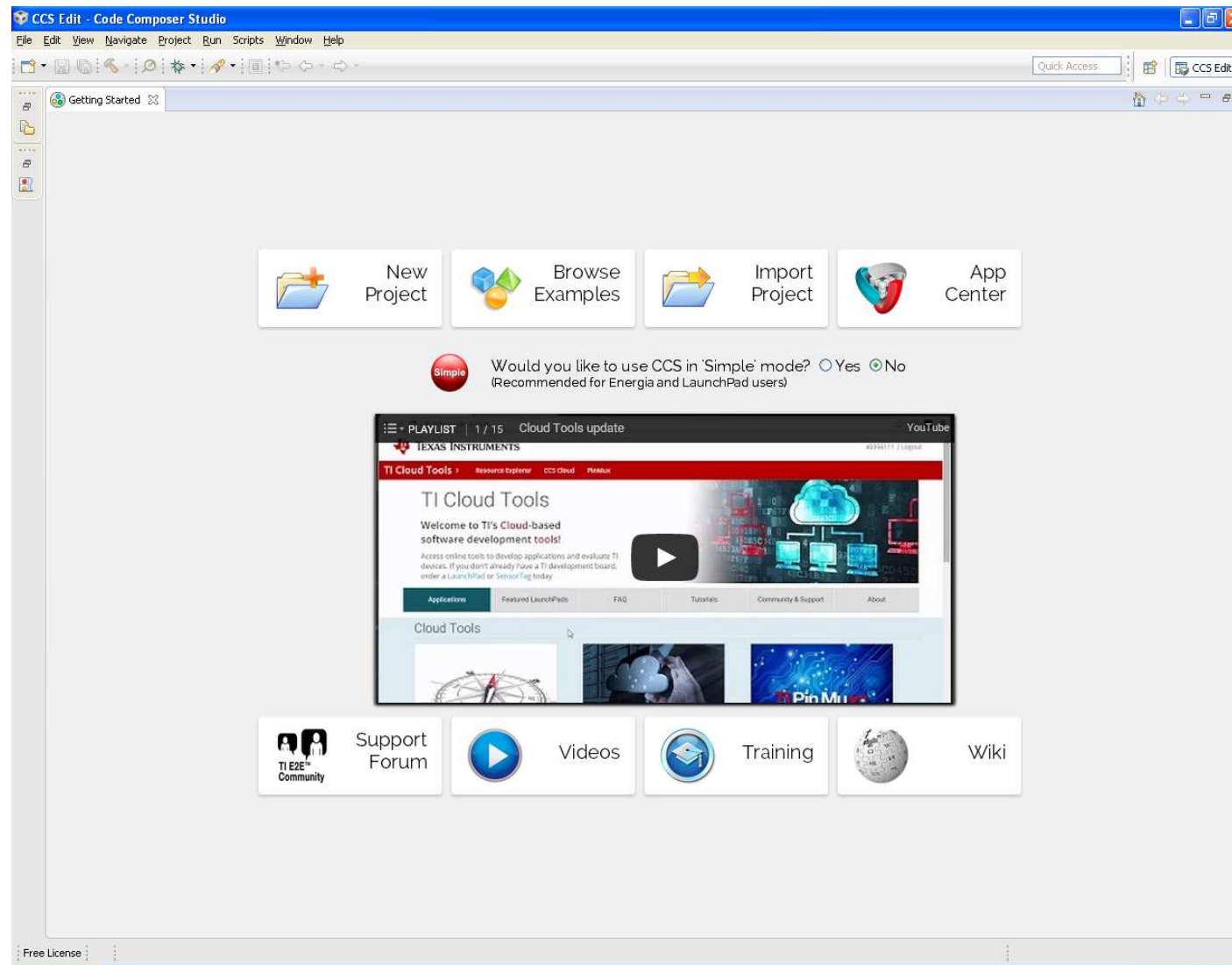
Копирование шаблонного проекта



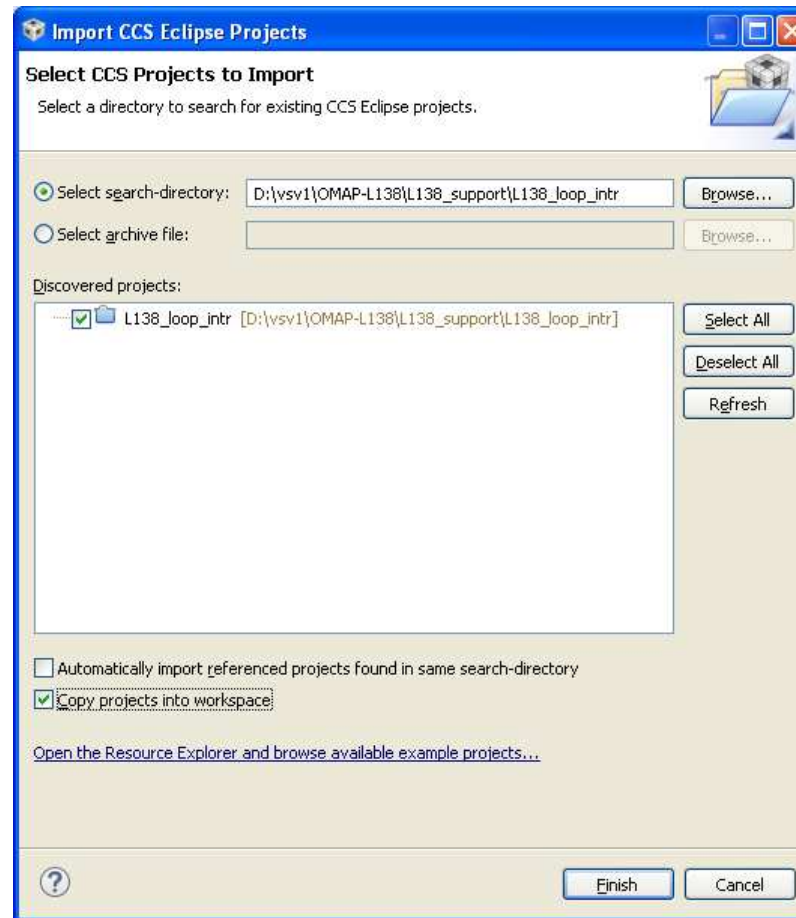
Открытие среды CCS, задание рабочей области



