La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

Aplicación de la biblioteca TI-RTOS para procesar audio con la tarjeta DK-TM4C129X

Santiago Espinosa Felipe✉, Moreno Espinosa José Antonio

Instituto de Electrónica y Mecatrónica, Universidad Tecnológica de la Mixteca
✉fsantiag@mixteco.utm.mx, jamoreno@mixteco.utm.mx

Resumen

Un sistema de tiempo real (STR) debe cumplir con requerimientos estrictos de temporización, es decir, sus respuestas deben generarse en intervalos de tiempo acotados. El diseño y evaluación de un STR se simplifica con la ayuda de un RTOS, que es un software base para gestionar la ejecución concurrente de tareas, sobre todo cuando éstas tienen diferentes prioridades. En este trabajo se demuestra que el uso del TI-RTOS simplifica el diseño de un STR con la tarjeta de desarrollo DK-TM4C129X de Texas Instruments®. Para ello, se diseñó un sistema que procesa audio y que realiza dos tareas principales: El procesamiento de la señal entrante y la atención de una interfaz de usuario basada en una pantalla LCD-Touch. La primer tarea es prioritaria y más frecuente que la segunda, aunque la CPU invierte mucho más tiempo en la segunda tarea. El procesamiento consiste en la aplicación de tres filtros centrados en diferentes frecuencias, con ganancias configurables por el usuario a través de barras de desplazamiento. Como resultados, el RTOS simplificó el desarrollo del sistema y se validó que el periodo entre muestras es suficiente para el procesamiento de los datos, las muestras se toman cada 125 us (8000 mps).

Palabras clave: Tiempo real, RTOS, Tiva-C, audio, filtros.

Abstract

A real-time system (RTS) must comply with strict timing requirements, it means that its responses must be generated in limited time intervals. The design and evaluation of a RTS is simplified with the use of an RTOS, which is a base software to manage the concurrent execution of tasks, especially when they have different priorities. This paper shows that the use of the TI-RTOS simplifies the design of an RTS with the Texas Instruments® DK-TM4C129X development card. For this, a system that processes audio was designed and it realizes two main tasks: The processing of the incoming signal and the attention of a user interface based on an LCD-Touch screen. The first task is priority and more frequent than the second, although the CPU spends much more time in the second task. The processing consists of the application of three filters centered on different frequencies, with user configurable gains through scrollbars. As a result, the RTOS simplified the development of the system and validated that the period between samples is enough for the data processing, the samples are taken every 125 us (8000 sps).

Keywords: Real time, RTOS, Tiva-C, audio, filters.

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

1. Introducción

El Sistema Operativo de Tiempo Real de Texas Instruments (TI-RTOS) consiste en un conjunto de herramientas integradas para dispositivos del mismo fabricante y está basado en un kernel multitarea llamado SYS/BIOS. El kernel facilita el diseño de sistemas en donde el tiempo de respuesta es vital para su funcionamiento, proporcionando una organización multi–procesos, abstracción de hardware, análisis en tiempo real y herramientas de configuración [1].

En este trabajo se utilizó al SYS/BIOS con la tarjeta de desarrollo DK-TM4C129X, la cual se muestra en la figura 1 y tiene las siguientes características [2]:

- Microcontrolador Tiva TM4C129X con núcleo ARM® Cortex™-M4 de 32 bits.
- Pantalla QVGA a color con touch screen resistivo.
- Puerto Ethernet 10BASE-T/100BASE-TX.
- Conector USB Micro-AB para anfitrión/dispositivo/OTG.
- Base para microSD.
- 3 botones y un LED RGB para aplicaciones del usuario.
- Voltaje de referencia de 3.0 V.
- Conectores para expansión
- Interfaz para depuración ICDI (*In-Circuit Debug Interface*).
- Botón de reset.

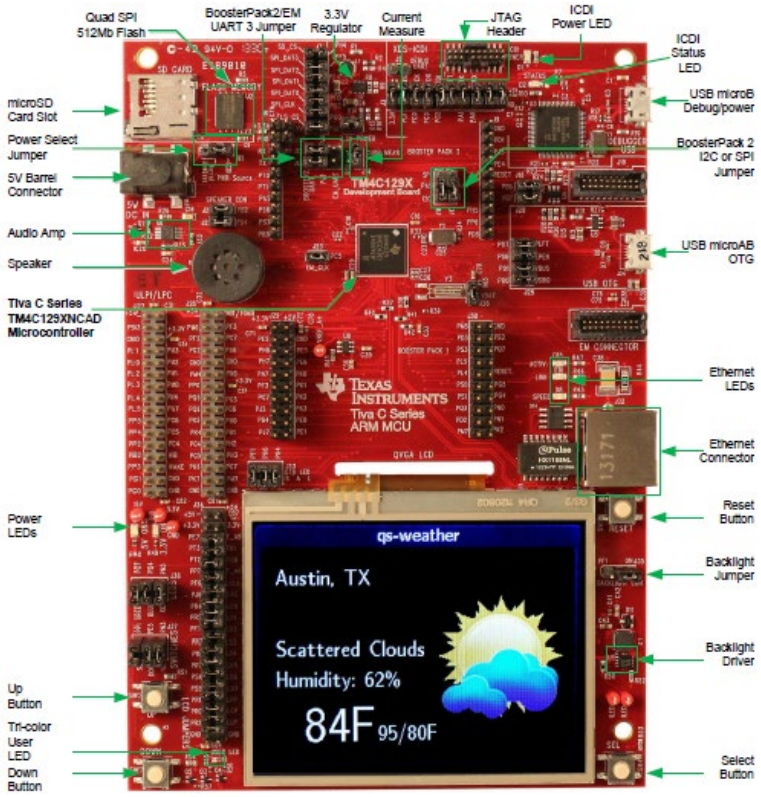


Figura 1. Tarjeta de desarrollo DK-TM4C129X.

El elemento principal de la tarjeta es el microcontrolador TM4C129X, éste está soportado por un núcleo ARM® Cortex™-M4 y puede operar hasta 120 MHz. El MCU tiene un mapa lineal de memoria con diferentes espacios: 1024 KB de Flash para código, 256 KB de RAM y 6 KB de EEPROM para datos, así como una ROM interna con rutinas optimizadas para el manejo de periféricos. El microcontrolador TM4C129X cuenta con un extenso número de periféricos, incluyendo puertos de entrada o salida, temporizadores, UART's, interfaces I2C, interfaces SPI y ADC's de 12 bits [3].

