

TABLE 2-continued

Logical Description of General Purpose IO (GPIO) Signal #4 (Block of bits only)			
Section/Bit Offset	# bits	Purpose	Sample Value
31:30	2	GPIO SignalType	2
29:29	1	GPIO Bus Flag (1 = true, 0 = false)	0
28:25	4	GPIO Bus Bits (# of bit-banged bus bits)	0x1
24:17	8	GPIO Bus Port Number	0x00
16:9	8	GPIO Bus Bit Location (register + pin)	0x00
8:4	5	GPIO Bus Bit Start Offset	00000b
3:3	1	Direction (0 = input/read, 1 = output/drive)	1b
2:2	1	Negative Logic (reverse true/false)	1b
1:1	1	Initial Value (true/false, subject to - logic)	1b
0:0	1	Open Drain (true/false, subject to - logic)	1b

[0089] Table 1 illustrates the block of bits parameters that described a certain Power Supply. That Power Supply used a number of logical input/output signals. These were called general purpose input/output (GPIO) signals. Table 2 illustrates how one of these logical GPIO signals, GPIO Signal #4, can itself be described. Table 2, includes a reference to a logical GPIO pin (as opposed to the purely logical GPIO signal).

[0090] The logical description (scheme) for Power Supply depends upon one or more logical descriptions (scheme instances) for GPIO signals. There are a number of scheme-to-scheme dependency/delegation relationships described by this example. Notice that some of the physical aspects of a GPIO pin are also described in the block of bits shown in Table 2. These physical, GPIO-pin aspects are interleaved with the logical GPIO-signal description (in the block of bits shown in Table 2). For instance, the Negative Logic, Initial Value and Open Drain attributes are really physical attributes of a GPIO pin (and not logical aspects of the logical GPIO signal).

[0091] While this sort of logical/physical interleaving is often done (for various reasons), it does not change the nature of this inventions logical versus physical description pairing. Nor does it change the conceptual separation of the two. For instance, the logical GPIO pin number shown in Table 2 (as 0x60) is mapped/associated with the physical ASIC pin number (which happens to be 78 in this particular exemplary implementation). There is a one-to-one, identity mapping between the logical GPIO-signal description and the physical GPIO-pin description. Just where various GPIO pin/signal attributes appear is irrelevant across such a one-to-one (identity) mapping. This interleaving of descriptive aspects is isomorphic with a strict logical-physical separation.

[0092] Consequently, other factors dominate the implementation choice between interleaving (various descriptive aspects) and separation (of various descriptive aspects). For instance, different specific makes/models of a power supply might require different GPIO signal pulse widths. While an electrical signal pulse width is surely a physical aspect, this

may be conveniently associated with the distinct logical power supply descriptions. It needs to be acquired from there into the physical description actually used. One may not want to set a single, static electrical signal pulse width for the pin. Although rare, different logical GPIO signals could be distinguished by the pulse width alone.

[0093] While this invention embraces all such refinements, the basic result remains—a logical description pairs with a physical description to form a complete declarative description of a given peripheral device (like a power supply).

[0094] Those of skill in the art will immediately realize that schemes for other sorts of peripheral devices, like fans, will all work in an analogous fashion. Fans, e.g., will use certain bits within the block of bits to identify which logical GPIO signals provide tachometer signals, which logical pulse width modulation (PWM) circuits to use (for a given fan instance) and so on. As with the power supply example, logical GPIO signals must then be mapped to physical GPIO pins (or to bit banded buses).

[0095] Ultimately, the physical description, appropriate for each instance of a logical scheme, is unambiguously referenced by the logical description values. Most often, the physical description to use (for any given logically described peripheral instance) is determined from the block of bits.

#### Host Hardware Environments

[0096] In some aspects, using the invention's approaches to declaratively describing the peripherals which constitute the hardware environment, this invention provides for declaratively describing a host hardware environment. For example, many embedded firmware/software hosts are essentially ASICs that embed many special purpose circuits. These integrated circuits might implement, e.g., a real-time clock, a random number generator, a UART (Universal Asynchronous Receiver Transmitter) or many other such functions. Each of these integrated sub-circuits can be treated as a specific sub-type of peripheral. In this case, the logical and physical descriptions according to embodiments of the present invention apply as if these circuits were just another peripheral scheme.

[0097] As with peripherals devices, the absence of some ASIC circuits, in some specific host hardware environment (for embedded software/firmware), will parallel the absence of some corresponding peripheral device. The same sort of (declarative) descriptive technique can indicate such an absence (for either the host ASIC circuit or for some equivalent peripheral device).

[0098] In practice, a logical scheme is introduced for things like non-volatile storage. As before, these logical descriptions are then mapped to corresponding physical description(s). For instance, one physical description can put non-volatile storage on the host ASIC while another physical description can put non-volatile storage on some off-chip flash memory part. Either way, it is logically just non-volatile storage. All such logically non-volatile storage components will share certain facets. The fixed-image, embedded software/firmware, according to embodiments of the present invention will manipulate each instance of non-volatile storage via the same logical non-volatile storage facet (interface). Some portion of the fixed-image, embedded software/firmware will be parameterized by the