Revista Elektron
ISSN: 2525-0159
revista.elektron@fi.uba.ar
Universidad de Buenos Aires
Argentina

Implementación de un adaptador de video por software en microcontrolador de doble núcleo
Revista Elektron, vol. 4, núm. 1, 2020, -Junio, pp. 21-26
Universidad de Buenos Aires
Argentina

elektron ámbar

Revista elektron, Vol. 4, No. 1, pp. 21-26 (2020)
https://doi.org/10.37537/rev.elektron.4.1.91.2020

Original Article

Implementación de un Adaptador de Video por Software en Microcontrolador de Doble Núcleo

Implementation of a Software Video Adapter in Dual Core Microcontroller

Santiago Germino
Laboratorio de Sistemas Embebidos
Facultad de Ingeniería - UBA
Buenos Aires, Argentina
sgermino@retro-ciaa.com

Recibido: 30/09/19; Aceptado: 13/01/20

***Resumen*—En este artículo se presentan las consideraciones y técnicas utilizadas para el diseño e implementación por software de un adaptador de video de alta performance en un kit de desarrollo con microcontrolador de doble núcleo. Se utiliza un núcleo exclusivamente para tal fin mientras que el segundo ejecuta la aplicación de forma concurrente.**

Palabras clave: computación gráfica; adaptador de video por software; doble núcleo.

***Abstract*—This article presents the considerations and techniques used for the design and software implementation of a high-performance video adapter in a development kit with a dual-core microcontroller, dedicating a core exclusively for that purpose while the second one is used to execute the application.**

Keywords: computer graphics; software video adapter; dual core.

I. Introducción

Un adaptador de video integra memoria para contener al menos una pantalla completa o cuadro de animación, capacidad de generar una señal de video y un puerto para conectar dicha señal a un monitor compatible [1].

Su implementación por software requiere un uso intensivo de recursos de memoria, manejo de puertos de E/S y temporizado preciso. Estas tareas pueden exigir hasta el 95 % del tiempo de un núcleo del microcontrolador. Por este motivo, un microcontrolador con al menos dos núcleos permite una implementación eficaz.

El kit de desarrollo educativo EDU-CIAA-NXP [2] (Fig. 1) pertenece al proyecto CIAA o Computadora Industrial Abierta Argentina. Integra un microcontrolador NXP LPC4337JBD144 de 204 MHz, un núcleo ARM Cortex-M0, un núcleo ARM Cortex-M4F, 136 KiB de SRAM y 1 MiB de Flash. La memoria SRAM y Flash es compartida por ambos núcleos [3].

El proyecto RETRO-CIAA [4] es una placa de expansión o «poncho» (Fig. 2) que suma conectividad y un nuevo firmware en donde se implementó un adaptador de video de alta prestación corriendo en el núcleo Cortex-M0 de la EDU-CIAA-NXP.

II. Motivación y objetivos

El uso de una interfaz gráfica de usuario está ampliamente extendido en sistemas embebidos. Aun así, debido a las restricciones de memoria, procesamiento o consumo de energía,

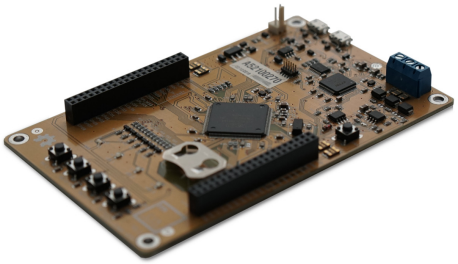

Fig. 1. Kit de desarrollo EDU-CIAA-NXP.

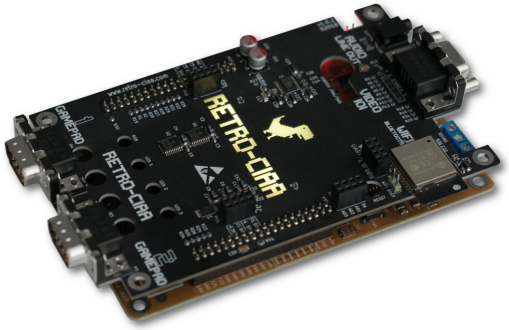

Fig. 2. Placa de expansión RETRO-CIAA sobre EDU-CIAA-NXP.

no siempre es factible utilizar librerías gráficas no diseñadas u optimizadas para funcionar con dichas limitaciones.

