IS3752: Multiple I2C Addressable LED Controller

Control 1200 LEDs/chip, 25 chips/bus, up to 30,000 LEDs

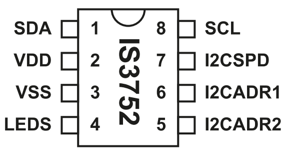
General Description

The IS3752 is an addressable LED controller IC (WS281x and compatible) accessed via I2C. Your microcontroller writes the desired LEDs color and brightness values into its memory map, and it automatically generates the data required to update the LEDs, thereby reducing your firmware complexity and development time.

It supports up to 25 configurable I2C addresses, allowing up to 25x IS3752 devices to operate on the same I2C-serial interface. Each IS3752 can control up to 1,200 LEDs. Using 25x IS3752 on the same I2C-serial interface enables control of up to 30,000 LEDs.

The IS3752 is available in an easy-to-solder SO8N (4.9×6 mm) package with 1.27 mm pitch, supporting an industrial temperature range (-40 to 85 °C). Special on-demand packages and temperature ranges are also available: UFQFPN 3×3 mm, WLCSP 1.7×1.42 mm; extended (-40 to 105 °C) and high (-40 to 125 °C) temperature ranges.

Part Number	Package	Op. Temperature
IS3752-S8-I	SO8N	-40ºC to 85ºC (105ºC)
For other packages and temperatures see Ordering Information		



Main Advantages

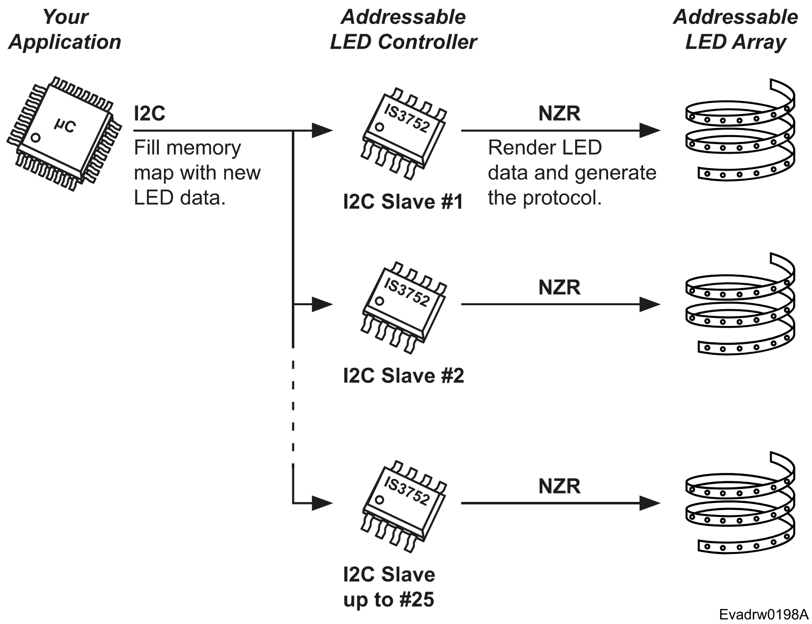
- Uses standard I2C-serial interface; no dedicated libraries required, making firmware easy to maintain and port.
- Control up to 1,200 LEDs per IS3752.
- Connect up to 25x IS3752 on the same I2C-serial interface.
- Control up to 30,000 LEDs from same I2C-serial interface.
- Reduces engineering time and costs.
- Reduces product time-to-market.
- Low-cost solution.
- Compact, easy-to-solder SO8N package (1.27 mm pitch).

Applications

- Custom lighting
- LED signaling
- Architectural lighting
- Status bars or meters
- OEM / Device Manufacturers

Characteristics

- Configurable I2C slave address range: 18 to 42.
- Compatible with: WS2811, WS2812, WS2812B, WS2812C, WS2813, WS2815, NeoPixel, SK6812, GS8208, and other LEDs using the same protocol.
- Compatible with 3-color and 4-color addressable LEDs.
- I2C speeds: 100 kHz, 400 kHz, and 1 MHz.



1 Specification

Absolute Maximum Ratings

Parameter				Min	Max	Unit
Input Voltage	VDD pin			-0.3	4	V
	SCL, SDA, LEDS pins			-0.3	6	
	I2CSPD, I2CADR1, I2CADR2 pins			-0.3	4	
Current Sourced/Sunk by any I/O or Control pin				-	±20	mA
Current Sourced/Sunk by sum of all I/O or Control pin				-	±80	
Temperature	Ambient Temperature (T_A) / [Junction Temperature (T_J)]	IS3752-S8-I	Industrial	-40	85 [105]	ºC
		IS3752-S8-E	Extended	-40	105 [125]	
		IS3752-S8-H	High	-40	125 [130]	
	Storage Temperature			-65	150	

Exceeding the specifications outlined in the Absolute Maximum Ratings could potentially lead to irreversible harm to the device. It's important to note that these ratings solely indicate stress limits and don't guarantee the device's functionality under such conditions, or any others not specified in the Recommended Operating Conditions. Prolonged exposure to conditions at or beyond the Absolute Maximum Ratings might compromise the reliability of the device.

Recommended Operation Conditions

Parameter	Symbol	Min	Nom	Max	Unit
Supply Voltage	V_{DD}	2.0	3.3	3.6	V
Input Voltage at SCL, SDA, LEDS pins	$V_{I/O\text{-}IN}$	-0.3	3.3	5.5	
Input Voltage at I2CSPD, I2CADR1, I2CADR2 pins	$V_{I2CSPD\text{-}IN}$	-0.3	0 to 3.3	3.6	
Source/Sink Current at SCL, SDA, LEDS pins	$I_{I/O\text{-}SS}$	-	±6	±15	mA

Electrical Characteristics

Parameter		Symbol	Min	Typ	Max	Unit
Current Consumption (T_A = 85ºC)		I_{OP}	-	3.50	4.10	mA
Input Voltage	Logical High-Level	V_{IH}	0.7xV_{DD}	-	-	V
	Logical Low-Level	V_{IL}	-	-	0.3xV_{DD}	

Electrical Specifications Revision B

LEDS Pin Timings

Parameter		Symbol	Min	Nom	Max	Unit
Reset (Low-Level Time)		T_R	338	-	-	ns
T0H		T_{0H}	360	-	380	
T1H		T_{1H}	800	-	820	
T0L		T_{0L}	860	-	880	
T1L		T_{1L}	430	-	450	
T0H+T0L		T_0	1.22	1.25	1.26	µs
T1H+T1L		T_1	1.23	1.25	1.27	
Total Frame Time	(1200 LEDs × 24 bits × 1.25 µs) + 0.338 µs	T_{FT}	36	-	-	ms
Frame Refresh Rate	((1200 LEDs × 24 bits × 1.25 µs) + 0.338 µs)-1	F_{RR}	-	-	27.27	Hz

2 Description

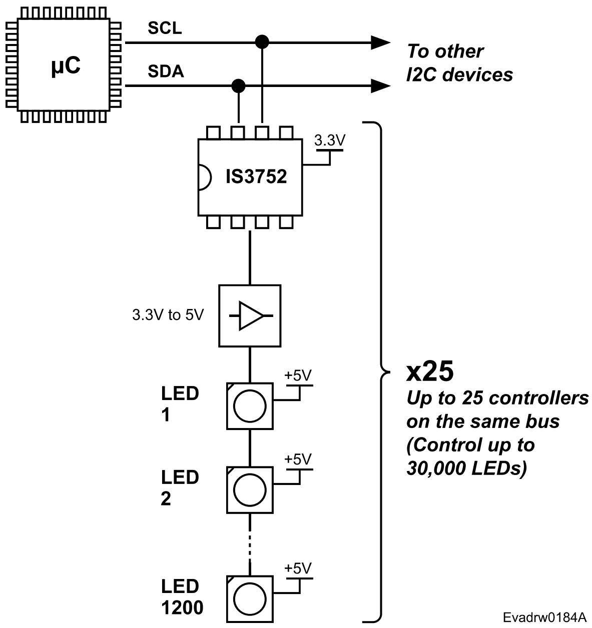
What is it, and what it does

The IS3752 is an addressable LED controller IC accessed via I2C. It supports up to 25 configurable I2C addresses, allowing up to 25x IS3752 devices to operate on the same I2C-serial interface.