Los recursos de la tarjeta DK-TM4C129X la hacen una opción muy adecuada para la evaluación del TI-RTOS. Como caso de estudio se eligió el desarrollo de un sistema para el procesamiento de audio, por ser una aplicación de tiempo real. El procesamiento consiste en la ejecución de tres filtros digitales tipo Butterworth a una señal de entrada, cada filtro tiene una ganancia independiente que puede ser modificada en tiempo de ejecución desde la pantalla LCD-Touch de la tarjeta.Esto significa que el tiempo de la CPU se debe distribuir en dos tareas principales, la ejecución de los algoritmos de procesamiento y la atención de una interfaz simple de usuario. La primer tarea es prioritaria porque cada muestra debe ser procesada antes de que llegue la siguiente. La segunda tarea ocurrirá con menos frecuencia aunque requiere de un tiempo de atención mucho mayor, porque implica el sondeo de la zona táctil así como la actualización de la información.

El audio se ingresa por medio de uno de los ADC con que cuenta el MCU, sin embargo, el MCU no cuenta con un DAC para la generación de resultados, para ello se utilizó al circuito integrado MCP4922 de la firma Microchip, éste es un DAC con interfaz SPI que también tiene una resolución de 12 bits y puede operar a una frecuencia de hasta 20 MHz [4].

El interés principal de este trabajo es el uso del TI-RTOS, como caso de estudio se elige procesar audio por ser una aplicación de tiempo real. El TI-RTOS ha sido usado en diversas aplicaciones, por ejemplo, en [5] se describe un sistema de supervisión médica que monitorea de forma remota varios parámetros de pacientes, se trata de un sistema integrado que maneja los parámetros críticos en unidades de hospital/enfermería, las muestras se procesan utilizando una CPU basada en ARM y para lograr métricas de rendimiento se usa al TI-RTOS. En [6], con el apoyo del TI-RTOS se optimiza un sistema de micro-mallas híbrido basado en fuentes de energía renovables y baterías, en tiempo real se calcula la energía requerida por los aparatos electrónicos en uso y se selecciona la mejor opción de suministro, como apoyo, el algoritmo recibe los parámetros de temperatura y velocidad del viento. En [7] se describe un sistema inalámbrico para el monitoreo de un oleoducto de crudo con una distancia de 1350 Km desde la estación de bombeo hasta los tanques de almacenamiento, se distribuyen 135 estaciones de monitoreo y señalización basadas en un MCU MSP430 de TI, las estaciones cuentan con una celda solar, batería y enlace WiFi, así como sensores de temperatura, fuego y nivel. Las tareas del MCU son agendadas con ayuda del TI-RTOS.

En la sección 2 se describe el diseño de los filtros que se aplican a la señal de audio, en la sección 3 se explica el diseño del sistema, con base en la metodología de sistemas embebidos. La sección 4 presenta los resultados alcanzados, sobre todo resaltando el uso del TI-RTOS y las conclusiones son descritas en la sección 5.

2. Diseño de los filtros

Para la tarea de procesamiento de audio se diseñó un banco de tres filtros tipo Butterworth, las muestras se tomarán a una razón de 8000 mps, por lo que para cumplir con el criterio de Nyquist, la frecuencia de la señal de entrada se debe limitar 4 KHz. Establecido el valor máximo de frecuencia, el rango se divide en seis octavas, las cuales tienen sus límites en 60, 120, 240, 500, 1000, 2000 y 4000 Hz.

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

La distribución de frecuencias en 3 filtros se fundamenta en [8], se plantea un filtro pasa bajas para las dos octavas más bajas con frecuencia de corte a -3 dB en 120 Hz, para las dos octavas más altas el filtro requerido es un pasa altas con frecuencia de corte a -3 dB en 2000 Hz y para las dos octavas centrales se requiere de un filtro pasa banda con frecuencia central en 490 Hz, la cual se obtiene de la ecuación 1, con f_l = 240 y f_u = 1000.

$$f_c = \sqrt{f_l f_u} \tag{1}$$

En la figura 2 se muestra la forma en que se aplican los filtros sobre la señal de entrada, en la salida se acumula el resultado y la ganancia de cada filtro es controlada de manera independiente mediante una barra de desplazamiento.

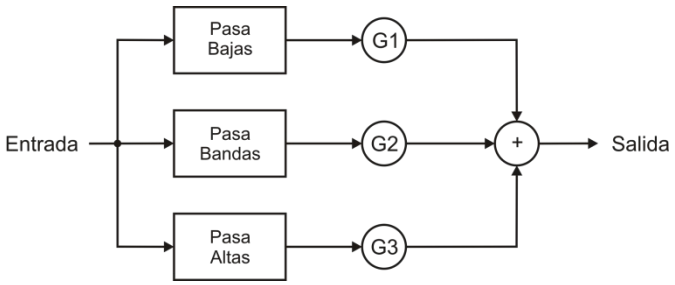


Figura 2. Organización de los filtros.

El diseño de los filtros se realizó usando la herramienta *fdatool* de Matlab, la cual ofrece una interfaz gráfica para seleccionar el tipo de filtro a diseñar con sus restricciones y devuelve los coeficientes del filtro. Usando las condiciones de diseño mencionadas anteriormente, *fdatool* devuelve los coeficientes mostrados en la tabla 1. Las bandas de rechazo de los tres filtros se eligieron menores a -60 dB, para que el orden de los filtros no fuera mayor de 8, de tal forma que el sistema los pueda realizar en tiempo real y el resultado de la ecualización sea notorio durante la ejecución.

Tabla 1. Coeficientes de los filtros digitales.

Filtro	Coeficientes
Pasa bajas	a0 = 1, a1 = -3.6159, a2 = 4.9198, a3 = -2.9841, a4 = 0.6807 b0 = 0.0000243, b1 = 0.0000971, b2 = 0.0001457, b3 = 0.0000971, b4 = 0.0000243
Pasa banda	a0 = 1.0, a1 = -3.3255227400418748, a2 = 5.3167114774765905, a3 = -5.6236534488876035, a4 = 4.4376562001855229, a5 = -2.5620298469228309, a6 = 1.018098177727131, a7 = -0.26201367183278318, a8 = 0.037673096418663427 b0 = 0.038029185045784168, b1=0, b2 = -0.15211674018313667, b3 = 0, b4 = 0.22817511027470502, b5 = 0, b6 = 0.15211674018313667, b7 = 0, b8 = 0.038029185045784168
Pasa alta	a0 = 1.0000, a1 = -0.3013, a2 = 0.8116, a3 = -0.1441, a4 = 0.1230, a5 = -0.0097, a6 = 0.0019 b0 = 0.0374, b1= -0.2242, b2 = 0.5605, b3 = -0.7474, b4 = 0.5605, b5 = -0.2242, b6 = 0.0374

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

3. Desarrollo del sistema

Por sus características, y aún con el uso del SYS/BIOS, el proyecto cae en la categoría de los sistemas embebidos, por lo que para su desarrollo y documentación se sigue la metodología propuesta por Arnold S. Berger [9], quien organiza el ciclo de desarrollo en 7 fases, como se muestra en la figura 3. La fase 7 se descarta porque queda fuera del alcance del trabajo.

