PARALLEL DIGITAL SIGNAL PROCESSING FOR EFFICIENT PIANO SYNTHESIS

Leonardo Gabrielli

Dipartimento di Ingegneria dell'Informazione
Università Politecnica delle Marche
Ancona, Italy

Stefano Zambon, Federico Fontana

Dipartimento di Matematica e Informatica
University of Udine
Udine, Italy

ABSTRACT

While computational acoustics techniques for musical instruments emulation reached a remarkable maturity due to continuous development in the last three decades, implementation into embedded digital instruments lags behind, with only a few notable commercial products to solely employ physics-based algorithms for acoustic instruments tone synthesis. In this paper a parallel DSP architecture for the efficient implementation of the acoustic piano on embedded processors is reported. The resulting model is able to provide faithful reproduction of the acoustic piano physical behaviour and can also be used as an engine for novel instruments that need to provide advanced multimodal output (haptics, spatial audio) with a low-cost embedded platform.

Index Terms— Sound synthesis, physical modeling, parallel computing, digital signal processing

1. INTRODUCTION

In contrast with traditional *signal-based* synthesis techniques, e.g. sampling or wavetable synthesis, *physics-based* algorithms try to imitate the sound of acoustic instruments at the *source* level, modeling the mechanical components and deriving efficient numerical schemes. The advantages are a better response to key velocity and higher-fidelity imitation of restrikes or other interaction phenomena such as sympathetic string resonance. Moreover, they are very flexible in terms of sound editing, offering the user a meaningful set of parameters with physical meaning (e.g. mass, hardness), and they can also easily provide additional physical signals that can be used e.g. for vibrotactile feedback that has been shown to enhance the realism of virtual piano instruments [1].

On the other hand, the computational requirements needed to achieve a realistic results are far more demanding than for traditional synthesis techniques. Especially on embedded processors it is, thus, desirable to exploit all the available parallel computing capabilities in order to achieve the desired throughput. This is particularly necessary in the context of emerging technologies for portable musical interfaces that need to be small and very power efficient, such as devices that can offer a keyboard-like experience on a generic surface like a table [2].

In this paper we review the technology behind a physics-based piano synthesis engine that has been used in digital pianos [3] and has partially been ported to a portable device that enables piano playing without any keyboard on a generic surface. After describing the software architecture and general algorithm used for the synthesis, we present some highlights on the parallel hardware architecture able to sustain the acoustic piano model computational requirements.

2. MODEL ARCHITECTURE

Following a *block-based* approach, we defined a set of signal-processing blocks corresponding to the main parts of the acoustic instrument, e.g. the hammer (*exciter*), the string (*resonator*) and the instrument body (*radiator*). *Physics-based* piano models often differ in the choice of the discretization technique used for the numerical solution of the string partial differential equation (PDE). Popular choices are finite differences or digital waveguides [5, 6]. During our preliminary investigation phase, we opted instead for a variation of *Modal Synthesis* [7], which we found sensibly better in terms of calibration flexibility and modelling accuracy.

In brief, the algorithm is based on the decomposition of the string displacement $y(x,t)$ into its orthogonal normal modes

$$y(x,t) = \sum_{n=1}^{N} y_n(t) \sin\left(\frac{n\pi x}{L}\right), \qquad (1)$$

where $y_n(t)$ are the instantaneous amplitudes of the modes, or *partials*. Substituting Eq. (1) into the PDE of the string motion results in an ordinary second-order differential equation for each partial, and thus the impulse response of the string becomes a sum of exponentially decaying sinusoidal functions.