Each IS3752 can control up to 1,200 LEDs. Using 25x IS3752 on the same I2C-serial interface enables control of up to 30,000 LEDs.

IS3752 is designed to control single-wire addressable LEDs using NZR coding, which is widely used in popular LEDs such as the WS2812 family.

The chip operates at a 3.3 V. Its I2C pins are 5 V tolerant, making it compatible with both 3.3 V and 5 V systems.



What problem it solves

Generating an addressable LED protocol requires tight timing, which increases CPU load and the number of ISR calls.

The use of the IS3752 is especially suitable for offloading LED protocol generation from the microcontroller, reducing CPU and ISR load. This simplifies firmware, improves debugability, and reduces overall engineering effort.

With the ability to configure up to 25 different I2C slave addresses, it allows you to break down your LED control challenge into multiple parts, making firmware development faster and easier to debug.

How it works

The IS3750 consists of three modules:

1. I2C-Serial Interface
2. Memory Map
3. LED Render

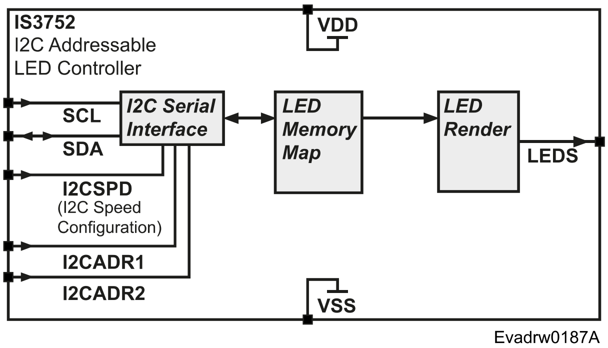


Figure 1: IS3752 Internal Block Diagram

Module 1: I2C-Serial Interface

Data is sent via I2C from your application (microcontroller, FPGA, single-board computer, or any I2C-master device) to the IS3752.

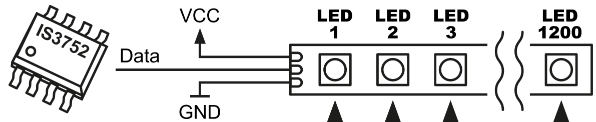
The IS3752 acts as an I2C-slave device, eliminating the need for a dedicated pin on the microcontroller, since it uses a shared bus.

The I2C-serial interface supports Standard Mode (100 kHz), Fast Mode (400 kHz), and Fast Mode Plus (1 MHz). A dedicated pin (I2CSPD) configures the appropriate internal filters for the selected speed. See section I2CSPD Pin for more details.

The I2C-slave addresses are configured by using resistors on the I2CADR1 and I2CADR2 pins. See section I2CADR1, I2CADR2 Pins for more details.

The IS3752 requires I2C clock stretching.

Module 2: Memory Map



IS3750 Memory Map	
0	**SHOW**
1	**LED 1 -** Green
2	**LED 1 -** Red
3	**LED 1 -** Blue
4	**LED 2 -** Green
5	**LED 2 -** Red
6	**LED 2 -** Blue
7	**LED 3 -** Green
8	**LED 3 -** Red
9	**LED 3 -** Blue
...	...
3598	**LED 1200 -** Green
3599	**LED 1200 -** Red
3600	**LED 1200 -** Blue
3601	CHIP_ID
3602	CHIP_REV

Evadrw0193A

The IS3750's internal memory map consists of a single page containing 3,603 registers, with addresses ranging from `0` to `3602`. These registers support both individual and block I2C read/write operations. The IS3752 uses 16-bit register addressing size, and registers size is 8-bit.

There are two relevant types of memory registers: the `SHOW` register (address `0`), and the `LEDx` registers (address `1` to `3600`).

The `LEDx` registers are used to store the data that will be sent to the LEDs; therefore, they define the brightness of each LED color. A value of `0` turns the LED color off, while a value of `255` sets it to full brightness.

All `LEDx` registers are sent consecutively to the LEDs, from register `1` to register `3600`, without taking into account the LED color order or the number of colors per LED. Therefore, the IS3752 is LED-agnostic

For example, in a GRB LED strip, registers 1, 2, and 3 correspond the first LED: register 1 controls the green channel, register 2 controls the red channel, and register 3 controls the blue channel. The same pattern applies to the remaining registers.

Module 3: LED Render
Once all `LEDx` registers are configured, you can trigger an update cycle for all LEDs by writing a `1` to the `SHOW` register. Each time a `1` is written to this register, the LEDs are updated with the contents of the memory map. If the `SHOW` register is not triggered, the LEDs will not update and will retain their previous values, even if you modify the memory map content.

This has two advantages:

1. You can easily perform selective modifications of the `LEDx` registers across different write operations.
2. When all modifications are ready, you can trigger `SHOW` so that all changes are applied simultaneously.

The `SHOW` register is automatically cleared after being triggered.

You can think of the IS3752 as an I2C memory device, this analogy helps in understanding its memory and I2C behavior.

The LEDS pin operates at 3.3 V. When interfacing with 5 V LEDs, a buffer, Schmitt trigger, or level shifter is required to adapt it's 3.3 V to 5 V. Refer to chapter Hardware Example for more details.

2.1 IS3752 is LED Agnostic

The IS3750 is agnostic to the LED model—it does not interpret or enforce a specific color sequence (e.g., RGB, GRB) or the number of colors per LED.

The controller simply transmits all data stored in the `LEDx` registers in sequence, without interpretation, only generating the NZR encoding.

If you are using 3-color LEDs (e.g., RGB or GRB), each LED consumes 3 bytes of data. In this case, the IS3750 can control up to 1,200 LEDs (3,600 registers ÷ 3 bytes per LED).

For 4-color LEDs (e.g., RGBW), each LED requires 4 bytes, allowing control of up to 900 LEDs (3,600 ÷ 4).

Note: The most common LED configuration on the market is a 3-channel GRB sequence.

Color Sequence Agnostic Example
Suppose you write the value `255` to address `1` and then trigger the `SHOW` register.

If you are using a GRB LED strip, the first LED will display green, since the first `LEDx` register maps to the first color channel of the first LED (green in this example).

However, if you are using an RGB LED strip instead of GRB, writing to the same first `LEDx` register will result in red output, since the first `LEDx` register maps to the first color channel of the first LED (red in this example).

Note: GRB is the most common color sequence in popular addressable LEDs such as the WS2812B family.

Color Count Agnostic Example
Suppose you write the value 255 to address 4 and then trigger the `SHOW` register.

With a GRB LED strip, the second LED will light up green (since its first byte is at address 4).

With a GRBW LED strip, the first LED will show white, as address 4 corresponds to its fourth channel.

Note: Most addressable LEDs in the market are 3-channel (RGB or GRB).

2.2 Troubleshooting

For proper operation, ensure that both hardware and firmware are correctly configured by following the next validation steps below.

Hardware Validation

First, ensure you have a proper hardware design by verifying:

1. The VDD pin is supplied with 3.3 V.
2. A non-inverting buffer connects the LEDS pin of the IS3752 to the data-in pin of the addressable LEDs.
3. Your LEDs are properly supplied:
    1. Critical: your power supply current capability should be at least 1.5 times the maximum consumption of all LEDs at full brightness with all colors active. A common issue is an under-rated power supply, resulting in supply sagging, which leads to your microcontroller and the IS3752 repeatedly resetting, causing a cyclical flash on the LEDs.
    2. The power supply voltage matches the LED voltage. Some LEDs are 5 V powered, others 12 V, etc.
    3. The ground reference (0 V) of the IS3752 is properly connected to the ground reference of the non-inverting buffer and the LEDs.
4. Your microcontroller is connected via I2C to the SCL and SDA pins, with pull-up resistors to 3.3 V or 5 V.
5. The I2CSPD pin is tied to GND for 100 kHz operation, or configured accordingly for other speeds. See section I2CSPD Pin for more information.
6. The I2CADR1 and I2CADR2 pins are configured to define a valid I2C slave address, and your firmware correctly addresses that corresponding I2C address. See section I2CADR1, I2CADR2 Pins for more information.

