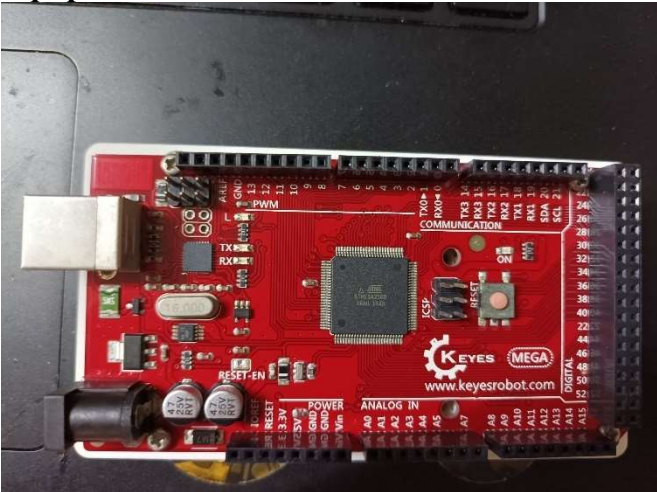


Future Hardware Testing Plans

To further validate the ACS Mini System's performance and security, the research project will incorporate two additional microcontroller boards for testing different encryption techniques. This expanded hardware testing phase aims to evaluate the system's data protection capabilities under various scenarios. The microcontrollers will be integrated into the existing setup, allowing for comprehensive testing of encryption algorithms and their impact on system performance. The testing results, including security assessments and performance metrics, will be documented and uploaded to the project's GitHub repository in the "documents" folder as a series of detailed reports. This iterative testing approach ensures the ACS Mini System meets the highest standards of data security and reliability, crucial for its successful deployment in construction environments.

Equipment: Arduino MEGA 2560



Tech specs

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Equipment: ESP8266 WeMos D1 R2



WeMOS D1:

The **WeMOS d1** which is an **Arduino compatible board** with the built-in **ESP8266 Wifi Module**. **WeMos D1** Arduino compatible development board is the cheapest WiFi-enabled board available today. It is a WiFi enabled based on the **ESP8266** chip. The board looks like an ordinary Arduino board. The dimensions and the pin layouts are exactly the same. So, this board is compatible with all the existing shields for **Arduino**. But don't expect them to work at once, since the libraries available for the **ESP8266 chip** are few so far. The boards, instead of an **ATMEGA** chip that standard Arduino boards use, use the impressive **ESP8266 Wifi chip**. The ESP8266EX chip that the **WeMos D1 board** uses offers:

- A 32 bit RISC CPU running at 80MHz
- 64Kb of instruction RAM and 96Kb of data RAM
- 4MB flash memory!
- Wi-Fi
- 16 GPIO pins
- I2C,SPI
- I2S
- 1 ADC

Test Result

Arduino MEGA 2560

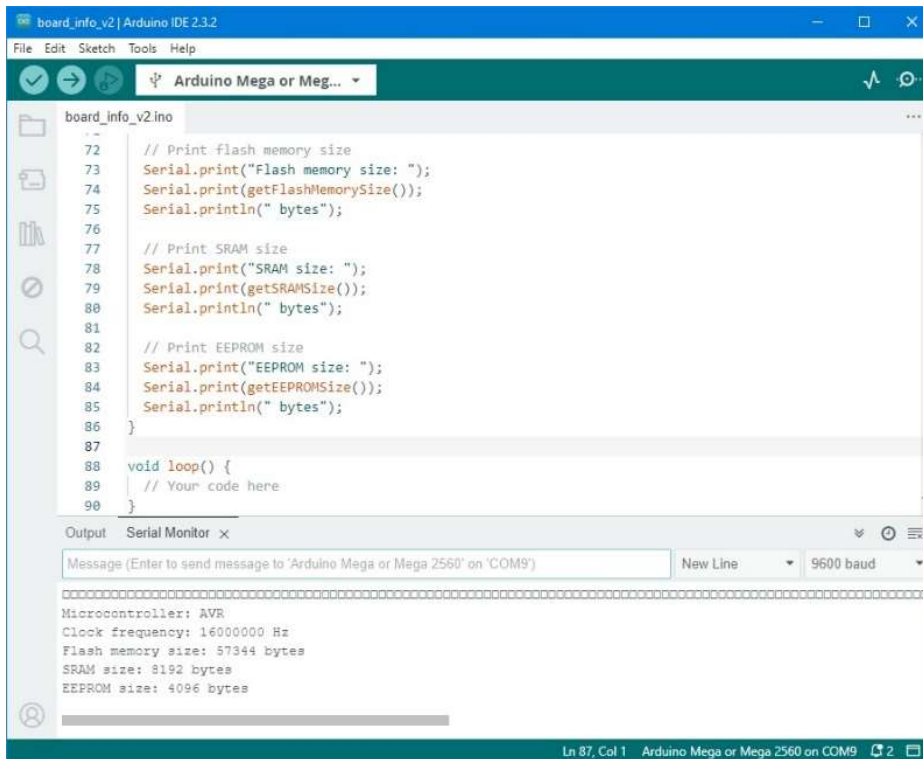
Encryption Algorithm	State Sizes (bytes)	Performance Tests (bytes per sec)	
		Encrypt	Decrypt
AES128	181	27962.53	14743.35
AES192	213	23267.88	12171.95
AES256	245	19923.04	10364.32
AESSmall128	34	23336.24	13492.51
AESSmall256	66	16852.51	9483.58
BLAKE2b	211	13482.85	
BLAKE2s	107	18305.26	
ChaCha20 128	132	19535.42	19533.18
ChaCha20 256	132	19535.42	19533.18
ChaCha12 128	132	30999.88	30994.18
ChaCha12 256	132	30999.88	30994.24
ChaCha8 128	132	43873.72	43862.29
ChaCha8 256	132	43873.60	43862.29
ChaCha+Poly1305	221	13110.32	13110.33
SHA-256	107	5985.56	
SHA3-256	205	16361.26	
SHAKE-256	206	16400.75	

