## **Data Processing Block - Data Processing**

When the data is received in this block it goes through several stages of processing before it is ready to be transmitted. These stages include: identification and error checking, concatenating data samples, applying IMU specific offsets and configurations to the data received from a particular IMU, calculating orientation from gyroscope and accelerometer data and finally fusing the data from the two using a complementary filter.

### **Identification and Error Checking**

Before the extracted data can be converted into useful orientation data, it has to be checked for errors first and the origin of the data must be identified in order to apply correct offsets and configurations to the data based on the IMU the data came from. As mentioned in the theory section of this report, each IMU has unique offsets due to manufacturing and different states of degradation. These offsets need to be eliminated in order for the data to be accurate.

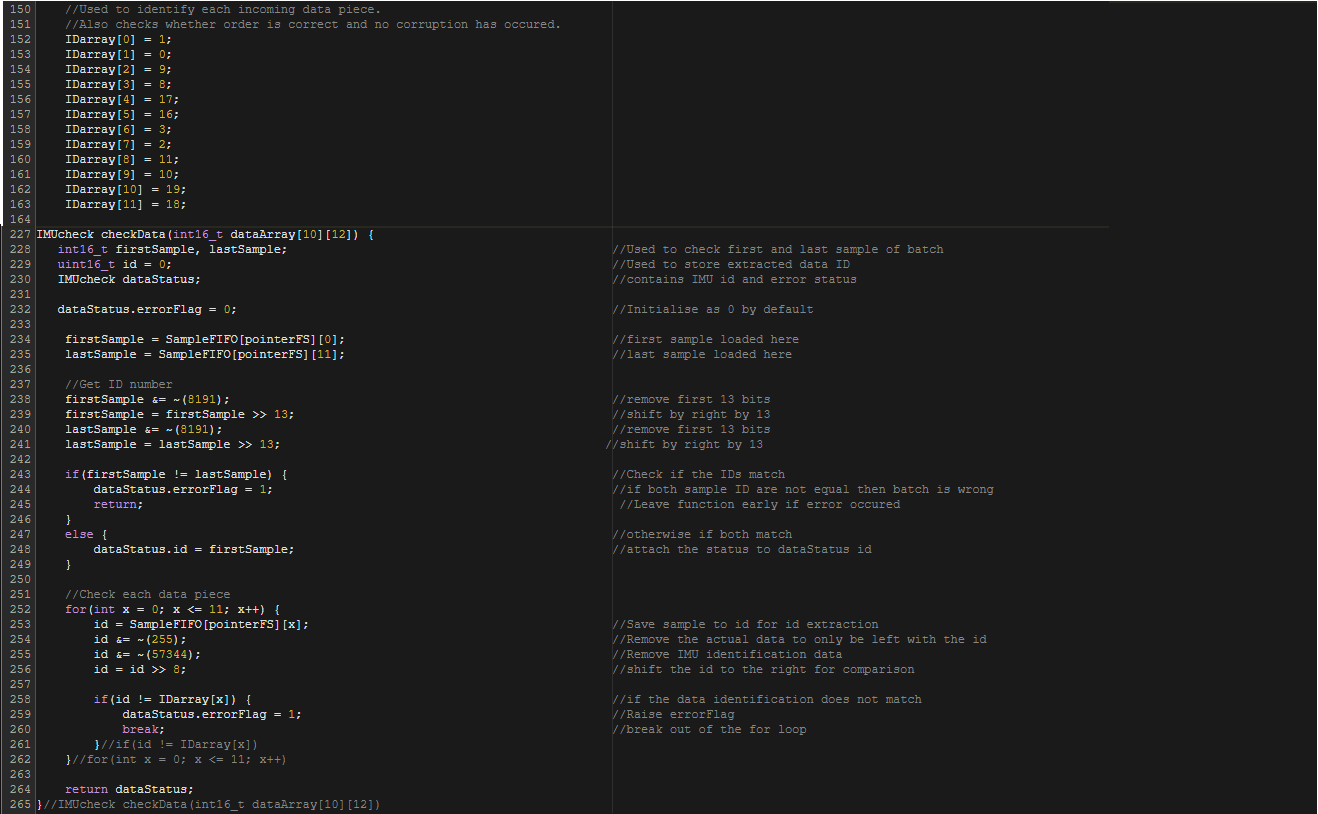
There are two simple checks employed to determine whether the data has undergone corruption. The first check is for alignment errors. The program checks the ID of the first and last sample in the data set obtained from the identification byte appended to each data piece in the Data Extraction Block. If the ID of the two does not match, data misalignment has occurred during transmission and therefore the data set is discarded. This check prevents the data from two different internal measurement units from being mixed which would cause errors in the orientation estimation. If the data is aligned, the second check is performed to determine whether any piece of data has been corrupted during transmission by checking bits 0 to 4 of the identification byte of each data piece against a table to determine whether a change has occurred. Because each data piece in the batch (of 12) has a unique combination of these 5 bits, which are always transmitted in the same order no matter which IMU the data comes from, an error can be determined if the value in any of these bits, in any data piece has changed or the order in which the they have been transmitted is not the same. If an error is detected in the second test, the data is also discarded. If, however, no errors are detected then the data is not discarded and is sent to the next stage of processing. The extract below depicts the code used for error checking.

