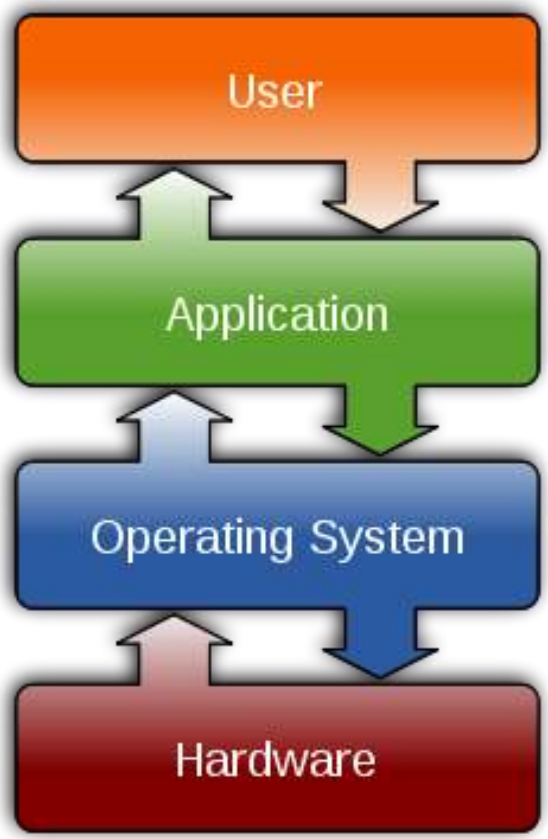
Introduction to CCSv5

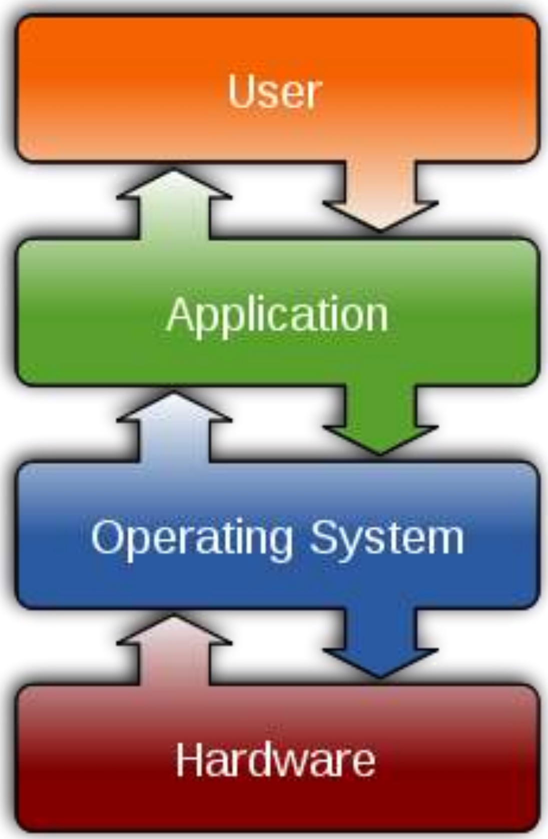
Outline

- **Intro to CCSv5**
  - Functional Overview
  - Perspectives
  - Projects
  - Target Configuration
  - Build Config & Options
  - Licensing/Pricing
  - CCSv5 – For More Info…

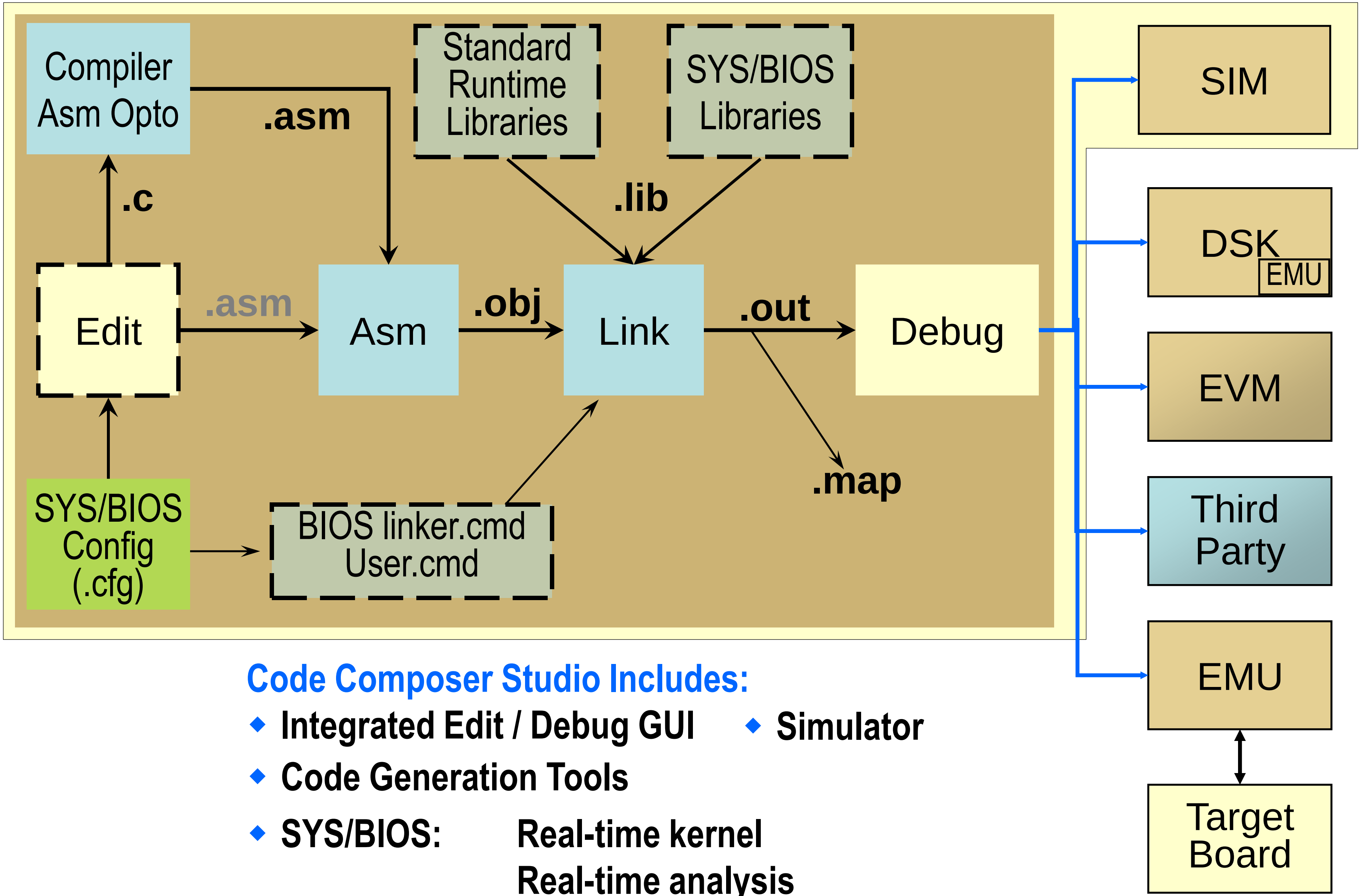


Outline

- Intro to CCSv5
  - **Functional Overview**
  - Perspectives
  - Projects
  - Target Configuration
  - Build Config & Options
  - Licensing/Pricing
  - CCSv5 – For More Info…



