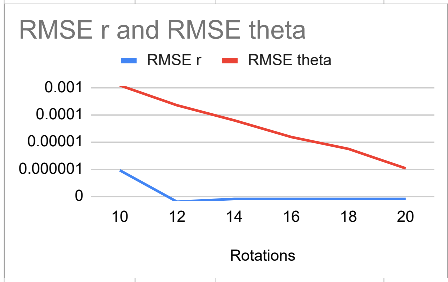
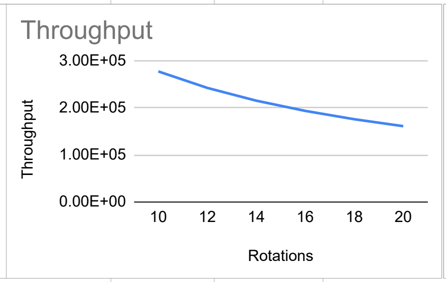
Question 1: One important design parameter is the number of rotations. Change that number to numbers between 10 and 20. This question should use a floating point implementation of CORDIC.

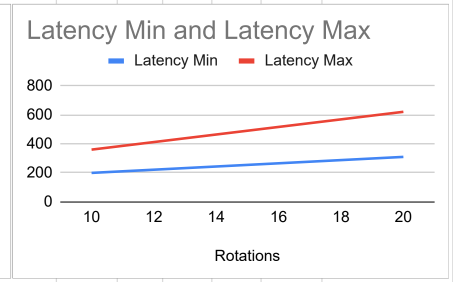
a) Create a table that shows resource usage, throughput, latency, and RMSE for each design you create. Use 10, 12, 14, 16, 18, and 20 rotations. You will need to add additional values to the table for 18 and 20 rotations. Chapter 3 has enough information to help you derive the additional angles and Kvalues.

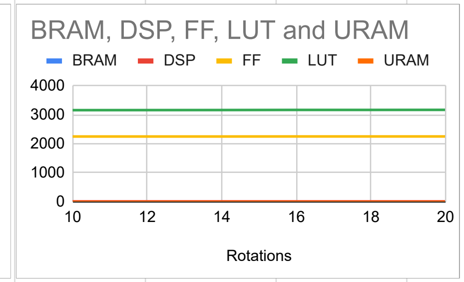
Rotations	Clock	RMSE r	RMSE theta	Latency Min	Latency Max	Throughput (latency*period)^-1 samples per second	BRAM	DSP	FF	LUT	URAM
10	1.00E-08	0.00000089694646	0.001191147021	200	360	2.78E+05	0	21	2251	3157	0
12	1.00E-08	0.000000062624721	0.0002210365928	222	412	2.43E+05	0	21	2251	3159	0
14	1.00E-08	0.000000079435864	0.00006164732622	244	464	2.16E+05	0	21	2251	3161	0
16	1.00E-08	0.000000079435864	0.00001499664722	266	516	1.94E+05	0	21	2253	3165	0
18	1.00E-08	0.000000079435864	0.000005490659987	288	568	1.76E+05	0	21	2253	3165	0
20	1.00E-08	0.000000079435864	0.000001056979272	310	620	1.61E+05	0	21	2253	3167	0

b) Plot throughput, resource usage, and RMSE (theta and r on the same plot) as a function of the number of rotations. Clearly label your axes and each datapoint.









c) At what number of rotations does the accuracy stop noticeably improving in the plot?

12 rotations

Question 2: Another important design parameter is the data type of the variables.

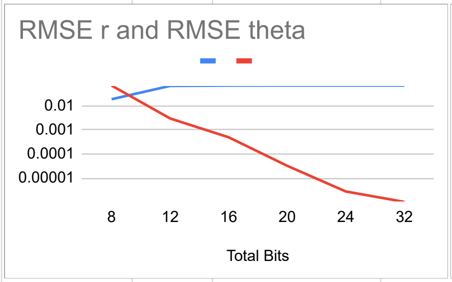
a) We will use the ap_fixed arbitrary precision data type for each variable. At most how many integer bits are required for each variable? Remember that this is a signed type. (Hint: consider the range of values that each variable can take on. You can use the float implementation to help you determine this. Think of the range of values of the variables r, x, y, and theta). Give an answer for each variable. The testbench assumes that x and y are normalized between [-1, 1].