Refer to chapter Hardware Example for more information about hardware design.

Firmware Validation

Before starting with your firmware, it is good practice to first validate that your microcontroller can properly communicate via I2C.

The simplest way to do this is to scan the I2C-serial interface and confirm that the IS3752 is detected on the bus.

Refer to application note *ISAN0001-How to scan I2C Serial Interface* for instructions on scanning the I2C-serial interface.

Once the IS3752 is detected on the bus, you can be confident that the I2C-serial interface is working correctly.

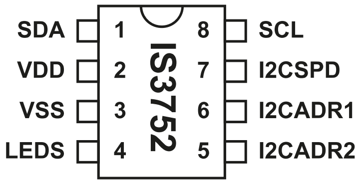
At this point, you can implement your I2C read and write routine. **The IS3752 uses 16-bit register addressing size with 8-bit data size and must support I2C clock stretching**. Understanding this is crucial for proper implementation of read and write routines.

To verify your read routine, read the `CHIP_ID` register at address `3601`. This register is read-only and always returns the fixed value `153`. If you obtain this value, communication is validated and you can safely begin your firmware.

Be aware when using I2C write operations in your firmware via interrupts or DMA. Writing all 3,600 LEDx registers at 100 kHz takes a significant amount of time, and non-blocking I2C transfers (interrupt or DMA-based) may overlap with a previous unfinished I2C write operation. Ensure that the previous I2C transfer has completed before starting a new write cycle.

Refer to *Firmware Examples* chapter for example codes in STM32, Arduino and Raspberry Pi platforms.

3 Pins



Pin	Name	Type	Description
1	SDA	Open-Drain 5 V-Tolerant	I2C data pin. Connects to your microcontroller. Open drain, it requires pull-up resistor.
2	VDD	Supply	3.3 V power supply pin. Bypass this pin to VSS with a 100 nF and 2.2 µF ceramic capacitors. Place them as close as possible to the IC.
3	VSS	Supply	0 V reference pin.
4	LEDS	Digital Output Push-Pull	3.3 V addressable LED data pin. The logic level on this pin must be shifted to 5 V to properly drive the LED's data input.
5	I2CADR2	Analog Input 0 to 3.3 V	I2C-serial interface slave address configuration pins. Do not leave these pins floating. For more details, see Table 1: I2C Address vs I2CADR1 and I2CADR2 Voltage Levels.
6	I2CADR1		
7	I2CSPD	Analog Input 0 to 3.3 V	I2C-serial interface speed selection pin. • For 100 kHz pull to GND. • For 400 kHz make a voltage divider of VDD/2 (1.65 V). • For 1 MHz pull to VDD (3.3 V).
8	SCL	Open Drain 5 V-Tolerant	I2C Clock pin. Open drain, it requires pull-up.

3.1 LEDS Pin

3.3 V Addressable LEDS Data Pin.

The Addressable LEDS Data Pin operates at 3.3 V. However, addressable LEDs typically require a 5 V logic level, so a non-inverting buffer is needed to shift the signal from 3.3 V to 5 V. There are many suitable buffers available; in the Hardware Example chapter, the 74LVC1G17 is used.

(Pin description continues on next page)

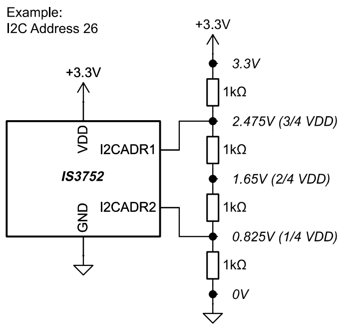
3.2 I2CADR1, I2CADR2 Pins

I2C-Serial Interface Address Configuration Pins

The combination of I2CADR1 and I2CADR2 pins, each supporting five discrete voltage levels, allows selection of up to 25 different I2C slave addresses. This enables up to 25x IS3752 devices to be connected on the same bus, providing control of up to 30,000 LEDs in total.

The five voltage levels are 0 V, 0.825 V, 1.65 V, 2.475 V, and 3.3 V. These can be generated using a resistor divider with four equal-value resistors in series. Each voltage level has a tolerance of ±0.41 V, allowing the use of standard 1% tolerance resistors. These pins are sampled only at power-up and must not be left floating.

In the following example, the IS3752 is assigned slave address 26, with I2CADR1 connected to 2.475 V, and I2CADR2 connected to 0.825 V.



The following table shows the I2C slave address as a function of the voltage levels applied to the I2CADR1 and I2CADR2 pins.

I2C Address	I2CADR1 (V)	I2CADR2 (V)
18	0	0
19	0.825	0
20	1.65	0
21	2.475	0
22	3.3	0
23	0	0.825
24	0.825	0.825
25	1.65	0.825
26	2.475	0.825
27	3.3	0.825
28	0	1.65
29	0.825	1.65
30	1.65	1.65
31	2.475	1.65
32	3.3	1.65
33	0	2.475
34	0.825	2.475
35	1.65	2.475
36	2.475	2.475
37	3.3	2.475
38	0	3.3
39	0.825	3.3
40	1.65	3.3
41	2.475	3.3
42	3.3	3.3

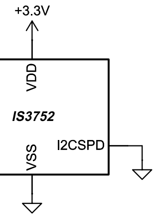
Table 1: I2C Address vs I2CADR1 and I2CADR2 Voltage Levels

3.3 I2CSPD Pin

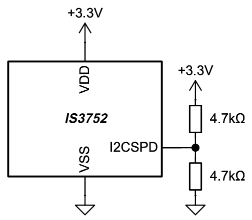
I2C-Serial Interface Speed Selection Pin.

This pin configures the IS3752 internal I2C-serial interface timings and filters to properly work with the selected bus speed. It is sampled only at power-up and must not be left floating.

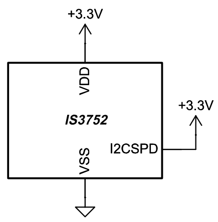
For a **100 kHz** setting, set the I2CSPD pin to VSS.



For a **400 kHz** setting, set the I2CSPD to 1.65 V (VDD/2) using a balanced voltage divider. This can be achieved by placing two 4.7 kΩ resistors from the I2CSPD pin: one to VDD and the other to VSS.



For a **1 MHz** setting, set the I2CSPD pin to 3.3 V.



4 Memory Map

For clarification purposes, the following Memory Map table assumes a 3-channel GRB LED strip, as this is the most common configuration. Therefore, LED numbers and color channels are indicated in the register names for clarity.

However, the IS3752 is LED-count and color-order agnostic. When using an RGB LED strip instead of GRB, Register 1 controls the red channel of LED 1. If a 4-channel LED strip is used, Register 4 controls the fourth channel of LED 1, rather than the first channel of a second LED.

