Draft for the UK PATENT APPLICATION

A Leak-Resistant Randomised Exponentiation Method for Cryptographic Boxes

Abstract.

This disclosure describes a means of designing exponentiation-based cryptographic devices so that secret internal keys cannot easily be deduced from the power used by the box, or from other externally measurable quantities such as timing or electro-magnetic radiation, which fluctuate during computations.

A new exponentiation algorithm, MIST, is described. It can be applied wherever exponentiation is used in cryptographic systems, whether hardware, software or a mixture of the two. It defeats current art in side channel attacks by avoiding pre-computations and providing different computation schemes for every exponentiation. Moreover, it requires only very limited extra resources. The method uses a random number generator to randomly vary operations, argument locations and other aspects. The general invention is supported by an exemplar apparatus which could be used in a typical smart card, rechargeable telephone card, conditional access module for set-top box, or other similar embedded VLSI chip.

Inventor: **Walter; Colin D.** (Manchester, UK)

Assignee: **Comodo Research Lab Ltd, 10 Hey St, Bradford, BD7 1DQ, UK**

Appl. No.: **0126317.7**

Filed: **2 November 2001**

Current U.S. Class: **380/30**; 380/1; 713/174; 713/194

Intern'l Class: H04L 009/30; H04L 009/00

Field of Search: 380/30,59,28,1

Attorney, Agent or Firm: **Appleyard Lees, 111 Piccadilly, Manchester M1 2HY, UK**

References Cited

U.S. Patent Documents

US 5991415: Method and apparatus for protecting public key schemes from timing and fault attacks.

Foreign Patent Documents

WO 99/35782: Leak-Resistant Cryptographic Method and Apparatus

Other References

K. Gandolfi, C. Mourtel & F. Olivier, "Electromagnetic Analysis: Concrete Results", Proceedings of *Workshop on Cryptographic Hardware and Embedded Systems*, Paris, May 14-16, 2001, Çetin Koç, David Naccache and Christof Paar (Editors), pp. 255-265.

Daniel M. Gordon, "A Survey of Fast Exponentiation Methods", Journal of Algorithms, volume 27 (1998), pp 129-146.

D. E. Knuth, The Art of Computer Programming, vol. 2, "Seminumerical Algorithms", 2nd Edition, Addison-Wesley, 1981, pp. 441-466.

P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems", *Advances in Cryptology, Proc Crypto 96*, Lecture Notes in Computer Science **1109**, N. Koblitz editor, Springer-Verlag, 1996, pp 104-113.

P. Kocher, J. Jaffe & B. Jun, "Differential Power Analysis", Advances in Cryptology – Crypto '99, Lecture Notes in Computer Science **1666**, M. Wiener (editor), Springer-Verlag, 1999, pp 388-397.

D. May, H.L. Muller & N.P. Smart, "Random Register Renaming to Foil DPA", Proceedings of *Workshop on Cryptographic Hardware and Embedded Systems*, Paris, May 14-16, 2001, Çetin Koç, David Naccache and Christof Paar (Editors), pp. 29-39.

T. S. Messerges, E. A. Dabbish, R. H. Sloan, "Power Analysis Attacks of Modular Exponentiation in Smartcards", Cryptographic Hardware and Embedded Systems (Proc CHES 99), C. Paar & Ç. Koç editors, Lecture Notes in Computer Science **1717**, Springer-Verlag, 1999, pp. 144-157.

E. Oswald & M. Aigner, "Randomized Addition-Subtraction Chains as a Countermeasure against Power Attacks", Proceedings of *Workshop on Cryptographic Hardware and Embedded Systems*, Paris, May 14-16, 2001, Çetin Koç, David Naccache and Christof Paar (Editors), pp. 40-52.

R. L. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Comm. ACM*, vol. **21**, 1978, pp. 120-126.

C. D. Walter, "Exponentiation using Division Chains", *IEEE Transactions on Computers*, vol. **47**, No. 7, July 1998, pp. 757-765.

C. D. Walter & S. Thompson, "Distinguishing Exponent Digits by Observing Modular Subtractions", Topics in Cryptology – CT-RSA 2001, D. Naccache (editor), Lecture Notes in Computer Science **2020**, Springer-Verlag, 2001, pp. 192-207.

