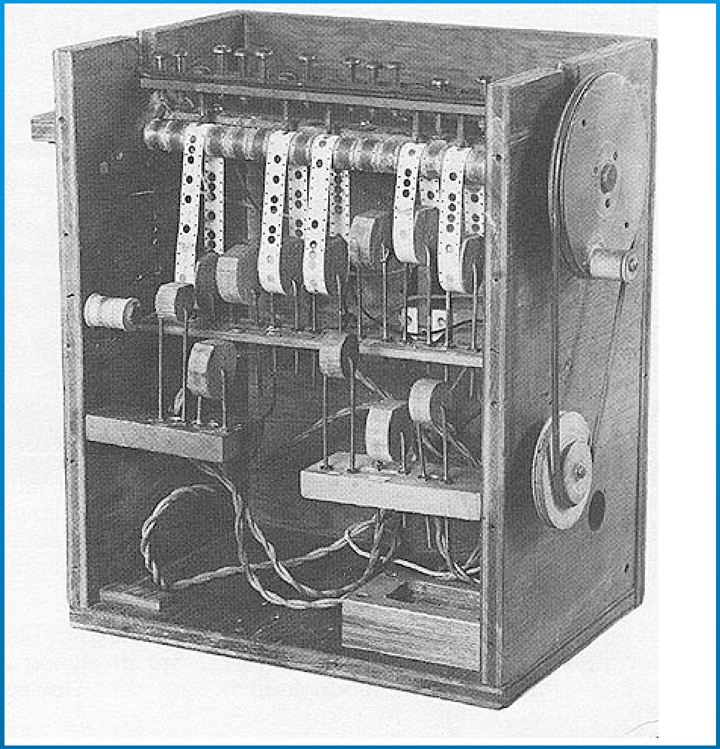
Cryptanalytic Devices



D.H. Lehmer’s movie film sieve

Table of Contents

RSA

1. Publicly described in 1977 by Ron Rivest, Adi Shamir, and Leonhard Alderman of MIT
2. Letters RSA are the initials of their surnames
3. An algorithm used for encryption
4. Based on two mathematical problems and the assumption that no efficient algorithm exists for solving them
5. First is the problem of factoring large numbers
6. Second called the RSA problem, defined as: the task of taking *e*th roots modulo a composite *n*: recovering a value *m* such that $c=m^e$ mod n, where (*e*, *n*) is an RSA public key and *c* is an RSA ciphertext (the encoded version of a method)

Purpose of Cryptanalytic Hardware

The purpose of the following devices is to perform integer factorization for very large numbers. There exist many methods to factor integers, however it is difficult for these to be feasible – having a reasonable cost, and fast enough speed – to be able to factor such large composite integers (for instance, 1024 bit composites) as are generally used by RSA. There is work done on software, and on hardware to work on an effective method to compute factorization; this presentation will focus on the hardware component.

The Number Field Sieve

- The Number Field Sieve (NFS), also called the general number field sieve, is the most efficient algorithm known for factoring large integers.
- The principle of the number field sieve (both special and general) can be understood as an extension of the simpler rational sieve. When using the rational sieve to factor a large number *n*, it is necessary to search for smooth numbers (i.e. numbers with small prime factors) of order *n* (given an integer a and a positive integer n with gcd(a,n) = 1, the multiplicative order of a modulo n is the smallest positive integer k with ak ≡ 1 (modulo n); the rarity of these causes the rational sieve to be impractical. The general number field sieve, on the other hand, only requires a search for smooth numbers of order $n^{1/d}$, where *d* is some integer greater than one. Since larger numbers are far less likely to be smooth than smaller numbers, this is the key to the efficiency of the number field sieve. But in order to achieve this speed-up, the number field sieve has to perform computations and factorizations in number fields. This results in many rather complicated aspects of the algorithm, as compared to the simpler rational sieve.
- This method is used by cryptanalytic devices, which usually break it up into two parts, the sieve part (which will be the part discussed), and the linear algebra part.

