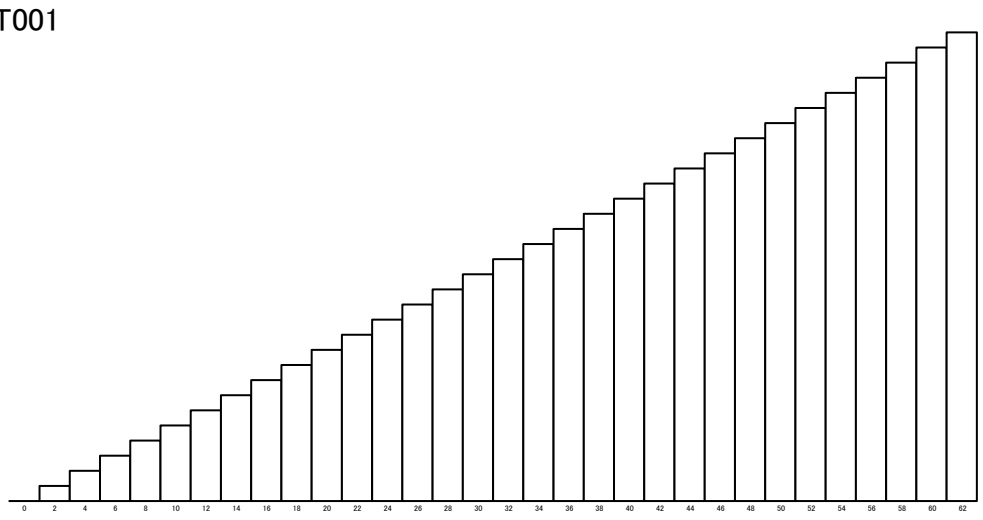
TEST001



440Hz

1/440[sec] / 32[sample] / (1[sec]/3579000[Hz])
= 3579000/(440*32)
= 254.1903...

254[clock/sample]
= 1/(254 * 32 *(1/3579000)) [Hz]
= 3579000 / (254*32)
= 440.32972...
→ 1[sample]あたり 254[clock] で処理すれば 440.32972..[Hz] で 32[sample] 周期となる。

設定値は、253,252,251, ... , 0, 253, 252, .. と繰り返す設定になるので 253 。

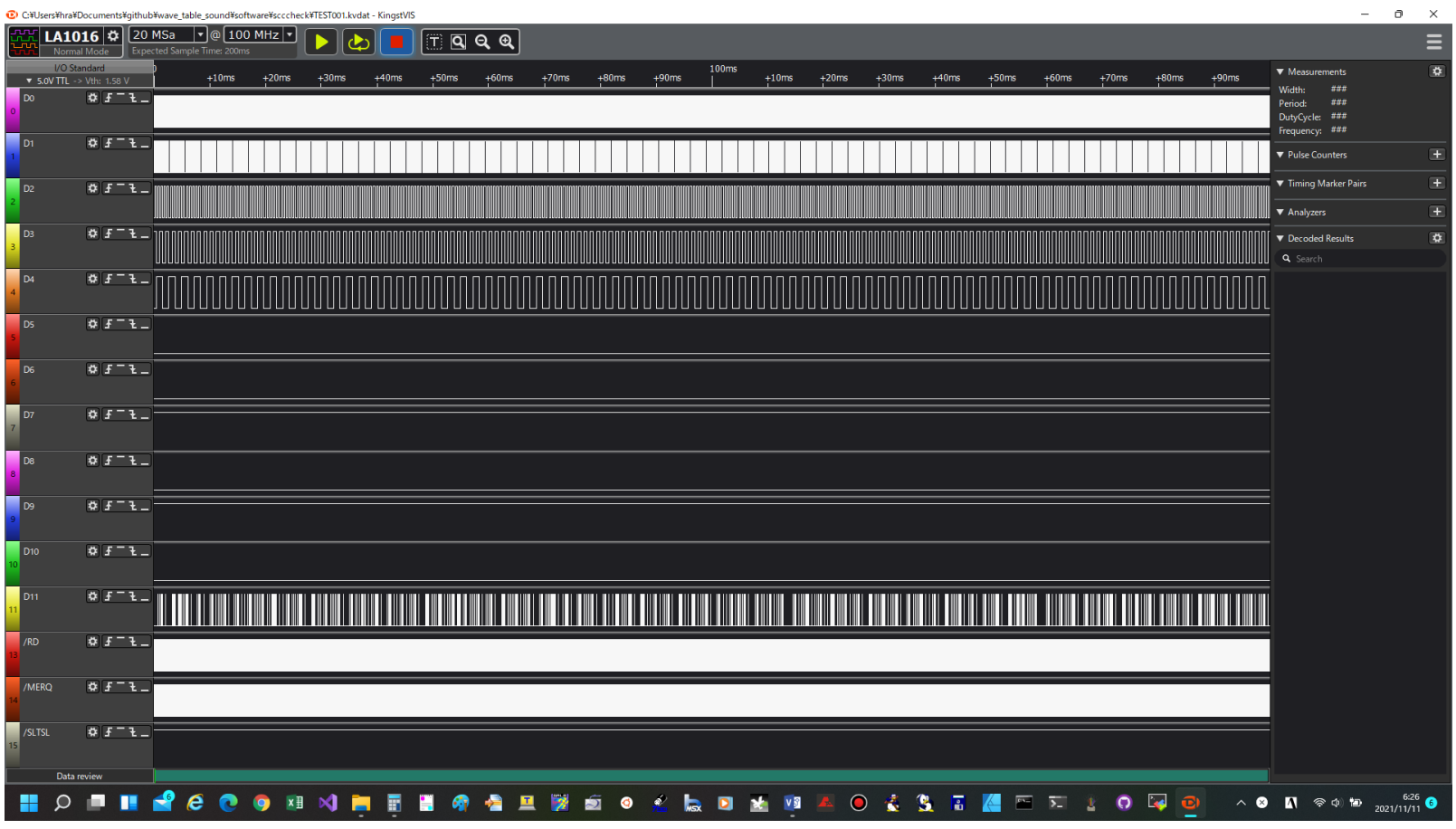
波形メモリに設定する値	出力値	volume=15 の場合の出力値
127	((127 * volume) >> 4) + 128	247
:	:	:
0	128	128
:	:	:
-128	((-128 * volume) >> 4) + 128	8

