double	ref_freq	250.00	Hz	
uint16_t	DAQ_period	50	us	*Must be int because it is the period for the interrupt service routine (ISR)*
		5.00E-05	s	
double	DAQ_rate	20000.00	Hz	
uint16_t	N_LUT	32768	samples	*Ensure N_LUT is power of 2 and we can use bitwise `&` to perform a fast integer modulo, see MULMOD*
uint16_t	BLOCK_SIZE	2000	samples	
double	BLOCK_RATE	10.00	blocks/s	
double	LUT_idx / DAQ_iter	409.60		*Fractional, non-ideal. Must be rounded to integer to be able to use in MULMOD.*
uint16_t	**ideal** LUT_idx / DAQ_iter	410		*Rounded to nearest integer.*
double	**ideal** ref_freq	250.24	Hz	*Hence, we must adjust ref_freq to reflect optimal value.*

MULMOD on integers and make sure modulus is power of 2 --> super fast, no integer overflow
https://www.geeksforgeeks.org/how-to-avoid-overflow-in-modular-multiplication/
https://stackoverflow.com/questions/12168348/ways-to-do-modulo-multiplication-with-primitive-types/12171020

		non-ideal ref_freq						
DAQ_iter	t [ms]	x = ref_freq*t	REF_X = sin(2*pi*x)	non ideal LUT_idx	non ideal modulo	ideal LUT_idx	ideal modulo	
##########	1.07E+08	2.68E+07	#NUM!	879610530611.20	2867.20	###########	17174	<-- Large iter value
0	0.000	0.0000	0.00	0.00	0.00	0	0	<-- Small iter values, starting at 0
1	0.050	0.0125	0.08	409.60	409.60	410	410	
2	0.100	0.0250	0.16	819.20	819.20	820	820	
3	0.150	0.0375	0.23	1228.80	1228.80	1230	1230	
4	0.200	0.0500	0.31	1638.40	1638.40	1640	1640	
5	0.250	0.0625	0.38	2048.00	2048.00	2050	2050	
6	0.300	0.0750	0.45	2457.60	2457.60	2460	2460	
7	0.350	0.0875	0.52	2867.20	2867.20	2870	2870	
8	0.400	0.1000	0.59	3276.80	3276.80	3280	3280	
9	0.450	0.1125	0.65	3686.40	3686.40	3690	3690	
10	0.500	0.1250	0.71	4096.00	4096.00	4100	4100	
11	0.550	0.1375	0.76	4505.60	4505.60	4510	4510	
12	0.600	0.1500	0.81	4915.20	4915.20	4920	4920	
13	0.650	0.1625	0.85	5324.80	5324.80	5330	5330	
14	0.700	0.1750	0.89	5734.40	5734.40	5740	5740	
15	0.750	0.1875	0.92	6144.00	6144.00	6150	6150	
16	0.800	0.2000	0.95	6553.60	6553.60	6560	6560	
17	0.850	0.2125	0.97	6963.20	6963.20	6970	6970	
18	0.900	0.2250	0.99	7372.80	7372.80	7380	7380	
19	0.950	0.2375	1.00	7782.40	7782.40	7790	7790	
20	1.000	0.2500	1.00	8192.00	8192.00	8200	8200	
21	1.050	0.2625	1.00	8601.60	8601.60	8610	8610	
22	1.100	0.2750	0.99	9011.20	9011.20	9020	9020	
23	1.150	0.2875	0.97	9420.80	9420.80	9430	9430	
24	1.200	0.3000	0.95	9830.40	9830.40	9840	9840	
25	1.250	0.3125	0.92	10240.00	10240.00	10250	10250	
26	1.300	0.3250	0.89	10649.60	10649.60	10660	10660	
27	1.350	0.3375	0.85	11059.20	11059.20	11070	11070	
28	1.400	0.3500	0.81	11468.80	11468.80	11480	11480	
29	1.450	0.3625	0.76	11878.40	11878.40	11890	11890	
30	1.500	0.3750	0.71	12288.00	12288.00	12300	12300	
31	1.550	0.3875	0.65	12697.60	12697.60	12710	12710	
32	1.600	0.4000	0.59	13107.20	13107.20	13120	13120	
33	1.650	0.4125	0.52	13516.80	13516.80	13530	13530	
34	1.700	0.4250	0.45	13926.40	13926.40	13940	13940	
35	1.750	0.4375	0.38	14336.00	14336.00	14350	14350	
36	1.800	0.4500	0.31	14745.60	14745.60	14760	14760	
37	1.850	0.4625	0.23	15155.20	15155.20	15170	15170	
38	1.900	0.4750	0.16	15564.80	15564.80	15580	15580	
39	1.950	0.4875	0.08	15974.40	15974.40	15990	15990	
40	2.000	0.5000	0.00	16384.00	16384.00	16400	16400	
41	2.050	0.5125	-0.08	16793.60	16793.60	16810	16810	
42	2.100	0.5250	-0.16	17203.20	17203.20	17220	17220	
43	2.150	0.5375	-0.23	17612.80	17612.80	17630	17630	
44	2.200	0.5500	-0.31	18022.40	18022.40	18040	18040	
45	2.250	0.5625	-0.38	18432.00	18432.00	18450	18450	
46	2.300	0.5750	-0.45	18841.60	18841.60	18860	18860	
47	2.350	0.5875	-0.52	19251.20	19251.20	19270	19270	