INSTITUTO TECNOLÓGICO DE COSTA RICA

ESCUELA DE INGENIERÍA ELECTRÓNICA

GenAero
AVIONICS ENGINEERING
MANUFACTURING

Informe de Proyecto de Graduación para optar por el título de

Ingeniero en Electrónica con el Grado Académico de Licenciatura

"Control Digital para un Sistema de Video"

General Aerospace Inc.

Estudiantes: Osvaldo Alvarado Bolívar

Juan Pablo Zawadzki Wisniewski

Profesor Asesor: Ing. Carlos Badilla Corrales

Cartago, Semestre 2 - 2003

Resumen

Dentro de la problemática de desarrollo de sistemas para el transporte aéreo, estos deben cumplir con estrictas normas eléctricas y físicas para su aprobación y aprovechamiento. Debido a esta situación, los dispositivos de esta índole por lo general resultan muy costosos con respecto a los dispositivos que se utilizan en otras áreas que no sean la aviación. Esto obligó a crear un sistema que cumpla con todas las normas y regulaciones pertinentes, y a su vez que sea un sistema con una gran flexibilidad y amplias posibilidades de actualización y crecimiento.

Para el desarrollo de este proyecto, la empresa General Aerospace Inc., ubicada en la ciudad de Miami, Florida, y dedicada a la venta de servicios de ingeniería para la aviación, se adentró en el área de la manufactura de controles de sistemas de entretenimiento en aviones privados y comerciales.

Para cumplir con los requerimientos generales de energía y tamaño en aviones, se utilizó un sistema computacional mínimo basado en un microprocesador marca Hitachi de tipo RISC y tecnología CMOS de alta velocidad y baja potencia, que se convierte en el elemento central del sistema de control de dispositivos eléctricos en este medio de transporte. La información generada por el sistema se despliega a través de una pantalla de tipo LCD y el ingreso de datos por parte del usuario se realiza a través de una pantalla de tacto.

El sistema actuador que se implementó es un espacio de direcciones virtuales que permite la comunicación del sistema mínimo con múltiples controladores, tales como televisores, luces y otros a través de un puerto serie en formato RS-232.

Este sistema permite por tanto la adaptación de nuevos dispositivos de control, ya sea de audio, video o funciones electromecánicas a un bajo costo de implementación y desarrollo.

Abstract

Inside the development problems of aerial transportation systems, these must accomplish severe electrical and physical standards for their approval and profit. Due to this situation, these kinds of dispositives generally are very expensive in relation with other dispositives that aren't used in the aviation area. Because of this, the system accomplishes all the necessary standards and regulations, besides doing a device with a big flexibility and larger actualization and growing possibilities.

For the project development, the company General Aerospace Inc., located in the city of Miami, Florida, and dedicated on the engineered aviation services sales, get involved in the area of entertainment control systems manufacture for private and commercial aircrafts.

To accomplish the general requirements of energy and size in the aircrafts, a minimal computing system was used, based on a RISC Hitachi microprocessor and CMOS technology with high speed and low power consumption, that becomes the central element of the control system for electrical dispositives in this transportation medium. The generated system information is displayed through an LCD screen and the user data go in to the system through a touch screen.

The implemented actuating system is a virtual direction space that allows the minimal system communication with multiple controllers, such as televisions, lights and others through a serial port with RS-232 format.

So the system allows the adaptation of new control dispositives for audio, video or electromechanical functions with low cost implementation and development.

Agradecimiento

Agradecemos al Msc. Ing. Carlos Badilla Corrales por su apoyo y sus valiosos consejos.

Agradecemos al Dipl. Ing. Pedro Murillo Fuentes por el interés mostrado para el buen término de este proyecto.

Agradecemos a don Jorge Cunillera Canalías por la ayuda constante.

Agradecemos a nuestras familias por el apoyo brindado.

Tabla de contenido

Índice de figuras

Índice de Tablas

Introducción

Actualmente en los aviones comerciales el sistema de video existente opera de la siguiente manera: una serie de televisores se encienden o apagan en paralelo y se exhibe una película para todos los pasajeros; estos no tienen control sobre el video, no pueden decidir que ver, ni si desean ver o no el programa presentado sobre el monitor.

Existen aviones privados que poseen este tipo de sistemas. Este sistema antes descrito para los pasajeros es molesto; para ellos es un problema no poder desconectar el video cuando no se está mirando o cuando no se desea ver. El proyecto pretende eliminarle esta molestia al cliente ofreciendo la comodidad deseada, el cual consiste básicamente en un control del sistema de video del avión, el cual sería operado por la aeromoza o persona encargada, y que permitiría bajo petición de los pasajeros, apagar o encender el (los) televisor (es) que se deseen.

La solución de este problema se compone de varias etapas, donde se pretende desarrollar el sistema central de control de video. Luego, se va a desarrollar la interfaz para los dispositivos de despliegue y entrada de datos del sistema de control. Además, se pretende realizar otras funciones de control, como manejo de encendido y regulación de intensidad de luces, y manejo de motores paso a paso para ocultar los televisores en la cabina de pasajeros del avión.

Posteriormente, gracias al aporte de este proyecto, se podrá controlar en forma independiente el sistema de video de cada pasajero, además del sistema de audio, luces de lectura, ventanas, hasta llegar a obtener un centro de control general de cada pasajero en el avión, de fácil manejo, dirigido al sector privado y que en un futuro llegará a implementarse en aviones comerciales.

Capítulo 1. Descripción de la empresa.

La empresa para la cual se realizará el proyecto es General Aerospace Inc., cuyo presidente es el Ing. Miguel Flores. Ésta se dedica a dar servicios de ingeniería en aviación tales como diseño, implementación e instalación de nuevos equipos y sistemas en aviones comerciales desde 1990. También ofrecen venta de paquetes de ingeniería para la modificación de sistemas existentes en el avión o nuevos sistemas que se deseen agregar en los aviones, los cuales consisten en el diagrama de cableado, diagramas para la modificación estructural, instrucciones de ingeniería y componentes y demás piezas necesarias para la instalación del sistema dentro de los aviones comerciales. Adicionalmente, ofrecen el servicio de instalación y la venta de soporte técnico. Asimismo se ofrece el servicio de actualización de manuales, que consiste en la reelaboración de los manuales de partes, de mantenimiento, de alambrado y el de cargas eléctricas.

General Aerospace también diseña sistemas de entretenimiento de audio y video disponibles en los aviones comerciales. Los servicios ofrecidos en esta área son el diseño de equipos, aprobación por parte de la autoridad reguladora de aviación de los Estados Unidos denominada FAA (Federal Aviation Administration) por sus siglas en inglés, manual del nuevo equipo, instrucciones de ingeniería para su instalación de acuerdo a las necesidades de cada cliente en específico y el modelo de avión, además de la instalación del mismo.

Cabe destacar que cuando un nuevo equipo es diseñado o uno existente es modificado (modificación interna en la circuitería del equipo) por parte de la empresa, éste se ve sometido a estudios de radiación electromagnética, cumplimiento de los estándares de aviación, y el efecto que el mismo tendrá sobre la navegabilidad de la aeronave.

Dentro de los productos que ofrece se encuentran todos los repuestos relacionados con su campo de trabajo.

General Aerospace está dirigido al mercado aeronáutico comercial, que ofrece sus servicios a aerolíneas a nivel mundial tales como Copa, TACA, Aerocontinente, Lanchile, Pluna, Rockwell Collins, entre otras.

El proyecto estará a cargo del departamento de control de calidad e ingeniería, dedicado al desarrollo de ingeniería en sistemas y equipos de audio y video en aviones comerciales. Dentro de este departamento la persona que coordina y asesora el proyecto es el Ing. Emilio Brenes Pereira, ingeniero en Electrónica egresado del ITCR, y con más de un año de laborar para esta empresa.

Este proyecto será el primero de un plan piloto, que tiene como objetivo abrir una sucursal de la empresa en Costa Rica dedicada al diseño e implementación de sistemas de entretenimiento para aviones. Este proyecto será una prueba para estudiar la capacidad de diseño en ingeniería de los profesionales en Costa Rica.

Capítulo 2. Definición del problema

2.1. Descripción del problema

En ciertos aviones comerciales y privados, se cuenta con una serie de televisores que se accionan simultáneamente, y en ellos se exhibe la misma película. El pasajero no cuenta con la posibilidad de tener un sistema de control que le permita ver o no la cinta que se observa en estos televisores. Lo que actualmente poseen la mayoría de los aviones, es un sistema en el cual el pasajero desde su silla puede seleccionar tres canales de audio que son: el audio de la cinta en inglés, el audio de la cinta en español, o puede escuchar un canal de música pregrabada; adicionalmente se tiene control sobre el nivel de volumen deseado. El sistema actual opera analógicamente.

La empresa para la que se realiza el proyecto desea incursionar más fuertemente en el campo del entretenimiento aéreo para aumentar sus mercados y expandirse como compañía especializada en sistemas de aviación. Para esto se intenta cambiar el antiguo sistema de control de video analógico por un sistema de control digital para el video, más flexible y que tenga la habilidad de crecimiento, con buena aceptación en el mercado meta.

2.2. Importancia del problema

Para la empresa es importante realizar este proyecto porque de esta manera está resolviendo la necesidad de un conjunto de clientes, desarrollando un producto nuevo en un gran mercado.

Además, pretende ampliar a futuro este proyecto del sistema básico de control de encendido y apagado de video en aviones, a diferentes etapas no solo de control de video, sino de dispositivos de audio, u otro tipo de dispositivos para la comodidad del pasajero.

Como se comentó en el apartado anterior, la empresa requiere de un sistema que además de cumplir con los objetivos planteados por los estudiantes, cumpla con las expectativas que esta tiene acerca del uso y beneficio que este pueda traerles.

Otro de los requerimientos que exige la empresa, es el desarrollo de un prototipo que cumpla con las normas mínimas impuestas para un control de dispositivos electrónicos en un transporte aéreo, para desarrollar con base en este, los restantes sistemas de control para su venta. Esto implica desarrollar un sistema que se pueda presentar como un producto de mercado para su futura comercialización, y que ofrezca ventajas competitivas sobre otros sistemas existentes en el mercado.

Es por estas razones que es importante para la empresa tener un sistema de control que permita manejar diferentes tareas, y no solamente dedicarse al control de televisores dentro de la cabina de pasajeros del avión, sino de poder controlar por medio de este sistema un mayor número de funciones de entretenimiento y comodidad para los pasajeros, como lo es el control de luces, el control de volumen de la señal de audio del programa o película que se presenta en el televisor, el despliegue de música, o bien manejar el sistema que permite ocultar los televisores dentro de un compartimiento en cabina.

2.3. Antecedentes del proyecto

En la industria aérea, debido al alto costo que tiene la construcción de un avión, estos se hacen pensando en que su vida útil sea larga, por este motivo los sistemas requeridos (audio y video, entre otros) son obsoletos. Como se mencionó anteriormente, el sistema de control de video de un gran número de aviones funciona de forma análoga y en paralelo, por lo que no tiene la flexibilidad que algunos clientes desean.

El desarrollo de este proyecto es algo novedoso y no existe un sistema parecido implementado en la mayoría de los aviones comerciales o privados. Para la empresa representa el inicio de un gran proyecto de gran beneficio para ellos y sus clientes, los cuales serán los usuarios del sistema. Por lo tanto, no existen soluciones ensayadas anteriormente por la empresa.

Por otro lado, este proyecto de graduación se convierte en la segunda parte de la primera etapa planteada por la empresa, para la solución del control general del sistema de entretenimiento. La primera parte del proyecto fue desarrollada para el Laboratorio de Estructuras de Microprocesadores durante el primer semestre del 2002, cuando se implemento el proyecto titulado " Control digital para un sistema de video" ,el cual fue asesorada por el Ing. Carlos Badilla Corrales, la cual consistió en la investigación y diseño del sistema de control de video, así como establecer los parámetros para la escogencia de los componentes de hardware y de software, desarrollando en esta parte las rutinas de inicialización del sistema de bajo nivel.

Capítulo 3. Marco Teórico

3.1. Descripción del hardware utilizado

Ya que se requiere una interfaz de alto nivel amigable al usuario, esto implica el desarrollo de un sistema computacional básico que pueda soportar el manejo de un sistema operativo, que permita el manejo de las funciones básicas del sistema por medio de software.

El hardware de un sistema computacional básico se denomina sistema mínimo y está compuesto principalmente por una unidad central de proceso (CPU), memoria de trabajo volátil, memoria de trabajo no volátil, memoria de almacenamiento secundario, y dispositivos de interfaz con el usuario.

3.1.1 Unidad Central de Procesos (CPU)

Esta se encarga de procesar la información y convertir las instrucciones en acciones deseadas por el usuario. Por lo general esta unidad es implementada dentro de un microprocesador semiconductor. En los últimos tiempos se han fabricado algunos procesadores que tienen un conjunto de instrucciones reducido en comparación con los procesadores Intel, y se han utilizado en sistemas portátiles como celulares, PDA, pocket PC, etc. Los cuales tienen como principales características utilizar tecnología CMOS de alta velocidad por lo cual tienen menos requerimientos de energía y son ideales para aparatos como éstos que utilizan batería.

Para este proyecto se utiliza un microprocesador marca Hitachi, modelo SH-7727. Este microprocesador cuenta con las siguientes características:

a. Pertenece a la familia de microcomputadores SuperH RISC de Hitachi.
b. El chip posee 240 pines.

c. Posee un bus de datos de 16 / 32 bits seleccionables y bus de direcciones de 32 bits.

d. Opera a una frecuencia de 160 MHz.

e. Tienen integradas las siguientes funciones periféricas: interface, y función de puerto serie universal (USB) permitiendo una velocidad máxima de transferencia de datos de 12 Mbps, controlador de LCD a color, convertidor analógico – digital de 6 canales, convertidor digital – analógico de 2 canales, controlador de PCMCIA para comunicación por infrarrojo, reloj de tiempo real, controlador de DMA de 4 canales, MMU (unidad de manejo de memoria) y un IrDA (asociador de datos por infrarrojo).

f. Tiene un CPU SH3-DSP, con una memoria interna de 16Kbytes para el manejo del controlador de DMA y para procesamiento de señales digitales.

g. Tiene una memoria caché de 16Kbytes.

h. Posee una interface serie de 3 canales, una análoga, una interface CODEC para señales de audio y una interface para conexión de memoria externa (de tipo SRAM, DRAM sincrónica, ROM, PCMCIA, y otros).

i. La alimentación externa es 3.0V a 3.6V.

3.1.2 Memoria de trabajo volátil

En esta residen los programas e información que el usuario desea procesar; este tipo de memoria es la más importante a nivel de procesos. Por lo general es implementada por arreglos de celdas básicas semiconductoras conocidas como memoria de acceso aleatorio (RAM); dependiendo de la tecnología se pueden tener: SRAM, DRAM, SDRAM, etc.

En este caso se hace uso de una memoria de lectura aleatoria de la marca Hyundai modelo HY57V281620AT-K, de tipo RAM dinámica sincrónica (SDRAM). Esta tiene una capacidad total de 32 Mbytes, con un tipo de empaquetado TSSOP, con un ancho de bus de datos de 32 bits. Utiliza una base de 54 pines, con una alimentación de 3.3V.

3.1.3 Memoria de trabajo no volátil

En esta memoria reside la información para la inicialización de los componentes de hardware y rutinas de servicio de interrupciones y carga del sistema operativo. Esta se implementa por medio de memorias de solo lectura (ROM), de compuerta flotante (EPROM) o las borrables eléctricamente (EEPROM, Flash ROM).

Para trabajar con el microprocesador Hitachi, se utiliza una memoria de solo lectura de la marca Intel, modelo E28F640, de tipo StrataFlash (FlashROM). Esta memoria cuenta con una capacidad de almacenamiento de 32 Mbytes y con un tipo de empaquetado de montaje superficial. Esta memoria tiene 56 pines y se alimenta con una fuente de 3.3V.

3.1.4 Dispositivos de interfaz con el usuario

Estos se encargan de desplegar e interpretar las tareas deseadas por el usuario. Dentro de estos subsistemas se tienen los de entrada de información tales como el teclado, el mouse, touchpanel, etc; y los de salida o despliegue de información tales como monitor, parlantes, impresora, etc.

El despliegue de información se realiza a través de una pantalla o monitor marca SHARP, modelo LQ10D-368. Este monitor tiene un tamaño de 10.4” (26 cm) del tipo LCD (pantalla de cristal liquido), con tecnología TFT (transistor de película delgada) o de matriz activa, el cual cumple con las especificaciones de la FAA (administración federal de aviación, según sus siglas en inglés) y que funciona en el formato estándar VGA.

Esta tiene una entrada del tipo RGB con compatibilidad VGA, que funciona con niveles de alimentación TTL (0 y 5 voltios) y bajos voltajes de entrada (-0.3 a 0.3 voltios). Utiliza un conector DF9BA-31P-1V, de la marca Hirose Electric Co. Ltd., el cual es un conector de 31 pines, en conexión de superficie.

Por otra parte, para el ingreso de información se utilizó una pantalla de tacto (o touch panel) marca Hampshire, modelo TSHARC, de 4 cables. Esta pantalla es del tipo análoga resistiva, compuesta por dos capas o componentes: la primera capa es de Plata, que es de baja resistencia y permite la conexión de esta pantalla con dispositivos electrónicos. La segunda capa es de Óxido de Indio (ITO), la cual es un conductor transparente. Las pantallas de tipo resistivo se basan en la decodificación de señales de voltaje. Para manejar las líneas de control de esta pantalla, se utiliza un controlador de pantalla de tacto o touch inverter.

3.1.5 Microcontroladores secundarios

Estos dispositivos tienen como función recibir la información por parte de la unidad central de procesos (CPU) y realizar las operaciones necesarias para manejar los dispositivos externos de entretenimiento. Entre las ventajas de este tipo de controladores se encuentran:

- Programación en lenguaje C y ensamblador.
- Integración de funciones periféricas, como relojes de tiempo real (RTC), acceso directo a memoria (DMA), control de interrupciones (INTC), manipulación directa de los pines de entrada/salida y otros.

Estos controladores son fabricados por empresas como Motorola, Texas Instrument, Microchip, entre otros.

Para este proyecto se utilizaron los controladores de la familia PIC16F87X de Microchip.

Dentro de las principales características de la CPU del microcontrolador son:

- Arquitectura tipo RISC.

- Frecuencia de operación máxima de 20MHz.
- Basado en tecnología Flash CMOS de alta velocidad y bajo consumo de potencia.
- Programación por medio de un lenguaje basado en C.

Dentro de las características perifericas se tienen:

- 3 timer para la sincronización de procesos y programas.
- Comunicación Serie sincronica y asincronica.
- Interrupción externa para eventos asincronicos.

3.2. Descripción del software utilizado

Un sistema mínimo como el planteado anteriormente debe utilizar un sistema operativo que permita dos funciones muy importantes, la primera es restringir a los usuarios el uso del hardware especializado; así como la administración de procesos para dar soporte a los procesos de multitarea. El sistema operativo utilizado en la plataforma LSH7727 es el Windows CE.net. Para la programación de una aplicación se utiliza el Visual C++, el cual se comunica con el sistema operativo para la sincronización de eventos y administración del hardware. En el siguiente apartado se hace una breve referencia a las herramientas de desarrollo y a la plataforma que le da soporte(WinCE)

3.2.1 Herramientas de diseño

Para la realización de los programas se utiliza el Microsoft Embedded Visual Tools 3.0, el cual es un paquete de dos programas, que incluye el Embedded Visual C++ y el Embedded Visual Basic. Además contiene un paquete de instalación para compilar y transferir directamente los programas elaborados en alguno de estos dos lenguajes a la plataforma de desarrollo, llamado WindowsCE LSH7727-10 SDK.

