Sistema Embebido de Monitoreo Web

Jorge Osio; Juan Czerwien; Walter Aróztegui; Jose Rapallini; Antonio Adrián Quijano
Centro de Técnicas Analógico – Digitales (CeTAD)
Facultad de Ingeniería – Universidad Nacional de La Plata
La Plata, Argentina
josio@gioia.ing.unlp.edu.ar; jczerwie@gioia.ing.unlp.edu.ar; josrap@ing.unlp.edu.ar; Quijano@ing.unlp.edu.ar

Abstract— **En este trabajo se presenta un sistema embebido que captura imágenes para ser preprocesadas y enviadas por una interfaz ethernet para su análisis remoto desde Internet. El prototipo funcional está formado por dos modulos, una micro-cámara y un dispositivo microprocesador que toma las imágenes y efectúan un primer tratamiento de los datos (reducción de ruido), almacenamiento y transmisión. Este sistema embebido es una parte del proyecto de ´Análisis de imágenes a distancia´, para el cual se diseñaron un conjunto de herramientas informáticas específicas (no presentadas en este artículo) que se ejecutan en las terminales de visualización.**

Palabras Clave: Sistemas Embebidos; Transmisión de Imágenes; Redes Ethernet

I. INTRODUCCIÓN

El desarrollo de un sistema embebido "micro-cámara IP" prevé la toma de datos de imágenes para su visualización mediante una IP fija que permite acceder por red, (LAN / WAN), al dispositivo. Las imágenes tomadas desde la cámara en tiempo real se pueden acceder desde cualquier lugar con un equipo que posea conectividad LAN o WAN.

uno de ellos se encarga de adquirir datos de imágenes los cuales son transmitidos por una red hacia una estación de trabajo denominada (control / servidor), que realiza funciones específicas: controlar y supervisar la transmisión de datos, almacenamiento y pre-procesamiento de las imágenes y como servidor hacia una red, la cual permite a través de cualquier terminal realizar la visualización, análisis y procesamiento digital de imágenes (Figura 1).

II. DETALLE DE LOS COMPONENTES

En función de los requerimientos técnicos del sistema propuesto se consideran para la realización del prototipo funcional los siguientes componentes:

A. Microcámara digiral:

Para su realización se seleccionó la micro-cámara VGA - CAM 100. Esta microcámara posee una serie de comandos que facilitan la sincronización y la obtención de la información adquirida, permite varios formatos de obtención de imágenes, entre los cuales se encuentra el formato de compresión JPEG.

El modulo VGA se vincula al exterior por una

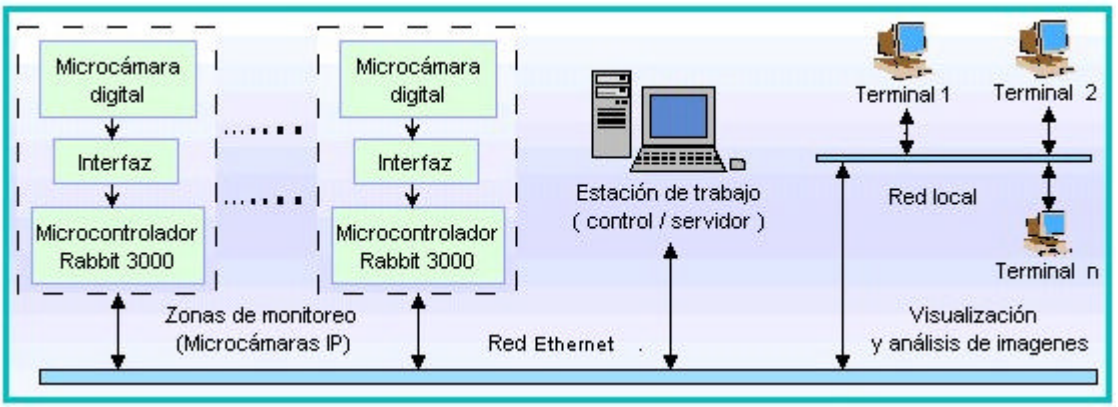

Figura 1. Diagrama en bloques general del desarrollo

Red de aplicación

Cada módulo de monitoreo, (que se encuentra alojado en diferentes zonas), conforma una microcámara IP que se compone de tres bloques fundamentales, la microcámara de video VGA, la interfaz digital y un microcontrolador. Cada

comunicación serie con nivel CMOS de 3.3 V. Un sencillo protocolo permite realizar la interfaz con el módulo microprocesador. En la Figura 2 se muestra la VGA -CAM.