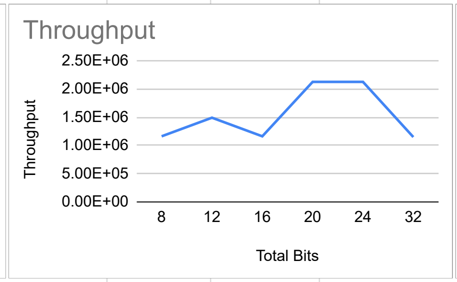
ap_fixed<W,I> covers values of -2^(I-1) - 2^-(W-I)
For x,y of range -1,1: 1 integer bit required
For r of range 0,1.414, scaled at K rotations (K=20 max) = r * K = 2.328: 3 integer bits required
For theta of range -pi,pi: 3 integer bits required
For Kvalues of range 0.0000...., 1: 1 integer bit required
For angles of range 0, 0.785: 1 integer bit required

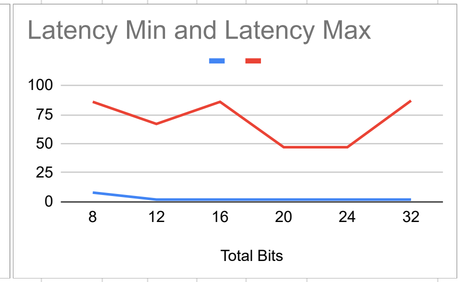
b) Now that you have fixed the number of integer bits (use the largest number of integer bits determined in **2a),** experiment with the number of total bits for each variable. Use the datatype for each variable. Create a table that shows resource usage, throughput, latency, and RMSE for each design you create. Create one plot each for resource utilization and RMSE vs total bits. Use 8, 12, 16, 20, 24, and 32 total bits.

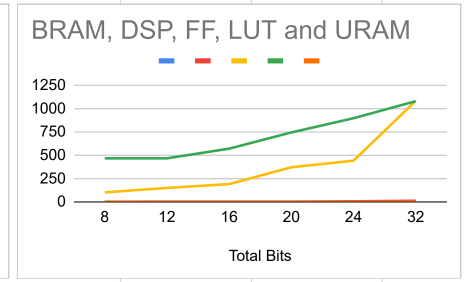
Total Bits	Clock	RMSE r	RMSE theta	Latency Min	Latency Max	Throughput (latency*period)^-1 samples per second	BRAM	DSP	FF	LUT	URAM
8	1.00E-08	0.0184454564	0.0676920563	8	86	1.16E+06	0	2	102	468	0
12	1.00E-08	0.06563318521	0.002942261053	2	67	1.49E+06	0	3	150	468	0
16	1.00E-08	0.06770890206	0.0004881987697	2	86	1.16E+06	0	3	191	572	0

20	1.00E-08	0.06795471162	0.00003220740473	2	47	2.13E+06	0	3	372	746	0
24	1.00E-08	0.0679717809	0.000002787146059	2	47	2.13E+06	0	6	443	899	0
32	1.00E-08	0.06797295064	0.000001032528871	2	87	1.15E+06	0	12	1086	1082	0



