به نام خدا

مستندات پروژه رمزنگاری به روش SDES

اعضای گروه:

علی باقرزاده 98521072

مهدی مصطفوی 98521495

پوریا زمانی 98521234

شرح وظایف هر نفر و تقسیم کار پروژه در [*ریپازیتوری گیت‌هاپ*](https://github.com/AliBagherz/DlProject) این پروژه به صورت کامل گنجانده شده است.

این پروژه شامل برنامه های رمزنگار و رمزگشا می باشد که با کمک کامپوننت های زیر پیاده شده است.

شرح کامپوننت ها: در این پروژه ما 13 کامپوننت پایه داریم که وابستگی به سایر کامپوننت ها ندارند و همچنین 3 کامپوننت مرکب داریم که در آن از سایر کامپوننت های قبلی استفاده شده است.

کامپوننت IP:

در این کامپوننت عددی 8 بیتی به عنوان ورودی گرفته میشود و جایگشت آنها تغییر پیدا میکند. به صورت:

IP(b0,b1,b2,b3,b4,b5,b6,b7) = b1b5b2b0b3b7b4b6

کامپوننت EP:

این کامپوننت یک عدد 4 بیتی را به عنوان ورودی گرفته و آنرا به صورت زیر به 8 بیت گسترش میدهد:

EP(b0,b1,b2,b3) = b3b0b1b2b1b2b3b0

کامپوننت P4:

کامپوننت p4 یک عدد 4 بیتی را به عنوان ورودی دریافت کرده و سپس به شکل زیر جایگشت آنرا تغییر میدهد:

P4(b0,b1,b2,b3) = b1b3b2b0

کامپوننت SW:

در این کامپوننت یک عدد 8 بیتی به عنوان ورودی گرفته میشود و جی 4 بیت Msb و 4 بیت Lsb آن با هم جابجا میشوند.

SW(b0,b1,b2,b3,b4,b5,b6,b7) = b4b5b6b7b0b1b2b3

کامپوننت IP-1:

در این کامپوننت عددی 8 بیتی به عنوان ورودی گرفته میشود و جایگشت آنها به شکل زیر تغییر پیدا میکند:

IP-1(b0,b1,b2,b3,b4,b5,b6,b7) = b3b0b2b4b6b1b7b5

کامپوننت P8:

این کامپوننت یک عدد 10 بیتی را به عنوان ورودی میگیرد و 8 بیت را به عنوان خروجی به شکل زیر جایگشت میدهد.

P8(b0,b1,b2,b3,b4,b5,b6,b7,b8,b9) = b5b2b6b3b7b4b9b8

کامپوننت P10:

این کامپوننت همانند کامپوننت P8،عدد 10 بیتی را به عنوان ورودی دریافت میکند با این تفاوت که خروجی آن یک عدد 10 بیتی با جایگشت زیر میباشد.

P8(b0,b1,b2,b3,b4,b5,b6,b7,b8,b9) = b2b4b1b6b3b9b0b8b7b5

کامپوننت Xor4Bit و Xor8Bit:

این دو کامپوننت دو عدد 4 و یا 8 بیتی را دریافت می کند و بیت های متناظر آن را با هم xor می کند و عدد 4 یا 8 بیتی نتیجه را خروجی می دهد.

کامپوننت Ls-1 و Ls-2:

این دو کامپوننت عددی 5 بیتی را ورودی گرفته و به ترتیب این اعداد را 1 و 2 بیت به صورت چرخشی به سمت چپ شیفت می دهند

Ls-1(b0,b1,b2,b3,b4) = b1b2b3b4b0

Ls-2(b0,b1,b2,b3,b4) = b2b3b4b0b1

کامپوننت S0:

این کامپوننت عددی 4 بیتی را ورودی می گیرد و با استفاده از ماتریس زیر عددی 2 بیتی را خروجی می دهد

دو بیت Msb تعیین کننده شماره سطر و دو بیت Lsb تعیین کننده شماره درایه مدنظر در ماتریس می باشند و در نهایت عدد آن درایه ماتریس را به صورت دوبیتی خروجی می‌دهیم.

