CS 577: Project Report

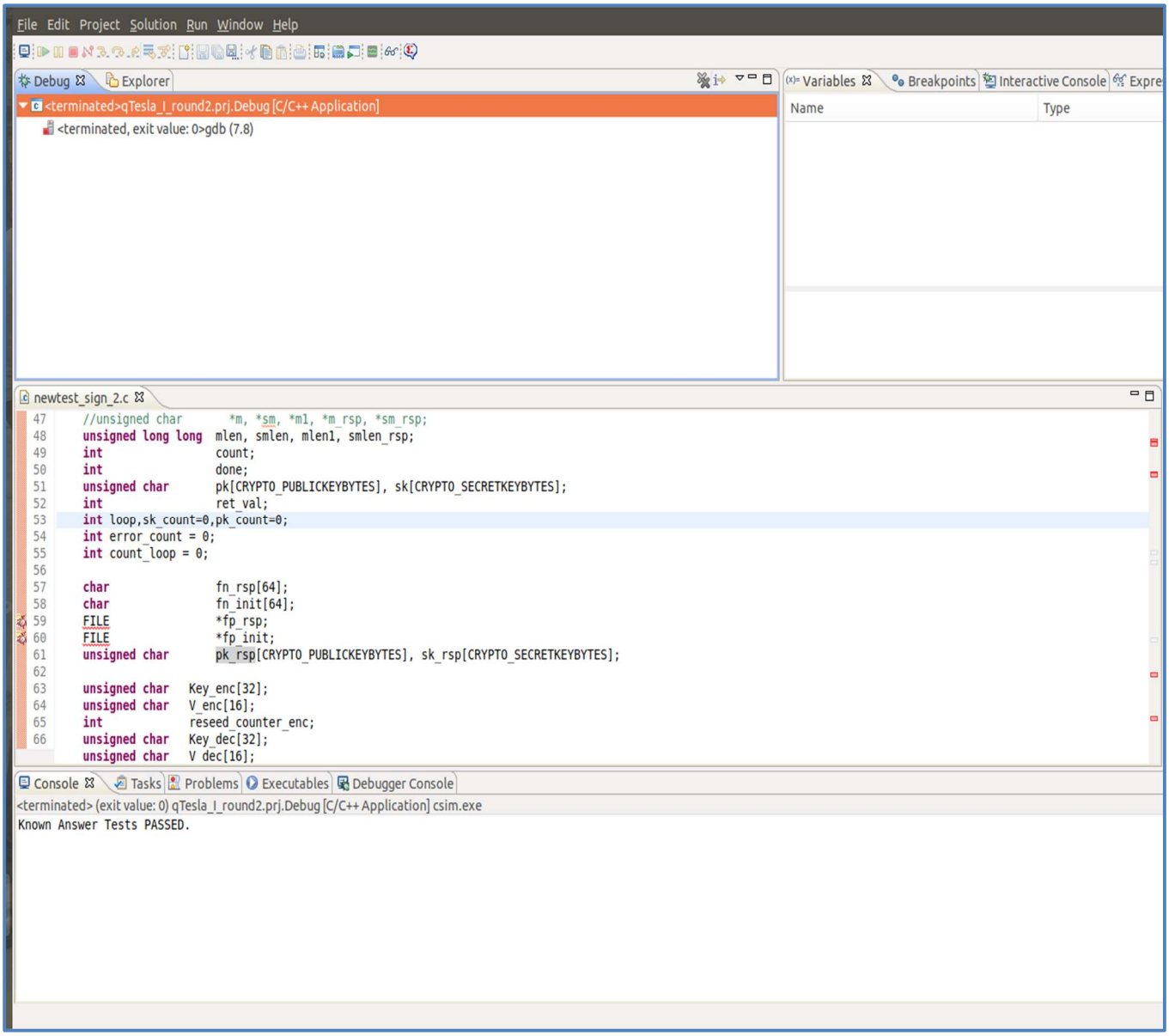
Project Number :	4
Group Number:	4
Name of the top modules:	crypto_sign_keypair
Link for GitHub Repo:	github.com/deepraj88/project4

