فاز اول (بیت‌کوین چگونه کار می‌کند؟)

بیت‌کوین[1] یک واحد پول دیجیتالی است که در سال ۲۰۰۹ توسط ساتوشی ناکاموتو معرفی شد. بیت‌کوین نوعی پول الکترونیکی است که به کاربران اجازه می‌دهد تا دارایی خود را از حسابی به حساب دیگر منتقل کنند؛ بدون اینکه نیاز به مداخله موسسات مالی و اعتباری مانند بانک‌ها وجود داشته باشد.

در سیستم‌های سنتی، موسسات مالی به عنوان محلی امن برای ذخیره و نگهداری دارایی افراد محسوب می‌شوند. در این سیستم‌ها، مدیریت انتقال دارایی و پول بین مشتری‌ها و سرویس‌گیرنده‌ها برعهده بانک است. سیستم سنتی مدیریت دارایی معایبی دارد از جمله اینکه انتقال الکترونیکی موجودی‌ها و تراکنش‌ها بین شعب مختلف بانک‌ها هزینه[2] زیادی را تحمیل می‌کند. این تراکنش علاوه بر هزینه فرآیندی زمان‌بر است. از همه مهم‌تر انتقال تراکنش‌ها نیاز به بستری امن دارد تا اطلاعات توسط افراد ناشناس سرقت نشود.

سایر سیستم‌های مدیریت دارایی مانند Visa، MasterCard و PayPal نیز هزینه زیادی را برای انتقال پول تحمیل می‌کنند. در مقابل بیت‌کوین به عنوان یک سیستم انتقال پول و ارز، واسطه‌ها را حذف کرده است و در عوض تراکنش‌ها را در یک شبکه نظیر به نظیر[3] انجام می‌دهد. در شبکه بیت‌کوین به جای استفاده از ابزارهای کمکی[4] برای امن کردن مسیر انتقال، از رمزنگاری استفاده می‌شود. برای ساخت و تصدیق امضاهای دیجیتالی که کاربران برای انجام تراکنش‌های خود از آن استفاده می‌کنند، از رمزنگاری مبتنی بر کلید عمومی و خصوصی استفاده می‌شود. این رمزنگاری اساس کار شبکه بیت‌کوین را تشکیل می‌دهد.

انتقال اطلاعات بین گیرنده و فرستنده نهایی بیت‌کوین در شبکه، زنجیره‌ای از بلاک‌ها را تشکیل می‌دهد که هر بلاک شامل رکورد انتقال بیت‌کوین مابین دو آدرس مشخص در شبکه است. هر بلاک قبل پیوستن به زنجیره نیاز به تصدیق دارد. در این زنجیره اگر یکی از بلاک‌ها دستکاری یا دچار تغییر شود، تمامی بلاک‌های بعد نیاز به تصدیق مجدد دارند. نکته دیگر اینکه زمانی که بلاک تراکنش بین دو کاربر به زنجیره بلاک متصل می‌شود، گیرنده مطمئن است که این تراکنش توسط تمامی کامپیوترهای شبکه رکورد شده است. این باعث می‌شود تا فرستنده نتواند بیت‌کوین مشابه را مجدداً[5] برای کاربر دیگری ارسال کند.

در شبکه بیت‌کوین (مجموعه‌ای از سیستم‌ها (Bitcoin Clients) که نرم افزار بیت‌کوین را اجرا می‌کنند و به اینترنت متصل هستند) هر بلاک قبل از پیوستن به زنجیره، نیاز به تأیید و تصدیق دارد تا از اعتبار بلاک

[1] Bitcoin
[2] Transaction Fee
[3] Peer-to-Peer
[4] Third Parity
[5] Double spending

اطمینان حاصل شود که به این فرآیند Proof of Work می‌گویند. این فرآیند نیاز به زمان محاسبات زیادی دارد. شبکه بیت‌کوین برای فرآیند Proof of Work از تابع SUDO-SHA-256 استفاده می‌کند. ضمنا لازم به ذکر است که هر کلاینت در شبکه بیت‌کوین علاوه بر اینکه می‌تواند بلاک جدیدی در شبکه تولید کند، می‌تواند عملیات Proof of Work را بر روی بلاک‌های موجود در شبکه بیت‌کوین انجام دهد. نکته دیگر آنکه هر کلاینت در شبکه تمامی بلاک‌های تائید شده را نگهداری می‌کند (دانلود و اجرای نرم افزار بیت‌کوین به معنی پیوستن شما به شبکه بیت‌کوین بوده و وجود این بلاک‌های تائید شده بر روی کلاینت را نشان می‌دهد).