C. D. Walter, "Sliding Windows succumbs to Big Mac Attack", Proceedings of *Workshop on Cryptographic Hardware and Embedded Systems*, Paris, May 14-16, 2001, Çetin Koç, David Naccache and Christof Paar (Editors), pp. 291-304.

Claims

What is being claimed is:

1. A modified, iterative version of implementing exponentiation using Walter's divisor chain method where:

   the divisor set is selected to be {2,3,5}; and

   each least non-negative residue is allowable for each residue; and

   one minimal length addition chain is associated with each divisor/residue pair; and

   a single addition chain is generated; and

   the residue powers are multiplied together sequentially.

2. In the method of claim 1, the improvement in which divisors are selected from the divisor set in a way unpredictable by an attacker so that the corresponding addition chain defines a randomised computation scheme for the exponentiation.
3. In the method of claim 2, the further improvement in which a different addition chain is generated for every one or more successive exponentiations.
4. In the method of claim 3, the improvement in which the locations for storing all computed products of the addition chain are selected in a way unpredictable by an attacker.
5. In the method of claim 4, the further improvement in which the exponent, modulus and message are first modified using the method of Shamir [US Patent 5991415].
6. All similar methods to those in claims 1, 2, 3, 4 and 5 in which a different divisor set is chosen in claim 1.
7. Similar methods to claim 6 in which any number of minimal length addition chains for each divisor are made available for the exponentiation scheme and, when the divisor is chosen, an appropriate addition chain is selected for it either randomly or predictably.
8. Similar methods to claim 7 in which addition chains of any length are made available for each divisor instead of only ones with minimal length.
9. In the method of claim 8, the improvement in which the decreasing value of the exponent is represented using a radix which is a multiple of each and every divisor.
10. In the apparatus of claim 2, the selection of a divisor which exactly divides the exponent if this is possible.
11. In the apparatus of claim 2, making no deterministic choices of the divisor during construction of the addition chain.
12. In the apparatus of claim 2, the further improvement under which the process for selecting a divisor is modified regularly and dynamically to an unpredictable state.
13. In the apparatus of claim 1, the further improvement in which the residues associated with each divisor are unrestricted.
14. In the apparatus of claim 2 and those of subsequent claims, the further improvement under which the process for selecting a divisor is varied at unpredictable intervals.

15. The further improvement that in the case of CRT exponentiation the apparatus of the previous claims be applied to both exponentiations.
16. The further improvement to the apparatus of claim 3 in which any or all of the subsequent claims are included.
17. Means for selecting the divisors in claim 2.
18. Means for selecting the locations in claim 4.
19. Means for selecting the addition chains in claim 7.
20. Means for selecting the divisor in claim 10.
21. Means for selecting the divisor in claim 11.
22. Means for selecting the divisor in claim 12.
23. Means for varying the decision process in claim 14.
24. Means for representing the exponent and performing division on it efficiently.
25. The further enhancement that the residue powers, whose product forms the required output, are distributed over several registers, which contain products of same, rather than being multiplied immediately into a single register.

Description

FIELD OF INVENTION

The present invention relates to novel techniques, methods and apparatus, for making number -theoretic public-key schemes (including encryption schemes, signature schemes, authentication schemes, key exchange schemes, key management schemes etc. using modular integer arithmetic, finite field arithmetic, elliptic curve groups, etc.) more resistant to attacks involving side channel leakage of data.

BACKGROUND OF INVENTION

1. Introduction.

Cryptographic systems are widely used in a variety of circumstances. Prominent amongst these in the public sector are the electronic transfer of cash between banks, the storage of unspent units in a telephone smartcard, the protection of private keys which allow access to TV channels via a set-top box, and the exchange of confidential material between different branches of an organisation via a public network. Some of these are required to work in a hostile environment where the data owner is not physically present to protect the information and a potential attacker has unrestricted access to the cryptographic device. Such devices may incorporate a variety of means to protect the data contained within them besides encryption, including physical shielding to reduce electro-magnetic radiation, capacitors to reduce power variation, random noise generators to obscure internal operations, limited life span before secret data is overwritten, and self-destruct mechanisms when tampering is detected.