جدول درستی و کارنو های خروجی در صفحه بعد قرار داده شده است

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| O1 | O0 | A2 | A2 | A1 | A0 |
| 1 | **0** | **0** | **0** | **0** | **0** |
| 0 | **0** | **1** | **0** | **0** | **0** |
| 1 | **1** | **0** | **1** | **0** | **0** |
| 0 | **1** | **1** | **1** | **0** | **0** |
| 1 | **1** | **0** | **0** | **1** | **0** |
| 0 | **1** | **1** | **0** | **1** | **0** |
| 1 | **0** | **0** | **1** | **1** | **0** |
| 0 | **0** | **1** | **1** | **1** | **0** |
| 0 | **0** | **0** | **0** | **0** | **1** |
| 0 | **1** | **1** | **0** | **0** | **1** |
| 1 | **0** | **0** | **1** | **0** | **1** |
| 1 | **1** | **1** | **1** | **0** | **1** |
| 1 | **1** | **0** | **0** | **1** | **1** |
| 1 | **0** | **1** | **0** | **1** | **1** |
| 1 | **1** | **0** | **1** | **1** | **1** |
| 0 | **1** | **1** | **1** | **1** | **1** |

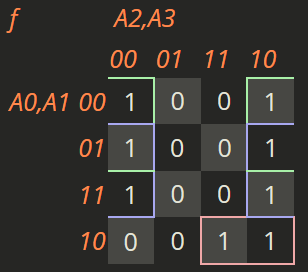
و حال می توانیم با رسم این دو کارنومپ روابط O0 و O1 را محاسبه کنیم.

جدول کارنو و رابطه O0:



O0 = A0'A1'A2 + A0'A1A2' + A0A1'A3 + A1A2'A3' + A0A1A2

جدول کارنو و رابطه O1:



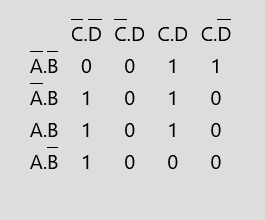
O1 = A0'A3' + A0A1'A2 + A1A3'

کامپوننت S1:

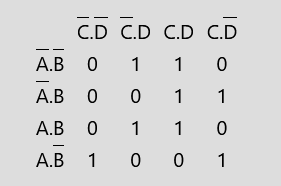
این کامپوننت عددی 4 بیتی را ورودی می گیرد و با استفاده از ماتریس زیر عددی 2 بیتی را خروجی می دهد

مشابه S0 است.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| O1 | O0 | D | C | B | A |
| 0 | **0** | **0** | **0** | **0** | **0** |
| 1 | **0** | **1** | **0** | **0** | **0** |
| 0 | **1** | **0** | **1** | **0** | **0** |
| 1 | **1** | **1** | **1** | **0** | **0** |
| 0 | **1** | **0** | **0** | **1** | **0** |
| 0 | **0** | **1** | **0** | **1** | **0** |
| 1 | **0** | **0** | **1** | **1** | **0** |
| 1 | **1** | **1** | **1** | **1** | **0** |
| 1 | **1** | **0** | **0** | **0** | **1** |
| 0 | **0** | **1** | **0** | **0** | **1** |
| 1 | **0** | **0** | **1** | **0** | **1** |
| 0 | **0** | **1** | **1** | **0** | **1** |
| 0 | **1** | **0** | **0** | **1** | **1** |
| 1 | **0** | **1** | **0** | **1** | **1** |
| 0 | **0** | **0** | **1** | **1** | **1** |
| 1 | **1** | **1** | **1** | **1** | **1** |

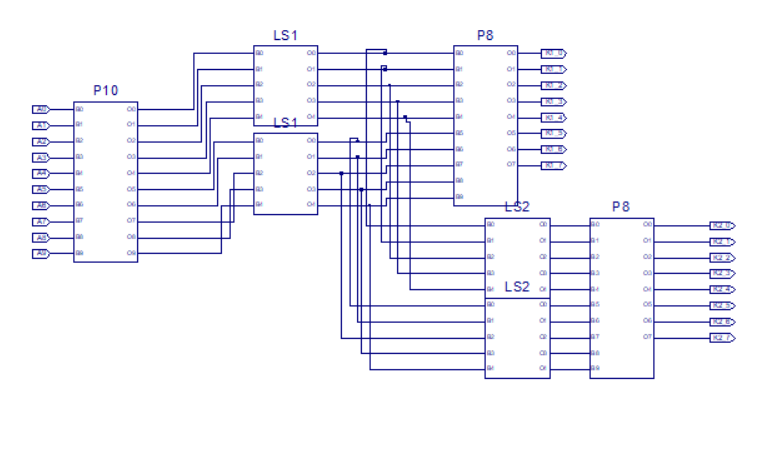


O0 = A'B'C + BC'D' + BCD + AC'D'

