D. Richard Brown III Associate Professor Worcester Polytechnic Institute Electrical and Computer Engineering Department drb@ece.wpi.edu

October 19-20, 2009

DIGITAL SIGNAL PROCESSING AND APPLICATIONS WITH THE TMS320C6713 DSK

Day I handouts







Technology for Innovators[™]

Workshop Goals

- Correctly install Texas Instruments Code Composer Studio IDE and DSK drivers
- Become familiar with
 - DSP basics
 - TMS320C6713 floating point DSP architecture
 - TMS320C6713 DSP starter kit (DSK)
 - Code composer studio integrated development environment (IDE)
 - Matlab design and analysis tools
- Learn how to program the C6713
 - Writing and compiling code
 - Fixing errors
 - Downloading code to the target and executing
 - Debugging
- Write and run useful programs on the C6713 DSK
- Learn about DSP applications
- Learn where to find help





Take Home Items

- "Digital Signal Processing and Applications with the C6713 and C6416 DSK" by Chassaing & Reay, 2008
- Texas Instruments TMS320C6713 DSK including
 - DSK board with TMS320C6713 DSP chip
 - USB cable
 - Power supply
 - CD with Code composer studio IDE (v3.1) and electronic documentation
 - DSK technical reference manual
 - DSK quick start installation guide
 - Matlab/Simulink trial CD and other promotional material







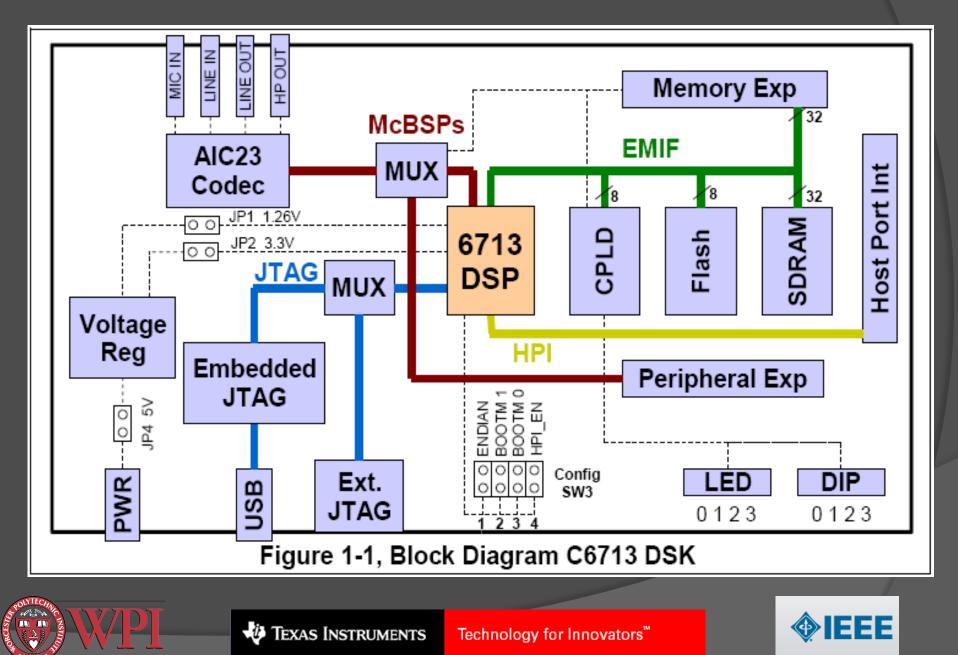
C6713 DSK Overview

- 225 MHz TMS320C6713 floating point DSP
- AIC23 stereo codec (ADC and DAC)
 - Ideal for audio applications
 - 8-96 kHz sample rates
- Memory
 - 16 MB dynamic RAM
 - 512 kB nonvolatile FLASH memory
- General purpose I/O
 - 4 LEDs
 - 4 DIP switches
- USB interface to PC

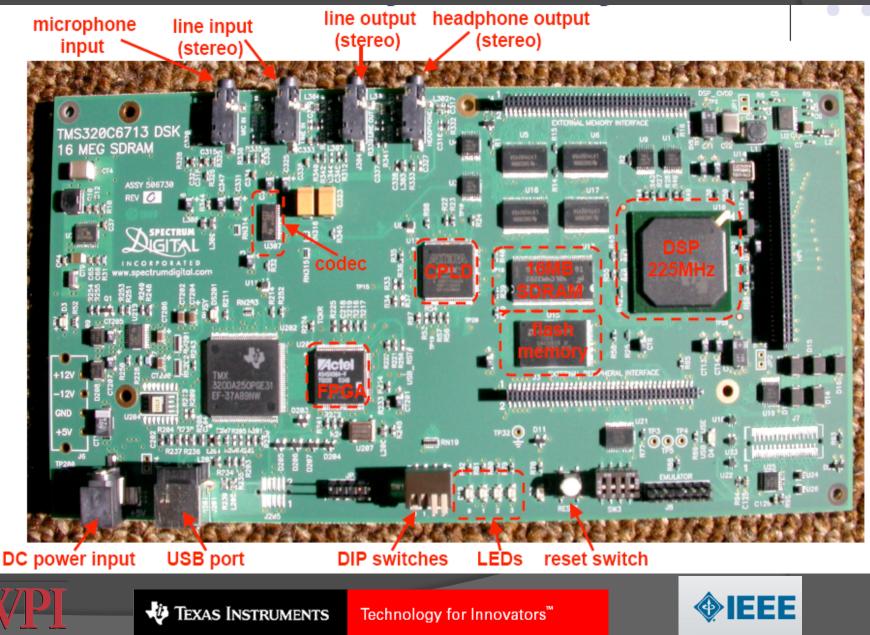




C6713 DSK Functional Block Diagram



C6713 DSK Physical Layout



Is my DSK working? DSK Power On Self Test

- Power up DSK and watch LEDs
- Power On Self Test (POST) program stored in FLASH memory automatically executes
- POST takes 10-15 seconds to complete
- All DSK subsystems are automatically tested
- During POST, a IkHz sinusoid is output from the AIC23 codec for I second
 - Listen with headphones or watch on oscilloscope
- If POST is successful, all four LEDs blink 3 times and then remain on





Is my DSK working? DSK Diagnostic Utility

- Install CCS 3.1
 - Directions in "Quick Start Installation Guide"
 - More detailed directions available on spinlab web site
 - Diagnostic utility automatically installed

IEXAS INSTRUMENTS

5 671 3DSK Diagnostics		5 671 3DSK Diagnostics	🛛
General Advanced Overall Diagnostic Test Diagnostic Status: Image: USB Diagnostics Image: Diagnostic Status: Image: DSP Diagnostics Image: DSP Diagnostics Image: DSP Diagnostics Image: DSK: Image: DSK: DIAGNOSTICS Image: DSK: <td>About Start Stop Reset Emu Reset DSK Save As</td> <td>General Advanced Overall Diagnostic Test Diagnostic Status: © USB Diagnostics Diagnostic Status: © DSP Diagnostics DSF Diagnostics © DSP Diagnostics DSK: © External Memory Component Value © Flash Diagnostics Codec Diagnostics © LED Diagnostics CPLD Version 2 © Dip Swt Diagnostics Diagnostics Diagnostics</td> <td>About Start Stop Beset Emu Reset DSK Save As</td>	About Start Stop Reset Emu Reset DSK Save As	General Advanced Overall Diagnostic Test Diagnostic Status: © USB Diagnostics Diagnostic Status: © DSP Diagnostics DSF Diagnostics © DSP Diagnostics DSK: © External Memory Component Value © Flash Diagnostics Codec Diagnostics © LED Diagnostics CPLD Version 2 © Dip Swt Diagnostics Diagnostics Diagnostics	About Start Stop Beset Emu Reset DSK Save As
Diagnostic Results		Diagnostic Results	

lechnology for Innovators

6713 DSK

Diagnostics Utility v3.1

Code Composer Studio IDE

- Connect power supply to DSK
- Wait for POST to complete
- Connect USB cable from PC to DSK
 - If this is the first time connecting the DSK, you may be asked to install a driver. The driver is on the Code Composer Studio CD and will automatically be found by Windows if the CD is in the drive.
- Launch Code Composer Studio C6713 DSK
- CCS will load and wait for your input

