Mapping Quantum Circuits to 2-D Quantum Architectures

Kamalika Datta[1] Abhoy Kole[2] Indranil Sengupta[3] Rolf Drechsler[4]

Abstract: We have been witnessing a rapid growth in quantum computing research over the years, with the emergence of demonstrable quantum computers of moderate size. The major issues that are faced to run a quantum algorithm reliably on these systems are: (i) lower qubit coherence period, (ii) noisy primitive gate operations, (iii) limited number of available physical qubits, and (iv) support of restricted set of 2-qubit operations. Overcoming these issues mandates physical resources that exceeds the capabilities of these noisy intermediate scale quantum (NISQ) systems. On the other hand, computation using bare qubits get further disturbed due to the inclusion of additional gates to mitigate the nearest neighbor constraints. In the present work, the 2-dimensional square, heavy-hex and fully hexagonal qubit coupling lattices are considered for mapping quantum circuits. The benefits are assessed in terms of minimal additional gates needed to satisfy the nearest neighbor (NN) constraint and the compilation complexity of mapping circuits on these architectures. From the experiments by mapping benchmark circuits on 16-qubit square and 65-qubit heavy-hex architectures from IBM as well as on a 64-qubit fully hexagonal architecture, it is observed that none of the square or heavy-hex lattice architecture provides uniform compilation advantage compared to the fully hexagonal architecture. It is expected that beyond the NISQ era, strongly connected lattices like hexagonal will become practically feasible.

1 Introduction

With the emergence of quantum computers in recent years, researchers have started to work on application mapping with respect to more realistic qubit architectures [IB]. The size of quantum computers in terms of the number of qubits has been steadily increasing over the years. For example, IBM has recently announced their 127-qubit Eagle processor [CDG21]. In quantum computing, qubits are used as the fundamental unit of information as opposed to bits in classical computing. The qubits can be regarded as hardware, on which the quantum gates operate in the axis of time.

In a quantum computer, the qubit architecture defines how the qubits are laid out physically and the ways they can interact. The concept of *coupling map* has been used in the earlier

[1] German Research Centre for Artificial Intelligence (DFKI), Cyber Physical System, Bremen, Germany
kamalika.datta@dfki.de
[2] German Research Centre for Artificial Intelligence (DFKI), Cyber Physical System, Bremen, Germany
abhoy.kole@dfki.de
[3] JIS University, Department of Computer Science and Engineering, Kolkata, India,
Indian Institute of Technology Kharagpur, Department of Computer Science and Engineering, India
indranil.sengupta@jisuniversity.ac.in
[4] University of Bremen, Group of Computer Architecture, Bremen, Germany,
German Research Centre for Artificial Intelligence (DFKI), Cyber Physical System, Bremen, Germany
drechsler@uni-bremen.de

generations of IBM Q series of quantum computers [IB], which specifies the way the physical qubits can interact during 2-qubit gate operations. Some other qubit architectures have also been proposed in recent years, like the *hexagonal architecture* [Li17, Ta18], where a qubit can be connected with up to six other neighboring qubits in a two dimensional (2-D) grid. In fact, all practical qubit architectures consider the layout of the qubits on a 2-D plane with realistic interconnections. The number of neighborhood connections to a qubit is determined by factors like physical layout, errors during gate operations, etc. [Na21]. One of the basic constraint in mapping a quantum circuit to these physical architectures is that the interacting qubits must be neighbors. For a large quantum circuit with fewer physical connectivity amongst qubits, it is indeed a major challenge to satisfy this constraint. In most of the cases, *Swap* gates are inserted to exchange the states of two qubits, where each Swap gate is implemented using three back-to-back CNOT gates. The consequent increase in the number of gates in the circuit makes it more prone to errors, and hence reducing the number of Swap gate operations is of vital importance.

In the present NISQ era, with the availability of limited number of noisy qubits, the degree of association of the qubits is limited to 2 or 3 (e.g., 2-D and heavy-hex architectures). However, with further developments in fabrication technology, emergence of strongly connected qubit architectures (e.g., hexagonal) is a strong possibility. The main objective of the present work is to compare some of the recent qubit architectures with the hexagonal architecture with respect to the cost of mapping quantum circuits, and evaluate how much reduction in gate overheads can be achieved.

In this paper we analyze the performances of 16-qubit IBM QX5 (Rueschlikon) and 65-qubit IBM Q65 (Hummingbird) architectures, as well as the hexagonal architecture. We propose methods to map a given quantum circuit to these architectures. The rest of the paper is organized as follows. Section 2 provides a survey of the selected IBM and the hexagonal architectures. Section 3 proposes the qubit mapping algorithms for the architectures selected for comparison. In section 4 we provide the experimental results followed by concluding remarks in section 5.

