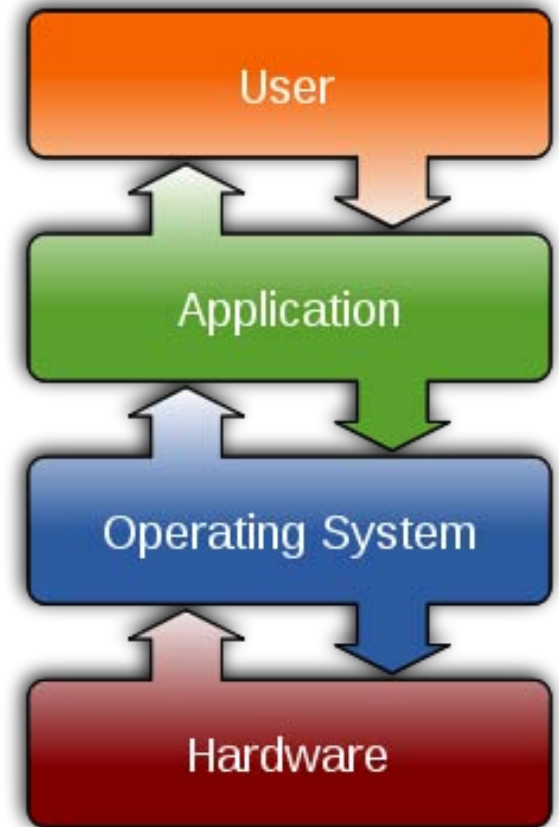


# Introduction to CCSv5

# Outline

## ◆ Intro to CCSv5

- ◆ Functional Overview
- ◆ Perspectives
- ◆ Projects
- ◆ Target Configuration
- ◆ Build Config & Options
- ◆ Licensing/Pricing
- ◆ CCSv5 – For More Info...