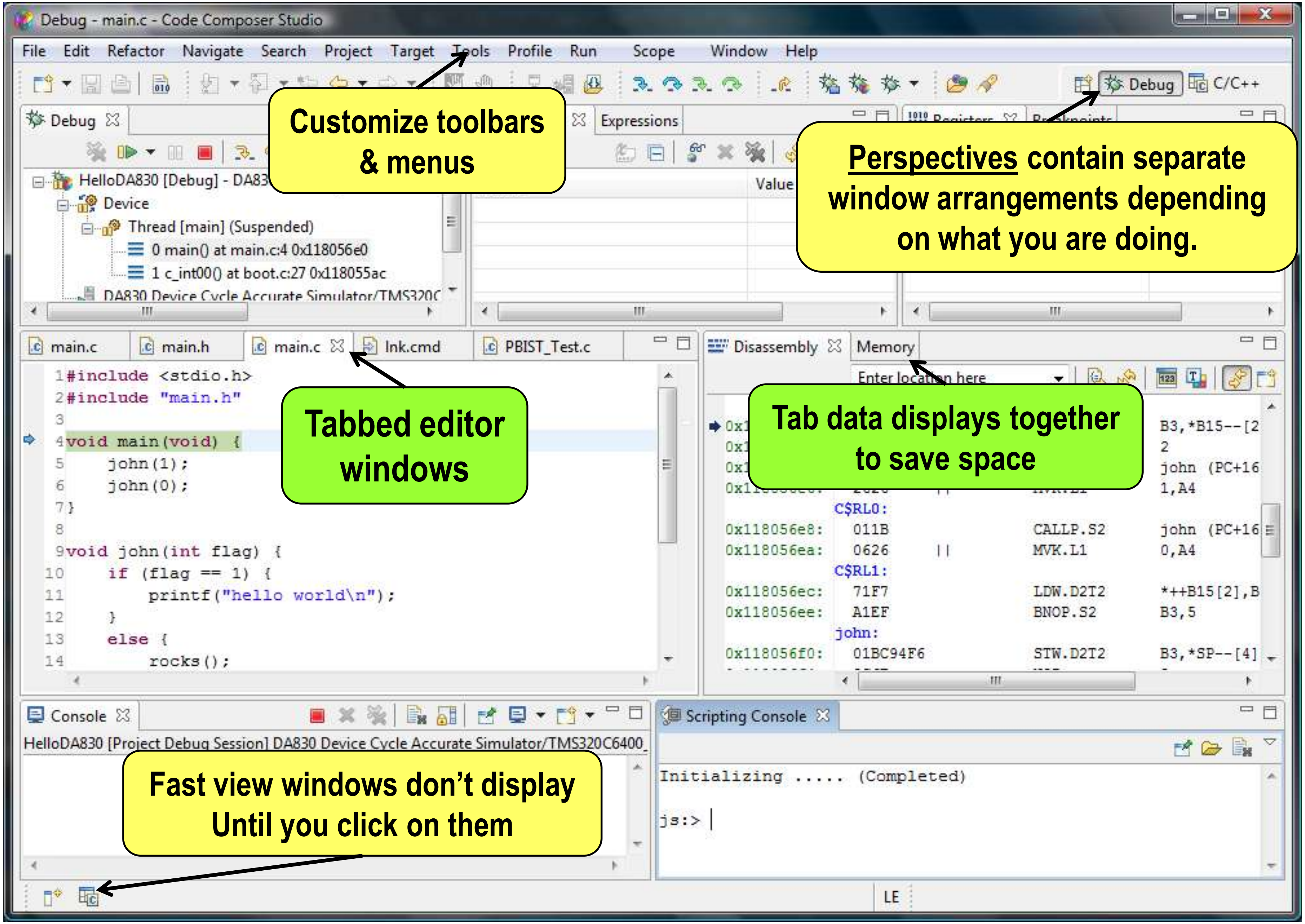
CCS Functional Overview



Code Composer Studio Includes:

- Integrated Edit / Debug GUI
- Simulator
- Code Generation Tools
- SYS/BIOS: Real-time kernel
  Real-time analysis

CCSv5 “GUI” Environment – Space Saving



HelloDA830 [Debug] - DA83
Device
Thread [main] (Suspended)
0 main() at main.c:4 0x118056e0
1 c_int00() at boot.c:27 0x118055ac
DA830 Device Cycle Accurate Simulator/TMS320C

```
#include <stdio.h>
#include "main.h"

void main(void) {
    john(1);
    john(0);
}

void john(int flag) {
    if (flag == 1) {
        printf("hello world\n");
    }
    else {
        rocks();
```

```
0x1                                      B3,*B15--[2
0x1                                      2
0x1                                      john (PC+16
0x11                         MVK.L1      1,A4
               C$RL0:
0x118056e8:    011B          CALLP.S2    john (PC+16
0x118056ea:    0626    ||    MVK.L1      0,A4
               C$RL1:
0x118056ec:    71F7          LDW.D2T2    *++B15[2],B
0x118056ee:    A1EF          BNOP.S2     B3,5
               john:
0x118056f0:    01BC94F6      STW.D2T2    B3,*SP--[4]
```

HelloDA830 [Project Debug Session] DA830 Device Cycle Accurate Simulator/TMS320C6400_

```
Initializing ..... (Completed)

js:>
```

Customize toolbars & menus

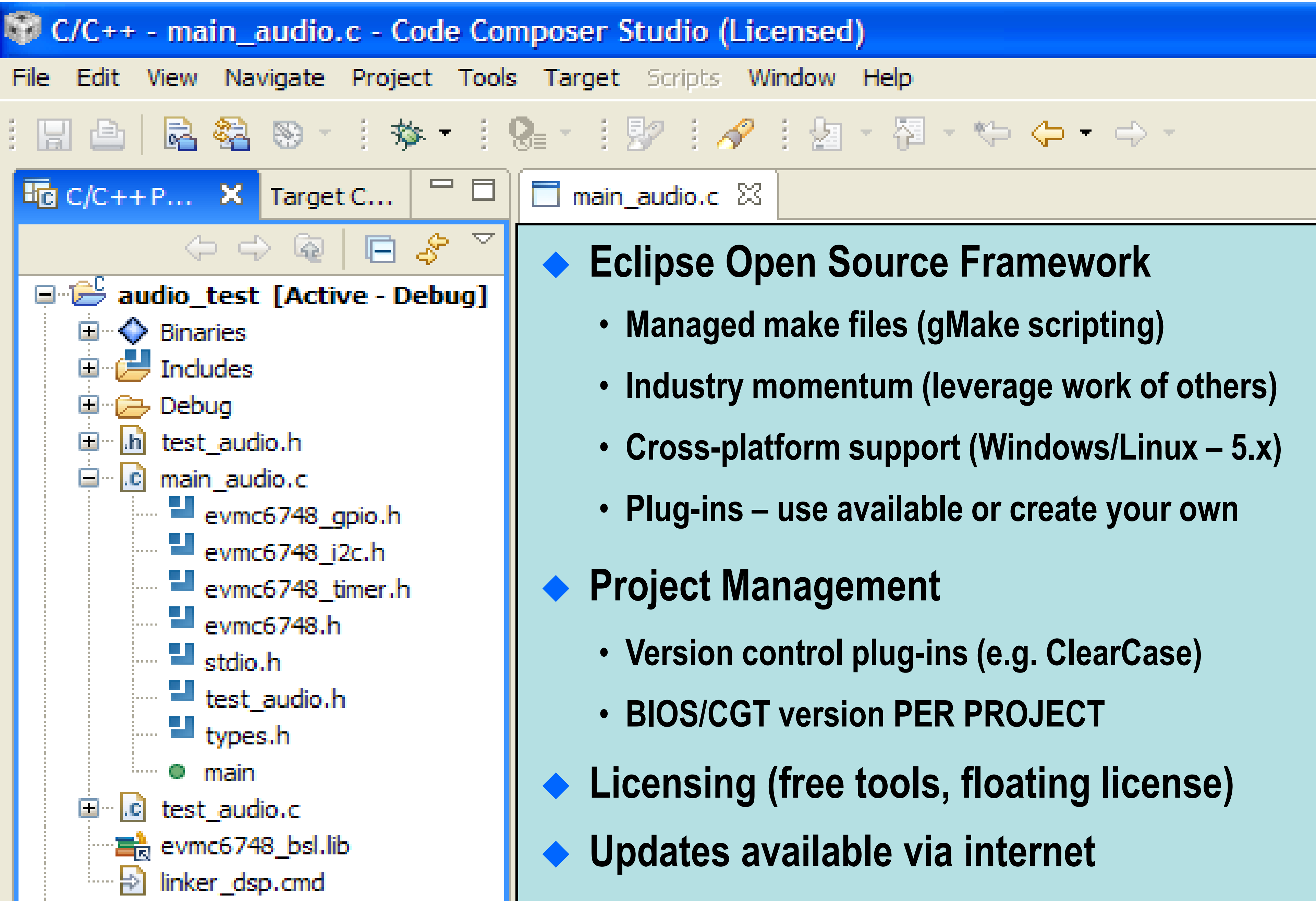
Perspectives contain separate window arrangements depending on what you are doing.