2 Recent Quantum Architectures

In this section we provide a brief review of the IBM QX5 and Q65 architectures and also the 2-D hexagonal architecture. The IBM Q series of quantum computers are based on qubits that are built using superconducting transmon, using materials like niobium and aluminum, patterned on a silicon substrate. IBM first came up with 5-qubit chips in 2016 and made it available in the cloud. Since then, IBM came up with several quantum chips of increasing capacity, like 16-qubit Canary and Rueschlikon, 27-qubit Falcon, 65-qubit Hummingbird, and 127-qubit Eagle [IB17]. In this paper we particularly focus on IBM's QX5 (Rueschlikon) and Q65 (Hummingbird) architectures. Earlier qubit architectures by IBM like the QX5 relied on *2-D lattice* structure. In contrast, the Q65 architecture uses the *heavy-hex lattice* structure, which is the topology used in recent IBM quantum computers. The choice of such topology is based on experimental evaluation to ensure reduced error-rates, which also provides room to explore error correcting codes. In these topologies, a single qubit is connected with a maximum of two or three neighboring qubits.

The IBM Q series architectures support a limited number of 2-qubit gate operations. Fig. 1 shows the qubit arrangements in the QX5 [Ga] and Q65 [IB20] architectures. An edge denotes coupling between physical qubits for conducting 2-qubit gate operations (e.g. CNOT) on such architectures. The QX5 architecture (see Fig. 1a) is more restrictive, where the edge directions indicate control and target qubits for 2-qubit CNOT operations, e.g. for a directed edge like $q_i \rightarrow q_j$ between physical qubits q_i and q_j, the QX5 architecture supports CNOT(q_i, q_j) operation, whereas it needs four additional Hadamard operations to conduct CNOT(q_j, q_i) operation [ZPW19].

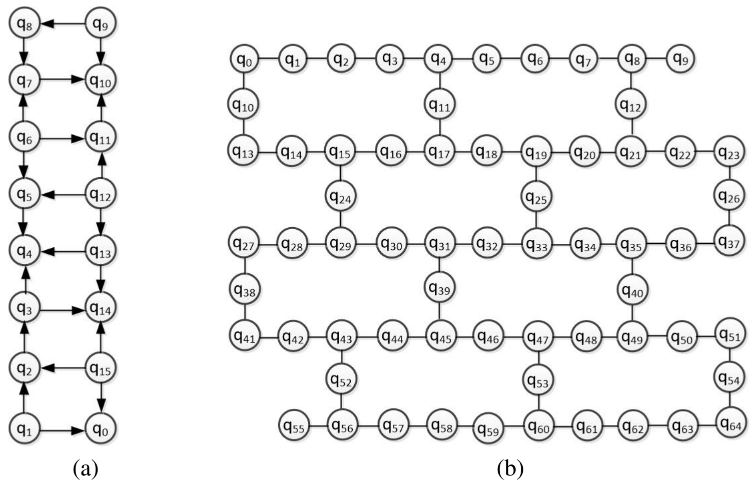
q_8 q_9
q_7 q_{10}
q_6 q_{11}
q_5 q_{12}
q_4 q_{13}
q_3 q_{14}
q_2 q_{15}
q_1 q_0
(a)
q_0 q_1 q_2 q_3 q_4 q_5 q_6 q_7 q_8 q_9
q_{10} q_{11} q_{12}
q_{13} q_{14} q_{15} q_{16} q_{17} q_{18} q_{19} q_{20} q_{21} q_{22} q_{23}
q_{24} q_{25} q_{26}
q_{27} q_{28} q_{29} q_{30} q_{31} q_{32} q_{33} q_{34} q_{35} q_{36} q_{37}
q_{38} q_{39} q_{40}
q_{41} q_{42} q_{43} q_{44} q_{45} q_{46} q_{47} q_{48} q_{49} q_{50} q_{51}
q_{52} q_{53} q_{54}
q_{55} q_{56} q_{57} q_{58} q_{59} q_{60} q_{61} q_{62} q_{63} q_{64}
(b)

Fig. 1: (a) IBM QX5 Rueschlikon architecture, (b) IBM Q65 Hummingbird architecture.

Recently quantum mechanical systems based on hexagonal architecture have been proposed [Li17, Ta18]. In this architecture, each qubit can have a maximum of 6 neighbors. Having more connectivity amongst neighbors provides flexibility in the mapping of quantum circuits. Fig. 2 shows a 6×5 hexagonal architecture where the height is 5 and width is 6. Recently few works have been reported on mapping quantum circuits to hexagonal structures [CL21, Da22]. As mentioned earlier, Swap gates are typically inserted to make a quantum circuit NN-compliant. In [CL21], Chang et al. have used Swap gates to make quantum circuits NN-compliant. Due to increased connectivity among qubits, the hexagonal architecture is found to reduce the number of Swap gates required as compared to existing architectures. In [Da22], Datta et al. have used the CNOT templates [RD12] as the cost metric for NN-compliance in hexagonal architecture, instead of Swap gates. The method uses a Genetic Algorithm based global ordering of qubits for generating the initial qubit placement, and is found to outperform other NN-mapping techniques in two dimensions.