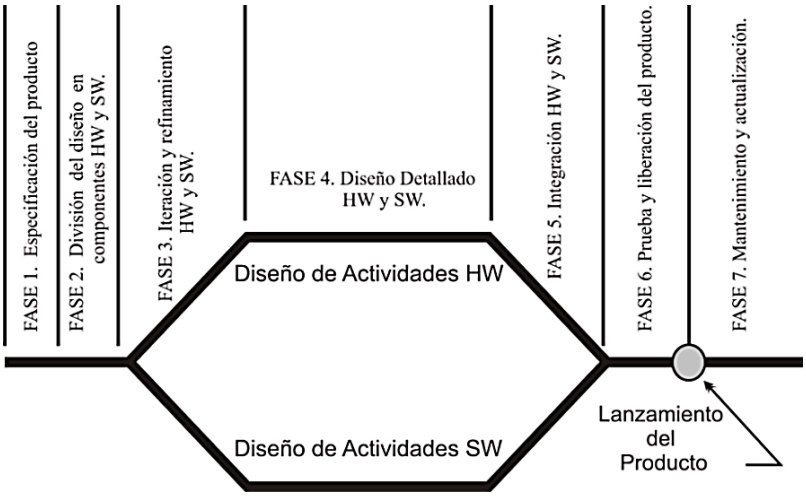


Figura 3. Ciclo de desarrollo de un sistema empotrado [9].

3.1 Especificación del producto

El objetivo del sistema es el procesamiento de audio con la tarjeta DK-TM4C129. Una señal de entrada es procesada por medio de tres filtros digitales centrados en diferentes frecuencias, la ganancia de los filtros es configurada por el usuario a través de tres barras de desplazamiento manejadas desde una pantalla LCD-Touch, incluida en la tarjeta. El elemento central de la tarjeta es el microcontrolador Tiva-C con número de serie TM4C129XNCZAD, éste cuenta con dos convertidores Analógico-Digital y uno de ellos es empleado para obtener la señal de entrada, se agrega un circuito para el acondicionamiento de la señal entrante.

Para la generación de la señal de salida se emplea un convertidor Digital-Analógico externo porque el MCU Tiva-C no cuenta con este recurso. Se trata del circuito integrado MCP4922 de la firma Microchip, éste es un DAC con interfaz SPI con una resolución de 12 bits y puede operar a una frecuencia de hasta 20 MHz [4], también se requiere de un circuito de acondicionamiento para acoplar la señal analógica de salida a una bocina. En la figura 4 se muestra un diagrama con la especificación del producto, las barras de desplazamiento manejan cantidades porcentuales.

1.1 División del diseño en componentes de Hardware y Software

Los componentes de Hardware pueden dividirse en elementos externos al MCU y recursos internos. Como elementos externos, el sistema requiere de módulos de acondicionamiento de señal, basados en amplificadores operacionales, así como el DAC y la pantalla LCD-Touch. De los recursos internos el sistema utiliza sus dos ACD's, el ADC-0 es para el sondeo de la zona táctil y el ADC-1 para la recepción de la señal de entrada. También se emplea un módulo SPI para el manejo del DAC externo.

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

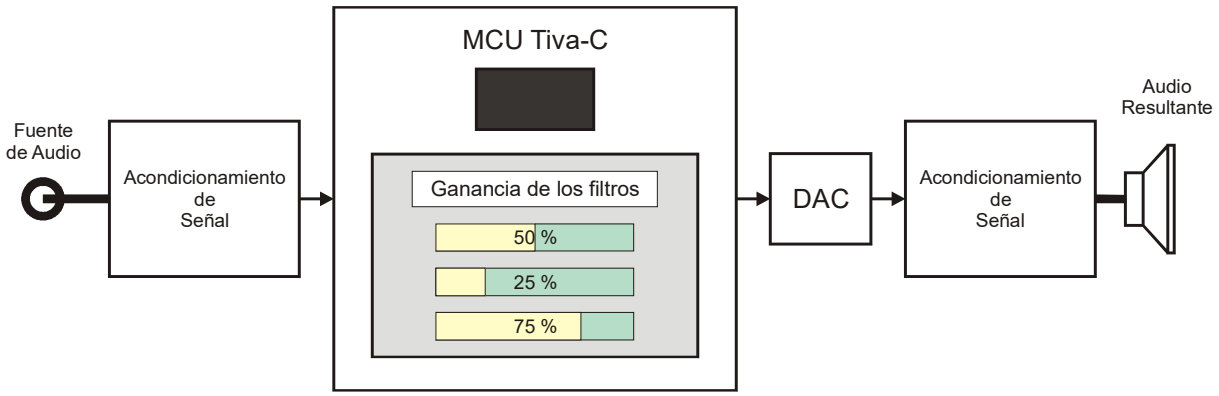


Figura 4. Especificación del producto.

Existe una relación funcional directa entre todos los componentes de Hardware, el bloque de acondicionamiento de la entrada ajusta la señal de audio a un rango de voltaje entre 0 y 3.3 V, que es el rango esperado por el ADC-1. El ADC-1 realiza conversiones dejando las muestras para su posterior procesamiento, el procesamiento considera las ganancias de los filtros definidas por el usuario, las cuales son obtenidas por medio del ADC-0. El resultado del procesamiento es enviado hacia el DAC externo por medio del módulo SPI, el DAC se encarga de generar la señal analógica para que sea acondicionada y enviada a la bocina.

Respecto al Software, el sistema requiere de tres componentes principales: La Biblioteca de Controladores Periféricos (*TivaWare Peripheral Driver Library*), la Biblioteca de Gráficos (*TivaWare Graphics Library*) y el SYS/BIOS. La primer biblioteca contiene funciones para la inicialización y el acceso a los recursos internos del MCU [10], la segunda incluye las funciones necesarias para el manejo de la pantalla, así como para la gestión de los eventos producidos en la zona táctil [11]. El SYS/BIOS es el componente del sistema que gestiona las diferentes tareas, asignando prioridades y recursos a cada una, manteniendo una ejecución en tiempo real [1].