WeMos D1 R2

ESP8266

Encryption Algorithm	State Sizes (bytes)	Performance Tests (bytes per sec)	
		Encrypt	Decrypt
AES128	188	167929.28	112976.16
AES192	220	140069.51	93693.05
AES256	252	120139.78	80031.85
AESSmall128	36	130264.94	93754.87
AESSmall256	68	98547.29	68881.32
BLAKE2b	224	713987.57	
BLAKE2s	120	809885.67	
ChaCha20 128	130	492989.95	490694.80
ChaCha20 256	132	493233.78	490833.70
ChaCha12 128	132	884736.58	880748.77
ChaCha12 256	132	884847.59	880748.77
ChaCha8 128	132	168903.35	163697.22
ChaCha8 256	132	169197.40	163697.22
ChaCha+Poly1305	240	578745.57	578593.84
SHA-256	120	814166.50	
SHA3-256	224	259523.29	
SHAKE-256	232	258027.23	

Arduino MEGA 2560 Testing Result



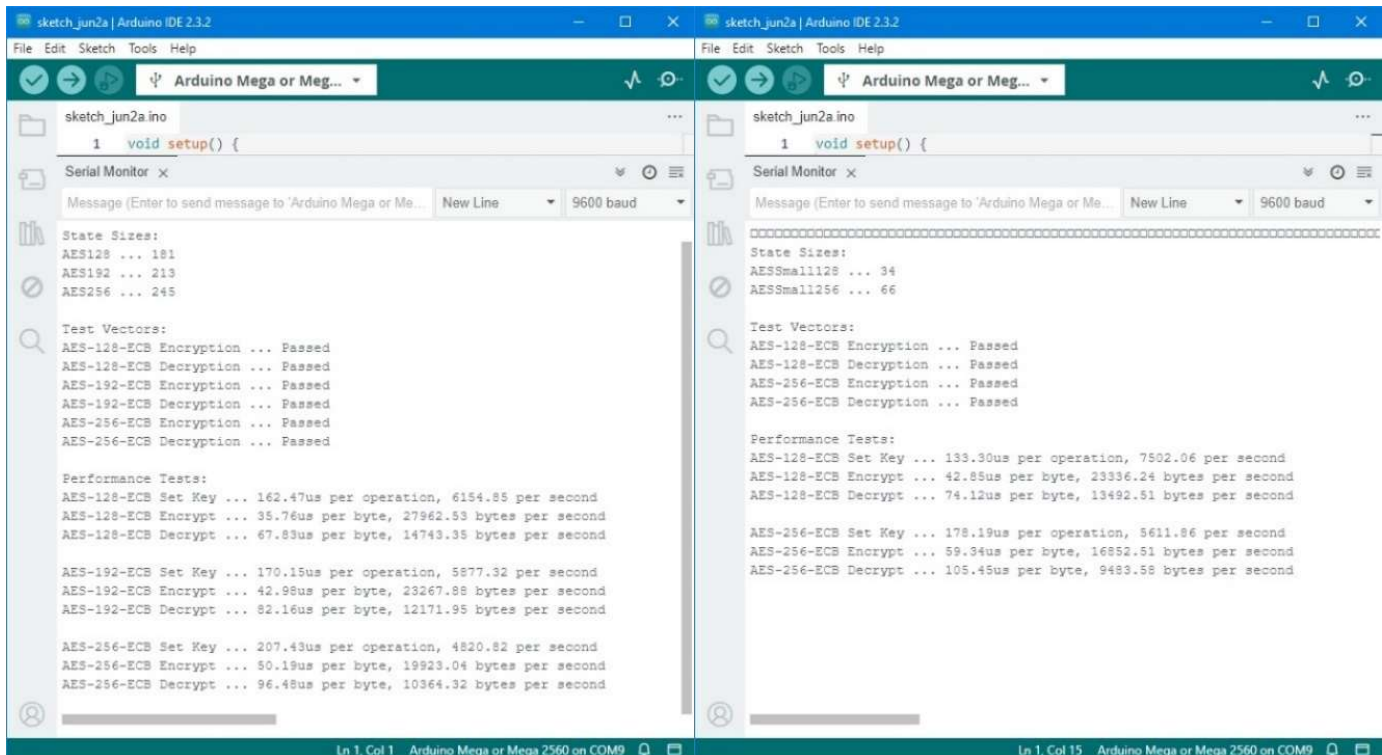
The screenshot shows the Arduino IDE 2.3.2 interface. The top toolbar includes icons for File, Edit, Sketch, Tools, and Help. The menu bar also lists these options. The toolbar dropdown shows 'Arduino Mega or Meg...'. The main editor displays the file 'board_info_v2.ino' with the following code:

```
72 // Print flash memory size
73 Serial.print("Flash memory size: ");
74 Serial.print(getFlashMemorySize());
75 Serial.println(" bytes");
76
77 // Print SRAM size
78 Serial.print("SRAM size: ");
79 Serial.print(getSRAMSize());
80 Serial.println(" bytes");
81
82 // Print EEPROM size
83 Serial.print("EEPROM size: ");
84 Serial.print(getEEPROMSize());
85 Serial.println(" bytes");
86 }
87
88 void loop() {
89   // Your code here
90 }
```

The Serial Monitor at the bottom shows the output of the code:

```
Microcontroller: AVR
Clock frequency: 16000000 Hz
Flash memory size: 57344 bytes
SRAM size: 8192 bytes
EEPROM size: 4096 bytes
```

The status bar at the bottom indicates 'Ln 87, Col 1 Arduino Mega or Mega 2560 on COM9'.



The image shows two side-by-side screenshots of the Arduino IDE 2.3.2 interface, both displaying the file 'sketch_jun2a.ino'.