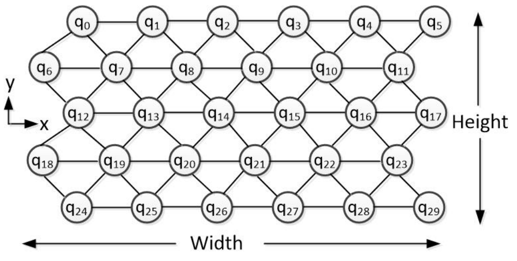

Fig. 2: Hexagonal qubit architecture.

It is evident that IBM's QX5 and Q65 architectures have less connectivity as compared to hexagonal architecture, and therefore have distinct advantages in terms of overall error rate. Although hexagonal architecture seems promising with higher connectivity, commercial products based on such architecture are expected only in the post-NISQ era.

3 Mapping Quantum Circuits to Various Architectures

In this section we discuss how the logical qubits in a quantum circuit can be mapped to various qubit architectures. For the sake of comparison, we have used Swap gates as the evaluation metric for all the methods. The overall flow of quantum circuit mapping is shown in Fig. 3. We first discuss the steps for qubit mapping in the Q65 architecture; the mapping for the QX5 architecture is very similar.

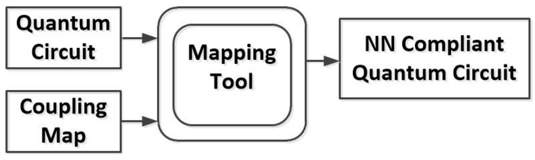

Fig. 3: The overall mapping flow.

3.1 Mapping Quantum Circuits to IBM's 65-bit Hummingbird Architecture

The problem of mapping quantum circuits to IBMQ architectures has been explored by several researchers [ZPW19, dADdS19, Ko20, Ni20, NBD21]. In most of these methods either 5-qubit (e.g., QX2 (Yorktown) and QX4 (Tenerife)) or 16-qubit (e.g., QX3 and QX5) Rueschlikon architectures have been considered. In this paper we have used QX5 as well as Q65 architectures for experimentation. A generalized architecture-independent qubit mapping tool has been developed for the purpose. The inputs to the tool are the architectural specification of the qubits (e.g., Q65), and the given quantum circuit. The algorithm generates the qubit placement, and calculates the number of Swap gates required to make the circuit NN-compliant. When we map a circuit with n logical qubits $(Q_0, \ldots, Q_{n-1})$ to the Q65 architecture, the n physical qubits can be selected in $\binom{65}{n}$ ways. It may be noted that the qubit association is bidirectional in Q65, i.e. an edge (q_i, q_j) in the coupling map indicates that either q_i or q_j can be used as target in a 2-qubit gate operation. Also, the degree of association of any qubit is 2 or 3. Referring to Fig. 1b, the qubit q_4 has the connections $q_4 \rightarrow q_3$, $q_4 \rightarrow q_5$, and $q_4 \rightarrow q_{11}$, and hence the degree of association is $deg(q_4) = 3$. Considering a quantum circuit, the degree of association is represented by connections between the logical qubits in the qubit interaction graph, $QIB = (V, E)$, where the vertices $Q_k \in V$ indicate logical qubits, and the weight (w_{kr}) of an edge $(Q_k, Q_r) \in E$ denotes the number of 2-qubit gate operations between Q_k and Q_r.

The following steps are used to map a quantum circuit to the IBM Q65 architecture. The process of mapping to the QX5 architecture is similar, with special care taken for the directional constraints in the coupling map.

(i) For every pair of physical qubits (q_i, q_j), d_{ij} denotes their distance as per the coupling map. If the qubits are adjacent, we set $d_{ij} = 0$. If they are non-adjacent, the distance is initially set to infinity (∞). To find the shortest distance between all pairs of non-adjacent qubits, we use Floyd-Warshall algorithm, which has an overall runtime of $O(n^3)$ for n physical qubits. This process needs to be performed only once for an architecture and the computed distance measure (d_{ij}) can be kept in a persistent storage to speed up the mapping process.

