ارائهٔ ابزاری کارآمد برای سنتز سطح بالای مبدل‌های دیجیتال مدارهای VLSI

محمدرضا اسماعیلی، سید حمید ظهیری* و سیدمحمد رضوی
دانشکده مهندسی برق و کامپیوتر، دانشگاه بیرجند، بیرجند، ایران

چکیده

امروزه مبدل‌های دیجیتال از مهم‌ترین ادوات یک سامانه پردازش سیگنال و داده به‌شمار رفته و به‌صورت گسترده در زمینه پردازش صوت، تصویر و سیگنال‌های حیاتی به‌کار گرفته می‌شوند. در طراحی مبدل‌های دیجیتال VLSI، سنتز سطح بالا (HLS) یکی از مراحل مهم و تأثیرگذار به شمار می رود. هدف اصلی از انجام این کار، کمینه‌کردن واحدهای پایه دیجیتالی مورد استفاده در پروژه مفروض جهت بهبود توان، تأخیر، و سطح مصرفی آن‌ها است. این کار عمدتاً با تحلیل گراف مسیر داده (DFG) اتفاق می‌افتد. بهبود در این مرحله، علاوه‌بر بازدهی بیشتر باعث کاهش زمان طراحی در مراحل پایین‌تر می‌شود. ماهیت پیچیده، گسترده و گسسته مسائل سنتز سطح بالا، باعث شده است که آنها در زمره مسائل بسیار دشوار در مهندسی مدارات VLSI به‌شمار آیند؛ از این‌رو استفاده از روش‌های فراابتکاری و هوش‌جمعی جهت حل پروژه‌های مرتبط با سنتز سطح بالا، گزینه‌ای مطلوب به نظر می‌رسد. در این مقاله روشی مبتنی‌بر الگوریتم فراابتکاری "شعله و پروانه"(MFO) جهت یافتن بهترین طرح سخت‌افزاری برای انواع مبدل‌های دیجیتال ارائه شده است. نتایج مقایسه‌ای در کنار نتایج حاصل از روش مبتنی بر الگوریتم ژنتیک (GA) نشان داد که روش پیشنهادی از توانایی بالاتری در ارائه ساختار سخت‌افزاری مناسب و سنتز سطح بالای انواع مبدل‌ها برخوردار است. همچنین ویژگی دیگر روش پیشنهادی، سرعت بالای آن در یافتن پاسخ بهینه است (میانگین برتری بیش از ۲۰٪ نسبت به GA).

واژگان کلیدی: سنتز سطح بالا، مسیرداده، مبدل‌های دیجیتال، الگوریتم‌های فراابتکاری، الگوریتم بهینه‌سازی شعله و پروانه

An efficient CAD tool for High-Level Synthesis of VLSI digital transformers

Mohammad Reza Esmaeili, Seyed Hamid Zahiri* & Seyed Mohammad Razavi
Department of Electrical Engineering, Faculty of Electrical Engineering and Computer, University of Birjand, Birjand, Iran

Abstract

Digital transformers are considered as one of the digital circuits being widely used in signal and data processing systems, audio and video processing, medical signal processing as well as telecommunication systems. Transforms such as Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT) and Fast Fourier Transform (FFT) are among the ones being commonly used in this area. As an illustration, the DCT is employed in compressing the images. Moreover, the FFT can be utilized in separating the signal spectrum in signal processing systems as fast as possible. The DWT is used in separating the signal spectrum in a variety of applications from signal processing to telecommunication systems, as well.

In order to build a VLSI circuit, several steps have to be taken from chip design to final construction. The first step in the synthesis of the integrated circuits is called high-level synthesis (HLS), in which a structural characteristic is obtained from a behavioral or algorithmic description. The resulting structural characteristic is equivalent to the one being considered in the behavioral description

* Corresponding author
*نویسندهٔ عهده‌دار مکاتبات

●تاریخ ارسال مقاله: ۱۳۹۷/۱۲/۱۴ ●تاریخ پذیرش: ۱۳۹۸/۱۱/۲ ●تاریخ انتشار: ۱۴۰۰/۱۰/۲۷ ●نوع مطالعه: پژوهشی

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]
[DOI: 10.52547/jsdp.18.3.3]

and it somehow represents the method for implementing the behavioral description as a result several structural descriptions could be implementable for each behavioral description. Therefore, depending on the intended use, the characteristic will be selected that outperforms the others. The main purpose of the HLS is to optimize the power consumption, the chip occupied area and delayed and is fulfilled by selecting the appropriate number of operating units and how they are implemented to the operators. This is generally accomplished through a graph analysis called the data flow graph (DFG) which is a graphical representation of the type and how the operators connect. In the DFG, each node is equivalent to an operator while the edges represent the relationship between these operators.

Experience has proved that if the level of design optimization is high, in addition to higher efficiency, the design time will be lower, which is why the researchers are far more interested in optimization at higher levels of design than the lower levels. The complex, extensive, and discrete nature of the HLS problems have been ranked them among the most complex problems in VLSI circuits engineering. Bearing this mind, using meta-heuristic and Swarm intelligence methods to solve high-level synthesis projects seems to be a favored option. In this paper, a heuristic method called Moth-Flame Optimization (MFO) has been used to solve the HLS problem in the design of digital transformer to find the optimal response. The MFO is a population-based heuristic algorithm that optimizes the problems using the laws of nature. The leading notion behind the MFO algorithm inspired from the moths' movements and their instinctive navigation during the night. In the MFO algorithm, the moths are like chromosomes in the GA and like the particles in the PSO algorithm. In order to compare and prove the efficiency of the proposed method, it was applied on the test data with the GA-based method separately but with the same initial conditions. The comparative results along with the results of the GA-based method demonstrated that the proposed method exhibits a higher ability to provide the appropriate hardware structure and high-level synthesis of various types of transformers. Another outstanding feature of the proposed method is its high speed of finding an optimal response with an average of more than 20% greater than the GA based method.

Keywords: High-Level Synthesis, Datapath, Digital Transformers, Meta-heuristic Algorithm, MFO Algorithm

۱- مقدمه

با پیشرفت فناوری و افزایش پیچیدگی طراحی‌ها و به‌جهت افزایش سرعت در طراحی‌های مدارات مجتمع با مقیاس‌پذیری بسیار بزرگ VLSI، نیاز به توسعه راه‌کارهای خودکار بهینه‌سازی به‌شدت احساس می‌شود. یکی از مراحل مهم در طراحی مدارات VLSI، سنتز سطح بالا (HLS) است. HLS، فازی از طراحی سیستم است که در آن از یک توصیف رفتاری و یا توصیف الگوریتمی، در یک زبان سطح بالا مانند زبان برنامه‌نویسی C و MATLAB و یا یک شمای گرافیکی مثل گراف مسیر داده (DFG) به یک مشخصه ساختاری می‌رسیم [1]. مشخصه ساختاری، با آنچه که در توصیف رفتاری مد نظر است؛ هم‌ارز بوده و به‌نوعی بیان‌گر روش پیاده‌سازی هتوصیف الگوریتمی و یا توصیف رفتاری است. برای هر توصیف رفتاری، چندین مشخصه ساختاری وجود دارد که خروجی همه آن‌ها یکسان و برابر با آن چیزی است که در توصیف رفتاری مد نظر بوده است. با این وجود تمام این مشخصه‌های ساختاری ویژگی‌های یکسانی ندارند و هر کدام ممکن است بسته به کاربرد موردنظر نسبت‌به سایرین از لحاظ توان مصرفی، سطح اشغال‌شده بر روی تراشه، و یا سرعت عملکرد و تأخیر، برتر باشد [2]. اهداف اصلی که در حین HLS باید بهینه شوند از قبیل موارد ذکرشده مانند تأخیر، سطح اشغال‌شده و توان، به‌طورمعمول در تضاد با یکدیگر هستند. بدین سبب راه‌کاری مورد نیاز است که بتوان به‌وسیله آن فضای طراحی را جستجو کرده، بهینه‌ترین جواب را بسته‌به کاربرد موردنظر پیدا کرد.

مسائل HLS جزو مسائل سخت مهندسی به‌شمار می‌آیند [3]. در حالت کلی حل مسائل HLS تا حدودی شبیه به مسأله معروف و سختِ فروشنده دوره‌گرد است [4] و با افزایش ابعاد مسأله، فضای طراحی نیز به‌صورت نمایی و به‌شدت افزایش می‌یابد؛ بنابراین قاعدتاً راه‌کارهای دستی برای حل چنین مسائلی مناسب نیستند، مگر برای مسائل با ابعاد بسیار کوچک. بدین سبب در سال‌های اخیر راه‌کارهای خودکار مبتنی‌بر سامانه‌های رایانه‌ای جهت جستجوی فضای طراحی و یافتن پاسخ‌های بهینه، توسعه داده شده‌اند.

در [5]، مراحل زمان‌بندی و اتصال در سنتز سطح بالا به‌صورت هم‌زمان انجام می‌گیرند. در این روش اولویت‌ها و حق تقدم‌ها و محدودیت‌های زمانی تحت برنامه‌ریزی خطی عدد صحیح (ILP)[1] فرموله می‌شوند. در [6-8] نیز بهینه‌سازی توان با استفاده از راه‌کار ILP صورت می‌پذیرد. سرعت بسیار پایین این روش در رسیدن به پاسخ، استفاده از آن را برای مسائل با ابعاد بزرگ درعمل غیرممکن می‌کند [9]. در [8, 10] استفاده از نظریه بازی‌ها

[1] Integer Linear Programming

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]

[DOI: 10.52547/jsdp.18.3.3]

مورد توجه قرار گرفته است. در [10]، با استفاده از نظریه بازی‌ها یک DFG زمان‌بندی‌شده، از لحاظ توان بهینه شده است. در این راه‌کار، تابع‌های عملیاتی به‌عنوان بازی‌کننده‌ها در نظر گرفته شده‌اند و تابع هزینه نیز همان توان‌مصرفی فرض شده است. بار محاسباتی این روش نیز بسیار سنگین بوده و زمان اجرای بسیار بزرگی را می‌طلبد، به‌خصوص در مورد مسائل واقعی و با ابعاد بالا که این هزینه بسیار چشم‌گیر بوده و به‌صورت نمایی افزایش پیدا می‌کند.

یک راه‌کار دیگر برای حل مسائل HLS بدین صورت است که برای حل مسأله ابتدا از یک جدول زمان‌بندی پیش‌فرض استفاده شده و مسأله حل می‌شود. سپس با اعمال مکرر تبدیلات، به‌سمت بهبود جواب اولیه حرکت می‌کنند [11]. ایراد اساسی راه‌کار موجود در این است که جواب‌های به‌دست‌آمده از این طریق به‌شدت به تبدیل مورد استفاده و روش‌های انتخاب تبدیل مناسب حساس هستند. در [12] با استفاده از روش PPL[1] به حل مسأله سنتز سطح بالا پرداخته شده است. استفاده از الگوریتم بازپخت شبیه‌سازی‌شده[2] نیز در [13-15] مورد توجه قرار گرفته است. در [16]، نویسندگان با استفاده از الگوریتم SA اقدام به زمان‌بندی، تخصیص و کاهش ثبات‌ها به‌صورت هم‌زمان کرده‌اند. روش دیگری به نام SALSA که در [17, 18] ارائه شده است نیز، تکنیک‌هایی بر پایه الگوریتم بازپخت شبیه‌سازی‌شده هستند که جهت حل مسائل سنتز سطح بالا مورد استفاده قرار گرفته‌اند. در [19] نیز از روشی بر پایه تکامل شبیه‌سازی‌شده جهت اجرای هم‌زمان مراحل زمان‌بندی و اتصال بهره گرفته شده است. در [20] پژوهش‌گران از روش MPC[3] بهره گرفته‌اند.