حاصل فرآیند Proof of Work استخراج[6] نامیده می‌شود که در نتیجه آن تراکنش جدیدی به زنجیره بلاک افزوده می‌شود. همه کلاینت‌ها در شبکه، اطلاعات مربوط به تراکنش‌های جدید را به اشتراک می‌گذارند، یعنی اگر تراکنش جدید رخ دهد هر کلاینت در شبکه این تراکنش را بر روی سیستم خود دریافت و نگهداری می‌کند. در این هنگام هر کلاینت این اختیار را دارد تا به استخراج بیت‌کوین بپردازد و تراکنش‌های جدید را به زنجیره بلاک اضافه کند. همانطور که قبلا گفته شد، اضافه کردن تراکنش، به لیست تراکنش‌های تائید شده (زنجیره بلاک) نیاز به قدرت پردازشی زیادی دارد که در ادامه دلیل این نیاز ذکر می‌شود. اما بعد از اضافه کردن بلاک به زنجیره، به عنوان پاداش ۲۵ بیت‌کوین به شخص برنده اهدا می‌شود، که موجب افزایش دارایی شخص می‌شود و نام استخراج (مشابه استخراج طلا از معدن) به همین دلیل اقتباس شده است.

تابع رمزنگاری sudo-SHA-256 (در این تابع با داشتن ورودی می‌توان به خروجی دست یافت ولی با داشتن خروجی به دست آوردن ورودی تقریباً غیرممکن است) یک پیام با طول متغیر (به عنوان ورودی) را به یک رشته با طول ثابت (به عنوان خروجی) نگاشت می‌کند (پیام با طول متغیر نشان دهنده تراکنشی است که در نظر داریم تا با فرآیند Hash آن را به زنجیره اضافه کنیم و پیام با طول ثابت تراکنشی است که باید به زنجیره اضافه شود). خروجی الگوریتم sudo-SHA-256 یک پیام ۲۵۶ بیتی است.

این الگوریتم شامل سه مرحله است:

مرحله اول: چسباندن و تجزیه[7]

به پیام خود یک عدد ’1‘ و به تعداد کافی ’0‘ اضافه می‌کنیم به طوری که طول کل پیام به پیمانه ۵۱۲ برابر ۴۴۸ گردد ($L+1+k=448 \bmod 512$). که L طول پیام اصلی و k تعداد صفرهای اضافه شده است. فرض کنید که می‌خواهیم پیام "abc" را رمزنگاری کنیم.

[6] Mining
[7] Padding and Parsing

طول پیام برحسب بیت (با فرض اینکه هر کاراکتر ۸ بیت است) برابر است با ۲۴ بیت. بیت ’1‘ را به انتهای ۲۴ بیت اضافه می‌کنیم و به تعداد ۴۲۳ بیت ’0‘ و در انتها عدد معادل L را به صورت یک عدد باینری ۶۴ بیتی به حاصل قبلی می‌افزاییم. حال خروجی مرحله اول به صورت زیر است، عدد باینری حاصل را به بلاک‌های ۵۱۲ بیتی ($M^{(1)}, M^{(2)}, ..., M^{(N)}$) تقسیم می‌کنیم. (در این مثال به دلیل کوتاه بودن طول پیام فقط یک بلاک ایجاد می‌شود ولی اگر طول پیام بزرگ باشد تعداد بلاک‌ها می‌تواند بیشتر شود).