La computación gráfica trata sobre las técnicas y algoritmos utilizados para generar gráficos por computadora. Promover su conocimiento y práctica seria imprescindible para lograr una eficiente implementación de funciones gráficas en sistemas con escasos recursos.

El kit EDU-CIAA-NXP se utiliza como herramienta educativa en diferentes secundarios, terciarios y carreras universitarias. A la fecha, mas de 2500 unidades se encuentran en poder de instituciones y particulares. La mayoría de proyectos, ejemplos y ejercicios disponibles [5], [6] solo utilizan el núcleo Cortex-M4F mientras que el Cortex-M0 se mantiene continuamente en estado de reset.

Estas observaciones motivaron a la búsqueda de agregar valor y prestaciones a la EDU-CIAA-NXP, utilizando sus capacidades ociosas para incrementar su utilidad como herramienta didáctica y posibilitar la práctica de computación gráfica.

El objetivo planteado se logró sumándole al kit un

TABLA I
BANCOS CONTIGUOS DE MEMORIA SRAM EN EL MICROCONTROLADOR LPC4337JBD144 DE NXP.

Tamaño	Bus	Dirección
32 KiB	LOCAL	10 00 00 00 h
40 KiB	LOCAL	10 08 00 00 h
64 KiB	AHB	20 00 00 00 h

adaptador de video por software -entre otras capacidades multimedia- mediante una expansión de bajo costo desarrollada por el proyecto RETRO-CIAA. Este artículo presenta la técnica en base a las decisiones de diseño del proyecto.

III. DISEÑO E IMPLEMENTACIÓN

A continuación se enumeran los elementos que componen esta implementación y los criterios de diseño utilizados.

III-A. Memoria de video

El *framebuffer* es un segmento contiguo en memoria que contiene los pixeles (mínima unidad de una imagen digital) que componen el cuadro de animación a visualizar [1]. Modificar la memoria del cuadro visible produce inestabilidad en la imagen en pantalla. Modificar sólo en los períodos inactivos de la señal de video reduce la capacidad de dibujo en un $95\,\%$. En consecuencia se decidió utilizar un esquema superador denominado *double-buffering* [1]: dos buffers iguales, uno visible y otro oculto en proceso de actualización. Los buffers intercambian roles toda vez que la señal de video termina de emitir un cuadro de animación.

Al ser una relación de compromiso, con esta técnica se gana en estabilidad y fluidez, pero se sacrifican otros aspectos como una mayor cantidad de memoria disponible para la aplicación o una mayor resolución en un solo *framebuffer* de mayor tamaño.

III-B. Resolución

La resolución de una imagen se define como su cantidad de pixeles en ancho y alto. En esta implementación cada uno de los dos *framebuffers* tienen una resolución de 256x144 pixeles. Esta resolución surge como compromiso entre la escasa SRAM en el microcontrolador LPC4337JBD144 de NXP y los bancos de memoria disponibles según se observa en la Tabla I.

Ademas, la resolución utilizada es coherente con la capacidad del microprocesador para manejar cierta cantidad de pixeles con soltura y es múltiplo de la resolución de la señal de video generada por el adaptador.

III-C. Formato de pixel

Un pixel contiene la información de color en un punto discreto de la imagen digital. El formato refiere a la cantidad total de bits por pixel y al reparto de esos bits entre los componentes rojo, verde y azul que describen color mediante la intensidad de primarios aditivos.

Para este proyecto se decidió utilizar un formato de pixel de $8\,\text{bit}$ repartidos en $3\,\text{bit}$ para rojo, $3\,\text{bit}$ para verde y $2\,\text{bit}$ para azul, lo que configura un formato RGB 3:3:2 tal como se observa en la Fig. 3. Con $8\,\text{bit}$ se obtiene una combinación total de $2^8 = 256$ colores disponibles.

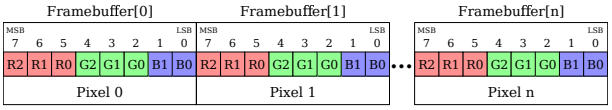

Fig. 3. Formato de pixel y pixeles en el *framebuffer*.

