انجمن کامپیوتر ایران
Computer Society of Iran

مرکز توسعه فناوری نیرو (متن)

روش جدید تولید الگوی آزمایش مبتنی بر الگوریتم ژنتیک، برای آزمایش مدارات الکترونیکی

شهرام بابائی
دانشگاه آزاد اسلامی واحد تبریز، ایران
sh.babaie@iaut.ac.ir

فرهاد نعمتی
دانشگاه آزاد اسلامی واحد تبریز، ایران
Farhad.nematy@gmail.com

نعیم رحمانی
دانشگاه آزاد اسلامی واحد تبریز، ایران
naeim.rahmani@gmail.com

چکیده: پیشرفت روز افزون علم الکترونیک همواره باعث تولید مدارات پیچیده تر و بوجود آمدن مدارات چند منظوره با وظیفه مندیهای پیچیده و حیاتی شده است. کاهش هزینه های تولید و حجم سخت افزاری ، همچنین افزایش قابلیت اطمینان و اتکاپذیری این سیستمها همواره چالشهای بزرگی سر راه محققان میباشند.آزمایش مرحله به مرحله مدارات در طی فرایند ساخت و آزمایش مدارات قبل از ورود به بازار به عنوان یکی از تکنیکهای افزایش قابلیت اطمینان مطرح میشود. یکی از مهمترین مسائل انجام آزمایش، تولید الگوی آزمایش بهینه میباشد؛که بتوان الگوهایی با طول کوتاهتر و تعداد کمتر تولید نمود.در این مقاله ضمن بررسی روشهای مختلف ارائه شده برای تولید الگوی آزمایش،مانند روشهای شبه تصادفی و روش فراگیر، یک روش تولید الگوی آزمایش مبتنی بر الگوریتم ژنتیک ارائه می دهیم.که قادر است برای آزمایش مدارات با تعداد ورودیهای زیاد الگوهای آزمایش بهینه ای تولید کند. الگوریتم ارائه شده را با الگوریتم شبه تصادفی که از پرکاربردترین روشهای تولید الگوی آزمایش میباشد مقایسه می کنیم. نتایج بدست آمده بر روی مدار های ISCAS'85 نشان می دهد که الگوریتم ارائه شده بسیار بهتر از روشهای ارائه شده قبلی عمل می کند؛ بعلاوه اینکه مجموعه الگوهای آزمایش تولید شده با این الگوریتم بسیار فشرده تر هستند.

واژه های کلیدی: آزمون پذیری، تولید الگوی آزمایش، الگوریتم ژنتیک، الگوهای شبه تصادفی

۱- مقدمه

وابستگی صنایع مختلف به فناوری الکترونیک روز به روز افزایش می-یابد؛ و افزایش پیچیدگی در آن شتاب بسیار زیادی داشته است. اما به کاربردن این گونه سیستمها مستلزم آزمایش آنها و اطمینان از صحت عملکرد آنها می‌باشد؛ چون در مدارهایی با کاربرد های حیاتی، هر گونه نقص و اشکال ممکن است عواقب بسیار خطرناکی داشته باشد. در آزمایش مدار، ورودی‌هایی به مدار تحت آزمایش اعمال می شود، که به آن الگوی آزمایش می گویند. پس از اعمال الگوهای آزمایش، پاسخ مدار به این تحریکات، با خروجی های منتظره مقایسه می‌شود، تا بتوان در مورد درست بودن یا معیوب بودن آن اظهار نظر کرد. بنابراین تولید الگوهای آزمایش و تحریکهای لازم، مرحله بسیار مهمی برای انجام آزمایش می‌باشد. روشهای مختلفی جهت تولید الگوهای آزمایش، از جمله، روش تولید الگوی فراگیر و روش شبه تصادفی ارائه شده است. روش اتوماتای سلولی و روش شیفت رجیستر فیدبک خطی دو نمونه از روش های تولید الگوی آزمایش شبه تصادفی می باشند[۱۰]. در روش آزمایش فراگیر برای یک مدار ترکیبی n ورودی، تمام ترکیبهای ممکن به مدار اعمال می‌شود؛ لذا این روش برای مدارات با بیش از ۲۰ ورودی مناسب نیست؛ زیرا همواره سعی می‌شود تعداد الگوهای آزمایش کم باشد، تا آزمایش با سرعت بیشتری انجام شود.

