VerilogBoy Handheld

Reference Manual

Hardware and Software Interface

Wenting Zhang

December 1, 2018

DRAFT

Preface

This documentation is mainly about the internal design of VerilogBoy Handheld. This document is intended to serve as a reference manual for developers. This is not a tutorial or a user manual or a service manual.

How to read this document

This is a warning about something.

0.1 Formatting of numbers

When a single bit is discussed in isolation, the value looks like this: `0`, `1`.

Binary numbers are prefixed with `0b` like this: `0b0101101`, `0b11011`, `0b00000000`. Values are prefixed with zeroes when necessary, so the total number of digits always matches the number of digits in the value.

Hexadecimal numbers are prefixed with `0x` like this: `0x1234`, `0xDEADBEEF`, `0xFF04`. Values are prefixed with zeroes when necessary, so the total number of characters always matches the number of nibbles in the value.

Examples:

	4-bit	8-bit	16-bit
Binary	`0b0101`	`0b10100101`	`0b0000101010100101`
Hexadecimal	`0x5`	`0xA5`	`0x0AA5`

0.2 Register definitions

Register 0.1: `0x1234` - This is a hardware register definition

R/W-0	R/W-1	U-1	R-0	R-1	R-x	W-1	U-0
VALUE<1:0>		–	BIGVAL<7:5>			FLAG	–
bit 7	6	5	4	3	2	1	bit 0

Top row legend:

R	Bit can be read.
W	Bit can be written. If the bit cannot be read, reading returns a constant value defined in the bit list of the register in question.
U	Unimplemented bit. Writing has no effect, and reading returns a constant value defined in the bit list of the register in question.
-n	Value after system reset: `0`, `1`, or x.
`1`	Bit is set.
`0`	Bit is cleared.
x	Bit is unknown (e.g. depends on external things such as user input).

Middle row legend:

VALUE<1:0>	Bits 1 and 0 of VALUE
–	Unimplemented bit
BIGVAL<7:5>	Bits 7, 6, 5 of BIGVAL
FLAG	Single-bit value FLAG

In this example:

- After system reset, VALUE is `0b01`, BIGVAL is either `0b010` or `0b011`, FLAG is `0b1`.
- Bits 5 and 0 are unimplemented. Bit 5 always returns `1`, and bit 0 always returns `0`.
- Both bits of VALUE can be read and written. When this register is written, bit 7 of the written value goes to bit 1 of VALUE.
- FLAG can only be written to, so reads return a value that is defined elsewhere.
- BIGVAL cannot be written to. Only bits 5-7 of BIGVAL are defined here, so look elsewhere for the low bits 0-4.

Contents

Part I

VerilogBoy Hardware Architecture

TODO

Part II

VerilogBoy MM Peripherials

Chapter 1

MIPI DSI controller (DSIC)

1.1 DSIC introduction

The display serial interface (DSI) is part of a group of communication protocols defined by the MIPI®Alliance. The DSIC implements a MIPI®DSI host controller and MIPI®D-PHY to provide an interface between the system and a DSI-compliant display.

Since the DSI specification is non-public and requires an NDA, the core was built using bits and pieces available throughout the Web: presentations, display controller/SOC datasheets, various application notes and Android kernel drivers. The author is not associated in any way with the MIPI Alliance. The core has never been verified for compliance with the DSI standard and it probably lacks many of its features.

1.2 DSIC main features

- Implement MIPI®D-PHY
- Transmission of command mode packets through the MMPB interface
- Transmission of video mode packets through the simple pixel FIFO interface
- Support up to one D-PHY data lanes by default
- Uni-direction transmission only
- Support non-continuous clock in D-PHY clock lane
- ECC and checksum capabilities
- Support 24-bit RGB mode

1.3 DSIC registers

Register 1.1: `0x10` - DSIC_CTL - MIPI DSI Control

U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
–	–	–	–	–	–	LP_REQ	CLK_EN
bit 7	6	5	4	3	2	1	bit 0

bit 7-2 **Unimplemented**: Read as `0`

bit 1 **LP_REQ**: Request to switch to LP mode
`0b1`= ??
`0b0`= ??

bit 0 **CLK_EN**: DSI clock lane output enable bit
`0b1`= DSI clock output is enabled
`0b0`= DSI clock output is disabled

Register 1.2: `0x11` - DSIC_TXDR - DSI LP Mode Transmission Data Register

`R/W-0`	`R/W-0`	`R/W-0`	`R/W-0`	`R/W-0`	`R/W-0`	`R/W-0`	`R/W-0`
`TXDR<7:0>`							
bit 7	6	5	4	3	2	1	bit 0

bit 7-0 **TXDR**: Raw byte to be sent in the LP mode.

Writing to this register will trigger the transmission immediately.

Appendices