FPGA-Based Circuit Model Emulation of Quantum Algorithms

Mahdi Aminian, Mehdi Saeedi, Morteza Saheb Zamani, Mehdi Sedighi

Quantum Design Automation Lab

Department of Computer Engineering & Information Technology

Amirkabir University of Technology

Outline

- Background
- Motivation
- Quantum Gates
- Emulation Method
- Experimental Results
- Conclusion

Background

- *State of qubit* represent as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \text{ (superposition state)}$$

- ‘α’ and ‘β’ are complex numbers and

$$||\alpha||^2 + ||\beta||^2 = 1$$

- Not possible to find the exact value of an unknown qubit using a measurement operator
- Upon measurement its state collapses to $|0\rangle$ or $|1\rangle$ with the probability of $||\alpha||^2$ and $||\beta||^2$
- A quantum system of size *N* can be constructed using the tensor product
- The property of working on multiple input states simultaneously leads to a significant parallelism in quantum algorithms

Motivation

- Several problems that cannot be executed on a classical machine as efficiently as a quantum computer
- Due to the lack of an existing quantum computer, simulating quantum algorithms on a classical computer is widely used
- Using simulation to verify the functionalities of quantum algorithms
- software simulation cannot profit the intrinsic parallelism of quantum algorithms completely
- using the parallel nature of hardware architectures

Quantum Gates

$$\alpha|0\rangle + \beta|1\rangle \quad \text{———}\boxed{X}\text{———} \quad \beta|0\rangle + \alpha|1\rangle$$

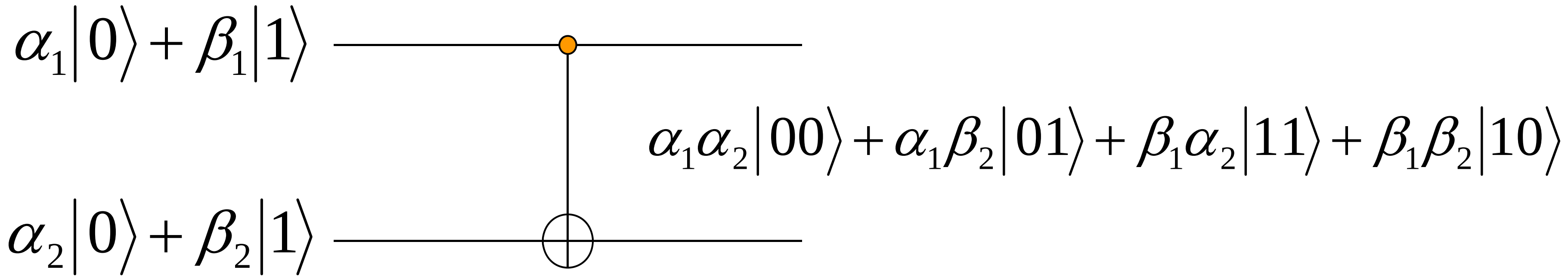
$$\alpha|0\rangle + \beta|1\rangle \quad \text{———}\boxed{Y}\text{———} \quad -i\beta|0\rangle + i\alpha|1\rangle$$

$$\alpha|0\rangle + \beta|1\rangle \quad \text{———}\boxed{Z}\text{———} \quad \alpha|0\rangle - \beta|1\rangle$$

$$\alpha|0\rangle + \beta|1\rangle \quad \text{———}\boxed{H}\text{———} \quad \frac{\alpha+\beta}{\sqrt{2}}|0\rangle + \frac{\alpha-\beta}{\sqrt{2}}|1\rangle$$

$$\alpha|0\rangle + \beta|1\rangle \quad \text{———}\boxed{PS}\text{———} \quad \alpha|0\rangle + e^{i\phi}\beta|1\rangle$$

