Programación en Tiempo Real (FreeRTOS)

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

Proyecto PAPIME PE109416

Manual para construir un robot de enjambre.

Programación en Tiempo Real (FreeRTOS)

Este manual forma parte de una serie de documentos diseñados para guiar a los participantes del Taller de Robótica de Enjambre: Nivel 4 – PROGRAMACIÓN EN TIEMPO REAL. Este cuarto taller práctico tiene como meta programar los microcontroladores ARM-Cortex con FreeRTOS.

Manual para construir un robot de enjambre.

Programación en Tiempo Real (FreeRTOS)

¿Qué es un sistema en Tiempo Real?

La palabra tiempo significa que el correcto funcionamiento de un sistema depende no sólo del resultado lógico que devuelve la computadora, también depende del tiempo en que se produce ese resultado. La palabra real quiere decir que la reacción de un sistema a eventos externos debe ocurrir durante su evolución

Fig. 4.1 Ejemplo de sistemas en Tiempo Real.

Los sistemas operativos en tiempo real fueron diseñados para dos clases generales de aplicaciones: de respuesta evento y de control de bucle cerrado. Las aplicaciones de respuesta de eventos, tales como la inspección visual automatizada de piezas de la línea de montaje, requieren una respuesta a un estímulo dado una cierta cantidad de tiempo. En este sistema de inspección visual, por ejemplo, cada parte debe ser fotografiado y analizado antes de la línea de montaje se mueva. Al programar cuidadosamente una

Sistema Operativo en Tiempo Real

En general, un sistema operativo (OS) es responsable de la gestión de los recursos de hardware de una computadora así como el almacenamiento y ejecución de aplicaciones en el equipo. Un RTOS realiza estas actividades, pero también esta especialmente diseñado para ejecutar aplicaciones con una sincronización muy precisa y un alto grado de fiabilidad. Esto puede ser muy importante en los sistemas de medición y automatización en donde el tiempo de inactividad es costoso o un retraso de algún programa podría causar un riesgo de seguridad.

En esta parte del taller los alumnos se enfrentan a la construcción de un robot de enjambre con freeRTOS.

aplicación que se ejecuta en un sistema operativo de tiempo real, los diseñadores que trabajan en la respuesta evento aplicaciones pueden garantizar que la respuesta va a suceder de manera determinista (dentro de una cierta cantidad máxima de tiempo). En vista de el ejemplo de inspección partes, utilizando un sistema operativo de propósito general podría resultar que una parte no se esté inspeccionando correctamente, por lo tanto, se tendría que retrasar la línea de montaje, haciendo que esa parte analizada se deseche, o se envíe como una parte potencialmente defectuosa.

Por el contrario, los sistemas de control en bucle cerrado, como el de un sistema de control de velocidad crucero del automóvil, continuamente procesan datos de feedback para ajustar uno o más salidas. Debido a que cada valor de Output depende de procesamiento de los datos de entrada en una cantidad fija de tiempo, es fundamental que los plazos de bucle se cumplan con el fin de asegurar que se producen las salidas correctas. ¿Que pasaría si un sistema de control de crucero no ha podido determinar que el ajuste del acelerador debe estar en un punto dado en el tiempo? Una vez más, el sistema operativo de tiempo real puede garantizar que los datos de entrada del sistema de control se procesen en la misma cantidad de tiempo.

La latencia de interrupción se mide como la cantidad de tiempo entre cuando un dispositivo genera una interrupción y cuando se atiende ese dispositivo. Mientras los sistemas operativos de propósito general pueden tener una cantidad variable de tiempo para responder a una interrupción determinada, los sistemas operativos en tiempo real deben garantizan que todas las interrupciones serán atendidos dentro de una cierta máxima cantidad de tiempo.

Gestión de procesos.

La Realización de tareas cuasi paralelo en un procesador utilizando procesos o hilos (proceso ligero) son:
• Mantener estados de procesos, gestión de colas de proceso.
• Tareas preventivas (cambio de contexto rápido) y rápido manejo de interrupciones.
•Planificación de la CPU (garantizando plazos, minimizando proceso de los tiempos de espera, la equidad en la concesión de recursos como potencia de cálculo).
•Proceso de sincronización (secciones críticas, semáforos, monitores, de exclusión mutua).
•Comunicación entre procesos (buffering).
•Apoyo de un reloj de tiempo real como una referencia de tiempo interna

¿En qué hardware se puede utilizar FreeRTOS?