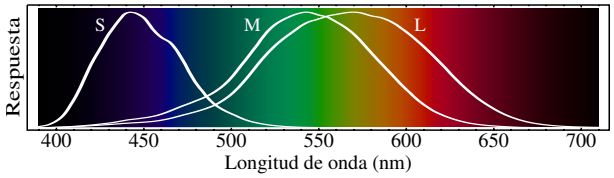

Fig. 4. Respuesta normalizada de los conos tipo S, M y L del ojo humano.

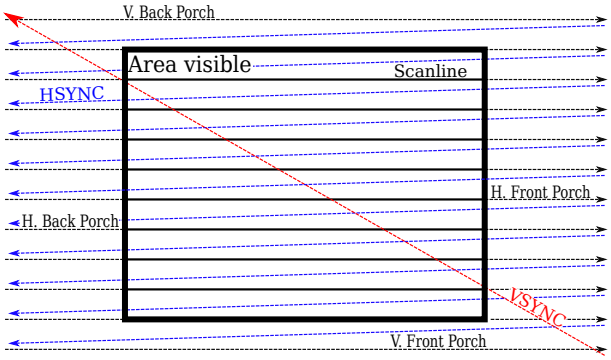

Fig. 5. Escaneo raster de una señal de video.

Se asignan $2\,\text{bit}$ al componente azul debido a que la sensibilidad del ojo humano (conos tipo S) es menor en ese rango de longitud de onda del espectro visible [7]. Esto puede apreciarse en la Fig. 4.

En este punto ya es posible calcular la cantidad de SRAM necesaria para implementar el adaptador de video por software (1) donde F_{res} es la resolución del *framebuffer*, F_n la cantidad de *framebuffers* y P_{bits} la cantidad de bits del formato de pixel.

$$\frac{F_{res} \times F_n \times P_{bits}}{8} = \frac{(256 \times 144) \times 2 \times 8}{8} = 72\,\text{KiB} \quad (1)$$

A continuación se revisan las señales de video necesarias y su interfaz eléctrica.

III-D. Raster scan

El proceso conocido como *raster scan* [1] define el temporizado de los elementos de una señal de video. La señal recorre la pantalla de izquierda a derecha -cada recorrido es una linea de pixeles o *scanline*- y de arriba hacia abajo -donde un barrido completo de todas las *scanlines* completa un cuadro de imagen-. Al final de cada linea se genera una demora que se denomina *horizontal blanking interval*. Del mismo modo, al terminar cada cuadro se genera una demora denominada *vertical blanking interval*. Los *blanking intervals* son áreas inactivas en donde no se dibujan pixeles.

El intervalo de *blanking* horizontal se divide en tres partes: *front porch*, pulso de sincronismo horizontal (HSYNC) y *back porch*. El intervalo de *blanking* vertical se divide en *front porch*, pulso de sincronismo vertical (VSYNC) y *back porch*. En la Fig. 5 se observa esta dinámica.

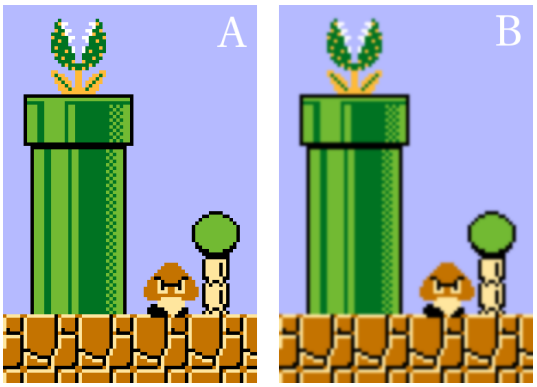

Fig. 6. Simulación del efecto de escalado en televisores LCD. (A) Fragmento de 74x110 de una señal de baja resolución en un monitor CRT. (B) Mismo contenido en una pantalla HDTV. Captura parcial de pantalla del emblemático videojuego *Super Mario Bros.* (Nintendo, 1985) para consola Nintendo Famicom.

III-E. Calidad de imagen

Durante varias décadas los tubos de rayos catódicos o CRT fueron utilizados para visualizar contenido en baja resolución, pero actualmente esa tecnología es obsoleta y fue completamente reemplazada por pantallas LCD HDTV.

