**You can use the book “Digital Signal Processing and Applications with the OMAP-L138 eXperimenter” with either the eXperimenter or the LCDK!**

Virtually all of the program examples for the eXperimenter from Chapters 1 to 6 have been ported to the LCDK. They have been tested using a Spectrum Digital XDS100v2 USB JTAG emulator to connect between Code Composer Studio version 5 running on a PC and the LCDK.

**Differences between the LCDK and eXperimenter implementations**

The main differences between the two implementations, *that will be apparent to the user*, are that

1. DIP switches #5 thru #8 on the LCDK are used in place of user DIP switches #1 thru #4 on the eXperimenter.
2. The LCDK provides a 3.5mm jack socket connected to the MIC input on the AIC3106 codec and hence an inexpensive PC microphone (as opposed to a more expensive dynamic microphone) may be connected to the MIC input socket on the LCDK.

There are differences between the two boards regarding the type of external memory used and the connections between the processor and the AIC3106 codec, but in the context of the program examples these differences are hidden from the user.

**Installing the LCDK files and using the book**

The book may be used with either the LCDK or the eXperimenter. Clearly the photograph, and detailed description, of the eXperimenter in Chapter 1 are not applicable to the LCDK. However, the program example source files for the LCDK are only very slightly changed from those for the eXperimenter in a way (described below) that leaves the program listings and descriptions in the book suitable for use with the LCDK.

Instead of copying the files provided for the eXperimenter so as to create a folder c:\eXperimenter, simply **copy the files provided for the LCDK so as to create a folder c:\LCDK, and when launching Code Composer Studio choose workspaces from that folder, e.g. c:\LCDK\L138\_chapter2**. Installation of Code Composer Studio, the LogicPD board support library (BSL), and DSPLIB should be carried out exactly as described in the book. A small number of functions from the LogicPD BSL that are compatible with the LCDK are used by the program examples provided (although there is currently no BSL written specifically for the LCDK).

**Support files for the LCDK**

The differences between the files used for the LCDK and those supplied for the eXperimenter are contained primarily in the support files in folder c:\LCDK\L138\_support. The example program source files are very close to those provided for the eXperimenter and therefore the LCDK may quite readily be used with the book.

The files contained in folder c:\LCDK\L138\_support are

1. linker\_dsp.cmd This is identical to the similarly named file supplied for the eXperimenter.
2. vectors\_intr.asm This is identical to the similarly named file supplied for the eXperimenter.
3. vectors\_poll.asm This is identical to the similarly named file supplied for the eXperimenter.
4. OMAP-L138\_LCDK.GEL This was downloaded from the LCDK User Guide wiki page http://processors.wiki.ti.com/index.php/LCDK\_User\_Guide
5. L138\_eXperimenter\_v2.ccxml This target configuration file is suitable for use with the inexpensive Spectrum Digital XDS100v2 USB JTAG emulator. Other JTAG emulators could be used with the LCDK but would require their own target configuration files. L138\_eXperimenter\_v2.ccxml uses GEL file OMAP-L138\_LCDK.GEL that may be downloaded from the LCDK User Guide wiki page http://processors.wiki.ti.com/index.php/LCDK\_User\_Guide.
6. L138\_LCDK\_aic3106\_init.h This, and the following file have been modified to account for the slight hardware differences between the LCDK and eXperimenter.
7. L138\_LCDK\_aic3106\_init.c
8. L138\_LCDK\_aic3106\_init\_triple.c This support file is used only for the triple-buffered EDMA example in Chapter 2 that is described further on in this document.

**Using the MIC input on the LCDK**

Functions L138\_initialise\_poll(), L138\_initialise\_intr(), and L138\_initialise\_edma() have been defined differently in file L138\_LCDK\_aic3106\_init.c. (compared to their original definitions in file L138\_aic3106\_init.c). The modified functions give the user the choice of using either the MIC or the LINE input on the LCDK. This is achieved using a fourth parameter passed to the aforementioned functions. Constants LCDK\_LINE\_INPUT and LCDK\_MIC\_INPUT are defined in file L138\_LCDK\_aic3106\_init.h and may be used to select one or other of the inputs, for example, in the following way

L138\_initialise\_edma(FS\_48000\_HZ,ADC\_GAIN\_0DB,DAC\_GAIN\_0DB,LCDK\_MIC\_INPUT);

Programs L138\_delay\_intr.c and L138\_echo\_intr.c in Chapter 2 use the MIC input by default but other programs may be adapted to use MIC, as opposed to LINE input.

**Extra program examples for the LCDK**

Some extra (compared to those supplied for the eXperimenter) program examples have been added.

1. **L138\_loop\_poll\_PGA.c** is provided in folder c:\LCDK\L138\_chapter2\L138\_loop\_poll as an alternative to program L138\_loop\_poll.c, and may be used by excluding source file L138\_loop\_poll.c from the project build and including source file L138\_loop\_poll\_PGA.c.

This program demonstrates the connection of the PGA output to the AIC3106 output, bypassing the digital section of the codec. The bandwidth of this analog signal path is greater than 48 kHz and this is independent of the value of the parameter fs passed to function L138\_initialise\_poll(). The parameter adc\_gain still has an effect but dac\_atten does not (see the AIC3106 datasheet, and Figure 2.3 on page 40 of the book).

This program may be used with the eXperimenter if copied into folder eXperimenter\L138\_chapter2\L138\_loop\_poll and the line

#include “L138\_LCDK\_aic3106\_init.h”

is replaced by the line

#include “L138\_aic3106\_init.h”

and the line L138\_initialise\_edma(FS\_48000\_HZ,ADC\_GAIN\_0DB,DAC\_GAIN\_0DB,LCDK\_LINE\_INPUT);

is replaced by the line

L138\_initialise\_edma(FS\_48000\_HZ,ADC\_GAIN\_0DB,DAC\_GAIN\_0DB);

1. **L138\_loop\_edma\_triple.c** is provided in folder LCDK\L138\_chapter2\L138\_loop\_edma as an alternative to program L138\_loop\_edma.c. It is a straightforward talk-thru program using DMA-based i/o but EDMA3 has been configured to implement a triple buffering scheme instead of ping-pong buffering. In order to run the program it is necessary to use the support files L138\_LCDK\_aic3106\_init\_triple.c and L138\_LCDK\_aic3106\_init\_triple.h in place of files L138\_LCDK\_aic3106\_init.c and L138\_LCDK\_aic3106\_init.h since these provide the definition of a modified function EDMA3\_PaRAM\_setup().

Two more linked PaRAM sets are used to set up the triple-buffered, as opposed to ping-pong, i/o scheme, but one less buffer is required.

When program L138\_loop\_edma\_triple() is halted, the DMA process continues to pass data from input to output without CPU intervention.

The triple buffering scheme used here could be used with any of the program examples in the book that use DMA-based i/o, with only minor changes to those programs.

1. **L138\_firdelta\_intr.c** is provided in folder LCDK\L138\_chapter3\L138\_firprn\_intr as an alternative to program L138\_firprn\_intr.c. Instead of reading input samples from the function prbs(), it reads input samples from an array initialised to contain a Kronecker delta, or unit impulse, sequence. Once the contents of the array have been read, the sequence is repeated. Correspondingly, the output from this program represents the impulse response of the FIR filter implemented, repeated at intervals of 32 milliseconds. In order to see this clearly using an oscilloscope, it may be necessary to alter the value of the constant AMPLITUDE, depending on the FIR filter coefficients used.