Many crypto-systems, signature systems and authentication systems employ exponentiation in some multiplicative group as part or all of the encryption, decryption, signing or verification processes. This includes, but is not limited to, the use of RSA in modular arithmetic and over elliptic curves (*see* Rivest, Shamir and Adleman), to Diffie-Hellman key exchange, to El Gamal encryption and to the Digital Signature Standard. Generally, the value of the exponent must be kept secret. However, if the same sequence of multiplications and squarings is used on every occasion to perform the exponentiation, then an attacker can average any information leaking from the device over a number of encryptions/decryptions in order to increase the signal to noise ratio. He may be able to do this sufficiently well to reduce the number of possible exponents to the point at which it is computationally feasible to deduce the secret exponent.

In particular, recent work has exposed fundamental weaknesses in externally powered embedded cryptographic devices. Methods called “simple power analysis” and “differential power analysis” sometimes enable secret keys to be obtained from such systems by measuring minute variations in execution time, current consumption or electro-magnetic radiation. These are described, for example, by Kocher in the proceedings of *Crypto 96*, by Kocher, Jaffe and Jun in the proceedings of *Crypto 99*, and in work at Gemplus by Gandolfi, Mourtel and Olivier.

Details of how this leakage can be used to recover secret keys has been further developed by a number of authors such as Messerges, Dabbish and Sloan. In particular, work by Walter & Thompson has shown that very few exponentiations are required to determine the secret

exponent if the same computational scheme is used repeatedly and appropriate monitoring equipment is available. Moreover, an article by Walter at *CHES 2001* shows that it may be possible to recover the secret exponent from a single exponentiation if a scheme is used in which the same multiplicands are used repeatedly.

2. Prior Art.

These methods of attack have led to some work on *secure implementations* of the algorithms involved. Much of this involves modification of the inputs to the exponentiation. For example, Shamir's patent US 5,991,415 describes how the exponent can be randomly changed, how the input for exponentiation may be randomly changed, and how the modulus might also be modified. All these methods can be used in combination with the inventions described here.

However, in the case of algorithms for performing the exponentiation itself, there have been few major advances recently. The main methods for performing exponentiation are described by Knuth. A more recent journal paper by Gordon bridges the gap between this and current state of the art. All this prior art in the public domain is exclusively directed towards *efficient* exponentiation, not *leak resistant* exponentiation. The most relevant material to this disclosure is a work by the algorithm inventor which is also on efficient exponentiation, namely, that by Walter in the IEEE Transactions on Computers.

At the CHES 2001 conference, Oswald and Aigner showed how random variation in the exponentiation scheme could be used as a counter-measure to side channel attacks, and provided an example for elliptic curve cryptography where an inverse can be calculated, but provided no way of doing this if the inverse of the initial input text were unavailable. Hence their method is inapplicable to integer RSA.

The invention here is for secure and leak-resistant implementation of exponentiation which is also efficient. The proper setting for this invention is the general structure referred to by mathematicians as a multiplicative group, and *all* applications of exponentiation in such settings are suitable for protection by the inventions described here, not just those mentioned explicitly in this disclosure. Whilst the exemplar apparatus described below uses modular arithmetic in the context of the RSA cryptosystem, no such restrictions to the invention are so implied. For example, elliptic curve cryptography and Diffie-Hellman key exchange can also be protected in part by this invention.

3. Background Theory.

The process of exponentiation can be described using a scheme which determines which partial results need to be multiplied together at any given time. In general, the exponentiation scheme is determined by an ***addition chain*** (*see* Knuth). For convenience in this exposition, an addition chain is represented more precisely here by a sequence of multiplication instructions which are written as triples of the form (i,j,k). Here (i,j,k) means multiply the contents of registers i and j together and write the contents into register k. For example, (1,1,2), (2,2,2), (1,2,1) defines an addition chain which Knuth would present as 1+1 = 2, 2+2 = 4, 1+4 = 5. If M is initially in register 1, then the first instruction is to compute $M^2 = M^{1+1} = M^1 \times M^1$ and put the result M^2 in register 2, the second instruction overwrites this, putting $M^4 = M^{2+2} = M^2 \times M^2$ into register 2, and the third instruction computes $M^5 = M^{1+4} = M^1 \times M^4$ and writes the result into register 1. So this is an addition chain which defines one scheme for exponentiating to the power 5 using only two registers and determines how those registers are

used. To complete the description, the requisite initialisation of the registers must be presented, and the output register determined. For the example just presented, this means initialising register 1 with M and reading the result from register 1.

