PORTLAND QUANTUM

LOGIC GROUP
Reversible Logic

- As proved by Landauer, using traditional *irreversible gates* leads to *energy dissipation*.
- Bennett showed that for power not to be dissipated in the circuit it is necessary that arbitrary circuit can be built from *reversible gates*. 
Reversible logic

Reversible are circuits (gates) that have the same number of inputs and outputs and have one-to-one mapping between vectors of inputs and outputs; thus the vector of input states can be always reconstructed from the vector of output states.
Quantum versus reversible computing

- **Quantum Computing** is a coming revolution – after recent demonstrations of quantum computers, there is no doubt about this fact. They are reversible.
- Top world universities, companies and government institutions are in a race.
- **Reversible computing** is the step-by-step way of scaling current computer technologies and is the path to future computing technologies, which all happen to use reversible logic.
- In addition, Reversible Computing will become mandatory because of the necessity to decrease power consumption.
But what can be done by researchers who have no access to well-funded laboratories, expensive equipment, and elite workshops?

- Portland Quantum Logic Group is an attempt to give an answer to the above question.
- We are a group of researchers interested in collaboration on reversible logic, quantum logic, reversible and quantum computing and nano-technologies.
- We share information about new developments, we collaborate on developing new methodologies, algorithms and circuits.
- We discuss theoretical problems and realization issues.
Portland Quantum Logic Group members include:

M. Perkowski, G. Greenwood, A. Al-Rabadi, A. Mishchenko, X. Song and M. Chrzanowska-Jeske from Department of Electrical and Computer Engineering, PSU,

B. Massey and S. Mocas from Department of Computer Science of PSU,

Andrew Fraser from System Science, PSU,

Pawel Kerntopf from Department of Computer Science, Technical University of Warsaw, Warsaw, Poland,

Svetlana Yanushkevich and Vlad Shmerko from Department of Electronics of Technical University of Szczecin, Poland,

Andrzej Buller from Information Science Division, Advanced Telecommunications Research Institute International (ATR), Kyoto, Japan,

Alan Coppola from Cypress Semiconductor Northwest and Oregon Graduate Institute,

Md. Mozammel Huq Azad Khan from Department of Computer Science and Engineering, East West University, Bangladesh,

Bernd Steinbach from Computer Science Department, Technical University of Freiberg, Germany,


Radomir Stankovic from University of Nis, Yugoslavia.
Our general goals are:

1. to develop efficient logic synthesis algorithms for reversible logic, and comprehensive theory of reversible logic synthesis, test and verification.

2. to implement reversible Field Programmable Gate Arrays (FPGAs) in CMOS technology for extremely low power, high testability and self-repair.

3. to create new theoretical algorithms for quantum computing.

4. to help one another to answer difficult research questions.

5. to work together on writing reports, journal and conference papers and software related to the above mentioned areas.

6. to compile a comprehensive list of relevant literature positions and Internet links.

7. to attend conferences and meetings and report about newest research in the world in the areas of reversible and quantum computing.
Our current work includes:

1. Designing circuits in reversible technologies
2. Designing gates in new reversible technologies.
3. Developing logic, system and layout methodologies for reversible and quantum logic (every quantum logic is reversible).
4. Designing systems (such as microprocessors) using reversible logic.
5. Developing highly efficient algorithms and computer programs for logic synthesis in reversible logic.
Our current work includes:

6. Developing test algorithms and design for test methods for reversible technologies.

7. Developing formal verification and validation methods for reversible systems.

8. Developing concepts of multiple-valued, fuzzy and continuous reversible logic.

9. Developing highly efficient data representations for reversible logic.
Background
Reversible logic

- A gate with $k$ inputs and $k$ outputs is called a $k*k$ gate.
- A conservative circuit preserves the number of logic values in all combinations.
- In balanced binary logic the circuit has half of minterms with value 1.
- In ternary logic one third of minterms have value 0, one third value 1 and one third has value 2 – this is preserved from inputs to outputs.
Reversible logic

- A circuit without constants on inputs which includes only reversible gates realizes on all outputs only balanced functions, therefore it can realize non-balanced functions only with garbage outputs.

