BipBip: A Lightweight Crypto Algorithm for Open Source Processors and AI Accelerators

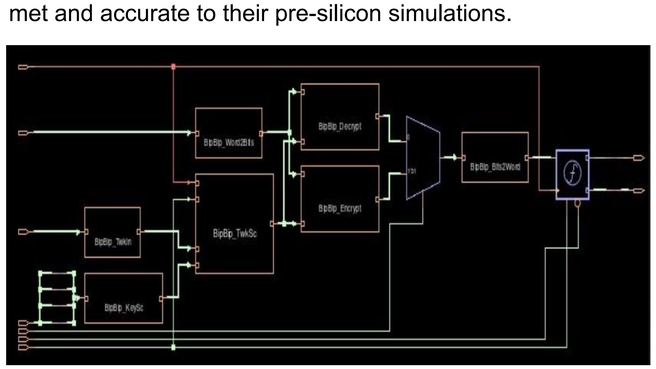
ALEX STOYANOV-ROBERTS, FAVOUR OKONKWO, JOSE DOMINGUEZ CORTEZ, JEREMIAH CONWAY and SHAMEER RAO
DR. KEVIN KORNEGAY, EUGENE DELOATCH ENDOWED PROFESSOR IN CYBERSECURITY ENGINEERING

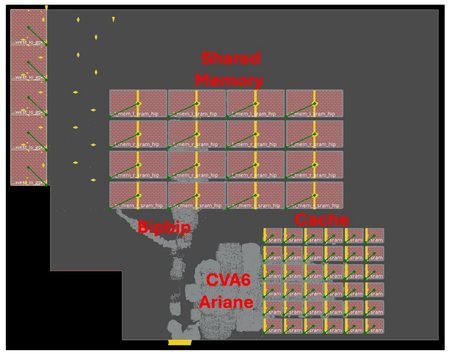
Introduction:

Lightweight cryptography is essential in systems constrained by power, performance, and area (PPA), particularly in IoT and embedded applications. To address these challenges,we present a RISC-V implementation of the BipBip algorithm, a symmetric tweakable block cipher tailored for fine-grained protection of pointer and contextual data in the CPU path of an SoC. Unlike larger-block ciphers like Deoxys-BC, PRESENT, Joltik, and Kisau, which are motivated by the need for pointer encryption, BipBip uses 24-bit blocks, a 40-bit tweak,and a 256-bit key to achieve ultra-low latency. This unique configuration supports contextual encryption without the overhead of nonce management, enabling predictable security in resource-limited systems such as AI on the edge. Our contributions emphasize RTL/FPGA implementation and SoC integration. We demonstrate a 2×2 mm block implementation of BipBip in Intel's 16 nm technology, operating at 300 MHz and with a full encryption-to-decryption path in two cycles. The lightweight primitive can be used in trusted microelectronics, requiring a small footprint, consistent latency, and integration with open-source CPUs and AI acceleration

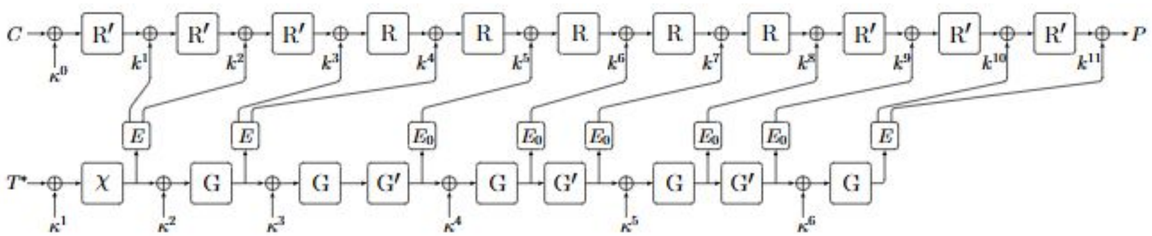
The Challenge:

Morgan State was tasked with developing a secure architecture for IoT devices. In partnership with Intel our team is working towards a RISC-V implementation of Intel's Cryptographic Capability Computing (C3). The literature surrounding C3 references BipBip as the cryptographic algorithm that was used internally, however the existing literature on BipBip only goes as far as synthesis on an advanced node. We decided to complete a hardware tapeout at 16 nm to validate that BipBip in our implementation would be feasible in Silicon, and that the timing constraints were met and accurate to their pre-silicon simulations.





