EEE3097S Second Progress Report

Group 3

Shameera Cassim and Tristyn Ferreiro
CSSSHA020, FRRTRI001

17 October 2022

Admin

Contributions

Task	**Completed by**
Admin	Both
Inter-subsystem Design	Both
IMU Module: Feature Discussion	Shameera Cassuim
IMU Module: Data Compatibility	Tristyn Ferreiro
IMU Module: Validation Testing	Both
IMU Module: Test Cases	Both
Experiment Setup: Joint Algorithms	Both
Experiment Setup: Overall	Both
Experiment Setup: IMU	Both
Experiment Setup: Compression	Tristyn Ferreiro
Experiment Setup: Encryption	Shameera Cassim
Results: Joint Algorithms	Both
Results: Overall	Both
Results: IMU	Both
Results: Compression	Tristyn Ferreiro
Results: Encryption	Shameera Cassim
Acceptance Test Procedures: Assessment table	Both
Acceptance Test Procedures: ATP Updates	Both

Project Timeline

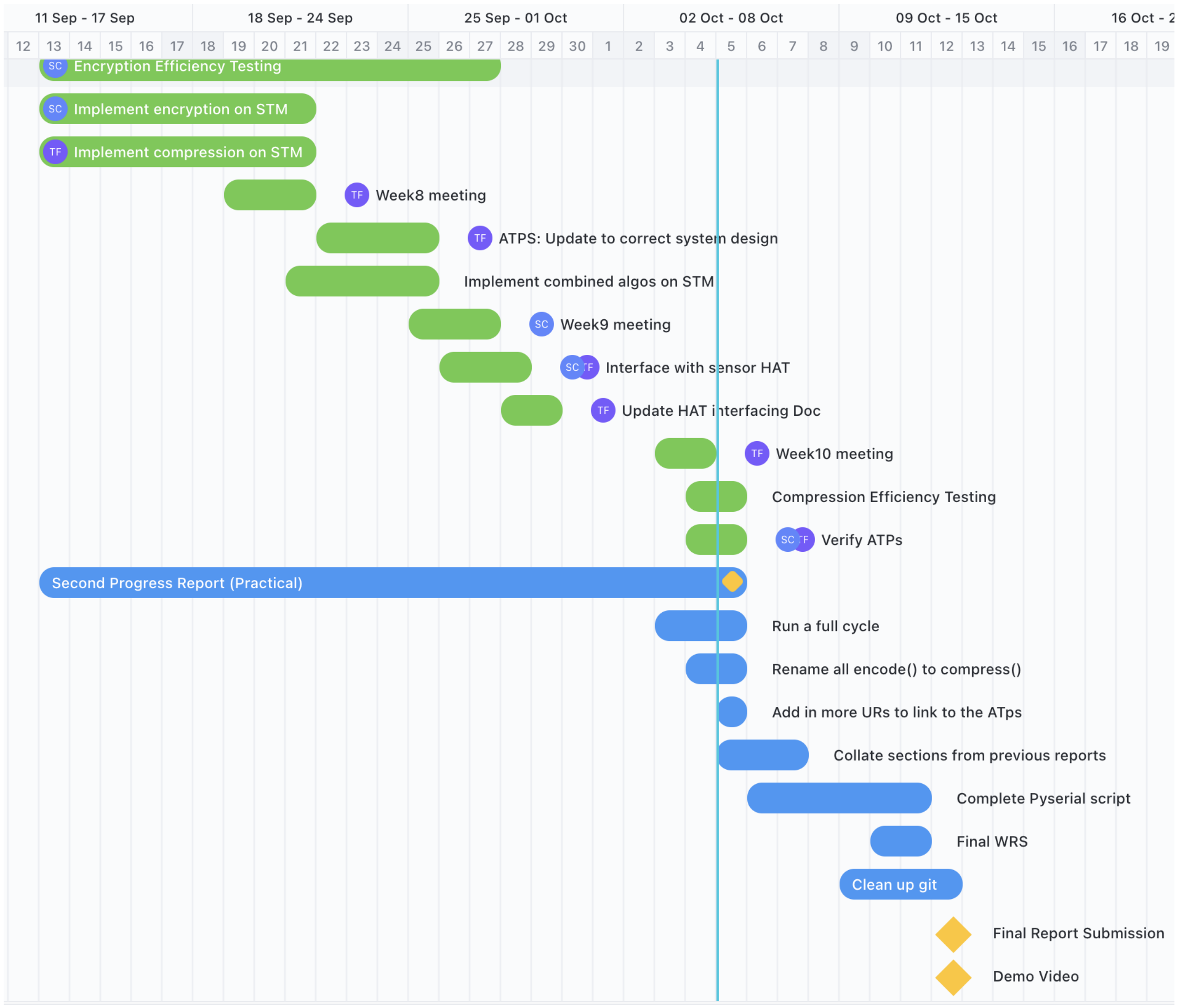
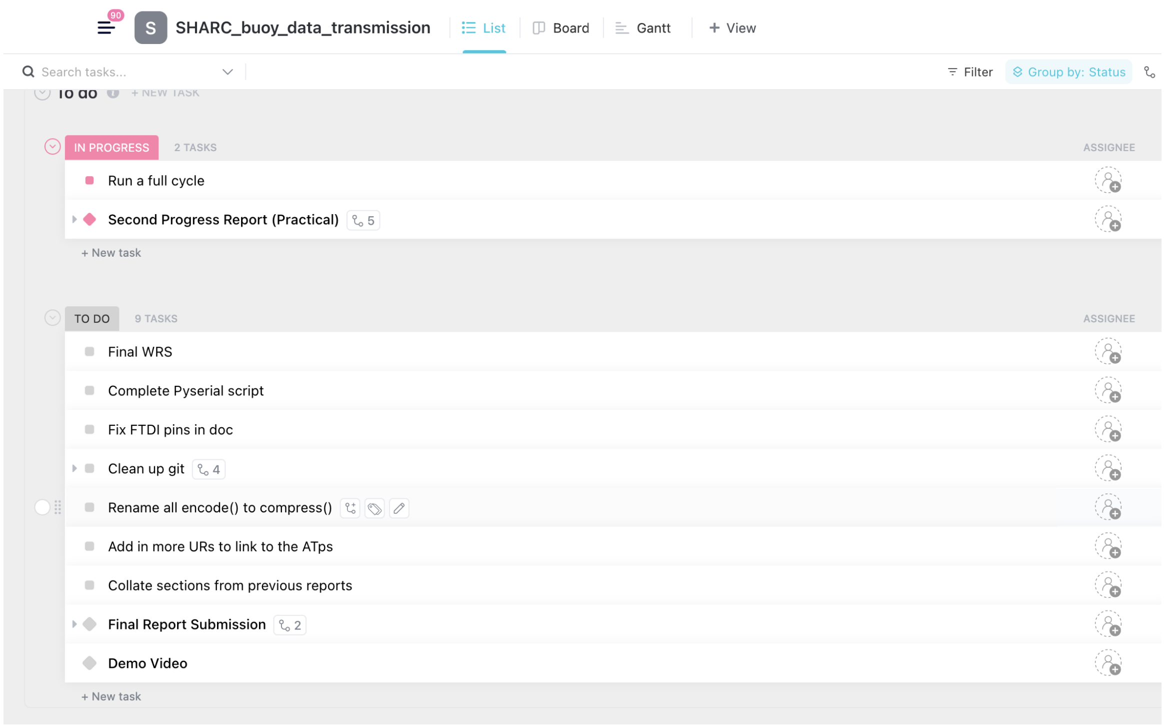


Figure 1: Timeline for the remainder of the project

Project Management tool



IN PROGRESS 2 TASKS ASSIGNEE

- Run a full cycle
- Second Progress Report (Practical) 5

+ New task

TO DO 9 TASKS ASSIGNEE

- Final WRS
- Complete Pyserial script
- Fix FTDI pins in doc
- Clean up git 4
- Rename all encode() to compress()
- Add in more URs to link to the ATps
- Collate sections from previous reports
- Final Report Submission 2
- Demo Video

+ New task

Figure 2: Screenshot of the project management tool showing the remainder of our tasks

The list above shows a break down of milestones and suggested tasks for the remainder of the project from the ClickUp task list. Some tasks have sub-tasks that need to be completed before the main task can be marked as completed. More tasks will be added as needed. Tasks not assigned to a specific person are either joint tasks or will be assigned in future.