- There are reversible circuits (such as Fredkin gate) that are conservative, but not all such circuits are conservative.

- Not all conservative gates are reversible. In reversible logic, the fanout of every signal, including primary inputs is one.

- The goal of this paper is to develop a method to build single-output reversible logic circuits in regular and non-regular structures, in binary and ternary logic, with minimized number of garbage output signals and gates.
The 2*2 Feynman gate, called also controlled-not or quantum XOR realizes functions \( P = A, \ Q = A \oplus B \), where operator \( \oplus \) denotes EXOR.

- When \( A = 0 \) then \( Q = B \), when \( A = 1 \) then \( Q = \neg B \).
- With \( B=0 \) Feynman gate is used as a fan-out gate.
- Every linear reversible function can be built by composing only 2*2 Feynman gates and inverters.
- With \( B=0 \) Feynman gate is used as a fan-out gate.
These three boolean expressions are identical, but lead to different physical realizations.

We note, however, that these implementations not only make use of the 1-bit NOT function and the 2-bit CONTROLLED NOT function, but also of the 2-bit exchanger, i.e. the gate that interchanges two logic variables (realizing \( P = B \) as well as \( Q = A \)).

This is an example of a general property: the EXCHANGER, the NOT, and the CONTROLLED NOT form a natural `generating set' for the twenty-four 2-bit reversible gates.

More precisely, each reversible 2-bit gate can be synthesized by taking one or zero CONTROLLED NOTs and adding one or zero EXCHANGERs and one or zero NOTs to the left and to the right of it.

Table 4: The three generators of the 2-bit reversible gates:
(a) EXCHANGER, (b) NOT, (c) CONTROLLED NOT.

Note that:
- neither the EXCHANGER nor the NOT is a true 2-bit gate,
- but the CONTROLLED NOT is one.

See Table 4
Fredkin Gate is a fundamental concept in reversible and quantum computing.

Every Boolean function can be build from 3 * 3 Fredkin gates:

\[
\begin{align*}
P &= A, \\
Q &= \text{if } A \text{ then } C \text{ else } B, \\
R &= \text{if } A \text{ then } B \text{ else } C.
\end{align*}
\]
The universality of sets of $3 \times 3$ conservative gates with respect to all $3 \times 3$ reversible gates was considered by Sasao and Kinoshita. They also proved that an arbitrary $n$-variable function can be implemented by $(n+3) \times (n+3)$ circuits using Fredkin gates but their approach leads to long (thus slow and expensive) cascades of gates.
• The 3 * 3 Toffoli gate is described by these equations:
  \[ P = A, \]
  \[ Q = B, \]
  \[ R = AB \oplus C, \]
• Toffoli gate is an example of \textit{two-through} gates, because two of its inputs are given to the output.
The Kerntopf gate is described by equations:

\[ P = 1 \oplus A \oplus B \oplus C \oplus AB, \]
\[ Q = 1 \oplus AB \oplus B \oplus C \oplus BC, \]
\[ R = 1 \oplus A \oplus B \oplus AC. \]

When \( C = 1 \) then \( P = A + B, \) \( Q = A \times B, \) \( R = !B, \)
so AND/OR gate is realized on outputs \( P \) and \( Q \) with \( C \) as the controlling input value.

When \( C = 0 \) then \( P = !A \times !B, \) \( Q = A + !B, \) \( R = A \oplus B. \)
Therefore for control input value 0 the gate realizes NAND and IMPLICATION on outputs $P$ and $Q$.

As we see, the 3*3 Kerntopf gate is not a one-through nor a two-through gate.

Despite theoretical advantages of Kerntopf gate over classical Fredkin and Toffoli gates, so far there are no published results on optical or CMOS realizations of this gate.
Multi-valued Fredkin Gate

- MVFG is described by equations:

  \[
  \begin{align*}
  P &= A \\
  Q &= B \\
  R &= C \text{ if } A < B \text{ else } R = D \\
  S &= D \text{ if } A < B \text{ else } S = C
  \end{align*}
  \]
Research on regular structures:
Our goals
The fundamental role of Shannon Expansion in binary logic is well known.

