

# CS 577: Project Report

<i>Project Number :</i>	4
<i>Group Number:</i>	4
<i>Name of the top modules:</i>	crypto_sign_keypair
<i>Link for GitHub Repo:</i>	<a href="https://github.com/deepraj88/project4">github.com/deepraj88/project4</a>

Group Members	Roll Numbers
1. Kapil Kukreja	194101024
2. Anasuya Jana	194101004
3. Megha Moon	194101033
4. Priyanka Khare	194101023

# INTRODUCTION

## PHASE-1

### 1. Running the algorithm

The screenshot shows a C/C++ development environment with the following interface elements:

- Top Bar:** File, Edit, Project, Solution, Run, Window, Help.
- Toolbars:** Standard and Debugging tools.
- Left Sidebar:** Debug (selected), Explorer.
- Middle Area:** A terminal window showing the output of the application run.
- Right Sidebar:** Variables, Breakpoints, Interactive Console, Expressions.
- Bottom Area:** Console, Tasks, Problems, Executables, Debugger Console.

**Code Editor Content:** newtest\_sign\_2.c

```
47 //unsigned char      *m, *sm, *m1, *m_rsp, *sm_rsp;
48 unsigned long long mlen, smlen, mlen1, smlen_rsp;
49 int count;
50 int done;
51 unsigned char pk[CRYPTO_PUBLICKEYBYTES], sk[CRYPTO_SECRETKEYBYTES];
52 int ret_val;
53 int loop,sk_count=0,pk_count=0;
54 int error_count = 0;
55 int count_loop = 0;
56
57 char fn_rsp[64];
58 char fn_init[64];
59 FILE *fp_rsp;
60 FILE *fp_init;
61 unsigned char pk_rsp[CRYPTO_PUBLICKEYBYTES], sk_rsp[CRYPTO_SECRETKEYBYTES];
62
63 unsigned char Key_enc[32];
64 unsigned char V_enc[16];
65 int reseed_counter_enc;
66 unsigned char Key_dec[32];
67 unsigned char V_dec[16];
```

**Terminal Output:**

```
<terminated> <terminated, exit value: 0> qTesla_I_round2.prj.Debug [C/C++ Application] csim.exe
Known Answer Tests PASSED.
```

## 1.2 Synthesis screenshot

Synthesis screenshot showing Utilization Estimates and Vivado HLS Console output.

**Utilization Estimates**

Name	BRAM_18KDSP48E	FF	LUT	URAM
DSP	-	-	-	-
Expression	-	8	0	513
FIFO	-	-	-	-
Instance	54	51	29155	142966
Memory	12	-	16	4
Multiplexer	-	-	-	865
Register	-	-	454	-
Total	66	59	29625	144348
Available	730	740269200	134600	0
Utilization (%)	9	7	11	107

**Vivado HLS Console**

```

INFO: [RTMG 210-282] Generating pipelined core: 'crypto_sign_keypair_tx'
INFO: [RTMG 210-278] Implementing memory 'poly_uniform_buf_ram (RAM)' using block RAMs.
INFO: [RTMG 210-278] Implementing memory 'crypto_sign_keypakbM_ram (RAM)' using block RAMs.
INFO: [RTMG 210-278] Implementing memory 'crypto_sign_kepancg_ram (RAM)' using distributed RAMs.
INFO: [RTMG 210-278] Implementing memory 'crypto_sign_kepaocq_ram (RAM)' using block RAMs.
INFO: [RTMG 210-278] Implementing memory 'crypto_sign_kepacpcA_ram (RAM)' using block RAMs.
INFO: [RTMG 210-278] Implementing memory 'crypto_sign_keparcU_ram (RAM)' using block RAMs.
INFO: [HLS 200-111] Finished generating all RTL models Time (s): cpu = 00:01:12 ; elapsed = 00:01:20 . Memory (M)
INFO: [VHDL 208-304] Generating VHDL RTL for crypto_sign_keypair.
INFO: [VLOG 209-307] Generating Verilog RTL for crypto_sign_keypair.
INFO: [HLS 200-112] Total elapsed time: 79.73 seconds; peak allocated memory: 491.484 MB.
Finished C synthesis.

```

### 1.3 C/RTL co-simulation screenshot

The screenshot shows the Vivado HLS interface with two main windows.