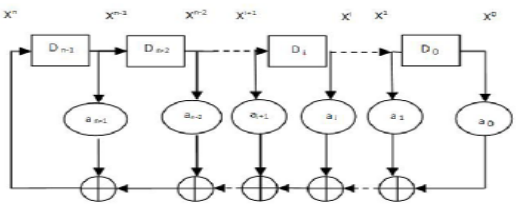
ما در این مقاله قصد داریم با استفاده از الگوریتم ژنتیک، الگوهای آزمایش مناسب تهیه کنیم. یکی از کاربردهای الگوریتم ژنتیک در بهینه سازی مسائلی است که تعداد حالتهای ممکن خیلی زیاد باشد؛که این ویژگی در بهینه سازی الگوهای آزمایش از بین 2^n الگو، که معمولا تعداد زیادی است کمک بزرگی خواهد کرد. با توجه با اینکه الگوهای شبه تصادفی از پرکاربردترین الگوها برای انجام آزمایش می‌باشند، الگوهای تولید شده مبتنی بر الگوریتم ژنتیک را با الگوهای شبه تصادفی و با اعمال بر روی مداراهای ISCAS'85 مقایسه می‌کنیم. برای اینکار از ابزار شبیه سازی Turbo Tester استفاده می‌کنیم. قسمت های مختلف مقاله شامل موارد زیر می باشد: در قسمت ۲، با روش تولید الگوی آزمایش شبه تصادفی آشنا می‌شویم. در قسمت ۳، روش جدید تولید الگوی آزمایش مبتنی برالگوریتم ژنتیک شرح داده شده است. در قسمت ۴، الگوی پیشنهادی با تولید کننده الگوی شبه تصادفی مقایسه خواهد شد و در نهایت در قسمت ۵ به نتیجه گیری خواهیم پرداخت .

۲-تولید الگوی آزمایش شبه تصادفی

Archive of SID

www.SID.ir

شیفت رجیستر فیدبک خطی یک تولید کننده الگوی آزمایش شبه تصادفی است. تعداد متنوعی ازمولدهای الگوهای شبه تصادفی مبتنی بر شیفت رجیستر فیدبک خطی برای آزماینده های توکار در[۹-۱۱] ارائه شده است.در شکل (۱) ساختار عمومی شیفت رجیستر فیدبک خطی آورده شده است؛ که شامل تعدادی فیلیپ فلاپ نوع D و یک تابع فیدبک می‌باشد. تابع فیدبک با XOR کردن برخی از خروجی های فلیپ فلاپها، ورودی n امین فیلیپ فلاپ را تولید می‌کند . وجود یا عدم وجود گیت XOR در خروجی هر کدام از فیلیپ فلاپ ها توسط چند جمله ای تولید کننده مشخص می شود. تولید الگوهای آزمایش با ایجاد حالت اولیه در شیفت رجیستر فیدبک خطی آغاز می-شود.



شکل (۱): ساختار شیفت رجیستر فیدبک خطی

رابطه کلی مدار فیدبک بصورت فرمول(1) می باشد:

$$G(x) = x^n + a_{n-1}x^{n-1} + \ldots + a_i x^i + \ldots + a_1 x + a_0 \quad (1)$$
$$a_i = 0,1$$

که در آن ضرائب a_i ضرب کننده های عددی می باشند. یک چند جمله ای اول فیدبک مناسب برای ایجاد یک مولد شبه تصادفی با حداکثر دوره تناوب می باشد؛ و قادر است 2^n-1 الگوی مختلف تولید کند؛ که n تعداد بیت شیفت رجیستر می باشد.

. تعدادی از الگوهای تولید شده توسط مولد الگوی شبه تصادفی با چند جمله ای با حالت اولیه "۱۰۰۱۱"، در جدول (۱) آورده شده است.