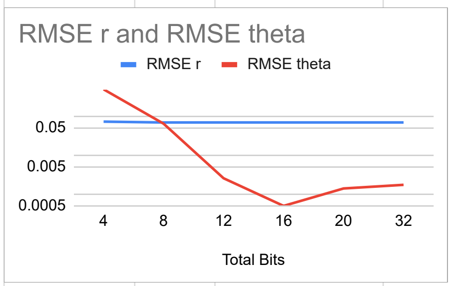


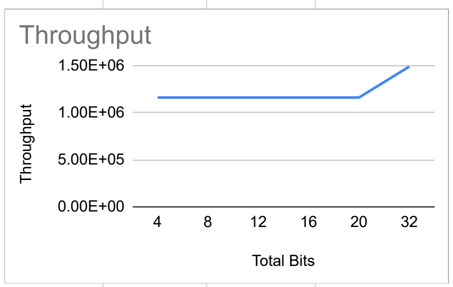


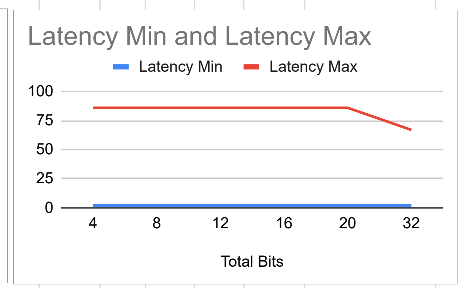


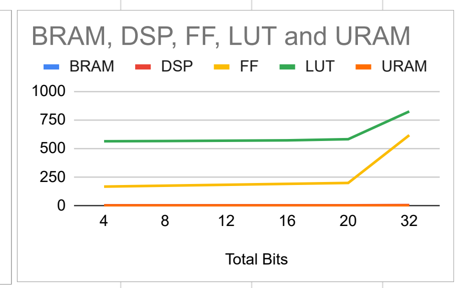
c) Use ap_fixed<16,3> for all variables. Now experiment with changing the type of **only** the CORDIC rotation tables (Kvalues and angles). Create a table that shows resource usage, throughput, latency, and RMSE for each design you create. Use 4, 8, 12, 16, 20, and 32 total bits. Also plot RMSE (one plot) as a function of the total number of bits for the data type.

Total Bits	Clock	RMSE r	RMSE theta	Latency Min	Latency Max	Throughput (latency*period)^-1 samples per second	BRAM	DSP	FF	LUT	URAM
4	1.00E-08	0.07104998082	0.4793570936	2	86	1.16E+06	0	3	167	564	0
8	1.00E-08	0.06757713109	0.0629594326	2	86	1.16E+06	0	3	175	566	0
12	1.00E-08	0.06770890206	0.002500157803	2	86	1.16E+06	0	3	183	569	0
16	1.00E-08	0.06770890206	0.0004881987697	2	86	1.16E+06	0	3	191	572	0
20	1.00E-08	0.06770890206	0.001367608551	2	86	1.16E+06	0	3	199	582	0
32	1.00E-08	0.06770890206	0.001695876708	2	67	1.49E+06	0	5	617	825	0









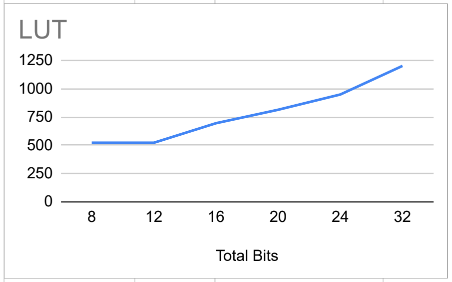
Question 3: What is the effect of using simple operations (add and shift) in the CORDIC as opposed to multiply and divide?

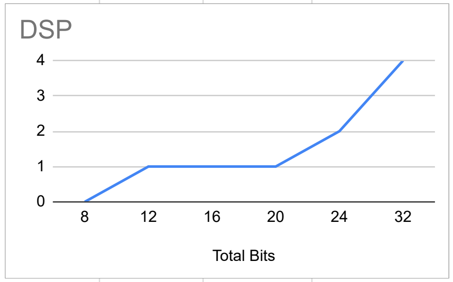
a) Now that you are using ap_fixed for all variables, change your implementation to use simple operations like add and shift instead of multiply and divide. Create a table that shows resource usage, throughput, latency, and RMSE for each design you create. Use 8, 12, 16, 20, 24, and 32 total bits. Use the implementation from 2b as a baseline for comparison. **You may keep the final multiplication to account for gain.**