**Cosimulation Report for 'crypto\_sign\_keypair'**

**Result**

RTL	Status	Latency			Interval		
		min	avg	max	min	avg	max
VHDL	NA	NA	NA	NA	NA	NA	
Verilog	Pass	684726	684726	684726	NA	NA	NA

Export the report(.html) using the [Export Wizard](#)

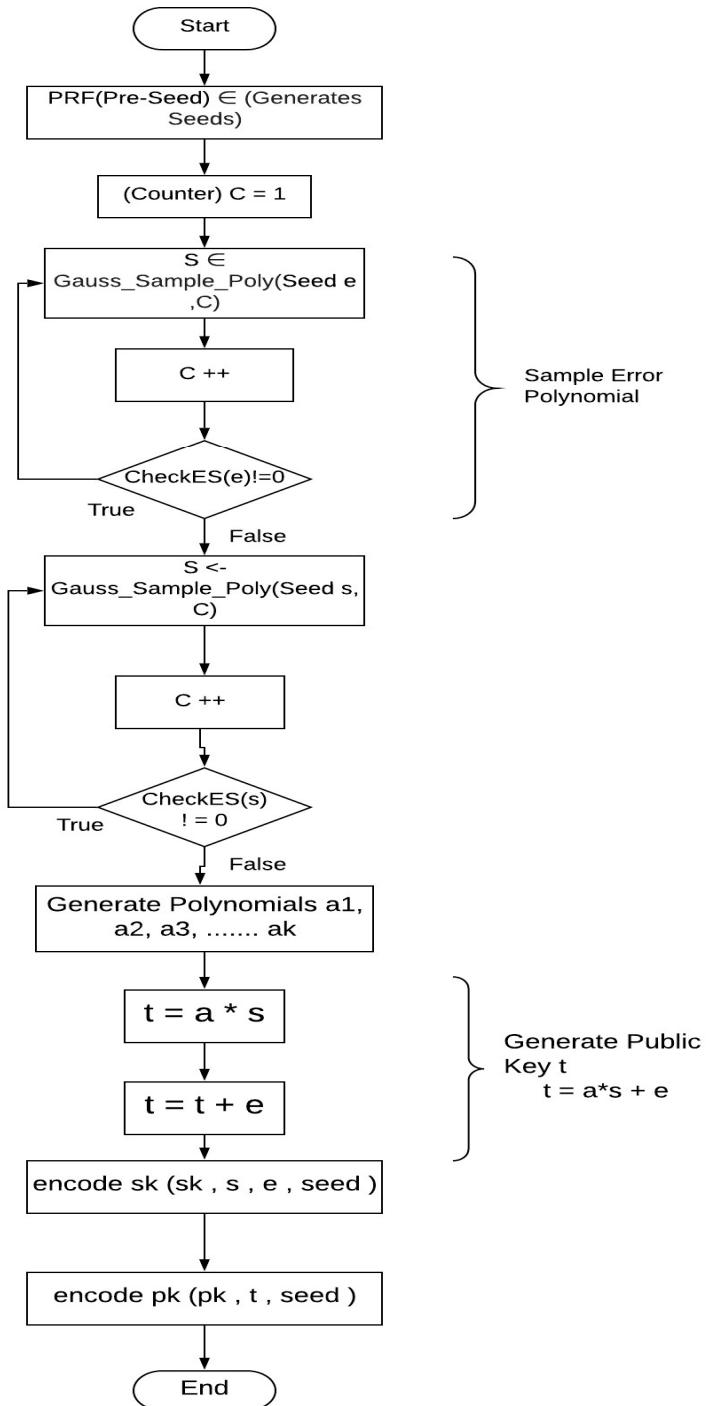
**Vivado HLS Console**

```
Console Errors Warnings DRCs
Vivado HLS Console
## quit
INFO: [Common 17-206] Exiting xsim at Sat Mar 14 18:54:28 2020...
INFO: [COSIM 212-316] Starting C post checking ...
Known Answer Tests PASSED.

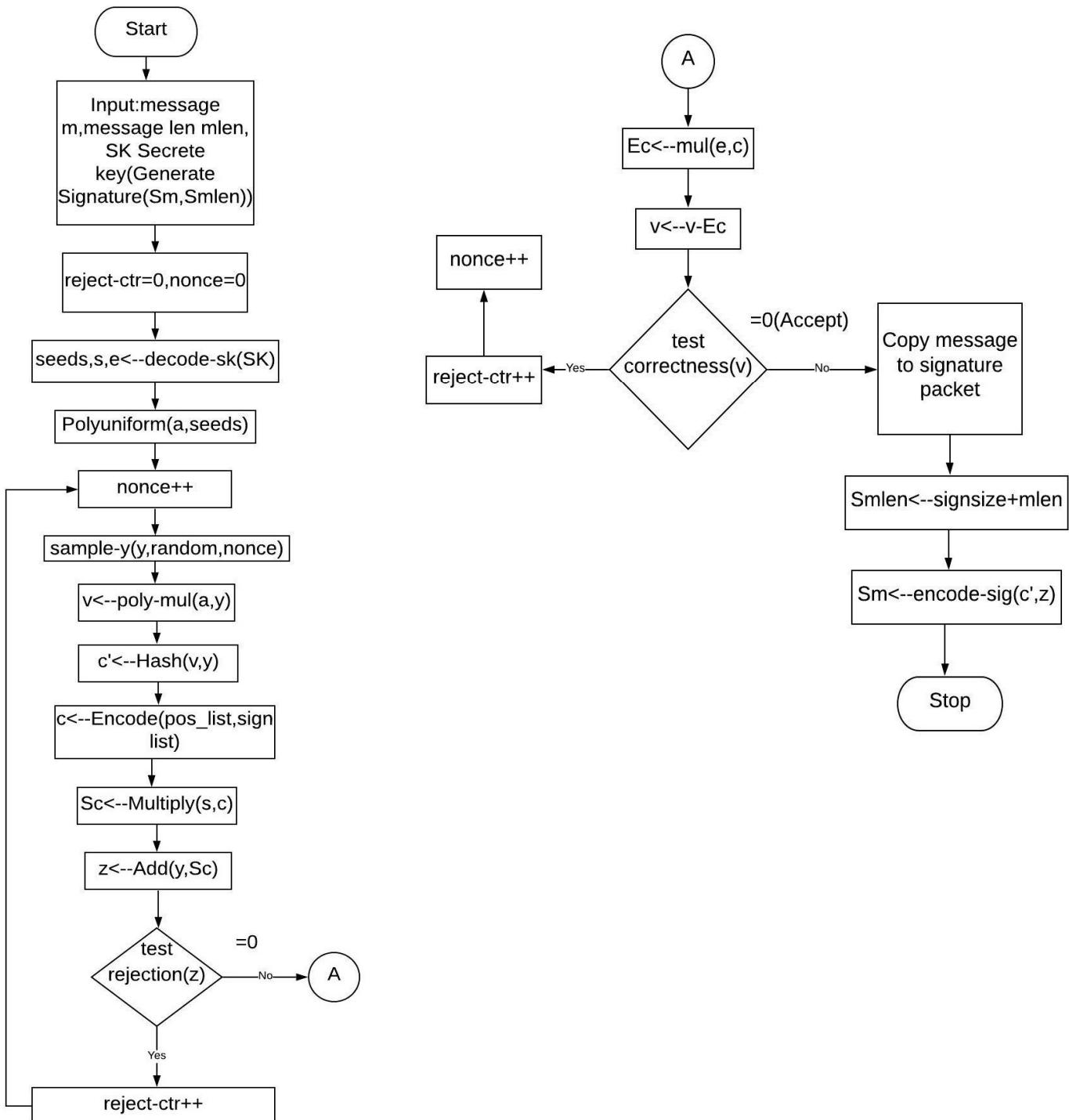
INFO: [COSIM 212-1000] *** C/RTL co-simulation finished: PASS ***
INFO: [COSIM 212-211] II is measurable only when transaction number is greater than 1 in RTL simulation. Otherwise
Finished C/RTL cosimulation.
```

