UNCUYO
UNIVERSIDAD NACIONAL DE CUYO

FACULTAD DE INGENIERÍA

Microcontroladores y

Electrónica de Potencia

Trabajo integrador:

Silla de Ruedas Electrónica

Alumno: Vignolo Alan

Legajo: 12667

Índice

Introducción

La tecnología ha avanzado rápidamente, influyendo en muchos aspectos de nuestra vida diaria, y uno de sus campos de aplicación es la asistencia médica. En este escenario, el propósito de este proyecto es diseñar una silla de ruedas comandada. Mediante la combinación de comandos y sensores, buscamos ofrecer una alternativa mejorada a quienes tienen movilidad reducida.

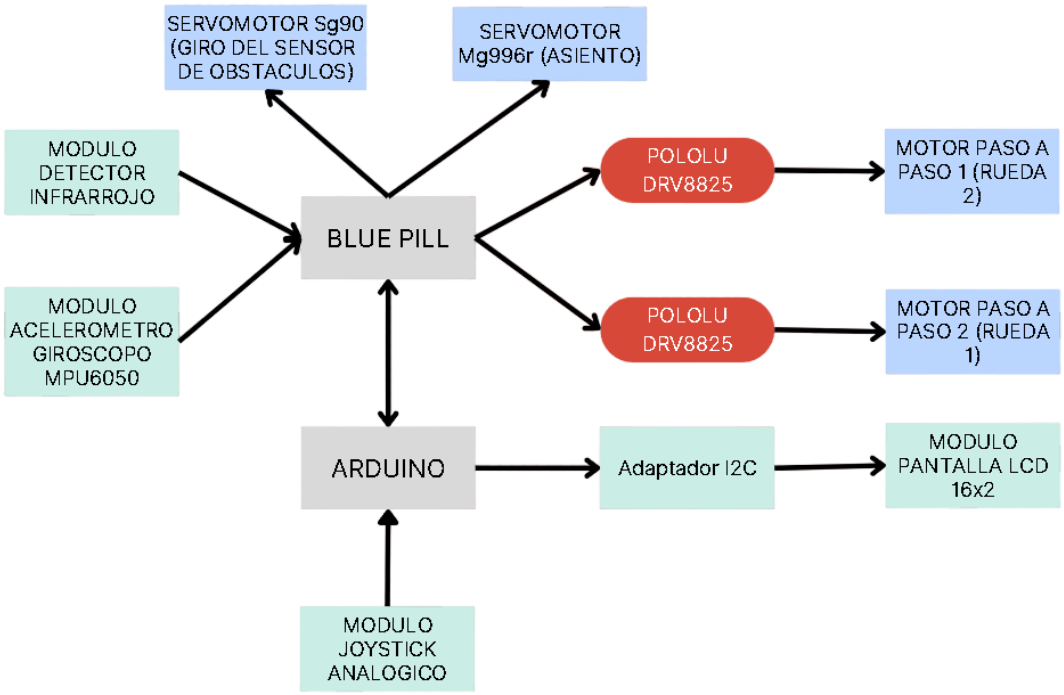
Mi interés en este proyecto surge de la observación directa de los desafíos diarios que enfrentan las personas con movilidad limitada. Las sillas de ruedas convencionales cumplen su función, pero aún presentan restricciones en su adaptabilidad a diferentes ambientes y situaciones.

La propuesta de esta silla de ruedas comandada es brindar más autonomía y seguridad. Equipada con sensores y actuadores, la silla está diseñada para desplazarse de forma comandada, esquivando obstáculos y adaptándose al terreno.

A lo largo de este informe, se describen en detalle el diseño, la construcción y las pruebas del sistema, abordando los componentes seleccionados, cómo interactúan entre sí y la estructura de programación que respalda el funcionamiento autónomo.

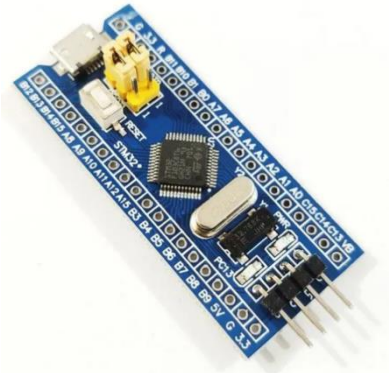
Esquema tecnológico

En el siguiente diagrama de bloques se ve la interconexión entre los diferentes módulos y los microcontroladores utilizados.



Detalle de módulos

Microcontrolador Blue Pill (STM32F103C8T6)



Características Principales:

- Arquitectura: ARM Cortex-M3 de 32 bits.
- Frecuencia de Trabajo: Hasta 72 MHz.
- Memoria: 64KB de Flash y 20KB de SRAM.
- Tensión de Trabajo: 2.0V a 3.6V.
- Periféricos: UART, SPI, I2C, GPIO, PWM, ADC, temporizadores.

Funcionamiento: El STM32F103C8T6 se programa mediante el entorno STM32 CubeIDE. Para la programación, se necesita un programador externo como el ST-Link (utilizado en este caso). Gracias a su arquitectura ARM de 32 bits, es capaz de realizar operaciones complejas, multitarea y gestionar varios periféricos simultáneamente.

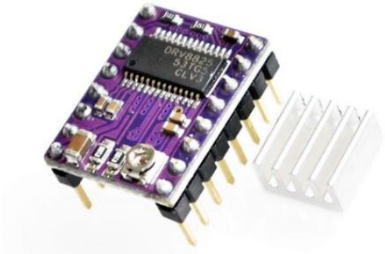
Microcontrolador Arduino Uno (ATmega328P)

Características Principales:

- Arquitectura: AVR de 8 bits.
- Frecuencia de Trabajo: 16 MHz.
- Tensión de Trabajo: 5V (recomendada).
- Memoria: 32KB de Flash, 2KB SRAM y 1KB EEPROM.
- Periféricos: 6 canales ADC, PWM, UART, SPI, I2C.

Funcionamiento: El ATmega328P se programa a través de Microchip Studio. Con una arquitectura de 8 bits, es adecuado para aplicaciones sencillas y se beneficia de una amplia comunidad que ha desarrollado múltiples bibliotecas para su uso.

Driver Pololu DRV8825 (Controlador de Motor Paso a Paso)

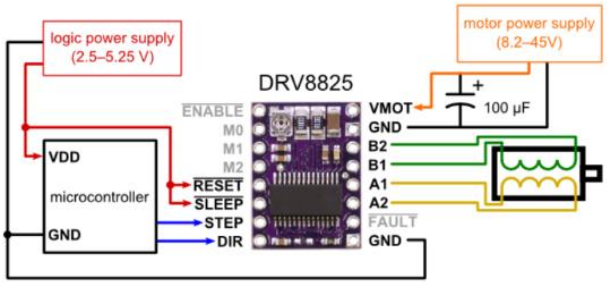


Características Principales:

- Microstepping: 1, 1/2, 1/4, 1/8, 1/16, y 1/32.
- Corriente Máxima: Hasta 2.5A por fase.
- Tensión de Alimentación: 8.2V a 45V.
- Frecuencia de Operación: Hasta 250 kHz.
- Protecciones: Sobre corriente, sobretensión y sobrecalentamiento.

