Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Grado Universitario en Ingeniería Electrónica Industrial
y Automática 2022-2023

Trabajo Fin de Grado

Desarrollo del control de movimiento y subsistemas modulares de un rover

Alba González Fernández
Tutor: Germán León
Cotutor: Ramón Rubio

Julio 2023

TABLA DE CONTENIDO

Anexo 1. ESP32

ESP32 Datasheet

Espressif Systems

October 8, 2016

About This Guide

This document provides introduction to the specifications of ESP32 hardware.
The document structure is as follows:

Chapter	Title	Subject
Chapter 1	Overview	An overview of ESP32, including featured solutions, basic and advanced features, applications and development support
Chapter 2	Pin Definitions	Introduction to the pin layout and descriptions
Chapter 3	Functional Description	Description of the major functional modules
Chapter 4	Peripheral Interface	Description of the peripheral interfaces integrated on ESP32
Chapter 5	Electrical Characteristics	The electrical characteristics and data of ESP32
Chapter 6	Package Information	The package details of ESP32
Chapter 7	Supported Resources	The related documents and community resources for ESP32
Appendix	Touch Sensor	The touch sensor design and layout guidelines

Release Notes

Date	Version	Release notes
2016.08	V1.0	First release

Disclaimer and Copyright Notice

Contents

List of Tables

List of Figures

1. Overview

ESP32 is a single chip 2.4 GHz Wi-Fi and Bluetooth combo chip designed with TSMC ultra low power 40 nm technology. It is designed and optimized for the best power performance, RF performance, robustness, versatility, features and reliability, for a wide variety of applications, and different power profiles.

1.1 Featured Solutions

1.1.1 Ultra Low Power Solution

ESP32 is designed for mobile, wearable electronics, and Internet of Things (IoT) applications. It has many features of the state-of-the-art low power chips, including fine resolution clock gating, power modes, and dynamic power scaling.

For instance, in a low-power IoT sensor hub application scenario, ESP32 is woken up periodically and only when a specified condition is detected; low duty cycle is used to minimize the amount of energy that the chip expends. The output power of the power amplifier is also adjustable to achieve an optimal trade off between communication range, data rate and power consumption.

Note:
For more information, refer to Section 3.7 RTC and Low-Power Management.

1.1.2 Complete Integration Solution

ESP32 is the most integrated solution for Wi-Fi + Bluetooth applications in the industry with less than 10 external components. ESP32 integrates the antenna switch, RF balun, power amplifier, low noise receive amplifier, filters, and power management modules. As such, the entire solution occupies minimal Printed Circuit Board (PCB) area.

ESP32 uses CMOS for single-chip fully-integrated radio and baseband, and also integrates advanced calibration circuitries that allow the solution to dynamically adjust itself to remove external circuit imperfections or adjust to changes in external conditions.

As such, the mass production of ESP32 solutions does not require expensive and specialized Wi-Fi test equipment.

1.2 Basic Protocols

1.2.1 Wi-Fi

- 802.11 b/g/n/e/i
- 802.11 n (2.4 GHz), up to 150 Mbps
- 802.11 e: QoS for wireless multimedia technology
- WMM-PS, UAPSD
- A-MPDU and A-MSDU aggregation
- Block ACK

- Fragmentation and defragmentation
- Automatic Beacon monitoring/scanning
- 802.11 i security features: pre-authentication and TSN
- Wi-Fi Protected Access (WPA)/WPA2/WPA2-Enterprise/Wi-Fi Protected Setup (WPS)
- Infrastructure BSS Station mode/SoftAP mode
- Wi-Fi Direct (P2P), P2P Discovery, P2P Group Owner mode and P2P Power Management
- UMA compliant and certified
- Antenna diversity and selection

Note:
For more information, refer to Section 3.5 Wi-Fi.