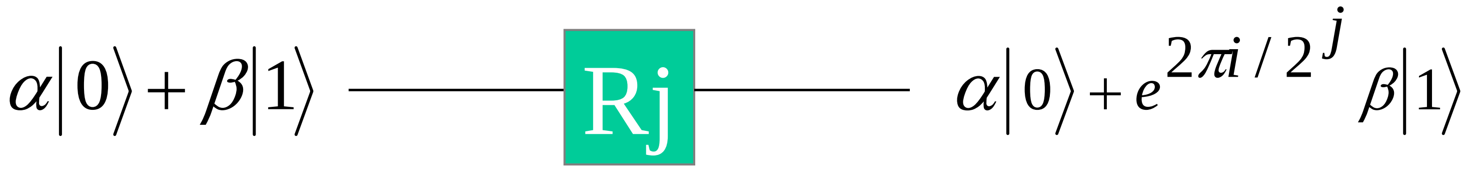
Quantum Gates (cont.)



$$\alpha_1|0\rangle+\beta_1|1\rangle$$

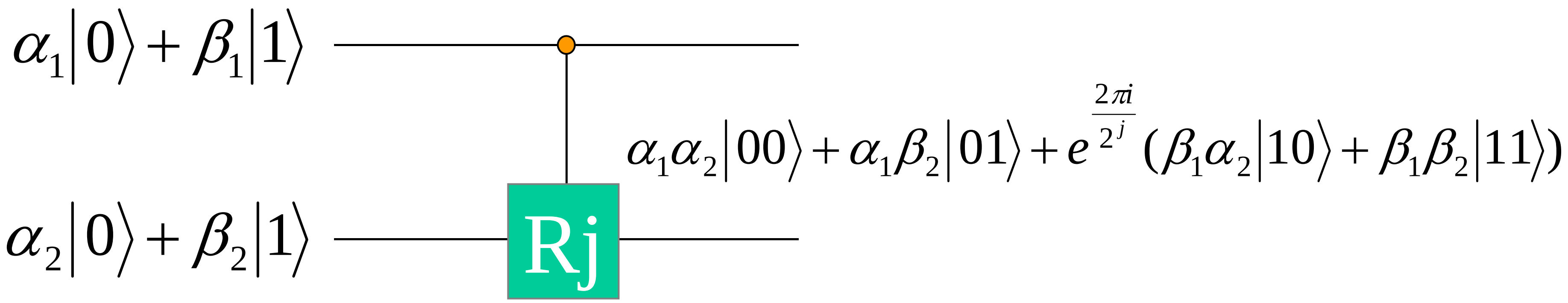
$$\alpha_2|0\rangle+\beta_2|1\rangle$$

$$\alpha_1\alpha_2|00\rangle+\alpha_1\beta_2|01\rangle+\beta_1\alpha_2|11\rangle+\beta_1\beta_2|10\rangle$$



