Anexo A

f01 – Sphere Model

$$f_1(x) = \sum_{i=1}^{30} (x_i)^2$$

-100 ≤ x_i ≤ 100

min $(f_1) = f_1(0, \ldots, 0) = 0$

f02 – Schwefel's Problem

$$f_2(x) = \sum_{i=1}^{30} |x_i| + \prod_{i=1}^{30} |x_i|$$

-10 ≤ x_i ≤ 10

min $(f_2) = f_2(0, \ldots, 0) = 0$

f03 – Schwefel's Problem

$$f_3(x) = \sum_{i=1}^{30} \left(\sum_{j=1}^{i} x_j \right)^2$$

-100 ≤ x_i ≤ 100

min $(f_3) = f_3(0, \ldots, 0) = 0$

f04 – Schwefel's Problem

$$f_4(x) = max_i \{[x_i], 1 \le i \le 30\}$$

-100 ≤ x_i ≤ 100

min $(f_4) = f_4(0, \ldots, 0) = 0$

f05 – Generalized Rosenbrock's Function

$$f_5(x) = \sum_{i=1}^{29} \left| 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right|$$

-30 ≤ x_i ≤ 30

min $(f_5) = f_5(1, \ldots, 1) = 0$

f06 – Step Function

$$f_6(x) = \sum_{i=1}^{30} (\lfloor x_i + 0.5 \rfloor)^2$$

-100 ≤ x_i ≤ 100

min $(f_6) = f_6(0, \ldots, 0) = 0$

f07 – Quartic Function with Noise

$$f_7(x) = \sum_{i=1}^{30} i x_i^4 + random[0,1)$$

-1.28 ≤ x_i ≤ 1.28

min $(f_7) = f_7(0, \ldots, 0) = 0$

f08 – Generalized Schwefel's Problem

$$f_8(x) = \sum_{i=1}^{30} \left(x_i \sin\left(\sqrt{|x_i|}\right)\right)$$

-500 ≤ x_i ≤ 500

min $(f_8) = f_8(420.9687, \ldots, 420.9687) = -12596.5$

f09 – Generalized Rastrigin's Problem

$$f_9(x) = \sum_{i=1}^{30} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$$

-5.12 ≤ x_i ≤ 5.12

min $(f_9) = f_9(0, \ldots, 0) = 0$

f10 – Ackley's Function

$$f_{10}(x) = -20e\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{30} x_i^2}\right) - e\left(\frac{1}{30}\sum_{i=1}^{30}\cos(2\pi x_i)\right) + 20 + e$$

-32 ≤ x_i ≤ 32

min $(f_{10}) = f_{10}(0, \ldots, 0) = 0$

f11 – Generalized Griewank's Function

$$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

-600 ≤ x_i ≤ 600

min $(f_{11}) = f_{11}(0, \ldots, 0) = 0$

Anexo B

g01

Min

$$f(x) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$$

$$\begin{aligned} g(x)_1 &= 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0 \\ g(x)_2 &= 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0 \\ g(x)_3 &= 2x_2 + 2x_3 + x_{11} + x_{12} - 10 \leq 0 \\ g(x)_4 &= -8x_1 + x_{10} \leq 0 \\ g(x)_5 &= -8x_2 + x_{11} \leq 0 \\ g(x)_6 &= -8x_3 + x_{12} \leq 0 \\ g(x)_7 &= -2x_4 - x_5 + x_{10} \leq 0 \\ g(x)_8 &= -2x_6 - x_7 + x_{11} \leq 0 \\ g(x)_9 &= -2x_8 - x_9 + x_{12} \leq 0 \end{aligned}$$

$0 \leq x_i \leq 1 (i = 1, ..., 9)$, $0 \leq x_i \leq 100 (i = 10, 11, 12)$ y $0 \leq x_{13} \leq 1$

$min(f) = f(1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1) = -15$

g02

Max

$$f(x) = \left|\frac{\sum_{i=1}^{n} cos^4(x_i) - 2\prod_{i=1}^{n} cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} ix_i^2}}\right|$$

$$\begin{aligned} g(x)_1 &= 0.75 - \prod_{i=1}^{n} x_i \leq 0 \\ g(x)_1 &= \sum_{i=1}^{n} x_i - 7.5n \leq 0 \end{aligned}$$

$n = 20$, $0 \leq x_i \leq 10 (i = 1, ..., n)$

$max(f) = f(x*) = -0.803619$

Anexo C

Micro-algoritmo de evolución diferencial.