1.2.2 Bluetooth

- Compliant with Bluetooth v4.2 BR/EDR and BLE specification
- Class-1, class-2 and class-3 transmitter without external power amplifier
- Enhanced power control
- +10 dBm transmitting power
- NZIF receiver with -98 dBm sensitivity
- Adaptive Frequency Hopping (AFH)
- Standard HCI based on SDIO/SPI/UART
- High speed UART HCI, up to 4 Mbps
- BT 4.2 controller and host stack
- Service Discover Protocol (SDP)
- General Access Profile (GAP)
- Security Manage Protocol (SMP)
- Bluetooth Low Energy (BLE)
- ATT/GATT
- HID
- All GATT-based profile supported
- SPP-Like GATT-based profile
- BLE Beacon
- A2DP/AVRCP/SPP, HSP/HFP, RFCOMM
- CVSD and SBC for audio codec
- Bluetooth Piconet and Scatternet

1.3 MCU and Advanced Features

1.3.1 CPU and Memory

- Xtensa® Dual-Core 32-bit LX6 microprocessors, up to 600 DMIPS
- 448 KByte ROM
- 520 KByte SRAM
- 16 KByte SRAM in RTC
- QSPI Flash/SRAM, up to 4 x 16 MBytes
- Power supply: 2.2 V to 3.6 V

1.3.2 Clocks and Timers

- Internal 8 MHz oscillator with calibration
- Internal RC oscillator with calibration
- External 2 MHz to 40 MHz crystal oscillator
- External 32 kHz crystal oscillator for RTC with calibration
- Two timer groups, including 2 x 64-bit timers and 1 x main watchdog in each group
- RTC timer with sub-second accuracy
- RTC watchdog

1.3.3 Advanced Peripheral Interfaces

- 12-bit SAR ADC up to 18 channels
- 2 × 8-bit D/A converters
- 10 × touch sensors
- Temperature sensor
- 4 × SPI
- 2 × I2S
- 2 × I2C
- 3 × UART
- 1 host (SD/eMMC/SDIO)
- 1 slave (SDIO/SPI)
- Ethernet MAC interface with dedicated DMA and IEEE 1588 support
- CAN 2.0
- IR (TX/RX)
- Motor PWM
- LED PWM up to 16 channels
- Hall sensor
- Ultra low power analog pre-amplifier

1.3.4 Security

- IEEE 802.11 standard security features all supported, including WFA, WPA/WPA2 and WAPI
- Secure boot
- Flash encryption
- 1024-bit OTP, up to 768-bit for customers
- Cryptographic hardware acceleration:
  - AES
  - HASH (SHA-2) library
  - RSA
  - ECC
  - Random Number Generator (RNG)

1.3.5 Development Support

- SDK Firmware for fast on-line programming
- Open source toolchains based on GCC

> Note:
> For more information, refer to Chapter 7 Supported Resources.

1.4 Application

- Generic low power IoT sensor hub
- Generic low power IoT loggers
- Video streaming from camera
- Over The Top (OTT) devices
- Music players
  - Internet music players
  - Audio streaming devices
- Wi-Fi enabled toys
  - Loggers
  - Proximity sensing toys
- Wi-Fi enabled speech recognition devices
- Audio headsets
- Smart power plugs
- Home automation
- Mesh network

- Industrial wireless control
- Baby monitors
- Wearable electronics
- Wi-Fi location-aware devices
- Security ID tags
- Healthcare
  - Proximity and movement monitoring trigger devices
  - Temperature sensing loggers