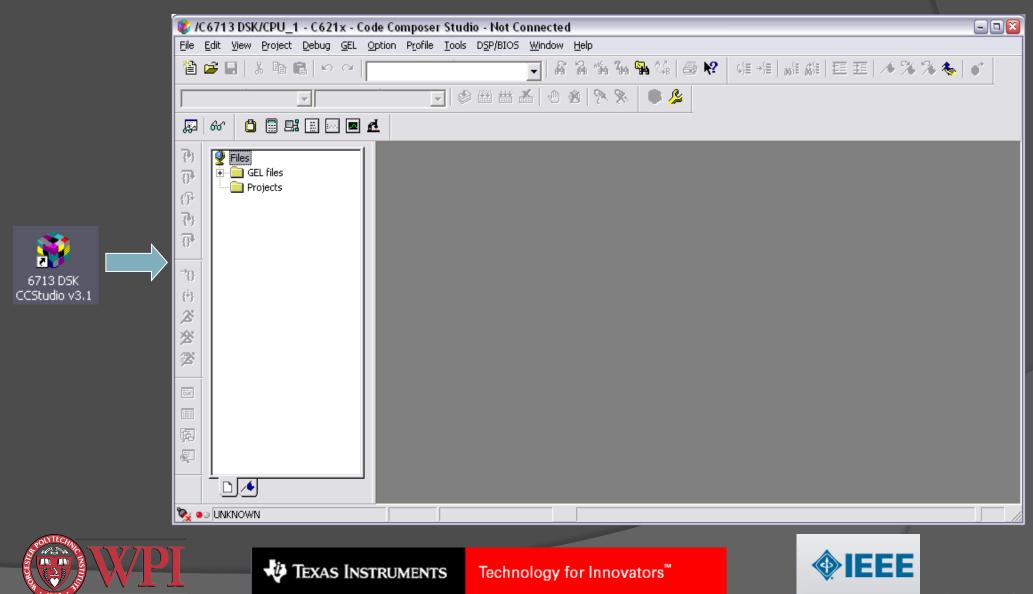




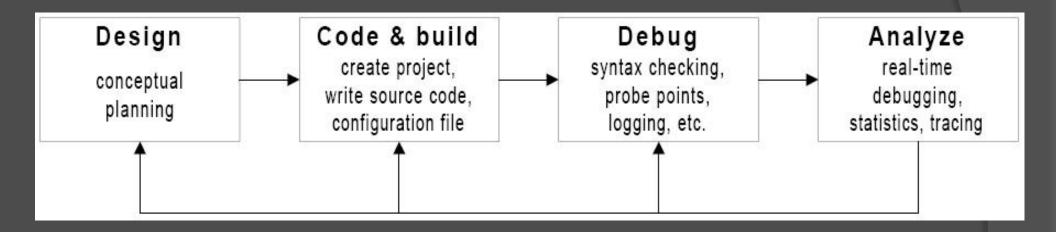


6713 DSK CCStudio v3.

Code Composer Studio IDE



CCS Integrated Development Environment



<u>Useful TI documentation (available online or on your hard drive)</u>: SPRU509EPDF CCS v3.1 IDE Getting Started Guide C6713DSK.HLP C6713 DSK specific help material

Note that your DSK includes CCS v3.1.







Connecting to the C6713 DSK

😰 /C 6713 DSK/CPU -1	- C621x - Code Composer S	tudio - Not Connee	cted							
	Debug GEL Option Profile To							omposer Studi		
쒑 🚅 🖬 🗼 🐴 🛙	Breakpoints		8 2 4	File Edit	View Project	: Debug G	EL Option	Profile Tools	DSP/BIOS Wind	low Help
	Probe Points			1 🗎 🖻	. X b		×		•	886 8
	Step Into	F11	9 🕸 🕅 🕅			-	<u> </u>			0 🙊 🔊 🖉
🔊 60 📋 🗐 📑	Step Over	F10		1	1					
🔊 🛷 📋 🖩 🖽	Step Out	Shift+F11		🛛 📈 60	' 📋 🗐 🖉	H 🗄 🖂	<u>a</u> 🛃			
🕑 🔮 Files	Run	F5					_1			
	Halt	Shift+F5		7) 🖗	Files			Disassembly		- 🗆 🔀
Projects	Animate	Alt+F5			GEL files				00006000	
(P	Run Free	Ctrl+F5		(P)	Projects				4 0010AA7A	
(ł)	Run to Cursor	Ctrl+F10							3 20000390	
	Set PC to Cursor	Ctrl+Shift+F10		()					2 00008000	
0.	Multiple Operation			() ()) 00000990 4 020000A8	
	Assembly/Source Stepping	×		→0				000008D8	3 00006000	
()	Reset CPU	Ctrl+R							C 023C82E6 D 00006000	
	Restart	Ctrl+Shift+F5		{ \ }					1 000000000 4 0210805A	
X	Go Main	Ctrl+M		a a caracteria de la ca			<			▶ .::
※	Reset Emulator	Ctrl+Shift+R								
28	Connect	Alt+C		🛛 🌋 👘	D 🕖		_			
	Restore Debug State				<u> </u>					
	Enable Thread Level Debugging			GEL S	tartUp Co	mplete.				
	Real-time Mode				-	-				
兩	Enable Rude Real-time Mode									
S	✓ Flush Pipeline on Halt					_				
				The target	t is now connect	ed 🛛				
						_				
🇞 💿 UNKNOWN			Toggle	🖌 💿 HA	ALTED					
	, , ,						,			/
POLYTECHIN										
									IEEE	
		xas Instrume	ENTS TO	echnology	/ for Innov	ators™			/ICEE	

Opening an Existing Project

🈻 /C6713 DSK	./CPU_1	- C671	x - C	ode Co	mposer	Studio
File Edit View	Project	Debug	GEL	Option	Profile	Tools D
12 🚔 🗐 💧	New.					
· · · · · · · · · · · · · · · · · · ·	Open					
]		xternal M		Э		1 🛸 🛙
🖾 60 🗂		t to Make				
🛛 🖓 🖓		iles to Pr	oject			
P Files	Save Close					ibly
🔐 🕂 🛄 GE		e Contro	I		•	. 08CO -08C4
()+ · · · · · · · · · · · · · · · · · ·	Comp	ile File			Ctrl F7	18C8
6	Build	lie i lie			E7	18CC
	Rebui	IA AII			1.7	08D0
- 0°	Stop I					D8D4
	Build					D8D8
0						-D8DC
(4)		gurations				08E0
2		Options				18F4
*		pecific Op				
	Funct	ion Level	Optio	ns		
🌋 🗖 d 📈	Projec	t Depen	dencie	s		
	Show	Project D	epena	dencies		
GEL Startl	Show	File Depe	endena	ties		
SEL Starte	Scan	All File De	epende	encies		
	Recer	nt Project	: Files		•	

Project->Open

Project Open			? 🛛
Look in: 🗀 mypro	jects	- 🗧 💣 🗉	≣-
Adaptc AdaptIDFIR AdaptIDFIRW AdaptIDFIRW AdaptIDIIR Adaptnoise Adaptnoise_2IN	Adaptpredict Adaptpredict_2IN Aliasing Aliasing BeatDetector SetDetector Solution_4led	bios_sine_ctrl bios_sine_intr bios_sine_intr bpsk Code_casm detect_play DFT	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
			>
File <u>n</u> ame:			<u>D</u> pen
Files of type: Proje	ect Files (*.pjt)	▼ (Cancel
			<u>H</u> elp

Select a .PJT file and press "Open". Chassaing example projects should be in c:\CCStudio_v3.1\myprojects\

