BENCHMARKING RSA ON SOFTCORE PROCESSORS WITH PYNQ-Z2

Caleb Pai, Winson Zhu, Paul Kim, Hyukjin Jeong

OVERVIEW

- RSA algorithm steps: key generation, encryption, decryption
- Performance bottleneck: modular exponentiation (thousands of modular multiplications)
- RSA private-key operations repeatedly compute: $m = c^d \mod N$

- Montgomery multiplication (avoids division, RSA runs much faster)
- FPGA logic implements $(A \cdot B) \mod N$

RSA ALGORITHM

- RSA is an asymmetric encryption algorithm using a public and a private key

Main Steps:

- Key Generation: Using Public and Private Keys
- Encryption: Sender encrypts the data using Public Key to get ciphertext
- Decryption: Decrypting the cipher test using Private Key to get the original data

HARDWARE PLATFORM

- Board: Pynq-Z2 (Zynq-7020 SoC)

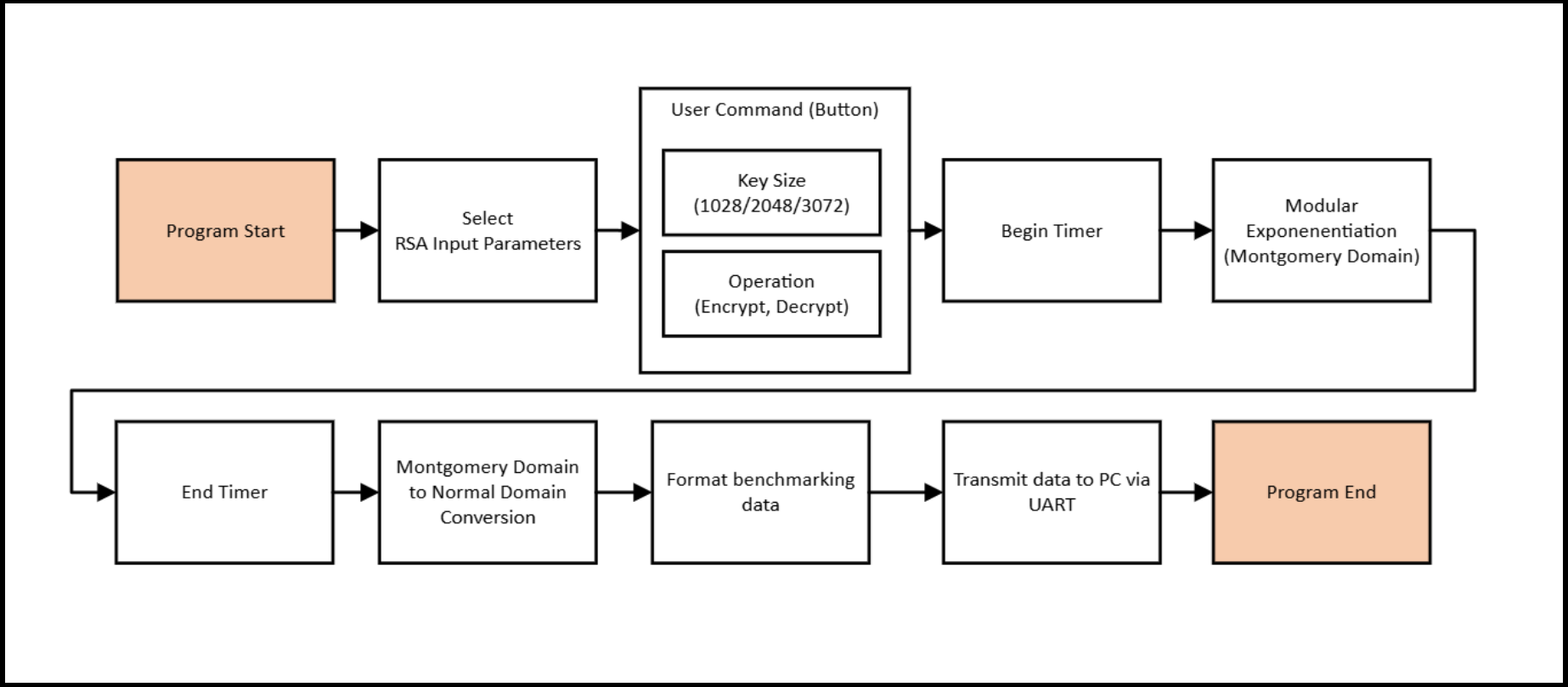
Components:

- Dual-core ARM Cortex-A9 (Processing System)
- Programmable Logic (FPGA fabric)

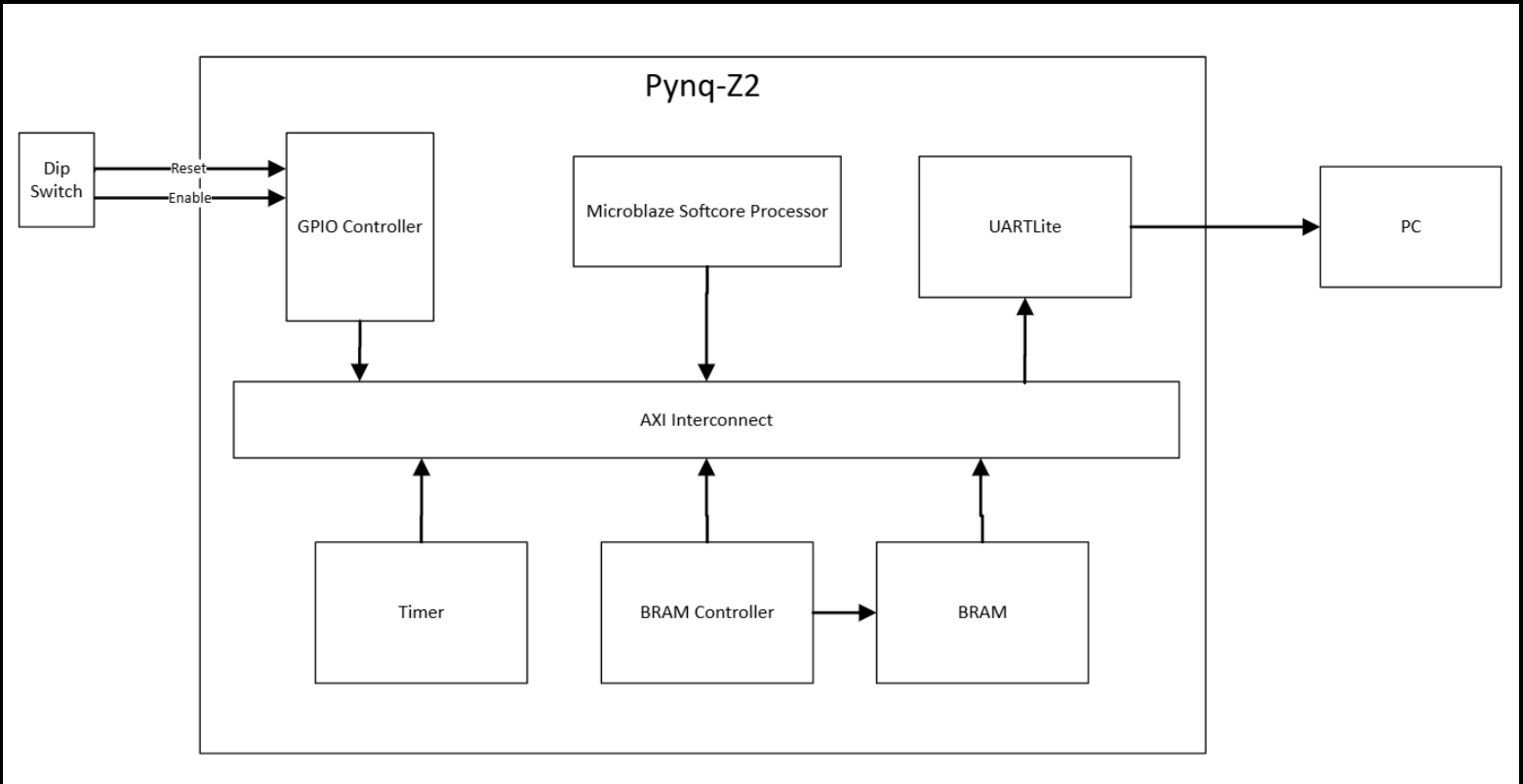
SOFTWARE BASELINE

- RSA executed entirely in software
- Platform: CPU only (no FPGA logic)
- Implemented in C and executed on Pynq-Z2 processing system
- Serves as reference to compare with hardware acceleration
- Measures encryption and decryption time for different key sizes

