Implementación mediante FPGA de un sistema SVM de verificación de locutor

Rafael Ramos-Lara[1], Mariano López-García[1], Enrique Cantó-Navarro[2], Luis Puente-Rodríguez[3]

[1] Universidad Politécnica de Catalunya, Victor Balaguer s/n, 08800 Vilanova i Geltrú
[2] Universidad Rovira i Virgili, Avda. Paisos Catalans, 26, 43007, Tarragona
[3] Universidad Carlos III de Madrid, C/ Butarque, 15, 28911, Leganes
{lara, lopezg}@eel.upc.edu, ecanto@etse.urv.es, lpuente@it.uc3m.es

Resumen. Los sistemas biométricos caracterizados por su alto nivel de seguridad se implementan habitualmente con sistemas procesadores de altas prestaciones como los ordenadores personales. Estos procesadores trabajan en un rango de frecuencias de GHz que les permiten realizar millones de operaciones por segundo, de forma que pueden ejecutar en tiempo real complejos algoritmos de verificación. Sin embargo, esta solución de implementación tiene el inconveniente del elevado coste. La utilización de dispositivos programables del tipo FPGA (Field Programmable Gate Array) permite obtener a bajo coste soluciones a medida con las que se consiguen elevadas velocidades de proceso similares a los sistemas µP de altas prestaciones. En este artículo se presenta el diseño e implementación sobre una FPGA de un sistema de verificación de locutor basado en los coeficientes Mel-Cepstrum y en un algoritmo de clasificación SVM (Support Vector Machines). Los resultados experimentales obtenidos con el diseño propuesto muestran una velocidad de proceso equiparable a la conseguida con un ordenador personal basado en el µP Pentium IV.

1 Introducción

El desarrollo de los algoritmos biométricos está principalmente enfocado en mejorar los índices de bondad de los sistemas de identificación y/o verificación, como la tasa de falso rechazo (TFR) o la tasa de falsa aceptación (TFA), dando por sentado que el sistema que ejecutará el algoritmo tiene suficiente capacidad computacional como para procesar toda la información en tiempo real. Esto ha dado lugar a algoritmos biométricos de alto nivel de eficiencia pero con una elevada carga computacional [1][2]. Unos de estos algoritmos biométricos es el de verificación de locutor que utiliza técnicas de clasificación basadas en SVM (Support Vector Machine) [3] combinadas con la utilización de coeficientes Mel-Cepstrum [4-6] que proporcionan buenos resultados con aceptables tasas de reconocimiento de usuario.

En la literatura existen algunas publicaciones que realizan implementaciones con FPGA de algoritmos de clasificación SVM, pero sólo un reducido grupo de ellas está dedicado a la verificación de locutores. En [7], los autores muestran las principales características de un sistema SVM de verificación de locutor implementado mediante una tarjeta inteligente

(Match-on-Card). Debido a las limitaciones de memoria de la tarjeta, los autores usan una media temporal de todas las tramas de voz como vector de parámetros, representando cada locución por un único vector de 24 componentes. El artículo también muestra la implementación en una FPGA de la etapa de clasificación SVM basada en una función exponencial. El diseño propuesto es capaz de realizar la clasificación de un vector 50 veces más rápido que el algoritmo de clasificación ejecutado en un PC Pentium IV trabajando con una frecuencia de reloj de 1.3GHz. En otras publicaciones se muestra sólo la implementación hardware de alguna parte específica del algoritmo de verificación o identificación de locutor como medida para disminuir el tiempo total de proceso [8][9].

Una posible solución para reducir el tiempo de proceso y al mismo tiempo los recursos necesarios para la implementación hardware de algoritmos de reconocimiento de voz consiste en utilizar un formato de representación numérica de coma fija en lugar del formato de coma flotante. En [10] los autores muestran una implementación ASIC de un sistema de reconocimiento de voz basado en los coeficientes Cesptrum donde se utiliza el formato de coma fija lo que permite reducir el área del diseño.

