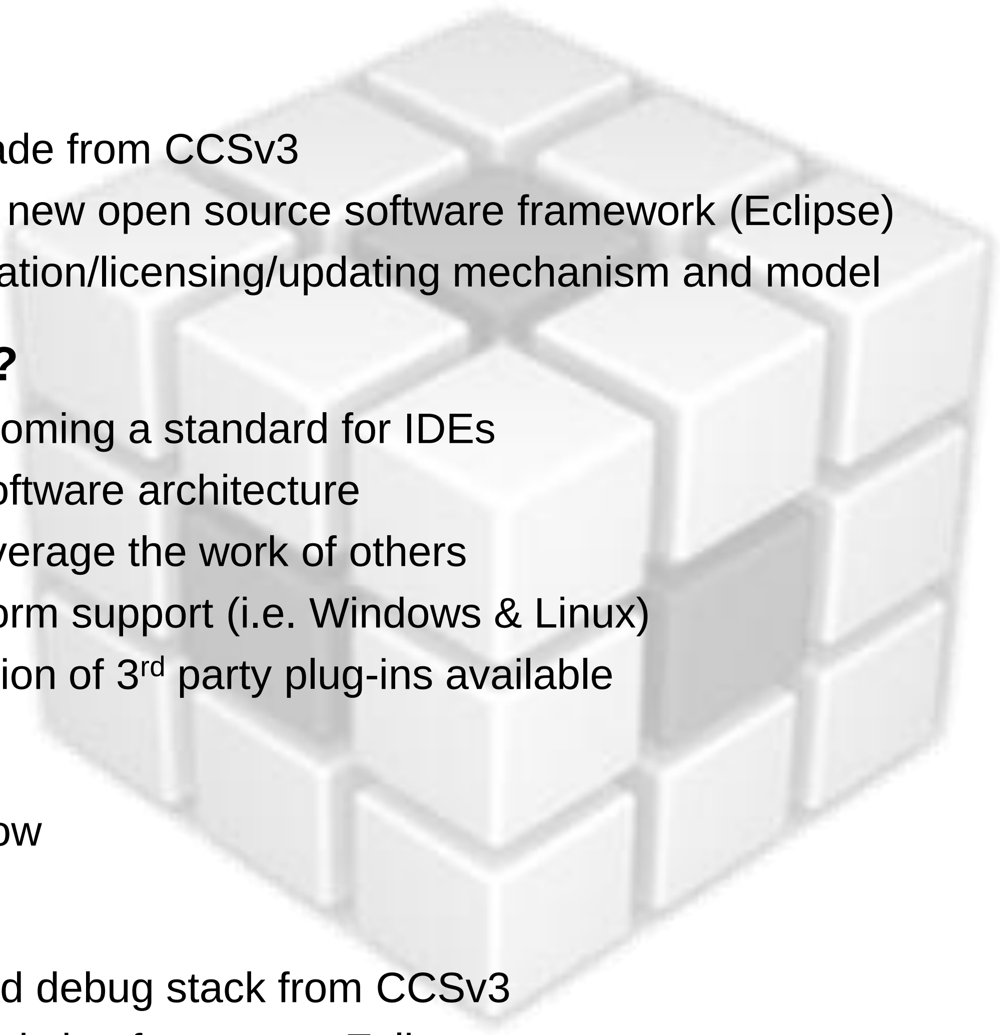
Code Composer Studio v5

Moving from CCSv3

Summary

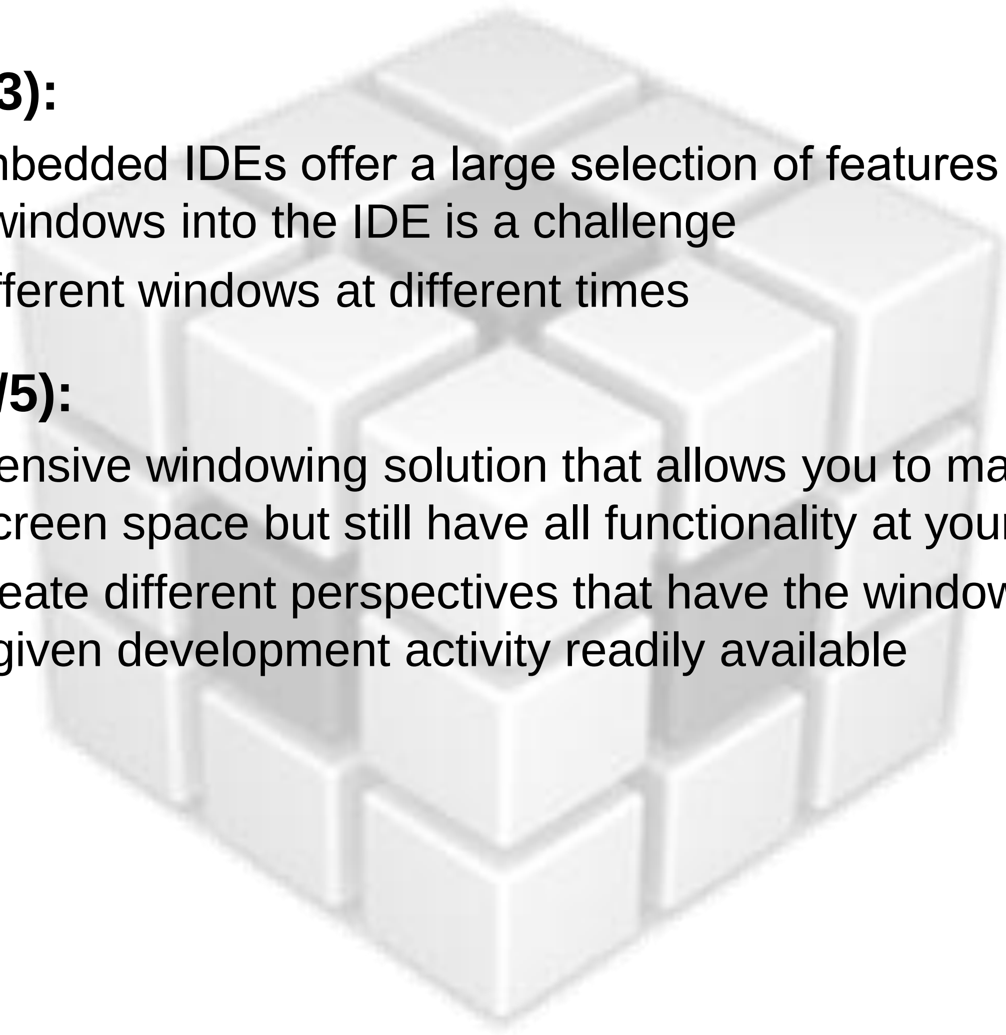


- **<u>What</u> is it?**
  - Major upgrade from CCSv3
  - Based on a new open source software framework (Eclipse)
  - New registration/licensing/updating mechanism and model
- **<u>Why</u> Eclipse?**
  - Quickly becoming a standard for IDEs
  - Excellent software architecture
  - Ability to leverage the work of others
  - Cross-platform support (i.e. Windows & Linux)
  - Wide selection of 3rd party plug-ins available
- **<u>When</u>?**
  - Available now
- **<u>How</u>?**
  - Restructured debug stack from CCSv3
  - Porting of existing features to Eclipse

Code Composer Studio v5

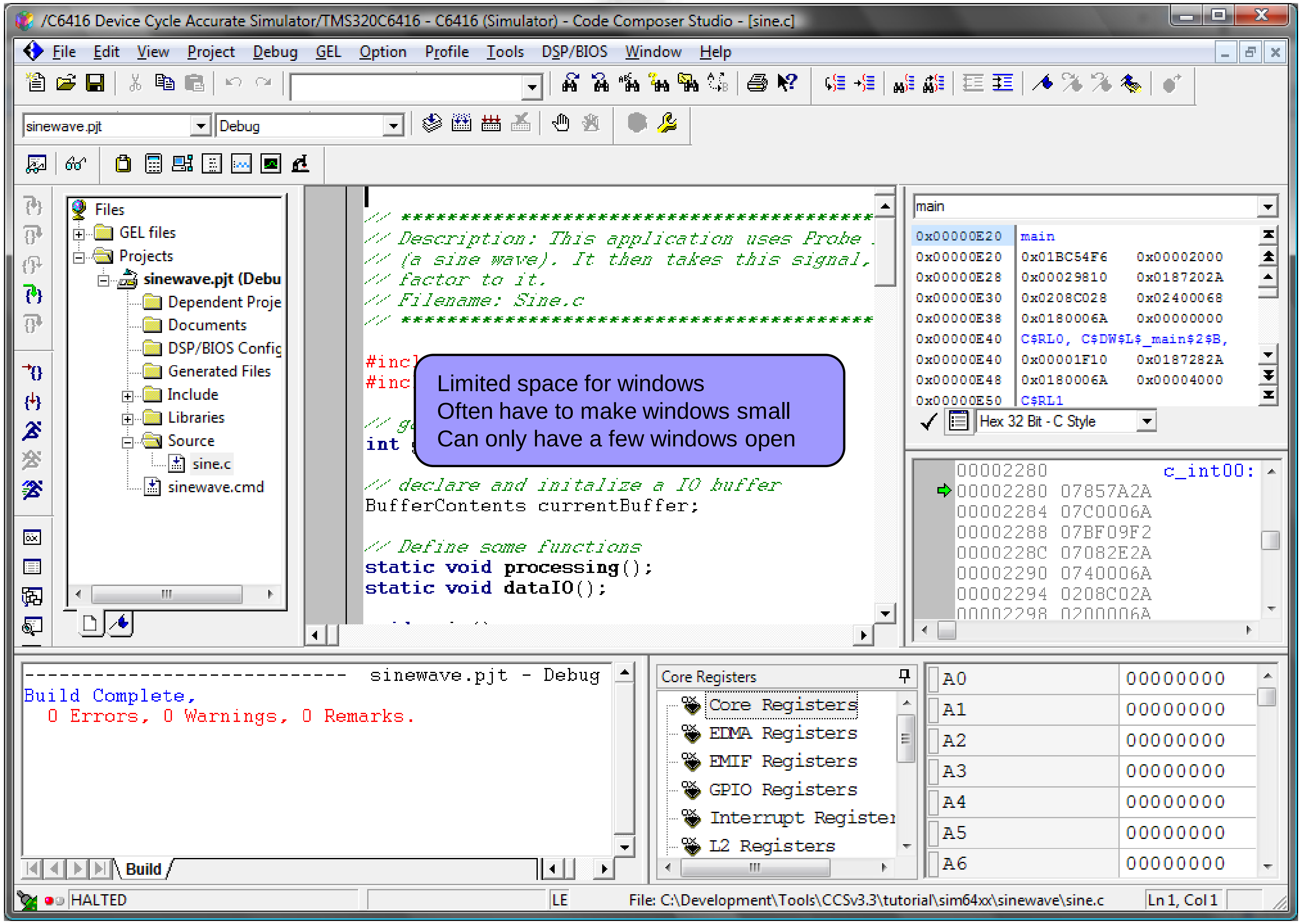
Key Benefits form CCSv3

Windowing Environment



- **Problems (v3):**
  - Today’s embedded IDEs offer a large selection of features however fitting all of your windows into the IDE is a challenge
  - You use different windows at different times
- **Solution (v4/5):**
  - A comprehensive windowing solution that allows you to maximize the available screen space but still have all functionality at your finger tips
  - Ability to create different perspectives that have the windows that you use most for a given development activity readily available

CCSv3.3 Environment



/C6416 Device Cycle Accurate Simulator/TMS320C6416 - C6416 (Simulator) - Code Composer Studio - [sine.c]

```
// ******************************************
// Description: This application uses Probe
// (a sine wave). It then takes this signal,
// factor to it.
// Filename: Sine.c
// ******************************************

#incl
#inc

// g
int

// declare and initalize a IO buffer
BufferContents currentBuffer;

// Define some functions
static void processing();
static void dataIO();
```

Limited space for windows
Often have to make windows small
Can only have a few windows open

```
0x00000E20  main
0x00000E20  0x01BC54F6  0x00002000
0x00000E28  0x00029810  0x0187202A
0x00000E30  0x0208C028  0x02400068
0x00000E38  0x0180006A  0x00000000
0x00000E40  C$RL0, C$DW$L$_main$2$B,
0x00000E40  0x00001F10  0x0187282A
0x00000E48  0x0180006A  0x00004000
0x00000E50  C$RL1
```

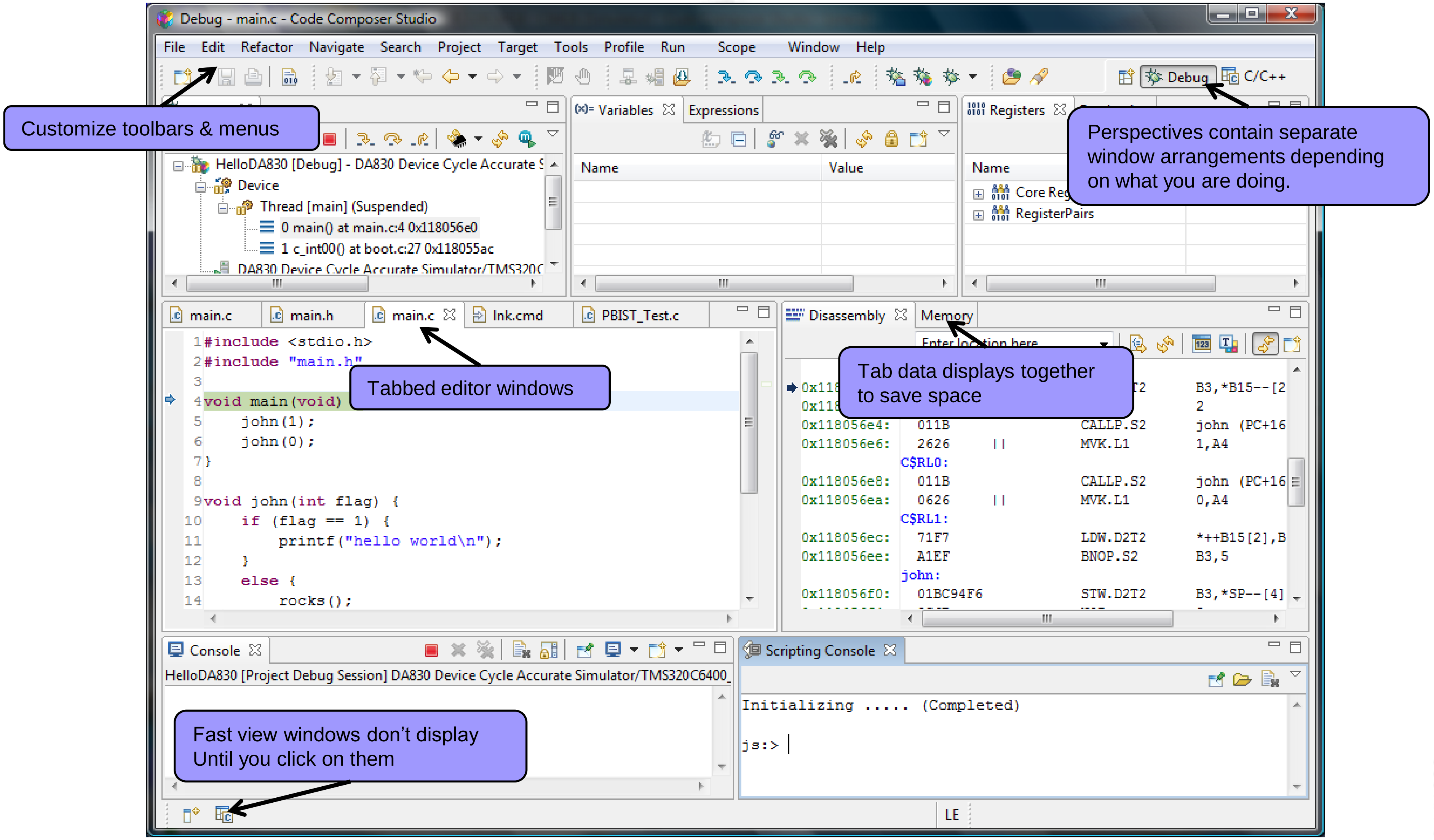
```
00002280           c_int00:
00002280 07857A2A
00002284 07C0006A
00002288 07BF09F2
0000228C 07082E2A
00002290 0740006A
00002294 0208C02A
00002298 0200006A
```

```
--------------------------- sinewave.pjt - Debug
Build Complete,
  0 Errors, 0 Warnings, 0 Remarks.
```

