Using the beagleboard as hardware to process sound

Rafael Vega & Daniel Gómez

Context and Motivation.

- Many long standing problems in the Latin American and Colombian context are rooted in a lack of information and unempowerment of the general population.
- Open-source and community based models can empower people in our local context.
- Rich and diverse musical traditions with little access to modern musical instruments.
- Time, talent, willingness and economic reasons to build your own tools instead of buying.

Main goal

The first goal was to approach free hardware and software and try to build the basis of a system that could be reconfigured for diverse uses such as a stompbox or as a synthesizer that enthusiasts and musicians could use as a first approach to programming audio and also as a universal tool for experimentation, performance and a DIY instrument platform.

First iterations of the prototype.

- C++, Chuck, CSound or PD?

First iterations of the prototype.

- C++, Chuck, CSound or PD?
- An open hardware platform with enough power and features for real-time audio and support for freely available software audio tools. The BeagleBoard and Agstrom Linux.

First iterations of the prototype.

- C++, Chuck, CSound or PD?
- An open hardware platform with enough power and features for real-time audio and support for freely available software audio tools. The BeagleBoard and Agstrom Linux.
- We decided to build a bridge between a PD patch and the Linux sound APIs. XookyNabox (wordplay for Chuck-in-a-box).

First iterations of the prototype.

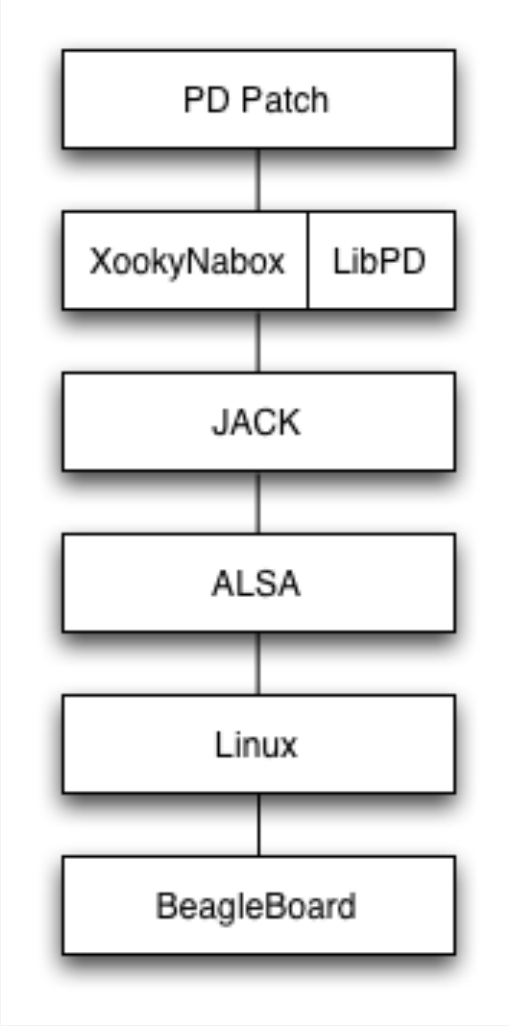
- C++, Chuck, CSound or PD?
- An open hardware platform with enough power and features for real-time audio and support for freely available software audio tools. The BeagleBoard and Agstrom Linux.
- We decided to build a bridge between a PD patch and the Linux sound APIs. XookyNabox (wordplay for Chuck-in-a-box).
- How to parse and execute a PD file? PDAnywhere, ZenGarden or libpd?

First iterations of the prototype.

- C++, Chuck, CSound or PD?
- An open hardware platform with enough power and features for real-time audio and support for freely available software audio tools. The BeagleBoard and Agstrom Linux.
- We decided to build a bridge between a PD patch and the Linux sound APIs. XookyNabox (wordplay for Chuck-in-a-box).
- How to parse and execute a PD file? PDAnywhere, ZenGarden or libpd?
- How do you talk to the systems' DAC and ADC? The ALSA API, RtAudio, JACK.