En este artículo se presenta la implementación hardware en coma fija sobre un dispositivo FPGA de un algoritmo de verificación de locutor basado en los coeficientes Mel-Cepstrum y en el algoritmo de clasificación SVM. Este artículo se organiza de la siguiente manera. En la sección 2 se describe la teoría del sistema de verificación de locutor junto con las distintas operaciones involucradas en el cálculo de los coeficientes Mel-Cepstrum, así como en el sistema de clasificación SVM. En la sección 3 se describen las principales características del diseño FPGA en coma fija del sistema de verificación de usuario. En la sección 4 se presentan los resultados experimentales obtenidos con el diseño FPGA en términos de tiempo de ejecución y error de clasificación comparándose dichos resultados con los obtenidos con la ejecución del algoritmo de verificación en código fuente C en un ordenador PC Pentium IV y en un microprocesador de gama media. Finalmente en la sección 5 se indican las conclusiones del presente trabajo.

2 Sistema de verificación de locutor

En la figura 1 se puede ver el diagrama de bloques del sistema de verificación de locutor en el cual se pueden diferenciar dos fases o procesos: la fase de entrenamiento donde se obtiene el modelo que se utiliza durante el proceso de verificación del usuario y la fase de funcionamiento donde se lleva a cabo la verificación propiamente dicha de la identidad del usuario. En los siguientes subapartados se presentan los detalles de cada fase.

2.1 Fase de entrenamiento

Esta fase consiste en obtener un modelo, compuesto por un conjunto de vectores de parámetros, que contiene las características de la voz del usuario que se pretende verificar así como de personas ajenas al usuario y que servirá para proceder a su identificación en base a una locución suya posterior. La obtención del modelo se realiza antes de la puesta en marcha del sistema de verificación (off-line) utilizando para ello un algoritmo de entrenamiento que se ejecuta en un PC.

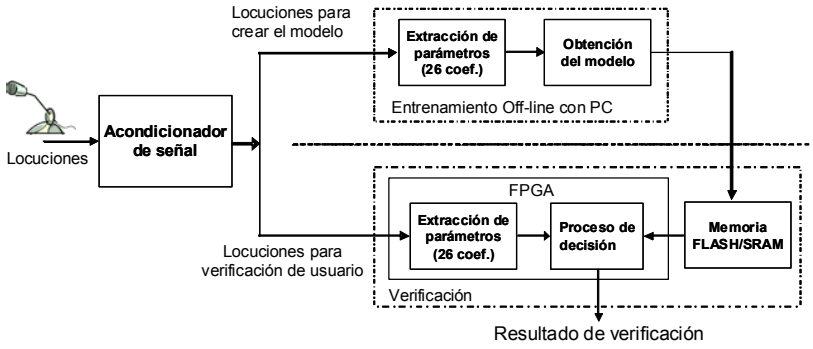

Fig. 1. Diagrama de bloques del sistema de verificación de voz

Para obtener el conjunto de vectores de parámetros que constituyen el modelo se recogen 4 locuciones de 14 segundos de duración por persona con una frecuencia de muestreo de 8kHz y 12 bits de resolución, provenientes de 52 locutores (26 hombres y 26 mujeres) entre los que se encuentra el propio usuario a identificar. A partir de las muestras recogidas se obtienen 4000 vectores del usuario a identificar y 4000 vectores correspondientes a personas ajenas al usuario. Los 8000 vectores se entregan al entrenador de SVM y como resultado se obtiene el modelo con los vectores soporte y los parámetros *gamma* y *rho* utilizados en el proceso de decisión.

2.2 Fase de funcionamiento

Esta fase se corresponde con el funcionamiento del sistema de verificación de voz y es la que se ha implementado mediante una FPGA. En esta fase se pueden distinguir dos sistemas: un sistema de extracción de parámetros que procesa la señal de voz del locutor para extraer un conjunto de vectores de parámetros y un sistema de decisión, basado en el algoritmo SVM, que compara los vectores de parámetros del locutor con el modelo de la persona que pretende ser. En los siguientes subapartados se describe con detalle cada uno de estos sistemas.