I2C Register Address	Register Name	Description
0 (0x0000)	**SHOW**	Send data to the LEDs
1 (0x0001)	**LED1G**	LED 1, color Green brightness
2 (0x0002)	**LED1R**	LED 1, color Red brightness
3 (0x0003)	**LED1B**	LED 1, color Blue brightness
4 (0x0004)	**LED2G**	LED 2, color Green brightness
5 (0x0005)	**LED2R**	LED 2, color Red brightness
6 (0x0006)	**LED2B**	LED 2, color Blue brightness
7 (0x0007)	**LED3G**	LED 3, color Green brightness
8 (0x0008)	**LED3R**	LED 3, color Red brightness
9 (0x0009)	**LED3B**	LED 3, color Blue brightness
...	(LED4 to LED1197)	
3592 (0x0E08)	**LED1198G**	LED 1198, color Green brightness
3593 (0x0E09)	**LED1198R**	LED 1198, color Red brightness
3594 (0x0E0A)	**LED1198B**	LED 1198, color Blue brightness
3595 (0x0E0B)	**LED1199G**	LED 1199, color Green brightness
3596 (0x0E0C)	**LED1199R**	LED 1199, color Red brightness
3597 (0x0E0D)	**LED1199B**	LED 1199, color Blue brightness
3598 (0x0E0E)	**LED1200G**	LED 1200, color Green brightness
3599 (0x0E0F)	**LED1200R**	LED 1200, color Red brightness
3600 (0x0E10)	**LED1200B**	LED 1200, color Blue brightness
3601 (0x0E11)	CHIP_ID	Chip identification number (constant over product life-cycle)
3602 (0x0E12)	CHIP_REV	Chip revision (may change without notice)

Table 2: IS3752's Memory Map[1]

1 Assumes GRB LEDs.

4.1 SHOW

Writing a `1` to this register triggers the rendering of all memory map `LEDx` data and sends it through the LEDS pin. The register is automatically reset to `0`.

You can send `LEDx` data in the same I2C write operation in which this register is set to `1`; the data will be applied in that same update cycle.

Writing a `0` makes no effect.

At power-up, the default value for this register is `0`, meaning all LEDs remain off.

Name:	`SHOW`
Description:	Send data to the LEDs
Address:	`0`
Memory Type:	Volatile RAM
Allowed Values:	`0` or `1`
Reset Values:	`0`

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
`SHOW`							

4.2 LEDx

`LEDx` registers store the color brightness values that are sent to the LEDs when a `1` is written to the `SHOW` register.

The value stored in each register is what will be directly sent to the LED. A value of `0` means the corresponding color is off, while a value of `255` means the color is at maximum brightness.

You can set the `SHOW` register to 1 in the same I2C write operation as the `LEDx` registers; the data will be sent to the LEDs in the same update cycle.

These LED registers are volatile RAM. You can modify them individually, or update all of them at once in a single I2C operation.

When the IS3752 loses power, the contents of the registers are lost. At power-up, the default value for all registers is `0`.

These registers can be both written to and read from. If needed, they can also be used as an extension of your microcontroller's RAM, helping to reduce memory usage in your project regarding to LED managing.

Writing to `LEDx` registers does not have any effect on the LEDs until the `SHOW` register is triggered.

Name:	`LEDx`
Description:	Brightness register of LEDx color
Address:	`1` to `3600`
Memory Type:	Volatile RAM
Allowed Values:	`0` to `255`
Reset Values:	`0`

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LEDx							

4.3 CHIP_ID

The CHIP_ID register contains the chip identifier, which is a fixed value of 153. This value is used for production tracking. It is stored in ROM and will not change throughout the product's life-cycle.

Since this register value is constant, reading it during firmware development can help verify that I2C communications are working and that the chip’s memory can be properly read.

This register is read-only.

Name: CHIP_ID
Description: Chip Identification Number
Address: 3601
Memory Type: ROM
Value: 153

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0	0	0	1	0	1	1	0

4.4 CHIP_REV

The CHIP_REV register indicates the chip revision.

This value is intended for production and product tracking. It is stored in ROM and may change throughout the product's life-cycle.

This register is read-only.

Name: CHIP_REV
Description: Chip Revision Number
Address: 3602
Memory Type: ROM
Value: Depends on the revision

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
-	-	-	-	-	-	-	-

5 I2C

The IS3752 operates as a Slave in the I2C-serial interface. It supports Standard Mode (100 kHz), Fast Mode (400 kHz), and Fast Mode Plus (1 MHz). The I2C-master device, typically a microcontroller or a microprocessor, initiates and manages all read operations to the Slave.

The IS3752 is represented on the bus by the I2C device slave address configured via I2CARD1 and I2CADR2 pins, and can range from 18 to 42. See section I2CADR1, I2CADR2 Pins for more information about the I2C slave address configuration.

Pull-up resistors are required on the SCL and SDA lines for proper operation. The resistor values depend on the bus capacitance and operating speed. Typical values are 4.7 kΩ for Standard Mode (100 kHz) and 2 kΩ for Fast Mode and Fast Mode Plus (400 kHz and 1 MHz).

The IS3752's I2C pins high state can be either 3.3 V or 5 V. A logical `0` is transmitted by pulling the line low, while a logical `1` is transmitted by releasing the line, allowing it to be pulled high by the pull-up resistor. The I2C-master controls the Serial Clock (SCL) line, which generates the synchronous clock used by the Serial Data (SDA) line to transmit data.

The memory map consists of 3,603 registers; therefore, addressing a register requires 2 bytes. Each register is 8 bits wide.

The IS3752 requires clock stretching.

The operability of the Read and Write commands of the IS3752 is very similar to an EEPROM memory. Thinking of the IS3752 as an EEPROM memory is a good analogy to quickly understand how to communicate with the device.

5.1 Highlights

I2C Device Slave Address: 18 to 42

I2C Memory Map Addressing Size: 16-bit (2x 8-bit)

I2C Memory Map Register Size: 8-bit

I2C Clock Stretching: Required

Compatible I2C Speeds:

- Standard Mode (100 kHz), recommended SCL and SDA pull-up value: 4.7 kΩ
- Fast Mode (400 kHz), recommended SCL and SDA pull-up value: 2 kΩ
- Fast Mode (10 MHz), recommended SCL and SDA pull-up value: 2 kΩ

Supported Operations:

- Single-Byte Write
- Multiple-Byte Write (up to 3,603 registers)
- Single-Byte Read
- Multiple-Byte Read (up to 3,603 registers)

Overreading and Overwriting the memory:

- If a write operation starts at a valid memory address (`0` to `3602`) and continues past the last valid address, it will roll over to address `0`.
- Starting a write operation to an invalid memory address (greater than `3602`) will result in a NACK and data will be discarded.
- If a read operation starts at a valid memory address (`0` to `3602`) and continues past the last valid address, it will roll over to address `0`.
- Starting a read operation at an invalid memory address (greater than `3602`) will return a value of 0xFF.

5.2 Single Byte Write

Writing a single byte is an action performed by your microcontroller (I2C-master) to write data to any register within the IS3752 memory map (I2C-slave), regardless of the last read or written position. To perform this action, the microcontroller must load the register address intended to be written into the IS3752's pointer register. Once the address is set, your microcontroller can send the data to be written.

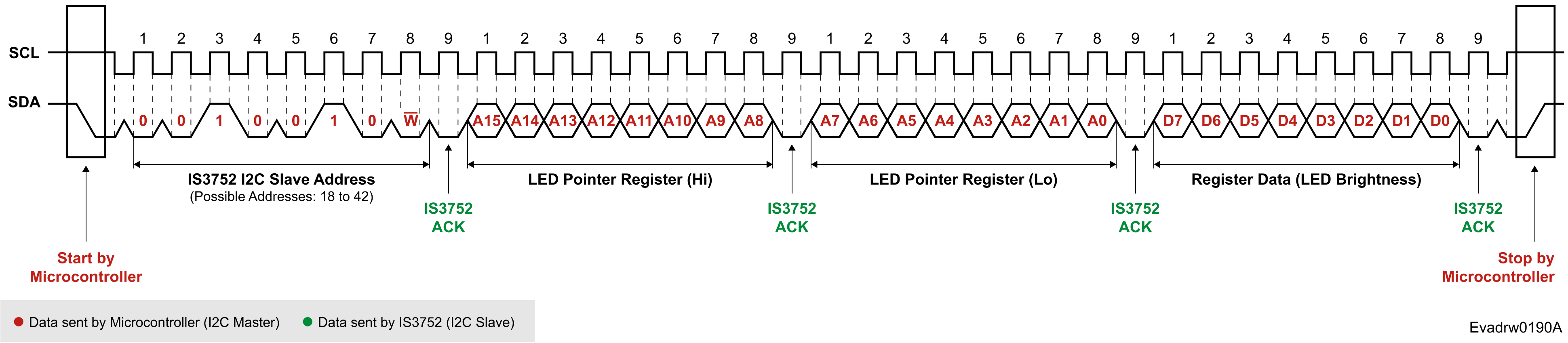
To initiate the single byte write operation, the following steps must be performed from the beginning: your microcontroller begins by pulling down the SDA line while the SCL line is high, creating a Start Condition. It then sends the IS3752 I2C device slave address, configured via the I2CADR1 and I2CADR2 pins (range 18 to 42), with the R/W bit set to `0` (indicating a write operation).