Figure 1: Code used to check whether data has been corrupted during transmission.

### **Concatenating Data**

Because data is extracted from the IMU in two parts, the most significant byte and the least significant byte, the two bytes needs to be concatenated together to make the data extracted usable. This is done by removing the identification bytes from the two halves, shifting the most significant byte to the left 8 spaces to make it the upper byte of the 16-bit data sample and lastly add the two bytes together. Figure to demonstrates how this is accomplished in software.

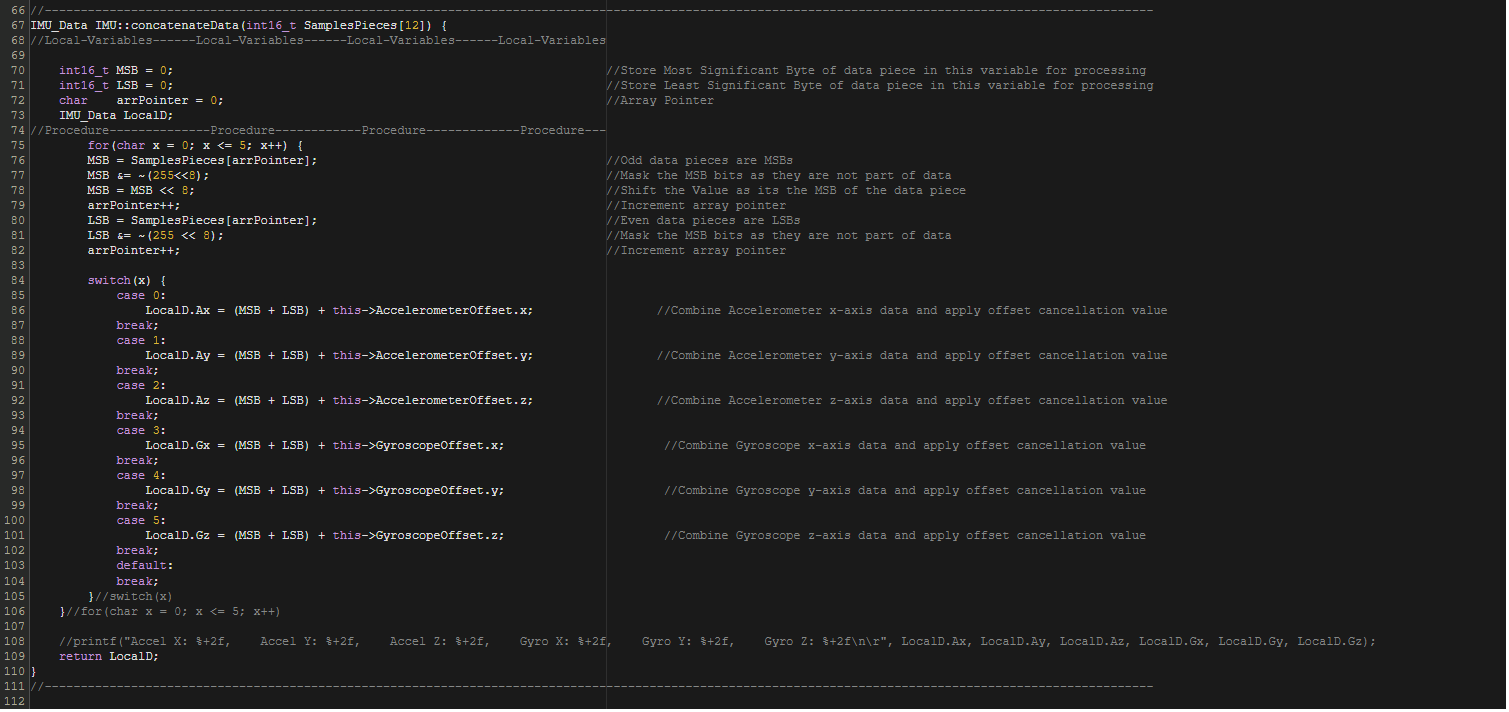
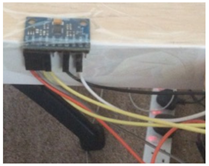


