D.3 Shor's algorithm

> If computers that you build are quantum,
> Then spies everywhere will all want 'em.
> Our codes will all fail,
> And they'll read our email,
> Till we get crypto that's quantum, and daunt 'em.
> — Jennifer and Peter Shor (Nielsen & Chuang, 2010, p. 216)

The widely used RSA public-key cryptography system is based on the difficulty of factoring large numbers.[11] The best classical algorithms are nearly exponential in the size of the input, $m = \ln M$. Specifically, the best current (2006) algorithm (the *number field sieve algorithm*) runs in time $e^{\mathcal{O}(m^{1/3}\ln^{2/3} m)}$. This is subexponential but very inefficient. Shor's quantum algorithm is bounded error-probability quantum polynomial time (BQP), specifically, $\mathcal{O}(m^3)$. Shor's algorithm was invented in 1994, inspired by Simon's algorithm.

Shor's algorithm reduces factoring to finding the period of a function. The connection between factoring and period finding can be understood as follows. Assume you are trying to factor M. Suppose you can find x such that $x^2 = 1 \pmod M$. Then $x^2 - 1 = 0 \pmod M$. Therefore $(x+1)(x-1) = 0 \pmod M$. Therefore both $x+1$ and $x-1$ have common factors with M (except in the trivial case $x = 1$, and so long as neither is a multiple of M). Now pick an a that is coprime (relatively prime) to M. If $a^r = 1 \pmod M$ and r happens to be even, we're done (since we can find a factor of M as explained above). (The smallest such r is called the *order* of a.) This r is the period of $a^x \pmod M$, since $a^{x+r} = a^x a^r = a^x \pmod M$.

In summary, if we can find the order of an appropriate a and it is even, then we can probably factor the number. To accomplish this, we need to find the period of $a^x \pmod M$, which can be determined through a Fourier transform.

Like the classical Fourier transform, the *quantum Fourier transform* puts all the amplitude of the function into multiples of the frequency (reciprocal period). Therefore, measuring the state yields the period with high probability.

[11] These section is based primarily on Rieffel & Polak (2000).

D.3.a Quantum Fourier transform

Before explaining Shor's algorithm, it's necessary to explain the quantum Fourier transform, and to do so it's helpful to begin with a review of the classical Fourier transform.

Let f be a function defined on $[0, 2\pi)$. We know it can be represented as a Fourier series,

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty}(a_k \cos kx + b_k \sin kx) = \frac{A_0}{2} + \sum_{k=1}^{\infty} A_k \cos(kx + \phi_k),$$

where $k = 0, 1, 2, \ldots$ represents the overtone series (natural number multiples of the fundamental frequency). You know also that the Fourier transform can be represented in the *ciscoid* (cosine + i sine) basis, where we define $u_k(x) \stackrel{\text{def}}{=} \text{cis}(-kx) = e^{-ikx}$. (The "$-$" sign is irrelevant, but will be convenient later.) The u_k are orthogonal but not normalized, so we divide them by 2π, since $\int_0^{2\pi} \cos^2 x + \sin^2 x \; \mathrm{d}x = 2\pi$. The Fourier series in this basis is $f(x) = \sum_{k=-\infty}^{\infty} \hat{f}_k \, \text{cis}(-kx)$. The Fourier coefficients are given by $\hat{f}_k = \frac{1}{2\pi}\langle u_k \mid f\rangle = \frac{1}{2\pi}\int_0^{2\pi} e^{ikx} f(x)\mathrm{d}x$. They give the amplitude and phase of the component signals u_k.

For the *discrete Fourier transform* (DFT) we suppose that f is represented by N samples, $f_j \stackrel{\text{def}}{=} f(x_j)$, where $x_j = 2\pi\frac{j}{N}$, with $j \in \mathbf{N} \stackrel{\text{def}}{=} \{0, 1, \ldots, N-1\}$. Let $\mathbf{f} = (f_0, \ldots, f_{N-1})^{\mathrm{T}}$. Note that the x_j are the $1/N$ segments of a circle. (Realistically, N is big.)