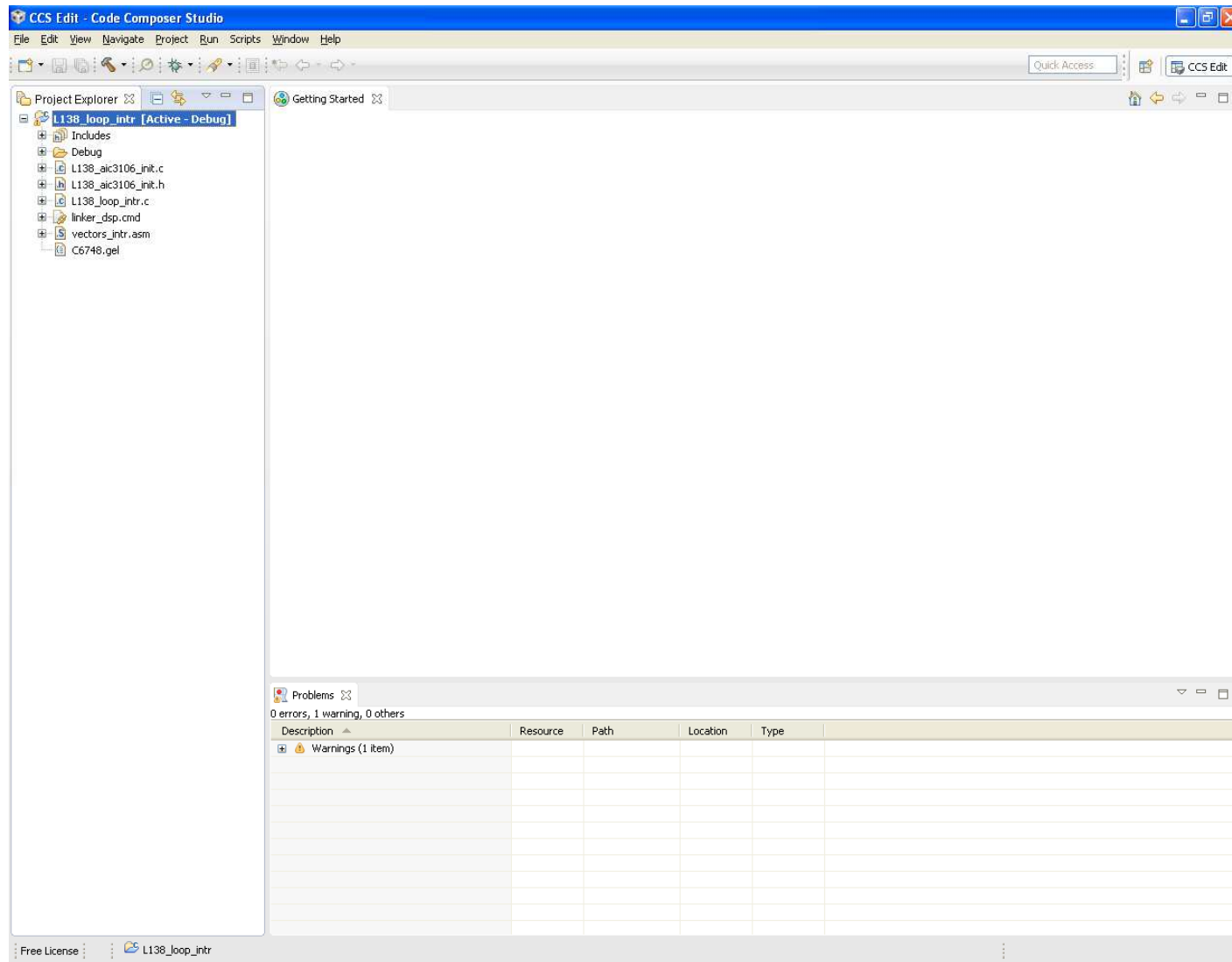
Импорт шаблонного проекта



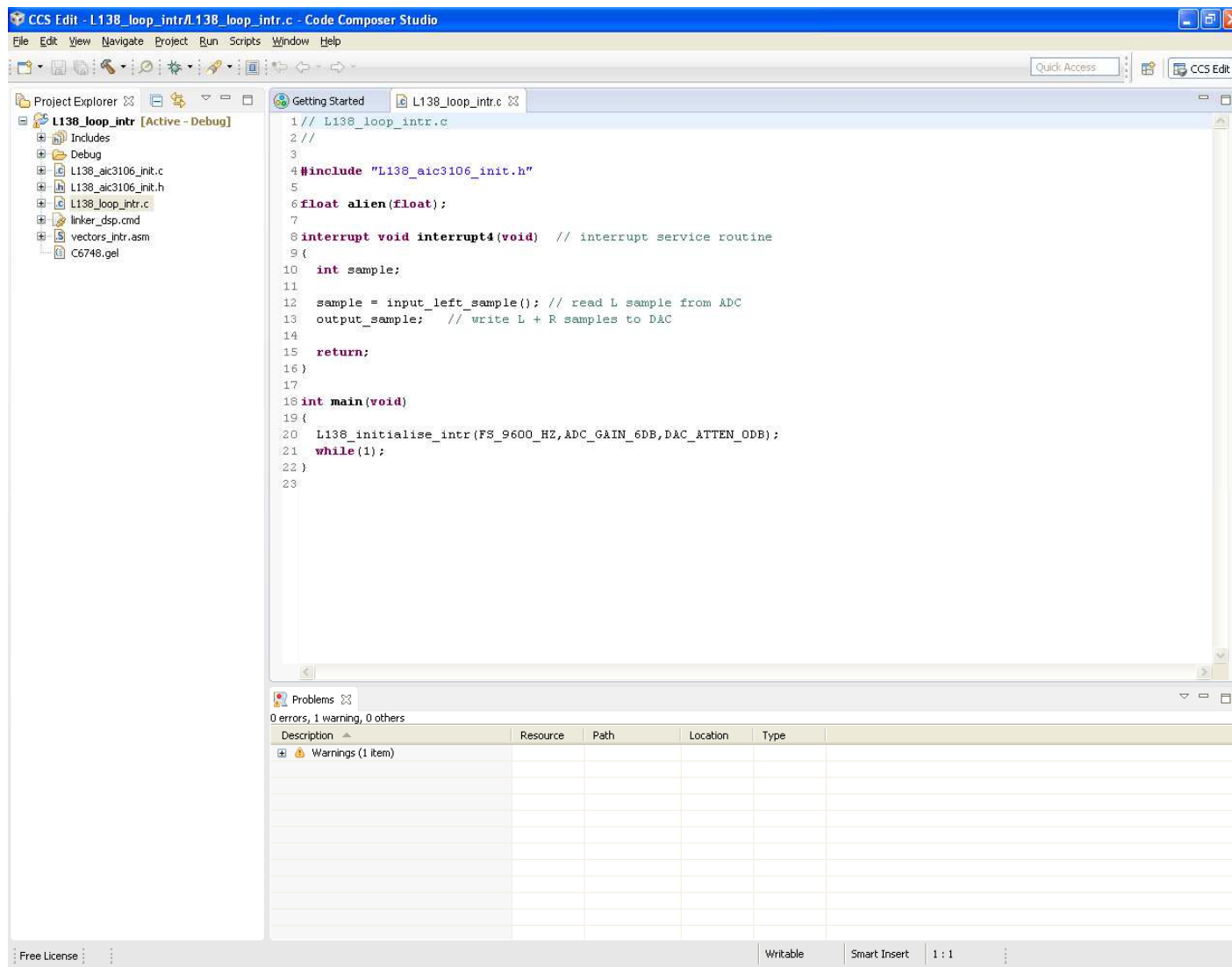
Импорт шаблонного проекта



Импорт шаблонного проекта



Анализ шаблонного проекта



Анализ шаблонного проекта

```
// L138_loop_intr.c
//

#include "L138_aic3106_init.h"

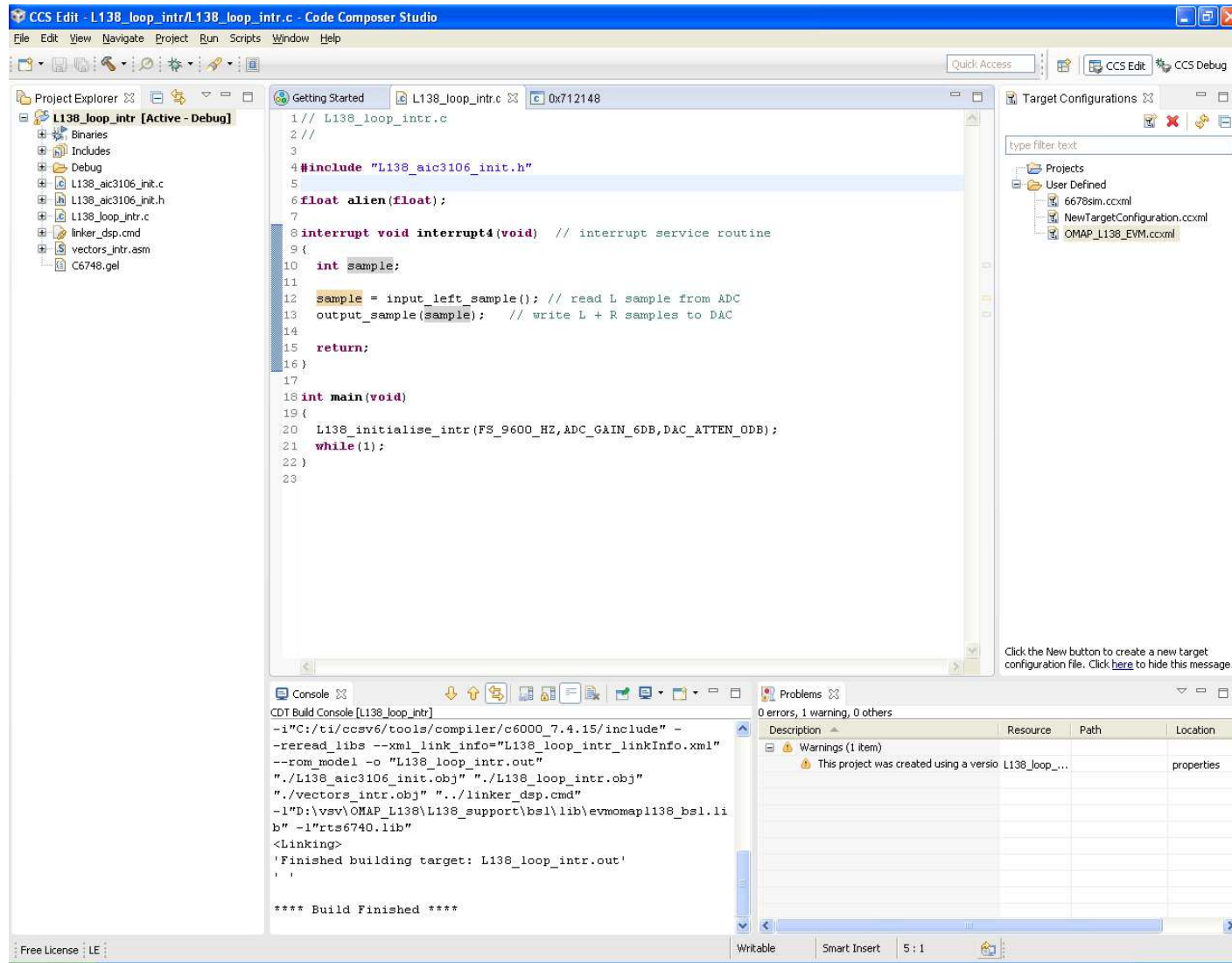