ساده‌ترین راه‌کار موجود، اجرای مستقل مراحل سنتز سطح بالاست. در این روش‌ها در هر زمان فقط یک عمل‌گر زمان‌بندی می‌شود و این روند به‌همین ترتیب ادامه پیدا می‌کند تا تمام عمل‌گرها زمان‌بندی شوند. پس از اینکه زمان‌بندی تمام شد، تخصیص واحدهای عملیاتی اجرا می‌شود. الگوریتم‌هایی مانند زمان‌بندی هر چه زودتر (ASAP)[4] [21]، زمان‌بندی هر چه دیرتر (ALAP)[5] [21]، زمان‌بندی مبتنی‌بر مسیر[6] [22]، زمان‌بندی نیروی هدایت شده[7] [23-26] و زمان‌بندی مسیر بحرانی[8] [27] از این دست روش‌ها هستند. تعداد زیادی از این روش‌ها از روش‌های هوشمند حریصانه نیز در حل مسائل کمک می‌گیرند [3, 23, 28]. الگوریتم‌هایی همانند الگوریتم ژنتیک و نسخه‌های مختلف آن نیز در چندین پژوهش ارائه شده‌اند [4, 29-33]. در [31] نویسندگان روشی مبتنی‌بر WSGA جهت بهینه کردن تأخیر و سطح اشغال شده در سنتز مسیر داده ارائه داده‌اند. در این پژوهش، نویسندگان با استفاده از الگوریتم WSGA اقدام به زمان‌بندی و تخصیص واحدهای عملیاتی به‌صورت هم‌زمان برای سنتز رفتاری DFG کرده‌اند. در [32]، نمونه دیگری از الگوریتم ژنتیک به نام WGA برای جستجوی فضای طراحی در طی فرآیند زمان‌بندی DFG، مورد استفاده قرار گرفته است. در [4]، نویسندگان از یک روش تکاملی چندهدفه به نام NSGA-II استفاده کرده‌اند. روش پیشنهادی در این پژوهش نیز، زمان‌بندی و تخصیص واحدهای عملیاتی را به‌طور هم‌زمان انجام می‌دهد. در [33] یک طرح و روش کاوش فضای طراحی با استفاده از الگوریتم NSGA-II برای بهینه‌سازی سطح و تأخیر، در مسائل HLS ارائه شده است. در این روش از رگرسیون خطی جهت محاسبه توابع هدف سطح و تأخیر استفاده شده است. از الگوریتم بهینه‌سازی گروه ذرات نیز در [4, 34-36] استفاده شده است. در [34]، روشی بر پایه الگوریتم PSO و با فرض منابع محدودبودن مسأله جهت کاهش زمان زمان‌بندی استفاده شده است. در [35] نویسندگان روشی برمبنای PSO ارائه داده‌اند که در آن زمان‌بندی و اتصال در مسیرداده یک فیلتر به‌صورت هم‌زمان صورت می‌پذیرد. در [4] نویسندگان جهت حل مسأله HLS روشی را بر پایه PSO به نام WPSO ارائه داده‌اند. همچنین پژوهش‌گران در [37] روشی را ارائه داده‌اند که در آن با استفاده از الگوریتم بهینه‌سازی تغذیه باکتریایی[9] اقدام به حل مسائل HLS نموده‌اند. در [38] با استفاده از الگوریتم ممتیک[10] سرعت و توان بهینه شده است. در [39] نیز با استفاده از الگوریتم کرم شب‌تابِ[11] بهبودیافته، به جستجوی فضای طراحی در مسأله سنتز سطح بالا پرداخته شده است. در این روش از ترکیب الگوریتم کرم شتاب و جستجوی ممنوعه[12] استفاده و درنهایت مسأله مورد نظر از لحاظ سرعت و توان بهینه شده است.

[1] parallel programming language
[2] Simulated Annealing
[3] Model Predictive Control
[4] As Soon As Possible
[5] As Late As Possible
[6] Path-based scheduling
[7] Force-directed scheduling
[8] Critical path scheduling
[9] Bacterial foraging optimization algorithm
[10] Memetic Algorithm
[11] Firefly Algorithm
[12] Tabu Search

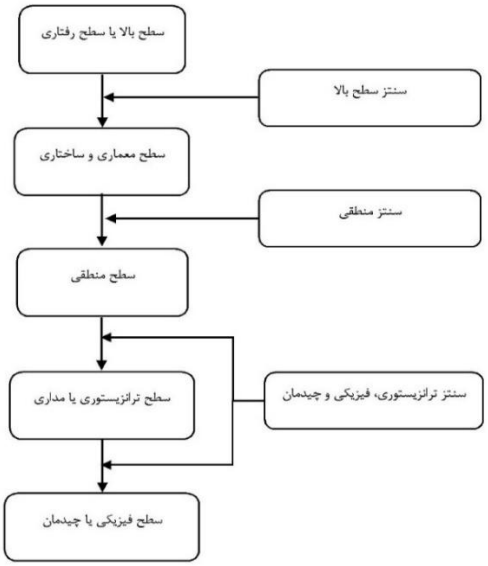
[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]
[DOI: 10.52547/jsdp.18.3.3]

وجود اهداف متضاد متعددی که باید بهینه شوند و همچنین وجود مراحل وابسته به یکدیگر در سنتز سطح بالا، استفاده از الگوریتم‌ها و روش‌های تکاملی و ابتکاری را در حل اینگونه مسائل جذاب می‌کند. الگوریتم‌هایی همانند الگوریتم ژنتیک و الگوریتم گروه ذرات از این جمله‌اند. این الگوریتم‌های تکاملی زیستیِ مبتنی‌بر جمعیت، این توانایی را دارند که گروه زیادی از جواب‌ها را، جهت اهداف مختلف، در یک اجرا ارائه دهند. در این مقاله نیز جهت سنتز سطح بالای مبدل‌های دیجیتال روشی مبتنی‌بر الگوریتم فراابتکاری MFO [1] ارائه شده است.

پیکربندی این مقاله بدین صورت است که در بخش ۲ سنتز سطح بالای مدارات VLSI توضیح داده شده است. در بخش ۳ روش پیشنهادی برمبنای الگوریتم فراابتکاری MFO ارائه و در بخش ۴ نتایج حاصل از شبیه‌سازی آورده شده است. در این بخش مقایسه‌ای از روش پیشنهادی با روش مبتنی‌بر الگوریتم ژنتیک نیز انجام و درنهایت بخش ۵ به نتیجه‌گیری اختصاص داده شده است.

۲- سنتز سطح بالای مدارات VLSI

جهت این‌که یک مدار VLSI پیاده‌سازی شود و بتوان از آن استفاده کرد، مراحل مختلفی باید طی شود. نخستین مرحله در طراحی مبدل‌های VLSI تعریف توصیف رفتاری و الگوریتمی مدار است. توصیف رفتاری کدی است در زبان برنامه‌نویسی C، MATLAB و یا VHDL که بیان‌گر شرایط و ویژگی‌های واحدهای محاسباتی مانند جمع‌کننده‌ها، ضرب‌کننده‌ها و ... است. این کد نحوه اجرای دستورات موردنظر را نشان می‌دهد. مرحله بعد تبدیل کد نوشته شده به یک توصیف ساختاری است که عملگرهای موردنظر در کد را به واحدهای عملیاتی در مسیرداده و یا واحد کنترلی نگاشت می‌کند [40, 41]. این توصیف و مشخصه ساختاری یا معماری را RTL [2] می‌نامند. پس از به‌دست‌آوردن طرح RTL موردنظر، این طرح به‌وسیله روش‌های سنتز منطقی به سطح گیت‌های منطقی تبدیل می‌شود. در این مرحله از سنتز، متناسب با آنچه در سطح RTL توصیف شده بود، گیت‌های منطقی در طرح قرار داده می‌شوند. مرحله بعد در طراحی VLSI، سطح ترانزیستور یا مداری نامیده می‌شود. در این مرحله مشخص می‌شود که مدار طراحی‌شده در مراحل قبل، توسط چه نوع ترانزیستورهایی و با چه فناوری پیاده‌سازی شود. مرحله بعد در طراحی، سنتز چیدمان است. این مرحله مشخص می‌کند که گیت‌های منطقی چگونه و به چه نحوی در طرح نهایی بر روی سطح تراشه که اغلب از جنس ویفر سیلیکون در نظر گرفته می‌شود، پیاده‌سازی شود. و درنهایت آخرین مرحله از طراحی VLSI، ساخت مدار موردنظر و پیاده‌سازی آن بر روی سطح ویفر با استفاده از ادوات و فناوری‌های موجود است. پس از این مرحله قطعه تولید شده می‌تواند در ادوات الکترونیکی خاص و بسته‌به کاربرد آن مورد استفاده قرار گیرد. شکل (۱) مراحل مختلف سنتز یک مدار VLSI را نشان می‌دهد که در آن طی مراحل مختلف طراحی، از یک توصیف رفتاری در زبان‌های سطح بالا به یک قطعه الکترونیکی ساخته شده بر روی ویفر سیلیکون می‌رسیم.



(شکل-۱): مراحل پیاده‌سازی یک طرح VLSI
(Figure-1): Implementation steps of a VLSI design

اگر مراحل اولیه طراحی را طراحی سطح بالا بنامیم و به‌همین ترتیب سایر مراحل را در سطوح پایین‌تر قرار دهیم، نخستین مرحله از سنتز یک تراشه را که در آن از یک توصیف رفتاری یا توصیف الگوریتمی به یک مشخصه ساختاری مانند RTL می‌رسیم، سنتز سطح بالا یا HLS می‌نامند.

مشاهده می‌شود که سطوح تجزیه و سنتز دیگری نیز وجود دارند که پایین‌تر از HLS قرار می‌گیرند، با این وجود تجربه ثابت کرده است که اگر جستجوی فضای طراحی در سطوح بالاتر انجام شود بازده به‌مراتب بهتری نسبت‌به حالتی دارد که این کار در سطوح پایین‌تر مانند سطوح منطقی و ترانزیستوری صورت می‌پذیرد [42].

توصیف رفتاری، مشخص‌کنندهٔ رفتار ورودی‌ها و خروجی‌ها فارغ از جزئیات موجود در سطوح پایین‌تر است،

[1] Moth-Flame Optimization
[2] Register Transfer Level

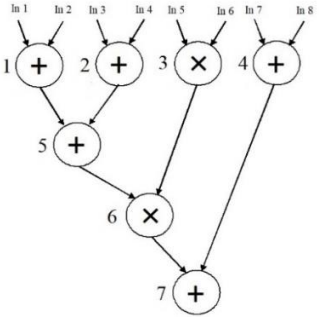
[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]
[DOI: 10.52547/jsdp.18.3.3]

که تحت یک زبان سطح بالا مانند زبان برنامه‌نویسی C، MATLAB و یا یک نمودار گرافیکی به نام گراف جریان داده (DFG) ارائه شده است. یک DFG، عمل‌گرهای موجود در یک توصیف رفتاری و تمام وابستگی‌های بین آن‌ها را به‌صورت گرافیکی نمایش می‌دهد. شکل (۲) نمونه‌ای از یک DFG را نشان می‌دهد. در هر DFG، گره‌ها بیانگر عمل‌گرها و یال‌ها تعیین‌کننده وابستگی بین این عمل‌گرها هستند. به‌طور مثال در شکل یادشده، پنج واحد عملیاتی جمع‌کننده و دو واحد عملیاتی ضرب‌کننده وجود دارد. در این مثال خاص نتایج حاصل از اجرای گره ۱ و ۲، برای اجرای گره ۵ مورد نیاز هستند. این مطلب را می‌توان برای گره ۶ هم در نظر گرفت که در آن نتایج حاصل از اجرای گره‌های ۳ و ۵، برای اجرای این گره لازم هستند.