Upon receiving the device slave address, the IS3752 acknowledges it. Subsequently, your microcontroller sends the two bytes of the register address it intends to write: the most significant byte first, followed by the least significant byte, each acknowledged by the IS3752.

Your microcontroller then sends the byte to be written, which the IS3752 acknowledges. Finally, your microcontroller issues a Stop Condition by raising the SDA line while the SCL line is high.

Invalid Memory Addressing

The valid memory range of the IS3752 for a write operation goes from addresses `0` to `3602`. If a Write Operation is performed with a Pointer Register higher than `3602`, the IS3752 will answer with a NACK.



5.3 Multiple Byte Write

The Multiple Byte Write operation functions similarly to the Single Byte Write, but allows writing a block of up to 3,603 registers in a single operation.

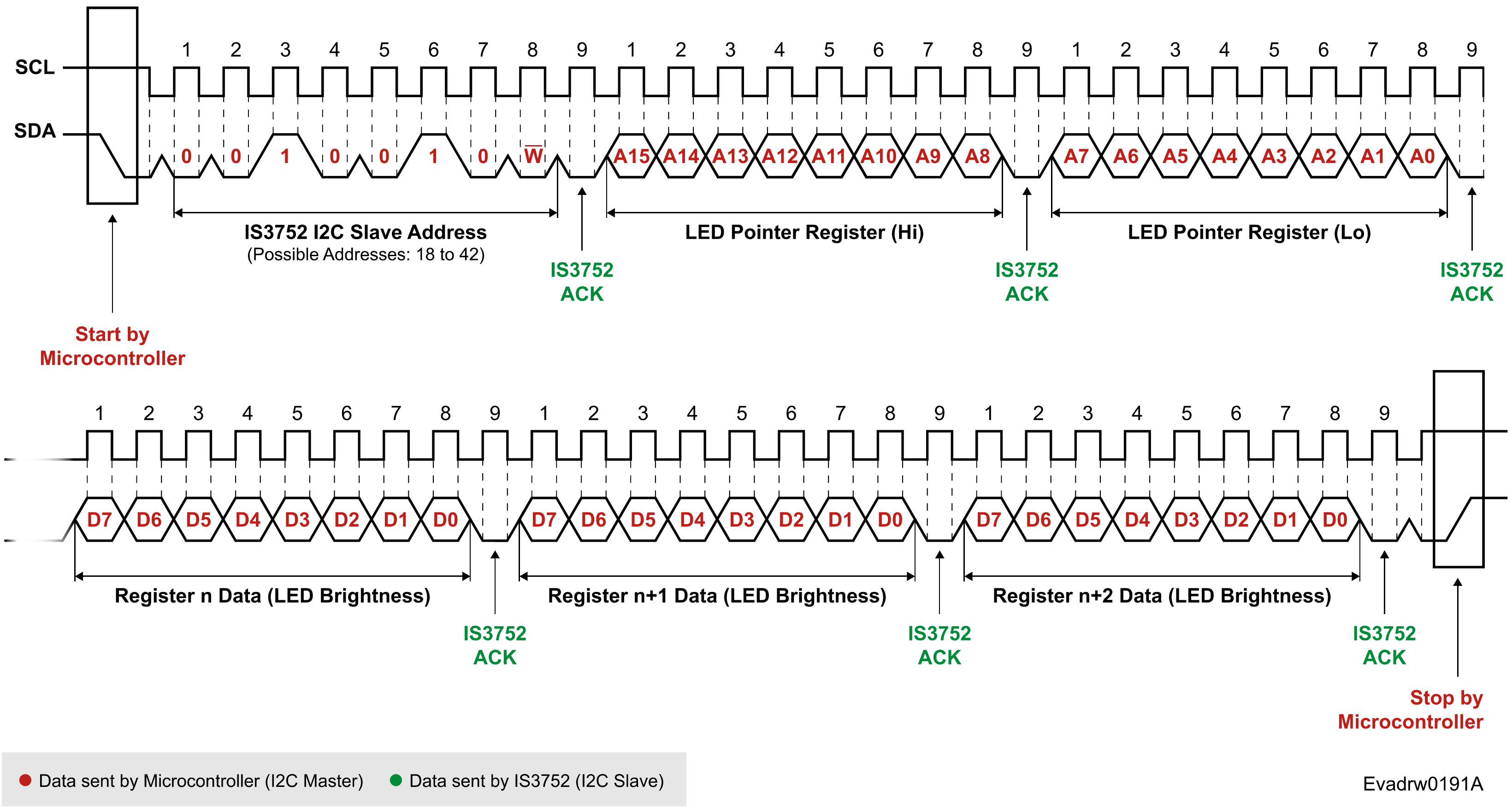
To perform a Multiple Byte Write operation, follow the same procedure as for a Single Byte Write until the first data byte is written. After writing the first byte, instead of generating a Stop Condition, your microcontroller should continue writing data bytes. To conclude the write operation, after sending the last data byte, your microcontroller should generate a Stop Condition.

Invalid Memory Addressing

The valid memory range of the IS3752 for a write operation goes from addresses `0` to `3602`.

If a Multiple Byte Write Operation is performed with a pointer register within the valid memory range (`0` to `3602`) but exceeds the last memory register (`3602`), a rollover to register `0` will occur.

If a Multiple Byte Write Operation is performed with a pointer register outside from the valid memory range (greater than `3602`), the IS3752 will respond with a NACK upon receiving the first data byte.



5.4 Single Byte Read

Reading a single byte is an action performed by your microcontroller (I2C-master) to access any register within the IS3752 memory map (I2C-slave), regardless of the last read position. To perform this action, your microcontroller must load the address of the register to be read into the IS3752's pointer register. Once the address is set, the microcontroller can retrieve the data from the specified register.

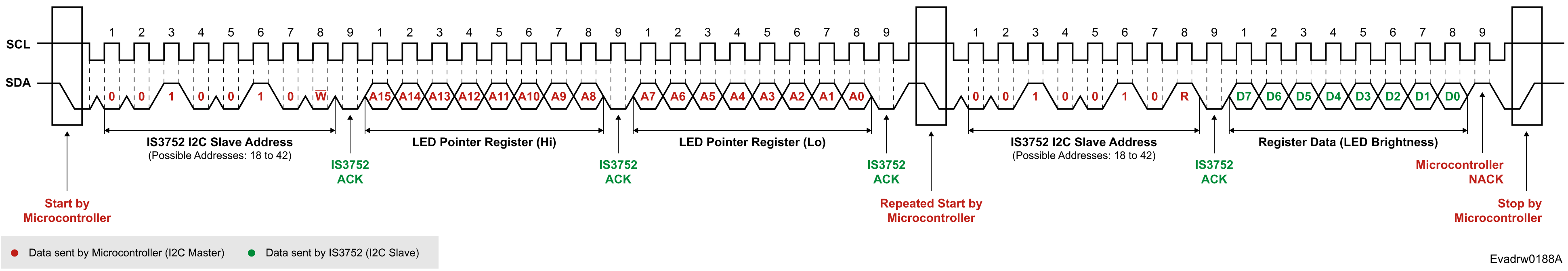
To initiate the Single Byte Read operation, the following steps must be performed from the beginning: your microcontroller starts by pulling SDA low while SCL is high to generate a Start Condition. It then sends the IS3752 I2C device slave address, configured via I2CADR1 and I2CADR2 pins (range 18 to 42), with the R/W bit set to `0` (indicating a write operation). Upon receiving the device slave address, the IS3752 acknowledges it. Subsequently, your microcontroller sends the two bytes of the pointer register address: the most significant byte first, followed by the less significant byte, each acknowledged by the IS3752. This sets the address of the next register to be read in the pointer register.