Other example projects for the C6713 can be found in c:\CCStudio_v3.1\examples\dsk6713







Compiling/Building a Project

Project->Build (F7)



TEXAS INSTRUMENTS



Loading and Running a Project on the C6713 DSK

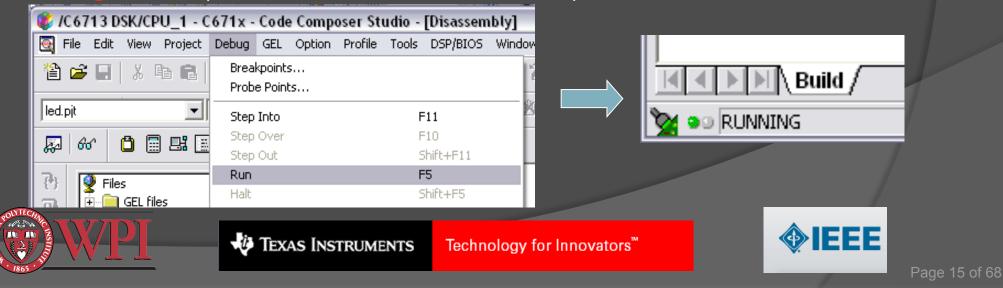
File-> Load Program (ctrl+L)

😻 I	C6713	B DS	SK/CP	U_1 - C	671x -	Code	e Comp	ose
()	File Ed	dit	View	Project	Debug	GEL	Option	Pri
×	New						- H 🗖	
	Oper	٦			Ctrl+	Ō	- 6	_
lec	Close	Э					- F	T.
	Save)			Ctrl+	S	- 6	-
2	Save	As.					- 11	_
_	Save	All					- 11	/*
{t)	Load	Dro	aron		Ctrl+	1		*
a			gram			_		*.
0°	Relo	ad P	rogran	n	Ctrl+	Shift+	·L	*/

Load Program	? 🛛
Look jn: 🗀 Debug	- 🖬 🖆 🔳
led.out	
File <u>n</u> ame:	<u>O</u> pen
Files of type: .out	✓ Cancel
	Help

Select the .out file in the project\Debug directory. Program is sent to DSK.

Debug->Run (F5 or the Run button 🖉



Halting a Running Program on the C6713 DSK

Debug->Halt (shift+F5 or the Halt button 🗾).

🌾 /C6713 DSK/CPU_1 - C	671x -	Code	e Comp	oser St	udio -	[Disassem	bly]		
🧕 File Edit View Project	Debug	GEL	Option	Profile	Tools	DSP/BIOS	Wind	ow	F
웥 🚅 🖬 🕺 🛍 💼		kpoint: e Point						2	Ŕ
led.pjt	Step	Into			F	11		21	9
🔊 60° 📋 🖩 🖽		Over				10			
	Step	Out			2	ihift+F11			
🔁 👰 Files	Run				F	5		I	
🙃 🚊 GEL files	Halt				5	ihift+F5		I	
	Anim	ate			- A	Alt+F5			







Tip: Fixing the search path

Add C:\CCStudio_v3.1\C6000\dsk6713\include to the search path

Project -> Build Options -> [Compiler tab] -> [Preprocessor catego

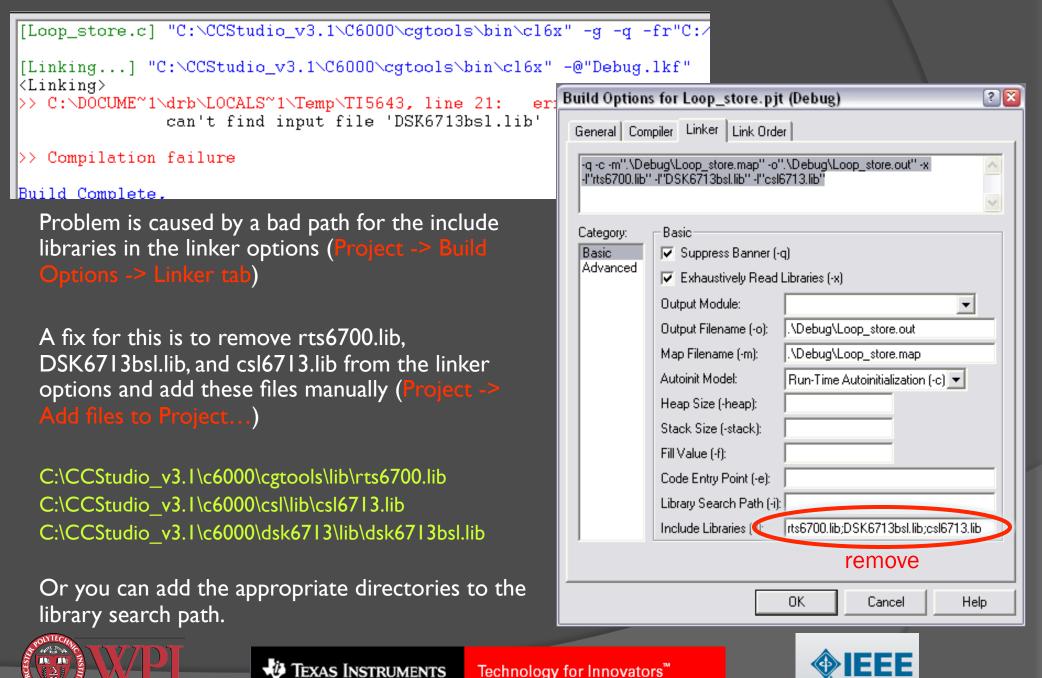
Build Options fo	or Sine8_LED.pjt (D	ebug)	? 🛛
General Compile	er Linker Link Order		
-g -s -fr"C:\CCStu -i"C:\CCStudio_v	udio_v3.1\MyProjects\sin /3.1\C6000\dsk6713\inc	e8_LED\Debug'' lude'' •d''CHIP_6713'' •mv6710	
Category: Basic Advanced Feedback Files Assembly Parser Preprocessor Diagnostics	Preprocessor Include Search Path (-i) Pre-Define Symbol (-u): Undefine Symbol (-u): Preprocessing: Continue with Comp	udio_v3.1\C6000\dsk6713\ind CHIP_6713 None	lude
	0	Cancel H	lelp







Tip: Problems Finding Files During Linking



Tip: Fixing the memory model

Change the memory model to "data=far"

Project -> Build Options -> [Compiler tab] -> [Advanced category

General Compiler Linker Link Order -g-s-fr"C:\CCStudio_v3.1\Cb000\dsk6713\include" -d"CHIP_6713" -mv6710 -mem_model:data=far Category: Advanced RTS Modifications: Defns No RTS Funcs Advanced Feedback Files Assembly Parser Preprocessor Diagnostics Interrupt Threshold (-mi): Interrupt Threshold (-mi): Speculate Threshold (-mi): Interrupt Threshold (-mi): Interrupt Threshold (-mi): Interrupt Threshold (-mi): Interrupt Threshold (-mi): Speculate Threshold (-mi): Interrupt Threshold (-mi):	Build Options f	or Sine8_LED.pjt (Del	bug)	? 🔀
-Precision of the second secon	General Compil	er Linker Link Order		
Basic RTS Modifications: Defns No RTS Funcs Advanced Auto Inline Threshold (-oi): Ittle Endian Feedback Endianness: Little Endian Assembly Parser Preprocessor Memory Models: Far (-mem_model:data=far) Diagnostics RTS Calls: Far Aggregate Aliasing: Near (-mem_model:data=far) Interrupt Threshold (-mi): Speculate Threshold (-mi): Turn Off Software Pipelining (-mu) Old 6400 Alignment Compatibility (-mb) Turn Off Reorder of Associative Floating Pt Ops (-mc) Use Function Subsections (-mo) Historic C Pointer to Const Alias Disambiguation (-ox)	-i"C:\CCStudio_	v3.1\C6000\dsk6713\inclu		<
	Basic Advanced Feedback Files Assembly Parser Preprocessor	RTS Modifications: Auto Inline Threshold (-oi) Endianness: Memory Models: RTS Calls: Aliasing: Interrupt Threshold (-n Speculate Threshold (-n Speculate Threshold (-n Old 6400 Alignment C Turn Off Reorder of A	Little Endian Far (mem_model: data=far) Far Aggregate Far (mem_model: data=far) Near (
		Historic C Pointer to C	onst Alias Disambiguation (-ox)	
OK Cancel Help		OK	Cancel	lelp







Optional: Suppress Linker Warnings

Project->Build Options [linker tab]

In the Advanced category, uncheck "warn about output sections".