(ii) A greedy approach is used to select the n physical qubits to map a n-qubit quantum circuit. The objective is two-fold: (i) to minimize the number of CNOT operations on uncoupled physical qubits in the mapped quantum circuit, and (ii) to reduce the search time for selecting a good set of n among m physical qubits for mapping. In the first step of qubit mapping, the logical qubit (Q_k) having highest degree of association is mapped to a physical qubit (q_i) that also has highest degree of association. In case of a tie, one of the qubits involved in the tie is chosen randomly. This process is repeated for all the remaining logical qubits, in descending orders of degree of association. At every step, the next physical qubit is selected based on the shortest distance from the already assigned qubits. For mapping n physical qubits, the average running time of this step is $O(n^2)$.

(iii) During the mapping of a quantum circuit to a physical qubit architecture, the selection of physical qubits and also the initial permutation of the logical qubits $(Q_0, \ldots, Q_{n-1})$ play an important roles in reducing the number of Swap gates and also the circuit depth. In general, finding the optimal initial permutation for n logical qubits is computationally hard, with a complexity of $O(n!)$. In this work we have used an evolutionary algorithm to find the initial permutation of logical qubits, some of the parameters for which are as follows.

- The number of Swap gates required for NN-compliance is used as the *fitness function* consistent with the edge weights of the corresponding QIG.
- The initial population size is taken as 30, crossover rate 70%, and mutation rate 17%. The algorithm is run over 500 generations, at the end of which the best solution gives the initial permutation.
- While generating the chromosomes for the next generation, the best 4 permutations from the current generation are copied. For crossover operation, a pair of permutations (P_i^c, P_j^c) from the current generation are picked out using roulette wheel selection method [Go89]. Using a randomly selected crossover point, first parts of this selected pair are copied to a new pair (P_1^n, P_2^n) for the next generation. To prevent redundant qubit mapping, the remaining missing qubits of the new pair (P_i^n, P_j^n) are filled up by reordering their appearance in the pair (P_i^c, P_j^c). During mutation, one of the operations like swapping positions of two randomly selected qubits, moving a randomly selected qubit to a vacant adjacent position or moving a qubit to a randomly selected vacant position is conducted randomly on permutations from the current generation before being copied in next generation.

(iv) After the initial placement of the logical qubits, some of the 2-qubit gate operations may not satisfy the coupling constraints, for which Swap gates are inserted. A Swap gate exchanges the states of a pair of qubits, which in turn changes the qubit permutation. Starting with the initial mapping, we traverse the quantum circuit from left to right, and use a k-gate lookahead approach to insert Swap gates before each 2-qubit gate operating on non-coupled qubits. The objective is to make the gate operation NN-compliant. Among the various possibilities that may exist, we select the one that minimizes the cost of mapping the k subsequent gates in the circuit.

For a given quantum circuit, the mapping tool tries to carry out a good logical to physical qubit mapping, and report the overhead for NN-compliance in terms of the number of Swap gates required.

3.2 Mapping Quantum Circuits to Hexagonal Architecture

We now discuss the problem of mapping a quantum circuit to the hexagonal qubit architecture. Fig. 2 depicts the coupling map of 30 qubits arranged in a 6×5 hexagonal array, with a maximum qubit coupling of 6. As a matter of convention, we number the qubits in row-major order, from left to right and then from top to bottom. If W and H denote the width and height respectively in the $W \times H$ arrangement of qubits, the total number of qubits will be $W.H$. The Cartesian co-ordinates of the i^{th} qubit (i.e., q_i) can be calculated as

$$\begin{aligned} x_i &= (1 - (y_i \,\%\, 2)) + 2\,(i \,\%\, W) \\ y_i &= i \div W \end{aligned}$$

where % and ÷ respectively denote modulus and integer division. This coordinate system guarantees that the Manhattan Distance (MD) between any two neighboring qubits is 2. The MD measure can be used to obtain the distance d_{ij} between physical qubits q_i and q_j as:

$$d_{ij} \quad = \quad max\left(|y_i - y_j|, \frac{MD(q_i, q_j)}{2}\right) - 1 \tag{1}$$

where the coordinates of the qubits q_i and q_j are (x_i, y_i) and (x_j, y_j) respectively.

The mapping approach is similar to the previously outlined approach for IBMQ architecture with minor changes. The following steps are followed to map the logical qubits of a quantum circuit to the hexagonal architecture:

(i) We create a weighted qubit interaction graph (QIG) from the given quantum circuit, in which every node Q_k denotes a logical qubit and an edge (Q_k, Q_r) denotes the presence of 2-qubit gate(s) between Q_k and Q_r. The weight of the edge w_{kr} indicates the number of 2-qubit gates between the pair of qubits.

(ii) We use an evolutionary algorithm to find the best initial mapping of the qubits. The earlier defined cost function (viz., the total number of Swap operation for obtaining NN-compliance) is used for optimization keeping population size, crossover rate, mutation rate and number of generations unchanged.

(iii) With the obtained initial mapping, the quantum circuit from left to right is traversed using the same k-gate lookahead approach, and Swap gates are inserted before each 2-qubit gate operating on non-coupled qubits for NN-compliance. Different possibilities are explored and the lowest cost alternative is selected for mapping the k subsequent gates in the circuit.