Next, the content of the pointer register needs to be read.

Your microcontroller generates a Repeated Start Condition, followed by the IS3752 I2C device slave address (range 18 to 42) with the R/W bit set to `1` (indicating a read operation), instructing the IS3752 to retrieve data. The IS3752 acknowledges and responds with the data, which your microcontroller does not acknowledge (NACK). Finally, your microcontroller issues a Stop Condition by raising the SDA line while the SCL is high.

Invalid Memory Addressing

The valid memory range of the IS3752 goes from addresses `0` to `3602`. If a Single Byte Read operation is performed with a pointer register higher than `3602`, the read result will be 0xFF.



5.5 Multiple Byte Read

Multiple Byte Read operation functions similarly to Single Byte Read but can read a block of up to 3,603 registers in a single operation, corresponding to the full memory map.

To perform a Multiple Byte Read operation, follow the same procedure as for a Single Byte Read until the first byte is received. After receiving the first byte, instead of generating a Not Acknowledge (NACK), your microcontroller should continue acknowledging (ACK) each received data byte from the IS3752 for as many bytes as it intends to read. To conclude the read operation, after reading the last data byte, your microcontroller should generate a Not Acknowledge (NACK) and a Stop Condition.

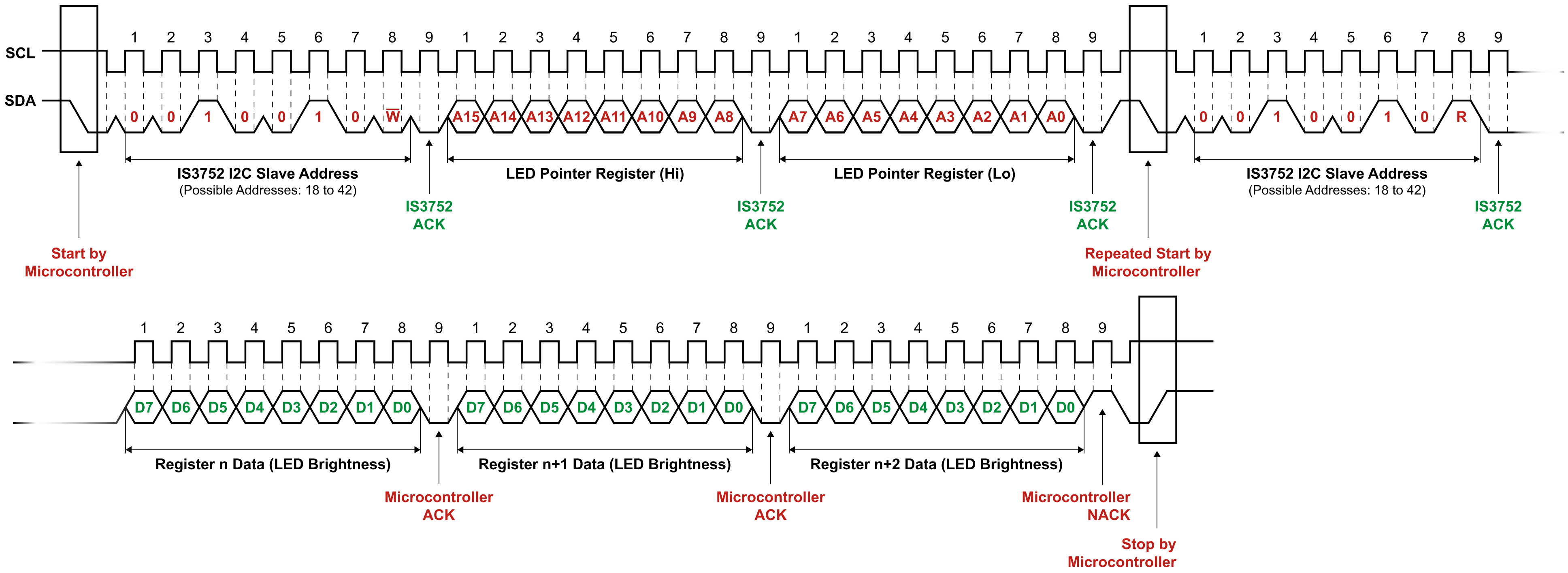
With each byte read, the pointer register increments by one.

Invalid Memory Addressing

The valid memory range of the IS3752 goes from addresses `0` to `3602`.

If the Multiple Byte Read operation is performed with a pointer register within the valid memory range (`0` to `3602`), but the data retrieval extends beyond register `3602`, a rollover to position `0` will occur.

If a Multiple Byte Read operation is performed with a pointer register value higher than `3602`, the read result will be `0xFF`.



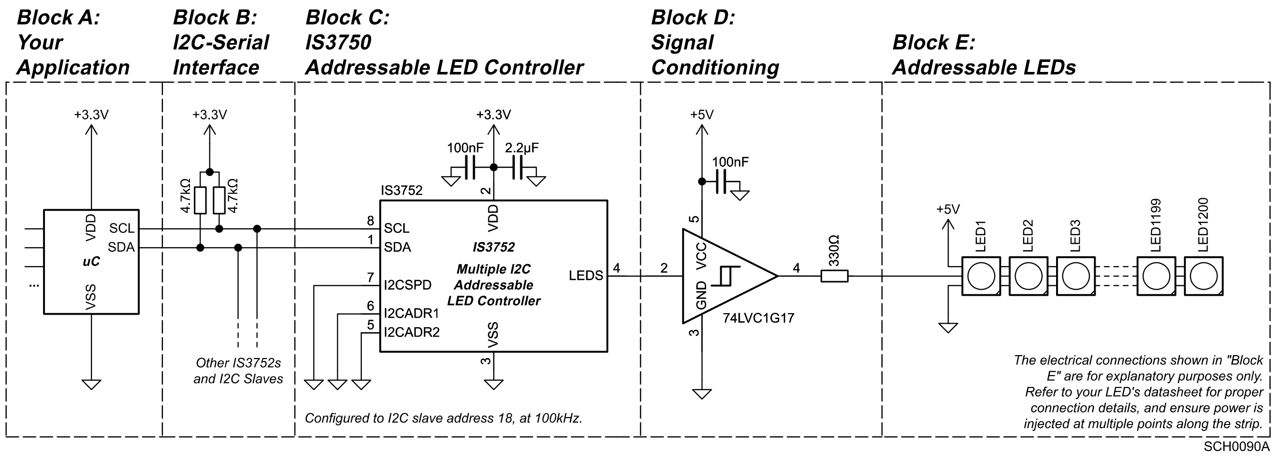
● Data sent by Microcontroller (I2C Master) ● Data sent by IS3752 (I2C Slave)

Evadrw0189A

6 Hardware Example

The following chapter represents an application design example for explanation proposals and is not part of the product standard. The customer must design his own solution, choose its most appropriate components and validate the final product according to the legislation and the addressable LED specifications.

This example shows how to use a microcontroller with the IS3752 to drive up to 1200 LEDs.



Block A: Your Application

This is the core of your project. Typically, it will be a microcontroller, FPGA, or an embedded computer like a Raspberry Pi, among others.

It's the part of the system where you want to offload CPU load, reduce RAM and Flash usage, and eliminate the need for timers—along with the stress of continuously handling timer interrupts. In short, it's the part you want to keep as clean and simplified as possible.