Alternatively, put values for stack and heap in the Basic category.

Build Options	for helloworld.pjt (Debug)	? 🔀
General Com	piler Linker Link Order	
-q -c -m".\Deb	oug\helloworld.map'' -o''.\Debug\helloworld.out'' -x	
Category: Basic Advanced	Advanced Disable Conditional Linking (-j) Disable Debug Symbol Merge (-b) Strip Symbolic Information (-s) Make Global Symbols Static (-h) Warn About Output Sections (-w) Resolve Symbols to First Library (-priority) Disable Size-based Allocation (default_order) XML Link Info File (xml_link_info=): Define Global Symbol (-g): Create Unresolved Ext Symbol (-u):	
	OK Cancel Help	





Things to Try

- Open the Sin8_LED project and fix the search path and the memory model (see previous pages). Then build, load, and run it.
 - Press DIP switch 0. You should see LED 0 light up and a 1kHz sinusoid should appear on the left channel of the codec. This is a good test to see if the DSK is working.
- Make an error in the source code Sin8_LED.c and build the project to see what happens.
- Change the amplitude of the sinusoid (gain variable), rebuild, reload, and see what happens.
- Modify the code to generate a 500Hz sinusoid.
- Open, build, and load other projects in "myprojects"





Creating a New Project (1 of 5)

I. Create new project Project->New

Project Creation	Project Creation 🛛 🔊						
Project <u>N</u> ame:	helloworld						
Location:	CACCOnstants 2 13 McDecise 343 hollows						
	C:\CCStudio_v3.1\MyProjects\hellowo						
<u>P</u> roject Type:	Executable (.out)						
<u>T</u> arget	TMS320C67XX						
	< <u>Back</u> Finish Cancel Help						





Technology for Innovators[™]



Creating a New Project (2 of 5)

- 2. Write your C code: File->New->Source File
- 3. Save it in your project directory (make sure it has a .c extension):
 File->Save
- 4. Add your C code to the project: Project->Add Files to Project







Creating a New Project (3 of 5)

- 5. Add required support files to project Project->Add Files to Project
 - a) myprojects\support\c6713dsk.cmd [linker command file – this or another cmd file is required]
 - b) c6000\cgtools\lib\rts6700.lib [run-time support library functions - required]
- 6. Add optional support files to project, e.g. Project->Add Files to Project
 - a) myprojects\support\vectors_poll.asm or vectors_intr.asm [used to set up interrupt vectors]
 - b) c6000\dsk6713\lib\dsk6713bsl.lib [DSK board support library functions – useful for interfacing to the codec, DIP switches, and LEDs]
 - c) c6000\csl\lib\csl6713.lib [chip support library functions]







Creating a New Project (4 of 5)

- 7. Set up the build options for C6713: Project -> Build Options (compiler tab)
 - Make sure target version is C671x
 - Also make sure Opt(imization) Level is "none" - this will help with debugging

Build Options f	or helloworld.pjt (Debug)	? 🛛		
General Compiler Linker Link Order				
-g -fr''\$(Proj_dir)\Debug'' -d''_DEBUG'' -mv6710				
Category: Basic Advanced Feedback Files Assembly Parser Preprocessor Diagnostics	Basic Target Version: C671x (-mv6710) ▼ Generate Debug Info: Full Symbolic Debug (-g) ▼ Opt Speed vs Size: Speed Most Critical (no -ms) ▼ Opt Level: None Program Level Opt.: None			
	OK Cancel	Help		







Creating a New Project (5 of 5)

- 8. Scan all file dependencies to automatically bring all header files and includes into the project:
 Project -> Scan all file dependencies
- 9. Build the project:
 Project -> Build
- If successful, load the .out file to the DSK:
 File -> Load Program
 Select the Debug directory. Select the .out file.
- II. Run it:

Debug -> Run or F5 or the run button.







A Simple Program to Try: "helloworld"

// helloworld.c// D. Richard Brown III// 19-Oct-2009

#include <stdio.h>

void main()

printf("Hello world.\n");

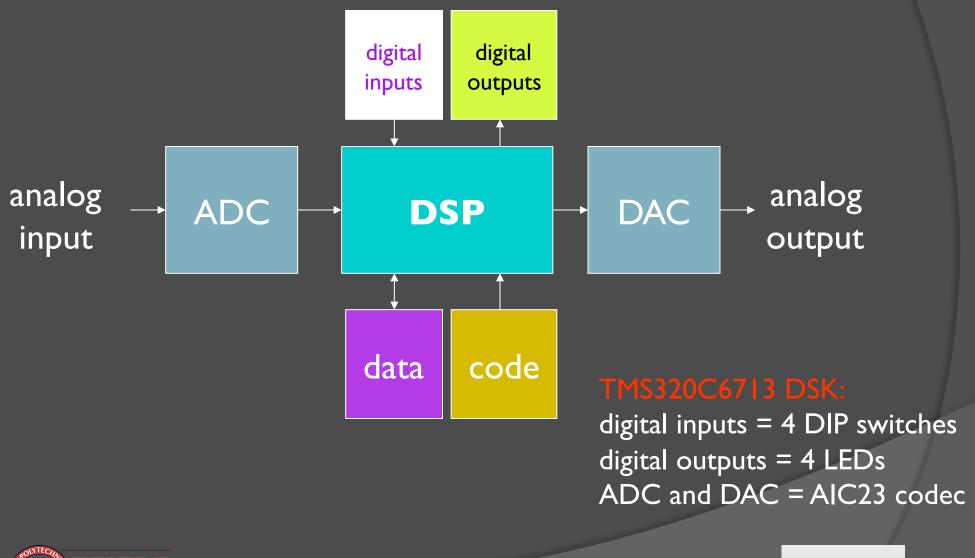




}



More Interesting Programs: Interfacing With the Real World







Interfacing With the DIP Switches and LEDs

LED and DIP switch interface functions are provided in dsk6713bsl.lib.

Initialize DIP/LEDs with DSK6713_DIP_init() and/or DSK6713_LED_init()
Read state of DIP switches with DSK6713_DIP_get(n)
Change state of LEDs with DSK6713_LED_on(n) or DSK6713_LED_off(n) or DSK6713_LED_toggle(n)
where n=0, 1, 2, or 3.

Documentation is available in C:\CCStudio_v3.1\docs\hlp\c6713dsk.hlp







Interfacing With the AIC23 codec: C6x Interrupt Basics

- Interrupt sources must be mapped to interrupt events
 - I6 "interrupt sources" (timers, serial ports, ...)
 - 12 "interrupt events" (INT4 to INT15)
- Interrupt events have associated "interrupt vectors". An "interrupt vector" is a special pointer to the start of the "interrupt service routine" (ISR).
- Interrupt vectors must be set up in your code (usually in the file "vectors.asm").
- You are also responsible for writing the ISR.







Setting Up an Interface With the AIC23 Codec (step I of 3)

We can write the ISR first:

```
49 interrupt void serialPortRcvISR()
50 {
51 Uint32 temp;
52
53 temp = MCBSP_read(DSK6713_AIC23_DATAHANDLE); // read L+R channels
54 MCBSP_write(DSK6713_AIC23_DATAHANDLE,temp); // write L+R channels
55 }
```

Remarks:

- MCBSP_read() requests samples from the codec's ADC
- MCBSP_write() sends samples to the codec's DAC
- This ISR simply reads in samples and then sends them back out.