$L = 3 \times 8 = 24$

$L + 1 + k = 448 \quad => \quad k = 423\,bit$

01100001	01100010	01100011	1	0000….0000	0000….11000
a, 8bit	b, 8bit	c, 8bit	1 bit pad	423 bit pad	L in binary

مرحله دوم: گسترش پیام[8]

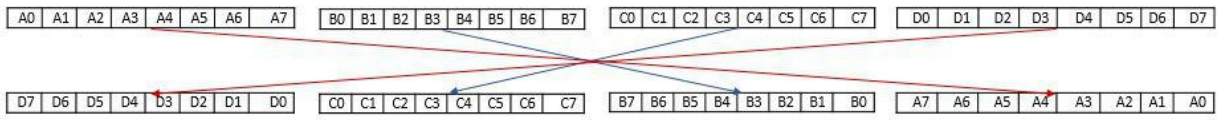
در این مرحله هر بلاک ۵۱۲ بیتی حاصل از مرحله قبل به ۱۶ بلاک ۳۲ بیتی ($M_t^{(i)} \ for\ 0 < t < 15$) تقسیم می‌شود. در ادامه هر بلاک ۵۱۲ بیتی به ۶۴ بلاک ۳۲ بیتی (W_t) به صورت زیر بسط می‌یابد:

$$W_t = \begin{cases} M_t^i & 0 < t < 15 \\ \sigma_1(W_{t-1}) + W_{t-6} + \sigma_0(W_{t-12}) + W_{t-18} & 16 < t < 63 \end{cases}$$

$$\sigma_0(x) = ROT_{17}(x) \oplus ROT_{14}(x) \oplus SHF_{12}(x)$$

$$\sigma_1(x) = ROT_9(x) \oplus ROT_{19}(x) \oplus SHF_9(x)$$

در فرمول‌های فوق $ROT_n(x)$ عملیات n مرتبه شیفت دورانی به راست است و $SHF_n(x)$ عملیات n مرتبه شیفت منطقی به راست x است و عملیات + نشان دهنده جمع باینری ۳۲ بیتی است. در مرحله بعد هر بلاک ۳۲ بیتی از یک تابع جایگشت[9] عبور داده می‌شود. دیاگرام زیر نشان‌دهنده این تابع است.



مرحله سوم: فشرده‌سازی[10]

در این مرحله W_iهای حاصل از مرحله قبل به عنوان ورودی این مرحله محسوب می‌شوند. تابع فشرده‌سازی دارای ۸ متغیر ۳۲ بیتی A…H است که با ۳۲ بیت اول بخش اعشاری ریشه دوم ۸ پرایم اول ($H_0^{(0)} - H_7^{(0)}$)

[8] Expansion
[9] Permutation Box
[10] Compression

تحویل مطابق جدول زمانبندی

در شروع هر بار فراخوانی تابع فشرده‌سازی مقداردهی اولیه می‌شوند. در ادامه تابع فشرده‌سازی ۶۴ بار تکرار می‌شود و به صورت زیر مشخص می‌شود:

$$H_1^{(0)} = 6a09e667- \quad H_2^{(0)} = bb67ae85- \quad H_3^{(0)} = 3c6ef372- \quad H_4^{(0)} = a54ff53a$$
$$H_5^{(0)} = 510e527f- \quad H_6^{(0)} = 9b05688c- \quad H_7^{(0)} = 1f83d9ab- \quad H_8^{(0)} = 5be0cd19$$
$$T_2 = H + \sum_1(E) + Ch(E,F,G) + K_t + W_t$$
$$T_1 = \sum_0(A) + Maj(A,B,C) + \sum_2(C+D)$$
$$H = G$$
$$F = E$$
$$D = C$$
$$B = A$$
$$G = F$$
$$E = D + T_1$$
$$C = B$$
$$A = 3T_1 - T_2$$

که در این روابط داریم:

$$Ch(x,y,z) = (x\ AND\ y) \oplus (\overline{y}\ \text{AND}\ z) \oplus (\overline{x}\ \text{AND}\ z)$$
$$Maj(x,y,z) = (x\ AND\ z) \oplus (x\ \text{AND}\ y) \oplus (y\ AND\ z)$$
$$\sum_0(x) = ROT_2(x) \oplus ROT_{13}(x) \oplus ROT_{22}(x) \oplus SHF_7(x)$$
$$\sum_1(x) = ROT_6(x) \oplus ROT_{11}(x) \oplus ROT_{25}(x)$$
$$\sum_2(x) = ROT_2(x) \oplus ROT_3(x) \oplus ROT_{15}(x) \oplus SHF_5(x)$$

