Lab 1
Introduction to TI's TMS320C6713 DSK Digital Signal Processing Board *

This laboratory introduces you to the TMS320C6713 DSK board module with:

- An overview of the functional blocks of the board
- Code Composer Studio (CCS)
- Writing, compiling, and running a simple program
- Learning how to generate a tone

1 Introduction

The TI's TMS320C6713 DSK is designed and optimized to perform digital signal processing operations. For short this DSP will be referred to as 'C6713'. The family of this DSP is referred to as 'C6x' or 'C6000'. 'C6713' is a high performance 32-bit floating-point DSP.

The basic operation in digital signal processing is solving the following equation:

$$y[n] = \sum_{k=0}^{M} b_k x[n-k] - \sum_{k=1}^{N} a_k y[n-k]$$

The DSP has to be able to perform the above operation very efficiently and very fast. For example, for a 100-tap FIR filter, where $M = 99$ and $N = 0$, the DSP will have to be able to store 99 samples, and perform 100 multiplication and 100 summation operations between every two samples. The DSP implements multiply-accumulate (MAC) hardware with circular addressing capabilities very efficiently.

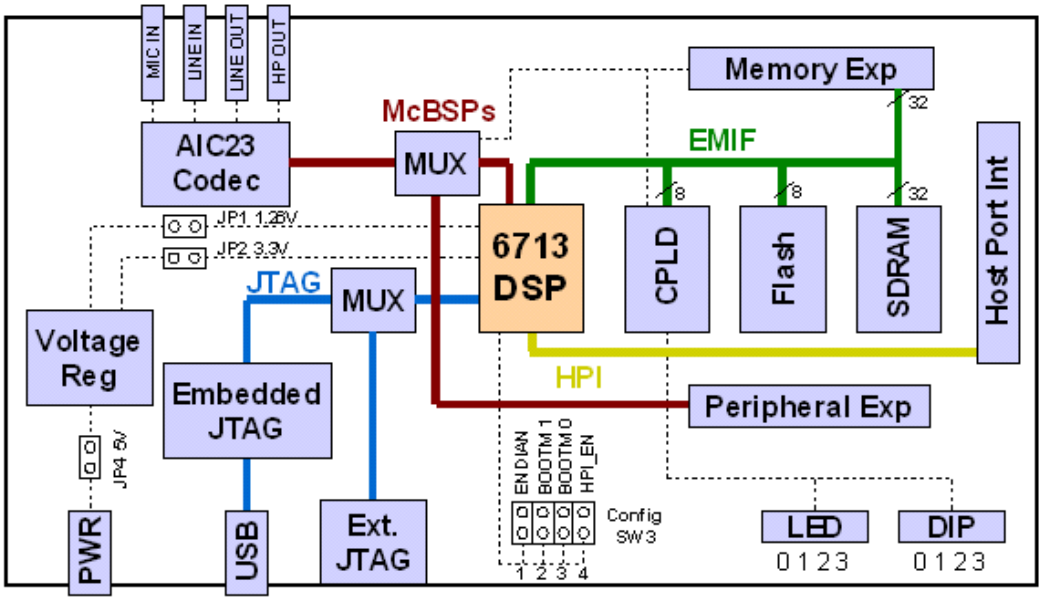
*This lab is based on previous labs written by Drs. Aly El-Osery and Hector Erives.

Some of the TMS320C6713DSK features shown in Figure 1 are

- 225 MHz TMS320C6713 Floating Point DSP
- AIC23 Stereo Codec 8KHz-96KHz sample rate, 16 to 32 bit samples, MIC IN, LINE-IN, LINE-OUT, and HP-OUT (headphone output) jacks.
- Four position user DIP Switches and four LEDs
- 1800 million instructions per second (MIPS) and 1350 MFLOPS
- Software board configuration through registers implemented in CPLD
- Standard expansion connectors for daughter card
- The maximum allowable input signal level at the LINE IN is 1 Vrms. However, the C6713 has a voltage divider circuit with a gain of 0.5 which allows for a maximum of 2 Vrms. Above this level, input signals will be distorted.

TI's Code Composer Studio is a development tool that will be used to program the DSP. The version that comes with the board includes a chip support library (CSL) and a board support library (BSL). The chip support library (CSL) provides a C-language interface for configuring and controlling on-chip peripherals. The BSL provides a C-language interface for configuring and controlling all on-board devices.