La VGA -CAM puede adicionarse como accesorio a un dispositivo controlador desde el que puede controlarse la

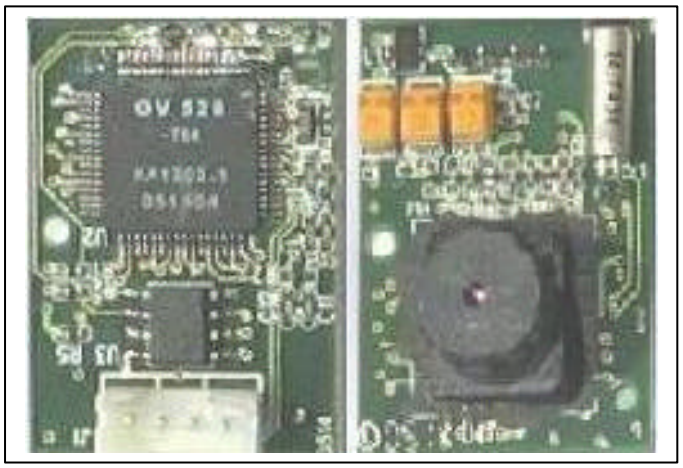

Figura 2. Imagen de la VGA – CAM 100

salida mediante comandos, seleccionando la captura de imágenes (fotos) a máxima resolución, en color o blanco y negro, representando la imagen en un solo cuadro. Cada cuadro (foto) es comprimido en formato Standard JPEG o en diferentes resoluciones en niveles de grises y transferido hacia el controlador (Host).

El Módulo VGA-CAM está formado por 3 bloques fundamentales, el "Sensor de Imágenes", el de "Compresión, Control y Transferencia" y el de "Memoria de Programa". El diagrama en bloques de la Cámara se muestra en la Figura 3 y se describe a continuación:

1) Bloque sensor de Imagenes

El módulo sensor de imágenes, consiste en un chip que es una cámara digital color VGA (OV7640/8) de la empresa OmniVision que posee una interfaz de 8-bits YCbCr.

2) Bloque de Compresión, Control y Transferencia (OV528)

Se compone de un JPEG CODEC (OV528), embebido en un chip controlador, que en conjunto actúan como puente serie, entre el sensor de la cámara y la salida del sistema. Es el encargado de transferir, comprimir y controlar la información (imagen) desde el propio chip de la cámara hacia un dispositivo externo. Los datos de video, en forma progresiva, son transmitidos en el formato de 8-bits YCbCr 422. La sincronización de la interfaz de la cámara, con los datos de la entrada de video y el desempeño de los modos de resolución deseados, tales como los modos de conversión de color, son configurados mediante comandos a través del bus serie, desde el dispositivo host.

3) Bloque de Memoria de Programa (EEPROM)

Consiste en una memoria de programa de tipo serial, la cual esta dotada de comandos con los cuales fácilmente se puede controlar mediante una interfaz desde un host externo. Su finalidad es la de almacenar el programa necesario para realizar la configuración de funcionamiento de la microcámara, como el tipo de compresión de imágenes, velocidad de transmisión y conversión a color o a niveles de grises.

Cumple dos funciones, permite capturar imágenes instantáneas (fotos) o actuar como una cámara filmadora de video realizando tomas de imágenes, las cuales pueden o no estar en movimiento. En ambos casos se puede realizar la adquisición en blanco y negro (niveles de grises), en color y también para éste último, con formato de compresión standard JPEG. Permite que las fotos puedan adquirirse a la máxima resolución tanto en color o en blanco y negro.

