# PRESENT CIPHER (32 BIT INPUT)
documentation

Krzysztof Gajewski
and opencores.org

www.opencores.org

gajos@opencores.org

# Change History

| Rev. | Chapter | Date | Description | Reviewer |
|---|---|---|---|---|
| 0.1 | all | 2014/09/05 | First draft | K. Gajewski |
| 0.2 | all | 2014/09/16 | Some small corrections with the text, typos, etc. | K. Gajewski |


All rights reserved - ©2014 Krzysztof Gajewski and opencores.org

# Contents



All rights reserved - ©2014 Krzysztof Gajewski
and opencores.org

# 1 Introduction

Present is ``ultra-lightweight´´ block cipher developed by A. Bogdanov et al. and proposed in 2007 [1]. It uses 64 bit data block and 80 bit or 128 bit key. This cipher consists of 32 rounds, during which:

- round key is added to plaintext
- plaintext goes through sBoxes (substitution boxes)
- plaintext after sBoxes goes through pLayer (permutation layer)
- round key is updated

After that, ciphertext feeds out the output. Briefly algorithm was shown in Fig. 1. In this project

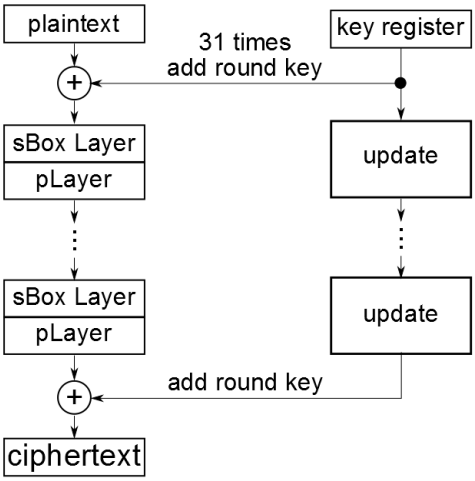


Figure 1: Briefly block scheme of the PRESENT block cipher

Present block cipher works with 80 bit key. Target was Xilinx® Spartan 3E XC3S500E [2] on Spartan 3E Starter Board [3] made by Digilent®. In comparison with "plain" Present cipher project, this core was modified to take 32 bit word at input (plus control word). Output is also 32 bit.

**NOTE:**

This is rather "historical" project and is not recommended for future use.


All rights reserved - ©2014 Krzysztof Gajewski and opencores.org

# 2 Interface

Top level component of the Present component with 32 bit input was shown in Fig. 2. All inputs and outputs are synchronous except `reset` signal and sampled at rising edge of the clock. Type for all signals is `STD_LOGIC` or `STD_LOGIC_VECTOR`.

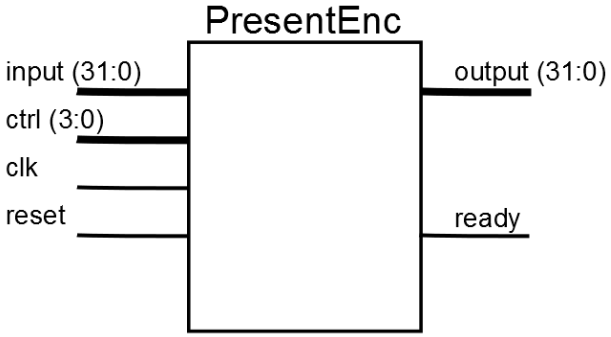


Figure 2: Top level of the Present component with 32 bit input

| Signal name | Width | In/Out | Description |
|---|---|---|---|
| `input` | 32 | in | input data - both key and plaintext. |
| `ctrl` | 4 | in | control bus for sending commands to the core. |
| `clk` | 1 | in | clock signal for the component |
| `reset` | 1 | in | *asynchronous* reset signal. |
| `output` | 32 | out | output data - ciphertext. |
| `ready` | 1 | out | signal informing about end of encoding process.<br>"0" - wait until end of data encoding.<br>"1" - end of the encoding process, output data available. |

Table 1: Input/Output signals of the Present component with 32 bit input

All rights reserved - ©2014 Krzysztof Gajewski
and opencores.org

# 3 State machine workflow

Overall internal structure of the Present component with 32 bit input is similar to the structure shown in [1]. To conform 64 bit plaintext, 80 bit key and 32 bit output data, multiplexer-like blocks was added to fit data. Additionally, control logic was added in the state machine. It was shown in Fig. 3.

