communications physics

ARTICLE

https://doi.org/10.1038/s42005-023-01191-3 OPEN

Scalable set of reversible parity gates for integer factorization

Martin Lanthaler [1✉], Benjamin E. Niehoff [2] & Wolfgang Lechner [1,2]

Classical microprocessors operate on irreversible gates, that, when combined with AND, half-adder and full-adder operations, execute complex tasks such as multiplication of integers. We introduce parity versions of all components of a multiplication circuit. The parity gates are reversible quantum gates based on the recently introduced parity transformation and build on ground-space encoding of the corresponding gate logic. Using a quantum optimization heuristic, e.g., an adiabatic quantum computing protocol, allows one to quantum mechanically reverse the process of multiplication and thus factor integers, which has applications in cryptography. Our parity approach builds on nearest-neighbor constraints equipped with local fields, able to encode the logic of a binary multiplication circuit in a modular and scalable way.

[1] Institute for Theoretical Physics, University of Innsbruck, Innsbruck, Austria. [2] Parity Quantum Computing GmbH, Innsbruck, Austria.
✉email: Martin.Lanthaler@uibk.ac.at

The inherent asymmetry in the complexity of multiplication and factorization has played a crucial role in cryptography and serves as the foundation for protocols such as RSA[1]. From the point of view of complexity theory, it is unlikely that the factoring problem is either in NP-complete or P, but it was proven that the problem lies in NP ∩ coNP[2]. As a search problem, factoring belongs to the class of total search problems TFNP and, more precisely, to PPA ∩ PPP under randomized reductions[3].

On the other hand, Shor's algorithm shows that factoring is an easy problem on quantum devices with an exponential speedup compared to any known classical algorithm[4,5], i.e., factoring belongs to the complexity class BQP. However, due to the extensive requirements on the number of qubits and quality of gates, Shor's algorithm is still limited to proof-of-concept demonstrations, far away from factoring numbers of magnitude used in real-world cryptosystems[6–11].

In the framework of the adiabatic quantum computer, integer factorization is recast as unconstrained optimization that is solved using adiabatic quantum dynamics[12–14]. Experimentally, this approach was first validated for the small number $143 = 11 \cdot 13$ using NMR technology[15–17]. Subsequently, larger bi-primes were factorized[18–21] using a novel approach based on multiplication tables[13]. In this approach, ancilla variables have to be introduced to mediate carry-overflows between different powers of two. Using ideas from algebraic geometry, i.e., Gröbner bases, the number of these auxiliary variables can be reduced by classical preprocessing, thereby enabling a demonstration of factorization up to 223.357[22]. Finally, simulations on D-Wave's hybrid quantum/classical simulator QBSOLV show, it is possible to factorize 1.005.973 on current hardware[23], (While this manuscript was under revision, this record has been broken in ref. [24] and ref. [25] building on a variational approach.). All these methods, which are based on variants of the multiplication table, reduce the integer factorization problem to a quadratic unconstrained binary optimization (QUBO) problem involving $\mathcal{O}(\log^2(n))$ logical qubits. To solve the optimization problem using adiabatic quantum computing techniques, the structure of the corresponding 2-local Hamiltonian must be mapped onto a connectivity graph available on hardware, e.g., via minor embedding[26,27].

In this article, we construct a novel reduction of the factorization problem to a parity-based spin model using a total of $\mathcal{O}(\log^2(n))$ physical qubits and interaction strengths of order $\mathcal{O}(1)$. That's a considerable improvement compared to previous approaches where $\mathcal{O}(\log^4(n))$ physical qubits and interaction strengths $\mathcal{O}(\log^2(n))$ are needed[13,19,28]. We propose to design a quantum factoring device by constructing a reversible version of a classical multiplication circuit built from basic AND, half-, and full-adder gates. This is accomplished by encoding each previously irreversible logic gate in the ground space of a spin Hamiltonian. By summing up all local "gate" Hamiltonians, we construct a spin glass model, which alongside constraints on input integers for multiplication or output integers for factorization, can perform either task [see Fig. 1]. The proposed Hamiltonians are tied to unit cells that can be arranged in a modular manner and are, therefore, easily scalable.

Results

Instead of following the path from multiplication table to QUBO to hardware graph, we propose to design a special-purpose quantum device with a hardware graph that directly implements the logic of a binary multiplication circuit. The framework in which this is done, is based on a promising approach to encode optimization problems into a spin system using local fields and quasi-local 3- and 4-body parity constraints[29]. In this more general context, the work presented here explores what's feasible within the parity framework, i.e., how the factoring problem can be tackled by building on parity 'plaquettes' equipped with local fields.

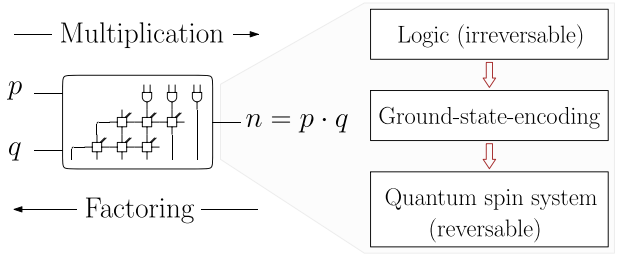

Fig. 1 General idea. By introducing reversible gates, factorization is considered a reverse multiplication. The problem is reformulated as an optimization problem by encoding the logic in the ground space of a spin Hamiltonian.

Overview. It is straightforward to build a Boolean circuit that takes as input the binary representations of two numbers p, q and outputs their product n. As depicted in Fig. 2, such a circuit can be built from basic AND gates and full-adder (FA) gates. Subsection "Ground state spin logic" shows how the logic of the circuit is implemented within a spin system by designing a Hamiltonian whose ground states encode valid input-output relations of these gates. With this, the Hamiltonian $H_{\text{circuit}} = \sum_\omega H^{(\omega)}_{\text{AND}} + \sum_\omega H^{(\omega)}_{\text{FA}}$ has a ground space spanned by states obeying the correct multiplicative relationship. To single out one specific triple (p, q, n), one adds a term $H_{\text{in}}(p, q)$, assigning an energy penalty to all states not having p and q as their inputs. Hence, finding the ground state of $H_{\text{product}} = H_{\text{circuit}} + H_{\text{in}}(p, q)$ would solve the (easy) task of multiplying numbers p and q. However, the same approach is applicable in the case of factorization: The output number n can be fixed by adding a term $H_{\text{factors}} = H_{\text{circuit}} + H_{\text{out}}(n)$. With this, the challenging task of factorization can be implemented via optimization, and in particular via the approach of quantum optimization.

