**FPGA Based Implementation Of Encryption And Decryption Of 16-Bit Data Using VHDL**

Paresh Kumar Pasayat, Janmejaya Rout, Kodanda Dhar Sa

**Abstract:** This paper deals with the techniques used for encoding of 16-bit data known as encryption and decoding of the encoded data to get the original 16-bit data known as decryption. The encryption and decryption processes are carried out by taking two chip codes each having 8-bit. The goal has been achieved by doing different operations on the input data, chip code and encoded data. This technique can be used in the field of wireless communication with error free transmission and reception of data from source and destination.

**Key words:** Encryption, decryption, chip code VHDL (Very High speed Integrated Circuit Hardware Description Language).

1. **Introduction:** The transformation of original data into an unreadable format is known as called encryption and the process of reversing it back to a readable form is known as decryption. The encryption and decryption of 16-bit data can be used in the digital communication technology for error free data transmission from one point to another point and to provide security to the data to achieve the confidentiality in data transmission.

2. **Description & Architecture:** In order to encrypt the 16-bit data, the following steps are used.

   **Step 1:** The input data having 16-bit (say I) is divided into two parts (I_1 & I_2) each having 8-bit.

   **Step 2:** The two data I_1 & I_2 are used to perform the XOR operations with the two chip codes (C_1 & C_2) resulting two outputs such as E_XOR_1 & E_XOR_2, i.e.
   
   \[ E_{XOR\_1} = I_1 \oplus C_1 \]
   \[ E_{XOR\_2} = I_2 \oplus C_2 \]

   **Step 3:** The results obtained after performing the xor operations are undergone for 1’s complement operations resulting E_C_1 & E_C_2.
   
   \[ E_{C\_1} = \neg E_{XOR\_1} \]
   \[ E_{C\_2} = \neg E_{XOR\_2} \]

   **Step 4:** The complemented outputs are used to perform left rotate operation where the datas are rotated by one bit towards left giving rise to two outputs such as E_LR_1 & E_LR_2.
   
   \[ E_{LR\_1} = E_{C\_1}(0) \& E_{C\_1}(7 \text{ downto } 1) \]
   \[ E_{LR\_2} = E_{C\_2}(0) \& E_{C\_2}(7 \text{ downto } 1) \]

   **Step 5:** The results having two 8-bit datas obtained after the left rotate operations are appended to get the 16-bit encoded data known as cipher data (say CIPHER_DATA).
   
   \[ \text{CIPHER\_DATA} = E_{LR\_1} \& E_{LR\_2} \]
In order to decrypt the 16-bit cipher data, the following steps are used.

Step 1: The cipher data having 16-bit is divided into two parts (C_DATA_1 & C_DATA_2) each having 8-bit.

Step 2: The outputs obtained after the bit separation are used to perform right rotate operations where the datas are rotated by one bit towards right giving rise to two outputs such as D_RR_1 & D_RR_2.

\[
D_{RR,1} = C_{DATA,1} \text{ (6 downto 0)} & C_{DATA,1} \text{ (7)} \\
D_{RR,2} = C_{DATA,2} \text{ (6 downto 0)} & C_{DATA,2} \text{ (7)} 
\]

Step 3: The results obtained after performing the right rotate operations are undergone for 1’s complement operations resulting D_C_1 & D_C_2.

\[
D_{C,1} = \text{not } D_{RR,1} \\
D_{C,2} = \text{not } D_{RR,2} 
\]

Step 4: The two datas resulted from the complementary operations are used to perform the XOR operations with the two chip codes which should be same as the chip codes used during the encryption of data resulting two outputs such as D_XOR_1 & D_XOR_2, i.e.

\[
D_{XOR,1} = D_{C,1} \text{ xor } C_{1} \\
D_{XOR,2} = D_{C,2} \text{ xor } C_{2} 
\]

Step 5: The results having two 8-bit datas obtained after the xor operations are appended to get the 16-bit decoded data known as original data.

\[
\text{DECODED\_DATA} = \text{ D\_XOR\_1 } \& \text{ D\_XOR\_2 } 
\]
3. Simulation Results:

- Figure 3: Simulation Result of bit separator

- Figure 4: Simulation Result of xor operation using chip_code_one

- Figure 5: Simulation Result of xor operation using chip_code_two

- Figure 6: Simulation Result of 1’s complement operation corresponding to chip_code_one

- Figure 7: Simulation Result of 1’s complement operation corresponding to chip_code_two

- Figure 8: Simulation Result of left rotate operation corresponding to chip_code_one

- Figure 9: Simulation Result of left rotate operation corresponding to chip_code_two

- Figure 10: Simulation Result of bit append operation giving rise to the desired 16-bit encrypted data
Figure 11: Simulation Result of encryption process showing the output of encryption block

Figure 12: Simulation Result of bit seperator

Figure 13: Simulation Result of right rotate operation corresponding to chip_code_one

Figure 14: Simulation Result of right rotate operation corresponding to chip_code_two

Figure 15: Simulation Result of 1’s complement operation corresponding to chip_code_one

Figure 16: Simulation Result of 1’s complement operation corresponding to chip_code_two

Figure 17: Simulation Result of xor operation using chip_code_one

Figure 18: Simulation Result of xor operation using chip_code_two

Figure 19: Simulation Result of bit append operation giving rise to the desired 16-bit decrypted data
Figure 20: Simulation Result of decryption process showing the output of decryption block

4. Conclusion:

The work done in this paper based on the encryption and decryption of 16-bit data can be used in the digital communication technology for error free data transmission from one point to another point and to protect the confidentiality of sensitive data in transmission. Further research is being carried out to implement this design in the processor unit of computer.

5. References:


