Timing-Aware End-to-End Circuit Compilation Framework for Modular Quantum Systems

Ching-Yao Huang and Wai-Kei Mak

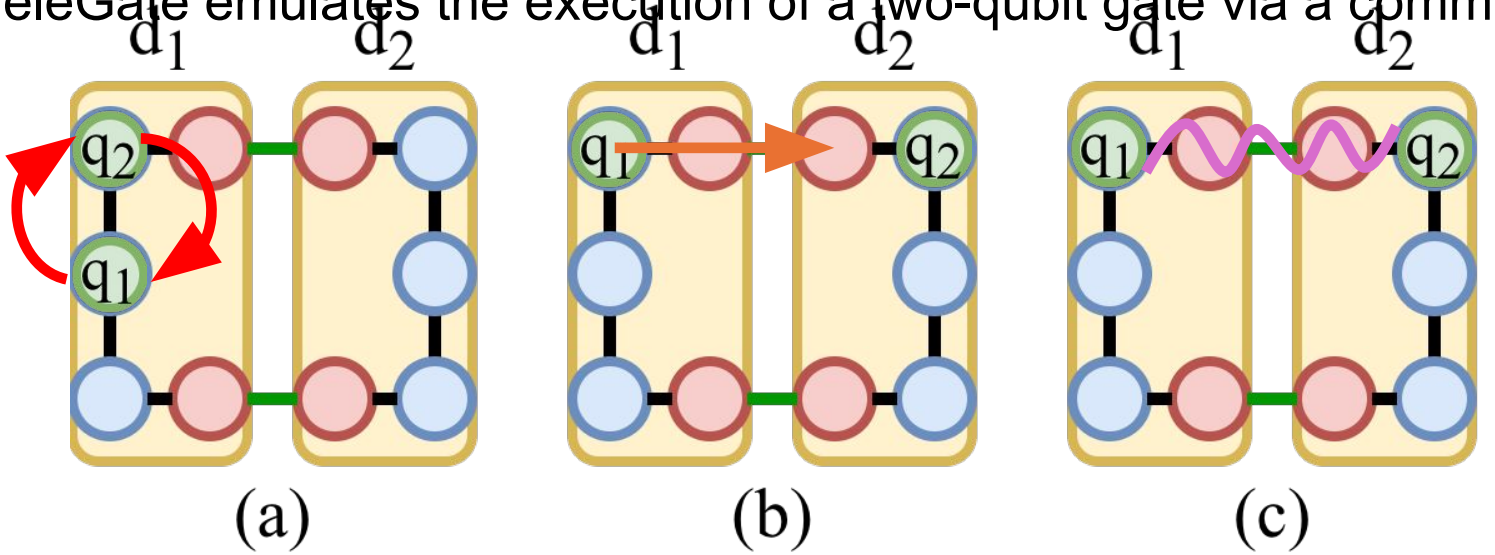
National Tsing Hua University, Taiwan

Introduction

- Fabrication challenges limit the scalability of a single QPU.
  - Congestion in qubit control wiring
  - Increased crosstalk among qubits
- Modular quantum system
  - Multiple QPUs are integrated into an interconnected quantum system.
  - Individual QPUs are connected through communication links.
- Compilation problem
  - Placement stage: map each logical qubit to a physical qubit.
  - Routing stage: insert inter- and intra-QPU operations to execute all the gates.

Preliminary

- Operations to move qubits
  - Intra-QPU operation: Swap
    - A Swap exchanges the mappings of two logical qubits within a QPU.
  - Inter-QPU operations: TeleData and TeleGate
    - A TeleData moves a logical qubit from one QPU to another via a communication link.
    - A TeleGate emulates the execution of a two-qubit gate via a communication link.



Preliminary

- Problem Formulation
  - Input: a modular quantum system and a logical circuit.
  - Output: a physical circuit.
    - The physical circuit is functionally equivalent to the logical circuit and can be executed on the system.
  - Objective: minimize the physical circuit latency.
    - We assume an as-soon-as-possible execution model.
    - The finish time of a gate g_ℓ:

