BME-MIT

Control engineering and image processing laboratory – DSP development technologies

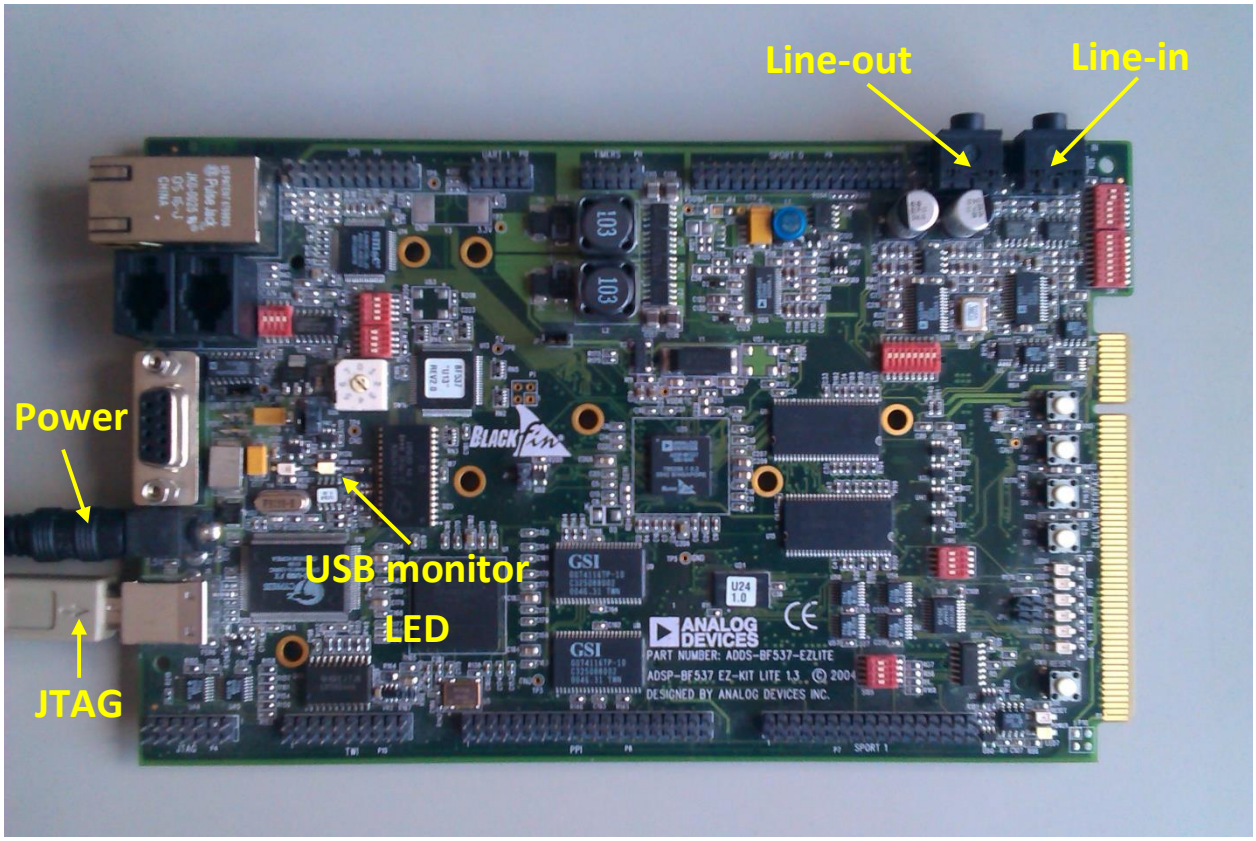
Introduction to DSP development systems

Orosz György

2015

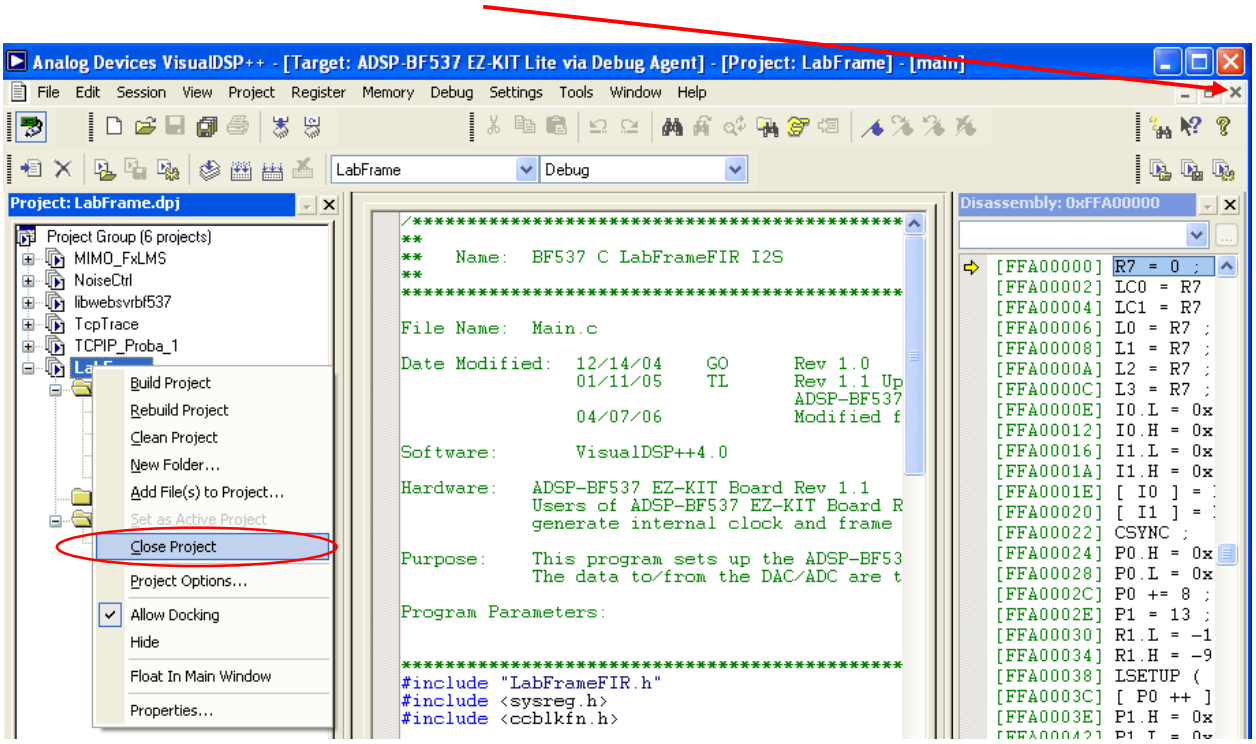
Preparation of the DSP board and the development environment

- Connect the DSP card to the PC. **Attention!** Plug the power cord first, then the USB JTAG cable. Stick to this order if you need to reset the DSP card.
- Wait until the „USB monitor" LED lights up! It means that the card has successfully connected to the computer.



- Start the development environment on the PC: Start menu → Programs → Analog Devices → VisualDSP++ **5.0** → VisualDSP++ Environment
  Important: please check twice that you are launching the 5.0 version of the program.

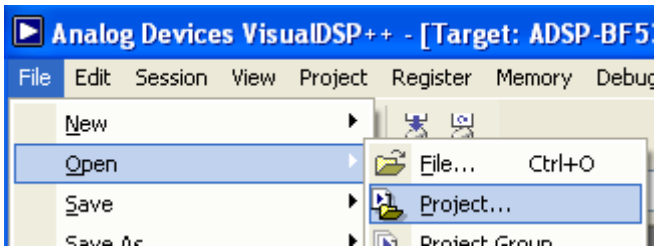
- Close all previously opened projects and windows!
    - Closing a project: right click on the project to be closed in the left window, then select „Close Project” (see the figure below).
    - Closing a window: close the opened windows by clicking on the X sign at the top right corner of the window.