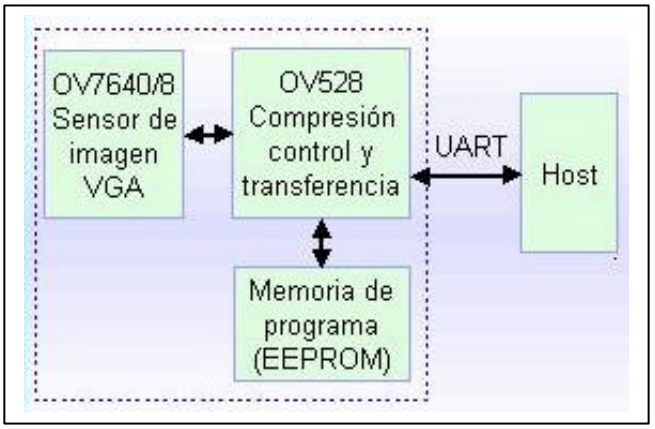

Figura 3. Bloques del Módulo VGA

B. Dispositivo Microprocesador

El dispositivo seleccionado es el Microcontrolador Rabbit 3000, por sus prestaciones en cuanto a velocidad de procesamiento, a espacio de memoria RAM y de ROM y a la gran cantidad de Librerías que provee el fabricante sobre TCP/IP. También posee módulos seriales eficientes para la comunicación con la cámara. Además, su sistema de desarrollo contiene la posibilidad de utilizar un ambiente de programación que permite incorporar funciones desde lenguaje ´C´ (Dynamic C), útiles para utilizarlo como ´Estación de Trabajo´ (Figura 1).

Características Principales:

- Microcontrolador de bus accesible.
- Posee un bus de direcciones de 20 bits, esto implica 1 MB de direccionamiento posible.
- Posee un bus de datos de 8 bits, lo que implica menos líneas de conexión.
- Su arquitectura es compatible con el Z-80, ello implica una gran ventaja debido a que resulta de una arquitectura conocida, con gran cantidad de software disponible, poderosos modos de direccionamiento acompañado de instrucciones lógicas y aritméticas de 8 y 16 bits y manejo de bits individuales.
- Es un microcontrolador, por lo tanto posee USARTs, timers, puertos de I/O (ports), WDT (Watchdog Timer), RTC (Real Time Clock). La glue-logic es interna, posee salidas "Chip Select", que elimina lógica externa para conexión de memorias, pudiendo generar de 0 a 4 wait-states.

- En el espacio de I/O, tiene salidas " I/O Strobes" suprimiendo lógica externa para la conexión de periféricos de I/O, pudiendo generar de 0 a 15 wait-states.
- Puede cargar el programa de arranque desde un puerto serie (serial boot).
- Posee un slave port, paralelo, que facilita la interconexión de procesadores, incluso puede bootear del slave port, siendo esclavo (sin memoria) de otro procesador.
- Su performance es comparable a la de muchos DSP. El kit de desarrollo se encuentra acompañado por un potente entorno de programación C con Librerías de funciones, soporte multitarea, ICD (In-Circuit Debugging) y TCP/IP.
- Posee 40 entradas y salidas paralelo (compartidas con los port serie). Algunas de las salidas pueden ser sincronizadas con los timers, lo que permite generar pulsos de precisión.
- Posee 6 ports serie, cinco de los cuales pueden operar en modo sincrónico/asincrónico, uno de ellos solamente en modo asincrónico. Las velocidades llegan a 1/32 de la frecuencia de clock para modo sincrónico y 1/6 para modo sincrónico (1/4 si la fuente de clock es interna).
- El oscilador principal de reloj utiliza un cristal o resonador externo. Las frecuencias típicas cubren un rango de 1,8 a 50MHz.

El microcontrolador seleccionado, es un diseño que evolucionó del dispositivo Z80/Z180 a un micro moderno, conservando los mismos registros, pero con un set de instrucciones extendido, que facilitan y refuerzan la potencia del procesador para manejar punteros y variables en 16-bits [2], [3] y [4]. Además la combinación Rabbit - con el software Dynamic C que forma parte del sistema de desarrollo, permite agregar bibliotecas de funciones con soporte de aritmética de coma flotante, característica importante para la productividad en sistemas pequeños, ya que resulta fácil resolver muchos problemas de programación de alto grado de complejidad. El grupo de instrucciones permite procesar rápidamente números en formato de coma flotante y por supuesto también enteros.

El procesador no posee memoria interna, la misma se adiciona mediante los *core-modules*, que son módulos pre-armados con una cierta capacidad de memoria ya instalada, pines para conexión con el mundo exterior, y según el modelo, controlador Ethernet y jack RJ-45 [5]

En alguna aplicación en especial, que demande mayor capacidad de memoria, simplemente se reemplaza el módulo por otro de mayor capacidad, sin necesidad de portar el diseño a un controlador de gama más alta.

