

# Quantum Computing 2

## the Gate to Quantum Computing

Maart 2004

Robin van Schendel  
Wim van Gool  
Sander Triebert  
Stephan van Keulen  
Ronald Swets  
Sander Kleywegt

## 1 Inleiding

Om de zoveel tijd wordt er weer een uitvinding gedaan die ons aan betere, snellere computers helpt. Waar we zijn begonnen met inmiddels klassieke buizen zijn we de laatste jaren alleen nog maar bezig om meer en meer steeds kleinere transistoren op chips aan te brengen. Het probleem is inmiddels verschoven van het maken van machines die zulke kleine elektronica kunnen maken naar manieren om te zorgen dat de elektronica niet interfereert. Men loopt aan tegen de effecten van de quantum fysica, waar atomen niet meer gezien kunnen worden als bouwstenen, maar als complete systemen op zich. Vooralsnog hebben de quantum mechanische effecten een negatieve invloed op de weg naar kleinere transistoren, maar dit nadeel kan misschien wel worden omgebogen tot een voordeel.

## 2 Geschiedenis

In 1981 op de “Conference on the Physics of Computation”, gehouden aan het “Massachusetts Institute of Technology”, gaf de Amerikaanse natuurkundige Richard Phillipps Feynman een lezing [8] waarin hij zich voorstelde hoe aspecten uit de quantum fysica gebruikt konden worden om er berekeningen mee uit te voeren. Het zou een klassieke computer een extreme hoeveelheid tijd kosten om een quantum fysisch experiment te simuleren. Dit betekent dat in die eenvoudig experimenten in feite enorme berekeningen werden uitgevoerd. Het zou misschien mogelijk zijn om deze berekeningen te gebruiken voor nuttige toepassingen, zo redeneerde hij. Hiermee was de basis voor wat we nu quantum computing noemen gelegd.

In 1985 publiceerde, de eveneens natuurkundige, David Deutsch een paper [9] waarin hij op theoretische wijze een quantum computer beschreef. Hierin werden de eigenschappen van de quantum fysica verweven met de theorie van

Alan Turing over de Turing Machines. Deutsch beschreef de universele quantum computer opgebouwd uit quantum gates, die met ten hoogste polynomiale vertraging alle andere quantum computers zou kunnen simuleren.

Quantum computing werd pas echt populair toen Peter Shor, die werkzaam was bij AT&T Laboratories, in 1994 een algoritme [10] bedacht waarmee in theorie de factorisatie van grote gehele getallen snel en efficiënt opgelost kon worden met behulp van een quantum computer. De factorisatie is voor klassieke computers een ingewikkeld probleem wat veel tijd kost. En waarbij de berekeningstijd exponentieel toeneemt als de lengte van het getal met één vergroot wordt. Juist dit is de reden dat de factorisatie van getallen wordt gebruikt in moderne encryptietechnieken zoals RSA. Met het algoritme zoals Shor het beschrijft zou het mogelijk zijn deze encryptie binnen zeer korte tijd te kraken. Een uitgebreide uitleg van het algoritme van Shor is te vinden in paragraaf 6.1.

In 1995 kwam Shor met een andere belangrijke uitvinding. Hij had mogelijkheden ontdekt om foutcorrectie binnen de quantum computer toe te passen [11]. Dit is een belangrijk resultaat omdat een quantum computer erg gevoelig is voor verstoringen uit de buitenwereld. Op dit moment wordt er nog door vele mensen onderzoek gedaan naar hoe de foutcorrectie op een betere manier kan worden toegepast, het zal namelijk een sleutelrol gaan spelen bij het construeren van een werkende quantum computer.

Twee jaar nadat Shor zijn algoritme voor factorisatie publiceerde werd er een ander algoritme gevonden dat een niet zo extreme versnelling oplevert als die van Shor, maar wel zeer veelvuldig gebruikt zou kunnen worden gezien de steeds groeiende behoefte aan opslag en bewerking van informatie in grote databanken: in 1996 kwam Lov Grover van Bell Labs met een efficiënt algoritme [7] voor het zoeken in een ongeordende database. Dit betekent dat alle brute-force algoritmen voor het oplossen van problemen die aan de volgende eisen voldoen met een kwadratische versnelling [12] te maken krijgen:

- De enige manier om het probleem op te lossen is verschillende antwoorden proberen
- Er zijn  $n$  mogelijk juiste antwoorden
- Het bekijken van elke van de  $n$  mogelijke antwoorden kost evenveel tijd
- Elk antwoord is even waarschijnlijk

Voor een klassieke computer is dit een  $O(n)$  probleem, met behulp van een quantum computer en het algoritme van Grover wordt dit probleem gereduceerd tot een probleem van  $O(\sqrt{n})$ .

In 1997 werden de eerste papers gepubliceerd met betrekking tot fysieke aspecten van de quantum computer. Hierin werden technieken beschreven die zouden kunnen leiden tot het maken van een quantum computer. Hierbij kan gebruik gemaakt worden van bijvoorbeeld de spin van atomen binnen een molecuul zoals uitgewerkt in paragraaf 7.3.

In 1998 werd de eerste quantum computer gemaakt op de universiteit van California-Berkeley door een groep wetenschappers onder leiding van Isaac Chuang. Het betrof een 2 qubit computer [13] die werd gemaakt met behulp van *nucleaire magnetische resonantie* wat kortweg neerkomt op het draaien van atoomkernen onder invloed van verschillende magnetische velden.

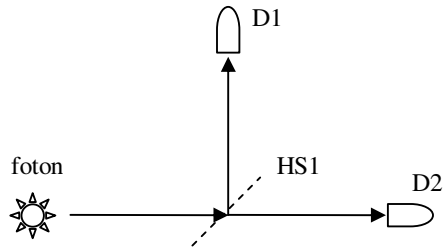
Voortbordurend op deze techniek werden in 1999, 2000 en 2001 respectievelijk de 3, 5 en 7 qubit quantum computers gemaakt [1,14,15,16]. De algoritmen die werden gedemonstreerd waren Grover's algoritme met de 3 qubit quantum computer. Met de 5 qubit quantum computer werd een volgorde algoritme bewerkstelligd. Als laatste werd met de 7 qubit het correcte antwoord gevonden van een factorisatie van het getal 15 met behulp van Shor's algoritme.

De 7 qubit quantum computer, zoals die gedemonstreerd werd in 2001, is tot nu toe de laatste noemenswaardige gebeurtenis in de geschiedenis van de quantum computer. Op het moment zijn er veel onderzoeken naar het realiseren van een quantum computer, dat er ooit een werkende quantum computer wordt gemaakt waar we echt wat mee kunnen is nog maar de vraag, maar de hoop is in ieder geval aanwezig.

### 3 Een Quantum Computer in het kort

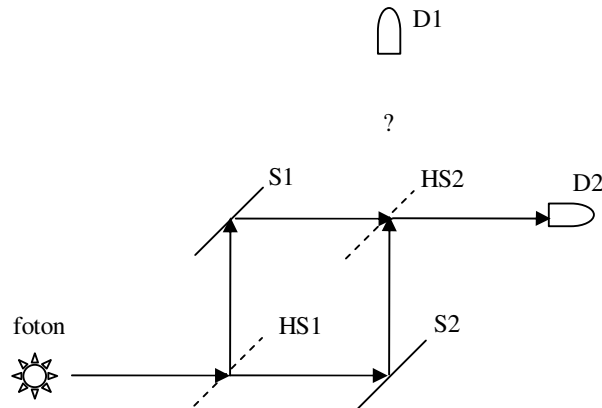
Een quantum computer verschilt in een aantal aspecten fundamenteel van de klassieke computer. Bij de klassieke computer worden aan het kleinste deeltje informatie, de bit, slechts 2 waarden toegekend. Een bit is ofwel 0 ofwel 1. Bij quantum computers ligt dit anders, daar kan een quantum bit (kortweg qubit) zich gedragen als een 0, als een 1, maar zich ook in een *superpositie* van die twee bevinden. Het qubit bevindt zich dan als het ware in zowel de 0 als de 1 positie tegelijk.

De standaard manier om dit verschijnsel uit te leggen is met behulp van een foton, twee half doorlatende spiegels (i.e. het foton heeft 50% kans om gereflecteerd te worden en 50% kans om door te gaan) en twee detectoren. In Figuur 3.1 is een opstelling gemaakt waarbij er één half doorlatende spiegel (HS1) geplaatst is in het pad van het foton. Zoals verwacht zal het foton een kans hebben van 50% om door detector 1 (D1) waargenomen te worden alsook een kans van 50% om door detector 2 (D2) waargenomen te worden.



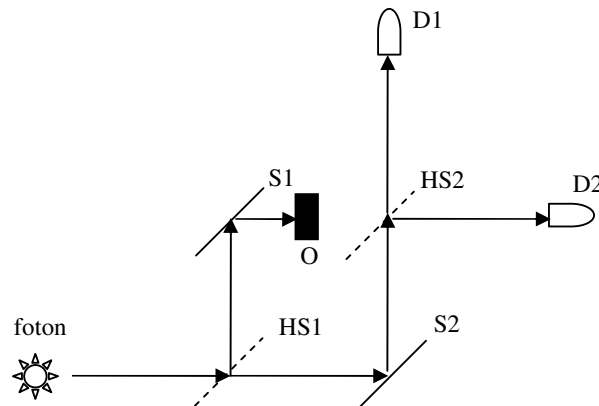
**Figuur 3.1** Half doorlatende spiegel

In Figuur 3.2 worden twee normale spiegels (S1 en S2) toegevoegd om beide mogelijke paden om te buigen zodat de paden weer bij elkaar komen. Precies op de kruising van de twee paden wordt een tweede half doorlatende spiegel (HS2) geplaatst waardoor er vier mogelijke paden ontstaan voor het foton waarvan er twee keer twee samenvallen bij de tweede half doorlatende spiegel. Het lijkt nu waarschijnlijk dat het foton wederom met 50% kans detector 1 (D1) bereikt en 50% kans heeft om detector 2 (D2) te bereiken. Het tegendeel blijkt waar, in 100% van de gevallen wordt het foton waargenomen door detector 2.