The left screenshot shows the code in the editor:

```
1 void setup() {
```

The Serial Monitor shows the output of the code:

```
State Sizes:
AES128 ... 181
AES192 ... 213
AES256 ... 245

Test Vectors:
AES-128-ECB Encryption ... Passed
AES-128-ECB Decryption ... Passed
AES-192-ECB Encryption ... Passed
AES-192-ECB Decryption ... Passed
AES-256-ECB Encryption ... Passed
AES-256-ECB Decryption ... Passed

Performance Tests:
AES-128-ECB Set Key ... 162.47us per operation, 6154.85 per second
AES-128-ECB Encrypt ... 35.76us per byte, 27962.53 bytes per second
AES-128-ECB Decrypt ... 67.83us per byte, 14743.35 bytes per second

AES-192-ECB Set Key ... 170.15us per operation, 5877.32 per second
AES-192-ECB Encrypt ... 42.98us per byte, 23267.88 bytes per second
AES-192-ECB Decrypt ... 82.16us per byte, 12171.95 bytes per second

AES-256-ECB Set Key ... 207.43us per operation, 4820.82 per second
AES-256-ECB Encrypt ... 50.19us per byte, 19923.04 bytes per second
AES-256-ECB Decrypt ... 96.48us per byte, 10364.32 bytes per second
```

The status bar at the bottom indicates 'Ln 1, Col 1 Arduino Mega or Mega 2560 on COM9'.

The right screenshot shows the same code in the editor:

```
1 void setup() {
```

The Serial Monitor shows the output of the code:

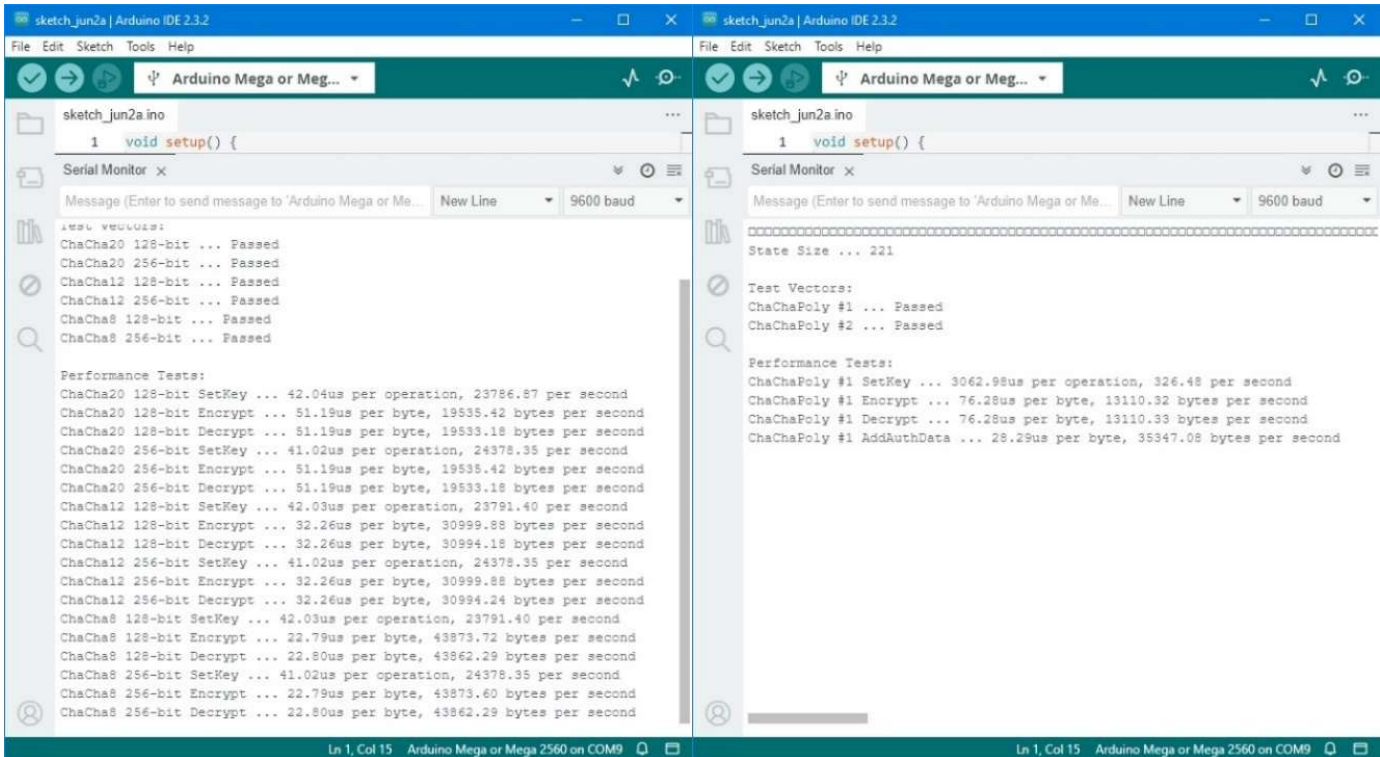
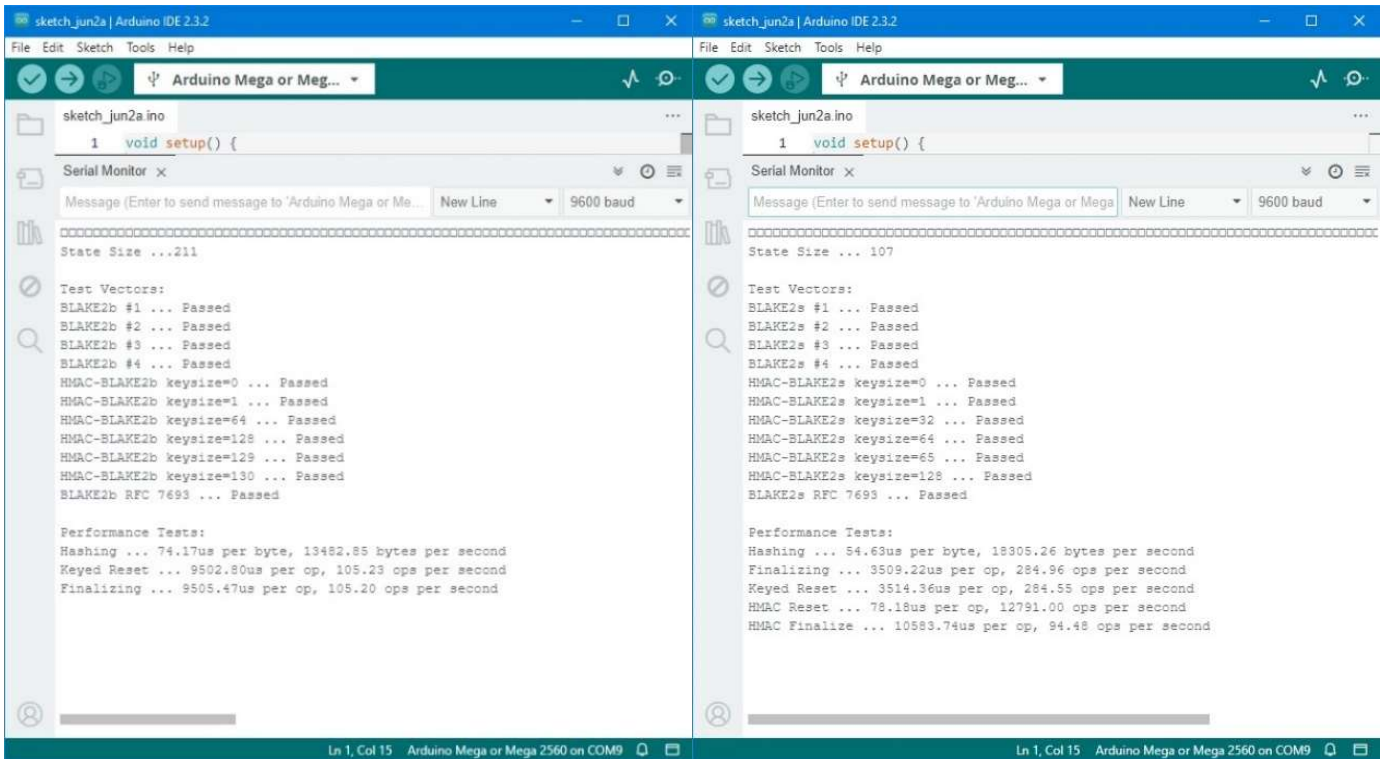
```
State Sizes:
AESSmall128 ... 34
AESSmall1256 ... 66

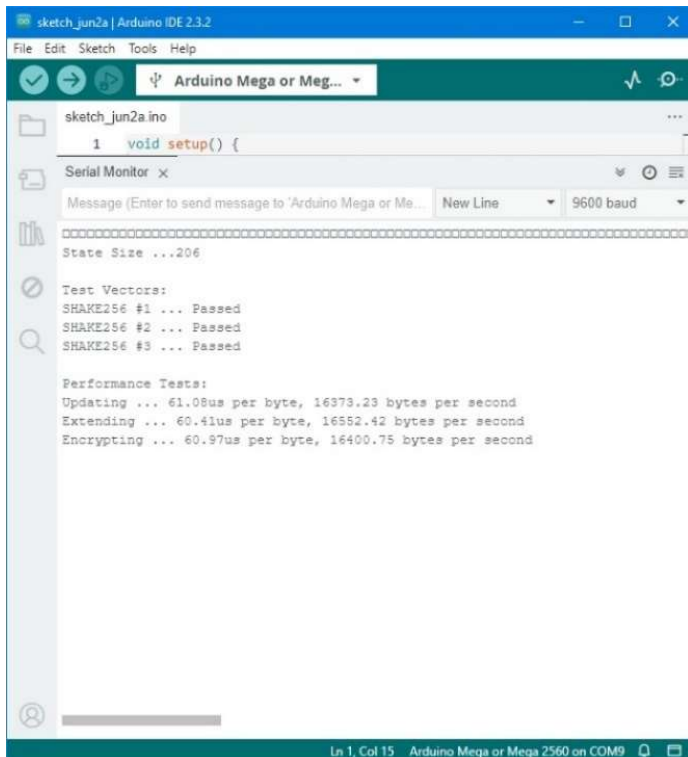
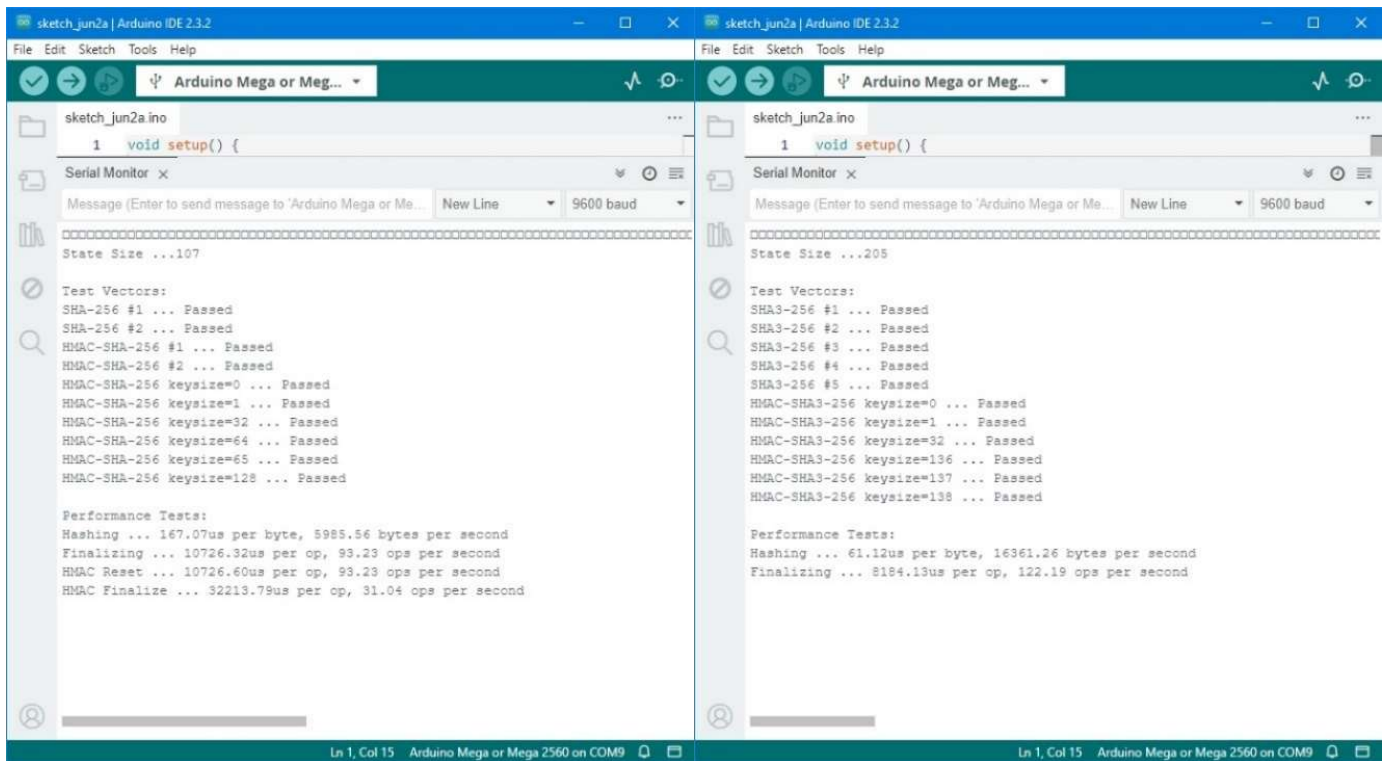
Test Vectors:
AES-128-ECB Encryption ... Passed
AES-128-ECB Decryption ... Passed
AES-256-ECB Encryption ... Passed
AES-256-ECB Decryption ... Passed

Performance Tests:
AES-128-ECB Set Key ... 133.30us per operation, 7502.06 per second
AES-128-ECB Encrypt ... 42.88us per byte, 23336.24 bytes per second
AES-128-ECB Decrypt ... 74.12us per byte, 13492.51 bytes per second

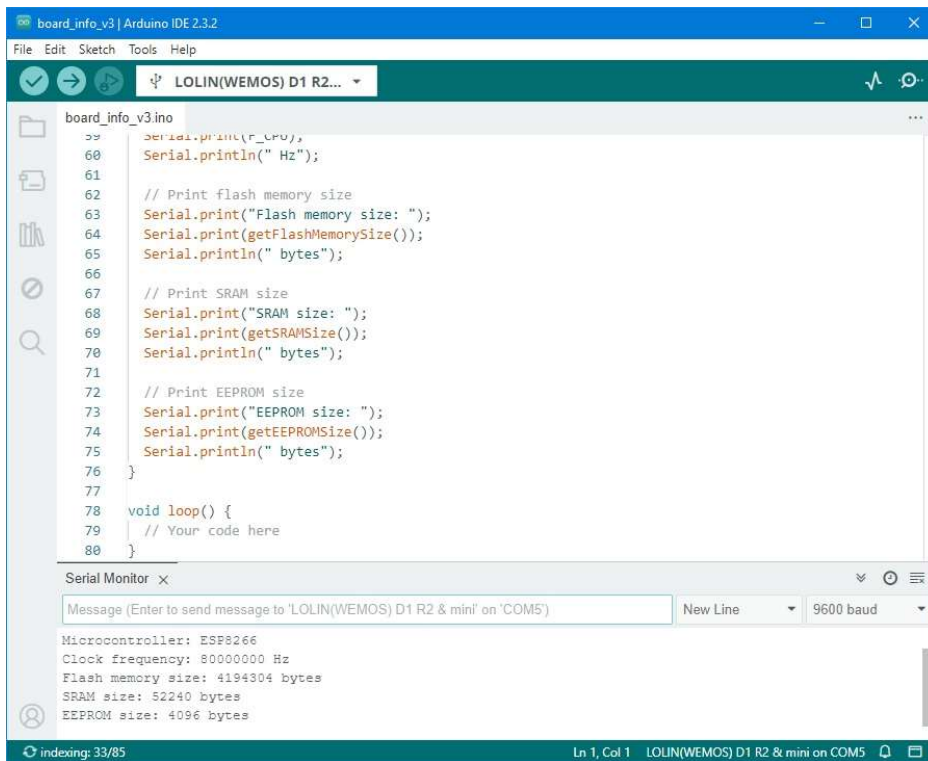
AES-256-ECB Set Key ... 178.19us per operation, 5611.86 per second
AES-256-ECB Encrypt ... 59.34us per byte, 16852.51 bytes per second
AES-256-ECB Decrypt ... 105.45us per byte, 9483.58 bytes per second
```