La característica que hace sobresaliente a los microcontroladores Rabbit radica en su entorno de programación denominado Dynamic C [5] el cual resulta compatible con ANSI C. Además, Posee una gran cantidad de librerías, entre las que se destacan aquellas que permiten mantener un stack TCP/IP completo, con servidor FTP, HTTP y POP3, además no requiere demasiado espacio de memoria para las mismas.

El compilador Dynamic C es básicamente un compilador en C con algunas instrucciones y conceptos de alto nivel. A continuación se describen los conceptos e instrucciones, propios de Dynamic C[5] más usados:

- Las **Costates (Co-sentencias)** permiten implementar un sistema de múltiples tareas. Estas co-sentencias son cooperativas ya que su ejecución debe suspenderse y reanudarse de forma voluntaria. El cuerpo de una costate es una tarea (lista ordenada de operaciones a realizar). Cada costate tiene su propio puntero de programa para determinar que item en la lista debe ejecutarse. Al inicializar el sistema multitareas el puntero se le asigna a la primera instrucción
- Las **Cofunctions (Co-functions)** son similares a las constantes, pero tiene un formato similar a las funciones ya que se les puede pasar argumentos y devuelven un resultado. Estas son llamadas desde una costate para ejecutar una determinada función.
- Las **Sentencias de control** (control statements) se utilizan para distribuir el procesamiento entre las distintas tareas.

III. DESCRIPCIÓN DEL PROGRAMA DE APLICACIÓN

Organigrama del Programa de Aplicación

El programa de Aplicación se puede diagramar en 4 Partes Principales:

1) *Definición y Configuración de librerías:*
   a) *Librerías TCP/IP y Ethernet*
   b) *Librerías HTTP*
2) *Configuración de Puertos y Modulos*
3) *Funciones del Bucle Principal*

Definición y Configuración de Librerías:

La definición de las librerías es requerida para utilizar una IP fija en el programa y la interfaz Ethernet por defecto.

En las Librerías TCP/IP se debe configurar:

- Dirección de IP
- Mascara de Subred
- Puerta de Enlace
- Nombre del servidor

También se deben definir y configurar las librerías de HTTP. En las mismas se especifica el tamaño de los buffers (buffer máximo de HTTP; Máximo Socket TCP; Nombre máximo de HTTP; nombre máximo SSPEC; Máximo Socket UDP) y las definiciones Generales como máximo tamaño de paquetes, máxima cantidad de fallas y máximo retardo de frame.

Configuración de Puertos y Modulos

La placa, que se encarga de configurar los puertos, tanto el puerto serial que se comunica con la CAM, como los puertos que manejan los leds indicadores. Además, se inicializa la red y se inicializa el servidor

Entre las variables a inicializar se encuentran; estado de la CAM, contador de fallas, imagen en cache y contador de sincronismo.

Funciones del Loop Principal

Luego de toda la configuración necesaria para inicializar el sistema, se inicializa el mismo mediante un While(1).

Las funciones dentro de este while son llamadas mediante constantes *(Co-sentencias)* que permiten implementar un sistema de múltiples tareas.

La primer constante es:

```
costate { wfd MasterCam();
        }
```

Contempla la comunicación con la micro-cámara serial. La sentencia de control wfd significa waitfordone y espera a que se termine de ejecutar la función.

La segunda constate es:

```
costate { http_handler();
        }
```

Se encarga de manejar el servidor web, con la posibilidad de refresco automático de la página web que permite acceder a la cámara.

Descripción de la función MasterCam():

La función MasterCam() implementa la máquina de estados que permite al microcontrolador tomar el control de la cámara.

Esta función está implementada mediante un switch(StatusCam), el cual tiene varias opciones (case) de envío de comandos (formados por varios parámetros). La Tabla I muestra todos los comandos [6], (reconocidos por la micro-cámara), usados en este sistema.

El valor de StatusCam permite acceder a cada uno de los cases, inicialmente este valor es 0x00; que equivale al comando de sincronización de la cámara SYNC (case CAM_SYNC).

Luego el comando se envía mediante la función SendCamCommand() y se espera la respuesta de la CAM. Una vez recibida dicha respuesta se envía el comando "ACK", confirmando que la respuesta anterior fue exitosa. Antes de salir del case se guarda en StatusCam el valor de CAM_SETUP. Lo que permite enviar los comandos de configuración "Initial" y "Set Package Size".