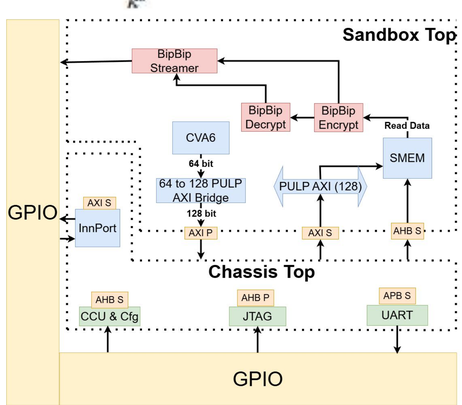
BipBip Schematic inside the Cipher and Layout



BipBip Structure

Project Description and Full Layout:

The fully assembled design includes Intel's IP, which handles communications to and from the system via UART, JTAG, and the Innovation port. Our custom design uses these options to access the system and communicate with external devices.The fully assembled design includes Intel's IP, which handles communications to and from the system via UART, JTAG, and the Innovation port. Our custom design uses these options to access the system and communicate with external devices.



Block Diagram of SOC highlighting custom design and Intel IP

BipBip Algorithm Overview:

Cipher Design Overview:

- Symmetric Encryption/Decryption
- Operates on 24-bit adjustable input.
- Incorporates a custom 24-bit S-box for enhanced non-linearity with 53 bit 2-boxes for the tweak and round functions.
- Five distinct bit-permutation layers (PI1–PI5) for high diffusion - 3 Inverse boxes for decrypt.
- SPN architecture supports efficient implementation in software and hardware.

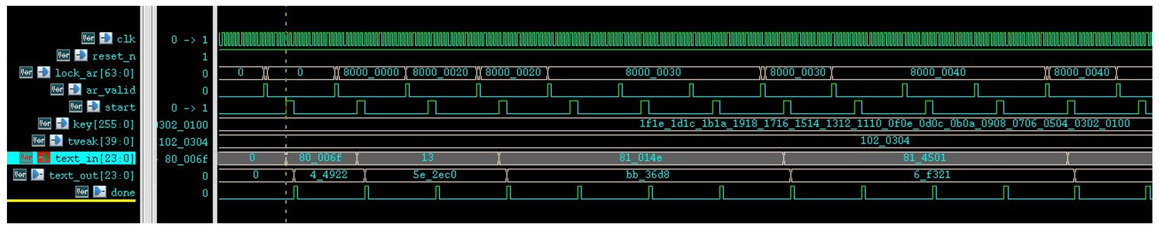
Tweakable Key Schedule:

- 53-bit tweak path processed through CHI and permutation layers.
- Round Key Extraction: RKE0 and RKE1 dynamically generate round keys.
- Inspired by LRW/QARMA constructions.
- Supports block-level domain separation and reuse protection.

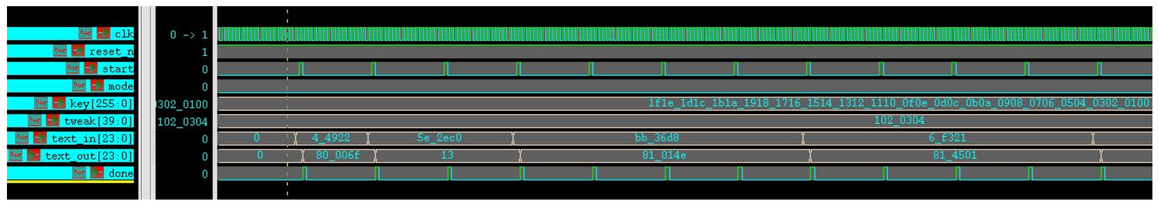
Round Structure

- 12 rounds total: alternating between Core and Shell rounds.
- Core: S-box → Permutation → Linear Mix → Permutation. S-boxes based on BBB and IBBB.
- Shell: Lightweight transformation using S-box + permutation.
- XOR-based round key injection at every stage.

Results & Synthesis:



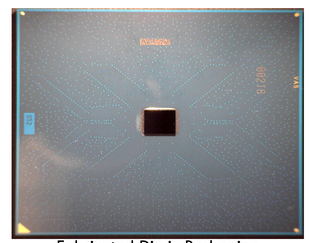
Encryption of SMEM data between set ADDR range



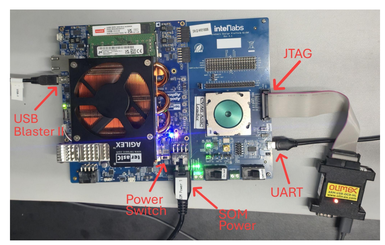
Decryption of SMEM data between set ADDR range

Silicon Bring-Up:

After receiving the Silicon back from the foundry, we began bringup efforts using the apollo agilex FPGA platform, align with a chip socket inside an SOM expansion board. The FPGA is connected to the chip via JTAG and is able to preload custom firmware into the SMEM on the Silicon. This allows us to perform read and write operations on the regions of memory that BipBip is set to encode and decode. We have these outputting to the screen via UART, and a logic analyzer tapped into the GPIO pins, along with an oscilloscope to measure the voltage of the chip when BipBip is being used, along with observing both the encrypted and decrypted outputs.



Fabricated Die in Packaging



Bring-Up platform used to test silicon