La Evolución Diferencial es un algoritmo evolutivo propuesto por Storn y Price [10] para resolver problemas de optimización principalmente en espacios continuos y en el cual las variables se representan mediante números reales. Actualmente se ha utilizado en problemas de alta complejidad con muy buenos resultados [11].

En la ED se genera de forma aleatoria una población inicial de individuos (vectores originales), de los cuales se seleccionan tres, también aleatoriamente y que sean distintos entre sí. A cada uno de estos individuos se le denomina vector objetivo y se denotan por *r1*, *r2* y *r3*, en donde *r3* es el vector base o principal. Con ayuda de un operador especial (*F*) se realiza una combinación lineal con las diferencias entre *r1*, *r2* y *r3*, con la intención de generar nuevos vectores ruidosos, a esta etapa se le conoce como mutación. *F* es entonces, un factor que escala la diferencia entre vectores. Posteriormente se generan vectores de prueba a partir de la recombinación de vectores ruidosos y vectores originales, lo anterior se logra a través de un parámetro denominado tasa de recombinación (*CR*). Para finalizar el algoritmo se procede a seleccionar el vector que se copiará a la generación siguiente, comparando los vectores de prueba con los originales en base a su aptitud.

La versión más común de la ED es la denominada **DE/rand/1/bin**; que fue la que se abordó en este trabajo y se lista en Algoritmo 1, se recomienda referirse a [12] para mayores detalles del mismo. La cantidad de generaciones, el tamaño de la población y los parámetros *CR* y *F*, son definidos por el usuario.

Algoritmo 1. Evolución Diferencial estándar, versión DE/rand/1/bin

Begin
  $G = 0$
  Crear aleatoriamente la población inicial $\vec{x}_G \; \forall i, \; i = 1, \ldots, NP$
  Evaluar $f(\vec{x}_G) \; \forall i, \; i = 1, \ldots, NP$
  For $G = 1$ to G_{max} **Do**
    For $i = 1$ to NP **Do**
      Seleccionar aleatoriamente $r_1 \neq r_2 \neq r_3$
      $j_{rand} = \text{randint}(1,D)$
      For $j = 1$ to D **Do**
        If $(rand_j[0,1) < CR$ or $j = j_{rand})$ **then**
          $u^i_{j,G+1} = x^{r_3}_{j,G} + F\left(x^{r_1}_{j,G} - x^{r_2}_{j,G}\right)$
        Else
          $u^i_{j,G+1} = u^i_{j,G}$
        End if
      End for
      If $\left(f\left(\vec{u}^i_{G+1}\right) \leq f\left(\vec{x}^i_G\right)\right)$ **then**
        $\vec{x}^i_{G+1} = \vec{u}^i_{G+1}$
      Else
        $\vec{x}^i_{G+1} = \vec{x}^i_G$
      End if
    End For
    $G = G + 1$
  End For
End

La versión con población reducida de este mismo algoritmo se observa modularmente en la Figura 2. En nuestros experimentos con ED utilizamos una población de 5 individuos, ya que se comprobó funcionalmente que con un menor número de individuos se obtiene una convergencia prematura y utilizar más de 5 individuos no mejora los resultados. Se determinó que el criterio para alcanzar la convergencia nominal fuera al cabo de 5 generaciones, refiriéndose al ciclo interno del algoritmo. En este rubro también se realizaron pruebas con 10 generaciones, sin observarse alguna mejoría en los resultados.