Funcionamiento: El DRV8825 recibe señales de pulso y dirección desde un microcontrolador. La frecuencia del pulso determina la velocidad del motor, mientras que la señal de dirección determina el sentido de giro. El ajuste de corriente es esencial para maximizar el rendimiento y proteger el motor. Además, tiene 3 entradas para configurar el microstepping y una entrada que habilita o deshabilita el módulo.

Diagrama de conexiones:



Motor Paso a Paso Nema 17

Características Principales:

- Pasos por Revolución: 200 pasos.
- Ángulo de Paso: 1.8 grados.
- Corriente Máxima: 1.2A por fase.

- Tensión Nominal: 2.5V.

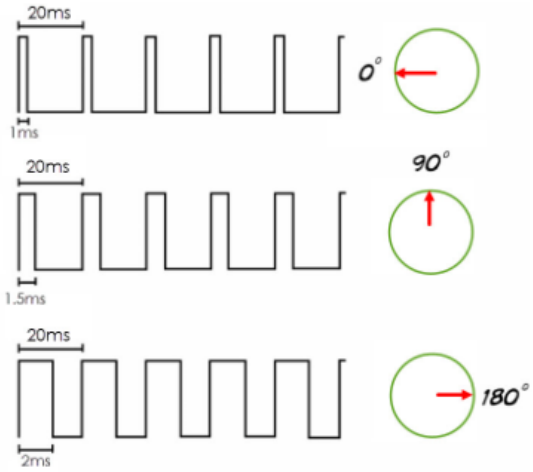
Funcionamiento: El motor Nema 17 avanza un paso gracias a la activación y desactivación secuencial de sus bobinas, que es controlada por una unidad como el DRV8825. La secuencia y la cantidad de pulsos determinan el ángulo total girado.

Servomotor SG90

Características Principales:

- Rango de Movimiento: 0 a 180 grados.
- Tensión de Operación: 4.8V a 6V.
- Frecuencia de PWM: 50Hz (un pulso cada 20ms).
- Duración de Pulso: De 1ms a 2ms.
- Torque: Aproximadamente 2.5 kg-cm (a 4.8V).

Funcionamiento: El SG90 espera una señal PWM. El ángulo del servo está determinado por la duración del pulso: un pulso de 1ms lo llevará a 0 grados y un pulso de 2ms a 180 grados.



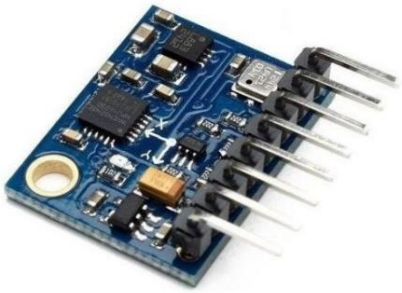
Servomotor MG996R

Características Principales:

- Par de Torsión: Aproximadamente 10 kg-cm.
- Rango de Movimiento: 0 a 180 grados.
- Tensión de Operación: 4.8V a 7.2V.
- Frecuencia de PWM: 50Hz.
- Duración de Pulso: De 1ms a 2ms.

Similar al SG90, el MG996R también utiliza señales PWM para determinar su posición. Sin embargo, cuenta con un mayor par de torsión, lo que le permite mover o soportar cargas más pesadas, haciéndolo adecuado para aplicaciones que requieran mayor fuerza.

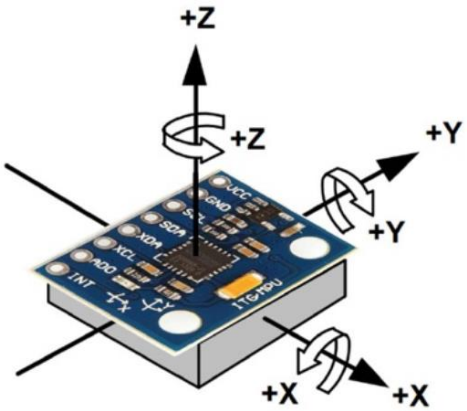
Módulo Acelerómetro Giroscopio MPU6050



Características Principales:

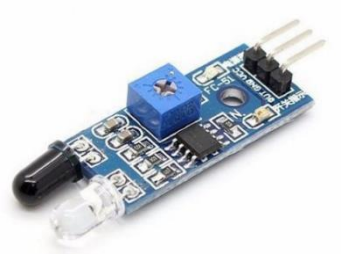
- Grados de Libertad: 6 (3 del acelerómetro y 3 del giroscopio).
- Rango de Aceleración: ±2, ±4, ±8, ±16 g.
- Rango del Giroscopio: ±250, ±500, ±1000, y ±2000 °/s.
- Tensión de Operación: 2.375V-3.46V.
- Comunicación: Protocolo I2C.
- Frecuencia de Muestreo: Hasta 8kHz para el giroscopio, 1kHz para el acelerómetro.

Funcionamiento: El MPU6050 combina un acelerómetro y un giroscopio en un solo dispositivo. El acelerómetro mide la aceleración en tres ejes, mientras que el giroscopio mide la velocidad angular. Estos datos son esenciales para determinar la orientación de un objeto en el espacio. Los ejes de referencia se ven a continuación:



Módulo Detector de Obstáculos Infrarrojo FC51

Características Principales:



- Distancia de Detección: de 2cm a 30cm ajustable por potenciómetro.
- Tensión de Funcionamiento: 3.3V a 5V.

Funcionamiento: El módulo utiliza un diodo emisor infrarrojo para enviar una señal y un receptor para detectar su reflejo. Si un objeto se encuentra dentro del rango de detección, la señal reflejada es capturada por el receptor, lo que provoca un cambio en la salida del módulo.

Módulo Joystick Analógico

Características Principales:

- Rango de Salida Analógica: 0V a 5V o 0V a 3.3V, dependiendo de la alimentación.
- Ejes: X e Y.
- Tensión de Operación: 3.3V a 5V.

Funcionamiento: El joystick cuenta con dos potenciómetros lineales, uno por eje. Al mover el joystick, se altera la resistencia de los potenciómetros, generando un cambio en la salida analógica proporcional al ángulo de inclinación, que luego se puede medir con un ADC.

Módulo Pantalla LCD 16x2 HD44780

Características Principales:

- Número de Caracteres: 32 (16 por fila).
- Controlador: HD44780.
- Modos de Comunicación: 8 bits.

Funcionamiento: Se comunica con microcontroladores a través de un bus de datos.

