

# Интегрированная среда программирования Code Composer Studio

Управление проектом:

- Исходные и объектные файлы
- Файл зависимостей

Полная C/C++ и ассемблер отладка:

- Смешанный режим
- Дизассемблер
- Установка точек останова и пробных точек

Редактор:

Окно  
статуса

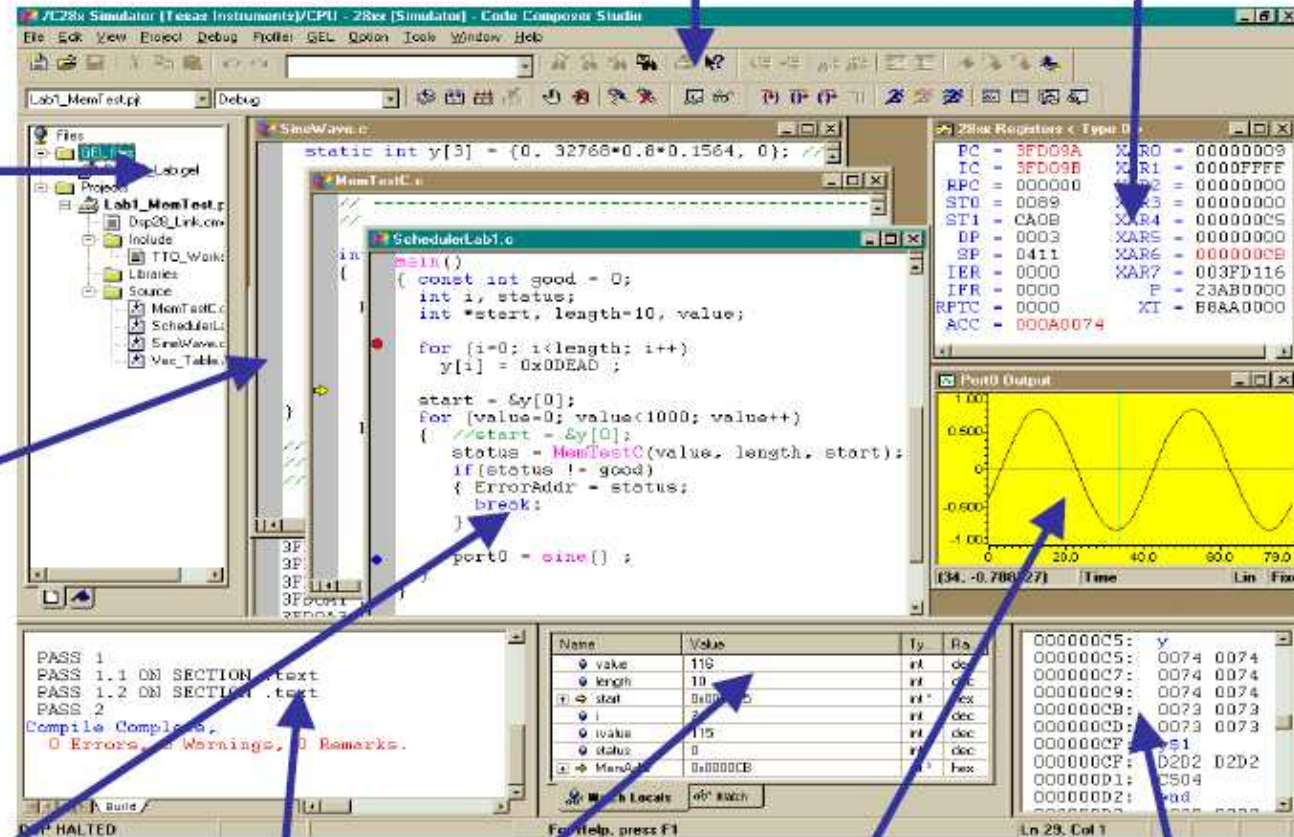
Окно  
просмотра

Графическое  
окно

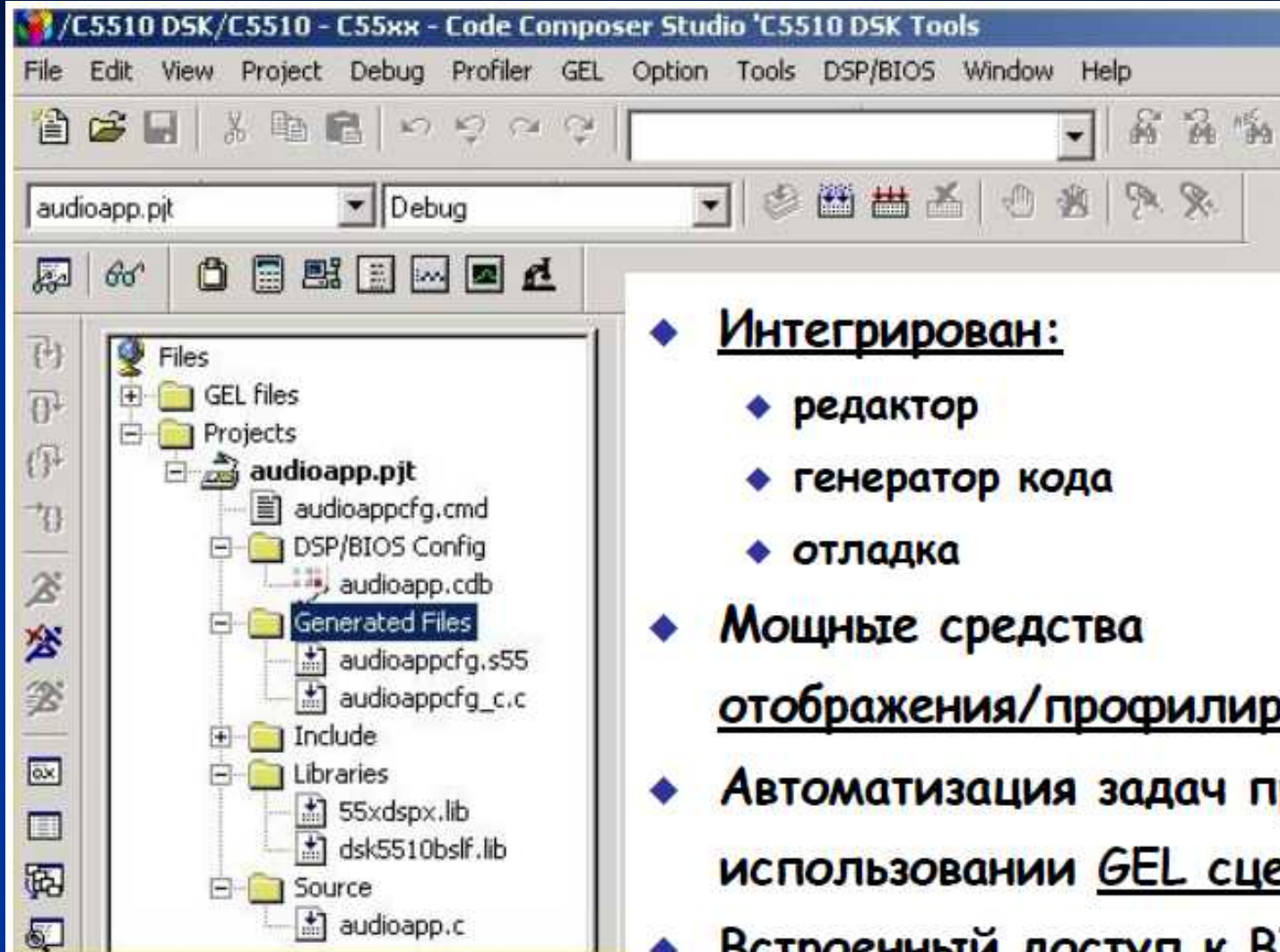
Дамп  
памяти

Меню и иконки

Окно  
регистров ЦП



# Интегрированная среда программирования *Code Composer Studio*



- ◆ Интегрирован:
  - ◆ редактор
  - ◆ генератор кода
  - ◆ отладка
- ◆ Мощные средства отображения/профилирования
- ◆ Автоматизация задач при использовании GEL сценариев
- ◆ Встроенный доступ к BIOS функциям

# *Интегрированная среда программирования Code Composer Studio*

---

- **Отладка**
  - *Загрузка исполняемого кода*
  - *Запуск в реальном времени или шаг за шагом*
  - *Точки останова*
  - *Профилирование*
  - *Сохранение дампа памяти в файл*
  - *Формирование сигнала по содержимому памяти*

# *Интегрированная среда программирования Code Composer Studio*

---

- *Для построения проекта необходимы:*
- *Исходные файлы (C,C++ и/или ассемблер) (\*.c, \*.asm)*
  - *Текстовые файлы с проверкой синтаксиса, в основном для ассемблера \*.asm*
- *Командный файл компоновки(\*.cmd)*
- *Файл проекта(\*.pjt)*
  - *Содержит исполняемый файл для CCS*
  - *Определяет все исходные файлы и командный файл компоновки, необходимые для компиляции исполняемого файла*



# Интегрированная среда программирования Code Composer Studio

- **Проект в CCS. Файл проекта \*.pjt содержит:**



## Исходные файлы:

- ◆ Исходные файлы (C, ASM)
- ◆ Библиотеки
- ◆ DSP/BIOS конфигурация
- ◆ Компоновщик и др.

## Настройки проекта:

- ◆ Опции компиляции (компилятор и asm)
- ◆ Конфигурации компилятора
- ◆ DSP/BIOS
- ◆ Компоновщик

# Интегрированная среда программирования Code Composer Studio

## ■ Применяемые инструкции и помощь online:

The screenshot displays the Code Composer Studio (CCS) environment. The main window is titled "/C54x Simulator (Texas Instruments)/CPU - C54X (Simulator) - Code Composer Studio". The menu bar includes File, Edit, View, Project, Debug, Profiler, GEL, Option, Tools, DSP/BIOS, Window, and Help. The Help menu is open, showing options: Contents, User Manuals, Tutorial, CCS on the Web, and About... The left sidebar shows the "TMS320C54x Code Composer Studio" project structure, including "Welcome to CCS IDE", "CCS Documentation", "What's New and Migration", "Using CCS IDE", "Code Composer Studio", and "TMS320C54x DSP Mnemonic Instruction Set". The "TMS320C54x DSP Mnemonic Instruction Set" is selected, and its contents are displayed in the main window. The text "summary below is included for convenience" is visible. The "Instruction Categories" section lists: [Arithmetic Instructions](#), [Control Instructions](#), [I/O Instructions](#), and [Load/Store Instructions](#).

