Lab Session on

Speech Processing using DSP TMS320C6713

8th July, 2014

Prof Bhupendra Fatania
Prof Amit Degada

STTP on Multimedia Signal Processing Theory and Applications, 30th June to 11th July, 2014
Institute of Technology, Nirma University, Ahmedabad

Electrical Engineering Department
Electronics & Communication Engineering Program
Institute of Technology,
Nirma University
Ahmedabad-382481
Outline

• Overview and Feature of TMS320C6713
• Applications
• Code Composer Studio CCS V3.1
• Demo 1 : Generation of Sine Wave
• Demo 2 : Creating an FIR Filter
Overview

• What is the need of DSP Chip?

• Major Companies: TI, Motorola, Analog Devices, Lucent Technologies etc?

Rapid Growth due to Fast, Flexible and low cost Solution made it to be used in wide variety of Applications
TI C6713

- Running at 225 MHz
- 4 I/O (mic in, line in, line out, headphone)
- 8k cache L1
- 256k cache L2
- 16M of SDRAM
- Floating point processor
- AIC23 -> stereo codec for I/O
- Sampling rate from 8 to 96kHz
TMS320C6713 DSK Hardware

- Mic In
- Line In
- Headphones
- Expansion
- C6713 CPU

- Codec
- Line Out

- +5V Supply
- USB
- JTAG
- LEDs
- RAM
- Switches
TMS320C6713 DSK

• This board plugs into the computer using a USB cable.
• There are stereo audio inputs for line (CD player / IPOD etc).
• There are stereo audio outputs for headphones or computer speakers.
• Input switches are provided to allow selection of up to 16 different settings.
• There are 4 LEDs which can be used as a bargraph.
TMS320C6713 DSK KIT
Typical C6713 DSK Setup

- USB to PC
- Headphones
- to +5V
- Microphone
TI C6713

- VLIW architecture
- Up to 8 instructions / cc
- 2 completely independent data paths
- 2 x 32 registers
- Connected to PC with an USB cable
AIC23

- Accept inputs of 6V p-p
- Supported sample rates:
  - 8, 16, 24, 32, 44.1, 48, 96 (KHz)
- Data length:
  - 16, 20, 24, 32 (bit)
- Automatically performs:
  - ADC, DAC, lowpass filtering, oversampling
- Multi-Channel Buffered Serial Port
TI Libraries

TI Software Foundation Libraries

Board Support Library (BSL)
- Board-level routines supporting DSK-specific hardware
- Higher level of abstraction than CSL
- BSL functions make use of CSL

Chip Support Library (CSL)
- Low-level routines supporting on-chip peripherals
  - Serial Ports
  - EDMA
  - EMIF
  - Cache
  - Timers
  - Etc.

CSL helps with:
1. Configure Peripherals
2. Managing Multiple Resources (e.g. McBSP channels)
Using AIC23 in your program

• Add “dsk6713bsl.lib” to your project
• Add CHIP_6713 to the preprocessor symbol list
  • Need to add it in Project Build Options

• Include “dsk6713.h” and “dsk6713_aic23.h”
  • Located in c6000/dsk6713/(include or lib)
Using AIC23 in your program

• Configure and create a codec handle

```c
DSK6713_AIC23_Config config = {
    0x0017, /* 0 DSK6713_AIC23_LEFTINVOL Left line input channel volume */
    0x0017, /* 1 DSK6713_AIC23_RIGHTINVOL Right line input channel volume */
    0x01f9, /* 2 DSK6713_AIC23_LEFTHPVOL Left channel headphone volume */
    0x01f9, /* 3 DSK6713_AIC23_RIGHTHPVOL Right channel headphone volume */
    0x0011, /* 4 DSK6713_AIC23_ANAPATH Analog audio path control */
    0x0000, /* 5 DSK6713_AIC23_DIGPATH Digital audio path control */
    0x0000, /* 6 DSK6713_AIC23_POWERDOWN Power down control */
    0x0043, /* 7 DSK6713_AIC23_DIGIF Digital audio interface format */
    0x0081, /* 8 DSK6713_AIC23_SAMPLERATE Sample rate control */
    0x0001 /* 9 DSK6713_AIC23_DIGACT Digital interface activation */
};
DSK6713_AIC23_CodecHandle hCodec;
```
Code Composer Studio

Diagram showing the workflow of the Code Composer Studio, including:
- Editor
- Compiler
- Asm Optimizer
- Asm
- Linker
- Link.cmd
- .map
- .out
- .obj
- .asm
- .sa
- .c/.cpp
What is a Project?