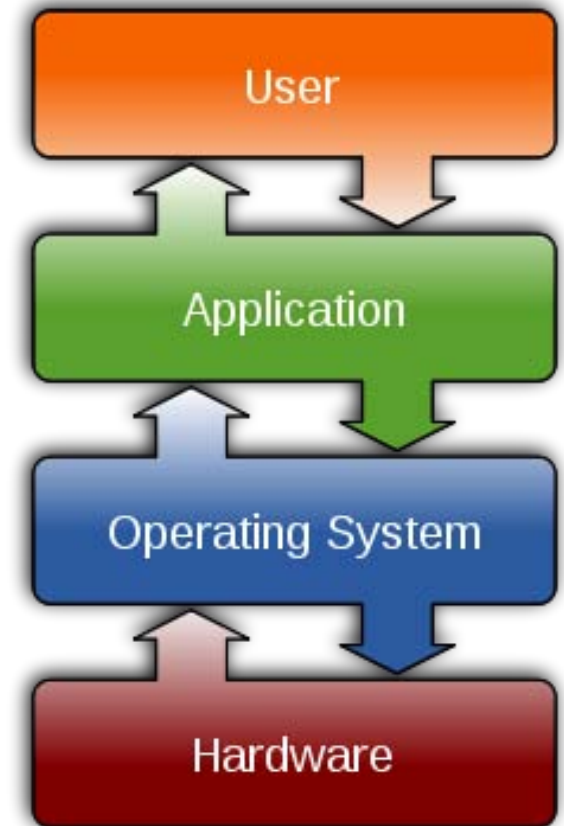


# Outline

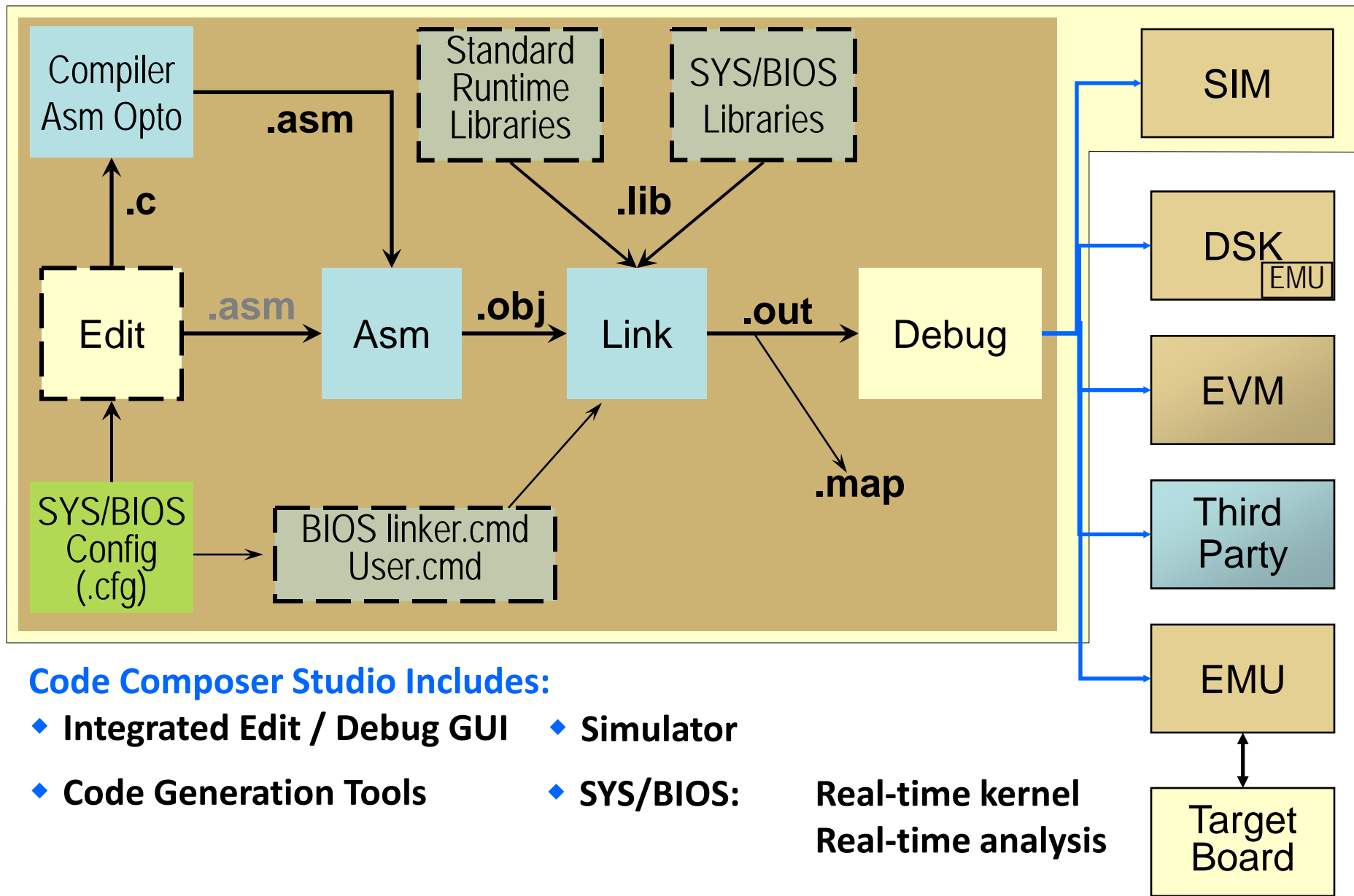
## ◆ Intro to CCSv5

### ◆ 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

The screenshot displays the Code Composer Studio (CCSv5) interface. The top menu bar includes File, Edit, Refactor, Navigate, Search, Project, Target, Tools, Profile, Run, Scope, Window, and Help. The toolbar contains various icons for file operations, debugging, and development. The main workspace is divided into several panes:

- Left Pane:** Shows the project structure for 'HelloDA830 [Debug] - DA830'. It includes a 'Device' section with 'Thread [main] (Suspended)' and a list of functions: '0 main() at main.c:4 0x118056e0' and '1 c\_int00() at boot.c:27 0x118055ac'. Below this is the 'DA830 Device Cycle Accurate Simulator/TMS320C6400'.
- Top Right Pane:** Contains 'Expressions', 'Registers', and 'Breakpoints' tabs. The 'Debug' button is highlighted.
- Bottom Left Pane:** The 'main.c' editor window shows the following code:

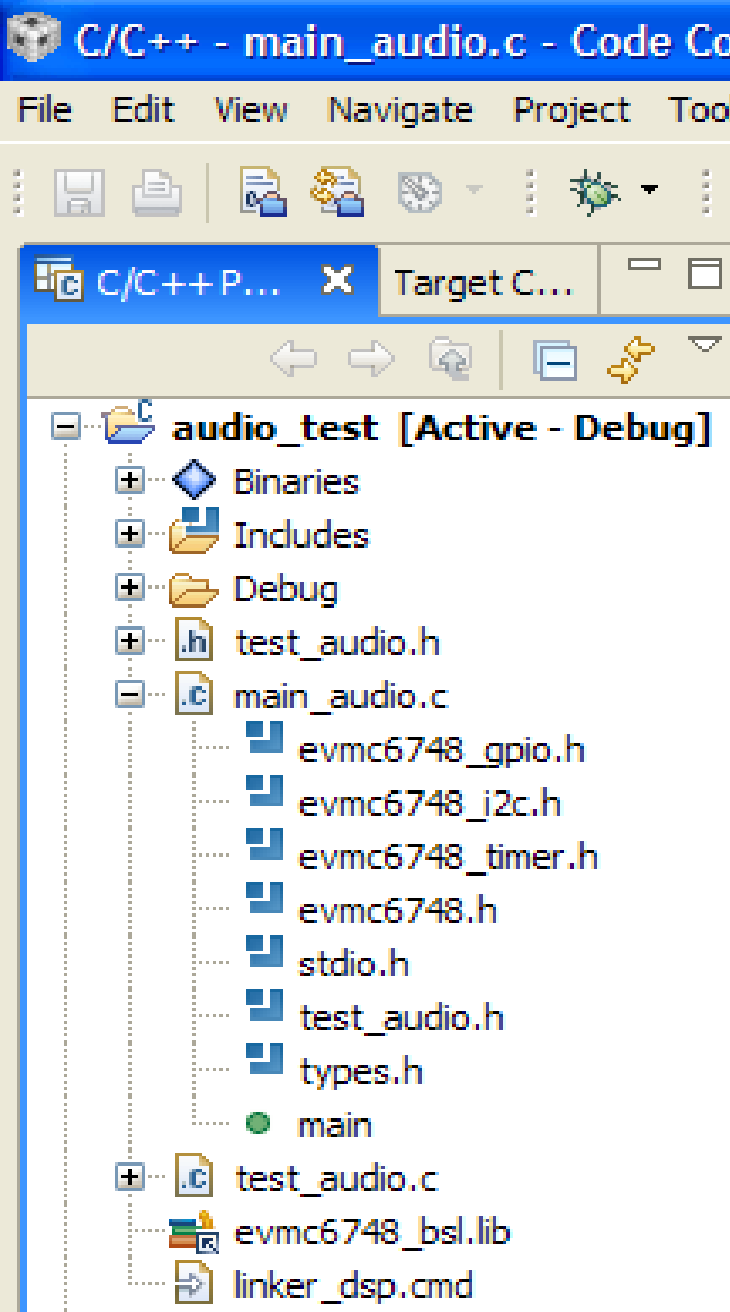
```
1#include <stdio.h>
2#include "main.h"
3
4void main(void) {
5    john(1);
6    john(0);
7}
8
9void john(int flag) {
10    if (flag == 1) {
11        printf("hello world\n");
12    }
13    else {
14        rocks();
15    }
16}
```
- Bottom Right Pane:** The 'Disassembly' and 'Memory' tabs are visible. The 'Disassembly' tab shows the following assembly code:

```
0x118056e8: 011B      CALLP.S2    john (PC+16
0x118056ea: 0626      MVK.L1     0,A4
0x118056ec: 71F7      LDW.D2T2   *++B15[2],B
0x118056ee: A1EF      BNOP.S2    B3,5
john:
0x118056f0: 01BC94F6  STW.D2T2   B3,*SP--[4]
```
- Bottom Pane:** The 'Console' and 'Scripting Console' tabs are visible. The 'Console' tab shows the output: 'HelloDA830 [Project Debug Session] DA830 Device Cycle Accurate Simulator/TMS320C6400'. The 'Scripting Console' tab shows the output: 'Initializing ..... (Completed)' and 'js:> |'.