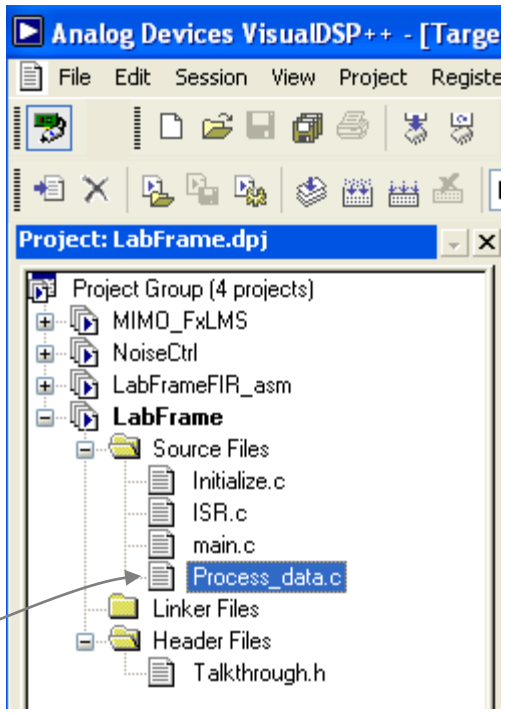


- Check if the ADSP-BF537 EZ-KIT Lite via Debug Agent text appears in the heading of the window. If not, go to the Session menu and click on Select session.
- Create your own working directory on the c: drive with a unique name which helps the identification of your student group.
- Download the sample project from the following address:
  https://www.mit.bme.hu/system/files/oktatas/targyak/7258/dsp1_0.zip
  Unzip the file.
- Connect the jack – BNC cables to the Line-in and Line-out connectors of the DSP card.
    - **Attention:**
        - Take care of the labels on the input and output connectors (Line-in/Line-out)! **Do not connect the Line out to the output of the signal generator!**
        - The -20 dB button must be pressed, otherwise the Line in could be overdriven! (signal generator: max. 20 V_{pp}, maximum input on the line in channel: 3 V_{pp})
    - Connect the „Line out” output to the oscilloscope (both channels).
    - Connect the „Line in” input to the signal generator (the channel does not matter). Set a sine wave output signal on the generator.

Opening a project

- Open the LabFrame.dpj project in the previously unzipped folder! To open a project: File menu → Open → Project…, and here the project file has to be selected!





- Now you can see the Process_data.c file in the project window under the LabFrame project among other source files. The Process_data.c file contains the following signal processing method:
  - Void Process_Data(void) is called every time a new sample is available from the ADC.
  - The iChannel0RightIn and iChannel0LeftIn variables contain the sample of the left and right channels.
  - The iChannel0RightOut and iChannel0LeftOut variables contain the data sent to the DAC.
- Compile the project! After the code is generated, it is automatically uploaded to the DSP card. Compiling options:
  - F7 key *or*
  - Project menu → Build (Rebuild all) *or*
  - clicking on the  button.
- Run the project! (After uploading the program, it will be automatically halted at the first instruction). To run the program:
  - F5 *or*
  - Debug menu → Run *or*
  - clicking on the  button.

  To stop the program:
  - Shift + F5 *or*

    - Debug menu → Halt *or*
    - clicking on the  button.
- The project simply forwards the signal of the right input channel to the left and right outputs. If no signal appears on the oscilloscope, switch to the other channel of the cable at the generator output.