Codec Data Format and How To Separate the Left/Right Channels

// we can use the union construct in C to have
// the same memory referenced by two different variables
union {Uint32 combo; short channel[2];} temp;

temp.channel[0] (short) temp.channel[1] (short)

temp.combo (Uint32)

// the McBSP functions require that we
// read/write data to/from the Uint32 variable
temp.combo = MCBSP_read(DSK6713_AIC23_DATAHANDLE);
MCBSP_write(DSK6713_AIC23_DATAHANDLE, temp.combo);

// but if we want to access the left/right channels individually
// we can do this through the short variables
Leftchannel = temp.channel[1];
Rightchannel = temp.channel[0];





Technology for Innovators[™]



Setting Up an Interface With the AIC23 Codec (step 2 of 3)

- Now we can set up the interrupt vector to point to the ISR.
- In this example, our ISR is called "serialPortRcvISR".
- We will link the codec interrupt event to INT 5.
- Here is the appropriate code in the vectors.asm file:

150	INT15:	
151	MVKL	.S2 _serialPortRcvISR, B0
152	MVKH	.S2 _serialPortRcvISR, B0
153	В	.S2 B0
154	NOP	
155	NOP	
156	NOP	
157	NOP	
158	NOP	





Setting Up an Interface With the AIC23 Codec (step 3 of 3)

Initialization steps:

- I. Initialize the DSK
- 2. Open the codec with the default configuration.

22 23

24

25

26

27

28

29

30

31

32

33 34

35

36

37

38

39

40

41

42

43

- 3. Configure multichannel buffered serial port (McBSP)
- Configure codec parameters, e.g. set the sampling rate
- 5. Configure and enable interrupts
- 6. Do normal processing (we just enter a loop here)

```
interrupt void serialPortRcvISR(void);
                                                            // ISR function prototype
void main()
£
    DSK6713_init(); // Initialize the board support library, must be called first
   hCodec = DSK6713_AIC23_openCodec(0, &config);
                                                            // Open the codec
   // Configure buffered serial ports for 32 bit operation
   // This allows transfer of both right and left channels in one read/write
   MCBSP_FSETS(SPCR1, RINTM, FRM);
   MCBSP_FSETS(SPCR1, XINTM, FRM);
   MCBSP_FSETS(RCR1, RWDLEN1, 32BIT);
   MCBSP_FSETS(XCR1, XWDLEN1, 32BIT);
    DSK6713_AIC23_setFreg(hCodec, DSK6713_AIC23_FRE0_48KHZ);
                                                               // set the sampling rate
   // Interrupt setup
   IRQ_globalDisable();
                                    // Globally disables interrupts
    IRQ_nmiEnable();
                                    // Enables the NMI interrupt
    IRQ_map(IRQ_EVT_RINT1,15);
                                    // Maps an event to a physical interrupt
                                    // Enables the event
   IRQ_enable(IRQ_EVT_RINT1);
   IRQ_globalEnable();
                                    // Globally enables interrupts
   while(1)
```





Setting the Sampling Rate

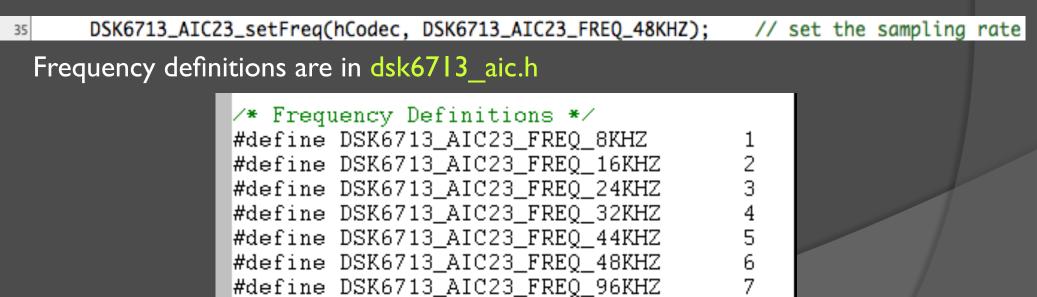
Here we open the codec with the default configuration:

26 hCodec = DSK6713_AIC23_openCodec(0, &config);

// Open the codec

The structure "config" is declared in dsk6713_aic23.h

Rather than editing the header file, we can change the sampling frequency after the initial configuration:









Other Codec Configuration

- Line-level input volume (individually controllable for left and right channels)
- Headphone output volume (individually controllable for left and right channels)
- Digital word size (16, 20, 24, or 32 bit)
- Other settings, e.g. byte order, etc. For more details, see:
 - dsk6713_aic23.h
 - Codec datasheet (TLV320AIC23B)
 - C:\CCStudio_v3.1\docs\hlp\c6713dsk.hlp





Some Things to Try

• Make a new project that:

- Polls DIP switch 0. If pressed, light up all four LEDs.
- Sets the sampling rate of the AIC23 codec to 44.1kHz.
- Uses an ISR to sample the left and right channels.
- Multiplies the left and right channels by a variable gain.
- Outputs the modified samples to the left and right channels.
- Sonus: Swap the channels, i.e. Left_in -> Right_out, Right_in -> Left_out, when DIP switch 0 is pressed.
- Sonus: Try changing the input/output volumes (hint: look at default configuration in dsk6713_aic23.h)







Lunch Break

Workshop resumes at 1:30pm...





Technology for Innovators[™]



Debugging and Other Useful Features of the CCS IDE

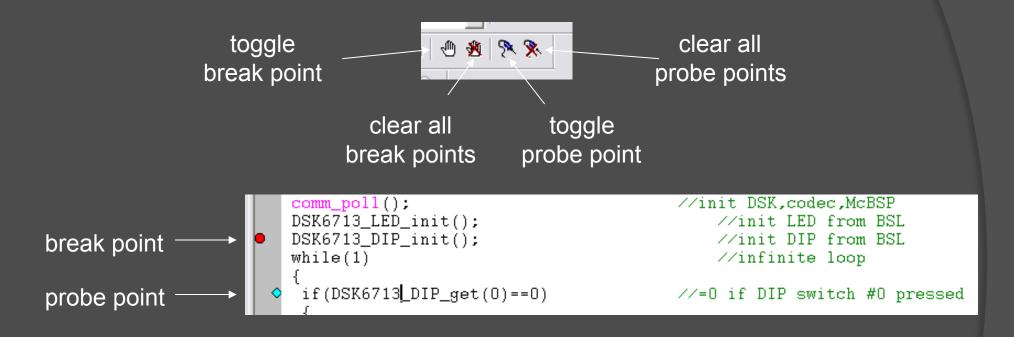
- Breakpoints
- Probe points
- Watch variables
- Plotting arrays of data
- Animation
- General Extension Language (GEL)







Breakpoints and Probe Points



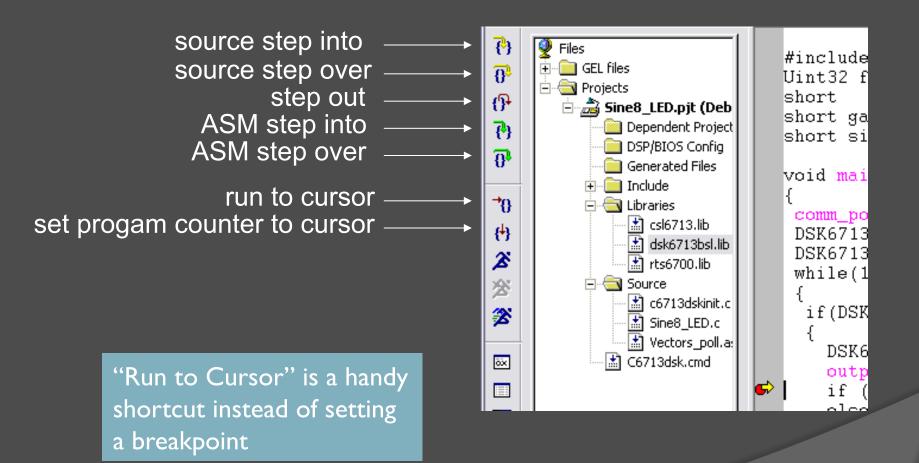
- Breakpoints: stop code execution at this point to allow state examination and step-by-step execution.
- Probe points: force window updates and/or read/write samples from/to a file at a specific point in your code.