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

$$\alpha|0\rangle+e^{2\pi i/2^j}\beta|1\rangle$$



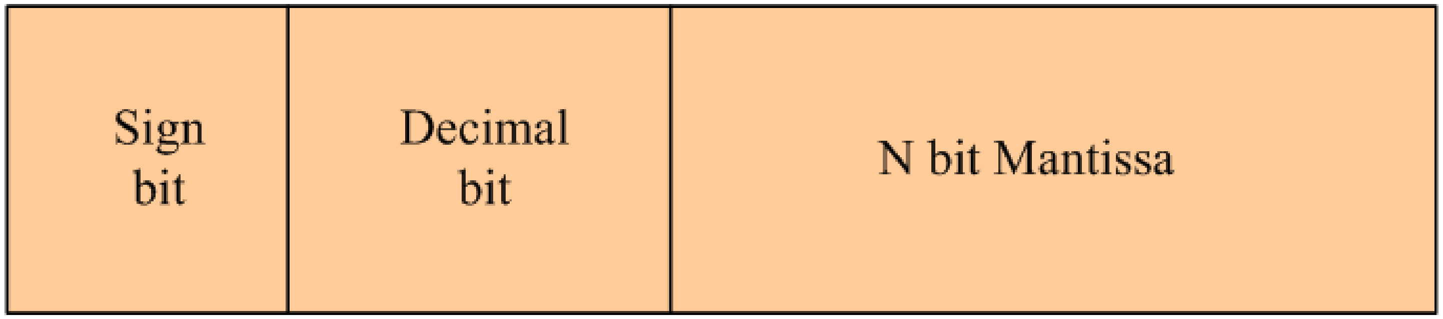
$$\alpha_1|0\rangle+\beta_1|1\rangle$$

$$\alpha_2|0\rangle+\beta_2|1\rangle$$

$$\alpha_1\alpha_2|00\rangle+\alpha_1\beta_2|01\rangle+e^{\frac{2\pi i}{2^j}}(\beta_1\alpha_2|10\rangle+\beta_1\beta_2|11\rangle)$$

Emulation Method

- Fixed-point numbers for represent complex coefficients of each qubit



- each qubit can be represented by four fixed-point numbers

Emulation Method (cont.)

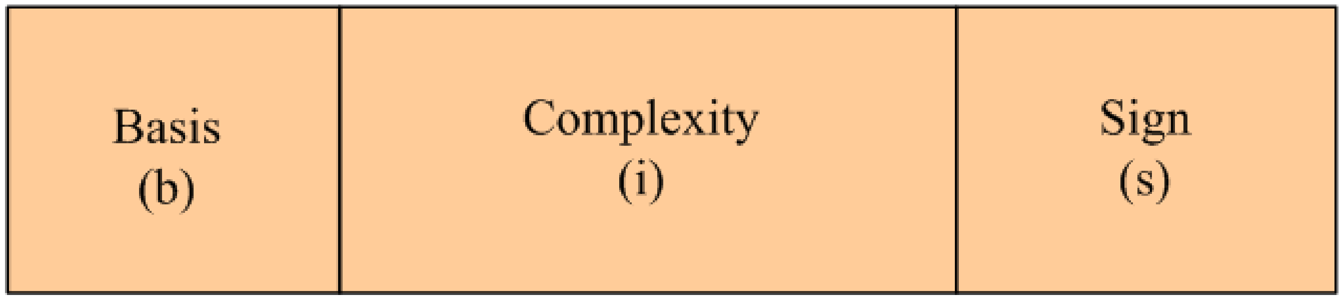
- Two group of gates
  - HRC group
    - Hadamard gate (H)
    - Phase-Shift gate (PS)
    - Rotate gate (R)
    - Controlled-Rotate gate (CR)
  - XYZC group
    - X gate (NOT)
    - Y gate
    - Z gate
    - Controlled-NOT gate (CNOT)