**Figuur 3.2** Een vreemd verschijnsel

Het lijkt er dus op dat het foton op een of andere manier beide paden tegelijk bewandelt. Hoe kan het foton anders weten dat er nog een andere weg mogelijk is? Om dit te testen is in één van de mogelijke paden een obstakel (O) geplaatst wat het foton ervan weerhoudt door te gaan naar de tweede half doorlatende spiegel, zie Figuur 3.3. Het blijkt dat beide detectoren dan wel weer een gelijke kans hebben om het foton waar te nemen.



Figuur 3.3 Test voor superpositie

Het feit dat het uitmaakt of een van de paden al dan niet geblokkeerd is lijkt voldoende bewijs te zijn voor het feit dat het foton op een of andere manier beide paden tegelijk bewandelt, we zeggen dan dat het foton zich in een superpositie bevindt.

Het idee dat een qubit zich in meerdere states tegelijk bevindt is elementair voor het begrip *quantum parallelisme*, hetgeen een quantum computer zo krachtig maakt. Stel dat we drie bits hebben die acht verschillende waarden kunnen uitdrukken. Bij de klassieke computer kunnen we een functie loslaten op één van de states waarin de drie bits zich bevinden. Als we echter qubits nemen, die zich allemaal in een superpositie bevinden van de twee basis states, kunnen we dezelfde functie in één keer toepassen op alle acht de verschillende mogelijkheden waarin de drie bits zich kunnen bevinden! Dit resulteert in het in één stap berekenen van alle mogelijk uitkomsten.

Een groot probleem is echter nog om dan het goede antwoord uit alle antwoorden te halen. Dit komt door de zogenaamde *entanglement* van verschillende deeltjes. Het lijkt er op dat verspreide deeltjes toch op één of andere vreemde manier met elkaar verward zijn. Het probleem hierbij is dan dat als we een qubit proberen te meten dit een verstoring van de hele quantum computer betekent omdat door het meten van dat ene deeltje we allerlei andere deeltjes beïnvloeden. Het zou dus kunnen dat door het bekijken van een antwoord, we andere (tussen)antwoorden ruïneren!

## 4 Introductie tot wiskundige begrippen

### 4.1 Inleiding

Deze paragraaf zal zich richten tot de algemene wiskundige begrippen en berekeningen die gebruikelijk zijn in de wereld van *quantum computing*. De hier formeel beschreven en aan de orde komende wiskunde zal in de komende

paragrafen uitgebreid en/of concreet gebruikt gaan worden. Informeel kunnen we zeggen dat we hier de bouwstenen gaan beschrijven die nodig zijn om thans in theorie quantum berekeningen uit te kunnen voeren.

## 4.2 Eén qubit

Zoals eerder beschreven is een qubit de meest elementaire bouwsteen in quantum computers. Een qubit wordt weergegeven met een vector in de 2-dimensionale ruimte, waar de dimensies de pure 0 en 1 states voorstellen. Als we voor de pure 0-state de normaalvector  $\begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle$  nemen en voor de pure 1-state  $\begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle$ , dan kunnen we met deze orthonormale set elke state weergeven op de volgende manier:

$$(Qubit)state = a \cdot |0\rangle + b \cdot |1\rangle = \begin{pmatrix} a \\ b \end{pmatrix} = |\varphi\rangle$$

Waarbij de volgende conditie altijd moet gelden:  $|a|^2 + |b|^2 = 1$  en  $a, b \in C$ .

Wanneer een qubit gemeten wordt, ongeacht in welke state deze zich bevindt, wordt altijd hetzij een 0, hetzij een 1 gemeten. De kans echter, welke van de twee gemeten wordt, wordt voor de 0-state en 1-state respectievelijk weergegeven door de waardes van  $|a|^2$  en  $|b|^2$ .

Omdat alle mogelijke kansen moeten optellen tot 1 volgt hieruit direct bovenstaande conditie. De coëfficiënten  $a$  en  $b$  zelf worden aangeduid als *probability amplitudes*.

Het manipuleren van de state kan door een qubit door een *gate* te sturen, wat wiskundig niets anders is dan het vermenigvuldigen van een matrix die bij de gate hoort met de state-vector van de qubit, ofwel het uitvoeren van *een lineaire transformatie* over de state-vector. Meer over gates volgt later in deze paragraaf.

## 4.3 Meer qubits

Stel nu, dat we  $n$  verschillende qubits  $|\varphi_1\rangle, |\varphi_2\rangle, \dots, |\varphi_n\rangle$  hebben die zich elk in een eigen state bevinden als hierboven beschreven, dat kunnen we al deze states combineren in één vector op de volgende manier:

$$|\varphi\rangle = \otimes_{i=1}^n |\varphi_i\rangle$$

Waarbij  $\otimes$  het *tensor product* is en  $|\varphi\rangle$  nu een  $2^n$ -dimensionale vector is. De ruimte waarin de totale state van één of meerdere qubits wordt weergegeven wordt ook wel de *Hilbertspace* genoemd, kortweg  $\mathcal{H}$ .

Het tensor product is als volgt gedefinieerd. We nemen twee vectoren:

$$|\varphi\rangle = \begin{pmatrix} a \\ b \end{pmatrix} \quad |\psi\rangle = \begin{pmatrix} c \\ d \end{pmatrix} \quad |\varphi\rangle \otimes |\psi\rangle = \begin{pmatrix} a \cdot \begin{pmatrix} c \\ d \end{pmatrix} \\ b \cdot \begin{pmatrix} c \\ d \end{pmatrix} \end{pmatrix} = \begin{pmatrix} ac \\ ad \\ bc \\ bd \end{pmatrix} = |\varpi\rangle$$

Het tensor product kan ook tussen matrices worden toegepast (i.e. in feite is een n-vector een (n x 1)-matrix). Dit wordt het *Kronecker product* genoemd. We kunnen de vector  $|\varpi\rangle$  nu weer als een combinatie van basis-states weergeven:

$$|\varpi\rangle = \begin{pmatrix} ac \\ ad \\ bc \\ bd \end{pmatrix} = ac \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} + ad \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} + bc \cdot \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} + bd \cdot \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Waarbij de basis-states nu ook genoteerd kunnen worden als

$$|00\rangle, |01\rangle, |10\rangle \text{ en } |11\rangle$$

Merk op dat voor de coëfficiënten moet gelden:

$$|ac|^2 + |ad|^2 + |bc|^2 + |bd|^2 = 1$$

Over het algemeen kan elke  $2^n$ -dimensionale vector weergegeven worden als een combinatie van  $2^n$  basis-states  $|q_1 q_2 \dots q_n\rangle$  met  $q_i \in \{0, 1\}$  (i.e. de basis-states vormen de orthonormale set die de  $2^n$  dimensionale Hilbertspace opspannen).

#### 4.4 Gates: introductie

Nu we een redelijk inzicht hebben in wat qubits voorstellen, kunnen we er ook wat mee gaan doen. Het quantum computing principe is gebaseerd op het

manipuleren en het uitlezen van de state van één of meerdere qubits. Het manipuleren van states van qubits gebeurt zoals gezegd door zogeheten *gates*. Deze zijn vergelijkbaar met de logische gates uit de klassieke computer.

Omdat we elke gate kunnen weergeven als een matrix en elke state kunnen weergeven als een vector, is de state-transformatie als volgt te noteren:

$$|\varphi'\rangle = A \cdot |\varphi\rangle$$

Waarbij A de matrix behorende bij de gate is,  $|\varphi\rangle$  de beginstate en  $|\varphi'\rangle$  de resulterende state is. Schematisch:

$$|\varphi\rangle \rightarrow [Gate] \rightarrow |\varphi'\rangle$$

Elke gate doet een transformatie zodanig dat  $|\varphi'\rangle$  van dezelfde dimensie is als  $|\varphi\rangle$ , met andere woorden, het is een lineaire transformatie van  $\mathcal{H} \rightarrow \mathcal{H}$ . Dit is inherent aan het feit dat elke gate evenveel input-qubits als output-qubits heeft.

Omdat één enkele qubit in feite een 2-dimensionale state-vector is, staat elke gate die een transformatie op een enkel qubit uitvoert voor een (2 x 2)-matrix:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} e \\ f \end{pmatrix} = \begin{pmatrix} ae + bf \\ ce + df \end{pmatrix}$$

Bovenstaande matrix moet zo gedefinieerd zijn dat ook voor de resulterende vector geldt:

$$|ae + bf|^2 + |ce + df|^2 = 1$$

Het ligt voor de hand dat elke willekeurige (n x n)-matrix aan de voorwaarde moet voldoen dat voor de resulterende (n x 1)-vector ook weer geldt dat

$$|a_1|^2 + |a_2|^2 + \dots + |a_n|^2 = 1 = \sqrt{\sum_1^n |a_i|^2}$$

De uitdrukking aan de rechterkant is de formule voor de *L2-norm* van een vector (de lengte van een vector in een n-dimensionale ruimte). We mogen in dit geval de wortel 'erbij schrijven' omdat we deze L2-norm gelijkstellen aan 1.

