

Re: How inaccurate is a 555 or 7555 REALLY?

Re: How inaccurate is a 555 or 7555 REALLY?

Source: <http://sci.tech-archive.net/Archive/sci.electronics.design/2006-12/msg01999.html>

- *From:* bill.sloman@xxxxxxxx
 - *Date:* 8 Dec 2006 14:06:22 -0800
-

John Fields wrote:

On 7 Dec 2006 17:30:41 -0800, bill.sloman@xxxxxxxx wrote:

John Fields wrote:

On 6 Dec 2006 16:18:51 -0800, bill.sloman@xxxxxxxx wrote:

John Fields wrote:

On 5 Dec 2006 21:21:13
-0800,
bill.sloman@xxxxxxxx
wrote:

Phil Allison
wrote:

<bill.sloman@xxxxxxxx>

John
Fields:

<snip>

Yes. I've seen them. All very nostalgic. Your recent exercise with a 4024 (the last post in this thread) reminds

Re: How inaccurate is a 555 or 7555 REALLY?

me of stuff I was doing in
1974 with a 4040, though I didn't decode
with three diodes and a
resistor, even back then.

Didn't know how? It's just an RDL AND...\$

Of course I knew how. But the resistors I would have had to use were
0.6" long and 0.2" wide on the board, so it would have wasted space,

Really?

ISTR that 1/8 watt resistors and 1N914s were available back then, so
the entire decoder would have used up the same real estate as an 8
pin DIP, as well as not using nearly the amount of power the TTL
glue you'd have to otherwise make the decoder from would.

1N914s weren't a problem, but company policy had fixed on one style of
600mW Philips metal film resistor, and 1/8 watt resistors weren't an
option.

not to mention confusing the final test technicians and the service
engineers.

LOL, you design to keep from confusing folks?

Of course. Unconfusing them takes time and costs money. All other
things being more or less equal, I'll go for the transparent design
every time.

A more modern programmable logic part –
like the Xilinx Coolrunner
series – with in-system programming, 1.8V
supply and some really
compact packages – looks as if it would be
even nicer, but I've not yet
been able to contrive an excuse to play with

Re: How inaccurate is a 555 or 7555 REALLY?

Re: How inaccurate is a 555 or 7555 REALLY?

one of these parts.

You seem to be gravitating toward the "use a PIC" mentality when, for the OP's purpose, suggesting that he use anything he'd have to learn to program would be ridiculous. But that's never stopped you before, huh?

A PIC may be programmable, but it isn't programmable logic, and can't do a lot of things that are easy in programmable logic.

Programming programmable logic can be handled through a variety of interfaces, depending on the manufacturers programming software (usually available free for everything except bleeding-edge new parts, which the Xilinx CoolRunner isn't – Xilinx bought it from Philips).

Most of the ones I've run across have included a graphical option, which makes the process similar to hooking up logic gates and bistables.

I prefer to write out a series of logic equations.

I didn't find any of these approaches difficult to learn – nothing remotely as difficult as mastering a computer language, where it took me a week to learn Fortran 4, and nearly a year before I could produce a well formatted page of output (including printed graphs). The MACRO-8 assembly language for the PDP-8 was much easier – mainly because the teletype output made more sense.

There is a certain learning curve in learning how to use Boolean logic, and you need extra time learn how to hook up CMOS parts to realise that logic – most people would be better off putting the extra time into learning how to program programmable logic.

The choice of which programmable part should probably be referred to comp.arch.fpga, but one thing is for sure – you will be able get all the logic into a single modern PLD.

All of which is irrelevant, not to mention boring.

So you don't use programmable logic parts. Why am I not surprised.

Re: How inaccurate is a 555 or 7555 REALLY?

Re: How inaccurate is a 555 or 7555 REALLY?

The glue logic approach was the appropriate way to solve the OP's problem. End of story.

If you say so John. But you do look a bit silly, bent over, with your head buried in the sand.

—

Bill Sloman, Nijmegen (but in Sydney at the moment)

.