(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 539 004**

(51) Int. CI.:

H04L 9/00 (2006.01)
H04L 9/06 (2006.01)
H04L 9/26 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(96) Fecha de presentación y número de la solicitud europea: **12.04.2012 E 12714311 (3)**

(97) Fecha y número de publicación de la concesión europea: **07.01.2015 EP 2700188**

(54) Título: **Método y aparato para la configuración IV no lineal en generadores de secuencia de clave**

(30) Prioridad:

21.04.2011 EP 11305482

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
25.06.2015

(73) Titular/es:

GEMALTO SA (100.0%)
6, rue de la Verrerie
92190 Meudon, FR

(72) Inventor/es:

GOUGET, ALINE y
PAILLIER, PASCAL

(74) Agente/Representante:

ISERN CUYAS, María Luisa

DESCRIPCIÓN

Método y aparato para la configuración IV no lineal en generadores de secuencia de clave.

La invención se refiere a un método y un aparato para la configuración IV no lineal en los generadores de corriente de clave.

Más precisamente, la invención describe la implementación de software eficiente de un cifrado de flujo orientado al hardware.

La invención se refiere a la tecnología MIFARE sin contacto utilizada dentro de las redes de transporte público para aplicaciones de gestión de venta de entradas y acceso.

En la siguiente descripción, se utilizarán algunas funciones lineales. Las funciones lineales se pueden implementar utilizando tablas y las tablas definen fácilmente las correspondiente funciones lineales. "Función lineal" y "tablas" se utilizarán de igual manera.

El principal bloque de construcción criptográfica de MIFARE es un cifrado de flujo llamado CRYPTO-1. La estructura general de CRYPTO-1 se puede esbozar tal como sigue:

1) Un registro de desplazamiento de estado de 48 bits indicado (en el tiempo t>0) por STt = (st, ..., st+47), con si € {0,1}, i≥0.

2) Una función configuración de clave

3) Una función de configuración de IV

4) Una función de generación de flujo de claves

La parte más crítica del cifrado de flujo CRYPTO-1 con respecto a las actuaciones de implementación de software es la función de configuración de IV, la cual es una función no lineal. Para la fase de configuración de IV, los sucesivos bits de realimentación no lineales utilizados para actualizar el registro de desplazamiento se calculan, para cada uno, como una función no lineal del estado anterior incluyendo los bits de realimentación anteriores.

Dejemos que STt0 = (st0, ..., st0+47) sea el estado del cifrado antes de realizar la función de configuración de IV (suponemos que la función de configuración de clave ha sido ya realizada). El estado STt0 será actualizado mediante el uso de una función boolena no lineal nlf y una IV de 32 bits = (IV0, ..., IV31) inyectada bit a bit en el estado de la siguiente manera:

PARA i = 0 a 31 HACER

$$st0+48+i = nfl\ (st0+i,..., st0+47+i, IVi)$$

$$STt0+i+1 = (st0+i+1,..., st0+i+48).$$

Después de la inyección de un IV de 32 bits en STt0, el nuevo estado del cifrado es STt0+32 = (st0+32, ..., st0+79). Ahora, el cifrado está listo para generar un flujo de clave, por ejemplo para completar el protocolo de autenticación Mifare.

La función nfl "efectivamente" depende sólo de 28 variables (en lugar de 49). Sin pérdida de generalidad, cualquier función F booleana n-variable puede ser representada utilizando el Formulario Normal Algebraico por:

$$F(x1,..., xn) = G(x1,..., xn)\ xor\ L(x1,..., xn),$$

donde L es una función booleana lineal y G(x1,..., xn) es una función no lineal. La función nlf cae dentro de una clase particular de funciones booleanas, es decir, funciones booleanas del tipo:

G(x1,..., xn) = GU (x1,..., xn-k) x ó GH (xn-k+1,..., xn).GV(x1,..., xn-k) donde k es un entero pequeño en comparación con n.

La función nlf está "orientada al bit". Una implementación directa de nlf hace uso de un gran número de selecciones de bits (básicamente, se puede implementar utilizando un desplazamiento derecha/izquierda y Byte-Y ó Byte-O). Los valores de bits se combinan a continuación utilizando bit-XO y bit-Y.

Una solución clásica para convertir una implementación orientada a bits en un implementación orientada a bytes es reemplazar selecciones de bits por tablas de consulta y operaciones orientadas a de bits por operaciones orientadas

a bytes. Sin embargo, esta solución clásica no se puede aplicar directamente cuando el valor de bit sj se calcula utilizando una función no lineal que no depende linealmente de sj-1 y/o sj-2 y/o ... y/o sj-8.

K. Mayes et al., "The MIFARE Classic storry" proporcina una visión global de la tarjeta MIFARE® Clásica.

w. Teepe, "Making the Best of Mifare Classic" describe contramedidas contra la recuperación y clonado de estado de MIFARE® Clásica.

La invención describe un método para actualizar un estado inyectando una IV utilizando un registro de desplazamiento de retroalimentación no lineal que únicamente hace uso de mirar arriba tablas y operaciones básicas en palabras de 8 bits.

El tamaño del estado interno del cifrado es N = 8*n. Un valor típico para n es n = 6 ó n = 32. En el tiempo t, el estado interno del cifrado está representado por STt = (STt[0], ..., STt[n-1]), o equivalentemente por STt = (st, ..., st+8n-1).

Suponemos que una función de configuración de clave se ha realizado en el estado del cifrado (por ejemplo, la clave se ha cargado en el estado) antes de comenzar la fase de configuración de IV. No se hace ninguna suposición sobre la función de configuración de clave.

Fase de configuración de IV:

Dejemos que STt0 = (st0, ...,st0+8n-1) sea el estado inicial del cifrado para la fase de configuración de IV, donde t0> 0 es un valor predeterminado. Durante esta fase, el estado interno del cifrado se actualiza usando un Valor de Inicialización (IV) de longitud 8*m bits, m>0 (un valor típico para m es m = 4, m = 8, m = 12 ó m = 16). Denotamos los bytes IV por: IV = (IV[0],..., IV[m-1]).

El estado final de la fase de configuración de IV es:

$$STt0+8m = (st0+8m,\ldots,st0+8(m+n)-1)$$

Debido a que la fase de configuración de IV no resultaría ser "fácil" de invertir, es decir, conocer el estado final de la fase de configuración de IV, no debería ser fácil recuperar el estado inicial de la fase de configuración de IV, la función utilizada para actualizar el estado durante el fase de instalación de IV es generalmente elegido que no sea lineal.

Actualización No Lineal del Estado (NLF).

Desde el estado inicial STt0, los estados sucesivos m STt0+8, STt0+16,..., STt0+8m se calculan mediante la inyección byte por byte de la IV en el estado a través de una función de actualización no lineal denominada 1-byteNLF.

En el tiempo t = t0, T+1=t0+8, ..., T+m-1 =t0+8*(m-1), el estado de cifrado STT+i, 0≤i≤m-1, es actualizado utilizando IV[i] mediante el cálculo de un valor-byte que es STT+i+1[n-1]. De hecho, asumiendo que en el tiempo T+i, el estado es:

$$STT+i = (STT+i[0], STT+i[1],\ldots, STT+i[n-1])$$

Entonces, el estado en el tiempo T+i+1 es:

$$STT+i+1 = (STT+i[1], STT+i[2]\ldots, STT+i[n-1], STT+i+1[n-1])$$

$$= (STT+i+1[0], STT+i+1[1]\ldots, STT+i+1[n-2], STT+i+1[n-1])$$

En otras palabras, STT+i+1[j]= STT+i[j+1], para j=0, ..., n-2, y sólo un valor de un-byte (es decir, STT+i+1[n-1]) necesita ser calculado. Calculamos este valor de byte mediante una función no lineal denominada 1-byteNLF.

La función 1-byteNLF. Esta función hace uso de:

1) La tablas de consulta n de tamaño (8 bits)*(8 bits) indicadas por LF0, LF1, ..., LFn-1 (posiblemente, existe (i,j), 0≤i<j<n tales que LFi = LFJ, y/o existe i, 0≤i<n, tal que LFi = Ø).

2) Las tablas de consulta 2n de tamaño (8 bits)*(4 bits) indicadas por BFI0, BFr0, BF11, BFr1, ..., BFIn-1, BFrn-1 (posiblemente, existe (i,j), 0≤i<j<n tal que BFxi = BFyj donde x, y€{r, l}, y/o existe i, 0≤i<n, tal que BFxi = Ø donde x €{r, l}). En implementaciones particulares es posible utilizar tablas n de tamaño (8 bits)*(8 bits). Los

tamaños de tablas anteriores hablan de tamaños útiles de tablas. Es posible almacenar una tabla de tamaño útil de (8 bits)*(4 bits), dentro de una tabla de tamaño real de (8 bits)*(8 bits).

3) Dos funciones de filtrado:

- Ga: {0,1}^{8*k} → {0,1}^8;
- Gb: {0,1}^{8*k} → {0,1}^8;

con k tal que 0< k ≤ n -1. Dependiendo de la implementación, los valores de entrada de Ga y Gb son por ejemplo k bytes ejemplo o palabras k/4 de 32 bits.

4) Una tabla de consulta 10 de tamaño (8 bits)*(8 bits)

El cálculo de STT+i+1[n-1] = (st+8(n+i-1), ..., st+8(n+i) -1) incluye los siguientes pasos:

1) Cálculo de los valores intermedios de 1-byte:

(S[T+i], para alguna i, 0 ≤ i ≤ n-1) representa una selección de 8 bits a partir de dos bytes de S.

Xi = BFli [S[T+i]], para alguna i, 0 ≤ i ≤ n-1

Yi = BFri [S[T+i]], para alguna i, 0 ≤ i ≤ n-1

2) Cálculo de dos valores de 1-byte U y V:

- U se calcula utilizando la función Ga, aplicada a 4 bytes. Cada uno de estos bytes se ha calculado utilizando un valor Xi, y un valor Yi. Esto también puede ser descrito de la siguiente manera:

U = Ga (Xi1, ..., Xik, Yi1, ..., Yik'), donde 1 ≤ k,k' ≤ 8*n

- V se calcula utilizando la función Gb, aplicada a 4 bytes. Cada uno de estos bytes se ha calculado utilizando un valor Xi, y un valor Yi. Esto también puede ser descrito de la siguiente manera:

V = Gb (Xi1, ..., Xik, Yi1, ..., Yik'), donde 1 ≤ k,k' ≤ 8*n

3) El cálculo de un valor temporal:

STT+i+1 [n-1] = XORj = 0, ..., n-1 (LFi (STT+i[j]))

Ventajosamente, se utiliza un paso adicional con el fin de actualizar el valor temporal STT+i+1[n-1] mediante la adición de, utilizando la operación XOR, una función lineal de un byte de valor de inicialización IV[i]. Esto también puede ser descrito de la siguiente manera

STT+i+1[n-1] = STT+i+1[n-1] XOR I0 [IV[i]]

4) El valor actual de STT+i+1[n-1] es próximamente actualizado bit por bit, de la siguiente manera. Para j= 0 a 7, calcular:

Un valor de byte temporal e calculado a partir de un valor BFxj[STT+i+1[n-1]], donde x€{r,l} y, para algunos j, 0≤ j ≤ n-1

Un valor de byte temporal t = (U Y e) XOR V

STT+i+1 [n-1] = STT+i+1 [n-1] XOR seleccionar (j, t),

donde:

- seleccionar (j,t) representa la selección de bits en el paso j que son los bits ij1, ..., ijw, 0 ≤ ijr ≤ 7 con 0 ≤ r ≤ 7, del valor de byte t.

Más precisamente, la invención es un método para llevar a cabo la "etapa de configuración del desafío lector en la fase de autenticación", de acuerdo con el protocolo de autenticación y encriptado MIFARE, en el "dispositivo de etiqueta", en donde, el tamaño del estado interno del cifrado, llamado "ST" es de 6 bytes, en el tiempo t, el estado interno del cifrado está representado por ST_t = (ST_t[0], ..., ST_t[5]), el desafío lector, llamado "RC" previamente indicado por IV, de longitud de 4 bytes, se denota por: RC = (RC[0], ..., RC[3]), una función lineal "LF" que realiza la

función de retroalimentación lineal del registro de desplazamiento de retroalimentación lineal Mifare, y una función no lineal, "NLF", realizando el generador de filtro de dos capas Mifare, se utilizan para actualizar el valor de dicho ST usando dicho RC.

Dicho procedimiento comprende al menos los siguientes pasos:

- Disposición de bits "ST" en el byte para almacenar, en un nibble los bits con un ranking par, y en el otro nibble los bits con un ranking impar.
- creación de cuatro tablas f1i, f1p, f2i, f2p
- desde el estado inicial de dicha etapa de configuración del desafío lector indicado por ST_{t0}, los sucesivos estados ST_{t0+8}, ST_{t0+16}, ST_{t0+24}, ST_{t0+32} se computan calculando byte por byte el valor de realimentación lineal utilizando la citada función LF
- inyectando byte por byte dicho "RC" en el citado ST utilizando la mencionada función NLF

Este método puede comprender una etapa preliminar de configuración de bits "ST" en bytes para almacenar, en un nibble los bits con un ranking par, y en el otro nibble los bits con un rango impar.