## 2. Flowchart

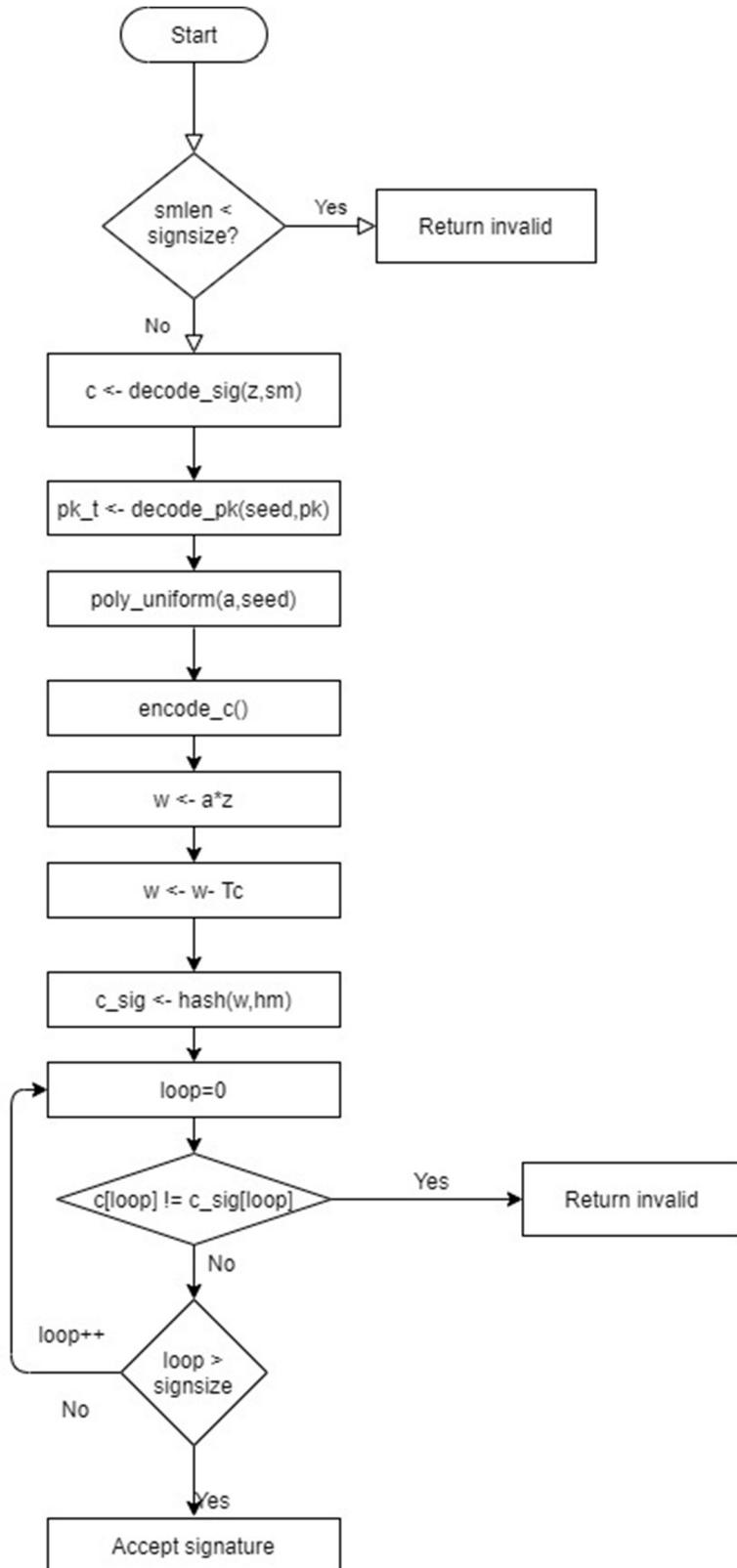
Flowchart 1: Key Generation (Generate Public, Private Key Pair)