In general, there exists a whole family of Hamiltonians able to encode the logic of the AND and the FA gates in its ground space. The choice made in subsection "Ground space spin logic" is motivated by the aspects of resources, i.e., the number of qubits and the number of interactions needed, and taking into account scalability. Instead of using these Hamiltonians directly, we use a parity representation of them as described in the subsection "Parity mapping", which reduces the degree and amount of interactions needed [see Fig. 3]. In the "Assembling instructions" subsection, we present two possible ways to implement the intergate connections. While the first layout [see Fig. 4] is efficient in terms of the number of auxiliary qubits, its geometry could make it difficult to implement due to its 3D nature. Therefore, we propose a planar version that uses parity constraints on a two-dimensional grid at the expense of additional qubits [see Fig. 5]. Either way, the resulting Hamiltonian

$$H^{\text{parity}}_{\text{factors}} := H^{\text{parity}}_{\text{circuit}} + H^{\text{parity}}_{\text{out}}(n) \tag{1}$$

is geometrical local and consists of unit cells s.t. factoring bigger numbers can be done in a scalable and modular way by adding more of these unit cells. Finally, we show how to encode the desired bi-prime n by defining $H_{\text{out}}(n)$ in the parity picture $H^{\text{parity}}_{\text{out}}(n)$. The resulting construction provides a scalable, geometric local, and programmable Hamiltonian whose ground state encodes the factors p and q s.t. $n = p \cdot q$ holds. The ground state can be prepared using quantum heuristics, e.g., quantum

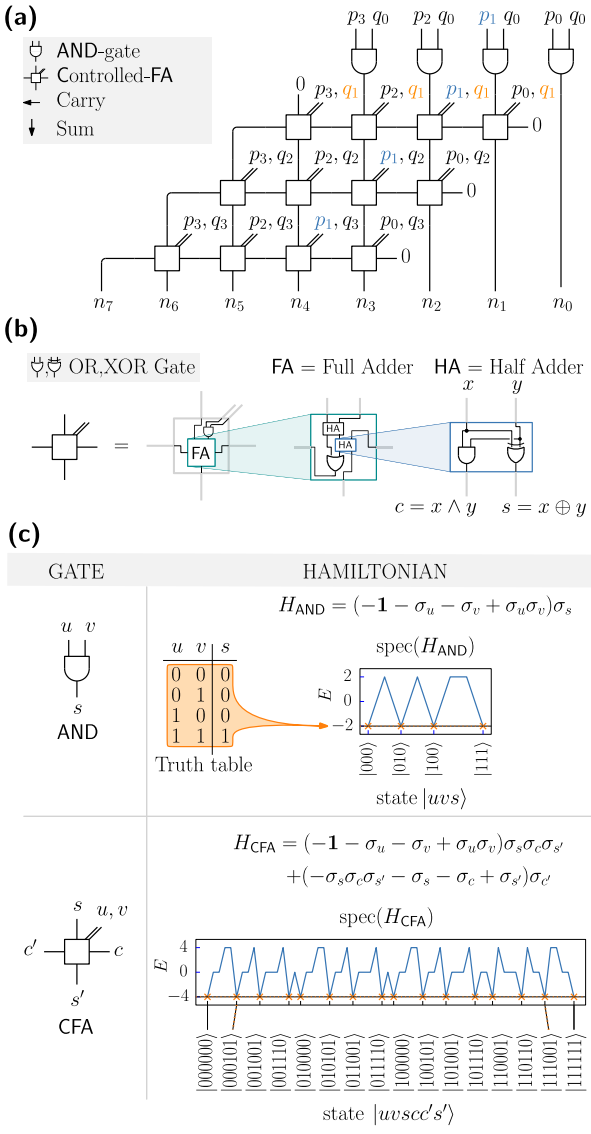

Fig. 2 Boolean multiplier circuit and ground-state-encoding. a Since $p_iq_j = p_i \wedge q_j$, the partial products p_iq_j can be formed using an AND gate. Subsequently, they can be summed up by full adders arranged on a 2D grid. Note that the gates share common inputs, as the values p_i and q_j repeat vertically or horizontally, respectively. **b** Full adders can be built from AND, OR, and XOR gates. **c** The AND gate can be encoded into the degenerated ground space of a Hamiltonian with four terms. Likewise, the controlled full adder can be encoded by a Hamiltonian consisting of eight terms.

annealing. In Supplementary Note 6, we offer a proof of principle by simulating the induced dynamics for a small instance.

Notation. We make receptive use of diagonal Hamiltonians of form

$$H = \sum_i a_i\sigma_z^{(i)} + \sum_{ij} a_{ij}\sigma_z^{(i)}\sigma_z^{(j)} + \sum_{ijk} a_{ijk}\sigma_z^{(i)}\sigma_z^{(j)}\sigma_z^{(k)} + \ldots . \quad (2)$$

with the Pauli σ_z defined by $\sigma_z := |0\rangle\langle 0| - |1\rangle\langle 1|$. Since Hamiltonians of the form of Eq. (2) are classical, we like to slim down the notation and write $\sigma \equiv \sigma_z$ s.t. σ_i denoted the Pauli-Z operator acting on qubits i. Terms $\sigma_i\sigma_j$ are used as a shorthand notation for the tensor product $\sigma_i \otimes \sigma_j$, where the subscript indicates which spin the operator acts. Natural numbers n, p, q are represented in binary as $n \equiv n_l \ldots n_1 n_0$ via $n = \sum_i n_i 2^i$ and $n_i \in \{0, 1\}$. Throughout this document, we deal with irreversible classical gates and

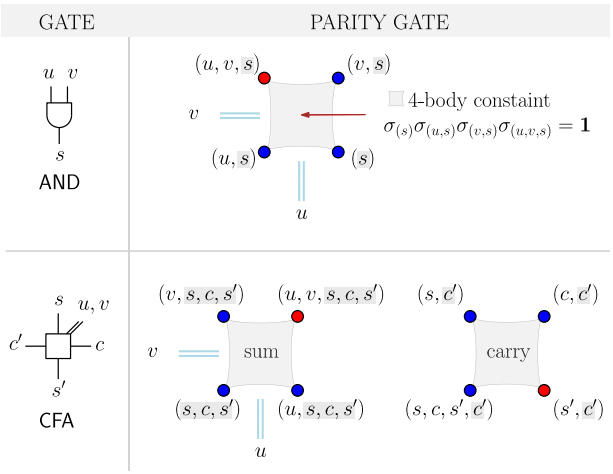

Fig. 3 Parity gates. AND and CFA gates are implemented via ground state coding on four or eight qubits, respectively [see Fig. 2c]. While individual physical qubits encode parity information (denoted by the tuple, e.g., $\sigma_{(u,v,s)}$), the gates are defined by local fields (red $+1$ and blue -1) and 4-body constraints $\prod_{\omega\in\square} \sigma_\omega = \mathbf{1}$ as additional penalty term (highlighted by gray squares). Thus, the operator H_{AND} is implemented on a single parity plaquette. Conversely, the Hamiltonian H_{CFA} is implementable by means of two parity plaquettes denoted by "sum" and "carry".

reversible (classical) gates encoded as the ground-state subspace of local spin Hamiltonians. The corresponding ground subspace is spanned by states defined by the logic of the classical gate and will be subsequently called the 'logical subspace'. At some point, it may be helpful to distinguish between the classical gate and a corresponding ground state implementation. Unless otherwise specified, the terminus gate should always refer to the classical gate, while corresponding parity implementations are denoted as 'parity gates'.