Git

The GitHub repository for this project can be found via this link: git

Contents

1 Subsystem Design **7**
1.1 Compression . . . 7
1.1.1 Algorithm Design . . . 7
1.2 Inter-Subsystem and Inter-Sub-subsystems Interactions . . . 7
1.2.1 Interfacing . . . 8

2 IMU Module **9**
2.1 Salient Features . . . 9
2.2 Data Compatibility with ICM-20649 . . . 9
2.3 Innovation in Validation Testing . . . 9
2.4 Validation Testing . . . 10
2.5 Code design . . . 10

3 Experiment Setup **11**
3.1 Overall System Functionality . . . 11
3.1.1 Compression and Encryption work together . . . 11
3.1.2 Compression and Encryption work on STM32F0 . . . 11
3.1.3 Compressing and Encrypting IMU data . . . 11
3.2 IMU . . . 12
3.2.1 IMU initialisation . . . 12
3.2.2 Acceleration Test . . . 12
3.2.3 Gyroscope Test . . . 12
3.2.4 Pendulum Test . . . 12
3.3 Wave test . . . 12
3.4 Compression . . . 13
3.4.1 General . . . 13
3.4.2 Efficiency Testing . . . 13
3.5 Encryption . . . 13
3.5.1 Efficiency Testing . . . 13
3.5.2 Breakability Testing . . . 13

4 Results **14**
4.1 Overall System Functionality . . . 14
4.1.1 Sending data from STM32F0 . . . 14
4.1.2 Interfacing IMU and STM32F0 . . . 14
4.1.3 Full System Efficiency . . . 14
4.1.4 Compression and Encryption work together . . . 15
4.2 IMU . . . 15
4.2.1 IMU initialisation . . . 15
4.2.2 Acceleration Test . . . 15
4.2.3 Gyroscope Test . . . 16
4.2.4 Pendulum Test . . . 17
4.2.5 Wave Test . . . 20
4.3 Compression . . . 21
4.3.1 Efficiency Testing . . . 21
4.4 Encryption . . . 22
4.4.1 General . . . 22
4.4.2 Efficiency Testing . . . 22

5 Acceptance Test Procedures **24**
5.1 AT01 25
5.2 AT02 25
5.3 AT03 26
5.4 AT04 26
5.5 AT05 26
5.6 AT06 27
5.7 AT07 28
5.8 AT08 28
5.9 AT09 29

6 Appendix A: Requirements **31**

Listings

1 Subsystem Design

This is a brief overview of the subsystem design and decisions made to date, some updates have been made since the Paper Design.

1.1 Compression

1.1.1 Algorithm Design

The implemented version of LZSS compression is based off of Haruhiko Okumura's LZSS encoder-decoder. Adjustments have been made to suit the requirements of this design however, the core algorithm uses Haruhiko Okumura's algorithm.

1.2 Inter-Subsystem and Inter-Sub-subsystems Interactions

Below are some changes made to the interactions described in the Paper Design and Progress Report.

The original system was designed to first compress the data and then to encrypt it, as encryption transforms data into high-entropy data, making it random while compression relies on patterns in order to preform well and reduce the data size. However, the encryption algorithm chosen, RSA, converts the input data into purely numerical data. This means while there isn't necessarily a pattern to the data, there are many repetitions of characters which allows for more efficient compression. Thus, the inter-subsystem design has been updated so that the data is first encrypted and then compressed. This is illustrated in the diagram below:

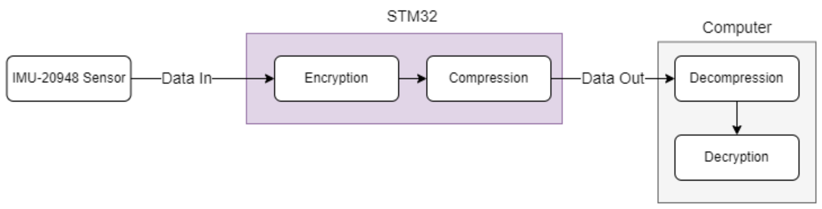


Figure 3: Diagram of Inter-Subsystem interactions

In order to test the system, it will be setup as follows:

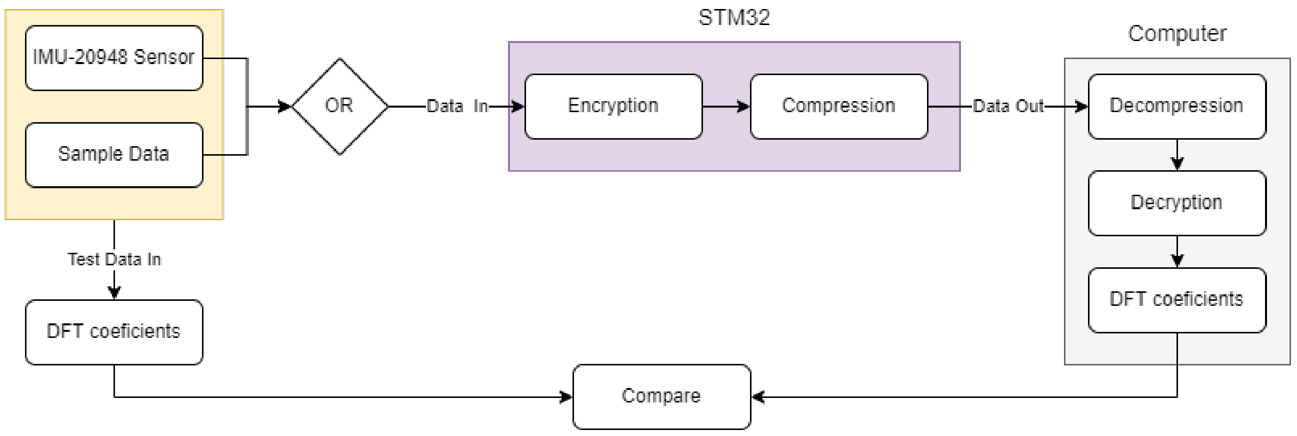


Figure 4: Diagram of Inter-Subsystem interactions - Testing and Development

As discussed in previous reports, all encryption and compression code has been written into a single c file. In this file, encryption writes the encrypted output to a global array which is accessed by the compression algorithm to be compressed and sent to a PC. This simplifies implementation and allows for a more cohesive code structure. This c file has been used for testing on a computer before transferring the code onto the Discovery board for system testing.

1.2.1 Interfacing

Below are updates made to the interfacing of the system, the Waveshare HAT B has been replaced by the SparkFun 9DoF IMU Breakout board.

The subsystems need to be connected physically and will be interfaced as described in table 1 below.

Table 1: Interfacing specifications

Interface	Description	Pins/Output
I001	SparkFun 9DoF IMU Breakout board to STM for data transfer (SPI)	• MISO: Breakout MISO* to STM PB14 • MOSI: Breakout MOSI* to STM PB15 • SCLK: Breakout SCLK* to STM PB13 • CS: Breakout CS* to STM PB12 (GPIO) • Power: Breakout 1V8-5V5* to STM 3V • GND: Breakout GND* to STM GND
I002	STM to FTDI for data transfer to computer (UART)	• STM PA10(RX) to FTDI TX • STM PA9(TX) to FTDI RX
I003	FTDI to computer for data transfer from STM	• Assuming the FTDI is connected to a Micro B USB port (which can be used to connect to a USB cable to connect to a computer), output via cable

*referring to pins labeled with a blue flag in the image below: [1]