■ LD	Smem,dst
■ ADD	Smem,dst
■ STL	src,Smem

# Интегрированная среда программирования Code Composer Studio

## ■ Командный файл компоновки:

```
MEMORY
{
    PAGE 0: VECS:    origin = 0080h, length = 0080h
                        /* Internal Program RAM */
    PRAM:    origin = 100h, length = 1f00h
                        /* Internal Program RAM */
    PAGE 1: SCRATCH:origin = 0060h, length = 0020h
                        /* Scratch Pad Data RAM */
    INRAM:    origin = 2000h, length = 1ffffh
                        /* Internal Data RAM */
}

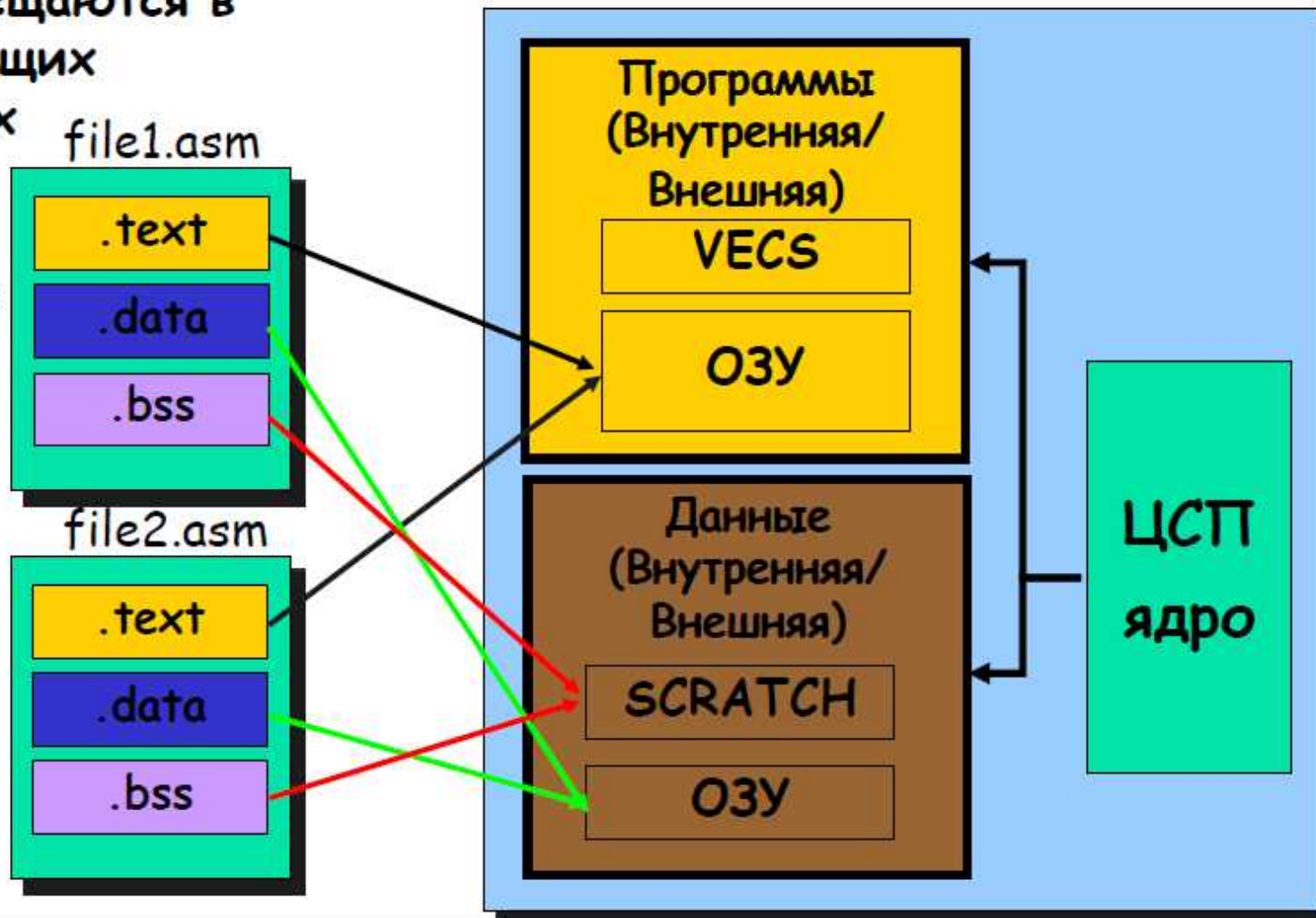
SECTIONS
{
    .text    >    PRAM PAGE 0
    .data    >    INRAM PAGE 1
    .bss     >    SCRATCH PAGE 1
}
```



# Интегрированная среда программирования Code Composer Studio

## ■ Пространства памяти и программные секции:

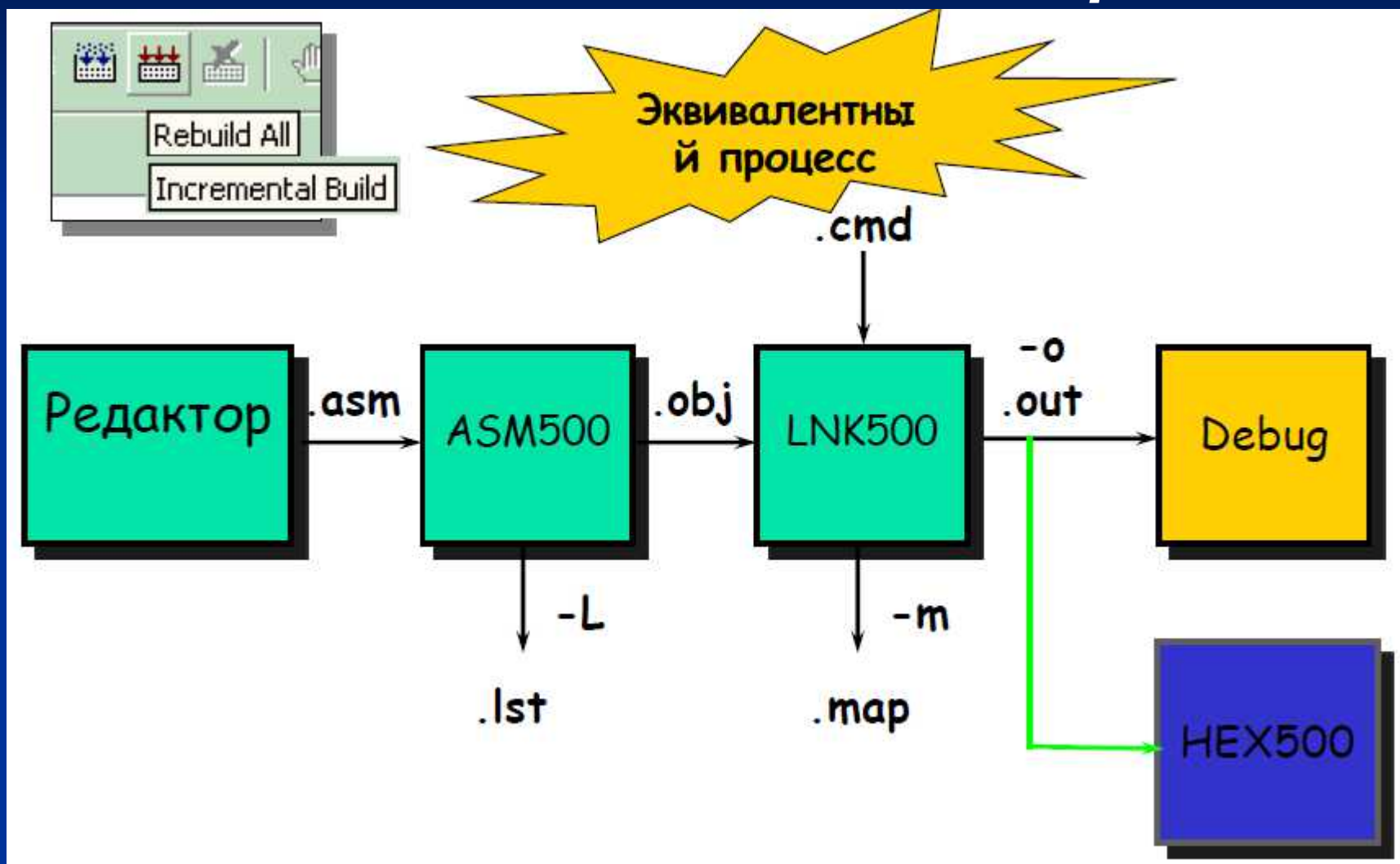
Секции размещаются в соответствующих пространствах памяти через компоновщик





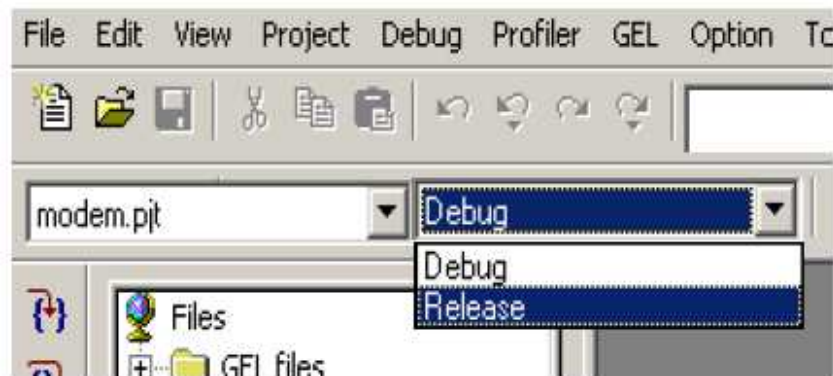
# Интегрированная среда программирования Code Composer Studio

## ■ Компиляция исполняемого файла:

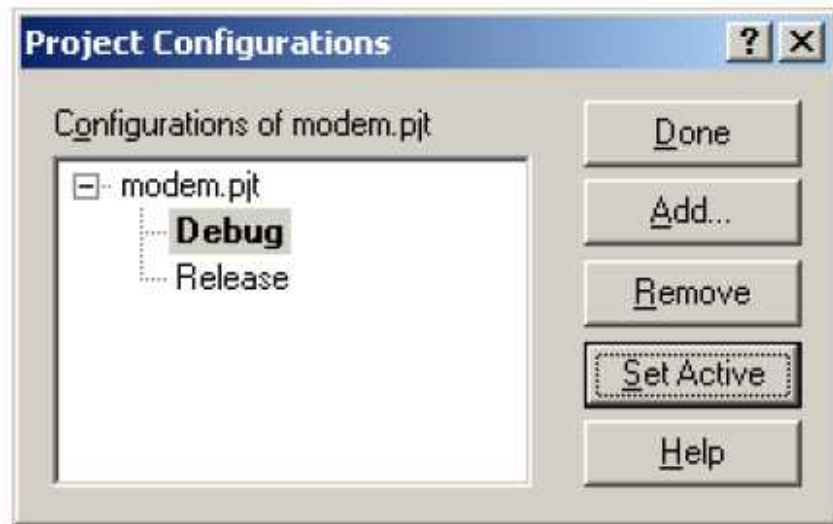


# Интегрированная среда программирования Code Composer Studio

## ■ Конфигурации компиляции по умолчанию:



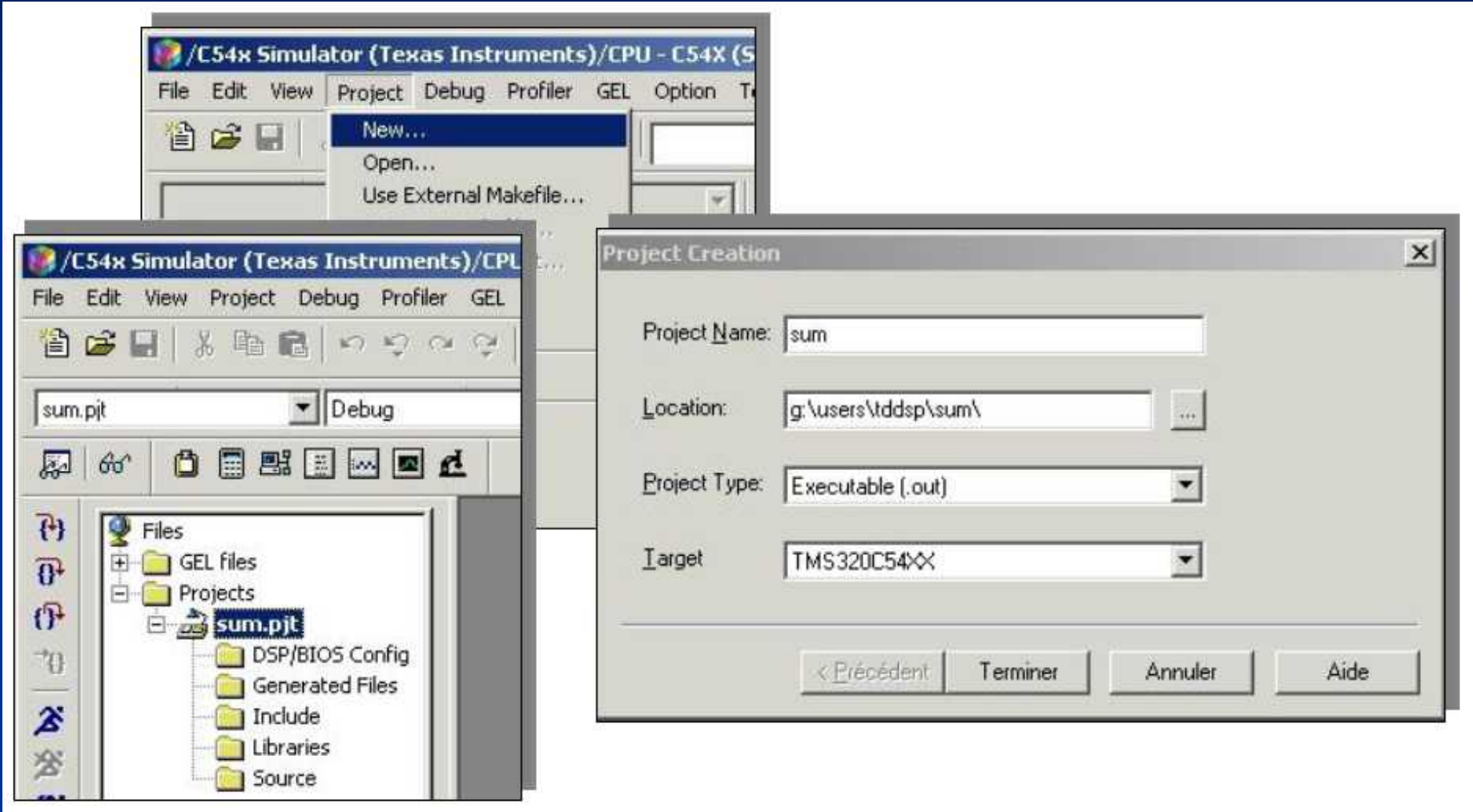
- ◆ Для нового проекта, CCS автоматически создает две конфигурации:
  - ◆ **Debug** (без оптимизации)
  - ◆ **Release** (оптимизированная)