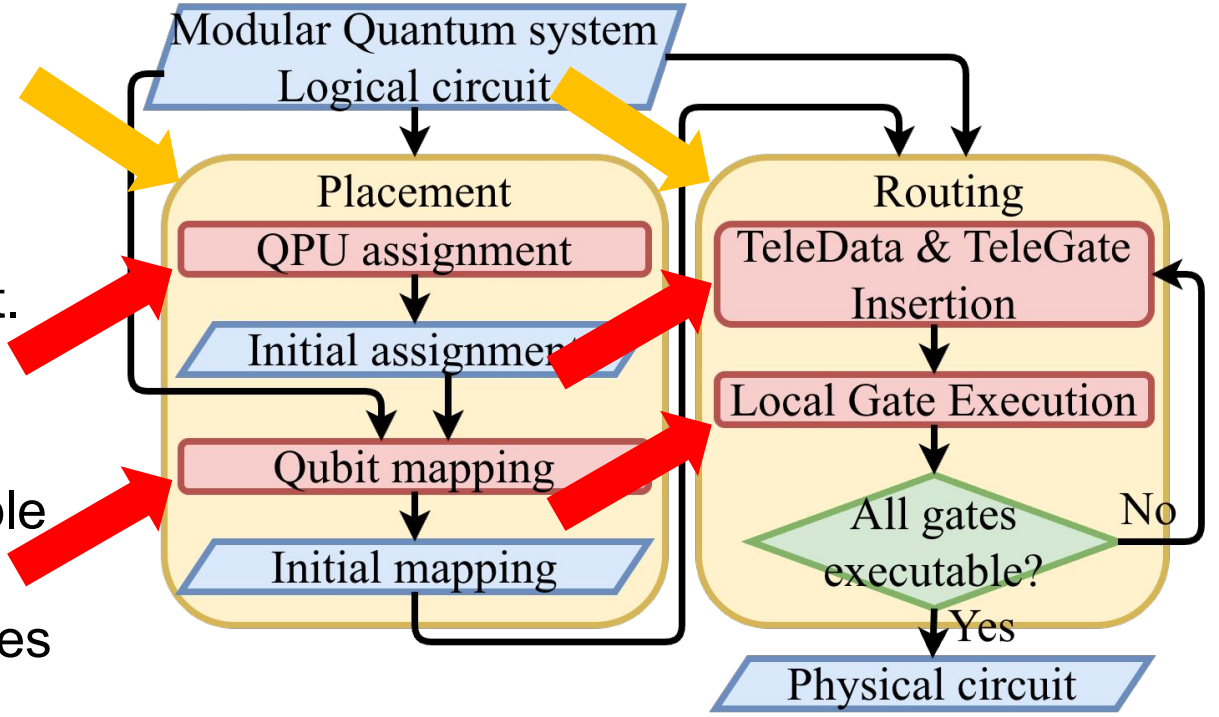
$$finish(g_\ell) = \max_{g'_\ell \in pred(g_\ell)} finish(g'_\ell) + \Delta_{g_\ell}$$

    - The latency of a physical circuit PC:

$$\max_{g_\ell \in PC} finish(g_\ell)$$

Proposed Framework

- Overall flow
  1. Placement Stage
     - Each logical qubit is first assigned to a QPU, and then mapped to a data qubit.
  2. Routing Stage
     - We iteratively select a TeleData or TeleGate for insertion and insert multiple Swap operations.
     - We repeat this process until all the gates in the logical circuit are executed.



Proposed Framework

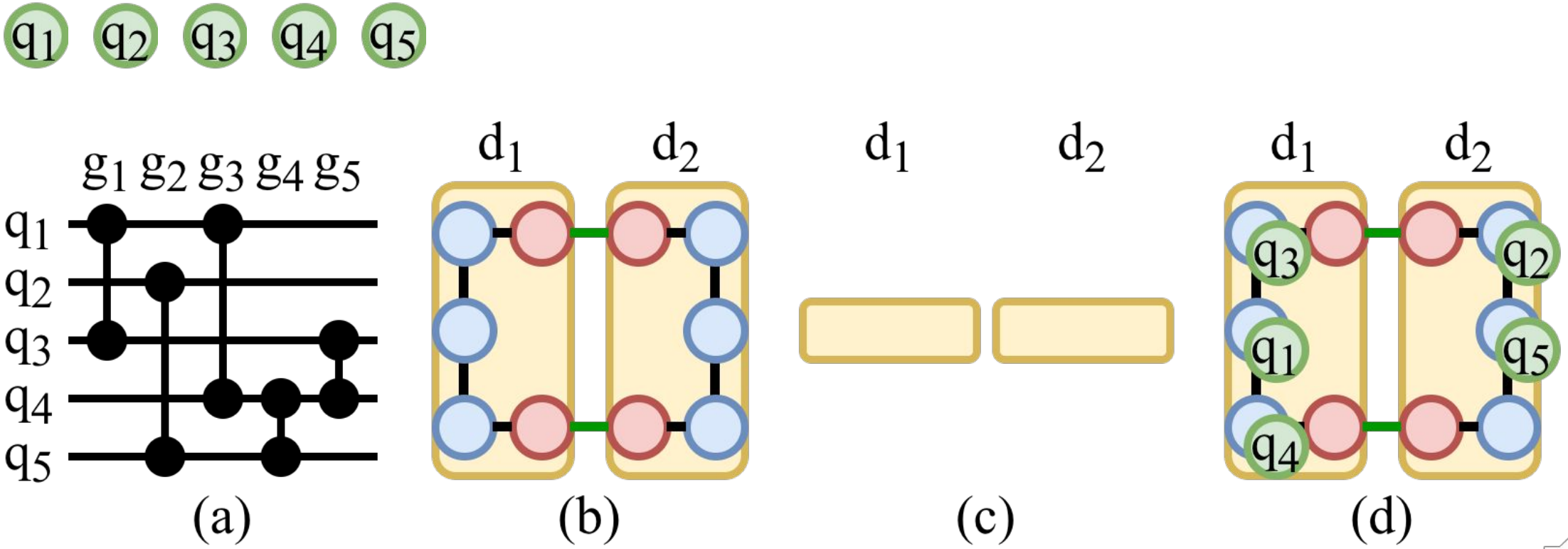
- Placement Stage
  1) Iterate through the two-qubit gates according to the topological ordering.
     - If q_i has already been assigned and q_j has not, then q_j is assigned to the QPU that is closest to the one holding q_i.
     - If neither q_i nor q_j are assigned, then we assign q_i and q_j to a QPU that minimizes the cost below.

