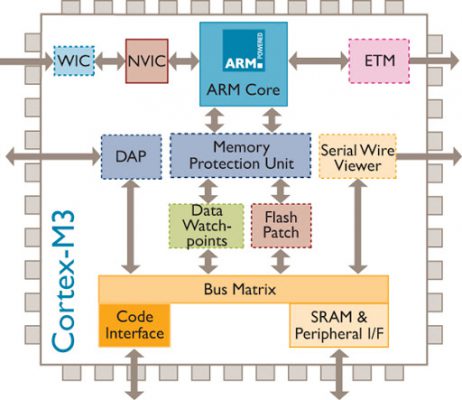
**KIẾN THỨC CƠ BẢN**

1. Kiến thức về vi điều khiển

1.1 Giới thiệu về vi điều khiển ARM Cotex M



**Hình 3.1** Kiến trúc của ARM Cortex M3 – Một trong những dòng vi điều khiển sử dụng rất phổ biến hiện nay

1.2. Ngôn ngữ lập trình C

Với các dòng vi điều khiển dùng cho các sản phẩm IOT, các yếu tố thường dùng để một sản phẩm có thể đem lại lợi ích về mặt kinh tế đó chính là hiệu suất cao nhưng giá thành phải thấp. Điều đó dẫn đến việc loại ngôn ngữ dùng để lập trình cho các dòng vi điều khiển phải có sự tối ưu về mặt tốc độ thực thi, tiết kiệm bộ nhớ. Ngôn ngữ C chính là ngôn ngữ đáp ứng được các yếu tố đó.

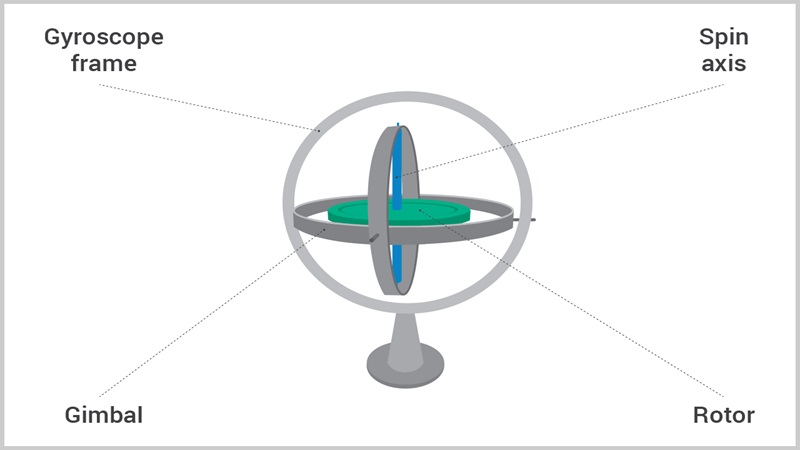


**Hình 3.2** Kí hiệu của ngôn ngữ lập trình C

2. Cảm biến gygroscope, cảm biến gia tốc và cảm biến từ trường

2.1. Cảm biến Gygroscope

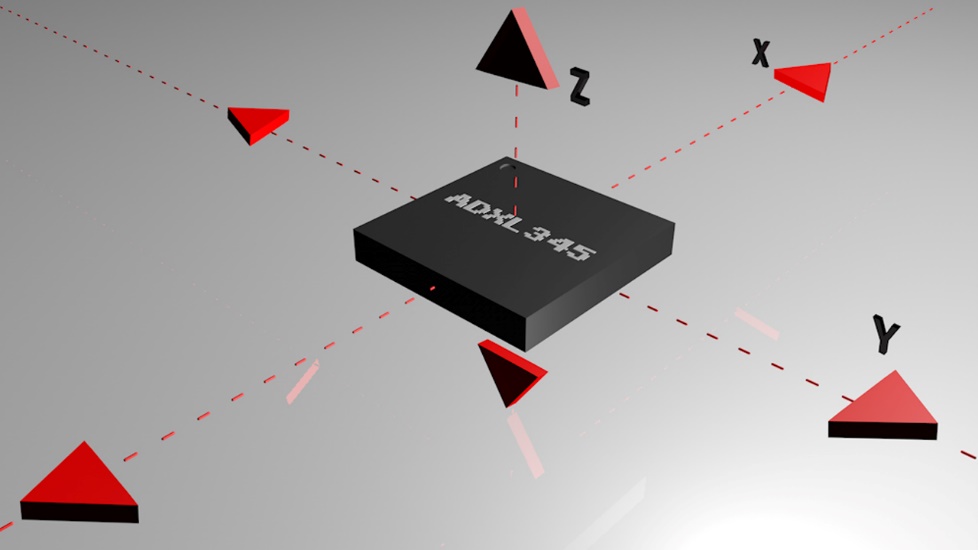
Cảm biến con quay hồi chuyển, thường được gọi là “gyroscope sensor” trong tiếng Anh. Đây là một thiết bị được thiết kế để đo và xác định góc quay hoặc thay đổi góc quay. Đặc điểm nổi bật của cảm biến này là khả năng cung cấp thông tin về ba trục không gian: trục x, trục y và trục z. Nhờ vậy, cảm biến có khả năng xác định hướng và sự chuyển động của một thiết bị trong không gian một cách chính xác.



2.2. Cảm biến gia tốc

Cảm biến gia tốc, hay còn gọi là cảm biến gia tốc kế, là một thiết bị điện tử được sử dụng để đo gia tốc, tức là thay đổi về tốc độ hoặc hướng chuyển động. Nó cho phép phát hiện khi vật thể tăng tốc, giảm tốc hoặc thay đổi hướng di chuyển.

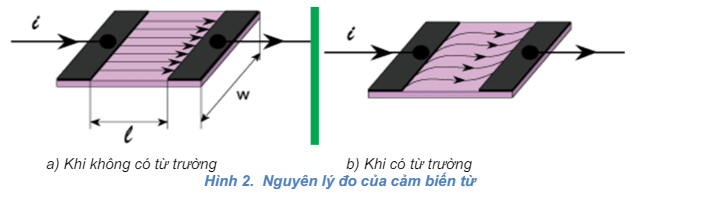
Cảm biến gia tốc ghi lại thông tin về sự biến đổi này, giúp các thiết bị điện tử hiểu và phản ứng với các chuyển động xung quanh. Ứng dụng của cảm biến này rất đa dạng, từ công nghiệp ô tô cho đến điện thoại thông minh và thiết bị y tế



2.3. Cảm biến từ trường

La bàn số sử dụng hai cảm biến từ trường nằm vuông góc nhau trên mặt phẳng nằm ngang để đo cường độ từ trường của trái đất. Tỉ số hai giá trị đo này cho phép tính được góc giữa các trục của cảm biến với đường sức từ của trái đất, đó chính là hướng (số chỉ) của la bàn từ.

