Proyecto – Parte 1

Resumen:

Fecha de entrega	Jueves 27 de mayo de 2021 (antes de la media noche)
Valor de la nota	10 %
Entregable	Un reporte en formato IEEE (en pdf), código funte

Lo que debe realizar

A usted se le dara una descripción de un sistema, para dicho sistema usted debe:

1. Describir por medio de un diagrama de bloques una arquitectura para implementar dicho sistema optimizado en area.
2. Describir por medio de un diagrama de bloques una arquitectura para implementar dicho sistema optimizado en desempeño.
3. Crear un modelo de sistema para cualquiera de las dos arquitecturas utilizando el lenguaje de programación de su preferencia (python, c++, perl, go, otro).

Consideraciones:

- Se puede realizar de forma individual o en grupos de dos o 3 personas
- Utilice una sección para cada arquitectura.
- Incluya el diagrama de bloques de cada arquitura.
- Explique porque considera que la arquitectura en específico esta optimizada en area o desempeño.
- Para el modelo de sistema. Incluya comentarios dentro del código fuente, no olvide modularizar su código.
- En el código recordar el tamaño de tanto los operandos como resultados son de un byte.
- Incluya resultados, demostrando que su modelo de sistema funciona.
- No olvide incluir las referencias bibliograficas, si las utilizó.
- El reporte debe ser un pdf utilzando el formato IEEE.
- Debe subir el pdf y el cógico fuente del modelo de sistema a mediación virtual.

Evaluación:

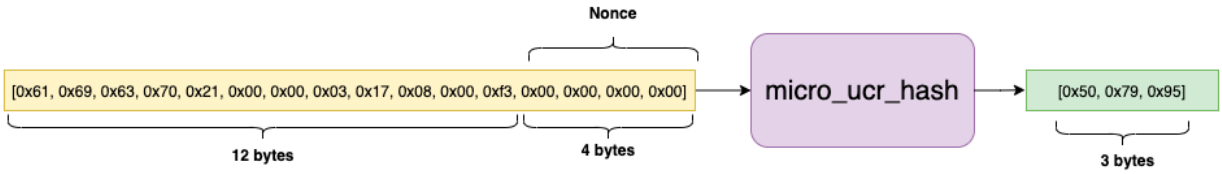
Rubro	**Valor**
Arquitectura optimizada en área	30 %
Arquitectura optimizada en desempeño	30 %
Modelo de sistema	40 %
Total	**100%**

Rubros opcionales (% adicional a la nota de este proyecto):

- Crear un modelo de sistema para la otra arquitectura. 10% adicional
- Explicar cual es el papel de un sistema como el diseñado en el contexto de las criptomonedas, cuales serías los objectivos de optimización más importantes y que problemas están generando las criptomonedas en el ecosistema de hardware. 10% adicional

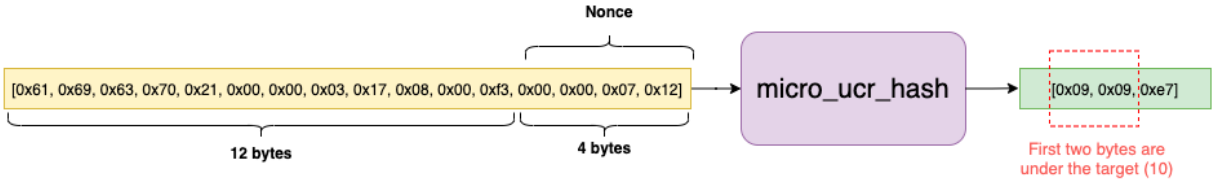
Descripción del sistema

Existe una función ¨micro_ucr_hash¨, que toma como entrada una sequencia de 16 bytes y genera una salida de 3 bytes:

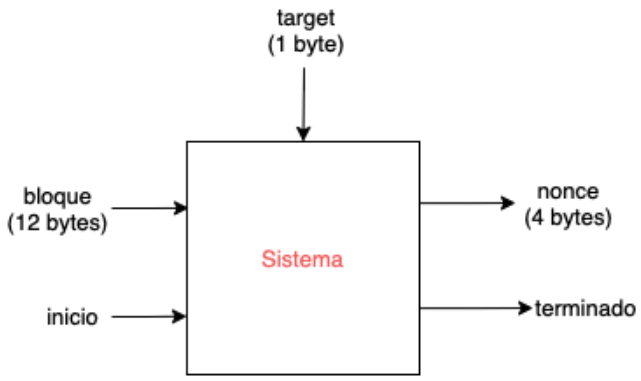


Es importante destacar que no es posible regenerar la entrada a partir de la salida.