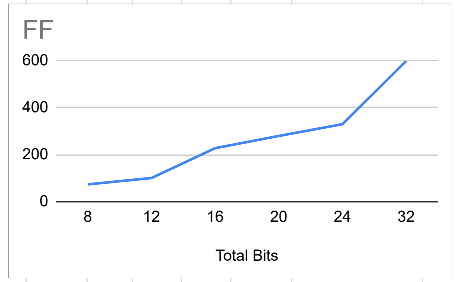
Total Bits	Clock	RMSE r	RMSE theta	Latency Min	Latency Max	Throughput (latency*period)^-1 samples per second	BRAM	DSP	FF	LUT	URAM
8	1.00E-08	0.1602807939	0.06341259181	2	27	3.70E+06	0	0	74	523	0
12	1.00E-08	0.07150160521	0.002536559012	2	27	3.70E+06	0	1	101	523	0
16	1.00E-08	0.06826100498	0.00039316123	2	46	2.17E+06	0	1	228	696	0
20	1.00E-08	0.06798420101	0.00001278748187	2	46	2.17E+06	0	1	280	815	0
24	1.00E-08	0.0679736957	0.000001458457064	2	46	2.17E+06	0	2	330	950	0

32	1.00E-08	0.06797296554	0.000001032528871	2	47	2.13E+06	0	4	599	1203	0

b) Create 3 separate plots for LUTs, DSPs, and FFs for each of these data types and each implementation that compares these results to the results from **2b**. Clearly label your axes and each datapoint. Use a different color/line style for each implementation.







Question 4: These questions all refer to the lookup table (LUT) implementation of the Cartesian to Polar transformation. This is in cordic_LUT/cordiccart2pol.cpp.

a) How does the input data type affect the size of the LUT? How does the output data type affect the size of the LUT? Precisely describe the relationship between input/output data types and the number of bits required for the LUT.

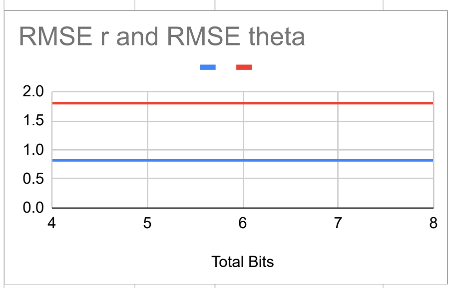
The input data type ap_fixed<W,I> increases LUT size exponentially as seen by 2^(2*W)
The output data type data_t does not affect LUT size, it only defines the size of values in the LUT

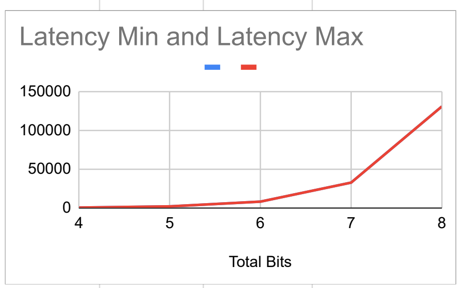
b) Create a table of resource usage, throughput, latency, and error vs number of total bits. Use the same number of integer bits for all data types (as in 2b). Use 4, 5, 6, 7, 8 total bits (8 bits will take some time to synthesize).

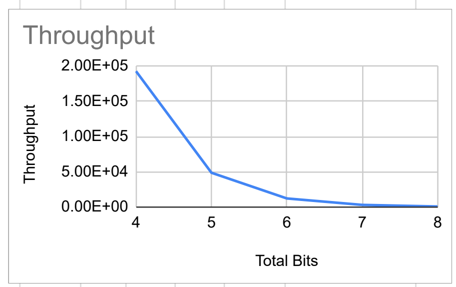
c) Plot all types of resource usage (LUTs, FFs, DSPs) as a function of the total number of bits for the data types. Make one plot for resource utilization

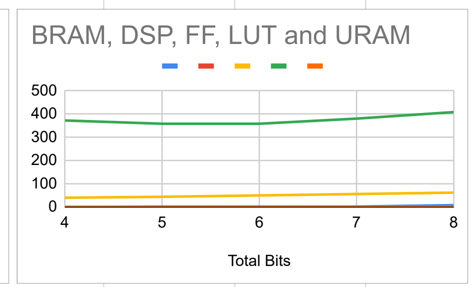
d) Plot RMSE as a function of the total number of bits for the data types.

Total Bits W, I	Clock	RMSE r	RMSE theta	Latency Min	Latency Max	Throughput (latency*period)^-1 samples per second	BRAM	DSP	FF	LUT	URAM
4	1.00E-08	0.8205698729	1.804754376	519	519	1.93E+05	0	0	40	372	0
5	1.00E-08	0.8205698729	1.804754376	2055	2055	4.87E+04	2	0	44	358	0
6	1.00E-08	0.8205698729	1.804754376	8199	8199	1.22E+04	2	0	50	358	0
7	1.00E-08	0.8205698729	1.804754376	32775	32775	3.05E+03	2	0	56	380	0
8	1.00E-08	0.8205698729	1.804754376	131079	131079	7.63E+02	8	0	62	408	0

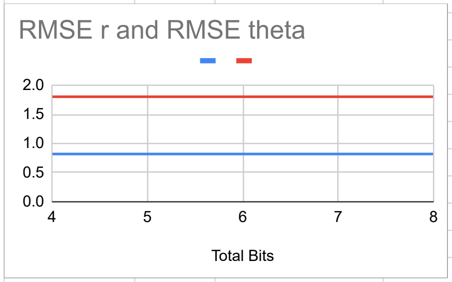


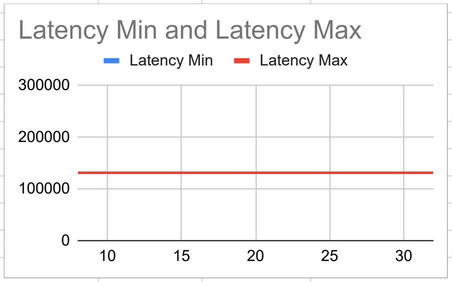


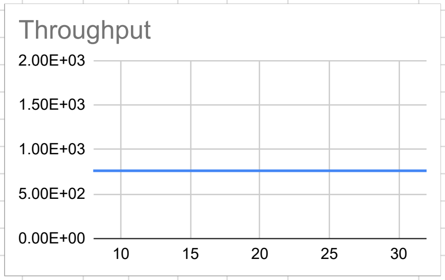


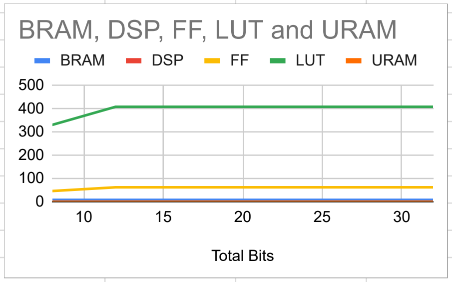


Total Bits ap_fixed	Clock	RMSE r	RMSE theta	Latency Min	Latency Max	Throughput (latency*period)^-1 samples per second	BRAM	DSP	FF	LUT	URAM
8	1.00E-08	0.8205698729	1.804754376	131079	131079	7.63E+02	8	0	46	330	0
12	1.00E-08	0.8205698729	1.804754376	131079	131079	7.63E+02	8	0	62	408	0
16	1.00E-08	0.8205698729	1.804754376	131079	131079	7.63E+02	8	0	62	408	0
20	1.00E-08	0.8205698729	1.804754376	131079	131079	7.63E+02	8	0	62	408	0
24	1.00E-08	0.8205698729	1.804754376	131079	131079	7.63E+02	8	0	62	408	0
32	1.00E-08	0.8205698729	1.804754376	131079	131079	7.63E+02	8	0	62	408	0









e) What advantages/disadvantages of the CORDIC implementation compared to the LUT-based implementation?

LUT Pros: Low resource utilization
LUT Cons: Low throughput / High Latency, Static
Cordic Pros: High throughput / Low Latency, High accuracy, Configurable
Cordic Cons: High resource utilization