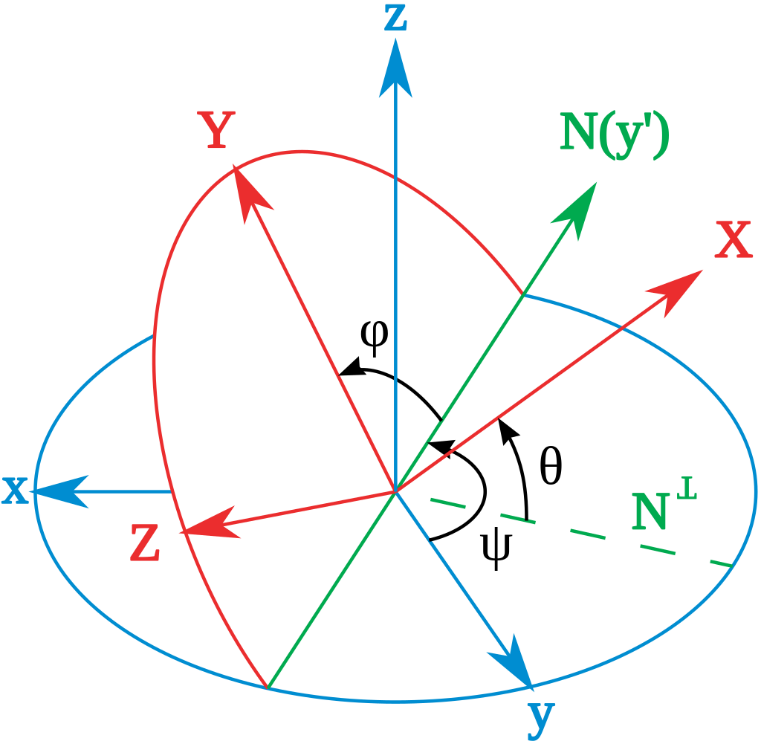
Trên thị trường hiện nay có nhiều loại cảm biến từ. Thông thường các cảm biến từ này có gắn 3 cảm biến dọc theo ba trục của hệ tọa độ Decac và có thể đo với độ nhạy rất cao. Cảm biến được chế tạo bằng hợp chất InSb (hợp chất của In – Indium và Sb – Antimony)[4] . Hợp chất này có tính chất như sau:

Khi cho dòng điện chạy qua hợp chất, nếu không có từ trường các điện tử sẽ chuyển động theo đường thẳng. Khi đặt hợp chất trong từ trường, dưới tác động của từ trường, các điện tử sẽ chuyển động trên các đường chéo, quãng đường chuyển động này dài hơn làm cho điện trở của vật liệu tăng lên. Người ta sử dụng hiện tượng này để chế tạo cảm biến đo cường độ từ trường 

3. Biểu diễn hướng của vật thể trong không gian

3.1. Biểu diễn hướng bằng 3 góc Euler

Góc Euler là ba góc được giới thiệu bởi Leonhard Euler để miêu tả định hướng của một vật thể rắn. Để miêu tả như một định hướng trong không gian ba chiều Euclide theo ba tham số được yêu cầu. Chúng có thể được đưa ra bằng nhiều cách và Góc Euler là một trong số chúng



3.2. Ma trận Quaternion

Quaternion được định nghĩa như là một số phức có ba thành phần ảo:

\boldsymbol{q} = w + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} 

Đừng mường tượng quaternion là một cái gì đó bốn chiều, đầu bạn sẽ nổ tung. Các thành phần ảo i, j, k  có thể được coi như ba vector đơn vị của trục tọa độ x, y, z. Và một quaternion có thể được viết lại như sau:

\boldsymbol{q} = w + \boldsymbol{v} 

Với \boldsymbol{v} là một vector trong không gian 3 chiều, và w là một đại lượng vô hướng chỉ độ lớn (nói chung là một con số). Dễ tưởng tượng hơn rồi đúng không!

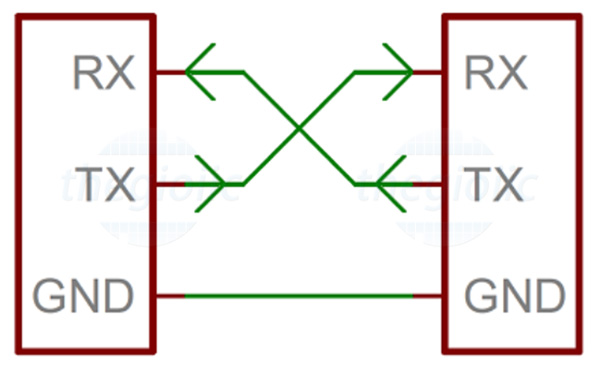
u_x^2 + u_y^2 + u_z^2 = 1 Áp dụng vào phép quay không gian của chúng ta. Giả sử bạn có một vector trục quay \boldsymbol{u}  (vector \boldsymbol{u}  phải là vector đơn vị, tức  ) và góc quay \theta (dương khi quay cùng chiều với vector, theo nguyên tắc nắm tay phải) thì chúng ta sẽ có một quaternion quay như sau:

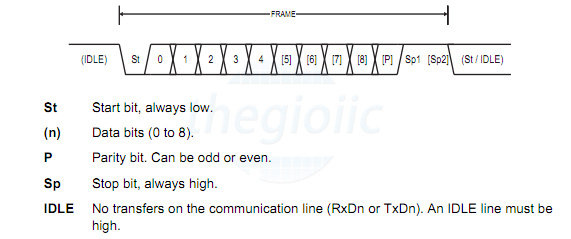
\boldsymbol{q} = \cos \frac{1}{2} \theta + \boldsymbol{u} \sin \frac{1}{2} \theta = \cos \frac{1}{2} \theta + (u_x \boldsymbol{i} + u_y \boldsymbol{j} + u_z \boldsymbol{k}) \sin \frac{1}{2} \theta 

4. Chuẩn giao tiếp UART và I2C

4.1. UART

UART (Universal Asynchronous Receiver-Transmitter) bộ truyền nhận dữ liệu nối tiếp bất đồng bộ, đây là một trong những giao thức truyền thông giữa thiết bị với thiết bị được sử dụng nhiều nhất. Giao tiếp UART được sử dụng nhiều trong các ứng dụng để giao tiếp với các module như: Wifi, Bluetooth, Xbee, module đầu đọc thẻ RFID với Raspberry Pi, Arduino hoặc vi điều khiển khác. Đây cũng là chuẩn giao tiếp thông dụng và phổ biến trong công nghiệp

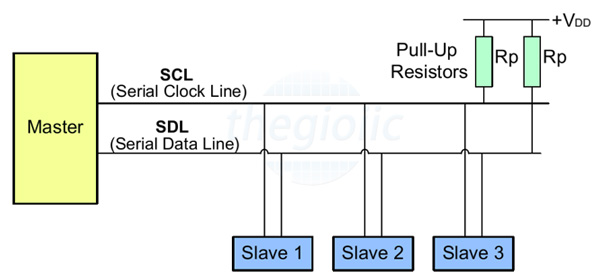




4.2. I2C

I2C hay IIC (Inter – Integrated Circuit) là 1 giao thức giao tiếp nối tiếp đồng bộ được phát triển bởi Philips Semiconductors, sử dụng để truyền nhận dữ liệu giữa các IC với nhau chỉ sử dụng hai đường truyền tín hiệu.

I2C kết hợp các tính năng tốt nhất của SPI và UART. I2C có thể kết nối nhiều slave với một master duy nhất (như SPI) và có thể có nhiều master điều khiển một hoặc nhiều slave. Điều này thực sự cần thiết khi muốn có nhiều hơn một vi điều khiển ghi dữ liệu vào một thẻ nhớ duy nhất hoặc hiển thị văn bản trên một màn hình LCD.



Giống như giao tiếp UART, I2C chỉ sử dụng hai dây để truyền dữ liệu giữa các thiết bị:

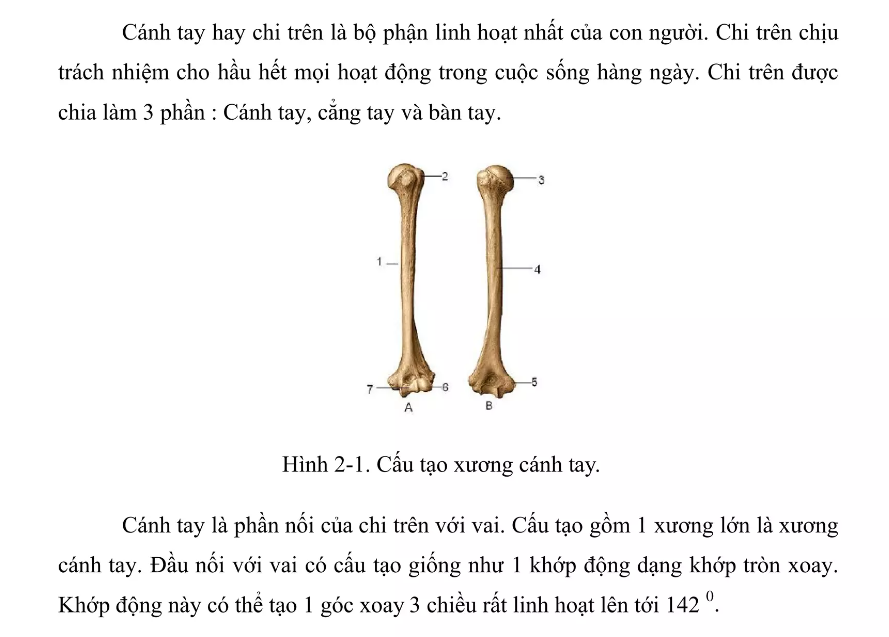
SDA (Serial Data) - đường truyền cho master và slave để gửi và nhận dữ liệu.