interrupt void interrupt4(void) // interrupt service routine
{
    int sample;

    sample = input_left_sample(); // read L sample from ADC
    output_sample(sample); // write L + R samples to DAC

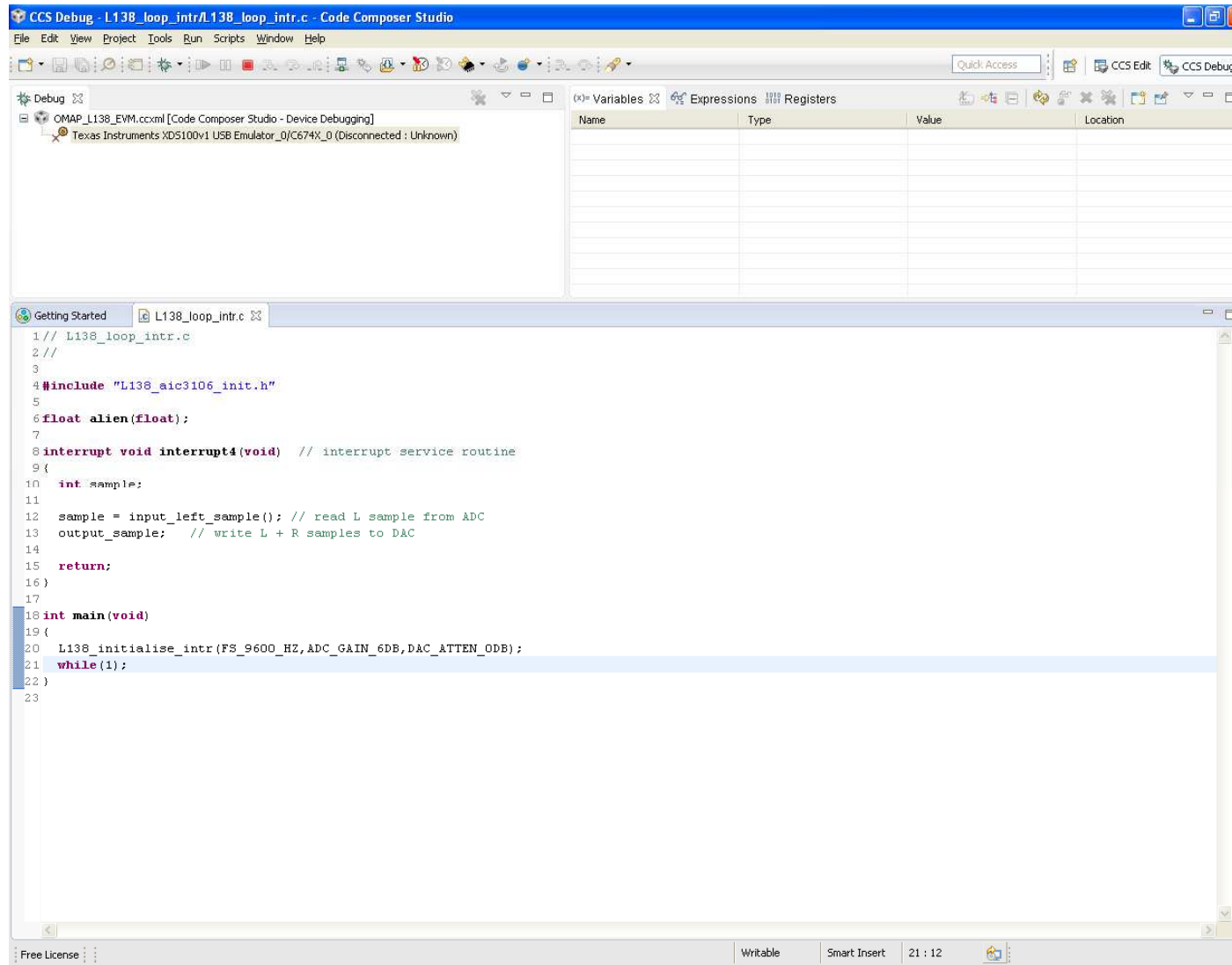
    return;
}

int main(void)
{
    L138_initialise_intr(FS_9600_HZ, ADC_GAIN_6DB, DAC_ATTEN_0DB);
    while(1);
}
```