En la siguiente figura se muestra una imagen de la interfaz de programación del software de programación utilizado.

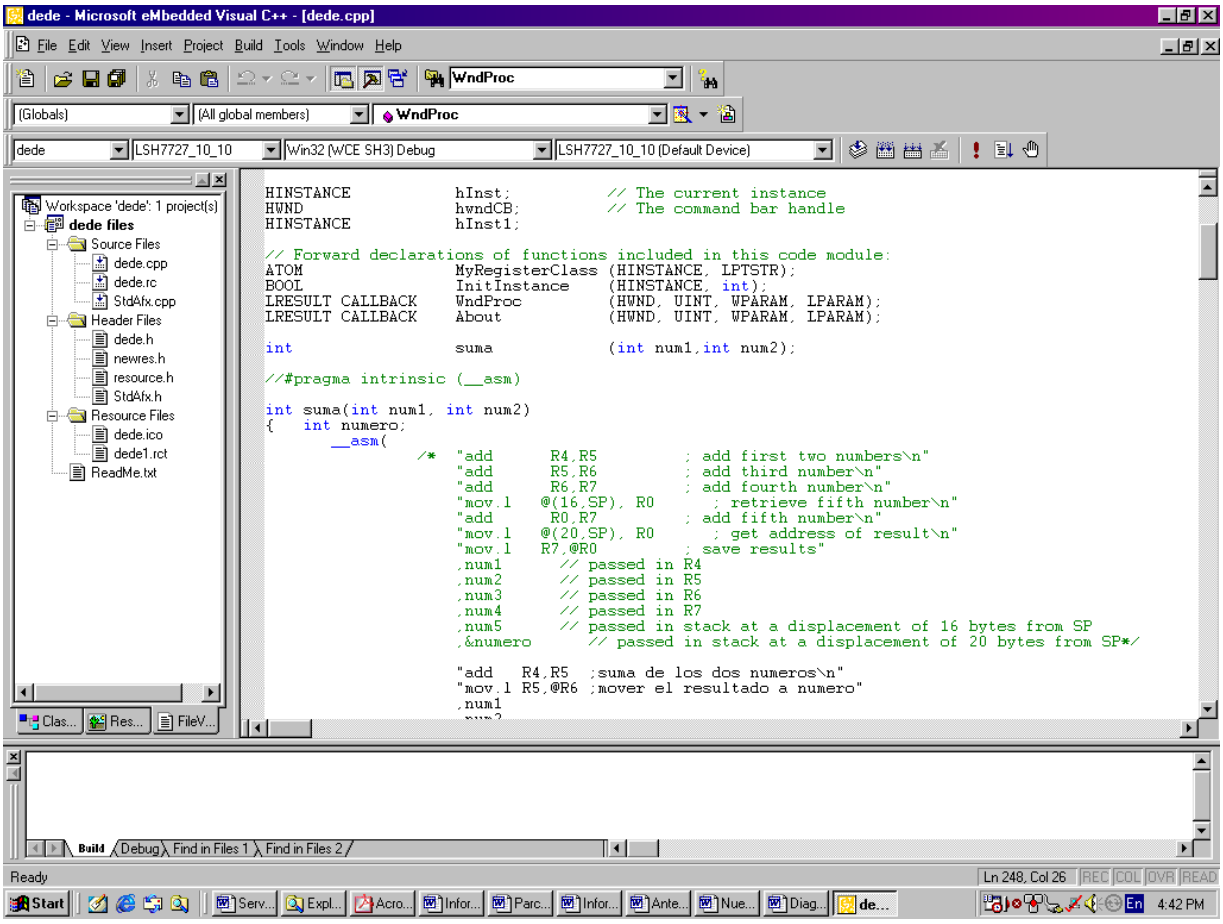

Figura 3.1 Interfaz del software de programación Microsoft Embedded C++ 3.0

Por otra parte, para realizar la programación de los dispositivos de control externos, se utiliza el software de programación PIC C Compiler, haciendo uso además del software MPLab para la programación de los microcontroladores.

3.2.2. Fundamentos del sistema operativo Windows CE

Para la programación de las funciones especiales que se desean implementar en la plataforma se debe tener un conocimiento básico del sistema operativo en el cual se esta trabajando, ya que este debe hacer la comunicación entre los diferentes eventos y la arquitectura a nivel de software. El siguiente es un resumen de los conceptos básicos que el Windows CE comprende.

3.2.2.1 Servicios del Kernel de Windows CE

El Kernel de WinCE es el corazón del sistema operativo, el cual da funcionalidad a los diferentes dispositivos periféricos. Las funciones de este se refieren al procesamiento de procesos, hilos y el manejador de memoria. Dentro de estos procesos y funciones se tiene: la creación, terminación, y sincronización de los procesos y los hilos del sistema.

Un proceso el cual representa una simple instancia de las aplicaciones que están corriendo y activa una o varias aplicaciones.

Los hilos activan una tarea de una aplicación y mantienen solo una aplicación funcionando una a la vez. Los niveles de prioridad de los hilos, el manejador del inversor de prioridad, el soporte de las interrupciones, y la administración de tiempo y esquematización se encuentran en la arquitectura del Kernel.

Trabajo con proceso e hilos.

Todas las aplicaciones que están en WinCE están conformadas por procesos y en uno o más hilos. Como se mencionó anteriormente un proceso es una simple instancia de una aplicación que esta corriendo. Un hilo es la unidad básica con la cual el sistema operativo asigna tiempo de procesamiento. Un hilo puede ejecutar cualquier parte del código de un proceso, incluyendo parte del que esta siendo ejecutado por otro hilo.

Procesamiento

WinCE puede correr 32 procesos en un mismo instante. Cada proceso inicia con la llamada a un simple hilo (llamado regularmente hilo primario), la cual da los parámetros adecuados para que pueda correr una aplicación. En WinCE crea un hilo primario cuando el cargador llama a la función WinMain. El proceso puede abrir un número de hilos adicionales los cuales solo son limitados por la cantidad de memoria RAM del sistema, el Kernel puede direccionar hasta 512Mbyte de memoria física.

Cuando el sistema operativo se inicializa, crea un espacio de memoria virtual simple de 4Gbyte, el cual es divido en 33 slot, de 32Mbyte. El espacio de memoria es compartido por todos los procesos. Cuando inicia un proceso el OS (sistema operativo) busca un slot[1] vacío en el espacio de memoria. El slot 0 se reserva para los procesos que están corriendo.

Además el OS crea una pila para cada hilo y un encabezado para el proceso. El máximo número de hilos depende de la cantidad de memoria en el sistema. Se puede asignar memoria arriba de los 32MB para cada slot, por medio de la función VirtualAlloc.

Cuando un proceso se inicializa, el OS guarda la pila, el encabezado, la sección de datos y las librerías DLL en el slot asignado al proceso (DLL, Stack, head, .exe).

[1] Slot: Unidad de medición de memoria virtual que corresponde a 32Mbyte

3.2.2.2 Desarrollo de aplicación en tiempo real.

Una aplicación de tiempo real es un proceso en el cual el tiempo es una variable crítica, tal como telecomunicaciones, manufactura de productos, dispositivos de adquisición de datos, etc.

Características de WinCE para este tipo de aplicación:

- Los niveles prioritarios de hilos son mayores.
- Manejo simplificado de la inversión de prioridad de los hilos
- Soporte total de interrupciones anidadas
- Control de la esquematización del tiempo

Niveles de Prioridad

Un proceso está compuesto por uno o varios hilos. Cada hilo representa una parte independiente del proceso. WinCE tiene 248 niveles de prioridad para 255 prioridades de hilos. Las prioridades se pueden definir a través de la función CeSetThreadPriority, se puede saber el número de prioridad de un hilo por medio de una función CeGetThreadPriority. La mayor prioridad esta dada por el cero y la menor prioridad esta dada por el 255.

Inversión de prioridad

La inversión de prioridad se produce cuando un hilo de más alta prioridad es bloqueado cuando está corriendo por un proceso de menor prioridad propio de un objeto del Kernel, tal como un mutex o una región critica que un hilo de más alta prioridad necesita.

Uso de las interrupciones anidadas

Las aplicaciones de tiempo real utilizan interrupciones que responden a eventos externos. WinCE utiliza dos factores para compartir el procesamiento de las interrupciones en dos pasos: una rutina de servicio de interrupciones (ISR), y un hilo para el servicio de interrupciones.

El OS asigna cada IRQ con un ISR, cuando las interrupciones son activadas y una interrupción ocurre, el Kernel llama a la ISR registrada para la interrupción. Cuando la ISR está corriendo, el Kernel desactiva todas las otras IRQ. Cuando el servicio de interrupción termina retorna un identificador de la interrupción e inicializa los eventos asociados con el evento. Cuando el Kernel inicializa los eventos, el ISR comienza el procesamiento.

Para prevenir la pérdida y retraso de las interrupciones de más alta prioridad se utilizan las interrupciones anidadas. La siguiente lista muestra como el OS maneja el anidamiento de llamadas a las rutinas de servicio de interrupción:

- El Kernel desactiva todas las otras IRQ del mismo y menor nivel de prioridad así la IRQ invoca la ISR.
- Si una IRQ de más alta prioridad se activa antes que el proceso sea completado, el Kernel guarda el estado del ISR que está corriendo.
- El Kernel llama al administrador de la nueva petición de interrupción.
- Cuando es completado, el Kernel carga el anterior ISR y continúa el procesamiento.

El WinCE puede anidar muchos servicios de interrupciones como las soportadas por hardware.

Manejo del tiempo y planeación

WinCE tiene herramientas con cualidades poderosas que dan el control sobre el tiempo de planeación de un hilo y la cantidad de datos que los hilos deben cargar. Además que aumentan la eficiencia en tiempo y activación de los hilos.

Composición del quantum de un hilo

El quantum de un hilo es definido como la unidad de tiempo mínima que un proceso es ejecutado, este es definido en 100ms, y es iniciado por el OEM en el OAL, este valor es constante en todo el sistema operativo.

En WinCe se activa una aplicación iniciando el quantum de un hilo por el hilo fundamental. Esto significa que se puede tomar el esquema de las necesidades de la aplicación. Para ajustar el tiempo del Quantum se cuenta con dos funciones: CeGetThreadQuantum, lo activa para recibir el quantum de un hilo específico; CeSetThreadQuantum le debe inicializar el quantum para un hilo.

Temporización de un hilo

La función GetThreadTimes obtiene la información de la temporización de un hilo, o sea, el tiempo que ha ejecutado en modo Kernel y en el modo usuario, este dato no incluye el tiempo gastado ejecutando hilos del sistema o esperando en un estado suspendido o bloqueado. Si el hilo ya terminó la función retorna el tiempo de salida del hilo.

3.2.2.3 Procesos

Creación de un proceso

Para empezar un proceso dentro de otro proceso, se llama a la función CreateProcess, el cual carga una nueva aplicación dentro de la memoria y crea un nuevo proceso con un nuevo hilo.

El prototipo de la función es:

BOOL *CreateProcess*(**LPCTSTR** lpApplicationName,

LPTSTR lpCommandLine,

LPSECURITY_ATTRIBUTES lpProcessAttributes,

LPSECURITY_ATTRIBUTES lpThreadAttributes,

BOOL bInheritHandles,

DWORD dwCreationFlags,

LPVOID lpEnvironment,

LPCTSTR lpCurrentDirectory,

LPSTARTUPINFO lpStartupInfo,

LPPROCESS_INFORMATION lpProcessInformation);

Ya que WinCE no soporta muchas de las funciones de seguridad o de directorios corriendo y no maneja la herencia, la mayor parte de los parámetros son puesto en cero o en NULL. Lo que produce la siguiente función prototipo:

BOOL CreateProcess(**LPCTSTR** lpApplicationName,

LPTSTR lpCommandLine, **NULL, NULL, FALSE,**

DWORD dwCreationFlags, **NULL, NULL, NULL,**

LPPROCESS_INFORMATION lpProcessInformation);

lpApplicationName contiene un puntero al nombre donde empieza la aplicación.

DwCreationFlags: este parámetro especifica el estado inicial en el cual inicia el proceso después de cargarse.

lpProcessInformation: este parámetro apunta a la estructura **PROCESS_INFORMATION** la cual contiene los datos del nuevo proceso, este parámetro puede estar en NULL.

Si el proceso no puede correr, la función **CreateProcess** retorna un valor de falso, el cual se puede recuperar con la función **GetLastError.**

Terminación de un proceso

La forma más común de terminar un proceso es tener que retornar a la función **WinMain.** Otra forma de terminarlos es por medio de la llamada a la función **ExitThread** del hilo primario, ya que WinCE termina automáticamente los procesos al terminar el hilo primario. Se puede determinar el código de salida llamando a la función **GetExitCodeProcess.** Especificando el manejador del proceso, el cual se puede obtener por la llamada a la función de creación del proceso o por medio de la función **OpenProcess.**

Algunas de las formas más comunes de terminar un proceso son:

- Si el proceso tiene un mensaje en la cola, manda un mensaje WM_CLOSE a la ventana principal de la ventana.
- Usando la función **TerminateProcess**, la cual no comunica a las librerías dinámicas que el proceso está terminado.

3.2.2.4 Hilos

Un hilo describe la ruta de ejecución dentro de un proceso. Los OS crean nuevos procesos e hilos en todo momento. El propósito de crear hilos es aprovechar el tiempo del CPU al máximo. Por ejemplo, en muchas aplicaciones, se crea un hilo separado que se dedica a las tareas de impresión mientras que el usuario continúa ejecutando otras tareas.

Cada hilo comparte todos los recursos del proceso, incluyendo el espacio de direcciones. Además, cada hilo tiene una pila, donde el linker inicializa todos los hilos creados en un proceso.

Por otro lado, un hilo contiene el estado de los registros del CPU, por lo cual se dice que conoce su contexto, y las entradas en la lista de ejecución del sistema de planeación. Se puede usar la función GetThreadContext para obtener el contexto de un hilo especifico. La función SetThreadContext inicializa el contexto de un hilo.

Cada hilo en un proceso opera de forma independiente. Los hilos que comparten los recursos comunes deben de coordinarse por medio de métodos de sincronización.

Una aplicación comienza cuando el sistema de planificado obtiene el control de ejecución del hilo. El sistema de planificador determina cual de los hilos debe correr y cuando éste debe correr. Los hilos de más baja prioridad deben esperar a que los hilos de más alta prioridad terminen.

Los hilos pueden estar en cualquiera de los siguientes estados: corriendo, suspendidos, durmiendo, bloqueados o terminados. Cuando todos los hilos están bloqueados, WinCE entra en el modo IDLE, el cual detiene la ejecución de instrucciones y consumo de poder en el CPU.

Creación y Terminación de un hilo

Para crear un hilo se llama a la función **CreateThread.** El prototipo de la función se presenta a continuación:

> **HANDLE** CreateThread(**LPSECURITY_ATTRIBUTES** lpThreadAttributes,
>
> **DWORD** dwStackSize, **LPTHREAD_START_ROUTINE** lpStartAddress,
>
> **LPVOID** lpParameter, **DWORD** dwCreationFlags, **LPDWORD** lpThreadId);

Ya que WinCE no soporta los parámetros *lpThreadAttributes* y *dwStackSize*, estos se deben poner en cero o NULL. Si la función termina exitosamente, ésta retorna el manejador y el identificador del nuevo hilo. Además se puede recuperar el identificador del hilo por medio de la llamada a la función **GetCurrentThreadId.**

En WinCE esta función retorna el agente actual del hilo. Se puede además obtener el manejador del hilo por medio de la función **GetCurrentThread.** El manejador del hilo retorna un seudohandle para el hilo, que es válido mientras el hilo permanezca activo.

Se puede terminar un hilo llamando a la función ExitThread, el cual libera los recursos que fueron usados por el hilo. Al aplicar esta función a un hilo primario se finaliza la aplicación ligada a este hilo.

Planificación de un hilo

La planificación en WinCE está basada en el algoritmo time-slice, que planea la ejecución de los hilos. Ya que WinCE no soporta las clases de prioridad, los procesos en los cuales corren hilos no afectan la prioridad de los hilos. Todas las prioridades pueden ser usadas por el mismo proceso, por lo cual un proceso puede tener 256 prioridades. Las 249 prioridades superiores son niveles de prioridad para los procesos de tiempo real.

Los hilos de mayor nivel de prioridad corren primero. Los hilos con la misma prioridad corren en modo de round-robin cuando un hilo ha sido detenido, todos los demás corren antes de que el hilo original pueda continuar. Un hilo de más baja prioridad puede correr hasta que los hilos de más alta prioridad hayan terminado o este haya sido bloqueado. Si un hilo está corriendo y otro de más alta prioridad se desbloquea, el hilo de más baja prioridad es inmediatamente suspendido.

Los hilos corren por un intervalo de tiempo específico-llamado quantum- el cual tiene un valor de 100ms. Un OEM puede especificar un valor de quantum diferente. Un hilo con un quantum-set en cero no puede ser apropiado excepto por un hilo de más alta prioridad o por una rutina de servicio de interrupción.

En la mayor parte la prioridad de los hilos es fija y no puede ser cambiado. Sin embargo, la inversión de prioridad vista anteriormente es una excepción a esta regla. Si un hilo de más baja prioridad está usando un recurso que un hilo de más alta prioridad está esperando, el Kernel levanta la prioridad del primer proceso hasta que este libere el recurso que el otro hilo necesita.

Para obtener la prioridad de un hilo se puede llamar a la función **CEGetThreadPriority.** Para cambiar la prioridad de un proceso se llama a la función **CESetThreadPriority.**

Suspensión de un hilo

Para suspender un hilo se utiliza la función **SuspendThread.** Se tienen además algunas funciones relacionadas como **ResumeThread, Sleep, GetThreadTime.**

Cuando se suspende un hilo por más de una vez, se pueden hacer algunos juegos de llamadas múltiples a las funciones de SuspendThread con el mismo número de llamadas a la función ResumeThread.

Uso del almacenamiento local de hilos

El almacenamiento local de hilos (TLS), es un método por el cual, cada hilo de un proceso multihilo busca una localidad en la cual guardar los datos de un hilo específico. Estas son situaciones muy importantes, en las cuales, se busca un hilo con acceso único a un dato. Este tipo de funciones solo pueden ser llamadas por una librería DLL o un proceso y no por un hilo.

3.2.2.5 Sincronización de procesos e hilos

En un OS es necesario tener hilos que corran en forma concurrente. Para esto es muy importante la sincronización de las actividades de varios hilos. En WinCE existen objetos de sincronización que permiten la sincronización de las acciones de los hilos con respecto a otros. Este objeto incluye: secciones críticas, mutexs, eventos y semáforos. Adicionalmente se pueden usar las funciones de interbloqueo.