Annotations highlight the following features:

- Customize toolbars & menus:** A yellow callout box with an arrow pointing to the toolbar.
- Perspectives contain separate window arrangements depending on what you are doing:** A yellow callout box with an arrow pointing to the 'Debug' button.
- Tabbed editor windows:** A green callout box with an arrow pointing to the 'main.c' tab in the editor.
- Tab data displays together to save space:** A green callout box with an arrow pointing to the 'Disassembly' and 'Memory' tabs.
- Fast view windows don't display until you click on them:** A yellow callout box with an arrow pointing to the 'Console' and 'Scripting Console' tabs.

# 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

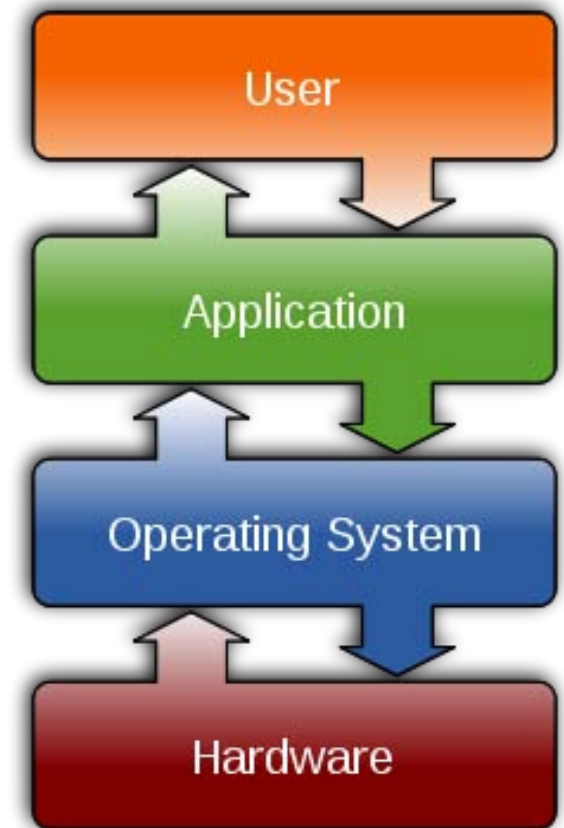
### ◆ Projects

### ◆ Target Configuration

### ◆ Build Config & Options

### ◆ Licensing/Pricing

### ◆ CCSv5 – For More Info...

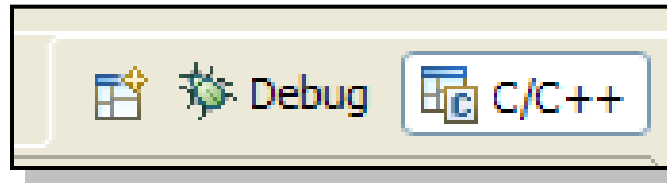


# Perspectives

- ◆ **Perspectives** – a set of windows, views and menus that correspond to a specific set of tasks
- ◆ Two **default perspectives** are provided with CCSv5:

## Debug

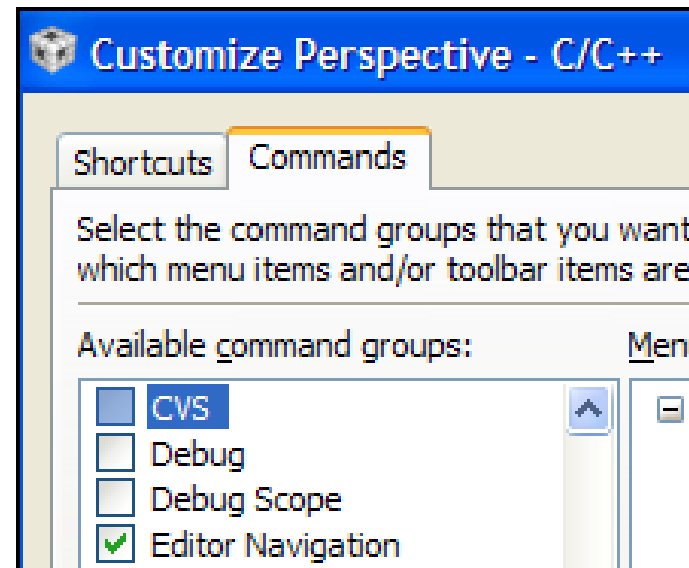
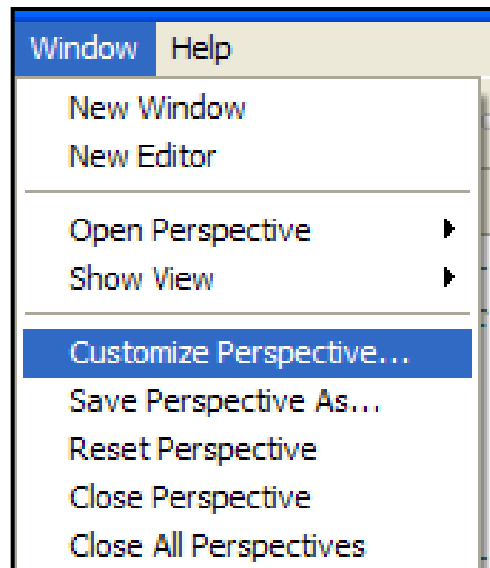
- Debug Views
- Watch/Memory
- Graphs, etc.