Block B: I2C-Serial Interface

This is the bus where you connect all your I2C slave devices. It can operate at either 3.3 V or 5 V, as the IS3752 I2C pins are 5 V tolerant.

The I2C-serial interface requires pull-up resistors on the SCL and SDA lines. Typical values are 4.7 kΩ for Standard Mode (100 kHz), and 2 kΩ for Fast Mode (400 kHz) and Fast Mode Plus (1 MHz).

The IS3752 I2C device slave address ranges 18 to 42. In the example address 18 is set by connecting I2CADR1 and I2CADR2 pins to VSS. If you require a different address, see Table 1: I2C Address vs I2CADR1 and I2CADR2 Voltage Levels.

Block C: IS3752

Place 100 nF and 2.2 µF decoupling capacitors close to the VDD supply pin.

Block D: Signal Conditioning

The LEDS output pin operates at 3.3 V. However, addressable LEDs typically require a 5 V logic level, so a non-inverting buffer is needed to shift the signal from 3.3 V to 5 V. There are many suitable buffers available; in this example, the 74LVC1G17 is used.

A series resistor—typically between 330 Ω and 470 Ω—is recommended between the buffer and the first addressable LED. This helps reduce signal ringing by adding impedance to better match the trace or wire, thereby damping reflections.

Note: this is not a pull-up or pull-down resistor; it's a series resistor placed directly between the buffer output and the LED’s data input.

Block E: Addressable LEDs

The IS3752 can control any number of LEDs in series, as long as the total number of data bytes does not exceed 3,600. This means it supports up to 1,200 3-color LEDs, 900 4-color LEDs, and so on. Typically, addressable LEDs are 3-color.

When using more than a few LEDs, special attention must be given to the power supply design. Poor PCB layout or cabling in the power domain can lead to burnt traces or wires, voltage drops (which my reset your microcontroller or the IS3752), and incorrect LED brightness or spurious flickering.

Always calculate the total current draw of all LEDs to properly size the power supply. Do not assume that certain colors won’t be used and therefore a smaller supply is sufficient. Always assume the worst case: all LED colors on at 100% brightness.

When powering LEDs from an external power supply, make sure the LEDs and the rest of the circuitry controlling the LEDs share the same voltage reference. The VSS of the LEDs and the VSS of your control circuit must be connected.

7 Firmware Examples

The following chapter provides example firmware for demonstration purposes only and does not form part of the product specification. The customer is responsible for implementing their own solution and for ensuring compliance with applicable legislation and the protocol specifications.

The IS3752 chip is very easy to use. It has been designed to facilitate the integration of addressable LEDs into your project.

Your microcontroller only needs to write the desired brightness values to the corresponding LED color registers (`LEDx`). A value of `0` turns the LED color channel off, while a value of `255` sets it to maximum brightness.

You can perform as many write operations as you need, and when you are ready, to update the LEDs acording to the memory map data, you only need to write a `1` to the `SHOW` register. Every time a `1` is written to the `SHOW` register the LEDs are updated. This register is automatically reset to `0`.

Think of how a classical I2C EEPROM works. It is essentially the same, with the only difference being that writing to a memory register is translated into LED brightness control after setting the `SHOW` register to `1`.

7.1 Pseudocode

Let's use 3 GRB addressable LEDs to create a classical traffic light sequence: green (first LED), amber (second LED), red (third LED), and so on, with 1 second between each step.

Since there are only 3 LEDs, registers `1` to `9` are used:

- Registers `1`, `2`, and `3` for the first LED
- Registers `4`, `5`, and `6` for the second LED
- Registers `7`, `8`, and `9` for the third LED

Execute the following steps in your while(1) loop:

Do the green:

0. Write `0` to registers `1` to `9` to ensure everything is cleared, and write `255` (full brightness) to register `1` (LED1, green color).
1. Show the content of the memory map on the LEDs by writing `1` to the `SHOW` register.
2. Wait 1 second.

Do the amber:

3. Write `0` to registers `1` to `9` to ensure everything is cleared, write `120` to register `4` (LED2, green color), and write `255` (full brightness) to register `5` (LED2, red color).
4. Show the content of the memory map on the LEDs by writing `1` to the `SHOW` register.
5. Wait 1 second.

Do the red:

6. Write `0` to registers `1` to `9` to ensure everything is cleared, and write `255` to register `8` (LED3, red color).
7. Show the content of the memory map on the LEDs by writing `1` to the `SHOW` register.
8. Wait 1 second.

7.2 STM32

Coding the IS3752 on an STM32 is very simple. You do not need any special libraries—using the I2C HAL functions `HAL_I2C_Mem_Read()` and `HAL_I2C_Mem_Write()` is sufficient to communicate with the IS3752.

Problem Statement
The example implements a traffic light simulation described in the Pseudocode section.

Warnings
Pay special attention on the `HAL_I2C_Mem_Write()` function:

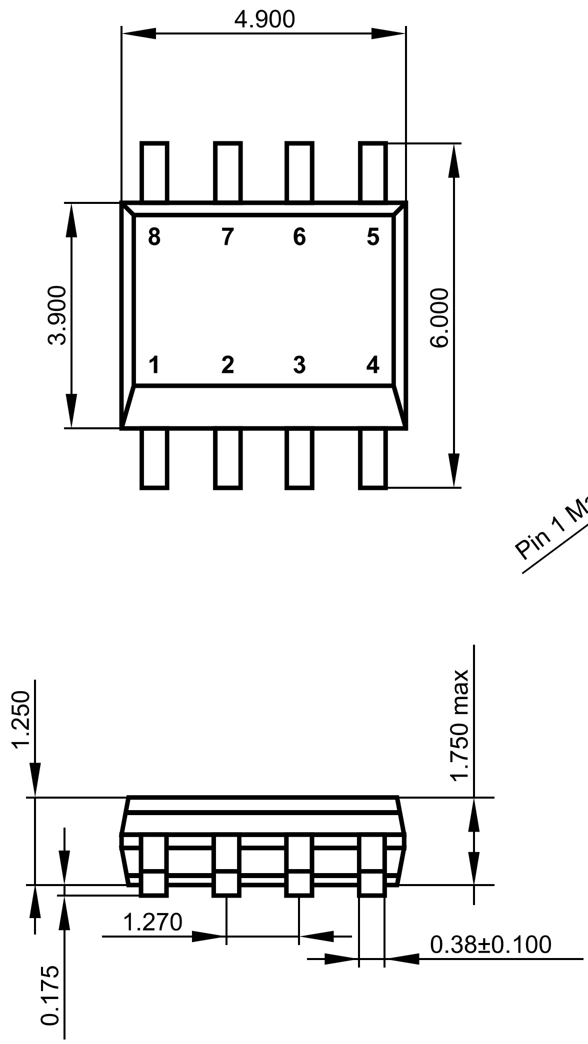
- The IS3752 uses 16-bit memory addressing. Always use `I2C_MEMADD_SIZE_16BIT` in the `HAL_I2C_Mem_Write()` function and never `I2C_MEMADD_SIZE_8BIT`.
- The IS3752 registers are 8-bit wide; always use `uint8_t` variables to store data written to its memory map, and never `uint16_t`.
- When using the blocking `HAL_I2C_Mem_Write()` function, ensure that the timeout is sufficiently large. Writing all `LEDx` registers (a total of 3,600) over I2C at 100 kHz takes a significant amount of time. If the timeout is too short, the transmission will be interrupted due to a self-timeout.
- When using non-blocking I2C write functions such as interrupt (`HAL_I2C_Mem_Write_IT()`) or DMA (`HAL_I2C_Mem_Write_DMA()`), ensure that the previous transmission has fully completed before starting a new one. Otherwise, one transfer will interrupt the other.

```
uint8_t vector[15];

// ======= Do color GREEN =======
vector[0] = 1; // Writing a 1 to the SHOW register will automatically SHOW the data on
the LEDs
// LED 1:
vector[1] = 255; // Green
vector[2] = 0; // Red
vector[3] = 0; // Blue
// LED 2:
vector[4] = 0; // Green
vector[5] = 0; // Red
vector[6] = 0; // Blue
// LED 3:
vector[7] = 0; // Green
vector[8] = 0; // Red
vector[9] = 0; // Blue
// Send the data to the IS3752:
HAL_I2C_Mem_Write(&hi2c1, (18 << 1), 0, I2C_MEMADD_SIZE_16BIT, vector, 10, 1000);
HAL_Delay(1000); // Wait 1 second

// ======= Do color AMBER =======
vector[0] = 1; // Writing a 1 to the SHOW register will automatically SHOW the data on
the LEDs
// LED 1:
vector[1] = 0; // Green
vector[2] = 0; // Red
vector[3] = 0; // Blue
// LED 2:
vector[4] = 120; // Green
vector[5] = 255; // Red
vector[6] = 0; // Blue
// LED 3:
vector[7] = 0; // Green
vector[8] = 0; // Red
vector[9] = 0; // Blue
// Send the data to the IS3752:
HAL_I2C_Mem_Write(&hi2c1, (18 << 1), 0, I2C_MEMADD_SIZE_16BIT, vector, 10, 1000);
HAL_Delay(1000); // Wait 1 second

// ======= Do color RED =======
vector[0] = 1; // Writing a 1 to the SHOW register will automatically SHOW the data on
the LEDs
```

```
// LED 1:
vector[1] = 0; // Green
vector[2] = 0; // Red
vector[3] = 0; // Blue
// LED 2:
vector[4] = 0; // Green
vector[5] = 0; // Red
vector[6] = 0; // Blue
// LED 3:
vector[7] = 0; // Green
vector[8] = 255; // Red
vector[9] = 0; // Blue
// Send the data to the IS3752:
HAL_I2C_Mem_Write(&hi2c1, (18 << 1), 0, I2C_MEMADD_SIZE_16BIT, vector, 10, 1000);
HAL_Delay(1000); // Wait 1 second
```