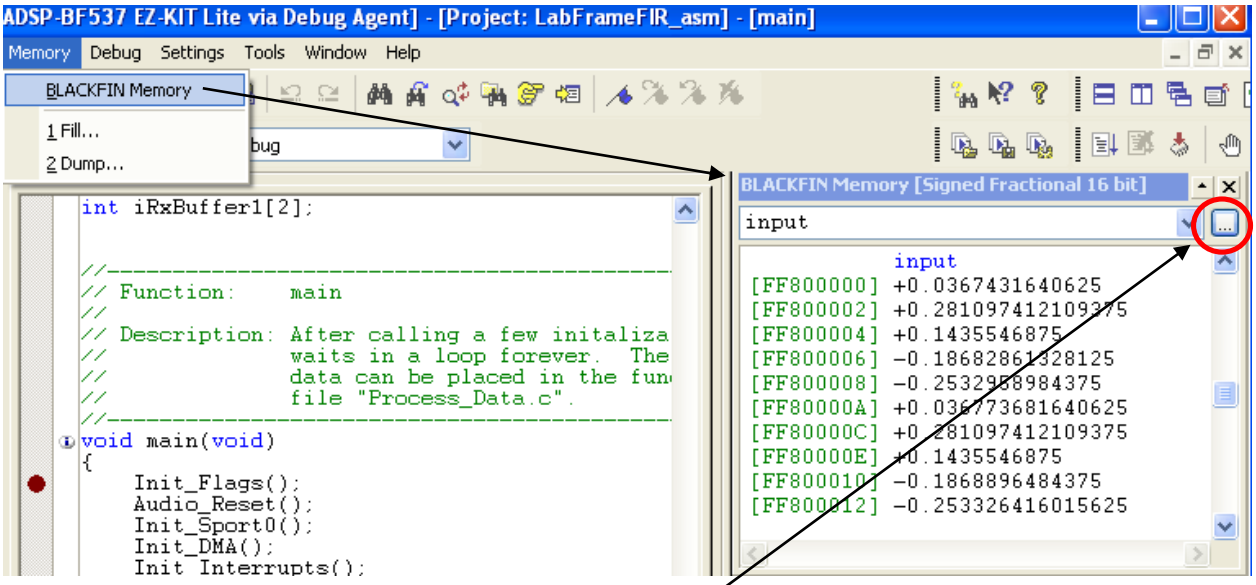
FIR filter with assembly routine

- Open the LabFrameFIR_3.dpj project! This is a FIR filter application: it filters the signal on the right channel of the input. The filtered signal is forwarded to the left channel of the output. On the right channel the original signal appears without being filtered.
- Study and understand the code. Notes:
    - The iChannel0RightIn >> 8 and out <<8 commands are shifting the data because the ADC and DAC are of 24 bits, but the DSP works with 16-bit data.
    - The filtering is done in the conv_asm function.
- Compile and run the project.
- Measure the cutoff frequencies of the filter using the signal generator and the oscilloscope. What type of filter is implemented? Compare the filter with the original one designed in Matlab: inside MATLAB open the filter design toolbox by the fdatool command. Open the bandpass.fda file inside the toolbox.

Debug functions of the IDE

Reading and writing variables in the memory.

- Inside the Memory menu select BLACKFIN Memory, and the „BLACKFIN Memory" window appears