El sistema va a recibir un bloque de 12 bits y debe adjuntar alguna combinacion de 4 bytes, que llamaremeos el *Nonce*, que logre generar una salida donde los primeros dos bytes sean menores a un *target*. Por ejemplo si el target es 10, un *Nonce* valido puede ser [0x00, 0x00, 0x01, 0xC1] ya que genera una salida donde los dos primeros bytes son 0x07 y 0x09, menores a 10.



En resumen, el sistema va a recibir un *bloque* de 12 bytes y un *target* de 1 byte, debe empezar a procesar cuando la entrada *inicio* se active. Al final el sistema debe retornar un *bounty* e indicar que el proceso se completo activando la salida *terminado*.



Función micro_ucr_hash (en todas las operaciones estamos trabajando con bytes):

1. Recibe como entrada un bloque de 16 bytes, el orden de los bytes es little endian:

   bloque[0:15] = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]

2. Inicializar 32 variables W (tamaño un byte) donde:

   W[i] = bloque[i] para 0 <= i <= 15

   W[i] = W[i-3] | W [i-9] ^ W[i-14] para 16 <= i <= 31

3. Inicializar tres variables:

   H[0] = 0x01

   H[1] = 0x89

   H[2] = 0xfe

4. Iterar 32 veces haciendo lo siguiente:

   a = H[0]

   b = H[1]

   c = H[2]

   Si 0 <= iteracion <= 16:

   k = 0x99

   x = a ^ b

   Si 17 <= iteracion <= 31:

   k = 0xa1

   x = a | b

   a = b ^ c

   b = c << 4

   c = x + k + W[iteracion]

5. En la ultima iteración:

   H[0] = H[0] + a

   H[1] = H[1] + b

   H[2] = H[2] + c

6. La salida sería la concatenación de los valores de H de la última iteración: [H[0], H[1], H[2]]

Ejemplos de la función hash:

Bloque	**micro_ucr_hash**
0x39 0x7d 0x9f 0x2f 0x40 0xca 0x9e 0x6c 0x6b 0x1f 0x33 0x24 0xfd 0xed 0x87 0x3c	0xf1 0x89 0x73
0xed 0x18 0xbe 0x0f 0x98 0x4a 0xe0 0xe2 0xe3 0x12 0x8e 0xfe 0x0f 0xa2 0x34 0x91	0x7a 0x19 0x6e
0x88 0x55 0xc7 0xac 0x8b 0x73 0xf8 0xf2 0x97 0x01 0xef 0xf1 0xba 0x0f 0x98 0xb3	0x97 0xe9 0x57
0x5b 0x71 0xfd 0x32 0xbd 0x79 0xb8 0x72 0xdb 0xe6 0x1c 0xf7 0xc0 0x09 0x65 0x18	0xce 0x59 0x73
0xd0 0xe1 0xaa 0xb6 0xae 0x1e 0xa2 0xd1 0x11 0x5d 0xd7 0x16 0x11 0x9f 0x29 0x2c	0x55 0xc9 0x91
0xeb 0xad 0x50 0x90 0x38 0x43 0xf9 0xc9 0xaa 0xad 0x6f 0x64 0xdf 0xd3 0x6f 0xa3	0xf0 0x79 0x95
0x87 0x3f 0x33 0xfa 0x4a 0x96 0xd3 0x41 0xa4 0xa1 0x6e 0xa4 0x95 0x91 0xa7 0xec	0x22 0x99 0x9b
0x26 0x6e 0x30 0xf5 0xe8 0x32 0xd7 0x54 0x76 0x5e 0xb5 0x81 0x70 0x43 0xce 0x8b	0x47 0xe9 0xf3
0xfe 0x35 0x36 0xdf 0x50 0x49 0x85 0x45 0x24 0x02 0x12 0xde 0xc6 0x09 0x59 0x2a	0xf2 0x99 0xbe
0x73 0xa9 0xde 0xbe 0x6a 0xf9 0xc9 0x5e 0x9f 0x05 0x2d 0x59 0x53 0xdb 0xd6 0x5c	0xbe 0x59 0xc0