یک راه‌کار ساده جهت پیاده‌سازی DFG شکل (۲)، متناظرکردن یک‌به‌یک هر گره در DFG با واحدهای عملیاتی مجزا است. برای مثال در این شکل، پنج عدد واحد جمع‌کننده و دو عدد واحد ضرب‌کننده جهت اجرای الگوریتم، مورد نیاز است. با این وجود به‌صورت محسوسی مشاهده می‌شود که این راه‌کار و تناظر یک‌به‌یک، باعث افزایش شدید در سطح اشغال‌شده بر روی تراشه می‌شود، بالاخص برای DFG هایی با تعداد گره‌های فراوان، این افزایش به‌شدت چشم‌گیر است. همچنین تعداد بالای واحدهای عملیاتی سبب افزایش توان‌مصرفی نیز می‌شود. راه‌کار دیگر جهت بهبود عملکرد و کاهش تعداد واحدهای عملیاتی، که به‌نوبه خود باعث کاهش سطح اشغال‌شده و کم‌شدن توان‌مصرفی می‌شود، به اشتراک‌گذاری واحدهایی است که عملیات یکسانی را انجام می‌دهند.

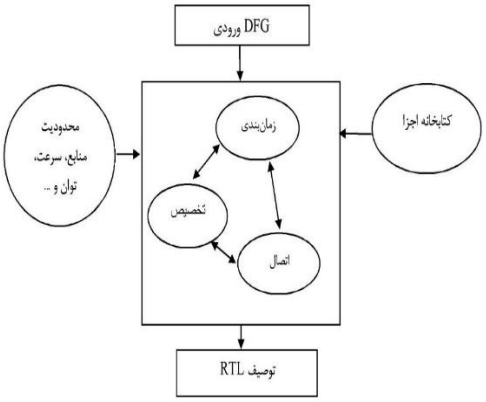
سنتز سطح بالا را می‌توان به دو بخش زیر تقسیم کرد:

- مسیرداده: که شامل واحدهای عملیاتی و حافظه می‌شود.
- واحد کنترل: که هماهنگ‌کننده جریان اطلاعات بین المان‌های مسیر داده است.



(شکل-۲): نمونه‌ای از یک DFG
(Figure-2): A sample of a DFG

بر اساس همین تقسیم‌بندی، سنتز سطح بالا (HLS) را می‌توان به دو بخش سنتز مسیرداده و سنتز کنترل‌کننده تقسیم کرد. سنتز مسیرداده بخش مهمی از HLS است که بهبود در این مرحله باعث بهبود عملکرد کلی سامانه می‌شود. سنتز مسیرداده نیز به‌نوبه خود شامل سه مرحله وابسته به یکدیگر است: زمان‌بندی[1]، تخصیص[2] و اتصال[3]، شکل (۳) شمایی کلی از این سه مرحله را نشان می‌دهد.



(شکل-۳): زیرکارهای HLS
(Figure-3): HLS subtasks

زمان‌بندی: زمان‌بندی، مجموعه‌ای از عملکردها در توصیف رفتاری را به مجموعه‌ای از گام‌های زمانی گسسته نگاشت می‌کند، به‌گونه‌ای که در آن تمام وابستگی‌ها و حق تقدم‌های مشخص شده، رعایت می‌شوند. یا به‌عبارت ساده‌تر، زمان‌بندی، گام زمانی را تعیین می‌کند که در آن یک گره اجرا می‌شود. شکل (۴) یک نمونه از زمان‌بندی‌شدنی را برای DFG ارائه شده در شکل (۲) نمایش می‌دهد. همان‌طور که ملاحظه می‌شود با توجه به زمان‌بندی انجام‌شده، DFG موردنظر در چهار گام زمانی اجرامی‌شود. گره ۱، ۲، ۳ و ۴ در گام زمانی نخست، گره ۵ در گام زمانی دوم و گره ۶ و ۷ در گام‌های زمانی سوم و چهارم اجرا می‌شوند. به مجموع گام‌های زمانی که در آن رفتار موردنظر اجرا می‌شود، طول زمان‌بندی می‌گویند. در مثال یادشده، طول زمان‌بندی برابر ۴ است، چون چهار گام زمانی مورد نیاز است تا الگوریتم اجرا شود.

تخصیص: تخصیص، تعداد منابع سخت‌افزاری را که می‌توانند به‌وسیله عملگرهای مشخص شده در توصیف رفتاری به اشتراک گذاشته شوند، تعیین می‌کند. به‌عنوان مثال، یک تخصیص ممکن برای DFG شکل (۲)، سه جمع‌کننده و یک ضرب‌کننده می‌تواند باشد، که با توجه به DFG زمان‌بندی‌شده شکل (۴)، این امر قابل درک است.

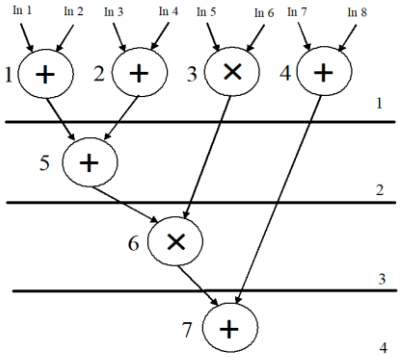
[1] Scheduling
[2] Allocation
[3] Binding

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]
[DOI: 10.52547/jsdp.18.3.3]

برخی از گره‌ها قابلیت تحرک[1] دارند، به‌گونه‌ای که به آن‌ها اجازه می‌دهد که در گام‌های زمانی مختلف اجرا شوند، بدون اینکه تأثیری بر روی ماهیت کلی مسأله داشته باشند. برای مثال در DFG زمان‌بندی‌شده شکل (۵)، گره ۳ به گام زمانی دوم و گره ۴ به گام زمانی سوم منتقل شده است. همان‌طور که مشاهده می‌شود، مزیت اجرای گره ۴ در گام زمانی سوم در این است که تعداد جمع‌کننده‌ها را می‌توان از سه به دو عدد کاهش داد، بدون این‌که در فرآیند اجرای الگوریتم خللی پیش بیاید.

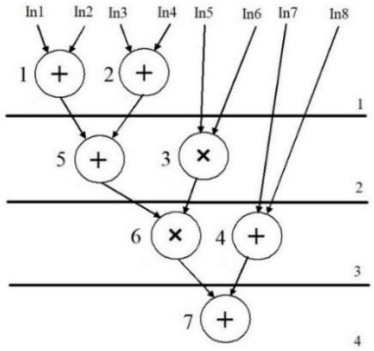
اتصال: در مرحله قبل تعداد منابع مورد نیاز برای پیاده سازی رفتار موردنظر مشخص شدند. حال در مرحله اتصال به‌طوردقیق مشخص می‌شود که کدام گره به کدام واحد عملیاتی اختصاص داده شود. به‌طور مثال در مورد شکل (۵)، می‌توان گره ۱ را به جمع‌کننده اول و گره ۲ را به جمع‌کننده دوم اختصاص داد.

این مراحل می‌توانند به‌صورت جداگانه و با هر ترتیبی اجرا شوند، با این وجود به دلیل وابستگی متقابل شدید هر سه مرحله به یکدیگر، اجرای هر مرحله، تأثیر چشم‌گیری بر اجرای سایر مراحل دارد و درنتیجه نهایی بسیار اثرگذار است. همچنین می‌توان هر سه زیرکار را به‌صورت هم‌زمان نیز اجرا کرد.



(شکل-۴): نمونه‌ای از DFG زمان‌بندی شده شکل (۲)

(Figure-4): An example of a scheduled DFG for figure (2)



(شکل-۵): نمونه‌ای دیگر از DFG زمان‌بندی شده شکل (۲)

(Figure-5): Another example of a scheduled DFG for figure (2)

[1] Mobility

نمونه‌ای از پرکاربردترین مدارات دیجیتال، مبدل‌های دیجیتال هستند که به‌صورت گسترده در مدارات VLSI و در کاربردهای خاص به‌کار برده می‌شوند. تبدیل کسینوسی گسسته (DCT)[2]، تبدیل موجک گسسته (DWT)[3]، تبدیل فوریه سریع (FFT)[4] و تبدیل کسینوسی گسسته معکوس (IDCT)[5] نمونه‌هایی از تبدیلات گسسته و دیجیتال هستند که به‌وفور در کاربردهای پردازش سیگنال، تصاویر و ویدئوها استفاده می‌شوند برای مثال از تبدیل کسینوسی گسسته می‌توان برای پردازش، فشرده‌سازی و بهبود تصاویر استفاده کرد [43-45] و یا تبدیل فوریه سریع که در جداسازی طیف سیگنال‌ها در کاربردهای پردازش سیگنال و فیلترها استفاده فراوانی دارد و به‌وسیله آن می‌توان عمل تبدیل فوریه گسسته را با سرعتی به‌مراتب بالاتر انجام داد [46, 47]. تبدیل موجک گسسته نیز از تبدیلات پرکاربرد است که در ساخت ادوات مورد نیاز در کاربردهای ارتباطی و مخابراتی، پردازش صوت و تصویر، پردازش سیگنال‌های حیاتی و ... می‌توان از آن استفاده کرد [43, 48]. شکل (۶) دو نمونه از گراف جریان داده تبدیلات استفاده‌شده در این مقاله را نشان می‌دهد [49].

۳- بیان روش پیشنهادی

به‌دلیل اینکه فضای جستجو در HLS فضایی گسترده و گسسته است، به‌نظر می‌رسد که استفاده از روش‌های مبتنی‌بر الگوریتم‌های فراابتکاری و هوش‌جمعی راه‌گشا باشند. در این مقاله برای نخستین‌بار از الگوریتم قدرتمند MFO در کنار روش GA به‌عنوان الگوریتم رقیب جهت بهینه‌سازی مسیر داده استفاده‌شده و درنهایت مقایسه‌ای بین عملکرد دو الگوریتم از لحاظ کارایی صورت گرفته است. در ادامه پس از معرفی اجمالی روش MFO، به بیان ساختار شعله‌ها و پروانه‌ها و تعریف تابع برازندگی برای حل مسأله سنتز سطح بالای مبدل‌های دیجیتال پرداخته شده است.

