Lab1
Introduction to TI's TMS320C6713 DSK Digital Signal Processing Board

Dr. Aly El-Osery

August 31, 2011

The purpose of this lab is to start getting familiar with TI's TMS320C6713 by learning the following.

- An overview of the functional blocks of the board.
- Code Composer Studio (CCS).
- Writing, compiling and running a simple code.
- Learn how to generate a tone.
- Learn how to read the DIP switches.

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.

Some of the TMS320C6713DSK features are

- 225 MHz TMS320C6713 Floating Point DSP

- AIC23 Stereo Codec 8KHz-96KHz sample rate, 16 to 32 bit samples, mic, line-in, line-out and speaker jacks
- Four position user DIP Switches and LEDs
- 1800 million instructions per second (MIPs) and 1350 MFLOPS.

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.

2 Lab

1. Connect the board to power.
2. Start CCS and begin a new project.
3. Create an new DSP/BIOS configuration file and add it to the project along with the `.cmd` file.
4. Under **Project/Build options**, make sure that you add the path `C6000/dsk6713/include`, `C6000/dsk6713/lib` and include the library `dsk6713bsl.lib`.
5. Discretize the continuous-time sine wave given by

   $$y(t) = \sin 2\pi f_0 t.$$

   Assume a sampling rate of $f_s = 8\text{kHz}$, and $f_0 = 1\text{kHz}$.
6. Write a code to generate the sine wave. Make sure that you will keep the angle from overflowing. Here 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`. 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.
7. Compile, load and run the code.
8. 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?
9. 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 Code parts, sample.c

```
/*
 *  Created by Aly El-Osery
 *  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 = { \
    0x0017,  /* 0 LEFTINVOL  Left line input channel volume */ \
    0x0017,  /* 1 RIGHTINVOL Right line input channel volume */\
    0x01f9,  /* 2 LEFTHPVOL  Left channel headphone volume */  \
    0x01f9,  /* 3 RIGHTHPVOL Right channel headphone volume */ \
    0x0011,  /* 4 ANAPATH    Analog audio path control */      \
    0x0000,  /* 5 DIGPATH    Digital audio path control */     \
    0x0000,  /* 6 POWERDOWN  Power down control */             \
    0x0043,  /* 7 DIGIF      Digital audio interface format */ \
    0x0081,  /* 8 SAMPLERATE Sample rate control */            \
    0x0001   /* 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, &config);
    DSK6713_AIC23_setFreq(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 */
    }
}
```