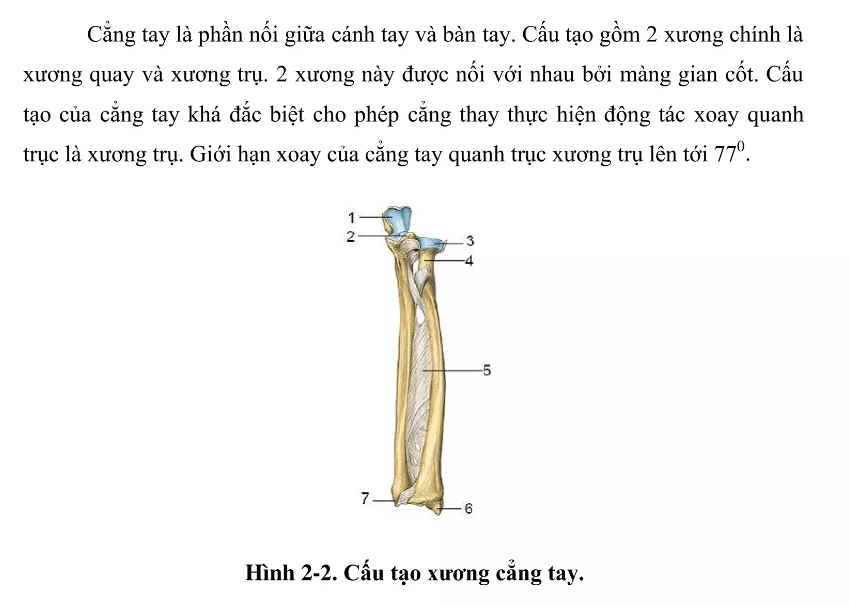
SCL (Serial Clock) - đường mang tín hiệu xung nhịp.

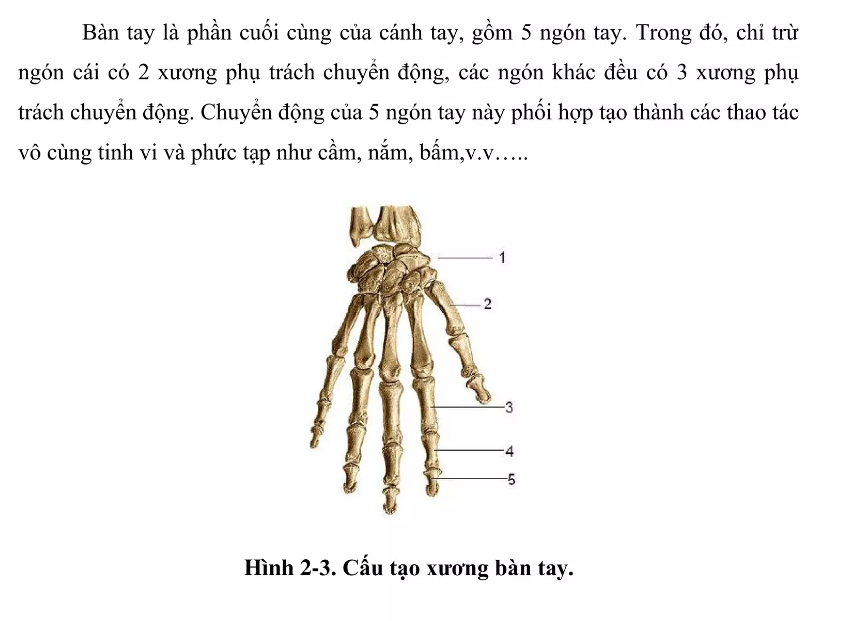
Các bit dữ liệu sẽ được truyền từng bit một dọc theo một đường duy nhất (SDA) theo các khoảng thời gian đều đặn được thiết lập bởi 1 tín hiệu đồng hồ (SCL).