Elke state-vector voldoet dus aan de L2-norm. Omdat de L2-norm behouden moet worden, zijn er bepaalde eisen aan elke willekeurige  $n \times n$ -matrix  $A$ , die een transformatie uitvoert op een state-vector.

Orthogonale matrices blijken precies aan deze eisen te voldoen. Deze matrices hebben de eigenschap dat hun kolomvectoren de  $n$ -dimensionale ruimte opspannen, dat zij *inverteerbaar* zijn en dat zelfs geldt dat hun *getransponeerde gelijk is aan hun geïnverteerde*:

$$AA^{-1} = AA^T = I$$

De inverteerbaarheid van de matrices is dus een belangrijke eigenschap van matrices die de werking van een gate wiskundig onderbouwen; derhalve wordt ook wel gezegd dat de werking van gates inverteerbaar is, omdat voor elke achterliggende matrix  $A$  voor gate  $G_A$  een matrix  $A^{-1}$  gevonden kan worden. Met deze matrix  $A^{-1}$  kan vervolgens gate  $G_{A^{-1}}$  gemaakt worden die precies het tegenovergestelde doet van gate  $G_A$ .

## 5 Gates (Simpel en combinaties)

### 5.1 Reversibility (Omkeerbaarheid)

Alle natuurkundige wetten zijn omkeerbaar. Dat wil zeggen dat aan de hand van de uitvoer altijd de invoer teruggerekend kan worden. Klassieke computers lijken in eerste instantie niet te voldoen aan deze regel, omdat er met gemak functies gemaakt kunnen worden die niet omkeerbaar zijn. Neem bijvoorbeeld een booleaanse AND-gate (zie Figuur 5.1).

Input 1	Input 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

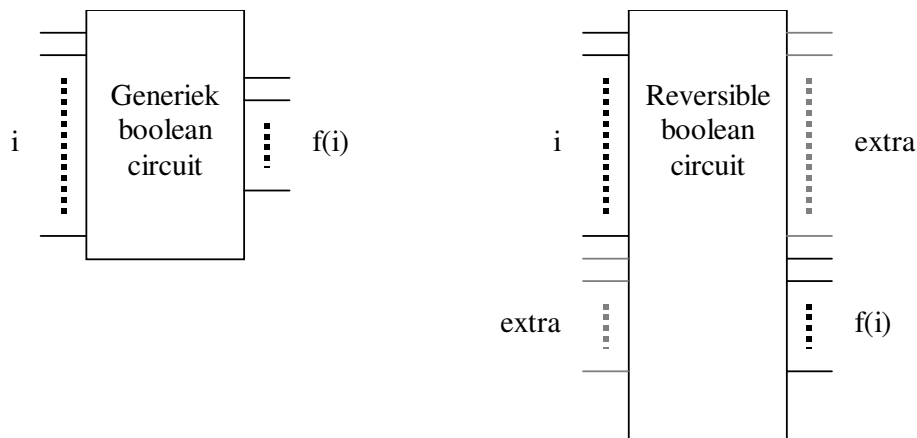
Figuur 5.1 Waarheidstabel AND-gate

In de waarheidstabel is makkelijk te zien dat er niet voor elke uitvoer een invoer af te leiden is. Dit is voldoende bewijs om te beweren dat de AND-gate niet omkeerbaar is.

Het antwoord op dit vraagstuk ligt in het verlies van warmte. Naast de gevraagde uitvoer produceert een klassieke AND-gate ook warmte, waarin de “verloren” informatie over de invoer zit.

In quantum computers is deze situatie ongeoorloofd. De afgifte van warmte hangt af van de invoer van de quantum gate. Die warmte heeft echter invloed op de state van de qubits en daarmee op de volgende invoer. Als de invoer verandert, wordt de uitvoer en de afgifte van warmte anders. Hierdoor verandert vervolgens de state van de qubits en is het kringetje rond. Quantum gates moeten dus omkeerbaar zijn [9,24].

De vraag is nu wanneer een gate omkeerbaar is. In 1980 is er een studie gedaan naar de reversibility van gates. Toffoli was één van de mensen die liet zien dat elk klassiek logisch circuit omkeerbaar gemaakt kan worden met slechts een aantal extra aansluitingen (Figuur 5.2).



**Figuur 5.2 Reversible Boolean Circuit**

Voor elke  $n$  inputs worden  $n$  outputs bijgemaakt en voor alle  $m$  outputs komen er  $m$  inputs bij. Totaal zijn er dan evenveel in- als outputs waardoor de gate omkeerbaar is. Er is namelijk voor elke invoer een uitvoer terug te vinden [25].

## 5.2 Quantum evaluatie

De meest eenvoudige states van qubits die er zijn, zijn  $|0\rangle$  en  $|1\rangle$ . Zoals vermeld in paragraaf 4 kunnen deze gerepresenteerd worden door de vectoren

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix} \text{ voor qubit } |0\rangle \quad \text{en}$$

$$\begin{pmatrix} 0 \\ 1 \end{pmatrix} \text{ voor qubit } |1\rangle.$$

Willen we nu een quantum state transitie uitvoeren op deze qubits (en dus de vectoren) dan dienen we een unitaire operatie  $U : C^2 \rightarrow C^2$  toe te passen. Deze wordt gevormd door een matrix

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \text{ en } |q_1\rangle = \begin{pmatrix} \alpha \\ \beta \end{pmatrix} \text{ zo danig dat,}$$

$$A|q_1\rangle = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \alpha a + \beta b \\ \alpha c + \beta d \end{pmatrix} = |q_2\rangle$$

Dit is een voorbeeld van een evaluatie van één enkel qubit. Zoals al eerder gebleken is, zijn er gates met meerdere input-qubits. Deze worden gerepresenteerd door een vector met een basis in  $C^2_1 \otimes C^2_2 \otimes \dots \otimes C^2_n$ .  $n$  is hier het aantal qubits. Er ontstaat dus een  $2^n$  dimensionale vector. De algemene notatie voor een evaluatie van  $n$  qubits luidt:

$$A|q_1 q_2 \dots q_n\rangle = \begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,2^n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,2^n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \dots & a_{m,2^n} \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_{2^n} \end{pmatrix} = |q'_1 q'_2 \dots q'_n\rangle$$

**Figuur 5.3 Evaluatie van  $n$  qubits**

We zien hier dat  $|q_1 q_2 \dots q_n\rangle$  gerepresenteerd wordt door de vector  $(v_1 \ v_2 \ \dots \ v_{2^n})^T$ . De definitie is terug te vinden in paragraaf 4.

### 5.3 Construeren van gates

Stel nu dat we de matrix voor het evalueren nog niet hebben en alleen zijn input en output. Dan kan door middel van het outer-product de bijbehorende evaluatiematrix worden afgeleid. Dit wordt duidelijk in de volgende berekening aan de hand van de NOT-gate:

$$|0\rangle\langle 1| = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes (0 \ 1) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \quad \text{input } |0\rangle \rightarrow \text{output } |1\rangle$$

$$|1\rangle\langle 0| = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes (1 \ 0) = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \quad \text{input } |1\rangle \rightarrow \text{output } |0\rangle$$

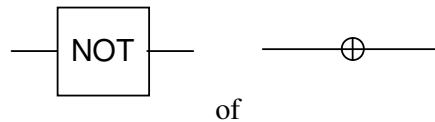
$$NOT = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

## 5.4 Single input quantum gates

De simpelste gates die we ons kunnen bedenken zijn gates met slechts één input. Doordat quantum-gates omkeerbaar moeten zijn hebben de gates met één input ook één output. Hieronder volgt een opsomming van enkele gates.

### 5.4.1 NOT gate

De NOT gate is waarschijnlijk de meest bekende gate bij degenen die met klassieke logische circuits werken. Deze gate inverteert de waarde van de qubit. Naast de klassieke werking (1 wordt 0 en 0 wordt 1) wordt de kans dat de qubit vervalt naar 0 verwisseld met de kans dat de qubit vervalt naar 1. Het symbool van de gate staat in Figuur 5.4.



Figuur 5.4 NOT gate

Laten we eens kijken naar het effect van een transformatiematrix op de qubits  $|0\rangle$  en  $|1\rangle$ . Hiervoor is de NOT-gate zeer geschikt. De NOT-gate heeft, zoals gezien in paragraaf 5.3 als transformatiematrix:

$$NOT = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Laten we deze los op het evaluatieproces, dan krijgen we de volgende twee vergelijkingen:

$$NOT|0\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = |1\rangle \quad \text{en}$$

$$NOT|1\rangle = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = |0\rangle$$

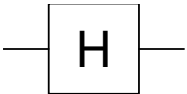
Verkort kan dit genoteerd worden in de volgende tabel:

Input	Output
$ 0\rangle$	$ 1\rangle$
$ 1\rangle$	$ 0\rangle$

**Figuur 5.5** Waarheidstabel NOT-gate

### 5.4.2 Hadamard gate

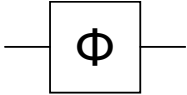
Voordat een quantum computer met meerdere invoerwaarden kan rekenen moeten qubits in superpositie staan. Hiervoor wordt de zogenaamde Hadamard gate gebruikt [27]. De Hadamard gate (H-gate) wordt over het algemeen aangegeven met het symbool in Figuur 5.6. Hier staan tevens de matrix en waarheidstabel.

