SG100 Whitepaper

General Design Considerations

The SG100 generator divides into a hardware noise generator and a software driver. Between the SG100 hardware and the SG100 software driver is a computer port with a port driver. The important question here is how to cleverly split our mission into two parts: one suitable for processing in the hardware and the other suitable for the software driver; and how to overcome possible problems with the computer port in the middle. These questions affects throughput, quality, and reliability.

In the SG100 generator we have chosen to do as little processing as possible in hardware. This results in poor statistical performance, when measured on the raw hardware, but actually gives us several advantages.

First: Building hardware is expensive, at least if we compare to a software solution of the same problem.

Second: If no processing of the noise takes place in hardware it is simple to calculate a figure just how well the device is operating. Not only do we have the opportunity to install run-rime testing of the device, we can actually remedy a situation with a low noise output, if we know when it occurs. This makes the SG100 generator more reliable than other similar devices which lack run-time testing.

Third: The compatibility is increased. SG100 works on almost any serial port, facilitating the use of the same hardware on all platforms.

Hardware Properties

- Hardware connectable to any computer.
- Powered from the computer port - no batteries or cables.
- Device automatically switches off on battery-powered computers.
- High resistance against power fluctuations.
- High resistance against external electromagnetic fields.
- Information feed into computer difficult to intercept.
- Runtime electrical and statistical testing.
- High output speed: up to 9.500 bytes/sec.
- Can be manufactured in a reliable way.
- No time-consuming factory adjustments.
- Pass CE requirements.

Design Considerations for Hardware.

We have chosen to manufacture the SG100 generator for 9-pin serial ports. Even if the PC-parallel/printer port provides higher communication speeds there could be problems when writing drivers for Win32 or UNIX-systems. As the noise process generates a serial stream of information any possible benefit of a parallel interface is clearly limited.

Only very little power can be obtained from a serial port. This has been a major construction problem, but now the device operates well even far out of the specification of the serial port. We have found, experimentally, that a number of PC-models have serial ports which operates below the RS-232C standard. The SG100 driver monitors power and if any power failure is detected this error is forwarded to the calling application.

The hardware has a power-off mode as well as an operating mode. When no more noise is needed the software driver will switch the SG100 to power-off mode. If your portable computer has automatic power saving (APM-hardware) this will work too.

Resistance against Radio Frequency (RF) fields has been in the specification from the beginning of our product development cycle. No customer has to fear that the operation of the SG100 generator can be intentionally influenced from any external RF-field. This has been accomplished by using a noise generating process with a high output level and by using a RF-shield casing. The SG100 generator has a built in RF-field filter.

The reader should note that all ordinary computers are somewhat sensitive to strong RF-fields. We estimate that almost all computer models will cease to operate when subject to radiation of a strength high enough to influence the SG100 generator.

The SG100 has a plastic casing which also protects against RF-fields. In the future we will also provide a metal casing for demanding customers with EMP/HEMP/TEMPEST protected computers and high security installations. Measurements of the actual levels of fields involved in this discussion will be published here, when available.

For a serial port the baudrate of the port must be specified. To overcome the fact that some computers don't support all baudrates, and to avoid specifying a low baudrate (like 9,600), we have choosen not to convert the output from the SG100 generator into byte-serial form before feeding it to the computer. The hardware outputs an irregular square wave that is feed into the computer UART which samples and converts the input stream into digital form. During this process the computer UART selects which parts of the noise stream that will be interpreted as start bits and which parts that will be ignored as stop bits. The decision of when to interpret the stream as "1" or "0" is up to the UART. You may use any baudrate below 100,000 that is accepted by the serial interface driver.

Not converting the SG100 output into byte-serial form simplifies hardware design and makes interception of the serial stream very difficult, as it is unknown what bits are selected as startbits and as the sampling in the UART is somewhat different on all computers.

Software Properties

- Windows-95 and Windows NT driver delivered with product.
- Easy to use API interface.
- Immediate action if the device fails.
- Fast response to the calling process.
- Interface for multiple processes reading noise.
- No cryptographic or statistical weaknesses.
- Do not deliver low quality noise when first called or if called repeatedly.
- Easy to include drivers/etc in OEM product.
- Source code access for OEM-customers.
- Drivers can be written for any platform.