🤴 Texas Instruments



Breakpoints



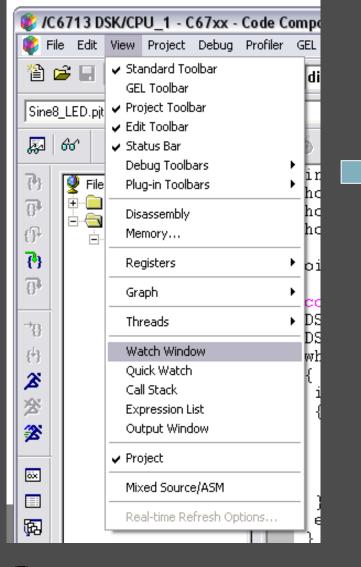


TEXAS INSTRUMENTS

Technology for Innovators[™]



Watch Variables



#3%

g	Profiler	GEL	_								
		di		Name			Value			Туре	Radix
		ir ha ha ha		[latch Locals	eer wa	tch 1				
		e oi	i						Ln 24, Col 1	NUM	
		► <mark>cc</mark> ► DS									
		► DS — DS wł		Name			Value			Туре	Radix
		{	i		loop gain		4			short short	dec dec
					Watch Locals	ස් ස ෘ	itch 1				
Dpl	ions								Ln 24, Col 1	NUM	
	Tex/	as II	NSTRUME	NTS	Technolo	ogy for	Innovators	•		IEE	E

Watch Variables

- In the Watch Locals tab, the debugger automatically displays the Name, Value, and Type of the variables that are *local* to the currently executing function.
- In the Watch tab, the debugger displays the Name, Value, and Type of the local and global variables and expressions that you specify.
- Can add/delete tabs.







Plotting Arrays of Data

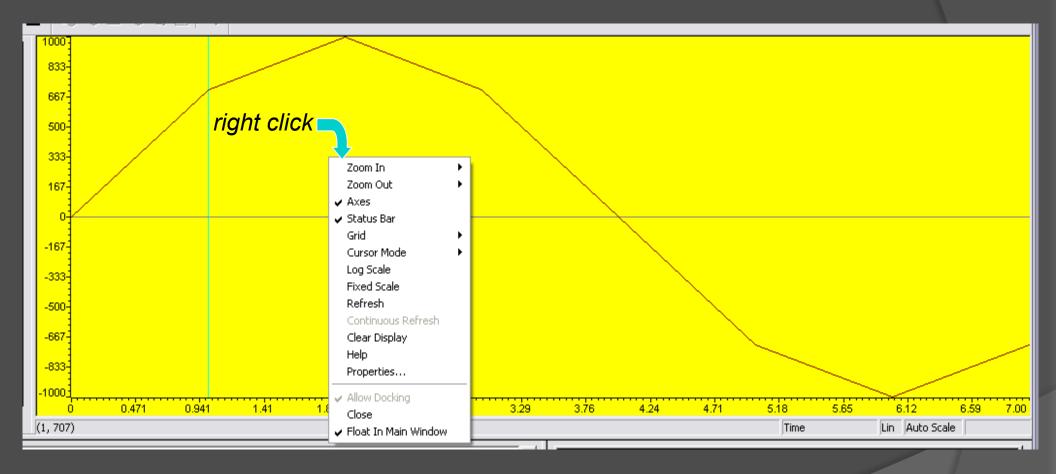
😻 /C6713 DSK/CPU_1 - C67xx - Code Composer Studio 'C6713 D						
🌾 File Edit	View Project Debug Profiler	GEL Option Tools PBC I				
Image: Sine 8_LED.pit Sine 8_LED.pit Image: Sine 8_LED.pit Image	 Standard Toolbar GEL Toolbar Project Toolbar Edit Toolbar Status Bar Debug Toolbars Plug-in Toolbars Disassembly Memory 	dip				
(4)	Registers	<pre>> oid main()</pre>				
- 194 - 1	Graph	Time/Frequency				
→ <u>0</u>	Threads	Constellation I Eye Diagram				
{ ¹ }	Watch Window Quick Watch	Image				
X I	Call Stack	{ if(DSK6713_DIP]				
谷 ※ ※	Expression List	11 (DSK6/15_DIF				
38	Output Window	DSK6713_LED_				
	🗸 Project	if (loop < 7				
	Mixed Source/ASM	else loop =				
	Real-time Refresh Options	- } else DSK6713 Li				
<mark>ゐ</mark>						

💀 Graph Property Dialog	×		
Display Type	Single Time		
Graph Title	Graphical Display		
Start Address	sine_table		
Acquisition Buffer Size	8		
Index Increment	1		
Display Data Size	8		
DSP Data Type	16-bit signed integer		
Q-value	0		
Sampling Rate (Hz)	8000		
Plot Data From	Left to Right		
Left-shifted Data Display	Yes		
Autoscale	On		
DC Value	0		
Axes Display	On		
Time Display Unit	\$		
Status Bar Display	On		
Magnitude Display Scale	Linear		
Data Plot Style	Line		
Grid Style	Zero Line		
Cursor Mode	Data Cursor		
	<u> </u>		





Graph Windows: Plotting Arrays of Data





TEXAS INSTRUMENTS

Technology for Innovators[™]



Probe Points

- Differ from breakpoints: Halt the DSP momentarily, perform an action, and then automatically resume execution.
- Some useful functions of probe points:
 - Connect probe point to graph window.
 - Connect probe point to file I/O to facilitate repeatable testing.
 - Note that probe points cause problems with real-time operation.
- For more details, see CCS Getting Started Guide (SPRU509F.PDF) or CCS help.







Using a Probe Point for File I/O

Basic idea:

- Create an input data file with the signals you want to use to test your code.
- Place a probe point in your code to read the contents of datafile into an array in the DSK memory.
- Place another probe point in your code to write the results (stored in an array on the DSK) to a datafile on the computer.







Example Matlab code to generate input data (noisy sinusoid)

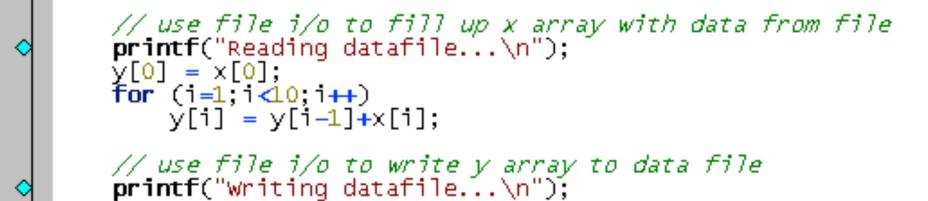
```
fs = 44100;
t = 0:1/fs:1;
f = 1000;
x = 0.5*sin(2*pi*f*t) + 0.1*randn(1,length(t));
fid = fopen('inputdata.dat','wt');
fprintf(fid,'1651 4 000 0 801\n');
fprintf(fid,'%f\n',x);
fclose(fid);
```







Setting Probe Points in CCS







Technology for Innovators[™]