Symbool:							
Matrix:	$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$						
Waarheidstabel:	<table border="1" style="margin-left: auto; margin-right: auto; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Input</th> <th style="text-align: center;">Output</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;"><math> 0\rangle</math></td> <td style="text-align: center;"><math>\frac{1}{\sqrt{2}}( 0\rangle +  1\rangle)</math></td> </tr> <tr> <td style="text-align: center;"><math> 1\rangle</math></td> <td style="text-align: center;"><math>\frac{1}{\sqrt{2}}( 0\rangle -  1\rangle)</math></td> </tr> </tbody> </table>	Input	Output	$ 0\rangle$	$\frac{1}{\sqrt{2}}( 0\rangle +  1\rangle)$	$ 1\rangle$	$\frac{1}{\sqrt{2}}( 0\rangle -  1\rangle)$
Input	Output						
$ 0\rangle$	$\frac{1}{\sqrt{2}}( 0\rangle +  1\rangle)$						
$ 1\rangle$	$\frac{1}{\sqrt{2}}( 0\rangle -  1\rangle)$						

**Figuur 5.6** Hadamard Gate

### 5.4.3 Phase Shift Gate

Deze gate wordt gebruikt om een qubit een andere fase te geven. Dit wordt specifiek gedaan om fasen van de superpositie gelijk te zetten of bepaalde interferentie (constructief of destructief) met andere qubits te veroorzaken [26,28]. Het symbool, de matrix en de waarheidstabel staan in Figuur 5.7.


Symbol:							
Matrix:	$Shift = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$						
Waarheidstabel:	<table border="1" data-bbox="756 648 985 793"> <thead> <tr> <th>Input</th> <th>Output</th> </tr> </thead> <tbody> <tr> <td><math> 0\rangle</math></td> <td><math> 0\rangle</math></td> </tr> <tr> <td><math> 1\rangle</math></td> <td><math>- 1\rangle</math></td> </tr> </tbody> </table>	Input	Output	$ 0\rangle$	$ 0\rangle$	$ 1\rangle$	$- 1\rangle$
Input	Output						
$ 0\rangle$	$ 0\rangle$						
$ 1\rangle$	$- 1\rangle$						

Figuur 5.7 Phase Shift Gate

#### 5.4.4 Square-root NOT gate

De square-root NOT gate is een bijzondere gate, omdat bij een gegeven input, de output pas bij meting vastligt. Echter, dit is nooit dezelfde waarde en lijkt willekeurig. Dit is een typisch geval van superpositie, omdat de qubit geen vaste waarde heeft. Voor de duidelijkheid wordt verwezen naar Figuur 3.2 met bijhorende uitleg [29].

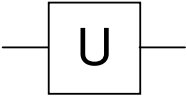
Twee square-root NOT gates maken een NOT gate. Figuur 5.8 geeft het symbool, de matrix en de waarheidstabel weer.

Symbol:							
Matrix:	$\sqrt{NOT} = \frac{1}{2} \begin{pmatrix} 1+i & 1-i \\ 1-i & 1+i \end{pmatrix}$						
Waarheidstabel:	<table border="1" data-bbox="711 1470 1110 1669"> <thead> <tr> <th>Input</th> <th>Output</th> </tr> </thead> <tbody> <tr> <td><math> 0\rangle</math></td> <td><math>\frac{1}{2}(1+i) 0\rangle + \frac{1}{2}(1-i) 1\rangle</math></td> </tr> <tr> <td><math> 1\rangle</math></td> <td><math>\frac{1}{2}(1-i) 0\rangle + \frac{1}{2}(1+i) 1\rangle</math></td> </tr> </tbody> </table>	Input	Output	$ 0\rangle$	$\frac{1}{2}(1+i) 0\rangle + \frac{1}{2}(1-i) 1\rangle$	$ 1\rangle$	$\frac{1}{2}(1-i) 0\rangle + \frac{1}{2}(1+i) 1\rangle$
Input	Output						
$ 0\rangle$	$\frac{1}{2}(1+i) 0\rangle + \frac{1}{2}(1-i) 1\rangle$						
$ 1\rangle$	$\frac{1}{2}(1-i) 0\rangle + \frac{1}{2}(1+i) 1\rangle$						

Figuur 5.8 Square-root NOT Gate

### 5.4.5 Unitary operation gate

Dit is een standaard gate die een *unitary operation* uitvoert. Deze zijn meestal opgebouwd uit Hadamard gates en phase shift gates. De exacte operatie die wordt uitgevoerd kan worden bepaald door de maker van de gate [28]. Het symbool, de matrix en de waarheidstabel staan in Figuur 5.9.

Symbol:							
Matrix:	$U = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$						
Waarheidstabel:	<table border="1" data-bbox="760 768 989 989"> <thead> <tr> <th>Input</th> <th>Output</th> </tr> </thead> <tbody> <tr> <td><math> 0\rangle</math></td> <td><math>\begin{pmatrix} a \\ c \end{pmatrix}</math></td> </tr> <tr> <td><math> 1\rangle</math></td> <td><math>\begin{pmatrix} b \\ d \end{pmatrix}</math></td> </tr> </tbody> </table>	Input	Output	$ 0\rangle$	$\begin{pmatrix} a \\ c \end{pmatrix}$	$ 1\rangle$	$\begin{pmatrix} b \\ d \end{pmatrix}$
Input	Output						
$ 0\rangle$	$\begin{pmatrix} a \\ c \end{pmatrix}$						
$ 1\rangle$	$\begin{pmatrix} b \\ d \end{pmatrix}$						

Figuur 5.9 Unitary Operation Gate

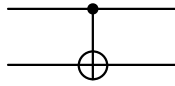
## 5.5 Multiple input quantum gates

### 5.5.1 Controlled NOT gate [CNOT]

De naam van deze gate zegt het al. Dit is een NOT gate met een control input. Als deze control input de waarde 1 krijgt, dan wordt de NOT geactiveerd. Als we dit in een waarheidstabel zouden zetten (Figuur 5.10) dan zien we dat dit overeenkomt met een zogenaamde XOR gate. Met het symbool in Figuur 5.11 wordt een CNOT aangegeven.

Control	Input	Output 1	Output 2
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Figuur 5.10 Waarheidstabel CNOT



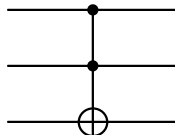
**Figuur 5.11 CNOT**

### 5.5.2 Toffoli gate [Controlled CNOT / $C^2$ NOT]

De Toffoli gate is een 3-input gate en wordt ook wel een controlled CNOT gate genoemd. Dit betekent niet meer dan een NOT gate met 2 control-inputs. Als beide 1 zijn, wordt de derde input geïnverteerd. Aan de hand van de waarheidstabel (Figuur 5.12) is te zien dat de gate als AND of als NAND kan fungeren: als Input gelijk is aan 0, dan is Output 3 de AND van Control 1 en Control 2. Bij Input gelijk aan 1, dan is Output 3 de NAND van Control 1 en Control 2. Het symbool voor de Toffoli gate staat in Figuur 5.13.

Control 1	Control 2	Input	Output 1	Output 2	Output 3
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

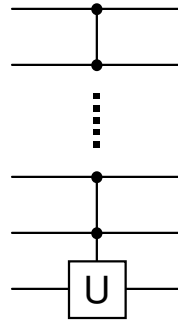
**Figuur 5.12 Waarheidstabel Toffoli Gate**



**Figuur 5.13 Toffoli Gate**

### 5.5.3 $C^n(U)$ gates

Elke unitary operation gate kan een n aantal control inputs hebben. De Toffoli gate valt ook onder deze gates. Hier is de unitary operation de NOT en zijn er twee control inputs. Elke controlled gate werkt in principe hetzelfde: als alle control inputs 1 zijn, wordt de unitary operation uitgevoerd op de laatste input [25,28,29]. Het algemene symbool staat in Figuur 5.14.



Figuur 5.14 Unitary Operation Gate

Uiteraard bestaan er nog veel meer soorten gates, deze worden hier echter niet verder besproken.

## 5.6 Aaneenschakeling van gates

Net als bij de klassieke gates bestaat er de mogelijkheid om quantum gates aan elkaar te koppelen. Stel er zijn twee matrices  $A : C^n \rightarrow C^m$  ( $A$  is dus een  $(m \times n)$  matrix) en  $B : C^q \rightarrow C^p$  (een  $(p \times q)$  matrix), dan is het tensorproduct  $A \otimes B : C^n \otimes C^q \rightarrow C^m \otimes C^p$ . Het resultaat is een  $(mp \times nq)$  matrix.

Als voorbeeld is gegeven de matrices  $A(m \times n)$  en  $B(m \times n)$  dan is het (*rechter*) *Kronecker product*:

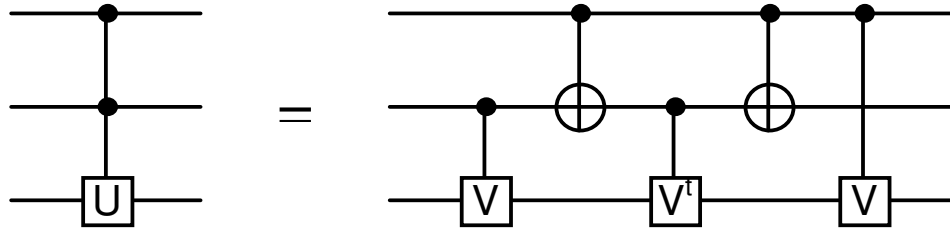
$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix}$$

Alle genoemde gates zijn omkeerbaar, maar ze zijn niet allemaal universeel. Een universele gate is een gate waarmee elk logisch circuit kan worden nagebouwd (Bij de klassieke circuits is dat de NAND gate). Het is niet moeilijk te bedenken dat een universele gate meer dan één input nodig heeft. Het is daarentegen niet eenvoudig te bepalen hoe groot een gate moet zijn om zeker weten dat deze universeel toepasbaar is.