Módulo Adaptador I2c Pcf8574t

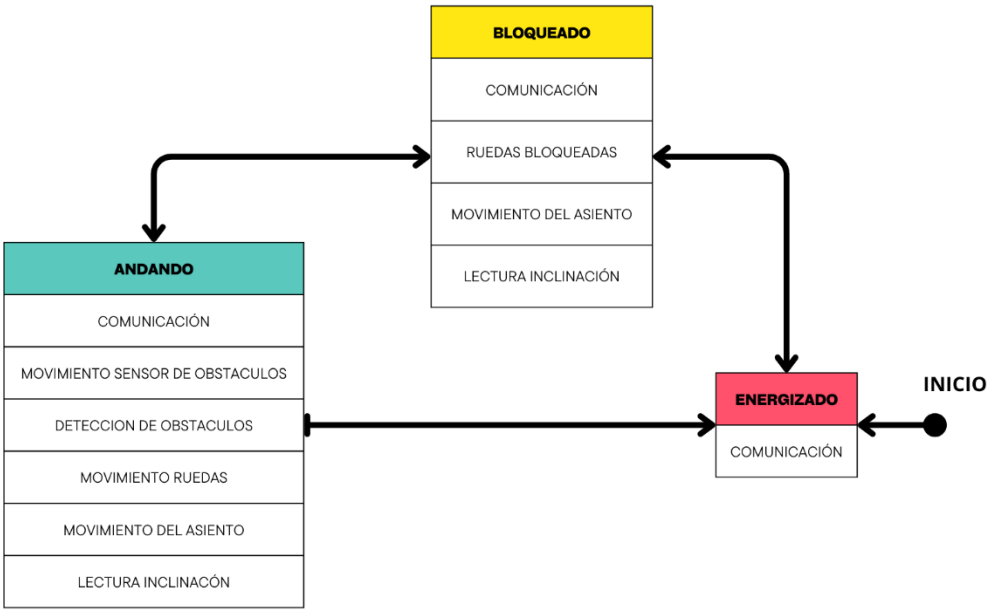
Características Principales:

- Tipo de Interfaz: I2C.
- Dirección I2C Configurable: Permite múltiples dispositivos en el mismo bus I2C.
- Pines de E/S Expandidos: Ofrece 8 pines de salid, que se pueden controlar mediante el bus I2C.
- Voltaje de Operación: Típicamente de 2.5V a 6V, compatible con la mayoría de los sistemas lógicos.

Funcionamiento: Funciona como un puente entre la interfaz I2C y componentes que no son nativamente compatibles con I2C, como pantallas LCD, botones y LEDs. Al asignar una dirección I2C única al módulo, este puede coexistir con otros dispositivos I2C en el mismo bus.

Funcionamiento general

A continuación, se puede ver un diagrama de estados de la maquina:



En cada estado de la máquina, se definen funciones específicas que pueden realizarse, y la transición entre estados es controlada por el usuario mediante dos interruptores. El primer interruptor alterna la máquina entre los estados de energizado y bloqueado, así como entre andando/bloqueado y energizado. El segundo cambia el estado de andando a bloqueado y viceversa. Esta configuración se ha diseñado para permitir apagar el equipo desde cualquier estado y asegurar que, al encenderse, la máquina comience en un estado de inactividad para prevenir incidentes.

Además, se ha establecido que la Blue Pill sea responsable de recopilar la información enviada por el Arduino, procesarla y tomar decisiones en tiempo real. A continuación, se detallan las distintas etapas del proceso.

Recopilación de Datos

MPU6050: El STM32 recopila datos del acelerómetro y giroscopio del MPU6050. Estas lecturas proporcionan información crucial sobre la orientación de la silla, permitiendo al sistema determinar si la silla se encuentra en una pendiente.

Detector de Obstáculos: El sensor de obstáculos infrarrojo proporciona entradas al STM32 sobre cualquier objeto que esté cerca de la silla. Esto es fundamental para evitar colisiones y asegurar la seguridad del usuario. Este valor es fácilmente regulable mediante su potenciómetro y esta diseñado para que la persona que lo vaya a usar seleccione el valor de la distancia a detectar.

Joystick Analógico: A través del Arduino, el STM32 recibe valores X-Y del joystick, que fluctúan entre 0 y 1023. Estas entradas representan la dirección y magnitud del movimiento deseado por el usuario.

Procesamiento

Interpretación de Entradas del Joystick: El STM32 procesa las coordenadas X-Y del joystick . Estas coordenadas se traducen en una dirección y velocidad deseadas por el usuario, permitiendo una interpolación a valores entre 0 y 1000 de velocidad, que luego se traducirán a valores de frecuencia.

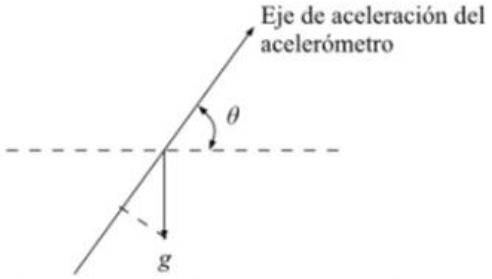
Rampa de Velocidad: Para evitar cambios abruptos en la velocidad y dirección, se implementa una rampa de velocidad. Esto significa que, en lugar de alcanzar la velocidad deseada inmediatamente, la silla aumenta o disminuye su velocidad de forma gradual. Esta técnica protege los motores y mejora la experiencia del usuario, evitando sacudidas. La rampa elegida para este caso se basa en una función lineal de distintas pendientes.

Estimación del Ángulo de Inclinación: Las lecturas del MPU6050, compuestas por datos del acelerómetro y giroscopio, se combinan usando un filtro complementario.
La técnica del ángulo complementario se aplica para obtener una estimación precisa de la orientación de la silla mediante la combinación óptima de los datos del acelerómetro y el giroscopio del MPU6050. Esta técnica emplea la fórmula:

$$\text{Ángulo} = \alpha \cdot (\text{Ángulo anterior} + \text{Velocidad angular} \cdot \text{delta tiempo}) + (1-\alpha) \cdot \text{Ángulo acelerómetro}$$

donde:

- α es el factor de filtro que determina la contribución relativa de los sensores.
- Ángulo anterior es el ángulo calculado en la iteración anterior.
- Velocidad angular es la medida del giroscopio.
- Delta tiempo es el tiempo transcurrido desde la última medición.
- Ángulo acelerómetro es el ángulo aproximado por el acelerómetro, el cual se calcula utilizando las mediciones de aceleración en los ejes X, Y y Z. Para un acelerómetro en reposo o en movimiento constante (donde la única aceleración percibida es la gravedad), el ángulo de inclinación respecto a la gravedad se puede calcular usando trigonometría básica.



$$\theta_x = \tan^{-1}\left(\frac{a_x}{\sqrt[2]{a_Y{}^2 + a_z{}^2}}\right)$$

$$\theta_y = \tan^{-1}\left(\frac{a_y}{\sqrt[2]{a_x{}^2 + a_z{}^2}}\right)$$

El valor de α varía entre 0 y 1 y es ajustado según la confianza que se tenga en las mediciones del giroscopio frente al acelerómetro. Para este caso se eligió un valor de 0,8 a partir de probar diferentes valores y comprobar que este es el de mejores resultados.