Tabbed editor windows

Tab data displays together to save space

Fast view windows don’t display Until you click on them

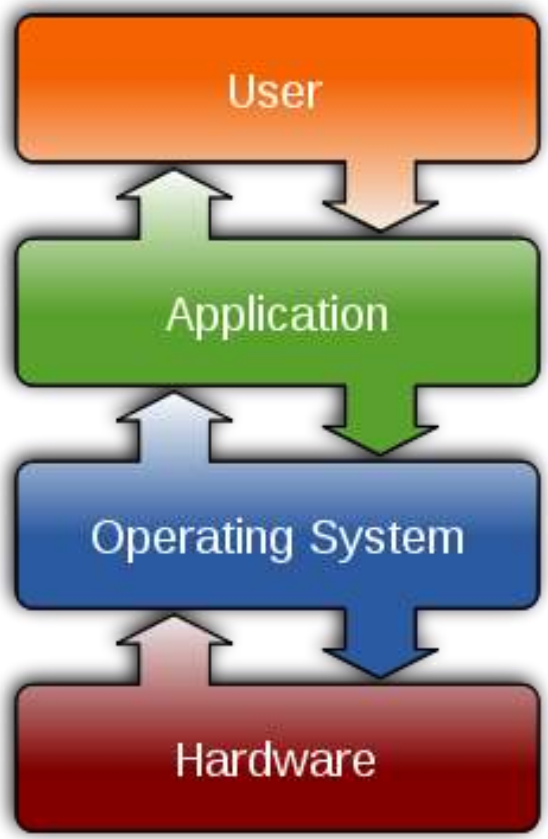
CCSv5 (Eclipse) Benefits



- Eclipse Open Source Framework
  - Managed make files (gMake scripting)
  - Industry momentum (leverage work of others)
  - Cross-platform support (Windows/Linux – 5.x)
  - Plug-ins – use available or create your own
- Project Management
  - Version control plug-ins (e.g. ClearCase)
  - BIOS/CGT version PER PROJECT
- Licensing (free tools, floating license)
- Updates available via internet

Outline

- Intro to CCSv5
  - Functional Overview
  - **Perspectives**
  - Projects
  - Target Configuration
  - Build Config & Options
  - Licensing/Pricing
  - CCSv5 – For More Info…



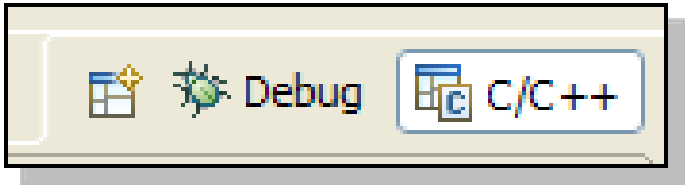
Perspectives

- ***Perspectives*** **– a set of windows, views and menus that correspond to a specific set of tasks**

- **Two *default perspectives* are provided with CCSv5:**

Debug

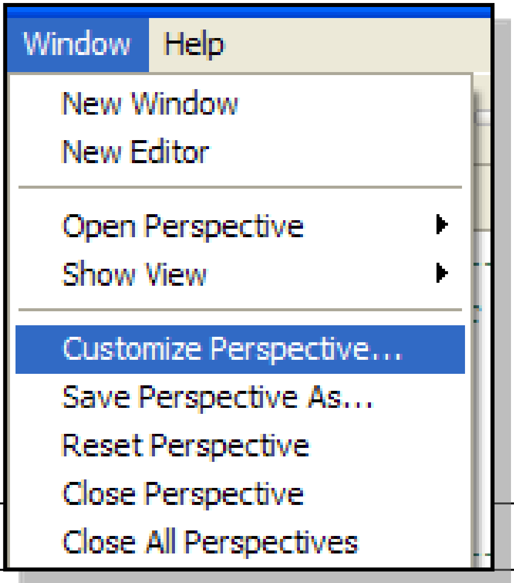
- Debug Views
- Watch/Memory
- Graphs, etc.

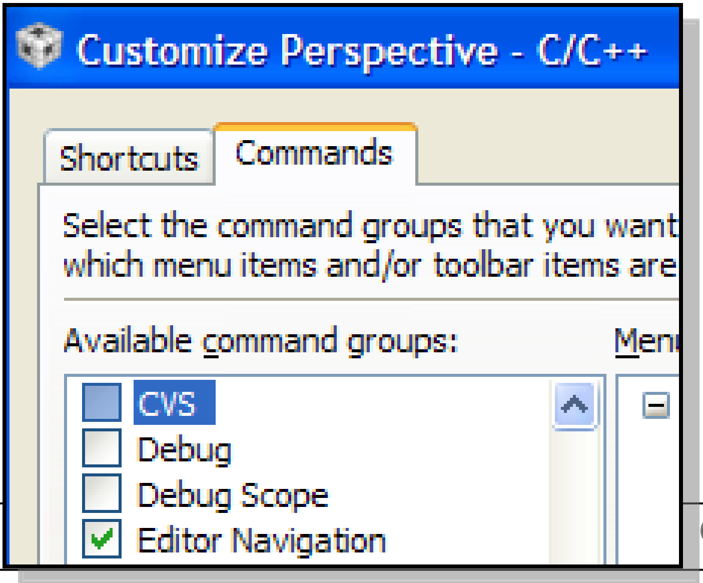


C/C++

- Code Dev't Views
- Project Contents
- Editor

- **Users can *customize perspectives* and save them:**





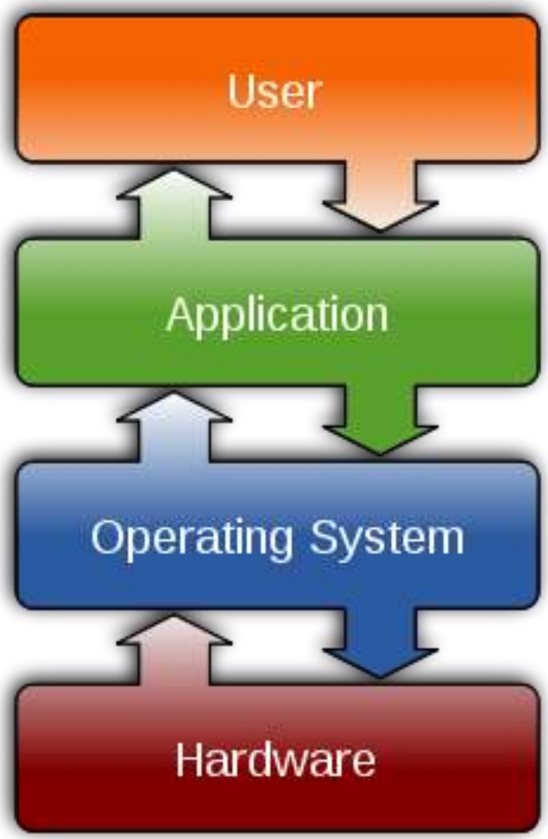
Select the command groups that you want
which menu items and/or toolbar items are

Available command groups:

□ CVS
□ Debug
□ Debug Scope
☑ Editor Navigation

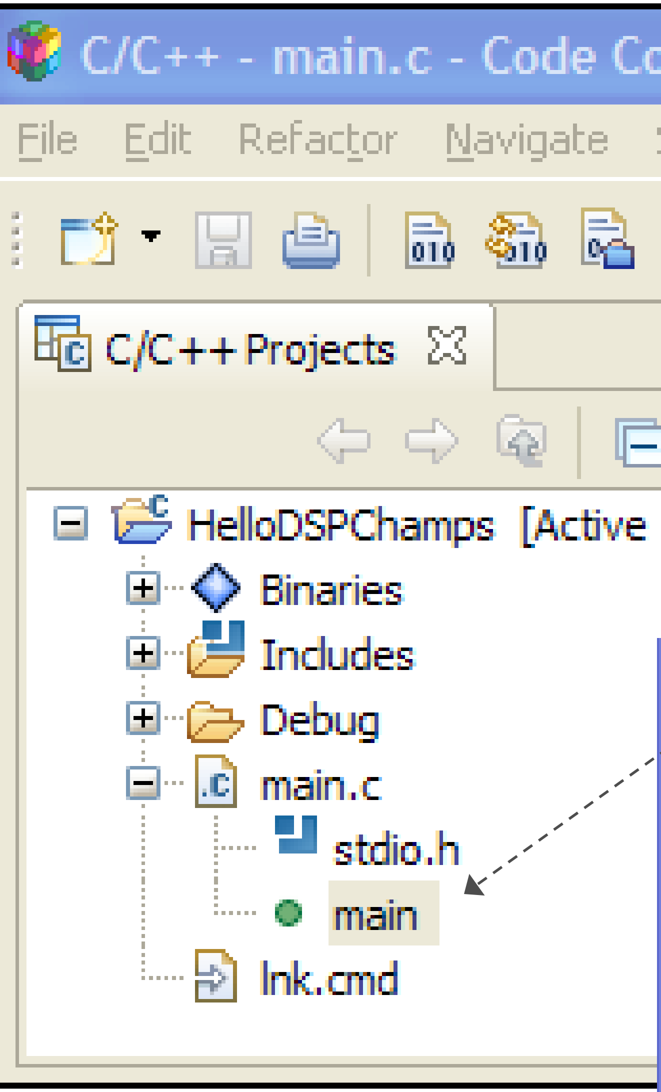
Outline

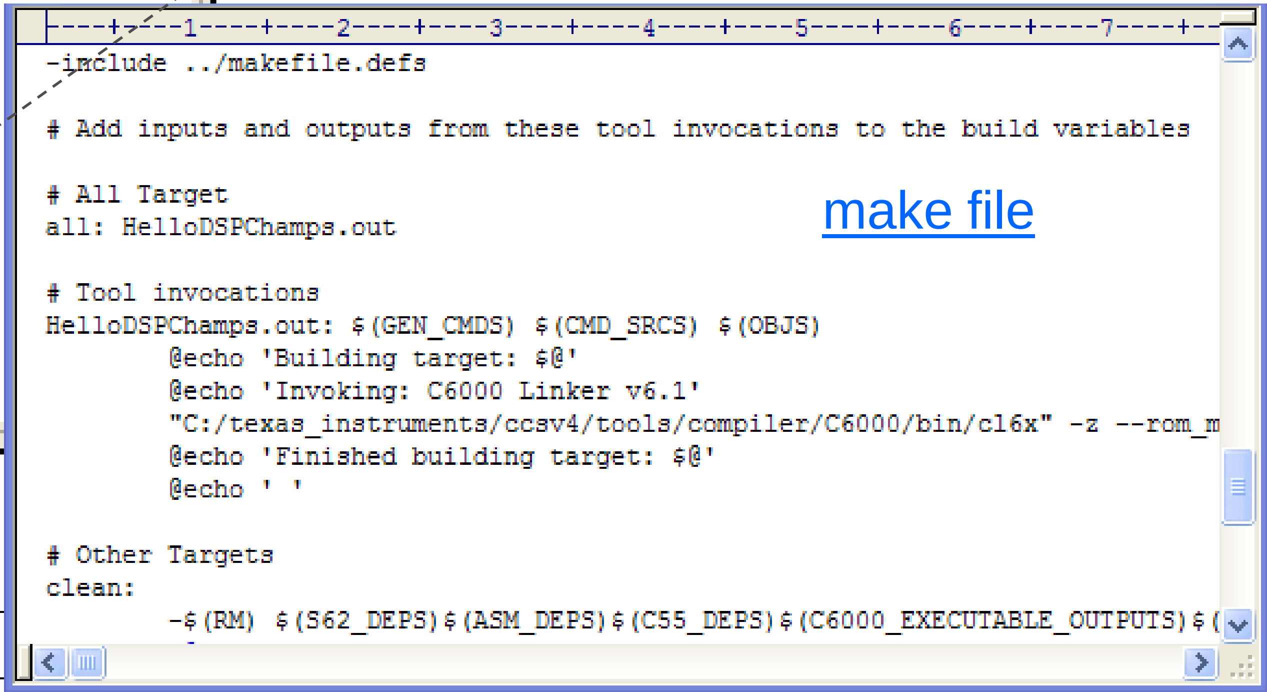
- Intro to CCSv5
  - Functional Overview
  - Perspectives
  - **Projects**
  - Target Configuration
  - Build Config & Options
  - Licensing/Pricing
  - CCSv5 – For More Info…



Eclipse “Projects”

- CCSv5 is PROJECT-centric
- Eclipse uses *managed makefiles* as their build scripts – as opposed to *pjt* files
- Eclipse projects are folder based
  - “Adding file” copies it to folder
  - “Linking file” references original file
  - Project explorer shows folder contents
- Project explorer lists functions





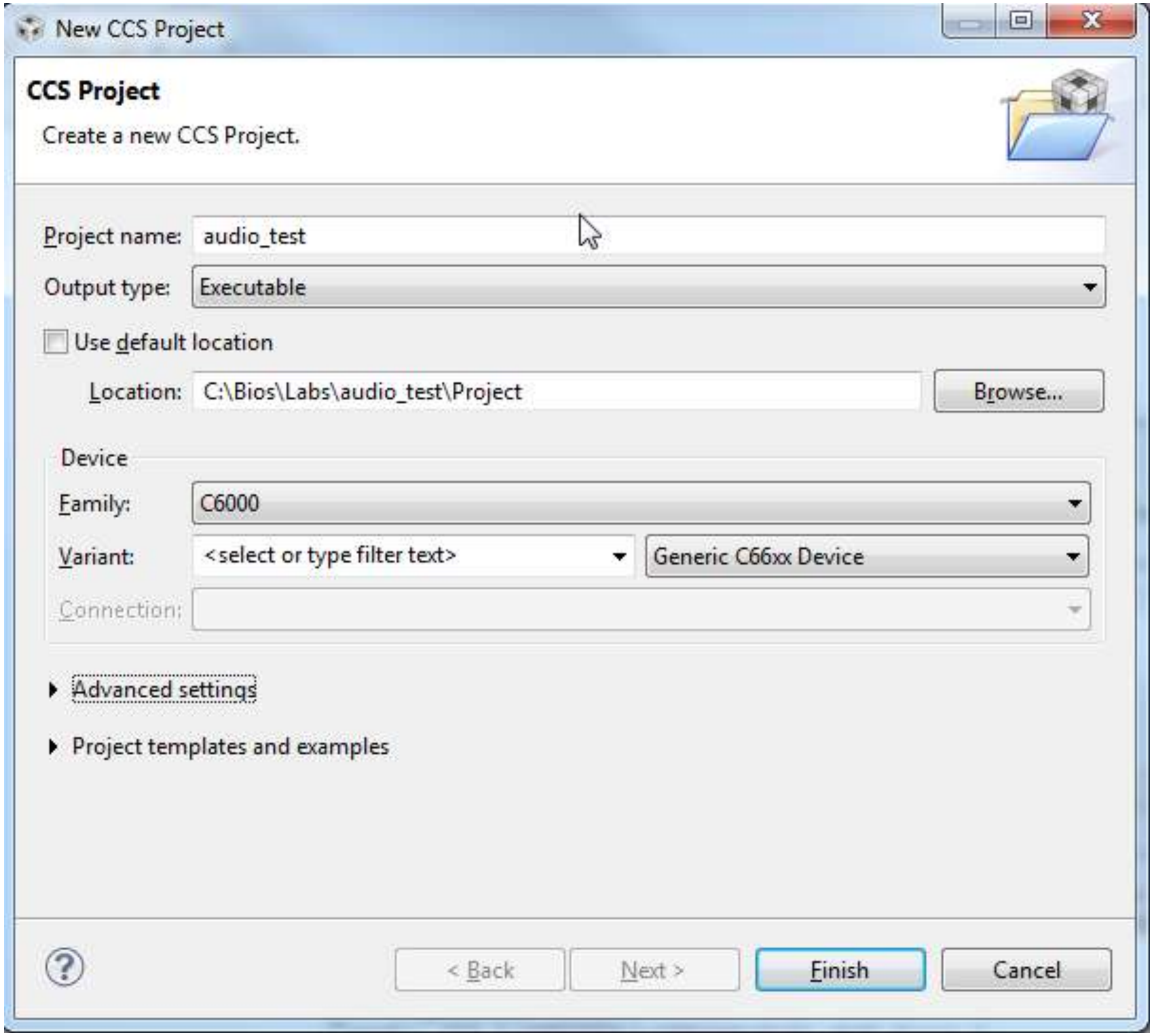
```
-include ../makefile.defs

# Add inputs and outputs from these tool invocations to the build variables

# All Target
all: HelloDSPChamps.out

# Tool invocations
HelloDSPChamps.out: $(GEN_CMDS) $(CMD_SRCS) $(OBJS)
	@echo 'Building target: $@'
	@echo 'Invoking: C6000 Linker v6.1'
	"C:/texas_instruments/ccsv4/tools/compiler/C6000/bin/cl6x" -z --rom_m
	@echo 'Finished building target: $@'
	@echo ' '

# Other Targets
clean:
	-$(RM) $(S62_DEPS)$(ASM_DEPS)$(C55_DEPS)$(C6000_EXECUTABLE_OUTPUTS)$(
```

make file

How do we create a NEW project?

Creating a New Project (1)

`File→New→CCS Project` (in C++ perspective)



New CCS Project

CCS Project

Create a new CCS Project.