DiVincenzo heeft echter in 1995 bewezen [30] dat een set van een aantal 2 input quantum gates universeel is. De bewijsvoering zal verder niet besproken

worden. Als de 2 input quantum gates universeel zijn, dan moet er een mogelijkheid bestaan om hiermee bijvoorbeeld een toffoli gate te bouwen.

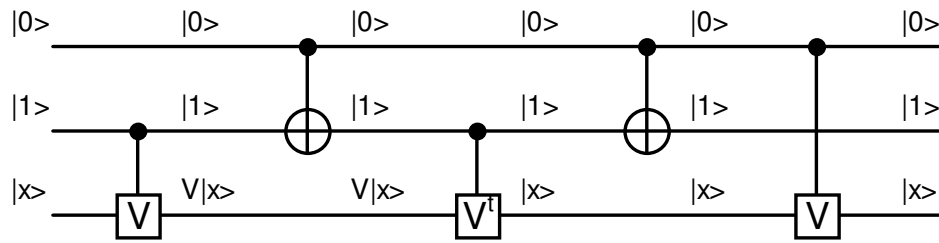
Volgens de methode (Figuur 5.15) van Barenco uit 1995 [30] blijkt dat elke  $C^n(U)$  gate van CNOT,  $C(V)$  en  $C(V^\dagger)$  gates kan worden gebouwd. Hierbij geldt dat  $V^2=U$ .



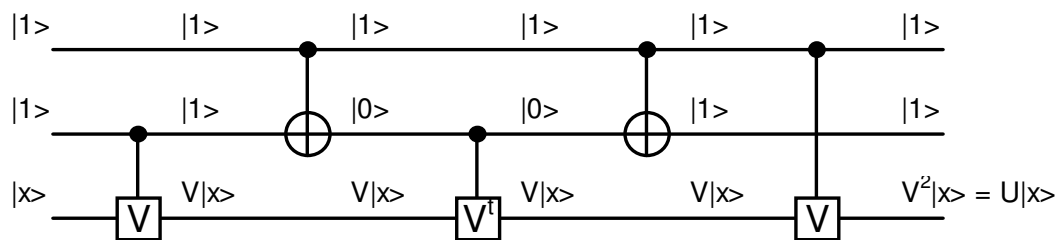
**Figuur 5.15 Methode van Barenco**

Met een eenvoudig getallenvoorbeeld kan worden nagegaan dat deze stelling klopt. Het netwerk heeft een grootte 5. Dat wil zeggen dat er 5 gates achteréén geschakeld zijn. Aan de hand van de waarheidstabellen gegeven bij de gates of gewoon door berekening kunnen de waarden worden nagegaan. In Figuur 5.16 en Figuur 5.17 staan twee voorbeeld inputs gegeven. Voor Figuur 5.17 zal een toelichting gegeven worden.

De eerste gate is een controlled unitary operation gate. De control qubit heeft waarde  $|1\rangle$  waardoor de unitary operation op de invoer  $|x\rangle$  wordt uitgevoerd. Dit geeft dus  $V|x\rangle$ . De tweede gate is een CNOT. De controlbit van de CNOT is  $|0\rangle$  waardoor de input  $|1\rangle$  dus niet geïnverteerd wordt. De derde gate werkt hetzelfde de eerste: De unitary operation  $V^\dagger$  wordt uitgevoerd op de invoer  $V|x\rangle$ . Zoals eerder beschreven heffen de operaties  $V$  en  $V^\dagger$  elkaar op en wordt de uitvoer dus netjes  $|x\rangle$ . De vierde gate is een CNOT met dezelfde invoer als de tweede CNOT. De vijfde en laatste gate is weer een controlled unitary operation gate. De control qubit is hier  $|0\rangle$  waardoor er geen operatie wordt uitgevoerd.



**Figuur 5.16 Voorbeeld 1: Unitary Operation**



**Figuur 5.17 Voorbeeld 2: Unitary Operation**

Een Toffoli gate is, zoals eerder vermeld, een  $C^n(U)$  gate met als eenheidsoperatie de NOT. Met bovenstaand voorbeeld (Figuur 5.16 en Figuur 5.17) is na te gaan dat de Toffoli gate met de methode van Barenco kan worden gebouwd [28,30].

## 6 Quantum Algoritmen

De vraag die nu rijst is wat we met al deze gates kunnen. Met andere woorden zijn er al bepaalde algoritmen ontwikkeld die speciaal voor een quantum computer zijn geschreven. Het antwoord hierop luidt een bescheiden ja. Er zijn inmiddels een aantal algoritmen bedacht [1,7].

Het meest bekende algoritme is wel het in 1994 door Peter Shor ontwikkelde factoring algoritme dat een getal in zijn priemfactoren kan ontbinden.

### 6.1 Shor's algoritme

Shor's algoritme is gebaseerd op een gegeven uit de Number Theory. Elke integer  $n$  heeft een unieke decompositie in priemfactoren. Deze decompositie is

bij een grote  $n$  echter een moeilijk berekenbaar probleem. De beste methode vandaag de dag bekend gebruikt een exponentiële tijd  $O(e^{c(\log n)^{1/3}(\log \log n)^{2/3}})$  [2].

De Number Theory biedt ook nog andere problemen. Bijvoorbeeld het vinden van de periode van een element. Vindt, gegeven  $x$  en  $n$ , de  $r$  (dit is de periode) zo dat  $x^r \equiv 1 \pmod{n}$ . Ook hier is er geen efficiënt algoritme bekend om het probleem op te lossen. Je zou denken wat hebben deze problemen met elkaar gemeen?

Miller heeft echter laten zien dat het factorenprobleem op te lossen is als we toegang hebben tot een orakel die de periode van een element kan vinden. Zijn reductie werkt als volgt: zorg dat  $n$  oneven en geen priemgetal is. Gebruik dan het volgende algoritme:

```
x = random({0, ..., n});
r = order(x, n); // berekent periode van x(mod n)
if (r == oneven OR xr/2 ≡ -1)
    failure;
else
    return gcd(xr/2 - 1, n); [3]
```

Alle methodes kunnen in polynomiale tijd gedaan worden, behalve het vinden van de periode [4]. Shor's doorbraak was dat hij een efficiënt quantum algoritme heeft uitgevonden om de periode van een element te bepalen. Het factorisatie algoritme is dus slechts Miller's reductie waarin het gegeven orakel vervangen is door een aanroep naar zijn quantum algoritme.

De enige exponentiële component in Miller's reductie was het vinden van de periode van  $n$ . Zolang er een methode wordt gebruikt die exponentiële tijd kost is de hele functie exponentieel. Shor heeft deze dus vervangen door een quantum computing methode die slechts polynomiale tijd kost.

Shor's algoritme kan opgebroken worden in een aantal simpele stappen:

- 1) Bepaal of  $n$  een priemgetal of een even getal is. Als dit het geval is gebruiken we het algoritme niet. Klassieke methoden volstaan hier.
- 2) Kies een getal  $q = 2^m$ , zodanig dat  $n^2 \leq q \leq 2n^2$  en een getal  $x$  copriem aan  $n$ .
- 3) Laadt een quantum register met

$$|reg1, reg2\rangle = \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a, 0\rangle.$$

Dit doen we door alle qubits in register 1 door een Hadamard gate te laten gaan. Vervolgens laten we de transformatie  $x^a \pmod{n}$  op  $reg2$  los.

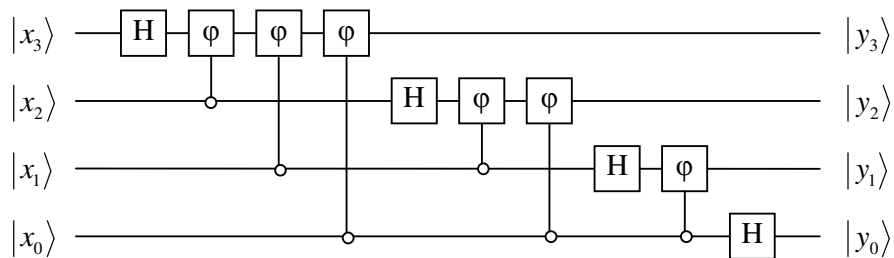
Dit doen we door middel van een unitary operatie gate. Hierna bevat het quantum register:

$$|reg1, reg2\rangle = \frac{1}{\sqrt{q}} \sum_{a=0}^{q-1} |a, x^a \bmod n\rangle$$

Meet de state van register 2. Als het resultaat van register 2  $k$  is, laat  $A = \{a' \mid x^{a'} \bmod n = k\}$  en  $\|A\|$  het aantal elementen van set  $A$  zijn, dan is de state gegeven door:

$$|reg1, reg2\rangle = \frac{1}{\sqrt{\|A\|}} \sum_{a' \in A} |a', k\rangle$$

- 4) Bereken de discrete fourier transformatie in register 1. Dit gebeurt door middel van een Discrete Fourier Transformation gate.



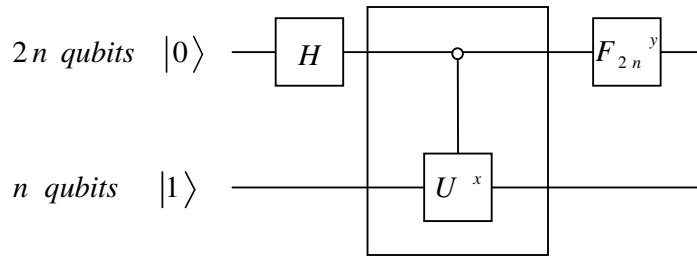
**Figuur 6.1 Discrete Fourier Transformation Gate**

Deze gate geeft de volgende output:

$$|reg1, reg2\rangle = \frac{1}{\sqrt{\|A\|}} \sum_{a' \in A} \frac{1}{\sqrt{q}} \sum_{c=0}^{q-1} e^{2\pi i a' c / q} |c, k\rangle$$

Laat het resultaat van register 1  $c'$  en de periode  $r$  zijn, deze  $c'$  is een meervoud van  $q/r$ .