4 Experimental Results

In this section we present the results of mapping various quantum benchmark circuits to the selected IBM and hexagonal qubit architectures using the approaches outlined in the previous section. Initially, the RevLib benchmarks [Wi08] available in reversible form are transpiled using Qiskit 0.19.0 [An21] with Python 3.16.12 setting the Qiskit parameters *basis_gates* and *optimiyation_levels* as (CX, U3) and 0, respectively before being used in our experiments. The qubit mapping algorithms are implemented using C++ and run on an Intel(R) Xeon(R) Gold 6132 processor with 96GB memory running CentOS 7.0 operating system.

We have shown two sets of experimental results. In Table 1, we show the mapping results for IBM QX5 and IBM Q65. The first three columns show the benchmark names, number of qubits (n), and number of CNOT gates in the original netlist. The next three columns show the number of extra CNOT gates required to make the circuit NN-compliant, the number of H-gates (specifically required to conduct CNOT operation consistent with the coupling direction), circuit depth and run-time for IBM QX5. The next three columns show the number of extra CNOT gates required, circuit depth and run-time for IBM Q65. The last three columns show the better architecture, % improvements in number of gates, and difference in run-times. We have chosen 51 benchmarks with $3 \leq n \leq 14$, out of which 21 benchmarks show better results for QX5, and 30 show better results for Q65. Out of the total 21 benchmarks for which QX5 gives better results, 11 of them have $n \geq 10$. But out of 30 benchmarks for which Q65 provides better results, most of them have $3 \leq n \leq 7$. It may be noted that in the QX5 architecture, the degree of association is 2 for only 4 qubits, and 3 for the remaining 12 qubits. In contrast for Q65, considering the first 16 qubits, 3 of them have degree of association as 3, while the rest have a value of 2. However, the Q65 architecture has better error resiliency and can map larger quantum circuits as compared to QX5.

Table 2 shows the mapping results for 8×8 hexagonal and IBM Q65 architectures. The first three columns show the benchmark names, number of qubits (n) and number of CNOT gates. The next six columns show the number of extra CNOT gates required for NN-compliance, circuit depth and run-time for the hexagonal and Q65 architectures respectively. The last two columns show the % improvement in CNOT gates and difference in run-time of the hexagonal architecture over Q65. It is evident from the results that the hexagonal architecture provides lower mapping overheads as compared to Q65, which is a result of the extended qubit coupling in the hexagonal architecture. However, it may be noted that the hexagonal architecture is not practically feasible in the present NISQ era; however, it can be a potential candidate in the future during the post-NISQ era.

5 Conclusion

In this paper we have analyzed the overheads of quantum circuit mapping on two IBM's quantum computer architectures, viz. 16-qubit Rueschlikon (QX5) and 65-qubit Hummingbird (QX65), and a 2-D hexagonal architecture. Both QX5 and Q65 architectures have a maximum qubit neighborhood of 3, with directional constraints existing between pair of coupled qubits in QX5 and no directional constraints in Q65. In fact, in the QX5 the qubits are laid out uniformly in a 2-D grid, while a heavy-hex lattice structure is used to connect the qubits in Q65. In contrast, the hexagonal architecture has a maximum qubit neighborhood of 6. Experimental results reveal that the hexagonal architecture incurs less gate overhead for mapping various benchmarks compared to QX5 and Q65, which is mainly because of the extended connectivity among qubits. Between QX5 and Q65, since the qubit interaction patterns are different, we see mixed results for the mapping of benchmark circuits. It may be noted that although hexagonal architecture provides better mapping, cloud based devices based on such extended neighborhood architectures are yet not available. In comparison, the IBM series of quantum computers in spite of their limitations have been physically fabricated and widely used by users from all over the globe. Hexagonal and other strongly connected architectures may emerge in the future during the post-NISQ era.

Tab. 1: Comparison of NN-mapping on IBM QX5 and Q65 architectures.