A0	00000000
A1	00000000
A2	00000000
A3	00000000
A4	00000000
A5	00000000
A6	00000000

HALTED | LE | File: C:\Development\Tools\CCSv3.3\tutorial\sim64xx\sinewave\sine.c | Ln 1, Col 1

CCSv4/5 Environment


Debug - main.c - Code Composer Studio
File Edit Refactor Navigate Search Project Target Tools Profile Run Scope Window Help
Debug
C/C++
Variables
Expressions
Name
Value
Registers
Core Reg
RegisterPairs
HelloDA830 [Debug] - DA830 Device Cycle Accurate S
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
main.c
main.h
main.c
lnk.cmd
PBIST_Test.c
Disassembly
Memory
Enter location here
Console
Scripting Console
LE

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

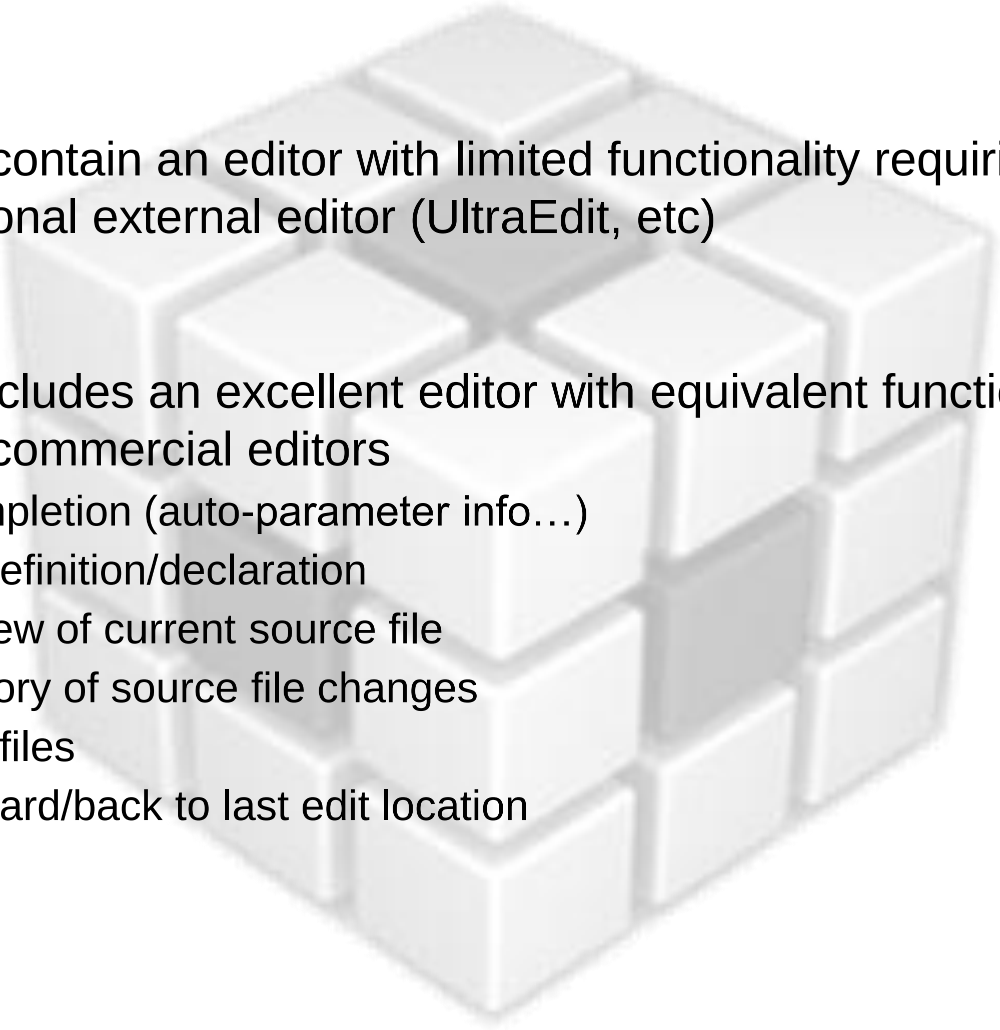
```
#include <stdio.h>
#include "main.h"

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

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

```
0x118                              2          B3,*B15--[2
0x118                                         2
0x118056e4:   011B          CALLP.S2          john (PC+16
0x118056e6:   2626     ||   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_
```

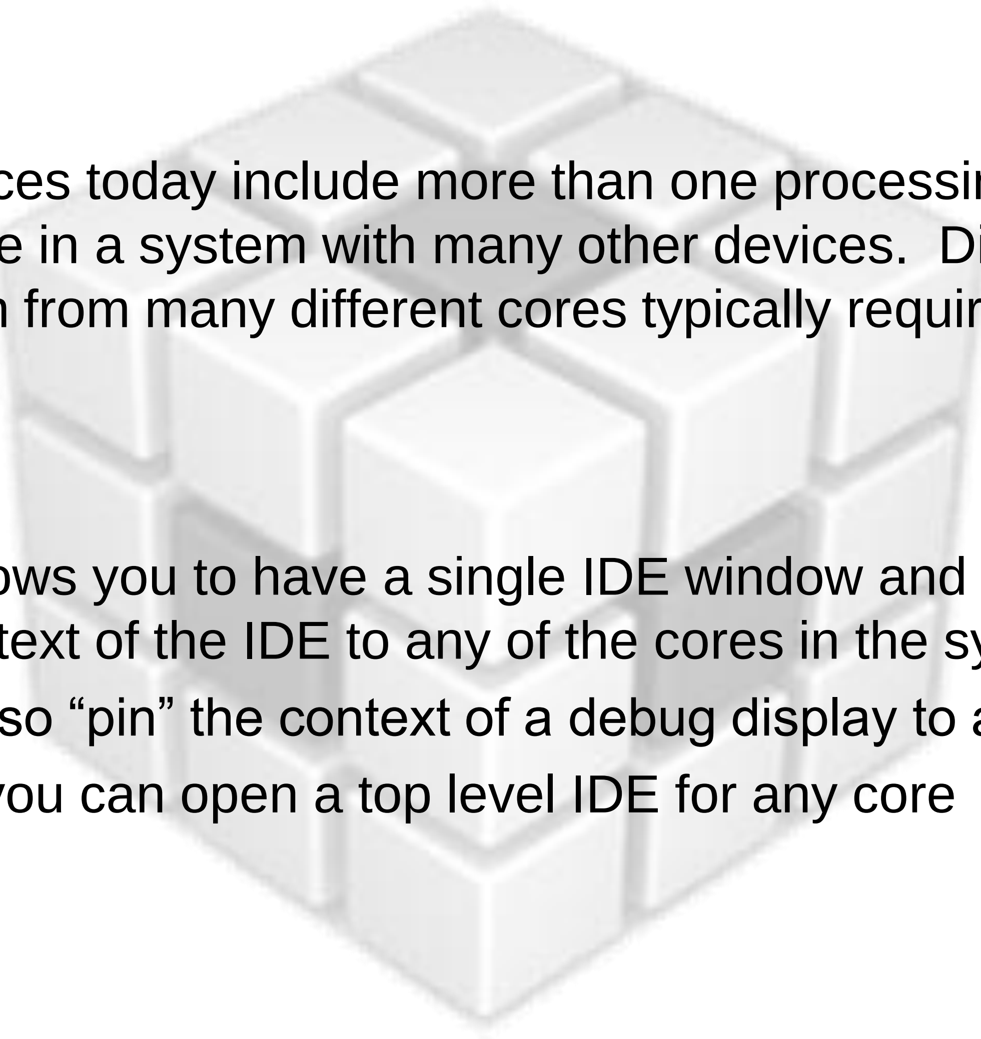
```
Initializing .....  (Completed)