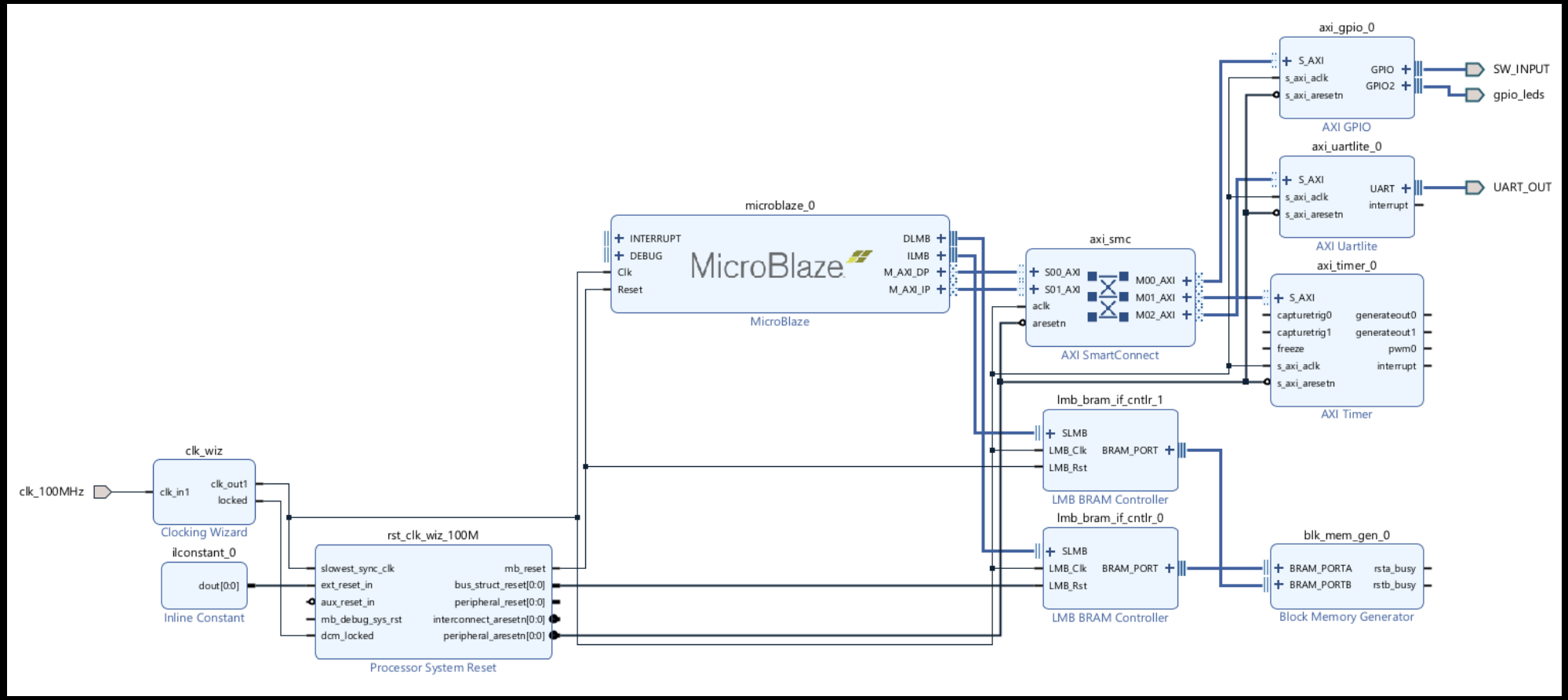
HIGH LEVEL BLOCK DIAGRAM



FPGA IP CORE BLOCK DIAGRAM



VIVADO BLOCK DIAGRAM



BENCHMARK

- Encryption time
- Decryption time
- Speedup = Software time / Hardware time (How much faster HW vs SW)

Example table:

Operation	SW (ms)	HW (ms)	Speedup x
RSA-2048 Encrypt			
RSA-2048 Decrypt			

WORK TO BE DONE

- Complete modular exponentiation function
- Benchmark and extract data
- Demo