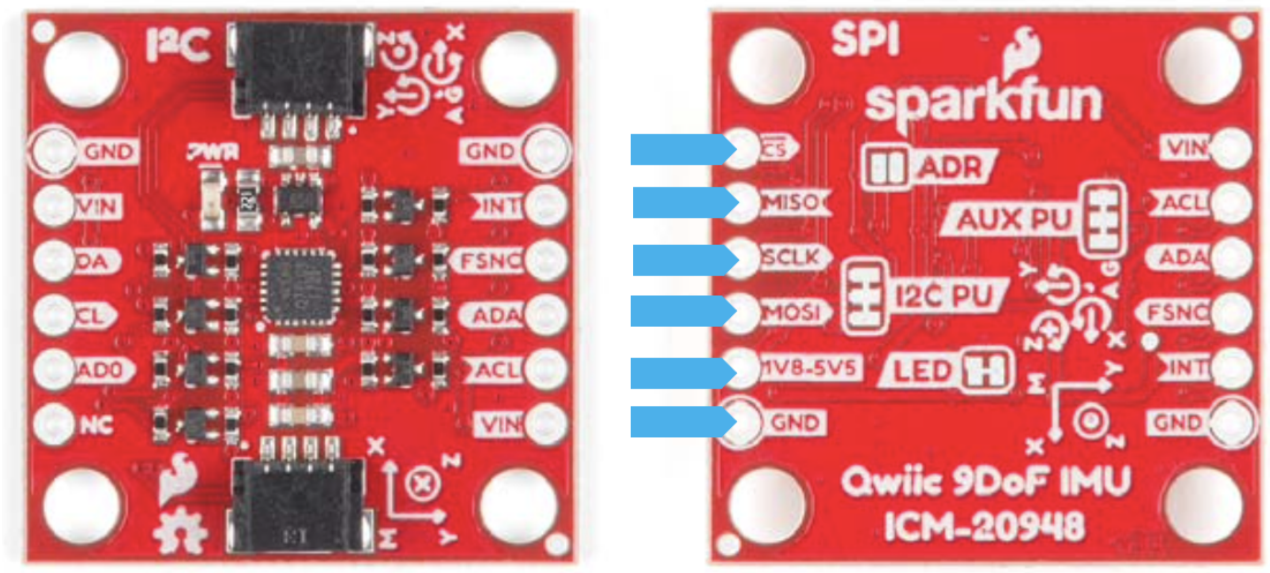


Figure 5: Front and back of SparkFun 9DoF IMU Breakout board

2 IMU Module

2.1 Salient Features

The IMU used in the testing and development of this system is the ICM-20948. However, the IMU used on the SHARC buoy, and hence the IMU intended to be used with the system, is the ICM-20649. While these IMU's are similar, they have some differences in what they can each do.

One of the most notable differences between the 2 for this application is that the ICM-20948 is a 9-axis device while the ICM-20649 is a 6-axis device. Both chips have a 3-axis accelerometer and a 3-axis gyroscope, however, the 20948 has a 3-axis compass which the 20649 does not. Thus, it is important to ensure that the system does not extract or use the compass readings. This will maintain that the system can be implemented using the 20649.

The ICM-20649 has higher full scale resolution for both the accelerometer and the gyroscope. The 20948 has a gyroscope with a maximum programmable FSR of $\pm 2000dps$ and an accelerometer with a maximum programmable FSR of $\pm 16g$. The corresponding values for the 20649 are $\pm 4000dps$ and $\pm 30g$ respectively. This means that the 20649 can read almost double the acceleration and double the angular velocity with accuracy than the 20948. However, for the purpose of the SHARC buoy, low values of $\pm 500dps$ and $\pm 2g$ [2] are needed; making system testing possible on the 20948.

The 20649 IMU has a wake-on-motion interrupt for low power operations. This feature is not available on the 20948. Thus, it will not be configured or used in this implementation but this feature should be used in the actual implementation on the SHARC buoy to conserve as much power as possible.

2.2 Data Compatibility with ICM-20649

The data used for testing the IMU and subsequently the entire system uses only the x,y and z accelerometer and gyroscope measurements.

Although the ICM-20948 used for testing has an additional onboard 3-axis compass, data will not be read from it. The ICM-20649 is compatible with an external compass but this is not used in the SHARC buoy design [2].

Both IMUs have onboard temperature sensors. However, the SHARC buoy system uses a dedicated temperature sensor and does not make use of IMU on-chip sensor [2]. Since the STM32F0 has limited RAM and Flash memory, recording these values would waste processing power and memory needed for testing. Thus, temperature measurements are not recorded.

Although the SparkFun board has an onboard magnetometer sensor this data is not used. It is not inside the scope of the project and the data is not useful in the context of the SHARC buoy [2].

Ultimately, by designing the system to only take gyroscope and accelerometer readings, the code is compatible with both the ICM-20948 and ICM-20649.

2.3 Innovation in Validation Testing

Pendulum
To check that the sensors are working as expected a pendulum system is used. This will clearly show trends in any form of acceleration allowing for thorough testing of both the gyroscope and the accelerometer.

Wave
At the time of this experiment, equipment needed to simulate the real SHARC bouy environment is unavailable. An alternative simulation setup is made using a breadboard and a bed sheet. The system is placed on the breadboard and the breadboard is placed on the sheet. The sheet is gently moved up and down and in different directions to simulate waves.

Testing this gentle motion also helps ensure that the system can sense the effect that wave swells have on a sheet of ice.

2.4 Validation Testing

In order to validate the IMU module is working, the following tests were run:

Table 2: IMU testing

IMU Test	Description	Success
IT01	Test that the startup sequence runs correctly by ensuring the initialization function is completed before any other functions are run	PASS
IT02	Test that the direction of measurements is recorded correctly by accelerating the sensor in the positive and negative x, y and z directions and confirm that the recorded value has the correct direction (accelerometer testing)	PASS
IT03	Test that the direction of measurements is recorded correctly by turning the sensor in the positive and negative x, y and z directions and confirm that the recorded value has the correct direction (gyroscope testing)	PASS
IT04	Test that magnitude of readings increase and decrease as expected relative motion of pendulum	PASS
IT05	Test that **all** relevant readings change if the sensor is moved in multiple axes as expected from wave motion (eg if there is acceleration in the x direction and the y direction, both readings should change)	PASS

2.5 Code design

The code used to interface with the ICM20948 is based on the stm32_hal_icm20948 github and has been adapted for the design requirements.

The code was adapted to be compatible with the STM32F0 not the STM32F4. The gyroscope and accelerometer reading methods were changed to read from the high and low registers for each value and then combine them into a 16 bit value. The calibration was changed to also read from high and low offset registers for all values.

3 Experiment Setup

The test data used for this report has been gathered using the SparkFun 9DoF IMU Breakout board.

3.1 Overall System Functionality

3.1.1 Compression and Encryption work together

Aims to determine that the algorithms work together, in line with AT06.

This ATP was tested in progress report 1 however, the system design has been changed and so this ATP is retested.

The compression and encryption algorithms are combined into a single c file. This combined program uses an array as input and prints the compressed-encrypted data to a file. The test data is passed into the program as an array and the encrypted data is then passed to the compression algorithm without user interaction. The expected output is a compressed-encrypted file.

After encryption and compression, the file is decompressed and decrypted. The contents of the file are verified against the original test data by comparing the fourier coefficients using FFTCompareScript.m (to ensure it meets SP01).The input data is a file containing the compressed-encrypted data and the expected output is a file containing the same data as in the input file.

3.1.2 Compression and Encryption work on STM32F0

Aims to determine that the algorithms work together on the STM32F0, in line with AT07.

The algorithms are flashed onto the STM32F0 using the SHARC_buoy_algorithms project. This combined program encrypts and compresses a hard-coded input array and sends the compressed-encrypted data over UART to a computer. The expected output is compressed-encrypted data.

The data received at the computer is then decompressed and decrypted. The contents of the file are verified against the original test data by comparing the fourier coefficients using FFTCompareScript.m (to ensure it meets SP01).

3.1.3 Compressing and Encrypting IMU data

Aims to determine that data read from the IMU can be encrypted and compressed on the STM32F0, in line with AT09.

The SHARC_buoy project is updated to read live data from the IMU and send it to encryption. The data read from the IMU is first sent over UART to a computer before being encrypted and compressed. The compressed-encrypted data is then sent over UART to the computer.

The data received at the computer is then decompressed and decrypted. The contents of the file are verified against the recorded live data by comparing the fourier coefficients using FFTCompareScript.m (to ensure it meets SP01).

3.2 IMU

Aims to determine that data can be read from the sensor and passed to a PC in line with AT08

The SparkFun 9DoF IMU Breakout is connected to the STM32F0 via SPI as described in *Table 1*. The ICM is configured to a full scale resolution of $\pm 500dps$ and $\pm 2g$ as specified on page 132 of Jamie's thesis [2] to be compatible with the SHARC Buoy system. The ICM20948 project is flashed onto the STM32F0 and reads the x,y and z values of the gyroscope and the accelerometer only. The measurement data is passed over UART to the computer.

The tests detailed in *Table 2* are carried out in this experiment.

3.2.1 IMU initialisation

Aims to test that the IMU is working in line with IT01

A startup message is added after all IMU initialisation methods have been called and transmitted over UART. The STM32F0 is plugged in, flashed and the serial output is checked for the ”IMU initialised” message.

3.2.2 Acceleration Test

Aims to test that the IMU is working in line with IT02

The sensor is jolted in the positive and negative x,y and z directions. The live measurements are transmitted to the computer over UART. The readings are expected to increase in magnitude. The readings are expected to increase in magnitude rapidly and then decrease back to the steady-state value.

3.2.3 Gyroscope Test

Aims to test that the IMU is working in line with IT03

The sensor is rotated in the positive and negative x, y and z directions.The live measurements are transmitted to the computer over UART. For this measurement, the change in the reading should be monitored - the values should decrease in the negative direction and increase in the positive direction.