音量指定は 8 にする。

(波形 * 音量) >> 4 が出力されるので、8 にすると 50% で出力される。
つまり、0,1,2,3,4, .. , 31, 0,1,2,3 ... が出力期待値となる。

1[sample] は、254[clock] になるので、その幅(時間)は、
(1[sec]/3579000[Hz]) * 254[clock] = 70.9[μ sec]

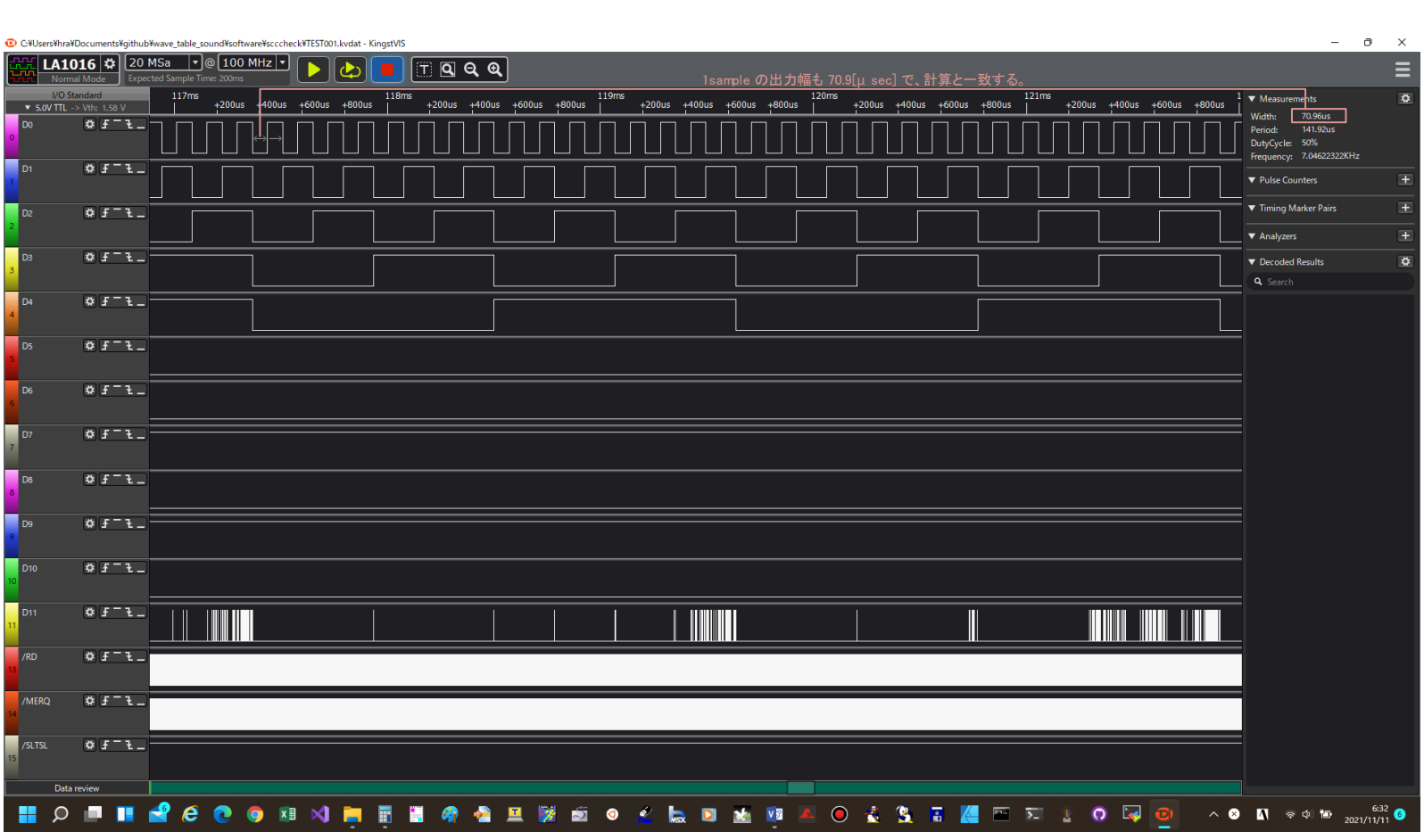
Ch.A だけ、こののこぎり波を音量8で出力。
Ch.B～Ch.E は、音量0で出力。