La actualización de la pantalla en televisores o monitores CRT es analógica por naturaleza y permite representar diversas resoluciones con buena fidelidad [8]. En contrapartida, los LCD poseen una resolución nativa o cantidad fija de pixeles tanto en horizontal como en vertical y deben aplicar un algoritmo de escalado digital si la señal no coincide con esta resolución, lo que produce distorsiones en la imagen original.

Como se observa en la Fig. 6, las imágenes en baja resolución generadas por computadora son particularmente propensas a visualizarse de forma suavizada o difuminada en un monitor LCD HDTV.

Por este motivo, a fin de obtener en pantalla una fiel representación de los pixeles en el *framebuffer* de baja resolución, es necesario generar una señal de video de alta definición escalando el contenido del *framebuffer* desde el mismo adaptador de video.

III-F. Señal de video

Se generó un modo de video compatible con el estándar de televisión de pantalla ancha y alta definición.

El modo elegido es 720p@60 Hz o un área activa de 1280x720 pixeles y frecuencia de actualización de pantalla de 60 Hz o 60 cuadros por segundo [9].

En la Fig. 7 se observa un diagrama completo de los elementos que componen la señal de video. La señal especifica una cantidad de pixeles mayor al modo de video elegido ya que se suman los intervalos de *blanking* al final de cada linea y de cada cuadro. Estos nuevos pixeles, inactivos son parte del temporizado y no se visualizan en pantalla.

Para generar el temporizado de la señal se utilizó el estándar VESA CVT [10]. Como resultado se obtuvo una frecuencia de sincronismo vertical o VSYNC de 59,86 Hz y una frecuencia de sincronismo horizontal o HSYNC de 44,77 kHz. Los pulsos de sincronismo son activo-bajo para HSYNC y activo-alto para VSYNC. En la Tabla II se lista el temporizado de cada elemento.

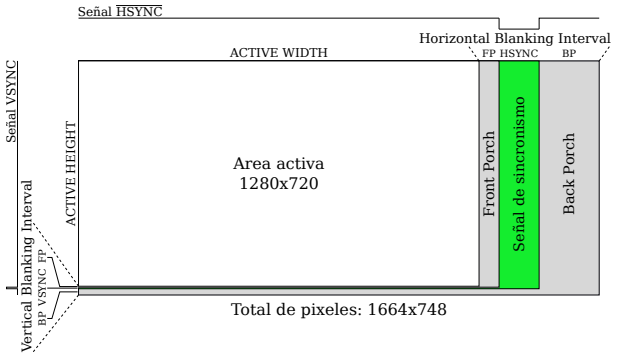

Fig. 7. Elementos de temporizado de la señal de video.

TABLA II
INTERVALOS DE TIEMPO DE LA SEÑAL DE VIDEO.

Parámetro	**Cantidad**	**Tiempo**
Horizontal Total (Line)	1664 pixeles	22,333 75 µs
Horizontal Active Pixels	1280 pixeles	17,179 80 µs
Horizontal Blanking Interval	384 pixeles	5,153 94 µs
Horizontal Front Porch	64 pixeles	0,858 99 µs
Horizontal Sync Width	128 pixeles	1,717 98 µs
Horizontal Back Porch	192 pixeles	2,576 97 µs
Vertical Total Lines	748 lineas	16,705 64 ms
Vertical Active Lines	720 lineas	16,080 30 ms
Vertical Blanking Interval	28 lineas	0,625 34 ms
Vertical Front Porch	3 lineas	0,067 00 ms
Vertical Sync Width	5 lineas	0,111 66 ms
Vertical Back Porch	20 lineas	0,446 67 ms

La cantidad total de pixeles de la señal de video es de 1664x748. Se calcula sumando los pixeles el área visible y los intervalos de *blanking* o pixeles no visibles.

El reloj de pixel o PCLK es la frecuencia a la cual el adaptador de video debe ser capaz de procesar pixeles. Se calcula teniendo en cuenta los visibles, los no visibles o areas de *blanking* y la frecuencia de actualización de la pantalla (2).

$$PCLK = (1664 \times 748) \times 59{,}86\,\text{Hz} = 74{,}50\,\text{MHz} \qquad (2)$$

En la ecuación 3 se calcula el periodo de un solo pixel o TPIXEL.

$$TPIXEL = \frac{1}{74{,}50\,\text{MHz}} = 13{,}421\,72\,\text{ns} \qquad (3)$$

De las interfaces de video más comunes como HDMI, DVI, DisplayPort o VGA se eligió esta ultima -incluso a pesar de su obsolescencia- por el bajo costo de implementación del hardware necesario para emitir señales compatibles.