Emulation of HRC group

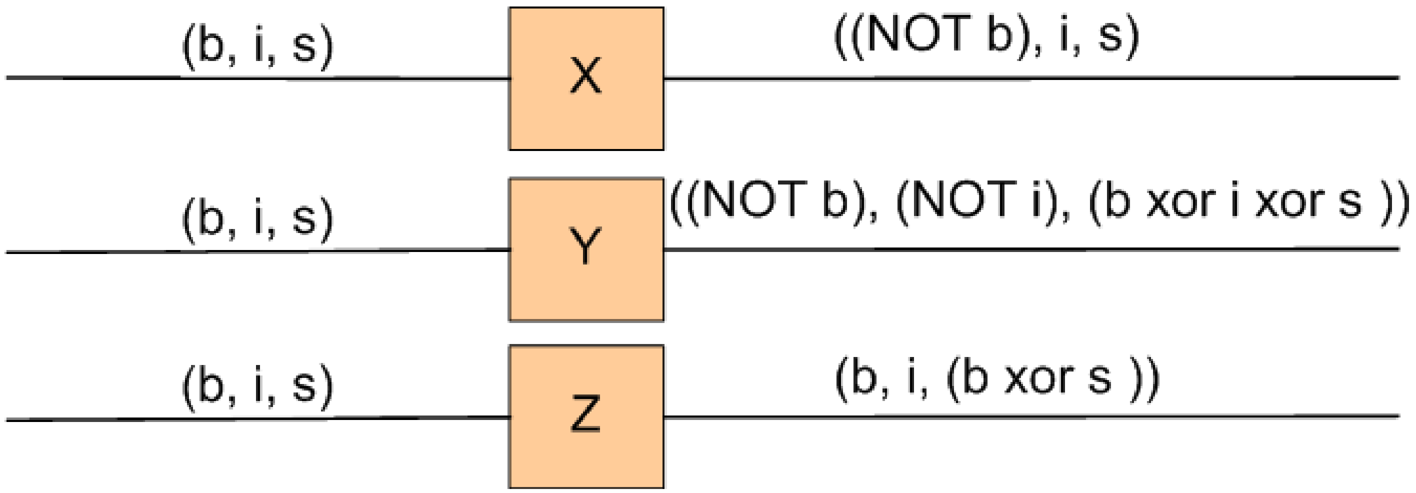
- *H* gate needs four multiplications and four additions on fixed-point numbers
- *PS* and *R* gates needs four multiplications and two additions
- *CR* gate rotating the target qubit if the control line is equal to 1
- *CR* gate produces an entanglement state
- In entanglement state, more resources are needed for simulating

Emulation of XYZC group