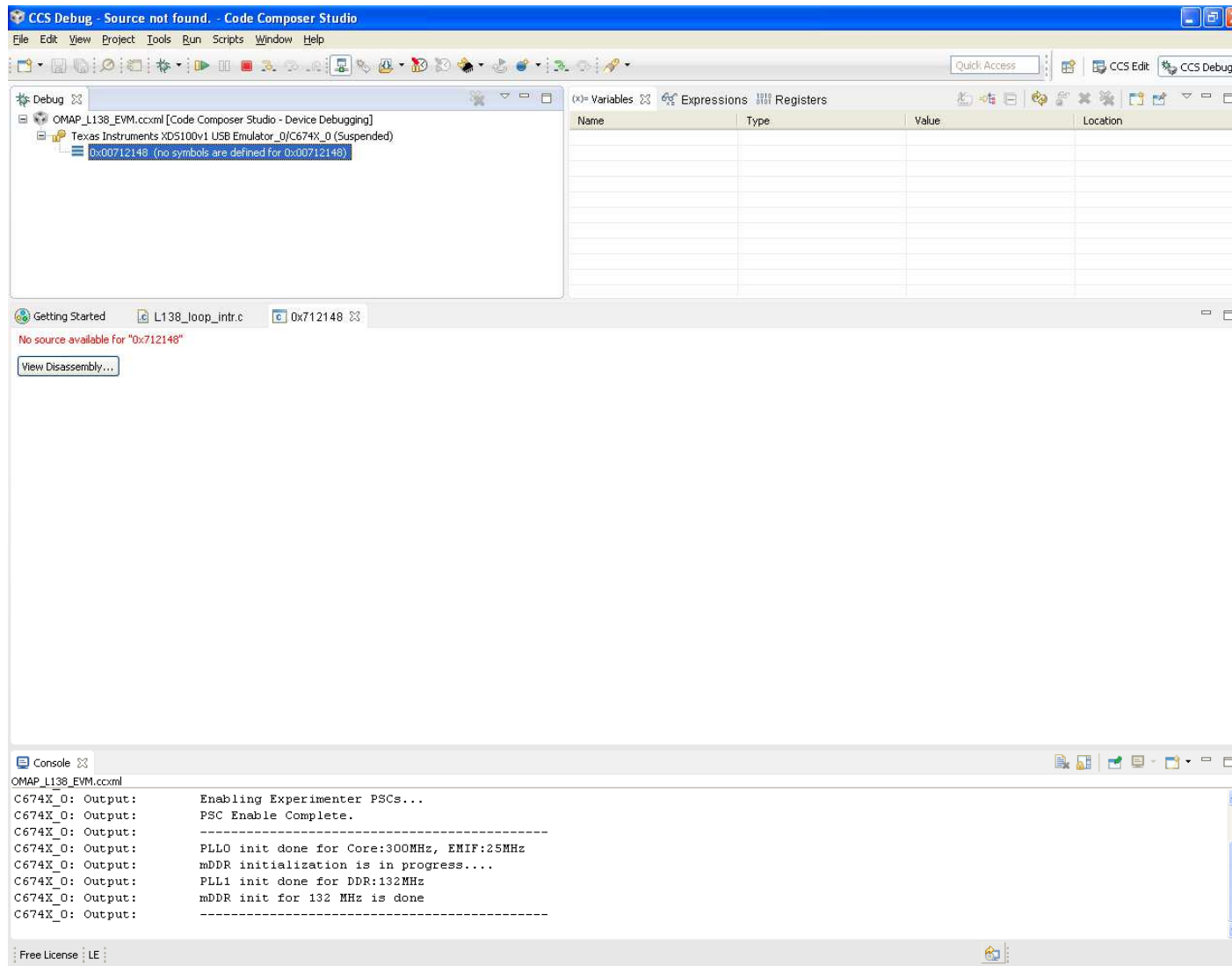

Сборка проекта, открытие файла конфигурации целевого оборудования



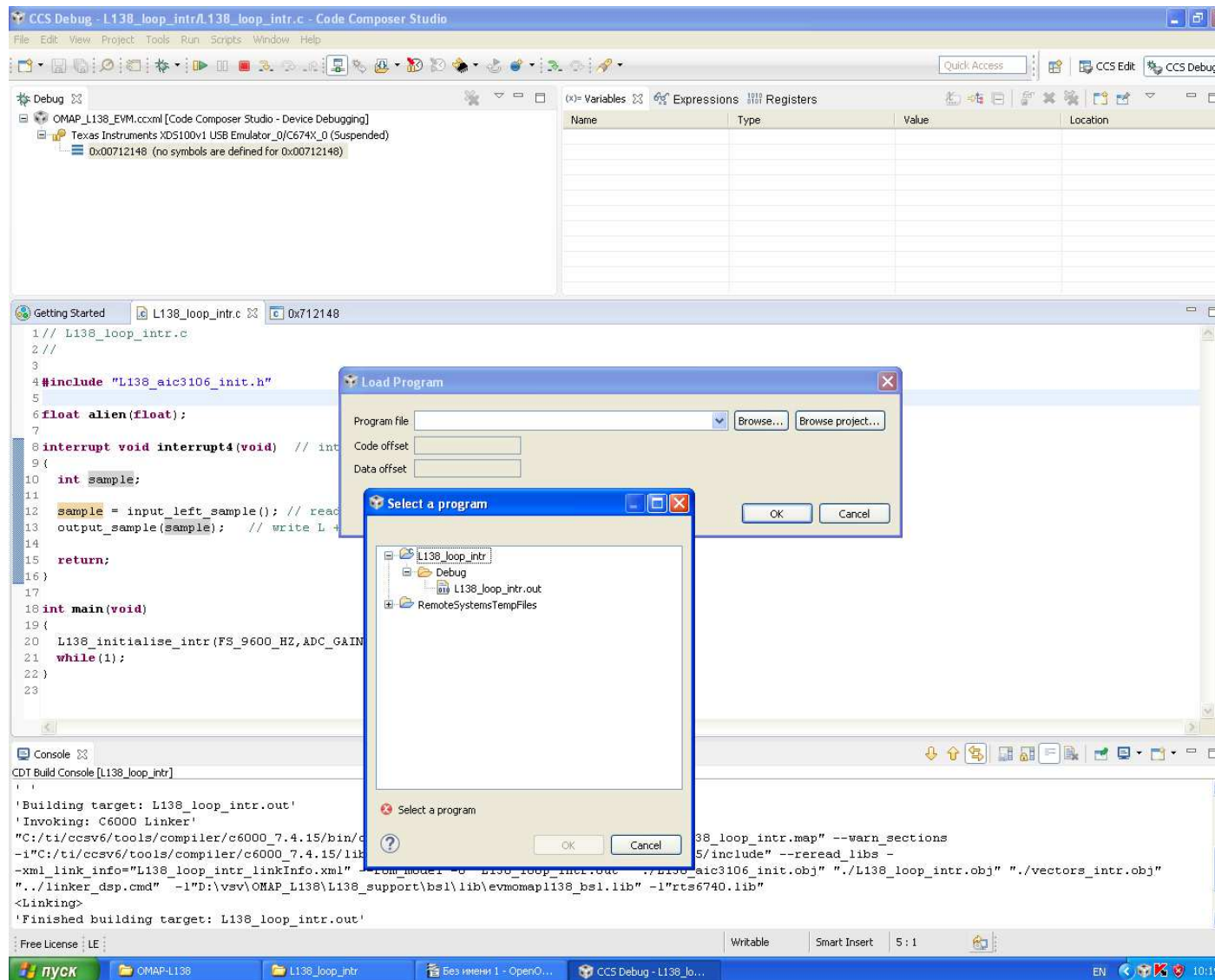
Перспектива отладки



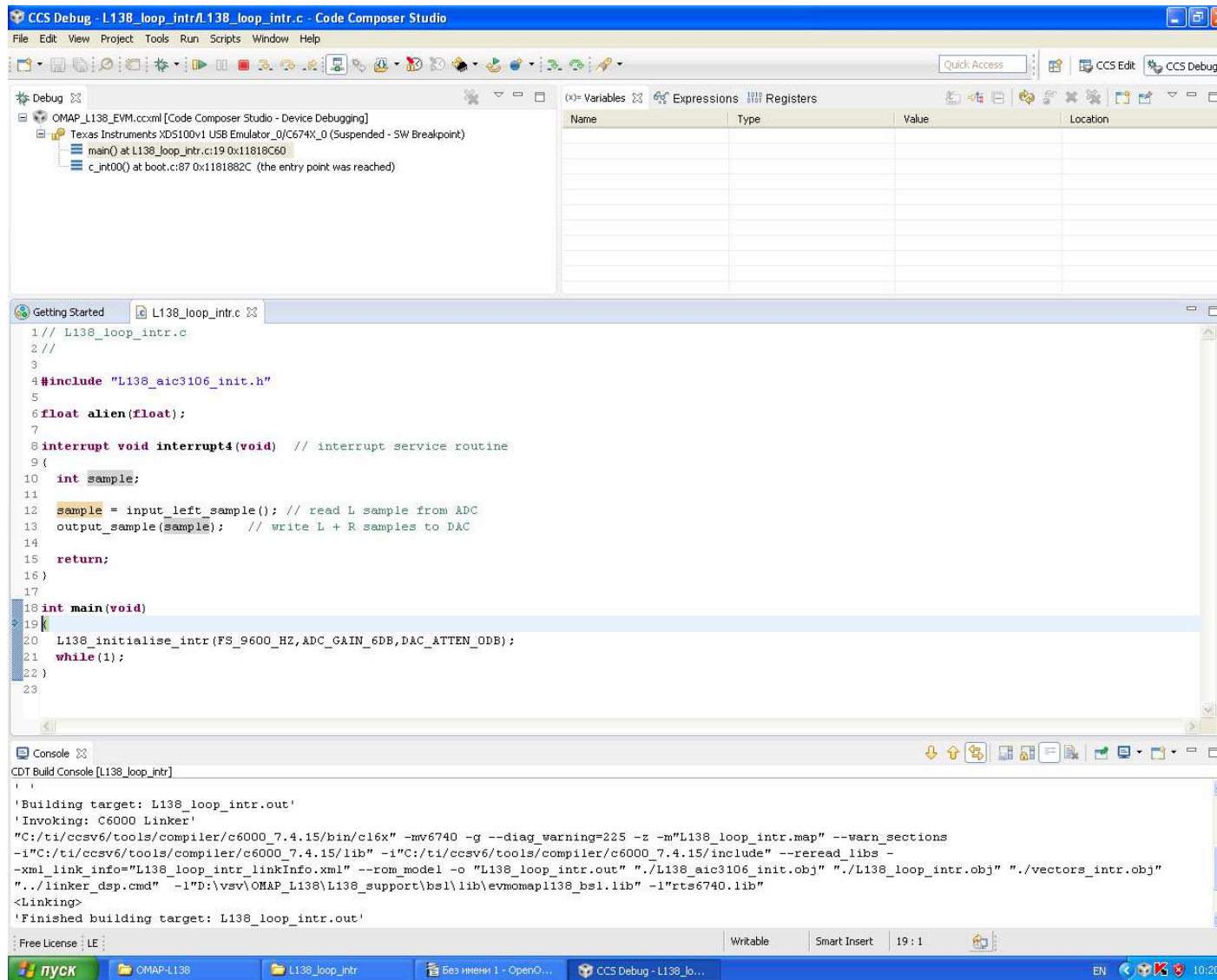
Подключение целевого оборудования



Загрузка программы



Пуск и проверка шаблонного проекта



Модификация шаблонного проекта

```
#include "L138_aic3106_init.h"

float my_function(float);

interrupt void interrupt4(void) // interrupt service routine
{
    int sample;
    float sample_float;

    sample = input_left_sample(); // read L sample from ADC
    sample_float = (float)sample;
    sample_float = my_function(sample_float);