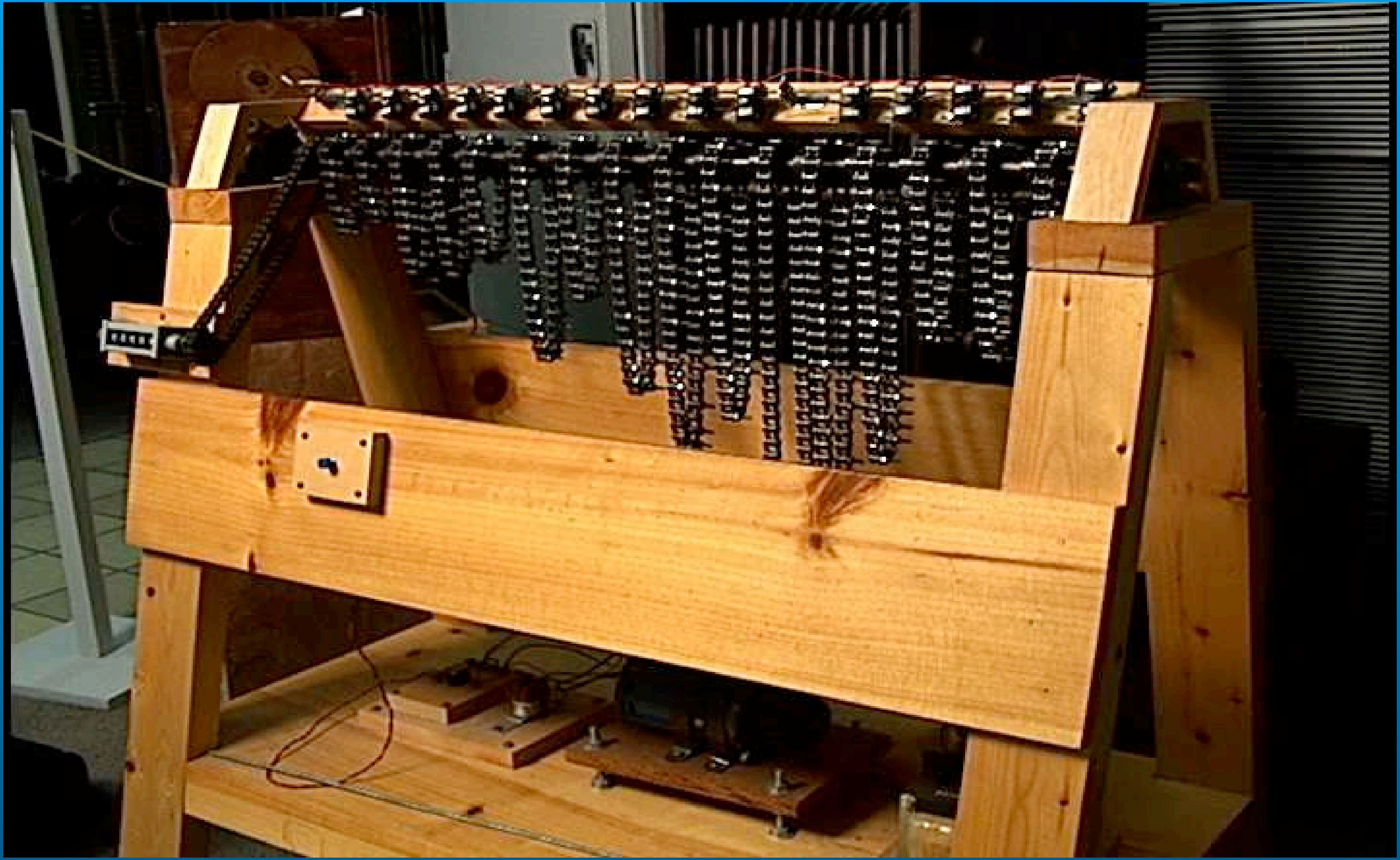
Historical Devices

- Employing mechanical devices for the sieving task was apparently first proposed by Frederick William Lawrence in 1896.
- Machine á Congruences, built by Eugène Olivier Carissan in 1919 as an improvement upon his brother's earlier design, is the first known sieving machine that operated successfully. It consists of 14 concentric brass rings, and employs electrical switches to identify events corresponding to (likely) squares in the Fermat method. It presently resides at the Conservatoire Nationale des Arts Métiers, Paris.
- Lehmer's mechanical sieves (mentioned in detail later) in the 1920's and 1930's, D. N. Lehmer and D. H. Lehmer built several mechanical sieving devices, employing various approaches: bicycle chains (1926, replica at the Computer History Museum), gears and photoelectric detectors (1932, presently at the Computer History Museum) and movie films (1936). In the 1970's, D. H. Lehmer also built electronic sieving devices using delay lines and shift registers (1970's).
- Extended Precision Operand Computer, AKA "Georgia Cracker“, is a 128-bit, highly parallel special-purpose computer for factoring using the Continued Fraction method, built by Jeffrey Smith and Samuel Wagstaff in 1982-3.
- Bomba (mentioned in detail later), electro-mechanical computers for breaking the German cipher Enigma. Built by Polish General Staff's Cipher Bureau in 1938 and successfully broke encryptions employing the basic variant of Enigma.
- Heath Robinson and Colossus Electro-mechanical computers for breaking the German cipher Lorenz. Built by the British Government Codes & Ciphers School at Bletchley Park (via the British Post Office's research center) circa 1943. Overall 10 such machines were built and operated with great success. Rebuilt from 1994 by Tony Sale and the Colossus Team.
-

