Implementation Project

Implement a hardware cipher / uncypher function on a microprocessor-based system.

Algorithm description

Precaution

The following method is not estimated accurate enough for today cryptanalysis methods, but it permits to demonstrate your abilities on microprocessors systems. The algorithm you will implement is named RC4 and is still used to encrypt data in the WEP (Wired Equivalent Privacy) and WPA (Wifi Protected Access) security protocol you can still use when you cypher data on WIFIand was used in many commercials' products.

Today AES256 or DES are considered more secure, but even if the main principle is quite similar, their descriptions are more complex and consequently their implementation uses too much time for this small project. Moreover, RC4 is particularly efficient for software implementation, but your hardware implementation should permit to improve data rate.

Algorithm generalities

This algorithm is used for dataflow operation. In a cryptographic system, the sender has to generate a key and exchange it with the receiver. This operation uses an asymmetric algorithm such as RSA which implies calculation on numbers as large as 1000 bits (300 digits in decimal) and uses prime numbers. This first part is slow but permits to share keys between sender and receiver. When it's done, some dataflow algorithms like RC4 permit to cypher/uncypher raw data at full speed.

Size:

Your implementation will cipher data byte after byte. You'll use a 5 bytes key. (40 bits), but it could handle a longer key without modification.

Description:

Mainly RC4 cypher data by applying a XOR gate between the data to encrypt, and a byte generated from the key. This byte came from an array of 256 bytes generated from the key. The content of this array changes each time you encrypt a new byte. With this principle, encryption and decryption if the same.

You will find the algorithm description, on Wikipedia: https://en.wikipedia.org/wiki/RC4

SUBJECT

At the beginning you will read and understand the C code provided in the project:

Copy, unzip and load the project RC4 provided on Moodle.
Q1) Measure the time used by NIOS processor to cypher a block of 256 bytes. (Opt OFF / Debug ON)

Now you will create a hardware IP to cypher a block of data.

Concerning the algorithm, you should have seen it has two parts:

- Generating an array of 256 bytes from the key (whatever is the size of the key)
- Cypher an decipher data. During this part the previous defined array is shuffled.

We propose to implement only the second part of the algorithm, because the first part could be done slowly by software.
But, the second part of the algorithm permits to cypher data on stream and your hardware implementation should increase the obtained data rate.

Q2) Create a new IP based on the file “HeadCypher.v”, then write a program to perform a write-read loop and measure the time between two read (or two write). Explain the interface.

The previous IP refers to the file “MD_CipherUser.v” which is on the directory rc4\IP. You have to add this file to your project.
Then in the following development you will modify this file “MD_CipherUser.v”. This method permits to compile the project directly by Quartus one the IP is insert in the system once (and generated once) by Plateform Designer.

Q3) Write and test an IP (by modifying the file “MD_CipherUser.v”) to receive this 256-bit array generated from the key and store it. This Ip should also permit to reread the recorder values.

Q4) Implement a state machine able to:

- Read consecutively two values in the array and swap them accordingly to the algorithm.
- Read the value useable to perform the XOR and apply it to the data to cipher

Q5) Realize the full IP able to cypher (uncypher) a data.

Q6) Write a program to use you IP. Measure the time used to cypher a block of data.

Q7) Modify your program and measure the new performances with: Optimization Level = O3 and debug level 0. Explain the program differences.

Q8) Change the processor for a NIOS2f. Report the time for the software ciphering. Then test your iP, Is it working? Explain.

Q7) Propose an organization to increase the obtained data rate (on NIOS2e).

Q8) [Optional] Implement a new version of your IP increasing the previous data rate, and measure the result.

Q9) [Optional] Use a NIOS 2f and rewrite your program consequently.