SpecQ: Speculative Local Quantum Error Correction with Superconducting Priority Selection

Abstract—As quantum computers become more fault-tolerant, the resources required for error correction continue to rapidly grow as a function of the code size and number of logical qubits in the system. In particular, the communication bandwidth between different temperatures in the quantum computer is growing at a rate that is not sustainable. We observe that we can defer and reduce communication by greedily calculating error corrections and speculatively applying them with small hardware overhead in the low-temperature layers. At the cost of accuracy and program execution time, this allows for a greater scale in terms of logical qubits and shorter quantum program execution times.

A. Hardware eval

In order to verify the correctness of our hardware designs, we have created gate-level implementations of each part (complex flag generation, priority score encoding, temporal arbiter). These implementations were then verified using the pulse transfer simulator Pylse [1] using extensive model-based testing. To perform the verification, thousands of input testcases were generated by sampling the distributions we observed during training. The measured outputs of our circuit simulation for these cases were compared to the results predicted by higher order models. The majority of the gate models used in the implementations are based on the open source RSFQ cell library from Sun Magnetics [2], supplemented with cell models we implemented and evaluated using WRSpice [?], a SPICE program that supports Josephson Junctions, like the Muller C and inverted C elements that were tuned to have almost identical delays. In Figure 1 we show the plot of a simulation showing the pulses in the data input (x) and select output (sel) wires of a small arbiter. The arbiter selects the two largest priority scores out of four ($k = 2, n = 4$). A small number of operands was used for this simulation in order to keep the plot interpretable. These simulations additionally provide accurate numbers for the latency and JJ count of various circuit components. These are given in Table I and Table II. As shown, the area cost for generating priority scores is fairly low for each logical qubit at only a few thousand JJs, and the overhead of the arbiter stays at the order of tens of thousands when managing many logical qubits ($n = 256$). This hardware cost is low enough to make our design viable for today's limited integration density, which is the largest roadblock for hardware meant to operate at the 4 Kelvin environment. Additionally, the combined delay of both priority score generation and arbiter barely crosses the nanosecond mark for the largest sizes considered, which is negligible compared to the total decoding time.

TABLE I
Area and delay measurements for arbiters supporting various numbers of logical qubits.

k	n	JJ count	latency (ps)
2	64	6971	379
4	128	25793	588
8	256	83345	847

TABLE II
Area and delay measurements for complex flag generation (CPX) and encoding to priority score (ENC).

d	CPX JJ cost	ENC JJ cost	CPX latency	ENC latency
3	176	390	95	143
5	592	430	105	157
7	1244	490	109	164
9	2132	570	116	171
11	3256	670	116	171
13	4616	790	119	174
15	6212	930	119	174
17	8044	1090	126	181
19	10112	1270	126	181
21	12416	1470	126	181

References

[1] M. Christensen, G. Tzimpragos, H. Kringen, J. Volk, T. Sherwood, and B. Hardekopf, "Pylse: A pulse-transfer level language for superconductor electronics," in Proceedings of the 43rd ACM SIGPLAN International Conference on Programming Language Design and Implementation, 2022, pp. 671–686.

[2] C. J. Fourie, K. Jackman, J. Delport, L. Schindler, T. Hall, P. Febvre, L. Iwanikow, O. Chen, C. L. Ayala, N. Yoshikawa et al., "Results from the coldflux superconductor integrated circuit design tool project," IEEE Transactions on Applied Superconductivity, 2023.

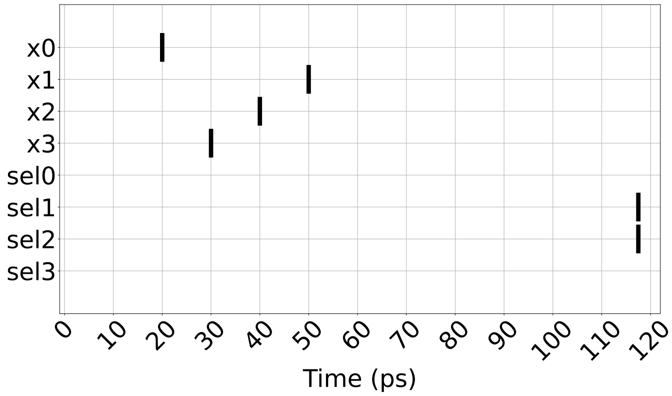

Fig. 1. Simulation of arbiter selecting the two highest scoring qubits out of four. The plot shows the time a pulse (indicated by a vertical bar) appears in each of the arbiter's ports. The score of each qubit is encoded in the arrival time of a pulse in the corresponing input port x0 to x3. After a short delay, the arbiter generates pulses at the sel ports of the qubits with the latest arriving inputs. In this case, sel1 and sel2.