Camellia Hardware Macro Specification

<table>
<thead>
<tr>
<th>Version</th>
<th>Update</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>2007/09/16</td>
<td>Initial version is created</td>
</tr>
<tr>
<td>0.1.1</td>
<td>2007/09/21</td>
<td>Timing chart (Fig. 7) is fixed</td>
</tr>
<tr>
<td>0.2</td>
<td>2007/09/25</td>
<td>Translated</td>
</tr>
</tbody>
</table>

1. Overview

1.1 Hardware macro overview

The features of this Camellia hardware macro are summarized in Table 1. Only the ECB (Electronic Code Book) mode is supported, but the other modes such as CBC (Cipher Block Chaining) can be easily supported by using additional data buffers and a control circuit.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Camellia</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data block size</td>
<td>128 bits</td>
</tr>
<tr>
<td>Key size</td>
<td>128 bits</td>
</tr>
<tr>
<td>Mode of operation</td>
<td>Electronic Code Book (ECB)</td>
</tr>
<tr>
<td>Source file name</td>
<td>Camellia.v</td>
</tr>
<tr>
<td>Description Language</td>
<td>Verilog-HDL</td>
</tr>
<tr>
<td>Top module name</td>
<td>Camellia</td>
</tr>
<tr>
<td>Throughput</td>
<td>128 bit / 23 clock</td>
</tr>
<tr>
<td>Round keys</td>
<td>On-the-fly</td>
</tr>
</tbody>
</table>

1.2 Algorithm overview

Camellia is a Feistel-type block cipher jointly developed by NTT (Nippon Telegraph and Telephone Corp.) and Mitsubishi Electric. Camellia supports 128-, 192-, and 256-bit keys. In the following, we describe 128-bit version. The detailed algorithm is described in the specification [1].

Fig. 1 shows the data randomization block including 18-round Feistel Network with functions $F$ and $FL/FL^{-1}$. A 64-bit round function $F$ consists of eight 8-bit S-boxes and an XOR network. Two 64-bit linear functions $FL$ and $FL^{-1}$ are given by AND, OR, XOR, and 1-bit rotation.

Round keys $kw_1$-$kw_4$, $kl_1$-$kl_4$, and $k_1$-$k_{18}$ are used in initial/final key addition, $F$ function, and $FL/FL^{-1}$ functions. The round keys are generated from a secret key $K_L$ and an intermediate key $K_d$ according to Table 1. $K_d$ is generated from $K_L$ using Feistel Network and $F$-function as shown in Fig. 2.
Fig. 1 Camellia encryption algorithm

Table 2  Generation rule for round keys

<table>
<thead>
<tr>
<th>Initial XOR</th>
<th>$kw_{0,28}$</th>
<th>$kw_{28,4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>F (Round 1)</td>
<td>$k_{1,0}$</td>
<td>$(K_L&lt;&lt;&lt;0)_0$</td>
</tr>
<tr>
<td>F (Round 2)</td>
<td>$k_{2,0}$</td>
<td>$(K_L&lt;&lt;&lt;0)_0$</td>
</tr>
<tr>
<td>F (Round 3)</td>
<td>$k_{3,0}$</td>
<td>$(K_L&lt;&lt;&lt;0)_0$</td>
</tr>
<tr>
<td>F (Round 4)</td>
<td>$k_{4,0}$</td>
<td>$(K_L&lt;&lt;&lt;0)_0$</td>
</tr>
<tr>
<td>F (Round 5)</td>
<td>$k_{5,0}$</td>
<td>$(K_L&lt;&lt;&lt;0)_0$</td>
</tr>
<tr>
<td>F (Round 6)</td>
<td>$k_{6,0}$</td>
<td>$(K_L&lt;&lt;&lt;0)_0$</td>
</tr>
<tr>
<td>$F_L$</td>
<td>$k_{7,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>$F_L^{-1}$</td>
<td>$k_{8,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 7)</td>
<td>$k_{9,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 8)</td>
<td>$k_{10,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 9)</td>
<td>$k_{11,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 10)</td>
<td>$k_{12,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 11)</td>
<td>$k_{13,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 12)</td>
<td>$k_{14,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>$F_L$</td>
<td>$k_{15,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>$F_L^{-1}$</td>
<td>$k_{16,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 13)</td>
<td>$k_{17,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 14)</td>
<td>$k_{18,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 15)</td>
<td>$k_{19,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 16)</td>
<td>$k_{20,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 17)</td>
<td>$k_{21,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>F (Round 18)</td>
<td>$k_{22,0}$</td>
<td>$(K_L&lt;&lt;&lt;5)_0$</td>
</tr>
<tr>
<td>Final XOR</td>
<td>$kw_{0,28}$</td>
<td>$(K_L&lt;&lt;&lt;11)_0$</td>
</tr>
<tr>
<td></td>
<td>$kw_{28,4}$</td>
<td>$(K_L&lt;&lt;&lt;11)_0$</td>
</tr>
</tbody>
</table>