    sample = (short)sample_float;
    output_sample(sample); // write L + R samples to DAC

    return;
}

int main(void)
{
    L138_initialise_intr(FS_9600_HZ, ADC_GAIN_6DB, DAC_ATTEN_0DB);
    while(1);
}
```

Цифровой процессор (r, s, P – в процессоре)

Цифровой сигнальный процессор

Алгоритмы обработки сигналов:

1. Сжатие/восстановление речи.
2. Криптография.
3. Синтез и распознавание речи.
4. Модемы.
5. Подавление шума.
6. Сжатие/восстановление изображений.
7. Эхокомпенсация.
8. Анализ спектра.

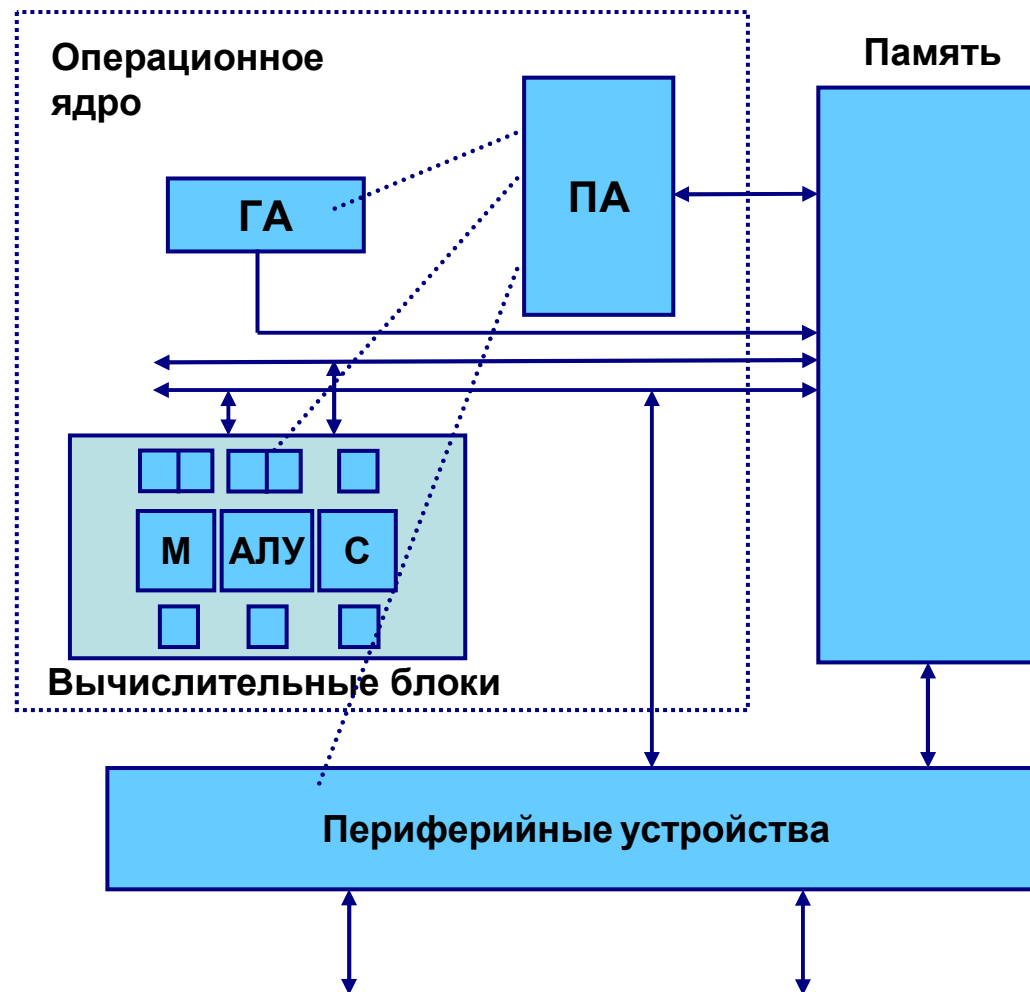


1. КИХ-фильтрация.
2. БИХ-фильтрация.
3. БПФ.
4. Адаптивная фильтрация.