js:>
```

Source Code Editor



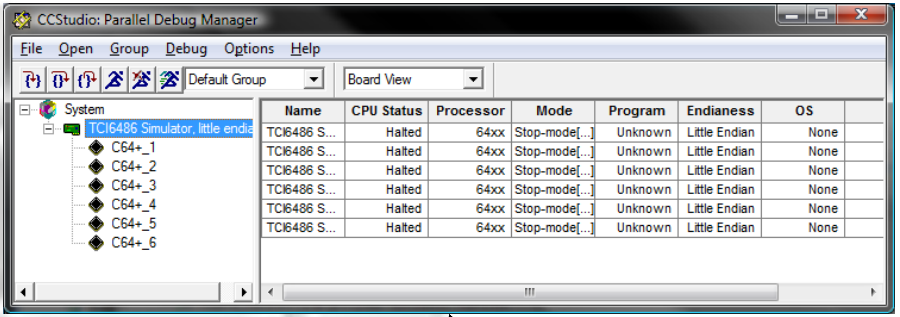
- **Problem:**
  - Most IDEs contain an editor with limited functionality requiring the purchase of an additional external editor (UltraEdit, etc)
- **Solution:**
  - CCSv4/5 includes an excellent editor with equivalent functionality to the majority of commercial editors
    - Code completion (auto-parameter info…)
    - Jump to definition/declaration
    - Outline view of current source file
    - Local history of source file changes
    - Compare files
    - Back/forward/back to last edit location
    - …

Multi-processor Environment



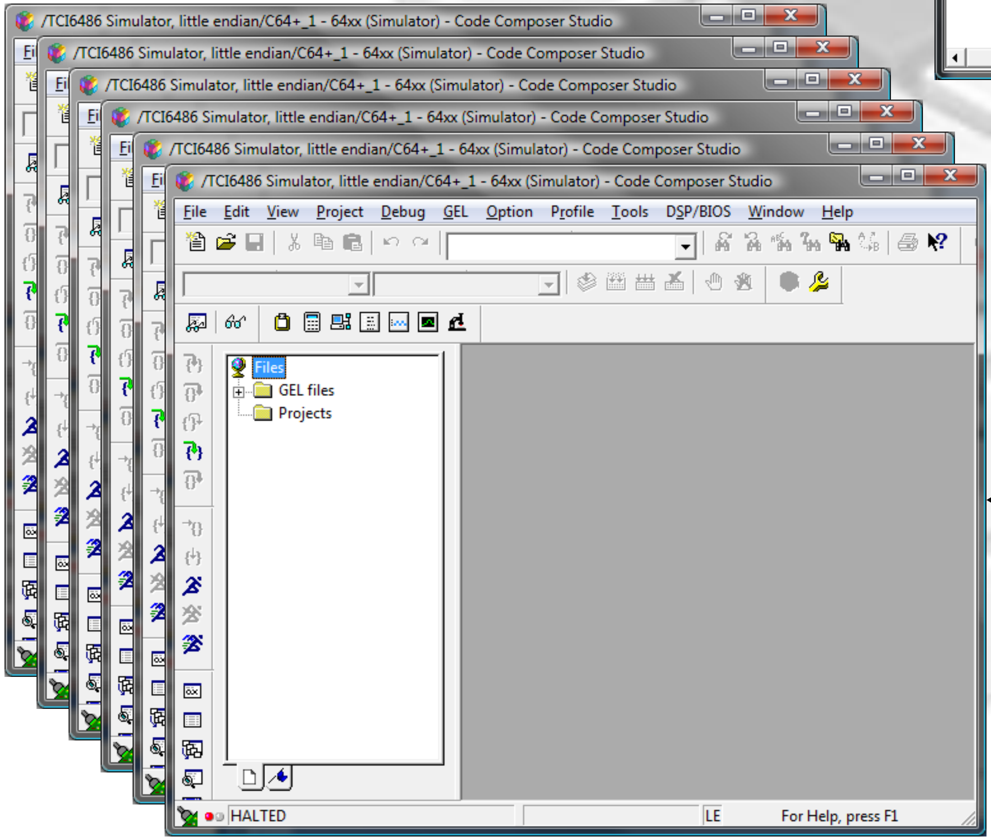
- **Problem:**
  - Many devices today include more than one processing core and often reside in a system with many other devices. Displaying debug information from many different cores typically requires many IDE windows.
- **Solution:**
  - CCSv4 allows you to have a single IDE window and to change the debug context of the IDE to any of the cores in the system.
  - You can also “pin” the context of a debug display to a specific core.
  - If desired you can open a top level IDE for any core

CCSv3.3 Multi-core Environment


CCStudio: Parallel Debug Manager
File Open Group Debug Options Help
Default Group
Board View
System
TCI6486 Simulator, little endia
C64+_1
C64+_2
C64+_3
C64+_4
C64+_5
C64+_6

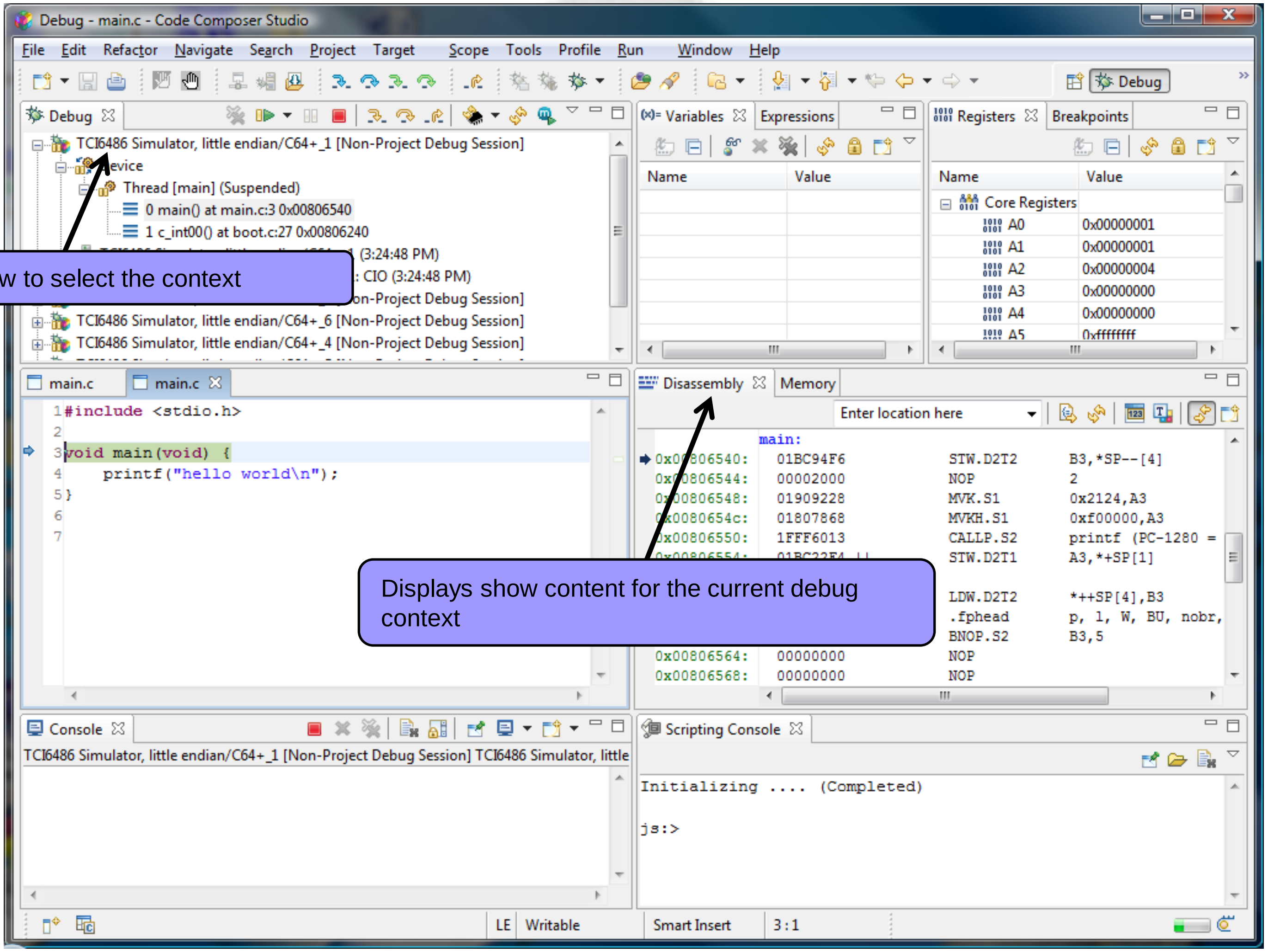
Name	CPU Status	Processor	Mode	Program	Endianess	OS
TCI6486 S...	Halted	64xx	Stop-mode[...]	Unknown	Little Endian	None
TCI6486 S...	Halted	64xx	Stop-mode[...]	Unknown	Little Endian	None
TCI6486 S...	Halted	64xx	Stop-mode[...]	Unknown	Little Endian	None
TCI6486 S...	Halted	64xx	Stop-mode[...]	Unknown	Little Endian	None
TCI6486 S...	Halted	64xx	Stop-mode[...]	Unknown	Little Endian	None
TCI6486 S...	Halted	64xx	Stop-mode[...]	Unknown	Little Endian	None

Parallel debug manager to see status of all cores


/TCI6486 Simulator, little endian/C64+_1 - 64xx (Simulator) - Code Composer Studio
File Edit View Project Debug GEL Option Profile Tools DSP/BIOS Window Help
Files
GEL files
Projects
HALTED
LE
For Help, press F1

Separate top level IDE windows for each core
Can actually run out of windows resources

CCSv4/5 Multi-core Environment


Debug - main.c - Code Composer Studio
File Edit Refactor Navigate Search Project Target Scope Tools Profile Run Window Help
Debug
Variables Expressions
Registers Breakpoints
Disassembly Memory
Enter location here
Console
Scripting Console
LE Writable Smart Insert 3:1

```
TCI6486 Simulator, little endian/C64+_1 [Non-Project Debug Session]
  evice
    Thread [main] (Suspended)
      0 main() at main.c:3 0x00806540
      1 c_int00() at boot.c:27 0x00806240
  (3:24:48 PM)
  : CIO (3:24:48 PM)
  on-Project Debug Session]
TCI6486 Simulator, little endian/C64+_6 [Non-Project Debug Session]
TCI6486 Simulator, little endian/C64+_4 [Non-Project Debug Session]
```

Use the Debug view to select the context

Name	Value
Core Registers	
A0	0x00000001
A1	0x00000001
A2	0x00000004
A3	0x00000000
A4	0x00000000
A5	0xffffffff

```c
#include <stdio.h>

void main(void) {
    printf("hello world\n");
}
```

```
main:
0x00806540:   01BC94F6      STW.D2T2    B3,*SP--[4]
0x00806544:   00002000      NOP         2
0x00806548:   01909228      MVK.S1      0x2124,A3
0x0080654c:   01807868      MVKH.S1     0xf00000,A3
0x00806550:   1FFF6013      CALLP.S2    printf (PC-1280 =
0x00806554:   01BC22F4 ||   STW.D2T1    A3,*+SP[1]
                            LDW.D2T2    *++SP[4],B3
                            .fphead     p, l, W, BU, nobr,
                            BNOP.S2     B3,5
0x00806564:   00000000      NOP
0x00806568:   00000000      NOP
```

Displays show content for the current debug context

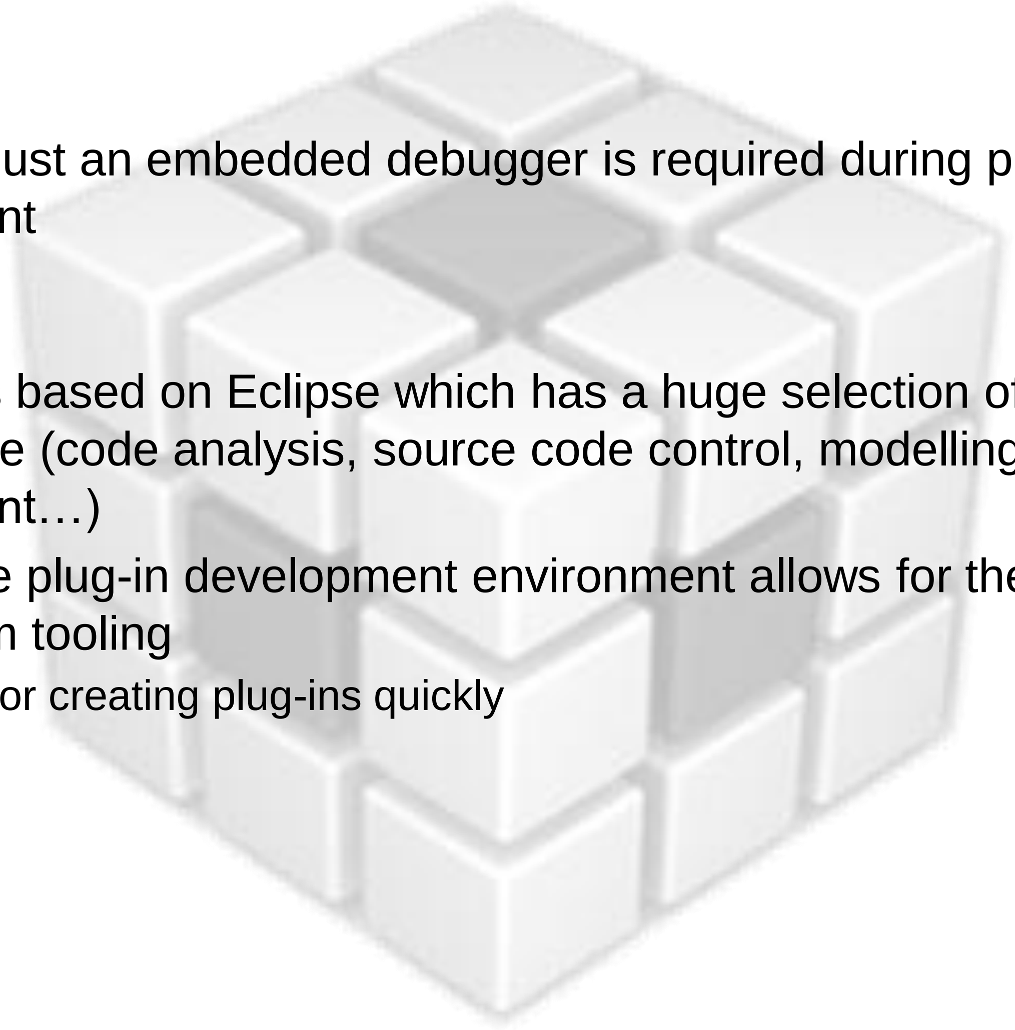
```
TCI6486 Simulator, little endian/C64+_1 [Non-Project Debug Session] TCI6486 Simulator, little
```

```
Initializing .... (Completed)