La interfaz VGA es analógica. Define la intensidad individual de los componentes de color rojo, verde y azul en un rango de tensión de 0 a 0,7 Vpp a 75 Ω de impedancia. Las señales de sincronismo horizontal y vertical son TTL [8].

RETRO-CIAA sólo utiliza la interfaz eléctrica del estándar VGA y el modo de video de alta resolución no es parte de dicho estándar. Aun así, los monitores LCD, televisores HDTV o adaptadores VGA a HDMI no tienen inconveniente en admitir la señal generada.

La señal analógica de componentes de color se implementa mediante un DAC ad-hoc simple y económico utilizando

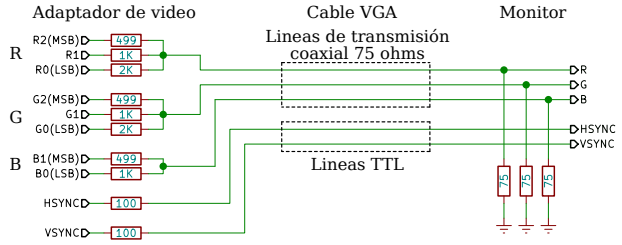

Fig. 8. Interfaz eléctrica de la señal de video.

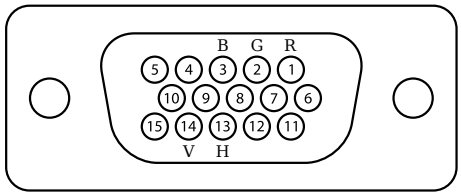

Fig. 9. Disposición de pines del conector estándar VGA.

8 salidas digitales y resistencias en paralelo, configurando un divisor de tensión en conjunto con la resistencia de terminación de 75 Ω del monitor. La posición de cada bit de más a menos significativo tiene un peso o aporte relativo equivalente en tensión. En la Fig. 8 se observa un esquema de la interfaz eléctrica.

Las señales digitales TTL se generan con 3,3 V para simplificar el diseño y ahorrar en componentes. Esta solución es similar a la utilizada en los kits de desarrollo FPGA Nexys con puerto VGA [11].

III-G. Puerto de video

El puerto expone las señales de video al exterior. En este caso se utiliza el puerto estándar VGA con conector DB-15HD hembra. En la Fig. 9 se observa la disposición de pines, donde (B, G, R) son los componentes de color analógicos azul, verde, rojo y (V, H) las señales de sincronismo vertical y horizontal TTL [8].

III-H. Escalado

En relación a la señal de video, el *framebuffer* contiene cinco veces menos pixeles tanto en horizontal como en vertical. Por este motivo, el adaptador de video implementa un método de escalado.

Cada pixel emitido en la señal de video se genera repitiendo cinco veces el valor de cada pixel en el framebuffer. A su vez, la lectura de cada linea de pixeles en el *framebuffer* se repite cinco veces para generar cinco lineas sucesivas e iguales en la señal de video. El resultado de esta operación es un escalado «al vuelo» del *framebuffer* con un factor de cinco en ambas dimensiones y una señal de video de alta resolución cuya imagen resultante es fiel al *framebuffer* original.

IV. Calculo de factibilidad

La elección de los parámetros de diseño del adaptador de video -en particular el modo de video y la señal a generar- esta restringida por la velocidad de reloj y las características de la arquitectura en donde se implementará el sistema.

El núcleo Cortex-M0 implementa la arquitectura ARMv6-M. La mayoría de las instrucciones se ejecutan de uno, dos o tres ciclos de reloj [12].

