Towards Better Standard Cell Library: Optimizing Compound Logic Gates for TFHE

Kotaro Matsuoka*
Yusuke Hoshizuki**
Takashi Sato*
Song Bian*
2021 Nov. 15

*Kyoto University
**AXELL CORPORATION
Agenda

1. TFHE & Standard Cell
2. Preliminary: Blind Rotate
3. Construction of proposed gates
4. DEMO: Workflow for a 16-bit multiplier
5. Conclusion
TFHE & Standard Cell
TFHE & Standard Cell

**TFHE: Torus Fully Homomorphic Encryption**
- Can evaluates Logic Circuits without decryption.
- Ordinary logic synthesis tools (ex: Yosys) can be used.

**Standard Cell: Optimized implementation of small logic functions**
- ex: NAND, Full Adder, AOI21, etc.
- Logic synthesis tools will represent the circuit using Standard Cells.

**Original TFHE only supports 2-input-1-output gates (ex: NAND) and MUX**
- More sophisticated gates can be used in the logic synthesis.

Propose some 3-input or multi-output gates. Around 2x performance benefit in DEMO.
Preliminary: Blind Rotate
## Encoded Message Space in Torus

**Torus (\(T\)):** \( \mathbb{R} \mod 1 \in \left[-\frac{1}{2}, \frac{1}{2}\right) \)

- The message space of TFHE
- Group of the fraction parts of real numbers
  - Supports addition (ex.: \(0.4 + 0.7 = 1.1 \equiv 0.1 \mod 1\))
  - Supports multiplication with integers (ex. \(3 \cdot 0.4 = 1.2 \equiv 0.2 \mod 1\))
- Need to encode the plaintext space, Binary \(\{0, 1\}\), into Torus.

### Encoded message space

\[ M_t = \left\{-\frac{1}{t}, \frac{1}{t}\right\} \]

- \(t \in \mathbb{N}\). Corresponds \(\{0, 1\}\) respectively.
- We use \(M_8\) and \(M_{12}\)
  - \(M_8\): Used in the original TFHE implementation
  - \(M_{12}\): More performance benefit but increase the decryption error rate
Blind Rotate

Blind Rotate: The core functionality of TFHE

- Can evaluate Look Up Table (LUT)
  - LUT is represented as a polynomial $TV[X] \in \mathbb{T}[X]/X^N + 1$
  - $\rho \in \mathbb{Z}/2N\mathbb{Z}$ is the (encrypted) index input
  - Output: The constant term of $X^{-\rho} \cdot TV[X]$

\[
TV[X] = \sum_{i=0}^{N-1} \mu_i \cdot X^i
\]

<table>
<thead>
<tr>
<th>$\mu_0$</th>
<th>$\mu_1$</th>
<th>$\mu_2$</th>
<th>$\mu_3$</th>
<th>$\mu_4$</th>
<th>$\mu_5$</th>
<th>$\mu_6$</th>
<th>$\mu_7$</th>
</tr>
</thead>
</table>

\[
X^{-3} \cdot TV[X]
\]

| $\mu_3$ | $\mu_4$ | $\mu_5$ | $\mu_6$ | $\mu_7$ | $-\mu_0$ | $-\mu_1$ | $-\mu_2$ |

Output: $\mu_3$

Figure 1: Blind Rotate ($\rho = 3, N = 8$)
Possible LUTs for BR have two constraints.

1. **Negacyclic Rotation:**
   \[ X^{-\rho} \cdot TV[X] = -X^{-(\rho+N)} \cdot TV[X] \]

2. **Linear Combination:**
   Only \( \frac{t}{4} \) degrees of freedom for LUT entries