Addition chains for the standard square-and-multiply and *m*-ary methods are easy to generate. For the *m*-ary method to compute a power of M, M^j is pre-computed and put into register j, say. Register 0 can be used to store the partial product, as it is created. It is initialised with 1. When $m = 2^i$, the exponent is represented with base *m*, and it has *i*-bit digits. The addition chain then consists of a repetition *i* squarings given by (0,0,0) followed, if j is non-zero, by a multiplication (0,j,0) where j is the next exponent digit, starting with the most significant exponent digit.

More efficient addition chains can be found using the method described by Walter in the IEEE Transactions on Computers. This uses the theory of ***division chains***. Suppose that exponentiation to the power E is required and that E can be expressed in the form E = FD+R. Then computing M^E can be reduced to computing M^D and M^R first and then computing $M^E = (M^D)^F \times M^R$. Raising to the power F here is done recursively in the same way, perhaps using a different value for D, and the process is repeated until the problem is reduced to raising to the power 0. The corresponding iterative version of this exponentiation algorithm simply multiplies all the powers M^R together as they are formed, and replaces M by M^D as the number which requires to be exponentiated. For each step, D is called the **divisor** and R the **residue**. The sequence of pairs (D,R) which reduces E to 0 is called a **division chain**. Walter initially selects a large set of divisors which have efficient addition chains for computing M^R and M^D in parallel, and then generates a large number of addition chains for M^E by always selecting the few most efficient divisors for extending the most efficient partial division chains constructed so far. Out of these many division chains, the most efficient is selected to provide an addition chain for computing M^E. Efficiency here is defined as the one with the shortest addition chain.

In detail, the algorithm for constructing one of these divisor chains can be presented in a suitable form for this invention using a Pascal-like pseudo-programming language as follows. The included modifications form part of the novel content of this invention.

Relevant component from Walter's Division Chain Exponentiation Algorithm adapted and suitably modified for the present purpose:

Inputs: *E* a non-negative integer;
M in the given multiplicative group;
Output: $ResultM = M^E$ in the multiplicative group;

Variables: *StartM* and *ResultM* in the given multiplicative group;
RemE, *D* and *R* non-negative integers;

```
Begin
    RemE ← E ;
    StartM ← M ;
    ResultM ← 1 ;
    While RemE > 0 do
    Begin
        Choose the most efficient D and R ;
        ResultM ← StartM^R × ResultM ;
        StartM ← StartM^D ;
```

RemE ← (RemE − R)/D ; { The choice of R must make this division exact }
{ Invariant: $M^E = StartM^{RemE} \times ResultM$ }
End
End

In each iteration of the loop here, the final values of StartM and ResultM are computed from the initial values using a single, fixed, optimal (i.e. shortest) addition chain which contains both D and R. In the discussion below, the addition chains for each divisor/residue pair (D,R) will be called addition *subchains*. When all of these are concatenated together they yield an addition chain which determines a scheme for computing any Eth power, as required. This complete addition chain, together with any relevant additional detail such as register locations, will be referred to as an *exponentiation scheme*.

An efficient choice for the divisor set tends to select divisors with few non-zero bits in their binary representations. Walter provides {2, 3, 5, 17, 33, 49, 65, 97, 129, 257, 513, 1025} as an example of a small set from which to choose divisors and he provides a *deterministic* method for selecting the divisor at each step. This divisor set is semi-successful at producing the very short addition chains required for efficient exponentiation. A much larger divisor set is better. This means that cryptographic devices do not generally have enough memory to use Walter's deterministic algorithm to speed up exponentiation. Indeed, to obtain a relatively efficient chain, Walter has to generate a large number of addition chains by choosing several of the most efficient divisors at each iteration in order to extend the best partially completed addition chains. It is counter-intuitive that the method will generate useful addition chains in this context.

SUMMARY OF THE INVENTION

The invention here provides a secure or leak-resistant implementation of exponentiation, such as is required for many public-key algorithms and protocols, such as RSA encryption, ElGamal encryption, Diffie-Hellman key exchange and the Digital Signature Standard. The power of the invention is that it has an efficient mechanism for generating a different exponentiation scheme every time it is used and that each such scheme is itself efficient. This means, firstly, that attacks on the system which use differential power analysis are rendered much less harmful because little meaningful data can be extracted by averaging over a number of different exponentiations. Secondly, it means that the mechanism makes little difference to the overall time which the cryptographic device requires for its operation. However, thirdly, it should be noted that this invention can be used in conjunction with all currently known inventions with the same purpose, because all of them involve modifying the inputs to the exponentiation whereas no such change is required for the method here.