Detección y Reacción a Obstáculos

Ángulo de Detección: El sensor de obstáculos está configurado para hacer una detección de obstáculos en la dirección del movimiento (ayudado por un servomotor), brindando una distancia apropiada para generar el frenado de las ruedas.

Decisiones basadas en Obstáculos: Cuando se detecta un obstáculo, se toma una decisión basada en su proximidad y dirección. El sistema bloquea el avance y solo permite el retroceso de la silla y del giro de forma distinguida. Ahora el giro esta provocado por el movimiento de una sola rueda, pivotando sobre la otra.

Actuación

Control de Motores Paso a Paso: El STM32 envía señales PWM a los controladores Pololu. Estas señales determinan la velocidad de accionamiento o de secuencia del motor, permitiendo el movimiento preciso de la silla en respuesta a las entradas del usuario.

Ajuste dirección: dependiendo el valor de Y introducido por el usuario, se ajusta la dirección de accionamiento de los pololu, permitiendo así el movimiento en ambos sentidos.

Servomotor y Estabilidad: Si el ángulo de inclinación de la silla se desvía más allá de un rango seguro, el STM32 envía comandos al servomotor para ajustar la posición de la silla. Esto no solo mantiene la silla nivelada, sino que también garantiza la seguridad y comodidad del usuario.

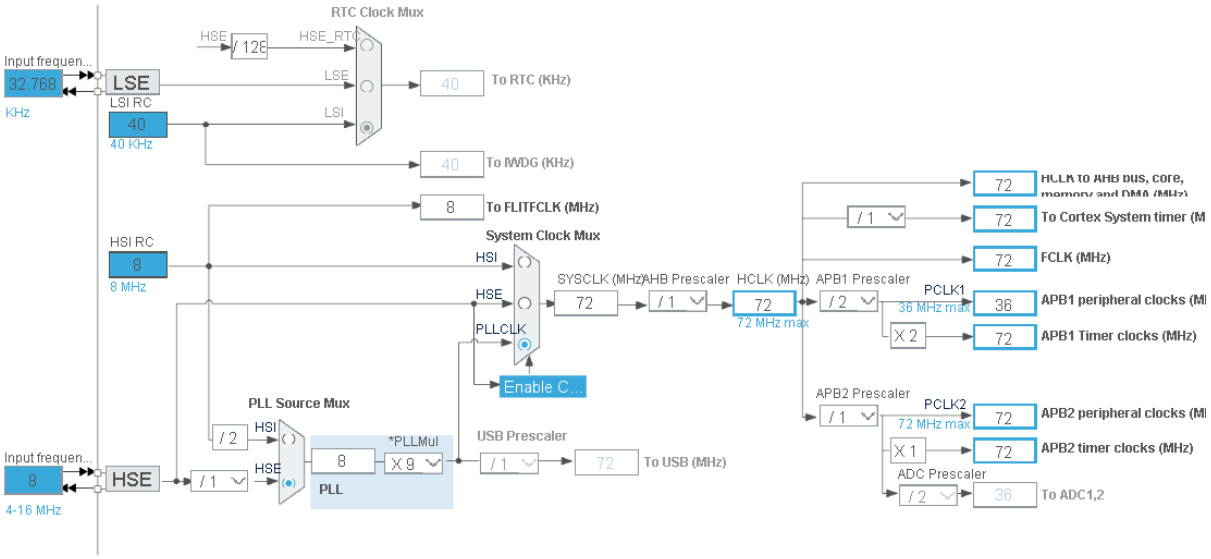
Control de Servomotor para Obstáculos: como se mencionó previamente, el sensor de obstáculos debe apuntar en todo momento en la dirección del movimiento, para esto se utiliza un servomotor controlador por la blue pill que mantendrá dicho sensor alineado todo el tiempo.

Comunicación

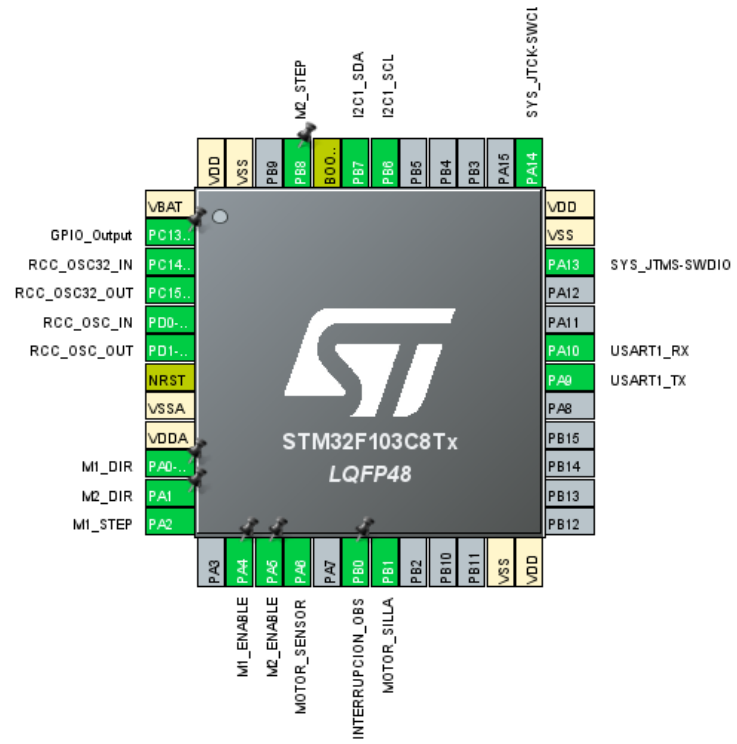
Feedback al Usuario: El STM32, a través de la comunicación UART con el Arduino, informa regularmente sobre el estado de la silla, permitiendo que el Arduino muestre esta información en una pantalla.

Programación STM32

Como primera instancia se configura el clock del microcontrolador en 72MHz:



A continuación, se muestra la configuración de los pines de la blue pill. El STM32 cube ID brinda una forma gráfica de hacer una configuración inicial de los pines y periféricos.



Prioridades en Interrupciones

- Prioridad 0: la interrupción del sensor de obstáculo tiene la prioridad máxima ya que esta debe impedir que la silla se estrelle contra algún objeto que se encuentre en frente. Vital para la seguridad.

- Prioridad 1: aquí es encuentra la comunicación por serie con el Arduino y la recepción de datos junto con la interrupción por desborde del timer 1, encargada de la rampa de velocidad en las ruedas y del manejo de los estados de la máquina de estados.