A pesar de los métodos de sincronización usados, un hilo se sincroniza al mismo, con respecto a los demás por medio de un objeto de sincronización y por medio de los estados de espera. Los objetos de sincronización llaman al sistema operativo, el cual, produce un evento antes de que el hilo se pueda ejecutar. Cuando un evento ocurre, el hilo es de nuevo elegido para ser planeado y para que obtenga tiempo de la CPU. Cuando es seleccionado el hilo continua la ejecución. El hilo ha sido sincronizado por medio de la ejecución de un evento.

Objetos de sección crítica

Cuando varios hilos tienen el acceso compartido a los mismos datos pueden interferir unos con otros. Un objeto de sección crítica protege una sección de código que accesa datos compartidos. Sin embargo, este objeto solo puede ser usado por un proceso o una librería dinámica (DLL), y no puede ser compartido por otro.

Las secciones críticas trabajan por medio de la llamada a las funciones **EnterCriticalSection** o **TryEntercriticalSection,** las cuales indican, que un hilo ha entrado a código de sección critica. Si otro hilo llama a estas funciones y referencia al mismo objeto de sección crítica, éste último es bloqueado, hasta que el primer hilo llame a la función **LeaveCriticalSection.** Este objeto puede proteger varias secciones de código. Para liberar los recursos utilizados en una sección crítica se utiliza la función **DeleteCriticalSection.**

Para poder usar la sección crítica, se debe primero declarar una estructura **CRITICAL_SECTION**, ya que otros hilos requieren punteros a esta estructura. Para crear un manejador para este objeto, se llama la función **InitializeCriticalSection.**

Objeto Mutex

Un objeto mutex es un objeto el cual indica cuando la señal es propia o no es propia. Este es utilizado para coordinar acciones de exclusión mutua, en procesos con datos o recursos compartidos. Sólo un hilo a la vez puede tener acceso al mutex en un instante.

El hilo llama a la función **CreateMutex** para crear un objeto de este tipo. El hilo que crea el mutex obtiene una respuesta inmediata especificando el nombre del objeto. Los hilos en otros procesos pueden abrir un manejador para un mutex existente, por medio de la invocación a esta función con el nombre del mutex. Si el mutex existe entonces la función **GetLastError** retorna el valor ERROR_ALREADY-EXISTS.

Cualquier hilo con un manejador a un mutex puede usar las funciones de espera para solicitar las propiedades de un objeto mutex. Si el mutex está siendo utilizado por otro hilo, la función de espera bloquea la solicitud hasta que el hilo libere los recursos que utiliza por medio de la función **ReleaseMutex.**

Objeto de eventos

El OS utiliza el objeto evento para notificar a un hilo para indicar que una tarea o un evento particular ha ocurrido. Para crear un evento se llama la función **CreateEvent.** A continuación se muestra el prototipo de la función:

HANDLE CreateEvent(NULL, **BOOL** bManualReset, **BOOL** bInitialState, **LPTSTR** lpname);

lpname se usa para poner el nombre al evento o dejarlo sin él.

BManualReset activa o desactiva automáticamente el estado, o sea, que puede ser activado manualmente.

BInitialState especifica si el evento es creado en un estado que emite señales o no las emite.

Un evento puede ser compartido por otros procesos. Los hilos de otros procesos pueden abrir un handle a un evento existente especificando el nombre en la función CreateEvent. Todos los elementos sincronizadores son guardados en la misma pila. Para determinar si un evento llama a un objeto ya existente o crea uno nuevo se llama a la función **GetLastError** inmediatamente después de la función de creación.

Para activar la espera de un evento se usa la función **SetEvent,** esta función no resetea automáticamente el evento a un estado inactivo. La función **PulseEvent** activa el evento y lo inicializa. Para eventos manuales, la función PulseEvent desbloquea todos los hilos que esperan el evento. Para los eventos automáticos, esta función solo desbloquea un hilo.

Un evento puede resetearse por sí mismo, sólo se necesita el uso de la función SetEvent para activarlo. El evento es automáticamente llevado a un estado de inactividad cuando un hilo es desbloqueado después de esperarlo. El reseteo manual se utiliza la función **ResetEvent.**

Un hilo simple puede especificar diferentes eventos en varias operaciones simultáneas.

Objeto Semáforo

Un semáforo es un objeto para la sincronización de interprocesos que mantiene una cuenta entre cero y el máximo valor. El estado del objeto se activa cuando el valor del semáforo es mayor a cero, y está desactivado cuando el valor del semáforo es cero. Los semáforos limitan el uso de los recursos por medio de un conteo de las entradas y salidas de los hilos. Cada vez que un hilo en estado de espera es liberado, en el estado de activación del semáforo la cuenta decrementa en uno.

Se utiliza la función **CreateSemaphore** para crear un semáforo con nombre o sin él. El handle retornado por esta función puede ser usada en cualquier función que requiera un handle para un semáforo. Cualquier hilo del proceso llamado puede especificar el andel del semáforo en una llamado de las funciones de espera.

Se usa la función **ReleaseSemaphore** para incrementar la cuenta del semáforo en una cantidad específica. La cuenta del semáforo nunca podrá ser menor que cero o mayor a la cantidad especificada inicialmente, en el parámetro *lInicialCount.*

Generalmente, una aplicación usa un semáforo para limitar el número de los hilos que pueden usar los recursos. Antes de que un hilo pueda usar un recurso debe especificar el handle del semáforo en la llamada a una de las funciones wait. Cuando las funciones de espera los retornan, el semáforo cuenta uno y el hilo en espera es liberado para que continúe su ejecución; cuando termina de usar el recurso, este llama a la función releasesemaphore para incrementar la cuanto en uno.

La función **ReleaseSemaphore** puede además ser utilizada durante la ejecución de una aplicación. La aplicación puede crear un semáforo con un conteo inicial de cero. Esta inicialización del estado del semáforo desactiva y bloquea todos los hilos para la accesibilidad y protección del recurso. Cuando la aplicación termina la inicialización, se usa la función ReleaseSemaphore para incrementar la cuenta a su máximo valor y permitir el acceso normal a los recursos protegidos.

Los semáforos pueden ser usados en procesos cruzados o dentro de un proceso simple. Los multiprocesos pueden tener handles del mismo semáforo. Esto produce la sincronización entre objetos.

Se usa la función CloseHandle para cerrar un handle. El sistema cierra el handle automáticamente cuando termina el proceso. Cuando el handle para un semáforo es cerrado el objeto semáforo es destruido.

Funciones de espera

WinCE soporta funciones de espera para objetos simples y múltiples el cual bloquea o desbloquea un hilo basado en la activación o desactivación del estado de un objeto. Los objetos simples deben llamar a la función **WaitForSingleObject.** Los objetos múltiples deben llamar a las funciones **WaitForMultipleObject y MsgWaitForMultipleObject.**

La función de espera para objetos simples activa un hilo en espera. Este puede ser un objeto de sincronización, tal como un mutex o un evento, o él puede ser un handle a un proceso e hilo. Los handles son activados cuando sus procesos o hilos terminan. Así, un proceso puede monitorear otros procesos o hilos y ejecutar alguna acción basado en el estado de estos.

Las funciones de espera para los objetos múltiples activan las llamadas a los hilos de un arreglo específico que contienen uno o más handles de los objetos de sincronización. Estas funciones pueden retornar algunos valores en algunos de los siguientes eventos:

- El estado de cualquiera de los objetos está activado.
- El intervalo del time-out ha transcurrido. Se puede establecer el time-out en un valor infinito.

WinCE no soporta las esperas para todos los objetos que están activados.

Capítulo 4. Solución propuesta

4.1 Antecedentes

Estudio del problema a resolver

En el control del video se desea implementar dos funciones: primeramente poder controlar el encendido y apagado de cada televisor en forma independiente, teniendo como característica principal una interfaz amigable con el usuario. Luego se pretende ampliar la capacidad de los canales de video a 14 canales diferentes. Esto se implementará en dos etapas respectivamente. La primera función o etapa es la que se pretende lograr con el desarrollo de este proyecto.

Para proyectos futuros, que no se involucran con el que se plantea, la empresa desea desarrollar un control del audio que permita tener dominio sobre el nivel de volumen, en dos canales (inglés y español), en forma digital. También, ampliar el sistema para controlar la intensidad de la luz de lectura de cada pasajero, en mayor o menor intensidad, y controlar automáticamente el cierre y apertura de las ventanillas del avión.

Por otro lado se debe tener claro el perfil del usuario de este sistema, así como los recursos disponibles dentro del avión, para llevar a cabo lo que se tiene propuesto. El personal de cabina será quien opere el sistema, a petición del pasajero. El sistema debe contar con una interfaz agradable, de fácil uso y que permita controlar otros sistemas instalados en el avión. Asimismo, debe ser un sistema que pueda actualizarse fácilmente.

En otros aspectos técnicos propios de la aviación, se debe tomar en cuenta los aspectos físicos tales como la turbulencia, vibraciones en despegue y aterrizaje, temperatura, espacio, y recursos energéticos limitados. Todos estos aspectos afectan considerablemente el rendimiento y funcionamiento de los equipos electrónicos, produciendo efectos negativos en la durabilidad y estabilidad de los sistemas.

A la hora de enfrentar un problema con estas características, se tiene que buscar una solución que permita a su vez cumplir con los objetivos que se plantea resolver durante el desarrollo del proyecto de graduación, así como los objetivos que tiene planteados la empresa a cargo para hacer uso o venta del sistema a desarrollar.

Para desarrollar el sistema de control completo, se tenía que implementar todo el sistema mínimo computacional para su funcionamiento, desde el diseño de las conexiones e interface entre el microprocesador y las unidades externas, como lo son la memoria de trabajo volátil y la memoria no volátil, la unidad de almacenamiento, la unidad de despliegue y la unidad de entrada de datos, hasta realizar físicamente la conexión de todos estos componentes del sistema en una placa de trabajo (o wire wrap). Este tipo de conexión permite realizar las funciones deseadas con el sistema de control, pero con una menor eficiencia (menor frecuencia de operación del microprocesador) y con una mayor probabilidad de obtener errores de transmisión e interpretación de datos o información que en un sistema computacional ya elaborado en un circuito impreso, por parte de una empresa de prestigio, y cumpliendo con todas las normas y requerimientos que este tipo de aplicación implica.

Por otra parte, una de las metas de la empresa responsable del proyecto, así como de los estudiantes a cargo, es la de ofrecer un sistema de mayor eficiencia, que permita realizar un mayor número de funciones en la cabina de pasajeros del avión con posibilidades de ser introducido exitosamente en el mercado aéreo.

Debido a estas razones, se tomó la decisión de adquirir y comprar una plataforma de trabajo desarrollada por la empresa Logic Product Development para un microprocesador de la marca Hitachi, modelo SH-7727, seleccionado en un principio para el desarrollo del proyecto. Esta plataforma de desarrollo (o application kit como se le conoce en inglés) permite cumplir con todas las características mencionadas, como mayor eficiencia, cumplimiento de normas aéreas, equipo aprobado para su venta comercial, probabilidad de errores reducida, etc. Además, presenta la gran ventaja de ser un equipo de bajo costo, en comparación al precio de diseñar y montar en un circuito impreso todo el sistema computacional mínimo desarrollado al principio del proyecto.

4.2 Descripción General

Al tratarse de un proyecto novedoso, la empresa no tenía una solución clara al problema. De aquí que la solución propuesta a continuación se deba al trabajo de investigación de los estudiantes y del ingeniero a cargo, y a la labor desarrollada durante la primera parte del proyecto, como se comentó anteriormente; y que además cuenta con la aprobación y soporte por parte de la empresa.

Para lograr cumplir con los objetivos planteados y además cumplir con los nuevos objetivos para el control de distintas funciones, se va a hacer uso de una plataforma de aplicación o desarrollo, como se explicó en los apartados anteriores. Esta plataforma no varía en su estructura general la solución propuesta al inicio del proyecto, más bien lo que hace es ampliar el número de recursos y permite realizar una mayor cantidad de funciones de control. En la siguiente figura, se muestra un diagrama general de solución del control digital:

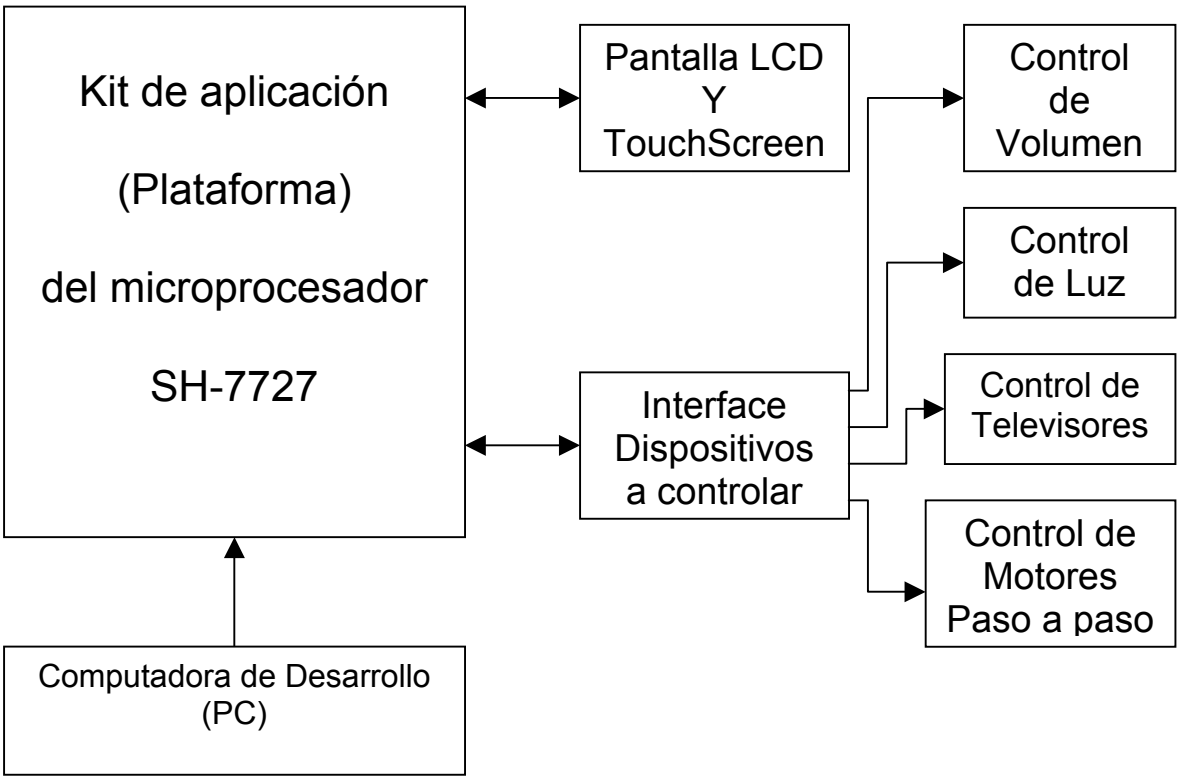

Figura 4.1 Diagrama General del Control Digital

4.2.1 Descripción de hardware

Como se observa en la figura 4.1, todas las funciones de control se realizan a partir de la plataforma de desarrollo o kit de aplicación del microprocesador SH-7727 de Hitachi. Esta plataforma está conectada con una pantalla de tipo LCD y su correspondiente pantalla de tacto (touchscreen). Asimismo, la plataforma será interfazada con subsistemas de control periféricos, los cuales controlarán funciones tales como: volumen del sistema central de audio, intensidad de luz en la cabina de pasajeros, encendido-apagado de los televisores y control de motores paso a paso. Además, a esta plataforma se podrá conectar una computadora de desarrollo tipo PC para poder realizar actualizaciones y cargas de programas en el almacenamiento secundario, y modificar las funciones de control de la plataforma.

4.2.2 Descripción de software

El sistema contará con una interfaz de alto nivel para interactuar con el usuario de los servicios de entretenimiento del avión, la cual cuando se activa una de las funciones se llaman las bibliotecas de hardware que tienen los protocolos de comunicación y control con los sistemas de bajo nivel; en este punto se activa la función que se desea, si da un error de funcionamiento se manda a la pantalla para que el usuario consulte al personal de soporte.

Las funciones de despliegue se harán por medio de una pantalla LCD ya que es delgada y tiene mejores cualidades de soporte mecánico que cualquier otro tipo de dispositivo de despliegue de datos. Para la captación de datos se utilizó una pantalla de tacto ya que es de más sencillo uso para el usuario, además de no requerir un espacio físico para que el usuario ingrese las instrucciones que desea realizar. Otro aspecto importante que se tomó en cuenta, es la imposibilidad de accesar a las configuraciones del sistema utilizando la pantalla de tacto.

4.3 Especificaciones del sistema

4.3.1 Especificaciones de hardware

Debido a la necesidad de una alta flexibilidad, para poder realizar actualizaciones que se apeguen a las necesidades del cliente y a las condiciones impuestas por los organismos internacionales especializados en esta área, es necesario desarrollar una plataforma que permita manejar un sistema operativo, lo cual tiene como característica principal independizar el software de interfaz con el usuario, del hardware existente. Además, el uso de un sistema operativo permite desarrollar una interfaz de alto nivel para el usuario con mayor facilidad y con mayores posibilidades de actualizaciones futuras.

Las plataformas (o computadoras) existentes cumplen con los requerimientos a nivel de software pero no cumplen con los requerimientos físicos propios de este medio de transporte (vibración, espacio, energía, temperatura).

Para soportar el manejo de un sistema operativo es necesario desarrollar un sistema computacional básico, que permita el manejo de las funciones básicas del sistema por medio de software. El hardware de un sistema computacional básico se denomina sistema mínimo y está compuesto principalmente por una unidad central de proceso (CPU), memoria de trabajo volátil, memoria de trabajo no volátil, memoria de almacenamiento secundario, y dispositivos de interfaz con el usuario.

Unidad Central de Procesos (CPU): Esta se encarga de procesar la información y convertir las instrucciones en acciones deseadas por el usuario. Por lo general esta unidad es implementada dentro de un microprocesador semiconductor.

Dadas las condiciones y requerimientos del problema a resolver, se debe escoger un microprocesador apto para realizar estas tareas. Dentro de las tareas que debe poder realizar se encuentran:

- Capacidad de uso de memoria virtual para manejo de memorias externas SDRAM de gran volumen de integración de 4Mbyte a 64Mbyte, así como una unidad de manejo de memoria y una unidad de manejo de caché para acelerar el uso de procesos y la multitarea.
- Velocidad de procesamiento de datos superior a los 100MIPS.
- Controladores de manejo de periféricos externos integrados, entre los más importantes para el sistema se buscó controlador del estado del bus, controlador de memoria cache, controlador de interrupciones y excepciones así como un controlador del LCD.
- Puertos de comunicación serie de alta velocidad (USB) superior a 5Mbps y de comunicación paralela.
- Control de interrupciones extendible.
- Direccionamiento de acceso directo a memoria para manejo de transferencias entre los dispositivos de almacenamiento secundario y la memoria principal, entre otros.
- Las menores dimensiones físicas y consumo de potencia, se alcanzan por medio de la alta integración y por medio de la tecnología CMOS de alta velocidad(Ver hojas de datos de la plataforma de desarrollo).