- 5) Door stap 3 t/m 6  $C \log(q)$  keer te herhalen kan de exacte periode  $r$  achterhaald worden [3]. Vervolgens kan eenvoudig de factor berekend worden door middel van  $\gcd(x^{r/2} - 1), \gcd(x^{r/2} + 1)$  [3,4].



Figuur 6.2 Shor's algoritme via gates

## 6.2 Error Correctie Algoritme [QEC]

We nemen hier aan dat de quantum computer volledig geïsoleerd is. In de praktijk zal dit echter zeker niet het geval zijn. Het eerste klaarblijkelijke effect is dat de quantum computer energie verliest. Dit gebeurt met snelheid  $t_{rel}$ . Het is relatief makkelijk om een systeem te maken waarvoor  $t_{rel}$  erg klein is en dus een redelijk aantal operaties voltooid kunnen worden. Een veel verraderlijker effect van imperfecte isolatie is decoherentie. Decoherentie wordt veroorzaakt door de voortdurende interactie van de quantum computer met de omgeving. Als gevolg hiervan lekt informatie over de states van de qubits uit naar de omgeving. Zo verliezen de states hun puurheid.

Dit brengt ons bij het probleem van error correction. Allereerst is er het Non-Cloning-Theorem [5], we kunnen geen kopieën maken van qubits wat het ons moeilijk maakt om eenvoudige error correction methoden toe te passen.

Dit bewijs wordt geleverd via een contradictie. We nemen aan dat we een 'box' hebben die een willekeurige qubit als invoer heeft en als uitvoer twee gelijke qubits. Gegeven invoer  $|0\rangle$  dan is de uitvoer  $|00\rangle$  en zo levert  $|1\rangle$   $|11\rangle$  op. Gegeven de willekeurige state  $\alpha|0\rangle + \beta|1\rangle$  dan zouden we de uitvoer als twee separabele qubits willen hebben:

$$(\alpha|0\rangle + \beta|1\rangle)(\alpha|0\rangle + \beta|1\rangle)$$

Maar de lineaire superpositie van  $|0\rangle$  en  $|1\rangle$  leidt tot de volgende output:

$$\alpha|00\rangle + \beta|11\rangle$$

Alleen als  $\alpha$  of  $\beta$  0 is levert dit het gewenste resultaat. De qubits zijn *entangled*. Daarom kan onze "box" niet bestaan.

Verder kunnen we de state van een qubit ook niet lezen om te controleren of er iets is fout gegaan, dit resulteert in het vervallen van de qubit in één state. Dit in het achterhoofd houdend geven we hier een voorbeeld van: het meest simpele error correction algoritme voor bit flip errors van grootte 1.

We gebruiken een drie qubit state. Laten we aannemen dat we beginnen met:

$$|\psi\rangle = a|000\rangle + b|111\rangle$$

Vervolgens kunnen we door een bitflip error in één van de volgende states raken:

$$a|000\rangle + b|111\rangle, a|100\rangle + b|011\rangle, a|010\rangle + b|101\rangle, a|001\rangle + b|110\rangle$$

Nu willen we dus graag zorgen dat we terugkomen in de beginsituatie:

$$|\psi\rangle = a|000\rangle + b|111\rangle$$

Dit doen we door middel van een aantal operaties [6]. Allereerst voegen we 2 qubits toe aan  $|\psi\rangle$ ,  $q_4$  en  $q_5$  vormen de check-qubits. Hierbij is  $|q_4\rangle \rightarrow |q_1 \oplus q_2\rangle$  en  $|q_5\rangle \rightarrow |q_1 \oplus q_3\rangle$ . Het hele plaatje wordt dan:

$$a|00000\rangle + b|11100\rangle, a|10011\rangle + b|01111\rangle, a|01010\rangle + b|10110\rangle, \\ a|00101\rangle + b|11001\rangle$$

Het lijkt nu alsof we er niks mee opgeschoten zijn, want we hebben nu nog steeds errors. Kijken we echter naar de laatste qubits  $q_4$  en  $q_5$  zien we dat ze per state gelijk zijn. Zij kunnen ons iets vertellen over de fout die is opgetreden. Als we afgaan op qubit  $q_4$  en  $q_5$  dan kunnen we daaraan zien welke operatie uitgevoerd moet worden. Bij 00 doen we niks, bij 11 flippen we  $q_1$ , bij 10 flippen we  $q_2$  en bij 01 flippen we  $q_3$ . Dit levert weer de volgende states op, waarbij we  $q_4$  en  $q_5$  weer buiten beschouwing laten:

$$|\psi\rangle = a|000\rangle + b|111\rangle$$

Passen we dit algoritme echter toe op een willekeurig aantal bit flips dan is de kans  $3p^2(1-p) + p^3$  dat de fout niet gecorrigeerd is, dat wil zeggen kleiner dan  $p$  (zolang de kans  $p$  op een bit flip gelijk is aan  $p < 1/2$ ).

Een meer realistisch kanaal levert echter ook random phase shifts op. De volgende functie representeert een random phase shift die kan optreden.

$$P(e\phi) = \begin{pmatrix} e^{i\epsilon\phi} & 0 \\ 0 & e^{-i\epsilon\phi} \end{pmatrix} \equiv \cos(\epsilon\phi)I + i\sin(\epsilon\phi)\sigma_z$$

Waarin  $I$  de identiteitsmatrix voorstelt,  $\epsilon$  een vaste hoeveelheid die de gemiddelde grootte van de rotaties inhoudt en  $0 < \phi < 2\pi$  een willekeurige hoek.  $\sigma_z$  is een operator die een phase shift inhoudt op één qubit. We kunnen met deze situatie omgaan door gebruik te maken van het vorige algoritme, alleen zetten we aan beide kanten van het kanaal een Hadamard gate waardoor het gezamenlijke effect van de phase shift  $HPH = \cos(\epsilon\phi)I + i\sin(\epsilon\phi)\sigma_x$  wordt.  $\sigma_x$  representeert hier een bit flip. Elke qubit ondervindt nu zeker een error die een combinatie is van de identiteitsmatrix en een bit flip. De kans is nu  $1 - 3p^2$  voor kleine  $p$ , dat de fout is gecorrigeerd [6].

Ook bestaan er projectiefouten. We kunnen dit beschouwen door de geprojecteerde qubit eerst naar een ander systeem te koppelen en dan de state van het andere systeem te negeren. Laat zo'n systeem, onder zijn mogelijke states, twee states  $|a\rangle_e$  en  $|\beta\rangle_e$  die dichtbij elkaar liggen.  $\langle a|\beta\rangle = 1 - \epsilon$ . De fout bestaat in de volgende koppeling tussen een qubit en het extra systeem:

$$(a|0\rangle + b|1\rangle)|a\rangle_e \rightarrow a|0\rangle|a\rangle_e + b|1\rangle|\beta\rangle_e$$

Dit is een verwikkeling tussen het qubit en het extra systeem. Een handig inzicht is om de *entangled* state op de volgende manier uit te drukken:

$$a|0\rangle|a\rangle_e + b|1\rangle|\beta\rangle_e \equiv \frac{1}{\sqrt{2}}(a|0\rangle + b|1\rangle)|+\rangle_e + (a|0\rangle + b|1\rangle)|-\rangle_e$$

waarin  $|\pm\rangle_e = (|a\rangle_e \pm |\beta\rangle_e)/\sqrt{2}$ . Dus de fout op de qubit wordt gezien als een combinatie van de identiteit en een phase shift en is corrigeerbaar zoals eerder [6].

De meest voorkomende fout die een qubit kan ondergaan is een algemene combinatie van rotaties in de *Hilbertspace*, gecombineerd met een mogelijke projectie op een as. We verkrijgen nu een volledig willekeurige verandering van state  $|\phi\rangle$  op de volgende manier:

$$|\phi\rangle|\psi_0\rangle_e \rightarrow \sum_{i=\{I,x,y,z\}} (\sigma_i|\phi\rangle)|\psi_i\rangle_e$$

Waarin het tweede systeem, dit wordt dus over het algemeen als de interactie met de omgeving opgevat, wordt gerepresenteerd door states  $|\psi_i\rangle_e$  [6].

Het blijkt zo te zijn dat om de meest algemeen mogelijke ruis op een kanaal te corrigeren het voldoende is om alleen maar bit flips en phase shifts fouten te corrigeren. Het lijkt alsof we dan alleen maar bit flip en phase flip corrigeren, maar dat is niet waar. Het bewijs hiervoor is te vinden in [6]. Het lijkt nu alsof de indruk is gewekt dat elk mogelijke fout gecorrigeerd kan worden in een kanaal met ruis. Een realistisch kanaal produceert echter ook oncorrigeerbare fouten. De essentie om het succes van QEC (*Quantum Error Correction*) te evalueren is om de kans te berekenen dat foutcorrectie werkt en die is gelijk aan het berekenen van de kans dat het kanaal een oncorrigeerbare fout produceert mits we natuurlijk alle corrigeerbare fouten corrigeren [6].

Er is voortdurend onderzoek naar QEC codes en naar het begrijpen van de grenzen aan hun grootte. De volgende stap in QEC is om niet alleen het kanaal, maar ook alle quantum gate operaties ruis te laten bevatten. Zo nemen we de stap naar een realistischer model van een quantum computer. Het is zeker niet vanzelfsprekend of QEC onder zulke condities nog steeds zouden slagen, omdat als we een juiste set qubits op basis van een set foute *ancilla's* corrigeren introduceren we in feite meer fouten.