- ◆ Добавление/удаление конфигураций компиляции через *Project Configurations*
- ◆ Редактирование конфигураций:
  1. Выбор активной
  2. Изменение опций
  3. Сохранение проекта

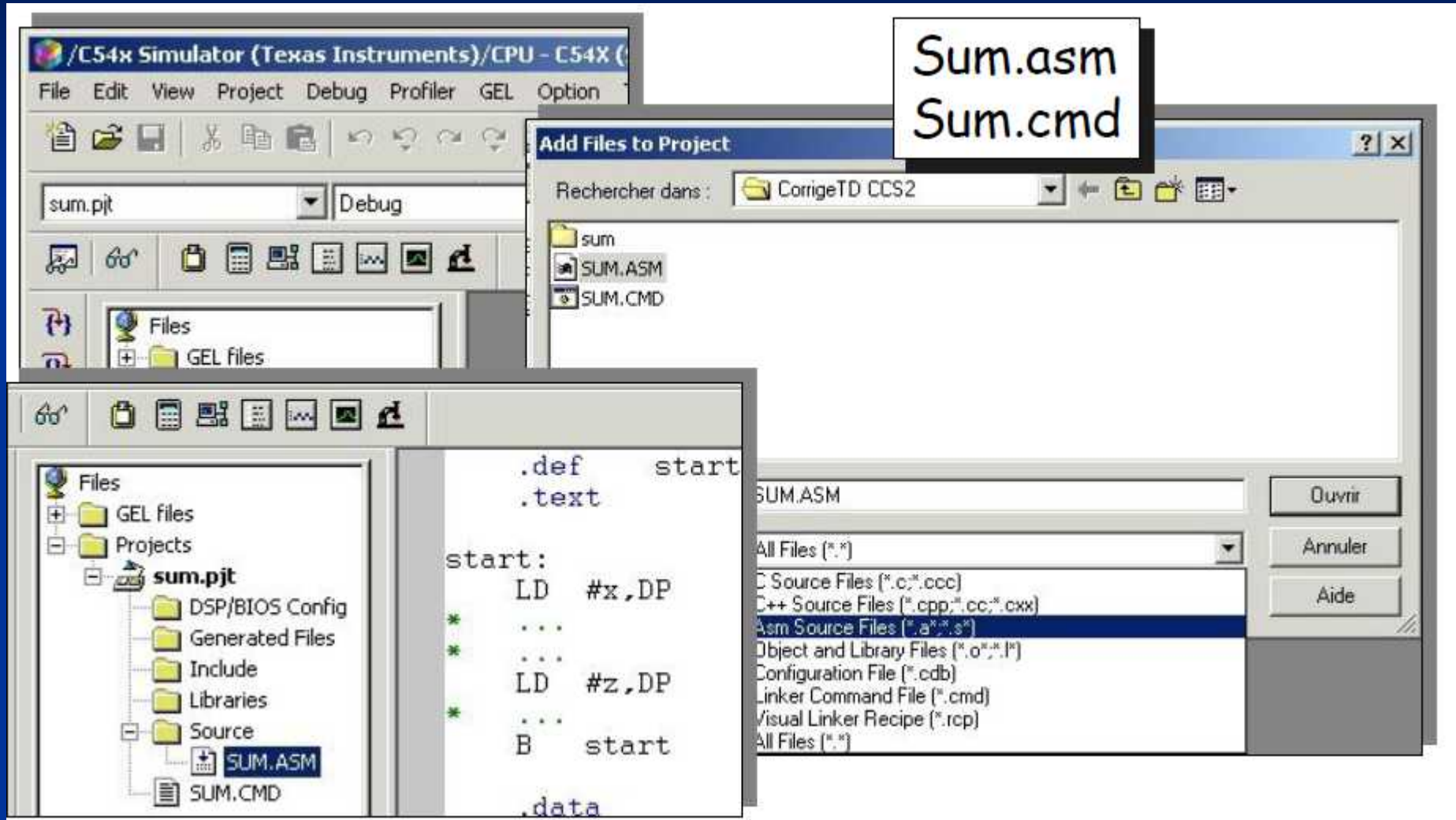
# Интегрированная среда программирования Code Composer Studio

## ■ Создание проекта:



# Интегрированная среда программирования Code Composer Studio

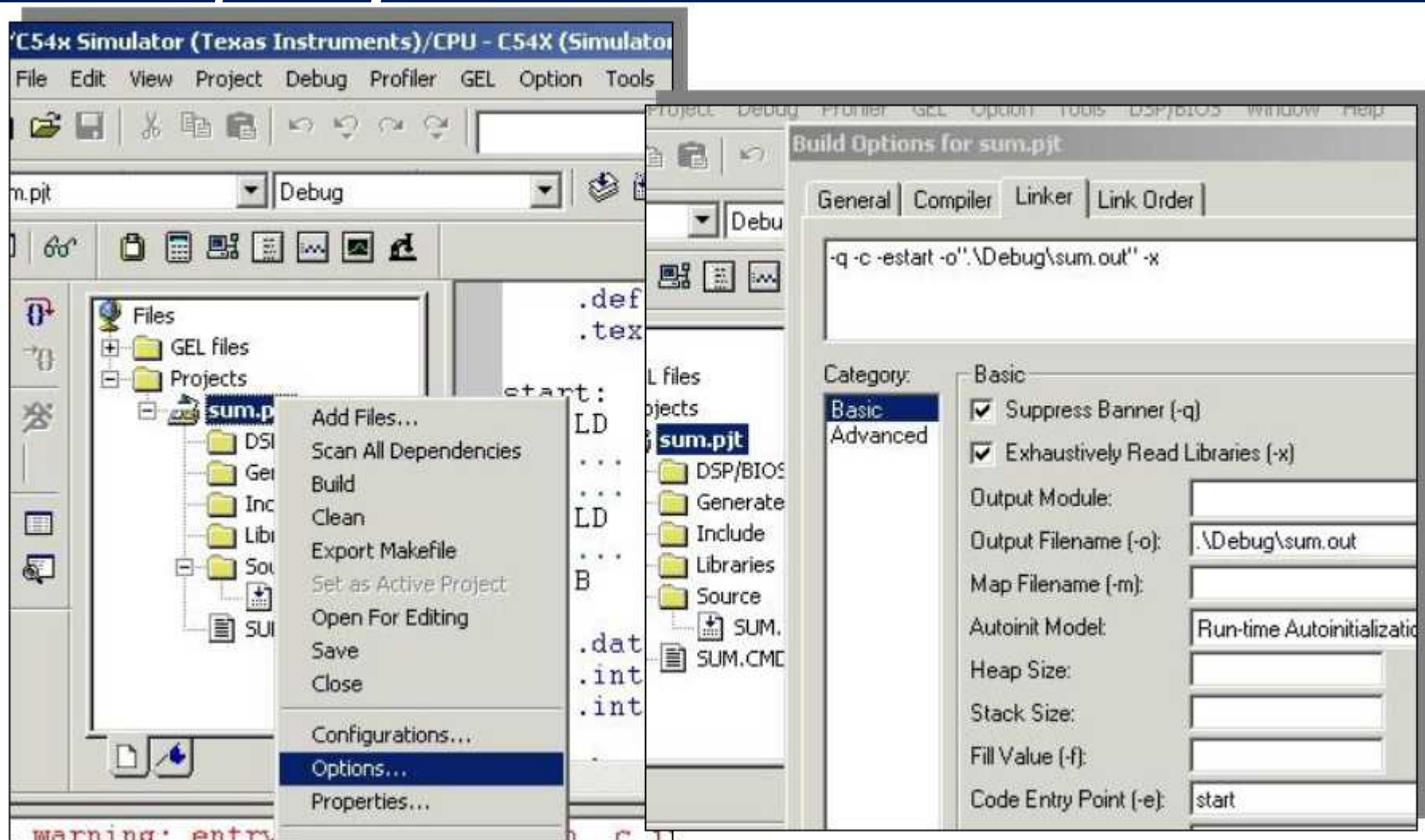
## ■ Создание проекта:





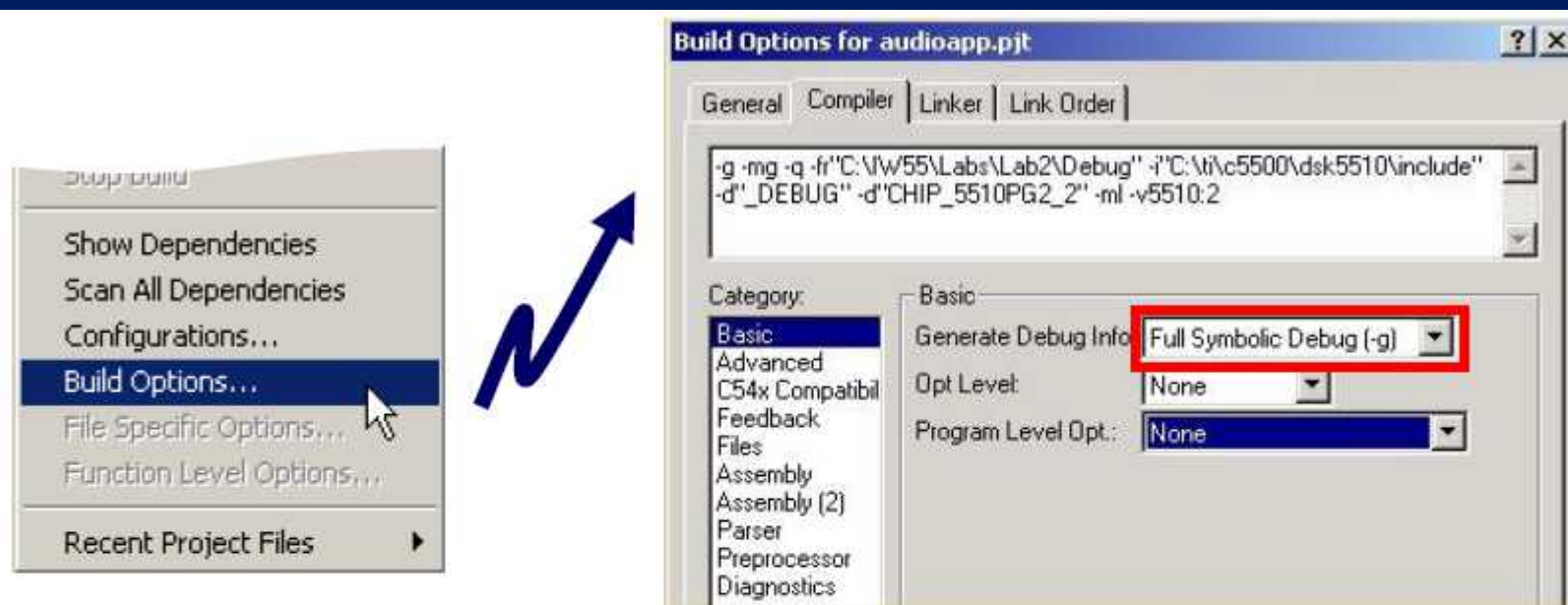
# Интегрированная среда программирования Code Composer Studio

