

# ISA Card Random Number Generator

Aaron Logue Aug 2002

The purpose of this card is to provide a source of cryptographically secure random numbers to a host PC. The card can operate in either polled or interrupt mode. Some other pages here provide more detailed information on [ISA bus interfacing](#), getting started with [SX microcontrollers](#), and the theory behind the [random number generator](#) circuit.



Microcontroller source code: [isarng.asm](#)

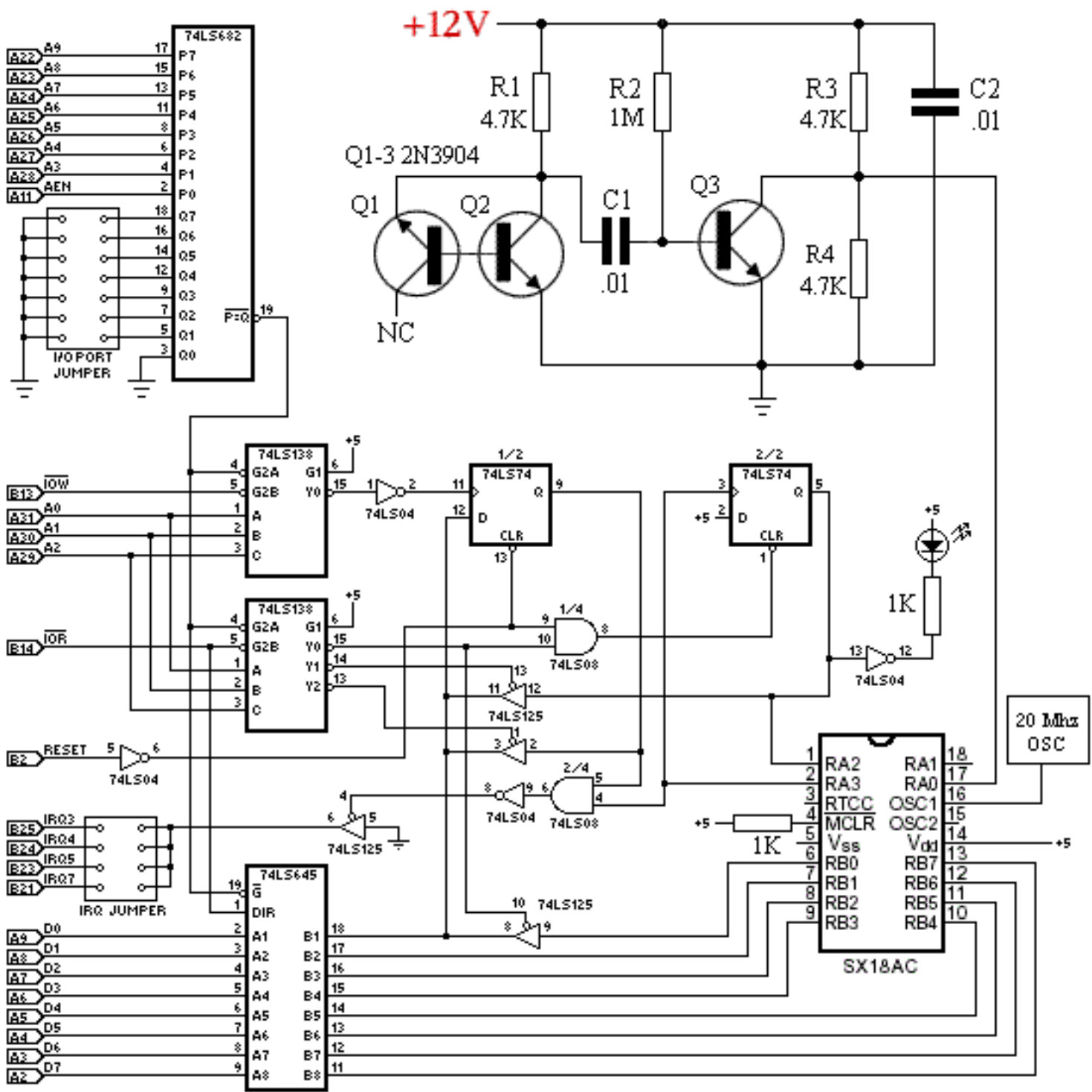
Hex file (20 Mhz oscillator): [isarng.hex](#)

Card exerciser executable for DOS: [rngtest.exe](#)

Exerciser source code: [rngtest.asm](#)

[Diehard](#) randomness test [results](#) using a 10M sample set

[DiehardC](#) randomness test [results](#) using same sample set



The PN junction of Q1 is reverse biased to produce avalanche noise. The resultant signal is amplified and connected to the RA0 pin on an SX microcontroller, where it is periodically sampled as an unpredictable source of noise. Random bits are derived from this noise source, and when a byte has been accumulated, the microcontroller checks the line state of RA2, and if it is low (indicating that the previous data byte has been read) then the byte is placed on the RB output pins and RA3 is toggled to indicate that a new byte is ready for transfer.