Memoria de trabajo volátil: En esta residen los programas e información que el usuario desea procesar; este tipo de memoria es la más importante a nivel de procesos. Por lo general es implementada por arreglos de celdas básicas semiconductoras conocidas como memoria de acceso aleatorio (RAM); dependiendo de la tecnología se pueden tener: SRAM, DRAM, SDRAM, etc.

En lo referente a la memoria RAM requerida, se necesita una tecnología con alta integración y la mayor velocidad posible con memorias SDRAM de 32Mbyte y dimensiones físicas de $2cm^2$, para acelerar de esa forma el manejo e intercambio de procesos. Otra característica importante que debe tener la memoria RAM es una alta capacidad de manejo de información, debido a que esta memoria funciona igualmente como memoria de video.

Memoria de trabajo no volátil: En esta memoria reside la información para la inicialización de los componentes de hardware, y las rutinas de servicio de interrupciones y carga del sistema operativo. Esta se implementa por medio de memorias de solo lectura (ROM), de compuerta flotante (EPROM) o las borrables eléctricamente (EEPROM, Flash ROM).

Dispositivo de interfaz con el usuario: Estos se encargan de interpretar la información del usuario y le reporta la información procesada por medios que son fácilmente interpretados por éste. Dentro de estos subsistemas se tienen los de entrada de información tales como el teclado, el mouse, touchpanel, etc; y los de salida o despliegue de información tales como monitor, parlantes, impresora, etc.

Por otra parte, se hace necesario el uso de una pantalla para el despliegue de alta definición unido a una pantalla de tacto, con el fin de cumplir con las especificaciones del problema.

4.3.2 Especificaciones de software

Un sistema mínimo requiere inicializar todos los componentes de hardware que lo componen para hacer uso de estos. Esta inicialización es conocida como rutinas de arranque, que a su vez configuran el sistema para la comunicación adecuada de todos los componentes. Se pueden distinguir varios momentos importantes de ejecución de estas rutinas:

- Arranque del microprocesador (Power On Reset) y realización del primer proceso de Fetch.
- Inicialización de los registros internos del microprocesador, en este se tiene el bus interno y los relojes del sistema.
- Inicialización y verificación de la memoria volátil.
- Inicialización de los dispositivos de hardware a utilizar.
- Leer el MBR (master boot record) en disco.
- Ejecutar el boot loader (cargador de sistema).
- Correr el sistema operativo.

A partir de las características propias del hardware del sistema, se hace necesario elaborar las correspondientes rutinas de inicialización y configuración de cada dispositivo.

Para poder operar el sistema, lo primero que se requiere hacer es inicializar los registros internos del microprocesador. Estos registros se dividen en dos grupos:

- Registros de control
- Registros de sistema

Por otra parte, para poder hacer uso de los dispositivos periféricos, es necesario especificar las características propias de cada uno, por medio de la programación de los registros internos de los controladores del CPU.

Para inicializar la unidad de almacenamiento, se debe obtener la información propia de este tipo de componentes (número de cilindros, cabezas y sectores, número de serie, capacidad, etc). Este proceso se realiza por medio de la programación de los comandos básicos requeridos por cualquier disco, y así poder realizar la transferencia de información a memoria principal.

Por otra parte, al tener un controlador de video integrado en el CPU, es necesario desarrollar las rutinas de despliegue de caracteres en pantalla y de manejo de gráficos. Estas rutinas, elaboradas en bajo nivel, deben ser compatibles con las instrucciones en alto nivel del sistema operativo.

Para adecuar el uso del sistema operativo a la plataforma que se desea implementar, se debe modificar la parte de más bajo nivel del sistema operativo (kernel), que es la encargada de administrar el hardware de acuerdo a los requerimientos de los programas de usuario.

Además se utiliza una computadora de desarrollo (PC) que permita realizar las modificaciones de software necesarias para que pueda manejar el hardware disponible. Esta información se envía desde esta computadora, a través de un puerto de comunicación serie, a la plataforma a desarrollar. La información ya almacenada en el disco tipo flash (FFD) debe ser instalada por medio de un algoritmo de configuración del sistema, el cual permite la lectura del MBR, la ejecución del boot loader y correr el sistema operativo.

Capítulo 5. Objetivos

5.1. Objetivo general

Implementar un sistema digital de control para el manejo de dispositivos de entretenimiento en aviones de aerolíneas privadas y comerciales.

5.2. Objetivos Específicos

5.2.1 Objetivos de hardware

1. Elaborar la interface de las líneas de alimentación del microprocesador Hitachi.

2. Diseñar e implementar la interface entre la unidad de control central (microprocesador) y la memoria de solo lectura (ROM).

3. Diseñar e implementar la interface entre la unidad de control central (microprocesador) y la memoria de lectura / escritura (SRAM).

4. Implementar la interface entre el microprocesador y la unidad de despliegue (pantalla LCD).

5. Crear la interface entre el microprocesador y la unidad de entrada de datos (touchscreen).

6. Diseñar e implementar el hardware de comunicación para la transferencia serie entre el microprocesador y la computadora de desarrollo (PC).

7. Desarrollar la interface entre los puertos del microprocesador y los opto - acopladores para el control de los relevadores de los televisores del avión.

8. Implementar la interface de conexión a distancia de las líneas de control de la pantalla LCD.

9. Desarrollar el control de intensidad y brillo de la pantalla LCD.

10. Implementar el controlador de activación de dispositivos del avión y generador de espacio de puertos secundario.

11. Diseñar e implementar la interface de control de intensidad y activación del sistema de luces del avión.

12. Desarrollar la interface de control de los motores paso a paso de los televisores en el avión.

5.2.2 Objetivos de software

1. Programar las rutinas de bajo nivel para la inicialización del controlador de LCD interno del microprocesador.

2. Diseñar e implementar el algoritmo de control para el despliegue en pantalla utilizando la pantalla LCD.

3. Diseñar e implementar el protocolo de comunicación para la transmisión serie entre el microprocesador y la computadora de desarrollo (PC).

4. Configurar el kernel básico del sistema operativo.

5. Instalar el sistema operativo en la unidad de almacenamiento.

6. Diseñar protocolos de prueba para comprobar la carga del sistema operativo.

7. Configurar las librerías de manejo de los dispositivos de hardware específicos de la aplicación.

8. Diseñar e implementar las rutinas de manejo de puertos de comunicación del microprocesador.

9. Realizar un protocolo de pruebas finales de funcionamiento del sistema de control de video.

10. Elaborar el algoritmo del dispositivo de control para el manejo del brillo e intensidad de la pantalla.

11. Programar el dispositivo de control para el control de encendido y apagado de los televisores.

12. Realizar la programación del dispositivo de direccionamiento y control de estado para el manejo de los drivers de los motores paso a paso.

13. Programar el dispositivo de control para el control de encendido e intensidad de luces de la cabina de pasajeros.

Capítulo 6. Metodología

6.1 Aspectos metodológicos relacionados con el hardware

1. Elaboración de la interface de las líneas de alimentación del microprocesador Hitachi.
2. Diseño e implementación de la interface entre la unidad de control central (microprocesador) y la memoria de solo lectura (ROM).
3. Diseño e implementación de la interface entre la unidad de control central (microprocesador) y la memoria de lectura / escritura (SRAM).
4. Implementación de la interface entre el microprocesador y la unidad de despliegue (pantalla LCD).
5. Creación de la interface entre el microprocesador y la unidad de entrada de datos (touchscreen).
6. Diseño e implementación del hardware de comunicación para la transferencia serie entre el microprocesador y la computadora de desarrollo (PC).

Estos objetivos se alcanzaron siguiendo la siguiente metodología:

a. Discusión con el asesor de la empresa sobre la aspectos técnicos de viabilidad y logística sobre el uso de la plataforma de desarrollo vía correo electrónico y vía telefónica.

b. Obtener por medio de internet la información de las características técnicas de la plataforma.

c. Discusión con el asesor de la empresa sobre la problemática de la memoria RAM que se requería para el diseño anterior.

d. Resolución final sobre el uso de la plataforma, se le envío un correo al asesor de la empresa.

e. Se realizó el pedido de compra.

7. Para el desarrollo de la interface entre los puertos del microprocesador y los opto -acopladores para el control de los relevadores de los televisores del avión se realizó:

   a. La escogencia de los componentes relevadores, optoacopladores, resistencias y transistores adecuados a los niveles de voltaje y corrientes del sistema.

   b. Diseño y pruebas del sistema de potencia.

   c. Integración al sistema de control de televisores.

8. Para el desarrollo de la interface de conexión a distancia de las líneas de control de la pantalla LCD se hizo:

   a. Investigación sobre el formado de transmisión LVDS (Conversión de señales a voltaje diferencial bajo), la cual se hizo por medio de internet.

   b. Búsqueda de los componentes adecuados para la transmisión en formato LVDS.

   c. Diseño de la interface de transmisión LVDS para el sistema de video.

   d. Búsqueda en internet de los componentes misceláneos para la transmisión y recepción (cables, jack, etc).

   e. Documentación general del formato LVDS (diagramas de conexión, diseño de tarjeta, etc).

9. Para el desarrollo del control de intensidad y brillo de la pantalla LCD, se realizó:

   a. Investigación sobre los métodos de control del brillo e intensidad en una pantalla LCD por medio de las hojas de datos generales de la pantalla.

   b. Diseño del control de brillo e intensidad por medio del método de PWM, en el inversor de la pantalla LCD.

10. Para el desarrollo del controlador de activación de dispositivos del avión y generador del espacio de puertos secundario.

a. Investigación sobre métodos de generación de espacio de puertos en sistemas computacionales.

b. Investigación y pruebas de los diferentes tipos de comunicación existentes en la plataforma de desarrollo.

c. Programación y pruebas del microcontrolador.

11. Para el desarrollo de la interface de control de intensidad del sistema de luces del avión se ha realizado:

a. Investigación sobre las características técnicas de las luces de la cabina del avión.

b. Determinación y diseño del control básico de las luces.

12. Para la implementación de la interface de control de los motores paso a paso de los televisores en el avión.

a. Investigación sobre los métodos de control de motores paso a paso.

b. Adaptación del sistema al espacio de puertos secundario, por medio de la conexión de una PAL22V10.

c. Programación y pruebas del microcontrolador con los motores.

6.2 Aspectos metodológicos relacionados con el software

1. Programación de las rutinas de bajo nivel para la inicialización del controlador de LCD interno del microprocesador.

2. Diseño e implementación del algoritmo de control para el despliegue en pantalla utilizando la pantalla LCD.

3. Diseño e implementación del protocolo de comunicación para la transmisión serie entre el microprocesador y la computadora de desarrollo (PC).

4. Configuración del kernel básico del sistema operativo.

5. Instalación del sistema operativo en la unidad de almacenamiento.

6. Diseño de protocolos de prueba para comprobar la carga del sistema operativo.

7. Configurar las librerías de manejo de los dispositivos de hardware específicos de la aplicación.

Estos objetivos se alcanzaron siguiendo el procedimiento metodológico de los puntos de 1 a 7, planteado en la sección 6.1.

8. Para el diseño del programa de manejo de puertos de la plataforma para direccionar y recibir información de los dispositivos de control externos se realizó:

   a. Una investigación del sistema operativo para conocer las detalles técnicos del acceso de puerto y obtener información en la pantalla.

   b. Hacer pruebas de despliegue en pantalla y direccionamiento de espacio en memoria.

   c. Programación en lenguaje C++ para los procesadores embebidos (embedded) de los procedimientos anteriores.

9. Realización de un protocolo de pruebas finales de funcionamiento del sistema de control de video.

Este objetivo se alcanzó siguiendo el procedimiento metodológico de los puntos de 1 a 7, planteado en la sección 6.1.

10. La programación del dispositivo para el control de brillo e intensidad de la pantalla se realizó:

    a. Diseño del algoritmo de control para el manejo del brillo e intensidad de la pantalla LCD.

    b. Programación del controlador de brillo e intensidad en un micro controlador 16F876 de Microchip.

    c. Pruebas del sistema mediante simulación de entradas.

11. Programación del dispositivo de control del encendido y apagado de los televisores.

12. Programación del dispositivo de direccionamiento y control de estado para el manejo de los drivers de los motores paso a paso.

Estos objetivos se alcanzaron siguiendo la siguiente metodología:

    a. Diseño del algoritmo para el controlador por medio del compilador en C del microcontrolador.

    b. Implementación y pruebas del algoritmo para el cumplimiento de las diferentes condiciones impuestas para este controlador por medio del compilador en lenguaje C del microcontrolador.

    c. Programación final y pruebas de puesta a punto con el sistema completo de control de televisores.

13. La programación del dispositivo de control para el control de encendido e intensidad de luces de la cabina de pasajeros se realizó mediante:

    a. Diseño del algoritmo de control para el manejo de encendido e intensidad de luces de en cabina.

b. Programación del controlador de luces en un micro controlador 16F876 de Microchip.

c. Pruebas del sistema mediante simulación de entradas.

Capítulo 7. Descripción detallada de la solución

7.1 Descripción general del funcionamiento

El sistema a implementar en esta etapa es una fracción de la implementación de un sistema mayor, en el cual se pretende un funcionamiento análogo al de una computadora, o una agenda de despliegue portátil (PDA). Este sistema se puede representar por medio de un diagrama de bloques, el cual se muestra a continuación.

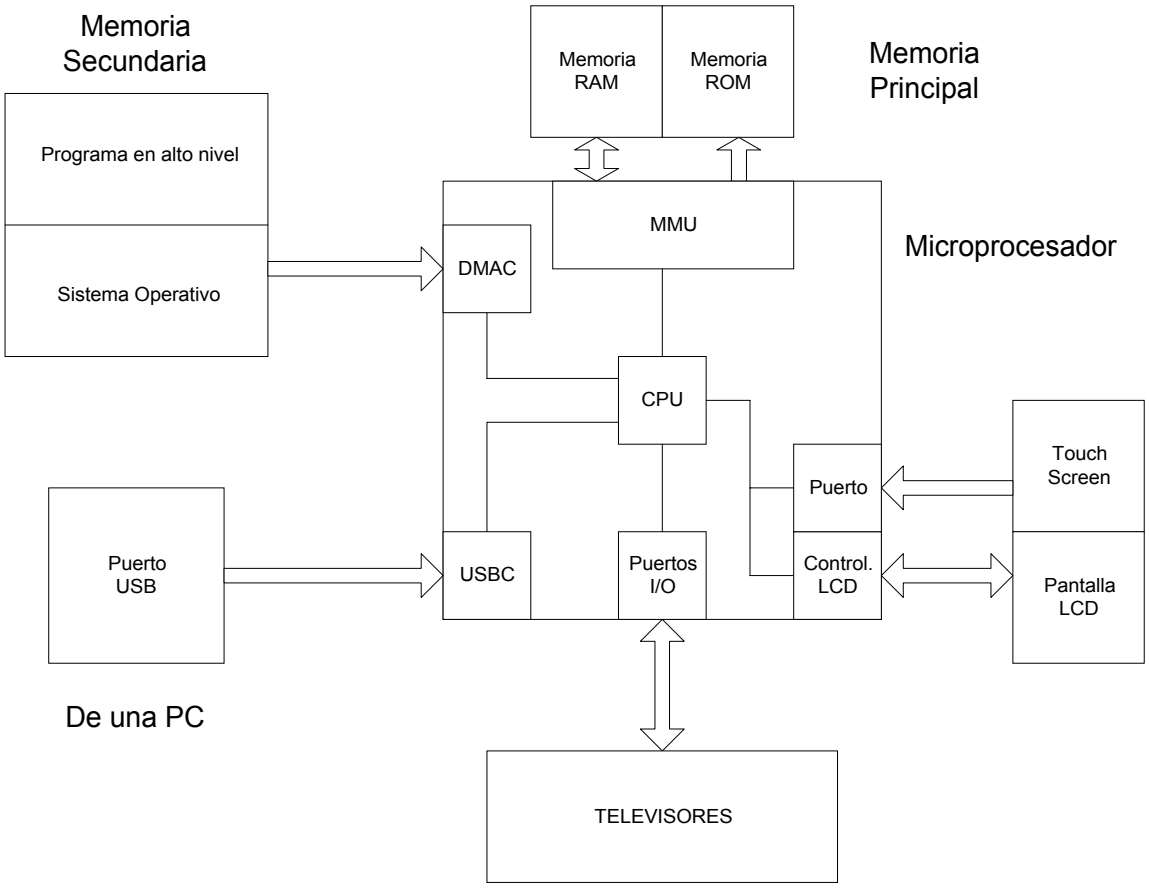

Figura 7.1 Diagrama de bloques del sistema por implementar.

7.2. Descripción del hardware

Para realizar el sistema completo de control de los dispositivos en la cabina de pasajeros del avión, se tuvo que diseñar toda una circuitería y un sistema externo de control para poder realizar diferentes funciones y no sobrecargar el uso de la plataforma de desarrollo del microprocesador SH-7727. En la siguiente figura se muestra un diagrama general de todo el sistema de control a implementar.

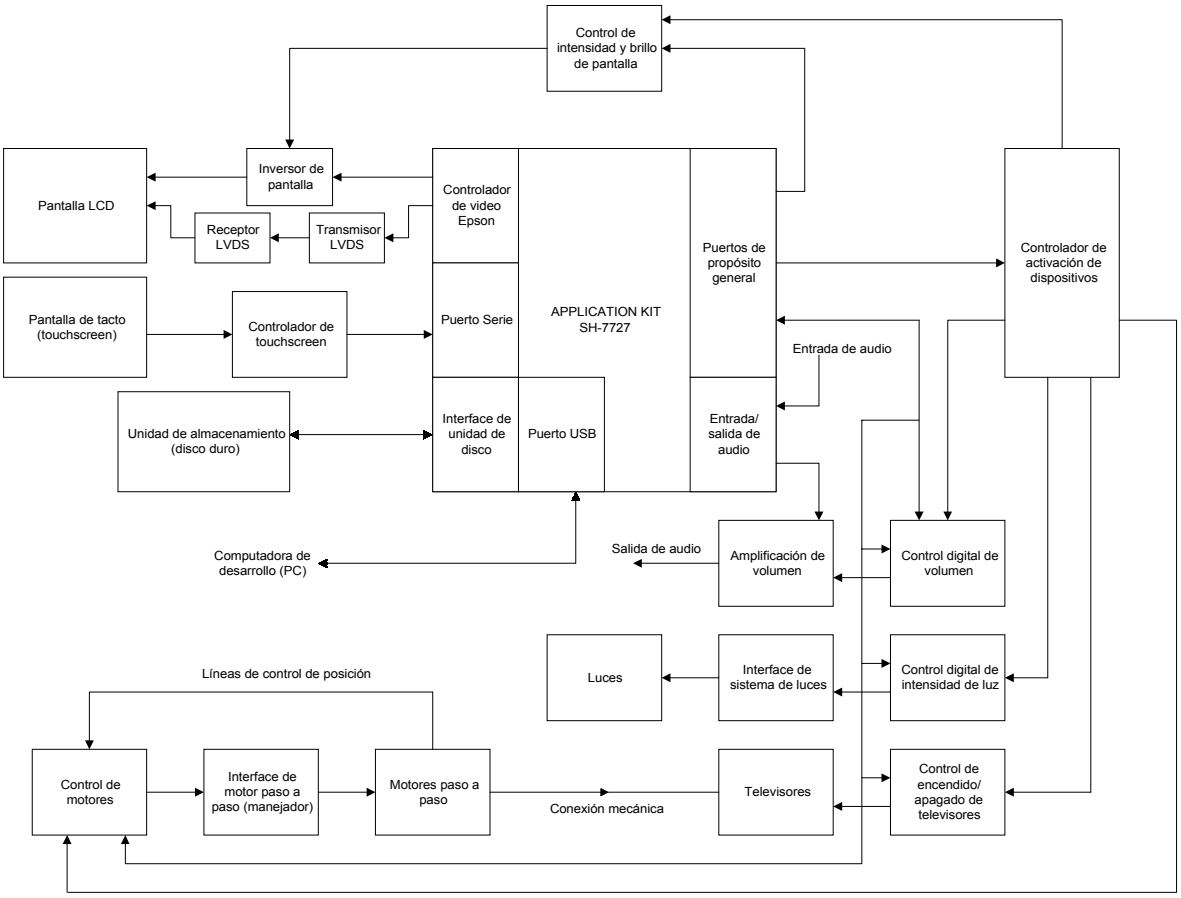

Figura 7.2 Diagrama de bloques del sistema de control

7.2.1 Control de brillo e intensidad de la pantalla LCD

Las pantallas de tipo LCD (pantallas de cristal líquido) requieren del uso de un inversor que permite convertir un voltaje de CD a un voltaje de CA para alimentar a la pantalla; además permite proporcionar a la pantalla un voltaje de arranque de más de 1000 voltios para poner a funcionar el tubo de pantalla, regular la corriente por este tubo y además proporcionar una entrada para una señal que controle el brillo e intensidad de la pantalla. Para poder hacer el control de estos dos parámetros a través del pin de entrada del inversor, se genera una señal PWM (modulación por ancho de pulso) a una frecuencia determinada, propia del tipo de pantalla utilizada. En la siguiente figura se muestra el diagrama de este sistema de control.

