

**PulseRain**  
TECHNOLOGY

**Doc# TRM-0922-01004, Rev 1.0.0**

Copyright © 2017

PulseRain Technology, LLC.

10555 Scripps Trl, San Diego, CA 92131



858-877-3485



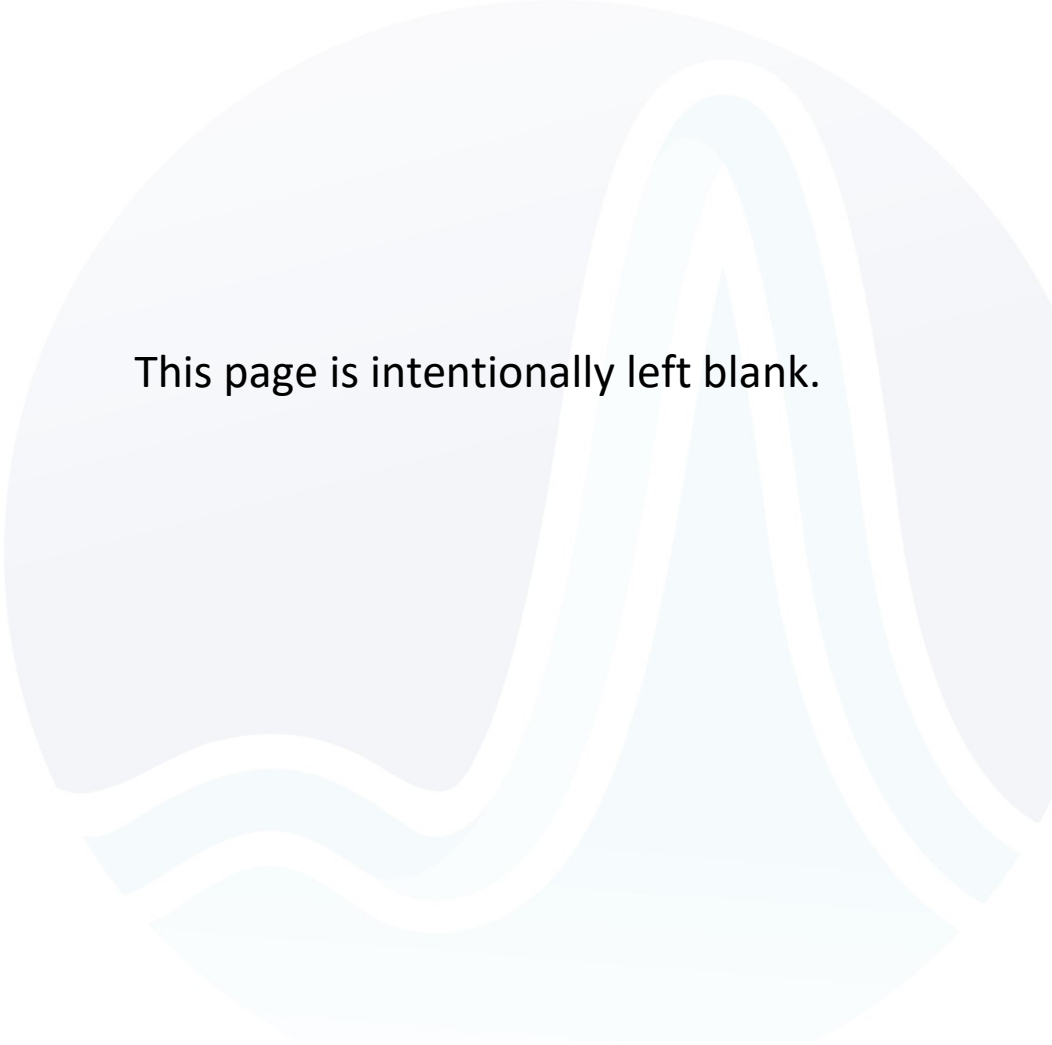
858-408-9550

<http://www.pulserain.com>

# PulseRain M10 – SRAM

## Technical Reference Manual

Sep, 2017



This page is intentionally left blank.

# Table of Contents

---

<b>REFERENCES.....</b>	<b>1</b>
<b>1 INTRODUCTION .....</b>	<b>2</b>
<b>2 HARDWARE .....</b>	<b>3</b>
2.1 ARCHITECTURE .....	3
2.2 PORT LIST.....	4
2.3 PIN ASSIGNMENT.....	4
2.4 REPOSITORY.....	4
<b>3 SOFTWARE .....</b>	<b>5</b>
3.1 REGISTER DEFINITION .....	5
3.2 ADDRESS MAP .....	6
3.3 WORK FLOW.....	6
3.4 ARDUINO LIBRARY.....	7
3.4.1 APIs.....	7
3.4.2 Examples.....	7