$$cost(d_m, q_i, q_j) = \sum_{q_k \in Q_{assigned}} (Dis_S[d_m][d_k] \times (wgt[q_k][q_i] + wgt[q_k][q_j]))$$

  2) Subsequently, we leverage a dynamic lookahead heuristic [41] to map each assigned logical qubit to a data qubit within the QPU it is assigned to.

Proposed Framework

- Placement Stage



Proposed Framework

- Routing Stage
  1) TeleData & TeleGate Insertion
     a) Identify all helpful operations.
        - A TeleData is helpful if it moves one of the operands of a two-qubit gate in the front layer to a QPU closer to the other operand.
        - A TeleGate is helpful if it emulates the execution of a two-qubit gate in the front layer.
     b) Compute the latency overhead and benefit of each help operation.
        - Latency overhead ≤ 0: the delay of the operation is covered by idle times of qubits and thus does not increase circuit latency.
        - Latency overhead > 0: the operation will lengthen the circuit latency.
        - Benefit: an operation that benefits more two-qubit gates at the front leads to a higher benefit.
     c) Select the operation with minimal latency overhead and maximal benefit for insertion.

Proposed Framework

- Routing Stage
  2) Local Gate Execution
     - In this step, we iteratively select a Swap operation for insertion until no local gates remain in the front layer.
       - Local gates: single-qubit gates and two-qubit gates whose operands reside on the same QPU.

