CRYPTO HASH ENGINE WITH BRUTE-FORCE NONCE SEARCH

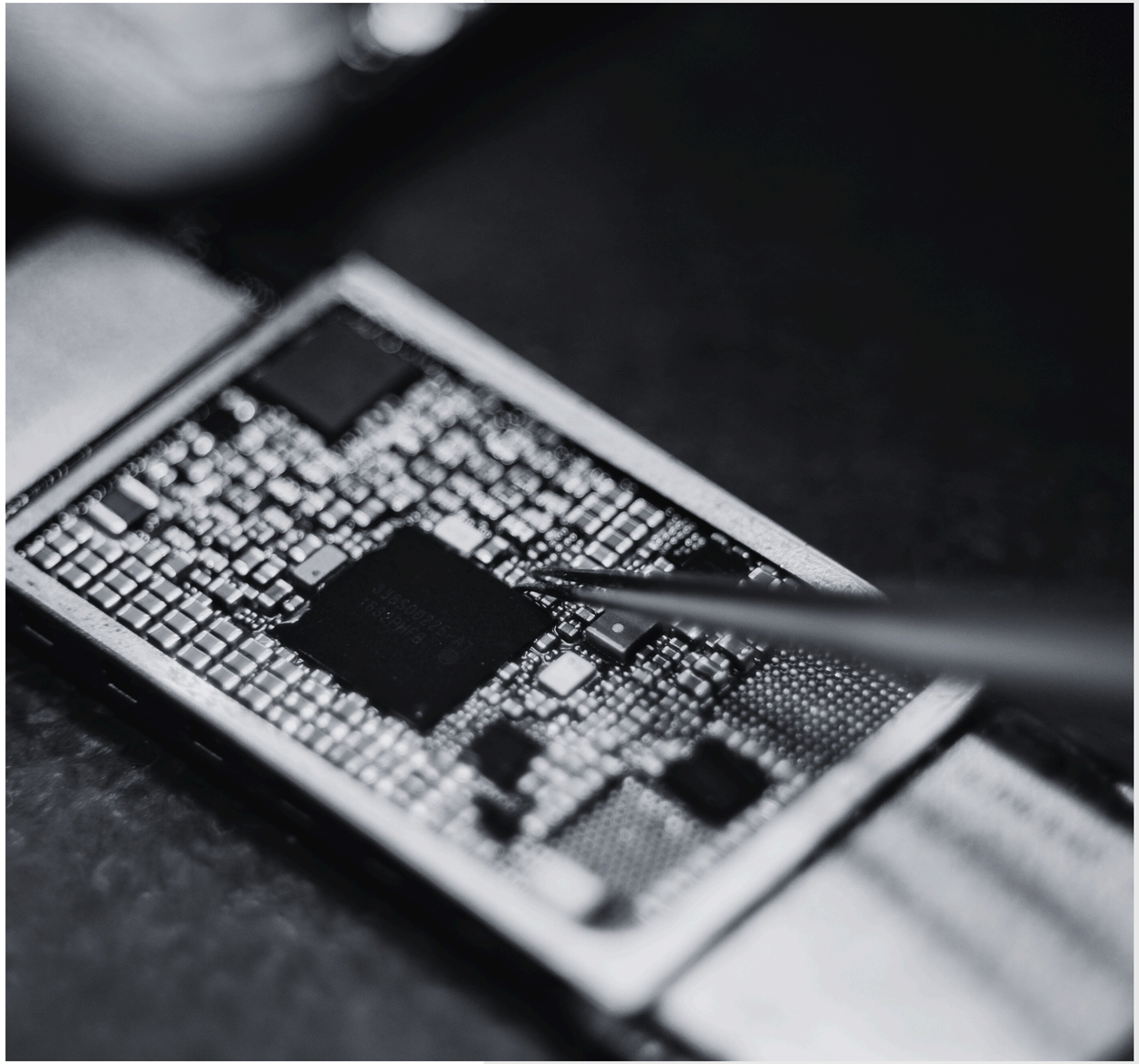
GROUP 7

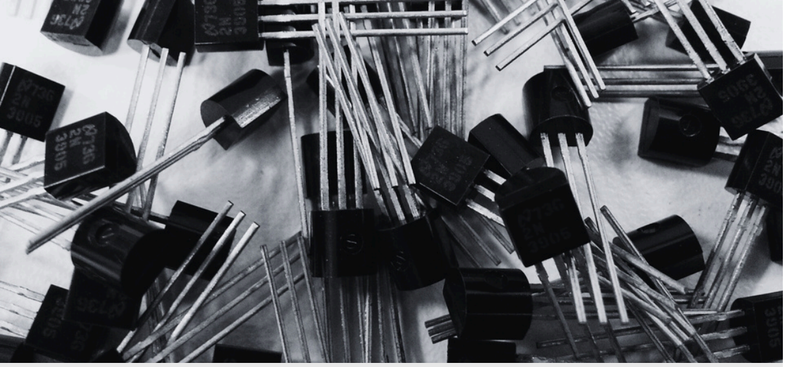
- Muhammad Fairuz Dzaki
- Muhammad Naufal Gilardino
- Raihan herhumadzib
- Raul fadhilah

BACKGROUND

CRYPTO HASH PLAYED AN IMPORTANT ROLE IN MODERN DIGITAL SECURITY SYSTEMS, INCLUDING BLOCKCHAIN MINING. ONE OF THE APPLICATIONS IS BITCOIN MINING, WHICH USES A PROOF-OF-WORK (POW) MECHANISM THAT REQUIRES MINERS TO REPEATEDLY COMPUTE HASH VALUES UNTIL A RESULT SMALLER THAN A PREDEFINED TARGET IS FOUND. IN REAL WORLD, THIS PROCESS IS USUALLY IMPLEMENTED USING SHA-256 ALGORITHM AND IS OPTIMISED USING ASIC HARDWARE.

HOWEVER, IMPLEMENTING FULL SHA-256 WILL BE TOO COMPLEX AT THIS TIME.

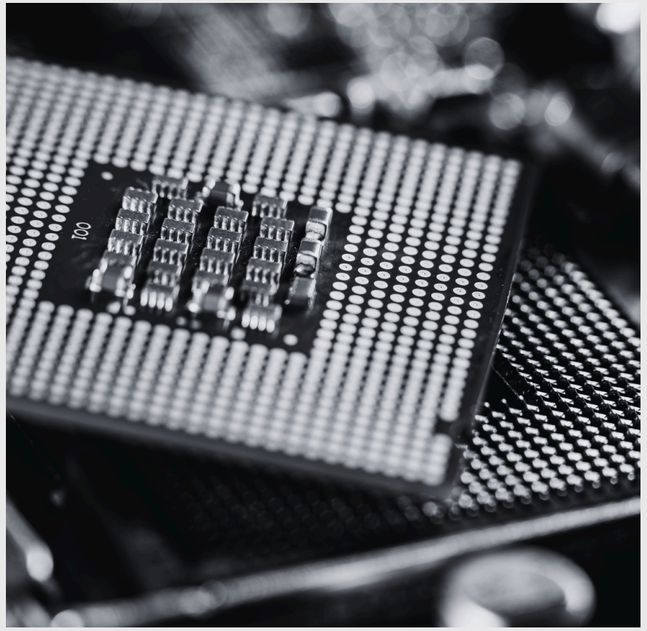


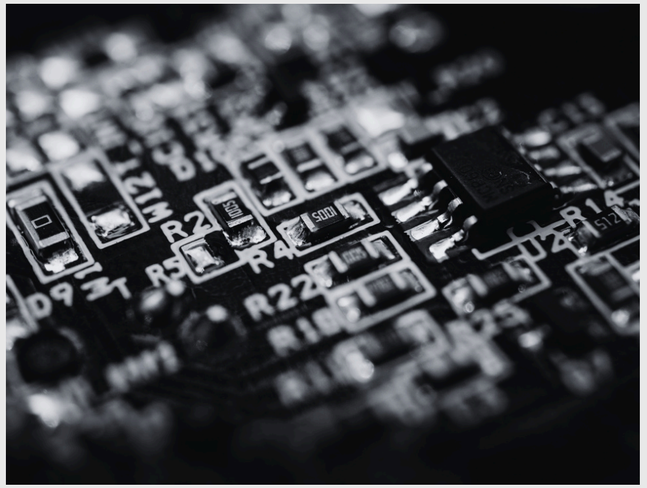


GIGGLING PLATYPUS CO.