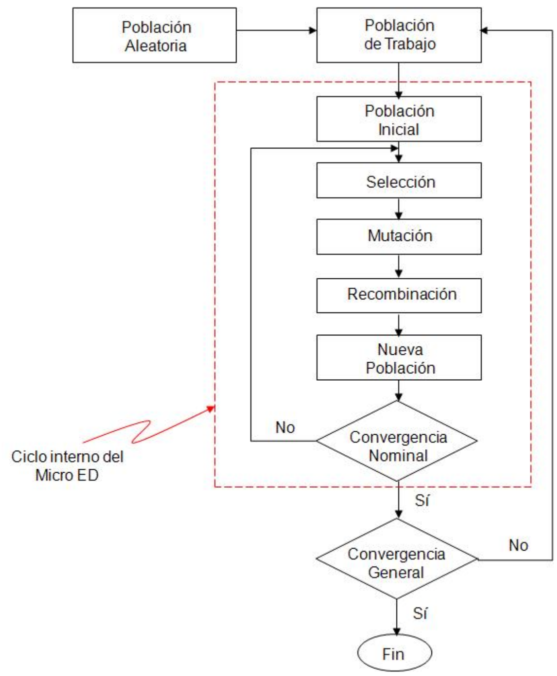

Figura 2. Diagrama a bloques del micro algoritmo de ED.

En la generación cero, utilizando una población aleatoria de 5 individuos, se procedió a copiar los mismos en la población de trabajo y en la población inicial. Nótese que para el ciclo interno en donde aplica la convergencia nominal, el algoritmo ED se implementa sin cambios en sus operadores y estrategias para evaluar la metodología de manera básica. Al alcanzarse la convergencia nominal es necesario determinar cuántos individuos se copiarán a la población de trabajo y aplicar el proceso de reinicialización. Después de una serie de experimentos, se decidió copiar los 4 mejores individuos y el restante se genera aleatoriamente. Los experimentos realizados contemplaron copiar 1, 2, 3 y 4 individuos con la mejor aptitud, sin embargo con 4 individuos se obtuvieron los mejores resultados. En este micro algoritmo la diversidad se mantiene gracias a las etapas de mutación y recombinación del propio algoritmo base de ED y al individuo aleatorio incorporado en la población de trabajo.

1. Experimentos y Resultados

Para los experimentos se utilizaron las siguientes funciones de alta dimensionalidad, referidas en [12]. ***f1*** y ***f2*** son funciones unimodales y separables. ***f5*** es una función multimodal y no separable.

f1 – Esfera de De Jong

$$f(x) = \sum_{i=1}^{30} (x_i)^2 \qquad (1)$$

$-100 \leq x_i \geq 100$
$min\ (f_1) = f_1\ (0, \ldots, 0) = 0$

f2 – Función techo

$$f(x) = \sum_{i=1}^{30} (\lfloor x_i + 0.5 \rfloor)^2 \qquad (2)$$

$-100 \leq x_i \geq 100$
$min\ (f_2) = f_2\ (0, \ldots, 0) = 0$

f5 – Función generalizada de Rosenbrock

$$f(x) = \sum_{i=1}^{29} \left| 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right| \qquad (3)$$

$-30 \leq x_i \geq 30$
$min\ (f_3) = f_3\ (1, \ldots, 1) = 0$

Las tres funciones fueron probadas en el micro algoritmos ED, realizando 20 ejecuciones para cada una y obteniendo los resultados que a continuación se discutirán. Se definieron los parámetros indicados en la Tabla 1, los cuales son sugeridos en [12]:

Tabla 1*: Parámetros utilizados para el micro algoritmo de ED.*

Función	*CR*	*F*
f1	0.9	$\in [0.3, 0.9]$
f2	0.0	$\in [0.3, 0.9]$
f5	0.0	$\in [0.3, 0.9]$

En la Tabla 2 se indican las generaciones utilizadas para cada función, así como los resultados experimentales alrededor del valor óptimo.

Tabla 2*: 20 ejecuciones del micro algoritmo de ED.*

Función	*Ciclo Externo*	*Ciclo interno*	*Mejor*	*Peor*	*Media*
f1	300	5	0.0	0.0064	0.00010
f2	200	5	0.0	1.0	0.20
f5	300	5	0.0	0.0010	0.00012

En la Figura 3, se aprecia el comportamiento del algoritmo de ED estándar en su versión **DE/rand/1/bin** con respecto a la aptitud, para ***f*2**. Se utilizó una población de 60 individuos, 1000 generaciones y los mismos valores de *CR* y *F* listados en la Tabla 1.