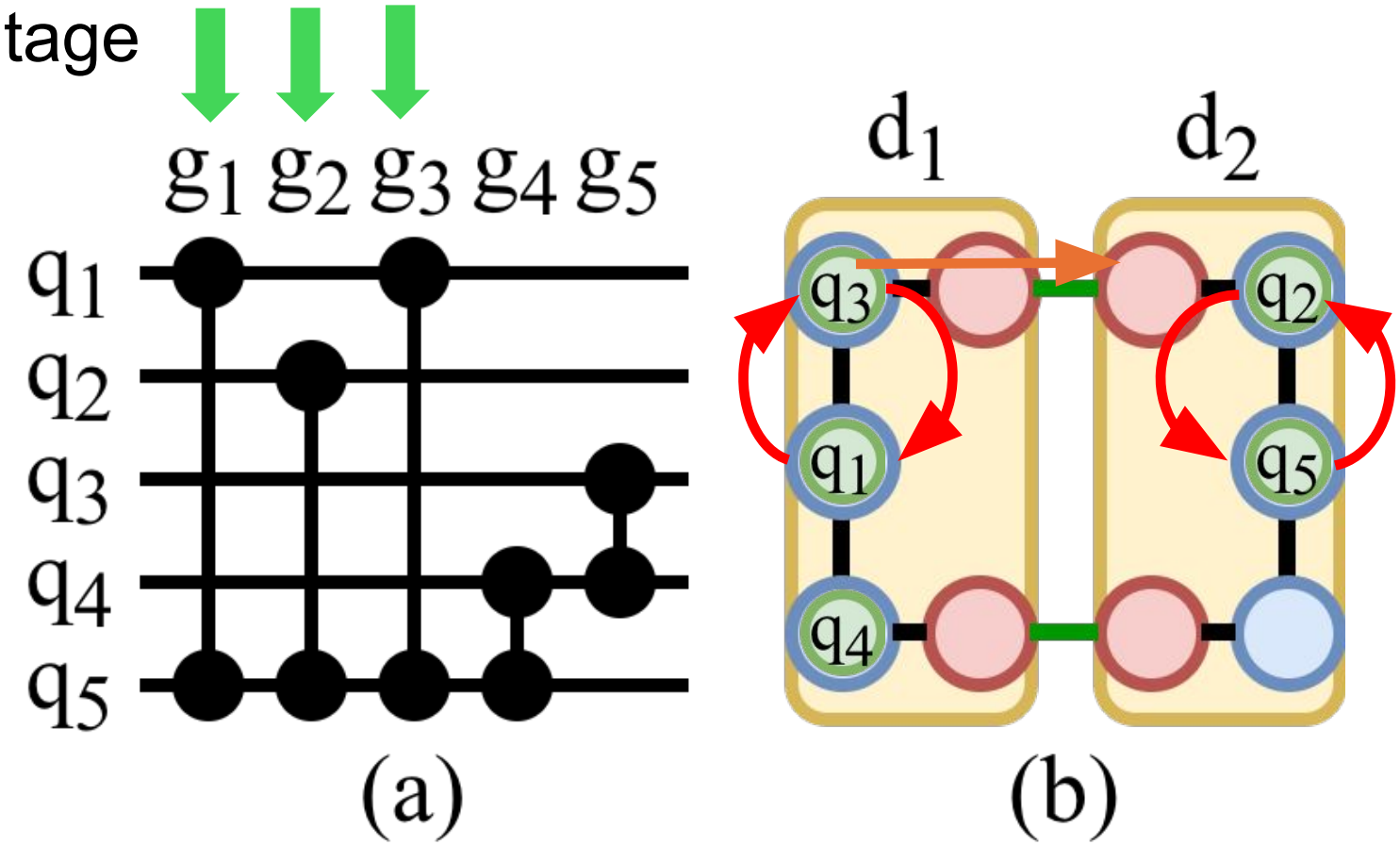
     a) Identify all helpful Swap operations.

     b) Compute the latency overhead and benefit of each helpful Swap.

     c) Select the Swap with minimal latency overhead and maximal benefit for insertion.

