Hello All!

Introduction

At first, let me introduce myself; my name is Dimitar and I am glad to be a member of Astfin / uCpbx
and now
Switchfin
teams.I am a DSP engineer and my major interest is to twist the FFT in a try to process given signal. For one involved in the field it will not be a surprise that Matlab is my favorite tool.

As you know the biggest challenge is actually to implement the tested and simulated idea in the fixed point DSP hardware. Why fixed point?

Well in the linear signal processing (the foundation of the DSP) it is all about repetitive multiplications and accumulations.

From hardware perspective however fixed point multiplier and adder need less area on the silicon die, crunch the numbers quicker and consumes less power than the corresponding floating point implementation.

So, why I am babble all this? Well I just wanted to express better how exited I was, at the moment I started with the Line Echo Canceller (LEC) project. It is about adaptive filtering one of the most interesting field in DSP, it is about fixed point implementation as well.

Overview

Now it is the right moment to say that actually my task was not to start from scratch, no no! Unfortunately I had only to test one of the most promising open echo cancellers software available. Thank you Mark about pointing me to the MIKET DSP Solutions web site! The author of this code apparently is a great scientist and at the same time they were generous enough to open the code. One huge thank you goes to MIKET DSP Solutions!!! I encourage everyone interested in this field to go and browse his site.

So at the beginning I had to select the test excitations. I was already told that the test signals defined in G.168 can be used for the convergence analysis but are far from being good for the perceptual assessment of the different echo cancellers. I decided to take the simplest approach and to record several speech samples - myself and Kalina

The next step was to build the model for the echo path. I tried to find good samples of the

typical recorded impulse response somewhere in the internet but unfortunately without success. So I just made several more or less artificial impulse responses. I have used a curve family with exponential positive and negative fronts and I was able to vary the time constants of this exponents and the delay of the impulse response. This way I was able to test with different delays and echo impulse response tails.It is not necessary to say that all this I did in Matlab environment, isn't it?

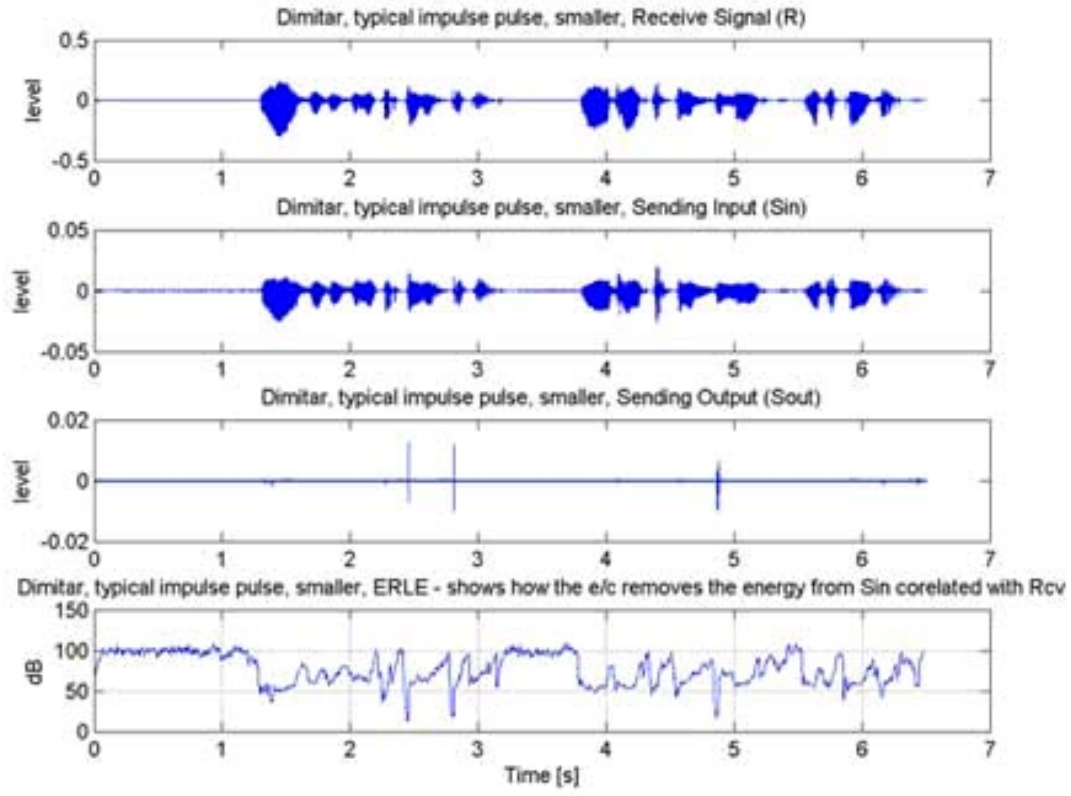
Oh I actually forgot to mention the biggest benefit of the MIKET DSP Solutions. He provided not only the general C code but also an optimized version for the C55x digital signal processors from Texas Instruments. So the most critical part are available in assembly.
Fortunately, I had an evolution version of the Code Composer Studio (CCS) from Texas Instruments
and it was my ideal tool to do the cycle true simulation. For the interface with Matlab I have used data files generated by Matlab and supported by CCS and data transfer was triggered by probe points in CCS.