Figure : C++ implementation of byte concatenation.

### **Applying Internal Measurement Unit Specific Parameters**

In order to determine the offsets of each IMU, each device was left stationary, facing up and parallel to the floor. This was done to ensure that there are no disturbances, rotations or tilt which could affect the measurement of the offset values by introducing errors. 5000 samples were collected and averaged in real-time using the equation below. The results obtained from this procedure provided the offset values for x, y and z axes of the gyroscope and accelerometer.

Equation 1: Ongoing average equation. represents the average and represents the current sample and n represents the sample number.

Figure 3: Image depicting the IMU attached to the table face up.

An IMU class was created in order to create IMU object in software that contain the IMU specific parameters such that when the orientation calculations are performed these offsets are added to the raw values data beforehand. Another parameter that needed to be considered was the sensitivity scale factor of each internal measurement unit. This setting, in essence, determines the maximum angular velocity (for gyroscope) and the maximum acceleration (for accelerometer) that can be measured. The setting in each IMU is by default zero which means that the gyroscope can measure angular velocity up to 250 degrees per second and the accelerometer can measure accelerations up to 2g. These settings were not changed as they were adequate for the application. Because the data has received from the IMUs is 16-bits wide in 2’s complement format, the value range is between -32,768 and 32,767. At a setting of zero for the sensitivity scale factor in the gyroscope, multiples of 131 are measured for each integer increase in angular velocity. What this means is that for a measured angular velocity of 2 degrees per second, the value measured would 262 (or -262 depending on the direction of rotation) or for angular velocity of 250 degrees per second the measured value would equal to 32,767 (or -32,768 depending on direction). For the accelerometer the value is 16,384 per g.

These raw accelerometer and gyroscope values need to be converted to acceleration and degrees per second respectively to obtain correct orientation angles in the later stage of processing. This conversion can be done by simply multiplying the raw accelerometer and gyroscope values by the reciprocal of their respective sensitivity scale factor. Figure 3 depicts the initialisation of the IMU objects with their unique offsets. To view all code related to the application of offsets and parameters visit **(Include link to related section).**

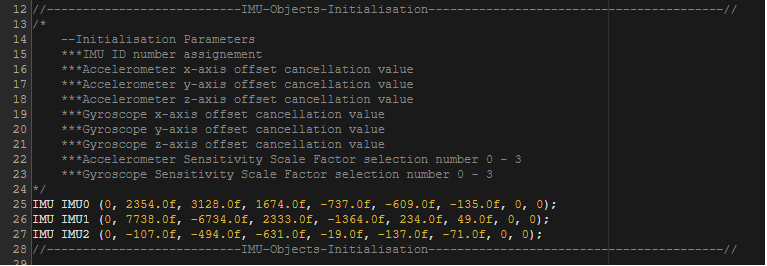


Figure :

### **Calculating Orientation and Fusing Data**

The last step to data processing is the orientation calculation and data fusion using the complementary filter. The following figures depict the implementation of the accelerometer, gyroscope and the complementary filter theory.

