Page Inventory Page - type 0x02

Every database has at least one Page Inventory Page (PIP) with the first one always being page 1, just after the database header page. If more are required, the current PIP points to the next PIP by way of the very last bit on the page itself. The C code representation of the PIP page is:

```
struct page_inv_page
{
   pag pip_header;
   SLONG pip_min;
   UCHAR pip_bits[1];
};
```

Pip_header: The PIP starts off with a standard page header.

Pip_min: Four bytes, signed. Bytes 0x10 - 0x13 on the page. This is the bit number of the first page, on this PIP, which is currently free for use.

Pip_bits: Bytes 0x14 onwards. The remainder of the page, is an array of single bits where each bit represents a page in the database. If the bit is set (1) then that page is free for use. If the bit is unset (0) then the page has been used.

If the database is large, and requires another PIP elsewhere in the database, then the last bit on this PIP represents the page number for the next PIP. For example, on a 4,096 byte page we have a total of 4,076 bytes to represent different pages in the database. As each byte has 8 bits, we have a total of 32,608 pages before we need a new PIP.

In a brand new database, a hex dump of the first few bytes of page 1, the first PIP, looks like the following:

```
Offset     Data                                                  Description
-------------------------------------------------------------------------------
------
00001000   02 00 39 30 31 00 00 00   00 00 00 00 a1 00 00 00  Standard
Header
00001010   a1 00 00 00                                        pip_min (low
endian)
00001014   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  pip_bits[]
00001024   00 00 00 00 fe ff ff ff   ff ff ff ff ff ff ff ff
```

In the above, we see that pip_min has the value 0x000000a1 and the following 20 bytes, the first part of the pip_bits array, are all zero. From this, it would appear that page 0xa1 is the first available page in the database for user tables etc. and that all the pages up to that one have already been used for the system tables and indices etc.

Looking at the bitmap again, page 0xa1 will be represented by byte 0x14, bit 0x01 of the bitmap. This is byte 0x00001028 bit 1. We can see that this byte currently has the value 0xfe and bit 0x00 is already in use. So, our array is correct and so is our pip_min value - the next available page is indeed 0xa1.

If we look at the hexdump of that particular page, at address 0x000a1000, we see that it is actually the first byte past the current end of file, so our brand new blank database has been created with just enough space to hold all the system tables and indexes and nothing else.