The main technique is a means through which each time exponentiation to the power E is required, a new, randomly generated addition chain is constructed either before, or in parallel with, performing the exponentiation. No two such exponentiation schemes are usually the same, although the same scheme can be re-used if desired. No other pre-computation is involved for individual exponentiations. In the preferred embodiment, an addition scheme is for the exponentiation scheme is generated using a novel modification of the divisor chain method of Walter and a random number generator (RNG) box to select the divisors.

The invention consists of a number of novel methods, not all of which need be used, and all of which can be constructed differently from the particular cases described in the exemplar

exponentiation box. Many of the other novel ideas of the invention involve the introduction of further random aspects which ensure more variation for each exponentiation, in order to make cryptanalysis even more difficult. These are set out in the claims and preferred embodiments.

24 Claims, 1 Drawing Sheet

BRIEF DESCRIPTION OF THE FIGURES

Figure 1 shows schematically the method and apparatus of the invention in one of its preferred embodiments.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

INTRODUCTION

The preferred embodiment of the novel exponentiation device contains a version of the exponentiation machine described in the "Background of Invention" section, which is adapted from Walter's division chain exponentiation algorithm. Other embodiments may use equivalent variants of this in which, for example, the list of divisors is chosen before any exponentiation is performed rather than in parallel with it. In order to build an embodiment of the invention, this machine is augmented and enhanced by the various novel methods and apparatus which are set out in the claims. It will now be described, referring to the drawing provided in Figure 1. Since the mathematical notation and programming code are unambiguous and well understood by those of skill in the art, no detailed description of same is deemed necessary for a full and precise understanding of the present invention.

The invention and novel ideas presented herein are applicable in the area of number-theoretic public-key schemes (including encryption schemes, signature schemes, authentication schemes, key exchange schemes, key management schemes etc. using modular integer arithmetic, finite field arithmetic, elliptic curve groups, etc.) where it is required to protect the secrecy of the private key in a hostile environment. The invention is applicable wherever exponentiation is used and provides an apparatus for performing the exponentiation which is resistant to attacks involving side channel leakage of data.

The apparatus for computing one value of M^E at a time (claim 1) is illustrated in the Figure 1. This is to be performed in the common mathematical structure referred to as a group, and this group will be written using multiplicative notation. In the context of the (integer) RSA crypto-system or Diffie-Hellman key exchange with modulus N, the group is that of the integers mod N under multiplication. With this as the specified group, it is unnecessary to write the "mod N" explicitly. In the context of elliptic curve cryptography, the group is that of the points on the elliptic curve under what is normally referred to as addition. For uniformity with the integer cases, this group will be written multiplicatively as well. So, following normal mathematical practice, it should be understood that in all applicable contexts, the exponentiation is described with multiplication as the group operation.

OVERVIEW

The drawing shows the inputs M and E provided in boxes **10** and **21**, and the required output which appears in box **23**. There are two main processes which may run interleaved or sequentially, or in some combination of these, using a single processor or multiple processors.

These processes are i) the selection of the exponentiation scheme, which includes mainly the choice of addition chain (boxes **10** through **20** in Figure 1), and ii) the application of the exponentiation scheme to perform an exponentiation (boxes **21** through **23** in Figure 1).

During the process of selecting an exponentiation scheme, the exponent E in box **10** is copied to RemE in box **11**, where, for convenience, it may have a non-standard representation (claim 9) in order to make more efficient its division by any chosen divisor. After the divisor D is selected in box **13**, the new value for RemE and the residue R are computed by box **15** from the formulae $R \leftarrow \text{RemE mod D}$ and $\text{RemE} \leftarrow (\text{RemE} - R)/D$. The value of RemE in box **11** is updated, and the pair (D,R) is passed on to box **17** which stores this data and further choices ready for the exponentiation. The exponentiation may take place immediately in box **22** or it may be performed later.

During the process of performing the exponentiation, the register selector in box **18** first determines the locations of StartM, ResultM and TempM. Then the initial value M in box **21** is loaded into StartM in box **22** and ResultM is initialised with 1 in box **22**. Then (using the notation described in the background theory) the addition chain operation triples (i,j,k) are repeatedly obtained from box **22**, modified by box **18** if necessary, and applied to perform a multiplication and storing of the product. When the last triple has been processed, box **23** obtains the result M^E from ResultM in box **22**. This embodiment uses just the three registers named in box **22**. However, other embodiments may use more registers as desired.