5. Kiến thức cơ bản về cánh tay người

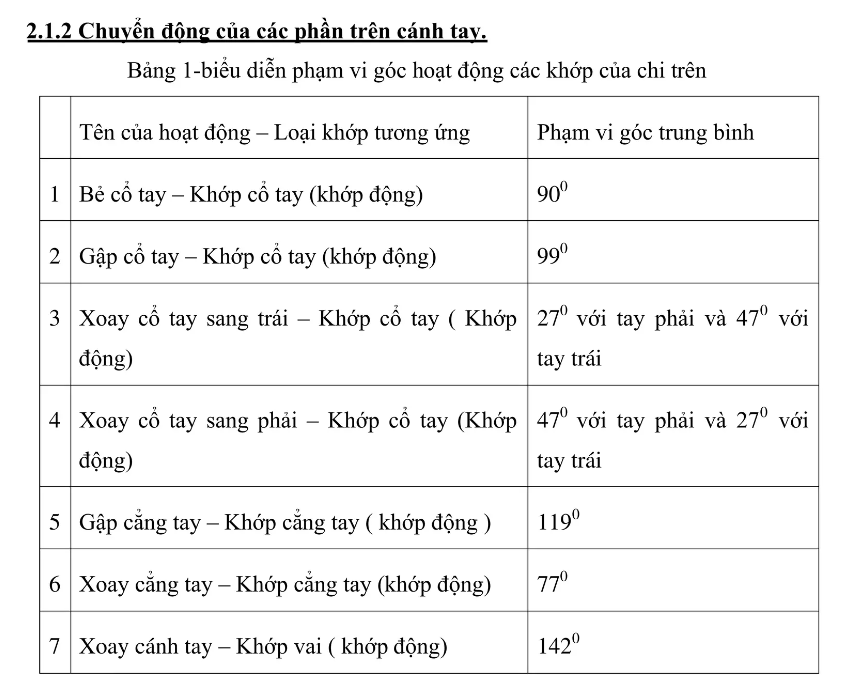
5.1 Cấu tạo cánh tay







5.2 Chuyển động của các phần trên cánh tay



Các bước thực hiện làm 7-DOF Wireless Human Arm Motion-Tracking System

**1. Mục tiêu và định hướng thiết kế**

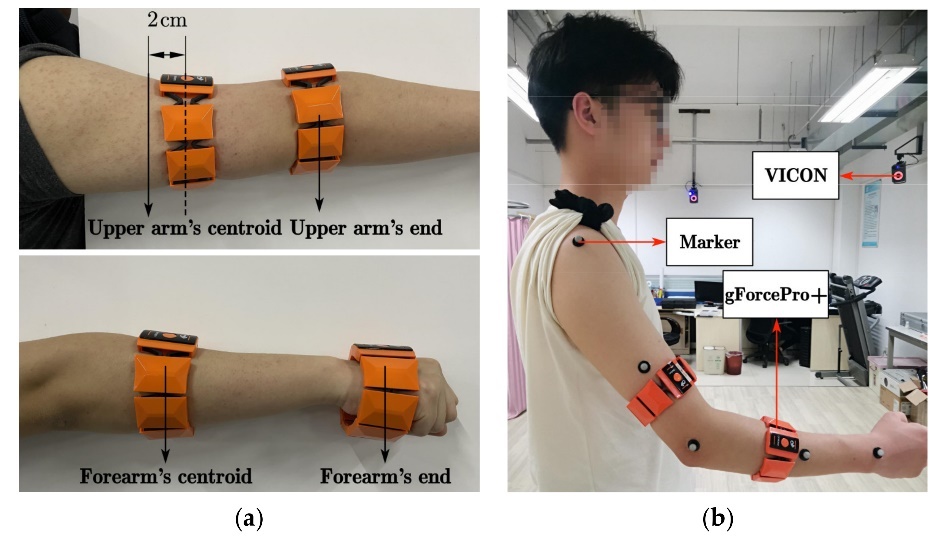
- Trên thế giới hiện nay, việc điều khiển các cánh tay robot chủ yếu dựa vào việc lập trình trên các phần mềm robot hoặc điều khiển trực tiếp trên các thiết bị cầm tay như FlexPendant của hãng ABB, E-controller của Kawasaki, SmartPAD của KUKA, ... Điểm chung của các thiết bị này là đều sử dụng các nút nhấn, màn hình cảm ứng, ... để thao tác và lập trình các quỹ đạo cho robot.

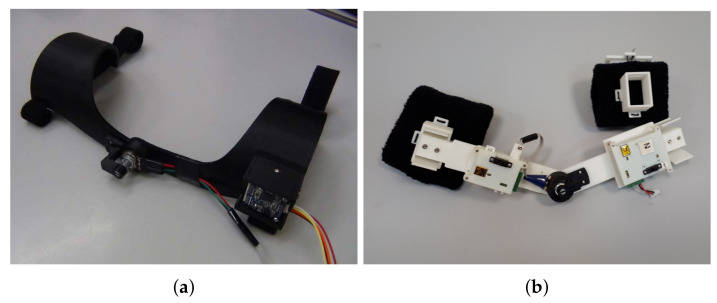


Tuy nhiên, những phương pháp này có nhược điểm là không được tự nhiên và tiện lợi khi phải thực hiện tạo từng quỹ đạo và chạy thử trong không gian thực tế.

- Một trong những hướng tiếp cận mới trên thế giới trong việc điều khiển robot hiện nay chính là sử dụng các chuyển động thu được từ cơ thể con người để thực hiện điều khiển cánh tay robot.

Đề tài IMU Motion Capture Method with Adaptive Tremor Attenuation in Teleoperation Robot System năm 2022 đã sử dụng các gForcePro+ armband để thực hiện các định vị trí của cánh tay và lực tạo ra bởi các cơ để điều khiển cánh tay robot. Tuy nhiên, các thiết bị đeo có giá rất đắt và không phù hợp nếu chỉ điều khiển các cánh tay robot nhỏ.



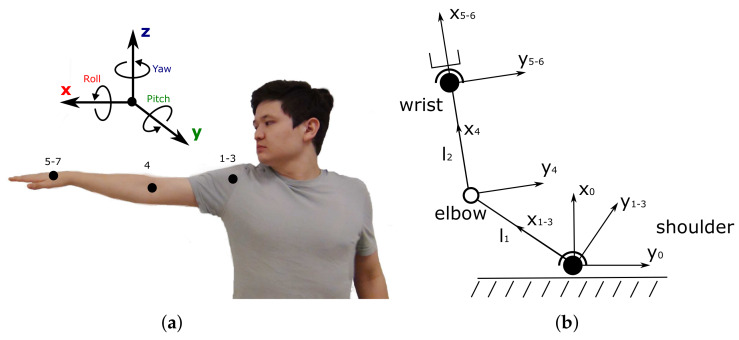
Đề tài An Open-Source 7-DOF Wireless Human Arm Motion-Tracking System for Use in Robotics Research sử dụng các cảm biến quán tính và biến trở để thực hiện tính toán. Với thiết kế và nguyên lí đơn giản, có thể giúp giảm chi phí chế tạo và điều khiển.



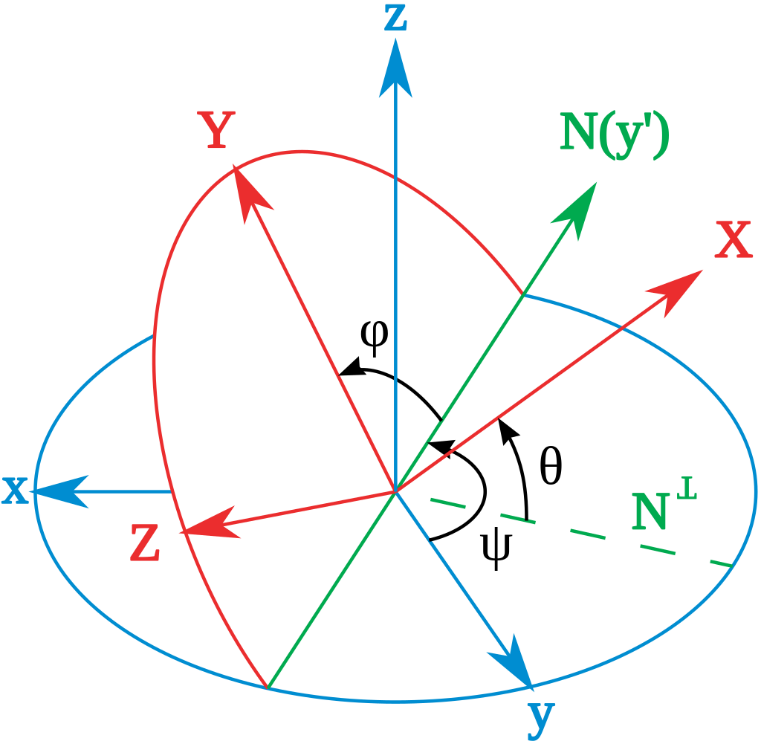
Theo đó, dựa vào nguyên lí của đề tài trên, ta sẽ thiết kế và chế tạo lại một thiết bị theo dõi chuyển động cánh tay người để thực hiện điều khiển cánh tay robot.

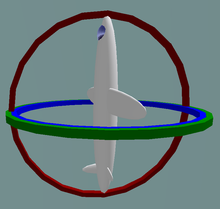
**2. Thiết kế nguyên lí và tính động học cho hệ thống**

- Kết quả cuối cùng mà hệ thống theo dõi chuyển động trả ra chính là vị trí của cánh tay trong không gian so với gốc tọa độ là vai. Ta có sơ đồ như sau:



Gỉa sử ta đã có được giá trị các góc roll, pitch, yaw của vai được đọc từ cảm biến IMU cùng với góc ở khủy tay được đọc từ biến trở,

Ta sẽ thực hiện nhân các ma trận chuyển vị để tính ra được vị trí x, y, z của đầu cánh tay.

Tuy nhiên, khi thực hiện chuyển các giá trị góc về roll, pitch, yaw, một hiện tượng gọi là Gimbal lock sẽ xảy ra làm cho các giá trị góc đọc về bị sai lệch.