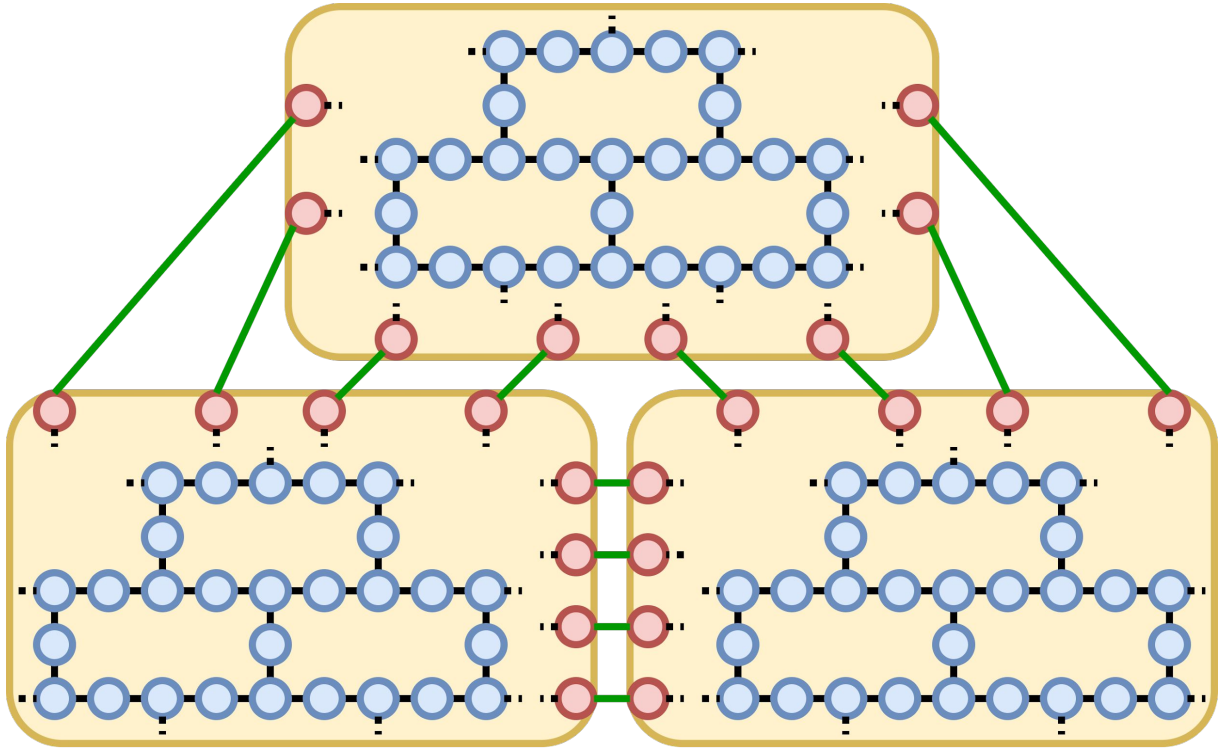
Proposed Framework

- Routing Stage



Experimental Results

- A realistic modular quantum system consisting of three Flamingo-class QPUs.
  - Each Flamingo-class QPU contains 156 physical qubits.



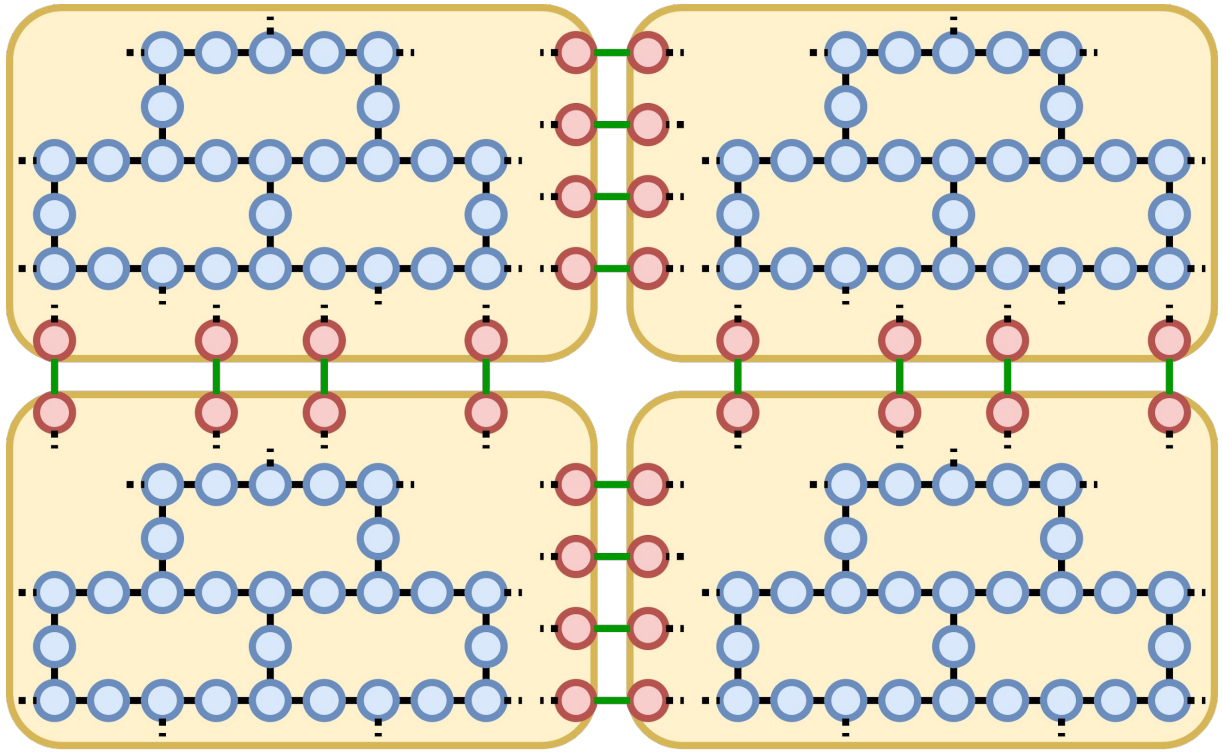
Experimental Results

Table 2: Comparison of our compilation results with those by [19] and [28] for benchmarks in B_{Revlib} and B_{Algo} on a realistic three-QPU modular quantum system. The "$|Q|$", "$|G_1|$", "$|G_2|$", and "Depth" columns show the number of logical qubits, the number of single-qubit gates, the number of two-qubit gates, and the depth for each benchmark, respectively. "Latency" is the compiled physical circuit latency and "#TT" is the number of inserted TeleData and TeleGate.