Flowchart 2: Crypto Sign (Generate Signature)



Flowchart 3 : Verifies Signature Sm



### 3. Result

FPGA Part	Name of Top Module	FF	LUT	BRAM	DS P	Latency	II
Artix-7 xc7a200t-fbg676-2	crypto-sign-keypair	29625	144348	66	59	684726	N.A.

#### 3.1 Explain the result

The current usage of the various components, such as Flip-flops, Lookup Table, BRAM etc., available on FGPA board is as shown in the table above. The result generated after RTL-C co-simulation shows the latency of the RTL design as on the board with current design and usage of various components.

#### 3.2 Problems and its solution

Major challenge in design is LUT usage which is originally at 107% of the available limit. The goal is to reduce the area by reducing the components usage by targeting each component, critically LUT. Further trying to reduce the area, and then proceeding to reduce latency is a major goal.

## PHASE-2

The target FPGA board is **xc7a200t-fbg676-2**.

Benchmark	Type (Area /Latency)	Resource Utilization				Latency		Major Optimizations
		UT	FF	DSP	RAM	No of Clock cycle/lat- ency	Clock period	
Baseline		144348 (107%)	29625 (11%)	59 (7%)	66 (9%)	6,84,726	12.65ns	NA
Optimization 1	Area	70388 (52%)	18316 (6%)	59 (7%)	94 (12%)	11,30,537	12.65ns	keccakStatePermute() In fips202.c
Optimization 2	Area	43683 (32%)	12121 (4%)	59 (7%)	94 (12%)	6,85,167	12.65ns	Reducing no. of duplicate instances of same function
Optimization 3	Area	28146 (20%)	11267 (4%)	57 (7%)	74 (10%)	12,32,661	12.65ns	As in 2 + some functions rewriting
Optimization 4	Latency	129138 (95%)	76481 (28%)	58 (7%)	67 (9%)	5,98,919	12.65ns	Pipelining in check_ES() Loop unrolling in gauss.c

➔ Optimization 1:

**Target:** To reduce LUT usage

**Observation:** Very high LUT usage in function keccakStatePermute() in fips202.c

**Solution:** There were huge no. of local variables and operations on them. Mapped these to a 2-D Array to shift it to using BRAM instead.

➔ Optimization 2:

**Target:** To reduce LUT usage at original latency

**Observation:** Duplication of instances (such as KeccakSqueezeBlocks) at different hierarchy levels.

**Solution:** Inline recursive the top functions with explicit inline=off on such functions.

➔ Optimization 3:

**Target:** To further reduce area.

**Observation:** Same as 1 in multiple locations

**Solution:** Same as 1.

➔ Optimization 4:

**Target:** To reduce latency

**Observation:** Some loops having very high iteration latency

**Approach:** Reducing iteration latency could reduce latency significantly. Thus, applied Pipeline.

**Problem:** RAM doesn't have enough ports to pipeline the function. As in check\_ES()

**Solution:** Array Partition.