Open Firmware

Recommended Practice:

Interrupt Mapping

Version 0.5

2/29/96

Published by the Open Firmware Working Group

Revision History

1. Introduction

This recommended practice defines a mapping mechanism between bus-specific interrupt values, as reported via the `"interrupts"` property of the bus's child nodes, and a system platform's "native" interrupt facility.

1.1. Purpose

The base Open Firmware specification (IEEE Std 1275-1994) defines a generic bus-specific property (`"interrupts"`) that is used to specify the interrupt "levels" used by that bus's devices. Each bus binding to Open Firmware must define the format and interpretation of its `"interrupts"` properties.

However, there is no mechanism for determining how these bus-specific interrupts eventually get reported to the interrupt reporting hardware on a particular platform. This recommended practice defines such a mechanism.

1.2. Scope

2. References and Definitions

2.1. References

2.2. Definitions

3. The interrupt reporting model

The basic new concept is to define an *interrupt tree* that represents the interrupt hierarchy of the platform. Each interrupt nexus in the interrupt tree represents where some mapping of `"interrupts"` property values must be done. At the top of the interrupt tree, a platform-specific interrupt value is produced; each platform binding must define the interpretation of this value (e.g., source number for Open PIC).

Since the interrupt tree may not match the bus tree, which is what the Open Firmware device tree represents, a new property is introduced that denotes the interrupt tree ordering from devices upwards (`"interrupt-parent"`). If a node does not contain an explicit parent (i.e., does not the "interrupt-parent" property), that node's parent in the device tree is assumed to be the interrupt parent.

At each level in the interrupt tree, a mapping may need to take place between the child interrupt space and the parent's. This is represented by a new property called `"interrupts-map"`. This property defines the mapping of interrupt "levels", as reported in the `"interrupts"` property of interrupt children. The `"interrupts-map"` property can represent wiring conventions (e.g., PCI cards with on-board PCI-to-PCI bridges, or platform routing of PCI interrupt pins) or more complicated mappings (e.g., VME interrupt controller to PCI interrupts).

The `"interrupts-map"` property is a table that maps from interrupt values in the child's interrupt space to values in the parent's. In order to interpret the `"interrupts-map"` property, the number of cells required for the child space and for the parent space must be defined. Each interrupt nexus node (i.e., a node with an `"interrupts-map"` property) must define the `"#interrupt-cells"` property that defines the number for cells required for its children. The `"#interrupt-cells"` property of the interrupt nexus node's parent defines the number of cells in the parent’s interrupt space.

A client can determine the platform’s ultimate interrupt value for a given device by traversing the tree from a device node upwards in the interrupt tree, using the information provided by the `"interrupts-map"`properties along the way until the top of the interrupt tree is reached.

If the `"interrupts-map"`property is missing, or if a corresponding entry is not found, the mapping is assumed to be the identity mapping. In addition, a bus binding may define "wild-card" values for components of its child interrupt components.

The definition of the `"interrupts-map"`property is a "template"; bus bindings define the details for that bus’s bus nodes.

4. Interrupt nexus properties

The following properties are defined for interrupt nexus nodes by this recommended practice.

`"interrupts-map"` S

Standard *property-name* to define the interrupt mappings.

prop-encoded-array:
Arbitrary number of interrupt mapping entries.

Each mapping entry contains an interrupt specifier in the format of the child node’s and the corresponding interrupt specifier in the form of the parent interrupt space.

`"#interrupt-cells"` S

Standard *property-name* to define the number of cells in its child interrupt space.

prop-encoded-array:
An integer, encoded as with encode-int, that denotes the number of cells required to represent an interrupt specifier in its child nodes.

5. Interrupt child properties

The following properties are defined for children of an interrupt nexus node. Note that since an interrupt nexus node may be both the child of one interrupt space and the parent of another, it must have both sets of properties.

`"interrupt-parent"` S

Standard *property-name* to denote the interrupt tree parent of this node.

prop-encoded-array:
An integer, encoded as with encode-int, that is the *phandle* of the interrupt nexus node that is the parent (within the interrupt tree) of this node.

6. Examples

6.1. PCI bus

The `"interrupts-map"`for a PCI bus node has the format:

```
    child-interrupt child-dev#  parent-interrupt
```

(each encoded as with **`encode-int`**).

Where the *child-dev#* is a child's device number value from its config-address, and the *child-interrupt* is the value reported in the child's `"interrupts"`property. The *parent-interrupt* is the value corresponding to the bus node's mapping of that interrupt.

For a PCI host bridge, the *parent-interrupt* would (typically) be the platforms "native" interrupt value. For a PCI-to-PCI bridge on a plug-in card, the *parent-interrupt* would be the interrupt pin to which that child's interrupt is connected and the PCI device# of the bridge.

Note: this requires a change to the PCI Binding to change the representation of the `"interrupts"`property. The current definition is a single cell, containing a copy of the interrupt-pin registers. The new definition adds the device# of the device as another int, thus requiring 2 cells to represent an interrrupt.

6.2. CHRP platform

The native interrupt representation is defined to be the Open PIC interrupt source number if the value is positive. A negative value is interpreted as an 8259-style interrupt whose value is the 8259 interrupt level minus 16; i.e., an 8259 level 0 interrupt is -16 and interrupt level 15 is -1. This is to allow Open PIC interrupt source 0 to be used as a normal interrupt source in the future.

Thus, the device tree node of the Open PIC interrupt controller should be defined as the "root" of the interrupt tree. Its `"#interrupt-cells"`value would be 1.

The PCI host bridge would be a logical place to perform the mapping between PCI `"interrupts"` properties and the platform. It would have an `"#interrupt-cells"`value of 2 and would have an `"interrupts-map"`property to devine the mapping of PCI "interrupts" to Open PIC interrupts.