

Bela - Bug #1333

rt_printf and audio glitches

2015-07-15 09:37 PM - Giulio Moro

Status:	Closed	Start date:	2015-07-15
Priority:	Low	Due date:	
Assignee:		% Done:	0%
Category:		Estimated time:	0.00 hour
Target version:			

Description

rt_printf() and audio glitches

When printing to the console in BeagleRT you should always use the Xenomai functions `rt_printf()` and `rt_sprintf()` instead of their corresponding `printf()/sprintf()`. This improves performances quite a lot as this way they are handled directly by Xenomai and there is no need to switch back to the Linux kernel and then back again to Xenomai.

Despite this, sometimes glitches in the audio can still be heard when using `rt_printf()` in the audio thread. This is especially true when printing long strings and/or the number of frames per block is very small.

For instance, the code below will cause a click in the audio for each `rt_printf` when run with `./BeagleRT -p 1` (2 sample long audio blocks)

The code is printing a long string on purpose. Shorter strings will cause less issues, but will still occasionally cause glitches. I use a trick to force the compiler not to optimize out the CPU cycles I am wasting on purpose!

If you take out the for loop with the 24 sinewaves, performances will improve, but you will still have some clicks from time to time: this means that glitches do NOT occur ONLY when the system is overloaded to full CPU usage.

A solution to this is not to put `rt_printf()` statements in the audio thread if you really need to run your code with a frame size of two samples (`-p 1`). If you run the program with larger block sizes you will probably not encounter any issues. In case you need your code to run with a small block size and print to the console, the best option is to have the `rt_printf()` statements executed in a separate thread with low priority. You can use BeagleRT's AuxiliaryTask for this (which builds on Xenomai's threading infrastructure) and schedule an auxiliary task with low priority whenever you need to print.

```
int gCount=0;
void render(int numMatrixFrames, int numAudioFrames, float *audioIn, float *audioOut,
            uint16_t *matrixIn, uint16_t *matrixOut)
{
    for(int n = 0; n < numAudioFrames; n++) {
        float out = 0.8f * sinf(gPhase);
        float out2=0.1;
        for(int i=0; i<24; i++){
            out2=sinf(out2+gPhase); //Waste some CPU cycles. We need to prevent the compiler from optimizing this, so
            we always reuse the previous output as an input ...
        }
        out = 0.8f * sinf(gPhase);
        gPhase += 2.0 * M_PI * gFrequency * gInverseSampleRate;
        if(gPhase > 2.0 * M_PI)
            gPhase -= 2.0 * M_PI;

        audioOut[n * gNumAudioChannels] = out;
        audioOut[n * gNumAudioChannels + 1] = out2/1000000000 + out; //and here we "mix in" some of out2 so the
        compiler does not optimize it out
        gCount++;
        if(0==(gCount&32767))
            rt_printf("%d abcdefghijklmnopqrstuvwxyz abcdefghijklmnopqrstuvwxyz\n",gCount);
        abcdefghijklmnopqrstuvwxyz\n",gCount);
    }
}
```

History

- #1 - 2016-06-25 08:38 PM - Giulio Moro
- Status changed from New to Closed

Just keep in mind that CPU time is finite and printing takes time, regardless of the fact that it is handled by xenomai or kernel. So just don't overdo it.