Project name: audio_test

Output type: Executable

□ Use default location

Location: C:\Bios\Labs\audio_test\Project

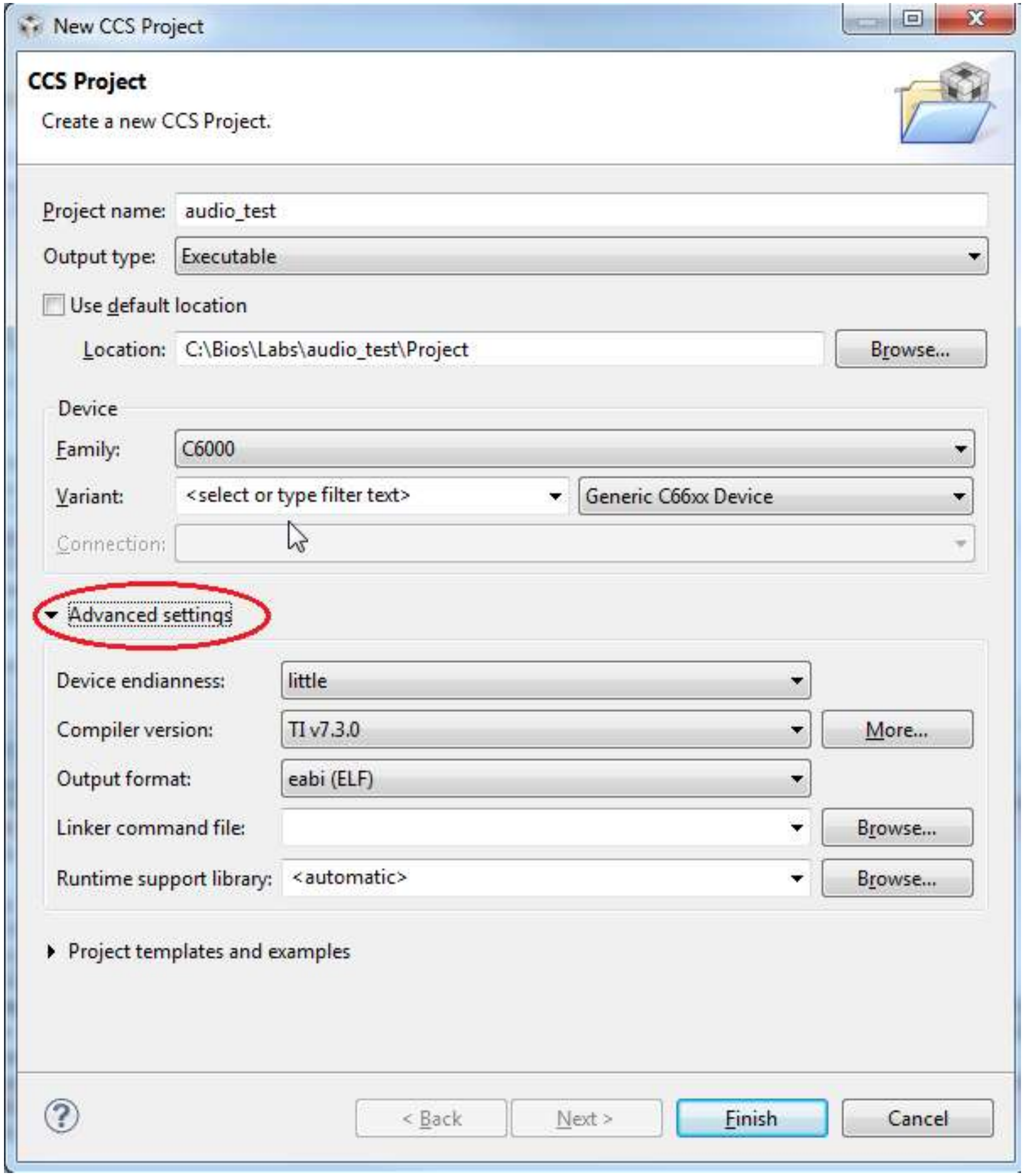
Device

Family: C6000

Variant: <select or type filter text> Generic C66xx Device

Connection:

Creating a New Project (2)



CCS Project

Create a new CCS Project.

Project name: audio_test

Output type: Executable

☐ Use default location

Location: C:\Bios\Labs\audio_test\Project

Device

Family: C6000

Variant: <select or type filter text> Generic C66xx Device

Connection:

▾ Advanced settings

Device endianness: little

Compiler version: TI v7.3.0

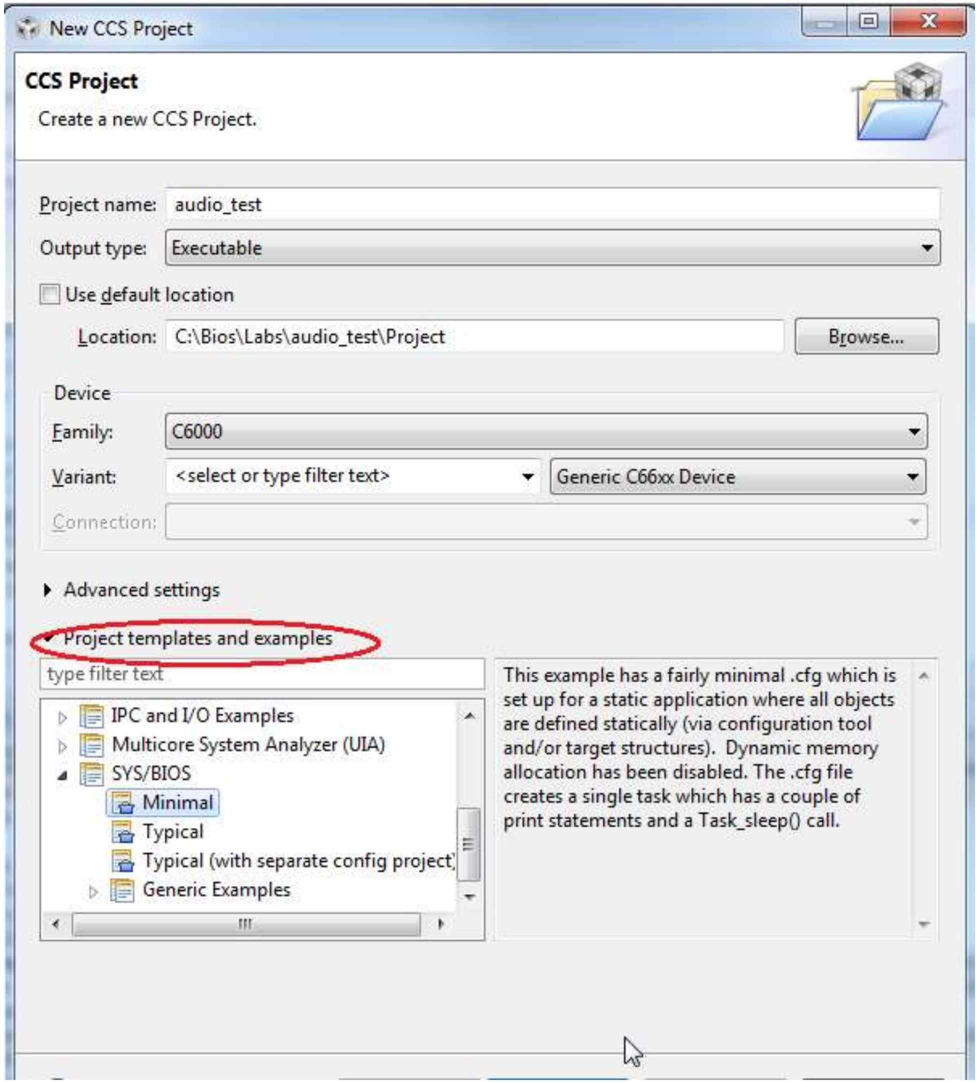
Output format: eabi (ELF)

Linker command file:

Runtime support library: <automatic>

▸ Project templates and examples

Creating a New Project (3)



New CCS Project

CCS Project

Create a new CCS Project.

Project name: audio_test

Output type: Executable

□ Use default location

Location: C:\Bios\Labs\audio_test\Project

Device

Family: C6000

Variant: <select or type filter text> Generic C66xx Device

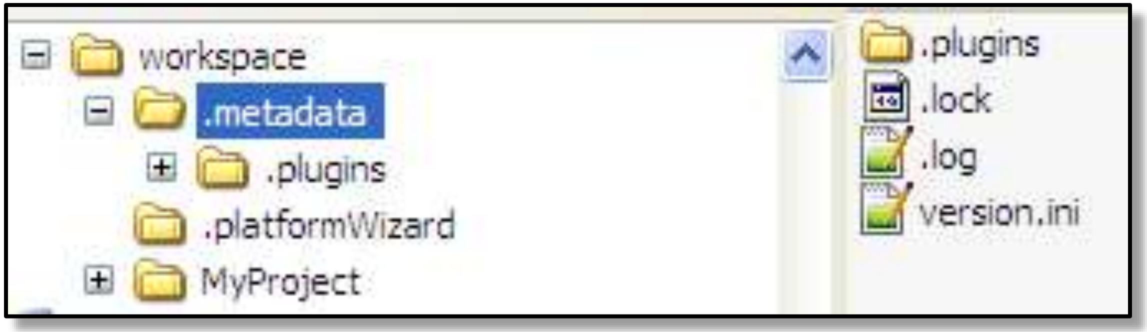
Connection:

This example has a fairly minimal .cfg which is set up for a static application where all objects are defined statically (via configuration tool and/or target structures). Dynamic memory allocation has been disabled. The .cfg file creates a single task which has a couple of print statements and a Task_sleep() call.