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

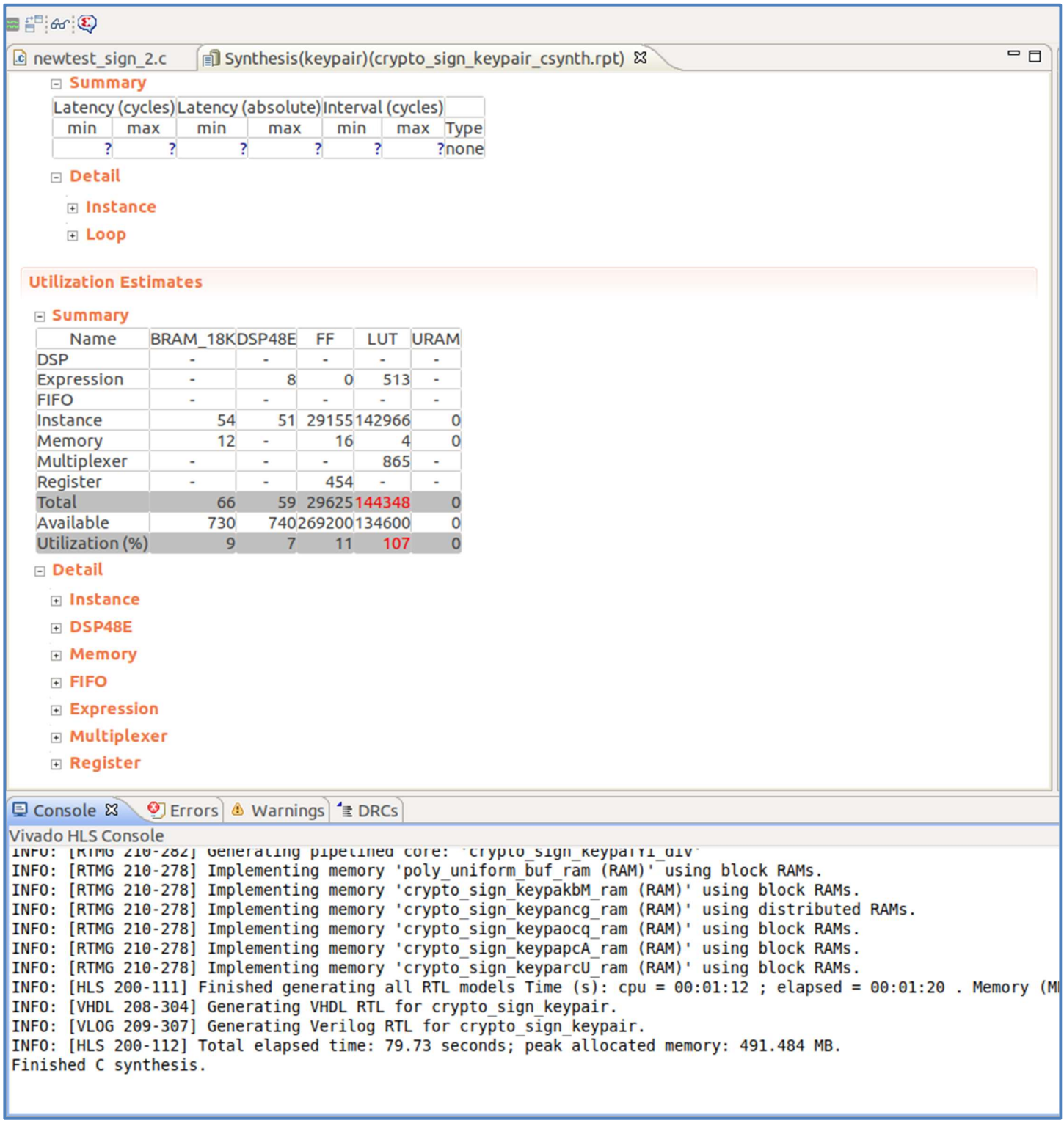
INTRODUCTION

PHASE-1

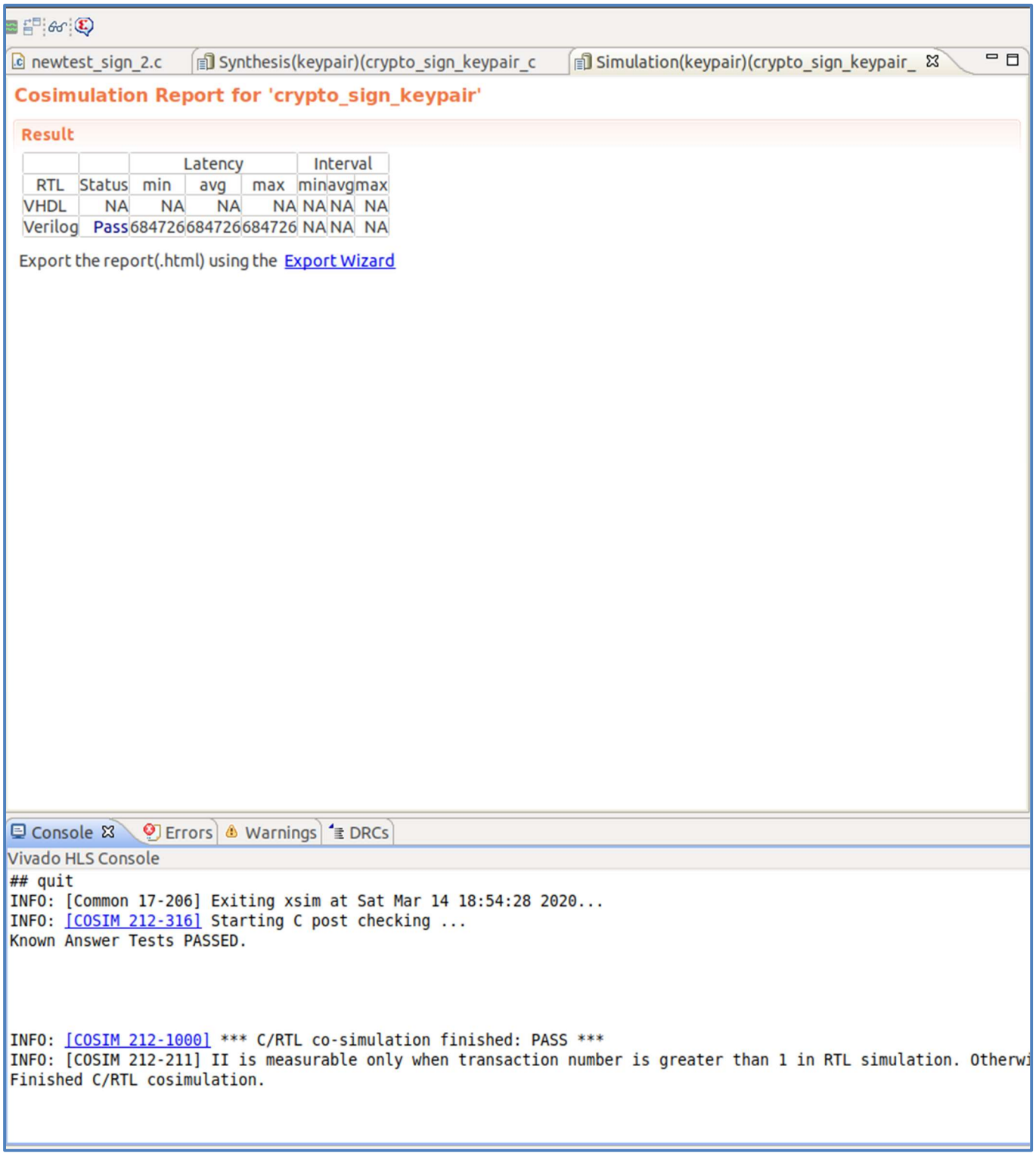
1. Running the algorithm



1.2 Synthesis screenshot

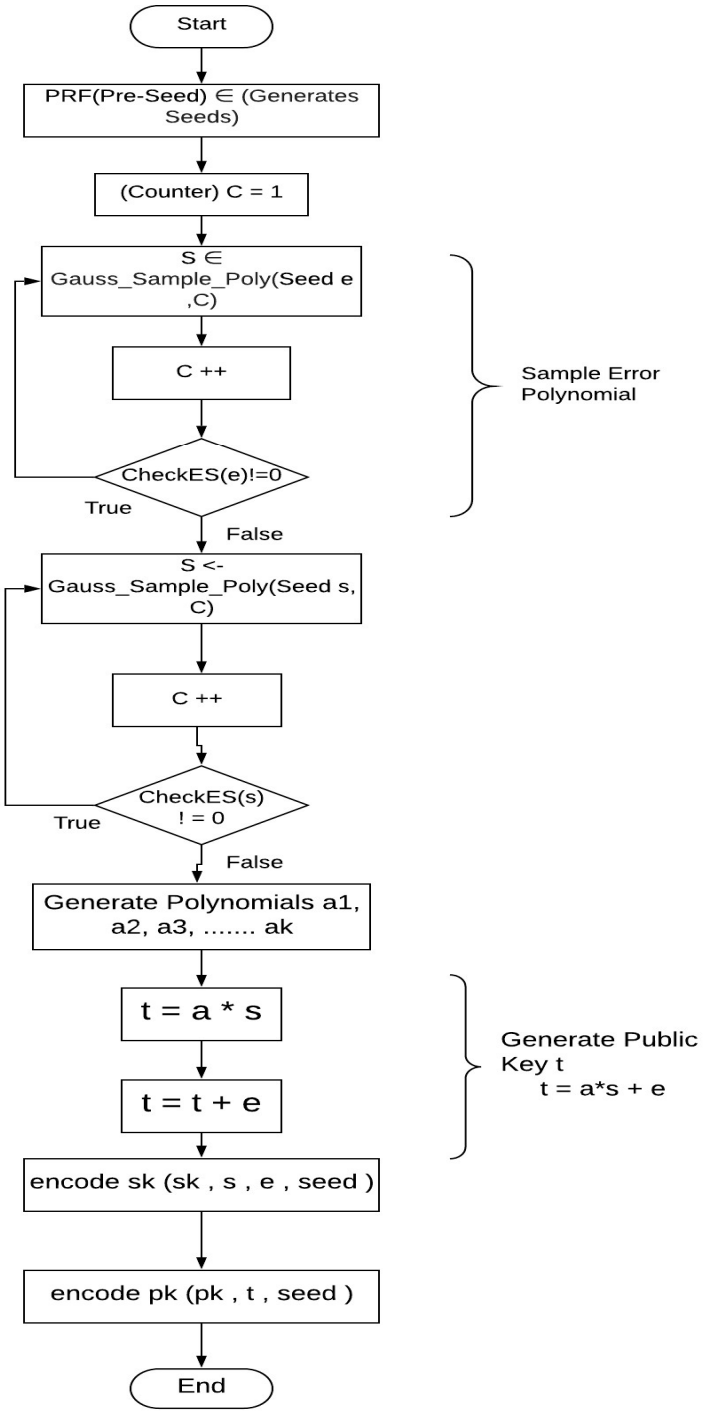


1.3 C/RTL co-simulation screenshot

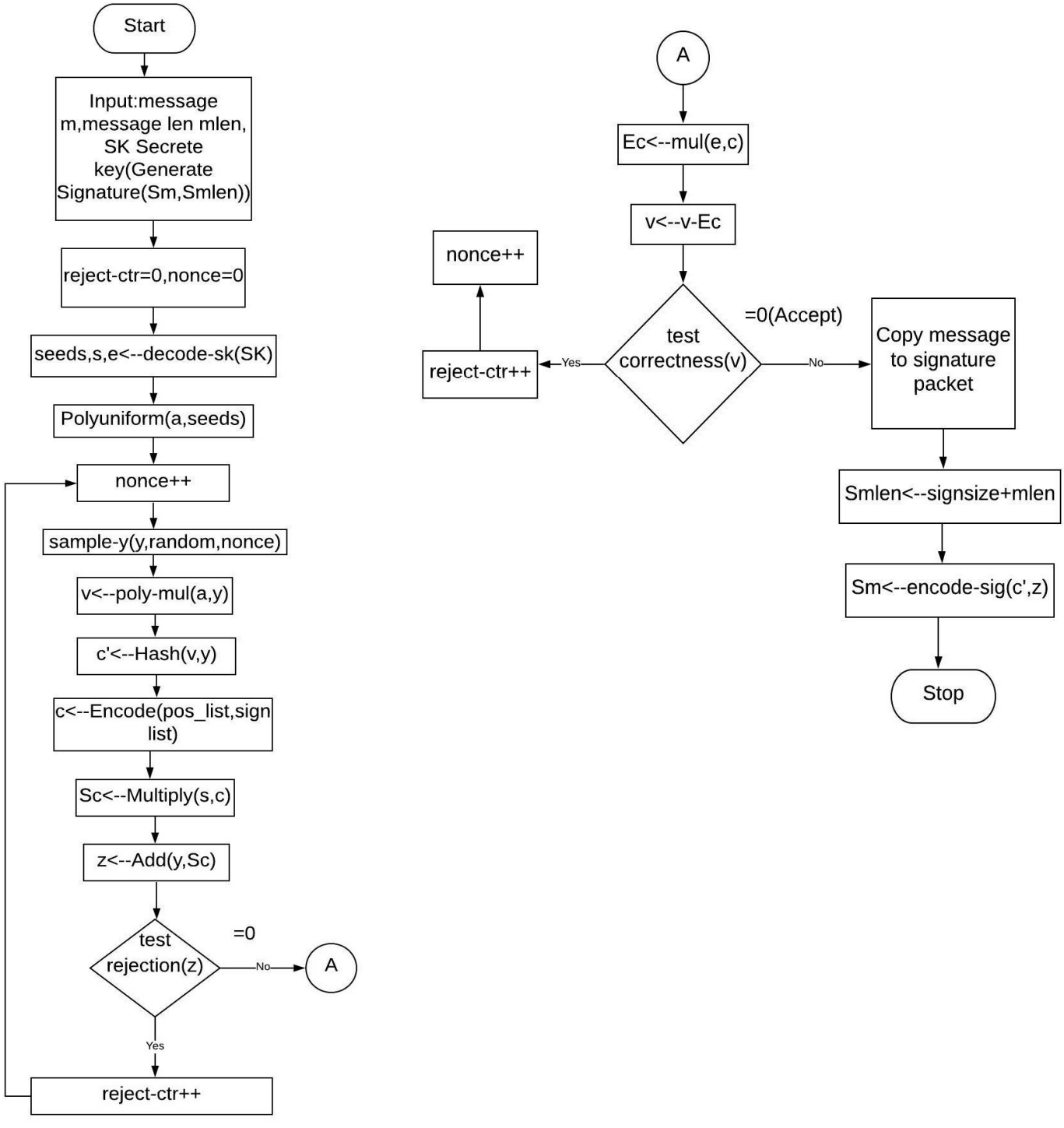