ورودی‌های K_t، ۶۴ ثابت ۳۲ بیتی هستند که با ۳۲ بیت اول بخش کسری ریشه سوم ۶۴ پرایم اول مقداردهی اولیه می‌کنند. بعد از ۶۴ مرتبه تکرار تابع فشرده‌سازی مقادیر میانی $H^{(i)}$ بصورت زیر محاسبه می‌شوند:

$$H_0^{(i)} = A + H_0^{(i-1)} - H_1^{(i)} = B + H_1^{(i-1)} - H_2^{(i)} = C + H_2^{(i-1)} - H_3^{(i)} = D + H_3^{(i-1)}$$
$$H_4^{(i)} = E + H_4^{(i-1)} - H_5^{(i)} = F + H_5^{(i-1)} - H_6^{(i)} = G + H_6^{(i-1)} - H_7^{(i)} = H + H_7^{(i-1)}$$

و مقادیر K_t بصورت زیر مقدار دهی اولیه می‌شوند:

$$428a298-71374491-b5c0\,fbcf\,-e9b5dba5-3956c25b-59\,f111\,f1-923\,f82a4-ab1c5ed5$$
$$d807aa98-12835b01-243185be-550c7dc3-72be5d74-80deb1fe-9bdc06a7-c19bf174$$
$$e49b69c1-efbe4786-0\,fc19dc6-240ca1cc-2de92c6\,f\,-4a7484aa-5cb0a9dc-76\,f988da$$
$$983e5152-a831c66d-b00327c8-bf\,597\,fc7-c6e00bf\,3-d5a79147-06ca6351-14292967$$
$$27b70a85-2e1b2138-4d\,2c6dfc-53380d13-650a7354-766a0abb-81c2c92e-92722c85$$
$$a2bfe8a1-a81a664b-c24b8b70-c76c51a3-d192e819-d6990624-\,f\,40e3585-106aa070$$
$$19a4c116-1e376c08-2748774c-34b0bcb5-391c0cb3-4ed8aa4a-5b9cca4\,f\,-682e6\,ff\,3$$
$$748\,f\,82ee-78a5636\,f\,-84c87814-8cc70208-90befffa\,-a4506ceb-be49a3\,f\,7-c67178\,f\,2$$

الگوریتم فشرده‌سازی sodu-SHA-256 بر روی تمامی بلاک‌های ۵۱۲ بیتی دیگر تکرار می‌شود. ۲۵۶ بیت خروجی ($H^{(N)}$) با پشت سرهم قرار دادن (Concatenate) مقادیر زیر حاصل می‌شود:

$$H^{(N)} = H_0^{(N)} \,\&\, H_1^{(N)} \,\&\, H_2^{(N)} \,\&\, H_3^{(N)} \,\&\, H_4^{(N)} \,\&\, H_5^{(N)} \,\&\, H_6^{(N)} \,\&\, H_7^{(N)}$$

هر بلاک در زنجیره دارای سرآیند است که اطلاعاتی را در مورد بلاک می‌دهد. بخش‌های مختلف سرآیند در شکل ۱ نشان داده شده است و در جدول نیز بخش‌های مختلف به تفکیک معرفی شده است.

Version	hashPrevBlock	hashMerkelRoot	Time	Difficulty	nonce
4 byte	32 byte	32 byte	4 byte	4 byte	4 byte

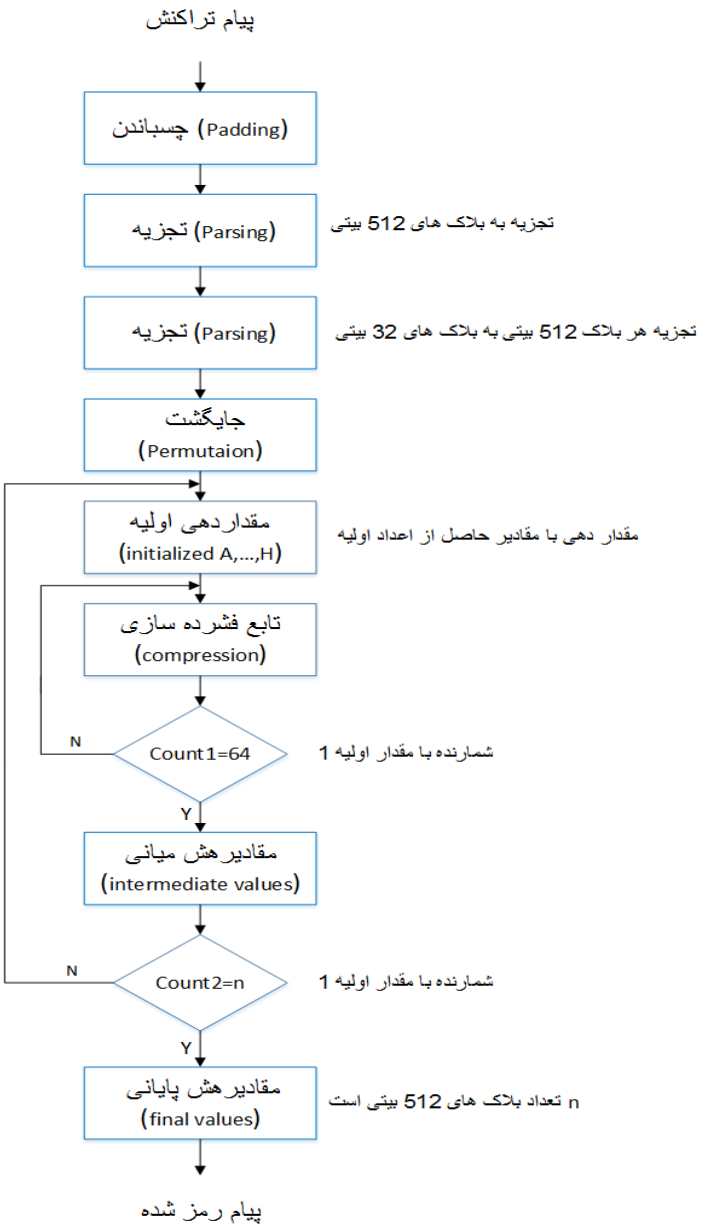
شکل ۱. ساختار بلاک سرآیند

بخش	هدف	زمان بروزرسانی	اندازه (بایت)
Version	شماره نسخه هر بلاک	زمان ارتقاء نرم افزار	۴
hashPervBlock	۲۵۶ بیت hash مربوط به بلاک قبلی	وقتی که بلاک جدید وارد می‌شود	۳۲
hashMerkleRoot	۲۵۶ بیت hash بر اساس همه تراکنش های درون بلاک	وقتی که تراکنش پذیرفته می‌شود	۳۲
Time	زمان حال بر حسب ثانیه از سال ۱۹۷۰	هر چند ثانیه	۴
Difficulty	هدف فعلی[11] در فرمت فشرده	زمان تعیین difficulty	۴
Nonce	عدد ۳۲ بیتی با شروع از مقدار صفر	زمان اجرای الگوریتم	۴

شبه کد مربوط به الگوریتم فشرده‌سازی به صورت زیر است:

[11] Current Target

```
block_header=<version+ prev_block+ merkel_root+ timestamp+ diff+ nonce >
nonce=0
hash=1
target=0x0000000000002816E000000000000000000000000000000000000000000000
While(hash>target)
{
    Hash=SHA256(SHA256(nonce+block_header))
    nonce++
}
```



شکل ۲. مراحل اجرای الگوریتم sudo-SHA-256

تحویل مطابق جدول زمانبندی

اهداف فاز اول:

۱. آشنایی با شبکه بیت‌کوین و سیستم پرداخت
۲. مطالعه الگوریتم استخراج بیت‌کوین
۳. پیاده سازی نرم افزاری الگوریتم رمزنگاری شبکه بیت‌کوین

توجه:

الگوریتم sudo-SHA256 مورد نظر در این پروژه با الگوریتم SHA256 استاندارد متفاوت است.