The status bar at the bottom indicates 'Ln 1, Col 15 Arduino Mega or Mega 2560 on COM9'.



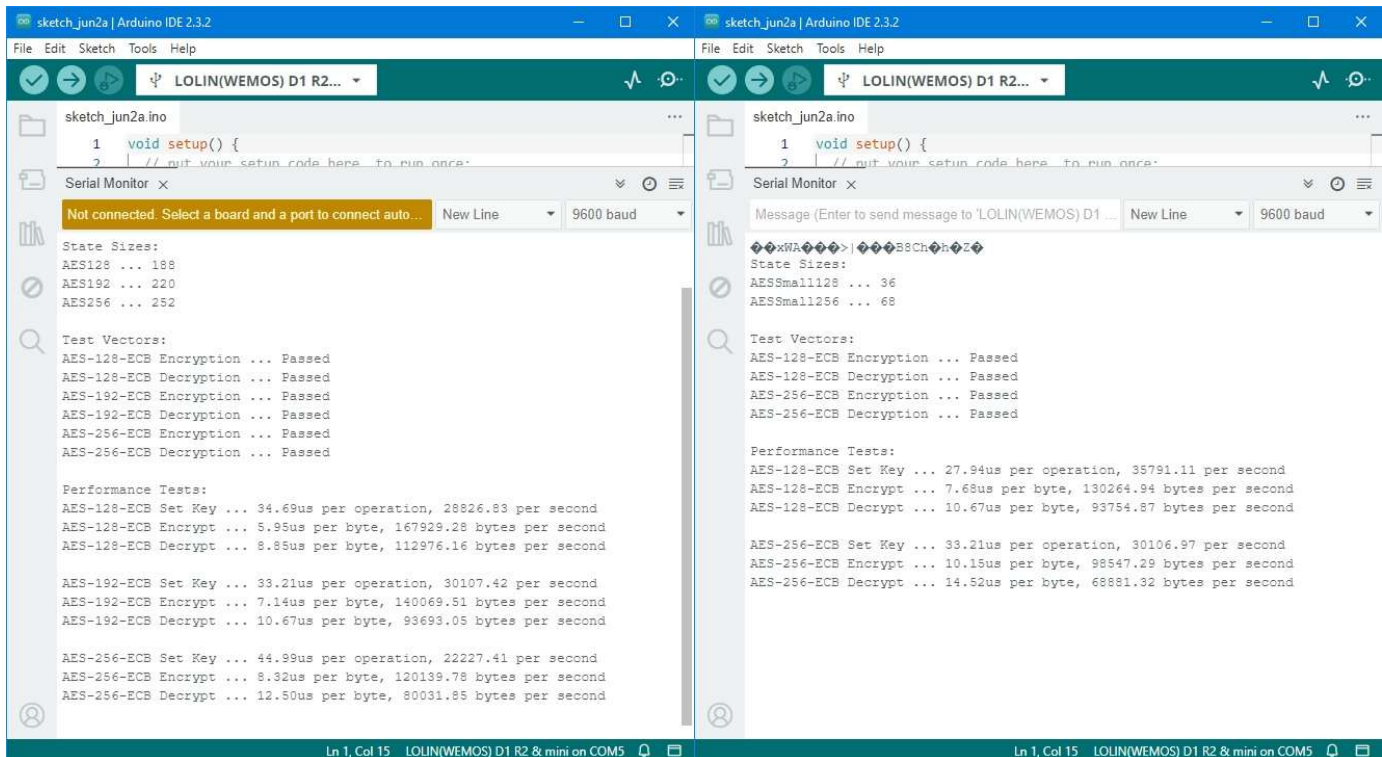


WeMos D1 R2 ESP8266 Testing Result



The screenshot shows the Arduino IDE interface with the file `board_info_v3.ino` open. The code prints various hardware specifications of the LOLIN(WEMOS) D1 R2. The Serial Monitor displays the following output:

```
Microcontroller: ESP8266
Clock frequency: 80000000 Hz
Flash memory size: 4194304 bytes
SRAM size: 52240 bytes
EEPROM size: 4096 bytes
```