Configuración de periféricos

UART

Mode: Asynchronous

Basic Parameters	
Baud Rate	9600 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1
Advanced Parameters	
Data Direction	Receive and Transmit
Over Sampling	16 Samples

I2C

Master Features	
I2C Speed Mode	Standard Mode
I2C Clock Speed...	100000
Slave Features	
Clock No Stretch...	Disabled
Primary Address ...	7-bit
Dual Address Ac...	Disabled
Primary slave ad...	0
General Call addr...	Disabled

TIMER 1

Clock Source: Internal Clock

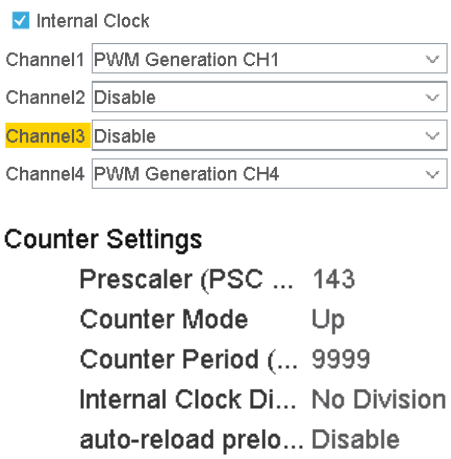
Counter Settings	
Prescaler (P...	7199
Counter Mode	Up
Counter Peri...	39
Internal Cloc...	No Division
Repetition C...	0
auto-reload p...	Disable

TIMER 2

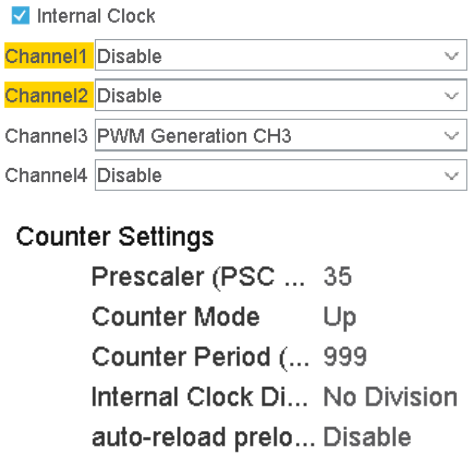
Clock Source: Internal Clock
Channel1: Disable
Channel2: Disable
Channel3: PWM Generation CH3
Channel4: Disable

Counter Settings	
Prescaler (PSC ...	35
Counter Mode	Up
Counter Period (...	999
Internal Clock Di...	No Division
auto-reload prelo...	Disable

TIMER 3



TIMER 4



Bibliotecas utilizadas

- main.h: es un archivo de encabezado que incluye definiciones, prototipos y configuraciones específicas del proyecto principal. En plataformas como STM32Cube, main.h suele incluir las configuraciones de los pines y las inicializaciones de periféricos.
- stdio.h: es la biblioteca estándar de entrada y salida en C. Provee funciones para realizar operaciones de entrada y salida.
- string.h: contiene definiciones de funciones para manipular cadenas y bloques de memoria, como strcpy(), strlen().
- math.h: es la biblioteca matemática estándar de C que provee funciones para operaciones matemáticas complejas, como pow(), sqrt().
- stdlib.h: es la biblioteca estándar de C para operaciones generales como gestión de memoria dinámica, conversión de datos, generación de números aleatorios, etc. Incluye funciones como atoi() y rand().
- stdbool.h: introduce un tipo booleano (bool) y los valores true y false en C, que originalmente no cuenta con un tipo de dato booleano nativo.
- ctype.h: provee funciones para clasificar y transformar caracteres individuales, como isdigit(), isalpha().
-

Funciones Codificadas

Gestión del MPU6050

- MPU6050_Init: Inicializa el sensor MPU6050. Establece los parámetros iniciales para que el dispositivo comience a funcionar.
- MPU6050_Reset: Envía una señal al MPU6050 para que se reinicie, útil en caso de comportamientos inesperados o errores de lectura.
- MPU6050_Read_Data: Solicita una serie de 14 bytes desde el MPU6050 que contiene las lecturas actuales del acelerómetro, giroscopio y temperatura.

- MPU6050_Get_ax: Calcula y devuelve la aceleración en el eje X utilizando los bytes relevantes obtenidos de MPU6050_Read_Data. El valor es convertido a una medida en g usando una constante de conversión.
- MPU6050_Get_ay: Similar a MPU6050_Get_ax, pero para el eje Y.
- MPU6050_Get_az: Similar a MPU6050_Get_ax, pero para el eje Z.
- MPU6050_Get_Gx: Devuelve la velocidad angular (rate) del giroscopio en el eje X. Nuevamente, utiliza una constante de conversión para obtener el valor en grados por segundo.
- MPU6050_Get_Gy: Similar a MPU6050_Get_Gx, pero para el eje Y.
- MPU6050_Get_Gz: Similar a MPU6050_Get_Gx, pero para el eje Z.
- AccionarMPU**:** Esta función inicia verificando el estado del MPU6050; en caso de errores, procede con un reinicio y reinicialización del sensor. Posteriormente, calcula el ángulo de inclinación y la velocidad angular usando los datos de aceleración y giroscopio del MPU6050. Aplica un filtro complementario para obtener una estimación precisa del ángulo, y si detecta una variación significativa en comparación con la medición anterior, activa el motor de la silla para ajustar su inclinación.

Acciones de Motor

- Accionar_Motor_Silla: Esta función controla el motor de la silla basándose en el ángulo de inclinación recibido. El ángulo se traduce en un valor PWM que determina la posición de la silla. La lógica detrás de esto es convertir un rango de ángulos en un rango de valores duty cycle, permitiendo que la silla se incline hacia adelante o hacia atrás en función del ángulo detectado por el MPU6050.
- Accionar_Motor_Sensor: Esta función ajusta el motor del sensor en función de una entrada 'x'. Traduce el valor 'x' en un valor PWM, que luego se utiliza para ajustar el motor que controla un sensor (posiblemente un sensor de proximidad o similar). Dependiendo de 'x', el sensor puede girar o inclinarse en una dirección específica.
- Rampa_Motor: Una función esencial que garantiza que los motores no cambien su velocidad de manera abrupta, evitando picos de corriente y reduciendo el desgaste del motor. La función evalúa si el motor necesita cambiar de dirección y, de ser así, reduce su velocidad a cero antes de cambiar. Luego, ajusta la velocidad en pequeños pasos (o "rampas") hasta llegar a la velocidad objetivo. Esta función también ajusta el registro de control de temporizador adecuado para generar la señal PWM correcta para el motor.
- Calcularpaso: Está vinculada a la lógica de rampa. Dependiendo de la diferencia entre la velocidad actual y la velocidad objetivo, esta función determina cuánto debería aumentar o disminuir la velocidad en cada ciclo. Si el motor necesita acelerar o desacelerar rápidamente, el "paso" será más grande. Si el cambio es más gradual, el paso es menor.
- Valor_ARR: Basada en un valor de entrada, esta función calcula y devuelve un valor para el registro ARR del temporizador. El valor del registro ARR determina la frecuencia de la señal PWM, que a su vez afecta la velocidad del motor.
- calcularVelocidadMotores: Una función central que traduce la posición actual del joystick en comandos de velocidad y dirección para los motores. Tiene en cuenta varios factores, como la detección de obstáculos y la posición central del joystick, para determinar cómo deberían

moverse los motores. Por ejemplo, si se detecta un obstáculo y el joystick indica un movimiento hacia adelante, los motores no se moverán.