La función de realimentación lineal se puede definir usando 6 tablas pre-calculadas de 8-bits x 8-bits L0, ..., L5 tal que:

$$LF\ (ST_{8t+i}\ [0], ..., ST_{8t+i}\ [5]) = L_5\ [ST_{8t+i}\ [5]]\ \emptyset L_4\ [ST_{8t+i}\ [4]]\ \emptyset\ ...\ \emptyset\ L_0\ [ST_{8t+i}\ [0]]$$

La función "NLF" se puede definir por:

- Un registro de tamaño de 8 bits, llamado "Buff", que se ha inicializado o actualizado utilizando la función lineal LF aplicada en ST y la tabla I0 se calcula para un byte de RC
- Cuatro tablas pre-calculadas de 8 bits x 8 bits f1i, f1p, f2i, f2p aplicadas en el estado ST
- una función "Ga", aplicada en cuatro valores de bytes "a", "b", "c" y "d" y que produce un valor de un byte, tal que:

$$Ga\ (a,b,c,d) = (((a\ \hat{}\ d)\ \&\ (a\ \hat{}\ b\ \hat{}\ (b\ \&\ c)))\ \hat{}\ 0xFF\hat{}\ d)$$

- una función "Gb", aplicada en cuatro valores de bytes "a", "b", "c" y "d" y que produce un valor de un byte, tal como:

$$Gb\ (a, b, c, d) = ((a\ \hat{}\ (b\ \&\ d))\ \&\ (c\ \hat{}\ a)\ \hat{}\ (a\ \&\ d))$$

- una función "G" que utiliza valores de dos bytes, indicados por U y V, que son las salidas de dichas funciones Ga, Gb, donde los bytes de entrada de Ga y Gb se calculan utilizando bits de ST y de las cuatro mesas f1i, f1p, f2i, f2p. Valores de dos bytes, indicados por UU y VV se inicializan con bits de dicho ST y, posiblemente, con bits de dicho registro Buff. Los valores de UU y VV son actualizados utilizando 8 pasos antes de la última actualización de dicho registro Buff utilizando los valores UU y VV cuando se calculan todos los bits de dicho registro Buff, el primer byte de dicho ST es borrado, todos los demás bytes son desplazados a la izquierda, y el registro Buff se inserta como sexto rango.

Otras características y ventajas de la presente invención se extraerán más claramente de la lectura de la siguiente descripción de un número de realizaciones preferidas de la invención con referencia a los correspondientes dibujos que la acompañan, en los que:

- La Figura 1 representa la operación tradicional realizada según el protocolo MIFARE
- La Figura 2 representa la implementación de la invención que nos ocupa

La Figura 1 representa el diseño de hardware del cifrado Cryptol del chip Classic Mifare. El núcleo del cifrado Cryptol es un registro de desplazamiento de retroalimentación lineal de 48 bits (11). En cada tic-tac del reloj, el bit de retroalimentación se calcula como una combinación lineal utilizando la operación XOR de los bits del estado actual, el registro se desplaza un bit a la izquierda de tal manera que se desecha el bit más a la izquierda y el bit más a la derecha es el bit de retroalimentación. El estado LFSR se filtra por una función booleana no lineal que está compuesta por dos capas de circuitos. La primera capa (12,13,14,15,16) hace uso de dos circuitos diferentes

correspondientes a dos diferentes funciones boolenas 4-variable f1 y f2. El circuito de la función booleana f1 se utiliza dos veces (12 y 15) y la función booleana f2 se utiliza en los otros tres lugares de la primera capa (13, 14 y 16). En cada tic-tac del reloj, los 5 bits de salida de la primera capa son los 5 bits de entrada de la segunda capa. La segunda capa (17) está compuesta por un circuito correspondiente a una función booleana 5-variable (17) y que produce un solo bit de salida (18).

La Figura 2 representa la implementación de software de la presente invención. El núcleo del cifrado Cryptol está representado por 6 bytes (21) correspondientes al estado de cifrado en algún tiempo T. El byte de retroalimentación lineal se calcula utilizando las tablas 6 (1 byte)*(1 byte) indicadas como L_0, L_1, L_2 , L_3, L_4 y L_5. La presente invención se dirige a la inserción del desafío lector (28) enviado durante un protocolo de autenticación. En el tiempo T + 1, el estado de cifrado se obtendrá descartando el byte más a la izquierda de (21) y añadiendo a la derecha el nuevo byte representado por (24).