- Using three extra bits added to each qubit coefficient



- The X, Y, Z gates are efficiently manipulated as below



Emulation of XYZC group (cont.)

- In entangled state, more extra bits added to each coefficient

Basis ($b_n \ldots b_1$)	Complexity (i)	Sign (s)

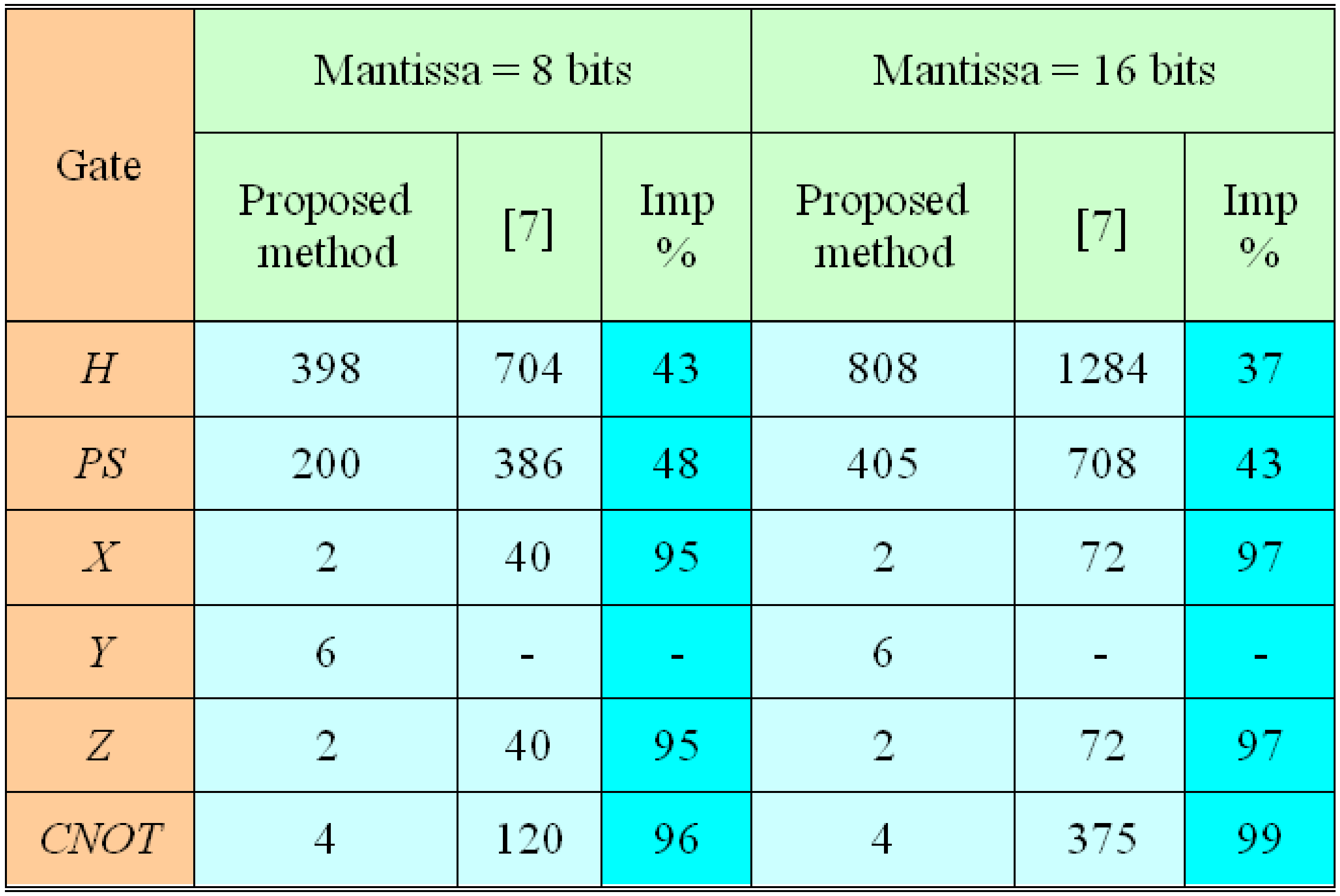
- ‘*n*’ is the number of qubits
- *CNOT* gate is implemented by XORing the control and target bits