Likewise each of the basis functions is represented by a vector of N samples: $\mathbf{u}_k \stackrel{\text{def}}{=} (u_{k,0}, \ldots, u_{k,N-1})^{\mathrm{T}}$. Thus we have a matrix of all the basis samples:

$$u_{kj} \stackrel{\text{def}}{=} \text{cis}(-kx_j) = e^{-2\pi ikj/N}, \; j \in \mathbf{N}.$$

In $e^{-2\pi ikj/N}$, note that $2\pi i$ represents a full cycle, k is the overtone, and j/N represents the fraction of a full cycle.

Recall that every complex number has N principal N^{th}-roots, and in particular the number 1 (unity) has N principal N^{th}-roots. Notice that N samples of the fundamental period correspond to the N principal N^{th}-roots of unity, that is, ω^j where (for a particular N) $\omega = e^{2\pi i/N}$. Hence, $u_{kj} = \omega^{-kj}$. That is, $\mathbf{u}_k = (\omega^{-k\cdot 0}, \omega^{-k\cdot 1} \ldots, \omega^{-k\cdot(N-1)})^{\mathrm{T}}$. It is easy to show that the vectors $\mathbf{u}_k$ are orthogonal, and in fact that $\mathbf{u}_k/\sqrt{N}$ are ON (exercise). Therefore, $\mathbf{f}$

can be represented by a Fourier series,

$$\mathbf{f} = \frac{1}{\sqrt{N}} \sum_{k \in \mathbf{N}} \hat{f}_k \mathbf{u}_k = \frac{1}{N} \sum_{k \in \mathbf{N}} (\mathbf{u}_k^\dagger \mathbf{f}) \mathbf{u}_k.$$

Define the *discrete Fourier transform* of the vector $\mathbf{f}$, $\hat{\mathbf{f}} = \mathrm{F}\mathbf{f}$, to be the vector of Fourier coefficients, $\hat{f}_k = \mathbf{u}_k^\dagger \mathbf{f}/\sqrt{N}$. Determine F as follows:

$$\hat{\mathbf{f}} = \begin{pmatrix} \hat{f}_0 \\ \hat{f}_1 \\ \vdots \\ \hat{f}_{N-1} \end{pmatrix} = \frac{1}{\sqrt{N}} \begin{pmatrix} \mathbf{u}_0^\dagger \mathbf{f} \\ \mathbf{u}_1^\dagger \mathbf{f} \\ \vdots \\ \mathbf{u}_{N-1}^\dagger \mathbf{f} \end{pmatrix} = \frac{1}{\sqrt{N}} \begin{pmatrix} \mathbf{u}_0^\dagger \\ \mathbf{u}_1^\dagger \\ \vdots \\ \mathbf{u}_{N-1}^\dagger \end{pmatrix} \mathbf{f}.$$

Therefore let

$$\mathrm{F} \stackrel{\text{def}}{=} \frac{1}{\sqrt{N}} \begin{pmatrix} \mathbf{u}_0^\dagger \\ \mathbf{u}_1^\dagger \\ \vdots \\ \mathbf{u}_{N-1}^\dagger \end{pmatrix} = \frac{1}{\sqrt{N}} \begin{pmatrix} \omega^{0\cdot 0} & \omega^{0\cdot 1} & \cdots & \omega^{0\cdot(N-1)} \\ \omega^{1\cdot 0} & \omega^{1\cdot 1} & \cdots & \omega^{1\cdot(N-1)} \\ \omega^{2\cdot 0} & \omega^{2\cdot 1} & \cdots & \omega^{2\cdot(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \omega^{(N-1)\cdot 0} & \omega^{(N-1)\cdot 1} & \cdots & \omega^{(N-1)\cdot(N-1)} \end{pmatrix}. \tag{III.25}$$

That is, $\mathrm{F}_{kj} = \overline{u_{kj}}/\sqrt{N} = \omega^{kj}/\sqrt{N}$ for $k, j \in \mathbf{N}$. Note that the "$-$" signs in the complex exponentials were eliminated by the conjugate transpose. F is unitary transformation (exercise).

The *fast Fourier transform* (FFT) reduces the number of operations required from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$.[12] It does this with a recursive algorithm that avoids recomputing values. However, it is restricted to powers of two, $N = 2^n$.