خروجی‌های مورد انتظار:

۱. پیاده سازی الگوریتم رمزنگاری با استفاده از زبان برنامه‌نویسی C یا C++ بر روی پردازنده Microblaze
۲. گزارش زمان اجرای الگوریتم (برای این بخش میانگین زمان ۵ اجرا را گزارش کنید).
۳. گزارش میزان استفاده از پردازنده (نیاز است تا حتماً مشخصات دقیق پردازنده، فرکانس کاری و نوع سیستم عامل (ویندوز یا لینوکس ۳۲ یا ۶۴ بیتی) مشخص شود).
۴. گزارش حافظه مصرفی.
۵. میزان target را برابر با مقدار زیر در نظر بگیرید. مقدار اولیه متغیر nonce برابر صفر است.
0x00000000000002816E0000000000000000000000000000000000000000000000

کلیات فاز دوم (هنر FPGA)

در این فاز تصمیم داریم تا تمام فرآیند مربوط به استخراج بیت‌کوین را به سخت‌افزار بسپاریم.

اهداف فاز دوم:

۱. پیاده‌سازی سخت‌افزاری الگوریتم استخراج بر روی بورد FPGA.
۲. مقایسه پیاده‌سازی‌های انجام شده تاکنون فازهای سه گانه (از لحاظ زمان اجرا)

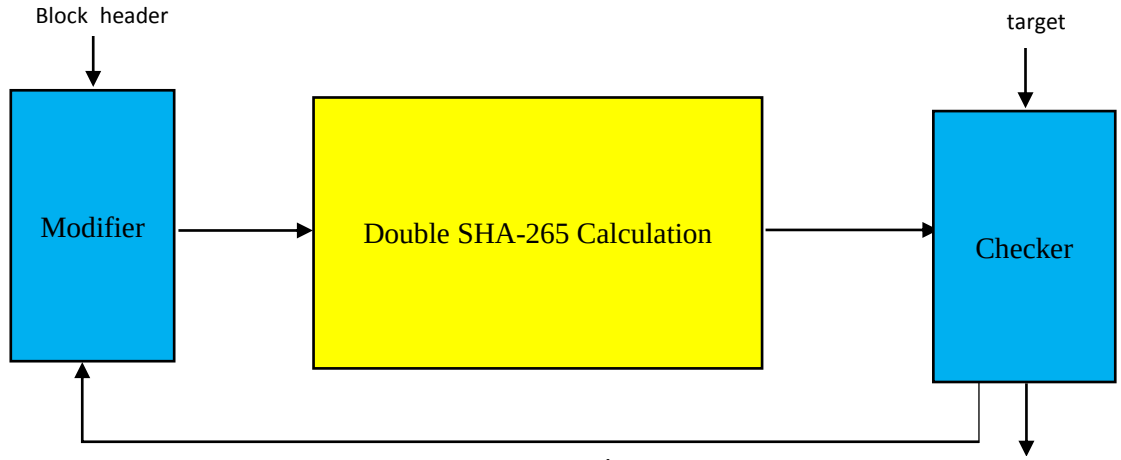
خروجی‌های مورد انتظار:

۱. پیاده سازی الگوریتم استخراج تماماً با استفاده از زبان توصیف سخت‌افزار VHDL.
۲. گزارش زمان اجرای الگوریتم (برای این بخش فرکانس ساعت و تعداد سیکل‌های ساعت گزارش شود).
۳. گزارش منابع مصرفی.
۴. میزان target را برابر با مقدار زیر در نظر بگیرید. مقدار اولیه متغیر nonce برابر صفر است.

0x00000000000002816E0000000000000000000000000000000000000000000000

کلیات فاز سوم (طراحی توأمان)

هنگامی که بیت‌کوین برای اولین بار معرفی شد، واحد پردازش مرکزی (CPU) از اینتل و AMD به عنوان استخراج‌گر استفاده شدند، اما آن‌ها به سرعت توسط واحد پردازش گرافیکی (GPU) از NVIDIA و AMD جایگزین شدند. CPUها تعداد نسبتاً کمی از واحد منطقی محاسباتی (ALU) دارند و برای اجرای نرم‌افزارها و تصمیم‌گیری‌های عمومی طراحی شده‌اند. GPUها توانایی انجام بسیاری از کارهای تکراری را دارند، زیرا تعداد زیادی ALU برای انجام عملیات‌های ریاضی در آن‌ها در نظر گرفته شده است. این ALUهای مشابه را می‌توان بارها و بارها برای اجرای Hashهای مختلف استفاده کرد؛ زیرا تعداد ALUها تأثیر مستقیم بر خروجی Hash دارد. FPGAها می‌توانند پیکربندی شوند تا الگوریتم SUDO-SHA-256 را با بهره‌وری بیشتر اجرا کنند.