After discretization, the input-output relation of the string block is realized as a parallel connection of N second-order all-pole resonators:

978-0-9928626-3-3/15/$31.00 ©2015 IEEE

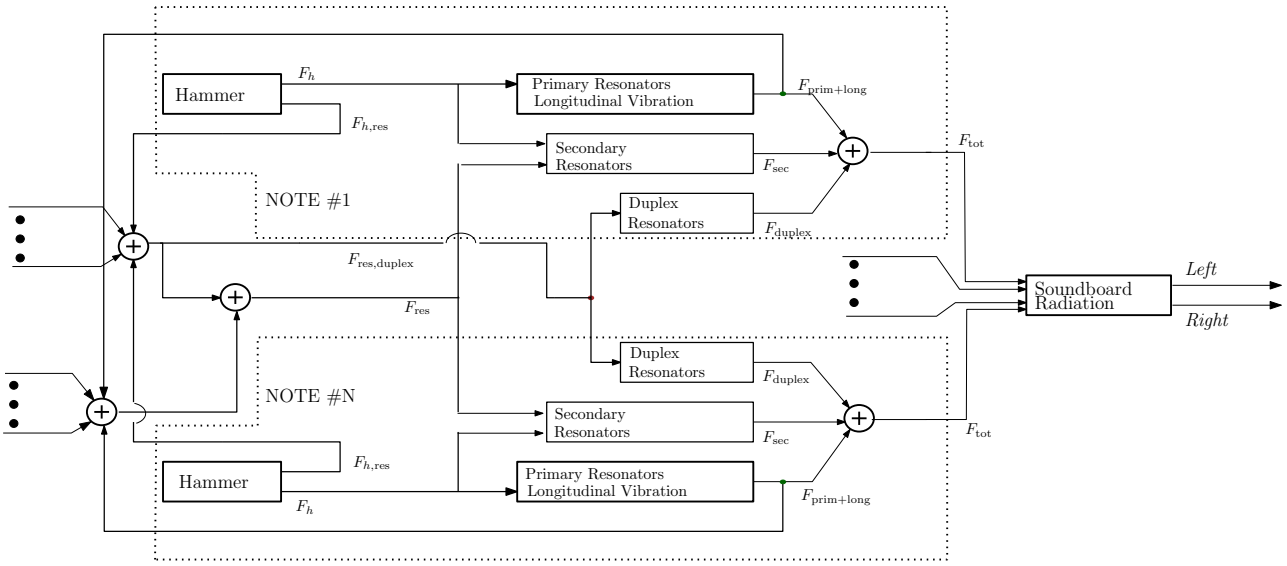

Fig. 1. Synthesis architecture of the acoustic piano model, showing the interactions between the notes and the soundboard model. Adapted from [4].

$$
\begin{aligned}
F_{\mathrm{string}}(z) &= H_{\mathrm{string}}(z)\, F_{\mathrm{h}}(z) \\
H_{\mathrm{string}}(z) &= \sum_{k=1}^{N} W_{\mathrm{in},k}\, H_{\mathrm{mode},k}(z)\, W_{\mathrm{out},k} \qquad (2) \\
H_{\mathrm{mode},k} &= \frac{b_{1,k} z^{-1}}{1 + a_{1,k} z^{-1} + a_{2,k} z^{-2}},
\end{aligned}
$$

where $F_{\mathrm{string}}(z)$ is the transversal force at the bridge, $F_{\mathrm{h}}(z)$ is the force coming from the hammer and $H_{\mathrm{mode},k}(z)$ are the transfer functions of the normal modes. The conversion between the physical variables (i.e. force, displacement) and the modal variables is regulated by a set of input and output weights $W_{\mathrm{in},k}$, $W_{\mathrm{out},k}$.

Even if the algorithm at its core is intrinsically parallelizable, we had to simplify the hammer-string interaction from the algorithm previously proposed in [7], partly for overcoming some calibration issues and also for better exploiting the capabilities of modern hardware. As it can be seen from the diagram in Fig. 1, the hammer model has a *feedforward* connection to the string model, allowing efficient block-based computation. We designed a parametric synthesis algorithm which is able to qualitatively reproduce the signals of the former feedback version employed in [7], but whose parameters are decoupled from the string parameters.