DETAILS

In detail, the choice of exponentiation scheme is made as follows. A divisor set is chosen and provided in box **12**. For the preferred embodiment, this set contains divisors 2, 3 and 5 only (claim 1), but almost any choice of a set of positive integers can be made instead (claim 6). Associated with each divisor D from this set, and integer R with $0 \le R < D$, an addition chain (*see* the section "Background Theory" for definitions) is chosen which includes R and D and has shortest length with this property (claim 1). These addition chains are put into box **16**. This is the preferred method for initialising box **16**, but (see claims 7 and 8) alternative and additional addition chains can be placed in box **16**. There may be several chains representing the same pair (D,R) (claim 7) and these chains do not need to be chosen with minimal length (claim 8), although faster exponentiation is achieved when this is done. In particular, these subchains may or may not contain "useless" operations which have no effect on the computation and could have been omitted. Furthermore, the set of residues from which R is chosen need not be restricted in any way (claim 13). Indeed, the set of residues associated with a divisor does not need to form a complete set of residues, nor do its elements need to be all non-negative, nor do they all need to be least non-negative. However, for each selectable pair (D,R) there must be at least one available addition chain placed in box **16**. Moreover, sufficient residues must be available for there to be at least one selectable pair (D,R) when an exponentiation scheme is being generated.

Before operation, the exponentiation apparatus must also be provided with a divisor selection processor (box **13**) which will select divisors in an unpredictable way (claim 2). There are many different ways to construct such a box. One such suitable embodiment for this is the following exemplar apparatus which illustrates some of the possible enhancements more clearly. This suggested construction (claim 17) obtains input from a random number generator in the interval [0,1] (RNG) provided by box **20** and uses the data in the divisor set provided by box **12**. It is written in the programming language Pascal.

```
D := 0 ;
If RNG < 7/8 then
    If 0 = RemE mod 2 then D := 2 else
    If 0 = RemE mod 5 then D := 5 else
    If 0 = RemE mod 3 then D := 3 ;
If D = 0 then
Begin
    p := RNG ;
    If p < 6/8 then D := 2 else
    If p < 7/8 then D := 3 else
                    D := 5 ;
End ;
```

Other embodiments may have random numbers supplied in a different interval, they may have different values replacing the occurrences of 7/8, 6/8 and 7/8, they may re-order some of the statements, or they may make other inconsequential changes as far as the functionality of the box is concerned. These are all deemed to be covered in this invention. In the program segment, RemE refers to the remaining part of the exponent E which still has to be processed, and is stored in box **11**. This construction (claim 21) is an embodiment of claim 11, namely that no deterministic choice is made for any divisor. An alternative construction for box **13**, which is useful for improving efficiency, is that a divisor which divides RemE is always chosen whenever possible (claim 10). This could be implemented by omitting the line "**If** RNG(x) < 7/8 **then**" in the above Pascal code, thereby obtaining the means of claim 20. A further enhancement is to update this selection process regularly in a random way (claim 12). One embodiment of this idea is for box **14** to update the values 7/8, 6/8 and 7/8 in the Pascal code for divisor selection at the start of each new exponentiation. Box **14** is supplied with random numbers from box **20** to enable this process to take place. The new values are restricted to being probabilities in the range [0,1] and are usually selected strictly between 0.5 and 1. This provides the means to implement claim 12 when applied for every new exponentiation (claim 22). This updating process can be combined with a random number from box **20** which is used to select unpredictable intervals between such updates (claim 14). Consequently, this provides a mechanism for enabling updates to a randomly determined selection process to occur at unpredictable intervals (claim 23).

When a divisor and residue pair (D,R) has been selected by box **15**, an addition subchain must be selected by box **17** from the list of such chains provided in box **16**. If there is more than one addition subchain associated with (D,R), then the random number generator (box **20**) is used to select one of them. Any selection process can be used for this, whether predictable or not (claim 7). Normally a weighted, random selection is made to improve cryptographic strength or to improve efficiency, such as by making non-minimal length subchains less likely to be chosen. In the case of there being n subchains for the chosen pair (D,R), a random number in the interval [0,1) from box **20** can be used to select the ith subchain when the random number lies in the sub-interval $[i/n,(i+1)/n)$ (claim 19).