Fig. 2 Intermediate key generation.
2. I/O ports

I/O ports of the Camellia macro are summarized in Table 3.

<table>
<thead>
<tr>
<th>Port name</th>
<th>Direction</th>
<th>Width</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kin</td>
<td>In</td>
<td>128</td>
<td>Key input</td>
</tr>
<tr>
<td>Din</td>
<td>In</td>
<td>128</td>
<td>Data input</td>
</tr>
<tr>
<td>Dout</td>
<td>Out</td>
<td>128</td>
<td>Data output</td>
</tr>
<tr>
<td>Krdy</td>
<td>In</td>
<td>1</td>
<td>When Krdy=1, a secret key is latched in an internal register, and the intermediate key generation process is executed. If Drdy and Krdy assigned to 1 at the same time, Krdy=1 has priority.</td>
</tr>
<tr>
<td>Drdy</td>
<td>In</td>
<td>1</td>
<td>When Drdy=1, a plaintext (or ciphertext) data is latched in an internal register and the encryption (or decryption) process is started.</td>
</tr>
<tr>
<td>EncDec</td>
<td>In</td>
<td>1</td>
<td>Encryption and decryption are executed when EncDec=0 and EncDec=1, respectively. Bit data on the port EncDec is stored in an internal register when encryption or decryption starts in response to Drdy=1.</td>
</tr>
<tr>
<td>RSTn</td>
<td>In</td>
<td>1</td>
<td>Reset signal. Sequencer logic and internal registers are reset when this signal is assigned to 0. The reset can be executed any time when the clock signal CLK is input, even if the enable signal EN=0.</td>
</tr>
<tr>
<td>EN</td>
<td>In</td>
<td>1</td>
<td>Enable signal. When EN=1, this macro is activated.</td>
</tr>
<tr>
<td>CLK</td>
<td>In</td>
<td>1</td>
<td>System clock. All registers are synchronized with the rising edge of this signal.</td>
</tr>
<tr>
<td>BSY</td>
<td>Out</td>
<td>1</td>
<td>Busy status flag. This signal is assigned to 1 while an encryption, decryption, or key generation process is executed. When this signal is 1, both Drdy and Krdy are ignored.</td>
</tr>
<tr>
<td>Kvld</td>
<td>Out</td>
<td>1</td>
<td>When round-key generation process is completed, this signal becomes 1 during the next one clock cycle, and then it goes 0. Soon after that, encryption and decryption processes are ready to start.</td>
</tr>
<tr>
<td>Dvld</td>
<td>Out</td>
<td>1</td>
<td>When encryption or decryption process is completed and cipher text or plain text are ready on the data output port Dout, this signal becomes 1 during the next one clock cycle, and then it goes 0.</td>
</tr>
</tbody>
</table>
3. Hardware Architecture

3.1 Datapath

A datapath of the Camellia macro is shown in Fig. 3. This macro executes 1-round operation in 1 clock cycle. A 128-bit block of plaintext are encrypted / decrypted in 16 clocks.