Benchmark Circuits			QX5				Q65			Improvement		
Name	n	#CNOT	#CNOT	#H	#Depth	t(s)	#CNOT	#Depth	t(s)	Arch.	Δ_g(%)	$\Delta_t(s)$
3_17_14	3	15	15	24	57	0.05	9	39	0.27	Q65	40.00	−0.22
4_49_17	4	35	21	72	122	0.08	27	92	0.42	QX5	−28.57	−0.34
4gt10-v1_81	5	49	39	156	167	0.09	36	113	0.55	Q65	7.69	−0.46
4gt12-v0_86	5	81	66	228	282	0.10	54	167	0.52	Q65	18.18	−0.42
4gt13_90	5	43	33	112	156	0.11	30	98	0.57	Q65	9.09	−0.46
4gt13_91	5	39	30	104	145	0.10	27	91	0.56	Q65	10.00	−0.46
4gt4-v0_73	5	127	102	348	408	0.11	90	269	0.54	Q65	11.76	−0.44
4gt5_77	5	42	30	128	135	0.10	12	84	0.54	Q65	60.00	−0.44
4mod5-bdd_287	7	27	27	52	89	0.15	30	68	0.78	QX5	−11.11	−0.64
4mod5-v1_23	5	28	27	60	93	0.10	27	73	0.54	Q65	0.00	−0.45
4mod7-v0_94	5	54	27	76	154	0.10	30	126	0.56	QX5	−11.11	−0.46
aj-e11_165	4	56	36	144	175	0.07	39	139	0.38	QX5	−8.33	−0.30
aj-e11_168	4	38	24	80	102	0.08	24	86	0.42	Q65	0.00	−0.34
alu-bdd_288	7	33	30	52	94	0.14	24	74	0.71	Q65	20.00	−0.58
alu-v2_30	5	160	126	432	528	0.10	120	363	0.55	Q65	4.76	−0.45
alu-v2_31	5	146	108	364	486	0.12	96	326	0.57	Q65	11.11	−0.45
alu-v4_36	5	40	42	92	131	0.10	39	112	0.5	Q65	7.14	−0.40
C17_204	7	156	132	292	463	0.14	189	355	0.71	QX5	−43.18	−0.58
cm42a_207	14	593	588	1420	1890	0.32	660	1286	1.27	QX5	−12.24	−0.96
dc1_221	11	642	633	1520	2110	0.30	696	1485	1.19	QX5	−9.95	−0.89
decod24-enable_126	6	116	111	328	430	0.13	108	283	0.66	Q65	2.70	−0.53
decod24-v1_41	4	29	24	72	91	0.08	12	60	0.41	Q65	50.00	−0.33
decod24-v3_45	4	47	36	132	154	0.08	18	103	0.4	Q65	50.00	−0.33
ex3_229	6	114	102	340	412	0.13	90	251	0.67	Q65	11.76	−0.54
ham7_104	7	114	108	296	387	0.13	102	261	0.64	Q65	5.56	−0.51
hwb4_49	4	85	54	204	260	0.07	45	197	0.36	Q65	16.67	−0.29
hwb5_55	5	141	141	416	505	0.11	126	325	0.6	Q65	10.64	−0.48
hwb6_58	6	183	201	512	670	0.13	198	452	0.66	Q65	1.49	−0.53
hwb7_61	7	6252	5718	14 516	19 706	4.43	5628	13 518	1.43	Q65	1.57	3.00
hwb7_62	7	4142	3828	9840	12 847	2.07	3864	8736	1.17	QX5	−0.94	0.90
mini_alu_305	10	67	84	148	180	0.20	63	134	1.03	Q65	25.00	−0.83
mod5adder_127	6	165	165	476	562	0.12	150	398	0.61	Q65	9.09	−0.49
mod5adder_128	6	110	87	276	404	0.12	93	261	0.65	QX5	−6.90	−0.52
mod8-10_177	5	133	129	324	482	0.11	96	298	0.57	Q65	25.58	−0.46
mod8-10_178	5	99	78	288	319	0.11	78	235	0.56	Q65	0.00	−0.45
one-two-three-v0_97	5	98	87	276	354	0.10	81	243	0.54	Q65	6.90	−0.43
one-two-three-v1_99	5	47	48	152	173	0.10	45	123	0.52	Q65	6.25	−0.42
pm1_249	14	593	588	1348	1857	0.32	618	1356	1.29	QX5	−5.10	−0.97
rd32_273	5	160	93	372	472	0.11	138	361	0.5	QX5	−48.39	−0.39
rd53_131	7	160	135	252	422	0.12	141	361	0.63	QX5	−4.44	−0.50
rd53_135	7	108	99	216	329	0.14	114	256	0.75	QX5	−15.15	−0.60
rd73_140	10	90	90	212	270	0.20	87	193	1.02	Q65	3.33	−0.82
rd73_252	10	2380	2094	4588	6635	0.89	2730	5041	1.4	QX5	−30.37	−0.51
sqn_258	10	4726	4191	8760	12 173	2.32	5172	10 672	1.66	QX5	−23.41	0.67
sqrt8_260	12	1711	1512	3032	4444	0.58	1785	3676	1.42	QX5	−18.06	−0.84
sym6_145	7	1302	1188	3056	4037	0.36	1023	2708	0.75	Q65	13.89	−0.38
sym6_316	14	107	138	280	326	0.24	210	300	1.2	QX5	−52.17	−0.96
sym9_146	12	128	69	308	306	0.23	138	257	1.22	QX5	−100.00	−0.99
sym9_148	10	7224	6273	16 560	22 996	6.40	6711	15 670	2	QX5	−6.98	4.41
wim_266	11	326	324	828	1067	0.21	354	788	0.94	QX5	−9.26	−0.73
z4ml_269	11	1248	1131	2636	3589	0.43	1533	2827	1.2	QX5	−35.54	−0.77

Tab. 2: Comparison of NN-mapping on hexagonal 64-qubit and IBM Q65 architectures.