Figure 1: Key features of the C6713 DSK



2 Lab

1. Connect the board to power.
2. Start CCS and begin a new project. Create a new project using **Project** → **New**, then input the name of the project, location of the project, type of project (executable), and the target will be the TMS320C67 board. An executable file (*.out) will be created in the ./Debug directory and this is the file that will be downloaded onto the board. When done press finish.
3. Create a new DSP/BIOS configuration file (rename it as `YourProgram.cdb`) and add it to the project along with `YourProgram.cmd` file, using **Project** → **Add Files to Project**
   (a) File + New + DSP/BIOS Configuration...
   (b) Select `dsk6713.cdb`
   (c) File + Save As `YourProgram.cdb` (in place of `Config1.cdb`)
   (d) Project + Add Files to Project (locate the file `YourProgram.cdb` just created, searching it by the suffix `.cdb`)
4. Add the file `YourProgram.cmd` to the project (Project + Add Files to Project) searching it by the suffix `.cmd`
5. Under **Project/Build options**
   (a) Under **Compiler Options**, in the Basic category set Target Version to C671x, and in the Preprocessor category set Pre-Define Symbol to CHIP_6713. Set the Include Search Path to where the
   ```
   ...\C6000\dsk6713\include
   ```
   directory is in your computer (if it is not already set).
   (b) under **Linker Options**, the Basic Category set the Output File name to
   ```
   .\Debug\YourProgram.out
   ```
   , the Library Search Path to the
   ```
   ...\C6000\dsk6713\lib
   ```
   directory (if it is not already set) and include Libraries to `rts6700.lib`, `dsk6713bsl.lib`; `cs6713.lib`.
6. Discretize the continuous-time sine wave given by
   $$y(t) = \sin 2\pi f_0 t$$
   Assume a sampling rate of $f_s = 8$kHz, and $f_0 = 1$kHz.
7. Write a code to generate the sine wave. Make sure that you will keep the angle from overflowing. Below is a sample file that might be useful, **Program 1**. Make sure you replace the include file `labcfg.h` with the name of your configuration file and attach to it cfg.h (i.e., `YourProgramcfg.h`). The two `DSK6713 AIC23 write(hCodec,data)` are used to write to the left and right channel. Later this will be modified so that you can write to both channels with one command.

8. Compile (from **Project**), load (from **File** → **load program** `YourProgram.out`) and run the code (from **Debug**).

9. Connect your headphones or speakers to the output of the codec. Do you hear a tone ? if not, what do you think the problem is ?

10. Use the watch window to change the frequency to 500Hz, to do so you will need to declare the frequency variable as `volatile`. Then highlight the frequency variable, right-click and select `Add to Watch Window`. The option `volatile` tells the DSP to reload the frequency every time it is needed.

Program 1

```
/* Created by Aly ElOsery (Last modified : 08/27/2010) */

#include labcfg.h
#include dsk6713 . h
#include dsk6713_aic23.h
#include <math . h>

/* Codec  configuration settings */
DSK6713 AIC23 Config config = { \
0 x0017 ,   /* 0 LEFTINVOL   Left  line  input  channel volume */ \
0 x0017 ,   /* 1 RIGHTINVOL  Right  line  input  channel  volume */ \
0 x01f9 ,   /* 2 LEFTHPVOL   Left channel  headphone  volume */ \
0 x01f9 ,   /* 3 RIGHTHPVOL  Right channel  headphone  volume */ \
0 x0011 ,   /* 4 ANAPATH     Analog audio path control */
0 x0000 ,   /* 5 DIGPATH     Digital audio path control */ \
0 x0000 ,   /* 6 POWERDOWN   Power down  control */ \
0 x0043 ,   /* 7 DIGIF       Digital audio interface format */ \
0 x0081 ,   /* 8 SAMPLERATE  Sample rate control */ \
0 x0001     /* 9 DIGACT      Digital interface activation */\
};

/* YOU MAY NEED SOME MORE CODE HERE */

void main ()
{
DSK6713 AIC23 CodecHandle  hCodec ;
Uint16 data ;

/* YOU MAY NEED ADDITIONAL DECLARATIONS HERE */

/* Initialize the board support library, must be called first */
DSK6713 init ( ) ;

/* Start  the  codec */
hCodec = DSK6713_ AIC23_openCodec ( 0 , &c o n f i g ) ;
DSK6713_AIC23 set_Freq ( hCodec , DSK6713 AIC23 FREQ 8KHZ ) ;

for(;;)
{

/* YOU MAY NEED SOME MORE CODE HERE */

/* Send a sample to the left channel */
while(! DSK6713 AIC23 write(hCodec,data)) ;

/* Send a sample to the right channel */
while(!DSK6713 AIC23 write(hCodec,data)) ;

/* YOU MAY NEED SOME MORE CODE HERE */
   }
}
```