2.2.1 Sistema de extracción de parámetros

La extracción de parámetros tiene como objetivo obtener a partir de una locución, de la persona objeto de verificación, un conjunto de vectores de parámetros que modelan las características de su voz y que se deben confrontar con los vectores del modelo para verificar su identidad. Durante el proceso de extracción de parámetros la señal proveniente de la locución del usuario se enventana en tramos de 25ms (200 muestras), durante los cuales la señal se puede considerar seudo-estacionaria, con un solapamiento de 10ms (80 muestras). De cada trama se obtiene un vector que contiene 26 parámetros o coeficientes: el primero se corresponde con la energía localizada en la ventana temporal, los 12 siguientes se basan en los Mel-Frequency Cepstral Coefficients (MFCC) [4-6] que representan la envolvente espectral de la señal de voz y los treces últimos coeficientes son los parámetros diferencia-

les o de velocidad de los 13 primeros coeficientes que se obtienen procesando N vectores adyacentes.

En la figura 2 se muestra la secuencia de operaciones a implementar para calcular los coeficientes de los vectores. La primera operación consiste en un pre-procesado de las muestras de la trama que incluye una normalización de la señal obtenida restándole su valor medio:

$$\bar{s}(n) = s(n) - \bar{S} \qquad con \qquad \bar{S} = \frac{\sum_{n=0}^{N-1} s(n)}{N} \qquad para \qquad 0 \le n \le N-1 \tag{1}$$

donde N es el número total de muestras de la trama. A continuación se somete la señal normalizada a un filtro pre-énfasis para compensar la menor amplitud de las componentes de alta frecuencia. La expresión del filtro pre-enfasis viene dada por:

$$y(n) = \bar{s}(n) - k \cdot \bar{s}(n-1) \qquad con \qquad 0 \le n \le N-1 \qquad y \qquad k = 0.97 \tag{2}$$

A partir de la trama pre-procesada *y(n)* se obtiene el primer coeficiente del vector que se corresponde con el logaritmo de la energía localizada de la trama y que viene dado por la expresión:

$$E = \ln\left(\sum_{n=0}^{N-1} y^2(n)\right) \qquad con \qquad 0 \le n \le N-1 \tag{3}$$

Después del pre-procesado, la trama se somete a una ventana de tipo Hamming que suaviza los efectos del enventanado rectangular. Al resultado obtenido se añaden ceros hasta conseguir una trama de 256 muestras de la que se calcula la transformada rápida de Fourier. Sobre la amplitud de la FFT se aplica un banco de filtros Mel [6] que simulan la respuesta de la membrana basilar del oído humano. Los 12 coeficientes Mel-Cepstrum se obtienen calculando la transformada inversa de Fourier del logaritmo de la respuesta del filtro Mel. Teniendo en cuenta que el logaritmo de la respuesta del filtro Mel es real y simétrica, la transformada inversa de Fourier se reduce a la transformada discreta del coseno.

Finalmente, los 13 últimos coeficientes del vector se corresponden con los parámetros diferenciales o de velocidad que proporcionan información sobre el comportamiento dinámico de la señal de voz y que vienen dados por:

$$\Delta C_i(n) = \frac{\sum_{k=-N}^{k=N} k \cdot C_i(n+k)}{\sum_{k=-N}^{k=N} k^2} \qquad con \qquad 0 \le i \le 12 \tag{4}$$

donde C_0 es el coeficiente de la energía localizada, $C_1,\ldots,C_{12}$ son los MFCC (Mel-Frequency Cepstral Coefficients) de la señal de voz y N =2 es el número de vectores adyacentes utilizados para calcular el promedio.

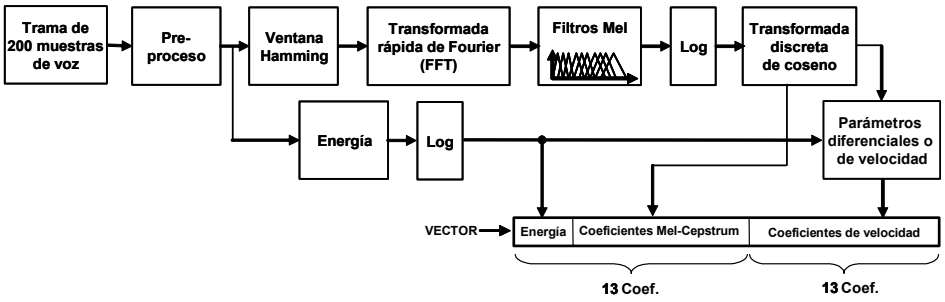

Fig. 2. Diagrama de bloques usado para obtener el vector de coeficientes.