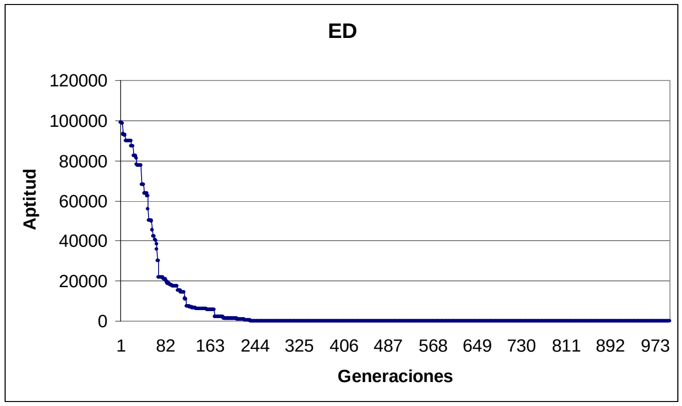

*Figura 3. Aptitud para **f2** con ED estándar.*

Utilizando una población de 5 individuos y los parámetros indicados en la Tabla 2, se obtuvo la gráfica exhibida en la Figura 4, para ***f*2** probada en el micro algoritmo de ED.

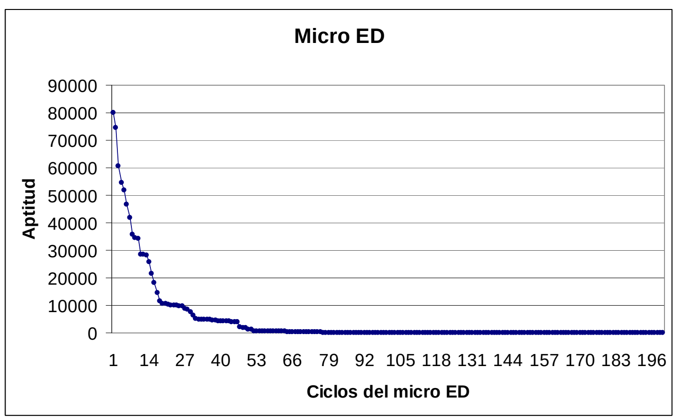

*Figura 4. Aptitud para **f2** con micro algoritmo de ED.*

Es importante notar que para las funciones experimentales, el desempeño del algoritmo de población reducida es tan eficiente como el del algoritmo en su versión estándar. La cantidad de evaluaciones a la función objetivo es similar, considerando el ciclo interno del micro algoritmo.

2. Conclusiones

El micro algoritmo de ED (en su versión *DE/rand/1/bin*) generó buenos resultado con base en la metodología aplicada y con respecto a las funciones seleccionadas. En los resultados experimentales se obtuvo el óptimo para las tres funciones de prueba de alta dimensionalidad, lo que denota resultados alentadores para continuar con la investigación. Es posible afirmar que el desempeño del micro algoritmo de ED es tan bueno como el de su contraparte estándar, aunque con menores requerimientos de espacio en memoria de datos, lo que permite enfocar el diseño hacia aplicaciones con un bajo costo computacional o a implementaciones en circuitos embebidos (p. e. dispositivos de lógica programable y microcontroladores convencionales) refiriéndose al denominado *hardware evolutivo*. El manejo de espacios restringidos y la optimización con múltiples objetivos se pueden incorporar al esquema básico planteado, seguramente éste será un trabajo a futuro, al igual que explorar la adaptación de otros algoritmos bioinspirados comunes en la resolución de problemas de optimización.

Referencias

1. D. Ashlock. Evolutionary Computation for Modeling and Optimization. Springer, first edition, 2005.
2. M. Munetomo, Y. Satake. Enhancing Model-building Efficiency in Extended Compact Genetic Algorithms. Systems, Man and Cybernetics, 2006. ICSMC '06. IEEE International Conference on Volume 3, 8-11 Oct. 2006 Page(s):2362 – 2367.3.
3. J. Torresen. Possibilities and limitations of applying evolvable hardware to real-world application. In Field-Programmable Logic and Applications: 10th International Conference on Field Programmable Logic and Applications (FPL-2000), volume 1896 of Lecture Notes in Computer Science, R. W. Hartenstein et al. (eds.), Springer-Verlag, 2000, pp. 230–239.
4. D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley, Reading, MA, 1989.
5. K. Krishnakumar. Micro-genetic algorithms for stationary and non-stationary function optimization. In SPIE Proceedings: Intelligent Control and Adaptive systems, pages 289–296, 1989.
6. G. Toscano, Carlos. A. Coello. A Micro-Genetic Algorithm for multiobjective optimization. First International Conference on Evolutionary Multi-criterion Optimization. Springer – Verlag, Lecture Notes in Computer Science number 1993, 2001, pp. 126 – 140.
7. J. C. Fuentes, C. A. Coello, Handling Constraints in Particle Swarm Optimization Using a Small Population Size, in Lecture Notes in Computer Science, Springer. MICAI 2007: Advances in Artificial Intelligence, Volume 4827/2007.
8. J. C. Fuentes. Un nuevo Algoritmo de optimización basado en optimización mediante cúmulos de partículas utilizando tamaños de población pequeños. Tesis de Maestría. CINVESTAV- IPN. México, 2008.
9. T. Bäck, D. B. Fogel, and Z. Michalewicz, editors. Handbook of Evolutionary Computation. Institute of Physics Publishing and Oxford University Press, New York, 1997.
10. R. Storn, K. Price. Differential Evolution - a Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. Technical Report TR-95-012, ICSI, March 1995, ftp.icsi.berkeley.edu. Accedido por última vez el 20 de junio de 2009.