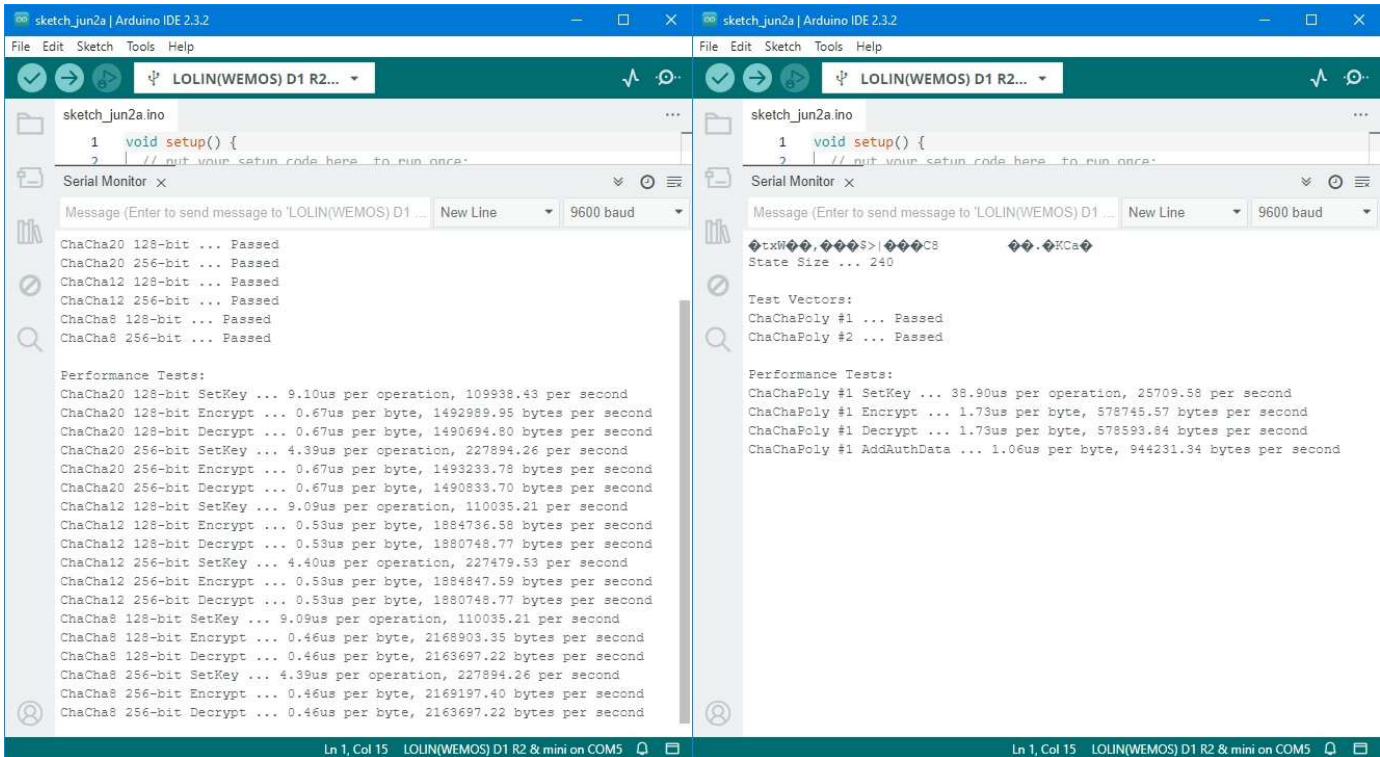
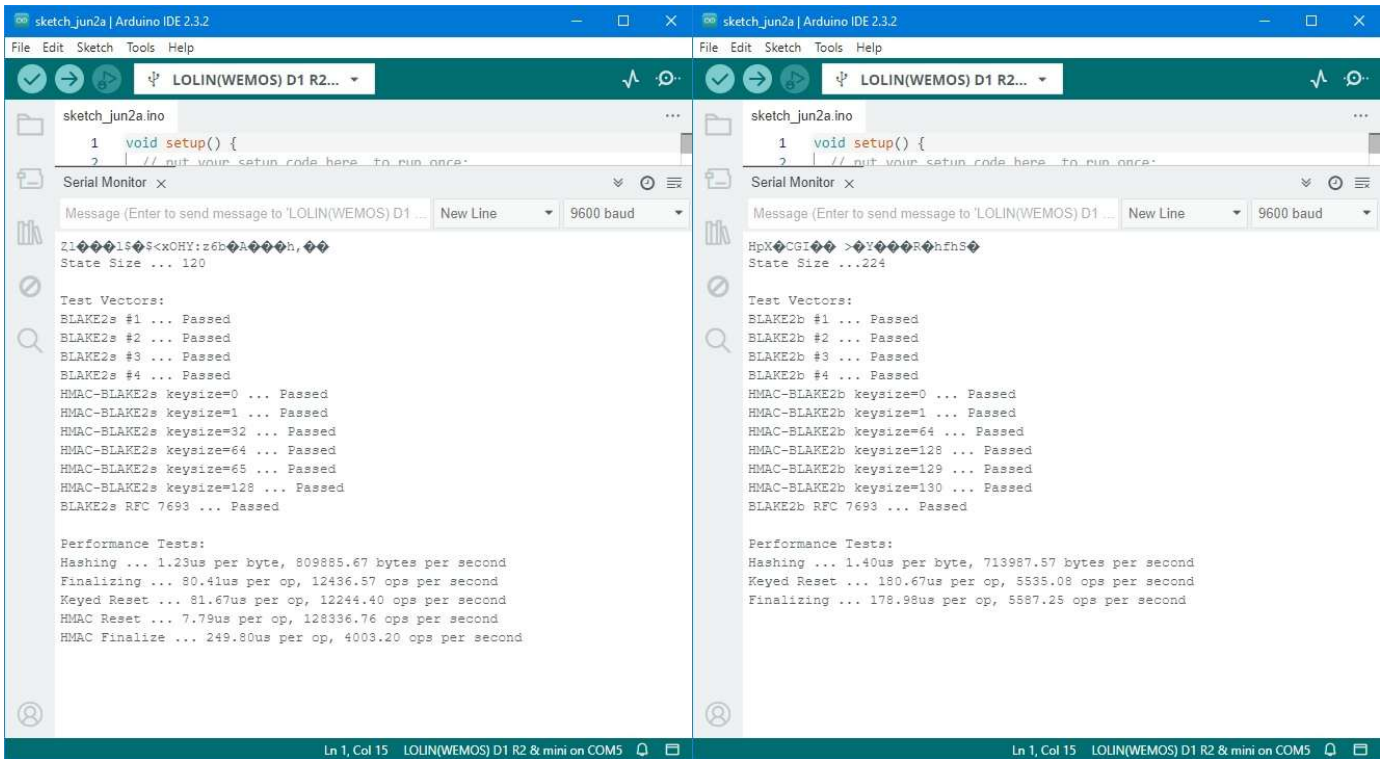
The image displays two side-by-side screenshots of the Arduino IDE, both showing the `sketch_jun2a.ino` file. The left screenshot shows the initial state where the board is not connected. The right screenshot shows the test results after a successful connection.