2.2.2 Sistema de decisión basado en SVM

En la figura 3 se muestra la secuencia de operaciones involucradas en el proceso de decisión. En este proceso se realiza un análisis de los vectores, obtenidos a partir de locución de la persona cuya identidad se debe verificar, comparando cada vector con los vectores soporte del modelo y determinando con ello si pertenece o no al usuario que pretende ser. La comparación de un vector con el modelo se traduce en el cálculo de la siguiente expresión:

$$G = \sum_{j=1}^{K} P_j \cdot e^{-\gamma \sum_{i=1}^{26} (x_i - z_{ji})^2} \qquad (5)$$

donde K es el número de vectores soporte del modelo (en nuestro caso 3634), z_{ji} son los coeficientes de los vectores del modelo, x_i son los coeficientes del vector de la persona que se pretende verificar, P_j es el lagrangiano del vector soporte j, y γ es una constante que se ajusta en el proceso de entrenamiento en base a las características de los datos utilizados para construir el modelo.

Se considera que un vector del locutor pertenece al usuario que pretende ser si cumple que el resultado de evaluar la expresión (5) alcanza el umbral *rho* de decisión obtenido en la fase de entrenamiento, esto es:

$$\begin{aligned} G - rho \geq 0 \quad &\Rightarrow \quad pertenece\ al\ usuario \\ G - rho < 0 \quad &\Rightarrow \quad no\ pertenece\ al\ usuario \end{aligned} \qquad (6)$$

Si la proporción de vectores pertenecientes al usuario supera el nivel fijado en la fase de entrenamiento se considera probada la identidad del locutor.

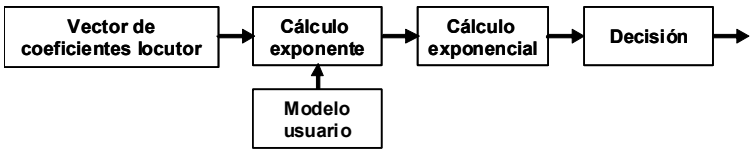

Fig. 3. Diagrama de bloques del proceso de decisión.

3 Diseño del sistema de verificación

En la figura 4 se muestra la arquitectura del sistema de verificación de voz basado en una FPGA. En el diseño FPGA realizado se pueden distinguir tres bloques principales: un bloque de extracción de parámetros que contiene todos los elementos que permiten realizar las operaciones que se muestran en la figura 2, un bloque de decisión que calcula las expresiones (5) y (6) y finalmente un microprocesador PicoBlaze de 8 bits que controla y gestiona el funcionamiento del sistema de verificación. Entre otras funciones el microprocesador PicoBlaze gestiona un sistema de comunicación basado en el puerto RS232 que junto con un menú de opciones permite la descarga desde un PC del modelo de vectores soporte utilizado en la fase de decisión.

Otros elementos importantes a destacar son un bloque de 16Mbytes de memoria FLASH que se encarga de almacenar el modelo de vectores soporte y un bloque de memoria SRAM de 2Mbytes de capacidad y 12ns de tiempo de acceso cuya función es almacenar temporalmente el modelo, de forma que el bloque de decisión puede acceder más rápidamente a los vectores del modelo reduciendo considerablemente, de esta forma, el tiempo de ejecución del algoritmo.

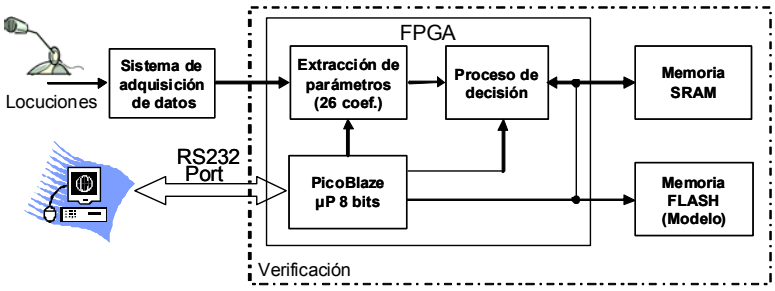

Fig. 4. Arquitectura del sistema de verificación basado en FPGA.