Anexo D

Bio-inspired architecture design for a PWM module

Abstract

This paper presents the analysis and design of a Pulse Width Modulation embedded unit as an alternative to convert digital signals to analog signals extrapolating the biological principle of DNA hybridization. Biological hybridization process is a binding event between two complementary DNA single strands and that leads to the formation of a double-stranded helix. The proposed technique to detect the hybridization is based on direct comparison of data, allowing a high degree of parallelism in the Bio-inspired architecture specially designed to regulate the light intensity of LED lamps as a home automation application.

1. Introduction

In digital electronic design, a PWM (Pulse Width Modulation) unit is basically a digital – analog converter (DAC) which can be used in solutions that do not require a high frequency of sampling, it converts a binary data into a series of pulses, so that the pulse duration is directly proportional to the value of binary data. PWM is widely used in different areas of control systems, such as robotics, industrial process control, power control systems, and others [1]. With regard to home automation (domotics), PWM has been very useful in regulating the light intensity of lamps (LED, fluorescent and incandescent), as well as regulating valves and small engines that automate some process [2], [3].

When designing with programmable logic devices (FPGA, CPLD), embedded realization of such converters is common [4], so this paper proposes an alternative to design a PWM unit simulating the combination of DNA strands in order to regulate the light intensity of a LED lamp in an application for home automation and so contribute to energy savings. The artificial DNA hybridization suggests a simple mechanical that compare parallel binary strands stored in data registers looking for evidence that they are complementary to each other to validate a subsequent action as a response.

This approach is very similar to evaluation of inference rules in an expert system that tries to model the reasoning of a human operator [5].

2. Electronic detection of DNA hybridization

DNA (Deoxyribonucleic Acid) is a chemical substance that is found in the nucleus of cells, which stores the basic code of all life translated as biological instructions. In the molecular genetics theory about DNA hybridization, single strands of DNA from two different species are allowed to join together to form hybrid double helices, much like a twisted ladder. These hybrid segments of DNA are used to determine the evolutionary relatedness of organisms by examining how similar (or dissimilar) the DNA base pair sequences are; in other words, the degree of hybridization is proportional to the degree of similarity between the molecules of DNA from the two species.

The way to detect the hybridization of two single strands of DNA has been replicated in the field of electronics through DNA Array, also called DNA chip or Gene array [6]. This chip is a matrix structure on which are distributed DNA single strands that have been implanted into a silicon base. These sequences have a simple fixed value that when they are incubated with strands injected up to the chip, generating helices of DNA that can be detected through electronic tools. Engineering has shown that DNA chips can be used to store and evaluate in parallel, Boolean or fuzzy rules [5], [7].

In electronic design, hybridization can be implemented using a reference register that will be compared in parallel with others test registers, looking for evidence that they are complementary. In Figure 1 test register represents the simple sequence of DNA with a fixed value and reference register is the DNA strand that is injected or introduced to chip. To maintain the analogy with biological hybridization, the reference register is unique and will be compared with all test registers in parallel form.

Note that to make this comparison, both registers must have the same size and only if they are complementary, the flag output will be high logical value. To verify that the bits of both registers are mutually complementary, XOR logic gates are used for bitwise comparison.