Để tránh khỏi hiện tượng này, ta sẽ biểu diễn các góc xoay dưới dạng một ma trận Quaternion. Quaternion được định nghĩa như là một số phức có ba thành phần ảo:

\boldsymbol{q} = w + x\boldsymbol{i} + y\boldsymbol{j} + z\boldsymbol{k} 

Đừng mường tượng quaternion là một cái gì đó bốn chiều, đầu bạn sẽ nổ tung. Các thành phần ảo i, j, k  có thể được coi như ba vector đơn vị của trục tọa độ x, y, z. Và một quaternion có thể được viết lại như sau:

\boldsymbol{q} = w + \boldsymbol{v} 

Với \boldsymbol{v} là một vector trong không gian 3 chiều, và w là một đại lượng vô hướng chỉ độ lớn (nói chung là một con số).

u_x^2 + u_y^2 + u_z^2 = 1 Áp dụng vào phép quay không gian của chúng ta. Giả sử bạn có một vector trục quay \boldsymbol{u}  (vector \boldsymbol{u}  phải là vector đơn vị, tức  ) và góc quay \theta (dương khi quay cùng chiều với vector, theo nguyên tắc nắm tay phải) thì chúng ta sẽ có một quaternion quay như sau:

\boldsymbol{q} = \cos \frac{1}{2} \theta + \boldsymbol{u} \sin \frac{1}{2} \theta = \cos \frac{1}{2} \theta + (u_x \boldsymbol{i} + u_y \boldsymbol{j} + u_z \boldsymbol{k}) \sin \frac{1}{2} \theta 

Trong một nghiên cứu được thực hiện vào năm 2009, Sebastian Madgwick đã phát triển một thuật toán lọc nhiễu sử dụng cho IMU và AHRS để thực hiện xử lí đọc được. Thuật toán có tên là Madgwick được open-source và cho ra kết quả là một ma trận Quaternion gồm 4 giá trị q0, q1, q2, q3, q4.

Tiếp theo, ta sẽ tiếp tục chuyển đổi ma trận Quaternion về ma trận quay để tiến hành tính toán.

1. Compute the square of the quaternion norm:

n = w^2 + x^2 + y^2 + z^2

2. If the norm is zero, the quaternion is invalid.

3. Compute the matrix elements as follows:

a11 = (w^2 + x^2 - y^2 - z^2) / n

a12 = 2 \* (x\*y - w\*z) / n

a13 = 2 \* (x\*z + w\*y) / n

a21 = 2 \* (x\*y + w\*z) / n

a22 = (w^2 - x^2 + y^2 - z^2) / n

a23 = 2 \* (y\*z - w\*x) / n

a31 = 2 \* (x\*z - w\*y) / n

a32 = 2 \* (y\*z + w\*x) / n

Sau khi ta có được ma trận xoay tại khớp vai, ta tiến hành nhân thêm các ma trận tịnh tiến và xoay để ra được vị trí cuối cùng.

Tổng hợp lại, tín hiệu từ cảm biến sẽ trải qua các bước sau:



Động học của hệ thống:

result =

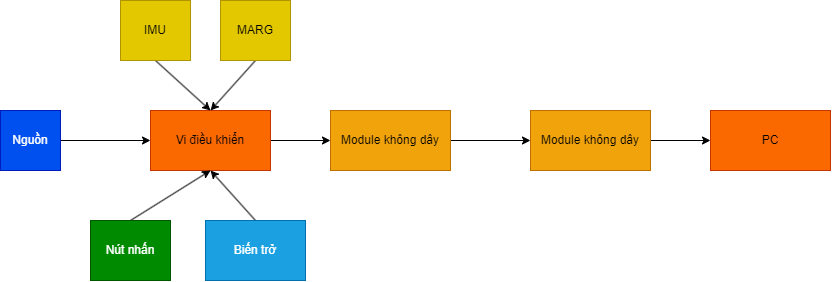
l2\*((ce\*(w^2 + x^2 - y^2 - z^2))/(w^2 + x^2 + y^2 + z^2) - (se\*(2\*w\*z - 2\*x\*y))/(w^2 + x^2 + y^2 + z^2)) + (l1\*(w^2 + x^2 - y^2 - z^2))/(w^2 + x^2 + y^2 + z^2)

l2\*((se\*(w^2 - x^2 + y^2 - z^2))/(w^2 + x^2 + y^2 + z^2) + (ce\*(2\*w\*z + 2\*x\*y))/(w^2 + x^2 + y^2 + z^2)) + (l1\*(2\*w\*z + 2\*x\*y))/(w^2 + x^2 + y^2 + z^2)

- l2\*((ce\*(2\*w\*y - 2\*x\*z))/(w^2 + x^2 + y^2 + z^2) - (se\*(2\*w\*x + 2\*y\*z))/(w^2 + x^2 + y^2 + z^2)) - (l1\*(2\*w\*y - 2\*x\*z))/(w^2 + x^2 + y^2 + z^2)

**3. Thiết kế hệ thống**

Ta có tổng thể thiết kế hệ thống như sau:



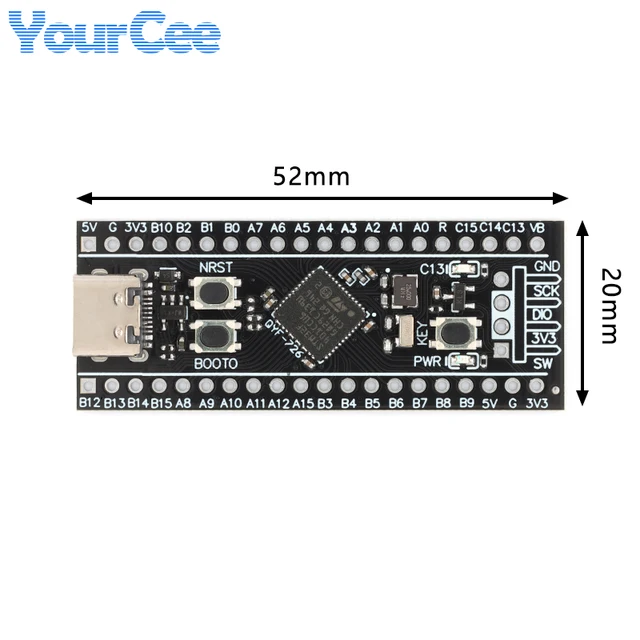
**4. Tính toán và chọn lựa linh kiện**

1. Vi điều khiển

- Hiện nay, các dòng vi xử lí STM32 với kiến trúc ARM đang rất phổ biến với tốc độ tính toán nhanh và giá thành rẻ cùng nguồn hàng phổ biến.

- Với tiêu chí nhỏ gọn, tốc độ xử lí nhanh và giá thành hợp lí, ta sẽ lựa chọn các module tích hợp sẵn có trên thị trường.

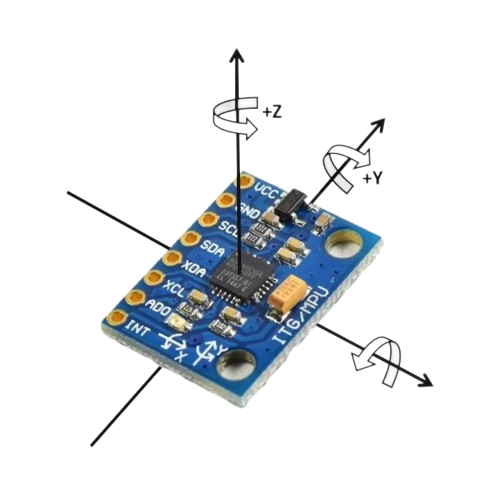
- Vi điều khiển STM32F401CCU với kiến trúc ARM Cotex-M4 với tốc độ tối đa 84 MHz và flash 256 Kb là lựa chọn phù hợp với kích thước rất nhỏ gọn cùng với tốc độ tính toán cao và giá thành rẻ.



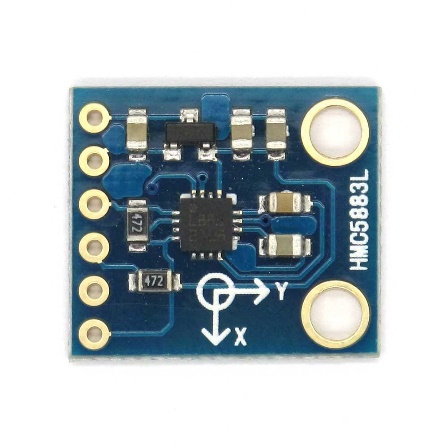
2. Cảm biến IMU và MARG

- Hiện nay có rất nhiều loại module tích hợp sẵn cảm biến IMU và MARG. Tuy nhiên, qua thực tế, các loại module này thường dễ bị làm giả làm cho tín hiệu đầu vào bị nhiễu nhiều. Một số loại module khác lại có giá thành cao và nguồn hàng hiếm. Do đó, ta sẽ lựa chọn cảm biến với tiêu chí giá thành thấp, tín hiệu ổn định và nguồn hàng phổ biến.