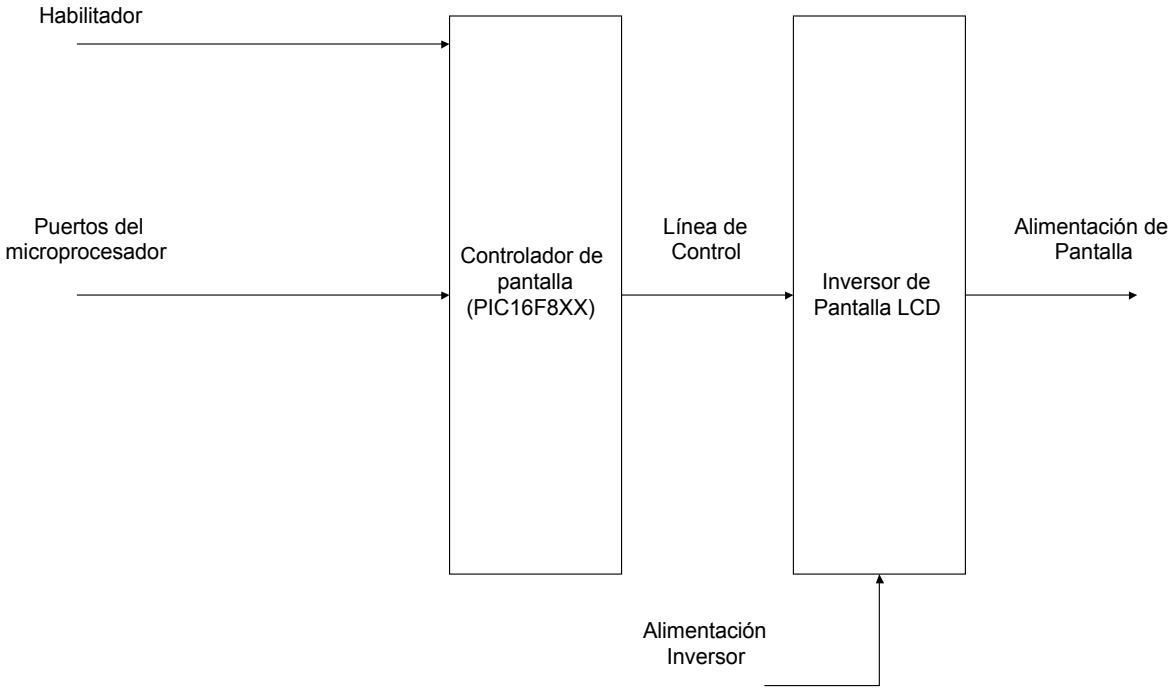

Figura 7.3 Diagrama del control de brillo e intensidad de pantalla

7.2.2 Control de activación de dispositivos en la cabina del avión

Al tener un sistema de control central que permite manejar diferentes dispositivos en la cabina de pasajeros del avión, se debe tener la capacidad de poder controlar cada dispositivo en el momento deseado, pero a su vez deshabilitar esta función cuando no se esté utilizando. Para esto se hace uso de una lógica de decodificación que permite activar y desactivar cada dispositivo, a la vez que nos permite hacer uso de un mayor número de dispositivos de control.

En la siguiente figura se muestra el diagrama del controlador para manejar la activación y desactivación de los subsistemas de control de dispositivos.

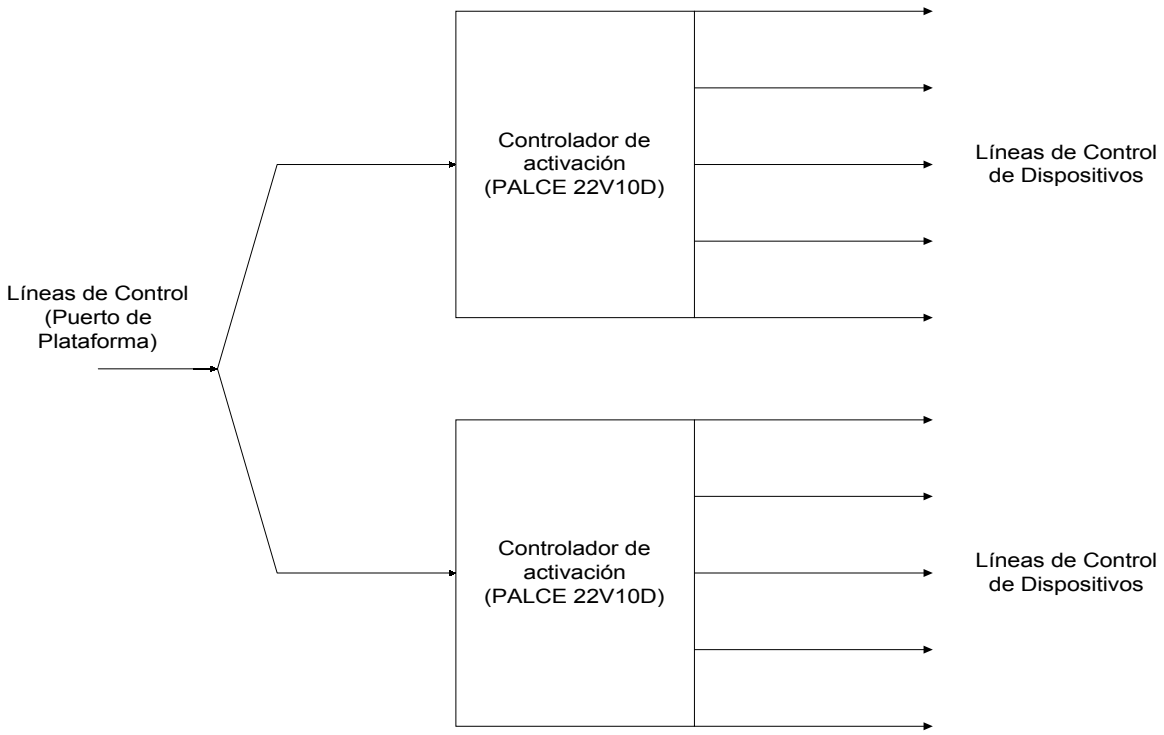

Figura 7.4 Diagrama del controlador de activación de dispositivos

7.2.3 Control de intensidad de luz en la cabina de pasajeros

En la cabina de pasajeros de distintos aviones existe un sistema de luces que permite encenderlas y apagarlas, pero que no regula su intensidad. Para controlar este parámetro, se genera a partir de la plataforma una señal digital que pasa a través de un controlador, el cual regula un amplificador de voltaje que permite aumentar o disminuir la intensidad de las luces en la cabina de pasajeros.

En la siguiente figura se muestra el diagrama para el control de la intensidad de las luces en cabina.

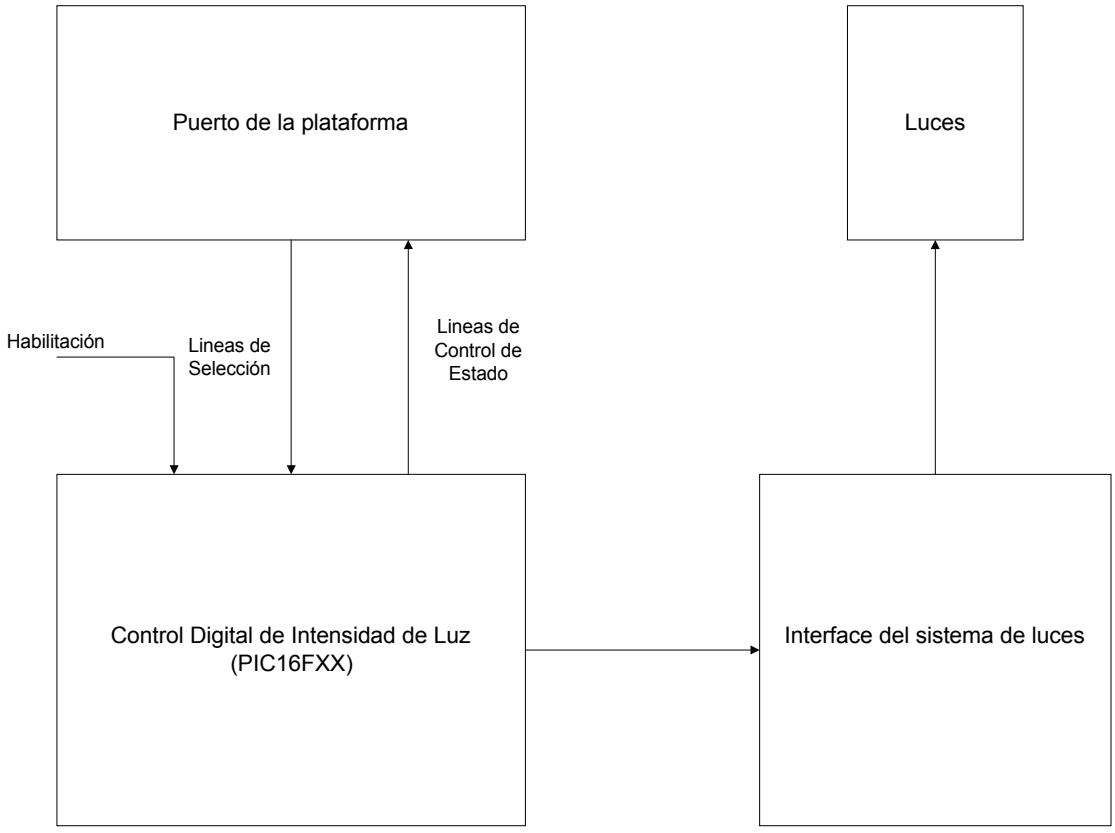

Figura 7.5 Diagrama del control de intensidad de luz en cabina

7.2.4 Control de encendido / apagado y posición de televisores

En la cabina de pasajeros de ciertos aviones comerciales y privados existe el inconveniente de que el sistema de video enciende todos los televisores en forma paralela. Por medio de la plataforma del SH-7727 se envía una señal digital a un controlador, que maneja los relevadores de encendido y apagado de los televisores individualmente.

Por otra parte, cada televisor se encuentra montado en una base diseñada de forma que permita ocultar el televisor dentro del panel del cielo en la cabina del avión. Para lograr esto, se requiere de una interface para cumplir con los requerimientos de voltaje y corriente de este tipo de motores, así como la secuencia para realizar el movimiento paso a paso de estos y la dirección de giro. Además, por medio de un sistema de control de posición se sensa la posición de cada televisor, de forma que se conozca el estado de cada uno. Para cumplir este objetivo la plataforma administra un sistema de drivers (NMJ3517), que a su vez controlan la corriente y giro de los motores paso a paso de cada televisor.

En la siguiente figura se muestra el diagrama para el control del encendido / apagado y posición de los televisores en la cabina de pasajeros del avión.

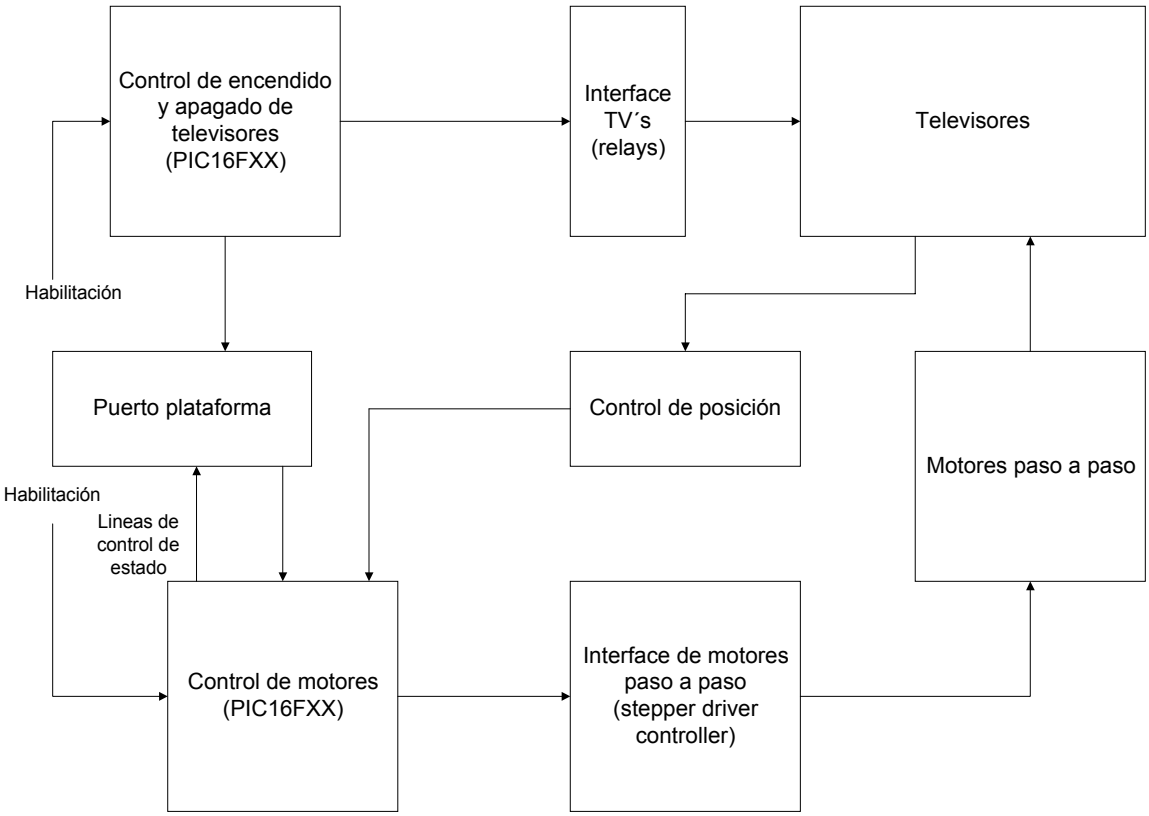

Figura 7.6 Diagrama de control de encendido/apagado y posición de los televisores

7.2.5 Interface en formato LVDS para la pantalla LCD

Para realizar la interface entre el controlador y la pantalla de tipo LCD, es necesario realizar una interface entre las líneas que salen del controlador de video y las de entrada a la pantalla, con el fin de poder enviar esta señal a varios metros de distancia (entre 15 y 20 metros), además que tenga alta inmunidad al ruido, bajo nivel de emisiones electromagnéticas (EMI) y con bajo consumo de potencia. Para lograr esto se requiere del siguiente equipo:

- Un convertidor-transmisor de TTL a LVDS.
- Un convertidor-receptor de LVDS a TTL.
- Un cable de conexión entre el controlador de LCD y el transmisor, y un cable de conexión entre el receptor y la pantalla.
- Un cable para la conexión entre el transmisor y el receptor.
- Conectores para cable.

Para ejemplificar esta interface, se puede observar el siguiente diagrama.

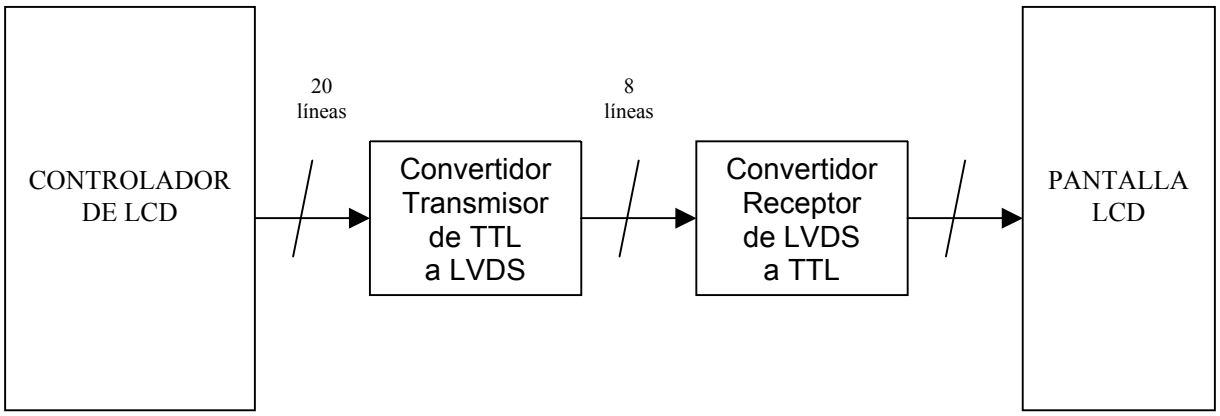

Figura 7.7 Diagrama de bloques del control LVDS para la pantalla LCD

En caso de tener una distancia mayor que 20 metros, se deberá implementar el siguiente circuito o etapa adicional.