3.2.4 Pendulum Test

Aims to test that the IMU is working in line with IT04

The IMU, STM32F0 and FTDI are all placed on a bread board; carefully positioned to ensure the board hangs evenly when stationary. The board is then pulled up to one side and let go, allowing it to swing back and forth until coming to rest. The live measurements are transmitted to the computer over UART. The readings are expected to oscillate with oscillation amplitudes decreasing with time.

3.3 Wave test

Aims to test that the IMU is working in line with IT05

The IMU, STM32F0 and FTDI are all placed on a bread board and then placed on a sheet. The sheet is slowly moved up and down to simulate the swells of a wave. The corners of the sheet are lifted alternately to test different swell centres. The live measurements are transmitted to the computer over UART. The readings are expected to show the general trend of the swell of the wave with a clear rise and fall from the peak of the swell.

3.4 Compression

3.4.1 General

Aims to test that the compression subsystem works inline with AT02 and AT03.
The same tests as the previous report are conducted except the IMU data is used. For the compression part: The input is a csv file containing numerical measurement values taken over a period of 10 minutes. The expected output is an encoded lzss compressed file which is smaller in size than the original data. For the decompression part: The expected input is the lzss encoded file and the expected output is a csv file containing the same data as in the input csv file.

The fourier transform of the original and decompressed data is determined and the coefficients are compared to ensure that at least 25% of the coefficients have been preserved. This process is automated using the FFTCompareScript.m script.

3.4.2 Efficiency Testing

The input data would be an integer array of encrypted data and the expected output would be an integer array of encrypted-compressed data. The output is expected to be smaller than the input. The expectation is that as the file size increases the time take to compress the file will increase linearly but not exponentially.

3.5 Encryption

Testing efficiency and breakability of the encryption subsystem.

AT04 and AT05 require the evaluation of the encryption and decryption algorithms and validation that they do actually encrypt and decrypt the data. Both ATPs were tested and passed in the previous report.

3.5.1 Efficiency Testing

A further requirement of the system is to be efficient as there is minimal power available on the buoy. This means that the system should spend as little time as possible in an active state to conserve battery life. In order to test this, different sets of data with varying sizes will be encrypted and decrypted. The process of encryption will be timed (as only encryption will occur on the microcontroller) to find the optimal amount of data to be passed through the system at a time. The data sizes to be tested are:

The input data would be a character array of data read from the IMU and the output is expected to be an integer array of encrypted data to be passed to the compression algorithm.

3.5.2 Breakability Testing

The data that will be encrypted and compressed for transmission is not of a highly sensitive nature. For this reason, while encryption is important, it is not as critically important as other factors of the system design and will not be tested exhaustively. In order to ensure that the encryption does work, the FFT coefficents of the original data will be compared to those of the encrypted data. This will highlight that the encrypted data is not the same as the original data.

If breakability were to be tested, it could be checked by attempting to decrypt the data without knowing what the key is. This could be done by brute force - simply trying out different combinations for the private key (variables n and d), p and q. Each of these values were hard-coded to be an 8-bit value and thus, there would be $(2^8)^3 = 16777216$ different combinations of these variables. This would be quite an intensive process. It is noted that a longer key would be more secure, however, due to memory constraints on the STM32F0, the key was chosen to be 8 bits.

It must also be noted that due to the system design, the data will be encrypted and then compressed before it is transmitted from the STM32. This means that in order to access the data, one would have to know how to decompress it, the encryption used and the windows and keys used for compression and encryption respectively.

4 Results

4.1 Overall System Functionality

4.1.1 Sending data from STM32F0

AT01 PASS

Tihs was tested and discussed in progress report 1. The data was transmitted successfully to the PC over UART using the code available on our GitHub. It was verified using verify.py.

4.1.2 Interfacing IMU and STM32F0

AT08 PASS
In order to read from the sensor, data must be passed from the sensor to the STM32F0 and from the STM32F0 to the computer via UART.

Accelerometer and gyroscope data was read from the sensor and passed to the STM32F0 via SPI interfacing. The data was then transmitted to the PC over UART. The code used for this interfacing is available here. This code transmits data every 100ms from the sensor to the PC and the readings can be read using a serial terminal.

4.1.3 Full System Efficiency

The following results were obtained from timing the implementation of the encryption and compression algorithms using the HAL_GetTick function:

Table 3: Full System Efficiecy

Test	No. of Readings	No. of Characters	Ticks Taken
1	1	36	6
2	5	179	32
3	10	360	65
4	15	545	101
5	20	730	140

This data is plotted together with the individual efficiencies which are discussed under the individual subsystem results: *Figure 6*

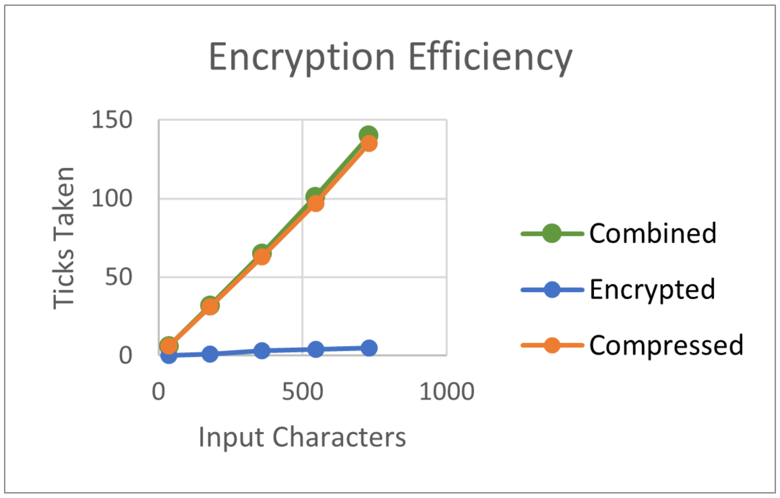


Figure 6: Plot of efficiency data

4.1.4 Compression and Encryption work together

AT06 and AT07 PASS

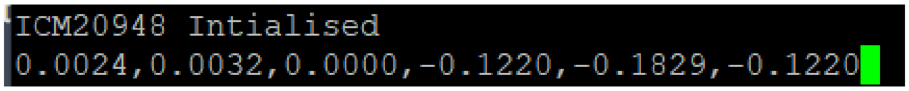
This was tested and discussed in progress report 1. The algorithms work together in a single c file both on a PC and on the STM32F0.

4.2 IMU

4.2.1 IMU initialisation

IT01 pass

The ICM20948 project is successfully updated to transmit the ”IMU initialised” message. *Figure 7* shows the message received at the computer after startup followed by the first reading from the sensor.



```
ICM20948 Intialised
0.0024,0.0032,0.0000,-0.1220,-0.1829,-0.1220
```

Figure 7: Diagram of Inter-Subsystem interactions

4.2.2 Acceleration Test

IT02 pass

The raw data used to plot *Figure 8* can be found here.

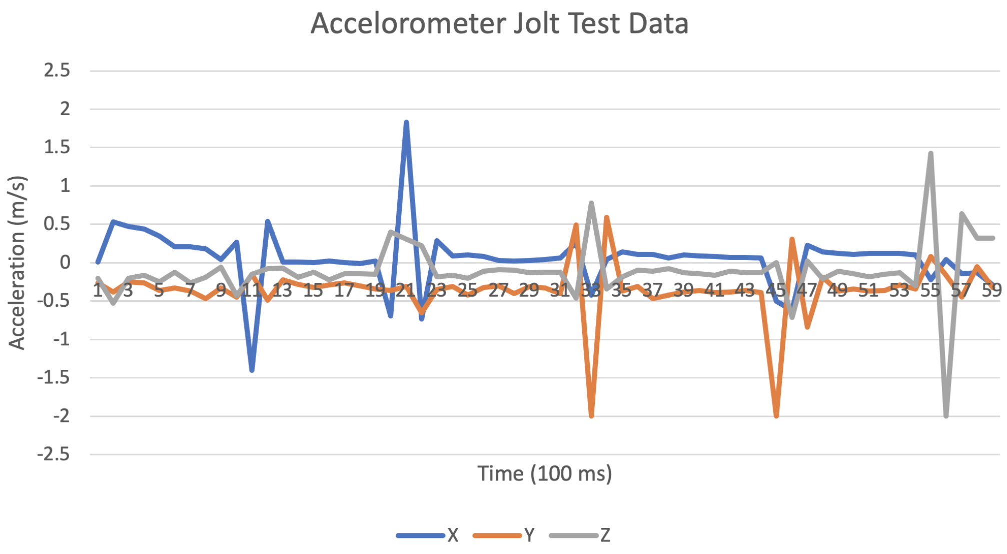


Figure 8: Plot of accelerometer jolt test data

Looking at *Figure 8* it is clear that the jolts are easily recorded by the sensor. Sharp movements in the positive and negative x, y and z directions are clearly shown by the spikes in the plot.