Connecting Probe Points to File I/O File -> File I/O

File I/O		
File Input File Output		
C:\Documents and Settings\Administrator\Desktc Add File	Break/Probe Points	
Remove File	Breakpoints Probe Points	
. 🔲 Wrap Around	Probe type: Probe at Location	Add
	Location:	Replace
Probe Point: Connected	Count: 1	View Location
Address: x Length: 0x0a Add Probe Point	Expression:	
	Connect To: No Connection	
OK Cancel Apply Help	Probe Point: Stereoloop.c line 52 (???) -> FILE IN:C:\\data.txt	Delete
File I/O	▼stereoloop.c line 58 (???)> FILE OUT:C:\\Desktop\a.out	Enable All
File Input File Output		Disable All
		Delete All
C:\Documents and Settings\Administrator\Desktc Add File		
Remove File		
	OK Cancel Apply	
Probe Point: Connected		
Address: y		
Length: 0x0a Add Probe Point	See SPRU509 for more	details.
OK Cancel Apply Help		
		EEE
TEXAS INSTRUMENTS	Technology for Innovators [™]	

Using a Probe Point to Update a Graph Window

- First create a graph using View->Graph
- Then go to Debug->Probe Points to connect a probe point to the graph Break/Probe Points

Break/Probe Poi	nts					
Breakpoints Probe Points						
Probe type: Location:	Probe at Location	Add				
Count:	1	Replace View Location				
Expression: Connect To:	No Connection					
Probe Point:	Watch Window FILE IN:C:\\data.txt FILE OUT:C:\\Desktop\a.out	Delete				
v stereoloop.c lir	Graphical Display Corr.c Desktop ta.out	Enable All				
		Disable All				
		Delete All				
	OK Cancel Apply	Help				





Animation



Runs the program until a breakpoint is encountered

- At the breakpoint, execution stops and all windows not connected to any Probe Points are updated.
- Program execution then automatically resumes
- Useful for updating graphical displays
- Note: Animation will cause problems with realtime operation
- Can pause execution at each breakpoint: Animate Speed (0-9s) (zero = no pause)



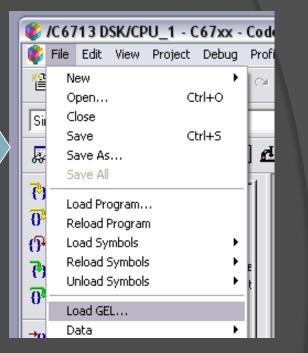


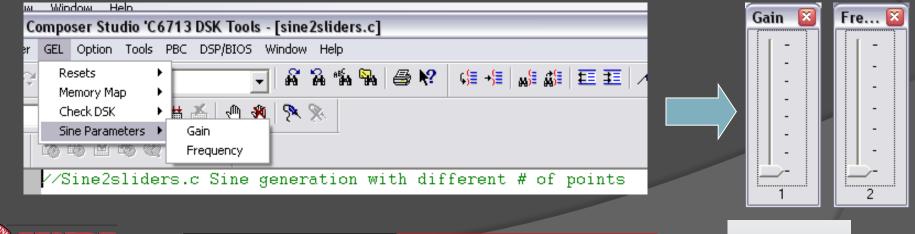


General Extension Language

- Create functions to extend the functionality of Code Composer Studio
- GEL files are not loaded with a project
- Often used to change variables "on-the-fly"
- Examples from Chassaing textbook: sin2sliders.pjt and sin2sliders.gel

- its





General Extension Language

- Useful GEL files can be pretty simple
- From sin2sliders.gel:

```
/*Sine2sliders.gel Two sliders to vary gain and frequency*/
menuitem "Sine Parameters"
slider Gain(1,8,1,1,gain_parameter) /*incr by 1,up to 8*/
{
    gain = gain_parameter; /*vary gain*/
}
slider Frequency(2,8,2,2,frequency_parameter) /*incr by 2,up to 8*/
{
    frequency = frequency_parameter; /*vary frequency*/
}
```

 Syntax details can be found in CCS help: Help->Contents->Making a Code Composer Studio Project -> Building and Running your Project -> Automating Tasks with General Extension Language (GEL)





Some Things to Try

- Try out the debugging tools on the code you wrote in the morning session
 - breakpoints
 - watch variables
 - step into, step over, step out
- Try out the CCS plotting tools
 - Modify your code to have a buffer (i.e., store samples in an array) and plot the contents in a graph window
- Try out file I/O with probe points and/or updating a graph with probe points
- Try to have CCS animate a plot window via probe points and/or animation
- Modify your stereo in/out project to have the output gain changeable via a GEL slider







Finite Impulse Response (FIR) Filters

• Frequently used in real-time DSP systems

- Simple to implement
- Guaranteed to be stable
- Can have nice properties, e.g. linear phase
- Input/output relationship

$$y[n] = \sum_{m=0}^{M-1} h[m]x[n-m]$$

x = input, y = output, h = filter coefficients, M = # of filter coefficients



🤣 Texas Instruments



Creating FIR Filters

I. Design filter

- Type: low pass, high pass, band pass, band stop, ...
- Filter order M
- Desired frequency response
- 2. Decide on a realization structure
- 3. Decide how coefficients will be quantized.
- 4. Compute quantized coefficients
- 5. Decide how everything else will be quantized (input samples, output samples, result of multiplies, result of additions)
- 6. Write code to realize filter

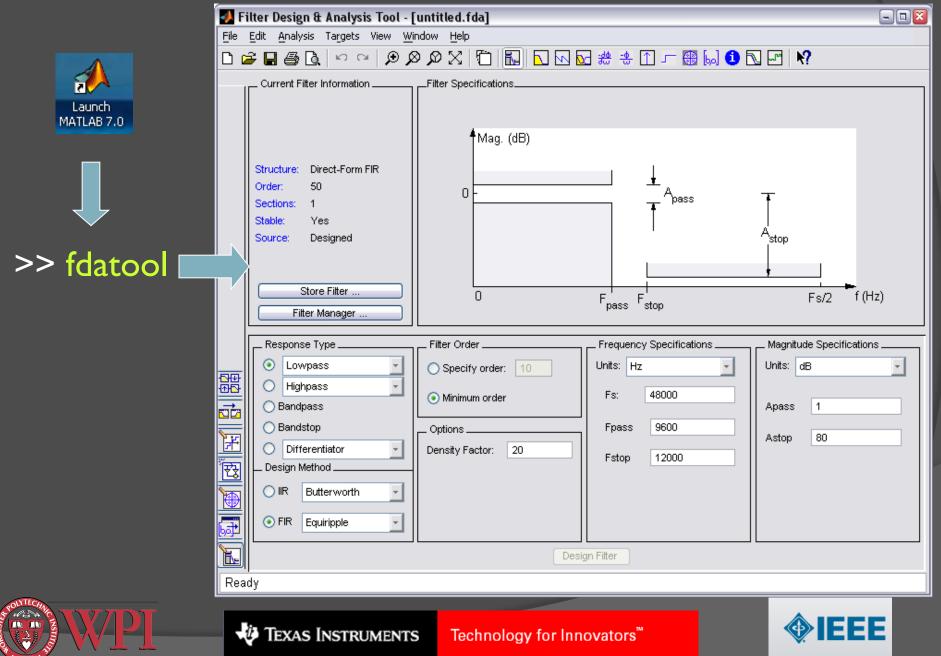
- CCS
- 7. Test filter and compare to theoretical expectations





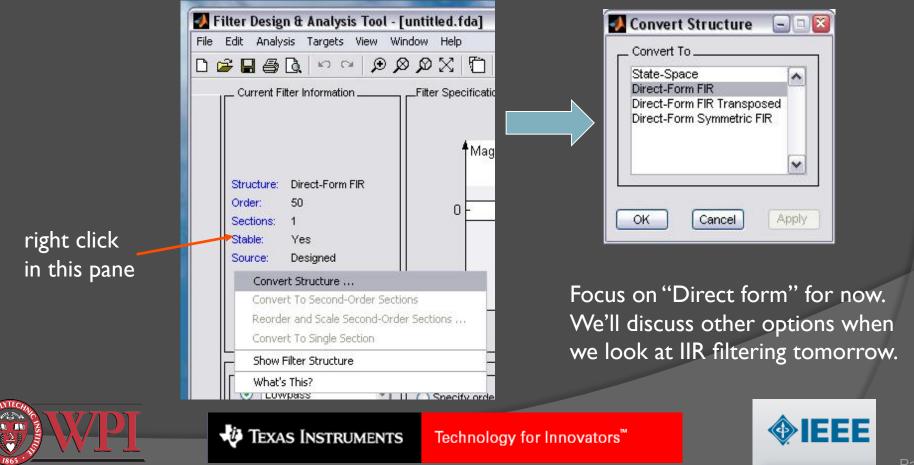


Designing FIR Filters



Filter Realization Structures

- Lots of different structures available
 - Direct form I, direct form II, transposed forms, cascade, parallel, lattice, ...
 - All have same input/output relationship
 - Choice of structure affects computational complexity and how quantization errors are manifested through the filter