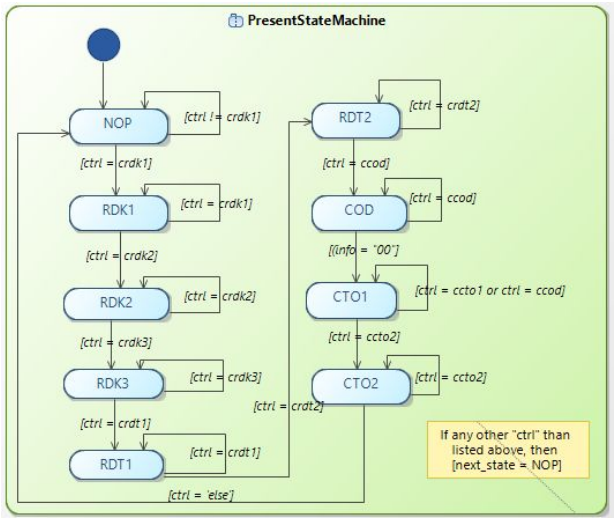


Figure 3: State machine of the Present component

State machine consist of nine states `NOP`, `RDK1`, `RDK2`, `RDK3`, `RDT1`, `RDT2`, `COD`, `CTO1`, `CTO2`. `NOP` is the default state after resetting the core. This state is active as long as control bus (`ctrl`) don't have `crdk1` command at the input.

`RDKx` states are responsible for reading the key from the input. They are changing when suitable command appears at the `ctrl` input (3). When another commands appear, the state is changing to the `NOP` state. When command are left constant, given state is not changing.

`RDTx` states are responsible for reading the plaintext from the input. They are changing when suitable command appears at the `ctrl` input (3). When another commands appear, the state is changing to `NOP` state. When command are left constant, given state is not changing.

During the `COD` state encoding process start. If encoding process ends (after 32 clock cycles, `info = "00"` signal from the counter), state machine automaticly goes to the `CTO1` state. When commands another than `ccod` appear, the state is changing to the `NOP` state. When command are left constant encoding process is working.

`CTOx` states are responsible for sending the ciphertext to the output. They are changing when suitable command appears at the `ctrl` input (3). When another commands appear, the state is changing to the `NOP` state. When command are left constant, given state is not changing.


All rights reserved - ©2014 Krzysztof Gajewski and opencores.org

# 4 FPGA implementations

The component has only been verified on a Xilinx® Spartan 3E XC3S500E FPGA in FG320 package and synthesized with Xilinx ISE 14.2. Appropriate setup files was prepared with the use of ISE Project Navigator, but Makefile scripts was also written. Suitable files was stored in `./32BitIO/syn/XC3ES500/` directory. Implementation in FPGA device **was not done** in this project (due to rather historical issues and nonconventional build of these core). Makefile was tested in Windows 8 with the use of Cygwin for 64-bit Windows.