Methodes die een hoge kans van slagen hebben, zelfs als alle betrokken operaties onvolmaakt zijn worden *fault tolerance* genoemd [6]. Dit is mogelijk door het gebruik van simpele ideeën zoals herhaling en ook van wat subtielere ideeën die ons in staat stellen om netwerken van operaties te construeren waarbij de routes waardoor een fout kan propageren beperkt zijn. Het onderzoek aan QEC en *fault tolerance* operaties gaat onverminderd verder.

## 7 Fysieke aspecten van de Quantum Computer

### 7.1 Hoe iets kleins van groot belang kan zijn

Na alle eerder besproken aspecten van de quantum computer, wordt er nu gekeken naar de fysieke aspecten van de quantum computer. Het bouwen van een quantum computer is tot op heden nog steeds niet 'gelukt'. Echter we moeten de term 'gelukt' niet te letterlijk nemen. Zoals met elke nieuwe ontwikkeling gaan er jaren overheen tot iets wat in volgende generaties als voor lief wordt genomen, ontwikkeld is.

Vooruitstrevende ontwikkelingen komen altijd in fases. We hebben het simuleren en doorberekenen van wat een quantum computer zou moeten en kunnen doen al aardig onder de knie, ondanks de beperkte rekenkracht van onze klassieke computers. Echter legt het traject van de hardware ontwikkeling een

langere weg af. Binnen de theorie van de quantum computer, wordt er op verschillende manier gedacht over hoe er een quantum computer gebouwd moet worden. Als er wordt gesproken over het bouwen van een quantum computer, dan wordt er eigenlijk gesproken over hoe de gates die benodigd zijn om een computer te realiseren gebouwd moeten worden. Maar ook over hoe er voor moet worden gezorgd dat daadwerkelijk een qubit ontstaat, en hoe de gates moeten worden opgebouwd om bewerkingen mee uit te voeren. De basis van de quantum computer zal gevormd worden door die kleine maar cruciale aspecten, welke indien in juiste mate aanwezig, uitgevoerd en gefabriceerd, een bruikbare quantum computer zullen opleveren.

## **7.2 Niemand heeft nog gelijk, maar iedereen denkt te weten hoe het moet**

Over het realiseren van een quantum computer bestaan meerdere ideeën [17,18,19,20]. Een bewezen theorie is de praktijk, dan en slechts dan als de resulterende praktijk ook de theorie aannemelijk heeft gemaakt. De tot op heden bestaande theorieën zijn in de praktijk alleen op een kleine schaal te bewijzen. De terugkoppeling van het werkende praktijkvoorbeeld van een volwaardige quantum computer naar de theorie erachter bestaat nog niet.

Hoe een quantum computer echt gemaakt kan worden, ligt nog niet eenduidig vast, omdat alle wetenschappers denken gelijk te hebben. Wel zijn er bepaalde veronderstelde eisen waaraan een quantum computer moet voldoen op fysiek gebied. Binnen de verschillende stromingen van theorieën komen deze eisen toch overeen, namelijk:

- 1) Lange coherence tijd (i.e. de tijd waarin een superpositie behouden blijft, gebruikelijk variërend van milliseconden tot seconden)
- 2) Een quantum computer moet werken met qubits
- 3) De resultaten van een quantum computer moeten kunnen worden uitgelezen

Hieruit blijkt dat toch hetzelfde gedacht wordt over het concept van de quantum computer. De vraag is of dit gehanteerde concept, de juiste is.

## **7.3 Nuclear Magnetic Resonance Quantum Computer weet raad**

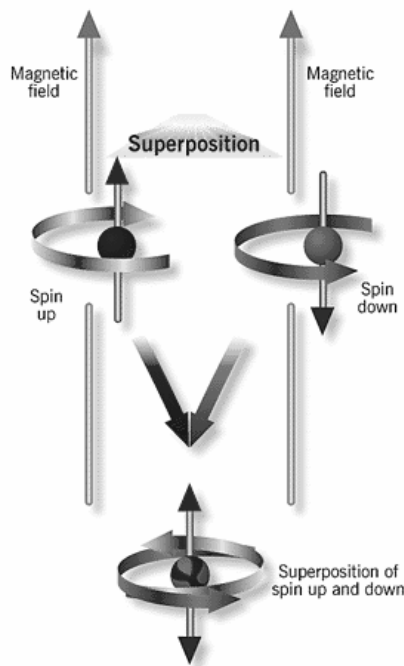
Binnen de stroming van de *Nuclear Magnetic Resonance Quantum Computer*, afgekort de NMRQC, zijn concreet uitgewerkte en geteste ideeën over de quantum computer [17,18,19]. Over deze manier van werken bestaat al

meer dan 50 jaar ervaring. Momenteel is men bezig met het bouwen van een NMRQC op basis van het *liquid state* principe. Deze is gebaseerd op de chemische omgeving waarin de NMRQC zich bevindt.

Op basis van het huidige onderzoek en gedane ontdekkingen binnen deze vorm van Quantum Computing zal er een beschrijving worden gegeven op de eerder genoemde eisen die verondersteld worden bij een quantum computer met betrekking tot de NMRQC. Wel moet erbij vermeld worden dat deze methode te weinig capaciteit gaat leveren om een groot aantal qubits te huisvesten. Daarentegen levert deze methode wel nieuwe kennis op ten aanzien van de quantum computer, waaronder de tekortkomingen van de NMRQC zelf.

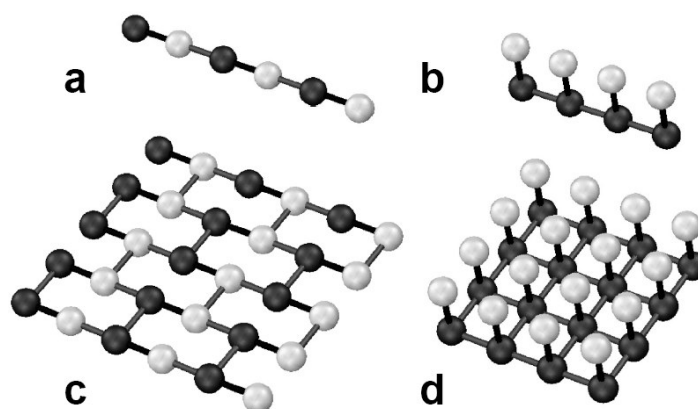
Een NMRQC gebruikt de spin van atomen binnen een molecuul als basis voor de quantum omgeving. Deze spin oftewel rotatie gaat tegen het er opstaande magnetische veld in of gaat ermee mee. Zo krijgt men de logische states 1 en 0. De spin heeft een bepaalde resonantie frequentie. Deze resonantie frequentie is per molecuul verschillend. Afhankelijk van de chemische omgeving waarin een molecuul zich bevindt zal de resonantie frequentie ook anders zijn [17].

Op deze manier is het mogelijk de spins van de atomen per molecuul apart aan te spreken. Zo kan de richting van de spin in elke willekeurige richting getikt worden, en kan elke state verkregen worden, inclusief een superpositie (zie Figuur 7.1 Spin van atomen [22]). Op deze wijze is het mogelijk om enkele qubit gates te maken.



**Figuur 7.1 Spin van atomen**

Om gates te maken die uit meerdere qubits bestaan, is het nodig dat er interactie is tussen de qubits. Dit wordt vertaald naar het fysieke deel door middel van het aan elkaar koppelen van de spins met behulp van de gemeenschappelijke elektronen. Zo ontstaat een chain of grid netwerk (al dan niet multidimensionaal [23]) van gekoppelde spins (zie Figuur 7.2 Netwerken van atomen). Deze geschakelde spins samen kunnen dan de states bereiken die in meervoudige qubit gates te bewerkstelligen zijn. Ook dit zal dan gaan door de resonantie frequentie zo te bepalen dat alle spins zich draaien naar de bepaalde states van de qubits. Aan het concept van de gates is al eerder aandacht besteed.



**Figuur 7.2 Netwerken van atomen**

- a. Horizontaal, 1 dimensionaal
- b. Verticaal, 1 dimensionaal
- c. Horizontaal, 2 dimensionaal
- d. Verticaal, 2 dimensionaal

Een ander probleem dat zich voordoet bij de NMRQC is de decoherence van de spin, oftewel het verlies van de nauwkeurigheid van de superstate. Beïnvloed door de omgeving zal de staat waarin de spin zich bevindt gaan schommelen. Hier kan gedacht worden aan de fluctuaties in de spin van het atoom, die naarmate de omgeving meer invloed krijgt en neemt, steeds meer gaan afwijken. Het is daarom ook belangrijk om de tijd waarin dit fenomeen gaat plaats vinden, zo lang mogelijk te maken. Ook bij het uitlezen van de state waarin een spin zich bevindt zal er dus invloed van buitenaf zijn. Ook dit zorgt voor fluctuaties in de spin. Een manier om dit binnen de NMRQC op te lossen is het gebruik van de error correcting algoritmes, zie hiervoor paragraaf 6.

De voorstelling en opstelling van de NMRQC voor een klein aantal bewerkingen op qubits ligt niet binnen praktisch handbereik. Zeker niet als het

gaat om grote voltallige bewerkingen. Deze totale manier van werken, Nuclear Magnetic Resonance, zal later niet de toekomst zijn. Wel zal deze mede de toekomst bepalen, want met betrekking tot onderzoek op gebied van de quantum computer op zich, zal elke stap voorwaarts, een stap in de goede richting zijn.