El código fuente soportado oficialmente es propiedad exclusiva de Amazon Web Services La parte específica de la arquitectura del microcontrolador de un puerto FreeRTOS se denomina capa de puerto. Cada capa de puerto es 'soportada oficialmente' o 'contribuida'.

Ficha 4.1

Actividad 4.1.

a) Ver https://www.freertos.org/RTOS_ports.html
b) Ver https://docs.aws.amazon.com/es_es/freertos/latest/userguide/freertos-getting-started.html

Entorno de desarrollo para IoT.

El **internet de las cosas** (en inglés, *Internet of Things*, abreviado *IoT*, es un concepto que se refiere a una interconexión digital de objetos cotidianos con internet. Los microcontroladores (MCU) constan de un solo chip que contiene un procesador simple y están presentes en muchos dispositivos, incluidos electrodomésticos, sensores, monitores de actividad física, sistemas de automatización industriales y automóviles. Muchos de estos pequeños dispositivos podrían beneficiarse de su conexión con la nube o conexión local con otros dispositivos. Actualmente Amazon es una de las compañías que promueve sus servicios en la nube y proporciona un entorno de programación para MCU usando FreeRTOS.

Amazon FreeRTOS es un sistema operativo popular de código abierto para microcontroladores, con bibliotecas de software que permiten conectar de forma sencilla y segura sus pequeños dispositivos de poca potencia con los servicios en la nube de Amazon Web Server (AWS).

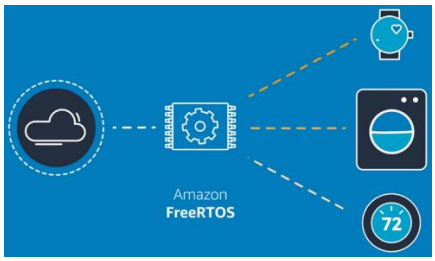


Fig. 4.2 Amazon FreeRTOS.

Ficha 4.2

Actividad 4.2.

a) Revisar la documentación de Amazon FreeRTOS y seleccione el chip más adecuado.
b) Siga la Guía de Programación de Espressif para instalar en Windows el entorno MSYS2 las herramientas de desarrollo ESP-IDF

UN ENFOQUE DE COMPETENCIAS.

La FES Aragón actualmente está pasando un proceso de Certificación en su Modelo de Enseñanza en sus carreras de Ingeniería, por lo que a continuación se propone una rubrica para calificar el nivel de comprensión de los integrantes del Taller:

Rúbrica de Evaluación – Los alumnos de Ingeniería de la FES Aragón Aplican, analizan y sintetizan procesos de diseño de ingeniería que resulten en proyectos que cumplen las necesidades especificadas.

Criterios de Desempeño	**Indicadores dimensionales**	**Tipo de Rúbrica:** Analítica.			
		Lo supera (10)	**Lo logra (8)**	**Parcialmente (6)**	**No lo logra (0)**
Realizar experimentos siguiendo el protocolo establecido.	Muestra los métodos y equipos seleccionados para la experimentación.	Explica métodos y equipos para la experimentación enfocada a los sistemas con microprocesador.	Muestra el experimento funcionando, pero no sabe cómo funciona el sistema con microprocesador.	Realiza experimentos siguiendo parcialmente el protocolo establecido.	No es capaz de seguir el protocolo establecido para la realización de experimentos.
Utiliza información experimental para el análisis, evaluación y diseño con ESP32.	Registra la eficiencia del diseño a través de simulaciones y pruebas en hardware y software.	Analiza información experimental relevante para realizar un nuevo diseño.	Utiliza información experimental para realizar el mismo proceso de diseño.	Clasifica la información experimental sin buenos resultados en las pruebas.	Ignora la información experimental en el diseño.

Experimento 4.1.

Una compañía de aplicaciones de software para el IoT, solicita la asesoría de los Ingenieros en Electrónica de la FES Aragón para instalar el entorno de programación más adecuado. (FreeRTOS con lenguaje C) para desarrollar aplicaciones con la Tarjeta de Desarrollo ESP32 DevKitC

Construcción del Modelo.

Un equipo de estudiantes selecciona las herramientas de desarrollo necesarias para el diseño de aplicaciones en IoT.

Documentar el modelo y proceso.

¿Cuáles son las herramientas que selecciono y por qué?
¿Qué herramienta matemática utilizo?
¿Qué características se deben considerar para instalar el entorno de programación más adecuado?
¿Qué fuentes y recursos utilizo?

Refinación mediante autoevaluación

Notifique las problemáticas encontradas y su solución.

Especifique las consideraciones de seguridad, costo y técnicas requeridas.

Generación del Modelo.