---

## References

1. MicroChip 23A1024/23LC1024 1Mbit SPI Serial SRAM with SDI and SQI Interface Datasheet, DS20005142C
2. The schematic of PulseRain M10 board, Doc# SH-0922-0039, Rev 1.0, 02/2017

# 1 Introduction

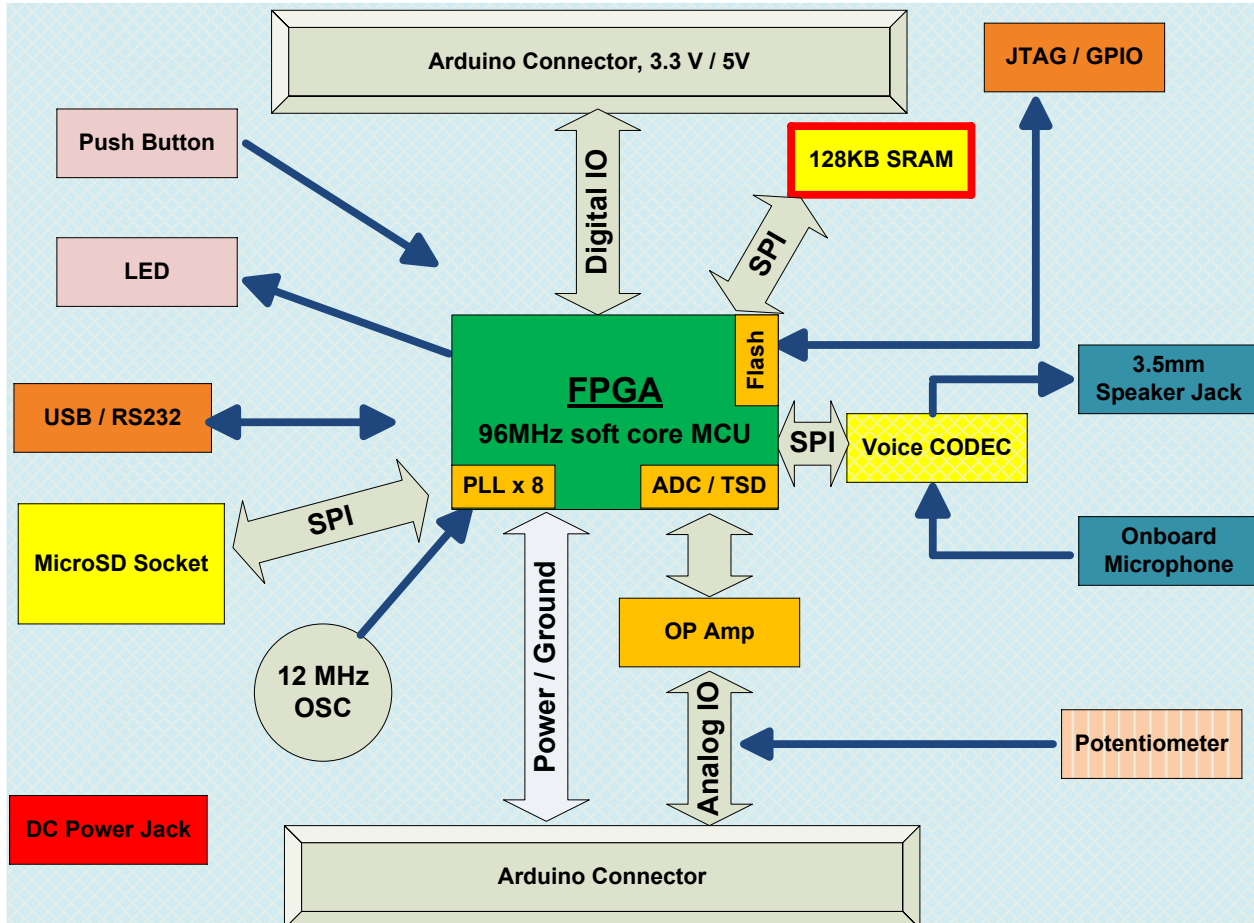


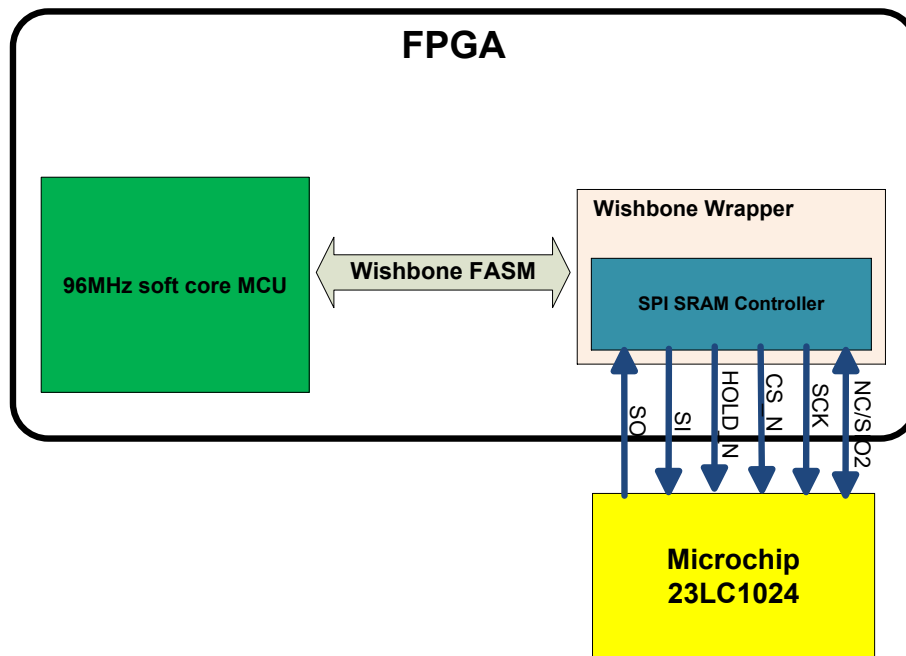
Figure 1-1 The Close View of M10

As shown Figure 1-1, the M10 board takes a distinctive technical approach by embedding an open source soft MCU core (96MHz) into an Intel MAX10 FPGA, while offering an Arduino compatible software interface and form factors. Among all the onboard peripherals, there is a 128KB SRAM with SPI interface. Accordingly PulseRain Technology has designed an open source controller to interact with the SRAM from the FPGA side. And this document contains the technical details of this controller.

## 2 Hardware

### 2.1 Architecture

The M10 board has a Serial SRAM from Microchip (part number 23LC1024). It supports up to 20MHz clock rate with a capacity of 128KB. It has 3 bus modes: SPI, SDI and SQI. Currently the SRAM controller contained in the FPGA image only supports SPI mode with 16MHz clock rate. However, at PCB level, the M10 board puts no constraint on the bus connections. In other words, users are welcome to design their own SRAM controller for SDI/SQI mode with higher clock rate.



**Figure 2-1 The FPGA Controller for Serial SRAM**

As shown in Figure 2-1, the FPGA controller for 23LC1024 is mainly composed of two parts: the Controller and the Wishbone wrapper. The Wishbone wrapper is used to communicate with the MCU core for register read/write. And the controller communicates with the 23LC1024 through 5 pins:

- SCK: 16 MHz clock from FPGA to 23LC1024