Benchmark Circuits			Hexagonal (8 × 8)			IBM Q65			Improvement	
Name	n	#CNOT	#CNOT	#Depth	t(s)	#CNOT	#Depth	t(s)	$\Delta_g(\%)$	$\Delta_t(s)$
4_49_16	4	78	12	140	0.34	51	189	0.41	76.47	0.07
4gt12-v0_86	5	81	24	159	0.49	54	167	0.52	55.56	0.03
4mod5-v0_20	5	9	0	13	0.00	6	19	0.44	100.00	0.44
4mod5-v1_22	5	10	0	13	0.00	6	19	0.45	100.00	0.45
4mod5-v1_23	5	28	6	53	0.41	27	73	0.54	77.78	0.13
5xp1_194	17	4777	1494	8448	3.09	5649	10 214	2.79	73.55	−0.30
add6_196	19	24 745	7266	44 041	8.32	32 499	54 162	12.21	77.64	3.89
alu-v2_30	5	160	57	309	0.51	120	363	0.53	52.50	0.02
alu-v2_31	5	146	27	270	0.49	96	326	0.56	71.88	0.07
alu1_198	20	324	180	454	2.54	438	541	1.39	58.90	−1.15
alu3_200	18	17 787	4032	30 709	5.32	20 376	37 210	7.46	80.21	2.14
C7552_205	21	3301	1212	6011	5.03	4458	7529	3.69	72.81	−1.34
cm150a_210	22	3653	954	6558	5.59	3735	8071	3.45	74.46	−2.14
cm163a_213	29	4911	1110	8722	7.23	5802	10 427	7.54	80.87	0.31
cu_219	25	16 396	3381	28 280	9.73	21 339	38 623	16.14	84.16	6.41
cycle10_293	39	222	327	463	6.33	693	586	4.10	52.81	−2.23
decod_217	21	3301	1284	6019	4.84	4185	7187	3.66	69.32	−1.18
dk17_224	21	13 154	3096	23 029	7.45	19 125	30 663	10.10	83.81	2.65
dk27_225	17	725	297	1281	2.86	1146	1849	2.03	74.08	−0.83
ham15_298	45	313	573	598	9.09	40 722	40 234	5.56	98.59	−3.53
ham7_299	21	151	126	294	2.93	318	427	1.73	60.38	−1.20
hwb6_301	46	589	804	1026	12.50	1863	1529	6.72	56.84	−5.78
hwb7_59	7	8423	1815	15 466	1.32	7260	18 500	1.79	75.00	0.47
hwb8_116	8	12 245	3843	22 708	1.97	13 101	26 210	2.70	70.67	0.73
inc_237	16	5582	1845	9923	3.19	360 666	366 212	5.12	99.49	1.93
life_238	10	19 936	3831	35 063	2.91	20 799	43 599	4.14	81.58	1.23
max46_240	10	22 924	4230	40 067	3.07	22 815	48 798	4.56	81.46	1.49
mod5adder_306	32	337	333	564	6.64	867	815	3.19	61.59	−3.45
mux_246	22	3652	1074	6585	6.39	3747	7598	3.45	71.34	−2.94
parity_247	17	16	36	28	1.07	87	59	1.39	58.62	0.32
pcler8_248	21	956	345	1597	3.84	1284	2063	2.13	73.13	−1.71
plus127mod8192_308	25	91	78	168	2.72	237	242	1.93	67.09	−0.79
plus63mod4096_309	23	82	66	160	2.53	177	220	1.77	62.71	−0.76
plus63mod8192_310	25	89	75	170	2.19	276	275	1.65	72.83	−0.54
rd73_312	25	246	207	487	4.35	522	651	2.11	60.34	−2.24
rd84_253	12	9677	2769	17 209	3.25	10 338	21 784	3.32	73.22	0.07
rd84_313	34	343	333	504	6.69	777	859	3.59	57.14	−3.10
sqr6_259	17	2292	828	4064	3.46	3114	5086	2.51	73.41	−0.95
sym9_193	10	39 296	7131	68 170	3.92	42 186	81 443	7.46	83.10	3.54
sym9_317	27	240	225	414	4.39	555	593	2.41	59.46	−1.98
t481_263	17	376	141	741	2.24	510	859	1.22	72.35	−1.02
urf1_278	9	24 916	14 307	46 644	5.06	40 344	60 383	7.23	64.54	2.17
urf2_152	8	30 180	18 735	65 365	4.87	39 120	86 416	6.45	52.11	1.58
urf3_279	10	66 435	31 281	121 272	11.01	102 738	160 603	18.21	69.55	7.20
x2_267	17	1990	780	3607	2.96	2439	4632	1.81	68.02	−1.15

Bibliography

[An21] Anis, Md S.; A.-Mitchell; Abraham, H. et al.: , Qiskit: An Open-source Framework for Quantum Computing, 2021. [Accessed: 2021-12-20].