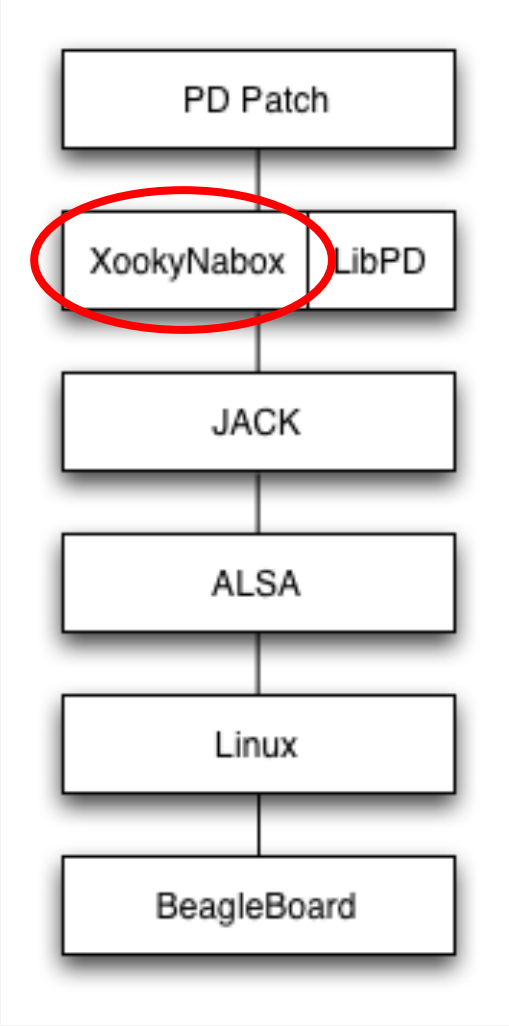
The approach that worked

General Architecture



The approach that worked

General Architecture



The approach that worked

Interleaved vs. non-interleaved audio.

```
int process(jack_nframes_t nframes, void *arg){
   // in1, in2, out1, out2 are pointers to the input and output signals

   // Jack uses mono ports and pd expects interleaved stereo buffers.
   for(unsigned int i=0; i<nframes; i++){
      input[i*2] = *in1;
      input[(i*2)+1] = *in2;
      in1++; in2++;
   }
   // PD Magic!
   libpd_process_float(input, output);

   for(unsigned int i=0; i<nframes; i++){
      *out1 = output[i*2];
      *out2 = output[(i*2)+1];
      out1++; out2++;
   };
   return 0;
}
```

The patches

Pd vanilla patches.

Basic processing: ringmod, filtering and clipping.

Basic tone generation: oscillators, lfo, basic filtering

Crashes with [line~] and with setting the phase to an [osc~] and [phasor~]

No envelope testing.

Conclusions and Future Work.

- The use of pd patches in a portable, light and relatively low cost computer makes a new generation of easily programmable customized audio hardware foreseeable.

Conclusions and Future Work.

- The use of pd patches in a portable, light and relatively low cost computer makes a new generation of easily programmable customized audio hardware foreseeable.
- The development of an interactive electronic bridge to allow communication of human-interface devices (accelerometers, potentiometers, sliders, buttons, etc) to control the patch in real time is a next step in the project.

Conclusions and Future Work.

- The use of pd patches in a portable, light and relatively low cost computer makes a new generation of easily programmable customized audio hardware foreseeable.
- The development of an interactive electronic bridge to allow communication of human-interface devices (accelerometers, potentiometers, sliders, buttons, etc) to control the patch in real time is a next step in the project.
- There are still complications in the loading of some pd patches related with specific connections and objects and more thorough debugging has to be made.

Conclusions and Future Work.

- The use of pd patches in a portable, light and relatively low cost computer makes a new generation of easily programmable customized audio hardware foreseeable.
- We need to explore ways to leverage the DSP hardware present in the BeagleBoard.
- The development of an interactive electronic bridge to allow communication of human-interface devices (accelerometers, potentiometers, sliders, buttons, etc) to control the patch in real time is a next step in the project.
- There are still complications in the loading of some pd patches related with specific connections and objects and more thorough debugging has to be made.
- More work is required in the divulgation of our work to potential users.

Links and contact.

- http://elsoftwarehamuerto.org/articulos/691/puredata-beagleboard/
- rvega@elsoftwarehamuerto.org
- dgomez@icesi.edu.co

Acknowledgements.

- Activata.
- Brolin and the Unloquer hacker space.
- ICESI University.
- Juan Reyes.
- PD, Linux and BeagleBoard communities.
- LAC 2012 organizers.