						[23][28]			[19] placement + [19] routing			Our placement + [19] routing			[19] placement + Our routing			Our placement + Our routing		
Benchmark set	Benchmark	$\|Q\|$	$\|G_1\|$	$\|G_2\|$	Depth	Latency (μs)	#TT	Runtime (s)	Latency (μs)	#TT	Runtime (s)	Latency (μs)	#TT	Runtime (s)	Latency (μs)	#TT	Runtime (s)	Latency (μs)	#TT	Runtime (s)
B_{Revlib}	spla_315	489	61042	5873	583.8	4113.8	1040	44.09	7323.5	388	0.13	5789.0	**84**	0.13	2382.9	430	0.13	**1795.3**	**84**	0.16
	apex2_289	498	61635	5919	185.8	2620.6	634	60.65	5108.6	265	0.12	4723.2	114	0.13	2039.9	301	0.14	**1520.5**	**108**	0.23
	avg16_324	576	80496	7452	867.2	4415.1	252	64.38	7032.6	138	0.16	6832.4	**41**	0.17	2492.4	200	0.17	**2251.1**	49	0.26
	cpu_register_32_357	615	70325	8841	885.4	6003.9	236	56.68	9731.4	191	0.12	9627.3	**35**	0.17	2722.7	167	0.16	**2315.5**	**35**	0.19
	cpu_register_32_429	616	43277	4771	327.1	9655.0	957	536.79	15757.5	730	0.14	9651.2	**383**	0.19	3153.5	643	0.23	**2081.8**	385	0.49
	pdc_307	619	67505	6450	216.9	2826.9	674	62.1	5208.0	254	0.14	5325.6	**135**	0.16	2200.1	298	0.17	**2124.5**	254	0.26
	bubblesort_32_413	623	210798	21581	1842.4	14664.8	2171	209.45	43541.0	1137	0.59	40419.0	450	0.44	7924.8	855	0.44	**7855.5**	**354**	0.81
	cpu_control_unit_354	646	101752	10411	1153.5	5476.2	622	159.9	12723.3	264	0.22	11418.7	127	0.21	3383.2	235	0.38	**3230.0**	**118**	0.64
	cpu_control_unit_426	647	69340	7102	601.4	4626.5	908	112.73	15927.2	678	0.19	22463.3	1266	0.32	**2585.5**	**374**	0.38	4269.8	950	0.5
	cpu_alu_32bit_329	756	2105086	216314	27876.6		timeout		233236.3	1603	17.73	229219.0	1601	7.54	**64688.5**	**1584**	62.62	65055.3	1614	72.45
	bubblesort_16_436	813	122142	12563	716.1	8386.5	1586	141.59	23093.0	812	0.32	17647.0	**240**	0.31	6174.0	1099	0.4	**5992.6**	335	0.7
	cps_292	923	87799	8381	302.2	13432.0	926	162.97	8743.8	525	0.26	8599.7	**327**	0.25	**3441.0**	599	0.32	3738.0	380	0.39
	apex5_290	1025	107063	10314	287.4	51179.2	1009	596.06	10905.5	583	0.32	9270.3	368	0.31	4608.0	624	0.42	**4052.4**	**367**	0.57
	frg2_297	1219	128886	12372	459.6	83871.8	982	567.8	11231.8	**369**	0.35	10066.2	**369**	0.37	4281.6	455	0.44	**3564.0**	496	0.54
	Norm1					1.0	1.0	1.0	1.762	0.54	0.002	1.691	0.325	0.002	0.487	0.547	0.002	0.465	0.328	0.003
	Norm2					1.43	1.986	942.327	1.0	1.0	1.0	0.93	0.605	1.058	0.31	1.043	1.463	0.286	0.64	2.121
B_{Algo}	VQE_q500	500	6994	499	54.1	113.9	2	10.02	378.0	2	0.05	396.3	**1**	0.07	97.8	2	0.05	**95.2**	**1**	0.07
	VQE_q900	900	12594	899	97.4	239.2	**2**	12.65	739.2	**2**	0.13	721.1	**2**	0.13	214.6	**2**	0.14	**182.4**	**2**	0.14
	VQE_q1300	1300	18194	1299	140.7	332.0	**2**	14.77	1042.3	**2**	0.22	1071.3	**2**	0.21	**322.0**	**2**	0.23	326.2	**2**	0.23
	QFT_q500	500	1872750	249500	239.8		timeout		139907.5	444	4.94	139717.5	**126**	5.64	60272.2	3040	6.98			
	QFT_q900	900	6070950	809100	432.1		timeout		390877.2	7566	18.2	347292.0	1285	57.97	**202564.3**	10755	33.13	299671.1	**1263**	93.94
	DJ_q500	500	5991	499	66.0	13282.7	284	64.14	3518.6	274	0.08	296.3	**1**	0.08	707.3	256	0.09			
	DJ_q900	900	10791	899	118.9	5172.9	470	33.45	5826.1	463	0.18	529.2	**2**	0.15	1101.5	442	0.2	**321.2**	**2**	0.16
	DJ_q1300	1300	15591	1299	171.8	50204.8	726	184.36	8580.2	702	0.29	770.5	**2**	0.24	2187.3	670	0.33	**454.4**	**2**	0.26
	GSR_q500	500	1500	2000	42.1	953.9	75	32.41	5370.7	380	0.13	2249.2	83	0.1	618.1	48	0.11	**554.7**	**32**	0.2
	GSR_q900	900	2700	3600	75.7	35446.8	99	197.99	8035.8	519	0.25	4241.5	156	0.19	1133.6	**48**	0.27	**1106.5**	**48**	0.36
	GSR_q1300	1300	3900	5200	109.4	21013.2	129	207.18	6663.4	211	0.31	5303.8	132	0.27	1787.6	61	0.48	**1583.4**	**48**	0.51
	Norm1					1.0	1.0	1.0	1.92	1.985	0.005	1.399	0.691	0.005	0.422	0.818	0.005	0.373	0.422	0.006
	Norm2					2.133	0.787	417.065	1.0	1.0	1.0	0.631	0.374	1.161	0.271	1.329	1.216	0.233	0.391	1.834