Emulation of XYZC group (cont.)

- When both XYZC and HRC gates are used, method is changed
- gate operations are implemented by a coefficient swapping operator using intermediate registers as below:
  - *X* gate swaps the complex coefficients
  - *Y* gate swaps coefficients and multiplying the complex number i (or –i)
  - *Z* gate implemented by multiplying -1 to the complex coefficient *β*
  - *CNOT* gate swaps appreciate coefficients in entangled state

Results: Implementation of each gate

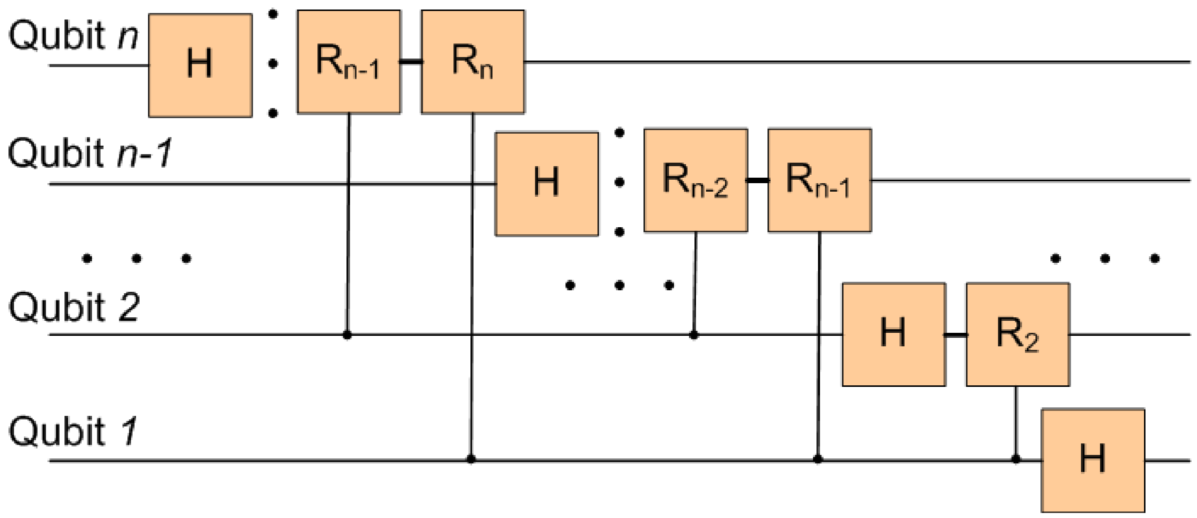
- Implementation with VHDL
- Using *ALTERA STRATIX EP1S80B956C6* device for synthesis