Para cada byte del Desafío Lector (RC), la presente invención calcula el valor del nuevo byte (24). Se inicializa primero con la operación byte-XOR entre el byte de retroalimentación lineal y el byte (28) que es una función lineal del byte del desafío lector calculada utilizando una tabla (1 byte)*(1 byte). A continuación, la invención calcula dos valores de bytes U y V usando dos funciones G1 (21) y G2 (26). El valor del byte (24) se actualiza utilizando 8 pasos. Para cada uno de los 8 pasos, se calcula una función booleana 4-variable f2 de tal manera que los bits de entrada se encuentran en el byte más a la derecha del estado actual (21) y/o del byte del siguiente de estado (28). Para el primer paso, las conexiones se encuentran en el estado actual tal como se representa por (23). Para el último paso, las conexiones de función booleana 4-variable se encuentran en el byte del nuevo estado tal como se representa por (24). El cálculo de f2 se realiza utilizando la tabla f2i o f2p.

El bit de salida de (23) se utiliza para actualizar UU y/o VV. Finalmente, UU y VV se utilizan para la actualización final de (24).

REIVINDICACIONES

1. Método para realizar una "etapa de configuración de desafío lector en una fase de autenticación", de acuerdo con un protocolo de autenticación y cifrado MIFARE© utilizando cifrado CRYPTO-1, en un "dispositivo de etiqueta", en donde, el tamaño de un estado interno de cifrado, llamado "ST" es de 6 bytes , en el tiempo t, el estado interno de cifrado está representado por $ST_t = (ST_t[0],..., ST_t[5])$, un desafío lector (28), llamado "RC", de longitud de 4 bytes, se indica por: RC = (RC[0], ..., RC[3]) una función lineal "LF" que realiza una función de retroalimentación lineal de un registro de desplazamiento de retroalimentación lineal cifrado CRYPTO-1 Mifare©, y una función no lineal, "NLF", realizando un generador de filtro de dos capas cifrado CRYPTO-1 Mifare© utilizando dos funciones de filtrado Ga y Gb, para actualizar el valor de dicho ST usando dicho RC, **caracterizado porque** dicho método comprende al menos las siguientes etapas:

- creación de cuatro tablas f1i, f1p, f2i, f2p

- desde un estado inicial de dicha etapa de configuración del desafío lector indicado como ST_{t0}, los sucesivos estados ST_{t0+8}, ST_{t0+16}, ST_{t0+24}, ST_{t0+32} se calculan mediante el cálculo byte por byte de un valor de retroalimentación lineal utilizando dicha función LF

- insertando byte por byte dicha "RC" en la mencionada ST utilizando dicha función NLF, calculándose los bytes de entrada de Ga y Gb utilizando bits de ST y las cuatro tablas f1i, f1p, f2i, f2p.

2. Método según la reivindicación 1, **caracterizado porque** dicho método comprende una etapa preliminar de disposición de bits "ST" en un byte para almacenar, en un nibble los bits con rango par, y en el otro nibble los bits con rango impar.

3. Método según la reivindicación 1 o la reivindicación 2, **caracterizado porque** dicha función de retroalimentación lineal se define utilizando 6 tablas pre-calculadas de 8 bits x 8 bits L_0, ..., L_5 tal que:

$$LF (ST_{8t+i} [0], ..., ST_{8t+i} [5]) = L_5 [ST_{8t+i} [5]] \varnothing L_4 [ST_{8t+i} [4]] \varnothing ... \varnothing L_0 [ST_{8t+i} [0]]$$

4. Método según la reivindicación 3, **caracterizado porque**, dicha función "NLF" se define por:

- Un registro de tamaño de 8 bits, llamado "Buff", que se ha inicializado o actualizado utilizando la función lineal LF aplicada en ST y la tabla I0 se calcula para un byte de RC

- Cuatro tablas pre-calculadas de 8 bits x 8 bits f1i, f1p, f2i, f2p aplicadas en el estado ST

- una función "Ga", aplicada en cuatro valores de bytes "a", "b", "c" y "d" y que produce un valor de un byte, tal que:

$$Ga (a,b,c,d) = (((a \wedge d) \& (a \wedge b \wedge (b \& c))) \wedge 0xFF \wedge d)$$