The *quantum Fourier transform* (QFT) is even faster, $\mathcal{O}(\log^2 N)$, that is, $\mathcal{O}(n^2)$. However, because the spectrum is encoded in the amplitudes of the quantum state, we cannot get them all. Like the FFT, the QFT is restricted to powers of two, $N = 2^n$. The QFT transforms the amplitudes of a quantum state:

$$U_{\mathrm{QFT}} \sum_{j \in \mathbf{N}} f_j |j\rangle = \sum_{k \in \mathbf{N}} \hat{f}_k |k\rangle,$$

[12] The FFT is $\mathcal{O}(N \log N)$, but $N > M^2 = e^{2m}$. Therefore the FFT is $\mathcal{O}(M^2 \log M^2) = \mathcal{O}(M^2 \log M) = \mathcal{O}(m e^{2m})$

where $\hat{\mathbf{f}} \stackrel{\text{def}}{=} \mathrm{F}\mathbf{f}$.

Suppose f has period r, and suppose that the period evenly divides the number of samples, $r \mid N$. Then all the amplitude of $\hat{f}$ should be at multiples of its fundamental frequency, N/r. If $r \not| N$, then the amplitude will be concentrated *near* multiples of N/r. The approximation is improved by using larger n.

The FFT (and QFT) are implemented in terms of additions and multiplications by various roots of unity (powers of ω). In QFT, these are phase shifts. In fact, the QFT can be implemented with $n(n+1)/2$ gates of two types: (1) One is H_j, the Hadamard transformation of the jth qubit. (2) The other is a controlled phase-shift $S_{j,k}$, which uses qubit x_j to control whether it does a particular phase shift on the $|1\rangle$ component of qubit x_k. That is, $S_{j,k}|x_j x_k\rangle \mapsto |x_j x'_k\rangle$ is defined by

$$S_{j,k} \stackrel{\text{def}}{=} |00\rangle\langle 00| + |01\rangle\langle 01| + |10\rangle\langle 10| + e^{i\theta_{k-j}}|11\rangle\langle 11|,$$

where $\theta_{k-j} = \pi/2^{k-j}$. That is, the phase shift depends on the indices j and k.

It can be shown that the QFT can be defined:[13]

$$U_{\mathrm{QFT}} = \prod_{j=0}^{n-1} H_j \prod_{k=j+1}^{n-1} S_{j,k}.$$

This is $\mathcal{O}(n^2)$ gates.

D.3.b Shor's algorithm step by step

Shor's algorithm depends on many results from number theory, which are outside of the scope of this course. Since this is not a course in cryptography or number theory, I will just illustrate the ideas.

algorithm Shor:

[13] See Rieffel & Polak (2000) for this, with a detailed explanation in Nielsen & Chuang (2010, §5.1, pp. 517–21).

Input: Suppose we are factoring M (and $M = 21$ will be used for concrete examples, but of course the goal is to factor very large numbers). Let $m \stackrel{\text{def}}{=} \lceil \lg M \rceil = 5$ in the case $M = 21$.

Step 1: Pick a random number $a < M$. If a and M are not coprime (relatively prime), we are done. (Euclid's algorithm is $\mathcal{O}(m^2) = \mathcal{O}(\log^2 M)$.) For our example, suppose we pick $a = 11$, which is relatively prime with 21.

Modular exponentiation: Let $g(x) \stackrel{\text{def}}{=} a^x \pmod M$, for $x \in \mathbf{M} \stackrel{\text{def}}{=} \{0, 1, \ldots, M-1\}$. This takes $\mathcal{O}(m^3)$ gates and is the most complex part of the algorithm! (Reversible circuits typically use m^3 gates for m qubits.) In our example case, $g(x) = 11^x \pmod{21}$, so

$$g(x) = \underbrace{1, 11, 16, 8, 4, 2,}_{\text{period}} 1, 11, 16, 8, 4, \ldots$$

In order to get a good QFT approximation, pick n such that $M^2 \leq 2^n < 2M^2$. Let $N \stackrel{\text{def}}{=} 2^n$. Equivalently, pick n such that $2 \lg M \leq n < 2 \lg M + 1$, that is, roughly twice as many qubits as in M. Note that although the number of samples is $N = 2^n$, we need only n qubits (thanks to the tensor product). This is where quantum computation gets its speedup over classical computation; M is very large, so $N > M^2$ is extremely large. The QFT computes all these in parallel. For our example $M = 21$, and so we pick $n = 9$ for $N = 512$ since $441 \leq 512 < 882$. Therefore $m = 5$.