Ground state spin logic. The idea behind ground-state spin logic, as described in[30], is to embed a set of bit-strings $\mathcal{S} \subseteq \{0,1\}^m$ into the ground-state subspace of a spin Hamiltonian $H_{\mathcal{S}}$. Consider the AND gate, which defines four valid bit configurations (u, v, $s = u \wedge v$). The corresponding Hamiltonian H_{AND} is characterized by its degenerated lowest energy subspace

$$\mathcal{L}(H_{\mathrm{AND}}) = \mathrm{span}\{|000\rangle, |010\rangle, |100\rangle, |111\rangle\}. \quad (3)$$

It is easy to find a whole family of Hamiltonians with the desired subspace defined by Eq. (3). One particular choice is given by

$$H_{\mathrm{AND}} := (-\mathbf{1} - \sigma_u - \sigma_v + \sigma_u\sigma_v)\sigma_s. \quad (4)$$

We use this embedding because it has some desirable properties: The Hamiltonian consists of only four terms, the minimum number of terms required. The coupling strengths are -1 or 1, and the spectrum of H_{AND} takes only two values $\{-2, 2\}$, as shown in Fig. 2c. Furthermore, each of the indices occurs with even frequency (after expanding the expression the 'u' and 'v' occur twice and the 's' four times).

Similarly, the multiplicative relation between two integers can be encoded into the degenerated, low-energy ground space of a spin Hamiltonian. Figure 2a shows a possible circuit based on AND, half-adder, and full-adder gates. The AND implements the binary multiplication between bits p_i and q_j via the relation $p_i \wedge q_i = p_iq_j$, and further, the FA (full-adder) proceeds a previous carry overflow c and a sum s into a new sum and carry variable s' respectively c' (binary variables) such that

$$s + c + p_i \cdot q_j = 2c' + s' \quad (5)$$

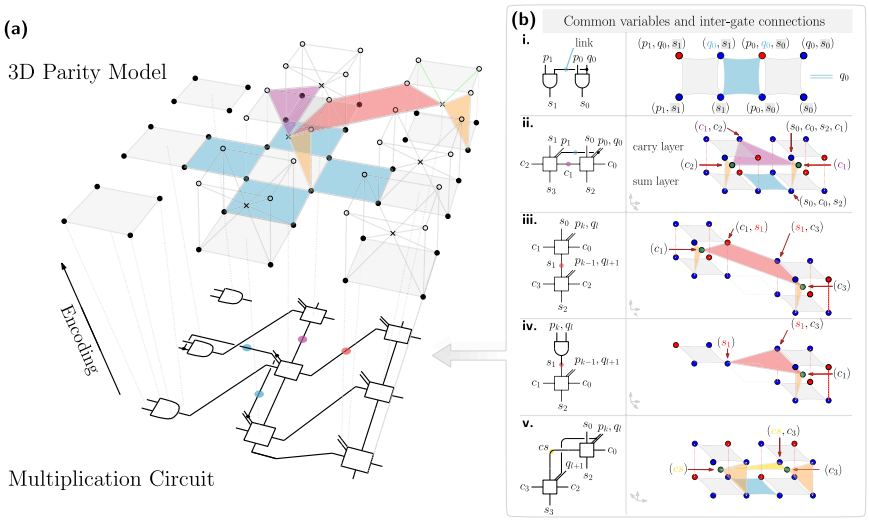

Fig. 4 Blueprint for parity factorization - Three-dimensional layout. a The multiplication circuit is mapped onto an array of physical qubits arranged on three layers of a 3D lattice. Dots and crosses represent qubits on different layers while faces represent constraints [see Supplementary Figure 2 for the labeling of the qubits and Supplementary Fig. 3 for the plot including all constraints]. Beside boundaries, the whole architecture can be constructed from repeating unit cell consisting of nine qubits connected to neighboring cells via additional constraints (blue, red, violet, yellow, orange). For illustrative purposes, only a selected number of qubits and constraints are displayed. **b** Building instructions. Each connection between logical gates translates into 3-body or 4-body parity constraints (blue, red, violet, yellow). The device is programmable in the sense that a given bi-prime n can be encoded by imposing additional 2-body constraints (**a** green). Thus the ground state $|n = pq\rangle$ reveals the factors p and q.

holds. See Fig. 2c and consult Supplementary Note 1 for further details.

Let us call the unit cell, build from an AND gate and a full-adder gate, a CFA-gate (**C**ontrolled-**F**ull**A**dder) [see Fig. 2b]. W.l.o.g. the whole circuit can be constructed from CFA gates arranged on a 2D grid with horizontal, vertical, and diagonal connections. The CFA gate operates on six bits, of which four act as input bits. The input q_j can be interpreted as control: When set to zero the gate acts as a half adder - processing the carry and sum inputs only. When set to one, it includes the p_i bit - acting as a full-adder. There are 16 valid bit configurations. These bit-strings can be encoded into the ground space of a spin Hamiltonian consisting of eight terms:

$$\begin{aligned} H_{\mathrm{CFA}} := & (-\mathbf{1} - \sigma_u - \sigma_v + \sigma_u\sigma_v)\sigma_s\sigma_c\sigma_{s'} \\ & + (-\sigma_s\sigma_c\sigma_{s'} - \sigma_s - \sigma_c + \sigma_{s'})\sigma_{c'}. \end{aligned} \tag{6}$$

Here, s and s' are the sum in- and outputs while c and c' denote the carry in-, and outputs, respectively. The two remaining nodes, u and v, are the inputs of the AND gate. Figure 2(c) shows the spectrum of this Hamiltonian. The ground-state manifold has energy -4 while the other states have an energy of 0 or $+4$, respectively. Remarkably, the first four terms are very similar to an AND gate. A formal replacement of the term σ_s with the product $\sigma_s\sigma_c\sigma_{s'}$ yields them from the AND gate Hamiltonian Eq. (4). Similar to the AND gate, this part of the Hamiltonian matches the parity of (s, c, s') according to the input on qubits u and v following the logic of an AND gate. Since they do not interact with the carry output c', we call these terms the 'sum terms'. However, without the 'carry terms' - the other four terms - the ground subspace would be 32-fold degenerate, which allows all possible states without fixing c'. Adding these carry terms removes this degeneracy and penalizes all states not having the correct carry bit.