Se presentan resultados y posibles mejoras.

Efectividad

Análisis de costos y portabilidad.

Conclusiones técnicas, éticas y oportunidades.

PWM para el control de motores con FreeRTOS.

El control de la velocidad de un motor de CC usando la Regulación por Ancho de Pulso está basada en el ajuste de voltaje en forma de una onda cuadrada, la energía que recibe el motor disminuirá de manera proporcional a la relación entre la parte alta (habilita corriente) y baja (cero corrientes) del ciclo de la onda cuadrada.

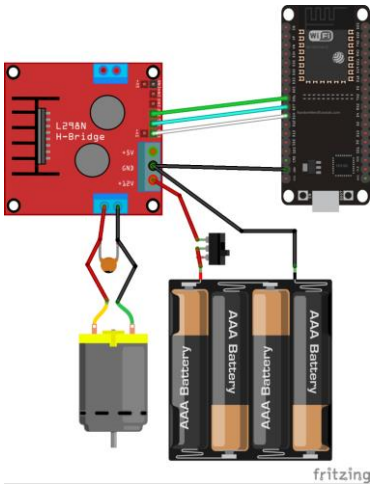


Fig. 4.3 ESP32 para controlar motores con PWM.

Los ejemplos proporcionados por la herramienta ESP-IDF de ESPRESSIF son un buen inicio para familiarizarse con la programación de freeRTOS. En particular, el manejo de periféricos (GPIO, ADC, UART, etc) ubicados por lo general en:

C:\msys32\home\USUARIO\esp\esp-idf\examples\peripherals

Ficha 4.3

Actividad 4.3.

a) Abra una ventana de terminal MSYS2 MINGW32 ejecutando C:\msys32\mingw32.exe . El entorno en esta ventana es un shell bash. Moverse al directorio llamado “esp”

b) Copiar el directorio “mcpwm_brushed_dc_control”

```
cd esp
cp -r $IDF_PATH/examples/mcpwm/mcpwm_brushed_dc_control .
cd mcpwm_brushed_dc_control
```

c) Costruir y flashear la aplicación
```
make flash
```

d) Descargar el esquemático de la tarjeta para ver los pines de asignados al PWM.

Experimento 4.2.

El Laboratorio REEX de la FES Aragón, lanza una convocatoria para los alumnos de Ingeniería en Electrónica de la FES-Aragón para diseñar de un robot móvil controlado por la Tarjeta de Desarrollo ESP32 DevKitC, usando el sistema operativo FreeRTOS.

Construcción del modelo.

Un equipo de estudiantes desarrolla la primera etapa:

I. Construir el chasis del robot con dimensiones máximas de 10x10x5 cm
II. Adaptar un banco de baterías de 4 pilas AA, para proporcionar 9V dc para los motores y 5V dc para la parte digital (se sugiere utilizar el módulo elevador de voltaje XL6009E1).
III. Colocar un puente H para controlar la dirección de los motores (se sugiere el módulo DRV8833)
IV. Modificar el programa “mcpwm_brushed_dc_control_example.c” para hacer la siguiente secuencia:

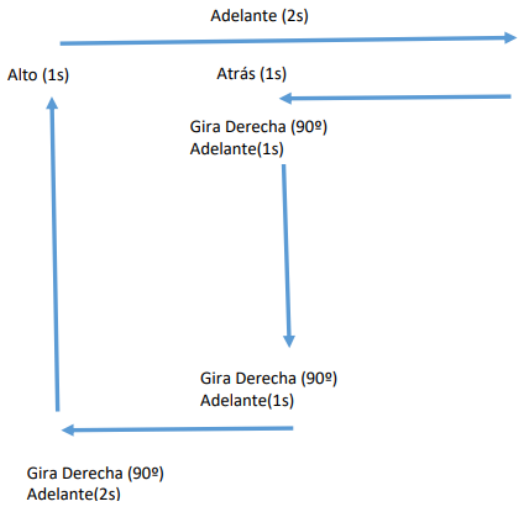


Fig. 4.4 ESP32 para controlar motores con PWM.

Documentar el modelo y proceso.

¿Cuáles son las herramientas que selecciono y por qué?
¿Qué herramienta matemática utilizo?
¿Qué características se deben considerar para instalar el entorno de programación más adecuado?
¿Qué fuentes y recursos utilizo?

Refinación mediante autoevaluación

Notifique las problemáticas encontradas y su solución.
Especifique las consideraciones de seguridad, costo y técnicas requeridas.

Generación del Modelo.

Se presentan resultados y posibles mejoras.