D. N. Lehmer and D. H. Lehmer’s Mechanical Sieves

- The name of Lehmer is famous in the mathematical sieve process because two different Lehmers were involved. D. N. Lehmer was the father and D. H. Lehmer was the son.
- The original "bicycle chain" sieve was made in in 1926. The geared sieve dates from 1932. It is made up of gears of different sizes representing numbers in exactly the same way that the bicycle chains or strips of movie film were of different length. Holes in each gear represented various remainders when these numbers were divided by others and they could be plugged up with toothpicks to only leave a certain set of these numbers open. A bright light on one side of the machine would shine on the gears and, if the holes lined up, then it would reach the other side and trip a flip-flop (an electronic circuit which has two stable states and thereby is capable of serving as one bit of memory) attached to the photocell – perhaps the first time that an electronic flip-flop was used in a computing device (they had been used earlier in counters for cosmic rays and this is what gave Lehmer the idea). The output of the flip-flop was amplified and would be used to stop the motor spinning the wheels – thus allowing you to see the solution to the problem

Lehmer's Bicycle Chain Sieve



Bomba

- The ***Bomba***, or ***Bomba kryptologiczna*** (Polish for "**Bomb**" or "**Cryptologic bomb**") was a special-purpose machine designed about October 1938 by Polish Cipher Bureau cryptologist Marian Rejewski to break German Enigma machine ciphers.
- The German Enigma used a combination key to control the operation of the machine: rotor order, which rotors to install, which ring setting for each rotor, which initial setting for each rotor, and the settings of the stecker plugboard. The rotor settings were trigrams (for example, "NJR") to indicate the way the operator was to set the machine. German Enigma operators were issued lists of these keys, one key for each day. For added security, however, each individual message was encrypted using an additional key modification. The operator randomly selected a trigram rotor setting for each message (for example, "PDN"). This message key would be typed twice ("PDNPDN") and encrypted, using the daily key (all the rest of those settings). At this point each operator would reset his machine to the message key, which would then be used for the rest of the message. Because the configuration of the Enigma's rotor set changed with each depression of a key, the repetition would not be obvious in the ciphertext since the same plaintext letters would encrypt to different ciphertext letters.
- This procedure, which seemed secure to the Germans, was nonetheless a cryptographic error. Using the knowledge that the first three letters of a message were the same as the second three, Polish mathematician Marian Rejewski was able to determine the internal wirings of the Enigma machine and thus to reconstruct the logical structure of the device.