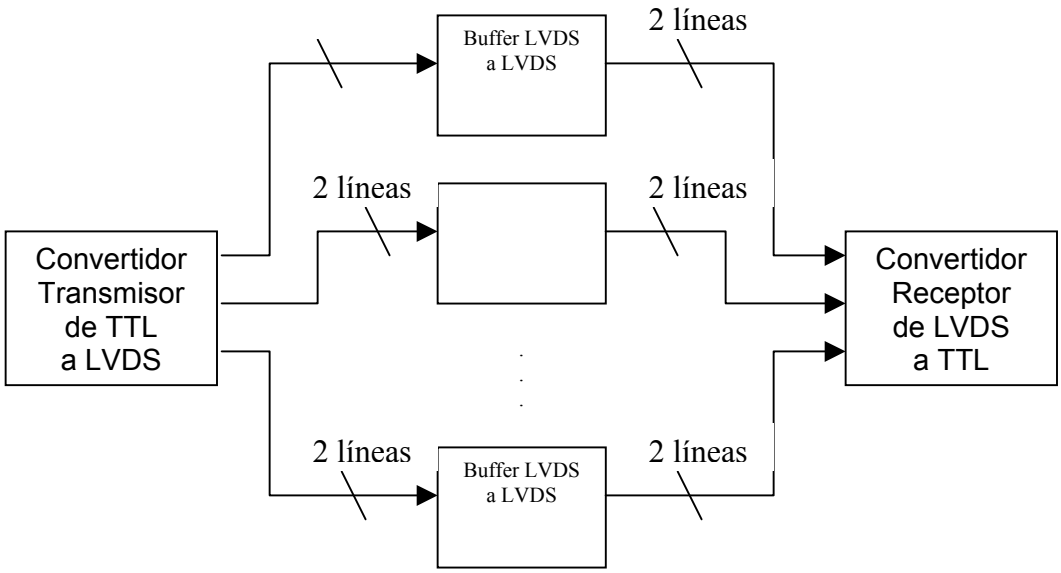

Figura 7.8 Diagrama de bloques del control LVDS con buffers para la pantalla LCD

7.3. Descripción del software

Con el fin de realizar todas las funciones de control del sistema, no solo es necesario realizar la programación de un algoritmo de manejo de puertos e interface para la plataforma utilizada. Este sistema de control está compuesto, como se observa en la figura 7.2, de todo un sistema adicional externo de control a la plataforma de desarrollo, que permite realizar un gran número de funciones de control, así como permite no recargar el uso de la plataforma y da la posibilidad de expansión al sistema.

Diferentes funciones de control externas se realizan solamente con la implementación de circuitos integrados específicos para realizar un determinada tarea, como lo es la utilización de un CI para la conversión a formato LVDS, o bien el inversor para la pantalla LCD. Pero existen otras funciones, como lo es el control de encendido y apagado de televisores, el control de posición para los motores paso a paso o bien el manejo del brillo e intensidad de pantalla, que requieren programar un circuito externo para poder realizar estas tareas. Estas funciones se realizan haciendo uso de diferentes controladores de interrupciones programables (PIC), los cuales se programan haciendo uso del software del programación PIC C Compiler o bien del software MPLab (ambos en lenguaje C para este tipo de dispositivos).

Entre los diferentes algoritmos que se requieren para realizar las funciones de control externas se encuentran:

- Programación de un microcontrolador para el manejo del brillo e intensidad de la pantalla LCD.
- Programación de un microcontrolador para el manejo del encendido y apagado de los televisores en la cabina de pasajeros.
- Programación de un microcontrolador para el control de posición de los televisores en la cabina de pasajeros.
- Programación de un microcontrolador para el manejo del encendido y apagado de luces en la cabina de pasajeros.

A continuación se detalla cada una de las diferentes etapas de programación para las funciones de control externas.

7.3.1 Manejo del brillo e intensidad de la pantalla LCD

El control de estos dos parámetros, se realiza mediante una señal que controla un pin de entrada en el inversor de la pantalla LCD. Esta entrada de control es una señal pulsante por la cual se controla el ancho de pulso y el rango de amplitud de éste, controlando así el brillo y la intensidad respectivamente. Por otra parte es necesario controlar la frecuencia de esta señal para cumplir con los requerimientos del inversor de este tipo de pantallas. Para poder realizar esta función se realizó la programación de un controlador PIC16F876, el cual maneja las siguientes líneas:

1. Seis líneas de entrada con la dirección para determinar el valor de brillo de la pantalla, la intensidad de la pantalla (rango) y la frecuencia de la señal de control.
2. Dos líneas de entrada que permiten realizar y escoger la programación en diferentes tiempos ya sea del brillo, intensidad o frecuencia.
3. Una línea de entrada que permite escoger la habilitación o no del sistema de control de la pantalla.
4. Una línea de salida hacia el inversor de la pantalla LCD para el control de estos parámetros.

Esta señal de salida no puede ir directamente conectada al pin de control del inversor de la pantalla, debido a que no cumple con los requerimientos de voltaje y corriente de este último. Para poder lograr esto, se hace uso de un opto-acoplador conectado a la salida del PIC y que permite acoplar esta señal con la entrada del inversor de la pantalla LCD.

7.3.2 Manejo del encendido y apagado de los televisores en la cabina de pasajeros

Para realizar el control del encendido y apagado de los televisores en la cabina de pasajeros del avión, se utiliza un controlador PIC16F87X, el cual se encarga de recibir un código de la plataforma y realizar la función sobre el o los televisores determinados; de igual forma recibe una señal de cada televisor para controlar el estado del mismo, y le envía esta información a la plataforma para determinar cualquier error que pueda presentarse. Este controlador maneja las siguientes líneas:

1. Cuatro líneas de entrada que permiten direccionar los registros internos del controlador para la activación y desactivación de televisores.
2. Seis líneas de entrada para controlar el estado de cada televisor individualmente.
3. Una línea de entrada para habilitar o deshabilitar este controlador.
4. Seis líneas de salida, cada una para encender o apagar cada televisor en la cabina de pasajeros a través de un relevador (relay).
5. Tres líneas de salida de datos para informar a la plataforma del estado de funcionamiento de cada televisor en cabina.

Para encender o apagar cada televisor, alimentados por una señal de 28 V en CD propia del avión, se debe hacer uso de un relevador por cada televisor que permita disparar por medio de una señal de tipo TTL (proveniente del PIC) la alimentación a cada televisor.

7.3.3 Control de posición de los televisores en la cabina de pasajeros

Como se comentó anteriormente, cada televisor se encuentra montado en una base la cual permite ocultarlos en el panel de cielo del avión, haciendo uso de un motor paso a paso. Para realizar esta función se utiliza un controlador PIC16F87X, que permite controlar la posición de cada televisor por medio de un sistema de interruptores en cada uno, además de enviar las señales de dirección y velocidad de desplazamiento a cada manejador de los motores paso a paso para ocultar o desplegar cada televisor; además, este controlador le envía a la plataforma un conjunto de datos sobre el estado de la posición de cada televisor y así determinar cualquier error.

El controlador de interrupciones maneja las siguientes líneas:

1. Cuatro líneas de entrada que provienen de la plataforma para controlar la posición de cada televisor.
2. Una línea de entrada que permite habilitar o deshabilitar este dispositivo de control.
3. Tres líneas de entrada para conocer la posición de cada televisor y determinar la señal de envio a los manejadores de cada motor paso a paso.
4. Seis líneas de salida para controlar la velocidad y el desplazamiento de los motores paso a paso a través de cada manejador stepper.
5. Seis líneas de salida para controlar la dirección de giro de cada motor paso a paso a través de los manejadores stepper.
6. Tres líneas de salida de datos para informar a la plataforma de la posición de cada televisor en cabina.

Para controlar la posición de los televisores, se utiliza un conjunto de interruptores que indican si el televisor se encuentra oculto (arriba) o desplegado (abajo). Estas señales de cada interruptor (dos por televisor), se envían a un circuito integrado PALCE22V10 el cual codifica de doce a tres líneas, y las envía al controlador para manejar los motores paso a paso.

Cada motor paso a paso se conecta a un CI stepper driver (manejador de motores paso a paso) que se maneja por medio de una señal pulsante y una señal digital, que permite determinar la velocidad de giro de los motores y la dirección de giro respectivamente. Estas dos señales son las que se manejan por medio del controlador a través de las señales de los interruptores de posición y de las señales de direccionamiento provenientes de la plataforma.

7.3.4 Manejo del encendido y apagado de luces en la cabina de pasajeros

De igual forma que para el control de los televisores, es necesario controlar el encendido y apagado de dos filas de luces en la cabina de pasajeros del avión. Para realizar esta función se utiliza un controlador PIC16F87X, el cual se encarga de recibir un código de direccionamiento de la plataforma para el encendido y apagado de las luces. Este controlador maneja las siguientes líneas:

1. Dos líneas de entrada de dirección para el control del encendido y apagado de las filas de luces en la cabina de pasajeros.
2. Una línea de entrada para la habilitación y deshabilitación de este dispositivo de control.
3. Dos líneas de salida para la activación de las luces en la cabina del avión.

Para realizar la activación de cada fila de luces en la cabina de pasajeros del avión, es necesario utilizar un relevador (relay) por cada fila para activar la alimentación necesaria para encender las luces, en este caso la alimentación de 28 V en CD propia del avión.

7.4. Alcances y limitaciones

Como se planteó al inicio del proyecto, uno de los objetivos era realizar la interface con el disco duro disponible, en este caso un disco tipo flash (FFD) con una interfaz de tipo IDE y formato ATA. Este tipo de unidad de almacenamiento no cuenta con un puerto específico en la plataforma de desarrollo con la que se está trabajando, diseñada específicamente para discos de tipo PCMCIA y CompactFlash. La conexión de software con la plataforma es realizable con este tipo de disco, pero se encuentra limitada la conexión de hardware (la conexión física) de este disco flash con los puertos de la plataforma de desarrollo, al no poder hacerse una conexión fiable con este; es decir, que se requeriría realizar un alambrado “casero” con cable para wire wrap, que no es conveniente para este sistema de hardware.

Luego de realizar un estudio y un diagrama de conexión entre la unidad de almacenamiento y la plataforma de desarrollo, se llegó a la conclusión de que lo mejor es adquirir una unidad de almacenamiento apta para conectarse con los puertos específicos del sistema. Además, ya que la plataforma de desarrollo ya cuenta con el sistema operativo instalado, no existe ningún problema de realización y finalización del proyecto debido a la no utilización de la unidad de almacenamiento.

7.5. Aporte de los estudiantes al proyecto.

Los estudiantes aportaran el diseño e implementación de un sistema computacional básico, tanto al nivel de hardware como al nivel de software, así como la asesoría que sea necesaria para la instalación de los prototipos, que estén listos para ello.

Dentro de las labores a las cuales no se comprometen están: el diseño de pistas, ni el montaje de los componentes en una tarjeta impresa, ya que esto excede las políticas planteadas por la Escuela de Electrónica en cuanto a la realización de un proyecto de graduación.

Este proyecto tiene la cualidad de unir la ingeniería en electrónica con la ingeniería en computación en un nivel en el cual no se logra vislumbrar la diferencia entre una y otra, ya que muchas de las rutinas y algoritmos tanto del arranque como del kernel del sistema operativo requieren un conocimiento más profundo del hardware. Mucha de esta tecnología proviene del extranjero, como lo son las tarjetas madre, las plataformas de desarrollo o dispositivos de despliegue portátil (PDA), por tanto, es un área en la que no ha habido la suficiente investigación en nuestro país, a pesar de existir excelentes teóricos en el tema, tanto en el área de electrónica como en el área de la computación.

Capítulo 8. Bibliografía

8.1 Manuales

1. Customer Service Division, *SH7727 Hardware Manual*. 1era Edición, Japón. (www.hitachisemiconductor.com) Agosto 2001.

2. Electronic Devices Sales & Marketing Group, *SuperH RISC engine, C/C++ compiler, Assembler, Optimizing Linkage Editor User's Manual*. 2era Edición, Japón. (www.hitachisemiconductor.com) Octubre 2001.

3. Logic Product Development *Quikstart Guide.* CD Rom. Minneapolis, USA: Logicpd, 2002.

4. Logic Product Development *Setup and User´s Guide.* CD Rom. Minneapolis, USA: Logicpd, 2002.

5. Logic Product Development *Hardware Specification.* CD Rom. Minneapolis, USA: Logicpd, 2002.

8.2. Hojas de datos

6. National Semiconductor Corp. *DS90CR561/DS90CR562 LVDS 18-Bit Color Flat Panel Display (FPD) Link.* Hoja de datos. USA: Julio, 1997

7. NEC. *High Isolation Voltage Single Transistor Type Multi Photocoupler Series.* Hoja de datos. USA: Octubre, 2001.

8. New Japan Radio Co. Ltd. *Stepper Motor Controller/Driver.* Hoja de datos. Japón.

8.3. Direcciones de correo:

9. techsupport@hsa.hitachi.com

Esta dirección de correo es de soporte técnico de Hitachi para obtener información acerca del microprocesador y el software de programación de este.

10. platformsupport@logicpd.com

Este correo es del soporte técnico de la plataforma de desarrollo utilizada.

Capítulo 9. Glosario

9.1. Glosario de términos

Buffer: Circuito integrado que acopla los niveles de corrientes y voltajes entre dos elementos electrónicos; es bidireccional.

Codec: Dispositivo codificador y filtrador de señales análogicas.

Disco Flash (FFD): Unidad de almacenamiento secundaria semiconductora basada en tecnología de la memoria flash, que tiene una interfaz similar a la de un disco duro de platos magnéticos.

Kernel: Capa de más bajo nivel de un sistema operativo, que se encarga de ocultar las peculiaridades del hardware a los programas de usuario.

Linker: Rutina que permite ligar o unir dos funciones.

Sistema Operativo (OS): Conjunto de programas para la administración de recursos limitados en un sistema computacional, por medio de la administración de procesos que se realiza a través de la asignación de tiempo de la CPU (procesos activados y bloqueados).

Slot: Espacio o campo en memoria para almacenar datos y realizar diferentes funciones.

Touchscreen: Dispositivo de entrada de datos que da las coordenadas de un punto en la pantalla seleccionado por el usuario.

WinCE: Es el diminutivo del nombre de un sistema operativo de Microsoft para sistemas con un set de instrucciones reducido.

9.2. Abreviaturas

CA: Corriente alterna.

CD: Corriente directa.

DLL: Librería dinámica.

FAA: Administración Federal de Aviación.

IRQ: Solicitud de interrupción.

ISR: Rutina de servicio de interrupción.

LCD: Pantalla de cristal líquido.

LVDS: Conversión de señales de bajo voltaje diferencial.

PAL: Arreglo lógico programable.

PS/2: Puerto serial tipo 2.

PWM: Modulación por ancho de pulso.

MMC: Controlador de manejo de memoria.

RISC: Computadora de set de instrucciones reducido.

ROM: Memoria de solo lectura.

SRAM: Memoria de acceso aleatorio estática.

UART: Receptor –transmisor asincrónico universal.

USB: Bus de comunicación serie universal.

Apéndices

Apéndice A.1: Información general sobre el proyecto

A.1.1 Información de los estudiantes

Osvaldo Alvarado Bolívar

Cédula: 1-1052-0730 **Carné ITCR:** 9802078

Dirección de su residencia en época lectiva: Guadalupe de Goicoechea, San José, 75m. Este del Centro Comercial de Guadalupe, contiguo a Dispasa.

Dirección de su residencia en época no lectiva: Guadalupe de Goicoechea, San José, 75m. Este del Centro Comercial de Guadalupe, contiguo a Dispasa.

Teléfono en época lectiva: 256 7975

Teléfono en época no lectiva: 256 7975

E-mail: ocab_ab@hotmail.com

Juan Pablo Zawadzki Wisniewski

Cédula: 1-1024-003 **Carné ITCR:** 9615814

Dirección de su residencia en época lectiva: 800 metros norte de RECOPE, Ochomogo, Cartago.

Dirección de su residencia en época no lectiva: 800 metros norte de RECOPE, Ochomogo, Cartago.

Teléfono en época lectiva: 537 2041

Teléfono en época no lectiva: 537 2041

Teléfono celular: 399 1462 **Fax:** 551 0540

E-mail: jpzawazki@hotmail.com

A.1.2 Información del proyecto

Nombre del proyecto: "Control Digital para un Sistema de Video"

Área del proyecto: Sistemas Digitales

A.1.3 Información de la empresa

General Aerospace, Inc.

Zona: Florida, USA

Dirección: 932 Ponce de Leon Blvd.. Coral Gables, FL 33134

Teléfono: (305) 442-2124

Fax: (305) 447-1315

E-mail: genspace@worldnet.att.net

Actividad Principal: Servicios de Ingeniería en Aviación

A.1.4 Información del encargado de la empresa

Ing. Emilio Brenes Pereira

Área de formación: Ingeniería Electrónica

Departamento: Control de calidad e ingeniería

Teléfono: (305) 445-2124

E-mail: e1brenes@aol.com

A.1.5 Información del asesor por parte de la Escuela de Electrónica

Ing. Carlos Badilla Corrales

Puesto que ocupa: Profesor

Grado: Ing. Eléctrico grado maestría

Área: Sistemas digitales

Teléfono: 550 2694

Fax: 591 6629

E-mail: cbadilla@itcr.ac.cr

Apéndice A.2: Manuales de los controladores desarrollados en el proyecto

A.2.1 Registro serial de direccionamiento de memoria / Registro serial de datos en memoria

Descripción general

Este dispositivo funciona como un registro de direccionamiento y de control de datos en memoria de forma serial por medio de un puerto RS232. Convierte la información vía recepción serie en bytes de datos, control y direcciones, y de igual forma recibe datos y los transmite en forma serial. Tiene un bus de datos y direcciones de 8 bits, y una línea de selección de lectura / escritura.

Características

- Velocidad de transmisión: ≈37600 Baudios.
- Entradas y salidas compatibles con LS-TTL.
- Compatibilidad con diferentes formatos de comunicación serie.
- Bus de datos y direcciones independientes.
- Empaquetado: DIP40.
- Análogo al MDR / MAR de la arquitectura de microprocesadores.