جدول (۱) : دنباله الگوهای تولید شده با $G(X)=X^5+X^4+X+1$

گام	الگوهای شبه تصادفی
۱	۱۰۰۱۱
۲	۱۱۰۰۱
۳	۰۱۱۰۰
۴	۰۰۱۱۰
۵	۱۰۰۱۱

همانطور که می بینیم دنباله های تولید شده دارای خاصیت شبه تصادفی می‌باشند و می‌توانند به عنوان مولد الگوهای آزمایش در آزماینده توکار استفاده شود.[۹و۱۰]

۳-روش جدید تولید الگوی آزمایش مبتنی بر الگوریتم ژنتیک

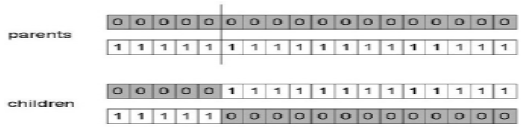
جان هلند، ابداع کننده الگوریتم ژنتیک، توانایی نمایش ساده بیتی را برای رمزنگاری ساختارهای پیچیده و قدرت تغییرات کوچک را در بهتر نمودن چنین ساختارهایی را نشان داد[۳].در الگوریتم ژنتیک ، یک راه حل ممکن از مسئله، یک فرد نامیده می‌شود. همانطور که ما اشخاص مختلف در جامعه داریم؛ همچنین راه حل های مختلف برای حل مسئله داریم که یکی بهتر از بقیه است. همه افراد با همدیگر جمعیت را تشکیل می‌دهند. در زمینه تولید الگوی آزمایش، بردار آزمایش متناظر با فرد و مجموعه بردارهای آزمایش متناظر با جمعیت خواهد بود. برای مثال یک بازنمایی الگوی آزمایش برای یک مدار با ۱۸ ورودی در شکل ۲ آورده شده است.

شکل (۲) : بازنمایی الگوی آزمایش

ابتدا یک مجموعه اتفاقی از بردار های آزمایش تولید می‌شود . این مجموعه به یک ابزار شبیه سازی داده می‌شود. حرکت از جمعیت تولید شده به صورت اتفاقی به سمت جمعیت وفق داده شده، یک محک خوب برای الگوریتم است، چون جواب نهایی توسط جستجو و مکانیسم های ترکیب الگوریتم تولید خواهد شد. البته برای کاربردهای واقعی می‌توان مقداردهی اولیه را با دقت و ساختارهای مفید انجام داد؛ که این باعث می شود خود جمعیت در ابتدای کار دارای برازندگی بالایی باشد؛ و تکامل را سرعت بخشد. مراحل بعدی الگوریتم به صورت تکراری اجرا می‌شود.

عمل ارزیابی برای سنجش میزان خوب بودن افراد در جمعیت به کار می‌رود و راه حل های بهتر امتیاز بیشتری می‌گیرند. راه حل های خوب در پروسه انتخاب، انتخاب خواهند شد؛ و راه حل‌های ضعیف رد می-شوند. بهترین بردار در جمعیت انتخاب شده و به انباره بردارهای انتخاب شده (حوضچه آمیزش) اضافه می‌شود. عمل انتخاب برای پیدا کردن دو یا چند کاندیدا برای ترکیب لازم است. بر اساس میزان کیفیت آنها بردارهای آزمایش بهتر در یک گروه آزمایش انتخاب می-شوند. بردارهای آزمایش بهتر احتمال بیشتری برای انتخاب شدن دارند. امکان دارد یک کاندیدا دوبار انتخاب شود که تولید مثل با خودش هیچ مشکلی ایجاد نمی‌کند؛ یعنی بردار انتخاب شده یک بردار خوب است و پتانسیل ژنتیکی خوب خودش را به نسل جدید انتقال می‌دهد.

تغییر اعضای ژنتیک متناظر از دو کروموزوم والد اجازه می‌دهد ژن های مفید در والد های مختلف در فرزند شان ترکیب شود. ترکیب یکی از مشخصه های کلیدی قدرت الگوریتم ژنتیک است. والد های موفق بیشتر تولید مثل می‌کنند و مشخصه های خوب دو والد ترکیب می شوند. روند انجام عمل ترکیب در شکل ۳ دیده می شود.

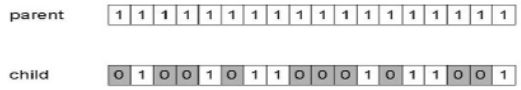


شکل (۳) : عملگر ترکیب

Archive of SID

www.SID.ir

جهش بعضی وقتها چیزهای مفیدی را به کروموزوم ها معرفی می‌کند. بدون جهش، همه افراد در جمعیت به زودی مثل هم می‌شوند. در شکل ۴ جهش را مشاهده می کنید.در همه بردارهای آزمایش ، هر بیت با احتمال p معکوس می‌شود. الگوریتم تکرار می‌شوند تا وقتی که همه خطاها پیدا شوند یا به یک تعداد تعریف شده نسل ها برسیم.