- Cảm biến 6 trục MPU6050 cho phép đo được 3 trục gia tốc và 3 trục gygroscope là 1 loại cảm biến rất phổ biến trên thị trường hiện nay.



- Cảm biến từ trường HMC5883L là một loại cảm biến có thể đo được giá trị từ trường Trái Đất theo 3 trục X, Y, Z



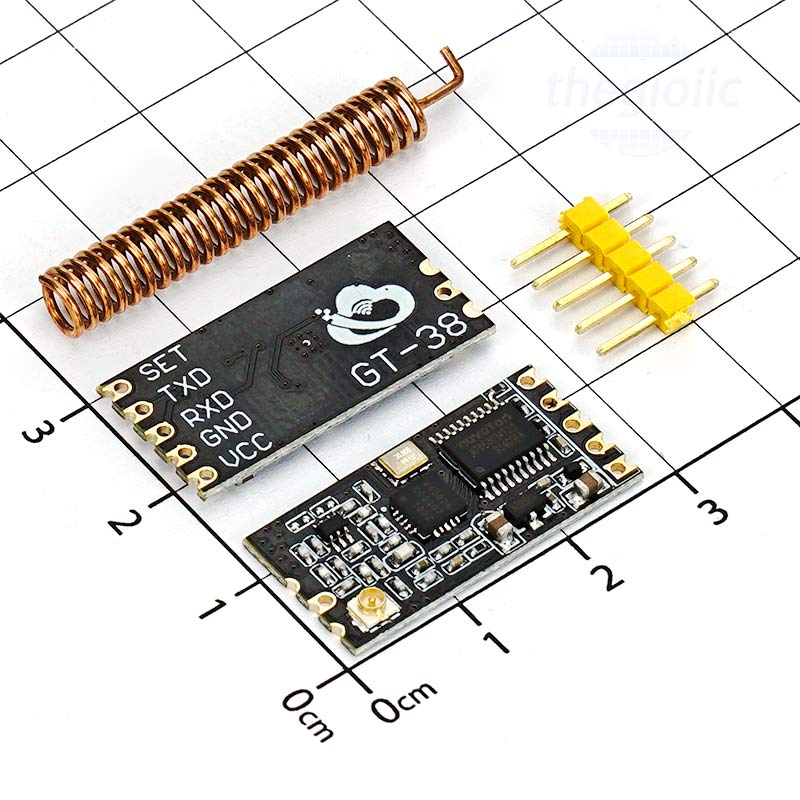
3. Biến trở

- Biến trở RV24YN 20S 10K là một loại biến trở được sản xuất tại Nhật Bản đáp ứng được tiêu chuẩn JIS. Với đường kính trục lớn có thể thuận tiện hơn cho việc thiết kế khớp nối tại khủy tay.



4. Module giao tiếp không dây

- Module UART không dây GT-38 là loại module cho phép truyền dữ liệu không dây qua giao thức UART. Với tốc độ theo datasheet có thể lên đến 100000 bps khi baudrate đạt 115200. Khoảng cách truyền có thể đạt tới 1.2 Km nếu dùng mode Ultra low distance.



5. Nguồn

Với tiêu chí nhỏ gọn và thời gian sử dụng lâu, ta sẽ sử dụng Pin Lipo 3.7V là loại pin Lithium cho khả năng sạc nhiều lần với kích thước rất nhỏ gọn.



Với dòng tiêu thụ ước tính của thiết bị là ~ 100 mA. Thiết bị có thể sử dụng được trong 20 giờ liên tục.

Để thuận tiện cho việc sạc và xả pin, ta sử dụng module mạch sạc tích hợp mạch nguồn 5V cho hệ thống



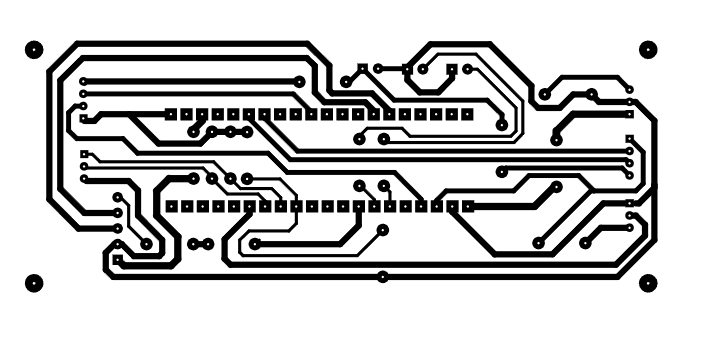
**5. Thiết kế và gia công mạch in**

1. Thiết kế schematic

2. Chọn lựa các linh kiện

3. Vẽ mạch

4. Gia công mạch in thủ công



**6. Giao tiếp với các cảm biến và xử lí tín hiệu trả về**

1. Lấy giá trị từ IMU

1.1 Giao tiếp I2C

- Trên STM32 ta thực hiện cấu hình để giao tiếp I2C, tốc độ 1000 bps.

- Địa chỉ I2C của MPU6050 mặc định là 0x68 << 1.

1.2 Init và cấu hình ban đầu

- Theo datasheet, ta có flow cấu hình cho IMU như sau:

+ Wake-up MP6050 bằng cách write 0x00 vào thanh ghi RA\_PWR\_MGMT\_1

+ Chọn Clock cho bộ PLL bằng cách tham chiếu với giá trị gygroscope của trục X

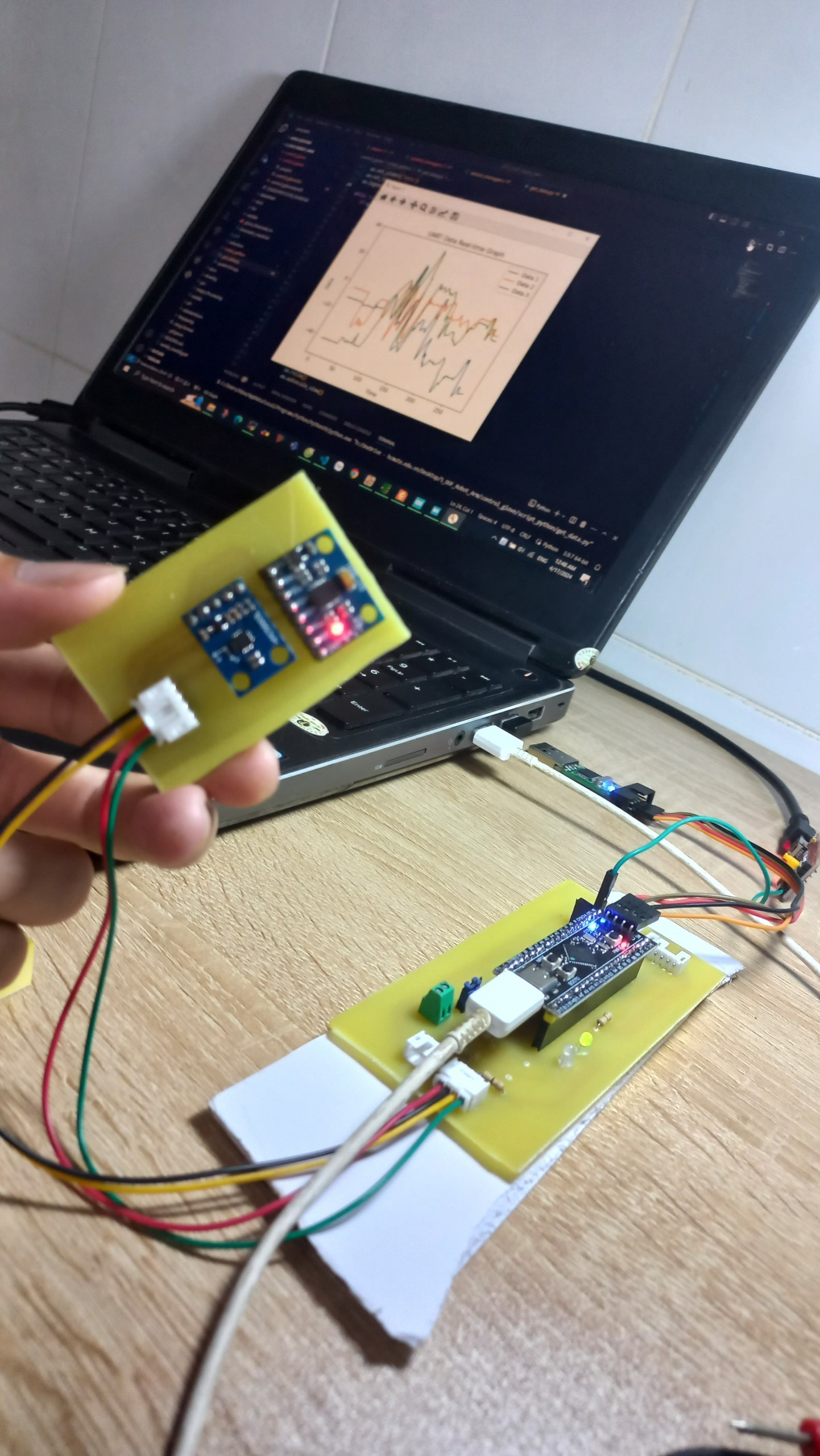
+ Set sample rate cho MPU6050

+ Cài đặt full scale range cho giá trị gygroscope là maximum 2000 độ/giây

+ Cài đặt full scale range cho giá trị gia tốc là 8g.

+ Cài đặt giá trị lọc thông thấp số Digital Low Pass Filter với acc là 260 Hz, delay 0 ms, gygro là 256 Hz, delay 0,98 ms.

1.3 Đọc giá trị gia tốc và gygro



1.4 Calib các giá trị ban đầu

- Giải thuật calib:

- So sánh giá trị trước và sau khi calib

2. Lấy giá trị từ cảm biến từ trường

1.1 Giao tiếp I2C

- Trên STM32 ta thực hiện cấu hình để giao tiếp I2C, tốc độ 1000 bps.

- Địa chỉ I2C của MPU6050 mặc định là 0x1E << 1.

1.2 Init cấu hình ban đầu

- Theo datasheet, ta có flow cấu hình như sau:

+ Chọn range cho cảm biến là RANGE\_1\_3GA

+ Chon mode đo là Continuos

+ Chọn tần số lấy mẫu là 75 Hz (maximum)

+ Chọn sample lấy là 1

1.3 Đọc giá trị từ trường

- Các giá trị từ trường của cảm biến có thể đọc được từ các thanh ghi:

REG\_OUT\_X\_M, REG\_OUT\_Y\_M, REG\_OUT\_Z\_M

1.4 Calib giá trị cảm biến từ trường

- Thuật toán calib:

3. Lấy giá trị từ biến trở

1. Đọc giá trị ADC

- Ta thực hiện cấu hình bộ ADC, chế độ 12-bits

- Ta có điện áp đầu vào có range từ 0V 🡪 3.3V, giá trị ADC từ 0 🡪 4095.

2. Lọc nhiễu sử dụng Low pass filter

- Ta sử dụng thuật toán low pass filter với hệ số Alpha là 0.03. Ta có kết quả:

**6. Tính toán giá trị vị trí cuối**

1. Áp dụng thuật toán lọc nhiễu Magdwick

- Các giá trị đầu vào: ax, ay, az, gx, gy, gz, mx, my, mz và tần số lấy mẫu

- Lưu đồ lấy data và xử lí bằng thuật toán Magdwick

- Chọn các hệ số Kp, Ki

- Kết quả khi convert về 3 góc Euler:

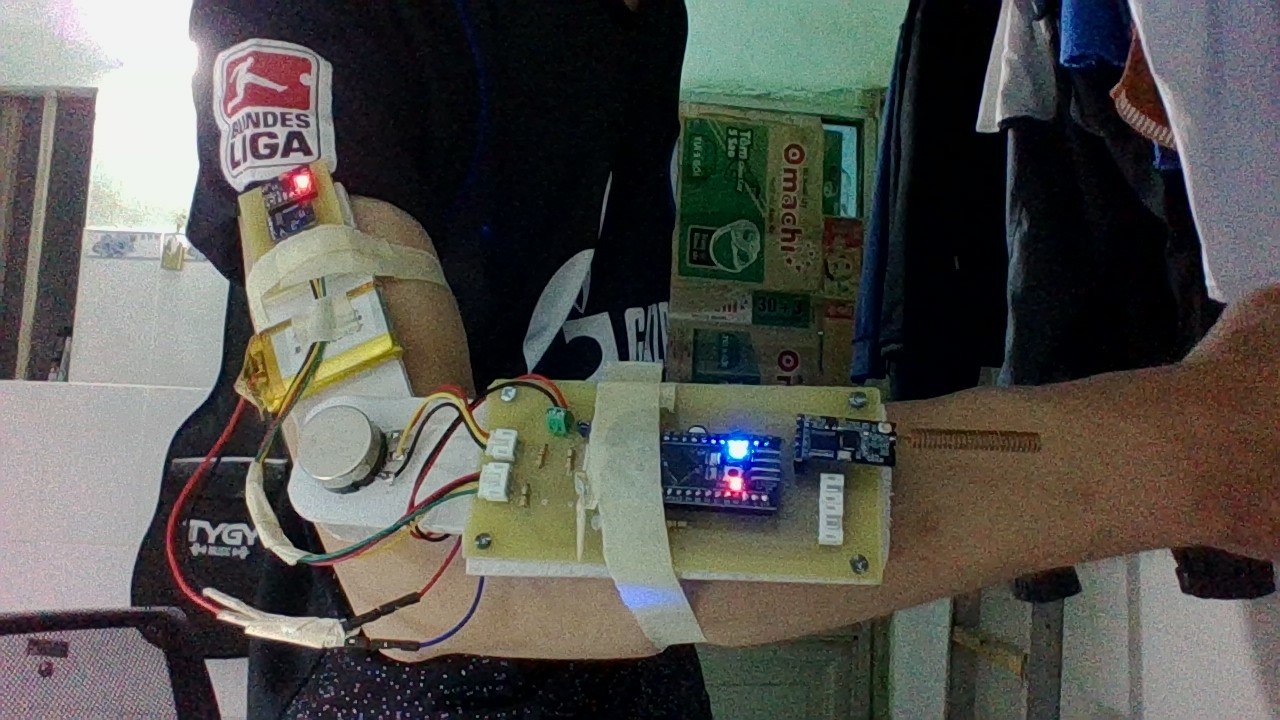
2. Chuyển đổi về giá trị vị trí cuối

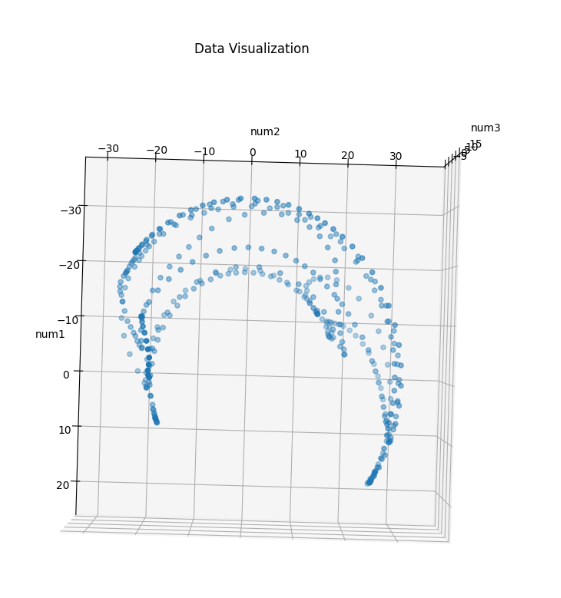
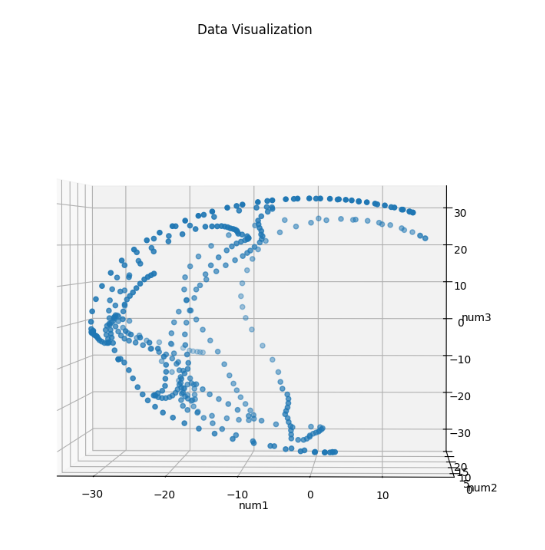
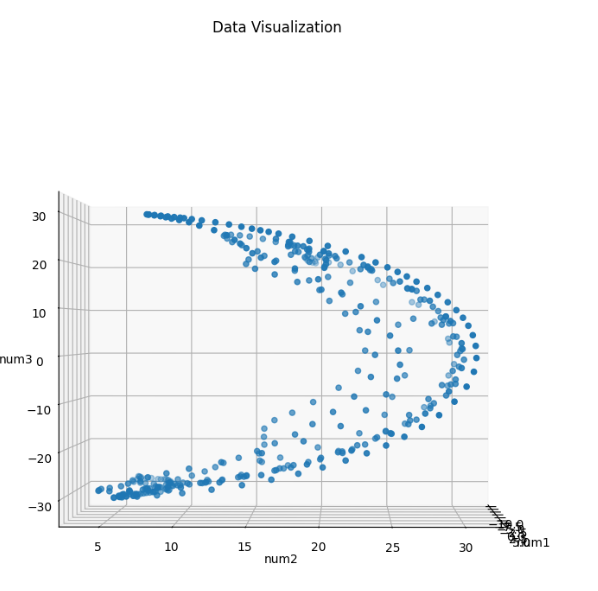
- Giá tri đầu vào: q0, q1, q2, q3 tính từ Magdwick và giá trị góc từ biến trở

2. Chế tạo mẫu thử và kiểm tra kết quả

- Nguyên lí: Đặt các cảm biến IMU, MARG ở vai và biến trở ở khủy tay

- Khoảng cách từ vai đến khủy tay chọn là 12 cm, từ khủy tay đến cuối là 20 cm



**7. Thiết kế hệ thống cơ khí**

- Thiết kế phần khung

- Thiết kế phần khớp nối

- Thiết kế phần dây đeo và tiếp xúc

- Thiết kế nắp pin và mạch sạc

- Kết quả

**8. Thiết kế giải thuật điều khiển cánh tay robot**

**- Giao thức truyền nhận**

1. Xác định frame truyền

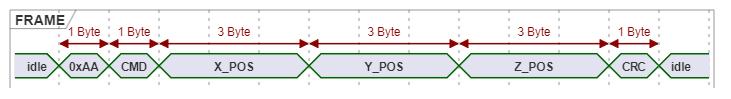
- Khi thực hiện truyền dữ liệu từ găng tay qua C#, ta cần truyền 3 thông số vị trí x, y, z là tọa độ của đầu găng tay. Các thông số vị trí có giá trị là số thực có giá trị từ -99,9999 đến 100,0000 mm (lấy 4 chữ số sau số thập phân). Vậy tổng số giá trị cần truyền cho mỗi giá trị vị trí là 2.000.000 giá trị. Suy ra cần ít nhất 3 bytes hay 24 bits để mã hóa. Ta có: 224 = 16.777.216 giá trị, lớn hơn số lượng giá trị cần thiết.

- Để thực thiện các câu lệnh từ găng tay lên C#, ta cần thêm 1 byte để mã hóa. Với 1 byte ta có thể mã hóa được 256 loại câu lệnh khác nhau.

- Trong quá trình tuyền nhận dữ liệu, có thể có xảy ra hiện tượng data bị hư hại do tác động của môi trường truyền dẫn, vì vậy cần thêm 1 byte CRC để đảm bảo tính an toàn của dữ liệu truyền đi.

- Để dễ dàng cho việc xử lí, ta sử dụng thêm 1 byte đầu tiên để báo hiệu 1 frame bắt đầu.

- Ta có kết quả sau khi cân nhắc các yếu tố:



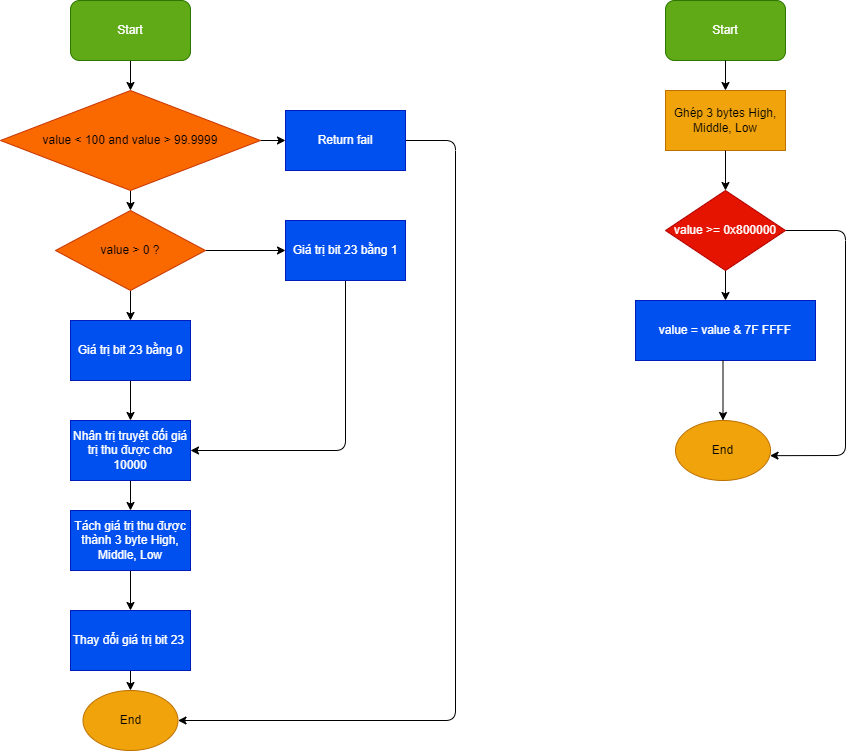
2. Quy ước câu lệnh

|  |  |
| --- | --- |
| **Command ID** | **Ý nghĩa** |
| 0x00 | Chỉ truyền giá trị vị trí |
| 0x01 | Truyền giá trị vị trí và start record |
| 0x02 | Truyền giá trị vị trí và stop record |
| 0x03 | Xóa record vừa tạo ra |
| 0x04 | Truyền giá trị vị trí và quay khớp 5 theo chiều dương |
| 0x05 | Truyền giá trị vị trí và quay khớp 5 theo chiều âm |

**3. Xử lí các giá trị vị trí**

- Các giá trị vị trí là các số thực (float) được mã hóa bởi 32 bits dữ liệu (ở dòng vi điều khiển 32 bits). Ta cần thêm 1 bit để mã hóa dấu.

Ta có: [1 bit dấu][23 bit giá trị]



Hình : Thuật toán mã hóa và giải mã giá trị vị trí