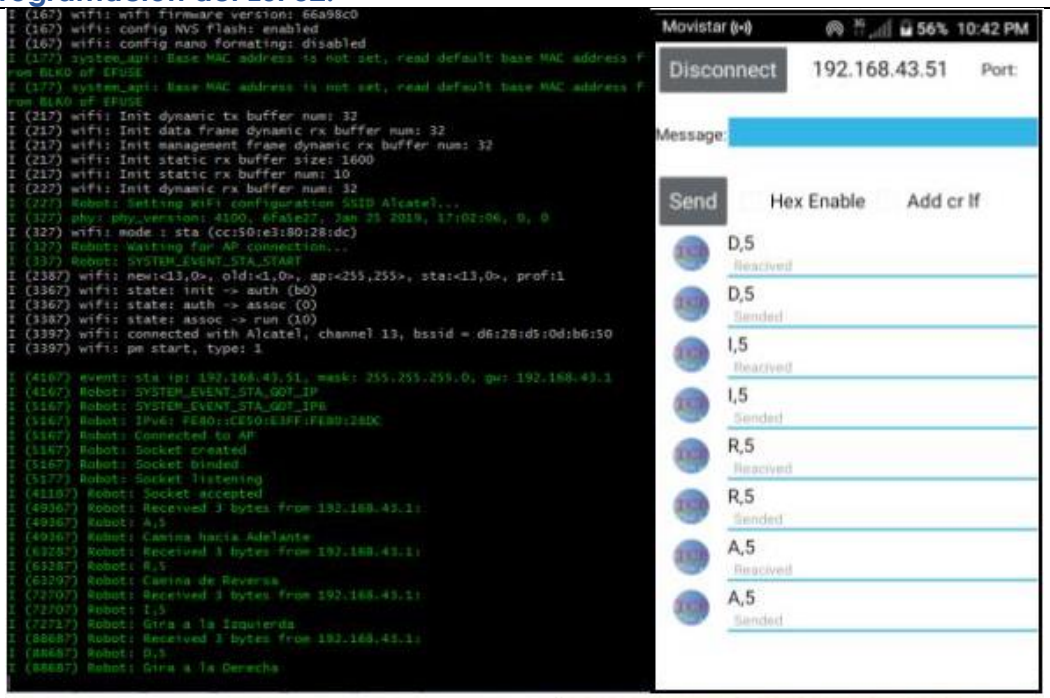
Efectividad

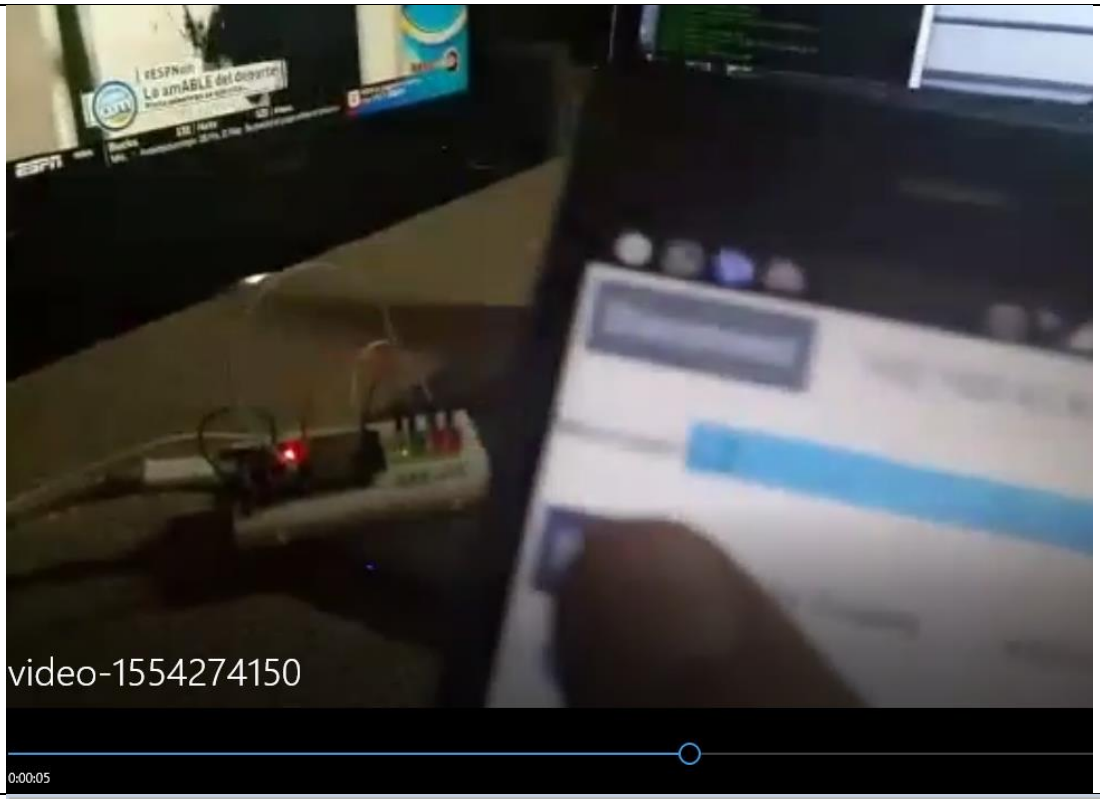
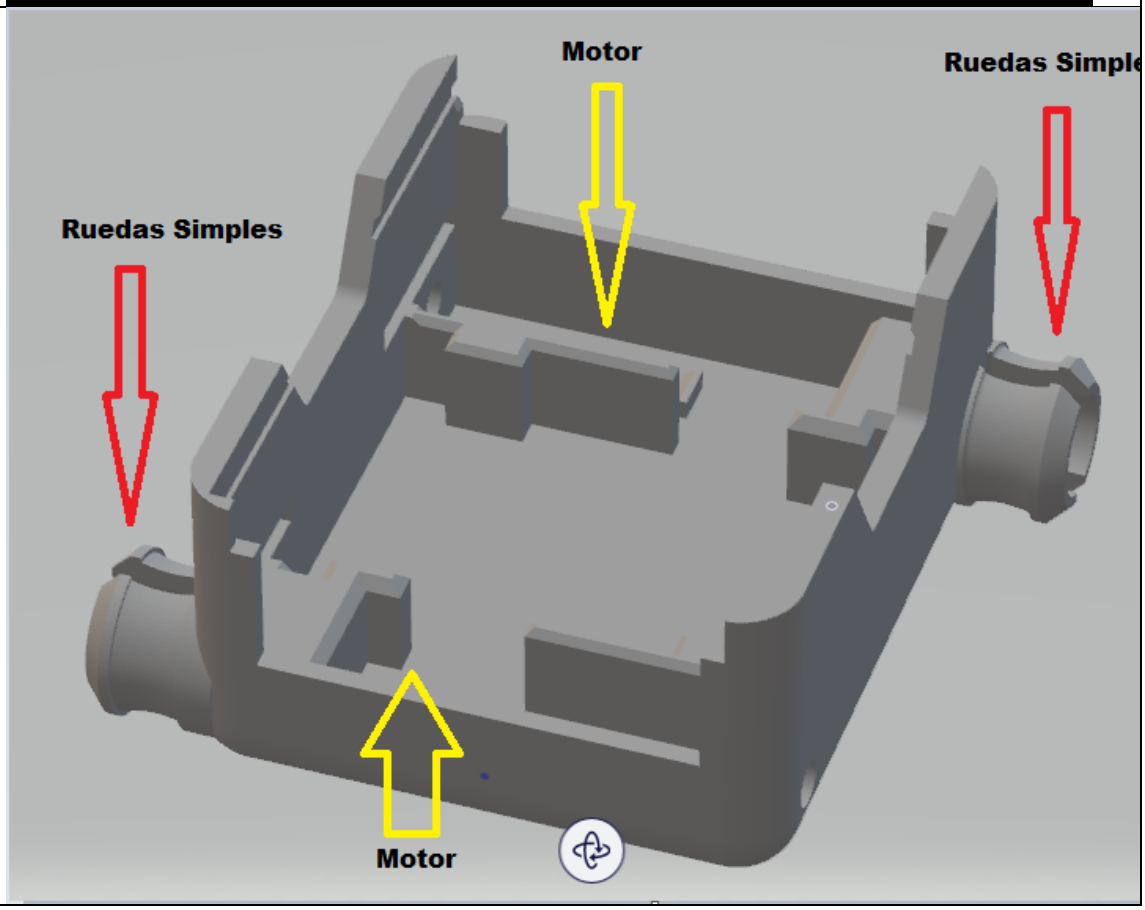
Análisis de costos y portabilidad.
Conclusiones técnicas, éticas y oportunidades.

Contribución programación del ESP32.

Resultados Experimento . 4.1

Programación del ESP32 DevKitC Con FreeRTOS



ESP32 DevKitC Programación TCP con FreeRTOS	
Diseño Sem. 2019-II	

Prototipo4.1 ESP32 con FreeRTOS.