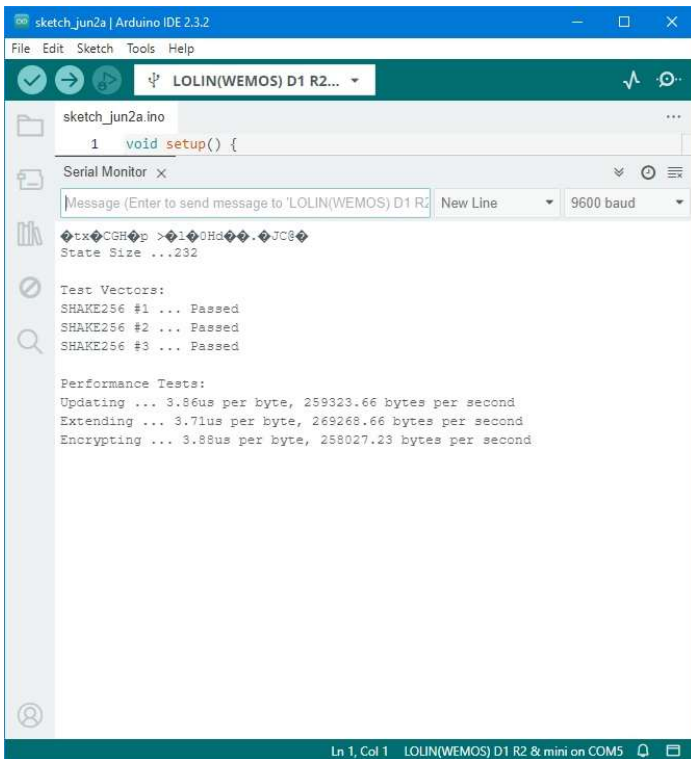
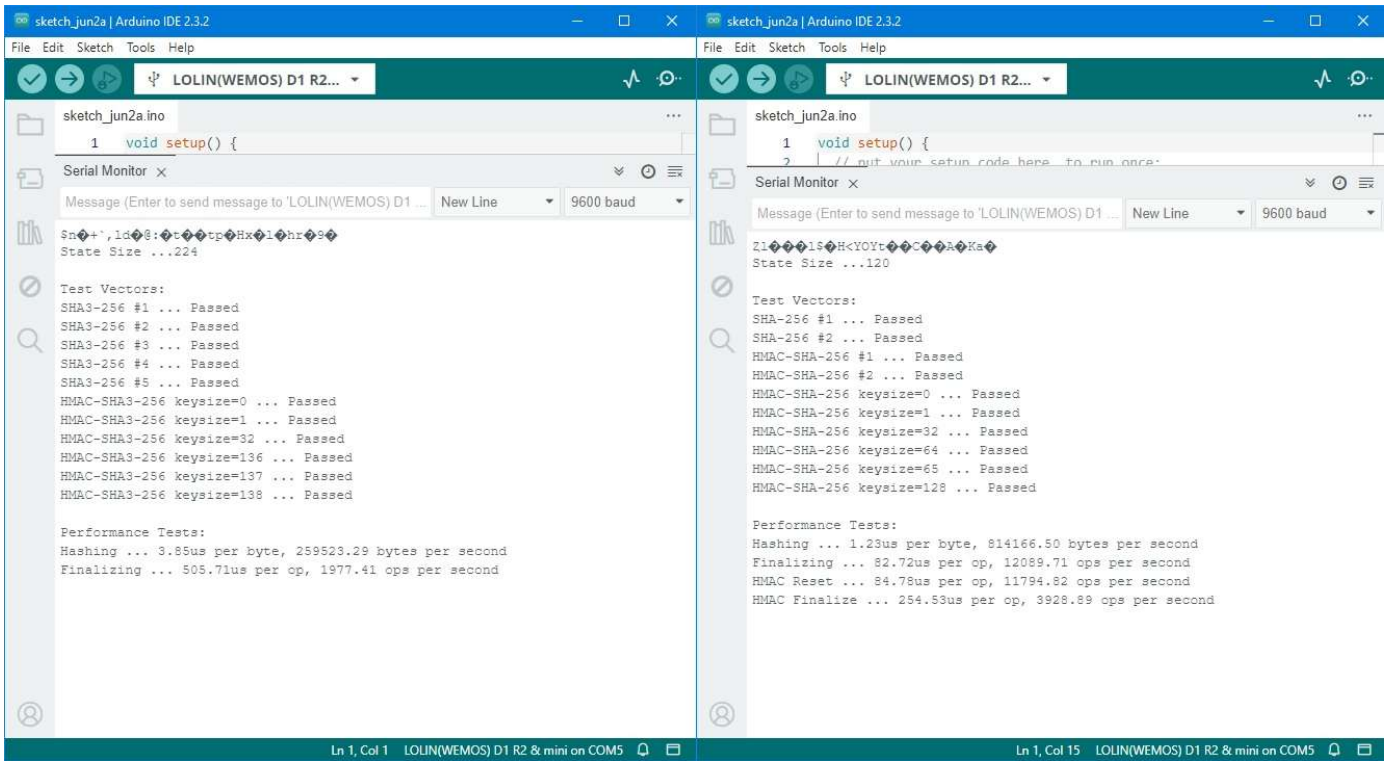
Left Screenshot (Not connected):

- Serial Monitor: "Not connected. Select a board and a port to connect auto..."
- Test Vectors: All tests (AES-128-ECB, AES-192-ECB, AES-256-ECB for both encryption and decryption) are listed as "Passed".
- Performance Tests: Detailed timing information for each operation (e.g., AES-128-ECB Set Key: 34.69us per operation, 28826.83 per second).

Right Screenshot (Connected):

- Serial Monitor: Shows the received data as `0x00000000000000000000000000000000`.
- Test Vectors: All tests are listed as "Passed".
- Performance Tests: Detailed timing information for each operation (e.g., AES-128-ECB Set Key: 27.94us per operation, 35791.11 per second).





Additional Conclusion and Recommendations

The additional performance evaluation of encryption algorithms on resource-constrained devices has provided valuable insights into their suitability for secure data monitoring in the construction industry. The results highlight the trade-offs between security strength, performance, and resource usage, which are crucial considerations for developing an affordable and efficient ACS Mini System tailored for construction SMEs.

Among the tested algorithms, ChaCha12 and ChaCha8 demonstrated remarkably high encryption and decryption performance, making them promising candidates for real-time data encryption and decryption. The AES algorithms (AES128, AES192, and AES256) also exhibited good performance, with higher throughput rates on the more powerful WeMos D1 R2 platform. Other algorithms as BLAKE2b, BLAKE2s, SHA-256, SHA3-256, and SHAKE-256 showed impressive performance, suitable for scenarios prioritizing data integrity and authentication over confidentiality. The ChaCha+Poly1305 algorithm, providing authenticated encryption, could be valuable for secure communication channels where data authenticity is crucial.

Recommendations:

1. Adopt ChaCha12 (128-bit or 256-bit) as the primary encryption algorithm for real-time data encryption and decryption within the ACS Mini System, leveraging its high throughput rates and security strength.
2. Implement ChaCha+Poly1305 for secure communication channels, ensuring both data confidentiality and integrity.
3. Leverage other algorithms like BLAKE2b, BLAKE2s, SHA-256, SHA3-256, and SHAKE-256 for specific use cases prioritizing data integrity and authentication.
4. Explore hardware acceleration techniques and code optimizations to further improve encryption algorithm performance on resource-constrained devices.
5. Continuously monitor the security landscape and periodically re-evaluate the selected algorithms to ensure they remain secure and efficient.
6. Collaborate with industry partners, regulatory bodies, and construction SMEs to establish best practices and standards for secure data monitoring, ensuring interoperability and widespread adoption.

By following these recommendations, the ACS Mini System project can enhance the security and efficiency of data monitoring for construction SMEs, addressing resource constraints while ensuring compliance with industry standards and regulations.