شکل (۴) : عملگر جهش

۴-مقایسه الگوی پیشنهادی با الگوی شبه تصادفی

در این قسمت قصد داریم الگوهای آزمایش رائه شده را با الگوهای شبه تصادفی مقایسه کنیم. برای شبیه سازی از ابزار TurboTester استفاده کردیم و شبیه سازی ها را روی مدارهای ISCAS'85 benchmark انجام دادیم. مدار های ISCAS'85 benchmark مدارهایی هستند که برای مقایسه تولید کننده های الگوی آزمایشی که محققان پیشنهاد می‌کنند استفاده می‌شوند. در اولین شبیه سازی تعداد بردارها یکسان در نظر گرفته نشد و نتیجه این بود که روش ژنتیک همیشه تعداد بردار آزمایش کمتری برای پوشش خطای یکسان با روش تصادفی نیاز دارد. نتایج را در جدول شماره ۲ مشاهده می‌کنید برای مثال روش تصادفی برای مدار c2670 از ۲۶۲۶ خطای موجود تعداد ۲۲۲۶ خطا را با ۹۴ بردارشناسایی می‌کند در حالی که روش ژنتیک تعداد ۲۲۴۲ خطا را با ۶۶ بردار شناسایی می‌کند ویا در مورد مدار c880 از۹۹۴ خطا روش تصادفی فقط ۹۸۸ خطا را به وسیله ۹۲ بردار و روش ژنتیک همه ۹۹۴ خطا را به وسیله ۴۹ بردار آشکار کرده و پوشش خطای ۱۰۰٪ دارد. یکبار هم شبیه سازی با تعداد بردار برابر برای هر دو روش انجام شد که درصد پوشش خطای روش ژنتیک بیشتر از روش تصادفی بود. برای مثال در مورد مدار c2670 با تعداد بردار یکسان روش تصادفی از ۲۶۲۶ خطای موجود فقط ۱۹۶۵ خطا را شناسایی کرد که درصد پوشش خطای آن برابر۷۴ ٪ است ولی روش ژنتیک ۲۲۲۶ خطا را شناسایی کرد که درصد پوشش خطای آن ۸۵ ٪ می‌باشد. نتایج را در جدول شماره ۳ می‌بینید

جدول شماره (۲) : نتایج شبیه سازی با تعداد بردار آزمایش متفاوت

		c432	C880	C499	C1908	C1355	C2670	C3540	C5315	C7552	C6288
تصادفی	تعداد خطا	۵۷۳/۶۱۶	۹۸۸/۹۹۴	۱۱۹۴/۱۲۰۲	۱۷۲۲/۱۷۳۲	۱۶۱۰/۱۶۱۸	۲۲۲۶/۲۶۲۶	۳۱۴۵/۳۲۹۶	۵۳۶۵/۵۴۲۴	۶۶۴۹/۷۱۰۴	۷۶۹۳/۷۷۴۴
	درصد پوشش	۹۳ ٪	۹۹/۳۹ ٪	۹۹/۳۳ ٪	۹۹/۴۲ ٪	۹۹/۵۰ ٪	۸۴/۷۶ ٪	۹۵/۴۱ ٪	۹۸/۸۹ ٪	۹۳/۵۹ ٪	۹۹/۳۴ ٪
	تعداد بردار	۷۶	۹۲	۱۰۱	۱۸۴	۹۹	۹۴	۲۵۰	۲۱۳	۲۷۸	۴۷
	زمان	۰/۰۱۵	۰/۰۹۴	۰/۰۷۸	۰/۱۷۲	۰/۱۰۹	۰/۱۷۲	۰/۲۹۶	۰/۳۱۳	۱/۶۷۱	۰/۳۹۱
ژنتیک	تعداد خطا	۵۷۳/۶۱۶	۹۹۴/۹۹۴	۱۱۹۴/۱۲۰۲	۱۷۲۲/۱۷۳۲	۱۶۱۰/۱۶۱۸	۲۲۴۲/۲۶۲۶	۳۱۴۷/۳۲۹۶	۵۳۶۴/۵۴۲۴	۶۷۳۵/۷۱۰۴	۷۶۹۳/۷۷۴۴
	درصد پوشش	۹۳ ٪	۱۰۰ ٪	۹۹/۳۳ ٪	۹۹/۴۸ ٪	۹۹/۵۰ ٪	۸۵/۳۷ ٪	۹۵/۴۷ ٪	۹۸/۸۹ ٪	۹۴/۸۰ ٪	۹۹/۳۴ ٪
	تعداد بردار	۵۱	۴۹	۸۶	۱۲۰	۸۶	۶۶	۱۵۵	۱۲۸	۱۸۵	۲۲
	زمان	۰/۲۵	۰/۳۹۱	۰/۴۰۶	۰/۵۹۳	۰/۵۹۳	۱/۷	۱/۱۲۵	۱/۹۲۱	۳/۲۸۱	۳/۲۵

