OSOROM Peripheral Reference

The Moroso Project

July 3, 2016

Contents

Chapter 1

Onboard Peripherals

1.1 Introduction

The OSOROM will include several onboard peripherals:

- A programmable timer
- A serial port
- A video controller
- An SD card controller
- A USB controller?

Access to all peripherals is through memory mapped registers; each peripheral has its own page of physical address space for control registers.

1.2 Peripheral Reference

1.2.1 Register Map

Timer	`TIMER_COUNT`	0x80000000
	`TIMER_TOP`	0x80000004
	`TIMER_CONTROL`	0x80000008
Serial Port	`SERIAL_BAUD`	0x80001000
	`SERIAL_DATA`	0x80001004
	`SERIAL_CONTROL`	0x80001008
	`SERIAL_STATUS`	0x8000100c
Video Controller		0x80002000
SD Controller		0x80003000
USB Controller		0x80004000

1.3 Timer

Each clock tick, the `TIMER_COUNT` register is incremented. If this value matches the value in the `TIMER_TOP` register, `TIMER_COUNT` is cleared and the `TIMER_INT` bit in `TIMER_CONTROL` is set.

1.3.1 Registers

TIMER_COUNT

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMER_COUNT															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER_COUNT															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:0 `TIMER_COUNT`: The current count, incremented each tick and reset to 0 when it matches the value in `TIMER_TOP`.

TIMER_TOP

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TIMER_TOP															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIMER_TOP															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:0 `TIMER_TOP`: The top value for the timer. When `TIMER_COUNT` is equal to this value, `TIMER_COUNT` will be reset to 0 and the `TIMER_INT` bit of `TIMER_CONTROL` will be set.

TIMER_CONTROL

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
(reserved)															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(reserved)													TIMER_EN	TIMER_INT_EN	TIMER_INT
													rw	rw	rw

Bit 2 `TIMER_EN`: Timer enable bit. The counter will be paused if this bit is set to 0.

Bit 1 `TIMER_INT_EN`: Timer interrupt enable. If this bit and `TIMER_INT` are both set, a timer interrupt will be generated.

Bit 0 `TIMER_INT`: Timer interrupt flag. Set by hardware when `TIMER_COUNT` is equal to `TIMER_TOP`.

1.4 Serial Port

The serial port contains two internal 8-bit registers, the receive and transmit buffer registers. These cannot be accessed directly, but instead are accessed through the `DATA` register.

When a value is written to the `DATA` register, it is written through to the transmit buffer. If a transmission is not in progress, a transmission will begin with the written value, and the `TX_EMPTY` bit will be set to indicate that the transmit buffer is ready for another write. If a value is written to `DATA` and a transmission is in progress, the value will be written to the transmit buffer and the `TX_EMPTY` bit will be cleared. When the transmission ends, if `TX_EMPTY` is clear, a new transmission will begin with the value in the transmit buffer, and the `TX_EMPTY` and `TX_COMPLETE` bits will be set.

Reads from the `DATA` register read the value in the receive buffer. When a byte is received, the value is stored in this buffer and the `RX_COMPLETE` bit is set.

Bits in the status register can be cleared by writing a 1 to them. (Clears are done with 1s rather than 0s to avoid potential race conditions.)

SERIAL_BAUD

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
`SERIAL_BAUD`															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
`SERIAL_BAUD`															
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:0 `SERIAL_BAUD`: A value that will determine the baud rate according to some formula we'll figure out later.

SERIAL_DATA

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
(reserved)															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(reserved)							`P`	`DATA`							
							rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit 8 `P`: Reserved for use as a parity bit. Reads as 0.

Bits 7:0 `DATA`: Data register. See description above.

SERIAL_CONTROL

31	···	20	19	18	17	16
(reserved)						

15	···	4	3	2	1	0
(reserved)			`RX_ENABLE`	`RX_COMPLETE_IE`	`TX_EMPTY_IE`	`TX_COMPLETE_IE`
			rw	rw	rw	rw

Bit 3 `RX_ENABLE`: Enable UART receiver.

Bit 2 `RXC_COMPLETE_IE`: Interrupt enable for `RX_COMPLETE`.

Bit 1 `TX_EMPTY_IE`: Interrupt enable for `TX_EMPTY`.

Bit 0 `TX_COMPLETE_IE`: Interrupt enable for `TX_COMPLETE`.

SERIAL_STATUS

31	···	20	19	18	17	16
(reserved)						

15	···	4	3	2	1	0
(reserved)			RX_ERROR	RX_COMPLETE	TX_EMPTY	TX_COMPLETE
			rw	rw	r	rw

Bit 3 `RX_ERROR`: Set by hardware when an error occurs when receiving a byte (e.g. missing stop bit). Cleared by writing a 1.

Bit 2 `RX_COMPLETE`: Set by hardware when a receive is complete. Should be cleared by software when the value in `DATA` is read by writing a 1 to this bit.

Bit 1 `TX_EMPTY`: Cleared by hardware when a value is written to `DATA` and a transmission is in progress. Set by hardware when a transmission begins.

Bit 0 `TX_COMPLETE`: Set by hardware when a transmission completes. Can be cleared by software by writing a 1 to this bit.

1.5 Video Controller

1.6 SD Card Controller

1.7 USB Controller