Step 2 (quantum parallelism): Apply U_g to the superposition

$$|\psi_0\rangle \stackrel{\text{def}}{=} H^{\otimes n}|0\rangle^{\otimes n} = \frac{1}{\sqrt{N}} \sum_{x \in \mathbf{N}} |x\rangle$$

to get

$$|\psi_1\rangle \stackrel{\text{def}}{=} U_g|\psi_0\rangle|0\rangle^{\otimes m} = \frac{1}{\sqrt{N}} \sum_{x \in \mathbf{N}} |x, g(x)\rangle.$$

For our example problem, 14 qubits are required [$n = 9$ for x and $m = 5$ for $g(x)$]. The quantum state looks like this (note the periodicity):

$$|\psi_1\rangle = \frac{1}{\sqrt{512}} \quad (\quad |0,1\rangle + |1,11\rangle + |2,16\rangle + |3,8\rangle + |4,4\rangle + |5,2\rangle +$$

$$|6,1\rangle + |7,11\rangle + |8,16\rangle + |9,8\rangle + |10,4\rangle + |11,2\rangle + \cdots)$$

Step 3 (measurement): The function g has a period r, which we want to transfer to the amplitudes of the state so that we can apply the QFT. This is accomplished by measuring (and discarding) the result register (as in Simon's algorithm).[14] Suppose the result register collapses into state g^* (e.g., $g^* = 8$). The input register will collapse into a superposition of all x such that $g(x) = g^*$. We can write it:

$$|\psi_2\rangle \stackrel{\text{def}}{=} \frac{1}{\mathcal{Z}} \sum_{x\in\mathbf{N} \text{ s.t. } g(x)=g^*} |x, g^*\rangle = \frac{1}{\mathcal{Z}} \sum_{x\in\mathbf{N}} f_x |x, g^*\rangle = \left[\frac{1}{\mathcal{Z}} \sum_{x\in\mathbf{N}} f_x |x\rangle\right] |g^*\rangle,$$

where

$$f_x \stackrel{\text{def}}{=} \begin{cases} 1, & \text{if } g(x) = g^* \\ 0, & \text{otherwise} \end{cases},$$

and $\mathcal{Z} \stackrel{\text{def}}{=} \sqrt{|\{x \mid g(x) = g^*\}|}$ is a normalization factor. For example,

$$\begin{aligned} |\psi_2\rangle &= \frac{1}{\mathcal{Z}}(|3,8\rangle + |9,8\rangle + |15,8\rangle + \cdots) \\ &= \frac{1}{\mathcal{Z}}(|3\rangle + |9\rangle + |15\rangle + \cdots)|8\rangle \end{aligned}$$

Note that the values x for which $f_x \neq 0$ differ from each other by the period; This produces a function f of very strong periodicity. As in Simon's algorithm, if we could measure two such x, we would have useful information, but we can't. Suppose we measure the result register and get $g^* = 8$. Fig. III.24 shows the corresponding $\mathbf{f} = (0,0,0,0,1,0,0,0,0,0,1,0,\ldots)$.

Step 4 (QFT): Apply the QFT to obtain,

$$\begin{aligned} |\psi_3\rangle &\stackrel{\text{def}}{=} U_{\text{QFT}} \left(\frac{1}{\mathcal{Z}} \sum_{x\in\mathbf{N}} f_x |x\rangle\right) \\ &= \frac{1}{\mathcal{Z}} \sum_{\hat{x}\in\mathbf{N}} \hat{f}_{\hat{x}} |\hat{x}\rangle. \end{aligned}$$

[14] As it turns out, this measurement of the result register can be avoided. This is in general true for "internal" measurement processes in quantum algorithms (Bernstein & Vazirani 1997).