جدول شماره (۳) : نتایج شبیه سازی با تعداد بردار یکسان

		c432	c880	c499	c1908	c1355	c2670	c3540	c5315	c7552	c6288
تصادفی	تعداد خطا	۴۸۶/۶۱۶	۸۲۷/۹۹۴	۱۰۴۲/۱۲۰۲	۱۱۷۶/۱۷۳۲	۱۳۰۵/۱۶۱۸	۱۹۶۵/۲۶۲۶	۲۵۵۰/۳۲۹۶	۵۰۶۰/۵۴۲۴	۶۲۲۰/۷۱۰۴	۷۵۱۵/۷۷۴۴
	درصد پوشش	۷۸/۸۹ ٪	۸۳/۱۹ ٪	۸۶/۶۸ ٪	۶۷/۸۹ ٪	۸۰/۶۵ ٪	۷۴/۸۲ ٪	۷۷/۳۶ ٪	۹۳/۲۸ ٪	۸۷/۵۵ ٪	۹۷/۰۴ ٪
	تعداد بردار	۵۱	۴۹	۸۶	۱۲۰	۸۶	۶۶	۱۵۵	۱۲۸	۱۸۵	۲۲
	زمان	۰/۰۱۵	۰/۰۰۰	۰/۰۰۰	۰/۰۰۰	۰/۰۰۰	۰/۰۱۵	۰/۰۳۱	۰/۰۴۷	۰/۱۲۵	۰/۰۳۱
ژنتیک	تعداد خطا	۵۷۳/۶۱۶	۹۹۴/۹۹۴	۱۱۹۴/۱۲۰۲	۱۷۲۳/۱۷۳۲	۱۶۱۰/۱۶۱۸	۲۲۲۶/۲۶۲۶	۳۱۴۷/۳۲۹۶	۵۳۶۴/۵۴۲۴	۶۷۳۵/۷۱۰۴	۷۶۹۳/۷۷۴۴
	درصد پوشش	۹۳ ٪	۱۰۰ ٪	۹۹/۳۳ ٪	۹۹/۴۸ ٪	۹۹/۵۰ ٪	۸۵/۳۷ ٪	۹۵/۴۷ ٪	۹۸/۸۹ ٪	۹۴/۸۰ ٪	۹۹/۳۴ ٪
	تعداد بردار	۵۱	۴۹	۸۶	۱۲۰	۸۶	۶۶	۱۵۵	۱۲۸	۱۸۵	۲۲
	زمان	۰/۲۵	۰/۳۹۱	۰/۴۰۶	۰/۵۹۳	۰/۵۹۳	۱/۷۰	۱/۱۲۵	۱/۹۲۱	۳/۲۸۱	۳/۲۵

