From uhclem Thu Sep 6 16:12:43 1984
To: bc cutler gordon iv ron uhclem tbul foggy!bc sneaky!gordon vaxnix!ron
From: trsvax!uhclem (Frank Durda IV)
Sent: Thu Sep 6 1984, 16:12:24 CDT
Subject: More benchmarks (snore)

I have reached a checkpoint in the improvement of the serial control system and in clock management on the Z80. Here is what I have to show at this point in the game: (all times are in seconds)

	waits 1.3.2	no/w 1.3.2	waits 1.3.3+	no/w 1.3.3+	waits 1.3.3++	no/w 1.3.3++	
Test 1	312	294	270	258	251	241	(seconds)
(delta)		+18	+42	+54	+61	+71	
Test 2	270	253	235	221	223	211	(seconds)
(delta)		+17	+35	+49	+47	+59	
Test 3	661	668	686	697	687	690	(seconds)
(delta)		-7	-25	-36	-26	-29	
Test 4	269	253	234	222	222	209	(seconds)
(delta)		+16	+35	+47	+47	+60	

waits = 2 wait states per Z80 opcode fetch
no/w = no wait states (on the Z80 side)

1.3.3+ is new serial code, larger fifo, new z80 clock, and 68k clock fudge
1.3.3++ is the same as 1.3.3+ with new video from Tom B.

Test 1 is /bin/time cat /usr/dict/words

Test 2 is /bin/time cat test.file
  (test.file is created via: cat /usr/dict/word | tr '\012' ' '
  >& test.file)

Test 3 is /bin/time cat /usr/dict/words | tr '\012' '\011'

Test 4 is /bin/time cat tab.file
  (tab.file is created via: cat /usr/dict/word | tr '\012' '\011'
  >& tab.file)

An interesting item above is that Test 3 takes longer on machines that do not have wait states. I assume this is caused by rotational latency in picking up data from the hard disks. Test 4 eliminates as much of the "on-the-fly" processing from test 3 and outputs the same data. It's results show that the machine without wait states executes the video/serial code under test faster.

It should also be pointed out that the speeds in column 1 should be the same as those collected by Tom B. For some reason, they are about 20 seconds lower. All of my tests were run on a 512k Model 16a, 1

15 meg drive and 1 meg of swap on that drive.

At any rate, the above results show that by combining the new video and serial code, we can see a 20% speed improvement and if you throw in a wait-stateless machine, that goes up to 23%.

Comments?

Frank Durda IV