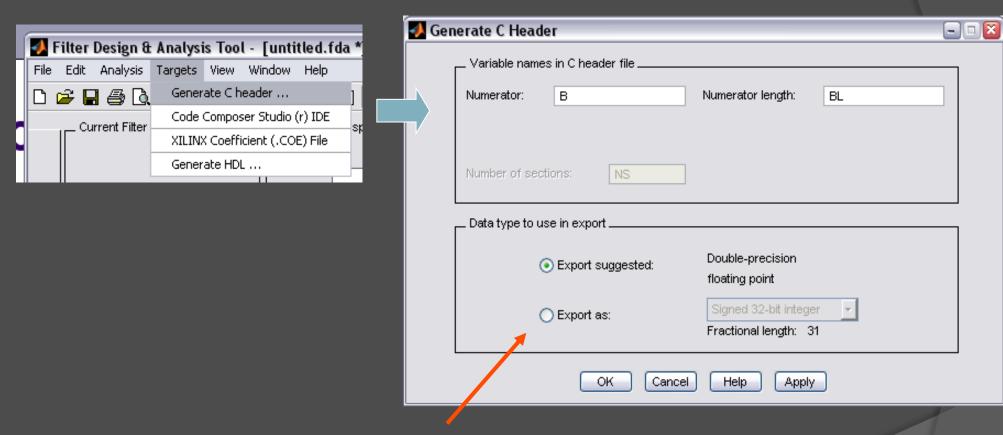


Compute FIR Filter Coefficients

🛃 Filter Design & Analysis Tool - [untitled.fda] 📃 🗆 🖾						
File Edit Analysis Targets View Window Help						
D 🚘 🖬 🚑 Q, 🗠 ལ 🗩 🖉 🖸 🔚 🔂 🖸 🖸 🖸 🔂 🖉 🕅 🖂 🖓 🖉 🔛 🕅						
	mation Filter Specifications					
	Form FIR eed er	▲ Apass ↑ Fpass Fstop	A _{stop} Fs/2 f (Hz)			
Response Type - O Lowpass Highpass Bandpass Bandstop Differentiato Design Method - O IIR Butterw FIR Equiripp	r T	Frequency Specifications	Magnitude Specifications			
TEXAS INSTRUMENTS Technology for Innovators [™]						

Page 60 of 6

Make Coefficient File For CCS



Here you can change the coefficient data type to match your desired quantization.







Example DP-FP Coefficient File

```
/*
 * Filter Coefficients (C Source) generated by the Filter Design and Analysis Tool
  Generated by MATLAB(R) 7.0 and the
 *
  Generated on: 19-Aug-2005 13:04:09
 *
 */
/*
 * Discrete-Time FIR Filter (real)
 * Filter Structure : Direct-Form FIR
 * Filter Order
                      : 8
 * Stable
                     : Yes
 * Linear Phase
                   : Yes (Type 1)
 */
/* General type conversion for MATLAB generated C-code */
#include "tmwtypes.h"
/*
 * Expected path to tmwtypes.h
 * C:\MATLAB7\extern\include\tmwtypes.h
 */
const int BL = 9;
const real64 T B[9] = {
                                                                0.2017873498839,
    0.02588139692752,
                       0.08678803067191,
                                            0.1518399865268,
     0.2205226777929,
                        0.2017873498839,
                                            0.1518399865268,
                                                               0.08678803067191,
    0.02588139692752
};
                                                                                   EEF
                        TEXAS INSTRUMENTS
                                              Technology for Innovators<sup>™</sup>
```

Quantization Considerations

- Sey choice: floating point vs. fixed point
- Advantages of floating point math:
 - Less quantization error
 - Don't have to worry about scaling factors
 - Less likelihood of overflow/underflow
 - Much easier to code
- Oisadvantages of floating point math:
 - Requires floating point DSP (higher cost, higher power)
 - Executes slower than fixed point
- C code allows you to "cast" variables into any datatype





Write Code to Realize FIR Filter

 Direct form I implies direct realization of the convolution equation

$$y[n] = \sum_{m=0}^{M-1} h[m]x[n-m]$$

- Some considerations:
 - Allocate buffer of length M for input samples.
 - Move input buffer pointer as new data comes in or move data?







FIR Filter Example Code

```
interrupt void serialPortRcvISR()
```

```
union {Uint32 combo; short channel[2];} temp;
int i = 0;
float result = 0.0;
```

```
temp.combo = MCBSP_read(DSK6713_AIC23_DATAHANDLE);
```

```
// Update array samples (move data - this is the slow way)
for( i = N-1; i >= 1; i-- )
        samples[i] = samples[i-1];
samples[0] = (float)temp.channel[0]; // store right channel
```

Note that all math here is floating point. Filter coefficients are also assumed to be floating point.







Some Things to Try

Try creating an FIR filter with the following specs: \bigcirc

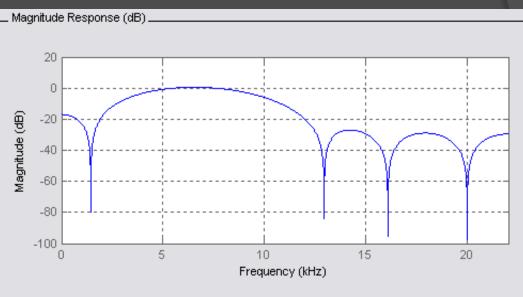
- Bandpass
- 8th order Direct Form I
- Least-squares design
- 44100Hz sampling rate
- FstopI = 3000Hz
- $F_{passI} = 4000Hz$
- Fpass2 = 8000Hz
- Fstop2 = I2000Hz
- Equal weighting in all bands
- All floating point math (single or double precision)
- Use an oscilloscope and a function generator to compare the magnitude response of your filter to the theoretical prediction.







Page 66 of 68



Workshop Day I Summary

What you learned today:

- Basics of the TMS320C6713 DSK and Code Composer Studio
- How to test the DSK
- How to open, build, load, and run existing projects
- How to create, build, load, and run new projects
- How to interface with DSK I/O (LEDs, DIP switches, and the AIC23 codec)
- How to debug code in CCS including
 - Setting and clearing breakpoints and probe points
 - Setting up watch variables
 - Plotting arrays of data
 - Animation
- How to use, modify, and create GEL files in CCS.
- How to use Matlab's filter design/analysis tool "fdatool"
- How to implement an FIR filter on the C6713







Workshop Day I Reference Material

- Chassaing and Reay textbook Chapters 1-2, and 4
- CCS Help system
- SPRU509F.PDF CCS v3.1 IDE Getting Started Guide
- C6713DSK.HLP C6713 DSK specific help material
- AIC23 Codec datasheet
- DSK Quick Start Guide (included in your DSK box)
- Spectrum Digital TMS320C6713 DSK reference (included in your DSK box)
- TMS320C6000 Programmer's Guide (SPRU198G.PDF)
- Matlab fdatool help (>> doc fdatool)
- Detailed CCS IDE and DSK drivers install guide at http://spinlab.wpi.edu/teaching.html

Latest TI documentation available at http://www.ti.com/sc/docs/psheets/man_dsp.htm