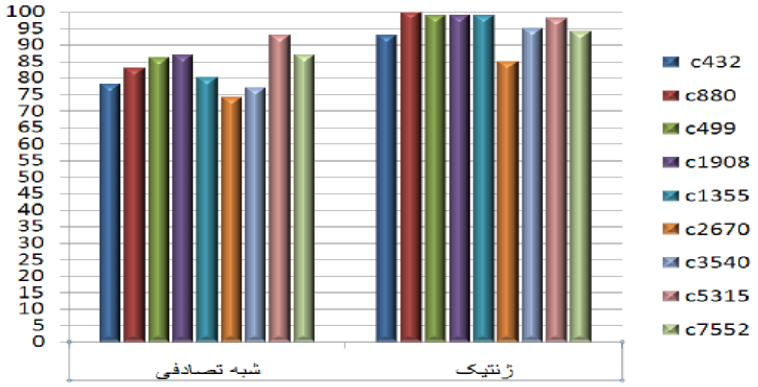
Archive of SID

www.SID.ir

زمان انجام آزمایش برای روش تصادفی کمی کوتاهتر از روش ژنتیک می باشد اما باید توجه داشت که چون عمل تولید الگوی آزمایش فقط یک بار انجام می‌شود و دفعات متعدد از آن برای آزمایش مدار تحت آزمایش استفاده می‌شود، طولانی بودن زمان تولید الگوی آزمایش اهمیت ندارد.

۵- نتیجه گیری

در این مقاله ما دو روش ژنتیک و تصادفی را مقایسه کردیم که بر اساس نتایج بدست آمده کارایی تولید کننده ژنتیک در مورد مدارهایی که تعداد زیادی ورودی دارند ثابت می شود. پس از مقایسه الگوهای آزمایش ارائه شده با الگوهای شبه تصادفی میتوان نتیجه گرفت که تعداد الگوی آزمایش مبتنی بر الگوریتم ژنتیک بمراتب کمتر از الگوهای شبه تصادفی برای پوشش خطای یکسان می باشد. حتی دیدیم که الگوهای ارائه شده قادر به پوشش تعداد خطای بیشتر هم می باشند. در فاز دوم شبیه سازی که تعداد الگوهای آزمایش در هر دو روش ثابت نگه داشته شده بود، دیدیم که الگوهای ژنتیک تا حد قابل ملاحظه ای پوشش خطای بیشتری نسبت به الگوهای شبه تصادفی دارا می باشند. نمودار شماره ۱ به صورت مقایسه‌ای درصد پوشش هر دو روش را نشان می‌دهد. ما نتایجمان را با نتایج در [۲] مقایسه کردیم. روش آنها همه خطاها را برای c7552,c2670 پیدا کردند ولی روش ما بردارهای آزمایش کمتری را به کار می‌برد. همچنین نتایج ما بهتر از نتایج بدست آمده در [۱۱] می‌باشد.



نمودار شماره (۱) : نمودار مقایسه ای درصد پوشش خط

مراجع

[1] D. G. Saab, Y. G. Saab, J. A. Abraham, “CRIS: A test Cultivation Program for Sequential VLSI Circuits”,Intl. Conf. Computer-Aided Design, Nov. 1992, pp. 216-219.

[2] I. Pomeranz, S. M. Reddy, “On improving Genetic Optimization based Test Generation”, Proc. Of European Design and Test Conference 1997, pp. 506-511

[3] J.H. Holland, “Adaptation in Natural and Artificial Systems”, University of Michigan Press, 1975

[4] E. M. Rudnick, J. H. Patel, G. S. Greenstein, T. M. Niermann “Sequential Circuit Generation in a Genetic Framework”, 31st ACM/IEEE Design Automation Conference

[5] Goldberg “Genetic algorithms”, Addison-Wesley USA,1991

[6] F. Berglez, H. Fujiwara, “A Neutral Netlist of 10 Combinational Benchmark Circuits and a Target Translator in Fortran”, Proc. of the Int. Test Conf., pp.785-794, 1985.

[7] S. R. Ladd, “Genetic Algorithms in C++”, M&T Books, 1996

[8] R.Ubar, J.Raik, P.Paomets, E.Ivask, G.Jervan, A.Markus. “Low-Cost CAD System for Teaching Digital Test”, .Microelectronics Education. World Scientific Publishing Co. Pte. Ltd. pp. 185-188, Grenoble, France, Feb. 1996

[9] A. Crouch. *Design-for-test for Digital IC’s and Embedded Core Systems*. Prentice Hall, 1999,347 p.

[10] M. Bushnell, V. Agrawal. *Essentials of Electronic Testing for Digital Memory & Mixed Signal VLSI Circuits*. Kluwer Academic Publishers, 2000, 712 p.

[11] Aleksejev, Jevgeni; Jutman, Artur; Ubar, Raimund (2006). LFSR polynomial and seed selection using genetic algorithm. Baltic Electronics Conference. IEEE, 2006, 179 - 182.

Archive of SID

www.SID.ir