Aunque el algoritmo original de verificación en código C trabaja con formato de coma flotante de doble precisión, en esta implementación con FPGA se han realizado todas las operaciones en formato de coma fija, donde cada variable real consiste en una parte entera, de M bits de precisión, y una parte fraccional, de N bits de precisión. Utilizar el formato de coma fija ha permitido reducir sensiblemente el tiempo de proceso y los recursos necesarios con respecto a realizar un diseño hardware en coma flotante. Por ejemplo, el área nece-

saria para implementar un simple sumador en coma flotante en una FPGA del modelo Spartan 3 es aproximadamente de 240 CLB slices con una latencia de 13 ciclos de reloj. Sin embargo, un sumador de dos números enteros de 32 bits ocupa tan solo 16 CLB slices y tarda un ciclo de reloj en realizar la suma.

La utilización del formato en coma fija genera un error con respecto a la utilización del formato en coma flotante, este error depende de los valores de M y N para cada una de las operaciones involucradas en el proceso de verificación. Un valor elevado de M y N permite disminuir el error de cálculo pero incrementa sustancialmente el área y los recursos necesarios para implementar el diseño sobre una FPGA y en general aumenta el tiempo de proceso. Se debe, por tanto, llegar a una solución de compromiso que combine un error bajo, con un área reducida y una elevada velocidad de proceso. La elección de los valores adecuados de los parámetros M y N se ha realizado con ayuda de un procedimiento desarrollado en el entorno MatLab que calcula el error entre el resultado obtenido con el código fuente en coma flotante y el resultado obtenido con un modelo que simula el diseño FPGA en coma fija. En las tablas 1 y 2 se muestran los valores de M y N escogidos para cada operación del sistema de extracción de parámetros y el sistema de decisión respectivamente.

Función		M (bits parte entera)	N (bits parte fraccional)
Pre-procesado		15	8
Ventana Hamming		15	9
FFT	DFT. Coef.	23	9
	(Re^2+Im^2)	42	8
	Modulo DFT	21	10
Filtro Mel		21	2
Logaritmo		6	14
Transformada discreta del coseno		6	18
Coeficientes de velocidad		6	14

Tabla 1. Valores M y N seleccionados para realizar las operaciones asociadas al sistema de extracción de parámetros.

Función		M (bits parte entera)	N (bits parte fraccional)
Exponente	Resta de coeficientes	7	22
	Cuadrado de la resta	13	14
	Acumulación	18	14
Exponencial		0	18
Decisión (G-rho)	Multiplicación por el lagrangiano	6	31
	Sumatorio de exponenciales	18	31

Tabla 2. Valores M y N seleccionados para realizar las operaciones asociadas al sistema de decisión.

Para disminuir el tiempo de proceso se ha tabulado la función trigonométrica que interviene en el cálculo de la transformada inversa del coseno en una look-up table de 312 elementos codificados con 16 bits de precisión. Por otra parte, las funciones raíz cuadrada, logaritmo y exponencial se han implementado siguiendo los algoritmos recursivos descritos

en [11] que combinan una elevada velocidad de proceso con una reducida área de implementación.

Para optimizar aún más el tiempo de ejecución del sistema de decisión se ha parelizado el cálculo del exponente y de la exponencial de la expresión (5) (ver figura 5). El cálculo de un exponente tiene una duración de 60 periodos de reloj, mientras que el cálculo de la exponencial, la multiplicación por su coeficiente P_j y la acumulación del resultado dura 53 periodos de reloj. Por tanto, el tiempo total de ejecución del proceso de decisión viene dado por:

$$\text{Tiempo de ejecución} = K \cdot 60 T_{CLK} + 53 \cdot T_{CLK} \qquad (7)$$

donde K es el número de vectores soporte del modelo.

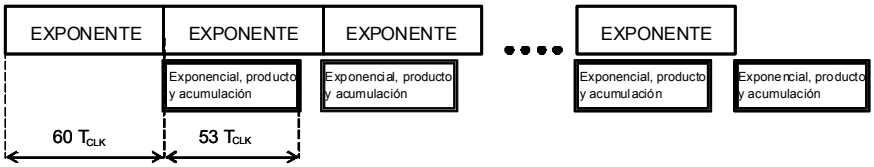

Fig. 5. Secuencia de cálculo de la expresión (5).