El proyecto también requiere la inclusión de los controladores del LCD-Touch, dos bibliotecas de funciones desarrolladas por *Kentec Electronics Limited*, fabricante de la pantalla. La primera se llama **Kentec320x240x16_ssd2119_8bit** y contiene funciones para el manejo de la pantalla, la segunda biblioteca es **touch** y contiene funciones para la inicialización y sondeo de la zona táctil [12].

1.2 Diseño detallado del Hardware y Software

En la figura 5 se muestra el circuito de acondicionamiento para la señal de entrada, el circuito filtra la señal y agrega un nivel de CD de 1.6 Volts para evitar niveles de voltaje negativos y hacer posible la conversión a una señal digital, el diseño del circuito fue tomado completamente de [13], en donde se describe su funcionamiento.

La señal analógica ingresa al MCU en la terminal PE3, como se puede ver en la figura 6, donde también se muestra la conexión del microcontrolador con el DAC MCP4922, que es manejado por una interfaz SPI. Como resultado de la conversión también se genera un voltaje positivo que debe ser acondicionado para que pueda ser reproducido en una bocina.

En la figura 7 se muestra el circuito de acondicionamiento para la salida, éste se basa en el circuito integrado MC31119 que es un amplificador de audio de baja potencia. La configuración también fue tomada de [13] y el circuito recibe el voltaje generado por el DAC.

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

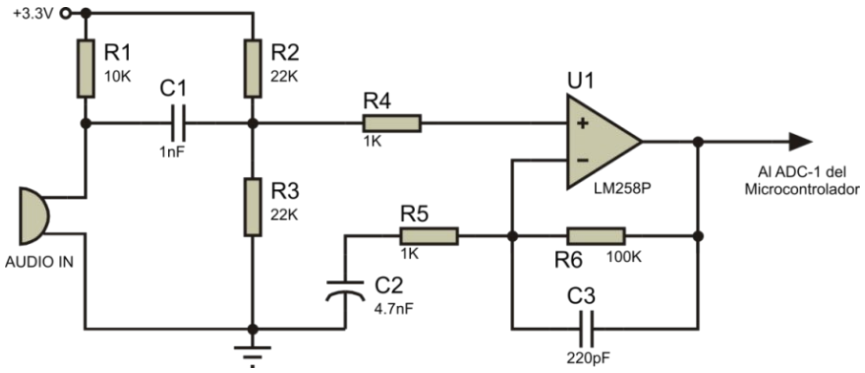


Figura 5. Circuito para acondicionar la señal de entrada.

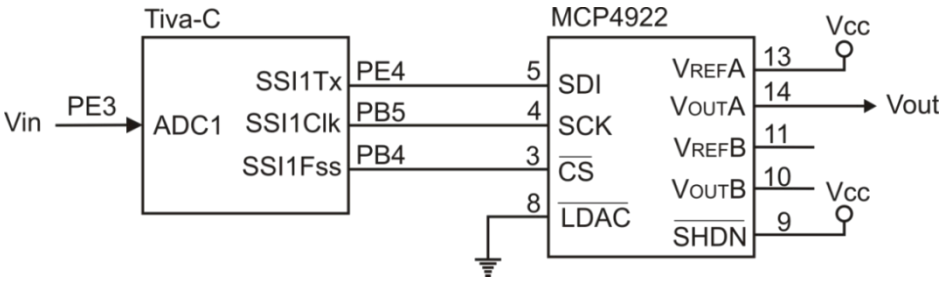


Figura 6. Conexión del Microcontrolador con el DAC.

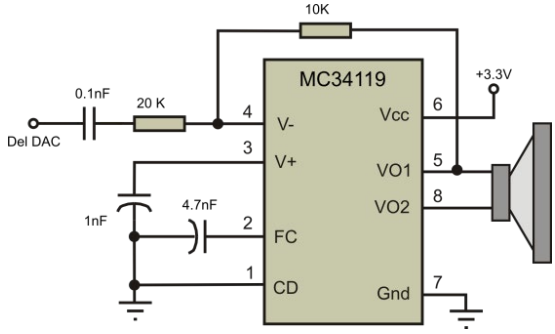


Figura 7. Amplificador de audio para la salida.

El software se organizó con la ayuda del BIOS, el BIOS permite el manejo de diferentes tipos de procesos que varían en propiedades, como se puede ver en la figura 8. El esquema de planificación es preventivo, es decir, si un proceso se está ejecutando y ocurre otro con prioridad mayor, el primero se suspende para dar paso al segundo, concluido el segundo se continuará con el primer proceso.

Los procesos que se incluyen en el sistema son:

- **Timer2A:** Es una **Hwi** generada por el temporizador 2 y sirve para definir el periodo de muestreo, la interrupción se genera en forma periódica a una frecuencia de 8 KHz y en su ISR básicamente se inicia una conversión de la señal analógica de entrada.

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

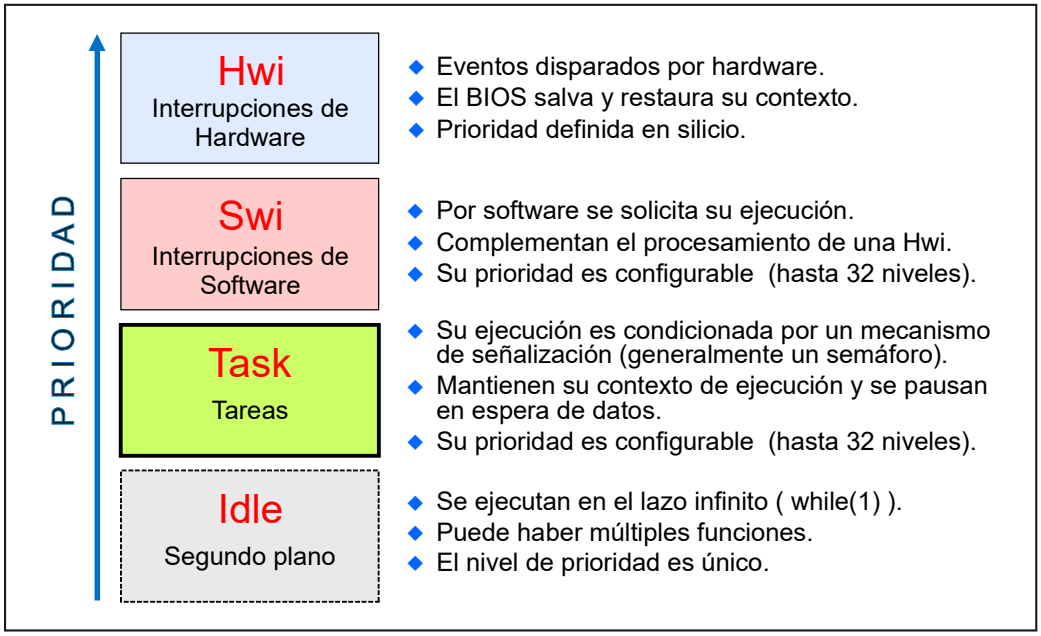


Figura 8. Tipos de procesos que puede manejar el BIOS.