۳-۱- الگوریتم MFO

الگوریتم MFO یکی از الگوریتم‌های فراابتکاری مبتنی‌بر جمعیت است که در [50] ارائه شده است. این الگوریتم نیز همانند سایر روش‌های فراابتکاری با الگوبرداری از طبیعت در جهت یافتن پاسخ‌های بهینه در حل مسائل گام

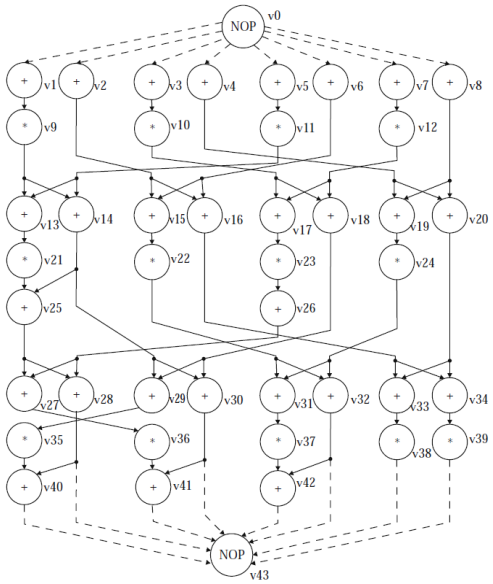
[2] Discrete Cosine Transform
[3] Discrete Wavelet Transform
[4] Fast Fourier Transform
[5] Inverse Discrete Cosine Transform

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]
[DOI: 10.52547/jsdp.18.3.3]

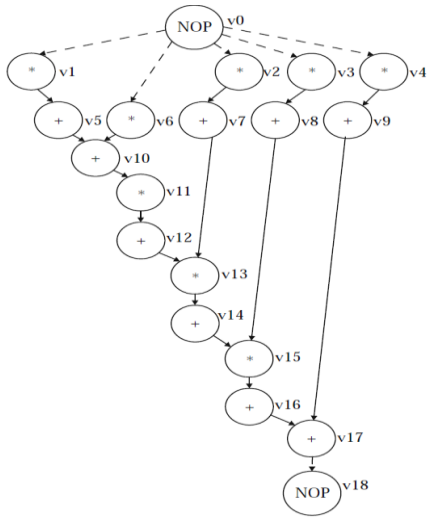
برمی‌دارد. ایده الگوریتم MFO برگرفته از پروانه‌ها و دستگاه مسیریابی غریزی آن‌ها در شب و از طریق نور ماه است که به مسیریابی متقاطع مشهور است. پروانه‌ها چنین مسیریابی را در مقابل نورهای مصنوعی نیز با اندکی تفاوت از خود نشان داده و به سمت منبع نور حرکت می‌کنند.

$$M = \begin{pmatrix} m_{11} & \cdots & m_{1d} \\ \vdots & \ddots & \vdots \\ m_{n1} & \cdots & m_{nd} \end{pmatrix} \quad (۱)$$

در این روش پروانه‌ها را می‌توان مانند کروموزم‌ها در الگوریتم ژنتیک یا ذرات در الگوریتم PSO [51] در نظر گرفت. متغیرها نیز همان موقعیت پروانه‌ها در فضا هستند. موقعیت پروانه‌ها را می‌توان به‌صورت رابطه (۱) تعریف کرد.



(الف)



(ب)

(شکل-۶): الف- گراف جریان داده تبدیل کسینوسی گسسته (DCT) ب- گراف جریان داده تبدیل موجک گسسته (DWT)

(Figure-6): a-Discrete cosine transform data flow graph (DCT) b- Discrete wavelet transform data flow graph (DWT)

که در آن n تعداد پروانه‌ها یا همان پاسخ‌هاست و d ابعاد مسأله است. مقدار برازندگی هر پروانه نیز در ماتریسی به نام OM به‌صورت رابطه (۲) ذخیره می‌شود.

$$OM = \begin{pmatrix} OM_1 \\ OM_2 \\ \vdots \\ OM_n \end{pmatrix} \quad (۲)$$

که در آن n تعداد پروانه‌ها و OM_i برازندگی i اُمین پروانه است. بطور مشابه دو ماتریس دیگر نیز برای شعله‌ها به‌صورت رابطه (۳) تعریف می‌شود:

$$M = \begin{pmatrix} f_{11} & \cdots & f_{1d} \\ \vdots & \ddots & \vdots \\ f_{n1} & \cdots & f_{nd} \end{pmatrix}$$

$$OF = \begin{pmatrix} OF_1 \\ OF_2 \\ \vdots \\ OF_n \end{pmatrix} \quad (۳)$$

در ادامه موقعیت جدید هر پروانه طبق رابطه (۴) به‌دست می‌آید:

$$S(M_i.F_j) = D_i \times e^{bt} \times \cos(2\pi t) + f_j \quad (۴)$$

که در آن $S(M_i.F_j)$ موقعیت جدید پروانه i اُم، b ثابتی است جهت تعیین شکل مارپیچی لگاریتمی، t عددی تصادفی در بازه [۱و۱-] و D_i بیان‌کننده فاصله بین i اُمین پروانه با j اُمین شعله است که به‌صورت رابطه (۵) به‌دست می‌آید:

$$D_i = |f_j - M_i| \quad (۵)$$

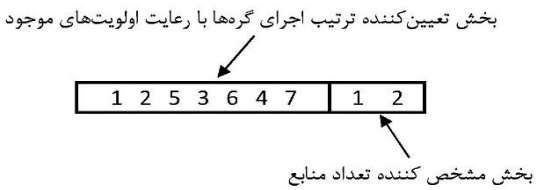
که در آن f_j بیان‌گر شعله j اُم و M_i پروانه i اُم است.

۲-۳- تعریف نمونه‌ها

در اینجا هم پروانه‌ها و هم شعله‌ها هر دو جواب شدنی مسأله هستند، با این تفاوت که شعله‌ها بهترین موقعیت پروانه‌ها تا این لحظه هستند و پروانه‌ها جواب‌هایی هستند که جهت رسیدن به نقطه بهینه در فضای جستجو در اطرف شعله‌ها حرکت می‌کنند. هر پروانه اطراف یک شعله حرکت کرده تا زمانی که یک موقعیت بهتر از آن را یافته و با موقعیت شعله جایگزین می‌کند. در این الگوریتم جهت فرار از نقاط بهینه محلی و بهبود قابلیت جستجوی سراسری، هر پروانه موقعیت خود را فقط با شعله متناظر با خود در ماتریس شعله‌های مرتب شده، به‌روز می‌کند. برای

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]
[DOI: 10.52547/jsdp.18.3.3]

مثال نخستین پروانه در ماتریس پروانه‌ها با بهترین شعله و آخرین پروانه با بدترین شعله در ماتریس مرتب‌شده شعله‌ها موقعیت خود را به‌روز می‌کنند. بدین ترتیب نقاط بیشتری در فضای طراحی جستجو شده و قابلیت جستجوی سراسری الگوریتم بهبود می‌یابد. در این مقاله موقعیت هر پروانه و یا شعله به‌صورت شکل (۷) است.



(شکل-۷): تعریف موقعیت پروانه‌ها و شعله‌ها
(Figure-7): Definition of the position of the moths and the flames

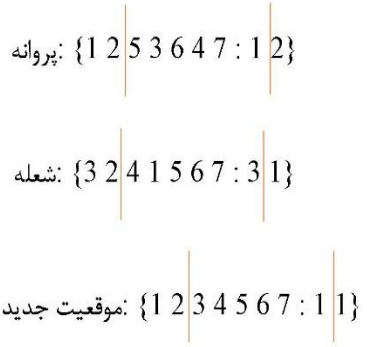
در این تعریف موقعیت هر پروانه به دو بخش و ناحیه تقسیم می‌شود. بخش نخست و سمت چپ بیان‌گر ترتیب و حق تقدم اجرای گره‌ها در DFG است و بخش دوم و سمت راست، نشان‌گر تعداد واحدهای عملیاتی از هر نوع است. ترتیب قرارگیری شماره گره‌ها در بخش حق تقدم، نشان می‌دهد که کدام گره زودتر اجرا شود. برای مثال در بالا ابتدا عمل‌گر ۱، سپس عمل‌گر ۲ و بعد از آن عمل‌گر پنجم اجرا و این روند به‌همین ترتیب تا اجرای همه گره‌ها انجام می‌شود. به این نکته باید توجه شود که در تمام پروانه‌ها و شعله‌ها باید اولویت اجرای گره‌ها رعایت شود؛ برای مثال در DFG یادشده در شکل (۲)، گره ۵ به‌حتم باید پس از اجرای گره‌های ۱ و ۲ قرار گیرد؛ زیرا برای اجرای آن نیاز به خروجی‌های عمل‌گرهای ۱ و ۲ است، ولی گره ۳ به گره ۱ و ۲ وابسته نبوده و می‌تواند پیش یا پس از آن‌ها قرار گیرد. به‌روزرسانی موقعیت پروانه‌ها با استفاده از موقعیت پروانه و موقعیت شعله متناظر با آن پروانه نیز، طبق روش گفته‌شده در ادامه، صورت گرفته و موقعیت جدید پروانه محاسبه می‌شود [31]. بروزرسانی موقعیت پروانه‌ها در بخش اولویت‌ها به‌گونه‌ای تعریف شده است که پس از اعمال آن، تمام اولویت‌های اجرای گره‌ها رعایت شده‌اند.

ابتدا در بخش اولویت‌های پروانه و شعله متناظر با آن، یک نقطه طبق رابطه (۶) در نظر گرفته می‌شود:

$$Q = T - Round\left(\left(T-1\right) - I \times \left(\frac{T-2}{Max_i}\right)\right) \qquad (۶)$$

که در آن Q نقطه مورد نظر، T تعداد کل عمل‌گرها در DFG، I تکرار حال حاضر و Max_i بیشینه تکرارها هستند.

تمام گره‌های قبل از این نقطه در پروانه، بدون تغییر در موقعیت جدید ظاهر می‌شود؛ سپس برای تولید مابقی موقعیت جدید، از اطلاعات و اولویت‌های موجود در شعله متناظر استفاده می‌شود. بدین صورت که از ابتدای موقعیت شعله موردنظر شروع به‌حرکت کرده و گره‌هایی را که در موقعیت جدید پروانه وجود ندارند، به‌ترتیب جایگزین می‌کنیم. جهت تولید موقعیت جدید در بخش تعداد واحدهای عملیاتی، پس از انتخاب یک نقطه به‌طورکامل تصادفی در این بخش، گره‌های بعد از این نقطه در پروانه و شعله با یکدیگر تعویض می‌شوند. در شکل (۸) نمونه‌ای از انجام این روش برای یک پروانه و شعله فرضی برای DFG یادشده در شکل (۲) نشان داده شده است:



(شکل-۸): مثالی از به‌روزرسانی موقعیت پروانه‌ها
(Figure-8): An example of updating the position of the moths

همچنین جهت اینکه قابلیت استخراج در روند اجرای الگوریتم بهبود یابد، تعداد شعله‌ها نیز در هر تکرار از طریق رابطه (۷) به‌روز می‌شوند:

$$Flame_{Number} = Round\left(N - I \times \left(\frac{N-1}{Max_i}\right)\right) \qquad (۷)$$

که در آن N بیشینه تعداد شعله‌ها یا همان جمعیت اولیه، I تکرار حال حاضر و Max_i بیشینه تکرارها هستند. بدین ترتیب در انتهای الگوریتم همه پروانه‌ها در اطراف بهترین شعله باقی‌مانده حرکت می‌کنند.

