DESIGNING A CO-PROCESSOR FOR IMPLEMENTATION OF SNOW 3G ALGORITHM.

MUGDHA JADHAO
IIT ROORKEE (SECOND YEAR, ELECTRICAL)
UNDER THE GUIDANCE - PROF. SANDEEP K. SHUKLA
OUTLINE

- SNOW 3G
- ARCHITECTURES IMPLEMENTED
  - DESIGN FLOWCHART
  - TIMING DIAGRAMS
  - DESIGN REPORT
- EEA1 ALGORITHM
- EIA1 ALGORITHM
- FINAL IMPLEMENTATION
- FUTURE WORK
- MAJOR LEARNINGS FROM THE PROJECT
PROBLEM CONTEXT

- 4G wireless networks operate entirely on the TCP/IP architectural suite.
- The LTE/SAE main core network architecture demands that security functions should be optimally and efficiently embedded into the overall architecture.
- Thus, cryptographic algorithms play an important role in the LTE/SAE architecture.
- In order to provide security protection in LTE, 3GPP has specified three sets of secure algorithm to protect confidentiality and integrity for data communication, which are 128-EEA1/128-EIA1, 128-EEA2/128-EIA2, and 128-EEA3/128-EIA3 respectively.
WHY HARDWARE OVER SOFTWARE?
MERIT OF THE PROJECT

- What is FPGA?
- Little about FPGA Architecture.
- Why FPGA over ASIC or Microcontroller.
# Week Report

<table>
<thead>
<tr>
<th>WEEK</th>
<th>WORK DONE</th>
</tr>
</thead>
</table>
| **WEEK 1** | 1. Elementary differences between Microcontroller, Asic and FPGA.  
3. Difference between hardware and software languages.  
4. Difference between simulator and tool.  
5. SOC design flow.  
6. Hardware and Software codesign basics. |
| **WEEK 2** | 7. Xilinx Vivado Tool.  
8. Designed an 8 bit ALU and implemented on Artix 7 FPGA.  
9. Setting up Microblaze on FPGA Artix 7 board and adding custom IP to the design.  
10. ASIC design flow.  
12. Innovus tool. |
WEEK 3
13. Implemented ALU ASIC in Genus.
14. Implemented pipelining techniques to reduce critical path in ALU.
15. Implemented clock gating to reduce power consumption.
16. Implemented optimized ALU as custom IP on the FPGA Artix 7 board.
17. Designed Multiplier using Carry Save Adder algorithm.
18. Designed Modulo 8 counter.
19. Studied Cordic IP architecture.

WEEK 4
20. Learnt Verilog HDL.

WEEK 5
25. Design.
27. Simulation.
29. Implementation.

WEEK 6

WEEK 7

WEEK 8
SNOW 3G

- SNOW 3G is a word-oriented stream cipher that generates a sequence of 32-bit words under the control of a 128-bit key and a 128-bit initialization variable.
- SNOW 3G is used as the core component of both UEA2 and UIA2.
- Main components of Snow 3g:
  - MULx
  - MULxPOW
  - 32x32 bit S-box Sr
  - 32x32 bit S-box Sq
  - Linear Feedback Shift Register (LFSR)
  - Finite State Machine (FSM)
Snow 3g operated in two modes:

**Initialization mode**

1. \( v = (s_0 32 \oplus s_2) \oplus \text{MUL}_\alpha(s_0,0) \oplus (s_{11} 32 \oplus \text{DIV}_\alpha(s_{11},3) \oplus F) \)
2. \((s_0, s_1, \cdots, s_{14}) \leftarrow (s_1, s_2, \cdots, s_{15})\)
3. \( s_{15} = v \)

**Clocking FSM**

1. \( F = (s_{15} \oplus R_1) \oplus R_2 \)
2. \( r = R_2 (R_3 \oplus s_5) \)
3. \( R_3 = S_2(R_2) \)
4. \( R_2 = S_1(R_1) \)
5. \( R_1 = r \)
SNOW 3G Keystream Mode

1. \( v = (s_0 \oplus 32 \oplus 8) \oplus \text{MUL}_\alpha(s_0,0) \oplus s_2 \oplus (s_{11} \oplus 32 \oplus 8) \oplus \text{DIV}_\alpha(s_{11},3) \)

2. \((s_0, s_1, \cdots, s_{14}) \leftarrow (s_1, s_2, \cdots, s_{15})\)

3. \(s_{15} = v\)
ARCHITECTURES IMPLEMENTED

TWO ARCHITECTURES:

- Architecture I: Used lookup table for recursive function.
- Architecture II: Designed recursive function modules instead using lookup tables.