3. Pulse width modulation (PWM)

In digital electronic circuits a PWM embedded unit is usually accomplished by connecting a binary counter and a comparator circuit together; this last one will determine when data applied to the entrance of the unit is less than the value of binary counter constantly changing. At the output of the PWM unit, is necessary to connect a low-pass RC filter to determine the voltage of the analog signal equivalent to digital data entered [8]. The entire period of a PWM cycle is equal to the product of the period clock signal reference (system clock) with 2^n, where n is the number of bits of the counter circuit proposed.

In the particularity of this work were considered 4-bit data registers, so in [8] is demonstrated that a converter of more than 8 bits does not get more benefits, so 16 different levels (at a rate of 2^4) of luminous intensity are adequate.

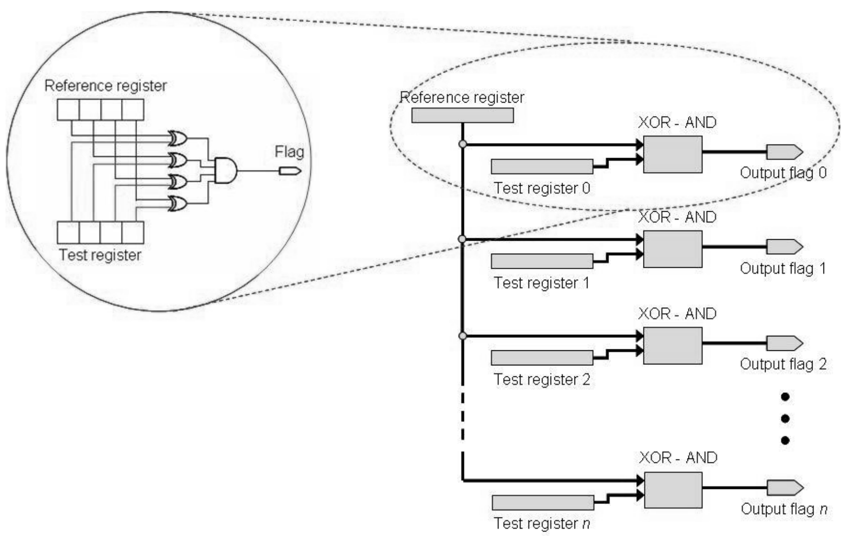

Figure 1. Standard model of artificial hybridization with digital electronic.

4. Bio-inspired architecture

We consider a modular design consisting of three parts. First part of this realization uses a reference register to be compared in parallel with every test registers, simultaneously.
According to the scheme established with 4-bit registers, 16 test registers were designed with fixed and different values, and a single reference register which establishes the digital data to convert to their analog equivalent in a range of 0 to 5 V, for example, a binary input equal to 0000 will get an analog voltage of 0V; in the same way, a binary input equal to 1111 will get the maximum analog value equal to 5V.

The fixed values of the test registers were assigned complementary in correspondence to the values that they can take at the rate of 2^n, in other words, for a binary entry 0000, stored in the reference register, the test register labeled as "0" will store a binary data 1111; for an entry 0001, it was assigned 1110 in the test register labeled as "1". Just the only one output from the module in Figure 1 is a data flag that indicates a successful hybridization through a high logical level.

In the second part of our design each flag from the previous stage actives individually a tri-state data register which is connected to a common output bus. This time were considered 16-bit tri-state registers to improve the resolution of the converter. In this scheme, only a single tri-state register may be enabled at a time. Each tri-state register has a fixed value assigned according to the binary value that will be delivered as result of modulation.

The third and final stage of architecture, is the stage of control, consisting of a 16-bit multiplexer that with help of a 4-bit binary counter, performs a binary sweep of the only tri-state register enabled, starting with the most significant bit. The same binary counter determines that after 16 pulses clock, when the unit PWM has finished data conversion, the reference register can receive a new data to process.

Figure 2 shows the schematic diagram of the complete unit designed with the three parts. All modules were described in VHDL for logic synthesis.