D10～D0 の 11bit が SCCの出力信号。

D[10:0] = 11'b010100?????

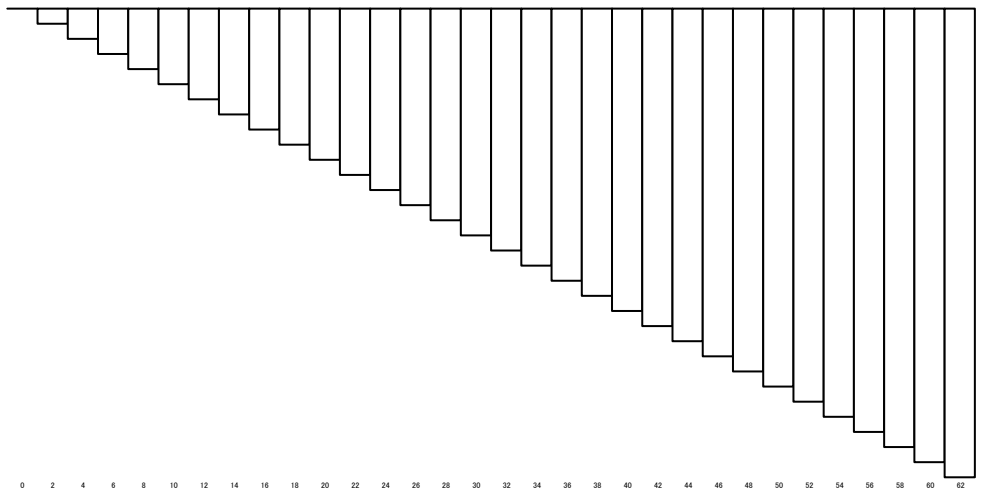
各チャネルの信号値 0 が 出力値 128 として出てくると考えると、Ch.A～E の 5ch が加算されて、ここに 0101 = 5 が出てくる。



D[10:0] = 11'b010100?????

波形を拡大してみると、????? の部分に 0,1,2,3, .. ,31,0,1, .. と出力されているのが確認できる。

TEST002



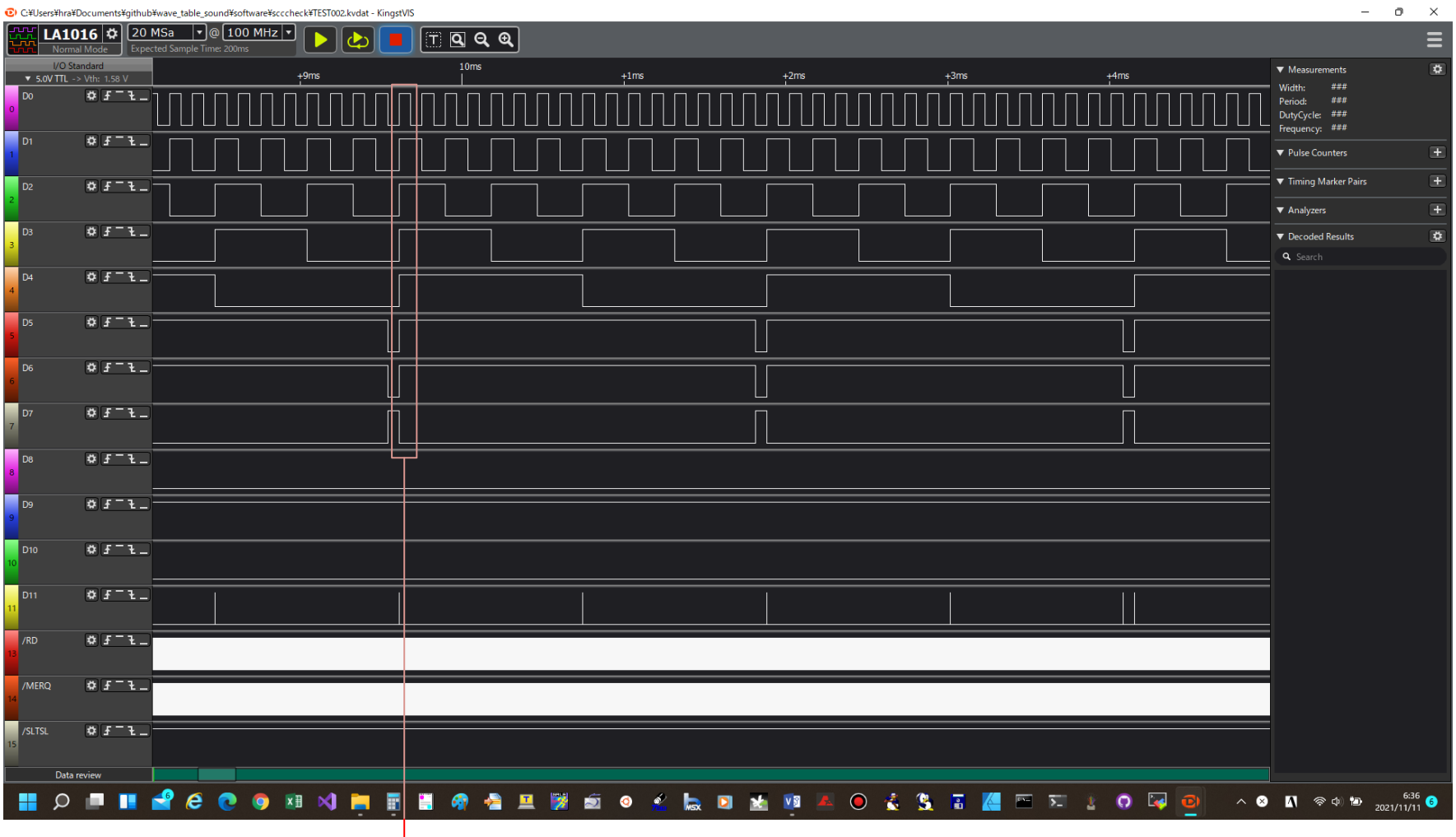
TEST002 は、波形を 0, -2, -4, .. と、TEST001に対して符号反転したものである。
8bit幅の2の補数形式なので、-2 は、254 と等価である。同様に -4 は 252、-6 は 250 となる。