2. Flowchart

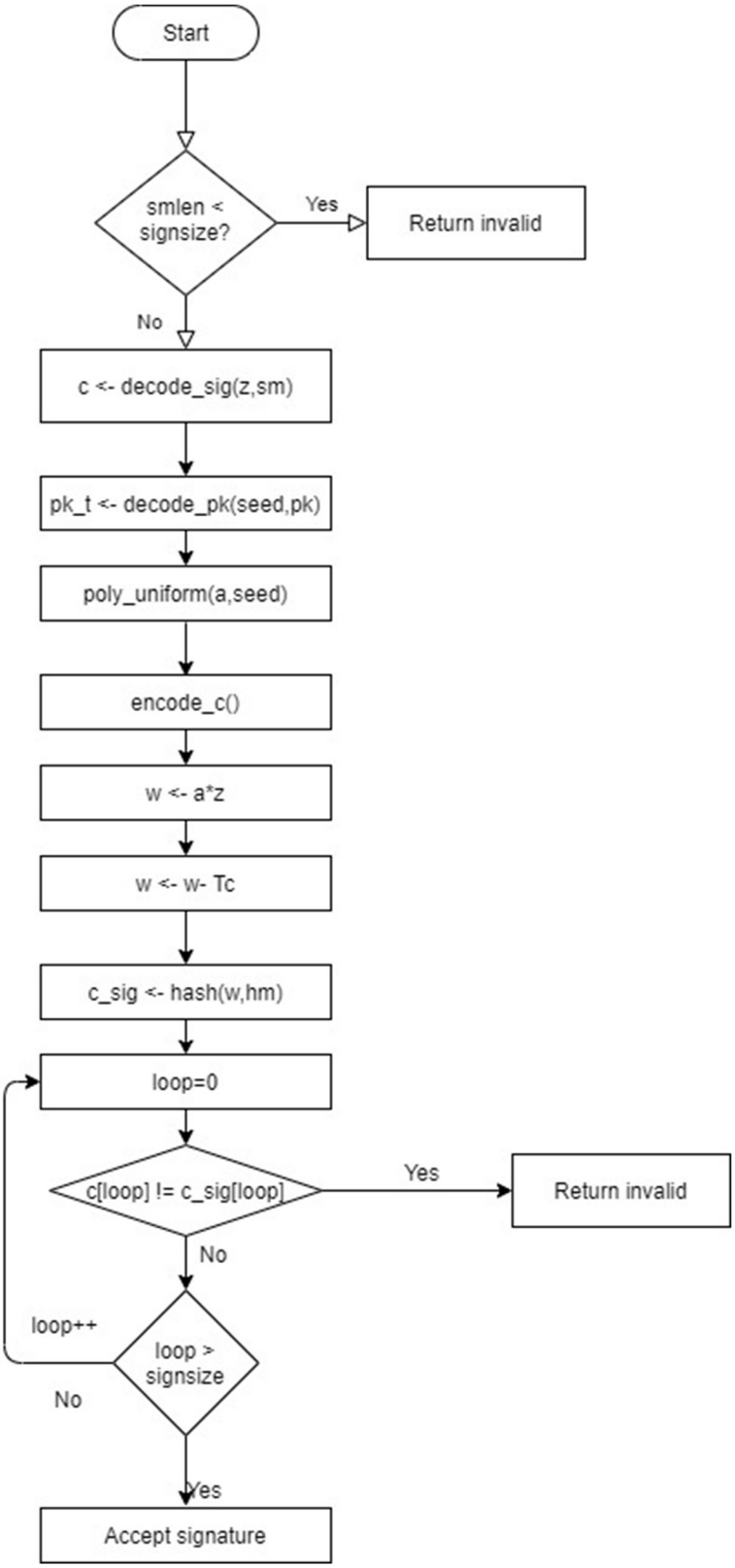
Flowchart 1: Key Generation (Generate Public, Private Key Pair)



Flowchart 2: Crypto Sign (Generate Signature)



Flowchart 3 : Verifies Signature Sm



3. Result

FPGA Part	Name of Top Module	FF	LUT	BRAM	DSP	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
		LUT	FF	DSP	RAM	No of Clock cycle/latency	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 nstances 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.