When the addition chain has been selected by box **17**, the register location selector (box **18**) can be used to modify the default locations of variables used in the computation of M^E. Using input from the RNG (box **20**), these locations can be chosen randomly from available RAM, but will normally just permute the locations within the space that the apparatus requires for storage of various powers of M. Then, when a new power of M is computed, it will be written into the random location determined by box **18** (claim 4). For most

embodiments, three registers are used, (these are called StartM, ResultM and TempM in box **22** of the figure), and frequently only two values need to be kept (namely StartM and ResultM in this embodiment). This leaves a third which is free to be overwritten. So a freshly computed product can overwrite either its previous value or, when available, any other value which is not going to be used subsequently.

The data for the addition chains and locations can be managed using the notation described in the background section: (*i*,*j*,*k*) means this: take the group elements in locations *i* and *j*, compute their product, and write the result into location *k*. A typical addition subchain for the divisor-residue pair (5,3) is stored in box **16** as the sequence of four triples (112, 121, 133, 121) where 1 denotes the register called StartM in box **22**, 2 denotes the register TempM and 3 denotes the register called ResultM. Triple (133) illustrates the preferred means by which residue powers are multiplied together (claim 1) for this subchain. Rather than letting box **18** compute which register locations can be permuted, box **16** can be supplied with extra subchains representing alternative location choices. So subchain (112, 121, 133, 122) writes the last product into location 2 instead of the 1 used in the previous subchain. The random selection of locations for writing products to is therefore achieved via the random selection of an addition subchain from box **16** for the divisor/residue pair (claim18). Then box **18** just needs to record that StartM will be in location 2, not 1, for the next subchain. Observe that the storing of such variables in random locations does not involve writing to a pre-determined location and then copying from there to the random location.

FINAL REMARKS

This apparatus presented in the drawing can be built with any or all of the enhancements presented above and in the claims (claim 16) and can be used for repeated exponentiations where the exponentiation scheme generated by the apparatus is changed with any desired frequency, whether regular or irregular (claim 3). Normally, each exponentiation would be performed with a freshly generated scheme, but it could also be changed at fixed, regular intervals or at randomly selected irregular intervals.

The Chinese Remainder Theorem (CRT) enables a large exponentiation to be re-expressed in terms of several smaller exponentiations. The invention here applies equally well to the component exponentiations, since it applies to any exponentiation (claim 15). There are many other means of modifying the exponentiation for increased efficiency or increased security or for other purposes. Whenever such modifications require exponentiations to be performed, the present invention may be applied in a similar fashion. In particular, the invention of Shamir (US Patent 5991415) can be applied to the exponent and other inputs, providing a new exponentiation to which the present invention can also be applied (claim 5).

The random number generator (RNG) in box **20** may be implemented in many different ways. Different methods may be used for the different requests from other boxes of Figure 1 to box **20**. Any method is acceptable here and all methods are covered by the invention. Normally, an RNG generates a so-called pseudo-random sequence. In the preferred embodiment, this RNG should be secure against side channel attack.

The exponent RemE of box **11** may be represented with any radix, as may be E in box **10** (claim 9). A multiple of the least common multiple (LCM) of the divisors in box **12** is preferred. For the choice {2,3,5}, a base of 30, 60, 120 or 240 is ideal when the word size used to hold RemE in memory is 5, 6, 7 or 8 bits respectively. Division of RemE and

determination of residue R are then much easier: once divisor D is chosen (box **13**) box **15** can determine R from the lowest digit of RemE, and the next value of RemE can be obtained by a multiplication and a shift down, in a manner well understood by those skilled in this art (claim 24). For example, with an 8-bit processor, base 240 would be chosen and division by D would be performed using multiplication by 240/D and a shift down.

The most efficient method for combining the residue powers is to multiply them into a single variable, called ResultM in the above (as described in claim 1). However, several variables ResultM0, ResultM1, ..., ResultMk might be set up, initialised, and the residue powers multiplied into a pre-determined or randomly chosen one of these (claim 25). Their locations may be changed randomly, of course (claim 4). The product of the contents of these registers must be calculated in order to obtain the required output, and this may be done by multiplications spread out over the whole exponentiation instead of at the end only.