- **ADC1_Hwi:** Es una **Hwi** generada por el ADC1, la interrupción se genera cuando termina la conversión de una muestra analógica a un número digital. En su ISR se obtiene el nuevo valor digital y se habilita un semáforo que da paso a la tarea de procesamiento.

- **ADC_Touch:** Es una **Hwi** generada por el ADC0, el cual está conectado a la zona táctil de la pantalla. La ISR de esta **Hwi** se denomina **TouchScreenIntHandler** y es parte de la biblioteca **Touch**, su trabajo consiste en encolar los eventos producidos en la pantalla para que sean atendidos cuando no haya procesos de mayor prioridad.

- **NuevoVal:** Es una **Task** en donde se aplican los filtros a la secuencia de datos de entrada. El proceso se mantiene detenido en espera de una nueva muestra, un semáforo señaliza la existencia del nuevo valor. Una **Task** agiliza su ejecución porque mantiene su contexto en memoria, es decir, las variables conservan su valor sin ser globales. Una vez concluido el procesamiento se inicia la transferencia del resultado hacia el DAC mediante el protocolo SPI. Para minimizar el uso del procesador, el proceso no espera el fin de la transferencia.

- **WidgetMessageQueueProcess:** Es una función que también es parte de la biblioteca **Touch.** Esta función se coloca en el plano **Idle** quedando en el nivel de prioridad más bajo. Los eventos de la pantalla no se pierden porque son detectados por el proceso **ACD_Touch** que es un **Hwi**, sin embargo, su atención se realiza cuando no hay otras tareas en proceso.

En el programa principal se inicializa la pantalla LCD, se configura la interfaz de usuario y la zona táctil para la detección de eventos. También se realiza la configuración de los recursos empleados: El reloj para que el sistema trabaje a 120 MHz, el temporizador 1 para sincronizar el procesamiento a una frecuencia de 8 KHz, el ADC 1 para digitalizar la señal de entrada y la interfaz SPI para el manejo del DAC.

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

Una vez hecha la configuración y con el alta de procesos en el proyecto por medio del BIOS, el comportamiento del sistema queda determinado por los diferentes eventos, algunos sincronizados con el temporizador 1 y otros generados por el usuario por medio de la pantalla táctil.

1.3 Integración y prueba del sistema

El programa fue desarrollado en lenguaje C en el entorno de Code Composer Studio versión 6.0 (CCS). Una vez compilado y descargado en la tarjeta DK-TM4C129 se procede a la evaluación del sistema, en la figura 9 se muestra la tarjeta con el programa descargado y funcionando.

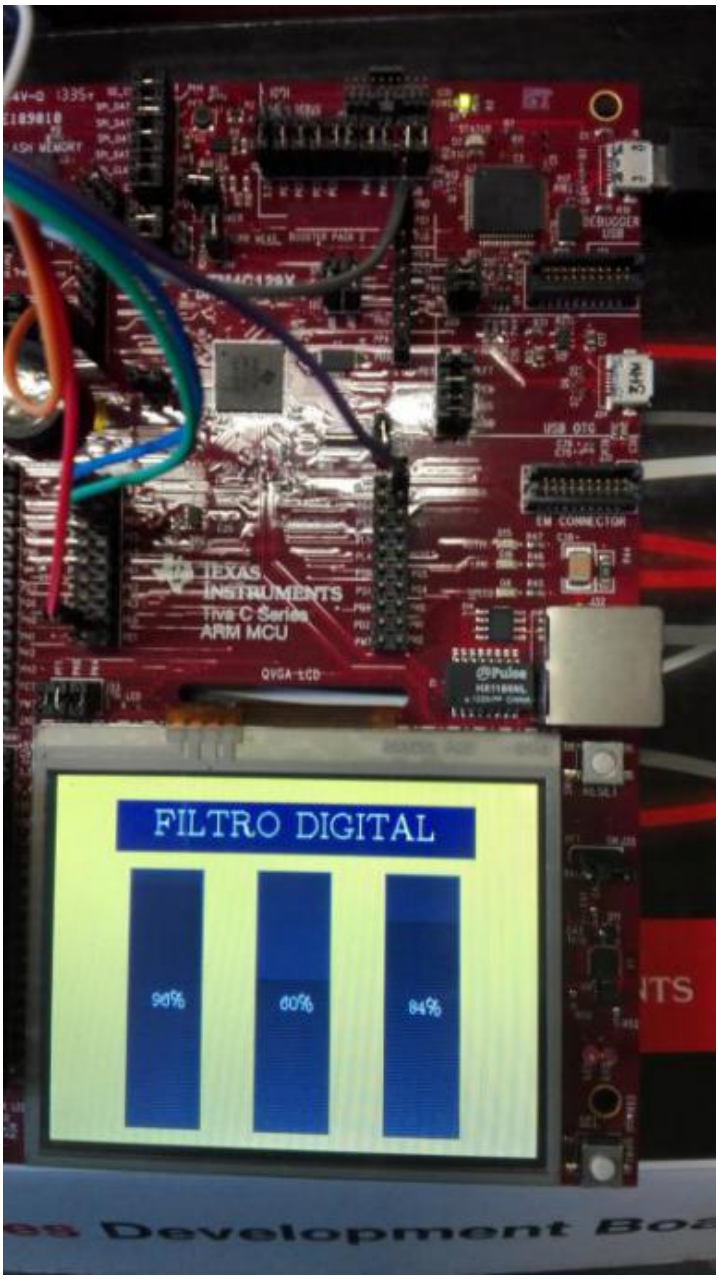
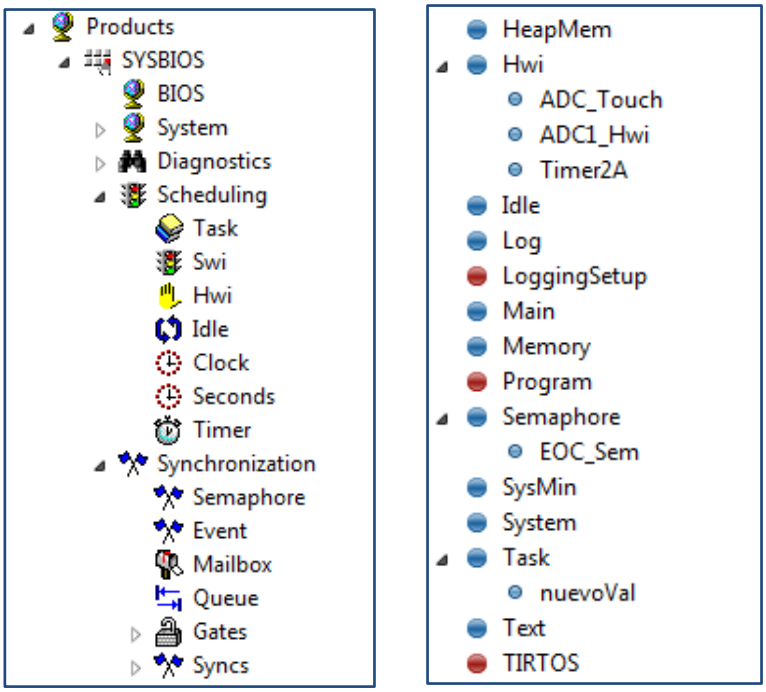