1.5 Block Diagram

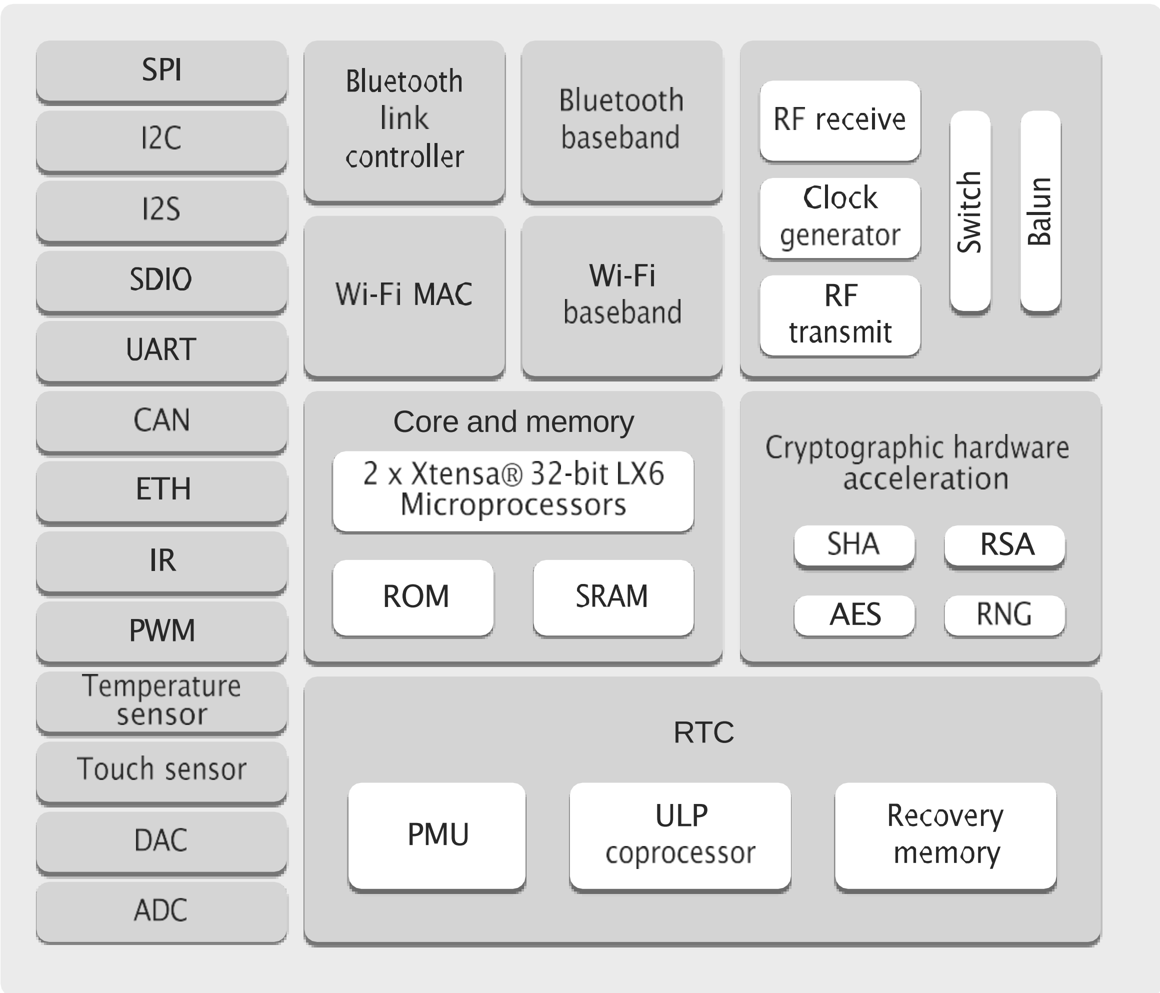


Figure 1: Function Block Diagram

2. Pin Definitions

2.1 Pin Layout

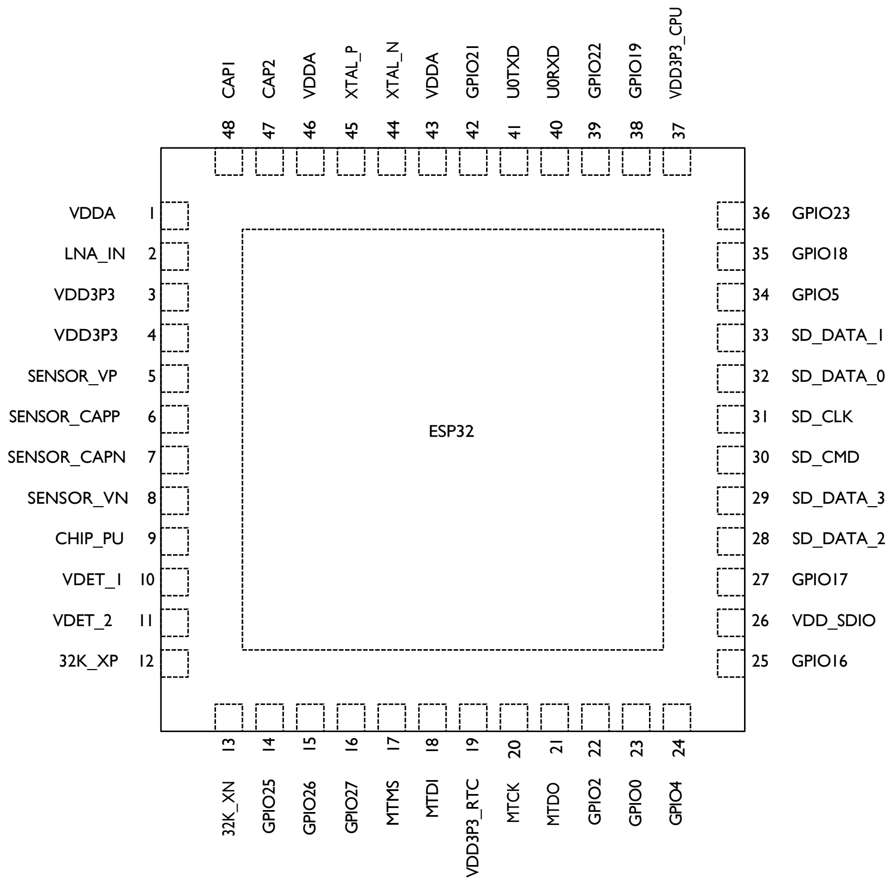


Figure 2: ESP32 Pin Layout

2.2 Pin Description

Table 1: Pin Description

Name	No.	Type	Function
			Analog
VDDA	1	P	Analog power supply (2.3V ~ 3.6V)
LNA_IN	2	I/O	RF input and output
VDD3P3	3	P	Amplifier power supply (2.3V ~ 3.6V)
VDD3P3	4	P	Amplifier power supply (2.3V ~ 3.6V)
			VDD3P3_RTC
SENSOR_VP	5	I	GPIO36, ADC_PRE_AMP, ADC1_CH0, RTC_GPIO0 Note: Connects 270 pF capacitor from SENSOR_VP to SENSOR_CAPP when used as ADC_PRE_AMP.