Shannon Expansion corresponds to a multiplexer gate. In reversible logic, an equivalent of multiplexer is the Fredkin gate.

- In this paper an equivalent of Shannon Expansion for reversible logic is created for the first time.
- We show its multiple-valued generalization using a new MV generalization of the Fredkin gate.
We introduce regular structures, both planar and three-dimensional, to realize arbitrary functions in binary and multiple-valued reversible logic, respectively.

- A problem in realizing functions in reversible logic is an excessive number of “garbage” output signals from gates.
- Our method effectively overcomes this problem.
- The method can be used in classical and quantum logic as well.
Levelized Structures
Standard Lattice Diagrams for continuous, multiple-valued and binary logic
Lattice Structure for Multiple-valued and Binary Logic

- Realizes every binary symmetric function
- Realizes every non-symmetric function by repeating variables
- Realizes piece-wise linear multivalued functions

Patented by Pierzchala and Perkowski 1994/1999
Lattice Structure for Multiple-valued and Binary Logic

- Cell has three inputs and two outputs
- Both outputs have the same function

Multi-valued output variable

Multiple-valued variables

Binary input variables

- Red nose
- Beard
- Red eyes

Kowalski
Al-Rabadi
Piotrowski
Perkowski
Lattice Structure for Multiple-valued and Binary Logic

- Redness of nose in interval [3,4]
- Length of beard an odd number
- Redness of eyes in intervals [2,4] or [7,9]
- Kowalski
- Al-Rabadi
- Piotrowski
- Perkowski
Lattice Structure for Multivalued and Binary Logic

Multivalued input variables

\[ A > B \]

\[ C < D \]

\[ E = G \]

Multi-valued output variable

\[ 0 \quad 1 \]

binary

Multivalued input variables

Kowalski
Al-Rabadi
Piotrowski
Perkowski
Lattice Structure for Multivalued and Binary Logic

Cell has 4 inputs and 2 outputs

Can we make the cell reversible?

Multi-valued output variable

A > B

C < D

E = G

E < G or G > 0

E < G and G < 0

Cell has 4 inputs and 2 outputs

Multi-valued input variables

Can we make the cell reversible?

Multi-valued input variables

Kowalski

Al-Rabadi

Piotrowski

Perkowski

Multivalued input variables
<table>
<thead>
<tr>
<th>Control</th>
<th>left</th>
<th>right</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>value</td>
<td>-</td>
<td>value</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>value</td>
<td>value</td>
</tr>
</tbody>
</table>

We want to make this cell reversible.

Values not separated.
Let us try to repeat control variable in output

Still not separated
Repeating variables will not help

Now it works!
This means that we added another MUX.
INPUTS OUTPUTS

Permutation gate

Control

<table>
<thead>
<tr>
<th></th>
<th>000</th>
<th>001</th>
<th>010</th>
<th>011</th>
<th>100</th>
<th>101</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>000</td>
<td>001</td>
<td>100</td>
<td>101</td>
<td>010</td>
<td>110</td>
</tr>
<tr>
<td>1</td>
<td>010</td>
<td>110</td>
<td>011</td>
<td>111</td>
<td>100</td>
<td>001</td>
</tr>
</tbody>
</table>

left  right

output  output1  output2
.... And we reinvented the Fredkin Gate ....!!!
Lattice Structure for Binary Logic

\[ F = S^{1,3} (A, B, C) \]
Three-Dimensional Realization of Multiple-Valued Functions using Reversible Logic

- M. Perkowski, A. Al-Rabadi, P. Kerntopf, A. Mishchenko, and M. Chrzanowska-Jeske
- PORTLAND QUANTUM LOGIC GROUP
Two-Dimensional Lattice Diagrams for reversible logic
Notation for Fredkin Gates

A circuit from two multiplexers

Its schematics

This is a reversible gate, one of many
Three Types of General Expansions

Forward Shannon

\[ f \]

A ->

\[ f_0 \]
\[ 0 \]
\[ 1 \]
\[ f_1 \]

f and A -> f_0 and f_1
Three Types of General Expansions

(b)

\[ g, h, \text{ and } A \rightarrow g_1A + h_0A' \]

Reverse Shannon
Three Types of General Expansions

A

\[ g h \]

\[ g0A' + h1A \]

\[ g1A + h0A' \]

Reversible Shannon

\[ g, h, and A \rightarrow g0A' + h1A and g1A + h0A' \]
Forward Shannon Expansion

- This graphical method can be easily extended to MV logic;
  - for instance for the ternary logic the Kmap is split into three Kmaps;
  - for $x_i = 0$, for $x_i = 1$, and for $x_i = 2$, respectively.

- Each of these maps has one-third cares and two-thirds don’t cares.
Treating the reversible Shannon Expansion as the *cofactor permuter* is a useful heuristic to create algorithms.

The method to synthesize such Lattices, called *Levelized Variable Decomposition Method* (for Shannon), presented here, is a special case of a general method of hierarchical decomposition of arbitrary reversible functions into reversible logic gates.
To distinguish this new general decomposition from the well-known decompositions of Ashenhurst, Curtis or Shannon, we call it the *Multi-purpose Portland Decomposition*, the *MP-decomposition* for short.
Generalization

- We mapped the logic function to a lattice structure of geometrical connections.
- There is nothing in our method to map to only this kind of structure.
- We can map to any selected regular structure.
- We can also map to an irregular structure with arbitrary connections.
Generalizations of Fredkin gate

- Observe, that this definition of the gate does not specify the type of signals.
- Thus they can be binary, multi-valued, fuzzy, continuous or complex.
- The only requirement is that the relation of order (<) is defined on them.
- It is interesting and important that a single reversible gate in binary logic has many generalizations in multiple-valued logic.
Generalizations of Fredkin gate

- Because it has been shown in [1] that there are many multiple-valued and multi-output \((k>3)\) generalizations of Fredkin gate, the name “modified” assigned by Picton is not correct.
- The generalization invented by him we will call the *Picton Gate*, while generalization of Fredkin-like gates we call “*new gates*”. 
Generalizations of Fredkin gate

• The exhaustive list of families of all such permutative multi-valued gates (both Shannon-like and Davio-like) has been presented in [1] and even more families in [18].

• These of the “new gates” that use multiplexers only are similar to the original Fredkin gate but they use multiple-valued multiplexers.

• Such multiplexers have been already realized in many technologies, including super-pass transistors [9], so building these new gates should be also possible.
Generalizations of Fredkin gate

- We believe therefore that they are good candidates for future reversible multiple-valued nano-technologies. The new generalization of Fredkin gate using multi-valued logic has additional advantages and is simpler. Let us observe, that equations for the binary 4 * 4 binary Fredkin gate can be rewritten as follows:

\[ P = A, \quad Q = \begin{cases} C & \text{if } A = 1 \\ B & \text{if } A = 0 \end{cases}, \quad R = \begin{cases} B & \text{if } A = 1 \\ D & \text{if } A = 0 \end{cases}, \quad S = \begin{cases} D & \text{if } A = 1 \\ C & \text{if } A = 0 \end{cases} \]

- Now, it can be easily generalized to a 4 * 4 ternary gate as follows:

\[ P = A, \quad Q = \begin{cases} B & \text{if } A = 2 \\ C & \text{if } A = 1 \\ D & \text{if } A = 0 \end{cases}, \quad R = \begin{cases} C & \text{if } A = 2 \\ D & \text{if } A = 1 \\ B & \text{if } A = 0 \end{cases}, \quad S = \begin{cases} D & \text{if } A = 2 \\ B & \text{if } A = 1 \\ C & \text{if } A = 0 \end{cases} \]
Three-Dimensional Lattice Diagrams for reversible logic
A ternary Fredkin gate

2D symbol

3D symbol
A two-dimensional structure of ternary Fredkin gates
A three-dimensional structure of ternary Fredkin gates, corresponding to the previous one
The general plan of the previous three-dimensional structure