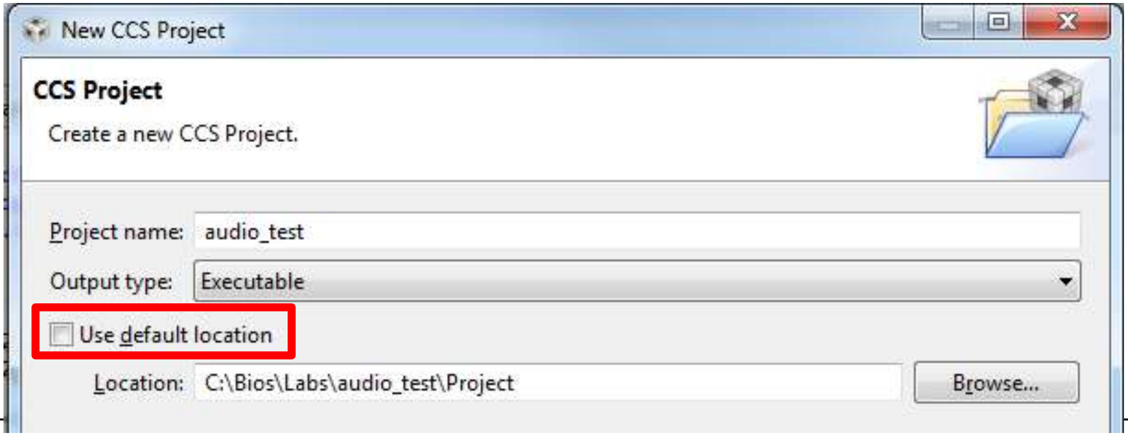
- **Not using SYS/BIOS?**
  - **Choose "Empty Project"**
- **Using SYS/BIOS?**
  - **Choose "Minimal" under SYS/BIOS**

Eclipse "Workspace"

- ***Workspace*** **– a "container" for Eclipse metadata and the default location for all projects**
- **Default Location: `\My Documents\workspace`:**



- **Can change "default" workspace location if desired**
- **User can also locate projects in specific folders:**



New CCS Project

CCS Project

Create a new CCS Project.

Project name: audio_test

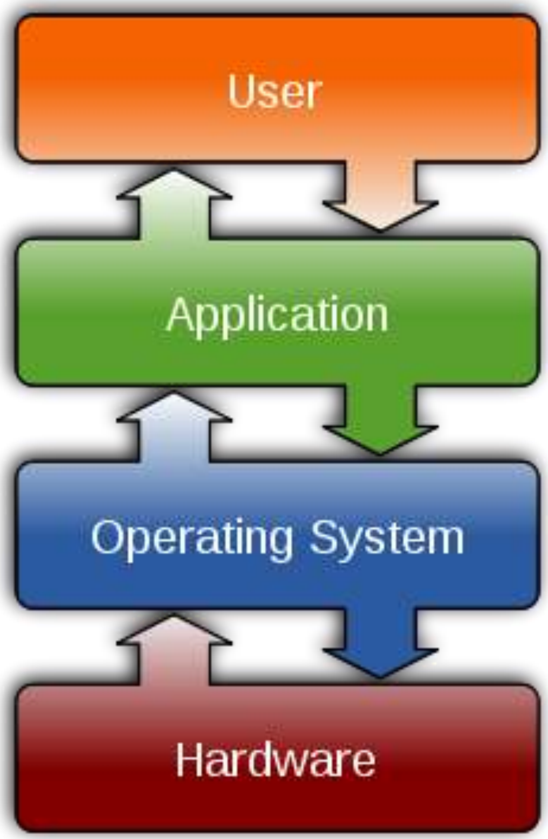
Output type: Executable

□ Use default location

Location: C:\Bios\Labs\audio_test\Project

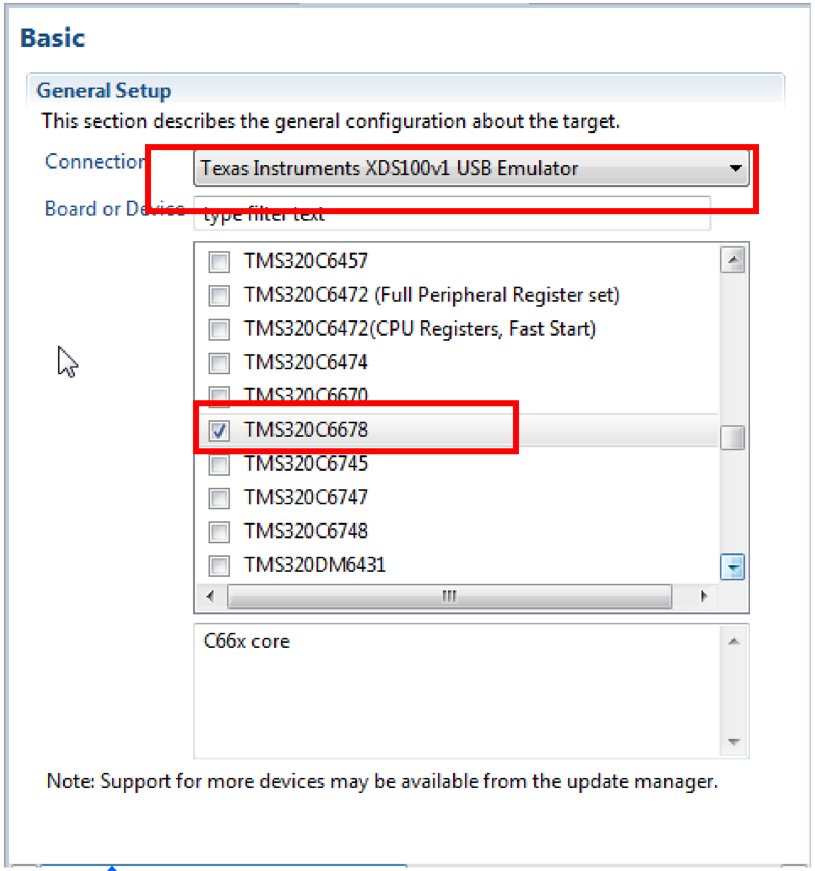
Outline

- Intro to CCSv5
  - Functional Overview
  - Perspectives
  - Projects
  - Target Configuration
  - Build Config & Options
  - Licensing/Pricing
  - CCSv5 – For More Info…
- Intro to SYS/BIOS



Creating a New Target Config File (.ccxml)

- ***Target Configuration*** **– defines your "target" – i.e. emulator/device used, GEL scripts (replaces the old CCS Setup)**
- **Use on a per-project basis (add to project or create User Defined)**



Basic

General Setup

This section describes the general configuration about the target.