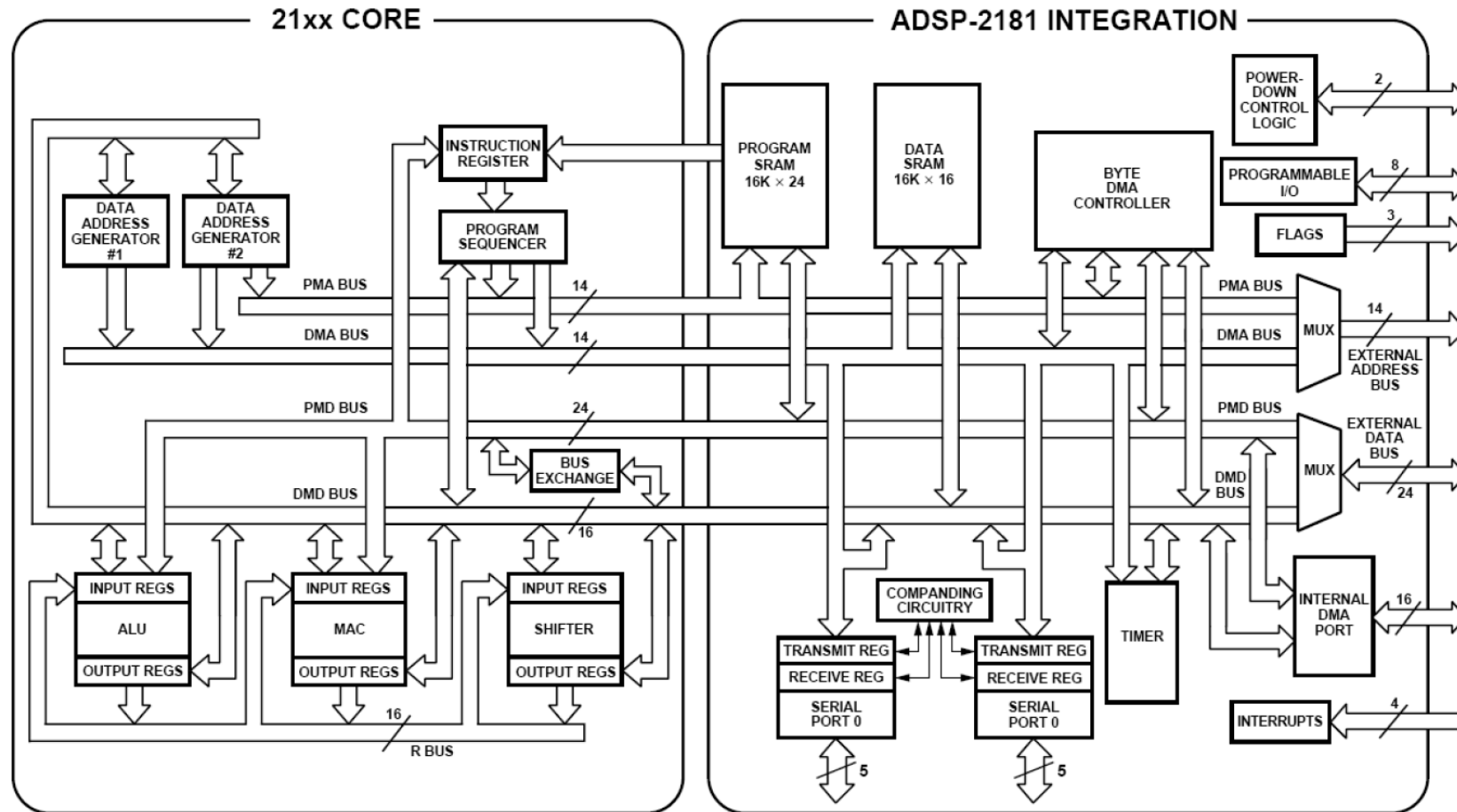
$$A = A + B \times C$$

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

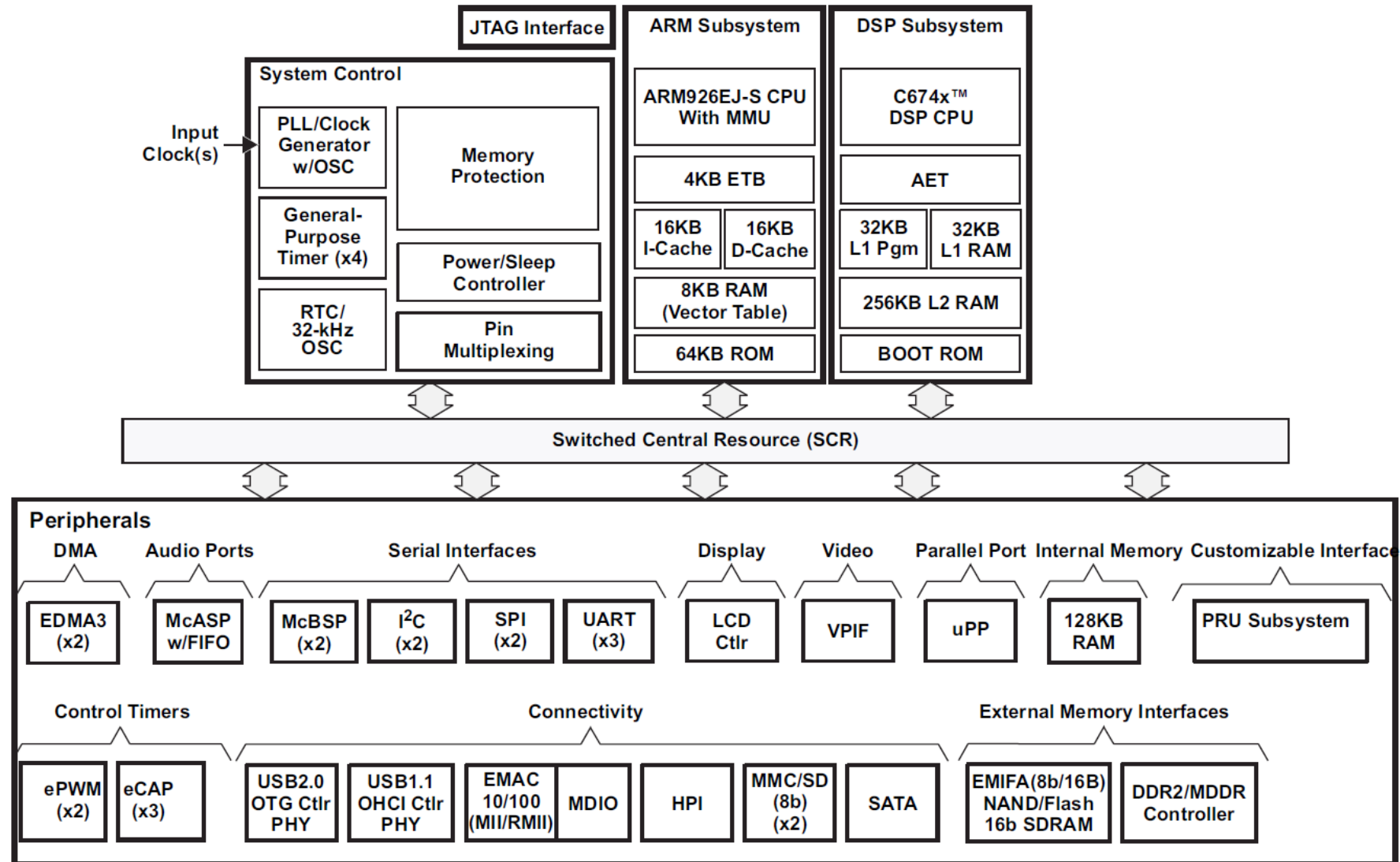


Классическая архитектура ЦСП

Пример архитектуры классического процессора (ADSP-2181)



Архитектура процессора OMAP-L138



