SpartanMC

Hardware parameters header file

Table of Contents

List of Figures

List of Tables

MANPAGE – HARDWARE.H(3)

NAME

hardware.h – Header file populating hardware implementation parameters for low level hardware access

SYNOPSIS

#include <system/hardware.h>

DESCRIPTION

The project specific generated header file **hardware.h** populates a number of key-value pairs reflecting all synthesis parameters passed to the hardware implementation process. This allows the firmware to be aware of certain features and parameters concerning the hardware platform it runs on.

The actual keys available depend on the system configuration specified in the system builder *jConfig*. Basically, each value choosen at the tab *Parameters* of each hardware component is mapped to a **#define** using the form

#define <KEY> <VALUE>.

Refer to the hardware documentation for details about the actual parameters defined by a certain hardware component.

KEYS

The list below gives an overview about the general format of available keys populated by **hardware.h**.

SB_<instance_name>_<parameter_name>

Value of hardware parameter <parameter_name> of module instance <instance_name>. E.g., the base address (parameter *BASE_ADR*) of module *uart_0* would be **SB_UART_0_BASE_ADR**.

SBI_REGION_<instance_name>_<suffix>

Provides information about the address space occupied by a certain hardware module. This only applies to processor instances and peripheral modules implementing DMA. **<suffix>** can be one of **MIN_ADDR**, **MAX_ADDR** or **BYTES** respectively giving the lower

or upper address boundaries or the number of bytes occupied by the components memory.

SBI_VERSION System builder version, which is currently 2.

SBI_CORE_ID 18-bit hexadecimal checksum of the current hardware design. Used to match a given firmware binary against a certain hardware design when using the bootloader (see *spmc-loader*(1)).

i_bits The number of interrupt lines provided by the interrupt controller (*intctrl* or *intctrl_p*), if any. When no interrupt controller is present, the value for **i_bits** is 0.

VALUES

The form a value is represented depends on the parameters value type as defined by the system builder *jConfig* or read from the respective module description. All values are in fact mapped to integer constants. To deal with float and string values, symbolic constants are used.

Integer values

Decimal and hexadecimal integer values are represented straight forward as shown in *jConfig* (e.g. **23**, **0x42**). Binary numbers are represented using their respective hexadecimal notation.

Float values

Float value parameters defined by the system builder are represented by symbolic constants of the form **SBFLOAT_<int>_<frac>**. E.g, the float number **2.68** woule become **SBFLOAT_2_68**. Note that this technique only allows for test of equality regarding a certain parameter. Performing real float arithmetics is not possible, which rather is limited by the fact that the SpartanMC currently does not support floating point in any way.

Boolean values

Boolean values are represented by integers of value **0** or **1** respectively standing for **false** or **true**. Constants of the form **SBBOOL_...** map the symbolic representation for each boolean parameter to their respective numeric values **0** or **1**. E.g., a boolean parameter with the symbolic meaning of **YES** or **NO** will provide the constants **#define SBBOOL_YES 1** and **#define SBBOOL_NO 0**. This allows you to use symbolic constants similar to as shown in the system builder when testing for values of boolean parameters.

String values

String values are mapped to symbolic constants of the form **SBSTRING_<value>**, where **<value>** is replaced by the original string value in upper case. All characters not allowed in a constant name are replaced by a underscore (_). E.g., the value of parameter VENDOR_STRING = "TU

Dresden" of component *usb11_0* would become **#define SB_USB11_VENDOR_STRING SBSTRING_TU_DRESDEN**. The symbolic constant **SBSTRING_TU_DRESDEN** is mapped to an arbitrary unique hexadecimal value.

FILES

<project_dir>/system/hardware.h	Header file to include for access to hardware parameters
<project_dir>/system/<subsystem_name>/hardware.h	Actual header file defining hardware parameters for the respective subsystem. Included by **hardware.h** depending on the actual subsystem the firmware is built for.

SEE ALSO

peripherals.h(3)

AUTHORS

Written by Markus Vogt