Interpretación y Gestión de Comandos UART

- Interpretar_Comando: Una vez que se recibe un comando a través de UART, esta función desempeña un papel crucial decodificándolo. Los comandos pueden venir en diferentes formatos, y esta función los interpreta adecuadamente para obtener los valores requeridos. Por ejemplo, un comando "XY" se dividiría para extraer los valores X e Y. Los comandos recibidos pueden influir en diversas acciones, como modificar la dirección del motor, ajustar la velocidad y cambiar la configuración del sistema.
- SendCurrentState: Una función auxiliar que toma una cadena de caracteres como entrada y la transmite a través de la interfaz UART. Esta se utiliza para enviar el estado actual de la silla al Arduino.

Programación ATMEGA328P

Configuración Inicial

Frecuencia de operación: El microcontrolador opera a una frecuencia de 16MHz. Esta configuración está dada por la definición #define F_CPU 16000000.

Configuración de Pines

Botones: Se han definido los pines PD2 y PD3 como entradas para los botones de cambio de estado. También se han habilitado resistencias pull-up internas para estos pines.

LCD: para facilitar su uso se incorpora un conversor I2C conectado al Arduino.

Configuración de Periféricos

UART
Baud Rate: Configurado para 9600 (dado por la definición #define brate 9600).
Bit de Parada: 1 bit.
Bits de Datos: 8 bits.
Habilitación de Interrupción por Recepción: Está habilitada para gestionar comandos entrantes.

ADC
Referencia de Voltaje: AVcc con capacitor externo en el pin AREF.
Prescaler: 128 (dado por ADCSRA).
Canales: Canales 0 y 1 utilizados para la lectura

Timer1
Configurado en modo CTC (Clear Timer on Compare Match). Se utiliza principalmente para generar un contador que incrementa cada milisegundo.
Prescaler: 64
Valor de comparación: 250

I2C
Tasa de Reloj: 100 kHz
Preescaler: Configurado en 1
Dirección I2C del LCD: 0x27
Pines de Comunicación: SCL (Pin 5), SDA (Pin 4)

Bibliotecas Utilizadas

avr/io.h: Esencial para cualquier programa AVR, ofrece capacidades de E/S específicas para AVR.

util/delay.h: Proporciona funciones de retraso.

stdlib.h: Contiene funciones generales y operaciones de conversión, como atoi().

stdio.h: Utilizado para operaciones de entrada/salida estándar.

Funciones Codificadas

Gestión de Botones

- configure_buttons: Esta función se encarga de configurar los pines PD2 y PD3 como entradas y habilitar las resistencias pull-up internas. Adicionalmente, configura las interrupciones externas INT0 e INT1 para que detecten flancos de bajada (falling edges), que corresponderían a una pulsación de botón.
- ISR(INT0_vec: Esta es una Rutina de Servicio de Interrupción (ISR) que se activa cuando se detecta una pulsación en el botón conectado al pin PD2 (INT0). Implementa un debounce básico y, en función de las condiciones actuales y anteriores, envía comandos o modifica el estado de la silla.
- ISR(INT1_vect): Similar a la ISR anterior, pero para el botón conectado al pin PD3 (INT1).

Comunicación UART

- mi_UART_Init: Función encargada de inicializar la comunicación UART con la tasa de baudios especificada. Configura registros relevantes y activa las interrupciones de recepción.
- mi_putc: Envía un único carácter c a través de la interfaz UART.
- mi_getc: Escucha y recibe un carácter desde la interfaz UART.
- mi_puts: Toma una cadena de caracteres str y la envía carácter por carácter a través de la interfaz UART.

Gestión ADC

- ADC_init: Función que inicializa el Convertidor Analógico a Digital (ADC) del microcontrolador. Configura la referencia de voltaje y el factor de división del ADC.
- ADC_GetData: Lee un valor analógico desde un canal específico del ADC. La función configura el canal de entrada, activa el ADC, inicia una conversión y luego devuelve el valor convertido.

Gestión I2C

- i2c_init: Esta función inicializa el bus I2C configurando la tasa de reloj a 100kHz y el preescaler a 1, preparando el sistema para la comunicación I2C.
- i2c_start: Genera una condición de inicio en el bus I2C, señalando el comienzo de una transmisión de datos.
- i2c_stop: Envía una señal de parada para terminar la transmisión de datos y liberar el bus I2C.
- i2c_write: Envía un byte de datos a través del bus I2C. Esta función es fundamental para la comunicación con dispositivos conectados al bus.
- i2c_read_ack: Lee un byte de datos del bus I2C y envía un bit de acuse de recibo (ACK), indicando que está listo para recibir más datos.
- i2c_read_nack: Similar a i2c_read_ack, pero envía un bit de no acuse de recibo (NACK), señalando el final de la lectura.
- lcd_send_nibble: Envía un nibble (4 bits) al LCD a través del adaptador I2C.
- i2c_send_byte: Función utilizada para enviar un byte completo al LCD. Se utiliza para controlar las operaciones del LCD, como la configuración del cursor o la visualización de caracteres.

Gestión LCD

- lcd_init: Inicializar la pantalla LCD
- lcd_write_command: Escribir comandos o datos en el LCD
- lcd_set_cursor: Configurar la posición del cursor en el LCD
- lcd_write_string: Escribir cadenas de caracteres en el LCD
- lcd_display_estado: Mostrar el estado actual en el LCD

Interpretación de Comandos

- interpretarcomando: Esta función se encarga de interpretar un comando recibido y modificar el estado de la silla de ruedas en consecuencia.
- ISR(USART_RX_vect): Rutina de servicio de interrupción que se activa cuando se recibe un dato por UART

Etapas de montaje y ensayos realizados

El proceso de montaje y ensayo de la silla de ruedas basada en STM32 fue sistemático y meticuloso para asegurar su correcto funcionamiento y garantizar la seguridad del usuario. A continuación, detallamos las etapas seguidas:

Etapas de Montaje

a. Configuración inicial del STM32: Antes de cualquier conexión, se aseguró que el STM32 estuviera programado con el bootloader correcto y que se pudiera comunicar con el entorno de desarrollo.
b. Conexión de Sensores y Actuadores: Se conectaron de forma secuencial y lógica el MPU6050, el sensor de obstáculos y los motores. También se conectó el joystick y el servomotor para el sensor de obstáculos.
c. Instalación de Circuitos de Potencia: Se montaron los controladores de motor y los circuitos de potencia, asegurándose de que cada uno tuviera la protección adecuada contra sobrecorrientes o cortocircuitos.
d. Integración con el Joystick y la Interfaz de Usuario: Se conectó el joystick al sistema, junto con cualquier otra interfaz que pudiera ser necesaria para el usuario, como una pantalla o indicadores LED.