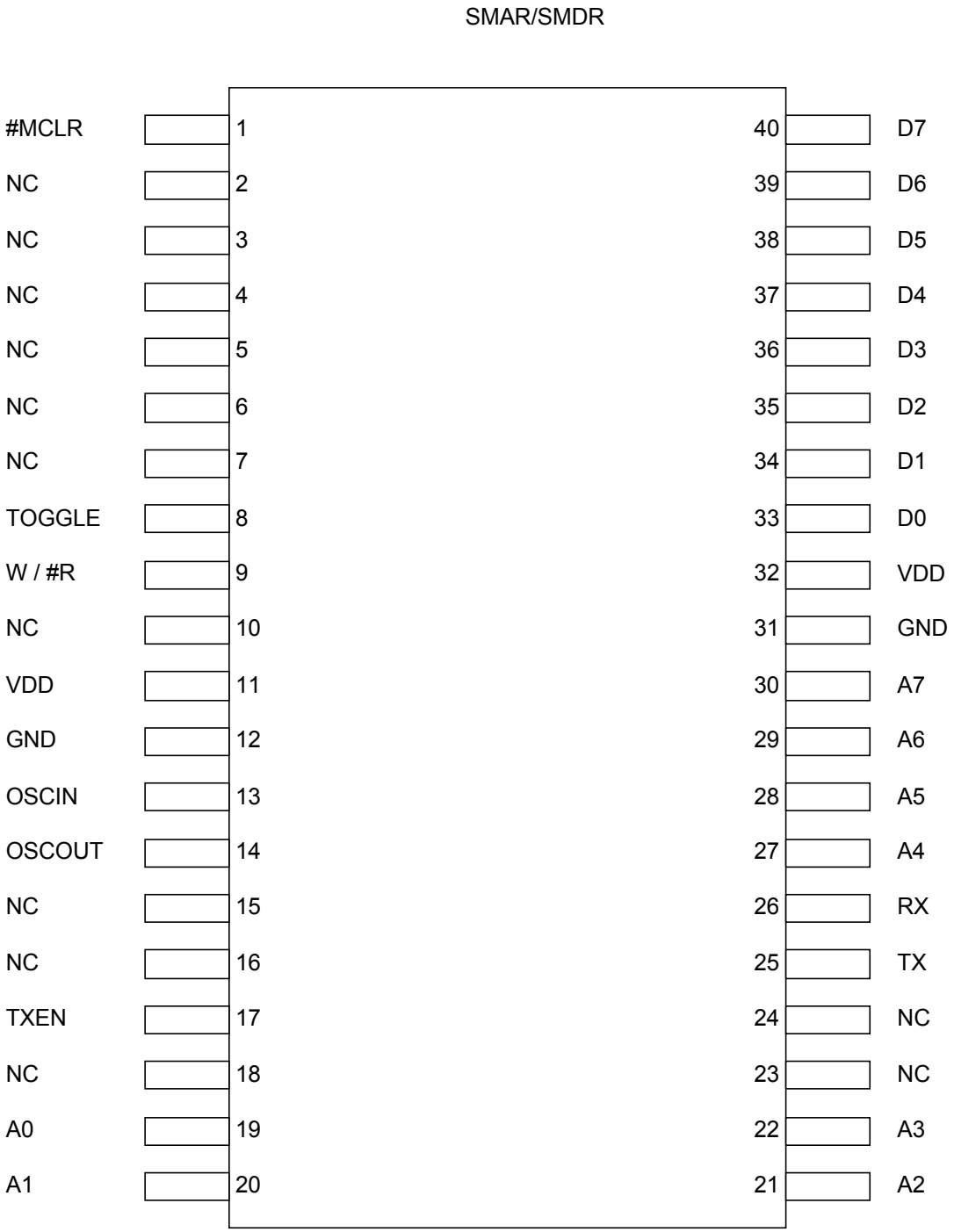

Figura A.1 Diagrama de pines del SMAR/SMDR

Configuración de Pines

Tabla A.1 Configuración de pines del SMAR/SMDR

Nombre del PIN	I/O	No.	Descripción
A[0..7]	I	19,20,21,22, 27,28,29,30	Bus de direcciones
D[0..7]	I/O	33,34,35,36, 37,38,39,40	Bus de datos bidireccional
W / #R	I	9	Selección de lectura – escritura
RX	I	26	Recepción de comunicación serie
TX	O	25	Transmisión de comunicación serie
TOGGLE	O	8	Señal cuadrada de sincronización
TXEN[2]	I	17	Activación de la transmisión
#MCLR	I	1	Reseteo general del controlador
OSCIN	I	13	Entrada de reloj
OSCOUT	O	14	Salida de reloj
VDD	I	11,32	Fuente alimentación +5V
GND	I	12,31	Tierra

[2] Este pin debe de estar siempre en un 1 lógico (conectado a VCC).

Diagrama de bloques

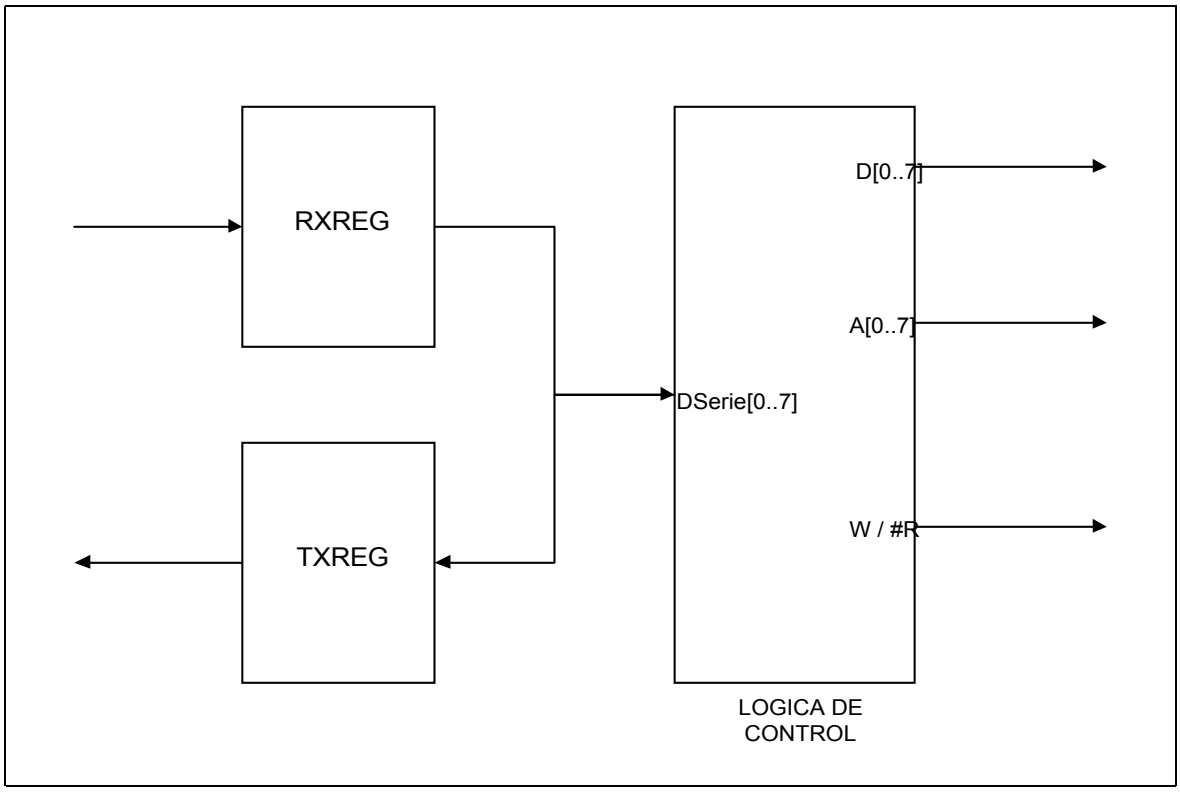

Figura A.2 Diagrama de bloques del SMAR/SMDR

A.2.2 Controlador de Televisores

Descripción general

Este dispositivo es un controlador de un sistema de Televisores, con un máximo de tres aparatos por integrado. Tiene un bus de datos de 8 bits, y un bus de direcciones de 4 bits.

Dentro de sus funciones primarias controla un motor paso a paso de posicionamiento del televisor, así como su respectivo sensor de posición de llegada. Permite la regulación vía software de la velocidad del motor y cuenta con una gama de registros de estado y programación que permiten usarlo en un sistema controlado por programas de alto nivel.

Características

- Frecuencia máxima de operación: ≈1KHz.
- Entradas y salidas compatibles con LS-TTL.
- Control completo de tres televisores por integrado.
- Empaquetado: DIP40.
- Completamente programable.
- Registros de estado.

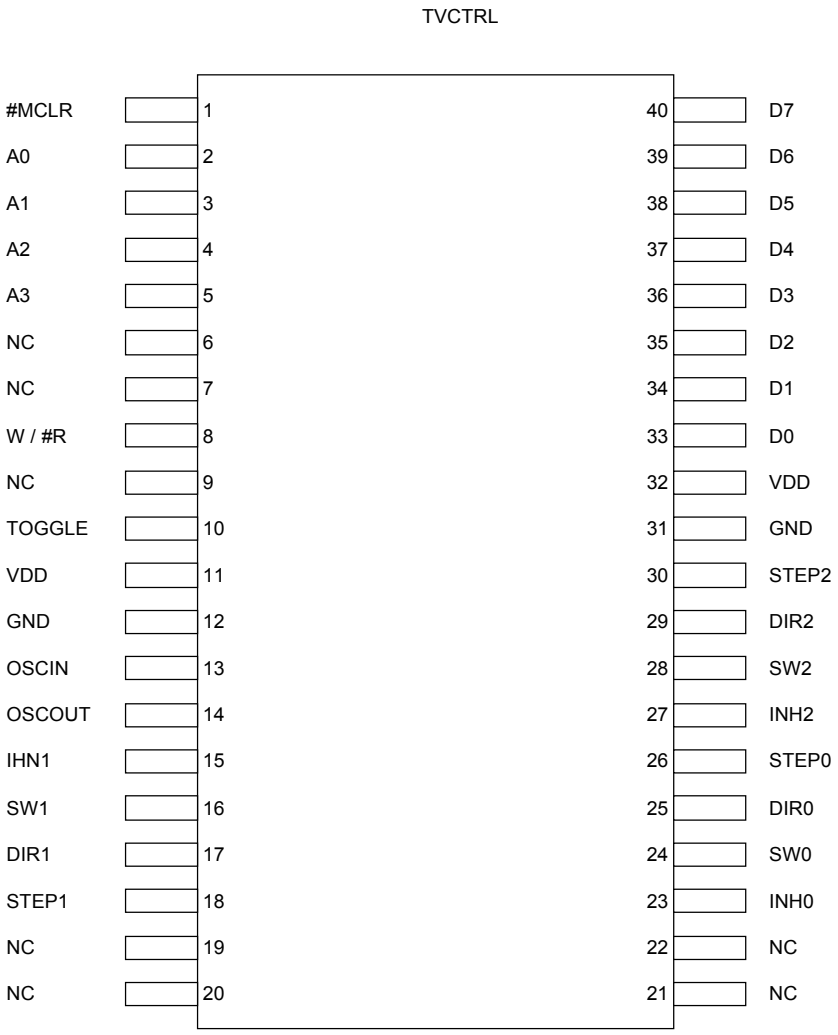

Figura A.3 Diagrama de pines del controlador de televisores

Configuración de Pines

Tabla A.2 Configuración de pines del controlador de televisores

Nombre del PIN	I/O	No.	Descripción
A[0..3]	I	2,3,4,5	Bus de direcciones
D[0..7]	I/O	33,34,35,36, 37,38,39,40	Bus de datos bidireccional
W / #R	I	8	Selección de lectura - escritura
IHN[0..2]	O	15,23,27	Deshabilitación de los motores Paso a Paso
SW[0..2]	I	16,24,28	Sensor de posición de los televisores
#STEP[0..2]	O	18,26,30	Señal pulsante para los pasos del motor
DIR[0..2]	O	17,25,29	Dirección de rotación del motor
TOGGLE	O	10	Señal cuadrada de sincronización
#MCLR	I	1	Reseteo general del controlador
OSCIN	I	13	Entrada de reloj
OSCOUT	O	14	Salida de reloj
VDD	I	11,32	Fuente alimentación +5V
GND	I	12,31	Tierra

Diagrama de bloques

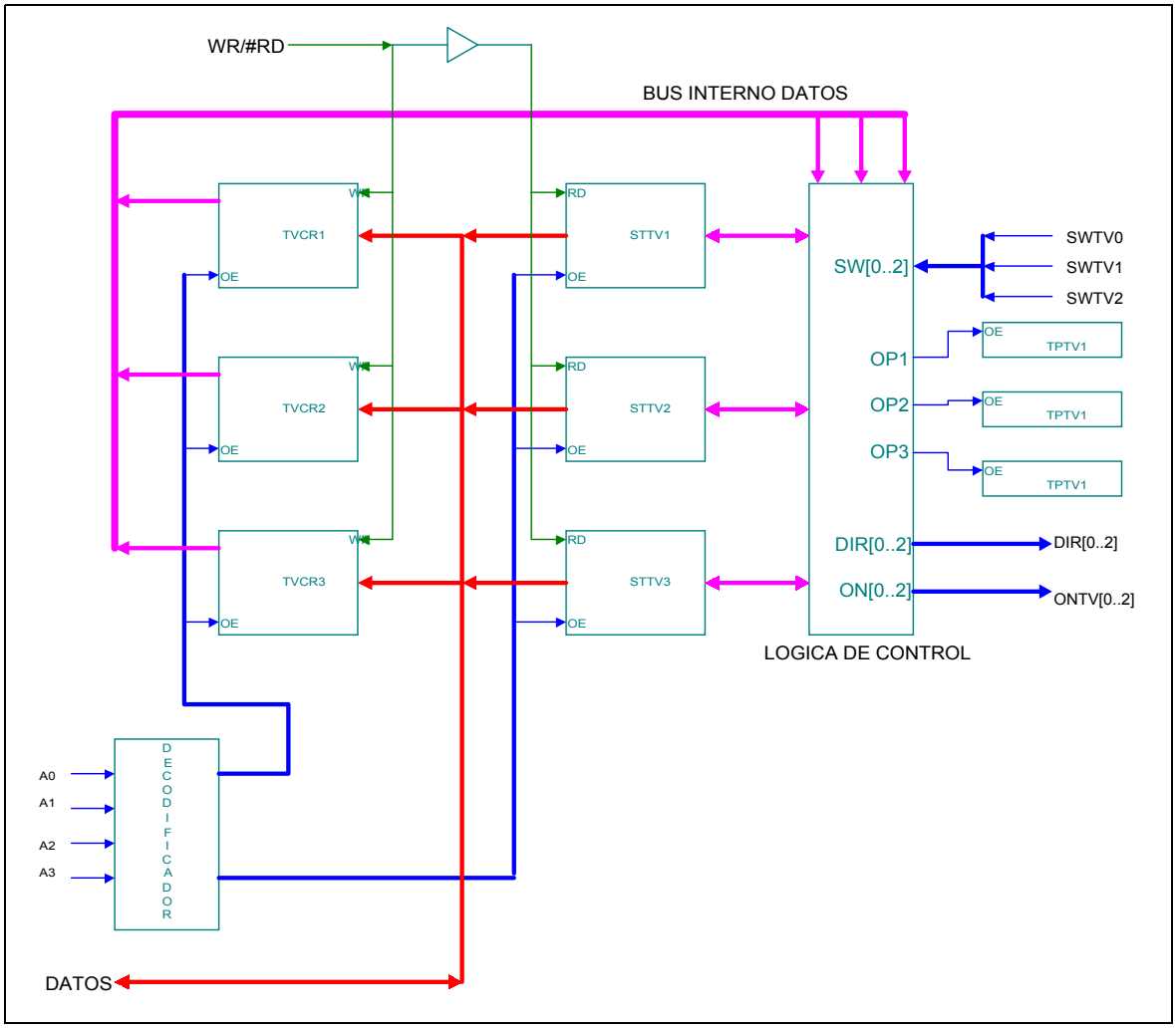

Figura A.4 Diagrama de bloques del controlador de televisores

Descripción Funcional

En la figura 3, se muestra el diagrama de flujo del funcionamiento interno del controlador de televisores. Cuando se conecta la energía al controlador, este inicia con el ciclo que determina si el registro de programación fue escrito; cuando este registro es escrito empieza el ciclo para realizar una de las cuatro funciones (subir, bajar, encender y apagar). El sistema está diseñado para determinar el estado actual de cada televisor y de cumplir con las condiciones se realiza la función deseada por el usuario.

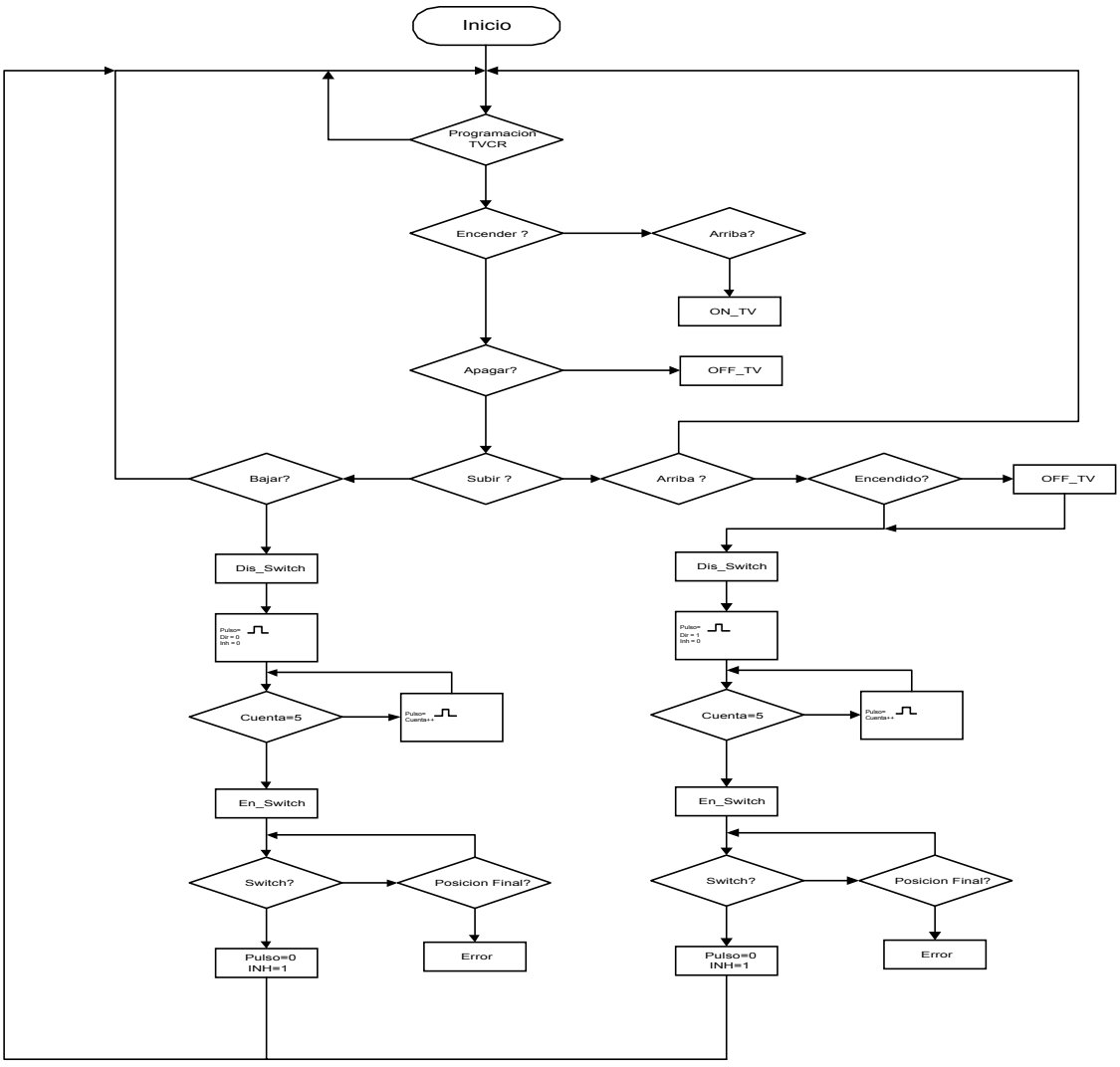

Figura A.5 Diagrama de flujo del controlador de televisores

A.2.3 Controlador de PWM programable

Descripción general

Este dispositivo es un controlador de modulación por ancho de pulso, con la capacidad de ser programado. Tiene un bus de datos de 8 bits, y una línea de selección de programación.

Por medio de este dispositivo se pueden controlar una gran variedad de hardware entre los que se encuentran control de intensidad de luces o bien control de intensidad de pantalla LCD.

Características

- Frecuencia máxima de operación: ≈250Hz.
- Entradas y salidas compatibles con LS-TTL.
- Control completo del ciclo de trabajo (0-100%).
- Empaquetado: DIP28.
- Completamente programable.