Figura 9. Sistema para procesar audio con la tarjeta DK-TM4C129X.

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

En la figura 10 se muestra parte del entorno de desarrollo, en 10(a) se pueden ver los productos del BIOS disponibles para la planificación y sincronización de tareas y en 10(b) los procesos empleados en el sistema, para las tareas descritas en la sección anterior. El semáforo *EOC_Sem* señaliza el final de una conversión del ADC, la tarea *nuevoVal* está en espera de esta señalización para realizar el procesamiento.



(a) (b)

Figura 10. Parte del Code Compouser Studio: (a) Productos disponibles para Planificación y Sincronización, y (b) Procesos empleados en el sistema.

2. Resultados

El BIOS cuenta con una Arquitectura de Instrumentación Unificada (UIA: *Unified Instrumentation Architecture*) que permite analizar la ejecución de procesos para validar los tiempos de respuesta, en la figura 11 se puede ver la ejecución gráfica del sistema, se observa cuando inicia y termina la **HWi** del temporizador, así como el momento en que se pone en alto el semáforo para continuar con la **Task** de procesamiento y cuando la **Task** queda suspendida en espera de otra señalización.

El punto X1 que aparece en la figura 11 (sobre el eje horizontal) señala el inicio de la **HWi** con el proceso **Timer2A**, éste ocurre en el tiempo 1 728 928 uS. El punto X2 marca el inició de otra ejecución y se produce en el instante 1 719 052 uS. La diferencia entre estos dos puntos es de 124 uS, lo que indica una frecuencia de muestreo de 8.064 KHz. Puede notarse en X2 que estaba otra tarea de baja prioridad, la cual fue temporalmente suspendida para dar paso a la **HWi**.

Otra herramienta para analizar la ejecución consiste en el uso de registros de eventos o sucesos. Cada registro se conoce como un **Log** y para generarlo, dentro del código se emplea la sentencia: Log_info0("Mensaje"). En la figura 12 se muestra el resultado del uso de **Logs** en el sistema, el tiempo está dado en nano segundos, por lo que se puede conocer con una gran aproximación el tiempo de ejecución de cada suceso.

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

Las llamadas a la función Log_info0 se colocaron en cada uno de los procesos. La **Task** envía dos mensajes, al principio del lazo manda el mensaje “Inicia procesamiento” y después de el cálculo de la salida manda el mensaje “Fin de procesamiento”. Entre los dos hay mensajes intermedios, debido a que la **Task** queda en espera de un semáforo. Puede notarse que no hay orden en el envío de los mensajes porque no es un sistema secuencial.

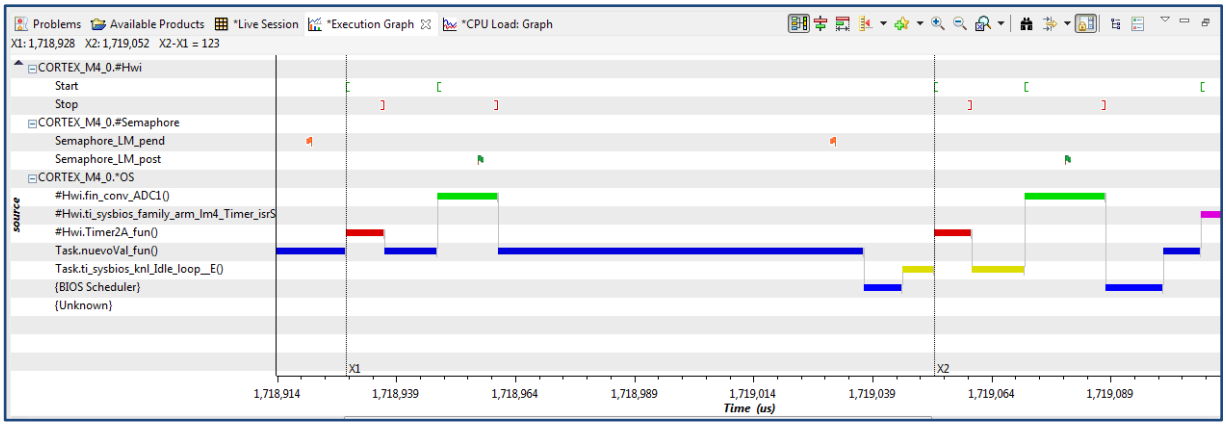


Figura 11. Ejecución gráfica del sistema.