Ultimately, the acclerometer is able to read acceleration.

4.2.3 Gyroscope Test

IT03 pass

The raw data used to plot *Figure 9* and *Figure 10* can be found at: jolt and roll.

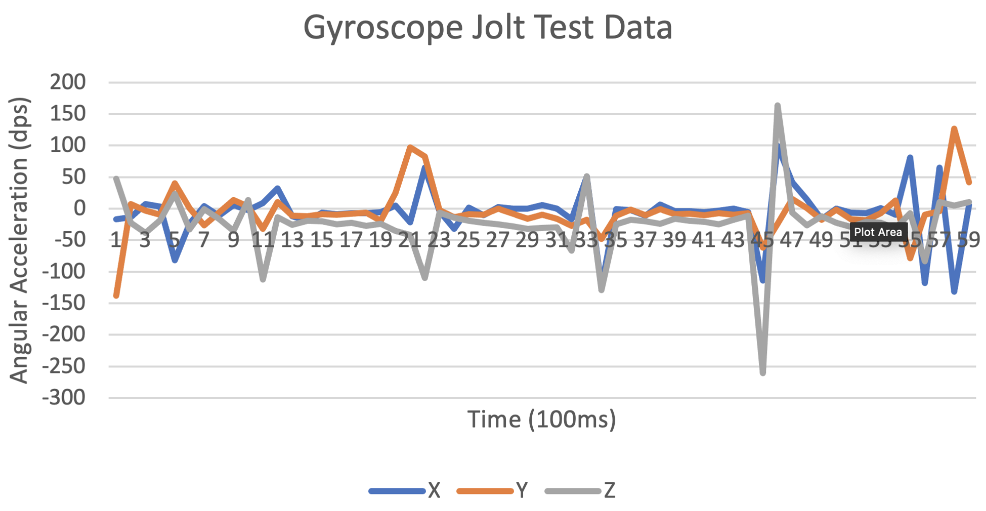


Figure 9: Plot of gyroscope jolt test data

Comparing the above data to *Figure 8* shows that the gyroscope does not show acceleration in one direction like what the accelerometer does. This is expected as the gyroscope measures angular acceleration. The spikes displayed show acceleration in more than one direction (angular acceleration). Looking at *Figure 8* again, the points where there are spikes in multiple axes correlates to the spikes seen in the gyroscope plot. The spikes at the end of the plot are likely due to noise.

A more suitable test for the gyroscope is the roll test discussed in the experiment setup. The results are plotted below: *Figure 10* below:

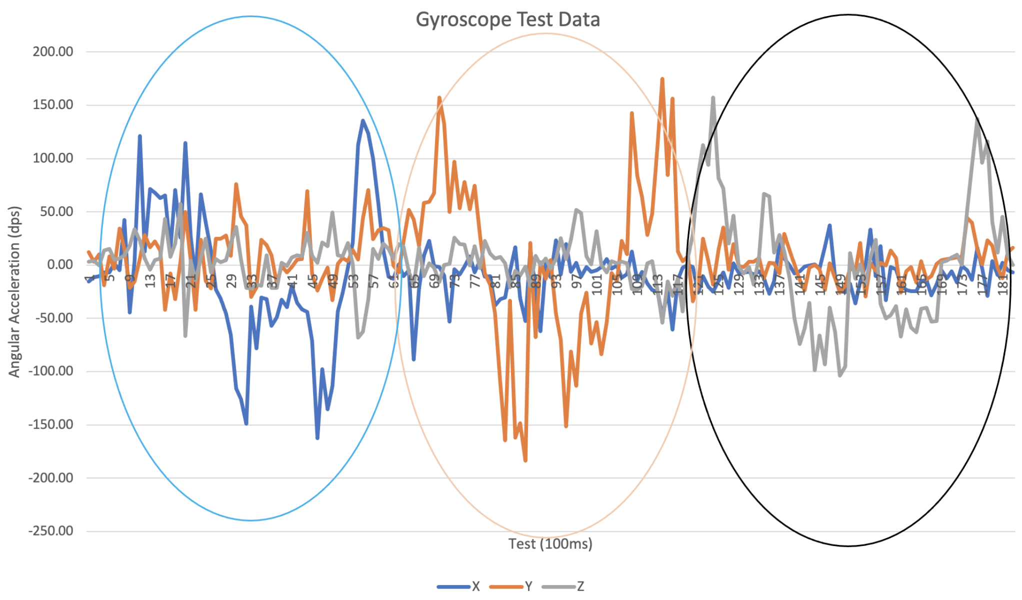


Figure 10: Plot of gyroscope roll test data

Looking at *Figure 10* it is clear where the sensor has been accelerated about a specific axes. The blue oval shows where the sensor has been rotated first anti-clockwise and then clockwise about the x axis. Focusing on the blue line, when the sensor is rotated anti-clockwise the general trend is decreasing and then on clockwise rotation the general trend is increasing. This is the expected outcome. Similar trends are seen in both the y and z axes as expected.

For each axes rotation there are also spikes in the other axes but these are not as significant. The simultaneous spikes in the other axes is due to noise caused by human motion. When rotating the sensor about a specific axes was difficult to keep it stable in all other axes which lead to the noise in the other axes.

Ultimately, the gyroscope is able to read angular acceleration.

4.2.4 Pendulum Test

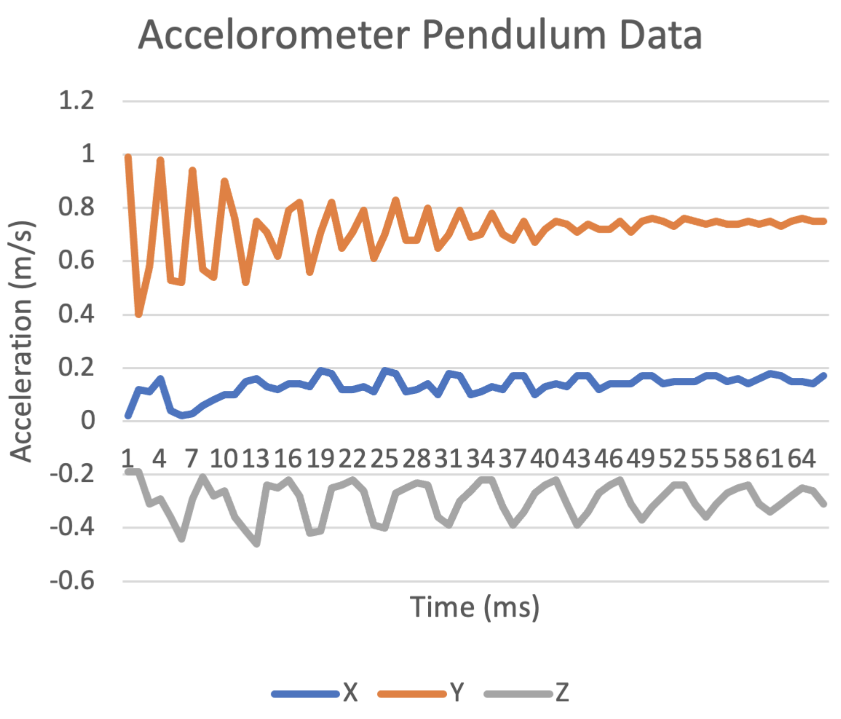
IT04 pass

The following data was obtained from the pendulum test. More readings were obtained but only 30 are included here as they sufficiently represent the data. The full raw data can be found here.