OBJECTIVES

- DESIGN STATE MACHINE FOR MINING CONTROLLERS
- PERFORM MULTIPLE MODULES (BEHAVIORIAL, DATAFLOW, STRUCTURAL,FSM, FUNCTIONS) INTO ONE SYSTEM
- UNDERSTANDING THE WORK CONCEPT FOR BLOCKCHAIN SYSTEM
- IMPLEMENTING A MINING ALGORITHM IN VHDL WITHIN THE MODULAR ARCHITECTURE





IMPLEMENTATION

THE HASH CORE IMPLEMENTS A LIGHTWEIGHT HASH FUNCTION FOR VHDL SIMULATION AND THIS IMPLEMENTATION IS A SIMPLIFIED ALGORITHM THAT MAINTAINS ITS CORE CRYPTOGRAPHIC PRINCIPLES WHILE BEING PRACTICAL FOR HARDWARE DESCRIPTION AND TESTING.

THE HASH FUNCTION OPERATES ON A STATE BASED ARCHITECTURE WITH MIXING OPERATIONS. AT INITIALIZATION THE STATE REGISTER IS LOADED WITH UPPER 32 BITS OF INPUT DATA,ESTABLISHING THE INITIAL MIXING STATE. THE ALGORITHM THEN PUTS THROUGH 8 COMPRESSION ROUNDS EACH CONTRIBUTED TO THE FINAL HASH VALUE THROUGH A SERIES OF TRANSFORMATIONS.

CONTROLLER FSM

IDLE :

```
WHEN IDLE =>
   HASH_START <= '0';
   NONCE_EN <= '0';
   NONCE_LOAD <= '1';
   DONE_FOUND <= '0';
   LATCH_NONCE <= '0';
   IF START_SEARCH = '1' THEN
      STATE <= LOAD;
   END IF;
```

LOAD :

```
WHEN LOAD =>
   NONCE_LOAD <= '1';
   HASH_START <= '1';
   NONCE_EN <= '0';
   LATCH_NONCE <= '0';
   STATE <= HASHING;
```

HASHING :

```
WHEN HASHING =>
   HASH_START <= '0';
   NONCE_LOAD <= '0';
   IF HASH_BUSY = '1' THEN
      STATE <= HASHING;
   ELSIF HASH_DONE = '1' THEN
      STATE <= CHECK;
   END IF;
```

FOUND :

```
WHEN FOUND =>
   LATCH_NONCE <= '0';
   HASH_START <= '0';
   NONCE_EN <= '0';
   IF START_SEARCH = '0' THEN
      STATE <= STOP;
   END IF;
```