۳-۳- تابع برازندگی

جهت ارزیابی پارامترهای سرعت، سطح اشغال‌شده و توان در دو روش یادشده و مقایسه عملکرد آن‌ها، از تابع برازندگی پیشنهادی به‌صورت رابطه (۸) استفاده شده است:

$$Fitness = W_1 \times \frac{L_t}{L_{Max}} + W_2 \times \frac{A_t}{A_{Max}} + W_3 \times \frac{P_t}{P_{Max}} \qquad (۸)$$

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]
[DOI: 10.52547/jsdp.18.3.3]

که در آن *Fitness* همان تابع برازندگی، L_t طول زمان‌بندی نمونه مورد ارزیابی، L_{Max} بزرگترین طول زمان‌بندی موجود در جمعیت حال حاضر، A_t مجموع ترانزیستورهای موجود در عمل‌گرها و ثبات‌ها، A_{Max} بزرگترین سطح موجود در جمعیت حاضر، P_t توان‌مصرفی عمل‌گرها و P_{Max} بیشترین توان در جمعیت موجود است. W_1، W_2 و W_3 نیز ضرایب جهت وزن‌دهی و ارزش‌گذاری بر روی پارامترها هستند. این سه ضریب، برحسب این‌که اولویت بهینه‌سازی، کدام یک از پارامترهای زمان، سطح اشغال‌شده و یا توان است، به‌گونه‌ای انتخاب می‌شوند تا مجموع آن‌ها همواره برابر یک شود. تعداد عمل‌گرها به‌صورت مستقیم از بخش تعداد واحدهای عملیاتی در کروموزوم (در GA) و یا شعله‌ها و پروانه‌ها (در MFO) به‌دست آمده و تعداد ثبات‌ها نیز به‌وسیله روش LEA[1] محاسبه می‌شود [25]. جهت مشاهده بهتر روند الگوریتم پیشنهادی شبه‌کد روش یادشده در ادامه آمده است:

```
Generate random initial population with respect to
node priorities and resource constraints.
For i=1 to population number
    Calculate Schedule length
    Calculate register number using Left Edge
    algorithm
End
Calculate fitness using Eq.6
For i=1 to Max_iteration
    Update flame number using Eq.5
    Generate and update the sorted matrix of
    Moths and Flames.
    Update Q using Eq.4
    For j=1 to Moth number
        Update the position of the Moth using Q
        and Fig. 8 with respect to the corresponding
        flame
        Calculate Schedule length
        Calculate register number using Left Edge
        Algorithm
    End
    Calculate fitness using Eq.6
End
```

۴- نتایج شبیه‌سازی

در این بخش نتایج حاصل از اجرای روش قدرتمند پیشنهادی مبتنی‌بر الگوریتم بهینه‌شده MFO در کنار روش مبتنی بر GA آورده شده است. جهت ارزیابی و مقایسه عملکرد و کارایی دو روش یادشده، این دو الگوریتم بر روی داده‌های آزمون DWT، MPEG، FFT، DCT و IDCT اعمال شده است. جدول (۱) مشخصات گراف هر کدام از داده‌های آزمون را نشان می‌دهد. نتایج حاصله شامل به‌دست‌آوردن بهینه‌ترین زمان اجرا، سطح اشغال شده و توان‌مصرفی است. در اینجا تأخیر و یا سرعت، همان طول زمان‌بندی یا گام‌های زمانی مورد نیاز برای اجرای کل DFG است. سطح اشغال‌شده از طریق محاسبه تعداد ترانزیستورهای موجود در عمل‌گرها و ثبات‌ها به‌دست آمده و توان‌مصرفی نیز مجموع توان عمل‌گرها در سیستم است [52].

(جدول-۱): مشخصات داده‌های آزمون

(Table-1): Specification of the benchmarks

	کاربرد	تعداد گره‌ها
DWT	پردازش سیگنال‌های دیجیتال، سیگنال‌های حیاتی، صوت، تصویر، حذف نویز و ...	17
MPEG	پردازش و فشرده‌سازی ویدئو	28
FFT	پردازش سیگنال و فیلترها	36
DCT	پردازش سیگنال و فشرده‌سازی تصویر	42
IDCT	پردازش سیگنال و فشرده‌سازی تصویر	42

پیاده‌سازی‌های مقاله، در نرم‌افزار MATLAB (R2015b) و سیستمی با مشخصات پردازش‌گر Cori7 6700HQ و ۸ گیگابایت حافظه رم پیاده‌سازی و اجرا شده است. در هر دو روش تعداد جمعیت اولیه برابر با ۲۰ نمونه است، همچنین تعداد تکرارها در هر بار اجرای الگوریتم برابر با ۱۰۰ در نظر گرفته شده است. در الگوریتم ژنتیک، ضریب تقاطع برابر ۰/۹ و ضریب عملگر جهش برابر ۰/۲ منظور شده است. هر الگوریتم، ۱۰ بار اجرا شده و میانگین جواب‌ها به عنوان جواب نهایی بدست آمده برای هر پارامتر ذکر شده است.

با توجه به حضور سه تابع هدف (سرعت، سطح اشغال‌شده بر روی تراشه و توان مصرفی) نمایش جبهه پرتو تخمین زده شده بوسیله روش پیشنهادی برای هر سه تابع به‌صورت هم‌زمان، سه بعدی خواهد شد و نخواهد توانست معیار سنجش بصری کارآمد و مناسبی باشد، لذا با ثابت درنظرگرفتن پارامتر مربوط به توان (W_3=0.3) با این معنی که اهمیت این هدف هرگز از سی درصد نسبت به سایر اهداف کمتر در نظر گرفته نشده است، جبهه پرتو در دو بعد برای تقابل دو هدف دیگر (سرعت و سطح اشغال شده) نمایش داده شده است. در اینجا اهمیت ویژه به مصرف توان داده شده و طبیعی است که همین رفتار را می‌توان با ثابت نگه‌داشتن ضرایب مربوط به دو پارامتر دیگر نیز تکرار کرد. جدول (۲) پاسخ میانگین و انحراف معیار نهایی دو روش یادشده به داده‌های آزمون را نشان می‌دهد. ذکر این نکته حائز اهمیت است که به‌دلیل بزرگی

[1] Left Edge Algorithm

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]

[DOI: 10.52547/jsdp.18.3.3]

مقادیر سطح اشغال شده و توان مصرفی، جهت نمایش و مقایسه بهتر داده‌ها از انحراف معیار لگاریتم پاسخ‌ها استفاده شده است. در این جدول W_3 برابر ۰/۳ در نظر گرفته شده و W_1 و W_2 از ۰/۱ تا ۰/۶ تغییر کرده به‌نحوی که مجموع هر سه ضریب همواره برابر یک است. همان‌طور که مشاهده می‌شود، روش پیشنهادی مبتنی‌بر الگوریتم MFO با بهبود در سرعت، سطح اشغال شده و توان، عملکرد بهتری نسبت‌به روش برمبنای الگوریتم GA دارد. می‌توان دید که با تغییر ضرایب مرتبط با سرعت و سطح یعنی W_1 و W_2 تغییرات قابل درکی در پاسخ‌های به‌دست‌آمده به‌وجود آمده است. با کاهش ضریب سرعت (W_1) و افزایش ضریب سطح اشغالی (W_2)، تأخیر افزایش یافته و سطح و توان‌مصرفی کاهش می‌یابند. برای روش پیشنهادی در مبدل DWT با کاهش W_1 و افزایش W_2، تأخیر از ۱۰/۲ به ۱۲/۶ افزایش یافته است سطح اشغال‌شده از تراشه نیز از ۵۸۵۲/۸ به ۳۲۴۳/۲ کاهش یافته و بهبود داشته است. همچنین توان از ۶۲۷۴/۳۲ به ۳۱۳۷/۱۶ کاهش پیدا کرده است. این پارامترها برای روش مبتنی‌بر الگوریتم ژنتیک به‌ترتیب افزایش تأخیر از ۱۰/۴ به ۱۲/۸، کاهش سطح اشغالی از ۵۸۸۱/۶ به ۳۲۵۷/۶ و کاهش توان از ۶۲۷۴/۳۲ به ۳۱۳۷/۱۶ را نشان می‌دهد.

برای سایر مبدل‌ها نیز مشاهده می‌شود که روش مبتنی‌بر MFO عملکرد بهتری را از خود نشان می‌دهد. متوسط بهبود هر پارامتر برای هر مبدل در جدول (۳) آورده شده است.

در جدول (۳) مشاهده می‌شود که به‌طور مثال در مبدل DWT تأخیر متوسط از ۱۲/۰۲ به ۱۱/۷۸ کاهش یافته که بهبودی ۱/۹۴ درصدی را نشان می‌دهد. سطح اشغالی برای پیاده‌سازی این مبدل نیز از ۳۸۵۷/۳۳ به ۳۷۹۲ و توان از ۳۸۵۴/۷ به ۳۸۰۲/۴۱ کاهش یافته است که به‌ترتیب بهبود ۱/۶۹ و ۱/۳۶ درصدی را نشان می‌دهد. در MPEG متوسط تأخیر از ۱۱/۵۲ به ۱۱/۳۳، سطح از ۶۳۷۵/۴۷ به ۶۲۰۸ و توان از ۵۹۷۸/۴ به ۵۹۱۹/۵۶ کاهش یافته است که به‌ترتیب ۱/۵۹، ۲/۶۳ و ۰/۹۸ درصد بهبود در عملکرد را نشان می‌دهد. شکل (۹) درصد بهبود عملکرد روش پیشنهادی نسبت‌به روش مبتنی‌بر الگوریتم GA را برای تمام داده‌های آزمون نشان می‌دهد.

(جدول-۲): مقایسه دو روش مبتنی‌بر GA و MFO در سنتز داده‌های آزمون

(Table-2): Comparison of two methods based on GA and MFO for the synthesis of the benchmarks

W_3=0.3			روش مبتنی‌بر GA [31]			روش پیشنهادی مبتنی‌بر MFO		
			تأخیر (تعداد گام زمانی)	سطح (تعداد ترانزیستورها)	توان (مجموع توان مصرفی عملگرها)	تاخیر (تعداد گام زمانی)	سطح (تعداد ترانزیستورها)	توان (مجموع توان مصرفی عملگرها)
DWT	W_1=0.6 W_2=0.1	*ave*	10.4	5881.6	6274.32	**10.2**	5852.8	6274.32
		std	0.5	0.005	0	0.4	0.004	0
	W_1=0.5 W_2=0.2	*ave*	11.3	3747.2	3789.8	11.2	3552	3551.74
		std	0.8	0.113	0.13	0.6	0.085	0.098
	W_1=0.4 W_2=0.3	*ave*	12.2	3564.8	3501.3	11.8	3491.2	3450.87
		std	0.9	0.082	0.094	0.6	0.085	0.095
	W_1=0.3 W_2=0.4	*ave*	12.7	3392	3288.45	12.4	3340.8	3263.24
		std	0.9	0.005	0.017	0.8	0.013	0.018
	W_1=0.2 W_2=0.5	*ave*	12.7	3300.8	**3137.16**	12.5	3272	**3137.16**
		std	0.5	0.009	**0**	1.5	0.01	0
	W_1=0.1 W_2=0.6	*ave*	12.8	3257.6	**3137.16**	12.6	**3243.2**	**3137.16**
		std	1	0.01	**0**	0.9	0.009	0
MPEG	W_1=0.6 W_2=0.1	*ave*	6.7	9161.6	9223.84	**6.6**	9169.6	9249.06
		std	0.7	0.05	0.052	0.6	0.05	0.052
	W_1=0.5 W_2=0.2	*ave*	7.1	8942.4	8910.12	7	8804.8	8884.91
		std	1.1	0.053	0.069	0.6	0.063	0.068
	W_1=0.4 W_2=0.3	*ave*	8.9	6979.2	6602.12	8.7	6635.2	6425.61
		std	0.3	0.008	0.008	0.4	0.008	0.009
	W_1=0.3 W_2=0.4	*ave*	13.8	4968	4355.68	13.5	4713.6	4280.03
		std	3.4	0.091	0.126	3.3	0.104	0.132
	W_1=0.2 W_2=0.5	*ave*	16.1	4203.2	3439.75	15.9	4072	3389.32
		std	0.9	0.016	0.013	0.7	0.011	0
	W_1=0.1 W_2=0.6	*ave*	16.5	3998.4	3338.88	16.3	**3852.8**	**3288.45**
		std	1.2	0.023	0.021	1.5	0.01	0.017
	W_1=0.6 W_2=0.1	*ave*	7.3	7136	6778.64	**7.1**	7136	6778.64
		std	0.5	0	0	0.3	0	0
	W_1=0.5 W_2=0.2	*ave*	9.7	5976	5448.12	9.5	5976	5448.12
		std	2.9	0.118	0.152	3	0.117	0.151
	W_1=0.4	*ave*	12.2	4955.2	4193.26	12	4720	4041.96

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]