## C/C++

- Code Dev't Views
- Project Contents
- Editor

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





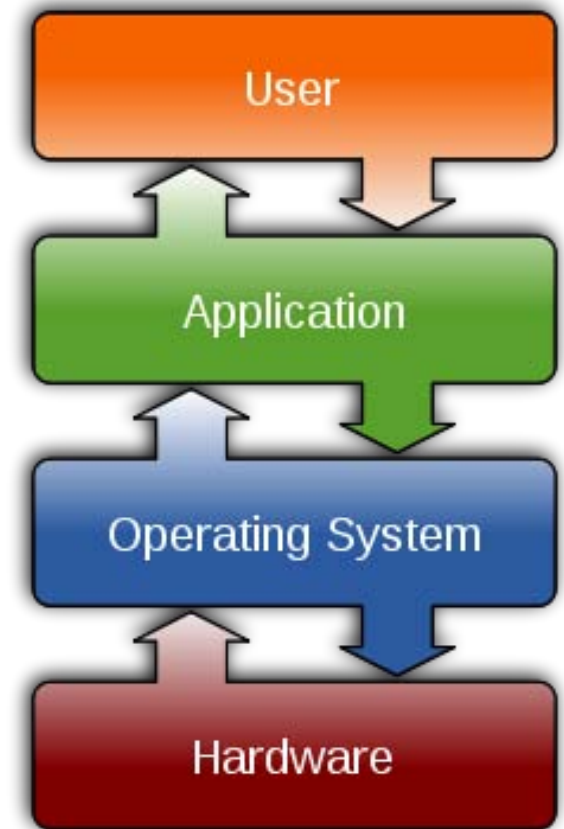
# Outline

## ◆ Intro to CCSv5

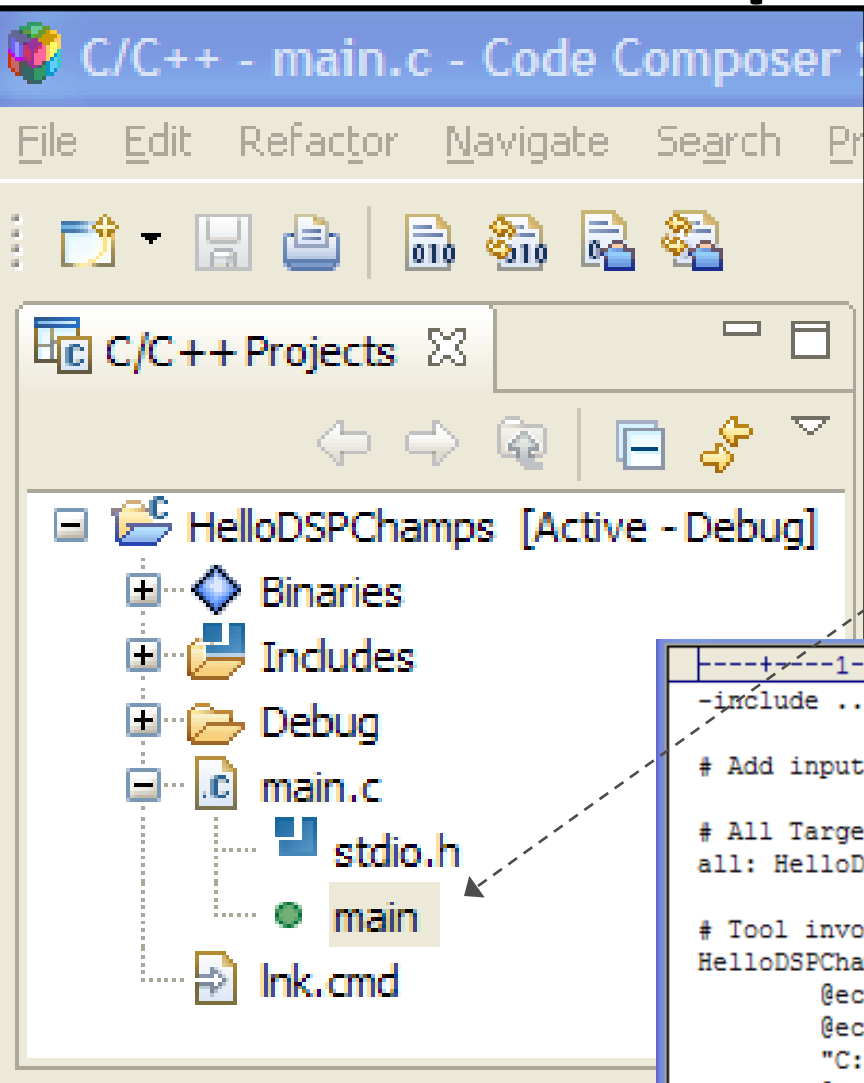
- ◆ Functional Overview
- ◆ Perspectives

### ◆ Projects

- ◆ Target Configuration
- ◆ Build Config & Options
- ◆ Licensing/Pricing
- ◆ CCSv5 – For More Info...