- CS\_N: chip select (active low), from FPGA to 23LC1024
- HOLD\_N: Active Low for suspending the bus transmission
- SO: Serial Data Out, from 23LC1024 to FPGA
- SI: Serial Data In, from FPGA to 23LC1024

There is a 6th signal pin on the PCB called NC/SIO2. It is currently not used by the SRAM controller.

And for SRAM, interrupt is not supported.

## 2.2 Port List

The port list of the SRAM Controller is defined in Table 2-1.

Group Name	Signal Name	In/Out	Bit Width	Description
Clock / Reset	clk	Input	1	Clock input, 96MHz
	reset	Input	1	Asynchronous reset, active low
	sync_reset	Input	1	Synchronous reset, active high
Host Interface	instruction_start	Input	1	enable for the instruction to be sent to 23LC1024
	instruction	Input	8	8 bit instruction to be sent to 23LC1024
	address	Input	24	the byte address for memory. The first 7 MSBs are don't care
	write_data	Input	8	byte data to be sent to 23LC1024.
	write_data_grasp	Output	1	A pulse to indicate that the write data has been accepted by the controller
	read_data_enable_out	Output	1	Pulse to indicate valid data is received from 23LC1024
	read_data	Output	8	byte data received from 23LC1024.
23LC1024 Interface	mem_so	Input	1	Serial Data Out, from 23LC1024 to FPGA
	mem_si	Output	1	Serial Data In, from FPGA to 23LC1024
	mem_hold_n	Output	1	HOLD_N, from FPGA to 23LC1024
	mem_cs_n	Output	1	chip select (active low), from FPGA to 23LC1024
	mem_sck	Output	1	16 MHz clock from FPGA to 23LC1024

**Table 2-1 Port List of SRAM Controller**

## 2.3 Pin Assignment

The pin assignment for Serial SRAM is as following for the onboard FPGA (10M08SAE144C8G):

Signal Name	FPGA Pin Assignment (10M08SAE144C8G)
mem_cs_n	29
mem_hold_n	26
mem_sck	27
mem_si	28
mem_so	32
NC/SIO2 (unused)	33