[CDG21] Chow, Jerry; Dial, Oliver; Gambetta, Jay: , IBM Quantum breaks the 100-qubit processor barrier. `https://research.ibm.com/blog/127-qubit-quantum-processor-eagle/`, 2021. [Online; accessed 16-November-2021].

[CL21] Chang, K. Y.; Lee, C. Y.: Mapping Nearest Neighbor Compliant Quantum Circuits onto a 2-D Hexagonal Architecture. IEEE Trans. on CAD of Integrated Circuits and Systems, pp. 1–14, 2021.

[Da22] Datta, K.; Kole, A.; Sengupta, I.; Drechsler, R.: Nearest Neighbor Mapping of Quantum Circuits to Two-Dimensional Hexagonal Qubit Architecture. In: Int'l Symp. on Multiple-Valued Logic. May 2022.

[dADdS19] de Almeida, A. A. A.; Dueck, G. W.; da Silva, A. C. R.: CNOT gate mappings to Clifford+T circuits in IBM architectures. In: Int'l Symp. on Multiple-Valued Logic. pp. 7–12, 2019.

[Ga] Gambetta, Jay: , IBM QX backend information. `https://github.com/Qiskit/ibmq-device-information`. [Accessed: 2022-04-20].

[Go89] Goldberg, D.: Genetic algorithms in search, optimization, and machine learning. Addision-Wesley Professional, 1989.

[IB] IBM: , IBM Q. `https://www.research.ibm.com/ibm-q/`. [Accessed: 2019-03-20].

[IB17] IBM: , IBM Quantum Processor Types. `https://quantum-computing.ibm.com/services/docs/services/manage/systems/processors/`, 2017.

[IB20] IBM: , The map view of IBM Hummingbird Architecture. `https://quantum-computing.ibm.com/services?services=systems&type=Hummingbird&system=ibmq_brooklyn/`, 2020.

[Ko20] Kole, A.; Hillmich, S.; Datta, K.; Wille, R.; Sengupta, I.: Improved Mapping of Quantum Circuits to IBM QX Architectures. IEEE Trans. on CAD of Integrated Circuits and Systems, 39(10):2375–2383, 2020.

[Li17] Litinski, D.; Kesselring, M. S.; Eisert, J.; von Oppen, F.: Combining Topological Hardware and Topological Software: Color-Code Quantum Computing with Topological Superconductor Networks. Physical Review, 7(3):031048, 2017.

[Na21] Nation, Paul; Paik, Hanhee; Cross, Andrew; Nazario, Zaira: , The IBM Quantum heavy hex lattice. `https://research.ibm.com/blog/heavy-hex-lattice/`, 2021. [Online; accessed 7-July-2021].

[NBD21] Niemann, P.; Bandyopadhyay, C.; Drechsler, R.: Combining SWAPs and remote Toffoli gates in the mapping to IBM QX architectures. In: Design Automation and Test in Europe. pp. 1–6, 2021.

[Ni20] Niemann, P.; de Almeida, A. A. A.; Dueck, G.; Drechsler, R.: Design Space Exploration in the Mapping of Reversible Circuits to IBM Quantum Computers. In: 23rd Euromicro Conference on Digital System Design (DSD). pp. 401–407, 2020.

[RD12] Rahman, M.; Dueck, G. W.: Synthesis of Linear Nearest Neighbor Quantum Circuits. In: 10th Int'l Workshop on Boolean Problems. pp. 1–9, 2012.

[Ta18] Tang, H. et al.: Experimental quantum fast hitting on hexagonal graphs. Nature Photonics, 12(12):754–758, 2018.

[Wi08] Wille, R.; Große, D.; Teuber, L.; Dueck, G.W.; Drechsler, R.: RevLib: An Online Resource for Reversible Functions and Reversible Circuits. In: Int'l Symp. on Multi-Valued Logic. pp. 220–225, 2008. RevLib is available at http://www.revlib.org.

[ZPW19] Zulehner, A.; Paler, A.; Wille, R.: An Efficient Methodology for Mapping Quantum Circuits to the IBM QX Architectures. IEEE Trans. on CAD of Integrated Circuits and Systems, 38(7):1226–1236, 2019.