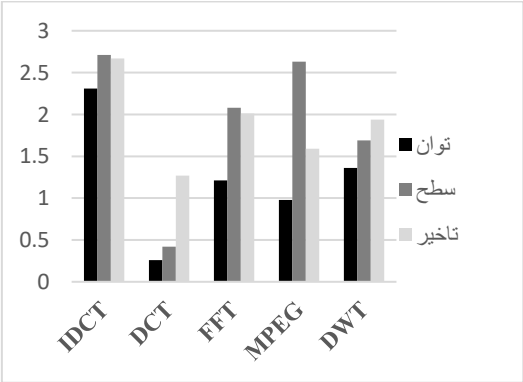
[DOI: 10.52547/jsdp.18.3.3]

FFT	W_2=0.3	*std*	2.2	0.093	0.121	2.2	0.096	0.123
	W_1=0.3 W_2=0.4	*ave*	13.3	4436.8	3540.61	13	4328	3515.4
		std	0.5	0.029	0.026	0	0.03	0.025
	W_1=0.2 W_2=0.5	*ave*	13.5	4400	3515.4	13.3	4297.6	3464.96
		std	0.5	0.027	0.016	0.4	0.024	0.015
	W_1=0.1 W_2=0.6	*ave*	13.7	4348.8	3490.18	13.4	**4144**	**3389.32**
		std	0.5	0.026	0.016	0.5	0	0
DCT	W_1=0.6 W_2=0.1	*ave*	9.7	7796.8	6728.2	**9.5**	7782.4	6728.2
		std	0.8	0.005	0.007	0.8	0.005	0.006
	W_1=0.5 W_2=0.2	*ave*	10.7	6968	5837.5	10.7	6945.6	5812.27
		std	2.3	0.089	0.12	2.3	0.089	0.123
	W_1=0.4 W_2=0.3	*ave*	12.4	6368	5134.41	12.3	6353.6	5109.19
		std	2.6	0.1	0.15	2.7	0.1	0.15
	W_1=0.3 W_2=0.4	*ave*	14.5	5382.4	3879.54	14.3	5332.8	3854.33
		std	2.1	0.06	0.092	2.1	0.058	0.092
	W_1=0.2 W_2=0.5	*ave*	15.3	5062.4	3616.26	15	5028.8	3591.04
		std	0.9	0.009	0.022	0.6	0.01	0.023
	W_1=0.1 W_2=0.6	*ave*	16	4961.6	**3515.4**	15.8	**4940.8**	3540.61
		std	0.8	0.007	0.016	0.7	0.009	0.16
IDCT	W_1=0.6 W_2=0.1	*ave*	7.4	8398.4	7506.93	**7.3**	8137.6	7243.65
		std	0.5	0.05	0.067	0.4	0.037	0.049
	W_1=0.5 W_2=0.2	*ave*	7.9	8193.6	7193.22	7.7	7881.6	6904.72
		std	0.3	0.038	0.05	0.5	0.009	0.008
	W_1=0.4 W_2=0.3	*ave*	10.9	6707.2	5361.35	10.5	6612.8	5336.14
		std	3.1	0.104	0.149	2.6	0.099	0.141
	W_1=0.3 W_2=0.4	*ave*	12.8	5644.8	4156.92	12.6	5556.8	4106.49
		std	1.8	0.058	0.088	1.6	0.06	0.087
	W_1=0.2 W_2=0.5	*ave*	13.6	5270.4	3717.12	13.2	5190.4	3691.91
		std	1.3	0.012	0.025	0.6	0.01	0.018
	W_1=0.1 W_2=0.6	*ave*	14.7	5150.4	3565.83	14.2	**4910.4**	**3490.18**
		std	1.1	0.011	0.029	1	0.004	0.016

(جدول-۳): میانگین پاسخ‌های دو روش مبتنی‌بر GA و MFO در سنتز داده‌های آزمون

(Table-3): The average results of the two methods based on the GA and the MFO in the synthesis of the benchmarks

	روش مبتنی‌بر GA			روش پیشنهادی مبتنی‌بر MFO		
	متوسط تأخیر	متوسط سطح	متوسط توان	متوسط تأخیر	متوسط سطح	متوسط توان
DWT	12.02	3857.33	3854.7	11.78	3792	3802.41
MPEG	11.52	6375.47	5978.4	11.33	6208	5919.56
FFT	11.62	5208.8	4494.37	11.38	5100.27	4439.73
DCT	13.1	6089.87	4785.22	12.93	6064	4772.6
IDCT	11.22	6560.8	5250.23	10.92	6381.6	5128.85



(شکل-۹): بهبود روش پیشنهادی نسبت‌به روش مبتنی‌بر الگوریتم GA (٪)

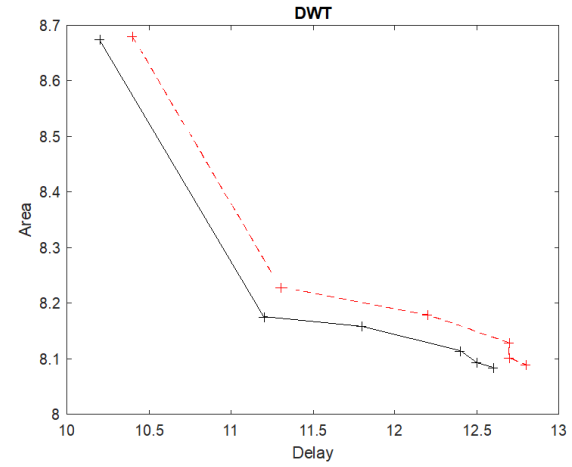
(Figure-9): Improvement of the proposed method compared to the GA-based method

با استفاده از شکل (۹) می‌توان مشاهده کرد که بهترین بهبود سرعت در مبدل IDCT با بهبود ۲/۶۷ درصد کاهش طول زمان‌بندی به‌دست آمده است. پس از آن مبدل FFT قرار دارد که با کاهش ۲/۰۱ درصدی در طول زمان‌بندی و تأخیر، عملکرد بهتر روش پیشنهادی را نشان می‌دهد. برای سطح اشغال‌شده از تراشه که همان تعداد ترانزیستورهای مورد نیاز برای پیاده‌سازی عمل‌گرها و ثبات‌هاست، بهترین بهبود را در مبدل IDCT با کاهش ۲/۷۱ درصدی از سطح اشغالی شاهد هستیم، پس از آن هم MPEG با ۲/۶۳٪ بهبود از لحاظ سطح اشغال شده بر روی تراشه قرار دارد. در مورد توان نیز بهبود ۲/۳۱ درصدی در مبدل IDCT مشاهده شده است. پس از آن نیز مبدل DWT با بهبود ۱/۳۶ درصدی در این زمینه بیان‌گر عملکرد بهتر روش پیشنهادی است. شکل (۱۰) نمود کاملی از نمایش تغییرات پاسخ‌ها بسته‌به ضرایب انتخابی را برای تمام داده‌های آزمون نشان می‌دهد. در این

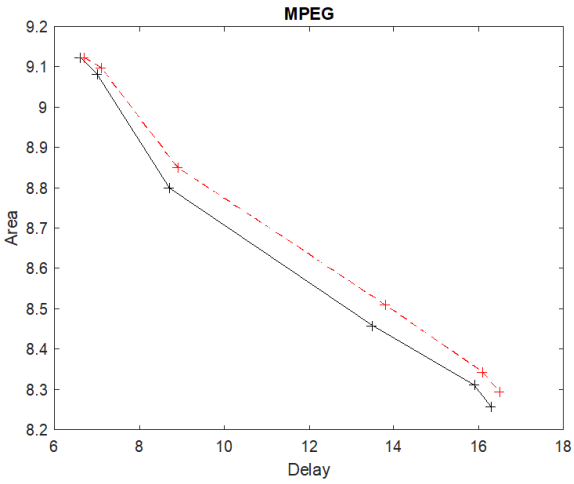
[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]
[DOI: 10.52547/jsdp.18.3.3]

شکل خط‌چین قرمز رنگ پاسخ‌های روش ژنتیک و خطوط سیاه پیوسته نیز پاسخ‌های مترادف، با استفاده از روش پیشنهادی را نشان می‌دهد. در این شکل W_3 برابر ۰/۳ در نظر گرفته شده و W_1 و W_2 مطابق جدول (۲) از ۰/۱ تا ۰/۶ تغییر کرده به‌نحوی که مجموع هر سه ضریب همواره برابر ۱ در نظر گرفته شده است. همچنین در این شکل به‌جهت نمایش و نمود بهتر، در محور عمودی از لگاریتم سطح اشغال شده استفاده شده است تا بتوان مقایسه بهتری از دو روش یادشده انجام داد.

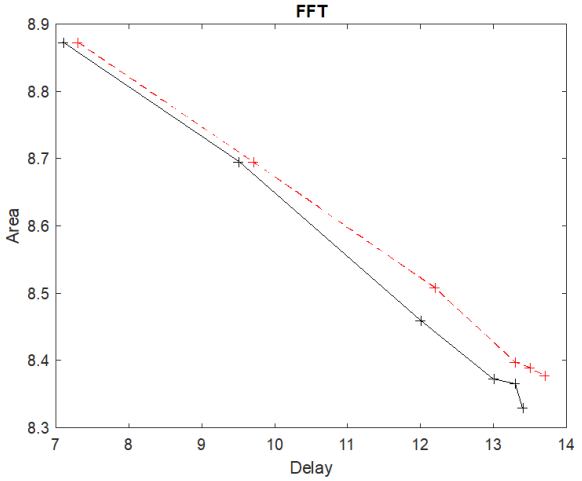
با توجه به شکل (۱۰) مشاهده می‌شود که در تمام داده‌های آزمون، پایین‌تربودن نمودار مربوط به روش پیشنهادی و درواقع کمتربودن سطح زیر این منحنی نسبت‌به منحنی قرمز رنگ مربوط به روش مبتنی‌بر الگوریتم ژنتیک، نشان‌دهندهٔ عملکرد بهتر روش پیشنهادی در یافتن پاسخ‌های بهینه است. تفاوت در دو منحنی یادشده در نمودار مربوط به مبدل IDCT کرد بیشتری دارد به‌نحوی که با داده‌های موجود در شکل (۹) که بهترین عملکرد روش پیشنهادی را در این مبدل (IDCT) بیان می‌داشت، هم‌ارز است.