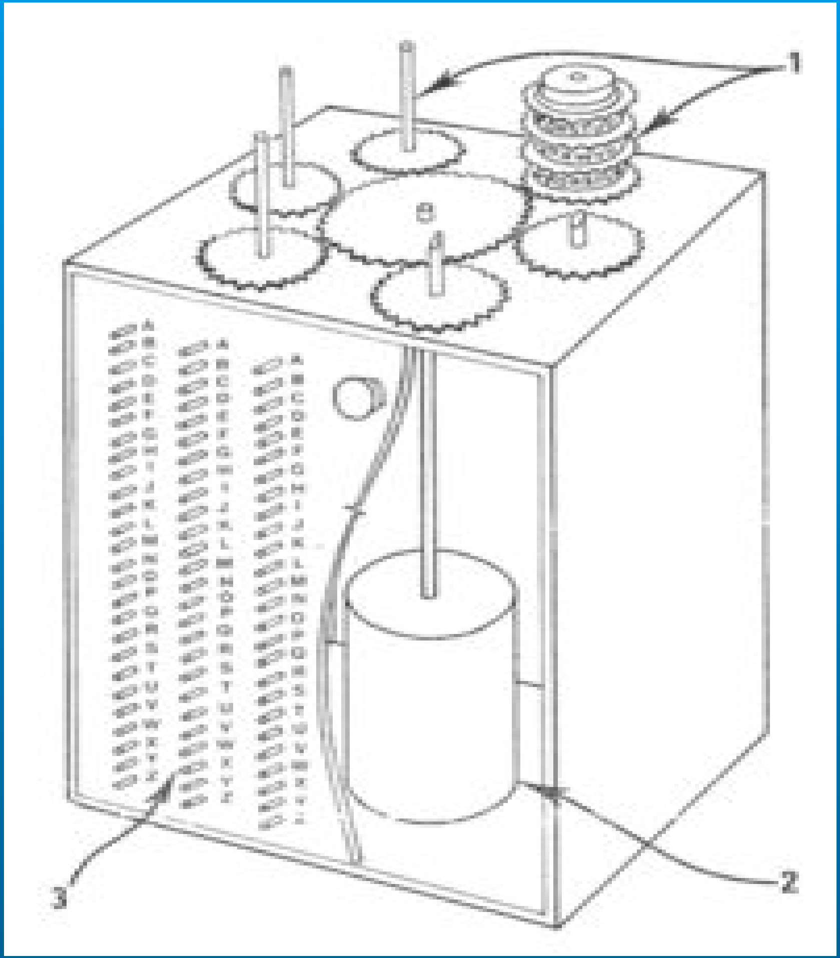
Bomba

Cryptologic bomb. Diagram from Marian Rejewski's papers.

1: Rotors (for clarity, only one 3-rotor set is shown).

2: Electric motor.

3: Switches.



Modern (Dates around 1999-2005) Cryptanalytic Devices

- These have never been built, though some of them are said to be feasible
- All use Number Field Sieve algorithm
- Devices to perform the sieving part of NFS are TWINKLE, TWIRL, and SHARK
- Devices to perform the linear algebra part of NFS are mesh-based, and pipelined (these will not be discussed)

TWINKLE

TWINKLE ("The Weizmann Institute Key Locating Engine") First modern sieving device. Never built. Uses non conventional electro-optical components.

The TWINKLE device is an optoelectronic device which is housed in an opaque blackened cylinder whose diameter is about 6 inches and whose height is about 10 inches. The bottom of the cylinder consists of a large collection of LEDs (light emitting diodes) which twinkle at various frequencies, and the top of the cylinder contains a photodetector which measures the total amount of light emitted at any given moment by all the LEDs. The photodetector alerts a connected PC whenever this total exceeds a certain threshold. Such events are related to the detection of possibly smooth numbers, and their precise timing is the only output of the TWINKLE device. Since these events are extremely rare, the PC can leisurely translate the timing of each reported event to a candidate modular square, verify its smoothness via trial division, and use it in a conventional implementation of the QS (Quadratic Sieve, used for same purpose as NFS) or NFS algorithms in order to factor the input n.

TWINKLE Sieving Technique

The standard PC implementation of the sieving technique assigns modular squares to array elements (using space) and loops over the primes (using time). The TWINKLE device assigns primes to LEDs (using space) and loops over the modular squares (using time), which reverses their roles. This is schematically described in Fig. 1.

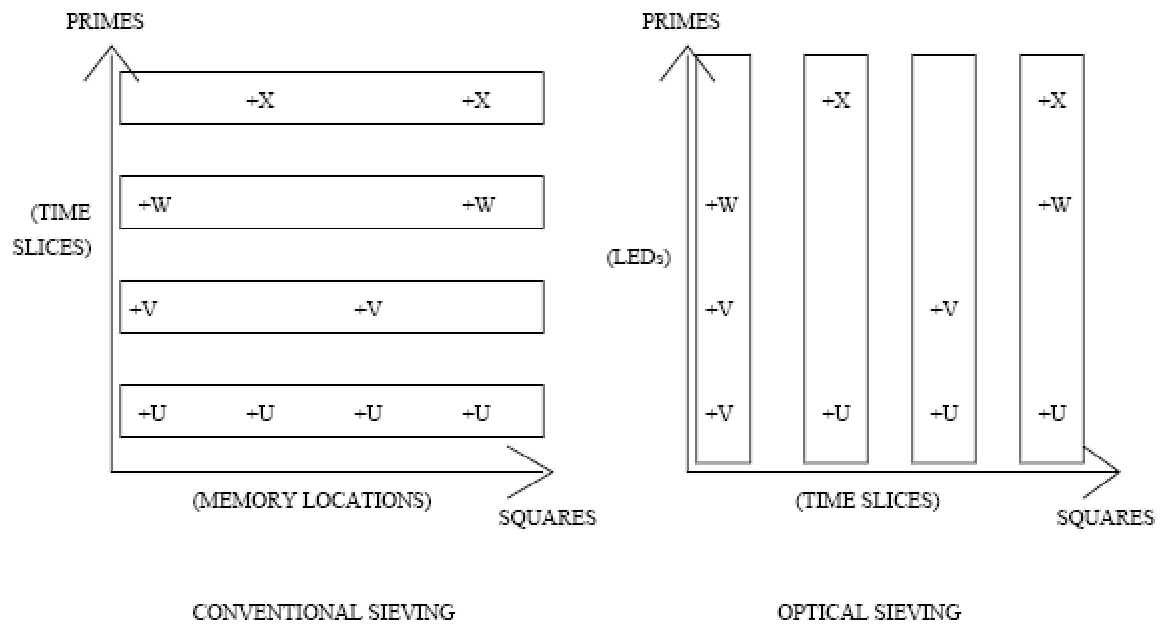


Figure 1: Conventional vs. optical sieving: the boxed operations are carried out at the same time slice