Folded architecture were used in both designs to reduce number of hardware resources.
SIMULATION : TIMING DIAGRAM
## DESIGN REPORT

<table>
<thead>
<tr>
<th>Platform</th>
<th>FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slices (without BRAM)</td>
<td>890</td>
</tr>
<tr>
<td>Frequency</td>
<td>287.52 MHz</td>
</tr>
<tr>
<td>Static Power</td>
<td>0.092 W</td>
</tr>
<tr>
<td>Dynamic Power</td>
<td>0.184 W</td>
</tr>
</tbody>
</table>

Critical path: In generate keystream module.
# LITERATURE SURVEY TABLE

<table>
<thead>
<tr>
<th>Sr.no</th>
<th>Platform</th>
<th>Year</th>
<th>Slices</th>
<th>Throughput (Mbps)</th>
<th>Area (kGE)</th>
<th>Frequency (MHz)</th>
<th>Throughput/Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FPGA</td>
<td>2016</td>
<td>-</td>
<td>10358</td>
<td>2,56</td>
<td>339</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>FPGA</td>
<td>2014</td>
<td>164</td>
<td>11712</td>
<td>-</td>
<td>366</td>
<td>71.4</td>
</tr>
<tr>
<td>3</td>
<td>FPGA</td>
<td>2012</td>
<td>188</td>
<td>10304</td>
<td>-</td>
<td>322</td>
<td>33.8</td>
</tr>
<tr>
<td>4</td>
<td>FPGA</td>
<td>2012</td>
<td>356</td>
<td>12036</td>
<td>-</td>
<td>376</td>
<td>33.8</td>
</tr>
<tr>
<td>5</td>
<td>FPGA</td>
<td>2009</td>
<td>164</td>
<td>11712</td>
<td>-</td>
<td>366</td>
<td>-</td>
</tr>
<tr>
<td>6</td>
<td>ASIC</td>
<td>2008</td>
<td>-</td>
<td>7968</td>
<td>-</td>
<td>249</td>
<td>-</td>
</tr>
</tbody>
</table>
EEA1 ALGORITHM

EEA1 FLOWCHART

ARCHITECTURE OF EEA1
SIMULATION OF EEA1 ALGORITHM
EIA1 ALGORITHM

**EIA1 FLOWCHART 1**

- **COUNT-1** || **FRESH** || **DIRECTION** || **0...0** || **000000000000000000000000** || **DIRECTION** || **000000000000000000000000**
- **IV1** || **IV2** || **IV3** || **IV4**
- **IK** || **K2** || **K3** || **K4** || **K5**
- **SNOW 3G**
- **z1** || **z2** || **z3** || **z4** || **z5** || **z6**
- **P** || **Q** || **OTP[0] ... OTP[31]**

**EIA1 FLOWCHART 2**

- **MESSAGE** || **0 ... 0**
- **M0** || **...** || **Md-2**
- **LENGTH**
- **Md-1**
- **z2** || **z3**
- **Q**
- **z3** || **z4**
- **MUL**
- **c0** || **c1** || **...** || **c31** (left 32 bits)
- **OTP[0] ... OTP[31]**
- **MAC-1**
FINAL IMPLEMENTATION ON FPGA
FUTURE WORK

- Hardware - software codesign in integrity algorithm for MUL recursive function.
- Use of BRAM to store static and dynamic lookup tables.
- Optimization of SNOW 3G core - to implement pipelining.
- Implement Architecture II.
MAJOR LEARNING FROM THE PROJECT

- Verilog Hardware Description Language.
- SOC Hardware-Software Design.
ACKNOWLEDGEMENT

● A heartfelt gratitude to Professor Sandeep K. Shukla for providing me with this opportunity.

● Special thanks to my mentors Asan Basiri Sir and Rourab Paul Sir for guiding me in the project.
THANK YOU!
REFERENCES