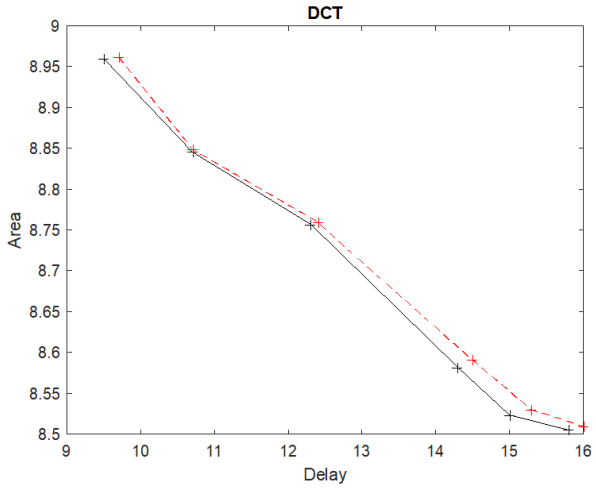
(الف)



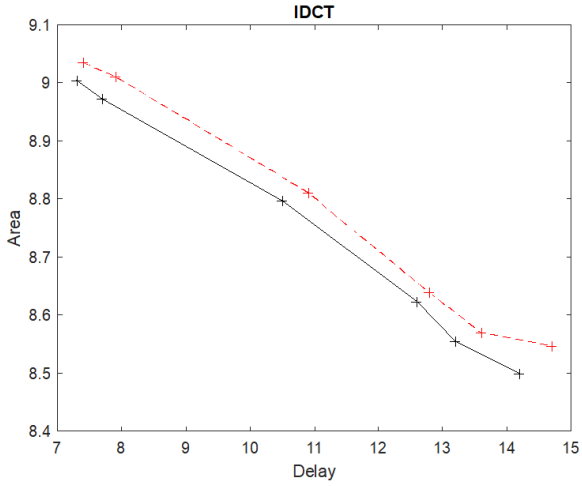
(ب)



(ج)



(د)



(ه)

(شکل-۱۰): پاسخ‌های به‌دست آمده با تغییر ضرایب W_1 و W_2 برای الف- تبدیل موجک گسسته (DWT) ب- MPEG ج- تبدیل فوریه سریع (FFT) د- تبدیل کسینوسی گسسته (DCT) ه- تبدیل کسینوسی گسسته معکوس (IDCT)

(Figure-10): Obtained results in terms of variation of W_1 and W_2 coefficients: a- Discrete Wavelet Transform (DWT) b-MPEG c-Fast Fourier Transform (FFT) d-Discrete Cosine Transform (DCT) e-Inverse Discrete Cosine Transform (IDCT)

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]

[DOI: 10.52547/jsdp.18.3.3]

جهت ارزیابی عملکرد روش پیشنهادی، آزمون تی بر روی داده‌های موجود اعمال شده و نتایج به‌دست‌آمده در جدول (۴) گزارش شده است.

در جدول یادشده نتایج حاصله برای دو حالت بهترین تأخیر (W_1=۰/۶، W_2=۰/۱ و W_3=۰/۳) و بهترین سطح اشغال شده (W_1=۰/۱، W_2=۰/۶ و W_3=۰/۳) هر مبدل نشان داده شده است. مطابق داده‌های موجود در این جدول و مقایسه آن با مقادیر استاندارد t با درجه آزادی برابر با ۹، می‌توان مشاهده کرد که در یافتن بهترین تأخیر در مبدل FFT با درصد اطمینان بیش از ۹۵ درصد، اختلاف معنی‌داری بین روش جدید پیشنهادی و روش مبتنی‌بر الگوریتم ژنتیک حاصل شده است. این مقدار در مبدل‌های DWT، MPEG، DCT و IDCT به ترتیب بیش از ۹۰، ۶۰، ۷۵ و ۷۵ درصد است. در مورد بهینه‌ترین سطح اشغال شده نیز در مبدل‌های MPEG، FFT و IDCT با درصد اطمینان بیش از ۹۹ درصد اختلاف معنی‌داری بین روش پیشنهادی و روش مبتنی‌بر GA به‌دست آمده است. در DWT و DCT با درصد اطمینان بیشتر از ۷۵ درصد می‌توان این بهبود را مشاهده کرد. درنهایت با توجه به نتایج به‌دست‌آمده، به‌طور میانگین با درصد اطمینان بیش از ۸۵ درصد، روش پیشنهادی عملکرد مناسب‌تری نسبت به روش مبتنی بر الگوریتم ژنتیک دارد.

(جدول-۴): نتایج حاصل از آزمون تی

(Table-4): Results of the t-test

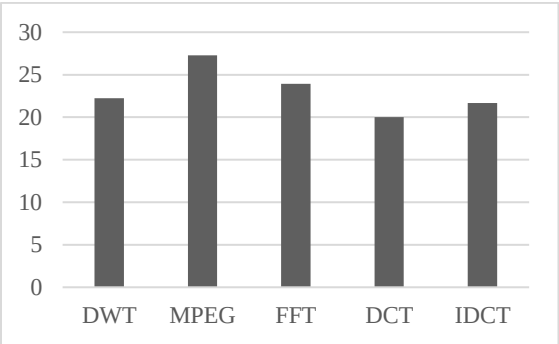
		t
DWT	Delay	-1.58
	Area	-0.75
MPEG	Delay	-0.52
	Area	-5.34
FFT	Delay	-2.1
	Area	-inf
DCT	Delay	-0.79
	Area	-0.73
IDCT	Delay	-0.79
	Area	-14.46

علاوه‌بر بهبود در جواب‌های به‌دست آمده، یکی از ویژگی‌های روش پیشنهادی سرعت بالای اجرای آن نسبت‌به روش مبتنی‌بر GA در رسیدن به پاسخ است. جدول (۵) متوسط سرعت اجرای دو الگوریتم در رسیدن به جواب را بر حسب ثانیه نشان می‌دهد. به‌وضوح مشخص است که سرعت اجرای الگوریتم مبتنی‌بر MFO به مراتب بیشتر از روش GA است.

جدول (۵): متوسط سرعت اجرای الگوریتم مبتنی‌بر MFO و روش مبتنی‌بر GA

(table-5): The average runtime of the MFO-based algorithm and the GA-based algorithm

	روش مبتنی‌بر GA [31]	روش مبتنی‌بر MFO
DWT	2.7	2.1
MPEG	5.5	4
FFT	7.1	5.4
DCT	9	7.2
IDCT	8.3	6.5



(شکل-۱۱): درصد بهبود سرعت اجرای الگوریتم

(Figure-11): Percentage of improvement in the algorithm runtime

جهت مقایسه بهتر در اجرای داده‌های آزمون، در شکل (۱۱) بهبود سرعت اجرای الگوریتم بر حسب درصد برای این داده‌ها نشان داده شده است. همان‌گونه که مشخص است، بهترین بهبود در این زمینه به MPEG اختصاص دارد که افزایش ۲۷/۲۷ درصدی سرعت اجرای الگوریتم را نسبت‌به روش مبتنی‌بر الگوریتم ژنتیک نشان می‌دهد، پس از آن نیز اجرای سنتز FFT با بهبود ۲۳/۹۴ درصدی در افزایش سرعت قرار دارد. در حالت کلی متوسط بهبود بالای بیست درصد در سرعت اجرای روش پیشنهادی نسبت‌به روش برمبنای الگوریتم GA، متضمن سرعت بالای این روش در رسیدن به جواب است.

۵- نتیجه‌گیری

همان‌طور که گفته شد، حل مسائل HLS به‌دلیل ماهیت آن‌ها ازجمله گسترده و گسسته‌بودن فضای جستجو و وجود اولویت‌های اجرا در حل مسأله، جزو مسائل سخت در مهندسی به‌شمار می‌روند. همچنین وجود جواب‌های متعددی که به یک پاسخ یکسان منتج می‌شوند نیز کار را پیچیده‌تر می‌کند. با این وجود استفاده از روش‌های ابتکاری که قبلا عملکرد خود را در حل چنین مسائلی نشان داده‌اند ممکن است باعث بهبود در یافتن جواب‌های بهینه شود. در این مقاله روشی مبتنی‌بر الگوریتم فراابتکاری MFO ارائه شد که پس از اعمال آن به

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]
[DOI: 10.52547/jsdp.18.3.3]

داده‌های آزمون استاندارد موجود، مشاهده شد که این روش در مقایسه با روش مبتنی‌بر الگوریتم تکاملی ژنتیک، با وجود شرایط اولیه یکسان (تعداد جمعیت اولیه و تعداد تکرارهای یکسان) از توانایی بالاتری در یافتن پاسخ بهینه سود می‌برد. همچنین علاوه بر یافتن پاسخ بهینه، یکی از ویژگی‌های روش پیشنهادی سرعت عملکرد بالای آن در رسیدن به جواب است. به‌گونه‌ای که به‌طور میانگین شاهد بهبود بیش از بیست درصدی در سرعت اجرای الگوریتم نسبت به روش مبتنی‌بر GA بوده‌ایم. به‌خصوص برای مسائلی با ابعاد بسیار بزرگ که به‌دلیل ابعاد بزرگ مسأله و پیچیدگی بیشتر پاسخ‌ها، سرعت اجرا از اولویت بالایی برخوردار است، روش پیشنهادی در یافتن سریع پاسخ بهینه برای چنین مسائلی، بسیار کاربردی خواهد بود.

6- References ۶- مراجع

[1] M. C. Bhuvaneswari, Application of Evolutionary Algorithms for Multi-Objective Optimization in VLSI and Embedded Systems. Springer, India 2015.

[2] S. Das, R. Maity, N. P. Maity, "VLSI-Based Pipeline Architecture for Reversible Image Watermarking by Difference Expansion with High-Level Synthesis Approach," *Circuits, Systems, and Signal processing*, vol. 37, no. 4, pp. 1575-1593, April 2018.

[3] D. Thomas, E. Lagnese, R.Walker, J. Nestor, J. Rajan, and R. Blackburn, Algorithmic and Register-Transfer Level Synthesis: The System Architect's Workbench. Kluwer, 1990.

[4] D. S. Harish Ram, M. C. Bhuvaneswari, S. S. Prabhu, "A Novel Framework for Applying Multiobjective GA and PSO Based Approaches for Simultaneous Area, Delay, and Power Optimization in High Level Synthesis of Datapaths," *VLSI Design*, vol. 2012, 2012.

[5] X. Tang, T. Jiang, A. Jones, P. Banerjee, "Behavioral synthesis of data-dominated circuits for minimal energy implementation," *in Proc. of 18th the International Conference on VLSI Design*,Jan. 2005, pp.3-7.

[6] N. Chabini, W. Wolf, "Unification of scheduling, binding, and retiming to reduce power consumption under timings and resources constraints," *IEEE Transactions on VLSI Systems*, vol. 13, no. 10, pp. 1113–1126, 2005.

[7] S. P. Mohanty, N. Ranganathan, S. K. Chappidi, "ILP models for simultaneous energy and transient power minimization during behavioral synthesis," *ACM Transaction on Design Automation of Electronic Systems*, vol. 11, no. 1, pp. 186–212, 2006.

[8] W. T. Shiue, "Peak power minimization using novel scheduling algorithm based on an ILP model," in Proc. of the 10th NASA Symposium on VLSI Design, 2002.

[9] A. Kumar, M. Bayoumi, "Multiple voltage-based scheduling methodology for low power in the high level synthesis," *in Proc. of the International Symposium on Circuits and Systems (ISCAS)*, 1999, pp. 371–379.

[10] A. K. Murugavel and N. Ranganathan, "A game theoretic approach for power optimization during behavioral synthesis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 6, pp. 1031–1043, 2003.

[11] R. K. Brayton, R. Camposano, G. De Micheli, R. Otten, J. van Eijndhoven, "The Yorktown silicon compiler system," *in Silicon Compilation, D. D. Gajski, Ed. Reading, MA: Addison-Wesley*, pp. 204–310, 1988.

[12] O. V. Nepomnyashchiy, I. V. Ryjenko, V. V. Shaydurov, N. Y. Sirotinina A. I. Postnikov, "The VLSI High-Level Synthesis for Building Onboard Spacecraft Control Systems," *in Anisimov K. et al. (eds) Proceedings of the Scientific-Practical Conference "Research and Development 2016*", Springer, Cham, 2018, pp. 229-238.

[13] R. Gopalan, C. Gopalakrishnan, S. Katkoori, "Leakage power driven behavioral synthesis of pipelined datapaths," *in IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 167–172, 11-12May 2005.

[14] S. P. Mohanty, R. Velagapudi, E. Kougianos, "Physical-aware simulated annealing optimization of gate leakage in nanoscale datapath circuits," *in Proc. of the Conference on Design, Automation and Test in Europe*, 6-10 March 2006, pp. 1191–1196.

[15] F. Su, K. Chakrabarty, "Unified high-level synthesis and module placement for defecttolerant microfluidic biochips," *in Proc. of the 42nd Annual Conference on Design automation*, 13-17 June 2005, pp. 825–830.