## ■ Опции проекта:



# Интегрированная среда программирования Code Composer Studio

## ■ Опции компилятора:

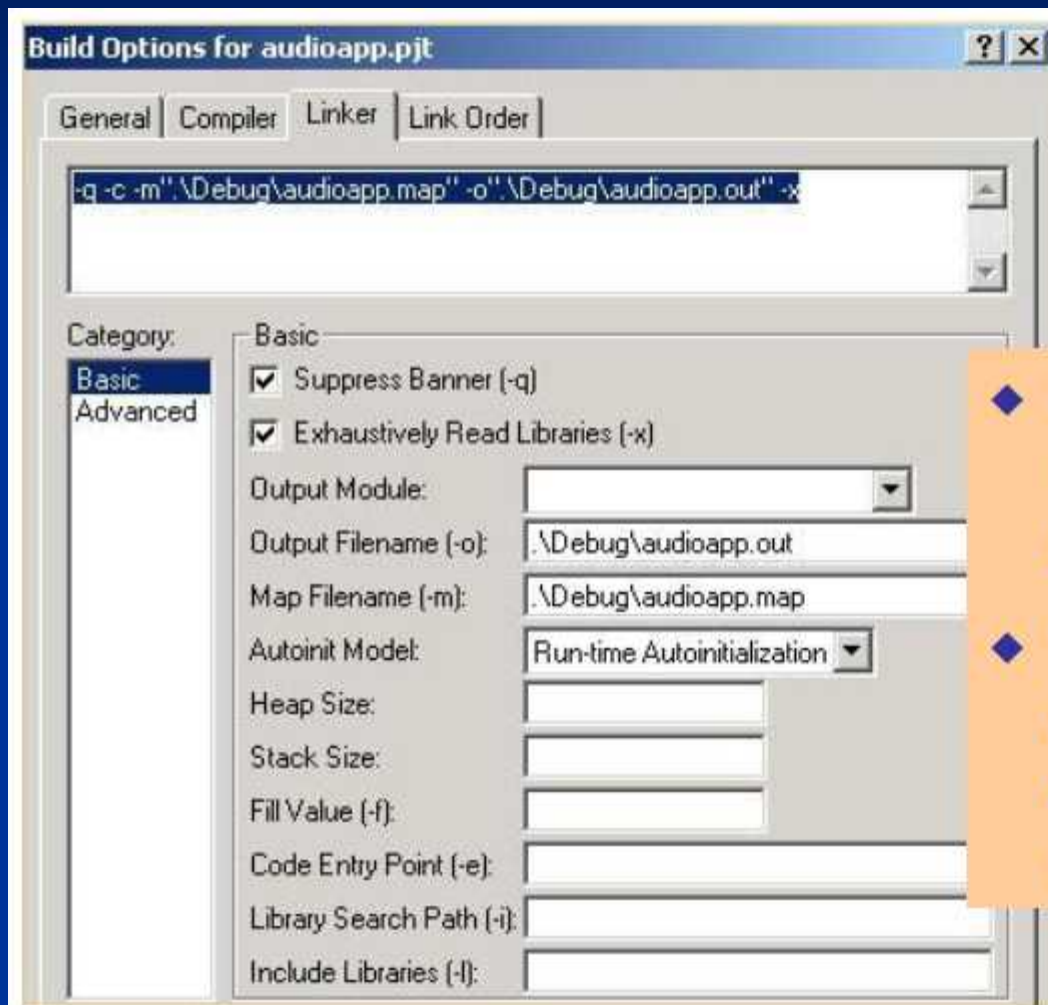


### ◆ Управление многими аспектами компиляции:

- Уровни оптимизации
- Выбранное устройство
- Опции компилятора/компоновщика
- Другие

# Интегрированная среда программирования Code Composer Studio

## ■ Опции компоновщика:

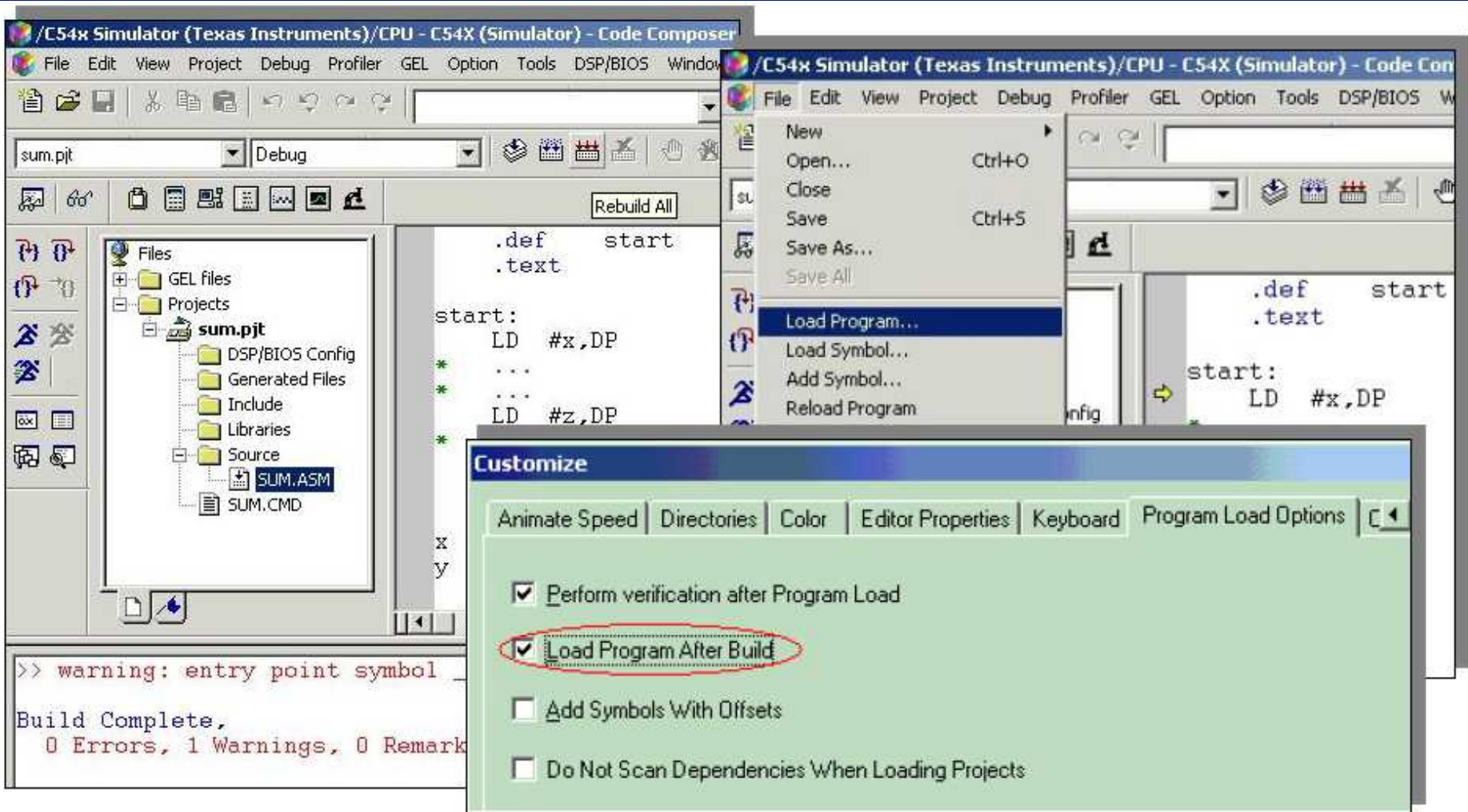


- ◆ Опции компоновщика показаны в конфигурации *Debug* для проекта *\*.pjt*
- ◆ ".\Debug\" указывает на директорию уровнем ниже места расположения проекта (.pjt)



# Интегрированная среда программирования Code Composer Studio

## ■ Построение проекта:





# Интегрированная среда программирования Code Composer Studio

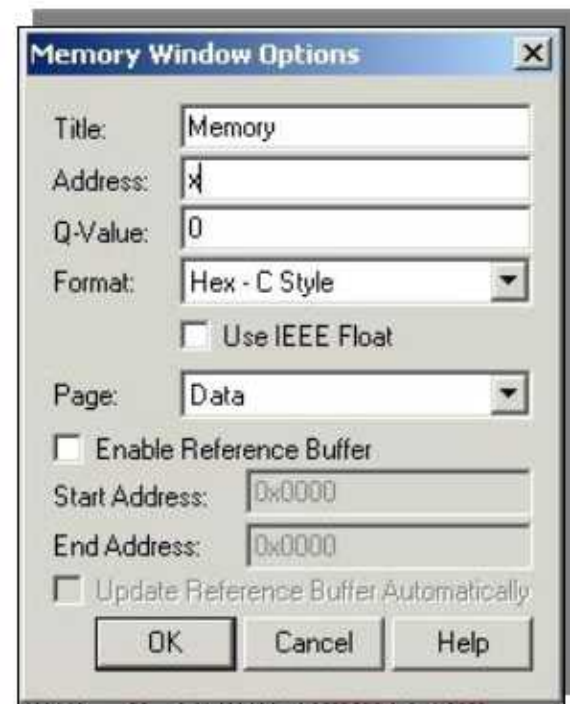
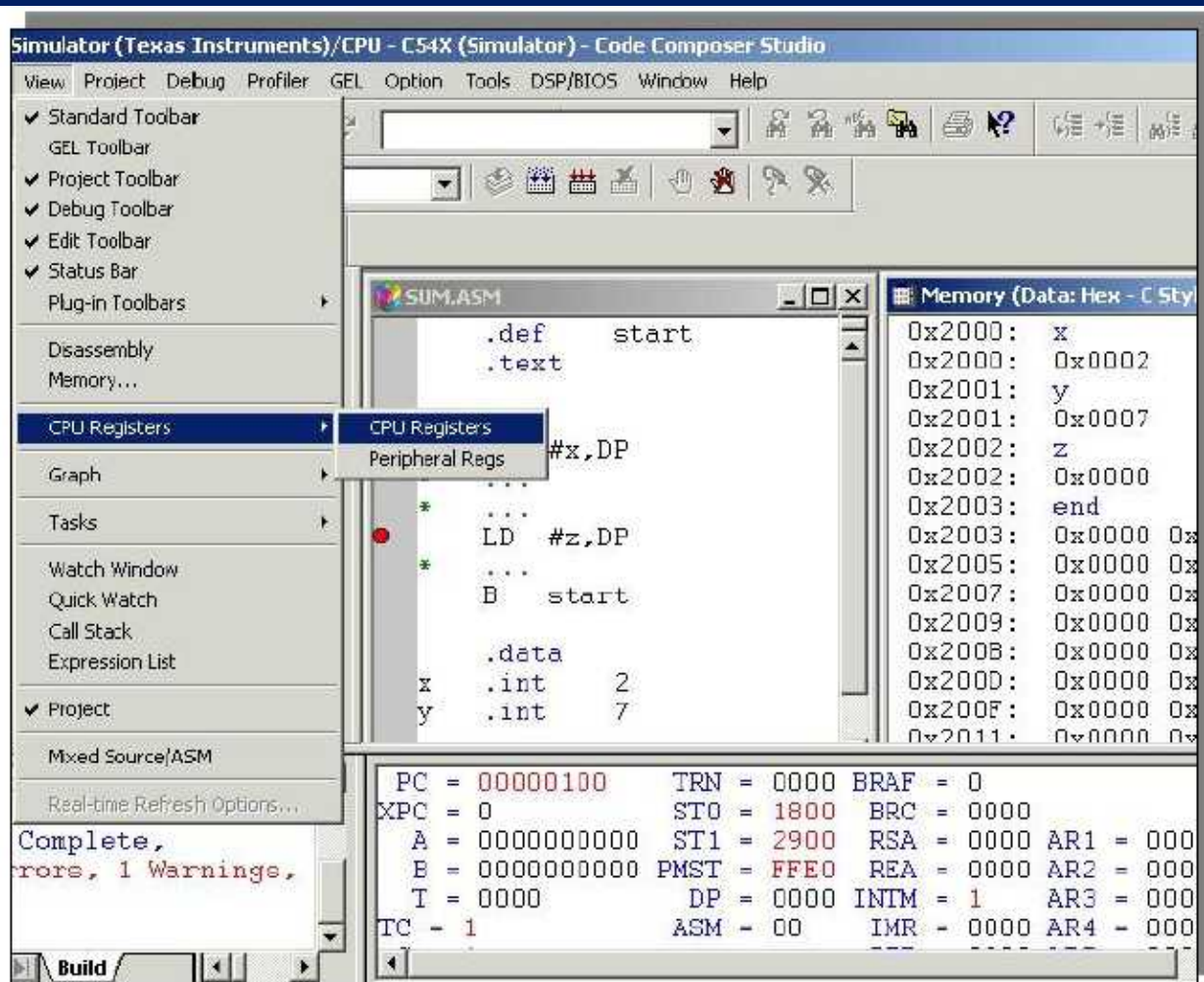
## ■ **Запуск проекта:**

The screenshot displays the Code Composer Studio interface. On the left, the 'Files' pane shows a project tree with folders like 'GEL files', 'Projects', 'DSP/BIOS Config', 'Generated Files', 'Include', 'Libraries', 'Source', and files 'SUM.ASM' and 'SUM.CMD'. The 'sum.pjt' project is selected. The main editor window shows assembly code with labels like 'start:', 'LD #2', and 'B st'. A yellow arrow points to the 'start:' label. On the right, the 'Debug' menu is open, showing options like 'Breakpoints...', 'Probe Points...', 'Step Into' (F8), 'Step Over' (F10), 'Step Out' (Shift F7), 'Run' (F5), 'Halt' (Shift F5), 'Animate' (F12), 'Run Free', 'Run to Cursor' (Ctrl F10), 'Multiple Operation...' (F11), and 'Reset CPU'. A text box on the right lists the following steps for running the project:

- По шагам
- В реальном времени
- Точки останова

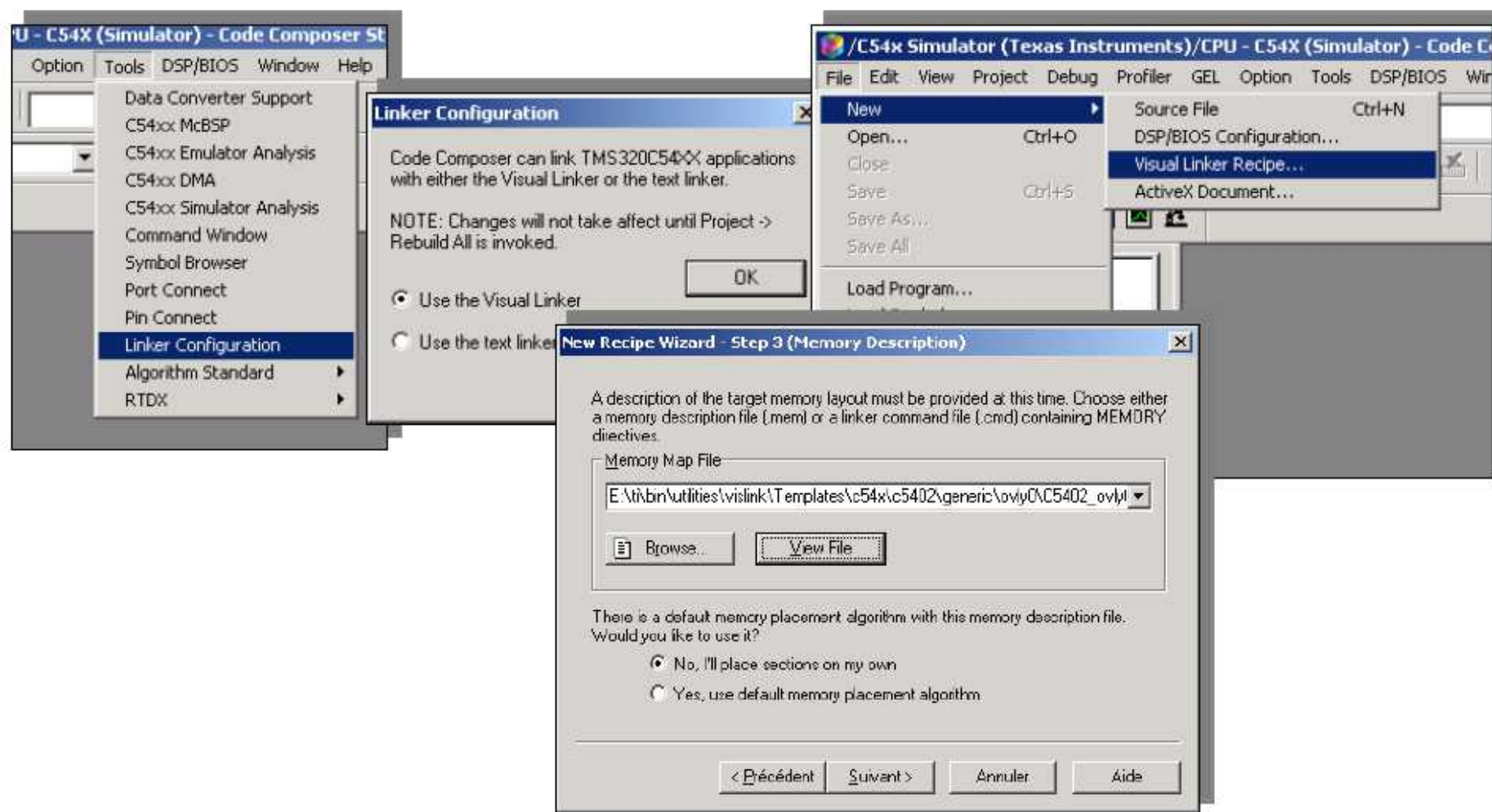
# Интегрированная среда программирования Code Composer Studio

## ■ Просмотр памяти и регистров ЦП:



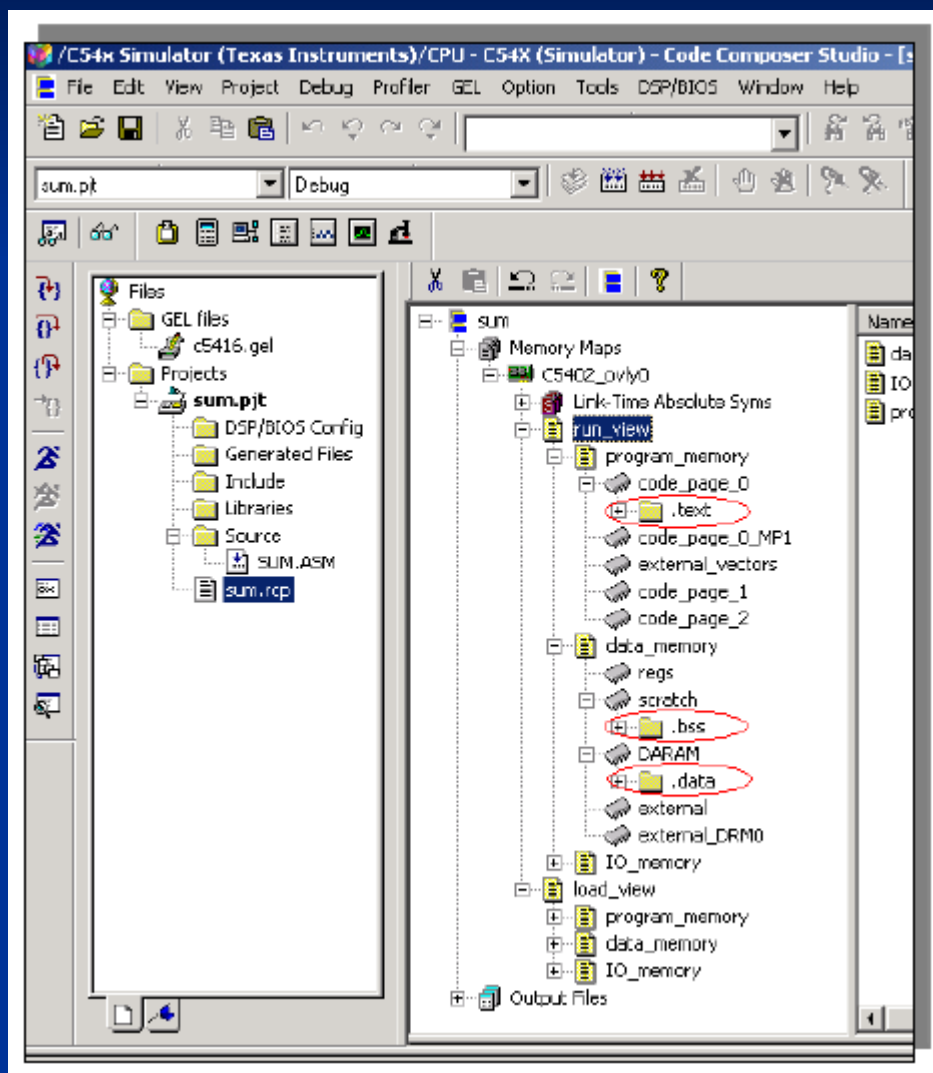
# Интегрированная среда программирования Code Composer Studio

## ■ Визуальный компоновщик:



# Интегрированная среда программирования Code Composer Studio

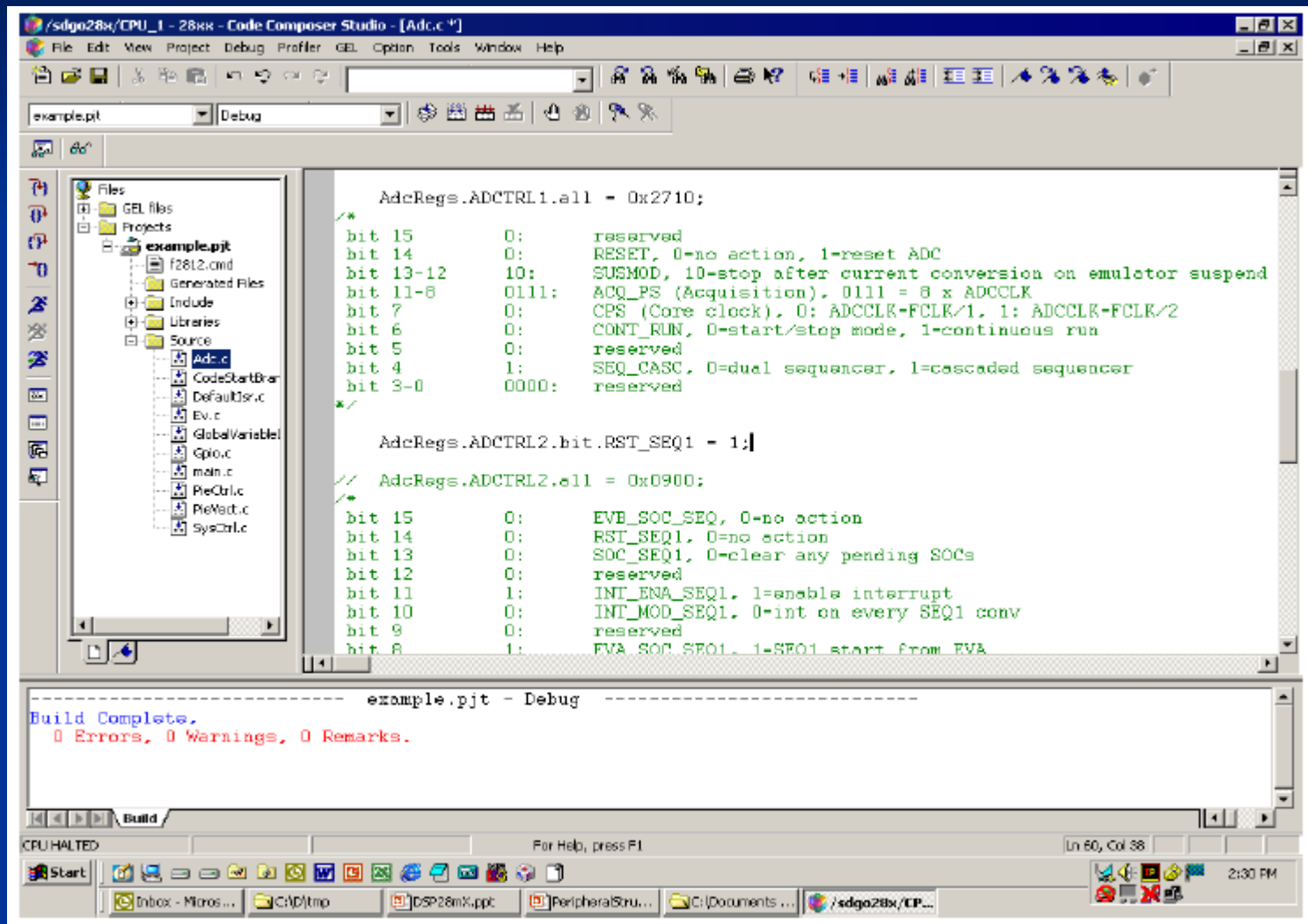
## ■ **Визуальный компоновщик:**





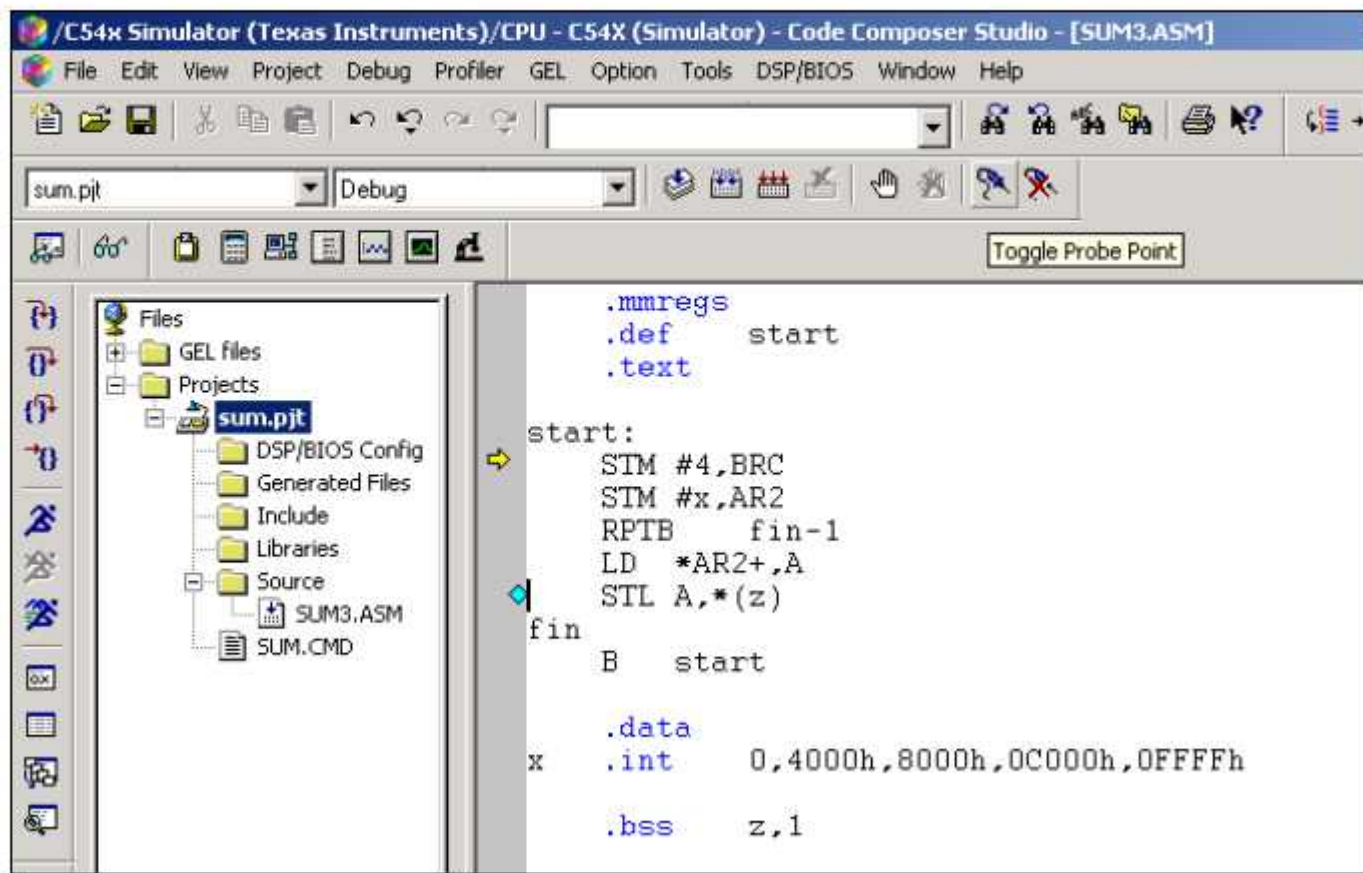
# Интегрированная среда программирования Code Composer Studio