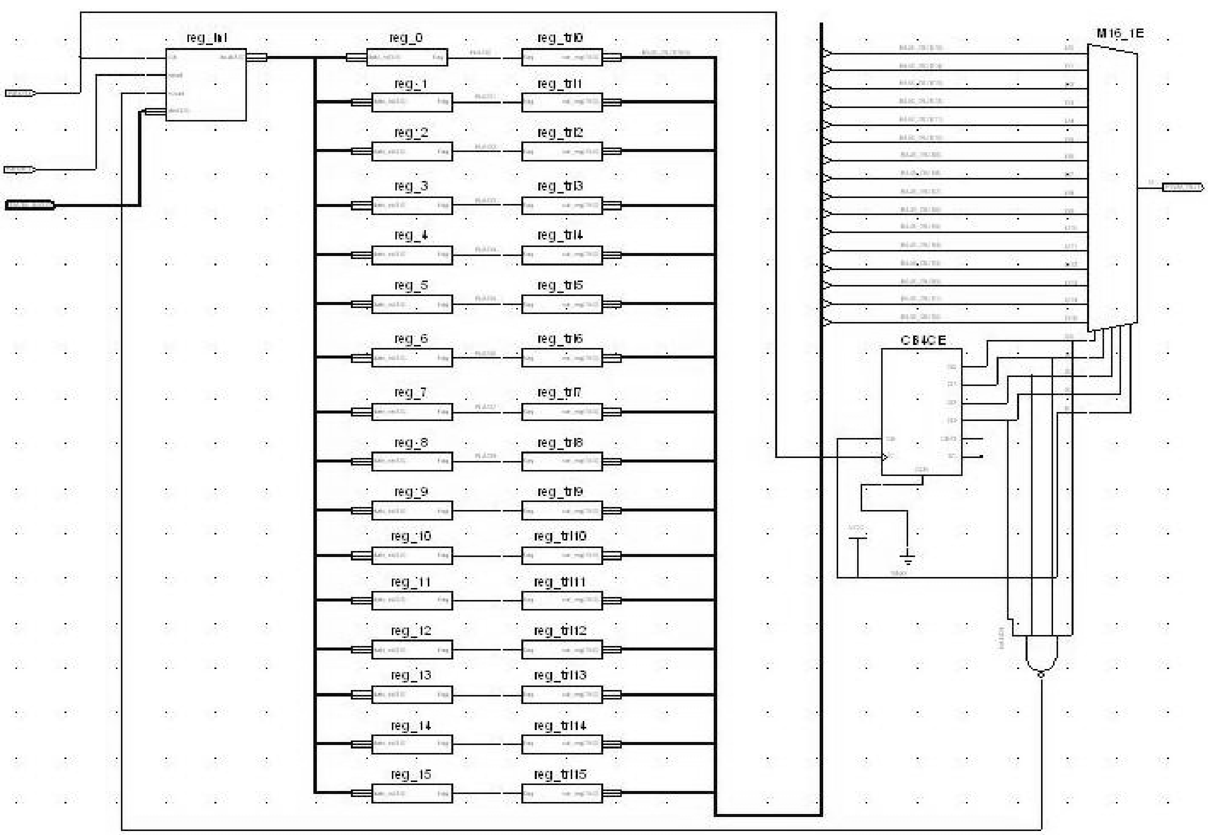

Figure 2. PWM parallel architecture using the principle of hybridization of DNA strands.

5. Results

The logic synthesis design was carried out using ISE Webpack v.8.1i, to configure a 3S200 Spartan 3 FPGA of vendor Xilinx. The PWM Bio-inspired unit implemented uses only 4% of the amount of internal resources of FPGA.

The PWM unit was tested at different frequencies, in each case calculating the values of RC filter. They were considering the following clock frequencies: 4MHz, 1MHz, 700Hz, 320Hz and 120Hz. Brief details of the conversion can be seen in Figure 3. It verifies the correct operation of PWM generator.

Tri-state registers	*Binary data*
reg_tri0	0000000000000000
reg_tri1	1000000000000000
reg_tri2	1100000000000000
reg_tri3	1110000000000000
reg_tri4	1111000000000000
reg_tri5	1111100000000000
reg_tri6	1111110000000000
reg_tri7	1111111000000000
reg_tri8	1111111100000000
reg_tri9	1111111110000000
reg_tri10	1111111111000000
reg_tri11	1111111111100000
reg_tri12	1111111111110000
reg_tri13	1111111111111000
reg_tri14	1111111111111100
reg_tri15	1111111111111110