4 inputs to a cell

4 outputs from a cell
The general plan of another three-dimensional structure

Observe, that three dimensional structures can be also created with standard 3*3 Fredkin and binary 4*4 Fredkin gates.
The general plan of yet another three-dimensional structure

3 inputs to a cell

3 outputs from a cell
Reversible Lattice Structure for Binary Logic

**Advantages**

- regular structure
- binary Fredkin Gate
- planar structure (good for Quantum Logic)
- Easy algorithmic creation
- Reasonable garbage

**Disadvantages**

- Variable ordering?
- Symmetrization?
- Garbage still exists
Do you remember that there are other binary expansions?

- **All Binary Expansions**
- Shannon - S
- Flipped Shannon - fS
- Positive Davio - pD
- Negative Davio - nD
- Flipped Positive Davio - fpD
- Flipped Negative Davio - fnD

**Ideas**
- Fredkin = <Var, S, fS>
- what about these?
  - <Var, pD, fpD>
  - <Var, nD, fnD>
  - <Var, nD, pD>
  - ....

we checked some of them to work
Do you remember that there are other component functions of reversible gates?

**All Binary Balanced Expansions:**
- ..... 
- Linear functions - L
- Negations - N
- Majorities - M

**Ideas**
- Fredkin = \(<\text{Var}, S, fS>\)
- what about these?
  - \(<\text{N}, pD, fpD>\)
  - \(<\text{Var}, M, fnD>\)
  - \(<\text{Var}, nD, L>\)
- ..... 

We checked some of them to work.
As you see, this opens a very broad area of research that will lead to invention of new reversible gates and regular structures that use them.

- Easy way to become a pioneer:
  - Investigate all combinations
  - Use genetic programming or other search methods to build structures and map functions to them
  - There is a place for many researchers
  - Nobody does this research

But this was only for binary

What about multivalued, fuzzy, arithmetic or other logics?
.... And we reinvented the Fredkin Gate ....!!!

• But what about the variant with two control signals?
Cell has 4 inputs and 4 outputs

Cell is reversible!

Multivalued input variables

Multi-valued output variable

MV and Generalized MV
Fredkin

Kowalski
Al-Rabadi
Piotrowski
Perkowski
Multi-valued logic generates less signals

Hence it generates less waste

Of course, it generates also less power, less connections and is easier to test
The real-life functions are multi-output.

Thus, there exists an opportunity to re-use some waste functions in other output functions.

This is a tough problem.

I do not know now how to solve it!

We need some group creativity.
Select other function of two variables

Select other pairs of VAR-type and NOT-type functions

Select other pairs of MUX-type functions
Generalized Multi-valued Fredkin Gate

• The number of these gates is astronomical

• We need both computer generation and some intelligence, simply generating them all would be a nonsense

• Very wide area of research

• It will give hints to gate designers what to look for
• The general levelized method can assume any structure of the layout, thus any order and choice of input signals of successive Reversible Shannon expansions.

• Assuming other type of structure, cascade or non-planar lattice with intersecting signals, this other type of structure would be created.

• For arbitrary structures, however, the method requires small modification: if the structure is too constrained, the structural equations have no solutions or the algorithm loops.
• This happens, for instance, when a Maitra Cascade structure is assumed for a function that is not Maitra-realizable.

• It happens also when we assume a levelized circuit of too narrow a bandwidth.

• Thus the algorithm must be modified to deal with these special cases.

• Finally, our general approach will work also for irregular structures. In such case, any pair of signals can be the inputs to the Reversible Shannon Expansion, regardless of their order. The signals are paired to give the smallest evaluated total complexity for the level.
- Arbitrary **symmetric function** can be realized in a lattice without repeated variables.
- Arbitrary (non-symmetric) function can be realized in a lattice with **repeated variables** (so-called **symmetrization**).
- Similar property exist for the presented method. This method **terminates for arbitrary function**, assuming that the variables are repeated in levels. Thus, if the leafs of the lattice are not constants after expanding for all input variables, some of these variables are used again in new levels of expansions, which we call variable repetition.
- Interestingly, the functions that do not require variable repetition in the Reversible Shannon Lattices **are not symmetric functions**.
- We work on the **characterization of the functions realizable in these structures without repetitions** and respective synthesis algorithms.
• We can impose during joining the structure of the three dimensional lattice. Such lattice is typical for some crystals.