js:>
```

Project Management

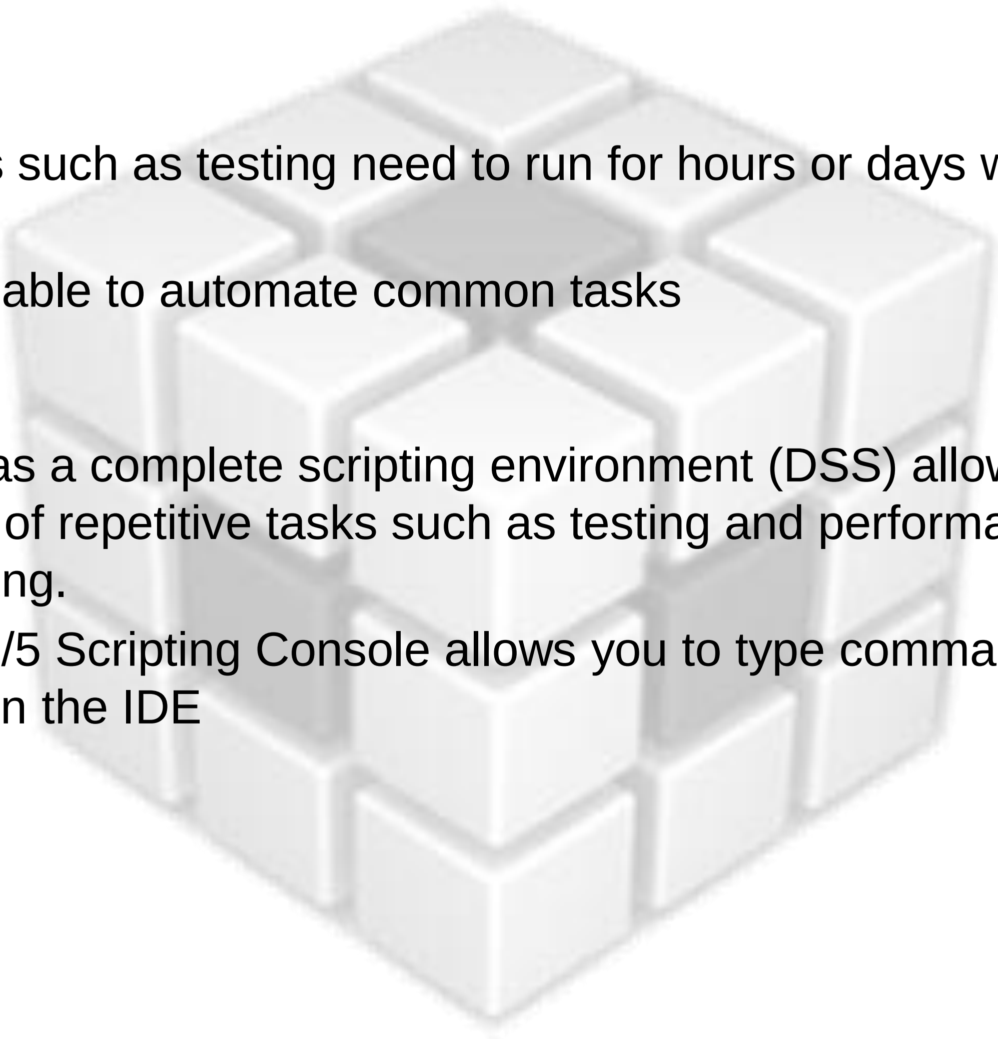
- **Problem:**
  - Typically you have more than one project on going at a time, with each project being at a different stage in development and often using different versions of compile tools, operating systems, and even target configurations
- **Solution:**
  - CCSv4/5 allows you to set the version of the compiler and DSP/BIOS that each individual project will use, allowing projects in maintenance mode to continue to use the tools they were deployed with and enabling new projects to use the latest high performance tools
  - CCSv4/5 allows you to associate a specific target configuration on a per project basis by allowing a project to have a target configuration file

Tool Integration & Customization



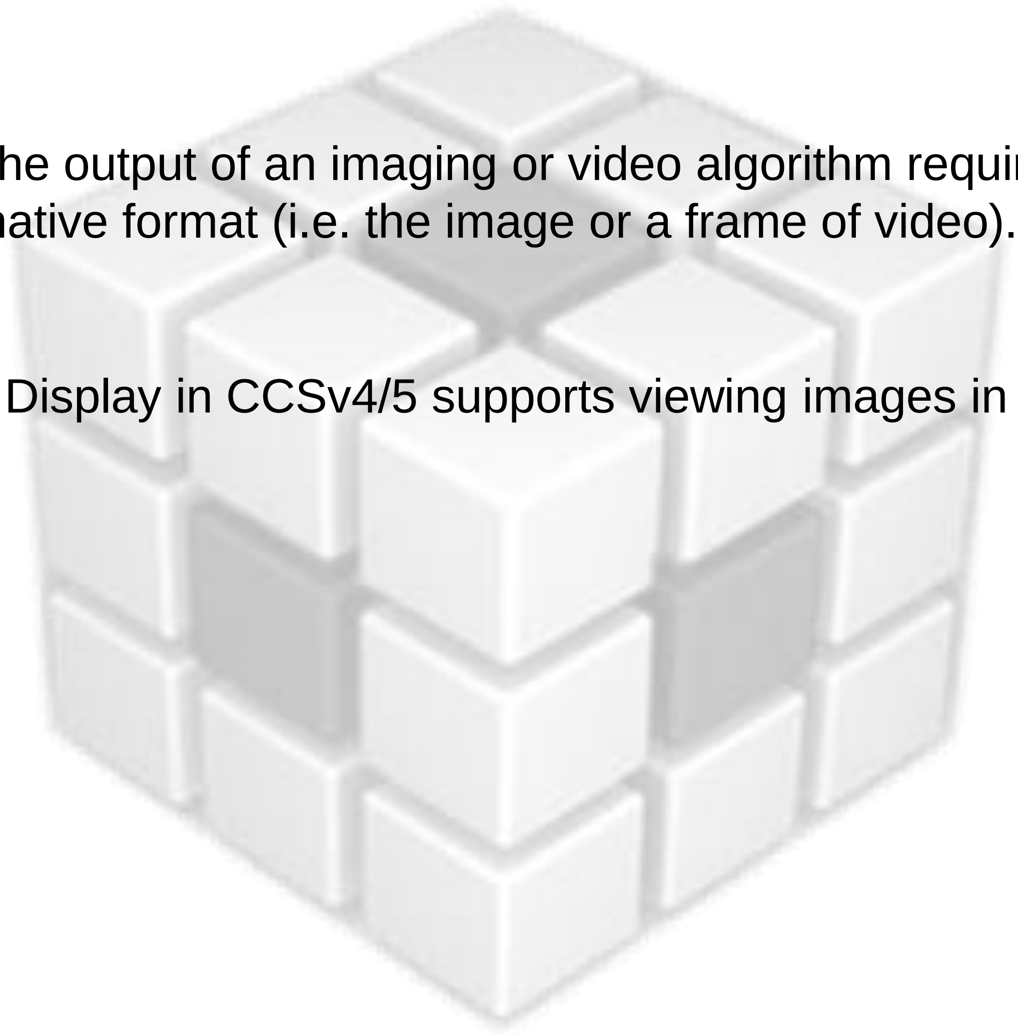
- **Problem:**
  - More than just an embedded debugger is required during product development
- **Solution:**
  - CCSv4/5 is based on Eclipse which has a huge selection of 3rd party plug-ins available (code analysis, source code control, modelling, Perl development…)
  - The Eclipse plug-in development environment allows for the creation of your own custom tooling
    - Wizards for creating plug-ins quickly

Scripting



- **Problem:**
  - Some tasks such as testing need to run for hours or days without user interaction
  - Need to be able to automate common tasks
- **Solution:**
  - CCSv4/5 has a complete scripting environment (DSS) allowing for the automation of repetitive tasks such as testing and performance benchmarking.
  - The CCSv4/5 Scripting Console allows you to type commands or to execute scripts within the IDE

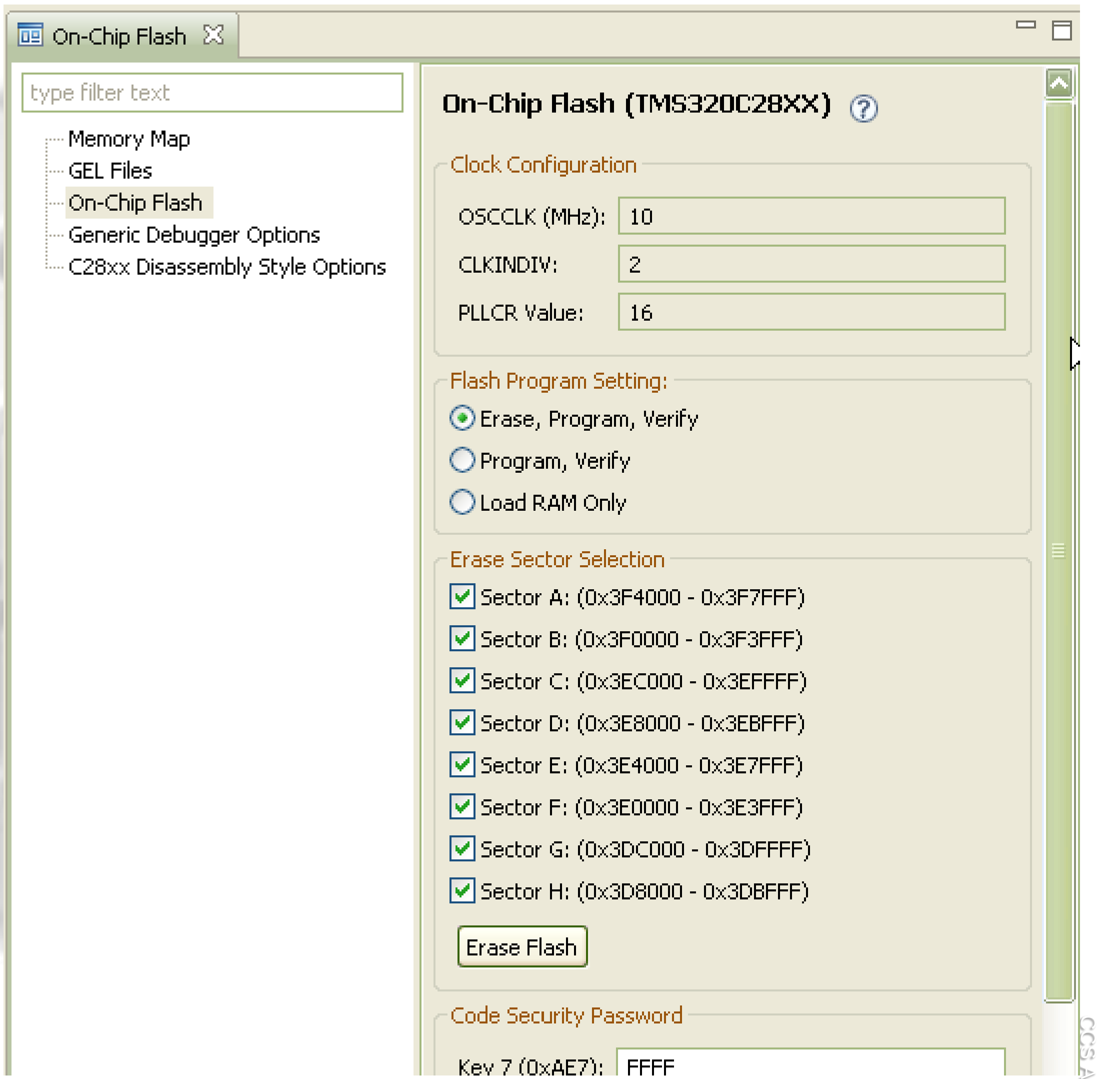
Image Analysis



- **Problem:**
  - Analyzing the output of an imaging or video algorithm requires looking at the data in its native format (i.e. the image or a frame of video).
- **Solution:**
  - The Image Display in CCSv4/5 supports viewing images in many different formats.

On-Chip Flash Programming

- **Problem:**
  - A separate flash programming utility is needed to write/load the program to flash
- **Solution:**
  - Flash programmer is tightly integrated with CCSv4/5
    - CCS will automatically write to flash when loading a program if the load address is in flash
    - No need to work with a separate utility to erase/program flash



On-Chip Flash (TMS320C28XX)

Clock Configuration

OSCCLK (MHz):	10
CLKINDIV:	2
PLLCR Value:	16

Flash Program Setting:

[✓ Erase, Program, Verify] Program, Verify Load RAM Only