The other blocks showed in the diagram can be also described as sets of parallel second-order resonators. They are used to model secondary features of piano strings which are not captured by the one-dimensional string PDE. *Longitudinal* string vibration is approximated with a feed-forward, parallel set of resonators whose parameters are set in order to match as closely as possible the output of the modulation-based algorithm proposed in [7]. *Secondary resonators* are slightly detuned groups of resonators which are needed for simulating the complex beating envelopes found in piano partials. In addition, they are employed for the simulation of the *sympathetic resonance effect*, since each note filters the signal obtained as a sum of all the primary resonators. Finally, the *duplex resonators* model the so called *duplex scale*, a portion of the strings above their speaking length giving a particular brilliance to piano sound.

Speaking in terms of signal processing, there is no actual feedback in the resulting diagram. However, the introduction of the aforementioned connections between generated tones is critical in parallel computation contexts, as it requires synchronization mechanisms when the workload is divided among several processors.

2.1. Soundboard Filtering

The output of the computation for each note is the total force generated at the bridge, which is then converted into the desired acoustic pressure signal using the soundboard radiation module. In the first versions of our piano synthesizers, this was implemented simply as a block-based convolution with an impulse response (IR) measured on a real piano using an instrumented hammer.

However, the behaviour of the soundboard vary significantly across different regions, so we developed an algorithm which is capable of computing 4 responses at the same time with a reduced computational cost [8]. At its core, it is based on the parallel second-order filters approximation of a given IR proposed by Bank [9, 10]. The actual filtering

	x86-i5	Cortex-A7	Cortex-A9	C674x
Reson/core	17300	3920	3800	5200
TDP/core	8.7 W	1.4 W	0.6 W	0.8 W
Core Freq	2.5GHz	1GHz	1GHz	456MHz

Table 1. Maximum number of resonators computed in a 1.4 ms time by general purpose and dedicated processors.

uses a common-pole description between the 4 regions, therefore parallelization is possible only for the zeros of the filters and, clearly, for the simultaneous processing of the two audio channels. If, as in our case, the buffer size is kept low (64 samples) to minimize latency, the speed-up of the algorithm is in the order of 50x compared to straightforward FFT-based convolution. Further details are given in [8].

3. PARALLEL IMPLEMENTATION

The model described in Section 2 employs around 15,000 resonators, of which 10,000 are running most of the time in free evolution, plus the soundboard modelling, requiring approximately 3 GFLOPS.

By expressing the computational cost in terms of resonators, a more implementation-oriented data is obtained, as floating point operations do not take into account memory transfers, instruction and data pipelining, etc. Benchmarks on a few general purpose architectures are shown in Table 1. The Texas Instruments C674x DSP is compared to ARM and x86 general purpose processors in terms of parallel computing of resonators. Resonator grouping, i.e. parallel computation of N oscillators must be hard-coded for the DSP to exploit its specific data path, while in the general purpose processors it is obtained by resorting to explicit SIMD instructions, SSE2 or NEON for x86 and ARM processors respectively. While in absolute terms the x86-i5 (Intel i5 3210M) reports the highest number of resonators per core, in terms of energy efficiency (resonator per Watt), the Cortex-A9 (Freescale iMX6Q) and the C674x provide the best results[1]. The C674x also provides the best performance in terms of resonators per MHz. The lower operating frequency and the presence of peripherals specifically targeted for embedded DSP uses make it a good candidate for embedded designs.

The computing hardware chosen for the digital piano features a mesh of 6 interconnected DSPs of the TI OMAPL137 family, featuring a C674x DSP and a ARM9 microcontroller unit.