4 Resultados experimentales

El sistema descrito se ha implementado sobre una FPGA modelo Spartan 3 XC3S2000 de Xilinx que dispone de 20480 Slices distribuidos entre un total de 5120 Bloques Lógicos Configurables (CLB), 40 multiplicadores dedicados de 18x18 bits, 40 bloques de memoria BRAM de 18kbits cada uno, y 4 DCM (Digital Clock Manager) entre otras prestaciones. De estos recursos se han utilizado en la implementación del diseño FPGA un total de 4381 Slices, 22 multiplicadores dedicados, 15 bloques de memoria BRAM y un DCM, con una frecuencia de funcionamiento de 50MHz.

Para probar las prestaciones del diseño se ha utilizado un fichero de datos de voz que contiene el equivalente a 1394 tramas de voz, y un modelo de usuario compuesto de 3634 vectores soporte. En primer lugar se han comparado los resultados obtenidos con la FPGA en el cálculo de la exponencial (5) con los obtenidos mediante el algoritmo en coma flotante diseñado en código C. A partir de estos resultados se ha calculado el error relativo medio definido como:

$$100 \cdot \sum_{j=1}^{1394} \left| \frac{\left(G_{float} - rho\right) - \left(G_{fija} - rho\right)}{G_{float} - rho} \right| \qquad (8)$$

donde G_{float} es el resultado de evaluar la expresión (5) en coma flotante con doble precisión y G_{fija} es el resultado obtenido utilizando coma fija.

El error relativo medio obtenido en la prueba realizada es del 1.446%, mientras que el número de errores en la determinación del signo de clasificación es de cero. Es importante remarcar que el resultado de la clasificación consiste únicamente en el signo de la expresión (6) no en su magnitud, por tanto, en relación a esta prueba el diseño FPGA no ha cometido ningún error de clasificación.

También se han comparado los resultados obtenidos por el diseño FPGA propuesto en cuanto a tiempo de ejecución con los obtenidos por dos plataformas diferentes que se han tomado como ejemplo representativo de microprocesadores de alta y media gama: un Intel Pentium IV a 1.5GHz y un µP MicoBlaze con unidad de coma flotante de simple precisión a 40MHz. Ambos microprocesadores ejecutan el algoritmo original en código C con formato en coma flotante de doble precisión. En la tabla 3 se muestra los tiempos de ejecución en fase de extracción de parámetros y en la fase de decisión para una sola trama de voz. Los resultados del MicroBlaze han sido obtenidos implementando este microprocesador sobre una FPGA Spartan 3. Como se puede observar en dicha tabla el MicroBlaze tarda un total de 2557.1ms en realizar el proceso de verificación de una sola trama de usuario, mientras que el diseño FPGA propuesto de coma fija tarda 4,65ms, un tiempo similar al obtenido con el microprocesador Pentium IV (4.61ms). El sistema de verificación mostrado en la figura 1 genera una nueva trama de voz cada 10ms, por tanto si se desea trabajar en tiempo real se requiere un sistema de procesado que sea capaz de ejecutar las fases de extracción de parámetros y decisión en un tiempo igual o menor a 10ms. Esto es posible utilizando un microprocesador de altas prestaciones o bien el diseño hardware dedicado en coma fija. La utilización de un microprocesador de gama media-baja imposibilitaría la ejecución en tiempo real.

Función	Pentium IV a 1.5 GHz	Microblaze a 40 MHz	Diseño FPGA 50 MHz
Pre-procesado	14.12 µs	3.13 ms	31.96 µs
Ventana Hamming	3.13 µs	151 µs	24 µs
FFT	63.36 µs	8.83 ms	30.22µs
Filtro Mel	45.45 µs	6.75 ms	116.48µs
Logaritmo	8.41 µs	17.30 ms	53.78 µs
Transformada inversa de coseno	102.57 µs	216.32 ms	26.46 µs
Coeficientes de velocidad	1.73 µs	620 µs	2.54 µs
Tiempo de ejecución de la extracción de parámetros para una trama	*238.77 µs*	*253.1 ms*	*285.44 µs*
Tiempo de ejecución de la fase de decisión	*4370.15 µs*	*2304 ms*	*4362 µs*
Tiempo total de ejecución de una trama	**4608.92 µs**	**2557.1 ms**	**4647.44µs**

Tabla 3. Tiempo de ejecución de los procesos de extracción de parámetros y decisión para un µP Pentium IV, para un µP Microblaze y para el diseño FPGA propuesto.