Design Considerations for the Software Driver

A driver for Windows 95 and Windows NT is delivered with the SG100 noise generator. This is the same driver that is used by Protego InfoSafe line of security products. The API of the driver (*.h file) is included as well as compiled demo programs in C/C++. Using the provided compiled demo program you may extract noise to a named file to facilitate an easy interface with almost any statistical or security product.

The driver may be linked directly into the EXE of an OEM product. If the customer wish to modify the driver himself the source code will be made available on a non-disclosure basis.

It is also possible to use the SG100 generator on other (non PC) platforms. In that case a request can be made for the complete driver specifications and algorithms.

The provided SG100 driver DLL operates the hardware and checks for hardware errors. A wide range of hardware errors can be detected and action taken accordingly. If the device fails any pending call for noise will be returned with an error status. If the user disconnects the SG100 generator a signal is given to the calling application to let the user reinstall the noise generator without any influence on the running application.

The driver do a continuous statistical check of the hardware. The driver then calculates the rate of information obtained form the SG100 generator. If the rate falls below 70% the device is considered faulty. A rate above 96% is acceptable without further action and if a rate lower than 96% is obtained this will be remedied by reading the noise stream twice, giving the driver access to the double amount of input. Reading twice, if necessary, will give us only half overall throughput.

The hardware, built according to a minimal noise-processing criteria, cannot create complex output correlations by itself. We will have a statistical bit bias from the hardware, with "1" and "0" not occurring with exactly 50%-50% distribution, and also a correlation between adjacent bits (and a correlation between the start bit and the first bit). We see that there cannot be any high complex correlations because the hardware lacks memory. After a short period of time a previous noise stream cannot influence the current noise stream.

Suppose we read four bytes and form a 32 bit integer A. Suppose we know that the current information flow is 94%. We then read a second 32-bit integer B, at some other point in time, maybe a few seconds later. We now know that A and B are independent and random, and also that we have statistical deficiencies, as indicated above. To form a 32-bit integer C with almost 100% information rate it is sufficient to add A and B using 32-bit binary add (with carry) C=A+B. We may picture this by an arrow A pointing on the circumference of a circle divided into 2^32 segments. The selection of A has some bias, making not all the 2^32 segments equally probable. If we now turn the arrow B steps forward it is visualised that if the bits of A and B has the independence property, above, the number C will have surprisingly good statistical properties. Note that 2*94%=192%; the number C, consisting of 32 bits, clearly cannot store any more information than 32 bits (100%).

In this way, using a similar but more complex "adding", the driver can guarantee that the minimum information flow is 96%. Using a SG100 developer package the reader may obtain the raw SG100 output and verify this for himself.

The software driver has a buffer of noise ready for reading to facilitate fast response when called. Currently this has been set to 32.000 bytes, as no more is necessary for any of the InfoSafe products. The buffer also has an additional purpose, here is the pre-processed input from the driver "whitened" to enable the output stream to pass any statistical or cryptographic test. This is done using a combination of statistical and cryptographic techniques. This relieve the customer, almost regardless of application, from the need to further improve the SG100 driver output. The detailed cryptographic and statistical methods can be obtained by customers of the SG100 development package.

The software driver uses process synchronisation allowing multiple processes to read noise simultaneously. The synchronisation also blocks processes if the noise buffer becomes empty or during an initial start-up period before the noise buffer has been thoroughly "whitened". When the noise buffer becomes empty the driver reads tree times the buffer size to allow filling the buffer with a maximum information content. We have seen that the minimal information input rate is 96%. The reader should note that it is possible to read this minimum information content only if an application asks for a very long continuous string of noise, and that the 32.000 first bytes of that long string will most probably be of 100% quality. To experimentally obtain the (maximum) 4% lack of information, the output must be intentionally processed to break the cryptographic operations, which will be most difficult mainly because of the small statistical deficiency searched for. This is practically infeasible for the string lengths that the drivers has been designed for.

Some demanding customers will not accept even the most diminutive statistical deficiency. Solving this problem will, in addition to other steps, including a halving of output throughput, forcing the information content to 100% at all times. If asked for, a double SG100 interface can be made available.