## Code Maestro:



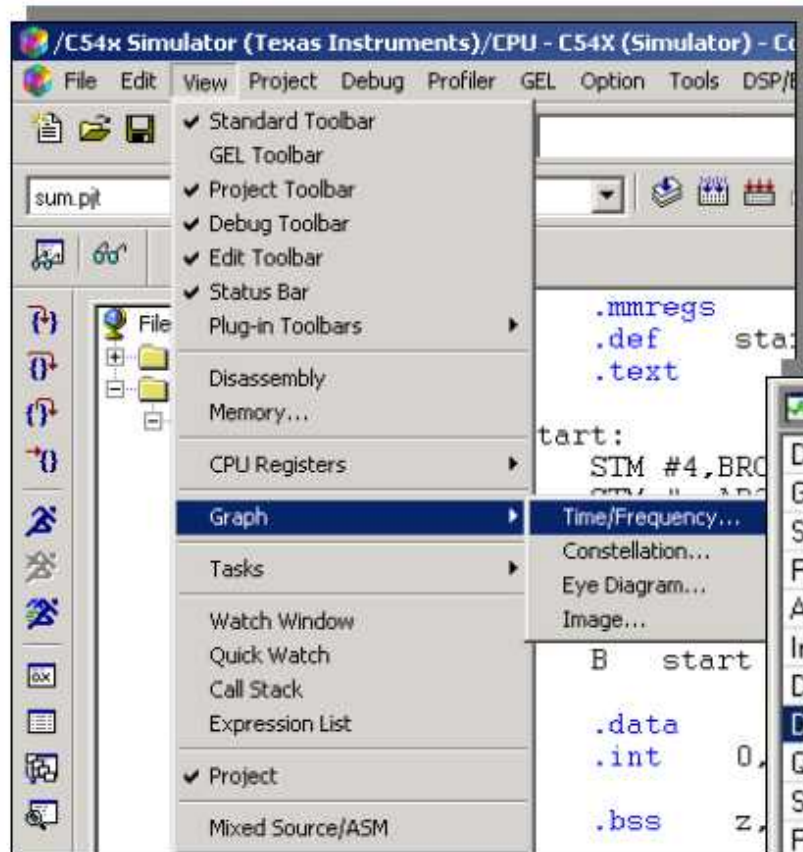
# Интегрированная среда программирования Code Composer Studio

- **Динамическое отображение графика:**
  - Встраивание Probe point



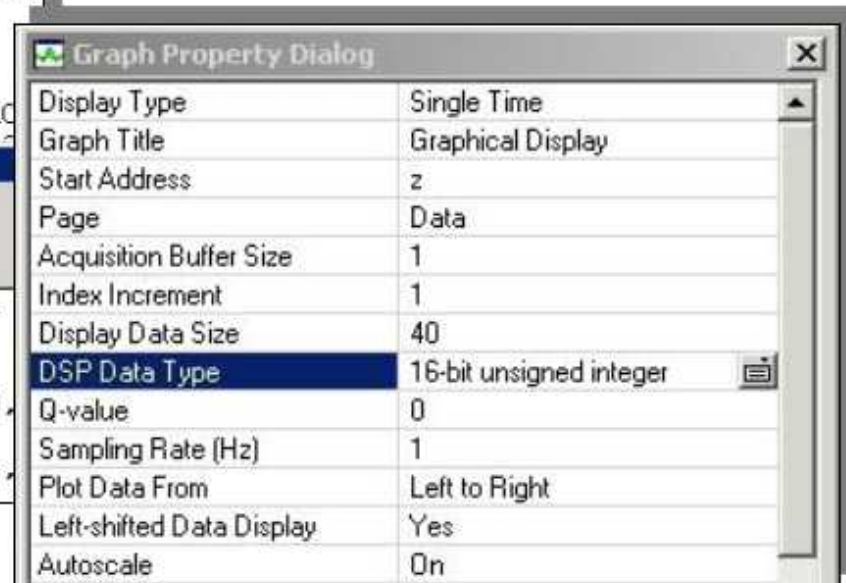
# Интегрированная среда программирования Code Composer Studio

## ■ Динамическое отображение графика:



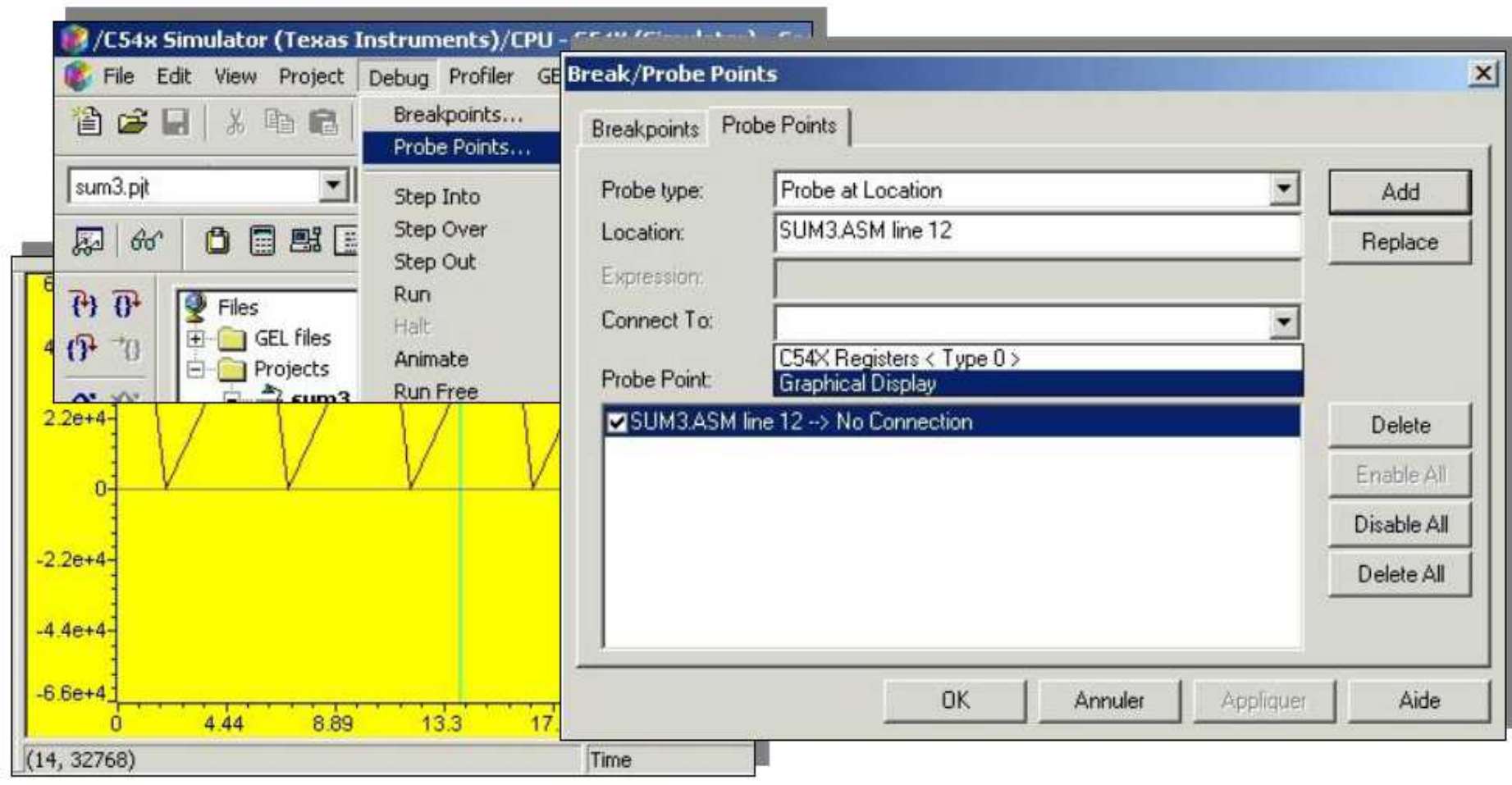
### -Конфигурация отображения

- Start adresse : z
- Acq buffer size : 1
- Display size : 40
- Data type : 16 bits unsigned



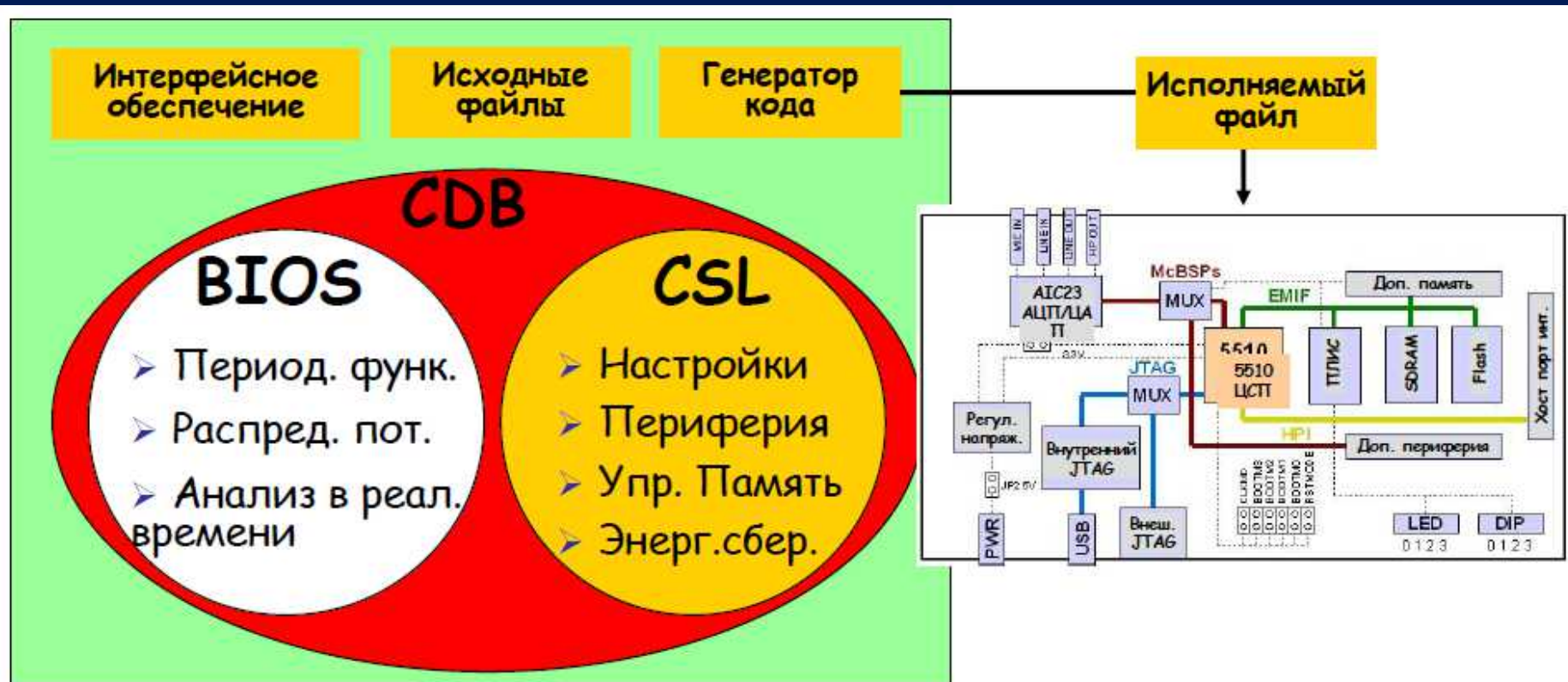
# Интегрированная среда программирования Code Composer Studio