TWINKLE Cell Design

Cell Design

The LED array is implemented on a single wafer of GaAs (Galium (III) arsenide, semiconductor). Each cell on this wafer contains one LED plus some circuitry which makes it flash for exactly one clock cycle every exactly pj clock cycles with an initial delay of exactly dj clock cycles. The high clock rate and extremely accurate timing requirements rule out analog control methods, and the unavoidable existence of bad cells in the wafer rules out a prewired assignment of primes to cells. Instead, we use identical cells throughout the wafer, and include in each cell two registers, A and B, which are loaded before the beginning of the sieving process with values corresponding to pj and dj , respectively. For a typical sieving run over m = 100; 000; 000 values, we need only log2(m) _ 27 bits in each one of these registers. The structure of each cell (described in Fig. 2) is very simple. Instead of using counters (with their more complicated designs and additional carry-induced delays), we use register B as a maximal length shift register based on a single XOR of two of its bits. It is driven by the clock, and runs until it enters the special state in which all its bits are "1". When this is recognized by the AND of all the bits of register B, the LED flashes, and register B is reloaded with the contents of register A (which remains unchanged throughout the computation). The initial values loaded into registers A and B are not the binary representations of pj and dj , but the easily computed states of the shift register which are that many steps before the special state of all "1". That's the whole cell design!

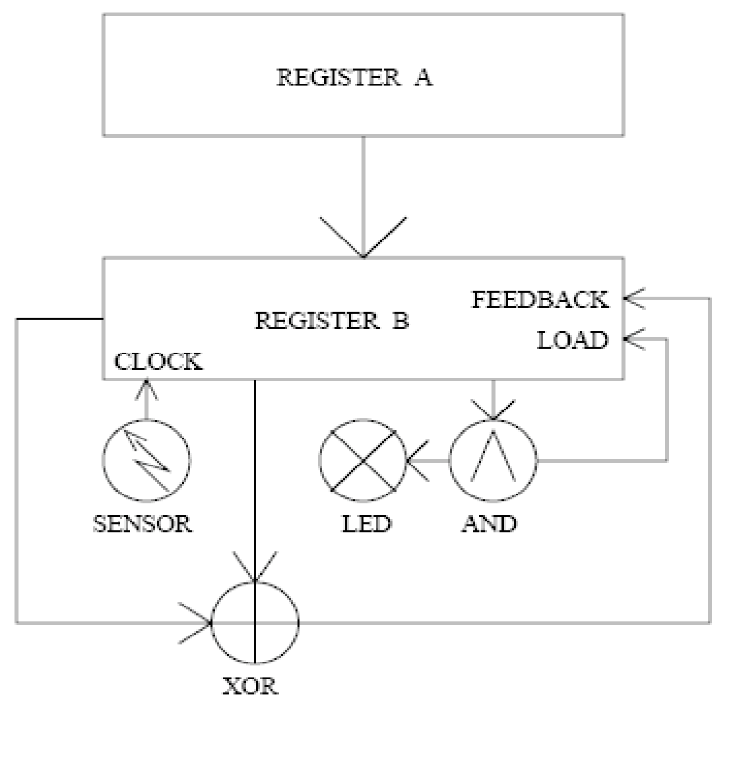


Figure 2: A single cell in the array

TWIRL

A new device that combines idea that that since it is inefficient to have unused memory cells sitting around, & merely storing the input is expensive, we should utilize it efficiently by appropriate parallelization, with the TWINKLE like approach of exchanging time and space. Whereas TWINKLE tests sieve location one by one serially, the new device handles thousands of sieve locations in parallel at every clock cycle. In addition, it is smaller and easier to construct: for 512-bit composites we can fit 79 independent sieving devices on a 30cm single silicon wafer, whereas each TWINKLE device requires a full GaAs (Galium (III) arsenide, semiconductor) wafer.