Gate	Mantissa = 8 bits			Mantissa = 16 bits		
	Proposed method	[7]	Imp %	Proposed method	[7]	Imp %
H	398	704	43	808	1284	37
PS	200	386	48	405	708	43
X	2	40	95	2	72	97
Y	6	-	-	6	-	-
Z	2	40	95	2	72	97
CNOT	4	120	96	4	375	99

Results: Quantum Fourier Transform (QFT)

- QFT circuit



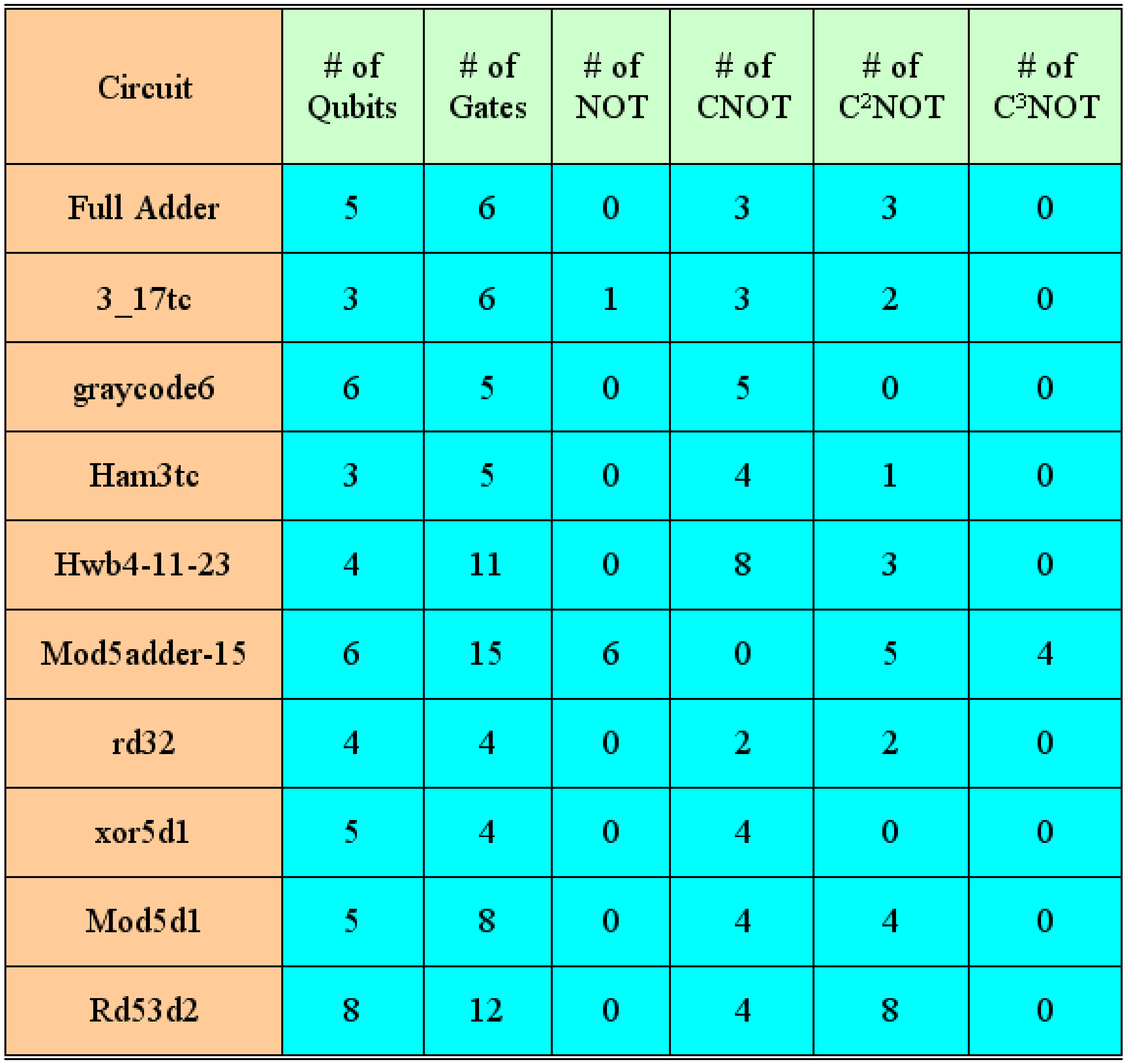
- synthesis results and run time for a 3-input QFT algorithm