Problems | Available Products | *Live Session | *Execution Graph | *CPU Load: Graph

	Type	Time	Err...	Master	Message	Event	EventClass
1	i	1718466891		CORTEX_M4_0	[../filtro_digital.c:131] Termina conversion	Log_L_info	Info
2	i	1718495350		CORTEX_M4_0	[../filtro_digital.c:147] Inicia el procesamiento	Log_L_info	Info
3	i	1718544483		CORTEX_M4_0	[../filtro_digital.c:161] Comienza envío	Log_L_info	Info
4	i	1718556316		CORTEX_M4_0	[../filtro_digital.c:122] Inicia conversion	Log_L_info	Info
5	i	1718575825		CORTEX_M4_0	[../filtro_digital.c:131] Termina conversion	Log_L_info	Info
6	i	1718595658		CORTEX_M4_0	[../filtro_digital.c:176] Fin del procesamiento	Log_L_info	Info
7	i	1718605016		CORTEX_M4_0	[../filtro_digital.c:147] Inicia el procesamiento	Log_L_info	Info
8	i	1718654158		CORTEX_M4_0	[../filtro_digital.c:161] Comienza envío	Log_L_info	Info
9	i	1718662141		CORTEX_M4_0	[../filtro_digital.c:176] Fin del procesamiento	Log_L_info	Info
10	i	1718684125		CORTEX_M4_0	[../filtro_digital.c:122] Inicia conversion	Log_L_info	Info
11	i	1718703516		CORTEX_M4_0	[../filtro_digital.c:131] Termina conversion	Log_L_info	Info
12	i	1718732783		CORTEX_M4_0	[../filtro_digital.c:147] Inicia el procesamiento	Log_L_info	Info
13	i	1718781925		CORTEX_M4_0	[../filtro_digital.c:161] Comienza envío	Log_L_info	Info
14	i	1718789908		CORTEX_M4_0	[../filtro_digital.c:176] Fin del procesamiento	Log_L_info	Info

Figura 12. Registros para analizar la ejecución del sistema.

Otro aspecto que se puede analizar con la ayuda de las herramientas es la carga de la CPU, la cual se puede ver en la figura 13. La CPU está trabajando en promedio a un 70 % de su capacidad de procesamiento. La línea no se mantiene constante porque durante esta prueba se generaron eventos en la pantalla táctil que requieren atención de la CPU.

Las herramientas de depuración requieren atención de la CPU y esto afecta la respuesta de la zona táctil porque sus eventos se gestionan en el nivel de prioridad más bajo, los otros procesos no se

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

ven afectados porque su prioridad es más alta. Por ello, es conveniente que los registros de eventos (**Logs**) se omitan en la versión final del proyecto.

Para mejorar la respuesta de un sistema, el entorno de CCS maneja dos configuraciones para la implementación de un proyecto (ver figura 14), el modo *debug* y el modo *release*, en el primer modo el entorno incluye código adicional que hace posible la depuración del proyecto y en el segundo este código se omite. La implementación final del sistema se hizo en modo *release* y con ello se favoreció significativamente la respuesta de la zona táctil, las otras tareas no se benefician por tener una mayor prioridad.

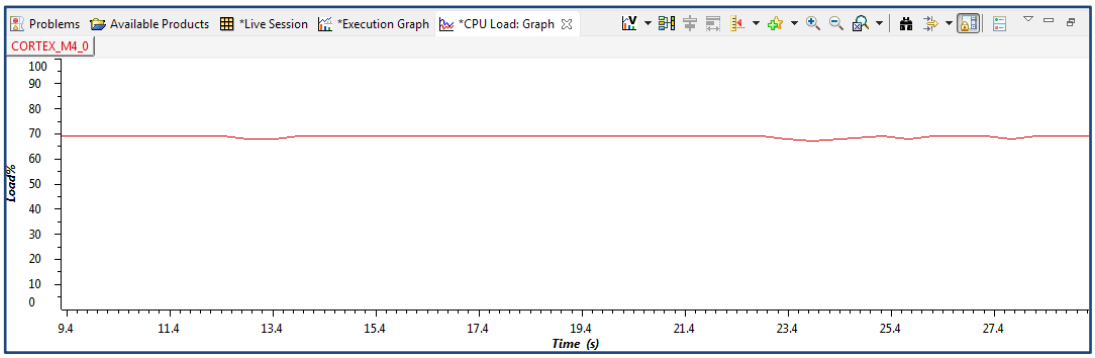


Figura 13. Carga de la CPU.

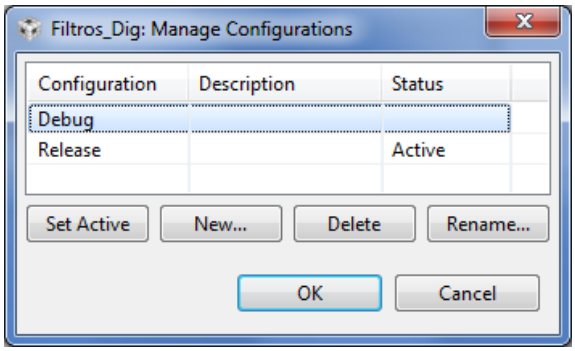


Figura 14. Configuraciones para la implementación de un proyecto en CCS.

Con todas las etapas conectadas se hicieron pruebas con una fuente de audio, los resultados fueron muy favorables porque se distingue claramente el efecto de cada uno de los filtros. Puesto que una señal de audio tiene amplitudes muy variables a lo largo del tiempo, con equipo básico de instrumentación no es posible estimar la frecuencia de corte. Por ello, el sistema también fue evaluado introduciendo una señal sinusoidal de un generador y con un osciloscopio se estimó la frecuencia de corte bajo el criterio de una amplitud de 3 dB por debajo de la frecuencia central (el valor de frecuencia con ganancia máxima), en la figura 15 se muestra la conexión del sistema con equipo de instrumentación y en la tabla 2 se muestran los resultados experimentales obtenidos.

La Mecatrónica en México, Septiembre 2018, Vol. 7, No. 3, páginas 100 – 113.
Disponible en línea en www.mecamex.net/revistas/LMEM
ISSN: 2448-7031, Asociación Mexicana de Mecatrónica A.C