Again, a whole family of Hamiltonians can encode the CFA logic. However, this Hamiltonian is desirable since (after expansion) it contains every index u, v, s, c, c' and s' an even number of times. We will now elaborate on why this property is crucial for our construction.

Parity mapping. Recently, a translation from an all-to-all 2-local spin model to a quasi-local model on a 2D lattice was introduced[31]. This approach can be generalized to include higher-order terms[29]. For each term appearing in Eq. (4) and Eq. (6) we introduce a physical spin. Imposing

$$\langle \sigma_{(i,j,k,\ldots)} \rangle_{\mathrm{phys}} = \langle \sigma_i\sigma_j\sigma_k\cdots \rangle_{\mathrm{logi}}, \tag{7}$$

the physical spin, subscribed by $(i, j, k, \ldots)$, is set to $|0\rangle_{\mathrm{phys}}$ if the logical system is in a state with an even number of $|1\rangle$'s on positions $i, j, k, \ldots$ and set to $|1\rangle_{\mathrm{phys}}$ otherwise. The newly introduced variable thus encodes the parity of a given subset of logical spins in a given state. In that sense, a 2-local term $\sigma_i\sigma_j$ - as discussed in[31] - only discriminate states according to their relative orientation between the spins s.t. parallel aligned spins map onto $|0\rangle_{\mathrm{phys}}$, and the subspace spanned by anti-parallel spins is mapped onto $|1\rangle_{\mathrm{phys}}$.

In the case of the AND gate, the Hamiltonian Eq. (4) has four terms. Consequently, we introduce four physical qubits. Let's call them $O := \sigma_s$, $U := \sigma_{(u,s)}$, $V := \sigma_{(v,s)}$ and $B := \sigma_{(u,v,s)}$. Rewritten in these new variables, the Hamiltonian H_{AND} reduces to a sum of local fields. Of special interest is the subspace of all physical states having a logical counterpart, i.e., they can be obtained by translating a logical state into the new variables. All such states from the logical subspace, obey the parity condition

$$OUVB = (\sigma_u)^2(\sigma_v)^2(\sigma_s)^4 = \mathbf{1}. \tag{8}$$

This holds because of our choice for the AND gate encoding Eq. (4) and the identity $(\sigma_i)^2 = \mathbf{1}$. Consequently, only every second basis state belongs to the logical subspace. This is reasonable since there are eight possible bit configurations (u, v, s), i.e., the Hilbertspace of three qubits is $2^3 = 8$ dimensional, and we map these to a system with four physical qubits with a total of 16 dimensions. The addition of a penalty term splits the set of physical states according to their parity and favors the logical subspace energetically. Summarizing, the parity

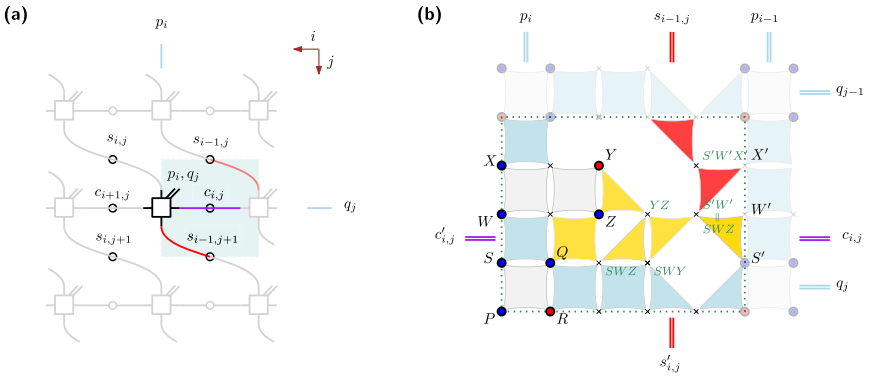

Fig. 5 Blueprint for parity factorization - Planar layout. a Multiplication circuit as an array of CFA cells. Cells are enumerated by their inputs, p_i and q_j. **b** Parity-based planar CFA unit cell. Compared to the 3D implementation, the unused space between primary plaquettes $(PQRS)_{ij}$ is enlarged to embed the second primary plaquette $(WXYZ)_{ij}$. Additional plaquettes transmit and process incoming and outgoing carry/sum variables. The multiplication circuit is then constructed by tiling the plane with unit cells, treating border cases differently, as shown in Supplementary Fig. 4.

version of the AND gates writes as

$$H_{\mathsf{AND}}^{\mathrm{parity}} := H_{\mathsf{AND}}^{\mathrm{local\ fields}} + H_{\mathsf{AND}}^{\mathrm{penalty}} = -O-U-V+B-kOUVB, \tag{9}$$

where $k>1$ [see Supplementary Note 4 for a detailed discussion of constraint strengths]. The corresponding qubits are arranged on a plaquette of four s.t. the 4-local penalty term is also local in a geometrical sense. While the output is directly accessible $O=\sigma_s$, the inputs u and v are encoded between horizontally or vertically adjacent qubits. Restricted to the subspace of states satisfying the parity constraint, we have the identities $BU=VO=\sigma_v$ and $BV=UO=\sigma_u$. If arranged according to Fig. 3, the plaquette encodes the variable v between horizontally adjacent qubits, while u is encoded vertically. In this sense, the plaquette acts as a wire, where information related to the variables u and v flow in perpendicular directions. These wires can be lengthened in one direction by extending them with additional plaquettes, as shown in Fig. 4(b.i). Here, the vertical encoded variable is q_0 and is transmitted via

$$\begin{aligned}\sigma_{q_0} &= \sigma_{s_0}\sigma_{(q_0,s_0)} = \sigma_{(s_0,p_0)}\sigma_{(p_0,q_0,s_0)}\\ &= \sigma_{(s_1)}\sigma_{(p_0,q_0,s_1)} = \sigma_{(p_1,s_1)}\sigma_{(p_1,p_0,q_0,s_1)},\end{aligned} \tag{10}$$

which is true for all states in the logical subspace.