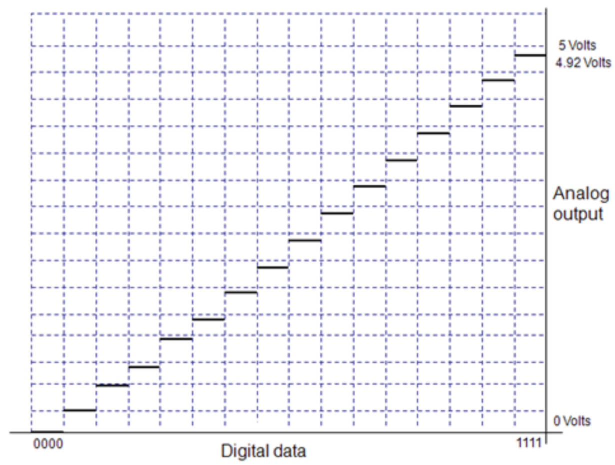

Figure 3. Details of the digital–analog conversion in practical experiments.

It was necessary to design an electronic power stage to attach the PWM output with the LED lamp used in laboratory. In Figure 4 we show the lamp used in our experiments. This lamp was designed with similar characteristics as commercial lamps with a panel of 18 x 18 high brightness LEDs.

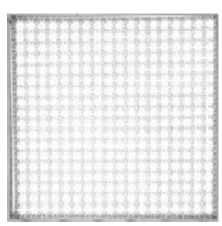

Figure 4. LED lamp used in laboratory.

6. Conclusions

The artificial hybridization of DNA single strands allows directing a design based on inference rules, into a natural parallelization that can be explored through implementation in programmable logic devices. This technique transferred to Bio-inspired hardware has allowed the completion of a functional PWM generator as an alternative to convert digital signals to analog signals.

Each tri-state register enabled individually, as a result of a successful hybridization, stores a value that can be updated in real-time increasing the scope of this proposal. This will allow attacking the quantification problem, which directly affects the converter resolution and that a conventional PWM unit could not stand it.

This modular design supports changing the content of the registers according to the needs of modulation, being possible to increase or decrease the resolution of the conversion.

It will be interesting to explore the capabilities of parallel evaluation of inference rules in expert systems more complex; in future work we will apply this same technique to create intelligent systems that allow the automatic regulation of the luminous intensity in conventional lamps, focusing more specialized home automation applications, encouraging a systematic energy saving.

References

[1] Zrilic, D. G.: Alternative Approach to Use of Pulse Width Modulation, Automation Congress, 2006. WAC '06. World IEEE, 24-26 July 2006, pp. 19–24.

[2] Alonso J. M.; Cardesin, J.; Calleja, A. J.; Rico-Secades, M.; García, J.: A fluorescent lamp electronic ballast for railway applications based on low cost microcontroller, Industry Applications Conference, 2003. 38th IAS Annual Meeting IEEE. Conference Record of the Volume 1, Issue , 12-16 Oct. 2003, pp. 523 – 530.

[3] Sandu, F.; Romanca, M.; Nedelcu, A.; Borza, P.; Dimova, R.: Remote and mobile control in domotics, Optimization of Electrical and Electronic Equipment, 2008. OPTIM 2008. 11th International Conference on 22-24 May 2008, pp. 225 – 228.

[4] Dan, D.; Su, C.; Joss, G.: FPGA implementation of PWM pattern generators [for PWM invertors], Canadian Conference on Electrical and Computer Engineering IEEE, 2001, vol.1, pp. 225-230.

[5] Delgado, A.: Rule base evaluation using DNA chips, Proceedings American Control Conference, pp. 3242 – 3245, Anchorage - Alaska, Mayo 8-10, 2002.

[6] Moeller, R.; Fritzsche, W.: Chip-based electrical detection of DNA. Nanobiotechnology, IEE Proceedings - Vol 152, Issue 1, 4 Feb. 2005, pp. 47 – 51.

[7] Delgado, A.: DNA chips as lookup tables for rule based systems. IEE Computing and Control Engineering Journal 2002, Vol. 13, No. 3, pp. 113-119.

[8] Yu, Z.; Mohammed, A.; Panahi, I.: A Review of three PWM Techniques, American Control Conference, 1997, Proceedings of the 1997. Vol. 1, 4-6 June 1997, pp. 257 – 261Khalil, Hassan K. Nonlinear Systems. Third Edition. Ed. Prentice Hall. 2002. ISBN 0-13-06-7389-7.