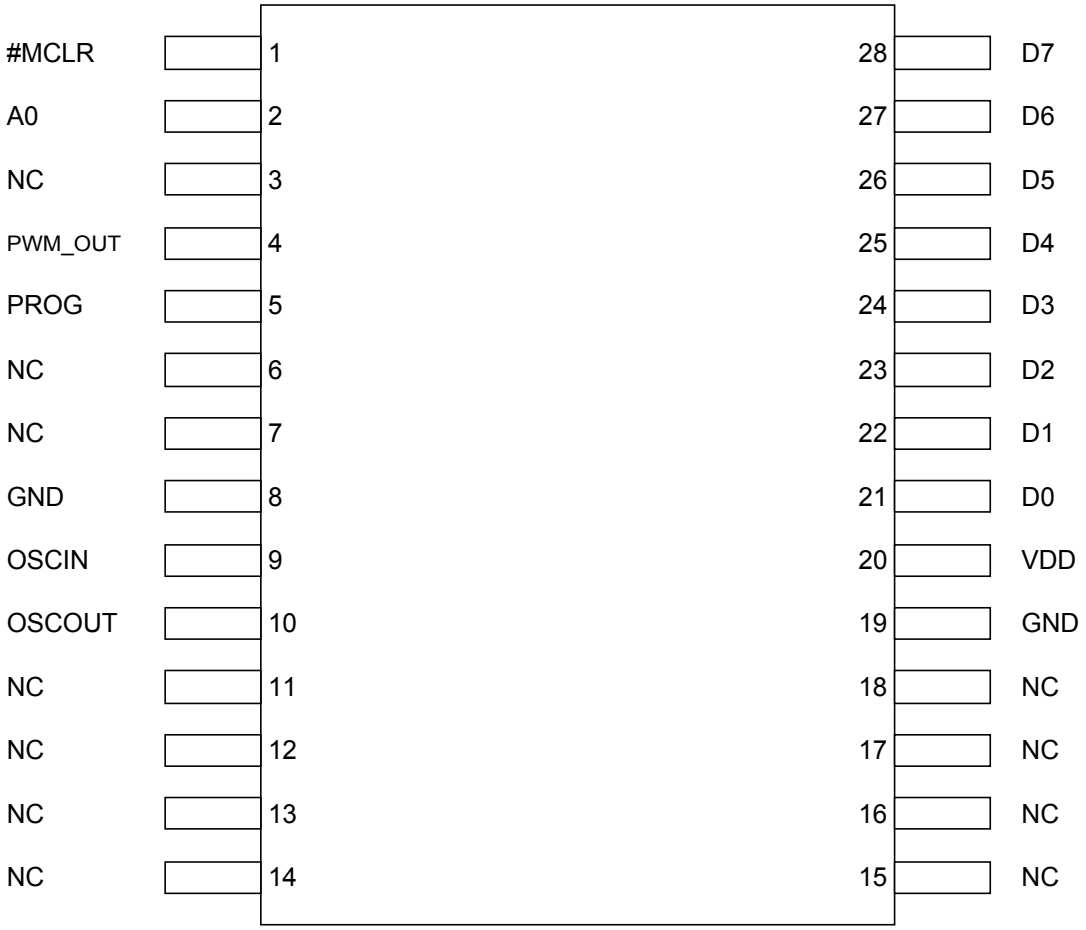

Figura A.6 Diagrama de pines del controlador de PWM

Configuración de Pines

Tabla A.3 Configuración de pines del controlador de PWM

Nombre del PIN	I/O	No.	Descripción
D[0..7]	I	21,22,23,24, 25,26,27,28	Bus de datos
PROG	I	5	Selección de estado Programación
PWM_OUT	O	4	Señal PWM de salida
#MCLR	I	1	Reseteo general del controlador
OSCIN	I	9	Entrada de reloj
OSCOUT	O	10	Salida de reloj
VDD	I	20	Fuente alimentación +5V
GND	I	8,19	Tierra

Diagrama de bloques

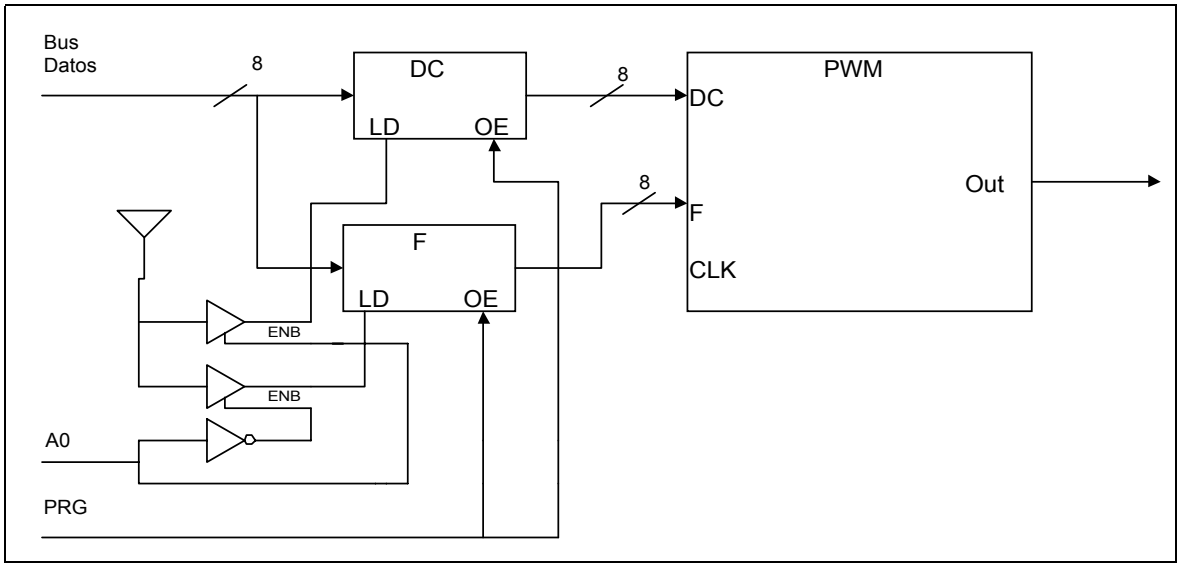

Figura A.7 Diagrama de bloques del controlador de PWM

A.2.4 Decodificador de direcciones

Descripción general

Este dispositivo decodifica el bus de direcciones del sistema y los convierte en las señales de habilitación de los diferentes controladores. Además genera las líneas para la escritura y lectura de los controladores que comparten el mismo bus de datos.

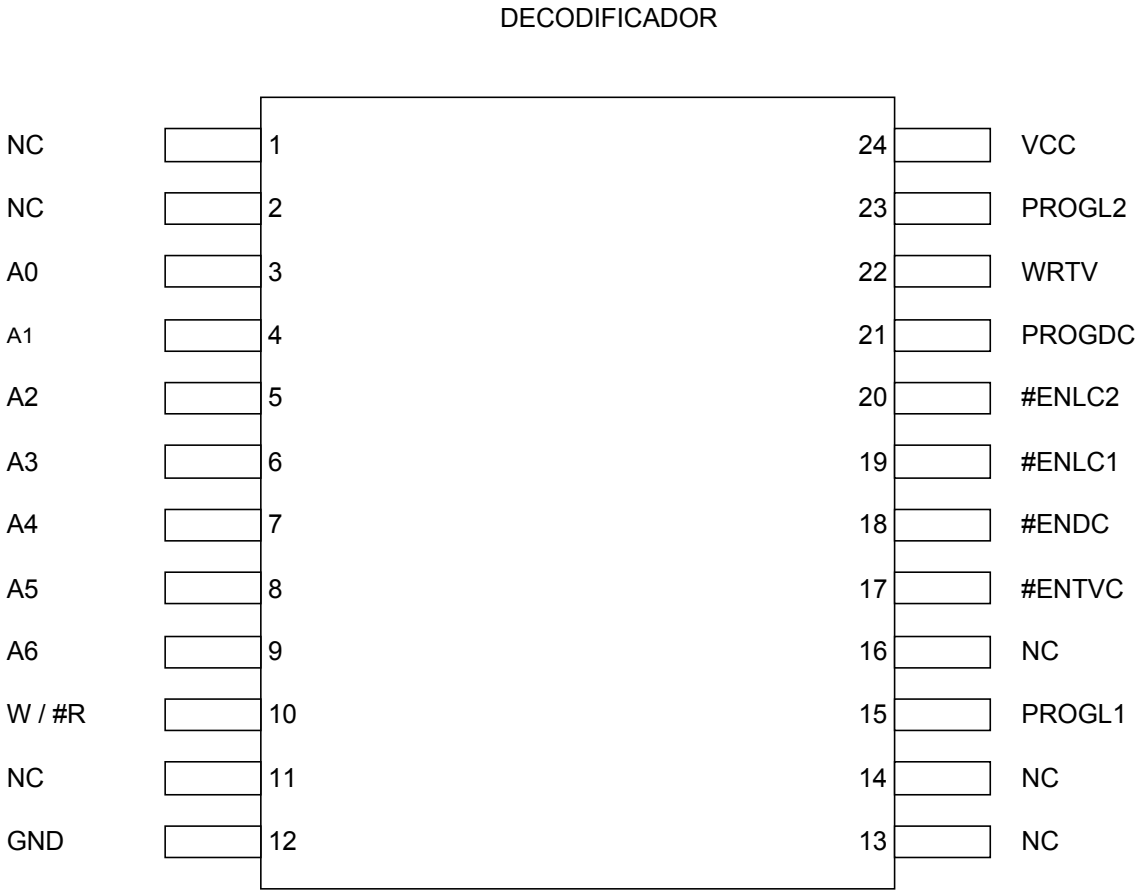

Figura A.8 Diagrama de pines del decodificador de direcciones

Configuración de Pines

Tabla A.4 Configuración de pines del decodificador de direcciones

Nombre del PIN	I/O	No.	Descripción
A[0..6]	I	3,4,5,6,7,8,9	Bus de direcciones
W / #R	I	10	Señal de lectura / escritura
#ENLC[1..2]	O	19,20	Habilitación controlador luz1 y luz2
#ENTVC	O	17	Habilitación controlador televisores
#ENDC	O	18	Habilitación controlador de brillo
PROGL[1..2]	O	15,23	Señal de lectura / escritura de control luces
PROGDC	O	21	Señal de lectura / escritura de control brillo
WRTV	O	22	Señal de lectura / escritura de control televisores
VCC	I	24	Fuente alimentación +5V
GND	I	12	Tierra

A.2.5 Diagrama de conexión general

En la siguiente figura, se muestra el diagrama completo de conexión del sistema de control implementado. En este diagrama se observa todos los controladores mencionados anteriormente y sus misceláneos (decodificadores, resistencias, capacitores, drivers, optoacopladores, etc).

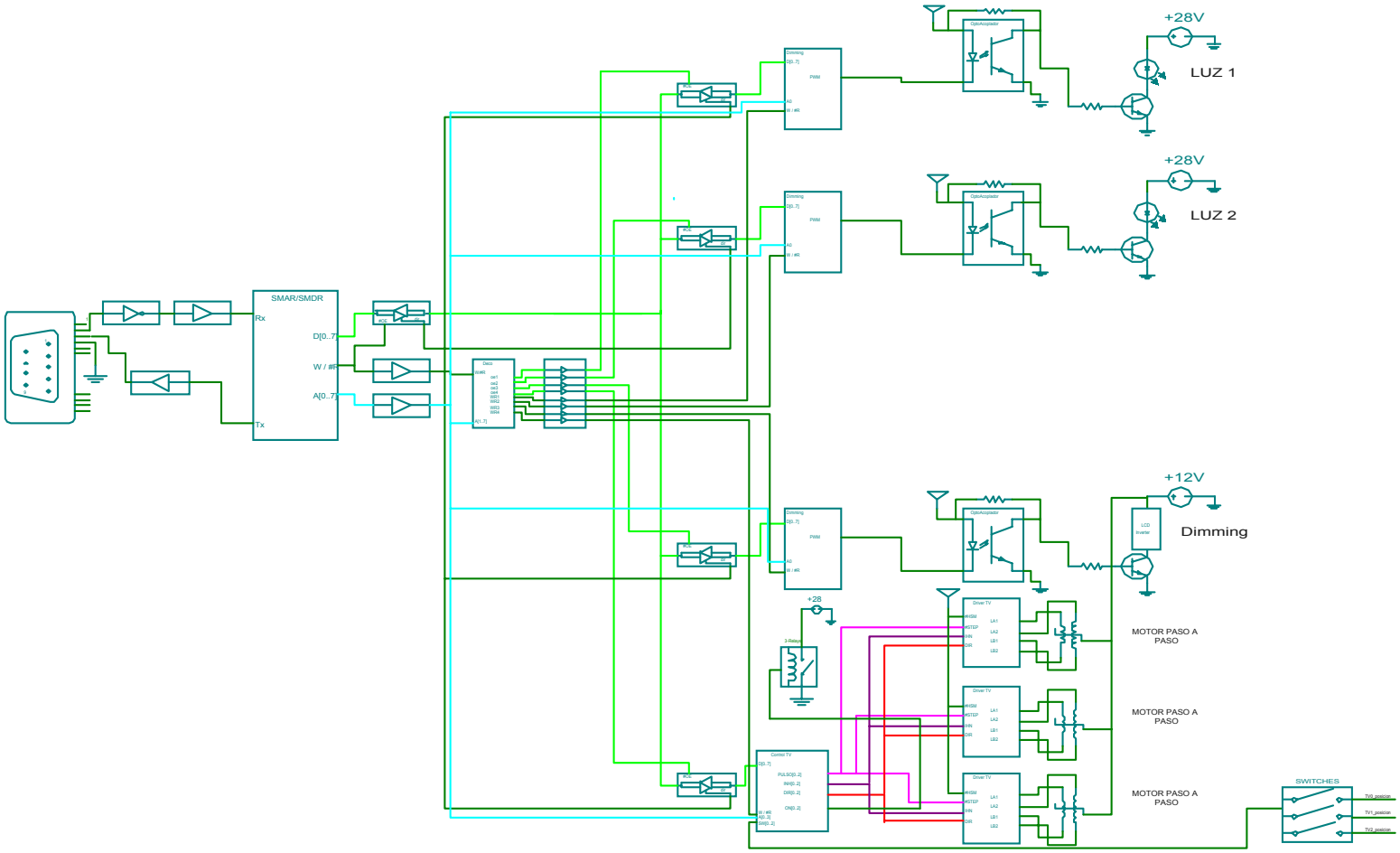

Figura A.9 Diagrama de conexión general del sistema de control

Anexos

Datos técnicos

2 Installation of the LSH7727-10

2.1 Connection Diagram

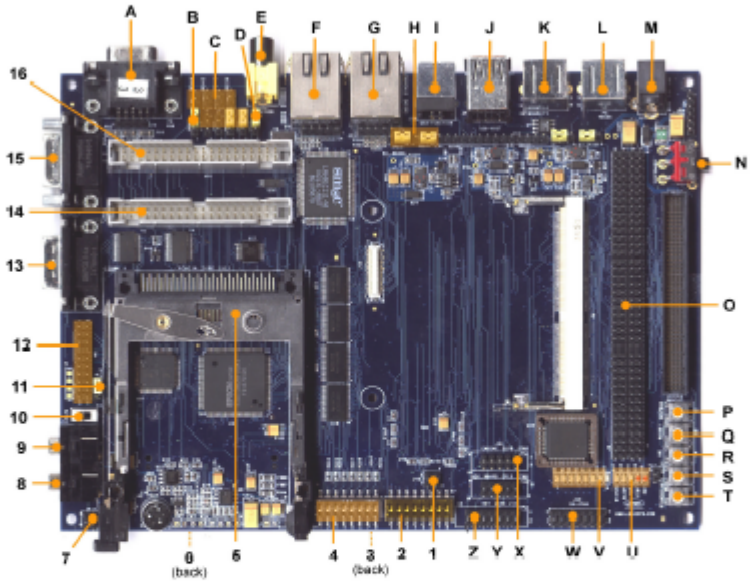

Figure 2.A – Connection Diagram for the LSH7727-10 Application Kit

A) VGA Video output. A standard PC-style SVGA monitor can be plugged into this jack to use as a video display.
B) Card Engine Serial port flow control bypass.
C) 8 jumpers controlling various LCD lines (see schematics).
D) Backlight power select jumper.
E) Composite Video.
F) Debug Board Ethernet. This jack is connected to the Ethernet controller located on the debug board.
G) Card Engine Ethernet. This jack is connected to the Ethernet controller located on the card engine.
H) USB Function/Host selection jumpers (see schematics).
I) USB Function.

J) USB Host.
K) Keyboard. A PS/2 style keyboard can be plugged in to this jack.
L) Mouse. This is a PS/2-style mouse connector.
M) Power. The included power dongle plugs in to this jack.
N) Power Switch. The on and off positions are clearly labeled on the circuit board.
O) Debug Headers J2 and J3 (see schematics).
P) Suspend Button. This button toggles suspend mode on the card engine.
Q) Reset. This button toggles the reset circuitry for the card engine.
R) NMI. This button is wired to the Non-Maskable Interrupt line on the card engine.
S) Master Reset. This button activates the reset line on both boards.
T) Standby. This button puts the system in standby mode.
U) Debug LEDs. These 8 LEDs are used for debugging purposes.
V) Status LEDs indicate the run state of the processor and application board.
W) SH-3 JTAG header. This header is the JTAG interface for the 7727.
X) Card Engine CPLD JTAG header.
Y) Application board CPLD JTAG header.
Z) H8 Tiny JTAG header. This header is the JTAG interface for the H8 Tiny processor
1) I2C header.
2) CPLD spare GPIO pins.
3) MMC connector (on back of board).
4) Operation control switches, see Boot Selection switches (dip switches at 4)
5) PCMCIA Slot.
6) CompactFlash Slot (back side of board).
7) AFE interface (see schematics JP10).
8) Headphone Slot. This is a stereo jack for standard 1/8" mini-stereo.
9) Microphone Slot. This is a 1/8" mono jack for an external microphone.
10) Mute Switch. The muted position is toward the center of the board.
11) On-board microphone en/disable.
12) Parallel port connector.
13) COM 0 Serial connector.
14) Epson LCD output. This is the LCD output from the Epson graphics chip on the debug board.
15) Debug Serial Connector.
16) SH-3 LCD output. This is the LCD output from the SH-3 7727 on the card engine.

2.2 Boot Selection switches (dip switches at 4)

1 On – boot from off-board EEPROM.
Off – boot from on-board flash memory.
2 On – Boot into Windows CE image.
Off – Boot into bootloader (connect serial terminal).
3 On – Use serial as boot transfer medium.
Off – Use Ethernet as boot transfer medium.
4 On – Use on-board Ethernet for debug.
Off – Use application board Ethernet for debug.
5 On – Use 7727 SCIF serial port for debug serial.
Off – Use application board UART for debug serial.
6 On – Use the Epson video chip for graphics.
Off – Use the 7727 LCD controller for graphics.
7 On – Epson graphics chip outputs to LCD.
Off – Epson graphics chip outputs to CRT/VGA.
8 On – 2 USB Host ports. (this functionality not supported in this release)
Off – 1 USB Host port, 1 USB Function port. (n/a)

Logic Product Development *Setup and User´s Guide.* CD Rom. Minneapolis, USA: Logicpd, 2002.

Esta información fue suministrada por el fabricante por la compra de la plataforma de desarrollo mostrada.