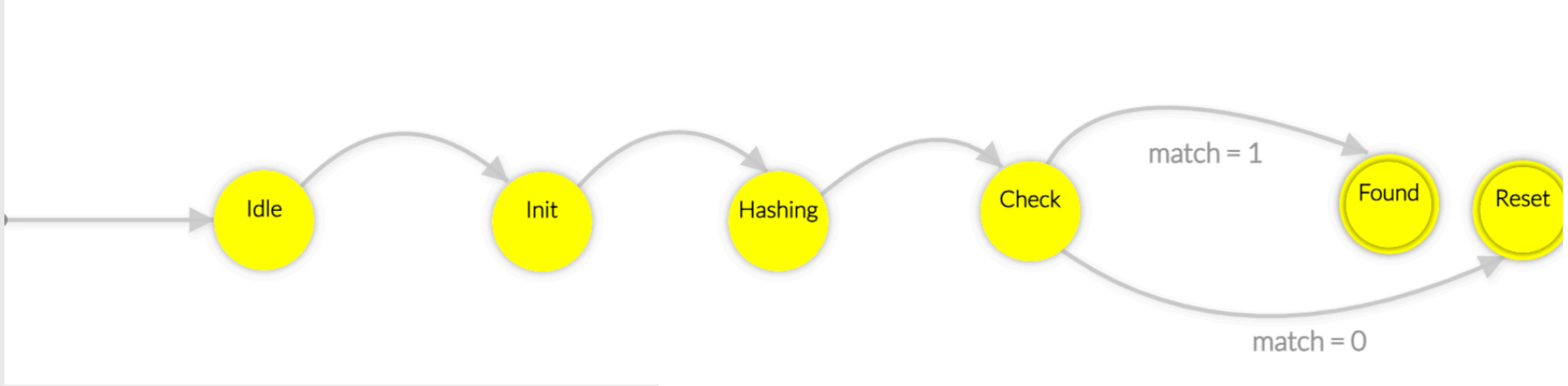
TESTING

The testbench includes a simple instrumentation to observe any system behavior. A custom hexadecimal formatting function can converts 32 bit std_logic_vectors into a readable 8character hex strings for console output. The clock generator produces a continuous 100 MHz clock signal with 10 ns period, providing the timing reference for all synchronous operations. Reset sequencing follows industry-standard practice: a 50 ns assertion period ensures all flip-flops reliably enter their reset states, followed by a 20 ns stabilization delay before test stimuli begin. Console reporting uses VHDL's report statement and text library to display test progress, found nonce values, resulting hash values, and success/failure indications.