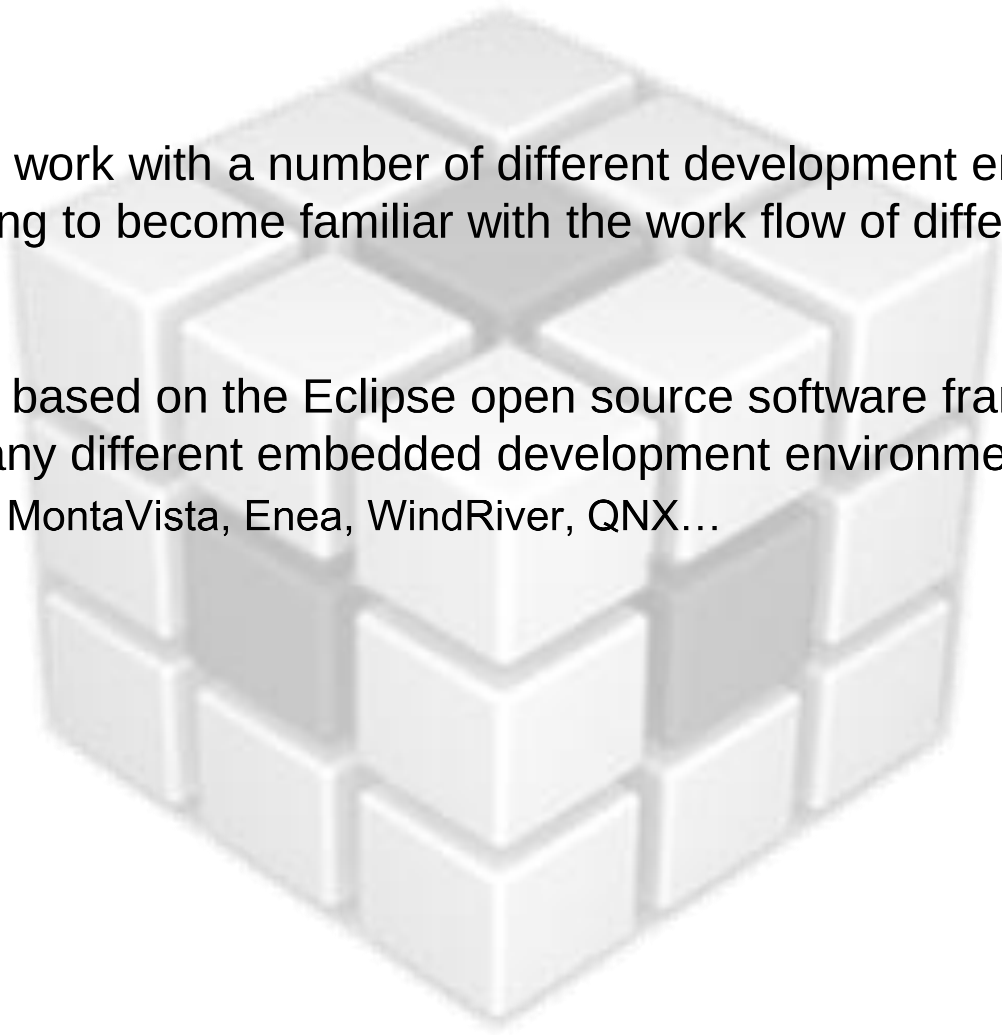
Erase Sector Selection

- [✓] Sector A: (0x3F4000 - 0x3F7FFF)
- [✓] Sector B: (0x3F0000 - 0x3F3FFF)
- [✓] Sector C: (0x3EC000 - 0x3EFFFF)
- [✓] Sector D: (0x3E8000 - 0x3EBFFF)
- [✓] Sector E: (0x3E4000 - 0x3E7FFF)
- [✓] Sector F: (0x3E0000 - 0x3E3FFF)
- [✓] Sector G: (0x3DC000 - 0x3DFFFF)
- [✓] Sector H: (0x3D8000 - 0x3DBFFF)

Code Security Password

Key 7 (0xAE7): FFFF

IDE Familiarity



- **Problem:**
  - Developers work with a number of different development environments. Thus needing to become familiar with the work flow of different tools.
- **Solution:**
  - CCSv4/5 is based on the Eclipse open source software framework which is used by many different embedded development environments:
    - ARM Ltd, MontaVista, Enea, WindRiver, QNX…

Licensing

- **Problems:**
  - Mid to large size customers want floating (server) license options
  - Free Evaluation Tools and DSK tools are out of date the day they are created
- **Solution:**
  - Integration of FlexNET licensing allows for a variety of licensing options.
  - Full tools, DSK tools, Free tools are all the same image and are kept up to date via the update manager

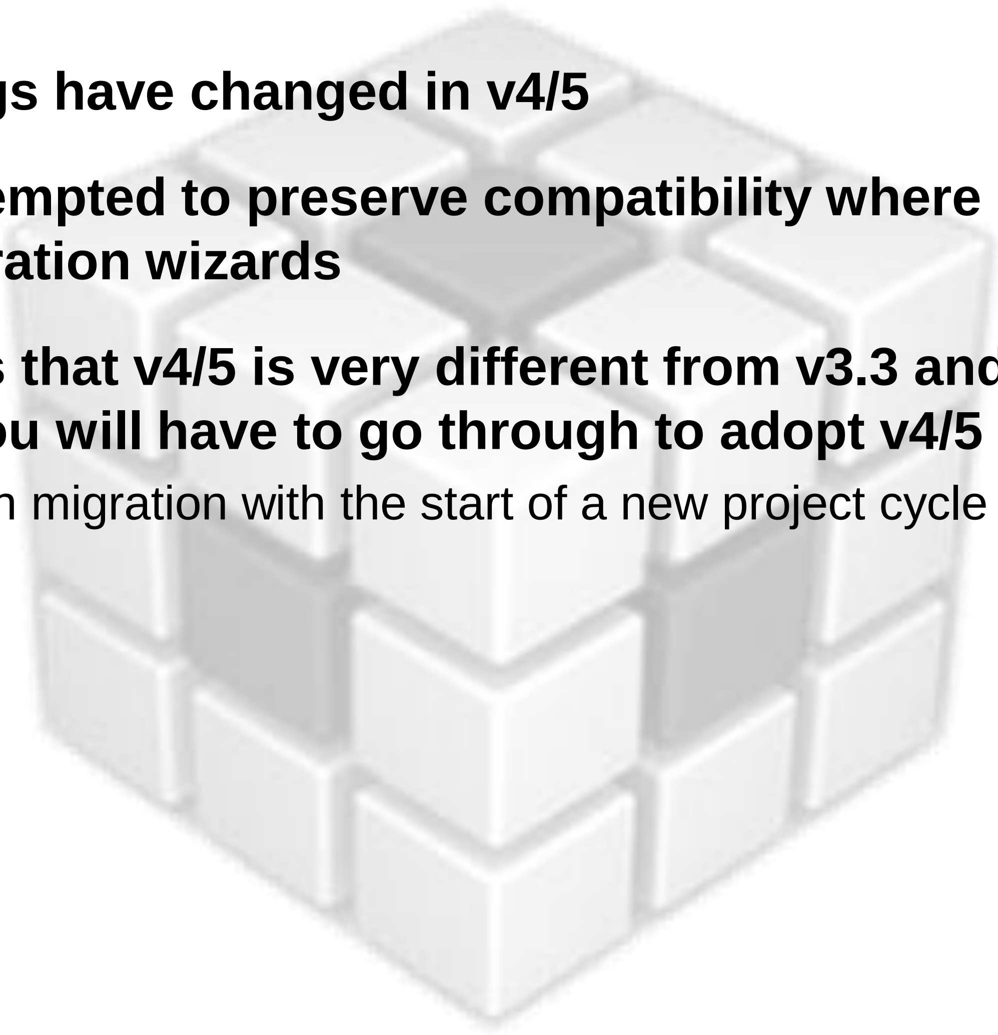
Update Delivery

- **Problems:**
  - People are unsure of what updates are needed
  - Downloading updates is painful
- **Solution:**
  - CCS will automatically check for updates on startup and indicate if content is available
  - Spectrum Digital & Blackhawk drivers are included in the CCS install
  - Service releases will only install content relevant to your installation (i.e. C2000 users only see C2000 content)
  - Much faster file server!!!!!!!!

Code Composer Studio v5

Main Changes from CCSv3

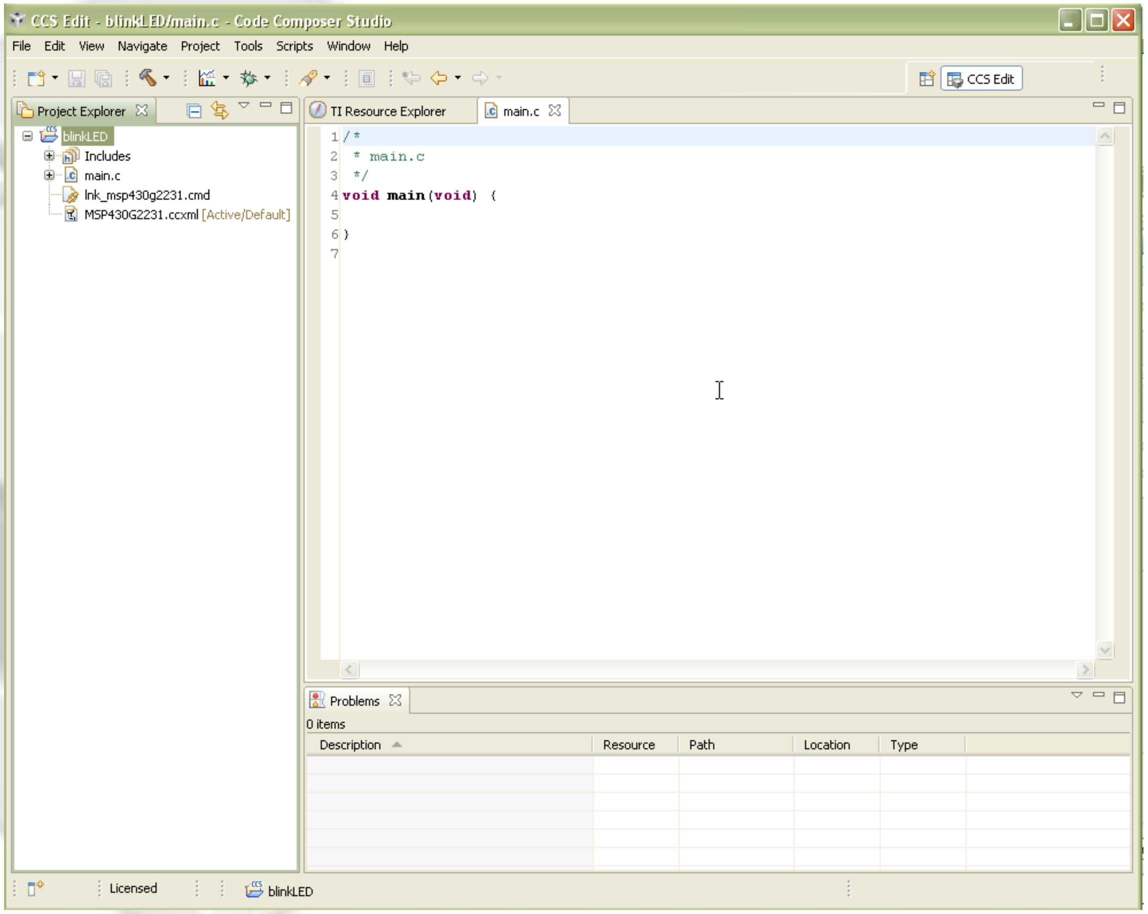
Summary



- **A lot of things have changed in v4/5**
- **We have attempted to preserve compatibility where possible and provide migration wizards**
- **The reality is that v4/5 is very different from v3.3 and there are steps that you will have to go through to adopt v4/5**
  - Best to align migration with the start of a new project cycle

IDE - Debugger integration

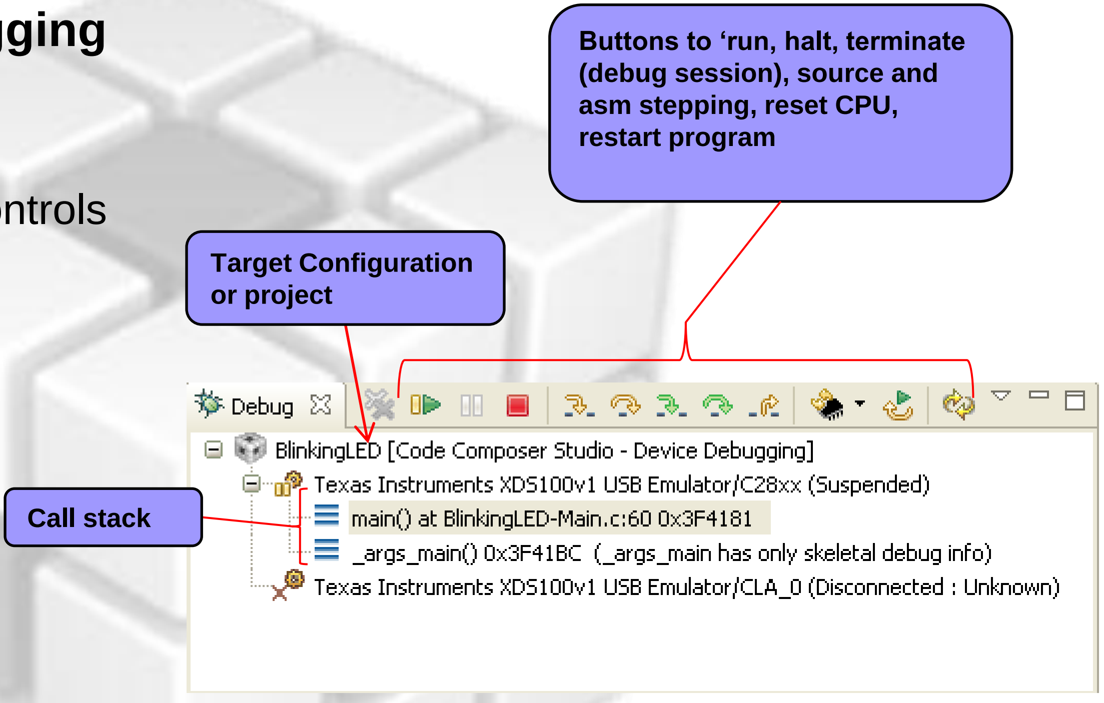
- **Debugger is not initialized when CCS starts**
  - It needs to be explicitly started (launch a debug session) and stopped (Terminate a debug session)
- **Allows for editing/code development to proceed without loading any debugger/driver components until needed**
- **Allows scripting/flashing tools to be used without closing CCS (just terminate the debugger and start a script)**