---

**Figure 2:** Negacyclic Rotation \((N = 8)\)
Possible LUTs for BR have two constraints.

1. Negacyclic Rotation: \( X^{-\rho} \cdot TV[X] = -X^{-(\rho+N)} \cdot TV[X] \)

2. Linear Combination: Only \( \frac{t}{4} \) degrees of freedom for LUT entries
   - There are only \( \frac{t}{2} \) possible values (without error) for \( \rho \)
   - \( p \): number of inputs, \( m_i \): encoded messages of inputs
   - \( a_i \in \mathbb{Z}, b \in [-\frac{t}{2}, \frac{t}{2}) \)
   - \( \rho \approx \lceil 2N \cdot (\sum_{i=0}^{p-1} a_i \cdot m_i + \frac{b}{t} \text{ mod } 1) \rceil \)

ex.) \( p = 2, m_i \in \mathbb{M}_8, a_i = 1, b = 1 \)
\[ \Rightarrow \rho \approx [2N \cdot (m_0 + m_1 + \frac{1}{8})] \in \{2N \cdot \frac{1}{8}, 2N \cdot \frac{3}{8}, -2N \cdot \frac{3}{8}, -2N \cdot \frac{1}{8}\} \]

\[
TV[X] = \sum_{i=0}^{1} \sum_{j=0}^{N-1} \mu_i \cdot X^{i \cdot \frac{N}{2} + j}
\]

\[
\begin{array}{cccccccc}
\mu_0 & \mu_0 & \mu_0 & \mu_0 & \mu_1 & \mu_1 & \mu_1 & \mu_1 \\
\end{array}
\]

Output for \( \rho = 2N \cdot \frac{1}{8} \)  
Output for \( \rho = 2N \cdot \frac{3}{8} \)
Construction of proposed gates
List of proposed gates

- $\mathbb{M}_8$
  - Half Adder (Gao’s method)
  - 2BR Full Adder

- $\mathbb{M}_12$
  - 1BR Full Adder (Multi Value)
  - AOI21 and OAI21
List of proposed gates

- $M_8$
  - Half Adder (Gao’s method)
  - 2BR Full Adder

- $M_{12}$
  - 1BR Full Adder (Multi Value)
  - AOI21 and OAI21
1BR Full Adder

- Full Adder = 3-input XOR gate + 3-input majority gate
  - 1BR Full Adder evaluate this by one Blind Rotate
- Extended from FHEW version\(^1\)
- Construction:
  1. Construct 3-input XOR
  2. Construct 3-input majority gate
  3. Merge above by Multi-value technique

---

3-input XOR

- 3-input XOR = $A \oplus B \oplus C$
- All inputs can be interchanged
  - $\rho \approx \lceil 2N \cdot (\sum a_i \cdot m_i + \frac{b}{i} \mod 1) \rceil$
  - $a_A = a_B = a_C = 1, b = 0$
- $\frac{\rho}{2N}$ of yellow and white satisfy:
  $2N \cdot -\frac{3}{12} + N \equiv 2N \cdot \frac{3}{12} \mod 2N$

**Table 1:** Output and $\frac{\rho}{2N}$ of 3-input XOR

<table>
<thead>
<tr>
<th>c\ab</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0/ $-\frac{3}{12}$</td>
<td>1/ $-\frac{1}{12}$</td>
<td>0/ $\frac{1}{12}$</td>
<td>1/ $-\frac{1}{12}$</td>
</tr>
<tr>
<td>1</td>
<td>1/ $-\frac{1}{12}$</td>
<td>0/ $\frac{1}{12}$</td>
<td>1/ $\frac{3}{12}$</td>
<td>0/ $\frac{1}{12}$</td>
</tr>
</tbody>
</table>

**Figure 3:** 3-input XOR LUT
3-input majority gate

- 3-input majority gate = 
  \((A \land B) \lor (B \land C) \lor (A \land C)\)
- All inputs can be interchanged
  - Same as 3-input XOR
  - \(a_A = a_B = a_C = 1, b = 0\)

Table 2: Output and \(\frac{\rho}{2N}\) of 3-input majority gate

<table>
<thead>
<tr>
<th>c\ab</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0/(-\frac{3}{12})</td>
<td>0/(-\frac{1}{12})</td>
<td>(\frac{1}{12})</td>
<td>0/(-\frac{1}{12})</td>
</tr>
<tr>
<td>1</td>
<td>0/(-\frac{1}{12})</td>
<td>(\frac{1}{12})</td>
<td>(\frac{3}{12})</td>
<td>(\frac{1}{12})</td>
</tr>
</tbody>
</table>

Figure 4: 3-input majority gate LUT
Multi-value technique

- Blind Rotate is a heavy operation.
- $TV[X]$ for 3-input XOR and majority gate have common divisor
  