# Eclipse “Projects”



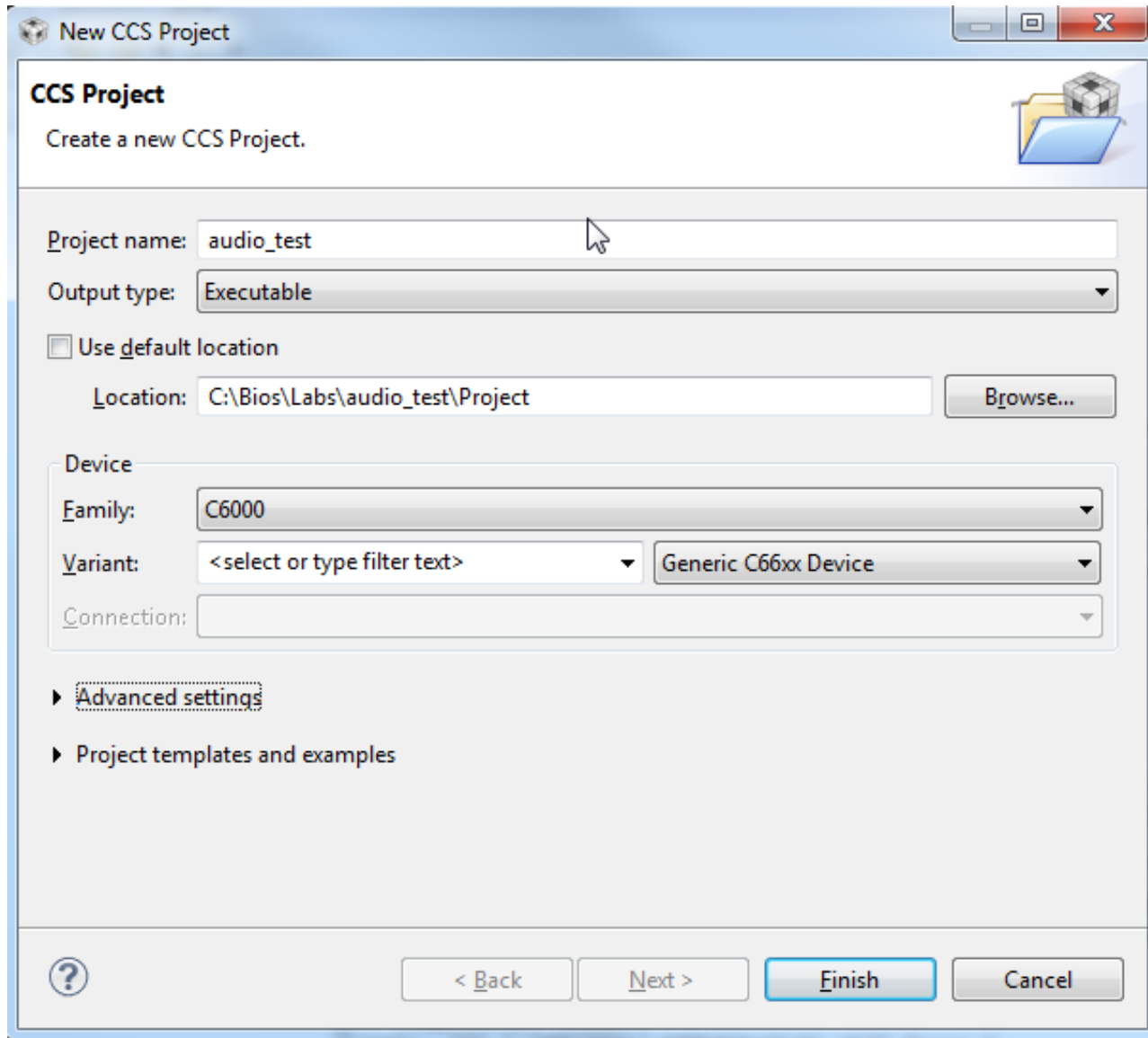
- ◆ CCSv5 is PROJECT-centric
- ◆ Eclipse uses managed makefiles as their build scripts – as opposed to *pjt* files
- ◆ Eclipse projects are folder based
  - ◆ “Adding file” copies it to folder
  - ◆ “Linking file” references original file
  - ◆ Project explorer shows folder contents
- ◆ Project explorer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

```
-----+-----1-----+-----2-----+-----3-----+-----4-----+-----5-----+-----6-----+-----7-----+-----  
-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) $(
```

How do we create  
a NEW project?

# Creating a New Project (1)

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



The screenshot shows the 'New CCS Project' dialog box. The title bar reads 'New CCS Project'. The main heading is 'CCS Project' with the instruction 'Create a new CCS Project.' and a folder icon. The 'Project name' field contains 'audio\_test'. The 'Output type' dropdown is set to 'Executable'. The 'Use default location' checkbox is unchecked. The 'Location' field shows 'C:\Bios\Labs\audio\_test\Project' with a 'Browse...' button. The 'Device' section includes 'Family' (C6000), 'Variant' (<select or type filter text>), and 'Connection' (Generic C66xx Device). At the bottom, there are expandable sections for 'Advanced settings' and 'Project templates and examples', and navigation buttons: '?', '< Back', 'Next >', 'Finish', and 'Cancel'.

**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 Browse...

Device

Family: C6000

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

Connection:

▶ Advanced settings

▶ Project templates and examples

? < Back Next > Finish Cancel

# Creating a New Project (2)

**New CCS Project**

**CCS Project**  
Create a new CCS Project.

Project name:

Output type:

☐ Use default location

Location:

Device

Family:

Variant:

Connection:

**Advanced settings**

Device endianness:

Compiler version:

Output format:

Linker command file:

Runtime support library:

Project templates and examples

# Creating a New Project (3)

**New CCS Project**

Create a new CCS Project.

Project name:

Output type:

☐ Use default location

Location:

Device

Family:

Variant:

Connection:

Advanced settings

Project templates and examples

type filter text

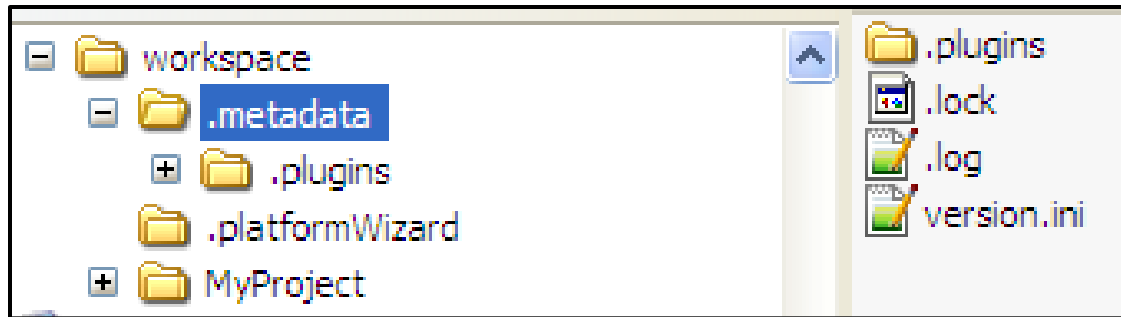
- IPC and I/O Examples
- Multicore System Analyzer (UIA)
- SYS/BIOS
  - Minimal
  - Typical
  - Typical (with separate config project)
- Generic Examples

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.