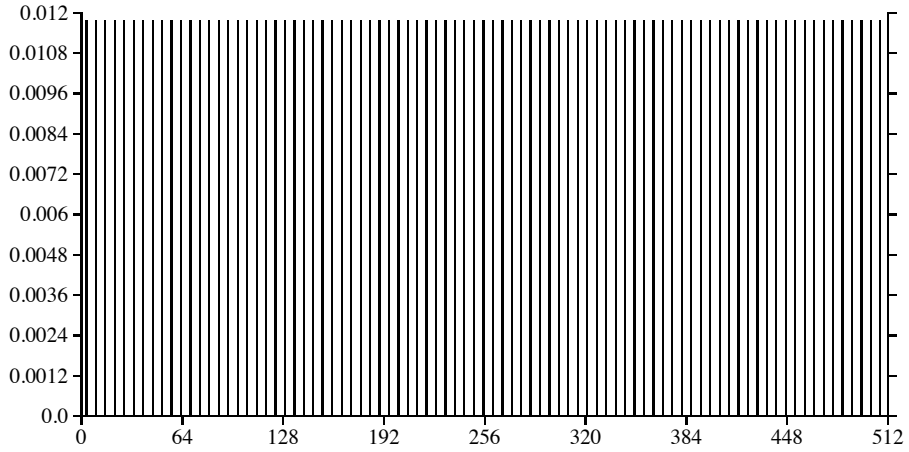

Figure III.24: Example probability distribution $|f_x|^2$ for state $Z^{-1}\sum_{x\in\mathbf{N}} f_x|x, 8\rangle$. In this example the period is $r = 6$ (e.g., at $x = 3, 9, 15, \ldots$). [fig. source: Rieffel & Polak (2000)]

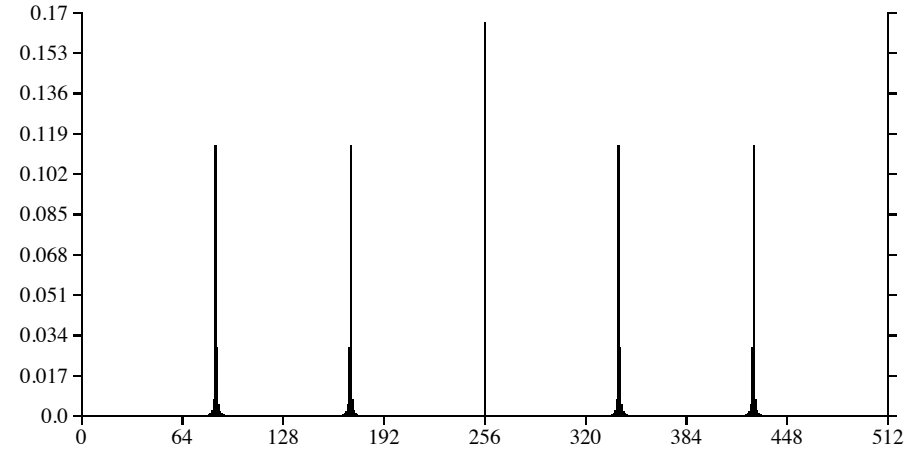

Figure III.25: Example probability distribution $|\hat{f}_{\hat{x}}|^2$ of the quantum Fourier transform of $\mathbf{f}$. The spectrum is concentrated near multiples of $N/6 = 512/6 = 85\ 1/3$, that is $85\ 1/3, 170\ 2/3, 256$, etc. [fig. source: Rieffel & Polak (2000)]

(The collapsed result register $|g^*\rangle$ has been omitted.)

If the period r divides $N = 2^n$, then $\hat{\mathbf{f}}$ will be nonzero only at multiples of the fundamental frequency N/r. That is, the nonzero components will be $|kN/r\rangle$. If it doesn't divide evenly, then the amplitude will be concentrated around these $|kN/r\rangle$. See Fig. III.24 and Fig. III.25 for examples of the probability distributions $|f_x|^2$ and $|\hat{f}_{\hat{x}}|^2$.

Step 5 (period extraction): Measure the state in the computational basis.

Period a power of 2: If $r \mid N$, then the resulting state will be $v \stackrel{\text{def}}{=} |kN/r\rangle$ for some $k \in \mathbf{N}$. Therefore $k/r = v/N$. If k and r are relatively prime, as is likely, then reducing the fraction v/N to lowest terms will produce r in the denominator. In this case the period is discovered.