Experimental Results

- A realistic modular quantum system consisting of four Flamingo-class QPUs.
  - Each Flamingo-class QPU contains 156 physical qubits.



Experimental Results

Table 3: Comparison of our compilation results with those by [19] and [28] for benchmarks in B_{Revlib} and B_{Algo} on a realistic four-QPU modular quantum system.

						[23][28]			[19] placement + [19] routing			Our placement + [19] routing			[19] placement + Our routing			Our placement + Our routing		
Benchmark set	Benchmark	$\|Q\|$	$\|G_1\|$	$\|G_2\|$	Depth (μs)	Latency (μs)	#TT	Runtime (s)	Latency (μs)	#TT	Runtime (s)	Latency (μs)	#TT	Runtime (s)	Latency (μs)	#TT	Runtime (s)	Latency (μs)	#TT	Runtime (s)
B_{Revlib}	spla_315	489	61042	5873	583.8	5573.9	1745	54.79	8294.3	519	0.12	5789.0	**84**	0.13	2398.6	616	0.15	**1795.3**	**84**	0.16
	apex2_289	498	61635	5919	185.8	3718.4	1204	75.53	5697.5	366	0.12	4723.2	114	0.13	2033.8	399	0.16	**1520.5**	**108**	0.24
	avg16_324	576	80496	7452	867.2	4840.8	450	75.81	7823.9	215	0.17	6832.4	**41**	0.17	2792.3	311	0.17	**2251.1**	49	0.27
	cpu_register_32_357	615	70325	8841	885.4	5246.0	346	68.31	10034.8	228	0.11	9627.3	**35**	0.17	2855.5	228	0.14	**2315.5**	**35**	0.19
	cpu_register_32_429	616	43277	4771	327.1	4689.5	1066	448.7	14718.3	651	0.12	9651.2	**383**	0.18	2772.9	643	0.21	**2081.8**	385	0.49
	pdc_307	619	67505	6450	216.9	3695.4	1152	73.56	5593.6	436	0.14	5325.6	**135**	0.16	2245.3	461	0.18	**2124.5**	254	0.27
	bubblesort_32_413	623	210798	21581	1842.4	13001.0	2246	207.62	37305.9	815	0.37	40419.0	450	0.42	**7850.5**	844	0.41	7855.5	**354**	0.81
	cpu_control_unit_354	646	101752	10411	1153.5	5795.0	1463	147.34	14105.4	411	0.21	11644.3	127	0.2	3444.4	447	0.3	**3129.2**	**110**	0.65
	cpu_control_unit_426	647	69340	7102	601.4	5414.0	1035	147.77	14638.9	692	0.18	22463.3	1266	0.3	**3170.7**	**614**	0.32	4269.8	950	0.5
	cpu_alu_32bit_329	756	2105086	216314	27876.6		timeout		236865.0	1859	11.74	229219.0	**1601**	3.26	65822.5	1810	80.12	**65055.3**	1614	73.61
	bubblesort_16_436	813	122142	12563	716.1	10386.3	2923	150.75	26675.8	1167	0.29	17647.0	**240**	0.3	7349.7	1456	0.41	**5992.6**	335	0.71
	cps_292	923	87799	8381	302.2	5717.8	1701	117.05	10950.4	815	0.25	9082.3	**377**	0.25	**3777.2**	890	0.33	3779.5	456	0.4
	apex5_290	1025	107063	10314	287.4	5668.1	1798	221.55	10479.7	581	0.28	10664.4	**503**	0.3	5009.0	821	0.43	**4453.6**	555	0.61
	frg2_297	1219	128886	12372	459.6	9670.1	1498	158.77	15053.1	785	0.37	11253.6	**526**	0.35	4447.9	714	0.45	**4075.6**	883	0.58
	bubblesort_32_437	1597	237708	24473	1352.4	23801.6	3677	504.63	49372.7	**1703**	0.83	65515.1	3172	1.18	**14302.7**	1873	1.01	18104.6	3573	2.05
	seq_314	1617	201388	19283	614.4	24428.6	3610	298.38	20293.8	1300	0.84	22992.2	**1212**	0.68	9270.9	1685	0.67	**9035.8**	1250	0.77