Connection: Texas Instruments XDS100v1 USB Emulator

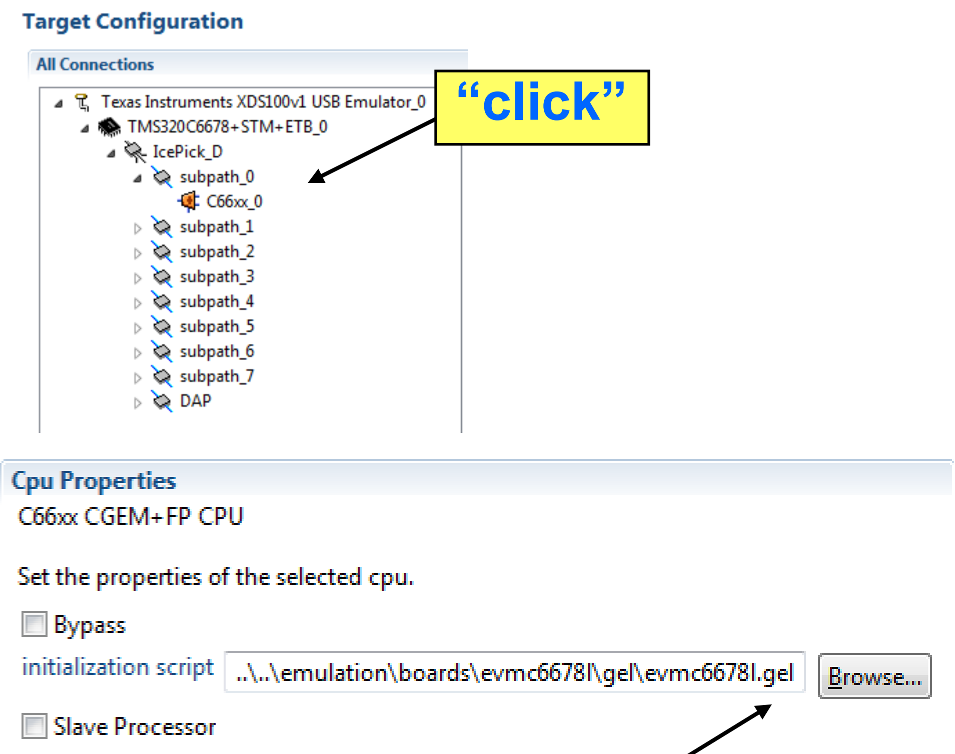
Board or Device: type filter text

- □ TMS320C6457
- □ TMS320C6472 (Full Peripheral Register set)
- □ TMS320C6472(CPU Registers, Fast Start)
- □ TMS320C6474
- □ TMS320C6670
- ☑ TMS320C6678
- □ TMS320C6745
- □ TMS320C6747
- □ TMS320C6748
- □ TMS320DM6431

C66x core

Note: Support for more devices may be available from the update manager.

Advanced Tab



Target Configuration

All Connections

```
Texas Instruments XDS100v1 USB Emulator_0
  TMS320C6678+STM+ETB_0
    IcePick_D
      subpath_0
        C66xx_0
      subpath_1
      subpath_2
      subpath_3
      subpath_4
      subpath_5
      subpath_6
      subpath_7
      DAP
```

"click"

Cpu Properties

C66xx CGEM+FP CPU

Set the properties of the selected cpu.

□ Bypass

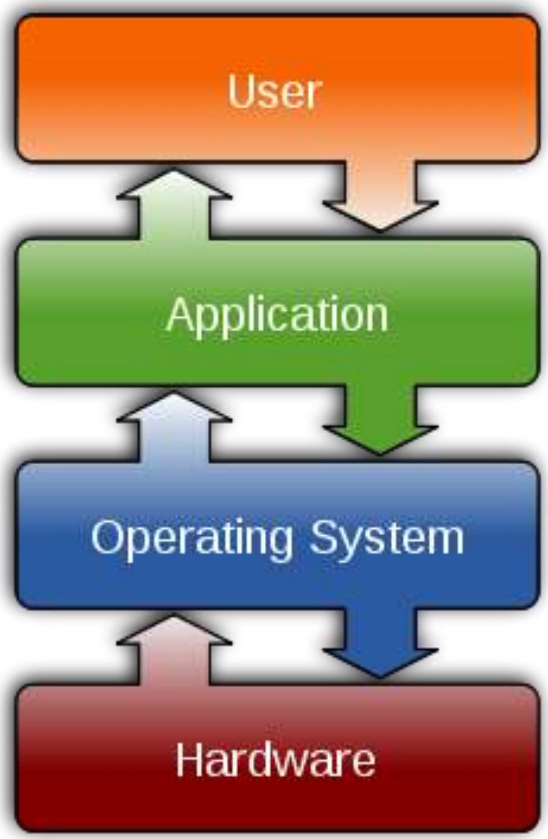
initialization script: ..\..\emulation\boards\evmc6678l\gel\evmc6678l.gel

□ Slave Processor

Specify GEL script here

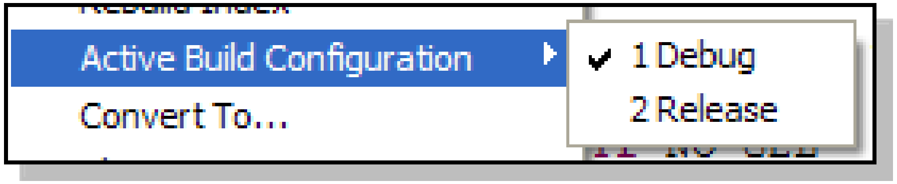
Outline

- Intro to CCSv5
  - Functional Overview
  - Perspectives
  - Projects
  - Target Configuration
  - Build Config & Options
  - Licensing/Pricing
  - CCSv5 – For More Info…



Two Default Build Configurations

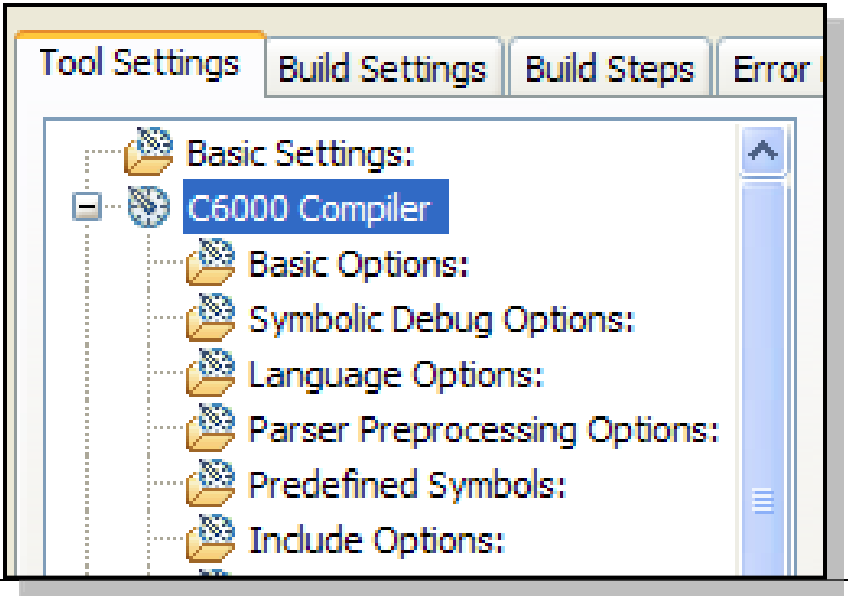
- ***Build Configuration*** **– a set of build options for the compiler and linker (e.g. optimization levels, include DIRs, debug symbols, etc.)**
- **CCSv5 comes std with two DEFAULT build configs: Debug & Release:**



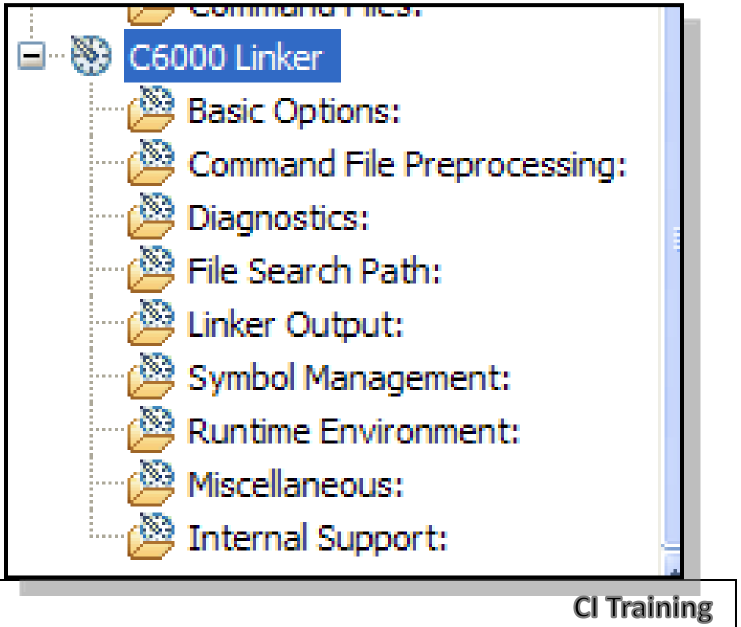
User can create their own config if desired