- una función "Gb", aplicada en cuatro valores de bytes "a", "b", "c" y "d" y que produce un valor de un byte, tal como:

$$Gb (a, b, c, d) = ((a \wedge (b \& d)) \& (c \wedge a) \wedge (a \& d))$$

- una función "G" que recibe las salidas de dichas funciones Ga, Gb y valores de dos bytes, indicados por U y V que se inicializan con bits de dicho ST y, valores de dos bytes, indicados por UU y VV que se inicializan con bits de dicho ST y posiblemente, con bits de dicho registro Buff, dichos valores UU y VV se actualizan utilizando 8 pasos antes de la actualización final de dicho registro Buff y el citado registro Buff.

- cuando todos los bits de dicho registro Buff se han calculado, el primer byte de dicho ST es borrado, todos los demás bytes se desplazan a la izquierda, y el registro Buff se inserta como sexto rango.

5. Método según la reivindicación 4, **caracterizado porque**, dicho registro de "Buff" contiene el byte actual de dicho RC para calcular, mediante el modo XOR con el byte proporcionado por la citada función lineal "LF".

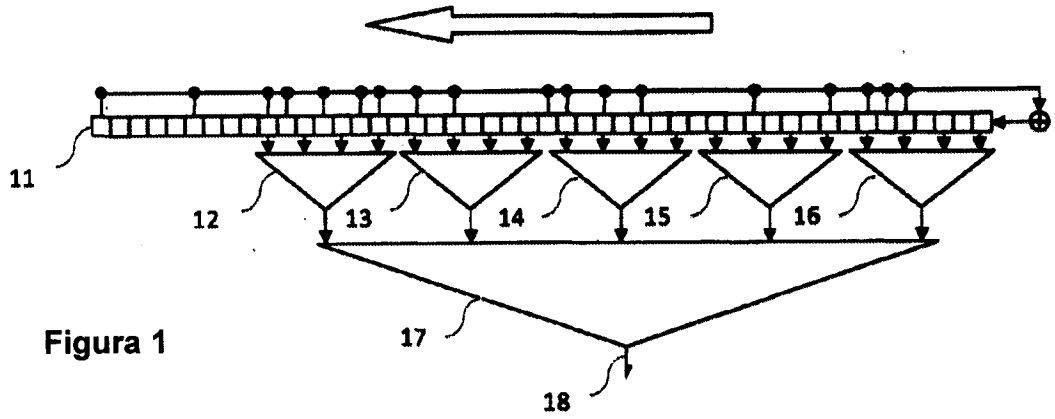

Figura 1

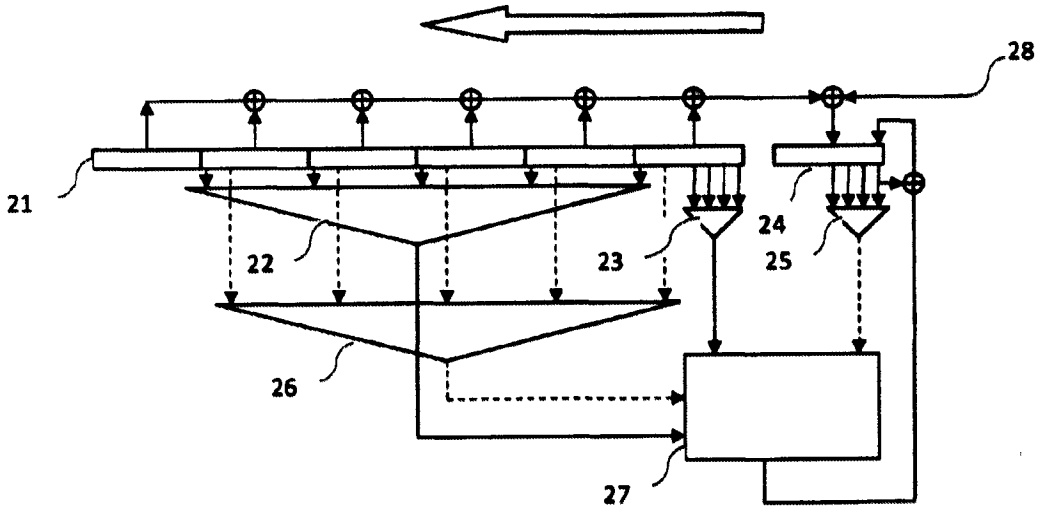

Figura 2