In the following section, we focus on the controlled full adder. As pointed out, the four sum-terms in H_{CFA} do not differ conceptually from the AND gate encoding Eq. (4). Consequently, we can assign them to a 4-body plaquette with the corresponding parity penalty. We call this the 'sum plaquette'. The corresponding variables should be denoted as $S=\sigma_{(\vdash)}$, $P=\sigma_{(u,\vdash)}$, $Q=\sigma_{(v,\vdash)}$ and $R=\sigma_{(u,v,\vdash)}$ up to formal substitution $\vdash := s, c, s'$. Due to construction, the other four terms can be identified with a physical spin each s.t. $\sigma_{(s,c,s',c')}\sigma_{(s,c')}\sigma_{(c,c')}\sigma_{(s',c')}=\mathbf{1}$ if, and only if they are connected via the parity mapping Eq. (7) to a logical state. Let us denote these spin variables with W, X, Y, and Z in the order $\sigma_{(s,c,s',c')}, \sigma_{(s,c')}, \sigma_{(c,c')}$ and $\sigma_{(s',c')}$. These terms can be combined in a second plaquette, which we call the 'carry plaquette' [see Fig. 3]. Consequently, the parity version of the CFA Hamiltonian introduced in Eq. (6) is given by

$$\begin{aligned}H_{\mathsf{CFA}}^{\mathrm{parity}} &:= H_{\mathsf{CFA}}^{\mathrm{local\ fields}} + H_{\mathsf{CFA}}^{\mathrm{penalty}}\\ &= -S-P-Q+R-W-X-Y+Z\\ &\quad -kSPQR-kWXYZ,\end{aligned} \tag{11}$$

with penalty $k>1$. To contrast Eq. (11) to a direct implementation of H_{CFA}, we only need 1-local fields and two 4-body terms instead of three 2-body, one 3-body, three times a 4-body and one 5-body term in Eq. (6). Restricted to the logical subspace we have $SPQR=\mathbf{1}$, $WXYZ=\mathbf{1}$ and the original variables can be calculated from the new ones, as given by the expressions in Table 1.

Table 1 Variable transformation.

$\sigma_u = RQ = PS$,	$\sigma_v = RP = QS$
$\sigma_c = SWZ$,	$\sigma_s = SWX$
$\sigma_{c'} = SW$,	$\sigma_{s'} = SWY$.

Input and output variables of CFA gates are encoded as products of physical spin operators.

Assembling instructions. In this section, we present two different ways to construct the multiplication circuit from primary parity gates. The computational subspace is a degenerated stabilizer space spanned by states encoding valid multiplications of L_p-bit times L_q-bit integers. For simplicity, we assume $L_p=L_q=l/2$. In the first proposal, we arrange the AND and CFA plaquettes on two layers and add auxiliary qubits in a middle layer. Inter-gate connections and common input variables are implemented via 3- or 4-body parity constraints imposed on geometrically local qubits. A second proposal shows how additional auxiliary qubits can be used to implement the circuit in a conceptional simpler, two-dimensional layout.

Furthermore, we show how the degeneracy of the ground space can be lifted by adding additional constraints via penalty terms. This isolates (apart from exchanging p and q) a single ground state, which encodes the information of the prime factors $n=p\cdot q$.

Three-layer layout. The whole multiplication circuit can be thought of as being composed of individual AND and CFA gates by defining common input variables and by applying one of these operations (building inter-gate connections):

a. Identify the sum output of an AND gate with a sum input of a CFA gate (sum-to-sum).
b. Connect two CFA gates 'horizontally', i.e., connect the carry output to the carry input (carry-to-carry).
c. Connect two CFA gates 'vertically' by identifying the sum output of the first with the sum input of the second gate (sum-to-sum).
d. Take the carry output of a CFA gate and feed it into the sum input of a second CFA gate (carry-to-sum).

a. Common inputs: The binary multiplication circuit [as shown in Fig. 2a] can be rearranged on a 2D grid. This is demonstrated

exemplary on a small instance in Supplementary Note 2. Arranged like this, the circuit has common input variables q_j, which repeate for all gates in row j. Likewise, the variable p_i is a common input for all gates in column i. On the parity side, AND and CFA gates are implemented via one or two plaquettes, respectively [see Fig. 3]. Let's denote the factor-related inputs (not carry or sum inputs) of the gate in row i and column j with u_{ij} and v_{ij}. In the parity picture, these inputs are encoded vertically and horizontally within the AND plaquette and within the sum-plaquette of the CFA gate, as depicted in Fig. 3. When these plaquettes are placed on a grid, additional four-body constraints enforce common inputs between these plaquettes. Horizontally introduced four-body constraints impose $u_{ij} = u_{i'j} = q_j$ for all i, i'. Likewise, vertical constraints fix all inputs $v_{ij} = v_{ij'} = p_i$ for all j, j'. Hence, these plaquettes can be placed in a first layer, where the additional parity constraints implement common input variables [see Fig. 4a and Supplementary Fig. 2b, additional constraints are highlighted in blue].

b. Inter-gate connections: Connections of type a-d are implemented via additional parity constraints, as shown in Fig. 4b. The key here is the introduction of one ancilla qubit per unit cell. This extra qubit is enforced to match the value of the carry output $\sigma_{c'} = WS$ [see Table 1]. This is imposed by adding an additional 3-body constraint. Type b connections are established between units $\mathrm{CFA}_{i,j}$ and $\mathrm{CFA}_{i+1,j}$ by enforcing

$$Z_{i+1,j}(SW)_{i+1,j}(SW)_{i,j} = \mathbf{1}, \tag{12}$$

or equivalently $\sigma_{(c_{i+1,j},c'_{i+1,j})}\sigma_{c'_{i+1,j}}\sigma_{c'_{i,j}} = \mathbf{1}$, which is only possible if $c_{i+1,j} = c'_{i,j}$. The additional parity connection is shown in Fig. 4b.ii. Connections of type c involve gates $\mathrm{CFA}_{i,j}$, and $\mathrm{CFA}_{i-1,j+1}$ and are imposed by setting

$$Y_{i,j}(WS)_{i,j}X_{i-1,j+1}(SW)_{i-1,j+1} = \mathbf{1}. \tag{13}$$

Since $SWY = \sigma_{s'}$ and $SWX = \sigma_s$ this is only possible if $s_{i-1,j+1} = s'_{i,j}$. The corresponding parity plaquette is drawn as a red polygon in Fig. 4(b.iii). Cases a and d are implemented similarly and are shown in Fig. 4(b.iv-v).

c. Programming: By adding the term $H_{\mathrm{out}}^{\mathrm{parity}}(n)$, we penalize all states not resulting in the correct output $n = p \cdot q$, i.e., states associated with input numbers p and q, which multiplied together, do not yield the correct integer n. The digits of n are available as sum outputs s' to CFA gates, located at the right and lower border of our construction. Since $\sigma_{s'} = SWY$, the bit n_k can be encoded into a 2-body interaction between qubit SW and qubit Y both present as physical qubits. While the first is the ancilla qubit $\sigma_{c'}$, the second is part of the carry-plaquette. More specifically, if $n_k = 0$, the constraint strength should be negative, while if $n_k = 1$, it should be positive. Moreover, boundary cases could have no previous carry bit, or sum bit, as the rightmost CFA gates. If we still want to use the same parity gates, we have to set these bits to zero. With the identification $\sigma_c = SWZ$, this is accomplished by imposing a two-body interaction between the ancilla qubit SW and qubit Z. Figure 4a highlights these interaction terms with a green line between the ancilla qubit and the corresponding qubits from the top layer.

d. Scaling: A circuit capable of multiplying $l/2$ times $l/2$ bit number produces an output n of size $l = \lfloor \log_2(n) \rfloor$ bits. Such a circuit consists of $l/2$ AND gates and $l/2(l/2-1)$ CFA gates. Including the carry qubits, there are $l(9l-10)/4$ physical qubits.