OMAP-L138 (ARM9 + C674x DSP)

■ CPU Cores

- ARM926EJ-S™ (MPU) 450/375MHz
- C674x DSP Core 450/375MHz
- 2 PRU Cores upto 150 MHz each

■ Peripherals (1.8/3.3V I/Os)

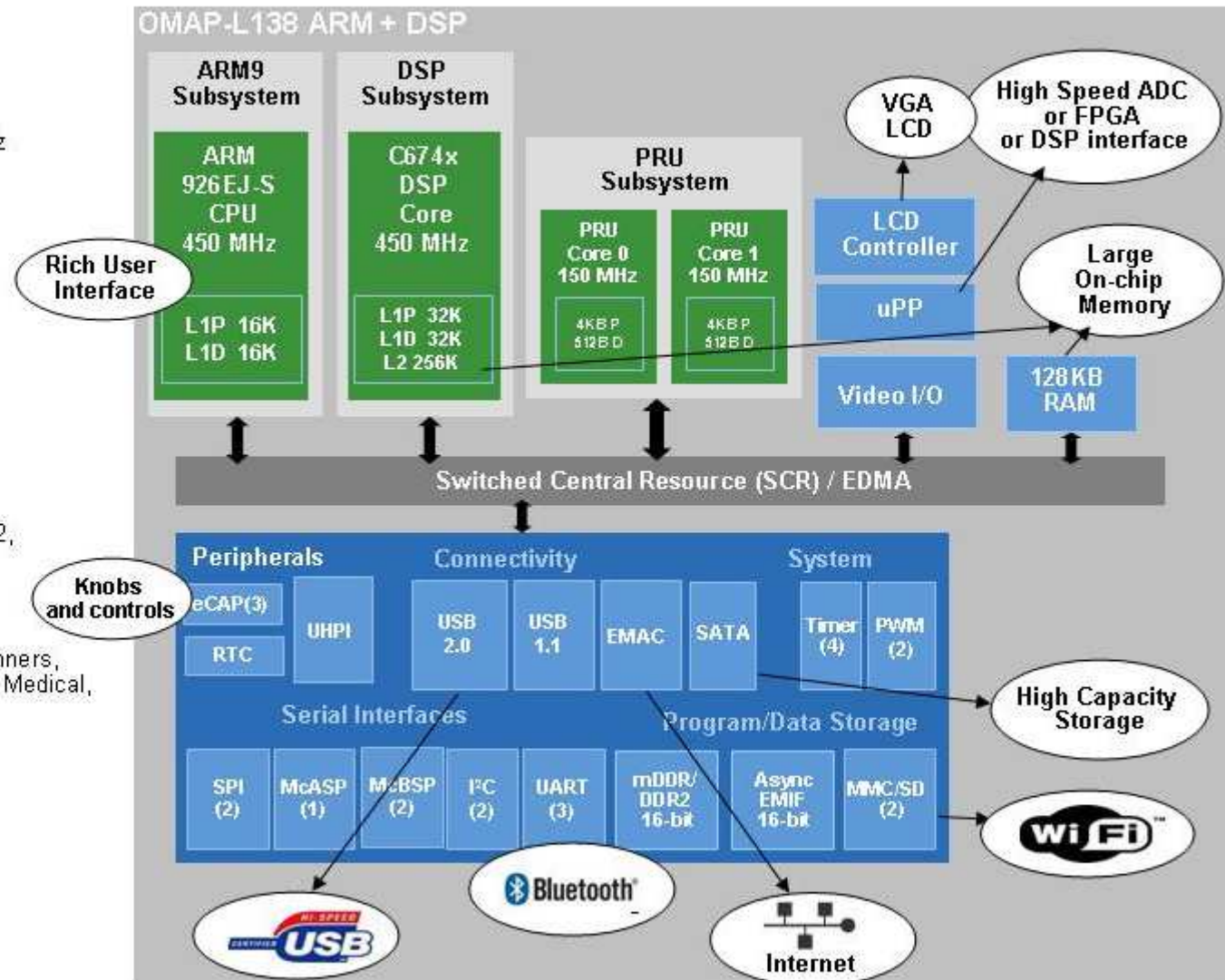
- 10/100 Ethernet MAC
- EMIFA - SDRAM/NAND Flash
- EMIFB - DDR (mDDR/DDR2)
- Video Port I/F, SATA, uPP, LCDC