```
/*
 * main.c
 */
void main(void) {

}

```

View: Debug

- **Central window during debugging that provides "context" to all other debug views**
  - User selection in debug view controls what other debug views display
- **Shows all processors and optionally non-debuggable devices (e.g. ICEPICK)**
- **Display call stacks**
- **Provides execution control (run/halt/step)**


Buttons to 'run, halt, terminate (debug session), source and asm stepping, reset CPU, restart program
Target Configuration or project
Call stack
Debug

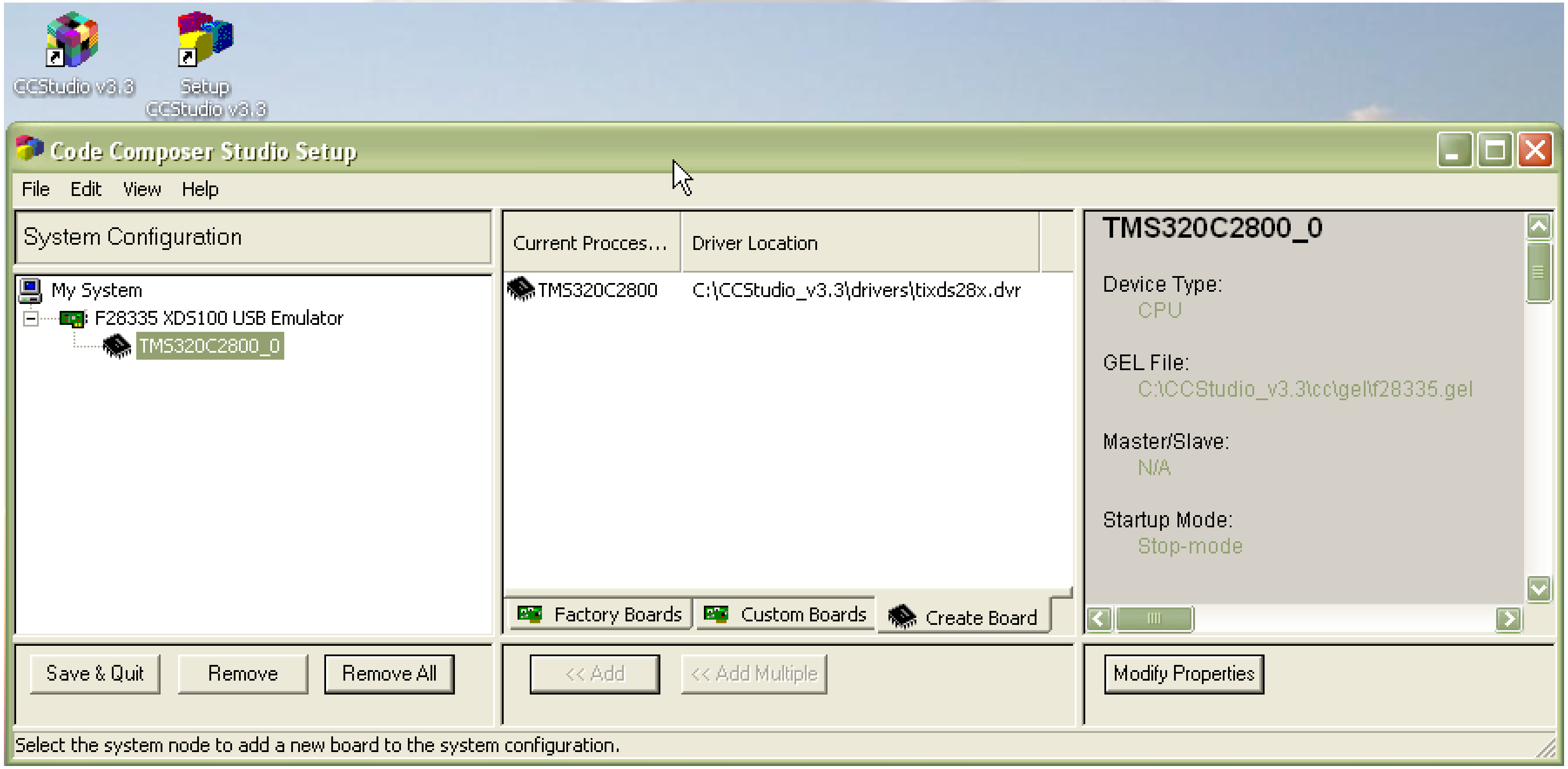
BlinkingLED [Code Composer Studio - Device Debugging]
Texas Instruments XDS100v1 USB Emulator/C28xx (Suspended)
main() at BlinkingLED-Main.c:60 0x3F4181
_args_main() 0x3F41BC (_args_main has only skeletal debug info)
Texas Instruments XDS100v1 USB Emulator/CLA_0 (Disconnected : Unknown)

Target Setup

- **Change to xml based target configuration**
  - Existing target configurations are not compatible
  - Device and connection are separated
    - You can change the emulator type that a configuration uses
    - 3P emulator vendors do not need to create import configurations for every device as you can just choose the emulator and the device
- **Setup is integrated into the IDE**
- **Target configurations can be included in a project so that a project automatically uses that configuration**

Target Setup (CCSv3)

Separate applications for the CCS debugger and CCS setup



My System
F28335 XDS100 USB Emulator
TMS320C2800_0

Current Procces...	Driver Location
TMS320C2800	C:\CCStudio_v3.3\drivers\tixds28x.dvr

TMS320C2800_0

Device Type:
CPU

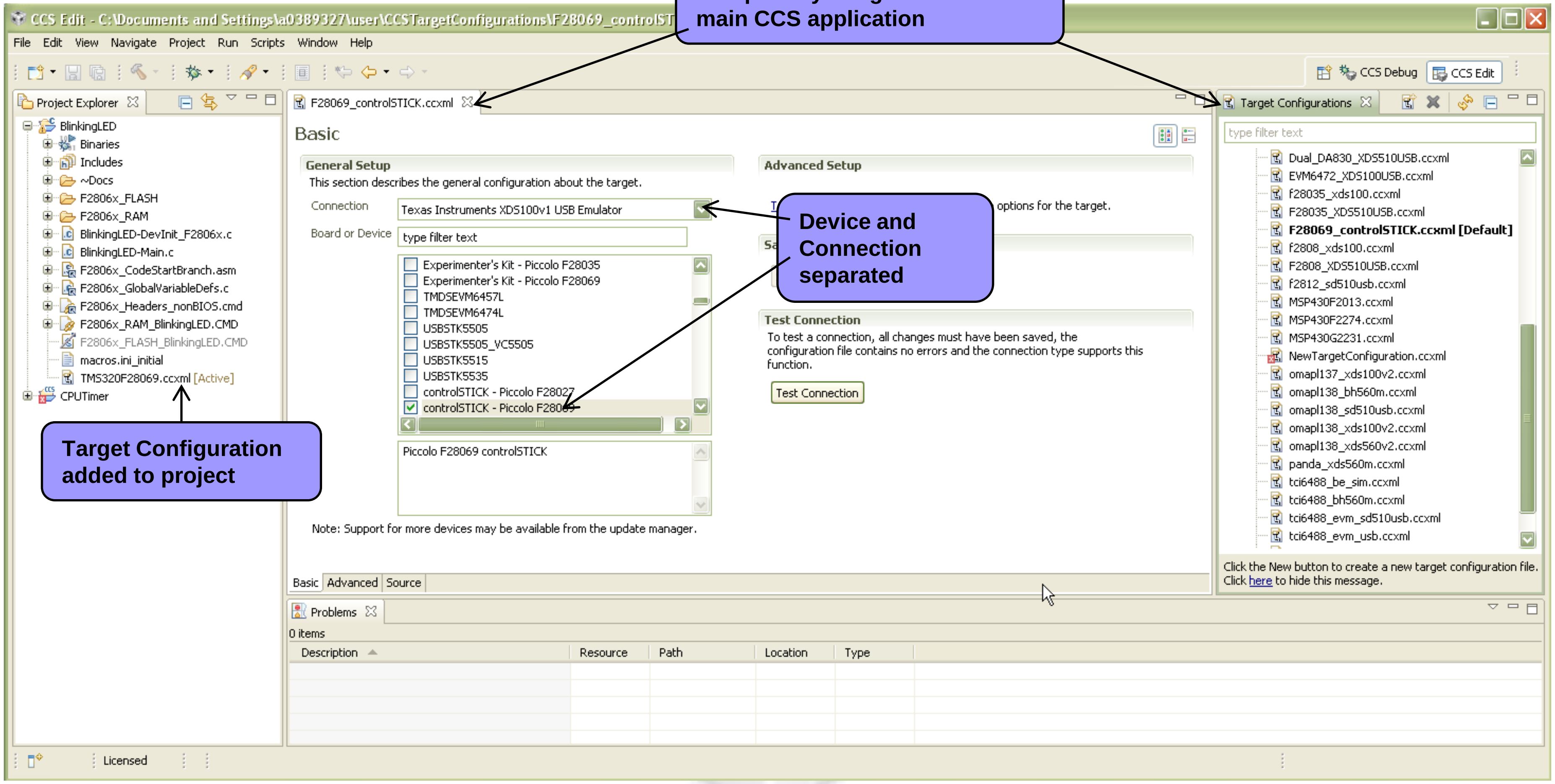
GEL File:
C:\CCStudio_v3.3\cc\gel\f28335.gel

Master/Slave:
N/A

Startup Mode:
Stop-mode

Select the system node to add a new board to the system configuration.

Target Setup (CCSv5)


CCS Edit - C:\Documents and Settings\a0389327\user\CCSTargetConfigurations\F28069_controlST
File Edit View Navigate Project Run Scripts Window Help
CCS Debug
CCS Edit
Project Explorer
F28069_controlSTICK.ccxml
Target Configurations
Basic
Advanced Setup
Test Connection
Basic Advanced Source
Problems
Licensed

Setup utility integrated with main CCS application

Device and Connection separated

Target Configuration added to project

Project Explorer:

- BlinkingLED
  - Binaries
  - Includes
  - ~Docs
  - F2806x_FLASH
  - F2806x_RAM
  - BlinkingLED-DevInit_F2806x.c
  - BlinkingLED-Main.c
  - F2806x_CodeStartBranch.asm
  - F2806x_GlobalVariableDefs.c
  - F2806x_Headers_nonBIOS.cmd
  - F2806x_RAM_BlinkingLED.CMD
  - F2806x_FLASH_BlinkingLED.CMD
  - macros.ini_initial
  - TMS320F28069.ccxml [Active]
- CPUTimer

General Setup

This section describes the general configuration about the target.

Connection	Texas Instruments XDS100v1 USB Emulator
Board or Device	type filter text

- [ ] Experimenter's Kit - Piccolo F28035
- [ ] Experimenter's Kit - Piccolo F28069
- [ ] TMDSEVM6457L
- [ ] TMDSEVM6474L
- [ ] USBSTK5505
- [ ] USBSTK5505_VC5505
- [ ] USBSTK5515
- [ ] USBSTK5535
- [ ] controlSTICK - Piccolo F28027
- [✓] controlSTICK - Piccolo F28069

Piccolo F28069 controlSTICK

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

Advanced Setup

options for the target.

Test Connection

To test a connection, all changes must have been saved, the configuration file contains no errors and the connection type supports this function.

Test Connection

Target Configurations:

type filter text

- Dual_DA830_XDS510USB.ccxml
- EVM6472_XDS100USB.ccxml
- f28035_xds100.ccxml
- F28035_XDS510USB.ccxml
- **F28069_controlSTICK.ccxml [Default]**
- f2808_xds100.ccxml
- F2808_XDS510USB.ccxml
- f2812_sd510usb.ccxml
- MSP430F2013.ccxml
- MSP430F2274.ccxml
- MSP430G2231.ccxml
- NewTargetConfiguration.ccxml
- omapl137_xds100v2.ccxml
- omapl138_bh560m.ccxml
- omapl138_sd510usb.ccxml
- omapl138_xds100v2.ccxml
- omapl138_xds560v2.ccxml
- panda_xds560m.ccxml
- tci6488_be_sim.ccxml
- tci6488_bh560m.ccxml
- tci6488_evm_sd510usb.ccxml
- tci6488_evm_usb.ccxml

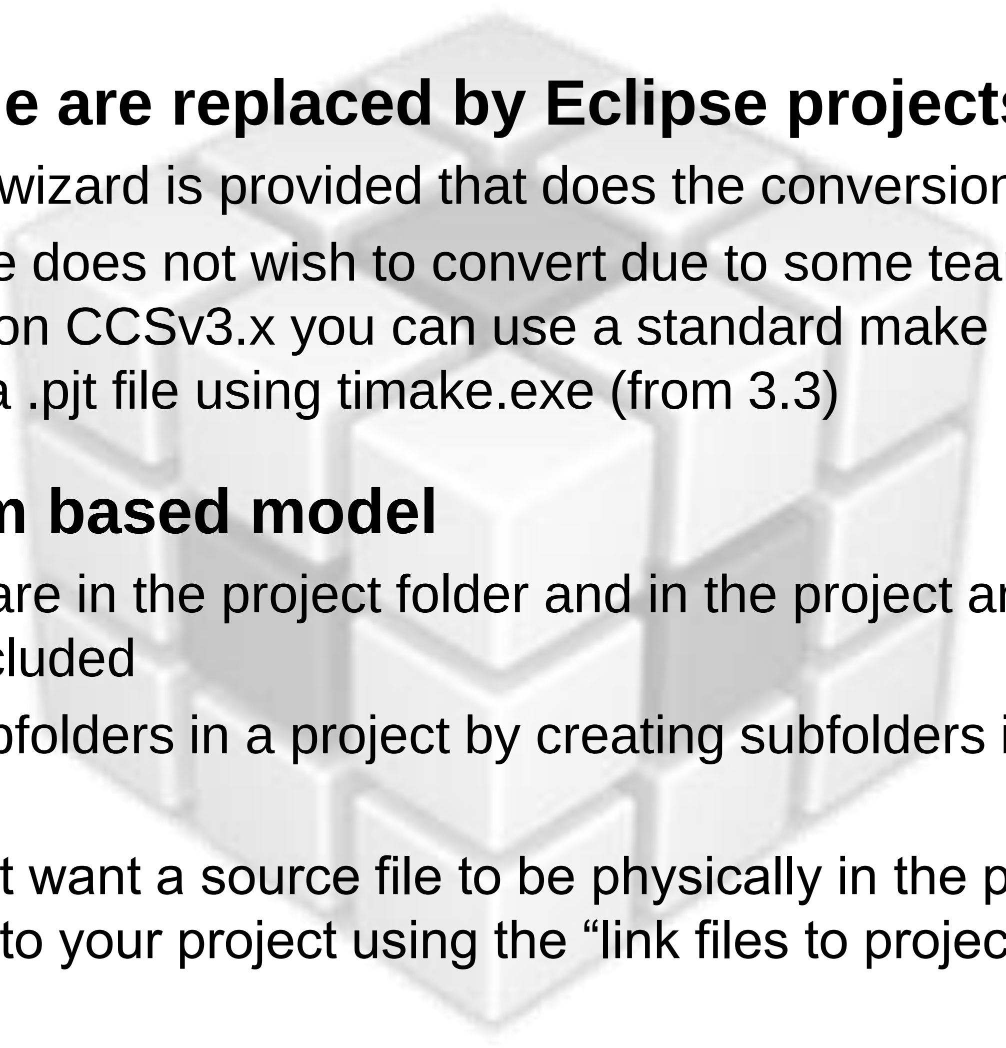
Click the New button to create a new target configuration file. Click here to hide this message.

Problems

0 items

Description	Resource	Path	Location	Type

CCS Projects



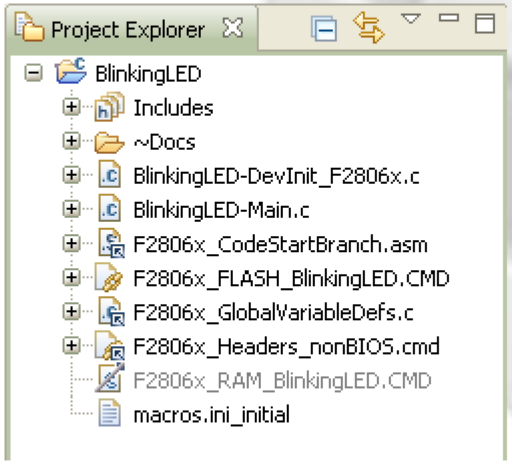
- **CCS .pjt file are replaced by Eclipse projects**
  - An import wizard is provided that does the conversion
  - If someone does not wish to convert due to some team members still being on CCSv3.x you can use a standard make project and tell it to build a .pjt file using timake.exe (from 3.3)
- **File system based model**