Synthesis results was given in Fig. 4

| Xilinx ®Spartan 3E XC3S500E FPGA in FG320 package | | | |
|---|---|---|---|
| **Parameter** | **Used** | **Available** | **Utilization** |
| Number of Slices | 313 | 4656 | 6% |
| Number of Slice Flip Flops | 262 | 9312 | 2% |
| Number of 4 input LUTs | 460 | 9312 | 4% |
| Number of bonded IOBs | 71 | 232 | 30% |
| Number of GCLKs | 1 | 24 | 4% |
| Minimum period | 4.250 ns | - | - |
| Maximum Frequency | 235 MHz | - | - |

Table 2: Synthesis results for Spartan 3E XC3S500E

Possible change in used FPGA device may be possible in steps given below[1]:

1. Copy `./32BitIO/syn/XC3ES500/` directory to another one like `./32BitIO/syn/YOUR_FPGA_SYMBOL/`
2. Go to `./32BitIO/syn/YOUR_FPGA_SYMBOL/` directory.
3. In `PresentEnc.xst` file modify the line `-p xc3s500e-5-fg320` to `-p YOUR_FPGA_CODE`
4. In `Makefile` file modify the line `PLATFORM=xc3s500e-fg320-5` to `PLATFORM=YOUR_FPGA_CODE`

[1] This solution was not tested and is based on my own observations.

All rights reserved - ©2014 Krzysztof Gajewski
and opencores.org

# 5 Simulation

Self-checking test bench were provided to the components used for Present encoder. In case of whole Present with 32 bit input encoder this test bench was not self-checking. This is due to historical character of this project. They are stored in `./32BitIO/bench/vhdl` directory. Suitable configuration files and Makefile used for running test bench was stored in `./32BitIO/sim/rtl_sim/bin` directory. Appropriate test vectors was taken from [1].

Makefile was prepared to make "manual run" of tests. If You want to perform it without gui, remove `-gui` option in Makefaile.


All rights reserved - ©2014 Krzysztof Gajewski and opencores.org

# 6 Troubleshooting

During work with Windows 8 64-bit and and Xilinx® ISE 64-bit some problems may occur:

1. Xilinx may be unable to open projects in Project Navigator.
2. When you run `make` in Cygwin and perform testbench it would be unable to open ISIM gui.
3. When you run ISIM gui (*.exe test bench file) it hangs out or anti virus protection opens.

To solve problems listed above you have to perform steps listed below:

1. You have to rename libraries `libPortabilityNOSH.dll` to `libPortability.dll` from `nt64` directories (http://www.gadgetfactory.net/2013/09/having-problems-installing-xilinx-ise-on-windows-8-64bit-here-is-a-fix-video-included/)
2. Firstly, install Cygwin X11 (http://stackoverflow.com/questions/9393462/cannot-launch-git-gui-using-cygwin-on-windows)
3. Temporary switch off anti virus protection.

All rights reserved - ©2014 Krzysztof Gajewski
and opencores.org

# 7 License and Liability

Copyright ©2013 Authors and OPENCORES.ORG

This source file may be used and distributed without restriction provided that this copyright statement is not removed from the file and that any derivative work contains the original copyright notice and the associated disclaimer.

This source file is free software; you can redistribute it and-or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This source is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this source; if not, download it from http://www.opencores.org/lgpl.shtml

Xilinx, Spartan3E is registered trademark of Xilinx Inc. 2100 Logic Drive, San Jose CA USA


All rights reserved - ©2014 Krzysztof Gajewski and opencores.org

# References

[1] A. Bogdanov, L. Knudsen, G. Leander, C. Paar, A. Poschmann, M. Robshaw, Y. Seurin, and C. Vikkelsoe, "Present: An ultra-lightweight block cipher," in *Cryptographic Hardware and Embedded Systems - CHES 2007*, ser. Lecture Notes in Computer Science, P. Paillier and I. Verbauwhede, Eds. Springer Berlin Heidelberg, 2007, vol. 4727, pp. 450–466. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-74735-2_31

[2] Xilinx. (2014, Feb.) Spartan-3e fpga family data sheet. [Online]. Available: http://www.xilinx.com/support/documentation/data_sheets/ds312.pdf

[3] Digilent. (2014, Feb.) Spartan 3e starter board. [Online]. Available: http://www.digilentinc.com/Products/Detail.cfm?Prod=S3EBOARD

All rights reserved - ©2014 Krzysztof Gajewski and opencores.org