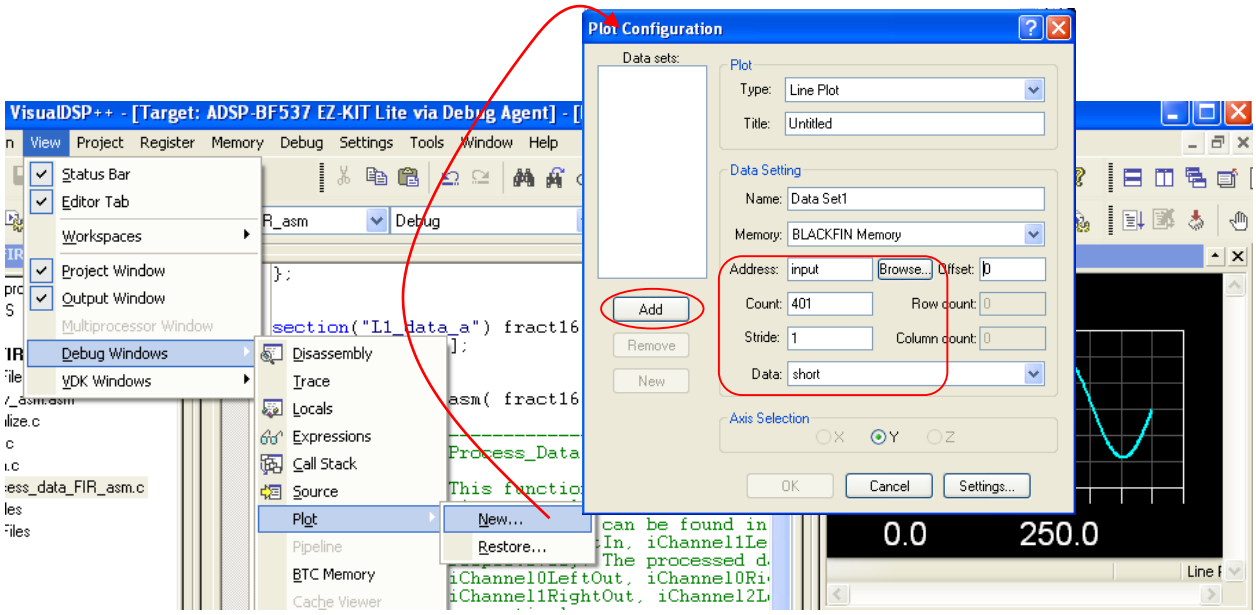


- Click on the button marked by the red circle. Now you can choose the variables you want to monitor. Select the array named input.
- By right-clicking on a variable and selecting „Select Format" you can choose the format of the data representation. For example, choose „Signed Fractional 16 bit" for fraction numbers, etc.

- By double clicking on a variable its value can be modified. After entering the new value, confirm it by hitting the Enter key. Then it will be updated in the memory of the DSP.
- These features can be used only if the DSP is in „Halt" state, so the running program must be halted first.
- To study some variables in the program, insert Breakpoints by double clicking next to the selected line of the code on the grey bar on the left. When the program arrives to the selected line, its state will change automatically from running to halted.

Plotting data arrays.

- Select View menu → Debug windows → Plot → New … Here you can specify the name, length and data type of the array you want to plot. Click add after specifying these parameters.



- The array variable can be selected with the Browse button. First select the input variable.
- Enter the length of the array to the Count field. For the input array this is 401.
- Stride defines the step size, its value should be 1.
- Select the number format in the Data list. Choose short.
- Add the array to the Data sets by clicking on the Add button. After clicking Ok, a figure plotting the array data appears.
- The DSP must be in Halt state to use these features. Use breakpoint(s).
- The sampling frequency of the ADC and DAC is 48 kHz. Adjust the frequency of the input signal until the sampling is coherent (an integer number of periods is measured of the input sine) and plot the result (e.g. measure 7 periods of the signal). Next measure a sine with noncoherent sampling by changing the signal frequency. Plot the result and explain it (circularity of the buffer).
- Modify some elements of the input array and plot it again.

- Plot and study the filter coefficients (coefs variable), explain the filter characteristics (bandpass filter = lowpass filter modulated by a sine wave).

Runtime-analysis

- Select Tools menu → Statistical Profiling → New Profile and open a new runtime statistics.
- Launch the program, and analyze the run-time of the different functions. Measure the run-time percentage of the conv_asm function.
- Modify the memory section of the input or the coefs variables (from `section("L1_data_a")` to `section("L1_data_b")`, or vice versa). The variables should be in the same memory section.
- Measure the run-time percentages again. What is the difference? (sometimes the previously measured statistics should be deleted manually: right click on the statistics and select „Clear profile").

Implementing FIR filter in C

- Open the LabFrameFIR_2.dpj project.
- Study the implementation of the circular buffer. Which functions are used for adding and multiplying data? Why?
- Measure the run-time percentage of the signal processing procedure.
- Select Project menu → „Project Options …". In the new window select Compile → General and „Additional options:" here delete the -force-circbuf option.
- Compile the project again and launch run-time statistics. Study the results.
- Add the -force-circbuf option again to the additional options.
- Perform some additions and multiplications with the * and + operators, and the fract16 library functions. Compare the result in multiple different cases.