  - Files that are in the project folder and in the project and will be built unless excluded
  - Create subfolders in a project by creating subfolders in the project folder
  - If you don't want a source file to be physically in the project folder you add it to your project using the "link files to project" feature

Eclipse Concept: Projects

- **Projects map to directories in the file system**
- **Files can be added or linked to the project**
  - Adding file to project
    - Copies the file into your project folder
  - Linking file to project
    - Makes a reference to the file in your project
    - File stays in its original location
    - CCS 3.x project files used this concept exclusively
- **Projects are either open or closed**
  - Closed Projects:
    - Still defined to the workspace, but it cannot be modified by the Workbench
    - The resources of a closed project will not appear in the Workbench, but the resources still reside on the local file system
    - Closed projects require less memory and are not scanned during routine activity
    - This differs from CCS 3.x, where closed projects do not appear at all in the CCS 3.x project view window.
- **Projects that are not part of the workspace must be imported into the active workspace before they can be opened**
  - Both CCSv4/5, CCE projects and legacy CCSv3 projects can be imported into the workspace

View: Project Explorer

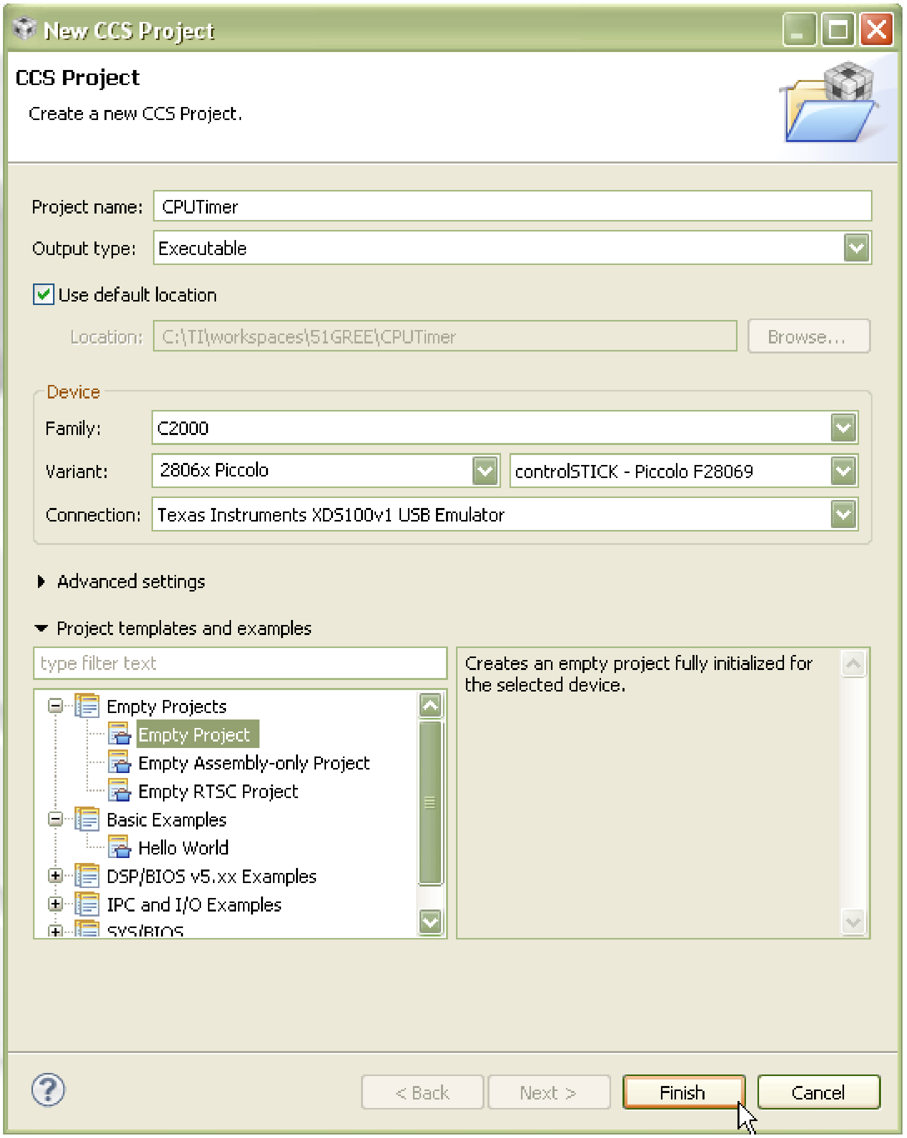
- **Displays all projects defined in the active workspace**
- **The view is mostly a representation of the file system of the project folder**
  - Linked files are marked with a special link graphic in the icon
- **Use filters to hide various file types to reduce clutter in the view**
  - Default is to filter CCS generated project files (.*)



BlinkingLED
- Includes
- ~Docs
- BlinkingLED-DevInit_F2806x.c
- BlinkingLED-Main.c
- F2806x_CodeStartBranch.asm
- F2806x_FLASH_BlinkingLED.CMD
- F2806x_GlobalVariableDefs.c
- F2806x_Headers_nonBIOS.cmd
- F2806x_RAM_BlinkingLED.CMD
- macros.ini_initial

New Project Wizard

- **Device Variant selection**
  - When you select the device variant it impacts the build options used for run-time models and the default rts library selection, linker command file selection…
  - Selecting a Connection type also will generate a target configuration file for the project
- **Specify a compiler version**
  - Expand the 'Advanced settings' section to specify a specific compiler version to use for the project
- **Project Templates**
  - Generates "ready to build" example projects



New CCS Project

CCS Project

Create a new CCS Project.

Project name: CPUTimer

Output type: Executable

✓ Use default location

Location: C:\TI\workspaces\51GREE\CPUTimer

Device

Family: C2000

Variant: 2806x Piccolo | controlSTICK - Piccolo F28069

Connection: Texas Instruments XDS100v1 USB Emulator

Advanced settings

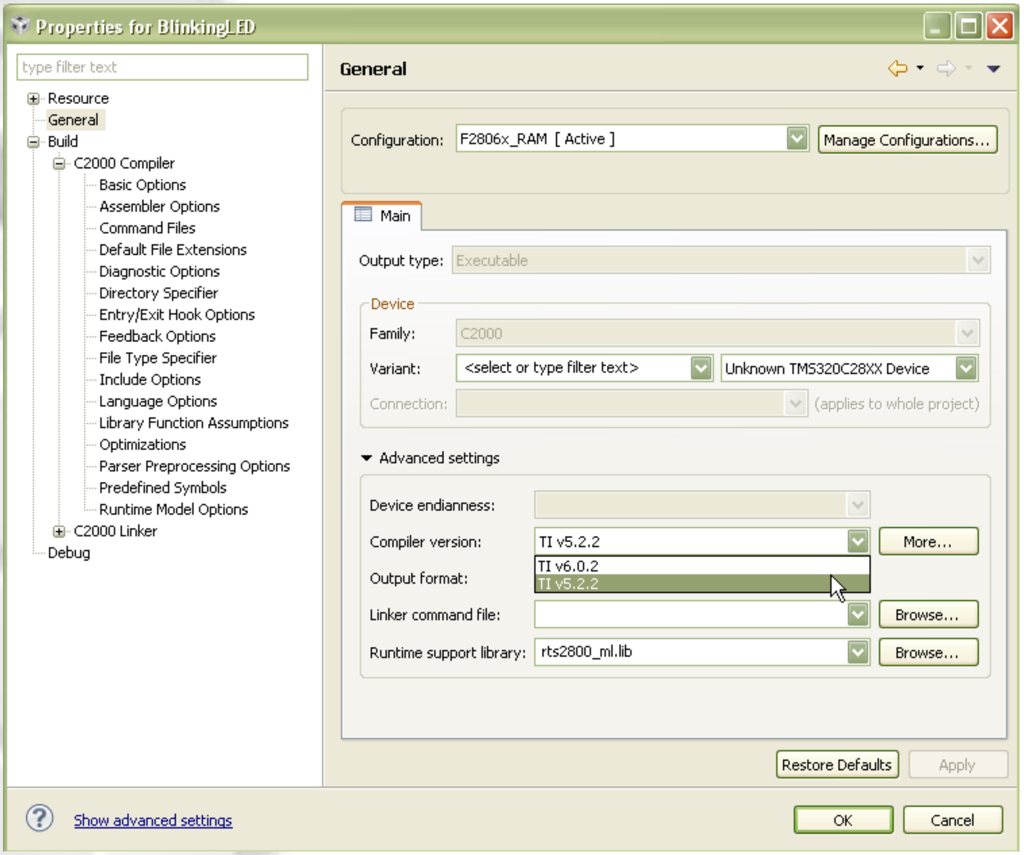
Project templates and examples

- Empty Projects
  - Empty Project
  - Empty Assembly-only Project
  - Empty RTSC Project
- Basic Examples
  - Hello World
- DSP/BIOS v5.xx Examples
- IPC and I/O Examples
- SYS/BIOS

Creates an empty project fully initialized for the selected device.

Compilers

- CCS supports building with and debugging the output of many different versions of TI compilers
- Within a project you can specify the version of the compiler to use for a particular project


Properties for BlinkingLED

type filter text

- Resource
- General
- Build
  - C2000 Compiler
    - Basic Options
    - Assembler Options
    - Command Files
    - Default File Extensions
    - Diagnostic Options
    - Directory Specifier
    - Entry/Exit Hook Options
    - Feedback Options
    - File Type Specifier
    - Include Options
    - Language Options
    - Library Function Assumptions
    - Optimizations
    - Parser Preprocessing Options
    - Predefined Symbols
    - Runtime Model Options
  - C2000 Linker
- Debug

General

Configuration: F2806x_RAM [Active] — Manage Configurations...

Main

Output type: Executable

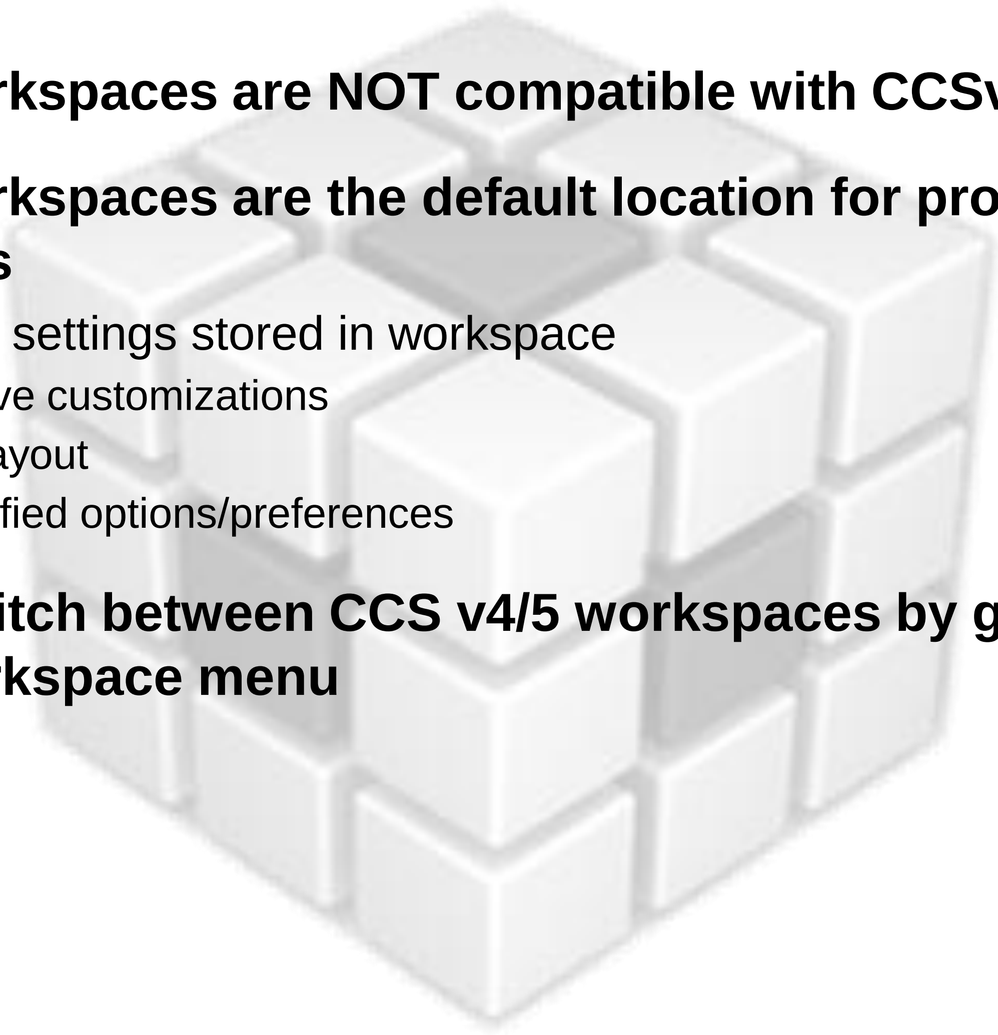
Device
Family: C2000
Variant: <select or type filter text> — Unknown TMS320C28XX Device
Connection: (applies to whole project)

Advanced settings
Device endianness:
Compiler version: TI v5.2.2 — More...
TI v6.0.2
TI v5.2.2
Output format:
Linker command file: — Browse...
Runtime support library: rts2800_ml.lib — Browse...

Restore Defaults — Apply

Show advanced settings — OK — Cancel

Workspaces



- **CCSv3.x workspaces are NOT compatible with CCSv4/5**