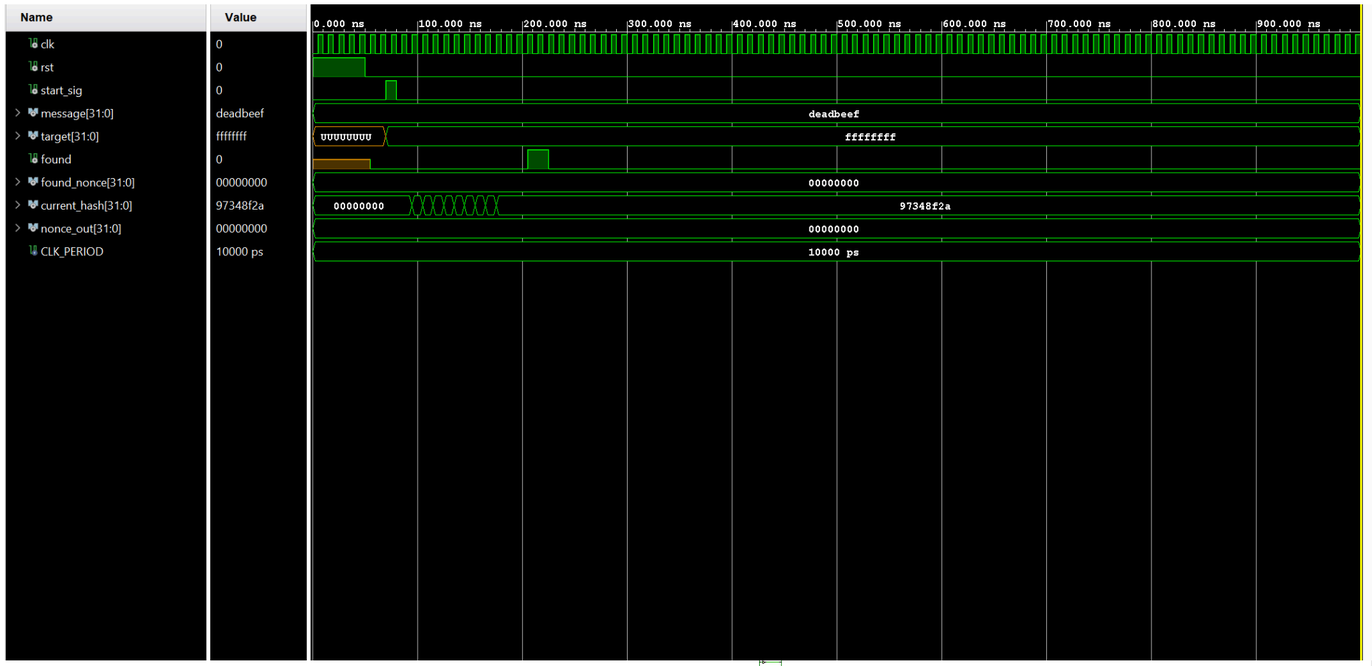
NORTH AMERICA

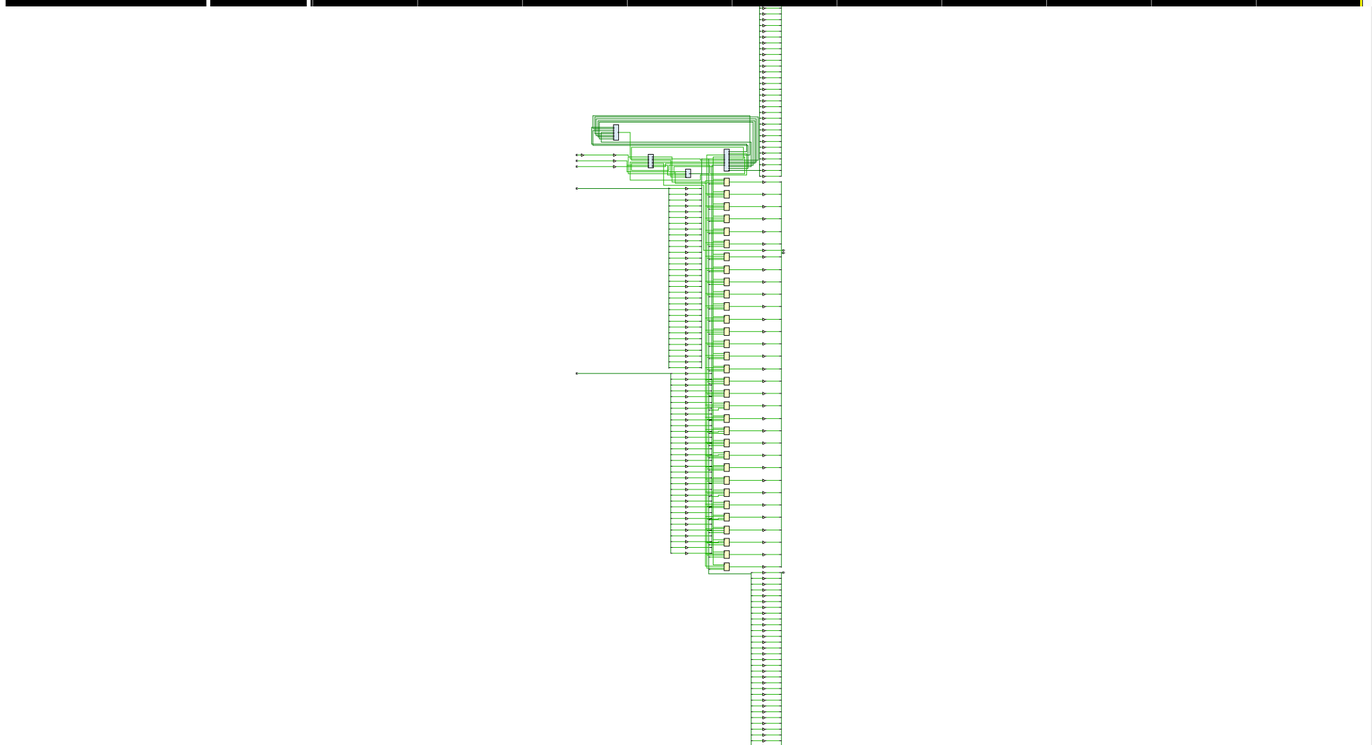
EUROPE

FINITE STATE MACHINE



RESULT


Name
Value
clk
0
rst
0
start_sig
0
message[31:0]
deadbeef
target[31:0]
ffffffff
found
0
found_nonce[31:0]
00000000
current_hash[31:0]
97348f2a
nonce_out[31:0]
00000000
CLK_PERIOD
10000 ps
0.000 ns
100.000 ns
200.000 ns
300.000 ns
400.000 ns
500.000 ns
600.000 ns
700.000 ns
800.000 ns
900.000 ns
deadbeef
ffffffff
uuuuuuuu
00000000
00000000
97348f2a
00000000
10000 ps



CONCLUSION

- **Successful PoW Implementation: The project achieved its primary objective by successfully translating the Bitcoin Proof-of-Work consensus mechanism into a functional VHDL hardware design.**
- **Modular Architecture: We constructed a robust system comprised of six integrated modules demonstrating how complex systems can be decomposed into manageable hardware units.**
- **Optimized Hashing Logic: By implementing a simplified 8-round SHA-256 core, we effectively demonstrated key cryptographic principles (XOR, rotation, modular addition).**
- **De-abstracting Blockchain: Rewriting the algorithm in hardware provided an understanding of blockchain mechanics that goes far beyond standard software abstractions.**

THANK **YOU** FOR LISTENING