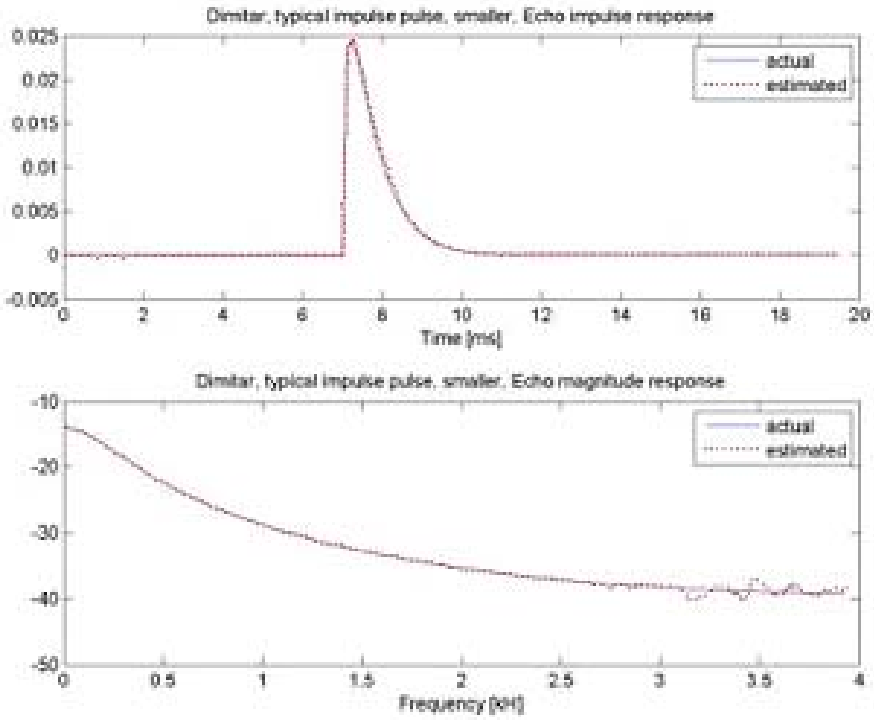
Now is the time to say something about the echo canceller algorithm I was about to test. As most modern one it uses two adaptive filters and update of the main filter is done only if the secondary filter reaches certain degree of convergence. The adaptation uses Reduced Complexity Recursive Least Square(RC-RLS) algorithm which is good compromise between performance and computational complexity. Under noise conditions as you probably know even the best adaptive filters will miss some echo component. That's why a Non Linear Processing (NLP) is used after the adaptive filtering to clip the remaining echo under given conditions. The beauty of the solution provided by MIKET DSP solutions is in the control logic they use. It is very complex and from the very beginning I had a feeling that I am dealing with the professional algorithm.

Experiments
So it was time for the first experiments. I applied my test data and recorded the adaptation of the main adaptive filter. Strange the filter didn't converge. The first think I thought that I am not applying the excitation signals properly. I read and read the companion manual but everything seemed to be OK with my setup. Then I decided to play with the parameters controlling the LEC but whatever I did the result remained the same. No convergence.
Hmm what should I do?
I noticed that the recorded samples contained some hum. You see a harmonic noise is one of

the biggest challenge for the adaptation process as it can result in detergency. So I pre filtered my samples (despite the band pass filtering which is included in the original LEC) but still the LEC behaved as it wasn't even there. It was necessary to go deeper in the code and to try to understand how it is working in more details. I have to admit that it was a week of investigation. The LEC is complex, believe me. My logic pointed me to the assembler function lec_gs_filter(). Something was not OK. Fortunately, MIKET DSP solutions left the corresponding C function available. It was an evening when I decided to experiment with the C counterpart of lec_gs_filter() and I got the convergence for the first time :) I was displaying the result in Matlab and I saw something like on the following figures. As you can see on the 3-rd subplot the residual echo is very small. On the second figure the actual and the estimated by the LEC impulse response are superimposed, in time and frequency domain.
Exciting :)

Some Results





Let me upload two speech samples without-lec.wav and with-lec.wav . It makes a difference, doesn't it?

I played with different echo impulse responses everything showed that the LEC is very good.

Then double talk. You know sometimes happens that both persons on the phone can talk at the same time.(especially when one of them is impatient) .This condition is a hard test for echo cancellers as some of them tend to diverge on double talk. I decided to simulate this as well and no divergence was detected.

Performance

So far so good. But how many independent LEC channels we can fit in the DSP we have in mind (200MHz C55x from Texas Instrument). The profiling capability of the CCS appeared to be perfect to answer. I measured that about 30000 CPU clocks are need in average per 5ms frame. So we can fit 5ms*200MHZ/30000 or about 30 channels in our DSP. With other words the LEC takes 30000/5ms or 6 MIPS/channel. I have discussed with David Rowe from www.rowetel.com
the author of
OSLEC
, another very good open echo canceller and he told me that 6MIPS is actually a very good achievement. Thank you David for the help!

What is coming next?

The simulation is nothing unless you confirmed in a real hardware.

We are already prototyping our DSP LEC module based on the described LEC. It will be used in our PR1 and BR4 appliances. For carrier grade echo cancellation we plan to have another module as well. It will be based on a chip from Zarlink Semiconductor

Stay tuned,

Dimitar