If the multiplication device is used to find the factors of an odd bi-prime $n = p \cdot q$, both p and q are odd, i.e., $p_0 = q_0 = 1$. This makes the first row of AND gates redundant, as $\mathrm{AND}(u, 1) = u$ holds. Consequently, $2l$ qubits related to the AND gates can be removed from our count s.t.

$$\#_{\mathrm{phys}} = \frac{9}{4}l(l-2) \tag{14}$$

indicates the number of physical qubits required within this approach [see Supplementary Figure 1].

Planar layout. Although the three-layer construction is more efficient in the number of qubits, it could be challenging to implement. So here we propose a planar layout at the expense of additional ancilla qubits.

For the sake of simplicity, let us first consider an infinite lattice of CFA_{ij} gates. Their inputs should be $p_i, q_j, s_{i,j}$ and $c_{i,j}$, as denoted in Fig. 5(a). Their respective outputs are $s'_{i,j}$ and $c'_{i,j}$ and should be connected to inputs $s_{i-1,j+1}$ and $c_{i+1,j}$.

a. Common inputs: As a starting point for the planar layout, we consider the bottom layer of the previous construction. As mentioned, horizontal lines "transmit" the q_j variables, while vertical lines encode and transmit the variables p_i.

b. Inter-gate connections: We slightly modify the arrangement of sum-plaquettes to make room for embedding the associated carry-plaquettes. As shown in Fig. 5(b), adjacent sum-plaquettes are no longer connected by single plaquettes but by a series of three(four)-body plaquettes. Trivially, a series of square plaquettes would act as a wire to carry the variables p_i and q_j. However, to create additional space, squares can be replaced with two consecutive triangles pointing at each other. The carry plaquette $(WXYZ)_{ij}$ is placed in an elongated shape in the enlarged space. This embedding allows the full-adders logic to be implemented in a (4×5)-qubit unit cell equipped with local fields and additional parity constraints. Similarly to a biological cell, this elementary cell has a boundary structure composed of sum plaquettes connected via wires. Not only do these wires carry information in one direction, but they can also act like a membrane that allows information to flow perpendicularly.

The reader might easily check if $PQRS = \mathbf{1}$, $WXYZ = \mathbf{1}$ holds and all the other parity constraints are full-filled, then the additional qubits have values like SW, SWX, SWY, and SWZ as shown in Fig. 5(b). According to Table 1, these spins encode the logical variables $\sigma_{c'}$,σ_s, $\sigma_{s'}$, and σ_c. In the presented layout, they are positioned s.t. information is expressed on boundary plaquettes encoded perpendicular to p_i or q_j wires. Thus, two horizontal-adjacent unit-cell directly enforces the carry inter-gate connection. Similarly, diagonally adjacent cells (i, j) and $(i-1, j+1)$ link their sum variables via an additional information path that leads through the elementary cell $(i, j+1)$. This path is represented by the red triangles in Fig. 5(b).

Unlike the infinite lattice of CFAs, the binary multiplier has specific boundary cases. These cases are considered in Supplementary Note 3.

c. Programming: The number to be factored is imposed by adding a Hamiltonian $H_{\mathrm{out}}^{\mathrm{parity}}(n)$ as a sum of local field terms. These local fields determine the state of the qubits associated with $SWY = \sigma_{s'}$. For unit cells at the lower boundary of the layout, these qubits are depicted in Fig. 5b. Additionaly, see Supplementary Fig. 4b for a discussion of the right-hand-side boundary cases. Finally, the bottom left unit allows accessing the most significant bit n_l via $\sigma_{c'} = SW$.

d. Scaling: As depicted in Fig. 5, a single CFA unit can be implemented within the parity framework with 18 qubits arranged on a 4×5 array. However, the number of qubits can be improved to 14 per unit cell by considering a planar implementation without restricting to a grid-based layout [see Supplementary Fig. 5]. With this, factoring a number with l bits

length (when both factors have $l/2$ bits) can be done at the cost of approximately $\#_{\text{phys2D}} \approx \frac{7}{2} l^2$ qubits in a planar layout.

Discussion

The parity-based Hamiltonians described in the last section can be used to factor numbers n of size l bits were we assume that both factors fit into a $l/2$-bit register. In general, a sufficient length of the factors is determined by the bounds $L_p = l := \lfloor \log_2(n) \rfloor$ and $L_q = \lceil \frac{1}{2}(l+1) \rceil - 1$[12]. Not knowing the length of the factors in advance is part of the factoring problem and makes it challenging. The extreme cases in which one of the factors is very small or contrary when both are nearly equal in size can be approached classically. Using, e.g., simple trial division, factors up to a certain threshold size of r bits could be checked. On the other hand, factoring algorithms such as Fermat's method performs well if both factors are close in value[5]. When using the RSA protocol, one is interested in making an attack as challenging as possible. Therefore, one can assume that neither of the factors is small nor the same size. To span this range, the circuit must be able to encode the multiplication of $(L_p - r)$ bit times L_q bit numbers, resulting in a l bit number. Without pre-processing, i.e., $r = 0$, the maximal resources needed are approximately $\frac{3}{2}\#_{\text{phys}}(l)$ qubits. This leads to an estimate of $3.4\, l^2$ qubits in the case of the three-layer proposal and $5.3\, l^2$ qubits in the planar case [versus $25\, l^2$ qubits given for a similar approach[32]].

The current state-of-the-art RSA2048 protocol uses a key that is 2048 bits in size. To attack such a problem following our three-layer proposal one would need around 14 million qubits before classical preprocessing.