Many choices described in this preferred embodiment are practical ones, in order that the invention be described clearly. However, the invention covers all possible choices without restriction. In particular, sections of program code might be replaced by functionally equivalent code in any programming language. This includes re-ordering the execution of some statements. The choice of divisor set {2,3,5} is for illustration only, and any set is covered in this invention. Addition subchains which contain the pair (D,R) have not been given explicitly as these can be constructed easily by a practitioner skilled in the art.

END OF PATENT DESCRIPTION

DRAWING SHEET 1 OF 1

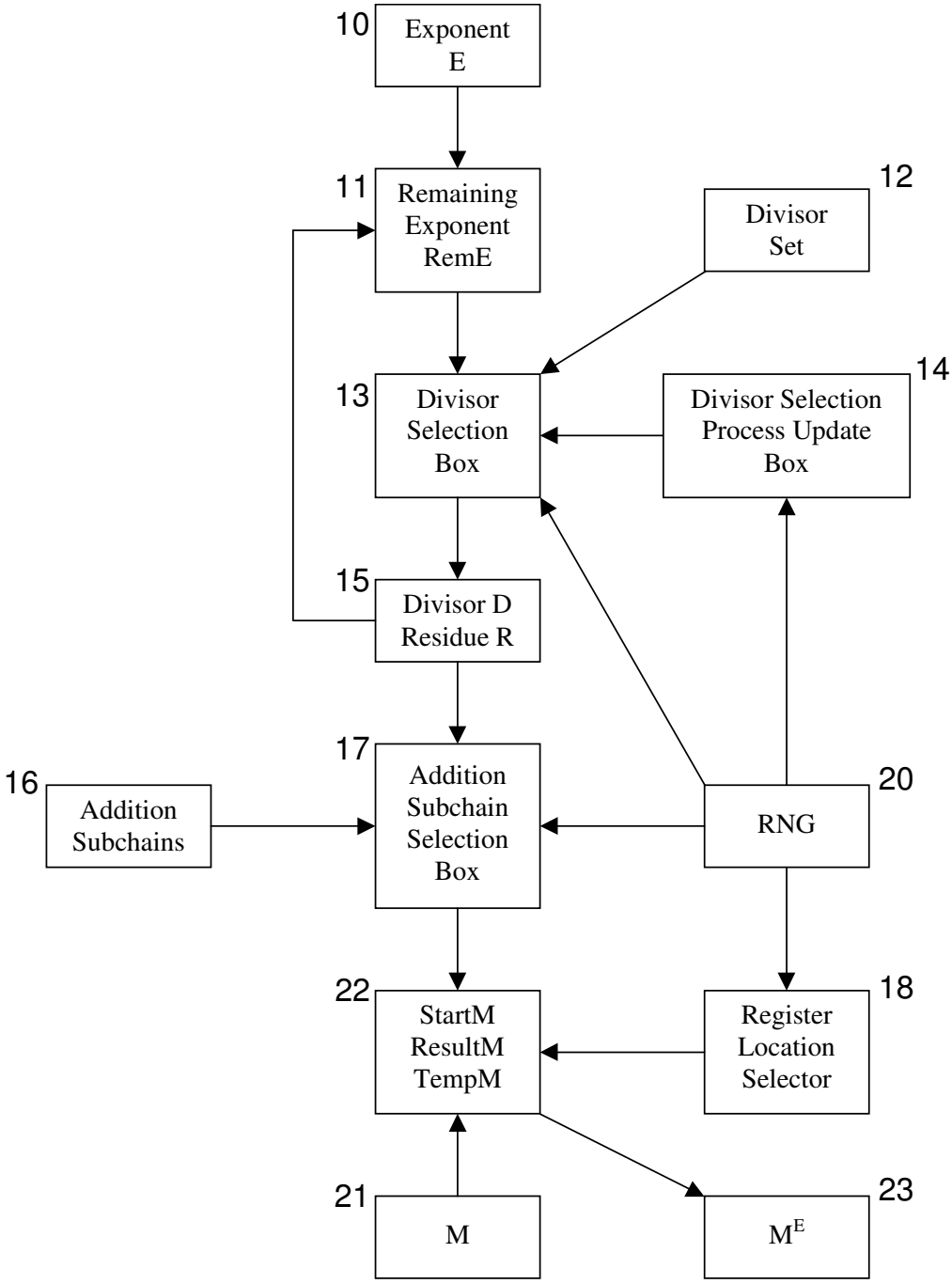

FIG. 1