Ensayos Realizados

Las pruebas y los ensayos realizados se pueden visualizar en el apartado de PRUEBAS de la sección de ANEXOS.

Pruebas Iniciales

a. Comunicación UART: Se verificó la correcta transmisión y recepción de datos entre el STM32 y otros dispositivos periféricos. Se enviaron y recibieron cadenas de prueba para asegurarse de que no hubiera pérdida de datos o corrupción.
b. Energización segura o reinicio: Se realizó un encendido y reinicio de la silla varias veces para confirmar que, en ningún escenario, la silla se moviera o actuara de manera inesperada.

Valores máximos de inclinación

a. Asiento: Se inclinó el asiento a su máxima capacidad en todas las direcciones, observando el comportamiento del sistema y asegurando que no superara los límites seguros.
b. Servo del Sensor de Obstáculos: Se movió el servomotor a sus extremos para determinar el rango máximo de detección del sensor.

Movimientos con Velocidad Variable

a. Hacia Adelante: Se desplazó la silla a diferentes velocidades hacia adelante, desde una marcha lenta hasta su máxima velocidad, observando la respuesta del sistema.
b. Hacia Atrás: Se siguió un protocolo similar al movimiento hacia adelante, pero en dirección inversa.

c. Giro: La silla fue puesta en modo de giro, haciendo que rotara sobre su eje central en ambas direcciones. Se evaluó la suavidad, precisión y velocidad de estos giros.

Pruebas de Frenado

a. Frenado Automático: Se comprobaría la eficiencia del sistema para detenerse automáticamente al detectar un obstáculo.
b. Frenado Manual: Se verificaría la respuesta de la silla al frenado manual a través de la interfaz del usuario.

Resultados, especificaciones finales

Grado de Cumplimiento de Metas y Objetivos

- Control Dinámico y Reactivo: La silla de ruedas, gracias a la programación y los componentes, he logrado un control reactivo y dinámico, permitiendo movimientos precisos en tiempo real según las entradas del usuario y las condiciones del entorno.
- Estabilidad y Alineación: A través del uso del MPU6050, se logró una estimación precisa del ángulo de inclinación. Esto ha permitido que la silla ajuste su posición para garantizar la seguridad del usuario.
- Detección de Obstáculos: El sensor infrarrojo ha sido efectivo en la detección de obstáculos en su camino, deteniendo la silla o reorientándola cuando se encuentra con un obstáculo.
- Comunicación y Feedback: La comunicación UART entre el STM32 y Arduino ha sido estable, permitiendo la transmisión eficiente de datos y proporcionando retroalimentación al usuario.
- Autonomía y Duración de la Batería: Aunque la silla ha mostrado un desempeño notable en sus funciones, la duración de la batería podría ser una limitación, en este proyecto no se ha contemplado el uso de batería ni el consumo ya que el prototipo funciona conectado a la red.

Especificaciones Técnicas

Consumo de Componentes Individuales (aproximado)

Para el calculo del consumo, hay que tener en cuenta el uso de motores de mayor tamaño y prestaciones, teniendo en cuenta el peso que tienen que mover. Como una posibilidad esta el Nema 34 que tiene un torque de 12Nm. El resto de elementos se dejarán tal cual están, pero hay que considerar que para pasar a una escala real muchos de ellos deberán cambiar. Como el consumo de estos es ínfimo comparado con el de los motores, se tomarán los valores del modelo a escala:

2x Motores Paso a Paso NEMA 34

- Corriente Nominal: 6.2A

Otros Componentes Electrónicos

- 2x Servos: 0.5A en máxima carga cada uno, total 1A.
- MPU6050: 0.005A o 5mA.
- Sensor de Obstáculos: 0.02A o 20mA.
- STM32 Blue Pill: Hasta 0.05A o 50mA cuando está activo.

- Arduino Uno: Aproximadamente 0.05A o 50mA.
- LCD 16x2: Aproximadamente 0.02A o 20mA.
- Joystick Analógico: Aproximadamente 0.005A o 5mA.
- Controladores Pololu A4988: Aproximadamente 0.005A o 5mA cada uno cuando están activos (sin considerar la corriente de los motores).

En cuanto al consumo total estimado para la silla de ruedas motorizada, se consideraron dos motores paso a paso NEMA 34, cada uno con un consumo de 6.2 amperios, resultando en un total combinado de 12.4 amperios. Adicionalmente, se añaden los otros componentes electrónicos, como los servos, sensores y controladores, que suman aproximadamente 1.155 amperios. Esto lleva a un consumo total aproximado de 13.555 amperios para el sistema completo.

Dado este consumo, y con el objetivo de alcanzar una autonomía de dos horas, se requiere una batería con una capacidad mínima de 27.11 amperios-hora. Para este fin, se sugiere el uso de una batería de 24 voltios con al menos 27.11 Ah. Se recomienda considerar baterías de tecnología LiFePO4 o de iones de litio, dada su capacidad para proporcionar una densidad de energía más alta y una mayor durabilidad en comparación con las baterías tradicionales de plomo-ácido.

Es importante destacar que estos cálculos son estimaciones teóricas y que la duración real de la batería puede variar en función de varios factores, como las condiciones de uso y el peso transportado por la silla de ruedas.

Es importante asegurarse de que todos los componentes sean compatibles y puedan funcionar de forma segura bajo las condiciones especificadas.

Precisión

- Detección de Obstáculos: Hasta 2cm dependiendo de las condiciones de iluminación.
- Estimación del Ángulo de Inclinación: Precisión de ±1° utilizando MPU6050.

Estimación de la Velocidad de la Silla de Ruedas con Frecuencia Ajustada

Para calcular la velocidad de la silla de ruedas con la frecuencia ajustada, es esencial considerar los siguientes parámetros:

Pulsos por Revolución (PPR) de los Motores Paso a Paso: Se emplea un motor paso a paso con un ángulo de paso estándar de 1.8 grados. Esto indica que son necesarios 200 pasos para que el motor complete una revolución total de 360 grados.

Diámetro de la Rueda: Las ruedas principales en una silla de ruedas convencional tienen un diámetro de aproximadamente 24 pulgadas, equivalente a 61 cm y 30,5 cm de radio.