Consider first a device that handles one sieve location per clock cycle, like TWINKLE, but
does so using a pipelined chain of electronic adders. Such a device would consist
of a long unidirectional bus, that connects millions of conditional adders in series. Each conditional adder is in charge of one progression Pi; when activated by an associated timer, it adds the value to the bus.
At time t, the z-th adder handles sieve location t − z. The first value to appear at the end of the pipeline is g(0), followed by g(1), . . . ,g(R), one per clock cycle.

Two main difficulties arise: the hardware has to work s times harder since time is compressed by a factor of s, and the additions corresponding to the same given progression Pi can occur at different lines of a thick pipeline. Our goal is to achieve this parallelism without simply duplicating all the counters and adders s times. We thus replace the simple TWINKLE-like cells by other units which we call stations. We partition

TWIRL Structure for a Largish Station

Progressions whose pi values are much larger than s emit values very seldom. For these largish primes it is beneficial to use expensive logic circuitry that handles many progressions but allows very compact storage of each progression. The resultant architecture is shown in Fig. 2. Each progression is represented as a progression triplet that is stored in a memory bank, using compact DRAM storage. The progression triplets are periodically inspected and updated by special-purpose processors, which identify emissions that should occur in the “near future” and create corresponding emission triplets. The emission triplets are passed into buffers that merge the outputs of several processors, perform fine-tuning of the timing and create delivery pairs. The delivery pairs are passed to pipelined delivery lines, consisting of a chain of delivery cells which carry the delivery pairs to the appropriate bus line and add their contribution.

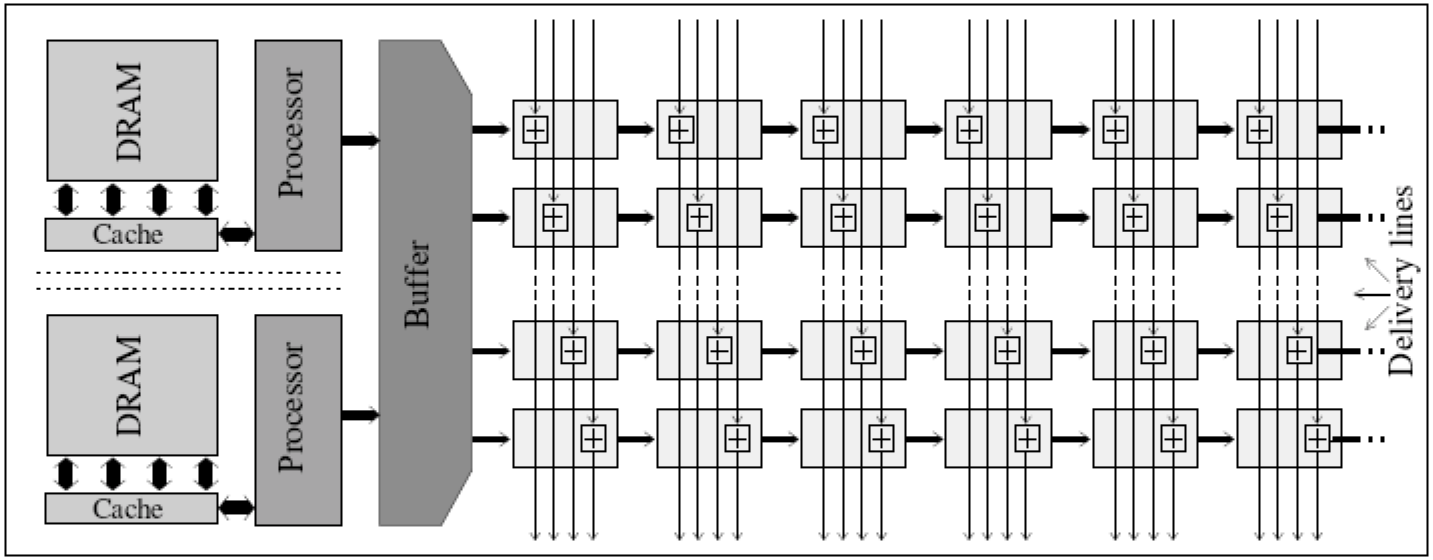


Fig. 2: Schematic structure of a largish station.

TWIRL Structure for a Smallish Station

Smallish Primes : for progressions with smaller pi values, each processor can handle very few progressions .Thus, the amortized cost of the processor, memory control circuitry and buffers is very high. Moreover, such progression cause emissions so often that communicating their emissions to distant bus lines (which is necessary if the state of each progression is maintained at some single physical location) would involve enormous communication bandwidth. We thus introduce another station design, which differs in several ways from the largish stations (see Fig 3).