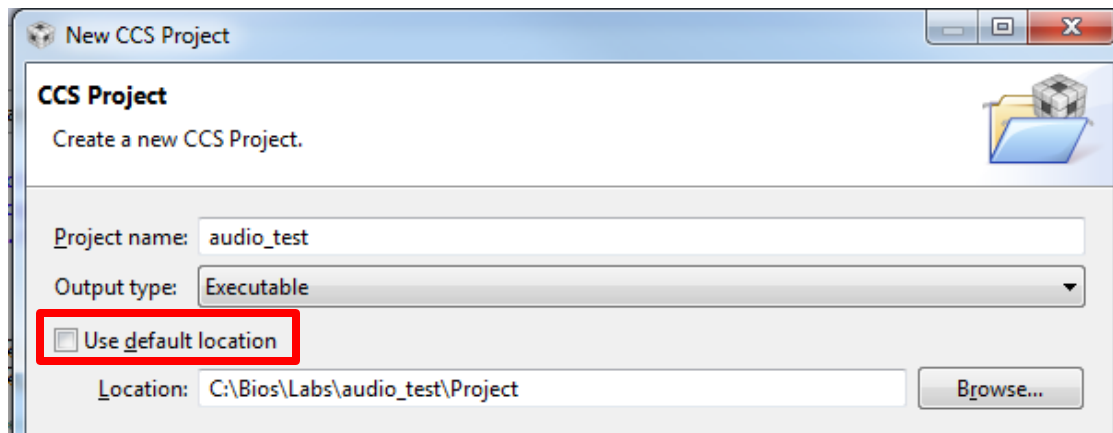
- Not using SYS/BIOS?
  - Choose “Empty Project”
- Using SYS/BIOS?
  - Choose “Minimal” under SYS/BIOS

# 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:



# Outline

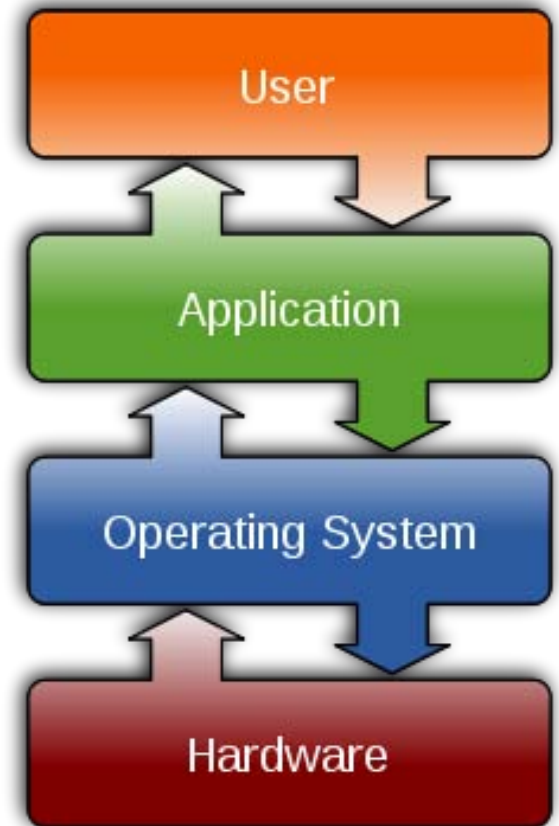
## ◆ Intro to CCSv5

- ◆ Functional Overview
- ◆ Perspectives
- ◆ Projects

## ◆ Target Configuration

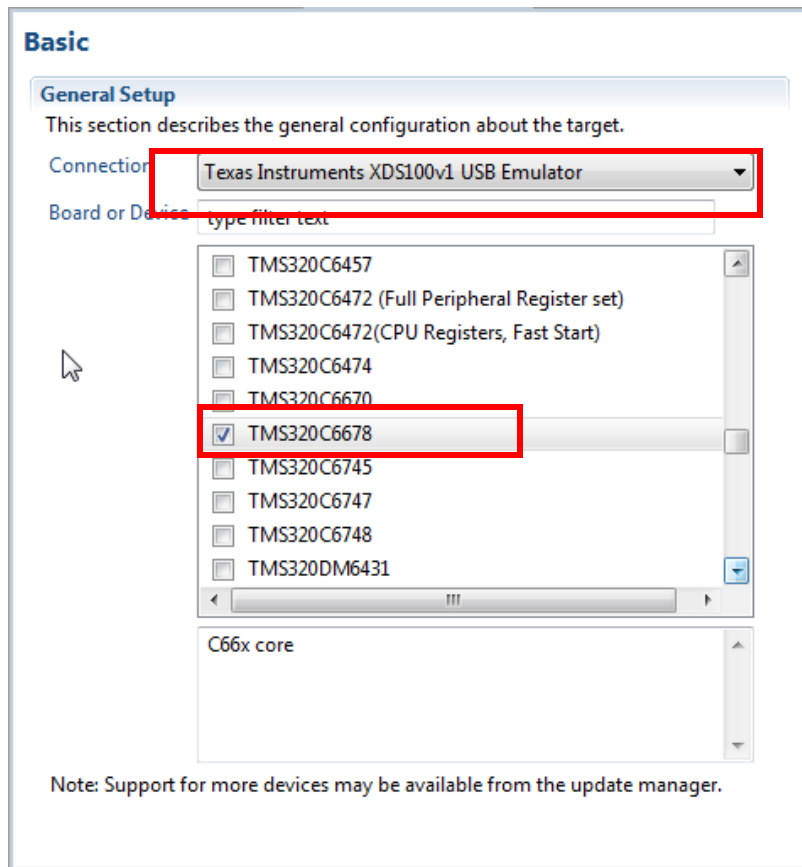
- ◆ Build Config & Options
- ◆ Licensing/Pricing
- ◆ CCSv5 – For More Info...