Alternative approaches for reformulating factorization as an optimization problem are summarized in Supplementary Note 1. They are based on the multiplication table [see Supplementary Table 1] by building factoring equations column by column. Squaring these leads to a QUBO problem involving $\mathcal{O}(l^2)$ logical variables. Mapping the QUBO problem to annealing hardware introduces another roughly quadratic overhead [see Supplementary Figure 1]. Our problem-specific approach reduces the hardware overhead s.t. we gain a scaling advantage over these approaches in the number of qubits [see also Supplementary Note 7, for a brief discussion on time complexity]. The parity gates and, thus, the whole factorization scheme can be implemented on a variety of platforms: From superconducting qubits like Transmon qubits, KPOs, or arrays of neutral Rydberg atoms[33–37]. See Supplementary Note 8 for a discussion of possible implementations.

Especially in platforms such as superconducting flux qubits, there is a fundamental trade-off between strong coupling and high coherence[38,39]. Here we see a major advantage of our factoring proposal compared to the QUBO approach. The range of interaction strengths in the QUBO model increases with system size and can vary from coupler to coupler. As size increases, these coupling strengths have to be controlled more precisely, challenging the performance of an annealer with a limited dynamic range. In our proposal, the functionality of the parity gates depends on precise local field conditions and parity constraints. Here the local fields only take values in $\{-1, 0, 1\}$ and are independent of the instance, i.e., they can be hard-coded into the device. On the other hand, there is no need for precise control over the strength of the parity constraints. Since they act as a penalty, it is sufficient to guarantee that they exceed a threshold, i.e., $k > 1$ [as described in Supplementary Note 4]. The number to be factored is programmed by turning on $\mathcal{O}(\log n)$ interactions. There, the magnitude of the interaction is not crucial, but rather its sign. Finally, the fact that information is encoded redundantly allows the parity implementation to provide a simple error detection and mitigation scheme, as discussed in Supplementary Note 5.

Methods

As our proposal builds on the parity framework, we provide a summary of the methodology behind the parity approach.

All-to-all connected optimization problems. Connectivity is a significant challenge in mapping optimization problems to hardware, especially when dealing with dense or all-to-all connected graphs. Ref. [31] proposed a solution to this problem by introducing a physical variable for each pair of logical variables $\sigma_{ij} := \sigma_i \sigma_j$. These variables are not independent of each other, which allows the original configuration space to be reclaimed in terms of a constrained subspace. Physical variables correspond to spins placed on a 2D grid, and constraints are enforced using penalties in the form of three- and four-body terms that penalize odd numbers of up spins on neighboring spins. The relevant constraints can be constructed from closed loops in the logical graph, e.g., $1-3-2-4$. Along such a closed loop, only an even amount of parity changes can occur. In the parity description, four spins, $\sigma_{13}, \sigma_{23}, \sigma_{24}$, and σ_{14}, are introduced to keep track of these changes. An even number of parity changes gives rise to constraint

$$\sigma_{13}\sigma_{23}\sigma_{24}\sigma_{14} = \mathbf{1}. \tag{15}$$

Introducing penalty terms for sufficiently many independent constraints, an Ising Hamiltonian $H_{\text{Ising}} = \sum_{i<j} J_{ij}\sigma_i\sigma_j$ is implemented through the Hamiltonian

$$H_{\text{par}} = \sum_{i<j} J_{ij}\sigma_{ij} - c_{\text{penalty}} \sum_{P\in\text{Plaq.}} \prod_{kl\in P} \sigma_{kl}, \tag{16}$$

build on programmable local fields $J_{ij}\sigma_{ij}$ and problem-independent penalty terms acting on plaquettes of neighboring spins. If the energy penalty c_{penalty} is strong enough, it separates the logical subspace from the rest, allowing a one-to-one correspondence with the logical model[40].

Higher order parity framework. As a generalization of ref. [31] the parity framework introduced in ref. [29] can be used to map a k-fold product of logical spins onto a single physical parity qubit, e.g., $J_{ijk}\sigma_i\sigma_j\sigma_k \to J_v\sigma_v$. This allows the direct encoding of combinatorial optimization problems with arbitrary higher-order k-body terms on a square lattice equipped with local fields J_v and parity constraints. Take, for example, a Hamiltonian including these four terms

$$\begin{aligned} H_{\text{part}} = {} & J_{12}\sigma_1\sigma_2 + J_{45}\sigma_4\sigma_5 \\ & + J_{123}\sigma_1\sigma_2\sigma_3 + J_{345}\sigma_3\sigma_4\sigma_5. \end{aligned} \tag{17}$$

Introducing four physical variables $\sigma_{24} := \sigma_2\sigma_4$, etc., for each term in Eq. (17) gives rise to the constraint $\sigma_{12}\sigma_{45}\sigma_{123}\sigma_{345} = \mathbf{1}$. This follows from

$$(\sigma_1\sigma_2)(\sigma_4\sigma_5)(\sigma_1\sigma_2\sigma_3)(\sigma_3\sigma_4\sigma_5) = \mathbf{1}, \tag{18}$$

using $(\sigma_i)^2 = \mathbf{1}$ and the fact that every logical variable appears twice. The terms specified in Eq. (17) may only represent a subset of terms we wish to implement on hardware. For example, the Hamiltonian could consist of two other terms, $J_{15}\sigma_1\sigma_5$ and $J_{24}\sigma_2\sigma_4$, which imply a second parity constraint $\sigma_{12}\sigma_{45}\sigma_{15}\sigma_{24} = \mathbf{1}$. Hence, it's possible to represent this Hamiltonian by six parity variables on two adjacent 4-body plaquettes, each equipped with a 4-body penalty term.

When introducing parity variables, the logical problem is embedded in a higher-dimensional Hilbert space. This embedding requires additional constraints to define the subspace containing the logical problem. A recent study ref. [29] investigated how to determine the necessary constraints and a corresponding layout for the physical spins, which they refer to as 'compiling' the problem for the parity framework. In this context, this work focuses on compiling the factoring problem by first identifying a suitable Hamiltonian for ground-state encoding.

Data availability

Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

Received: 29 October 2021; Accepted: 31 March 2023;
Published online: 17 April 2023

References

1. Rivest, R. L., Shamir, A. & Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **21**, 120–126 (1978).
2. Arora, S. & Barak, B. *Computational complexity: a modern approach* (Cambridge University Press, 2009).
3. Jerábek, E. Integer factoring and modular square roots. *J. Comput. Syst. Sci.* **82**, 380 (2016).
4. Shor, P. W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.* **26**, 1484–1509 (1997).