■ Package

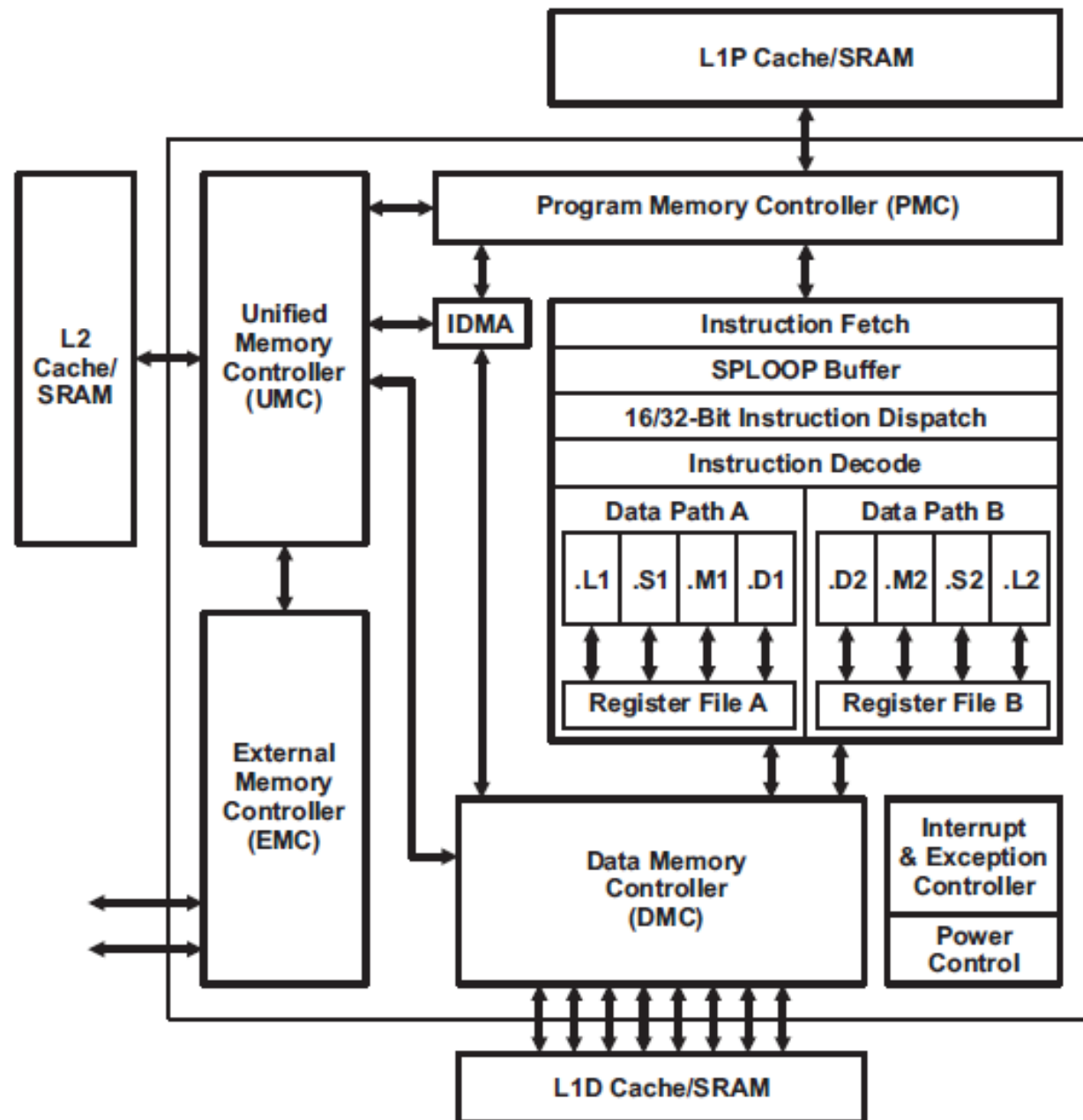
- 13 x13mm nFBGA (0.65mm), 16x16mm BGA (0.8mm)
- Pin to pin compatible with C6748/6/2, AM1808/6

■ Applications

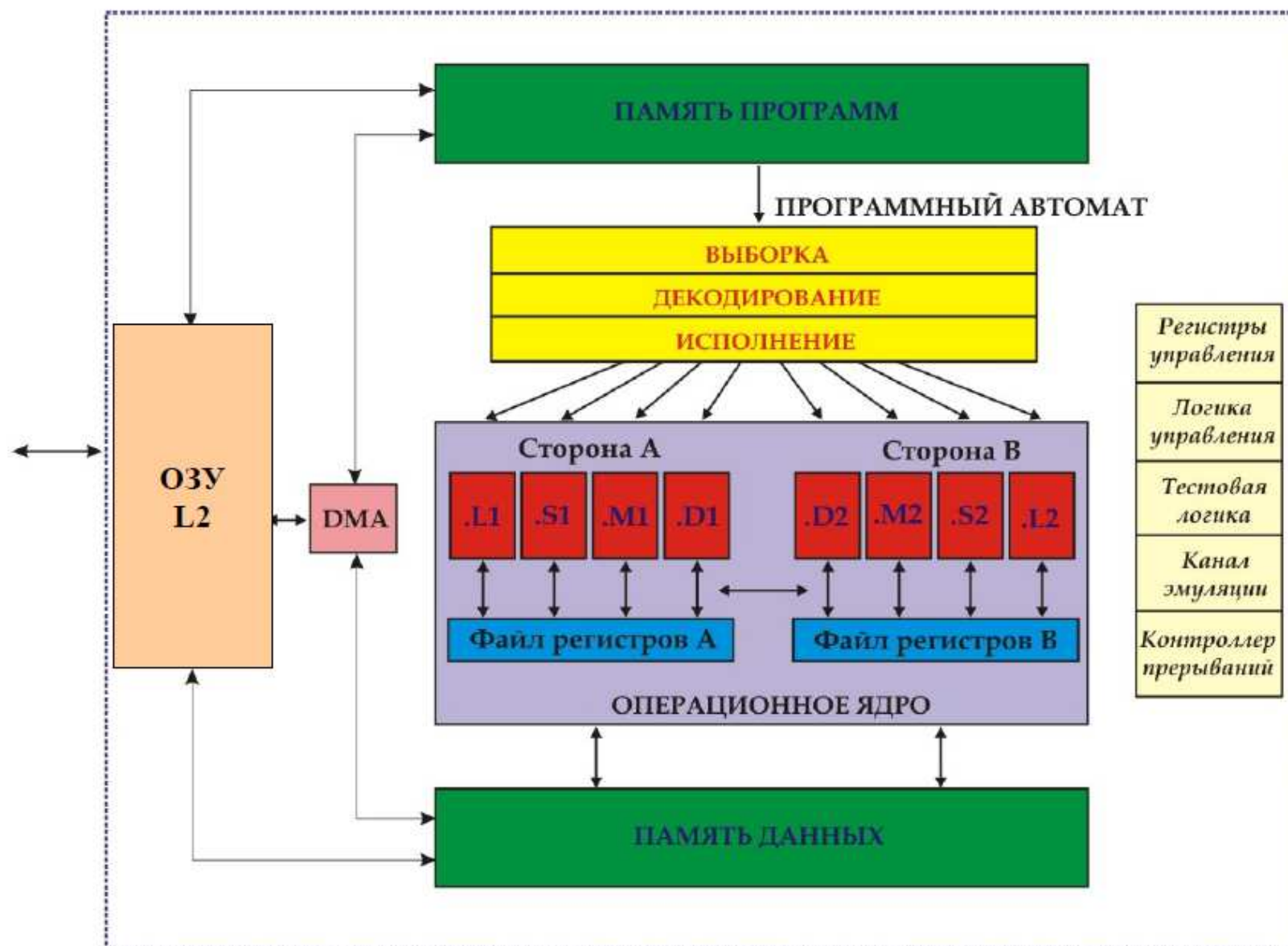
- Power Protection Systems, Test & Measurement, SDR, Bar Code Scanners, Portable Communications, Portable Medical, Portable Audio



Подсистема DSP



Подсистема DSP



Знакомство с ассемблером: основные команды

LDW .D1 *A3, A7

STW .D2 B1, *+B4(1)

ADDSP .L1 A3, A0, A2

MPYSP .M2 B1, B2, B3

NOP 4

ADD .L2 B0, B1, B2

SUB .L1 1, A0, A0

B .S1 _LOOP

Компиляция, ассемблер, дизассемблер

Обзор архитектуры процессора: память, регистры, вычислительные блоки

Поиск фрагментов кода, в которых происходит формирование эха и сдвиг линии задержки

Подробный анализ кода

Задания: найти регистр, в который записывается текущий отсчет; найти регистр, через который адресуется линия задержки; открыть окно отображения памяти и определить, по какому адресу размещается линия задержки и так далее.

Оценка времени обработки и затрат памяти

Вывод о работе в реальном времени

Понятие оптимизации; ее необходимость в нашем случае

Оценка затрат времени на разные участки кода

Выявление причины большого времени обработки

Поиск путей решения проблемы

Модификация программы с применением циклической буферизации

Модификация проекта

Отладка

Оценка новых временных затрат

Выводы