A secret key is contained in an internal register $kl$ through a 128-bit port $Kin$ in the key-scheduling. Then the intermediate key generation is started in the data randomization block. The obtained intermediate key is set to an internal register $ka$ after 6 clocks. Round keys are generated from the values in $kl$ and $ka$ on the fly.

An input data (plaintext for encryption, ciphertext for decryption) is set to an internal register $D_{\text{out\_reg}}$ through a 128-bit port $Din$ in the data-randomization block, An output data (ciphertext for encryption, plaintext for decryption) is obtained from a 128-bit port $Dout$.

![Fig.3 Datapath](image)

3.2 State Diagram

The state diagram of the Camellia sequencer and its description are shown in Fig. 4 and Table 4, respectively.
Fig. 4 State diagram of sequencer

Table 4 State diagram of sequencer logic

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDLE</td>
<td>Initial state. Only key input is accepted.</td>
</tr>
<tr>
<td>IDLE READY</td>
<td>Idling state where intermediate-key generation is finished. Both data input and key input are accepted.</td>
</tr>
<tr>
<td>KEY GET</td>
<td>States for intermediate-key generation.</td>
</tr>
<tr>
<td>KEY F_FUNC</td>
<td>St</td>
</tr>
<tr>
<td>KEY XOR</td>
<td>St</td>
</tr>
<tr>
<td>RANDOMIZE GET</td>
<td>States for data randomization (in encryption or decryption).</td>
</tr>
<tr>
<td>RANDOMIZE INITIAL XOR</td>
<td>St</td>
</tr>
<tr>
<td>RANDOMIZE F_FUNC</td>
<td>St</td>
</tr>
<tr>
<td>RANDOMIZE FL_1</td>
<td>St</td>
</tr>
<tr>
<td>RANDOMIZE FL_2</td>
<td>St</td>
</tr>
<tr>
<td>RANDOMIZE FINAL XOR</td>
<td>St</td>
</tr>
</tbody>
</table>

4. Timing Chart

Fig. 5 shows the timing chart of the key scheduling, encryption, and decryption process for the Camellia macro in the minimum cycles for the control signals. The operation are performed as follows.

CLK1: The sequencer logic is initialized by resetting RSTn to 0.
CLK2: By asserting Krdy=1, the 128-bit secret key on Kin is stored to an internal register. Soon after that, the key scheduling process is started, and BSY is set to 1.
CLK3–CLK8: The key scheduling process takes 6 clocks, and thus Kvlr and BSY are set to 1 and 0
in CLK8, respectively. The sequencer goes to the idling state “IDLE READY.”

**CLK9:** By asserting Drdy=1, the 128-bit input (plaintext) and the control signal EncDec are stored into internal registers. The encryption process is started in accordance with EncDec=0, and BSY is set to 1.

**CLK10~32:** The encryption takes 23 clocks, and thus it is completed in CLK32. The output data (ciphertext) is output from Dout and Dvld is set to 1 only in the 23rd clock (i.e., CLK32). The sequencer is set to “IDLE READY,” and BSY goes to 0 in CLK32.

**CLK33:** By asserting Drdy=1, the next operation is started. The 128-bit input (ciphertext) and the control signal EncDec are stored into internal registers. The decryption process is started in accordance with EncDec=1, and BSY is set to 1.

**CLK34~57:** The decryption also takes 23 clocks, and thus it is completed in CLK57. The output data (plaintext) is output from Dout and Dvld is set to 1 only in the 23rd clock (i.e., CLK56). The sequencer is set to “IDLE READY,” and BSY goes 0 in CLK57.

**Fig. 5** Timing Chart

5. **Reference**