Table 4: IMU data

Reading	**Acc X (g)**	**Acc Y(g)**	**Acc Z(g)**	**Gyro X(dps)**	**Gyro Y(dps)**	**Gyro Z(dps)**
1	0,07	0,27	-0,11	39,24	-219,86	44,56
2	0,03	0,42	-0,3	34,21	-277,48	-140,96
3	0,03	0,11	-0,12	-39,76	-168,9	-132,23
4	0,04	0,35	-0,03	-12,4	-196,46	-81,63
5	0,02	0,42	-0,24	59,04	-170,72	-80,56
6	-0,01	0,19	-0,39	164,49	3,3	-135,69
7	0,06	0,32	-0,24	137,18	85,88	-61,86
8	0,07	0,41	-0,06	74,47	51,71	-78,37
9	0,01	0,22	-0,14	59,97	-77,19	-179,34
10	-0,03	0,28	-0,08	22,72	-147,63	-155,69
11	0,08	0,46	-0,17	34,9	-197,15	-73,479
12	0,02	0,22	-0,27	100,06	-204,96	-51,85
13	0,06	0,22	-0,24	141,63	-63,95	-91,48
14	0,05	0,39	-0,17	95,19	50,35	-90,698
15	-0,03	0,27	-0,14	46,58	-11,33	-140,84
16	0	0,27	-0,11	28,44	-81,44	-138,69
17	0,05	0,44	-0,1	34,56	-194,61	-89,28
18	0,03	0,3	-0,15	76,34	-244,5	-49,73
19	0,05	0,26	-0,32	122,82	-123,4	-104,75
20	0,04	0,39	-0,24	97,83	50,89	-95,10,75
21	-0,01	0,31	-0,13	58,98	42,56	-109,565
22	0,02	0,3	-0,12	44,44	-42,72	-131,185
23	0,04	0,4	-0,08	33,94	-167,31	-103,02
24	0,02	0,36	-0,14	64,29	-244,79	-51,28
25	0,03	0,27	-0,27	103,63	-173,13	-88,11
26	0,05	0,34	-0,25	90,02	-4,92	-93,6811
27	0,01	0,32	-0,15	62,27	35,42	-97,4711
28	0,03	0,33	-0,13	43,11	-19,65	-118,35
29	0,04	0,37	-0,08	38,64	-123,4	-99,27
30	0,03	0,34	-0,12	53,19	-227,21	-63,08

Using the data in *Table 4*, *Figure 11* has been plotted.



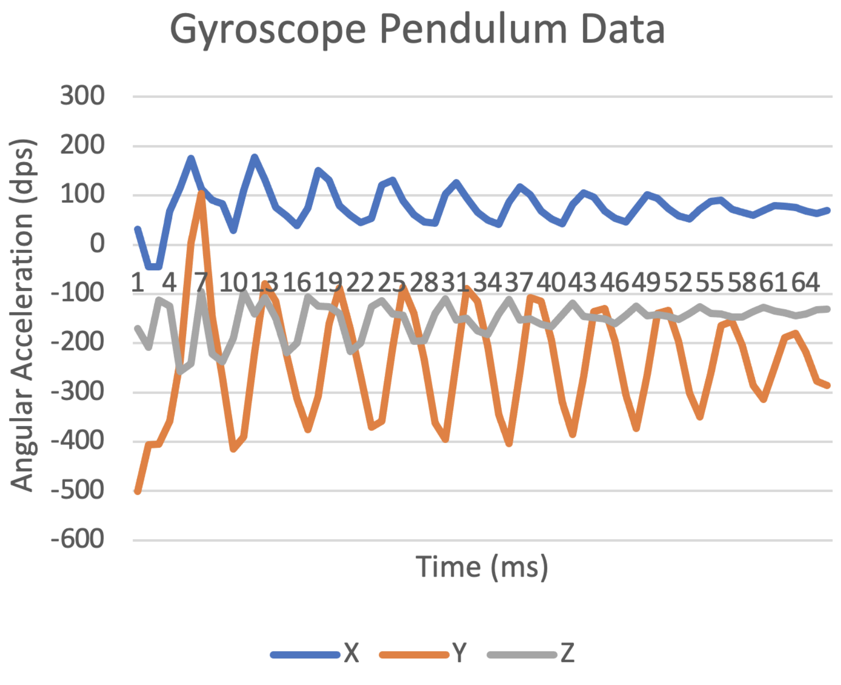


Figure 11: Plot of accelerometer and gyroscope pendulum test data

Looking at *Figure 11*, the recorded sensor data shows clear oscillations as expected. The acceleration plot shows oscillations with decreasing amplitudes. This is expected since a pendulum moves back and forth before coming to a stop. The pendulum was swung in the x-y plane which is shown by the strong oscillation trend in the y line. However, whilst running the experiment, the pendulum did not swing perfectly in the x-y plane which resulted in the oscillations in the z axis.

The gyroscope plot shows angular acceleration oscillations in all axes. The pendulum swings in multiple planes (x-y,y-z,x-z) throughout its movement. Over time, as the pendulum slows down, the amplitude of the oscillations decrease. This is as expected for a pendulum.

Ultimately, the accelerometer and gyroscope can take increasing and decreasing measurements relative to motion of the sensor.

4.2.5 Wave Test

IT05 pass

The following data was obtained from the wvae test. More readings were obtained but only 30 are included here as they sufficiently represent the data. The full raw data can be found here.

Table 5: IMU data

Reading	**Acc X (g)**	**Acc Y(g)**	**Acc Z(g)**	**Gyro X(dps)**	**Gyro Y(dps)**	**Gyro Z(dps)**
1	0.46	-0.84	-0.76	13.13	-19.71	-1.13
2	0.45	-0.80	-0.75	11.34	-24.79	0.46
3	0.48	-0.86	-0.71	12.79	-18.03	-0.85
4	0.44	-0.74	-0.63	17.01	-42.35	10.50
5	0.51	-0.85	-0.70	14.08	39.07	-38.23
6	0.39	-0.76	-0.44	11.51	6.11	-24.26
7	0.44	-0.62	-0.25	47.54	-51.30	32.38
8	0.34	-0.66	-0.47	-26.78	2.11	3.858
9	0.38	-0.70	-0.50	-27.37	-30.82	-1.53
10	0.45	-0.59	-0.33	18.81	-12.82	-25.51
11	0.22	-0.58	0.01	62.12	-13.34	15.981
12	-0.32	-0.11	0.03	-11.15	-3.51	0.631
13	-0.07	-0.29	-0.74	34.87	29.13	-83.24
14	0.27	-0.89	-0.64	-137.34	-230.66	21.02
15	0.20	0.32	-0.27	121.92	-61.82	-57.542
16	0.49	0.10	-0.65	-5.79	12.23	64.11542
17	0.71	-0.78	0.02	-216.38	179.85	13.332
18	0.81	0.12	0.61	37.24	-34.89	-107.372
19	-0.03	-0.39	-0.17	-9.42	8.90	10.052
20	0.09	-0.45	-0.22	-8.23	-5.94	-7.162
21	0.01	-0.63	-0.19	22.26	-27.11	18.672
22	-0.29	-0.87	0.36	-133.25	107.85	-20.76
23	-0.27	-0.75	-0.77	33.74	-90.24	-37.37
24	-0.80	0.22	0.65	328.72	-323.44	201.56
25	-0.21	0.24	0.59	72.15	-174.03	45.216
26	-0.04	-0.07	-0.75	84.17	-5.25	-5.406
27	-0.35	0.85	0.02	-193.71	15.34	16.636
28	0.27	-1.54	0.66	-66.87	52.98	-106.76
29	0.34	-0.21	-0.09	-228.85	-34.64	-6.46
30	0.12	-0.50	-0.66	-128.31	-17.51	34.79

Using the data in *Table 5*, *Figure 12* has been plotted.