((wave * volume) >> 4) + 128

この式が正しいのであれば、波形 -2 は、

((-2 * 8) >> 4) + 128
= (-16 >> 4) + 128
= -1 + 128
= 127

このことから、音量をかけた後の小数部を切り捨てる右シフトは、四捨五入などはせずに単純に下位ビットをカットしていることがわかる。



D[7:0] = 01111111 = 127

計算と一致する。

TEST003

波形は TEST001 と同じ。
再生中に波形の 5番地をひたすら読むテスト。

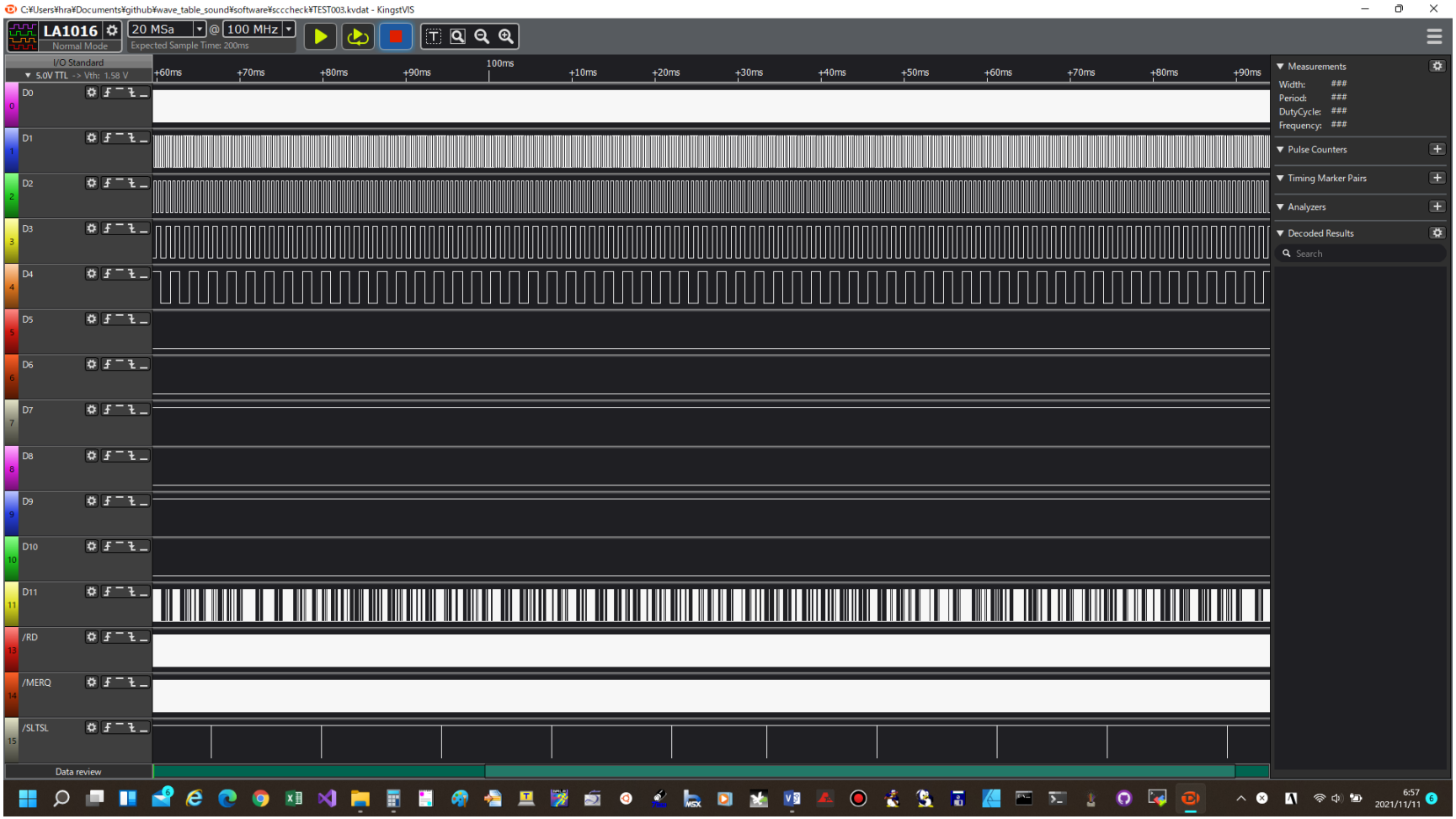
SCCに内蔵する波形メモリが 1RWタイプであれば、1clk 内でのアクセスは read 1つか write 1つのいずれかだけ。
CPUからの読み出しと、再生用の読み出しのタイミングがバッティングすると読み出し競合が起こるはず。