La cantidad de ciclos de reloj necesarios para generar una linea o *scanline* de la señal de video depende de la habilidad del programador para optimizar el código. No obstante, se pueden identificar algunas tareas necesarias y la cantidad mínima de ciclos, siendo doce el total resultante:

1. La lectura del *framebuffer* en memoria SRAM requiere de dos ciclos de reloj y se realiza mediante la instrucción de assembler LDR.
2. El reordenamiento de los bits a los puertos de E/S según restricciones del proyecto RETRO-CIAA [13] requiere de tres ciclos de reloj: dos ciclos para dos operaciones de corrimiento de bits y un ciclo para un OR binario (instrucciones LSRS, LSLS y ORRS).
3. La escritura de pines de E/S requiere dos ciclos si los pines se encuentran en el mismo puerto.
4. El incremento del puntero al *framebuffer* demanda un ciclo (instrucción ADDS).
5. El decremento de la cantidad de pixeles pendientes en la línea del *framebuffer* insume un ciclo (instrucción SUBS).
6. Si la cantidad de pixeles es mayor a 0, el inicio de una nueva iteración requiere tres ciclos (instrucción BNE).

En las ecuaciones (4) y (5) se calcula el periodo de cada ciclo de reloj que permitirá evaluar si el modo de video elegido es factible de implementar.

$$MCU_{clock} = 204\,\text{MHz} \tag{4}$$

$$MCU_{cycle} = \frac{1}{MCU_{clock}} = \frac{1}{204\,\text{MHz}} = 4{,}901\,96\,\text{ns} \tag{5}$$

En el apartado sobre la señal de video se calculó la constante TPIXEL. En la ecuación (6) se calcula la cantidad de ciclos del microcontrolador por cada pixel de la señal de video o CPIXEL.

$$CPIXEL = \frac{TPIXEL}{MCU_{\text{cycle}}} = \frac{13{,}421\,72\,\text{ns}}{4{,}901\,96\,\text{ns}} = 2{,}73803 \tag{6}$$

La señal de video contiene 1280x720 pixeles visibles. En el temporizado calculado se disponen de 2 ciclos de reloj por pixel en esa resolución, por lo tanto su implementación no es viable.

Sin embargo, según las decisiones de diseño tomadas teniendo en cuenta las restricciones de memoria y procesamiento, cada linea del *framebuffer* de 256 pixeles se repite 5 veces para generar una linea de 1280 pixeles. En cada iteración, un pixel del *framebuffer* genera 5 pixeles de la señal de video. En consecuencia, se dispone de cinco veces el tiempo calculado para *TPIXEL*. La nueva cantidad de ciclos se calcula en la ecuación (7).

$$CPIXEL5 = \frac{TPIXEL \times 5}{MCU_{\text{cycle}}} = \frac{67{,}1086\,\text{ns}}{4{,}901\,96\,\text{ns}} = 13{,}69015 \tag{7}$$

De esta forma se disponen de 13 o 14 ciclos, volviendo viable su implementación.

En la memoria del proyecto RETRO-CIAA [13] se revisa en detalle las implicancias de un número no entero de ciclos en la generación de la señal de video.

Fig. 10. Video por *streaming* desde tarjeta microSD y *transcoding* de 24 a 60 cuadros por segundo en el núcleo Cortex-M4F mientras el Cortex-M0 genera la señal de video.

Fig. 12. Efectos aplicados «al vuelo» por el núcleo Cortex-M0 mientras genera la señal de video: *scanlines* de monitor CRT y filtros de color a pantalla completa.

Fig. 11. Videojuego de ejemplo. La funciones de dibujo de lineas e imágenes corren sobre el núcleo Cortex-M4F. El desplazamiento sobre el mapa es suave y fluido [14].

Fig. 13. Se grafican variables de un acelerómetro en tiempo real. La resolución del *framebuffer* es adecuada para trabajos educativos.

V. Resultados obtenidos

La señal estándar de alta definición es compatible con televisores LCD HDTV y adaptadores VGA a HDMI. La imagen resultante es similar a la producida por las consolas de videojuegos de los años '80s. Ponderando la calidad de imagen resultante en un televisor LCD HDTV [14] contra proyectos similares [13] puede afirmarse que la fluidez, nitidez y estabilidad obtenida es sobresaliente para un sistema de este tipo. El núcleo Cortex-M4F queda totalmente disponible para ejecutar la aplicación y la memoria SRAM compartida permite que este núcleo dibuje sobre el área del *framebuffer* no visible utilizando instrucciones DSP y SIMD.

Estos resultados pueden apreciarse en las Figuras 10, 11, 12 y 13 que consisten en fotografiás de imágenes generadas por el proyecto RETRO-CIAA tomadas directamente de una pantalla LCD.