Luego se accede al case CAM_KEEPALIVE, que envía el comando de sincronismo "SINC" y el posterior "ACK" para verificar que la cámara se mantiene en sincronismo. Y antes de salir del case se carga la variable StatusCAM con CAM_SNAPSHOT.

Por lo tanto el paso siguiente es acceder al case CAM_SNAPSHOT. En donde se envía el comando "Snapshot", (que almacena una imagen en el buffer de la CAM), Seguido del comando "Get Picture" que permite recibir la imagen, en el servidor implementado por el procesador, mediante el bus serial. Por último, para recibir los paquetes de datos se enviarán tantos comandos "ACK" como paquetes de datos formen la imagen.

Nombre	**Comandos de Control Utilizados en el Sistema**	
	Parámetros del comando	***Descripción***
Initial	AA-01-00-07-03-01	Configura el tipo de Imagen a Obtener
Get Picture	AA-04-05-00-00-00	Envía la Imagen por el bus serial
Snapshot	AA-05-00-00-00-00	Captura una Imagen
Set Package Size	AA-06-08-00-02-00	Configura el tamaño de paquete
SYNC	AA-0D-00-00-00-00	Sincronismo de la CAM
ACK	AA-0E-0D-01-00-00	Confirmación de operación exitosa
Reset	AA-08-00-00-00-00	Resetea la VGA-CAM

TABLA I. COMANDOS DE CONTROL DE LA VGA-CAM UTILIZADOS

IV. IMPLEMENTACIÓN

En la Figura 4 se muestra el sistema completo formado por el Kit Rabbit 3720 y la micro-cámara CAM-VGA100.

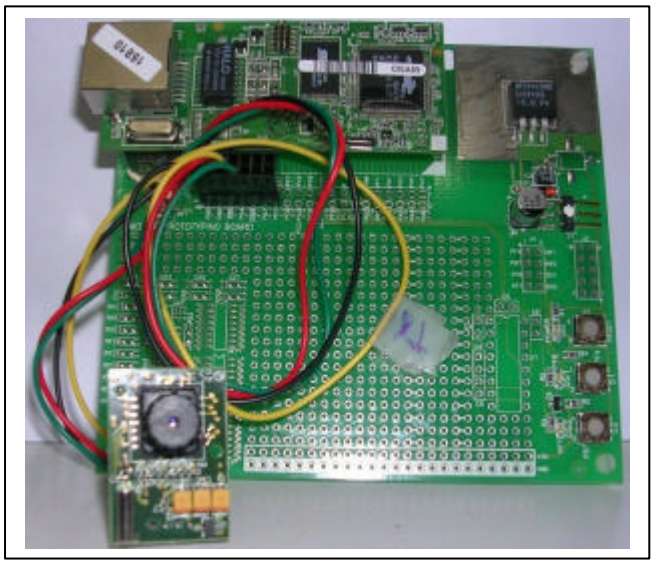

Figura 4. Sistema de monitéo web.

CONCLUSIONES

Este diseño, se caracteriza por la confiabilidad que provee el sistema de procesamiento empleado, que no solo es potente en cuanto a la velocidad de procesamiento, sino, también por la capacidad de almacenamiento, que permite implementar desde un servidor hasta una máquina de estados (para el control vía serial de la micro-cámara). Esta implementación permite tener un prototipo funcional para realizar todas las pruebas de los programas que se ejecutan en las terminales de visualización y análisis a través de la red y determinan las condiciones para la realización del sistema embebido propuesto.

REFERENCIAS

[1] Area SX s.r.l, http://www.areasx.com, acceso: octubre 2006.

[2] M. Morris Mano, "Arquitectura de Computadoras", Prentice Hall,1999.

[3] M. Morris Mano, "Fundamentos de Diseño Lógico y Computadoras", Prentice Hall, 2003.

[4]William Stallings, "Organización y Arquitectura de Computadores ", Prentice Hall, 2006.

[5] Srgio R. Caprile, "Desarrollo con Procesadores y Módulos Rabbit", GAMA Producción Gráficas, 2005.

[6] Round Solutions GmbH & Co KG, "CAM-VGA 100 User Manual", http://www.roundsolutions.com, acceso: septiembre del 2006.