原理的には、波形が変化するタイミング＝周波数カウンタが 0 になったタイミングだけ読めばいいので、CPU読み出しの際に必ずしもバッティングするとは限らない。

波形メモリが、D-FF で構成されている場合、読み出しは無制限なのでこのバッティングは発生しないが、
SCCが登場した時期を考えると、32byte＊4ch もの記憶素子を D-FF で構成するのは規模的にあり得ないと推定。

そもそも同期SRAMかどうかも怪しいので、次の1byteは事前に先読みしておいて、必要になったタイミングで「はいどうぞ」と必要な回路に渡しているかもしれない。
そうすると、CPUからの読み出し要求があった場合、そちらを優先して、先読み処理を遅らせるようにしていたらバッティングによる波形出力乱れは発生しないと思われる。

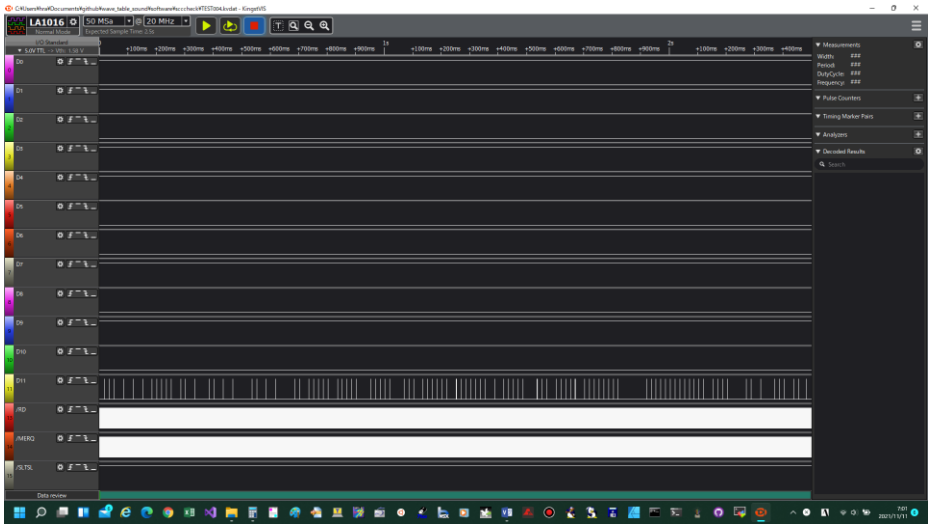
実際に、波形に乱れが無いことを確認できる。/SLTSL = 0 になっているタイミングがリードしているタイミング 。



TEST004

非同期のSRAMを使っていて、SRAMの読み出しに時間が掛かる（パイプライン化できない）として、
さらに CPU とのアクセス競合を防ぐ仕組みが入っているとすれば、
毎サイクル読まねばならなくなる「波長 0」という設定は不可能で、
時間が掛かる分も考慮すると極端に短い波長は実現できないことになる。