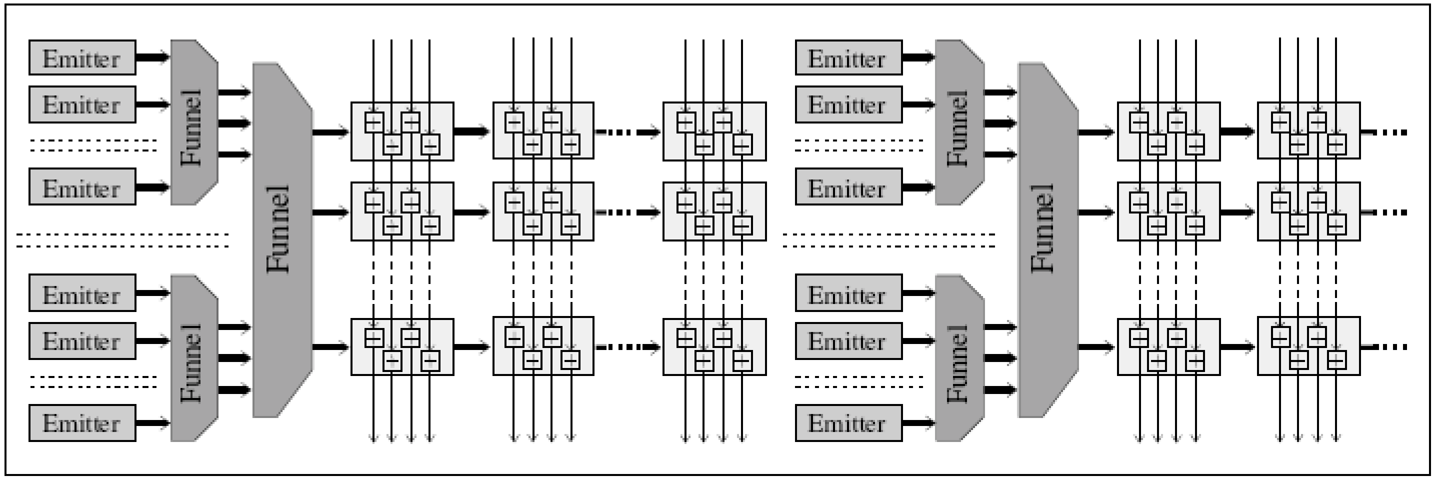


Fig. 3: Schematic structure of a smallish station.

TWIRL Structure for Tiny Primes

For very small primes, the amortized cost of the duplicated emitters, and in particular the related funnels, becomes too high. On the other hand, such progressions cause several emissions at every clock cycle, so it is less important to amortize the cost of delivery lines over several progressions. This leads to a third station design for the tiny primes.

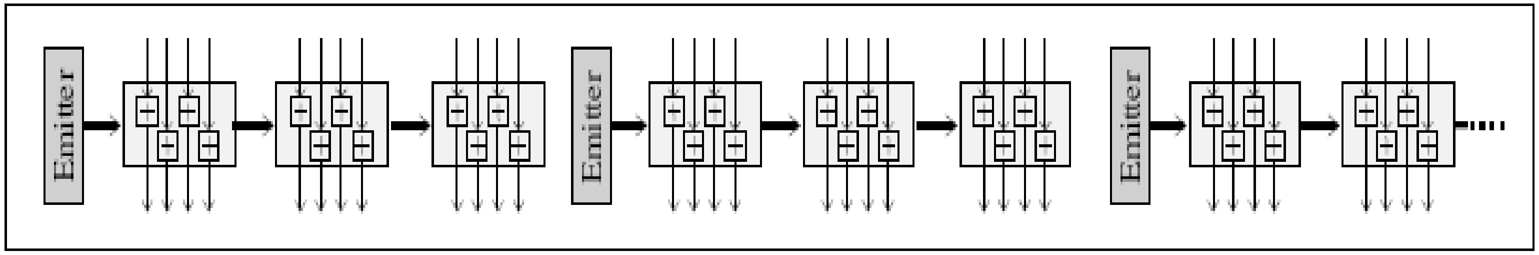


Fig. 4: Schematic structure of a tiny station, for a single progression.