## ***7.4 De toekomst zal bepalen waar nu naar gezocht wordt***

Naast de besproken theorie van de quantum computer, de NMRQC, zijn er nog verscheidene manieren om een quantum computer te realiseren.

### **7.4.1 All Silicon Quantum Computer**

Deze manier van werken brengt een bekend concept met zich mee, die van de computer op basis van siliconen. Dit concept plaatst individuele atomen in een siliconen omgeving, en stuurt deze elektronisch. Dit is een manier die op de huidige bekende technieken gebaseerd is [17].

### **7.4.2 Ion Trap**

De qubits worden hier opgeslagen in ionen. Er wordt hier ook gebruik gemaakt van de spin. Tevens spelen hier fotonen ook een rol. Hier is de qubit het atoom en foton. De ionen worden hier gekoeld tot bijna het absolute nulpunt, 0 Kelvin. In deze staat bewegen de ionen niet meer en is bewerking mogelijk [17,20].

### **7.4.3 Cavity QED**

Het principe hier is op basis van het atoom, het foton en de interactie tussen beide. Er wordt gebruik gemaakt voor een laser voor het opslaan en bewerken van een qubit. Het qubit is hier een enkele foton. [17,19,20]

### **7.4.4 Tot slot**

Bovenstaande manieren zijn niet uitgebreid behandeld vanwege de extra natuur- en scheikundige achtergrond die vereist is om deze manieren van werken te kunnen behappen. Ook bij de NMRQC, zit er achter de resonantie frequentie en de benodigde apparatuur om deze afstemmingsresonantie te

krijgen, grote delen wis- en natuurkunde, die verder gaat dan eerder behandelt. Tevens is er bij de NMRQC sprake van een langere ervaring met betrekking tot deze theorie.

Dit neemt niet weg dat wat al bepaald lijkt te zijn met betrekking tot de concepten aangaande de fysieke aspecten van de quantum computer, niet meer is dan een optelling van de huidige stand van zaken. Welke betrekking heeft op alle vlakken van de natuur-, schei-, en wiskunde, voortbordurend op de huidige stand van zaken van kennis binnen elk van de genoemde vakgebieden. Daarom geldt ook binnen het onderzoek van het fysieke deel van de quantum computer: wat niet is, kan nog komen én wat al is, zal vergaan.

## 8 Referenties

- [1] **Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer**  
Peter W. Shor  
January, 1996  
<http://www.arxiv.org/abs/quant-ph/9508027>
  
- [2] **The Development of the Number Field Sieve.**  
A.K. Lenstra and H.W. Lenstra.  
Springer Verlag, 1993  
LNCS 1554
  
- [3] **Quantum Complexity Theory II**  
Christian Scheideler  
Modern Complexity Theory  
Spring 2003  
[http://www.cs.jhu.edu/~scheideler/courses/600.471\\_S03/lecture\\_11.pdf](http://www.cs.jhu.edu/~scheideler/courses/600.471_S03/lecture_11.pdf)
  
- [4] **The Art of Computer Programming**  
D.E. Knuth  
Vol. 2, Addison Wesley, 1981
  
- [5] **Notes on Quantum Computing and Related Topics**  
Professor Douglas Ross en Professor Tony Hey  
<http://www.qtc.ecs.soton.ac.uk/lecture2/p5.html>
  
- [6] **Quantum Computing and Error Correction**  
Andrew M. Steane  
Department of Physics, University of Oxford,  
November 8, 2000  
[http://xxx.lanl.gov/PS\\_cache/quant-ph/pdf/0304/0304016.pdf](http://xxx.lanl.gov/PS_cache/quant-ph/pdf/0304/0304016.pdf)
  
- [7] **A fast quantum mechanical algorithm for database search**  
Lov K. Grover  
Bell Labs, Murray Hill NJ  
Mei 1996  
<http://arxiv.org/abs/quant-ph/9605043>
  
- [8] **Timeline of quantum computing**  
Wikipedia, the free encyclopedia  
[http://en.wikipedia.org/wiki/Timeline\\_of\\_quantum\\_computing](http://en.wikipedia.org/wiki/Timeline_of_quantum_computing)

- [9] **Quantum theory, the Church-Turing principle and the universal Quantum computer**  
David Deutsch,  
Proceedings of the Royal Society of London,  
1985,  
<http://www.qubit.org/oldsite/resource/deutsch85.pdf>
- [10] **Algorithms for quantum computation: Discrete logarithms and factoring**  
Peter W. Shor,  
Proc. 35nd Annual Symposium on Foundations of Computer Science  
(Shafi Goldwasser, ed.), IEEE Computer Society Press  
1994
- [11] **Scheme for reducing decoherence in quantum computer memory**  
Peter W. Shor,  
Phys. Rev. A **52**,  
1995
- [12] **Quantum Computation**  
Center for Numerical Simulation & Modeling  
<http://www2.latech.edu/~dgao/CNSM/quantumcomput.html>
- [13] **Experimental Implementation of Fast Quantum Searching**  
Isaac L. Chuang<sup>1</sup>, Neil Gershenfeld<sup>2</sup>, and Mark Kubinec,  
Physical Review Letters  
1998,  
<http://www.media.mit.edu/physics/publications/papers/98.03.grover.pdf>
- [14] **IBM-Led Team Unveils Most-Advanced Quantum Computer**  
15 Augustus 2000  
[http://www.research.ibm.com/resources/news/20000815\\_quantum.shtml](http://www.research.ibm.com/resources/news/20000815_quantum.shtml)
- [15] **Experimental Realization of an Order-Finding Algorithm with an NMR Quantum Computer**  
Lieven M.K. Vandersypen, Matthias Steffen, Gregory Breyta,  
Costantino S. Yannoni, Richard Cleve, and Isaac L. Chuang  
PHYSICAL REVIEW LETTERS, Vol. 85, Nr. 25  
18 December, 2000  
[http://luciano.stanford.edu/~lieven/papers/PRL\\_order.pdf](http://luciano.stanford.edu/~lieven/papers/PRL_order.pdf)

- [16] **IBM's Test-Tube Quantum Computer Makes History**  
SAN JOSE, California,  
19 December, 2001,  
[http://www.research.ibm.com/resources/news/20011219\\_quantum.shtml](http://www.research.ibm.com/resources/news/20011219_quantum.shtml)
- [17] **How to Build Your Own Quantum Computer – Lecture 19**  
Isaac Chuang  
Department of Mathematics, MIT  
13 November 2003  
[http://www-math.mit.edu/~chr/18.435/qc\\_lec19.pdf](http://www-math.mit.edu/~chr/18.435/qc_lec19.pdf)
- [18] **Nuclear Magnetic Resonance Quantum Computing**  
Matthias Steffen  
Quanta Group - MIT Media Lab  
Lopend Project  
<http://www.media.mit.edu/quanta/people/msteffen/nmrqc.htm>
- [19] **What is a Quantum Computer?**  
10 Oktober 2003  
iiRobotics Ltd.  
<http://www.iirobotics.com/webpages/hotstuff.php?ubre=89>
- [20] **Quantum Computing Survey**  
Clare Warwick  
11 Mei 1997  
<http://www.banished.demon.co.uk/quantum/Building.htm>
- [21] **Spin doctors**  
Justin Mullins  
New Scientist  
24 June 2000  
<http://www.newscientist.com/hottopics/quantum/quantum.jsp?id=22444700>
- [22] **Macroscopic/Microscopic World Interface – Lecture 14**  
All Schombert  
21st Century Science  
30 April 2003  
[http://zebu.uoregon.edu/~js/21st\\_century\\_science/lectures/lec14.html](http://zebu.uoregon.edu/~js/21st_century_science/lectures/lec14.html)

- [23] **Universal quantum computation with spin-1/2 pairs and Heisenberg exchange**  
Jeremy Levy  
Center for Oxide-Semiconductor Materials for Quantum Computation,  
and Department of Physics and Astronomy, University of Pittsburgh  
Januari 2001  
<http://arxiv.org/ftp/quant-ph/papers/0101/0101057.pdf>
- [24] **Reversible logic and quantum computers**  
Asher Peres  
Department of Physics, Technion - Israel Institute of Technology, 32000  
Haifa, Israel  
Maart 1985  
[http://prola.aps.org/pdf/PRA/v32/i6/p3266\\_1](http://prola.aps.org/pdf/PRA/v32/i6/p3266_1)
- [25] **Quantum Logic Circuits**  
John P. Hayes  
EECS Department, University of Michigan.  
Oktober 2001  
<http://vlsicad.eecs.umich.edu/Quantum/EECS598/lec/5.ppt>
- [26] **Basic Quantum Operations**  
Hendrik Weimer  
libquantum - Simulation of a quantum computer with a C library  
<http://www.enyo.de/libquantum/api/node4.html>
- [27] **Unitary Evolution, No Cloning Theorem, Superdense Coding**  
Berkely University, computer science department  
September 2003  
<http://www-inst.eecs.berkeley.edu/~cs191/lectures/lecture4.pdf>
- [28] **Quantum computation**  
J. Acebron  
2002/2003  
<http://www.dei.unipd.it/ricerca/cesp/academic/co/Acebron.pdf>
- [29] **Quantum Circuits**  
Jon Marshall  
June 2000  
<http://www.themilkyway.com/quantum/FinalReport/QuantumGates.html>

[30] **A Universal Two–Bit Gate for Quantum Computation**

Adriano Barenco

Clarendon Laboratory, Physics Department, University of Oxford, Parks  
Road, Oxford OX1 3PU, United Kingdom

Juni 1995

[http://arxiv.org/PS\\_cache/quant-ph/pdf/9505/9505016.pdf](http://arxiv.org/PS_cache/quant-ph/pdf/9505/9505016.pdf)