TEST004 では、TEST001 とほぼ同じ設定で、波長の設定だけ 8 にしてみたものである。

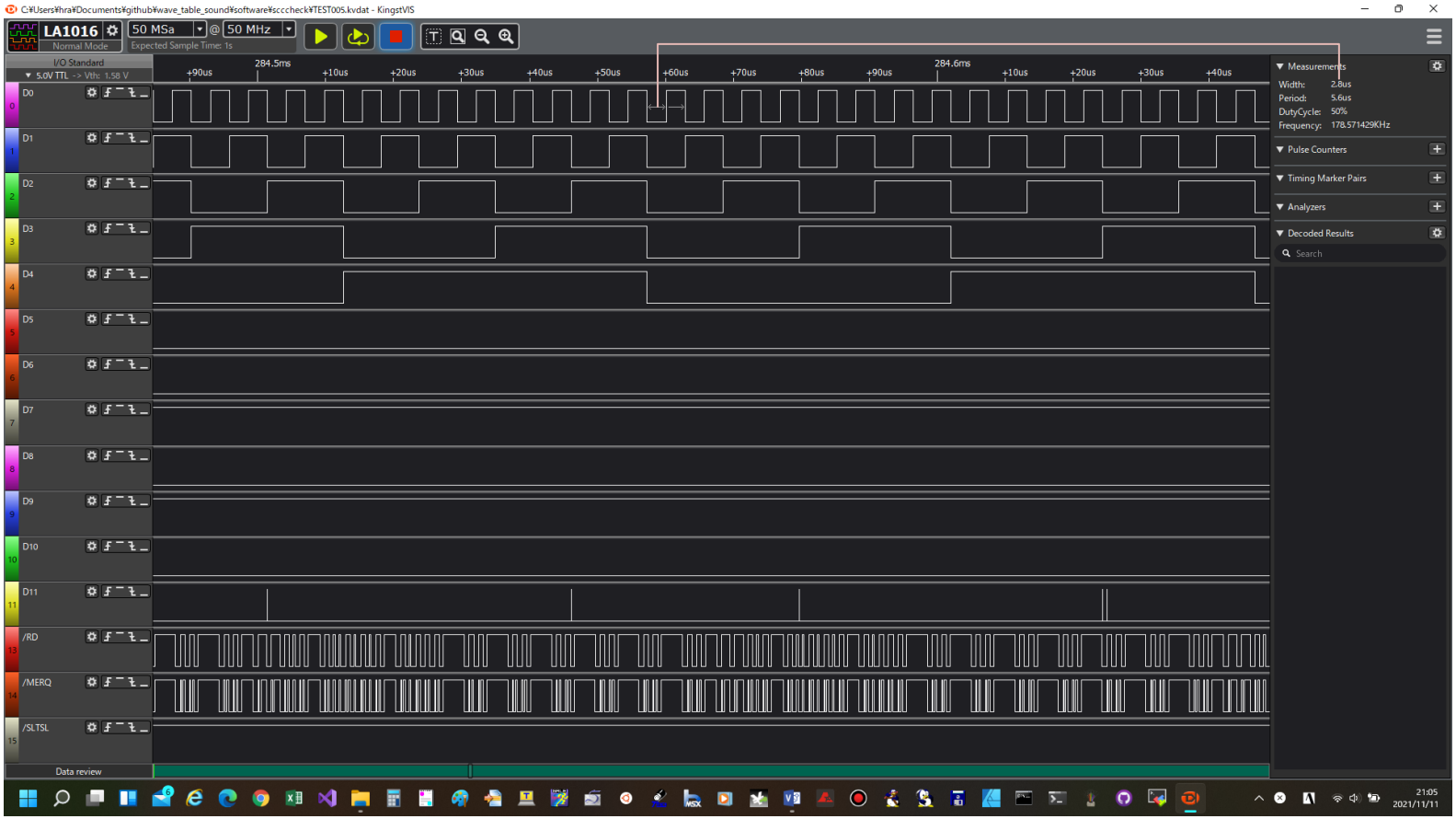


出力値が固定値に張り付くことを確認できる。
いろいろ値を変えてみると、0〜8 だと固定値 11 に張り付く。この 11 の意味は不明。

9 にすると、期待通りの波形が出てくるようになる。

TEST005

TEST004とほぼ同じで、波長を 9 にした場合。