## ■ Динамическое отображение графика:





# Интегрированная среда программирования Code Composer Studio



CDB - Конфигурационная база данных

BIOS - Операционная система

CSL - библиотеки поддержки кристаллов

# Интегрированная среда программирования Code Composer Studio

## ■ Условные обозначения в Ассемблере:



- ◆ Любой печатный ASCII текст разрешен
- ◆ Использовать расширение .asm для файлов
- ◆ Команды и директивы не могут начинаться с начала строки
- ◆ Комментарии к любой строке после точки с запятой

# Интегрированная среда программирования Code Composer Studio

## ■ Построение проекта:

Пример для обработки :  $z = x + y$

коде

get x  
add y  
store z

loop

КОНСТАНТЫ

x = 2  
y = 7

переменные

z

```
.text
start: LD    #X,DP
        LD    @x,A
        ADD   @y,A
        STL   A,*(z)
        B     start
```

```
.data
x       .int  2
y       .int  7
```

```
.bss    z,1
```



# Интегрированная среда программирования Code Composer Studio

## ■ Директивы ассемблера и типы данных:

### Основные директивы

<code>.sect</code>	<u>инициализируемая</u> секция для кода и данных
<code>.usect</code>	<u>не инициализируемая</u> секция create для данных
<code>.byte</code>	8-битная константа выравнивание по словам
<code>.int(.word)</code>	16-битная константа
<code>.long</code>	32-битная константа
<code>.ref/.def</code>	используется для определения символов
<code>.global</code>	совместно с <code>.ref</code> и <code>.def combined</code>

### Типы данных

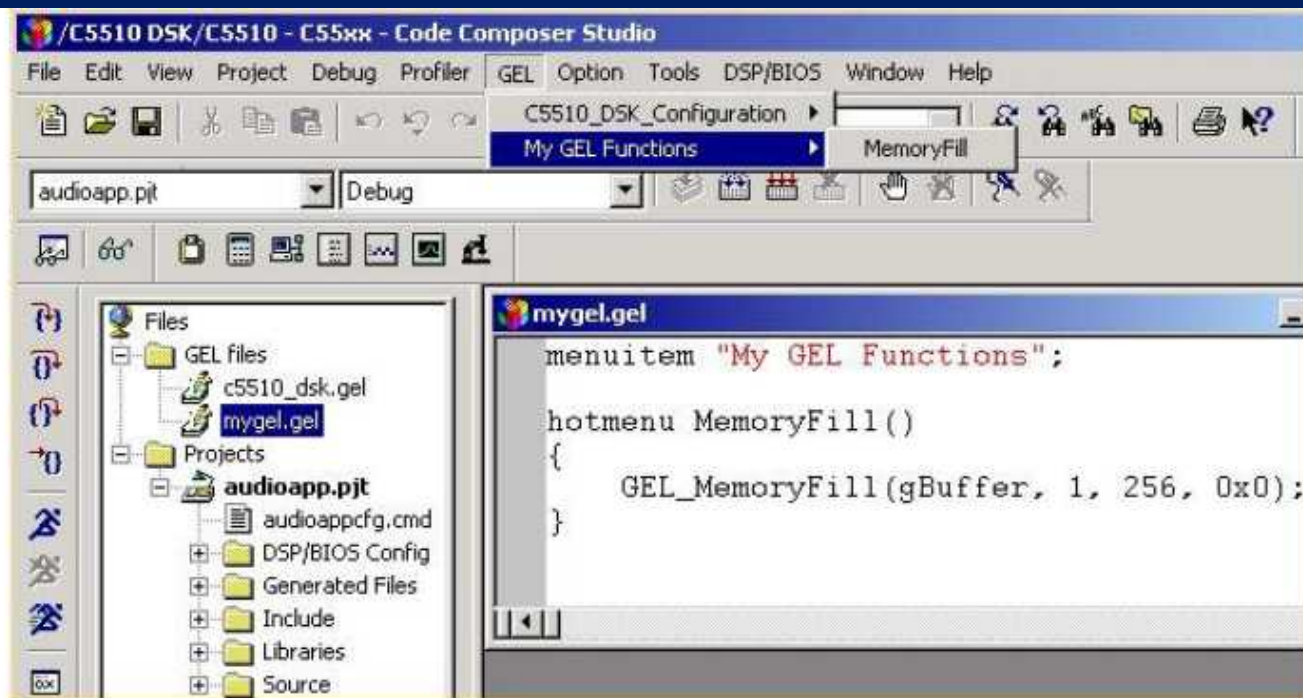
10	десятичный (default)
0Ah,	
0xA	шестнадцатеричный
1010b,	
1010B	бинарный

<code>.set/.equ</code>	равенство значению или символу
<code>.asg</code>	назначение ассемблерных констант



# Интегрированная среда программирования Code Composer Studio

## ■ Автоматизация при использовании GEL сценариев:



◆ GEL: General  
Extension  
Language

◆ Синтаксис языка C

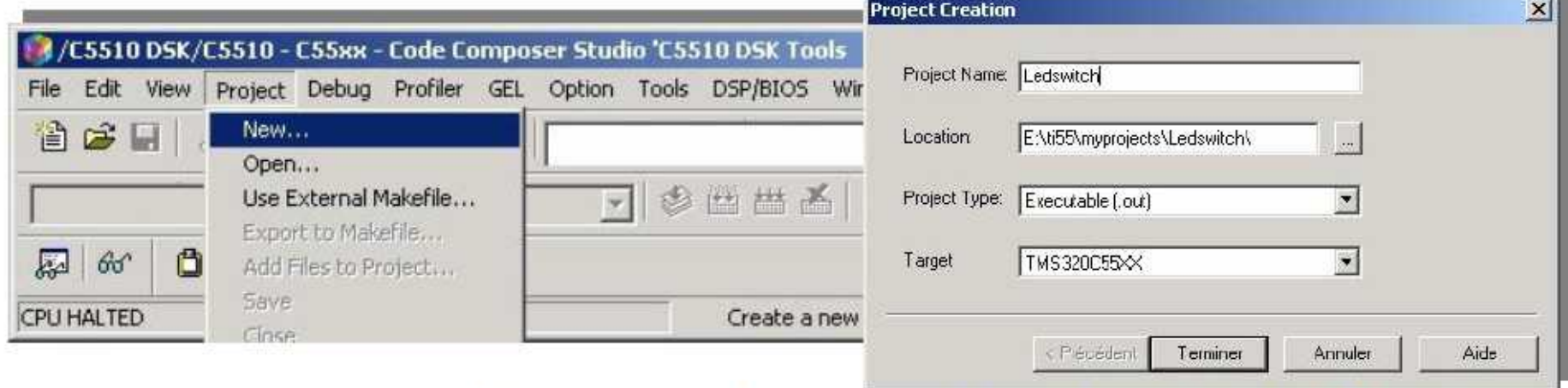
◆ Большое число команд отладки  
в виде GEL функций

◆ Написание собственных функций

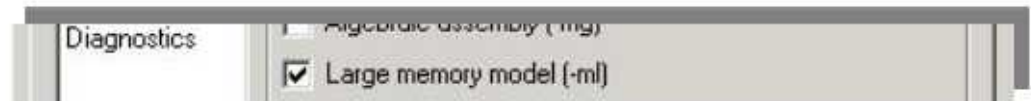
◆ Создание элементов меню GEL

# Code Composer Studio. Пример создания проекта

## ■ Создание нового проекта LedSwitch

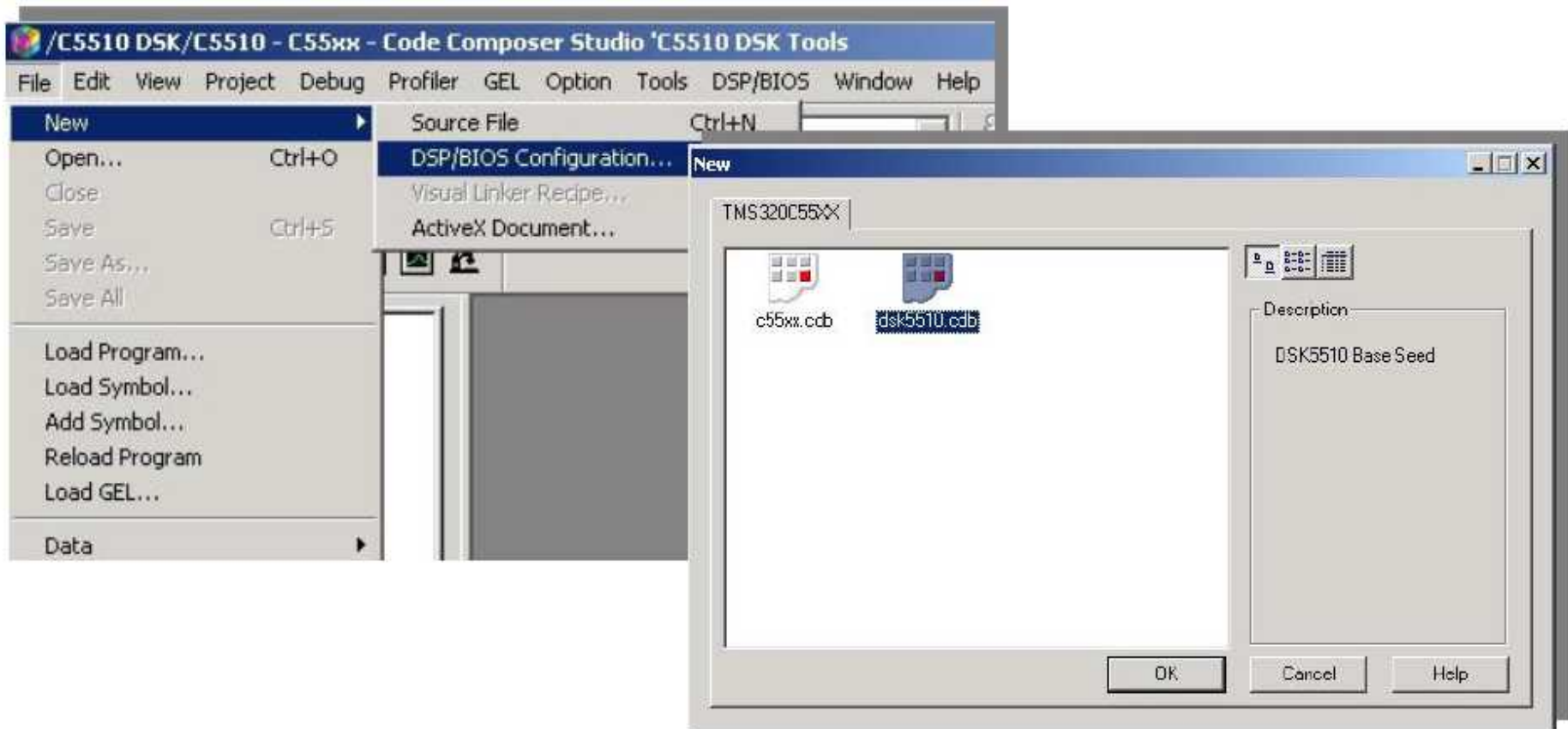


- Изменить модель памяти в меню « Project>Options » и выбрать « large memory model »