- **User can modify compiler/linker options via "Build Properties":**

Compiler

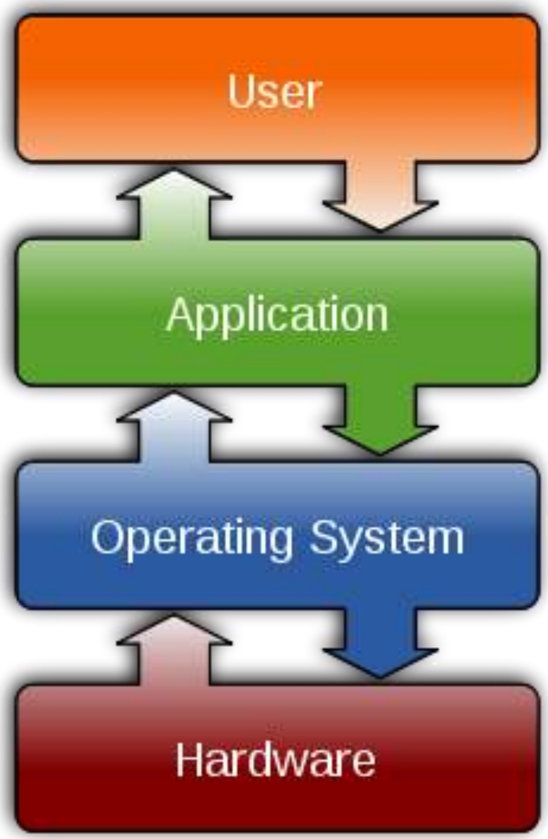


Linker



Outline

- Intro to CCSv5
  - Functional Overview
  - Perspectives
  - Projects
  - Target Configuration
  - Build Config & Options
  - Licensing/Pricing
  - CCSv5 – For More Info…



CCSv5 Licensing & Pricing

◆ Licensing

- Wide variety of options (node locked, floating, time based…)
- All versions (full, DSK, free tools) use same image
- Updates readily available via the internet

◆ Pricing

- Reasonable pricing – includes FREE options noted below

TMDSCCS-ALLN01 - Code Composer Studio IDE - Platinum Node Locked Single User (N01)

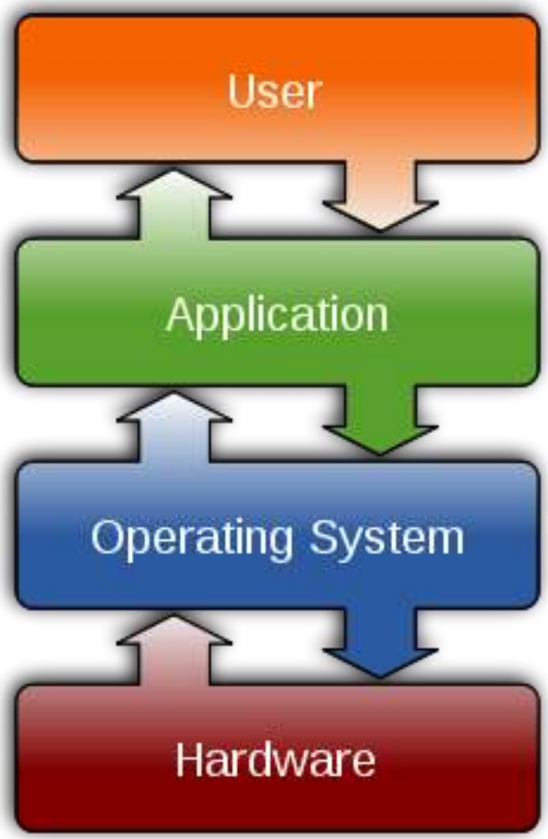
$495.00
Texas Instruments

Item	Description	Price
Platinum Eval Tools	Full tools with 30 day limit (all EMU)	FREE
Platinum Bundle	EVM, sim, XDS100 use	FREE ☺
Platinum Node Locked	Full tools tied to a machine	$495 (1)
Platinum Floating	Full tools shared across machines	$795 (1)
Microcontroller Core	MSP/C2000 code size limited	FREE
Microcontroller Node Locked	MSP/C2000	$445

☺ - recommended option: purchase Dev Kit, use XDS100v1-2, & Free CCSv5

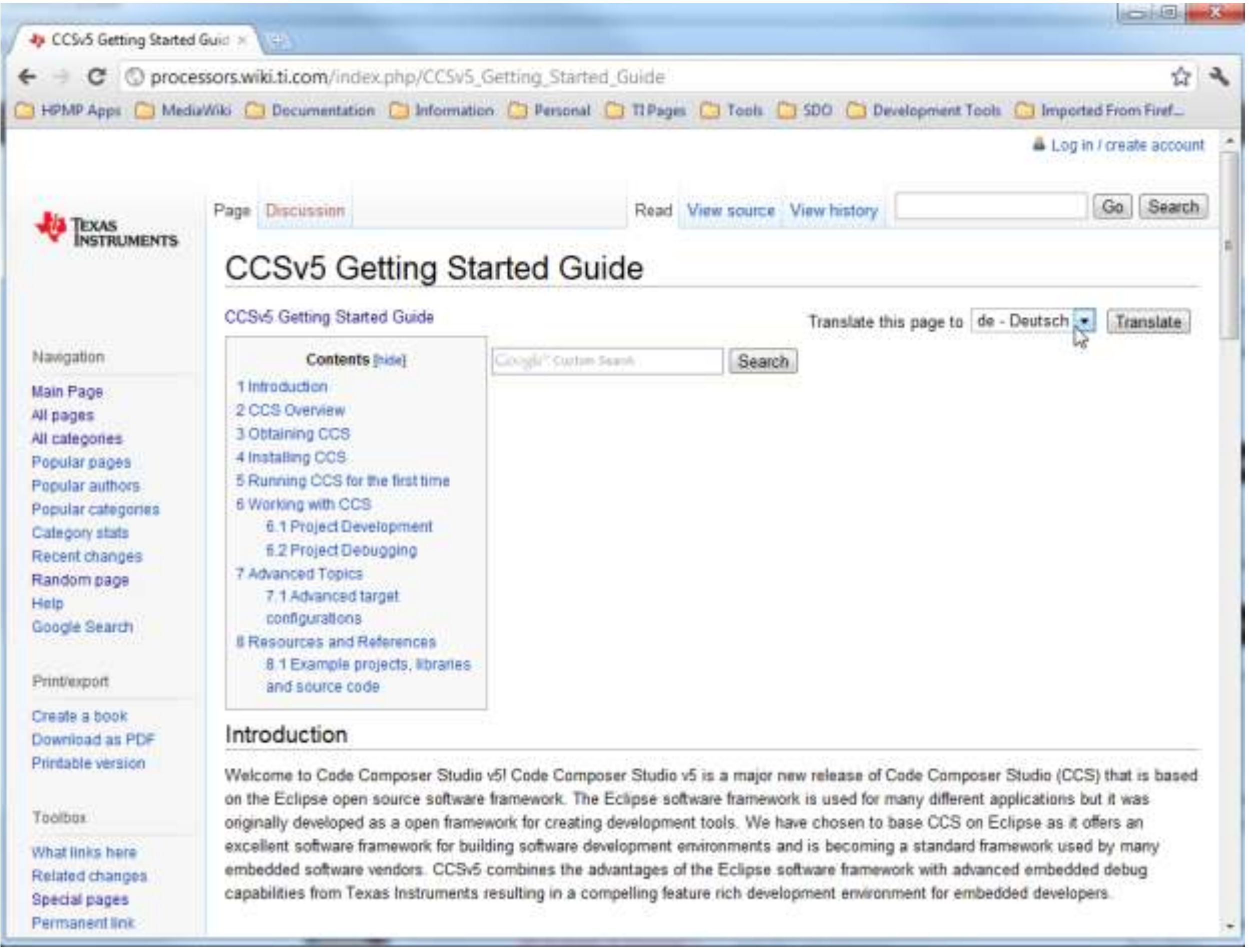
Outline

- Intro to CCSv5
  - Functional Overview
  - Perspectives
  - Projects
  - Target Configuration
  - Build Config & Options
  - Licensing/Pricing
  - CCSv5– For More Info…



CCSv5 – For More Info…

- **Links for:**
  - Downloading CCSv5
  - Installation Help
  - Licensing
  - Tutorials
  - BIOS Projects
  - Etc.



CCSv5 Getting Started Guide

CCSv5 Getting Started Guide

Translate this page to de - Deutsch [Translate]

Google Custom Search [Search]

Contents [hide]

Introduction

Welcome to Code Composer Studio v5! Code Composer Studio v5 is a major new release of Code Composer Studio (CCS) that is based on the Eclipse open source software framework. The Eclipse software framework is used for many different applications but it was originally developed as a open framework for creating development tools. We have chosen to base CCS on Eclipse as it offers an excellent software framework for building software development environments and is becoming a standard framework used by many embedded software vendors. CCSv5 combines the advantages of the Eclipse software framework with advanced embedded debug capabilities from Texas Instruments resulting in a compelling feature rich development environment for embedded developers.

Questions?