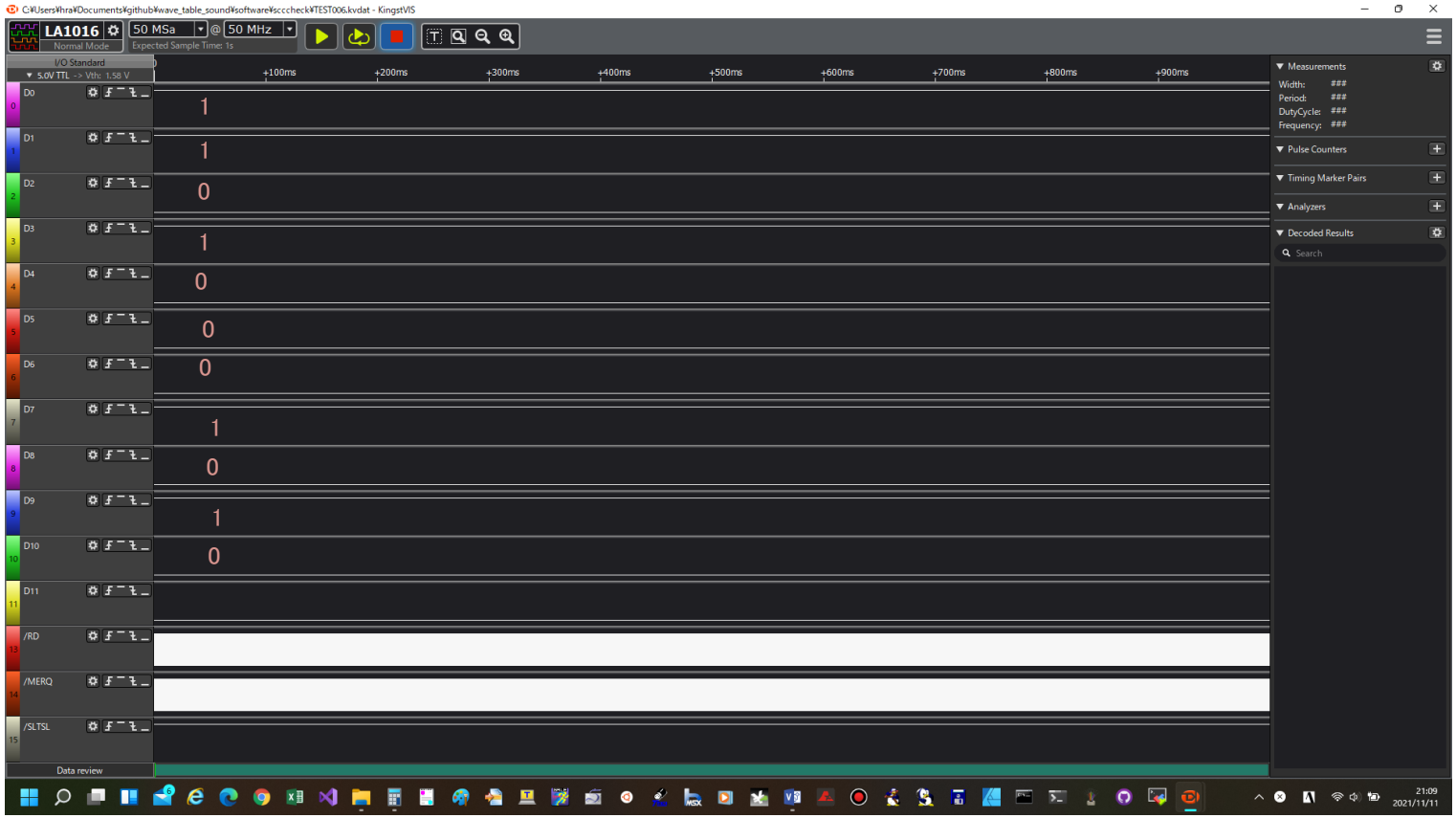


1[sample] = 2.8[μ sec] となっており、10[clk] の幅と一致する。期待通り。

波長が9 なのでカウンターは、9,8,7,6,5,4,3,2,1,0 の 10通りの値をとる。
(1000000[μ sec] / 3579000[Hz]) * 10[clk] = 2.79[μ sec]