Mantissa	LC Usage	Clock Frequency (MHz)	
		Proposed method	[7]
8	**3905**	**137.9**	-
16	**8197**	**131.3**	**82.1**

Mantissa (bits)	Run Time (Seconds)	
	Libquantum [17]	Proposed Method
16	$\mathbf{40 \times 10^{-6}}$	$\mathbf{46 \times 10^{-9}}$

Results: Implementation of Benchmark

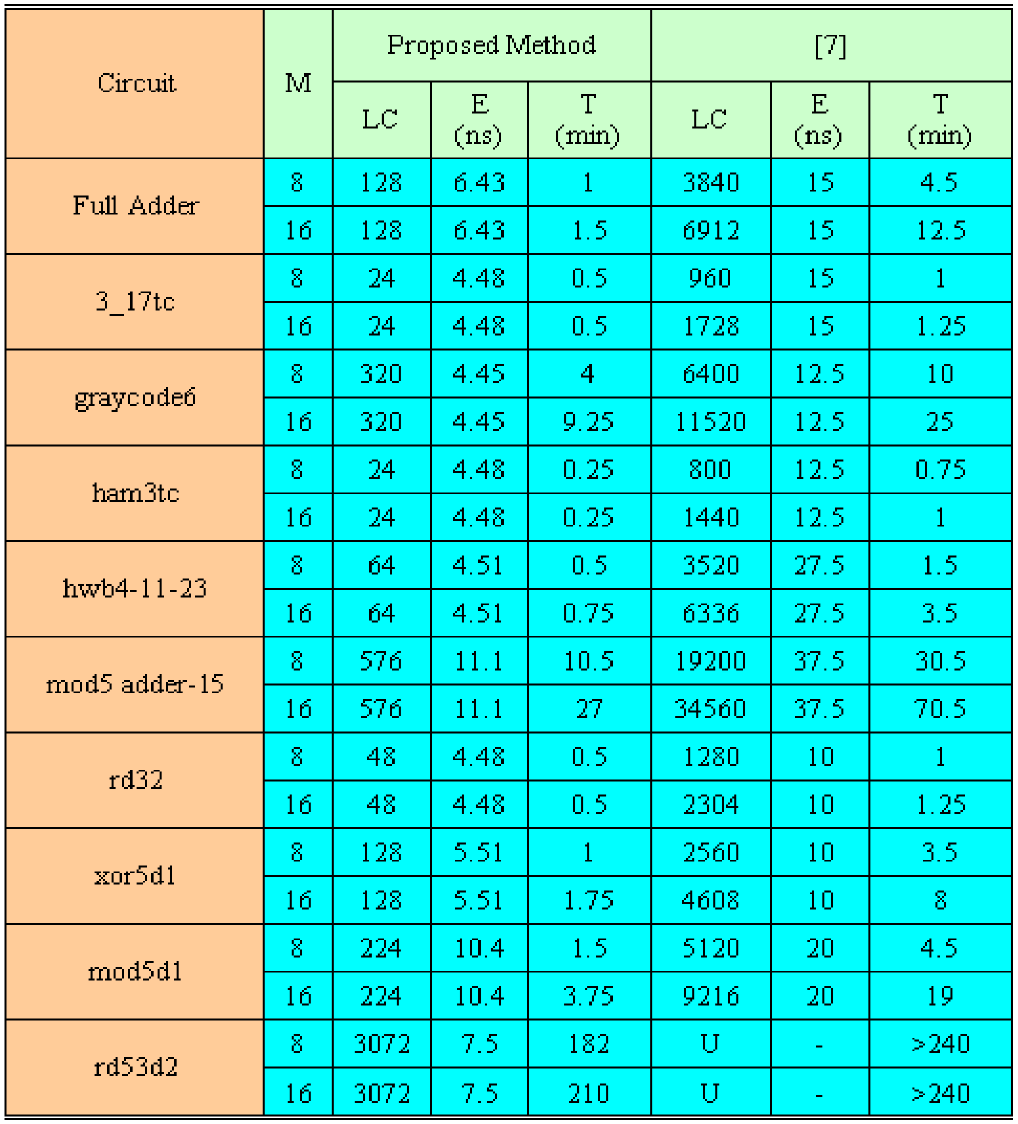
- **Specification of benchmarks**



Circuit	# of Qubits	# of Gates	# of NOT	# of CNOT	# of C^2NOT	# of C^3NOT
Full Adder	5	6	0	3	3	0
3_17tc	3	6	1	3	2	0
graycode6	6	5	0	5	0	0
Ham3tc	3	5	0	4	1	0
Hwb4-11-23	4	11	0	8	3	0
Mod5adder-15	6	15	6	0	5	4
rd32	4	4	0	2	2	0
xor5d1	5	4	0	4	0	0
Mod5d1	5	8	0	4	4	0
Rd53d2	8	12	0	4	8	0

Results: Implementation of Benchmark (cont.)

- **LC usage for synthesis of benchmarks**



Circuit	M	Proposed Method			[7]		
		LC	E (ns)	T (min)	LC	E (ns)	T (min)
Full Adder	8	128	6.43	1	3840	15	4.5
	16	128	6.43	1.5	6912	15	12.5
3_17tc	8	24	4.48	0.5	960	15	1
	16	24	4.48	0.5	1728	15	1.25
graycode6	8	320	4.45	4	6400	12.5	10
	16	320	4.45	9.25	11520	12.5	25
ham3tc	8	24	4.48	0.25	800	12.5	0.75
	16	24	4.48	0.25	1440	12.5	1
hwb4-11-23	8	64	4.51	0.5	3520	27.5	1.5
	16	64	4.51	0.75	6336	27.5	3.5
mod5 adder-15	8	576	11.1	10.5	19200	37.5	30.5
	16	576	11.1	27	34560	37.5	70.5
rd32	8	48	4.48	0.5	1280	10	1
	16	48	4.48	0.5	2304	10	1.25
xor5d1	8	128	5.51	1	2560	10	3.5
	16	128	5.51	1.75	4608	10	8
mod5d1	8	224	10.4	1.5	5120	20	4.5
	16	224	10.4	3.75	9216	20	19
rd53d2	8	3072	7.5	182	U	-	>240
	16	3072	7.5	210	U	-	>240

Conclusion

- Proposed an efficient quantum circuit emulation technique
- This method uses the parallelism of quantum algorithms
- Offers more efficiency than software simulation
- uses fewer logic cells compared with the available hardware emulation methods
- Using a novel representation schema
- Emulating the behaviors of various quantum gates

Questions?