5. Crandall, R. & Pomerance, C. *Prime Numbers: A Computational Perspective*, Lecture notes in statistics (Springer New York, 2006).
6. Monz, T. et al. Realization of a scalable shor algorithm, Science 351 (2015).
7. Amico, M., Saleem, Z. H. & Kumph, M. Experimental study of shor's factoring algorithm using the ibm q experience. *Phys. Rev. A* **100**, 012305 (2019).
8. Smolin, J. A., Smith, G. & Vargo, A. Oversimplifying quantum factoring. *Nature* **499**, 163–165 (2013).
9. Gidney, C. & Ekera, M. How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits. *Quantum* **5**, 433 (2021).
10. Gouzien, É. & Sangouard, N. Factoring 2048-bit rsa integers in 177 days with 13436 qubits and a multimode memory. *Phys. Rev. Lett.* **127**, 140503 (2021).
11. Dash, A., Sarmah, D., Behera, B. K. & Panigrahi, P. K. Exact search algorithm to factorize large biprimes and a triprime on ibm quantum computer. https://arxiv.org/abs/1805.10478 arXiv:1805.10478 (2018).
12. Burges, C. J. C. Factoring as optimization. tech. Rep. MSR-TR-2002-83 (2002).
13. Schaller, G. & Schützhold, R. The role of symmetries in adiabatic quantum algorithms. *Quantum Info. Comput.* **10**, 109–140 (2010).
14. Albash, T. & Lidar, D. A. Adiabatic quantum computation. *Rev. Mod. Phys.* **90**, 015002 (2018).
15. Xu, N. et al. Quantum factorization of 143 on a dipolar-coupling nuclear magnetic resonance system. *Phys. Rev. Lett.* **108**, 130501 (2012).
16. Pal, S., Moitra, S., Anjusha, V. S., Kumar, A. & Mahesh, T. S. Hybrid scheme for factorisation: Factoring 551 using a 3-qubit nmr quantum adiabatic processor. *Pramana* **92**, 26 (2019).
17. Li, Z. et al. High-fidelity adiabatic quantum computation using the intrinsic hamiltonian of a spin system: Application to the experimental factorization of 291311. https://arxiv.org/abs/1706.08061 arXiv:1706.08061 [quant-ph] (2017).
18. Xu, K. et al. Experimental adiabatic quantum factorization under ambient conditions based on a solid-state single spin system. *Phys. Rev. Lett.* **118**, 130504 (2017).
19. Jiang, S., Britt, K. A., McCaskey, A., Humble, T. & Kais, S. Quantum annealing for prime factorization. *Sci. Rep.* **8**, 17667 (2018).
20. Peng, W. et al. Factoring larger integers with fewer qubits via quantum annealing with optimized parameters. *Science China Physics, Mechanics, and Astronomy* **62**, 60311 (2019).
21. Warren, R. H. Factoring on a quantum annealing computer. *Quantum Info. Comput.* **19**, 252–261 (2019).
22. Dridi, R. & Alghassi, H. Prime factorization using quantum annealing and computational algebraic geometry. *Sci. Rep.* **7**, 43048 (2017).
23. Wang, B., Hu, F., Yao, H. & Wang, C. Prime factorization algorithm based on parameter optimization of ising model. *Sci. Rep.* **10**, 7106 (2020).
24. Yan, B. et al. Factoring integers with sublinear resources on a superconducting quantum processor (2022).
25. Hegade, N. N. & Solano, E. Digitized-counterdiabatic quantum factorization (2023).
26. Choi, V. Minor-embedding in adiabatic quantum computation: I. the parameter setting problem. *Quant. Inf. Process.* **7**, 193–209 (2008).
27. Choi, V. Minor-embedding in adiabatic quantum computation: II. minor-universal graph design. *Quant. Inf. Process.* **10**, 343–353 (2010).
28. Mengoni, R., Ottaviani, D. & Iorio, P. Breaking RSA security with a low noise d-wave 2000q quantum annealer: computational times, limitations and prospects, https://arxiv.org/abs/2005.02268 arXiv:2005.02268 (2020).
29. Ender, K., ter Hoeven, R., Niehoff, B. E., Drieb-Schön, M. & Lechner, W. Parity quantum optimization: compiler. *Quantum* **7**, 950 (2023).
30. Whitfield, J., Faccin, M. & Biamonte, J. Ground state spin logic. *Europhys. Lett.* **99**, 57004 (2012).
31. Lechner, W., Hauke, P. & Zoller, P. A quantum annealing architecture with all-to-all connectivity from local interactions. *Sci. Adv.* **1**, https://doi.org/10.1126/sciadv.1500838 (2015).
32. Maezawa, M., Imafuku, K., Hidaka, M., Koike, H., & Kawabata, S. Design of quantum annealing machine for prime factoring, 2017 16th International Superconductive Electronics Conference (ISEC), 1 (2017).
33. Leib, M., Zoller, P. & Lechner, W. A transmon quantum annealer: decomposing many-body ising constraints into pair interactions. *Quantum Science and Technology* **1**, 015008 (2016).
34. Glaetzle, A., Bijnen, R., Zoller, P. & Lechner, W. A coherent quantum annealer with rydberg atoms https://doi.org/10.1038/ncomms15813.
35. Puri, S., Andersen, C. K., Grimsmo, A. L. & Blais, A. Quantum annealing with all-to-all connected nonlinear oscillators. *Nat. Commun.* **8**, 15785 (2017).
36. Goto, H. Quantum computation based on quantum adiabatic bifurcations of kerr-nonlinear parametric oscillators. *J. Phys. Soc. Jpn* **88**, 061015 (2019).
37. Lanthaler, M., Dlaska, C., Ender, K. & Lechner, W. Rydberg-blockade-based parity quantum optimization. https://doi.org/10.48550/ARXIV.2210.05604 (2022).
38. Weber, S. J. et al. Coherent coupled qubits for quantum annealing. *Phys. Rev. Appl.* **8**, 014004 (2017).
39. Hauke, P., Katzgraber, H. G., Lechner, W., Nishimori, H. & Oliver, W. D. Perspectives of quantum annealing: methods and implementations. *Rep. Prog. Phys.* **83**, 054401 (2020).
40. Lanthaler, M. & Lechner, W. Minimal constraints in the parity formulation of optimization problems. *N. J. Phys.* **23**, 083039 (2021).

Acknowledgements

We thank Kilian Ender and Clemens Dlaska for useful discussions and comments on the paper. Work was supported by the Austrian Science Fund (FWF) through a START grant under Project No. Y1067-N27 and the SFB BeyondC Project No. F7108-N38, the Hauser-Raspe Foundation, and the European Union's Horizon 2020 research and innovation program under grant agreement No. 817482. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001120C0068. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA.

Author contributions

The paper was drafted and edited by L.M. and reviewed by W.L. The project was initiated and supervised by W.L., and content and calculations were developed by L.M. and B.N. In particular, B.N. provided the planar layout.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s42005-023-01191-3.

Correspondence and requests for materials should be addressed to Martin Lanthaler.

Peer review information *Communications Physics* thanks the anonymous reviewers for their contribution to the peer review of this work.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.