Usando la información anterior, podemos determinar la distancia que la silla de ruedas recorrerá en un solo paso del motor:

Distancia por Paso = $\frac{1.8}{360} \times (2 \times \pi \times 0.305)$

Distancia por Paso ≈ 0.009581 m

Con la nueva frecuencia máxima ajustada a 295 Hz (295 pasos por segundo) bajo el control del PWM:

Velocidad Lineal = 295×0,009581 m/s

Velocidad Lineal ≈ 2,8263 m/s o 10.17 km/h

Esto significa que la velocidad de la silla de ruedas puede oscilar desde 0 hasta 10,17 km/h dependiendo de la frecuencia con la que se accione el motor paso a paso.

Rangos de Trabajo

- Ángulo de Inclinación del Asiento: -30° a 30°.
- Ángulo del Servomotor para el Sensor de Obstáculos: -40° a 40°.

Conclusiones. Ensayo de ingeniería de producto.

Reflexión sobre el desarrollo realizado

El proyecto ha demostrado cómo la tecnología puede mejorar significativamente la movilidad y autonomía de personas con limitaciones físicas. La integración de sensores y microcontroladores ha permitido desarrollar un prototipo funcional que responde adecuadamente a los comandos del usuario y reacciona ante obstáculos. Sin embargo, ha sido evidente la necesidad de considerar la eficiencia energética y la robustez de los componentes para un uso prolongado y en diferentes entornos.

Perspectivas de mejoras del prototipo

Energía y Autonomía: Es fundamental incluir una batería de alta capacidad y durabilidad, y optimizar el consumo de energía de los componentes. Se podría considerar el uso de baterías de litio debido a su mayor densidad energética y menor peso.

Robustez y Durabilidad: Los componentes deben ser seleccionados pensando en la durabilidad y resistencia a condiciones adversas, como humedad y polvo.

Sensores Adicionales: Incorporar sensores de ultrasonido o LIDAR podría mejorar la detección de obstáculos en un rango más amplio, incluyendo la parte trasera de la silla.

Interfaz de Usuario Mejorada: Considerar una pantalla táctil para facilitar la configuración y el control por parte del usuario, así como la posibilidad de integrar controles de voz para una mayor accesibilidad.

Conectividad: Implementar conectividad Bluetooth o Wi-Fi para permitir el monitoreo remoto y la actualización de software.

Selección de componentes para un producto comercial

Microcontroladores: Evaluar la posibilidad de integrar un solo microcontrolador en lugar de dos, para simplificar la programación y reducir el consumo energético.

Materiales de Construcción: Utilizar aleaciones ligeras y resistentes para el chasis, como el aluminio, que ofrecen una buena relación entre peso y resistencia.

Ruedas y Motores: Seleccionar ruedas de mayor diámetro y motores con mayor torque para garantizar una movilidad efectiva en diferentes tipos de terreno.

Seguridad y Normativa: Asegurar que todos los componentes y el diseño final cumplan con las normativas de seguridad y accesibilidad vigentes.

Propuesta para el paso a un producto comercial o sistema de aplicación industrial

Escalabilidad de la Producción: Establecer procesos de fabricación que permitan escalar la producción sin comprometer la calidad.

Incremento en el uso de sensores: El uso de sensores en este tipo de herramientas es fundamental, para la seguridad de las personas. Un sensor que funcione en el reverso de la silla serviría para evitar colisiones en la situación más desfavorable.

Certificaciones y Pruebas: Obtener las certificaciones necesarias y realizar pruebas exhaustivas para garantizar la seguridad y fiabilidad del producto.

Estrategia de Mercado: Identificar el mercado objetivo y desarrollar estrategias de marketing enfocadas en las ventajas y características innovadoras del producto.

Accionar ante errores en los sensores/dispositivos

El acelerómetro/giróscopo MPU 6050, un componente clave para la estabilización y orientación de la silla de ruedas, presenta una delicadeza inherente en su construcción. Dada su tendencia a perder conexión con el microcontrolador, se implementó un protocolo de confirmación de conexión. En caso de una respuesta negativa, el sistema está diseñado para reiniciar automáticamente el sensor, intentando restablecer la comunicación y asegurando así la continuidad de sus funciones críticas.

En contraste, el sensor de proximidad muestra una mayor robustez y fiabilidad. Equipado con una luz indicadora, este sensor permite un seguimiento visual de su operatividad, facilitando el diagnóstico y mantenimiento. Aunque es menos propenso a fallos, si llegara a quedar inhabilitado, simplemente dejaría de detectar obstáculos sin interrumpir las operaciones generales de la silla. Cabe destacar que, en tal escenario, la responsabilidad de evitar obstáculos recae directamente en el usuario.

Para robustecer aún más el sistema, se considera esencial implementar un mecanismo de 'watchdog' o un sistema de supervisión similar. Esta funcionalidad actuaría como una red de seguridad contra fallos de comunicación, que podrían llevar a situaciones riesgosas como el embalamiento de las ruedas o el atascamiento de los motores. Una estrategia viable sería la implementación de un sistema de mensajes de chequeo con eco: si el controlador principal no recibe una respuesta dentro de un tiempo establecido, activaría un protocolo para llevar a la silla a un estado seguro en este caso energizado, para prevenir cualquier accidente.

Conclusiones personales

Trabajar en el desarrollo de la silla de ruedas comandada ha sido una experiencia enriquecedora y educativa. Este proyecto me ha permitido apreciar cómo la tecnología puede tener un impacto directo y positivo en la vida de las personas, especialmente en el ámbito de la asistencia médica. La combinación de la ingeniería mecánica y electrónica presentó desafíos interesantes, pero también oportunidades para aprender y crecer como ingeniero.

Una de las lecciones más valiosas ha sido entender la importancia de una interfaz de usuario bien diseñada. Comprender que la tecnología no solo se trata de cómo funciona, sino también de cómo interactúa con sus usuarios, ha sido fundamental. Este aprendizaje es algo que planeo llevar a futuros proyectos.

Además, este proyecto ha abierto caminos para explorar aún más el campo de la asistencia tecnológica. Veo oportunidades de mejora e innovación, especialmente con la aplicación de materias como control y sistemas. La idea de integrar conceptos de inteligencia artificial y aprendizaje automático para mejorar la autonomía y adaptabilidad de la silla es emocionante y algo que me gustaría explorar más adelante.

Referencias

- Apuntes catedra “Microcontroladores y Electrónica de Potencia” – Facultad de Ingeniería – UNCuyo
- https://www.pololu.com/file/0J450/a4988_DMOS_microstepping_driver_with_translator.pdf
- https://datasheetspdf.com/pdf-file/791970/TowerPro/SG90/1
- http://magicduino.com/Images/ItemsMedia/File/7203.pdf
- https://docs.arduino.cc/resources/datasheets/A000066-datasheet.pdf
- https://stm32-base.org/boards/STM32F103C8T6-Blue-Pill.html
- https://www.nxp.com/docs/en/data-sheet/PCF8574_PCF8574A.pdf
- https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
- https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf
- https://datasheetspdf.com/pdf-file/1402034/Joy-IT/KY-023/1