VI. Conclusiones

La técnica revisada posibilita agregar valor a un proyecto existente implementando una salida de video por software y sumando unas pocas resistencias y un conector para la señal. También permite bajar costos utilizando un microcontrolador sin soporte de video ó cubrir una necesidad especifica como puede ser generar una señal de alta definición desde un *framebuffer* de baja resolución, utilizar un formato de pixel especifico, generar una señal de video no estándar, etc.

Si bien esta técnica no es nueva, su uso junto a un *framebuffer* doble es poco común. Y generar una señal de alta definición realizando un escalado «al vuelo» de un *framebuffer* de baja resolución, es un enfoque novedoso.

El proyecto RETRO-CIAA agrega valor a la EDU-CIAA-NXP mediante la implementación de un adaptador de video por software, sin impacto para el núcleo utilizado comúnmente por las aplicaciones y con un costo de memoria SRAM significativo, pero aun así justificado en la posibilidad de contar con video de alta prestación.

Las decisiones de diseño fueron requerimientos del proyecto RETRO-CIAA. Otros proyectos pueden adaptar el uso de memoria, resolución, interfaz y modo de video según requerimientos particulares que modifiquen el balance de los recursos disponibles.

VII. Trabajo futuro

Como trabajo futuro se plantea promover este proyecto en la comunidad de usuarios del kit EDU-CIAA-NXP y la creación de ejemplos y material didáctico.

VIII. Agradecimientos

El autor agradece a Ariel Lutenberg y a Pablo Gomez por sus valiosos aportes, consejos y devoluciones, a Eric Pernia por su buena predisposición e ideas para dar a conocer el

proyecto RETRO-CIAA y al Simposio y Congreso Argentino de Sistemas Embebidos (SASE/CASE) por brindar la posibilidad de difundir este trabajo.

REFERENCIAS

[1] Foley, van Dam, Feiner, Hughes, «Computer Graphics, Principles and practice - Second edition in C», Addison Wesley, 1997.
[2] Proyecto CIAA, «Computadora Industria Abierta Argentina - EDU-CIAA-NXP», http://www.proyecto-ciaa.com.ar/devwiki/doku.php?id=desarrollo:edu-ciaa:edu-ciaa-nxp.
[3] NXP Semiconductors, «LPC43xx/LPC43Sxx ARM Cortex-M4/M0 multi-core microcontroller», version 2.3, July 2017.
[4] Santiago Germino, «Página principal del proyecto RETRO-CIAA», http://www.retro-ciaa.com.
[5] Eric Pernia, Martin Ribelotta y contribuidores, «Repositorio GitHub de la futura version 3 del firmware del proyecto CIAA», 3/12/2018, https://github.com/epernia/cese-edu-ciaa-template.
[6] Proyecto CIAA, «Ejemplos de uso de la CIAA», http://www.proyecto-ciaa.com.ar/index_quees_ejemplosuso.html.
[7] Stockman, MacLeod y Johnson, «2-deg cone fundamentals (based on the Stiles and Burch 2-deg CMFs», 1993, http://www.cvrl.org/database/text/cones/smj2.htm.
[8] Robert L. Myers, «Display Interfaces - Fundamentals and standards», Wiley-SID, 2002.
[9] The Advanced Television Systems Committee, Inc., «ATSC Digital Television Standard», A/53 Parts 1 - 6, 2007.
[10] Video Electronics Standards Association, «VESA Coordinated Video Timings (CVT) Standard», version 1.2, February 2013.
[11] Digilent Inc., «Nexys A7: FPGA Trainer Board Recommended for ECE Curriculum» https://store.digilentinc.com/nexys-a7-fpga-trainer-board-recommended-for-ece-curriculum.
[12] Joseph Yiu, «The Definitive Guide to ARM Cortex-M0 and Cortex-M0+ Processors - 2nd Ed.», Elsevier Inc, 2015.
[13] Santiago Germino, «RETRO-CIAA: Consola de videojuegos basada en EDU-CIAA-NXP», http://laboratorios.fi.uba.ar/lse/tesis/LSE-FIUBA-Trabajo-Final-CESE-Santiago-Germino-2018.pdf
[14] Santiago Germino, «RETRO-CIAA: Consola de videojuegos basada en EDU-CIAA-NXP - Demostración Defensa TF CESE», https://www.youtube.com/watch?v=tJM6_TdOuKQ.