5 Conclusiones

La implementación de sistemas de verificación y/o identificación biométrica se realiza generalmente mediante sistemas microprocesadores de altas prestaciones y elevado coste que permiten ejecutar en tiempo real los complejos algoritmos relacionados con este tipo de aplicaciones. En este artículo se ha presentado la implementación mediante una FPGA de bajo coste de un sistema de verificación de locutor basado en los coeficientes Mel-Cepstrum y en el algoritmo de clasificación SVM. Combinando un diseño hardware dedicado con un procesado de datos en coma fija de precisión variable se consigue, con una señal de reloj de tan sólo 50MHz, unas prestaciones similares en cuanto a tiempo de proceso a las obtenidas con un PC Pentium IV a 1.5GHz, así como un error relativo reducido. El sistema propuesto es capaz de realizar el proceso de extracción de vectores y el proceso de clasificación con un modelo compuesto de 3634 vectores en 285.44μs y 4362μs respectivamente. El error relativo obtenido en las pruebas realizadas es de tan sólo un 1.446 % en cuanto al valor de *G-rho*, siendo cero el número de vectores clasificados de forma errónea. Estos resultados permiten validar el diseño realizado y confirman a los dispositivos FPGA como componentes idóneos para diseñar con un coste razonable arquitecturas hardware específicas que resuelvan aplicaciones que requieran elevada velocidad de proceso que difícilmente serian realizables con otro tipo de dispositivos digitales de bajo coste.

Referencias

1. Lopez, M., Cantó, E.,: FPGA implementation of a Minutiae Extraction Fingerprint Algorithm: IEEE International Symposium on Industrial Electronics, Cambridge, U.K. (2008)
2. Cantó, E., Canyellas, N., Fons, M., Fons, F., López, M.,: FPGA Implementation of the Ridge Line Following Fingerprint Algorithm: 14th International Conference on Field-Programmable Logic and Applications, Springer-Verlag LNCS 3203, pp. 1087-1089, Antwerp, Belgium (2004)
3. Burges, C.J.C.,: A Tutorial on Support Vector Machines for Pattern Recognition. 1998 Kluwer Academic Publishers, Data Mining and Knowledge Discovery, vol. 2, pp. 121-167
4. Noll, A.M.: Cesptrum Pitch Determination. The Journal of the Acoustical Society of America, Volume 41, Number 2, pp. 293-309, 1967.
5. Childers, D. G., Skinner, D. P.: The Cepstrum: A Guide to Processing. Proceedings of the IEEE, Volume 65, Number 10, pp. 1428-1443, October 1977.
6. Rabiner, L., Juang Biing-Hwang: Fundamentals of Speech Recognition. Prentice-Hall, Englewood Cliffs, N.J., 1993.
7. Choi, W-Y., Ahn, D., Burn Pan, S., Chung, K., Chung, Y., Chung, S-H.: SVM-Based Speaker Verification System for Match-on-Card and its Hardware Implementation. ETRI Journal, Volume 28, Number 3, pp. 320-328, June 2006
8. Nedevschi, S., Patra, R., Brewer, E.,: Hardware Speech Recognition for User Interfaces in Los cost, Low Power Devices. 43rd Design Automation Conference, IEEE Press, California, USA, pp. 684-689, June 2005.
9. Melnikoff, S., Quigley, S.F., Rusell, M. J.,: Implementing a Simple Continuous Speech Recogniton System on an FPGA. Proceedings of the 10th Annual IEEE Symposium on Field-Programmable Custom Computing Machines, Napa, California, USA (2002)
10. Gin-Der Wu, Zhen-Wei Zhu :Chip Design of LPC-cepstrum for Speech Recognition. ACIS International Conference on Computer and Information Science (ICIS 2007)
11. Ercegovac, M.D.: Digital Aritmetic. Ed. Morgan Kaufmann, pp. 584-593.