

UDPi April FAQs

1) Does UDPi offer “data protection” for stolen data?

Yes. UDPi takes the lead and ensures data is secured at the hardware level with a flexible circuit solution. This technology is not just about data recovery and data loss monitoring, but also takes into account protecting data at every internal, external and peripheral point.

2) What is UDPi’s flexible circuit solution?

UDPi’s security is based around a flexible circuit, which is a variable circuit. Hardware is not generally hacked, but rather the software that uses the hardware can be vulnerable to hacking. Since UDPi’s flexible circuit resides in the hardware portion of a processor, it is not vulnerable to hacking.

For example, if one were to try to hack the physical electrical connection to their house, it would be nearly impossible. To find the circuit, one would have to destroy the walls of the house. However, if the electrical wiring circuit changes depending on which devices were active in the house, there would be too many variables.

The above example is what UDPi does for micros, depending on the code executed (thus its location), the codec (coding – decoding) circuit changes.

3) UDPi is located between the Instruction Fetcher and Destruction Decoder. What are the various implementation choices for the security?

The UDPi solution can be incorporated during the manufacturing process as a simple circuit, but can also be implemented in conjunction with a LFSR, XOR, or non-destructive alteration method to encode data.

4) Can other protection technologies that use software be added on top of UDPi?

Yes, other technologies can be used in conjunction with UDPi. This is primarily because UDPi is truly a hardware-based method, and does not require software-based formulas or heavy-duty number crunching. UDPi protects data at the hardware level, for example on hard disks as well as the microprocessor code and IP.

Software-based solutions can be very desirable for a number of applications and in a number of different markets. UDPi recognizes many companies like to use software for specific security applications, and UDPi is certainly compatible with the use of these solutions.

5) How does the UDPi key work? Is it similar to have execute only (no read) code?

UDPi hides data using pseudo-random block lengths and key lengths. The “key” is really a flexible circuit, as described in Question 2.

6) How does UDPi incorporate XOR functions?

UDPi adds complexity to basic XOR implementation. One example is to have a table of keys relating to a specific location in ROM. Key A is used for some portion of the ROM and Key B is used for another portion, etc...The key can change per block and the block size can change per key with UDPi’s method.

7) Can UDPi’s security be pseudo-random and what is the result?

Yes, UDPi can use pseudo-randomness to get a very long key that is secure. This technique makes sure that a character is not used twice in a key, adding security and not repeatability.

It is possible to use two LFSRs to generate a secure pseudo-random table of keys. UDPi can execute code incredibly fast, pairing high security with speed.