# OSOROM Peripheral Reference

The Moroso Project

July 21, 2020

# Contents

	poard Peripherals	2
1.1	Introduction	2
1.2	Peripheral Reference	2
		4
2.1	LED/switch/button interface	4
	Serial Port	
	Timer	
2.4	Video Controller	8
2.5	SD Card Controller	8
2.6	USB Controller	8

## Chapter 1

# **Onboard Peripherals**

#### 1.1 Introduction

The OSOROM includes several onboard peripherals:

- LED/button/switch IO
- A serial port
- An I<sup>2</sup>C controller
- An SD card controller
- An audio controller
- A video controller
- An interrupt controller

and will likely eventually have:

- A programmable timer
- 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 Peripheral Map

Each peripheral instance has an index; that index is used to determine the offset of its page of memory mapped registers, and is also used in the interrupt controller to enable or disable interrupts from that peripheral instance. The peripheral instances and their indices are:

Index	Name	Description
0	LEDSW	LEDs and switches
1	UART	Serial UART
2	I2C	I <sup>2</sup> C controller
3	SD	SD card controller
4	AUDIO	Audio controller
5	VIDEO	Video controller
6	INT	Interrupt controller
7	USB	USB controller (reserved)
8-15	TIMO-TIM7	Timers (reserved)

The base address for the memory mapped registers of a peripheral is the peripheral's index, in pages, from the start of the peripheral region (0x80000000). So, for example, interrupt registers (peripheral index 6) begin at 0x80006000.

### Chapter 2

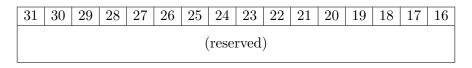
## Peripherals

### 2.1 LED/switch/button interface

The Moroso board has sixteen LEDs (eight green and eight red), as well as four buttons and ten switches; all are accessed through the LEDSW peripheral. Interrupts can be configured on state changes of any button or switch. In the future, more control over interrupts (such as the ability to trigger on a rising/falling edge) may be added.

#### 2.1.1 Registers

#### 0x0: LEDS



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

Bits 15:8 LEDS\_R: The value of the red LEDs (MSB corresponding to the leftmost red LED).

Bits 7:0 LEDS\_G: The value of the green LEDs (MSB corresponding to the leftmost green LED).

#### 0x4: SW\_BUTTONS

31	3     30   90   98   97   96						24	23	22	21	20	19	18	17	16
(nogowyod)									,	TIWE	CHES	3			
(reserved)					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

15												3	2	1	0
	(nocowyod)													CONS	
	(reserved)												rw	rw	rw

Bits 25:16 SWITCHES: The value of each switch (MSB corresponding to leftmost switch).

Bits 3:0 BUTTONS: The value of each button (MSB corresponding to the leftmost button).

#### 2.1.2 Register map

Offset	Register name
0x0	LEDS
0x4	SW_BUTTONS

#### 2.2 Serial Port

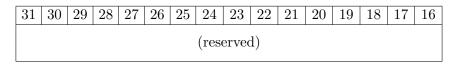
The serial port contains two internal 8-bit registers, the receive and transmit buffer registers. These cannot be accessed directly, but instead are accessed through the DATA register.

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.

#### 0x0: DATA



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

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

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

0x4: CSR

31   · · ·	21	20	19	18	17	16
			(rese	rved)		

	15		5	4	3	2	1	0
	(ro	serve	ad)	RX_ERROR	RX_COMPLETE	RX_ENABLE	TX_EMPTY	TX_COMPLETE
L	(16	SCI V	-u)	rw	rw	rw	r	rw

Bit 4 RX\_ERROR: Set by hardware on receive error.

Bit 3 RX\_COMPLETE: Set by hardware when a byte is received.

Bit 2 RX\_ENABLE: Enable UART receiver.

Bit 1 TX\_EMPTY: Indicates whether the TX buffer is empty. Set and cleared by hardware.

Bit 0 TX\_COMPLETE: Set by hardware when a byte transmission finishes.

#### 0x8: BAUD (not yet implemented)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
						SE	RIA	L_BA	UD						
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: Determines the baud rate as a function of the CPU clock:

Baud rate = 
$$\frac{f_{CPU}}{SERIAL\_BAUD + 1}$$

#### 2.2.1 Register map

Offset	Register name
0x0	DATA
0x4	CSR

### 2.3 Timer

Each clock tick, the COUNT register is incremented. If this value matches the value in the TOP register, COUNT is cleared and an interrupt is signaled.

#### 2.3.1 Registers

0x0: 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

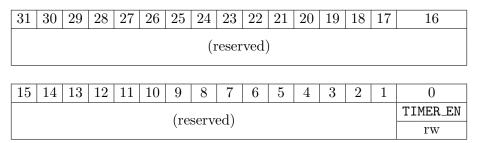
Bits 31:0 TIMER\_COUNT: The current count, incremented each tick and reset to 0 when it matches the value in TIMER\_TOP.

0x4: **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 an interrupt will be signaled.

#### 0x8: CONTROL



Bit 0 TIMER\_EN: Timer enable bit. The counter will be paused if this bit is set to 0.

#### 2.3.2 Register map

Offset	Register name						
0x0	COUNT						
0x4	TOP						
0x8	CONTROL						

- 2.4 Video Controller
- 2.5 SD Card Controller
- 2.6 USB Controller