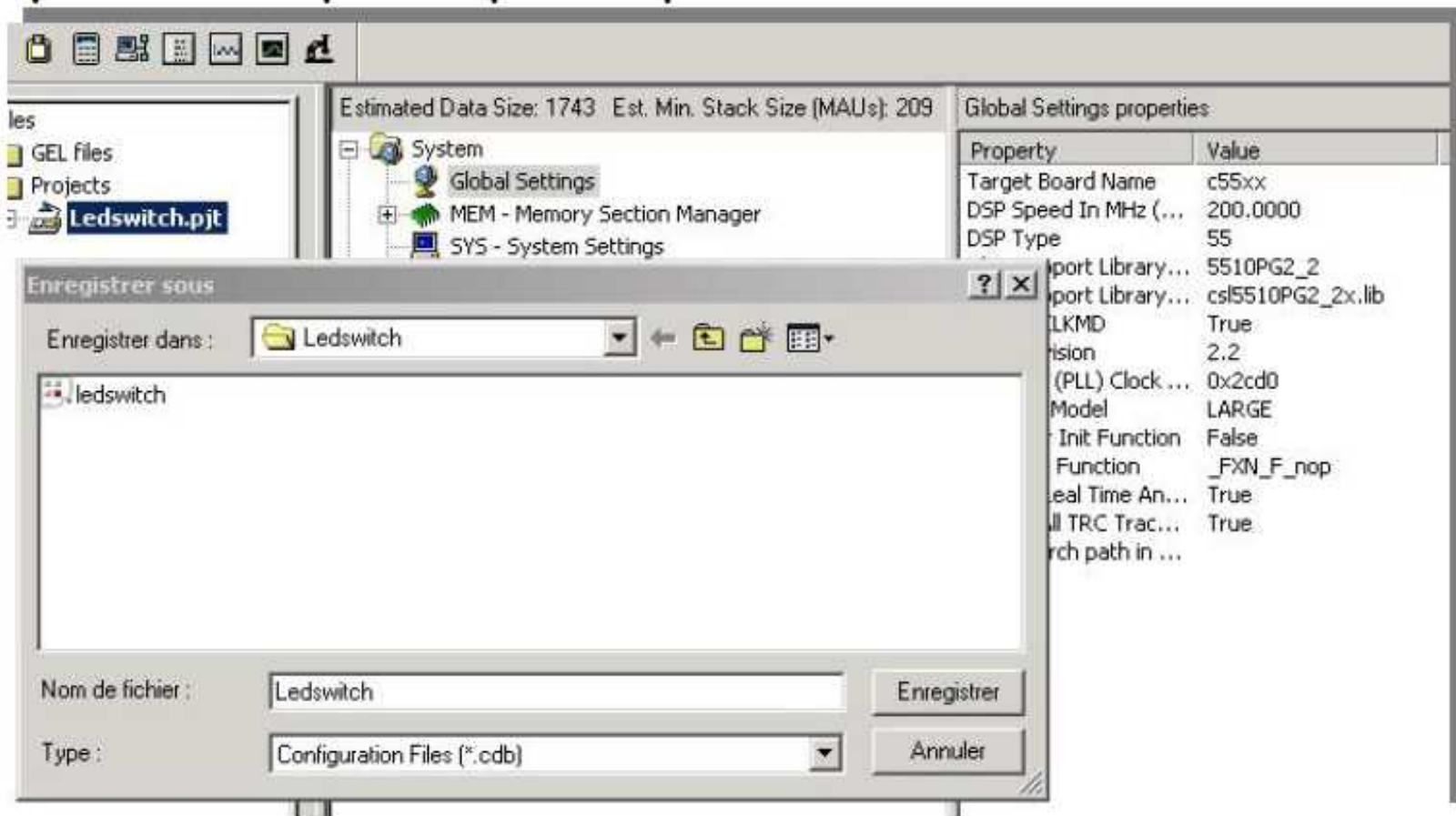
# Code Composer Studio. Пример создания проекта

- Выбрать новый конфигурационный файл (\*.cdb)



# Code Composer Studio. Пример создания проекта

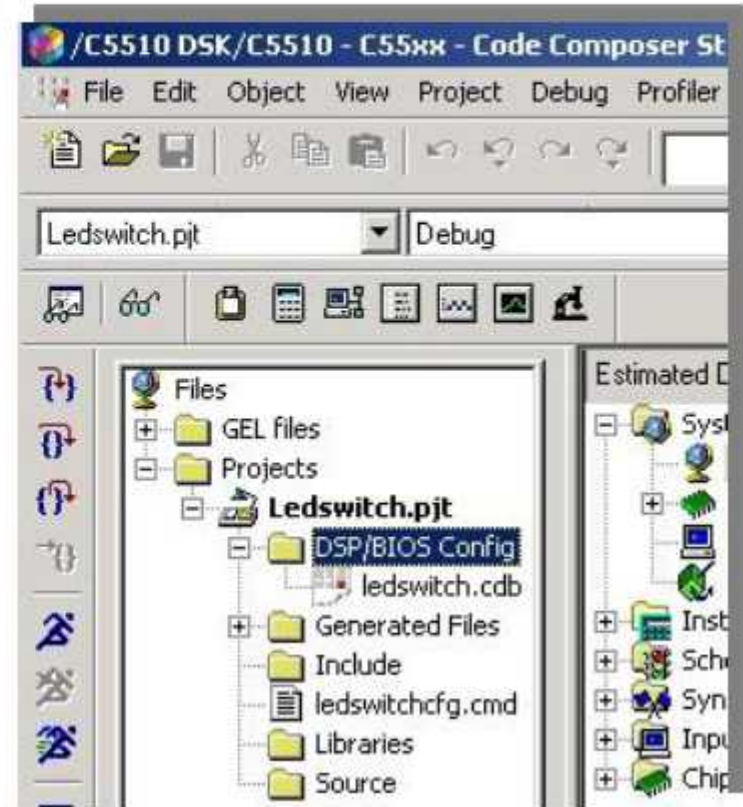
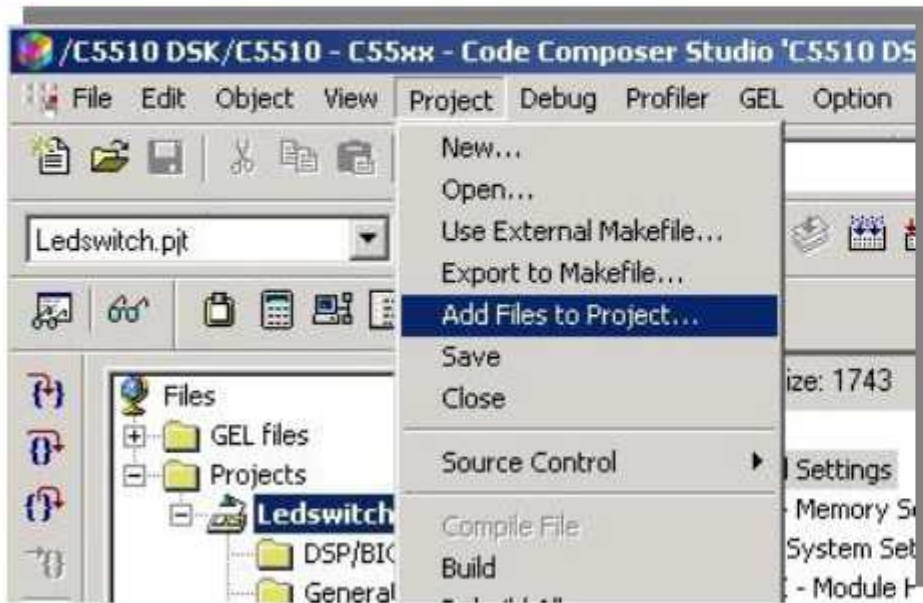
- Сохранить (File>Save) новый конфигурационный файл в директорию проекта





# Code Composer Studio. Пример создания проекта

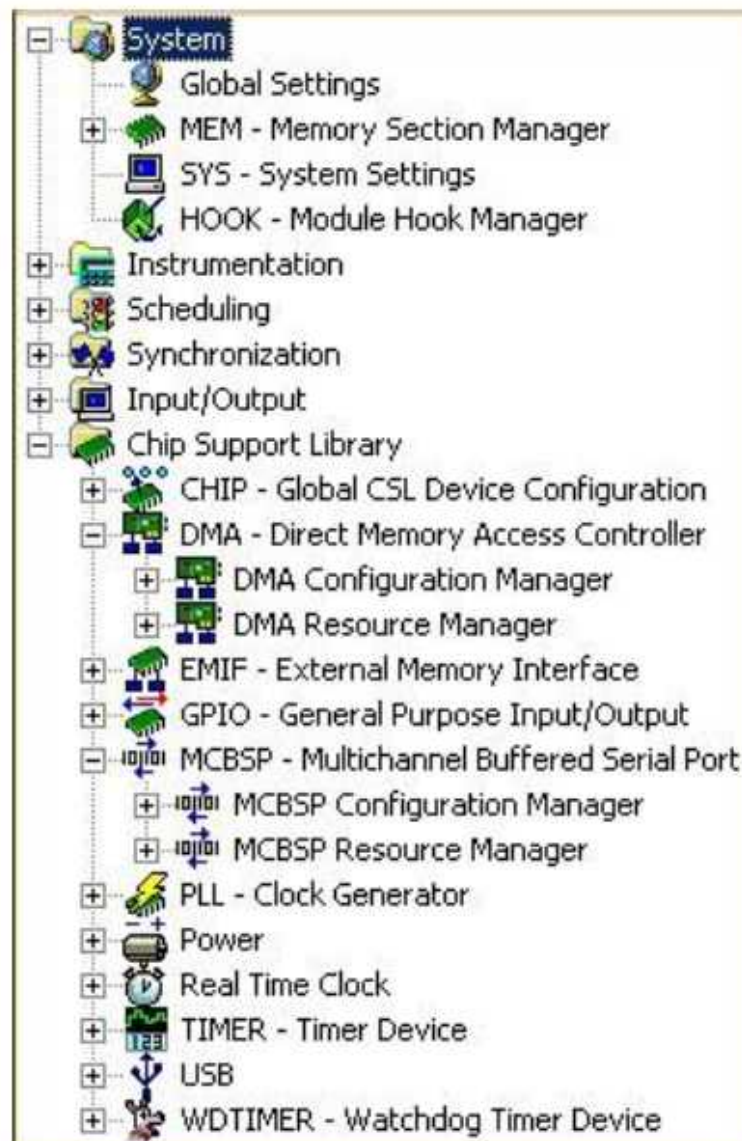
- Добавить в проект два сгенерированных файла: конфигурационный файл (\*.cdb) и командный файл компоновки (\*.cmd)



# Code Composer Studio. Пример создания проекта

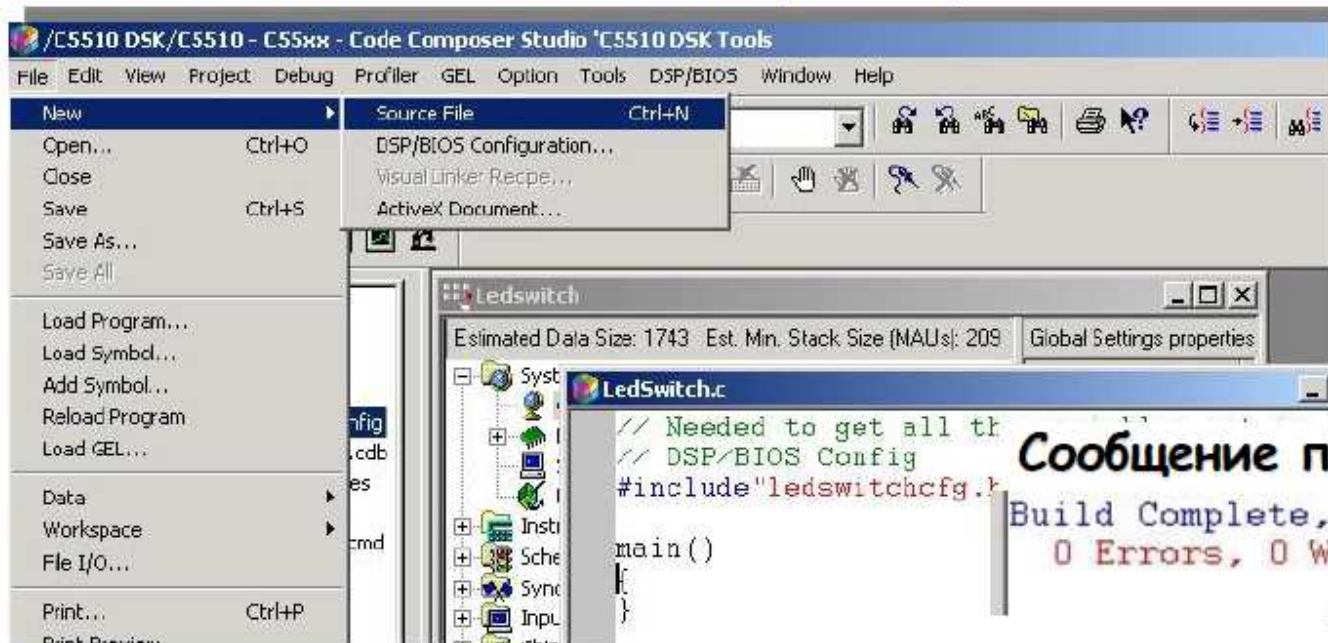
## ■ CDB – конфигурационная база данных:

- ◆ Настройка диспетчеров BIOS, RTA и других функций BIOS
- ◆ Автоматическое управление: библиотеками реального времени, векторами прерываний, системным сбросом и т.д.
- ◆ Управление конфигурацией памяти системы (создание CMD файла)
- ◆ CSL упрощает настройку периферии



# Code Composer Studio. Пример создания проекта

- Создать основной файл: LedSwitch.c, который должен включать сгенерированные заголовочные файлы
- Добавить его к проекту



**Сообщение после компиляции ...**  
Build Complete,  
0 Errors, 0 Warnings, 0 Remarks.



# Code Composer Studio. Пример создания проекта