• There are also several other three-dimensional structures corresponding to other types of bonds or constraints that exist in Nature (for example, *quantum dot computers*).

• This leads to very many new circuit types, which are reversible and multi-valued generalizations of Shannon Lattices, Kronecker Lattices, Fat Trees, and many other structures introduced in the past.
Conclusions and Future Work

- New concepts:
  - (1) Reversible Shannon Expansion for $k \times k$ binary Fredkin Gates ($k>2$),
  - (2) planar Reversible Fredkin Lattice structures for logic based on binary $3 \times 3$ Fredkin gates,
  - (3) Three-dimensional Reversible Fredkin Lattice structures for ternary logic based on $4 \times 4$ ternary new gates.

- We plan to design these gates in CMOS and Optical technologies.
Conclusions and Future Work

- A fundamental idea of this paper is the use of *three-dimensional space for reversible logic*
- Several realizations of reversible and quantum logic, such as for instance *quantum dots*, involve a geometrical space.
- For instance, in the *quantum dot model* this space is two-dimensional.
- Here we propose to *create three-dimensional regular structures*, because our physical world is three dimensional.
Conclusions and Future Work

• Thus, three dimensional regular structures provide the best use of space.

• Creating our theoretical model, we hope that some day a way to build such structures in matter, possibly at the quantum level, will be invented.
Symmetric functions
Good Use of two Multi-valued Fredkin Gates to create MIN/MAX gate

\[
\begin{align*}
\text{MIN}(A,B) &= A \geq B \\
\text{MAX}(A,B) &= A \geq B
\end{align*}
\]
Can be added in our future work, but is not necessary for this paper.
Property

- Even without ability of realizing any symmetric function of two variables, one can realize arbitrary single-output symmetric function of any number of variables:
  - by building a regular structure from MAX/MIN gates and next applying EXORing operations to find single-value symmetric coefficients and next OR gates to sum them.
Every Binary Symmetric Function can be composed of MIN/MAX gates: Example for three variables

\[
\begin{align*}
\text{MAX}(A,B,C) &= (A+B)+C = S^{1,2,3}(A,B,C) \\
\text{MIN}(A,B,C) &= (A*B)*C = S^3(A,B,C)
\end{align*}
\]

Indices of symmetric binary functions of 3 variables
Every single index Symmetric Function can be created by EXOR-ing last level gates of the previous regular expansion structure.
Example for four variables

MAX(A, B, C, D) = A + B + C + D

= S^{1,2,3,4}(A, B, C)

Example for four variables, EXOR level added

MAX(A,B) = A + B
MIN(A,B) = A * B

Max/M in gate

MAX(A,B,C) = (A+B)+C = S_{1,2,3}(A,B,C)
C(A+B)

MIN(A,B,C) = (A*B)*C = S_{3}(A,B,C)

Now it is obvious that any multi-output function can be created by OR-ing the outputs of EXOR level
Now we extend to Reversible Logic

\[
\begin{align*}
\text{MAX}(A,B,C) &= (A+B)+C = S_{1,2,3}(A,B,C) \\
\text{MIN}(A,B,C) &= (A\cdot B)\cdot C = S_{3}(A,B,C)
\end{align*}
\]

\[
\begin{align*}
\text{MAX}(A,B,C,D) &= A+B+C+D = S_{1,2,3,4}(A,B,C) \\
\text{MIN}(A,B,C,D) &= A\cdot B\cdot C\cdot D = S_{4}(A,B,C,D)
\end{align*}
\]

Denotes Feynman (controlled NOT) gate

Denotes fan-out gate
**Theorem for Binary Reversible Logic**

**Theorem 1**: Every positive unate (symmetric) function of 2 variables can be realized in 1 gate.

Every positive unate function of 3 variables can be realized in 1+2 gates.