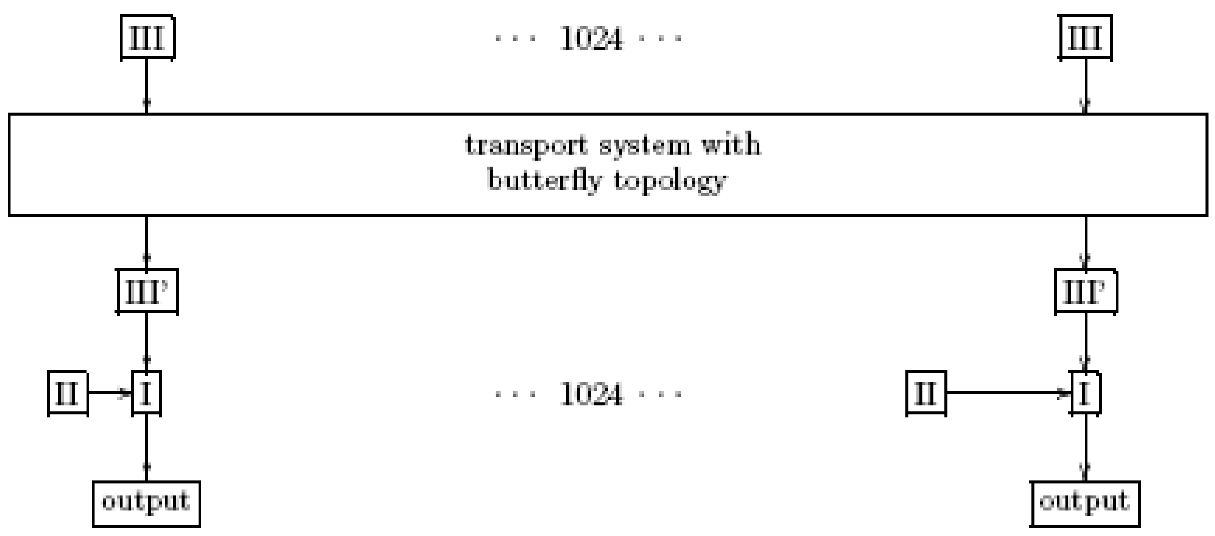
SHARK

- The architecture consists of 2300 identical isolated machines sieving in parallel.
- It uses lattice sieving. The actual sieving is done in very fast accessible memory cache. If this memory would be extremely cheap, we could construct a machine that sieves in some extremely large memory chip. Since this kind of memory is expensive we only use 32 MB of sieving cache memory.
- The sieving area is split into many small parts such that each part fits in the sieving cache.
- After the sieving of one small part is completed, the machine moves on to the next part until the whole sieving area has been scanned.
- The tricky part is to sort the sieving contributions such that all sieving contributions for a certain part are loaded into the sieving cache just before the sieving of that part starts.
- To achieve this, the data produced by the lattices corresponding to the larger primes of the factor base are sent through a specialized transport system with butterfly topology.
- The output of the sieve consists of potential sieving reports that still need to be checked for smoothness. This is done (after a quick compositeness test) by special hardware devices using the Elliptic Curve Method (ECM).
- The main difference to other proposed architectures is (in contrast to a giant monolithic ASIC (application-specific integrated circuit) its modular design composed of small ASICs connected by conventional data busses. The modularity is achieved by dividing the factor base into several parts and sorting the sieving data with a buttery transport system.

SHARK

The SHARK machine consists of parts I, II, III and a transport system (see Figure 1).

Figure 1: High-Level Schema of the SHARK Sieving Machine



Comparison

- TWINKLE: first (1999), based on electro-optics, mesh circuits (based on two-dimensional systolic arrays) would be capable of factoring 512-bit integers.
- TWIRL: successor to TWINKLE, based on parallel processing pipelines, would be able to factor 1024-bit composites in one year .
- SHARK: 2005, butterfly routing network, standard-sized chips, would be able to factor 1024-bit integer composites in one year, but would cost more than TWIRL.

Conclusion

- RSA coding assumes no efficient way to factor large integers.
- The Number Field Sieve considered to be the most effective method of factoring large integers.
- Computer software and hardware attempt large integer factorization.
- TWINKLE, TWIRL, SHARK theoretical devices for the Sieve.

Bibliography

1. http://ed-thelen.org/comp-hist/TheCompMusRep/TCMR-V04.html2.
2. http://ed-thelen.org/comp-hist/Mike-Williams-Lehmer.html
3. http://theory.csail.mit.edu/~tromer/cryptodev/
4. http://en.wikipedia.org/wiki/Number_field_sieve
5. http://en.wikipedia.org/wiki/Rsa
6. http://www.zyvra.org/chain/lehmerm.jpg
7. http://en.wikipedia.org/wiki/Bomba_%28cryptography%29
8. http://porsche.ls.fi.upm.es/Material/Articulos/twinkle.pdf
9. http://theory.csail.mit.edu/~tromer/papers/twirl.pdf
10. http://www.crypto.ruhr-uni-bochum.de/imperia/md/content/texte/publications/conferences/shark.pdf