

Re: randomized white noise = white noise?

Source: <http://sci.tech-archive.net/Archive/sci.electronics.design/2006-05/msg04392.html>

- *From:* James Waldby <j-waldby@xxxxxxxx>
 - *Date:* Thu, 25 May 2006 00:44:01 -0600
-

acannell@xxxxxxxx wrote:

Thanks for your very interesting posts Tom. I think I have found a "target" so to speak, to focus my efforts on. This algorithm:

```
b0 = 0.99765 * b0 + white * 0.0990460;
b1 = 0.96300 * b1 + white * 0.2965164;
b2 = 0.57000 * b2 + white * 1.0526913;
tmp = b0 + b1 + b2 + white * 0.1848;
```

Takes white noise samples and outputs pink noise samples. If I can find a way to perform it using the 400 or so instructions I have to work with on the ATTINY13, I have a way to generate pink noise real time using it! Maybe I can use a look up table or somehow do it so I dont need floating point....hmm

....

You might try a crude approximation like the following, which uses 16-bit fixed point arithmetic, with binary point at middle.

```
unsigned int b0, b1, b2, tmp, white, tw;
```

```
do {
```

```
<generate white>
```

```
tw = white/16;
```

```
tmp = tw*3;
```

```
b0 = b0 - b0/256 + tmp/2;
```

```
b1 = b1 - b1/16 + tw/4;
```

```
b2 = b2/2 + white + tw;
```

```
tmp += b0 + b1 + b2;
```

```
outp (tmp/256, PORTB);
```

```
while (1);
```

In code produced by the avr gcc port, this loop is around 75 instructions and 80 cycles if <generate white> is quite crude, eg white = ADDP + 16 * white;

and around 75 instructions and 80+m+d cycles if <generate white> is less crude, eg white = (ADDP + MULP * white) % MODP;

where m=cycles for rcall __mulhi3, d=cycles for rcall __udivmodhi4,

Re: randomized white noise = white noise?

and ADDP, MULP, MODP are appropriate numbers, eg numbers that might look like 2539, 27943, and 65521. [I don't know if the avr gcc port has a random() function; if it does, it's probably better and faster.]

You can calculate the values the above code corresponds to. Eg, $b0 - b0/256$ is $0.99610 * b0$, vs. your desired $0.99765 * b0$, while $3 * white/16$ is $white * 0.1875$ vs. desired $white * 0.1848$, and so forth. Also, the loop is fast enough that it should be ok with more filter stages (b3=..., b4=...). Perhaps that would give you more leeway with the constants.

-jiw

.