TEST006

TEST004とほぼ同じで、波長を 0 にした場合。

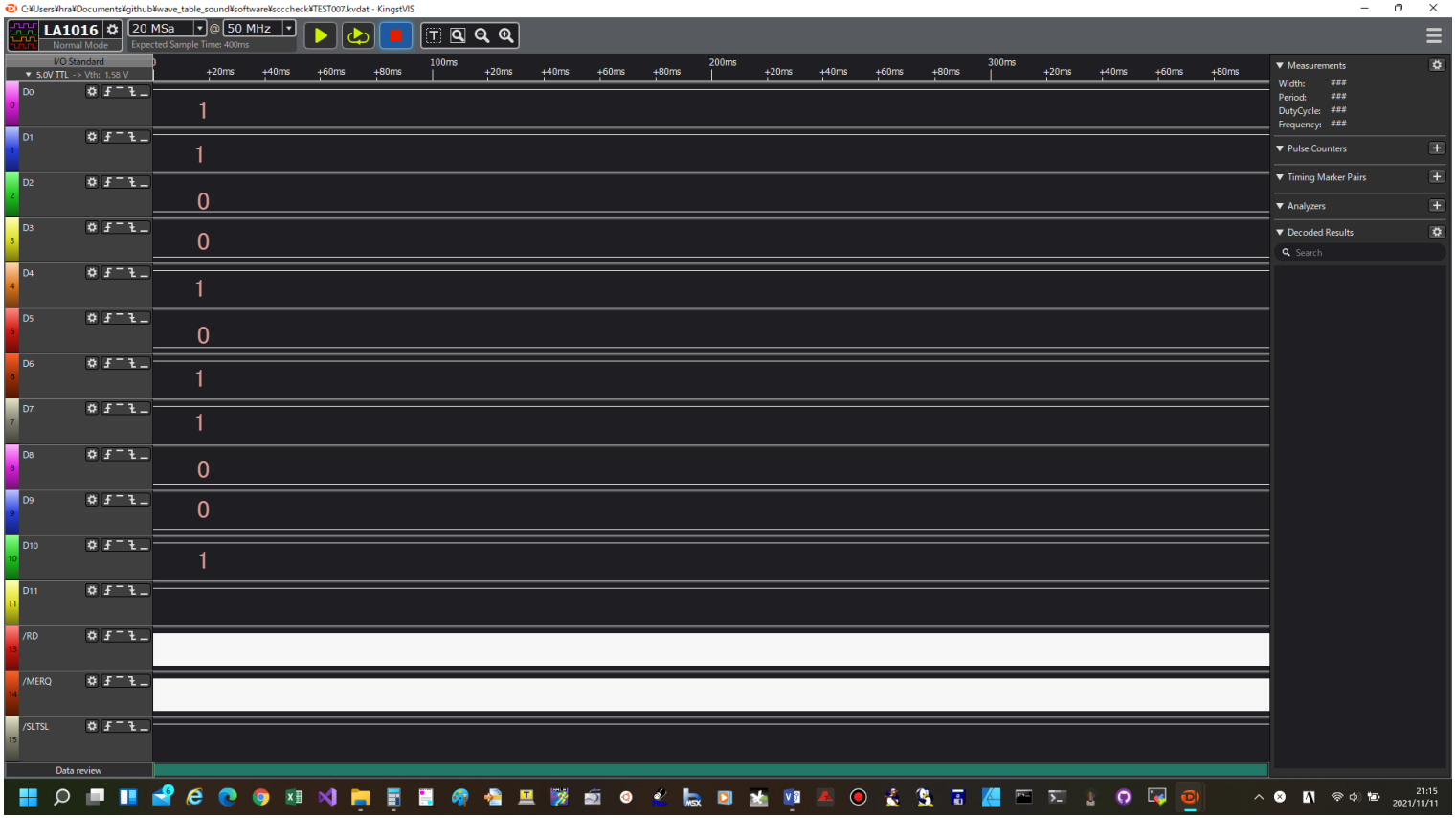


11'b01010001011

仮に、MUTE してる Ch.B～Ch.E が 0 だとすると、出力値は 128 となるため、
Ch.A の出力は 1011 (2進数) = 11 (10進数)
この値が何者なのか、この時点ではわからない。

TEST006

4ch の波形メモリをすべて 127 で埋めて、5ch の音量をすべて 15 にした場合。



11'b10011010011 = 1235

5ch すべて同じ値が出力されていると仮定すれば、5で割った値が 1ch の出力となる。

1235 / 5 = 247

+128 のオフセットが掛かっているので、

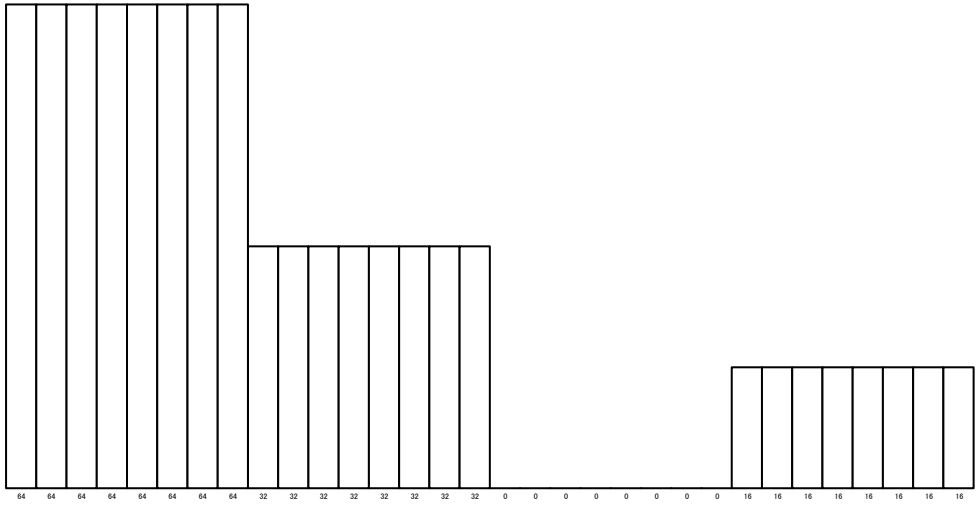
247 - 128 = 119

波形メモリは 127 を敷き詰めており、音量は 15 にしているので、

127 * 15 >> 4 = 119

で、一致する。

TEST008



Ch.D, E の波形メモリに上記波形を設定する。
Ch.D, E は有効(MUTE解除)して、Ch.Dは音量8, Ch.Eは音量0 にする。

sample31 の 16出力と、sample0 の 64出力が、
丁度 1sample分の 10clk で被っているタイミングがある。
(写真の A1-A2)

さらに、64出力の矩形が 10sample分の幅を持っている。
波形メモリ上では 8sample なので、この差が不明。