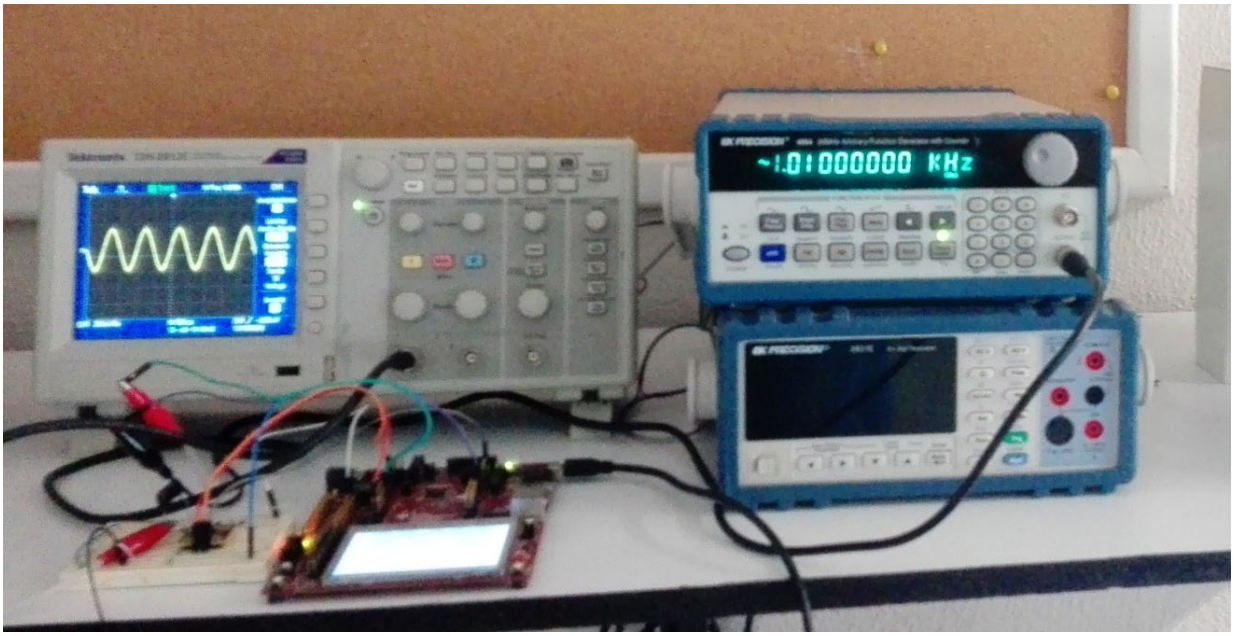


Figura 15. Evaluación del sistema con equipo de instrumentación.

Tabla 2. Caracterización práctica de los filtros digitales implementados.

Filtro	Frecuencia Baja	Frecuencia Central	Frecuencia Alta
1	No Aplica	150 Hz	192 Hz
2	475 Hz	680 Hz	1.3 KHz
3	1.8 KHz	2.2 KHz	No Aplica

3. Conclusiones

El uso del SYS/BIOS facilita el desarrollo de sistemas embebidos que incluyen tareas con diferentes prioridades y sus herramientas de instrumentación permiten validar el cumplimiento de las restricciones de tiempo establecidas, sin embargo, suele ser complicado o confuso analizar un sistema como un conjunto de tareas con prioridades y restricciones de tiempo, significa un cambio de paradigma que hace a un lado el estilo tradicional basado en el lazo infinito combinado con el manejo de interrupciones. Con este trabajo se muestra cómo aplicar este paradigma de diseño basado en un Sistema Operativo de Tiempo Real, el procesamiento de audio sólo es un caso de estudio.

Para este sistema en particular, la biblioteca para el manejo de la pantalla táctil incluye una función denominada *WidgetMessageQueueProcess* que se encarga de gestionar los eventos que ocurren en la pantalla, cada evento es encolado para su atención. La función debe estar en el lazo infinito para un sondeo continuo de posibles eventos. Puesto que no es posible modificar esta función, la solución que permitió incluir el uso de la zona táctil en una aplicación basada en el BIOS, consistió en realizar el llamado de la función con el nivel de prioridad **Idle**, éste es el nivel de menor prioridad por lo que las otras tareas tienen garantizada su ejecución a tiempo. El sistema trabaja adecuadamente, en la figura 12 se puede ver que la carga de la CPU está al rededor del 70 %, en caso de incrementar la cantidad de procesamiento para las otras tareas, se perdería la calidad en la respuesta y actualización de la pantalla táctil.

Con respecto a los filtros, existen diferencias importantes entre los valores propuestos para los cálculos y los valores obtenidos en forma práctica, esto puede deberse al orden de los filtros y a errores de medición. Sin embargo, el trabajo se centró principalmente en la evaluación y aplicación del BIOS en un sistema de tiempo real, por lo que ya no se buscó mejorar este aspecto del sistema.

En general, el comportamiento del sistema fue el esperado, con la evaluación de los resultados a través de las diferentes herramientas, se puede concluir que el uso del BIOS ayuda en la organización de sistemas que incluyan tareas que requieran atención en tiempo real.

Referencias

[1] Texas Instruments (Junio de 2016). TI-RTOS 2.20 User's Guide. Sitio web: http://www.ti.com/lit/ug/spruhd4m/spruhd4m.pdf. [Último acceso: Mayo de 2018].
[2] Texas Instruments (Octubre de 2016). Tiva TM4C129X Development Board User's Guide. http://www.ti.com/lit/ug/spmu360a/spmu360a.pdf. [Último acceso: Mayo de 2018].
[3] Texas Instruments-Production Data (2014). Tiva TM4C129XNCZAD Microcontroller Data Sheet. Sitio web: http://www.ti.com/lit/ds/symlink/tm4c129xnczad.pdf. [Último acceso: Mayo de 2018].
[4] Microchip Technology Inc. (2010). MCP4902/4912/4922 8/10/12-Bit Dual Voltage Output Digital-to-Analog Converter with SPI Interface. Sitio web: http://ww1.microchip.com/downloads/en/DeviceDoc/22250A.pdf. [Último acceso: Mayo de 2018].
[5] Neha S. Bidgar, Balamurugan M.S.; Fast, Intelligent and Secure an Embedded Health-care Supervisory System; International Journal of Reconfigurable and Embedded Systems (IJRES), Vol. 5, No. 2, July 2016, pp. 77~84.
[6] S. Balasubramaniam* and R. Kavithasudha, Sizing of Hybrid Renewable Energy Sources and Battery Systems in Residential Microgrids using Real Time Operating Systems, Journal of Chemical and Pharmaceutical Sciences, JCHPS Special Issue 11: July 2017, pp. 104~106.
[7] Reshma H, Radha R; Solar Powered Wireless Sensor Network For Pipeline Monitoring System, International Journal of Innovative Research and Advanced Studies (IJIRAS), Volume 5 Issue 7, July 2018, pp. 274~282.
[8] Zölzer, U. , Digital Audio Signal Processing, 2nd ed., John Wiley & Sons, England, 2008.
[9] Berger A. S., Embedded Systems Design: An Introduction to Processes, Tools, and Techniques. CMP Books, Lawrence, Kansas, 2002.
[10] TivaWare™ Peripheral Driver Library. User's Guide. Texas Instruments Incorporated. SW-TM4C-DRL-UG-2.1.0.12573. Copyright © 2006-2014.
[11] TivaWare™ Graphics Library. User's Guide. Texas Instruments Incorporated. SW-GRL-UG-1.0. Copyright © 2008-2013.
[12] Kentec Display, http://www.kentecdisplay.com/. [Último acceso: Agosto de 2018].
[13] J. W. Valvano, Embedded Systems: Real-Time Interfacing to ARM ® CortexTM-M Micro-controllers, http://users.ece.utexas.edu/~valvano/, ISBN: 978-1463590154, 2014.