Period not a power of 2: In the case r does not divide N, it's often possible to guess the period from a continued fraction expansion of v/N.[15] If v/N is sufficiently close to p/q, then a continued fraction expansion of v/N will contain the continued fraction expansion of p/q. For example, suppose the measurement returns $v = 427$, which is not a power of two. This is the result of the continued fraction expansion of v/N (in this case, 427/512) (see IQC):

i	a_i	p_i	q_i	ϵ_i
0	0	0	1	0.8339844
1	1	1	1	0.1990632
2	5	5	6	0.02352941
3	42	211	253	0.5

"which terminates with $6 = q_2 < M \leq q_3$. Thus, $q = 6$ is likely to be the period of f." [IQC]

Step 6 (finding a factor): The following computation applies however we

[15] See Rieffel & Polak (2000, App. B) for an explanation of this procedure and citations for why it works.

got the period q in Step 5. If the guess q is even, then $a^{q/2}+1$ and $a^{q/2}-1$ are likely to have common factors with M. Use the Euclidean algorithm to check this. The reason is as follows. If q is the period of $g(x) = a^x \pmod M$, then $a^q = 1 \pmod M$. This is because, if q is the period, then for all x, $g(x+q) = g(x)$, that is, $a^{q+x} = a^q a^x = a^x \pmod M$ for all x. Therefore $a^q - 1 = 0 \pmod M$. Hence,

$$(a^{q/2}+1)(a^{q/2}-1) = 0 \pmod M.$$

Therefore, unless one of the factors is a multiple of M (and hence $= 0 \bmod M$), one of them has a nontrivial common factor with M.

In the case of our example, the continued fraction gave us a guess $q = 6$, so with $a = 11$ we should consider $11^3+1 = 1332$ and $11^3-1 = 1330$. For $M = 21$ the Euclidean algorithm yields $\gcd(21, 1332) = 3$ and $\gcd(21, 1330) = 7$. We've factored 21!

Iteration: There are several reasons that the preceding steps might not have succeeded: (1) The value v projected from the spectrum might not be close enough to a multiple of N/r (D.3.b). (2) In D.3.b, k and r might not be relatively prime, so that the denominator is only a factor of the period, but not the period itself. (3) In D.3.b, one of the two factors turns out to be a multiple of M. (4) In D.3.b, q was odd. In these cases, a few repetitions of the preceding steps yields a factor of M.
□

D.3.c Recent progress

> To read our E-mail, how mean
> of the spies and their quantum machine;
> be comforted though,
> they do not yet know
> how to factorize twelve or fifteen.
> — Volker Strassen (Nielsen & Chuang, 2010, p. 216)