	Norm1					1.0	1.0	1.0	2.0	0.439	0.002	1.89	0.297	0.002	0.585	0.483	0.002	0.544	0.312	0.003
	Norm2					0.557	2.459	777.468	1.0	1.0	1.0	0.943	0.642	1.106	0.31	1.113	1.659	0.287	0.684	2.381
B_{Algo}	VQE_q500	500	6994	499	54.1	135.6	4	18.48	396.5	4	0.04	396.3	**1**	0.07	113.7	4	0.05	**95.2**	**1**	0.07
	VQE_q900	900	12594	899	97.4	308.4	5	21.62	730.8	4	0.11	721.4	**2**	0.14	208.3	4	0.12	**186.9**	**2**	0.14
	VQE_q1300	1300	18194	1299	140.7	382.4	4	23.97	1074.5	4	0.19	1057.1	**3**	0.2	312.9	4	0.21	**270.2**	**3**	0.22
	VQE_q1700	1700	23794	1699	183.9	445.9	**3**	26.81	1366.5	**3**	0.29	1380.5	**3**	0.28	**402.2**	**3**	0.31	402.6	**3**	0.31
	QFT_q500	500	1872750	249500	239.8		timeout		141599.9	607	5.08	139717.5	**126**	5.48	58816.6	10637	10.16	**48652.4**	541	18.75
	QFT_q900	900	6070950	809100	432.1		timeout		368613.3	1959	16.3	338169.0	**998**	34.76	**175993.4**	23240	39.88	299234.2	1263	94.34
	DJ_q500	500	5991	499	66.0	5101.8	406	31.35	3915.3	336	0.07	296.3	**1**	0.08	696.7	376	0.1	**180.3**	**1**	0.08
	DJ_q900	900	10791	899	118.9	7655.2	701	40.2	7040.3	596	0.16	530.9	**3**	0.15	1193.0	664	0.21	**322.9**	**3**	0.17
	DJ_q1300	1300	15591	1299	171.8	6092.7	1021	44.6	10269.4	871	0.28	761.8	**4**	0.23	1947.1	978	0.33	**462.0**	**4**	0.26
	DJ_q1700	1700	20391	1699	224.7	17962.9	1316	75.16	13517.5	1123	0.4	989.9	**4**	0.32	3082.1	1270	0.48	**597.2**	**4**	0.36
	GSR_q500	500	1500	2000	42.1	1116.8	143	42.97	5967.7	548	0.13	2249.2	83	0.1	700.6	64	0.11	**554.7**	**32**	0.21
	GSR_q900	900	2700	3600	75.7	1839.6	174	65.32	9456.6	968	0.26	4238.6	155	0.19	**1134.5**	96	0.25	1136.1	**65**	0.37
	GSR_q1300	1300	3900	5200	109.4	3064.7	214	106.07	13504.7	957	0.4	6507.0	266	0.28	1768.9	96	0.45	**1705.8**	**64**	0.5
	GSR_q1700	1700	5100	6800	143.1	3372.0	155	129.55	8962.7	321	0.4	8657.4	277	0.38	2079.8	**64**	0.61	**2041.3**	**64**	0.65
	Norm1					1.0	1.0	1.0	2.737	1.927	0.005	1.703	0.576	0.005	0.538	0.788	0.006	0.448	0.31	0.006
	Norm2					0.562	0.844	255.104	1.0	1.0	1.0	0.604	0.335	1.063	0.244	2.742	1.321	0.21	0.317	1.757

Conclusion

- In the placement stage, we propose a dependency- and interaction-aware assignment method to assign logical qubits that interact early in the circuit and frequently with each other to nearby QPUs in the system.
- In the routing stage, the selection of both inter- and intra-QPU operations is based on their circuit latency overhead as well as their benefit to subsequent gates.
- Improvement over [19] and [28]
  - Our approach outperformed the [19] with over 73.8% and 46.9% reduction in compiled circuit latency and number of inserted inter-QPU operations, respectively.
    - A similar advantage is also observed compared to [28].
  - The advantage of our approach over [19] and [28] is preserved on a realistic modular quantum system with 4 QPUs.