[16] S. Devadas, A. R. Newton, "Algorithms for hardware allocation in data path synthesis," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 8, no. 7, pp. 768-781, 1989.

[17] J. A. Nestor, G. Krishnamoorthy, "SALSA: A new approach to scheduling with timing constraints," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 12, pp. 1107–1122, 1993.

[18] G. Krishnamoorthy, J. A. Nestor, "Data path allocation using extended binding model," *in Proc. 29nd ACM/IEEE Design Automation Conf.*, 8-12 June 1992, pp. 279–284.

[19] T. A. Ly, J. T. Mowchenko, "Applying simulated evolution to high level synthesis," *IEEE Trans. on Computer-Aided Design of*

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]

[DOI: 10.52547/jsdp.18.3.3]

Algorithm based design space exploration," *in Proc. of the 12th International Symposium on Quality Electronic Design*, 14-16 March 2011, pp. 1-9.

[33] C. Pilato, D. Loiacono, A. Tumeo, F. Ferrandi, P. L. Lanzi, D. Sciuto, "Speeding-up expensive evaluations in high-level synthesis using solution modeling and fitness inheritance," *in Computational Intelligence in Expensive Optimization Problems*, vol. 2, 2010, pp. 701–723.

[34] R. F. Abdel-kader, "Particle Swarm Optimization for Constrained Instruction Scheduling," *VLSI design*, vol. 2008, no. 4, January 2008.

[35] S. A. Hashemi, B. Nowrouzian, "A novel particle swarm optimization for high-level synthesis of digital filters," *In Proc. of the 25th IEEE International Symposium on Circuits and Systems*, 20-23 May 2012, pp. 580–583.

[36] A. Sengupta, S. Bhadauria, S. P. Mohanty, "TL-HLS: Methodology for Low Cost Hardware Trojan Security Aware Scheduling with Optimal Loop Unrolling Factor during High Level Synthesis," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 36, no. 4, pp. 655-668, April 2017.

[37] S. Bhadauria, A. Sengupta, "Adaptive bacterial foraging driven datapath optimization: Exploring power-performance tradeoff in high level synthesis," *Applied Mathematics and Computation*, vol. 269, pp. 265-278, Oct. 2015.

[38] S. Rajmohan, N. Ramasubramanian, "A Memetic Algorithm based Design Space Exploration for Datapath Resource Allocation during High Level Synthesis," *Journal of Circuits, Systems and Computers*, 2019.

[39] S. Rajmohan, N. Ramasubramanian, "Group influence based improved firefly algorithm for Design Space Exploration of Datapath resource allocation," *Applied Intelligence*, vol. 49, no. 6, pp. 2084-2100, June 2019.

[40] C. Pilato, S. Garg, K. Wu, R. Karri, F. Regazzoni, "Securing Hardware Accelerators: A New Challenge for High-Level Synthesis," *IEEE Embedded Systems Letters*, vol. 10, no. 3, pp. 77-80, Sept. 2018.

[41] R. Nane et al., "A Survey and Evaluation of FPGA High-Level Synthesis Tools," *IEEE Transactions on Computer-Aided Designe of Integrated Circuits and Systems*, vol. 35, no. 10, pp. 1591-1604, Oct. 2016.

[42] N. S. Kim, J. Xiong, W. W. Hwu, "Heterogeneous Computing Meets Near-Memory Acceleration and High-Level Synthesis in the Post-Moore Era," *IEEE Micro*, vol. 37, no. 4, pp. 10-18, 2017.

Integrated Circuits and Systems, vol. 12, no. 3, pp. 389-409, 1993.

[20] S. Lucia, D. Navarro, O. Lucia, P. Zometa, R. Findeisen, "Optimized FPGA Implementation of Model Predictive Control for Embedded Systems Using High-Level Synthesis Tool," *IEEE Trans. on Industrial Informatics*, vol. 14, no. 1, pp. 137-145, Jan. 2018.

[21] G. De Micheli, Synthesis and Optimization of Digital Circuits. McGraw-Hill, New York 1994.

[22] R. Camposano, "Path-based scheduling for synthesis," *IEEE Trans. Comput. -Aided Des.*, vol. 10, pp. 85–93, 1991.

[23] P. G. Paulin, J. P. Knight, "Force-directed scheduling for the behavioral synthesis of ASICs," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 8, no. 6, pp. 661–679, 1989.

[24] S. Gupta, S. Katkoori, "Force-directed scheduling for dynamic power optimization," *in Proc. of the IEEE Computer Society Annual Symposium on VLSI*, 25-26 April 2002, pp. 68–73.

[25] S. H. Gerez, Algorithms for VLSI Design Automation. Wiley, 2004.

[26] S. Katkoori, R. Vemuri, "Scheduling for low power under resource and latency constraints," *in Proc. of the IEEE International Symposium on Circuits and Systems*, 28-31 May 2000, pp. 53–56.

[27] A. C. Parker, J. T. Pizarro, M. Mlinar, "Maha: A program for datapath synthesis," *in Proc. 23rd ACM/IEEE Design Automation Conf.*, 29 June- 2 July 1986, pp. 461–466.

[28] M. McFarland, T. J. Kowalski, "Incorporating bottom-up design into high-level synthesis," *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, vol. 9, no. 9, pp. 938–950, Sep. 1990.

[29] M. A. Elgamel, M. Bayoumi, "On low-power high-level synthesis using genetic algorithms," *in Proc. of the 9th International Conference on Electronics, Circuits and Systems*, 15-18 Sept. 2002, pp. 725–728.

[30] G. W. Grewal, T. C. Wilson, "An enhanced genetic algorithm for solving the high-level synthesis problems of scheduling, allocation, and binding," *International Journal of Computational Intelligence and Applications*, vol. 1, pp. 91-110, 2001.

[31] V. Krishnan, S. Katkoori, "A genetic algorithm for the design space exploration of datapaths during high-level synthesis," *IEEE Trans. Evol. Comput*, vol. 10, no. 3, pp. 213–229, 2006.

[32] A. Sengupta, R. Sedaghat, "Integrated scheduling, allocation and binding in high level synthesis using multi structure Genetic

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]

[DOI: 10.52547/jsdp.18.3.3]

[43] D. R. R. Freias, A. V. M. Inocencio, L. T. Lins, G. J. Alves, M. A. Bendetti, "A Parallel Implementation of the Discrete Wavelet Transform Applied to Real-Time EEG Signal Filtering," *in XXVI Brazilian Congress on Biomedical Engineering*, May 2019, pp. 17-23.

[44] C. Y. Pang, R. G. Zhou, B. Q. Hu, W. Hu, A. El-Rafei, "Signal and image compression using quantum discrete cosine transform," *Information Science*, vol. 473, pp. 121-141, January 2019.

[45] V. Mahale, M. M. H. Ali, P. L. Yannawar, A. Gaikwad, "Analysis of Image Inconsistency Based on Discrete Cosine Transform (DCT)," *in Proc. of Information and Communictuin Technology for Intelligent Systems*, Dec. 2018, vol. 1, pp. 563-571.

[46] A. N. Serov, A. A. Shatokhin, G. V. Antipov, "Sample Rate Converter As a Means of Reducing Measurement Error of the Voltage Spectrum by Application of FFT," in Proc. of 29th International Conference Radioelektronika (RADIOELEKTRONIKA), June. 2019.

[۴۷] سعید قاضی مغربی، فربیان خردادپور دیلمانی، "استفاده از تبدیل موجک در بهبود عملکرد OFDM به جای روش مرسوم مبتنی بر FFT"، پردازش علائم و داده‌ها، دوره ۱۶، شماره ۲، ص. ۱۳۶-۱۲۱، سال ۱۳۹۸.

[47] S. G. Maghrebi, F. K. Deylamani, "Using WPT as a New Method Instead of FFT for Improving the Performance of OFDM Modulation," *Signal and Data Processing*, vol. 16, no. 2, pp. 121-136, 2019.

[۴۸] اسماعیلی، محمدرضا، ظهیری، سید حمید، "تشخیص صرع در سیگنال EEG با استفاده از الگوریتم ابتکاری صفحات شیبدار (IPO)"، پردازش علائم و داده‌ها، دوره ۱۳، شماره ۴، ص. ۲۹-۴۲، سال ۱۳۹۵.

[48] M. R. Esmaeili, SH. Zahiri, "Epileptic seizure detection using Inclined Planes system Optimization algorithm (IPO)," *Signal and Data Processing*, vol. 13, no. 4, pp. 29-42, 2017.

[49] S. P. Mohanty, N. Ranganathan, E. Kougianos, P. Patra, Low-Power High-Level Synthesis for Nanoscale CMOS Circuits. Springer US, India 2008.

[50] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *in Knowledge-Based Systems*, vol. 89, pp. 228-249, Nov. 2015.

[51] J. Kennedy and R. Eberhart, "Particle swarm optimization," *in Proc. of IEEE Neural Networks*, 1995, pp. 1942-1948.

[52] M. Jhamb, Garima, H. Loohani, "Design, implementation and performance comparison of multiplier topologies in power-delay space," *in Engineering Science and Technology, an International Journal*, vol. 19, no. 1, pp. 355-363, March 2016.

محمدرضا اسماعیلی مدارک کارشناسی، کارشناسی ارشد و دکترای خود را در رشته الکترونیک از دانشگاه بیرجند به‌ترتیب در سال‌های ۱۳۹۱، ۱۳۹۳ و ۱۳۹۹ دریافت کرد. زمینه‌های مورد علاقه ایشان، پردازش سیگنال، پردازش سیگنال‌ها و علائم حیاتی، الگوریتم‌های بهینه‌سازی فراابتکاری و طراحی بهینه مدارات مجتمع است.

نشانی رایانامه ایشان عبارت است از:

Mr.esmaeili@birjand.ac.ir

سید حمید ظهیری ممقانی مدارک کارشناسی و کارشناسی ارشد خود را به‌ترتیب در رشته مهندسی برق الکترونیک از دانشگاه صنعتی شریف و تربیت مدرس دریافت کردند. ایشان در سال ۱۳۸۴ مدرک دکترای خود را در رشته مهندسی برق الکترونیک در دانشگاه فردوسی مشهد به اتمام رساندند. وی در حال حاضر استاد تمام گروه برق الکترونیک دانشکده مهندسی برق و کامپیوتر دانشگاه بیرجند هستند. زمینه‌های مورد علاقه ایشان، روش‌های بهینه‌سازی تکاملی، روش‌های بهینه‌سازی هوش جمعی، محاسبات نرم، تشخیص الگو، پردازش تصویر و پردازش سیگنال است.

نشانی رایانامه ایشان عبارت است از:

hzahiri@birjand.ac.ir

سید محمد رضوی مدرک کارشناسی خود را در رشته برق الکترونیک در سال ۱۳۷۳ از دانشگاه امیرکبیر و مدارک کارشناسی ارشد و دکترای خود را در همین رشته به‌ترتیب در سال‌های ۱۳۷۵ و ۱۳۸۵ از دانشگاه تربیت مدرس دریافت کرده است. وی در حال حاضر دانشیار گروه برق الکترونیک دانشکده مهندسی برق و کامپیوتر دانشگاه بیرجند هستند. زمینه‌های مورد علاقه ایشان، بازشناسی الگو، بازشناسی متون چاپی و دست‌نویس، پردازش تصویر و پردازش سیگنال و روش‌های بهینه‌سازی است.

نشانی رایانامه ایشان عبارت است از:

smrazavi@birjand.ac.ir

[Downloaded from jsdp.rcisp.ac.ir on 2025-11-18]

[DOI: 10.52547/jsdp.18.3.3]