If interrupts are enabled, the toggling of RA3 pulls the ISA bus IRQ line low and then lets it float high

again, signalling an interrupt. On the rising edge of RA3, the 74LS74 2/2 flip flop sets its Q output high. The flip flop's Q high output indicates that a byte is available and that it has not been read. The bit can be picked up by a polling read in the least significant data bit D0, and can also be sensed by the SX microcontroller to see if it's okay to trigger another interrupt.

PC interrupts are triggered on the rising edge of the IRQ signal. To generate an interrupt, the card pulls the IRQ line low for at least 100 nanoseconds and then releases it, relying on pullup resistors on the motherboard to create the rising edge needed to signal the interrupt controller.

The card is jumperable to allow its IO port base address to be set to any address in the IO port range 000h to 3f8h on 8 port boundaries. 300h, 308h, or 310h are good candidates. Here are the card's ports:

OUT baseaddr+0 Enable or disable interrupts. Set the LSB to 1 to enable interrupts, or to 0 to disable interrupts.

IN baseaddr+0 Read a data byte from the card and clear the Byte Available bit in the status register.

If a byte is available at the time that interrupts are enabled, it will be necessary to perform this read and clear the currently buffered data byte in order to "prime" the interrupt pump.

IN baseaddr+1 Read data availability status bit. B7 B6 B5 B4 B3 B2 B1 B0

B0 - this bit is 1 if an unread byte is available.  
B1 - B7 are indeterminate.

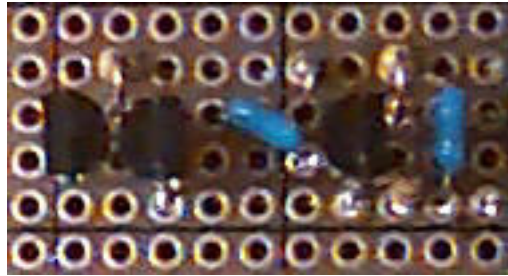
IN baseaddr+2 Read interrupt enable state  
B0 - this bit is 1 if interrupts are enabled.  
B1 - B7 are indeterminate.

It is possible to share interrupts with this card, provided that the interrupt handler is installed first in the chain, that it checks the status port to see whether or not this card generated the interrupt before reading the data byte, and that it jumps to the old handler if no byte is available (that is, if the card didn't generate the interrupt).

If building this card, I recommend first laying out the chip sockets with ground, power, and bypass capacitors, then wiring in the 74LS682 and 74LS138 address decoders. At that point, you can test the card and be sure that its address decoding is working properly; IN and OUT instructions to the different ports should yank on the correct 74LS138 output lines. Finish the card and plug in all chips except for the SX microcontroller, and test again to verify that the IN and OUT instructions can set and read the

interrupt enable flip flop. The LED should remain off, and the card should not generate any interrupts without the SX in place, even with interrupts enabled. An oscilloscope connected to the collector of Q3 should show white noise ranging from about 0 volts to about 4.5 volts. The card is then ready for the SX.

All the disclaimers you can think of regarding use in critical cryptography applications apply. Because this card uses the host PC's power supply, particular attention should be given to the possibility that signals present in the power supply may affect the bits derived from the transistor junction avalanche noise.



Jan 2004 addendum: While looking at the schematic trying to refresh my memory, I realized that one of the 74LS138s could be eliminated. There are two unused AND gates in the 74LS08.  $\sim P=Q$  and  $\sim IOW$  could be fed to an unused AND, and its output would work just like Y0, eliminating the write '138 address decoder chip.

SX18AC/DP chips are getting hard to come by. This circuit should work just fine with an SX28AC/DP, and the isarnng.asm code should work fine too. The only change needed would be to the FUSEX PIN bit in mydefs.inc