EE 451 – LAB 1

Introduction to Texas Instruments TMS320C6713 DSP Starter Kit (DSK) digital signal processing board

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

- 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.

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.

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

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, the DSP will have to be able to store 99 samples, and perform 100 multiplication and 100 summation operations between every two samples. Therefore the DSP implements multiply-accumulate (*MAC*) hardware with circular addressing capabilities very efficiently.

The Texas Instruments TMS320C6713DSK operates at 225MHz and has the following features:

- 16 Mbytes of synchronous DRAM on a 32-bit External Memory Interface (EMIF).
- 512 Kbytes, 8-bit interface (256Kb usable).
- An AIC23 stereo codec (coder /decoder) with 12 MHz system clock.
- 4 user LEDs and DIP switches through a Complex Programmable Logic Device (CPLD).
- Software board configuration through registers implemented in CPLD.

• Standard expansion connectors for daughter card.

These features are shown in Figure 1.

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 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.

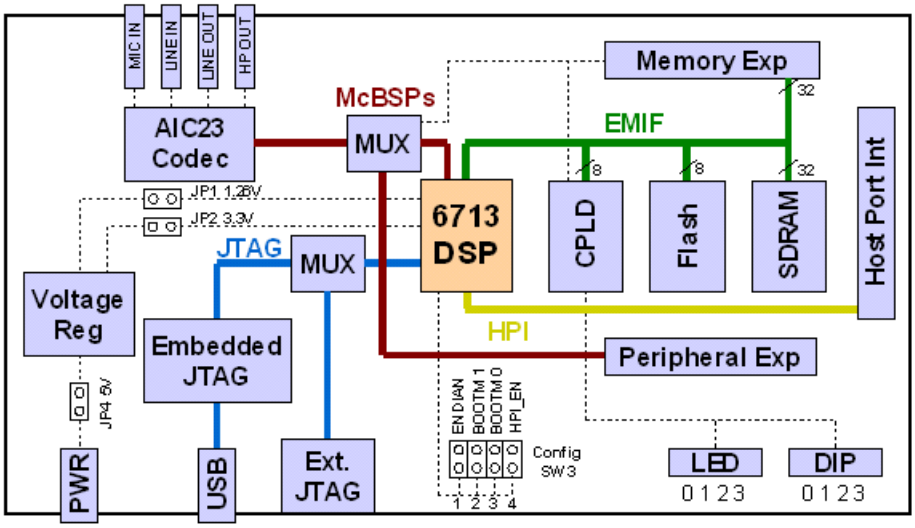

Figure 1. Key features of the C6713 DSK.

The Lab

Create a program that directs the AIC23 codec to generate a 1 KHz sine wave on the headphone output.

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.

3. Create a new source by using File → New → Source File and save it as a C file (i.e. YourProgram.c). This file will contain headers, define statements and the main()

program. To add source files, header, libraries and command files use Project → Add Files to Project… The header files that need to be included are:

- dsk6713.h which contains DSK6713 board specific I/O register definitions.
- dsk6713_aic23.h codec interface for AIC23 on the DSK6713 board.
- dsk6713config.h general initialization of the AIC23 (to be provided).
- math.h required when using math functions.

4. You need to set the compiler and linker options for the CS6713 DSK. You can access these setting by using Project → Build Options and then selecting Compiler and the Linker tabs to make the changes:

   - **Complier Options:** In the Basic category set Target Version to C671x. In the Advanced category set Memory Models to Far, 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).

   - **Linker Options:** In 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; csl6713.lib.

5. Build the project using Project → Build. This compiles and assembles all the C files, and the resulting object file(s) are then linked with the library files. In the building process all the dependent files are included too. Select File → Load Program to load YourProgram.out, and then Debug → Run to run the program. To stop execution of the program you need to use Debug → Halt.

6. Before you use the board, you need to initialize the board, start the codec, and set the AIC23 frequency to 8 KHz (the default sampling frequency is 48 KHz). Next, you need to discretize the continuous-time sine wave given by

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

   Assume that that $f_0 = 1$ KHz.

7. Use the template shown in Figure 2 to start your program.

```
//========= Lab1.c =========
//  This program generates a 1KHz sinusoidal signal
//  using the polling method
//


#include <math.h>
#include "dsk6713.h"
#include "dsk6713_aic23.h"                                  // codec support
#include "dsk6713config.h"

Uint32 fs = DSK6713_AIC23_FREQ_8KHZ;                        // set sampling rate
#define DSK6713_AIC23_INPUT_MIC 0x0015
#define DSK6713_AIC23_INPUT_LINE 0x0011
Uint16 inputsource=DSK6713_AIC23_INPUT_MIC;                 // select input

                                                            // sampling frequency

                                                            // freq. of signal
                                                            // amplitude
                                                            // theta

void main()
{

  DSK6713_init();                                           // call BSL to init DSK-EMIF,PLL)

  hAIC23_handle=DSK6713_AIC23_openCodec(0, &config);        // handle(pointer) to codec
  DSK6713_AIC23_setFreq(hAIC23_handle, fs);                 // set sample rate
  DSK6713_AIC23_rset(hAIC23_handle, 0x0004, inputsource);  // choose mic or line in
  MCBSP_config(DSK6713_AIC23_DATAHANDLE,&AIC23CfgData);     // interface 32 bits to AIC23

  MCBSP_start(DSK6713_AIC23_DATAHANDLE, MCBSP_XMIT_START | MCBSP_RCV_START |
  MCBSP_SRGR_START | MCBSP_SRGR_FRAMESYNC, 220);            // start data channel

  while(1)                                                  //infinite loop
  {

                                                            // compute theta increment
                                          . . .

       AIC_data.channel[LEFT]=(short)amp*sin(theta);        // data to Left channel
       while(!MCBSP_xrdy(DSK6713_AIC23_DATAHANDLE));        // wait for ready to transmit
       MCBSP_write(DSK6713_AIC23_DATAHANDLE,AIC_data.uint); // output left channel

  }
}
```

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. Use View → Quick Watch then type in the name of the frequency variable you used in the program. You need to declare this variable to be global.