The C674x DSP CPU is shown in Figure 2. It features a level 1 program and data cache memories of 32KB each. Two data paths are available for parallel processing, each comprising four different functional units, able to operate on different data types. To give an overview on floating point operations available on these units, the *.L* units allow several arithmetic

[1]The power estimates for the Cortex-A7 is relative to a Alwinner A20

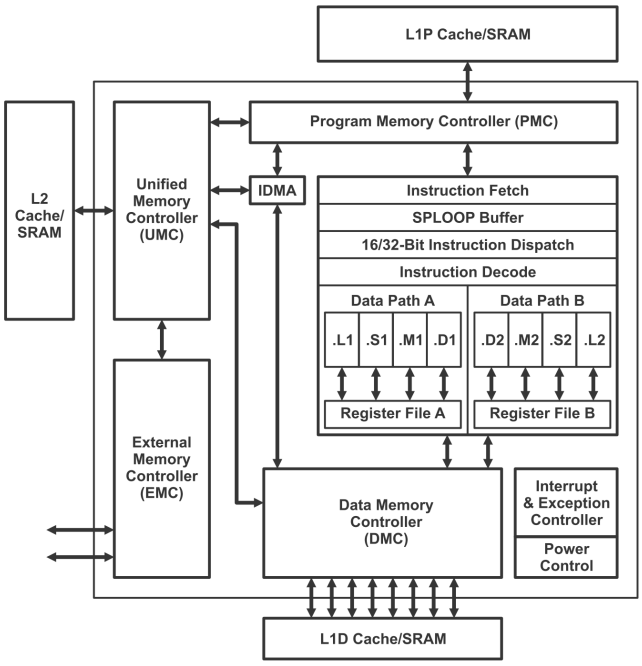

Fig. 2. Block diagram of the C674x CPU.

operations, and data precision conversion, or conversion to fixed or floating point. Unit *.S* allows floating point comparison, absolute value operations, reciprocal and reciprocal square-root, as well as operations between single and double precision values. Unit *.M* is devoted to any multiplication operation, including mixed-precision ones. Finally, unit *.D* includes double word (i.e. 64-bit) loading, address calculation, 32-bit logical operations. The CPU operates with data in register files A and B, which comprise 32 32-bit registers each, able to store data, pointers or conditions.

The DSPs interconnection of the digital piano implementation is depicted in Figure 3. The DSPs are assigned different functionalities. Four DSPs share the acoustic piano strings workload, including string resonances. A fifth one is in charge of other synthesis techniques, and the sixth one collects the outputs of the previous ones. The latter is also responsible for the soundboard algorithm and effects processing and provides the output signal to the DAC. The communication from the string DSPs to the soundboard DSP is carried over a slotted TDMA serial communication. Each slot carries samples to be conveyed to each spatial input of the soundboard. The string DSPs are also mesh-interconnected on a time-slotted bus in order to feed the others with the input signals for secondary and duplex resonators. Feedback signals are buffered. Each string DSP is in charge of part of the current voices (i.e. strings), secondary and duplex resonators. The voices are allocated on all four DSPs to share the workload. Since lower tones have a larger computational cost, contiguous tones are assigned a different DSP each to fairly balance the workload

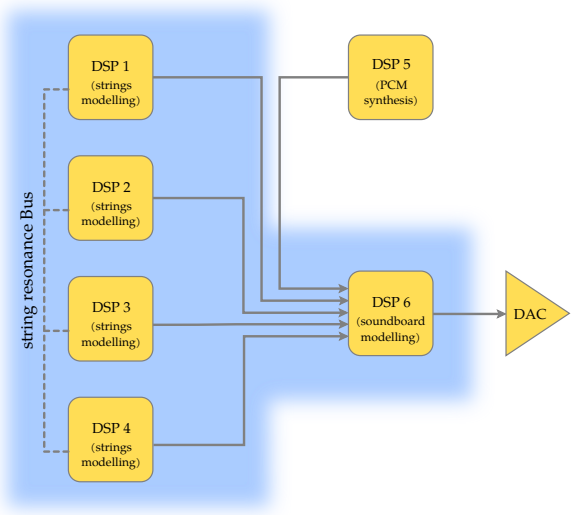

Fig. 3. Block diagram of parallel computing system. DSP 1 to 4 are devoted to string modelling; DSP 5 is responsible for PCM synthesis; DSP 6 collects all the signals and feeds the DAC. These connection take place on slotted TDMA serial lines.

among the DSPs. Moreover, partials which fall behind the hearing threshold are dynamically killed at runtime by analyzing their energy, thus reducing significantly the effective number of resonators computed in real-time.