- **CCSv4/5 workspaces are the default location for projects and CCS user settings**
  - E.g. of user settings stored in workspace
    - Perspective customizations
    - Window layout
    - user modified options/preferences
- **User can switch between CCS v4/5 workspaces by going to File->Switch Workspace menu**

Automation

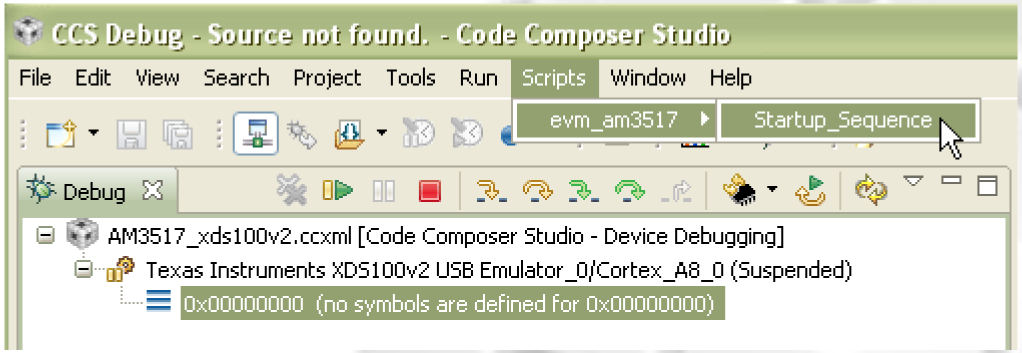
- **GEL**
  - GEL is still present in CCSv4/5
  - Functions that map to GUI related items such as opening a window or building a project are not available
- **New Scripting Environment (Debug Server Scripting)**
  - JAVA APIs
    - Default supported language is JavaScript
      - Perl, TCL, Python has also been used with DSS
  - Debug Server Scripting will be the main feature used for automation as it allows access to the functionality of the debugger
  - No support for project management
- **Scripting Console**
  - Window within CCS that provides access to the scripting environment as well as command driven operation of CCS

GEL

- **Startup Files**
  - Can still be GEL based and specified in Setup
- **Automation**
  - Most scripts will still work fine but it is recommended that you move regression test type activities to Debug Server Scripting
- **GEL Expressions**
  - Can still be used as conditions on breakpoints or start addresses or in the watch window
  - GEL is still the expression evaluator in the debugger
  - GEL hotmenus and dialogs are supported
- **Unsupported GEL functions**
  - GEL_WatchAdd(), project related functions
    - Mostly used by people to create peripheral register windows in the watch window. This should be migrated to register xml files
  - Project Build commands
  - GEL GUI actions

GEL Menus

- **Appear on the main menu under “Scripts”**

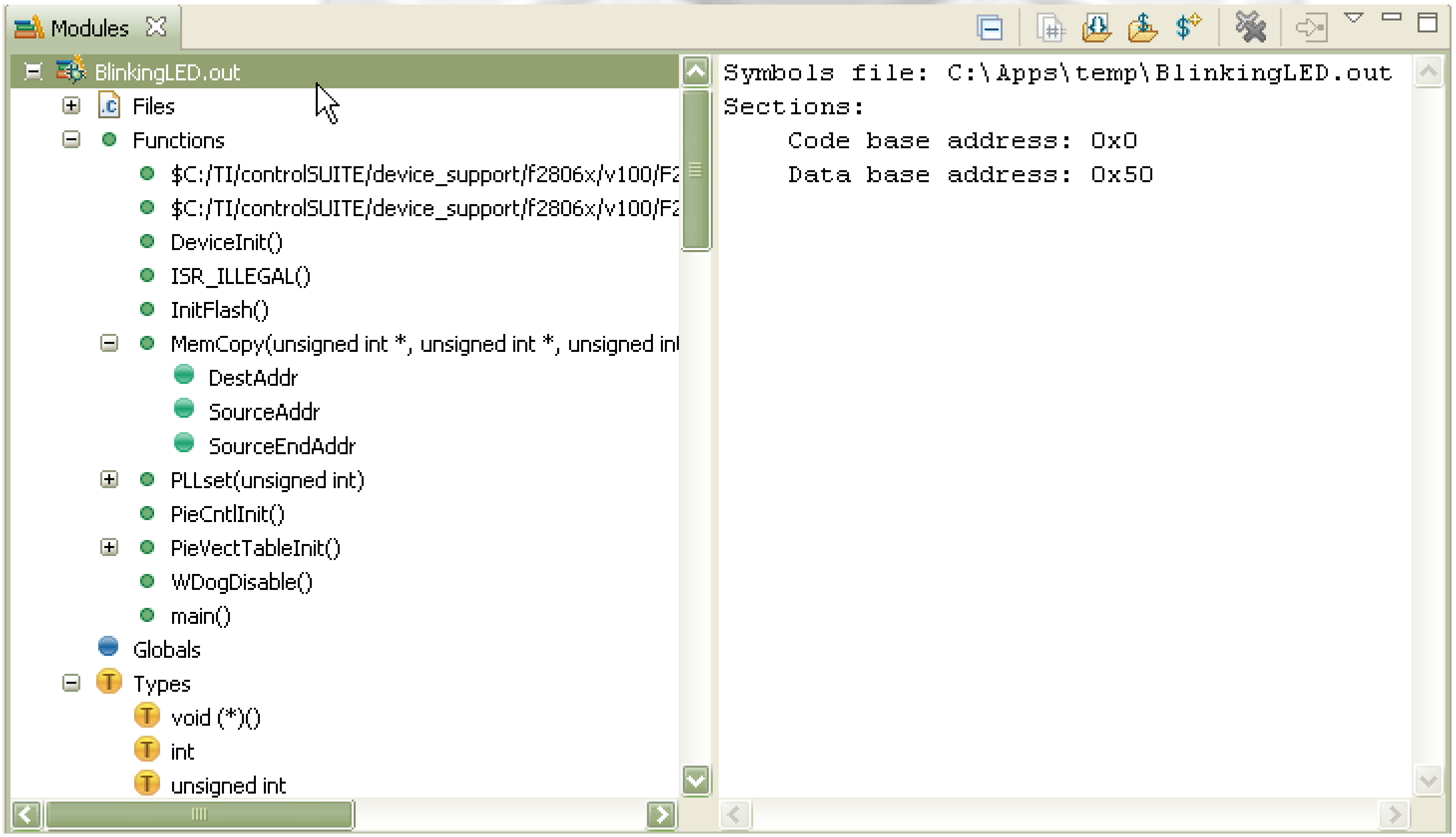

CCS Debug - Source not found. - Code Composer Studio
File Edit View Search Project Tools Run Scripts Window Help
evm_am3517
Startup_Sequence
Debug

AM3517_xds100v2.ccxml [Code Composer Studio - Device Debugging]
Texas Instruments XDS100v2 USB Emulator_0/Cortex_A8_0 (Suspended)
0x00000000 (no symbols are defined for 0x00000000)

- **GEL created menu items appear under Scripts menu only when Debugger has been started**
  - GEL engine is part of debugger
- **Debug view controls which script menu’s are available**
  - Only scripts applicable to currently selected processor are available

View: Modules

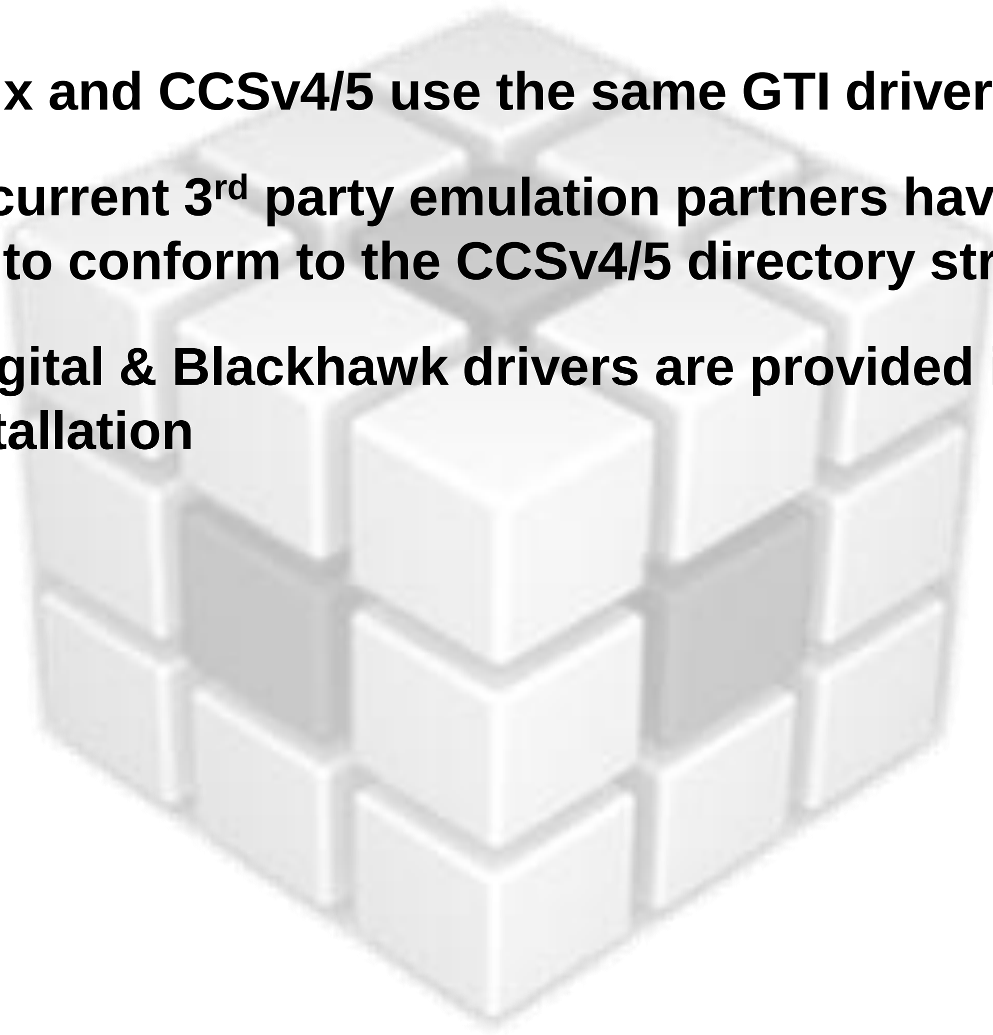
- **CCSv4/5 equivalent of the “Symbol Browser” in v3**
- **Provides information on all loaded symbol files**
- **‘View -> Other… -> Debug -> Modules’**



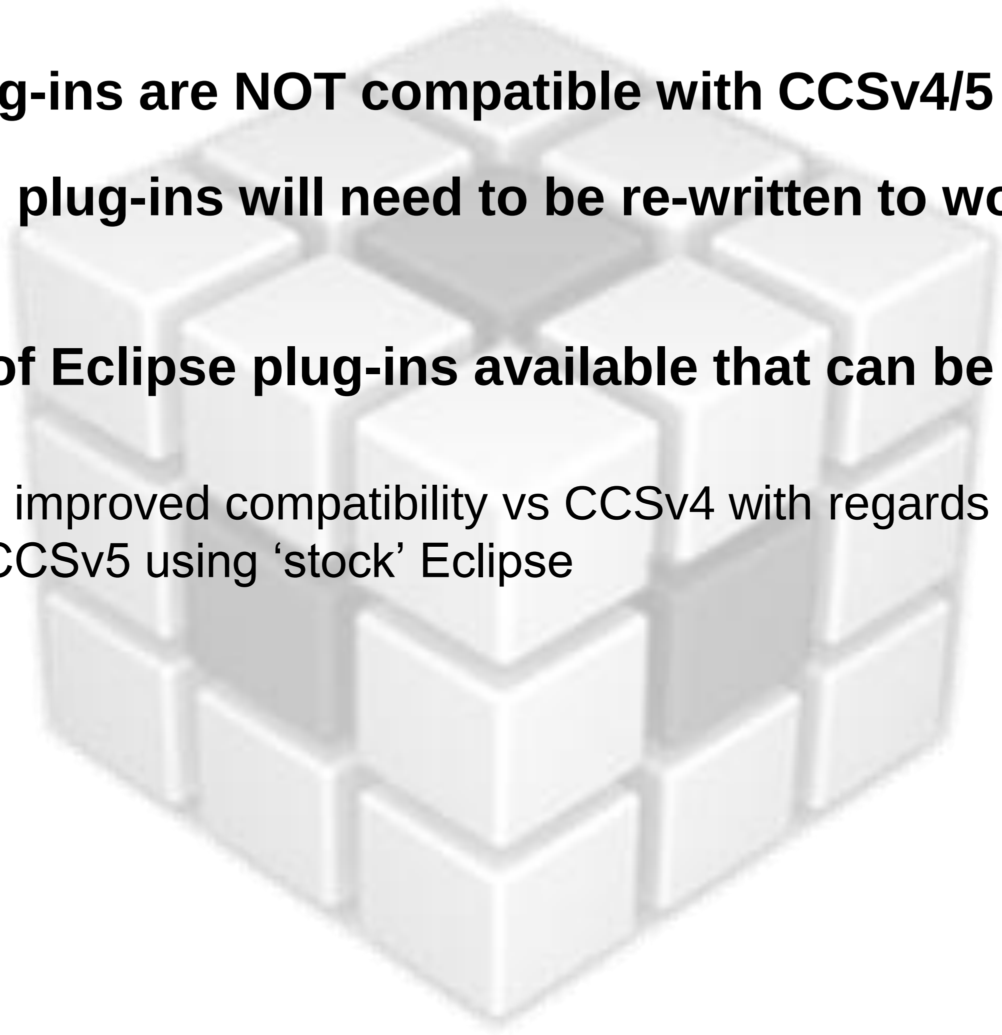
```
Symbols file: C:\Apps\temp\BlinkingLED.out
Sections:
    Code base address: 0x0
    Data base address: 0x50
```

Emulation Vendors

- Both CCSv3.x and CCSv4/5 use the same GTI driver interface
- Most of our current 3rd party emulation partners have updated their drivers to conform to the CCSv4/5 directory structure
- Spectrum Digital & Blackhawk drivers are provided in a base CCSv4/5 installation

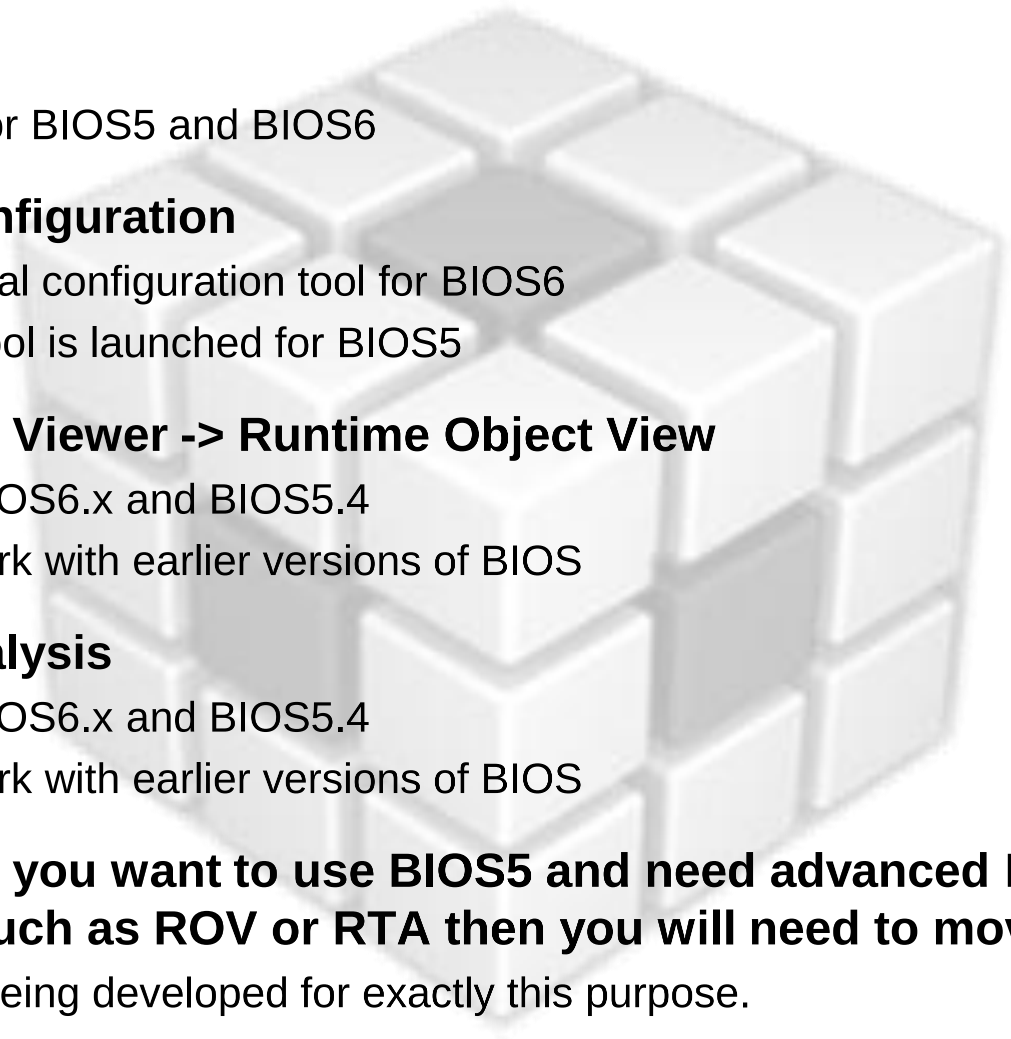


CCS Plug-ins



- **CCSv3.x plug-ins are NOT compatible with CCSv4/5**
- **Any existing plug-ins will need to be re-written to work with CCSv4/5**
- **Thousands of Eclipse plug-ins available that can be used with CCSv4/5**
  - CCSv5 has improved compatibility vs CCSv4 with regards to Eclipse plug-ins due to CCSv5 using ‘stock’ Eclipse

DSP/BIOS



- **Build**
  - Supported for BIOS5 and BIOS6
- **Graphical Configuration**
  - New graphical configuration tool for BIOS6
  - Old gtconf tool is launched for BIOS5
- **Kernel Object Viewer -> Runtime Object View**
  - Works for BIOS6.x and BIOS5.4
  - Does not work with earlier versions of BIOS
- **Real-time Analysis**
  - Works for BIOS6.x and BIOS5.4
  - Does not work with earlier versions of BIOS
- **In summary if you want to use BIOS5 and need advanced BIOS debug capabilities such as ROV or RTA then you will need to move to BIOS5.4.**
  - BIOS5.4 is being developed for exactly this purpose.

Register Definitions

- **CCSv3.3 and CCSv4/5 support register definition files**
  - You can reuse your files from CCSv3.3, some device level files may require modification to correctly include the scan path information
- **Using the Alias feature via GEL_WatchAdd()**
  - Some people used GEL files to create custom watch windows that are really peripheral register windows (instead of creating xml register files).
    - This does not work in CCSv4/5 - GEL_WatchAdd() is not available
      - Replaced by CCSv4/5 Scripting Console command ‘expAdd’

Questions?

TEXAS INSTRUMENTS