## ◆ Intro to SYS/BIOS



# Creating a New Target Config File (.ccxml)

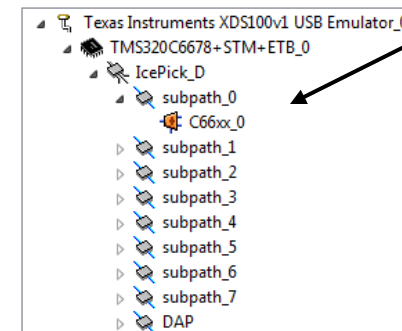
- ◆ **Target Configuration** – defines your “target” – i.e. emulator/device used, GEL scripts (replaces the old CCS Setup)
- ◆ Use on a per-project basis (add to project or create User Defined)



## Advanced Tab

### Target Configuration

#### All Connections



#### Cpu Properties

C66xx CGEM+FP CPU

Set the properties of the selected cpu.

☐ Bypass

initialization script: ..\..\emulation\boards\evmc6678\gel\evmc6678l.gel

☐ Slave Processor

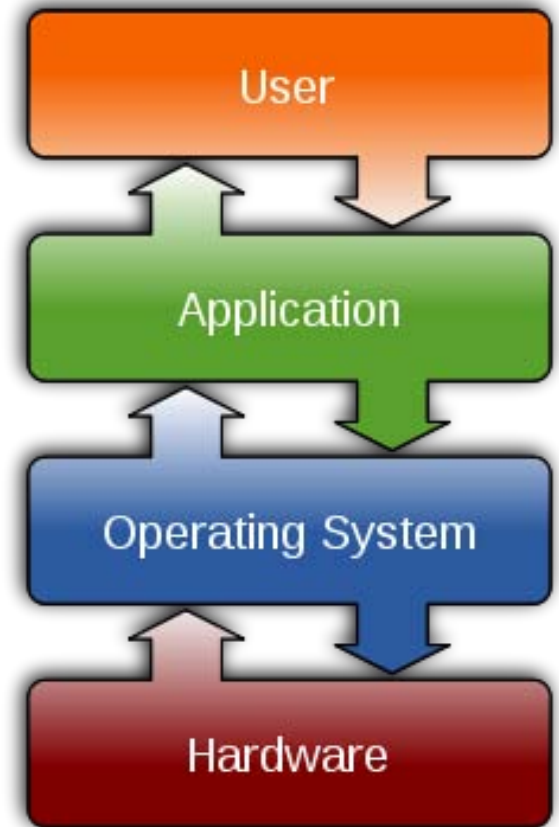
Specify GEL script here



# Outline

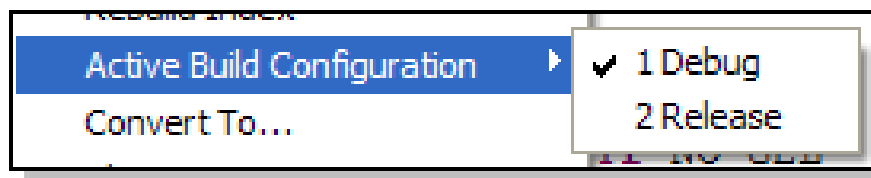
## ◆ Intro to CCSv5

- ◆ Functional Overview
- ◆ Perspectives
- ◆ Projects
- ◆ Target Configuration
- ◆ Build Config & Options
- ◆ Licensing/Pricing
- ◆ CCSv5 – For More Info...



# Two Default Build Configurations

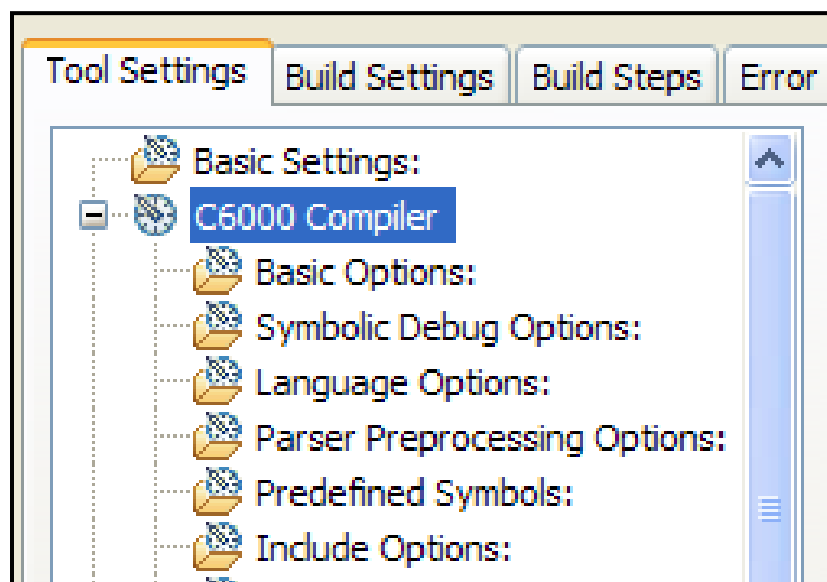
- ◆ **Build Configuration** – a set of build options for the compiler and linker (e.g. optimization levels, include DIRs, debug symbols, etc.)
- ◆ CCSv5 comes std with two DEFAULT build configs: Debug & Release:



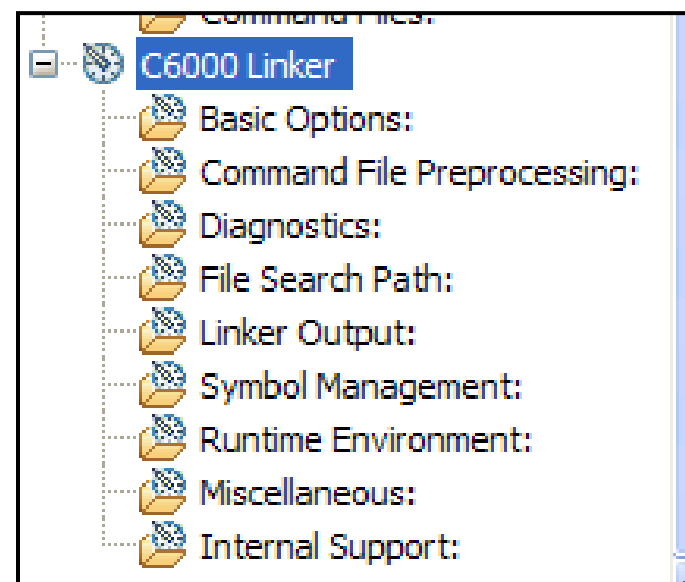
User can create their own config if desired

- ◆ User can modify compiler/linker options via “Build Properties”:

## Compiler



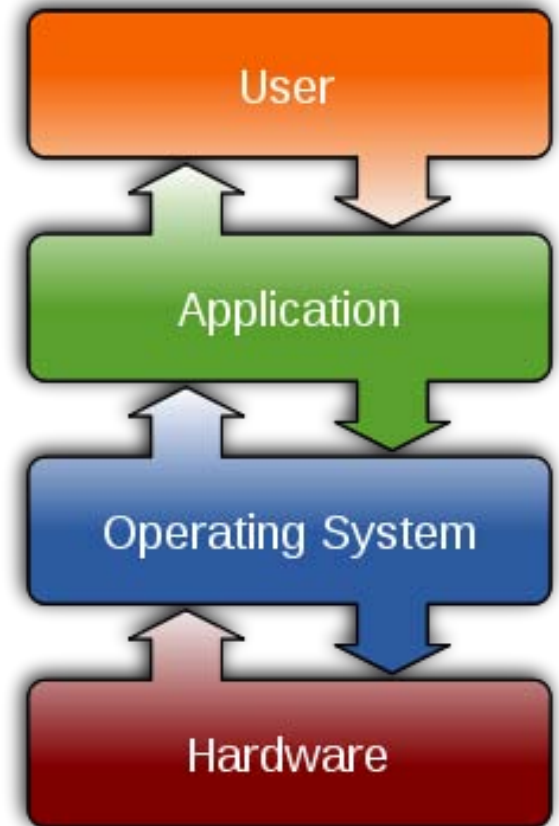
## Linker



# Outline

## ◆ Intro to CCSv5

- ◆ Functional Overview
- ◆ Perspectives
- ◆ Projects
- ◆ Target Configuration
- ◆ Build Config & Options
- ◆ Licensing/Pricing
- ◆ CCSv5 – For More Info...



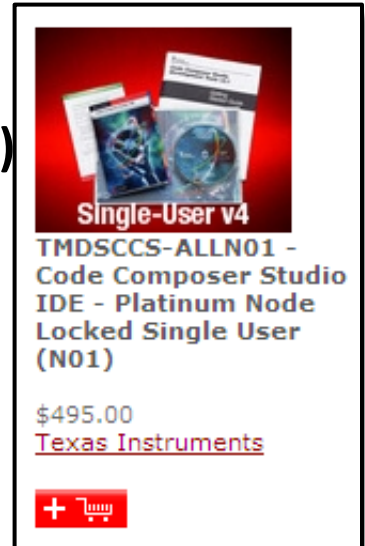
# CCSv5 Licensing & Pricing

## ◆ 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



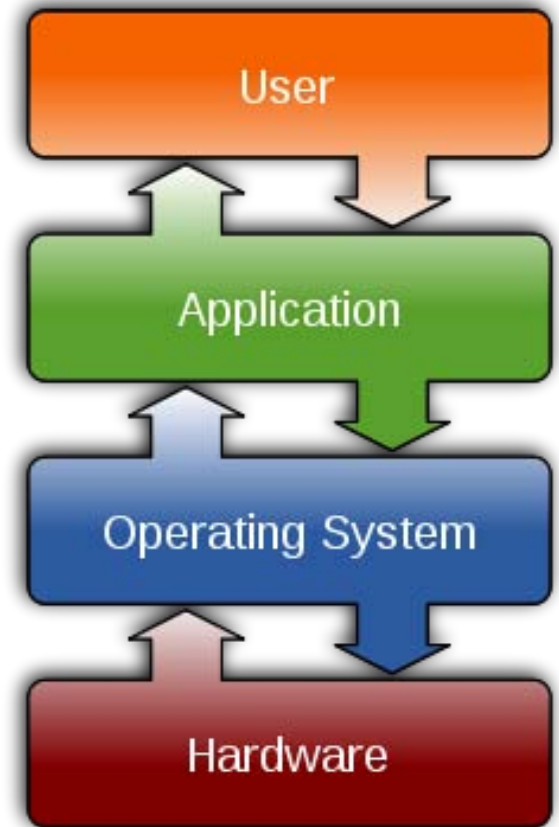
Item	Description	Price
Platinum Eval Tools	Full tools with 30 day limit (all EMU)	FREE
Platinum Bundle	EVM, sim, XDS100 use	FREE ☺
Platinum Node Locked	Full tools tied to a machine	\$495 (1)
Platinum Floating	Full tools shared across machines	\$795 (1)
Microcontroller Core	MSP/C2000 code size limited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

## Links for:

- Downloading CCSv5
- Installation Help
- Licensing
- Tutorials
- BIOS Projects
- ETC.



# Questions?