**Table 2-2 FPGA Pin Assignment**

## 2.4 Repository

The RTL code for SRAM controller and its Wishbone wrapper is part of PulseRain Technology's RTL library, which can be found on GitHub:

[https://github.com/PulseRain/PulseRain\\_rtl\\_lib](https://github.com/PulseRain/PulseRain_rtl_lib)

## 3 Software

### 3.1 Register Definition

The Wishbone wrapper shown in Figure 2-1 contains all the registers to control the Serial SRAM. In a nutshell, the registers are defined as following:

- INSTRUCTION (8 bit)

This register contains the 8-bit instruction to be sent to the 23LC1024. The valid values are:

Value	Description
0x03	Sequential read (SPI mode) from 23LC1024
0x02	Sequential write (SPI mode) to 23LC1024
0x05	Read mode register (RDMR)
0x01	Write mode register (WRMR)

**Table 3-1 Valid Value for INSTRUCTION Register**

- DATA (8 bit)

This register contains the byte data to be sent to 23LC1024, or the data received from 23LC1024.

- ADDRESS2 (8 bit), ADDRESS1 (8 bit) and ADDRESS0 (8 bit)

Those three registers contain the 17-bit byte address for 23LC1024, for which ADDRESS2 has the MSB while ADDRESS0 has the LSB. The top 7 bits of ADDRESS2 will be ignored.

- CSR (Control Status Register)

The bits for CSR are defined in Table 3-2:

Bits	R/W	Default	Name	Description
0	WO	0	sync_reset	Write 1 to synchronously reset the SRAM controller
1	RO	0	wr_busy_flag	This flag will be set to 1 when data is being written to the DATA register. And it will be cleared when the write data is accepted by the SRAM controller.
2	RO	0	data_avail_flag	This flag will be set to 1 when valid data is received from 23LC1024. And it will be cleared to 0 when the DATA register is being read.
3	RO	0	23LC1024_busy	This bit is the inverted output of <b>mem_cs</b> pin
7:4	N/A	0	RESERVED	RESERVED

**Table 3-2 Bit Map for CSR (Control Status Register)**



## 3.2 Address Map

The registers defined in Section 3.1 are mapped into MCU's address space, as shown in Table 3-3.

Address	Register Name
0xF9	SRAM_INSTRUCTION
0xFA	SRAM_DATA
0xFB	SRAM_ADDRESS2
0xFC	SRAM_ADDRESS1
0xFD	SRAM_ADDRESS0
0xFE	SRAM_CSR

**Table 3-3 Address Definition**

## 3.3 Work Flow

Before read/write the SRAM, the 23LC1024 has to be initialized by writing to its Mode register (Ref [1]). To initialize the 23LC1024, do the following:

1. Write 1 to CSR to synchronously reset the SRAM controller.
2. Write the intended mode configuration to DATA register.
3. Write 0x01 (WRMR) to the INSTRUCTION register.
4. Keep watching the 23LC1024\_busy flag in CSR register, until this flag becomes low.

To write a byte to SRAM, do the following:

1. Keep watching the 23LC1024\_busy flag in CSR register, until this flag becomes low.
2. Write the data to be sent into DATA register.
3. Write the 17-bit byte address into ADDRESS0, ADDRESS1, and ADDRESS2, where ADDRESS0 has the LSB while ADDRESS2 has the MSB.
4. Write 0x02 (Sequential Write) into INSTRUCTION register.

To read a byte from SRAM, do the following:

1. Keep watching the 23LC1024\_busy flag in CSR register, until this flag becomes low.
2. Write the 17-bit byte address into ADDRESS0, ADDRESS1, and ADDRESS2, where ADDRESS0 has the LSB while ADDRESS2 has the MSB.
3. Write 0x03 (Sequential Read) into INSTRUCTION register.
4. Keep watching the 23LC1024\_busy flag in CSR register, until this flag becomes low.
5. Wait for 3 more bus clock cycles (96MHz)
6. Read the received data from DATA register

### 3.4 Arduino Library

#### 3.4.1 APIs

- *void begin()*

Call this function to initialize the library and configure the 23LC1024.

- *void write (uint32\_t addr, uint8\_t data)*

Call this function to write a byte data into 23LC1024.

Parameters:

addr: 17-bit byte address

data: byte data to be sent to 23LC1024

- *uint8\_t read (uint32\_t addr)*

Call this function to read a byte data from 23LC1024.

Parameters:

addr: 17-bit byte address

Return Value:

the byte data received from 23LC1024

#### 3.4.2 Examples

To further facilitate the software development, the following examples can be referenced:

- *testSRAM*

This example comes with M10SRAM library. It is a simple test program that writes data into 23LC1024, and then read them back to verify the data integrity. The results will be displayed on Serial Port. To use this example, run it in Arduino IDE, and observe the results on Serial Monitor at 115200bps.