In this section we review recent progress in hardware implementation of

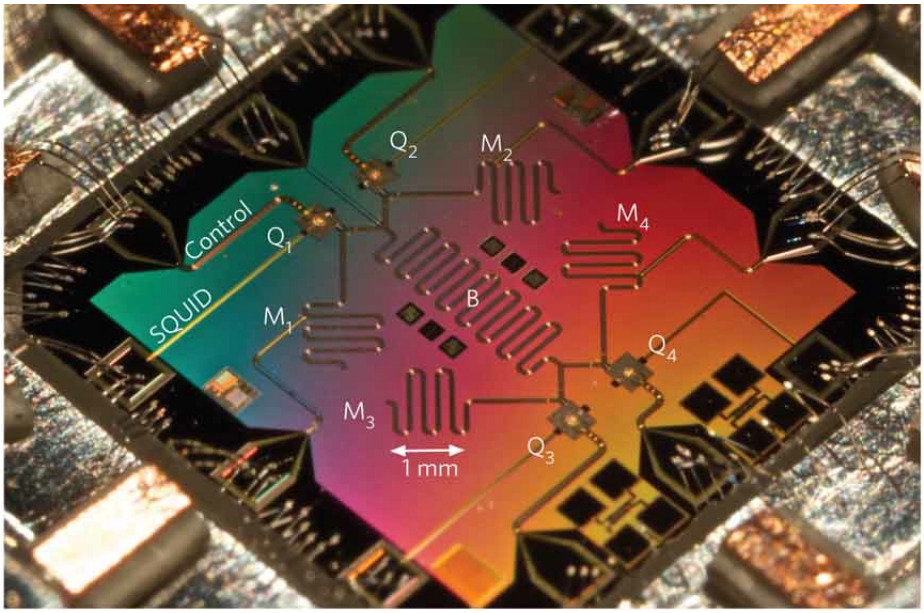

Figure III.26: Hardware implementation of Shor's algorithm developed at UCSB (2012). The M_j are quantum memory elements, B is a quantum "bus," and the Q_j are phase qubits that can be used to implement qubit operations between the bus and memory elements. [source: CPF]

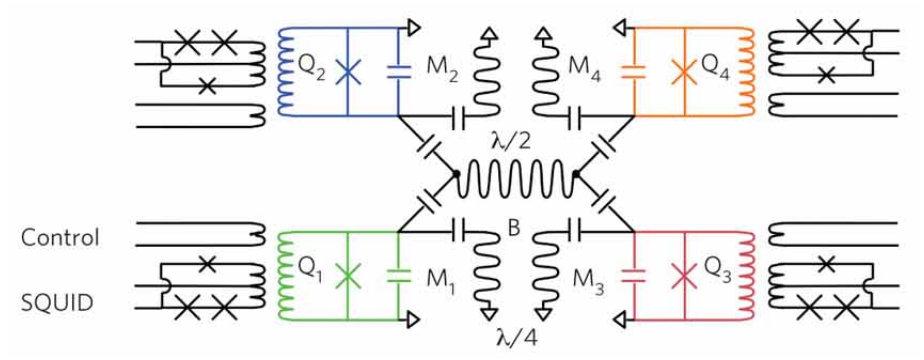

Figure III.27: Circuit of hardware implementation of Shor's algorithm developed at UCSB. [source: CPF]

Shor's algorithm.[16] In Aug. 2012 a group at UC Santa Barbara described a quantum implementation of Shor's algorithm that correctly factored 15 about 48% of the time (50% being the theoretical success rate). (There have been NMR hardware factorizations of 15 since 2001, but there is some doubt if entanglement was involved.) This is a 3-qubit compiled version of Shor's algorithm, where "compiled" means that the implementation of modular exponentiation is for fixed M and a. This compiled version used $a = 4$ as the coprime to $M = 15$. In this case the correct period $r = 2$. The device (Fig. III.26) has nine quantum devices, including four phase qubits and five superconducing co-planar waveguide (CPW) microwave resonators. The four CPWs (M_j) can be used as memory elements and fifth (B) can be used as a "bus" to mediate entangling operations. In effect the qubits Q_j can be read and written. Radio frequency pulses in the bias coil can be used to adjust the qubit's frequency, and gigahertz pulses can be used to manipulate and measure the qubit's state. SQUIDs are used for one-shot readout of the qubits. The qubits Q_j can be tuned into resonance with the bus B or memory elements M_j. The quantum processor can be used to implement the single-qubit gates X, Y, Z, H, and the two-qubit swap (iSWAP) and controlled-phase (C_ϕ) gates. The entanglement protocol can be scaled to an arbitrary number of qubits. The relaxation and dephasing times are about 200ns.

Another group has reported the quantum factoring of 21.[17] Their procedure operates by using one qubit instead of the n qubits in the (upper) control qubits. It does this by doing all the unitaries associated with the lowest-order control qubit, then for the next control qubit, updating the work register after each step, for n interations.

[16]This section is based primarily on Erik Lucero, R. Barends, Y. Chen, J. Kelly, M. Mariantoni, A. Megrant, P. O'Malley, D. Sank, A. Vainsencher, J. Wenner, T. White, Y. Yin, A. N. Cleland & John M. Martinis, "Computing prime factors with a Josephson phase qubit quantum processor." *Nature Physics* **8**, 719–723 (2012) `doi:10.1038/nphys2385` [CPF].

[17]See Martin-Lópex et al., "Experimental realization of Shor's quantum factoring algorithm using qubit recycling," *Nature Photonics* 6, 773–6 (2012).