Every positive unate function of 4 variables can be realized in 1+2+3 gates.

Every positive unate symmetric function of n variables can be realized in 
\[1+2+.. n-1 = n(n-1)/2\] MAX/MIN gates.

\[
\begin{align*}
\text{MAX}(A,B,C) &= (A+B)+C = S_{1,2,3}(A,B,C) \\
\text{MIN}(A,B,C) &= (A*B)*C = S_{3}(A,B,C) \\
\text{MAX}(A,B,C,D) &= A+B+C+D = S_{1,2,,3,4}(A,B,C) \\
\text{MIN}(A,B,C,D) &= A*B*C*D = S_{4}(A,B,C,D) \\
\end{align*}
\]
Theorems for Binary Reversible Logic

**Theorem 2**: Every single index totally symmetric function of \( n \) variables can be realized in \( n(n-1)/2 \) MAX/MIN gates, \( n-2 \) fan-out gates and \( n-1 \) Feynman gates.

**Theorem 3**: Every single-output totally symmetric function of \( n \) variables can be realized in \( n(n-1)/2 \) MAX/MIN gates, \( n-2 \) fan-out gates, \( n-1 \) Feynman gates and XXX? OR gates.

**NOT FINISHED HERE**
Theorems for Binary Reversible Logic

Arbitrary symmetric function can be created by exoring single indices

Next functions

S\(^1\)(A,B,C,D) → S\(^{2,3,4}\)(A,B,C,D)

S\(^2\)(A,B,C,D) → S\(^{3,4}\)(A,B,C,D)

S\(^4\)(A,B,C,D) → S\(^1\)(A,B,C,D)

S\(^1\)@S\(^2\)=S\(^{1,2}\)

S\(^1\)@S\(^4\)=S\(^{1,4}\)

S\(^{1,2}\)@S\(^4\)=S\(^{1,2,4}\)
Synthesis from (p)KFDDs
Compositions
and
Decompositions
Adders

- One possible implementation of a half adder

\[ \text{A xor B (Sum)} \]
\[ \text{A and B (Carry)} \]
How to build the Half-Adder?

AB xor 0

CCN

A xor B (Sum)

CN

A and B (Carry)

Feynman

Toffoli

Product-like

Sum-like

Here we guessed the solution from expressions.
Can we do this systematically?

This is what we would like to have

But this is not reversible,
Can we do this systematically?

Mapping of a half-adder

The simplest way to separate them is to add variable A
Can we do this systematically?

A xor B (Sum)
A and B (Carry)

This is what we would like to have

But this is not reversible, because more outputs than inputs
Can we do this systematically?

Can we do this systematically?

Can we do this systematically?

Can we do this systematically?

Can we do this systematically?
We invented a new reversible gate with three inputs and three outputs. But it is not Toffoli gate.
We showed that the new gate can be composed from Toffoli and Feynman Gates.

Because Toffoli gate is universal, every gate can be designed from it. But, generating a big waste of outputs.

Here we showed that we can generate the new gate only from two gates with only one “waste of outputs”.

“waste of outputs” should be the efficiency measure of the method.
Adders

- One possible implementation of a full adder
Adders

A XOR B
Sum

A
A XOR B
Sum

We repeat the input

We create a constant input
Another Systematic Method
**Needs separation**

(a) P, Q

(b) P = A @ B, Q = AB

(c) C = 0:

(d) A added to separate
Decomposing with Kerntopf’s Gate

A → Q=AB

B → P=A+B

C=1 → R=!B

f=AB (carry)

g=A @ B (Sum)

Garbage

Kerntopf

Feynman
Decomposing with Toffoli’s Gate

CCN

A

B

0

R=A
P=A @ B (Sum)
Q=AB (Carry)

CN

Feynman

Toffoli

AB @ 0
Adaptations of functional decomposition methods
Ashenhurst/Curtis like reversible decompositions
Bi-Decomposition-like reversible decompositions