  $TV_0[X] = \sum_{i=0}^{\lfloor \frac{N}{3} \rfloor - 1} \frac{1}{12} X^i$

- $TV[X]$ for 3-input XOR: $TV_{xor}[X] \cdot TV_0[X]$
  
  $TV_{xor}[X] = -1 + X^{\lfloor \frac{N}{3} \rfloor} - X^2 \cdot \lfloor \frac{N}{3} \rfloor$

- $TV[X]$ for 3-input majority gate: $TV_{majority}[X] \cdot TV_0[X]$
  
  $TV_{majority}[X] = 1 + X^{\lfloor \frac{N}{3} \rfloor} + X^2 \cdot \lfloor \frac{N}{3} \rfloor$

:: We can share the output of one BR

$i \in \{xor, majority\}$

$X^{-\rho} \cdot TV[X] = X^{-\rho} \cdot (TV_i[X] \cdot TV_0[X]) = TV_i[X] \cdot (X^{-\rho} \cdot TV_0[X])$

Blind Rotate
• AOI: AND OR Inverter

• \( \neg((A \land B) \lor C) \)
  • \( c \) is not interchangeable
  • Must be treated differently from \( a \) and \( b \)
  • \( a_A = a_B = 1, a_C = 2, b = 1 \)

### Table 3: Output and \( \frac{\rho}{2N} \) of AOI21

<table>
<thead>
<tr>
<th>( c \backslash ab )</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( 1/\frac{1}{12} )</td>
<td>( 1/\frac{1}{12} )</td>
<td>( 1/\frac{3}{12} )</td>
<td>( 1/\frac{3}{12} )</td>
</tr>
<tr>
<td>1</td>
<td>( 0/\frac{1}{12} )</td>
<td>( 0/\frac{1}{12} )</td>
<td>( 0/\frac{5}{12} )</td>
<td>( 0/\frac{3}{12} )</td>
</tr>
</tbody>
</table>

### Figure 5: AOI21 LUT

### Figure 6: AOI21
• OAI: OR AND Inverter
• \( \neg((A \lor B) \land C) \)
  • \( a_A = a_B = 1, a_C = 2, b = -1 \)
  • Same TV[\( X \)] but use a different pie

Table 4: Output and \( \frac{\rho}{2N} \) of OAI21

<table>
<thead>
<tr>
<th>c\ab</th>
<th>00</th>
<th>01</th>
<th>11</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( 1/ -\frac{5}{12} )</td>
<td>( 1/ -\frac{3}{12} )</td>
<td>( 1/ -\frac{1}{12} )</td>
<td>( 1/ -\frac{3}{12} )</td>
</tr>
<tr>
<td>1</td>
<td>( 1/ -\frac{1}{12} )</td>
<td>( 0/ \frac{1}{12} )</td>
<td>( 0/ \frac{3}{12} )</td>
<td>( 0/ \frac{1}{12} )</td>
</tr>
</tbody>
</table>

Figure 7: OAI21 LUT

Figure 8: OAI21

\[ \frac{\rho}{2N} = \frac{1}{12} \]
DEMO: Workflow for a 16-bit multiplier
List of Software

- **oveus-tfhe\(^2\):** Our implementation derived from TFHEpp\(^3\)
- **Yosys\(^4\):** Logic synthesis tool
- **Sudachi\(^5\):** Homomorphic logic circuit execution engine

---

\(^2\)https://github.com/axell-corp/oveus-tfhe
\(^3\)https://github.com/virtualsecureplatform/TFHEpp
\(^4\)https://github.com/YosysHQ/yosys
\(^5\)https://github.com/virtualsecureplatform/Sudachi
Conclusion
Conclusion

• We proposed some 3-input or multi-output gates
  • Demonstrated proposed gates give performance benefit in the real environment
  • Implementation: https://github.com/axell-corp/oveus-tfhe

• There is room for more sophisticated standard cells
  • More than depth 2 BR functions
    • SWAP gate in the implementation
  • \( t > 12 \) cases
    • 4-input AND and OR in \( \mathbb{M}_{16} \) in the implementation
This work is licensed under a Creative Commons “Attribution 4.0 International” license.