Project (.PJT) file contain:

References to files:
- Source
- Libraries
- Linker, etc …

Project settings:
- Compiler Options
- DSP/BIOS
- Linking, etc …

Let’s look more closely at Build Options and Configurations …
CCS Steps
Step 1: Kit Diagnosis. Check Idle and then Pass
Step 2: Kit Connection verification in CCS

Note: First Close C6713 Kit Diagnosis
Step 3: Create a Project

Note: Always Choose the saving Directory C:\CCStudio_V3.1\myprojects\
Step 4: Project Name, Location
Step 5: Source File

Note: Add Source File to write the main Code. Save as .c file
Code to Generate Sine Wave

Code
Step 6: Add File to Project
Step 7: Add C6713dskint.c file

Note: C:\CCStudio_V3.1\myprojects\support... Select Project ----> Add files

C6713dskinit.c file contains the function definitions for a low level routines including comm_poll() and output_left_sample().
Step 8: Add vectors_poll.asm file

Note: C:\CCStudio_V3.1\myprojects\support... Select Project ----> Add files

Vectors_poll.asm includes interrupt service table for C6713.
Step 9: Add library support files rts6700.lib

Note: C:\CCStudio_v3.1\c6000\cgtools\lib... Select Project ----> Add files

This is a run support file for C67xx architecture.
Step 10: Add library support files dsk6713bsl.lib

Note: C:\CCStudio_v3.1\c6000\dsk6713\lib... Select Project ----> Add files

This is a board support file for C6713 DSK (DSP Supported Kit).
Step 11: Add library support files csl6713.lib

Note: C:\CCStudio_v3.1\c6000\csl\lib... Select Project ----> Add files

This is a chip support file for C6713 Processor.
Step 12: Add linker command file c6713dsk.cmd

Note: C:\CCStudio_v3.1\c6000\cs\lib... Select Project ----> Add files

C6713.cmd is linker command file to create final.out file to load into Kit.
Step 13: Check Project layout

Note: It's without header files
Step 14: Add Header Files

Note: These are header file for the lib files added earlier
Step 15: Check Header files in Include folder

Note: These are header file for the lib files added earlier
Step 16: Check Header files in Include folder

Note: These are header file for the lib files added earlier
Step 17: Set Build options for Compiler

Note: The Code generation tool underlying CCS are Compiler, Assembler and Linker. Proper options need to be set before using it.
Step 18: Set Build options for Compiler....Basic

Note: It sets the target version to C671x.
Step 19: Set Build options for Compiler. Advanced

Note: It sets the and memory models to debug
Step 20: Set Build options for Compiler...

Preprocessor

Note: It sets predefined symbol to CHIP_6713 and search path for library
Step 21: Set Build options for Linker

Note: It sets the and memory models to debug
Step 22: Compile main source.c file

Note: It is to compile all the C file to create .asm file
Step 24: Load the Program

Note: It is to load the .out file in kit.
Step 25: Run the Project on Kit

Note: It is to Run the final .out file on kit
Add the GEL file to control the Gain using slider

Note: GEL stands for General Extension Language. It is similar to C language allows us to change the value of variable while processor is Running

Right click on GEL file ----> Add files. Write following code.

Menuitem “Sine Gain”

Slider Gain (0, 30, 4, 1, gain_parameter) /*incr by 4, up to 30 */
{
    gain = gain_parameter; /* Vary gain of Sine */
}
Thanks

Give Your Feedbacks at:
www.amitdegada.weebly.com/blog.html