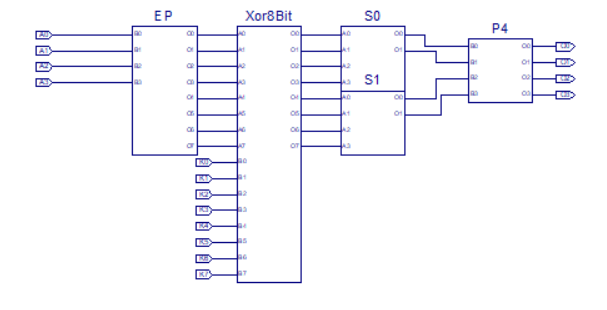


O1 = A'B'D + A'BC + AB'D' + ABD

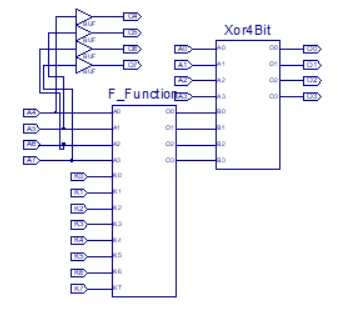
کامپونت key :  
این کامپوننت وظیفه‌ی طراحی کلیدهای k1 و k2 را با استفاده از 10 بیت ورودی دارد. ابتدا تمامی بیت ها وارد P10 شده، سپس وارد LS1 خواهد شد، قسمتی از این کلیدها با ورود به P8 باعث تولید k1 شده و قسمتی دیگر بعد از LS1 وارد LS2 و سپس P8 شده و k2 را تولید میکنند.



کامپوننت FFunction:  
این کامپوننت طبق توضیح مستندات پروژه تعریف شده است و زیر مجموعه ای از کامپوننت round می باشد. این کامپوننت 4 بیت ورودی گرفته و در نهایت 4 بیت خروجی میدهد.



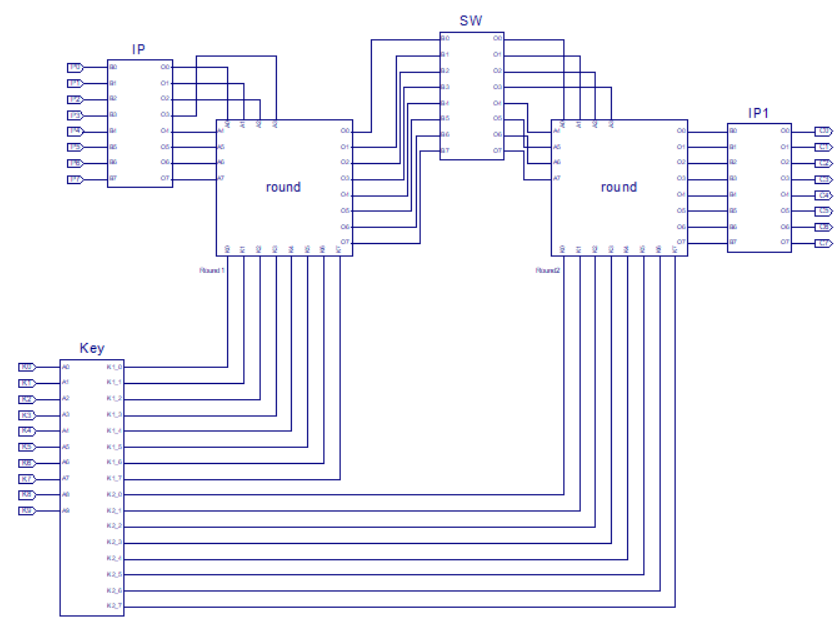
کامپوننت round:  
این کامپوننت برای سادگی بیشتر برنامه، محیط FFunction و XOR را در برمیگیرد. به صورت کلی میتواند محیط این کامپوننت را به صورت زیر آشکار کرد که شامل 8 بیت ورودی، 8 بیت کلید و 8 بیت خروجی است



برنامه Encrypt یا رمزنگاری:

این فایل متشکل از کامپوننت های round ,key, ip, ip-1,sw می‌باشد که مطابق نقشه زیر به هم متصل شده اند

ورودی این برنامه 10 بیت کلید و 8 بیت متن ساده می‌باشد که در خروجی 8 بیت متن رمزنگاری شده را برمیگرداند.



برنامه Decrypt یا رمزگشایی:

این برنامه کاملا برعکس رمزنگار عمل کرده و با گرفتن 8 بیت متن رمزنگاری شده و کلید 10 بیتی متن ساده را به ما خروجی می‌دهد. نقشه این برنامه کاملا مشابه با با برنامه رمز نگار می باشد با این تفاوت که کلید ها به صورت برعکس به round اول و دوم وارد می‌شود.