8 Mechanical

SO8N Package

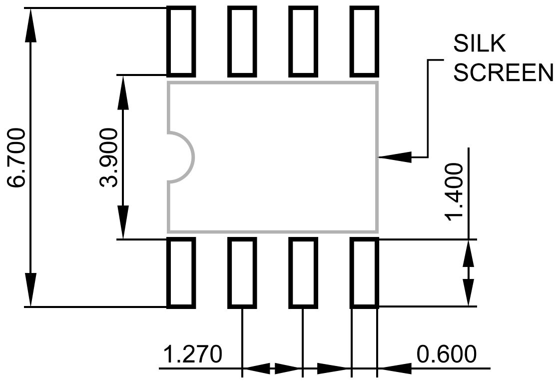


Units: millimeters

Notes:
This drawing is for general information only.
Drawing not to scale.

Evadrw0033B

SO8N Recommended Footprint



Units: millimeters

Notes:
This drawing is for general information only.
Drawing not to scale.

Evadrw0025r4

Ordering Information

Standard configuration

The standard configuration refers to the device variant that is mass-produced and offered at the most cost-efficient price.

Orderable Device	Status	Package	Pins	Temperature Range (Junction)
IS3752-S8-I	ACTIVE	SO8N	8	-40ºC to 85ºC (105ºC)

Special configurations

Special configurations are produced upon request only and are subject to a minimum order quantity. Please contact sales (Sales Contact) for more information.

Orderable Device	Status	Package	Pins	Temperature Range (Junction)
IS3752-S8-E	SPECIAL	SO8N (4.9×6 mm)	8	-40ºC to 105ºC (125ºC)
IS3752-S8-H				-40ºC to 125ºC (130ºC)
IS3752-W12-I		WLCSP (1.7×1.42 mm)	12	-40ºC to 85ºC (105ºC)
IS3752-W12-E				-40ºC to 105ºC (125ºC)
IS3752-W12-H				-40ºC to 125ºC (130ºC)
IS3752-U20-I		UFQFPN (3×3 mm)	20	-40ºC to 85ºC (105ºC)
IS3752-U20-E				-40ºC to 105ºC (125ºC)
IS3752-U20-H				-40ºC to 125ºC (130ºC)

ACTIVE: Product recommended for new designs.

SPECIAL: Available only for high-volume pre-orders. Contact sales.

NRND: Not recommended for new designs. Device is in production to support existing customers, but it is not recommended for use in new designs.

OBSOLETE: Production of the device has been discontinued.

Content

Appendix

Revision History

Document Revision

Date	Revision Code	Description
April 2026	ISDOC149A	- Initial Release.

Chip Revision

Chip Revision can be found in the `CHIP_REV` register of the memory map.

Date	Revision Code	Description
April 2026	**0**	- Initial Release.

Documentation Feedback

Feedback and error reporting on this document are very much appreciated. Please indicate the code or title of the document.

feedback@inacks.com

Sales Contact

For special order requirements, large volume orders, or scheduled orders, please contact our sales department at:

sales@inacks.com

Customization

INACKS can develop new products or customize existing ones to meet specific client needs. Please contact our engineering department at:

engineering@inacks.com

Trademarks

Disclaimer

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, INACKS does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. INACKS takes no responsibility for the content in this document if provided by an information source outside of INACKS.

In no event shall INACKS be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, INACKS's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of INACKS.

Right to make changes — INACKS reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — INACKS products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an INACKS product can reasonably be expected to result in personal injury, death or severe property or environmental damage. INACKS and its suppliers accept no liability for inclusion and/or use of INACKS products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Quick reference data — The Quick reference data is an extract of the product data given in the Limiting values and Characteristics sections of this document, and as such is not complete, exhaustive or legally binding.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. INACKS makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using INACKS products, and INACKS accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the INACKS product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

INACKS does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using INACKS products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). INACKS does not accept any liability in this respect.

Limiting values — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) will cause permanent damage to the device. Limiting values are stress ratings only and (proper) operation of the device at these or any other conditions above those given in the Recommended operating conditions section (if present) or the Characteristics sections of this document is not warranted. Constant or repeated exposure to limiting values will permanently and irreversibly affect the quality and reliability of the device.

Terms and conditions of commercial sale — INACKS products are sold subject to the general terms and conditions of commercial sale, as published at http://www.inacks.com/comercialsaleterms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. INACKS hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of INACKS products by customer.

No offer to sell or license — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Non-automotive qualified products — This INACKS product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. INACKS accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

Protocol Guidance Disclaimer: The information provided herein regarding the protocol is intended for guidance purposes only. While INACKS strive to provide accurate and up-to-date information, this content should not be considered a substitute for official protocol documentation. It is the responsibility of the client to consult and adhere to the official protocol documentation when designing or implementing systems based on this protocol.

INACKS make no representations or warranties, either expressed or implied, as to the accuracy, completeness, or reliability of the information contained in this document. INACKS shall not be held liable for any errors, omissions, or inaccuracies in the information or for any user's reliance on the information.

The client is solely responsible for verifying the suitability and compliance of the provided information with the official protocol standards and for ensuring that their implementation or usage of the protocol meets all required specifications and regulations. Any reliance on the information provided is strictly at the user's own risk.

Certification and Compliance Disclaimer: Please be advised that the product described herein has not been certified by any competent authority or organization responsible for protocol standards. INACKS do not guarantee that the chip meets any specific protocol compliance or certification standards.

It is the responsibility of the client to ensure that the final product incorporating this product is tested and certified according to the relevant protocol standards before use or commercialization. The certification process may result in the product passing or failing to meet these standards, and the outcome of such certification tests is beyond our control.

INACKS disclaim any liability for non-compliance with protocol standards and certification failures. The client acknowledges and agrees that they bear sole responsibility for any legal, compliance, or technical issues that arise due to the use of this product in their products, including but not limited to the acquisition of necessary protocol certifications.