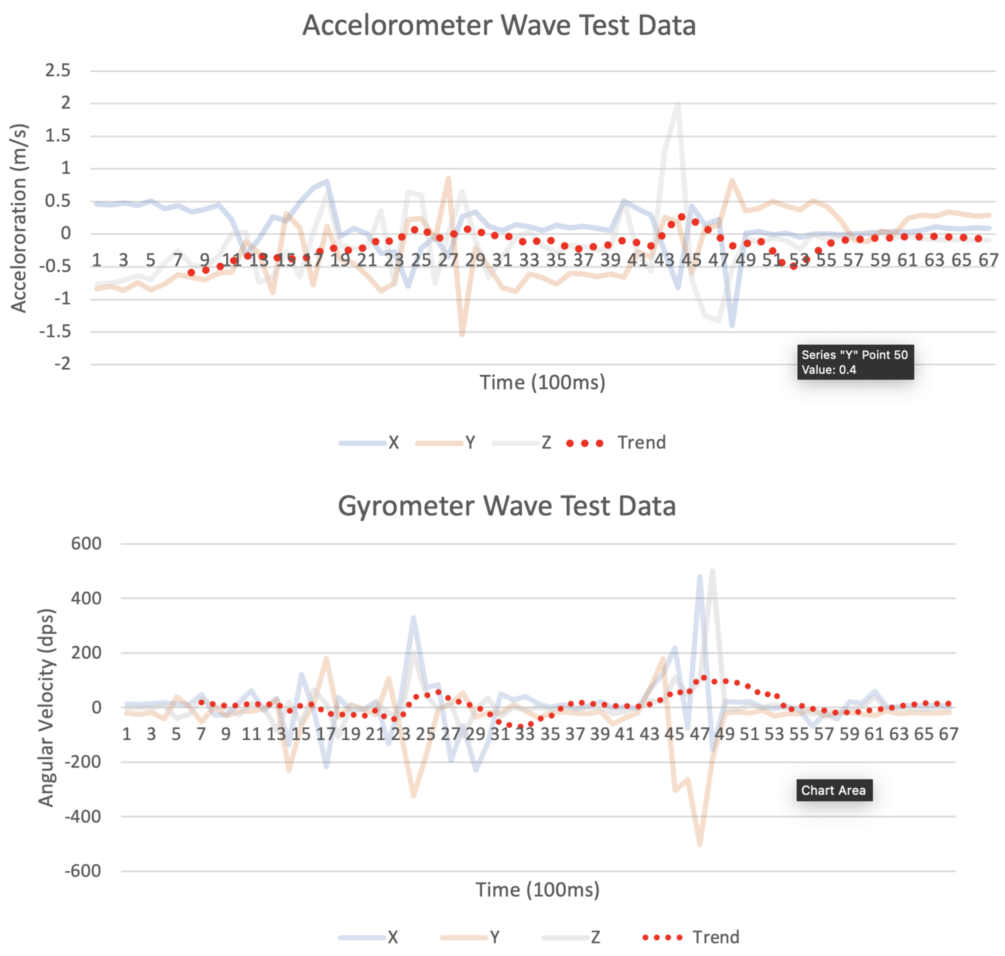


Figure 12: Plot of accelerometer and gyroscope pendulum test data

Looking at *Figure 12*, the recorded sensor data shows clear wave like trends as expected. Weighted average trend lines were added to both of the plots to show the general trend in the data. The motion of the wave causes acceleration in all three axes at the peak of the swell. From the peak, there is a gentle deceleration and then acceleration into the next swell. The trend lines are not as prominent as expected but this is likely due to the method used not being able to imitate wave swells well.

However, the general trends are there. The gyroscope shows the rise and fall to and from the peak of the swell. The peak of the swell is indicated by sharp spikes in all of the axes. Acceleration does not show the wave swells as clearly but rather indicates the general idea that there is acceleration in multiple directions at all times throughout the swell.

Ultimately, the sensor is able to sense slow changes in movement and still indicate the peaks of these slow changes.

4.3 Compression

subsubsectionGeneral *AT02 and AT03 Pass.*
This was tested and discussed in progress report 1. The algorithm works as expected.

4.3.1 Efficiency Testing

The following results were obtained from timing the compression algorithm using the HAL_GetTick function:

Table 6: Compression Efficiency

Test	No. of Readings	No. of Characters	Ticks Taken
1	1	36	6
2	5	179	31
3	10	360	63
4	15	545	97
5	20	730	135

This data is plotted: *Figure 13* below:

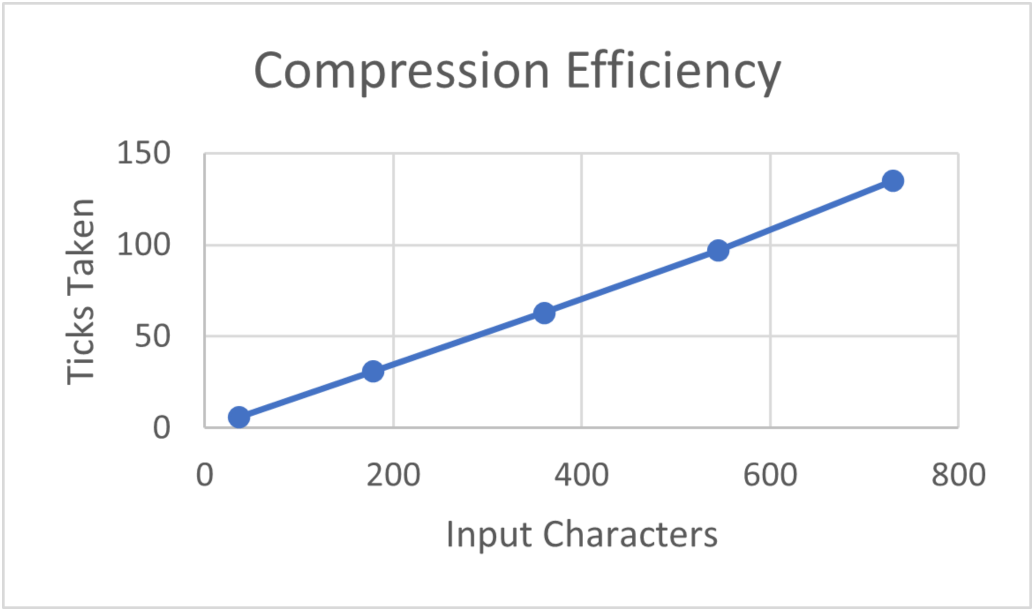


Figure 13: Plot of compression efficiency data

This is the expected trend and shows clear efficiency in the data. If the trend was exponential it would not be efficeint.

4.4 Encryption

4.4.1 General

AT04 and AT05 PASS.
This was tested and discussed in progress report 1. The algorithm works as expected.

4.4.2 Efficiency Testing

The following results were obtained from timing the encryption algorithm using the HAL_GetTick function:

Table 7: Encryption Efficiecy

Test	No. of Readings	No. of Characters	Ticks Taken
1	1	36	0
2	5	179	1
3	10	360	3
4	15	545	4
5	20	730	5

This data is plotted: *Figure 14* below:

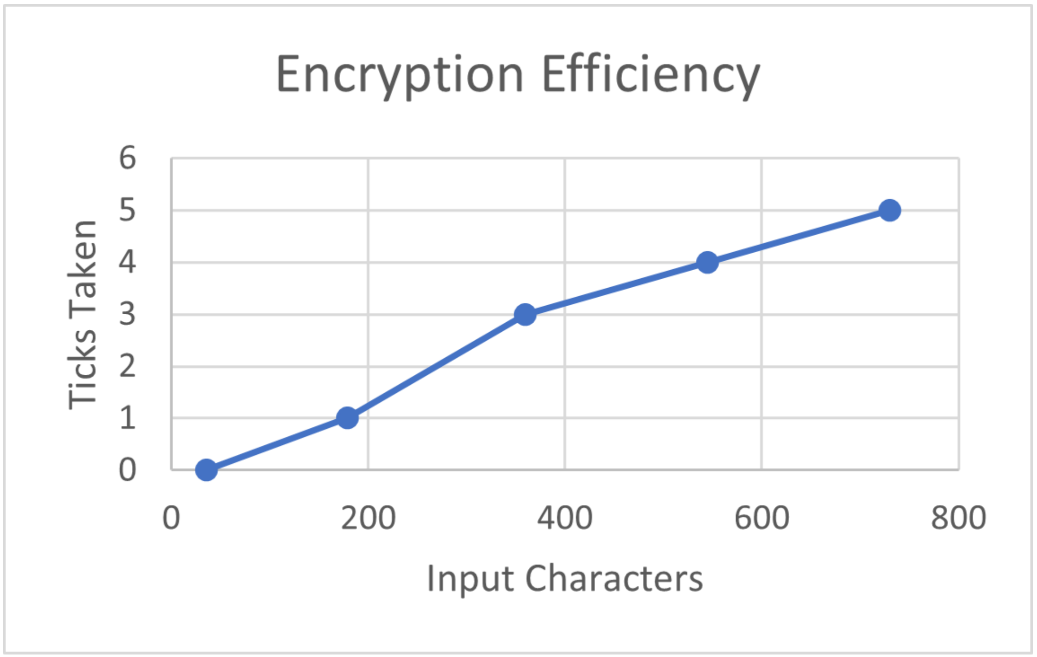


Figure 14: Plot of encryption efficiency data

5 Acceptance Test Procedures

Success and Failure tests:

Table 8: Assessment of ATP success or failure

Acceptance test	System	Success
AT01	Input	PASS
AT02	Compression	PASS
AT03	Compression	PASS
AT04	Encryption	PASS
AT05	Encryption	PASS
AT06	Full System	UNTESTED
AT07	Full System	UNTESTED
AT08	IMU	UNTESTED
AT09	Full system	UNTESTED

Overall, the order of ATPs 6-9 has been changed from previous reports. We felt that this fixes the fluidity of the ATPs as the previous order did not reflect the progression of testing.