شکل ۳. طراحی توأمان الگوریتم استخراج

در این فاز نیاز است تا وظایف مربوط به Checker که شرط حلقه را بررسی می‌کند و بخش مربوط به تعیین Block Header همچنان بر عهده نرم افزار باقی بماند و تنها پیاده‌سازی تابع SUDO-SHA-256 را به سخت‌افزار واگذار کنیم.

اهداف فاز سوم:

۱. پیاده سازی توأمان سخت‌افزاری (بر روی منابع سخت‌افزاری FPGA) و نرم‌افزاری (بر روی پردازنده Microblaze).
۲. گزارش زمان اجرای الگوریتم (برای این بخش میانگین زمان ۵ اجرا را گزارش کنید).
۳. گزارش میزان استفاده از منابع مصرفی FPGA.

توجه:

پیاده‌سازی این بخش **اختیاری** است. نمره اضافه این بخش ۳۵ درصد کل نمره پروژه است.

خروجی‌های مورد انتظار:

۱. پیاده سازی الگوریتم رمزنگاری با استفاده از زبان توصیف سخت‌افزار و بخش‌های Checker و Modifier با استفاده از زبان برنامه‌نویسی C یا C++ بر روی پردازنده Microblaze به صورت طراحی توأمان
۲. گزارش زمان اجرای الگوریتم (برای این بخش میانگین زمان ۵ اجرا را گزارش کنید).
۶. گزارش میزان استفاده از پردازنده (نیاز است تا حتماً مشخصات دقیق پردازنده، فرکانس کاری و نوع سیستم عامل (ویندوز یا لینوکس ۳۲ یا ۶۴ بیتی) مشخص شود).
۳. گزارش حافظه مصرفی.
۴. میزان target را برابر با مقدار زیر در نظر بگیرید. مقدار اولیه متغیر nonce برابر صفر است.

0x0000000000002816E000000000000000000000000000000000000000000000000

تحویل مطابق جدول زمانبندی

شرایط و زمانبندی تحویل پروژه به شرح جدول زیر است:

فاز	تاریخ
فاز اول	۱۳۹۷/۳/۷
فاز دوم	۱۳۹۷/۳/۲۱
فاز سوم	۱۳۹۷/۴/۱۷
تحویل حضوری	۱۳۹۷/۴/۱۷

موارد تحویلی در تمامی فازها:

۱. فایل کامل پروژه شامل تمامی کدهای VHDL
۲. شکل موج‌های شبیه‌سازی
۳. برنامه محک جهت تست طراحی
۴. گزارش سنتز شامل منابع مصرفی، توان مصرفی و فرکانس کاری مدار طراحی شده
۵. انجام پروژه در گروه‌های حداکثر دو نفره مجاز است. انجام پروژه به صورت تک نفره نمره اضافه نخواهد داشت.

نکته: در صورتی که حجم فایل‌های شما بیشتر از مقدار مجاز سایت درس می‌باشد (۲۰ مگابایت)، پروژه خود را در گوگل درایو، دراپ باکس یا وان درایو آپلود کنید و لینک اشتراک آن را در سایت درس بارگذاری کنید. پس از ارسال لینک فایل مذکور را تحت هیچ شرایطی ویرایش نکنید. در غیر این صورت نمره آن فاز صفر در نظر گرفته می‌شود.

۶. زمانبندی ارائه‌های حضوری متعاقباً اعلام خواهد شد. تحویل حضوری فقط در روز اعلام شده خواهد بود و پس از آن به هیچ عنوان پروژه تحویل گرفته نخواهد شد.
۷. در صورت عدم حضور در تحویل حضوری هیچ نمره‌ای به پروژه شما تعلق نخواهد گرفت.

j.talafy@aut.ac.ir
hanie.ghasemy@gmail.com

موفق باشید