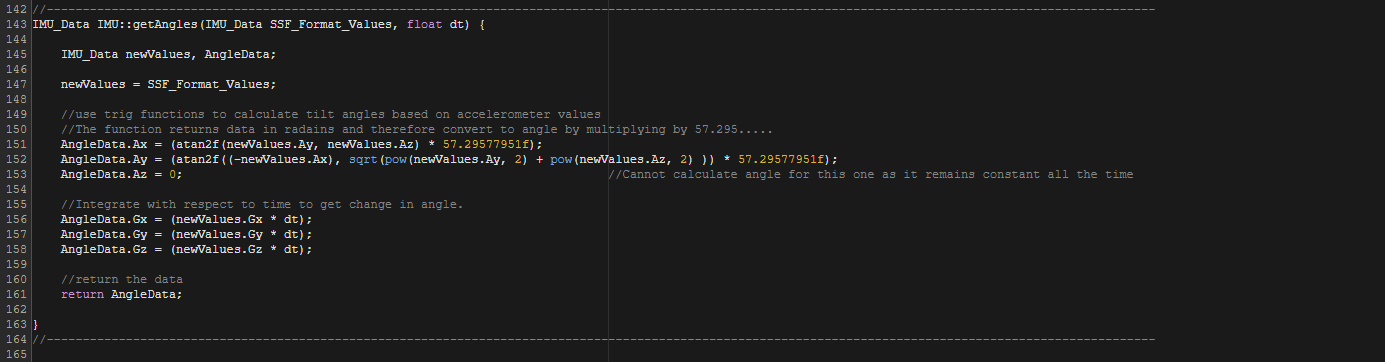


Figure : Calculating tilt angle and change in angle for complementary filter.

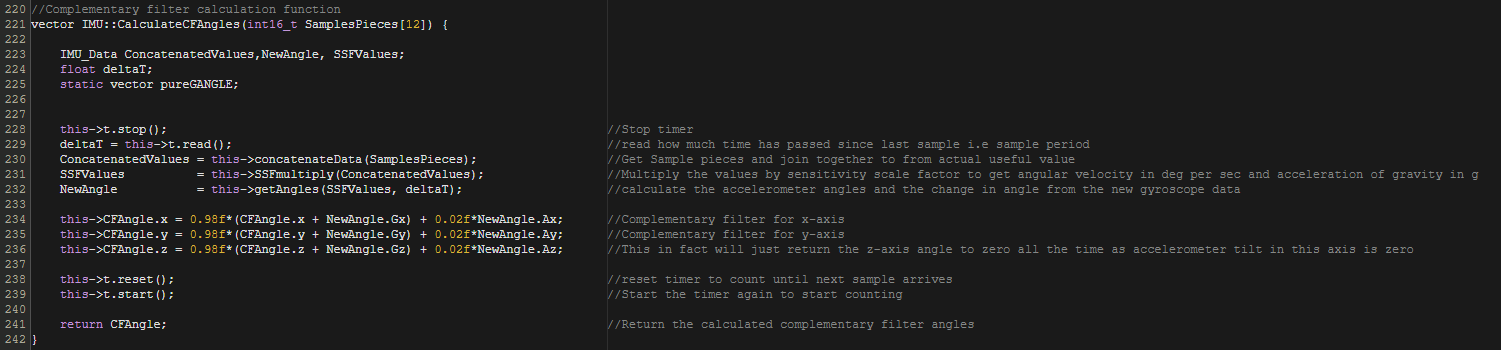


Figure : Fusing the gyroscope data and accelerometer data using the complementary filter.