AT06, AT07, AT09: updated to reflect the new system design i.e., compression and encryption being swapped around.
AT08: is a new ATP, added to test that the STM32F0 interfaces successfully with the IMU before adding the encryption and compression algorithms. This is a necessary test to minimise faults that could occur when combining the entire system. It also decreases fault testing that needs to be done should there be complications when adding in the encryption and compression algorithms.

Table 9: Acceptance Test Protocols used to determine that the system works to specification

Acceptance test	System	Description
AT01	Input	Ensure data is transferred correctly and completely from computer to STM32F051
AT02	Compression	Ensure that at least 25% of lower Fourier coefficients of data are preserved
AT03	Compression	Ensure file size is decreased after compression
AT04	Encryption	Ensure that the data is not readable without the key after encryption
AT05	Encryption	Data is readable and the same as the original once decrypted
AT06	Full system	Ensure compression and encryption works when implemented together on computer
AT07	Full System (no HAT)	Ensure compression and encryption works when implemented together on STM32F0
AT08	IMU sensor	Ensure that data is read from sensor and sent to computer
AT09	Full system	Ensure that data read from the sensor completes a full cycle meeting all requirements and above ATPs

5.1 AT01

AT01	**Connection Test**
Evaluation type	Comparison
Target	STM32F051
Test Protocol	1. Set up test case using a sample set of data 2. Pass data from device to STM32F051 3. Pass data from STM32F051 to computer. 4. Compare data read and passed to computer to original sample set of data.
Pass Condition	• Data is in the same format as original data • Data values are the same as original data • All original data is present in the new data
Fail condition	• Data format differs from original • Any data value is different from the original data • Any data values are missing Extra data values are present in the new data

5.2 AT02

AT02	**Fourier Coefficients Test**
Evaluation type	Comparison
Target	Compression Subsystem
Test Protocol	1. Set up test case using a sample set of data and take 25% the lower Fourier coefficients. 2. Compress the original data using the compression algorithm on the STM32F051. 3. Decompress the data and compute the Fourier coefficients on a computer. 4. Compare the Fourier coefficients to the test case.
Pass Condition	• All of the Fourier coefficients present in test case are present in decompressed data (extra coefficients may be present but not less).
Fail condition	• Fourier coefficients present in the test case are not present in the decompressed data.

5.3 AT03

AT03	**Compression Ratio Test**
Evaluation type	Comparison
Target	Compression Subsystem
Test Protocol	1. Set up test cases using sample set of data and make note of file sizes. 2. Compress the original data using the compression algorithm on a computer. 3. Compare the size of the new file to that of the original file.
Pass Condition	• New file size is less than original file size.
Fail condition	• New file size is greater than or equal to old file size.

5.4 AT04

AT04	**Encryption key Test**
Evaluation type	Algorithm Validation
Target	Encryption Subsystem
Test Protocol	1. Set up test cases using sample set of data. 2. Encrypt data using the encryption algorithm and key on the STM32F051. 3. Check if data can easily be deduced or read after encryption on a computer.
Pass Condition	• Data is not deducible or human readable after encryption.
Fail condition	• Data can easily be deduced or read after encryption.

5.5 AT05

AT05	**Decryption validation Test**
Evaluation type	Algorithm Validation
Target	Encryption Subsystem
Test Protocol	1. Set up test cases using sample set of data. 2. Encrypt data using the encryption algorithm and key on the STM32F051. 3. Decrypt data using the decryption algorithm and key on a computer. 4. Compare new data to original data set.
Pass Condition	• Decrypted data is identical to original sample set data.
Fail condition	• There are differences between sample set data and decrypted data.

5.6 AT06

AT06	**System Test**
Evaluation type	System execution
Target	Full System (no HAT and tested on computer)
Test Protocol	1. Set up test case using sample data set and compute 25% of the lower Fourier coefficients. 2. Pass sample data through encryption which must then pass onto compression algorithm without user interaction. 3. Perform decryption and then decompression on the data. 4. Compute 25% of lower Fourier coefficients of the data. 5. Compare the coefficients to the test case.
Pass Condition	• Encryption and compression and decryption and decompression complete without failure. • AT01 is met.
Fail condition	• Compressed data is not passed to encryption algorithm without human interaction. • Any of the algorithms fail. • Less than 25% of coefficients are retained. • AT01 is not met.

5.7 AT07

AT07	**System Test**
Evaluation type	System execution
Target	Full System (no HAT)
Test Protocol	1. Set up test case using sample data set and compute the Fourier coefficients. 2. Pass sample data through encryption and then compression algorithms on the STM32F051. 3. Perform decompression and then decryption on the computer. 4. Compute Fourier coefficients of the data. 5. Compare the coefficients to the test case and ensure that at least 25% of the coefficients are retained.
Pass Condition	• Encryption and compression and decryption and decompression complete without failure. • AT01 is met.
Fail condition	• Any of the algorithms fail. • Less than 25% of coefficients are retained. • AT01 is not met.

5.8 AT08

AT08	**Reading data from IMU sensor on STM32F0**
Evaluation type	System Execution
Target	Full System
Test Protocol	1. Data is read from the IMU sensor and sent to a computer by the STM32F051 2. Data from the IMU sensor is formatted. 3. Formatted data is sent from STM32F0 to computer.
Pass Condition	• Data is successfully read from the sensor and printed to the computer.
Fail condition	• Data is not read from the sensor (there is no change in the data or no data is printed to the computer) • Data is not transmitted correctly when formatted

5.9 AT09

AT09	**System Test with IMU sensor**
Evaluation type	System Execution
Target	Full System
Test Protocol	The IMU is connected to the STM32F051. The data from the sensor is first printed to the computer and then sent to be compressed (onboard the STM32F051). 1. The data printed to the computer is used as the sample data set and 25% of the Fourier coefficients are computed. 2. Data is sent through encryption and compression on the STM32F051 and then transferred to the computer. 3. Perform decompression and then decryption on the computer. 4. Compute 25% of lower Fourier coefficients of the data. 5. Compare the coefficients to the test case.
Pass Condition	• At least 25% of Fourier coefficients are retained after full system processing.
Fail condition	• AT06 is not met • AT07 is not met • AT08 is not met

References

[1] Waveshare, “Sense HAT (B) for Raspberry Pi, Multi Powerful Sensors,” [Online]. Available: https://www.waveshare.com/sense-hat-b.htm, [Accessed: 17 August 2022].

[2] J. N. Jacobson, “Sharc buoy: Robust firmware design for a novel, low-cost autonomous platform for the antarctic marginal ice zone in the southern ocean,” Ph.D. dissertation, Rondebosch, Cape Town South Africa, 2021.

6 Appendix A: Requirements

The ATP changes have been reflected in the traceability matrix below:

Table 10: Requirement Traceability Matrix

UR	FR	Specifications	ATP
UR01	FR01	SP01	AT02
UR02	FR06	SP06	AT09
UR03	FR05	SP05	
UR04	FR03, FR04	SP04	AT04, AT05
UR05	FR01, FR02	SP02	AT03
UR06	FR07	SP07	AT01

User Requirements

Table 11: User requirements used to determine the design and functionality of the desired IP

User Requirement	Description
UR01	System must minimise data loss
UR02	System must be compatible with ICM-20649 and ICM20948 IMU chips
UR03	System must use minimal computation to be power efficient
UR04	System must encrypt the data
UR05	System must compress data
UR06	System must be able to transmit data to another device

Functional Requirements

Analysis of the user requirements resulted in the following functional requirements which describe how the system will function.

Table 12: Requirements addressing the needs of the system

Requirement	Description
FR01	System to compress data such that 25% of the lower Fourier coefficients are recoverable after decompression
FR02	Compression subsystem to pass processed data to the encryption subsystem
FR03	All compressed data must be encrypted
FR04	Encryption subsystem to encrypt data in a manner that allows efficient and complete decryption
FR05	System must use minimal computation and processing
FR06	All code must be implementable for a system using either ICM-20649 or ICM-20948
FR07	System to transmit data via USB to another device for decryption and decompression

Specifications

Table 13: Specifications of design to repeat user requirements

Specification ID	**Description**
SP01	A minimum of 25% of the lower Fourier coefficients must be present after decompression
SP02	The compression ratio must be less than 1, i.e. the compressed file must be smaller than the original file.
SP03	100% of the compressed data must be recovered after decryption.
SP04	Less than 10% of the original data can be the same as the original file after encryption.
SP05	All code should be written in C
SP06	I2C must be run on fast mode when developing and testing (ICM-20649 standard)
SP07	System must transfer the data over UART following encryption