4. CONCLUSION

This paper presents technical aspects regarding the implementation of modern computational acoustics research outcomes into an industrial products. These include both digital pianos with a traditional keyboard interface and lightweight, portable devices that can augment any surface into a virtual musical instrument. Drawing from previous research effort, the current model allows for large computational savings with respect to other modal synthesis formulations, e.g. the logarithmic spaced filtering for the soundboard or the use of secondary resonators to emulate both sympathetic resonance and envelope effects (two-stage decay, beating). Notwithstanding this, optimization has been conducted to increase efficiency and allow headroom for further algorithms. The current oscillators implementation followed an iterative process that enabled to establish a trade-off between constraints (the time frame and available cache memory) and computational resources.

5. ACKNOWLEDGEMENT

The authors acknowledge support from the Proof of Concept Network national project entitled ”Virtual piano system on a tablet pc” (”Sistema di pianoforte virtuale su tablet pc”) called by the AREA Science Park in Trieste, participated by the University of Udine and Julia SRL, and co-financed by Viscount SpA through the University of Verona, all from Italy. The authors would also like to thank Marco del Fiasco and Balazs Bank for their valuable contribution.

REFERENCES

[1] Federico Fontana, Hanna Järveläinen, Stefano Papetti, Federico Avanzini, Francesco Zanini, and Valerio Zanini, “Perception of interactive vibrotactile cues on the acoustic grand and upright piano,” in *Proc. of Joint SMC and ICMC 2014 Conference*. National and Kapodistrian University of Athens, Greece, 2014.

[2] Yuri De Pra, Fausto Spoto, Federico Fontana, and Linmi Tao, “Infrared vs. ultrasonic finger detection on a virtual piano keyboard,” in *Proc. of Joint SMC and ICMC 2014 Conference*. National and Kapodistrian University of Athens, Greece, 2014.

[3] Stefano Zambon, Leonardo Gabrielli, and Balazs Bank, “Expressive physical modeling of keyboard instruments: From theory to implementation,” in *Audio Engineering Society Convention 134*. Audio Engineering Society, 2013.

[4] S. Zambon, E. Giordani, B. Bank, and F. Fontana, “A system to reproduce the sound of a stringed instrument,” Deposited PCT international patent, March 2013.

[5] J.O. Smith, “Physical Audio Signal Processing,” Available at `http://ccrma.stanford.edu/~ jos/pasp.html`, 2012.

[6] J.O. Smith and S.A. Van Duyne, “Commuted piano synthesis,” in *Proc. Int. Computer Music Conf.*, Banff, Canada, 1995, pp. 319–326.

[7] Balázs Bank, Stefano Zambon, and Federico Fontana, “A modal-based real-time piano synthesizer,” *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 18, no. 4, pp. 809–821, 2010.

[8] Stefano Zambon, “Distributed piano soundboard modeling with common-pole parallel filters,” in *Proceedings of the Stockholm Music Acoustics Conference (SMAC) 2013*. KTH Royal Institute of Technology, Stockholm, 2013.

[9] Balázs Bank, “Direct design of parallel second-order filters for instrument body modeling,” in *Proc. International Computer Music Conference (ICMC 2007), Copenhagen, Denmark*, 2007, pp. 458–465.

[10] Balázs Bank, “Magnitude-priority filter design,” *Journal of the Audio Engineering Society*, vol. 62, no. 7/8, pp. 485–492, 2014.