Name	No.	Type	Function
SENSOR_CAPP	6	I	GPIO37, ADC_PRE_AMP, ADC1_CH1, RTC_GPIO1 Note: Connects 270 pF capacitor from SENSOR_VP to SENSOR_CAPP when used as ADC_PRE_AMP.
SENSOR_CAPN	7	I	GPIO38, ADC1_CH2, ADC_PRE_AMP, RTC_GPIO2 Note: Connects 270 pF capacitor from SENSOR_VN to SENSOR_CAPN when used as ADC_PRE_AMP.
SENSOR_VN	8	I	GPIO39, ADC1_CH3, ADC_PRE_AMP, RTC_GPIO3 Note: Connects 270 pF capacitor from SENSOR_VN to SENSOR_CAPN when used as ADC_PRE_AMP.
CHIP_PU	9	I	Chip Enable (Active High) High: On, chip works properly Low: Off, chip works at the minimum power Note: Do not leave CHIP_PU pin floating
VDET_1	10	I	GPIO34, ADC1_CH6, RTC_GPIO4
VDET_2	11	I	GPIO35, ADC1_CH7, RTC_GPIO5
32K_XP	12	I/O	GPIO32, 32K_XP (32.768 kHz crystal oscillator input), ADC1_CH4, TOUCH9, RTC_GPIO9
32K_XN	13	I/O	GPIO33, 32K_XN (32.768 kHz crystal oscillator output), ADC1_CH5, TOUCH8, RTC_GPIO8
GPIO25	14	I/O	GPIO25, DAC_1, ADC2_CH8, RTC_GPIO6, EMAC_RXD0
GPIO26	15	I/O	GPIO26, DAC_2, ADC2_CH9, RTC_GPIO7, EMAC_RXD1
GPIO27	16	I/O	GPIO27, ADC2_CH7, TOUCH7, RTC_GPIO17, EMAC_RX_DV
MTMS	17	I/O	GPIO14, ADC2_CH6, TOUCH6, RTC_GPIO16, MTMS, HSPICLK, HS2_CLK, SD_CLK, EMAC_TXD2
MTDI	18	I/O	GPIO12, ADC2_CH5, TOUCH5, RTC_GPIO15, MTDI, HSPIQ, HS2_DATA2, SD_DATA2, EMAC_TXD3
VDD3P3_RTC	19	P	RTC IO power supply input (1.8V - 3.3V)
MTCK	20	I/O	GPIO13, ADC2_CH4, TOUCH4, RTC_GPIO14, MTCK, HSPID, HS2_DATA3, SD_DATA3, EMAC_RX_ER
MTDO	21	I/O	GPIO15, ADC2_CH3, TOUCH3, RTC_GPIO13, MTDO, HSPICS0, HS2_CMD, SD_CMD, EMAC_RXD3
GPIO2	22	I/O	GPIO2, ADC2_CH2, TOUCH2, RTC_GPIO12, HSPIWP, HS2_DATA0, SD_DATA0
GPIO0	23	I/O	GPIO0, ADC2_CH1, TOUCH1, RTC_GPIO11, CLK_OUT1, EMAC_TX_CLK
GPIO4	24	I/O	GPIO4, ADC2_CH0, TOUCH0, RTC_GPIO10, HSPIHD, HS2_DATA1, SD_DATA1, EMAC_TX_ER
VDD_SDIO			
GPIO16	25	I/O	GPIO16, HS1_DATA4, U2RXD, EMAC_CLK_OUT
VDD_SDIO	26	P	1.8V or 3.3V power supply output
GPIO17	27	I/O	GPIO17, HS1_DATA5, U2TXD, EMAC_CLK_OUT_180
SD_DATA_2	28	I/O	GPIO9, SD_DATA2, SPIHD, HS1_DATA2, U1RXD
SD_DATA_3	29	I/O	GPIO10, SD_DATA3, SPIWP, HS1_DATA3, U1TXD
SD_CMD	30	I/O	GPIO11, SD_CMD, SPICS0, HS1_CMD, U1RTS
SD_CLK	31	I/O	GPIO6, SD_CLK, SPICLK, HS1_CLK, U1CTS

Name	No.	Type	Function
SD_DATA_0	32	I/O	GPIO7, SD_DATA0, SPIQ, HS1_DATA0, U2RTS
SD_DATA_1	33	I/O	GPIO8, SD_DATA1, SPID, HS1_DATA1, U2CTS
VDD3P3_CPU			
GPIO5	34	I/O	GPIO5, VSPICS0, HS1_DATA6, EMAC_RX_CLK
GPIO18	35	I/O	GPIO18, VSPICLK, HS1_DATA7
GPIO23	36	I/O	GPIO23, VSPID, HS1_STROBE
VDD3P3_CPU	37	P	CPU IO power supply input (1.8V - 3.3V)
GPIO19	38	I/O	GPIO19, VSPIQ, U0CTS, EMAC_TXD0
GPIO22	39	I/O	GPIO22, VSPIWP, U0RTS, EMAC_TXD1
U0RXD	40	I/O	GPIO3, U0RXD, CLK_OUT2
U0TXD	41	I/O	GPIO1, U0TXD, CLK_OUT3, EMAC_RXD2
GPIO21	42	I/O	GPIO21, VSPIHD, EMAC_TX_EN
Analog			
VDDA	43	I/O	Analog power supply (2.3V - 3.6V)
XTAL_N	44	O	External crystal output
XTAL_P	45	I	External crystal input
VDDA	46	P	Digital power supply for PLL (2.3V - 3.6V)
CAP2	47	I	Connects with a 3 nF capacitor and 20 kΩ resistor in parallel to CAP1
CAP1	48	I	Connects with a 10 nF series capacitor to ground

2.3 Power Scheme

ESP32 digital pins are divided into three different power domains:

- VDD3P3_RTC
- VDD3P3_CPU
- VDD_SDIO

VDD3P3_RTC is also the input power supply for RTC and CPU. VDD3P3_CPU is also the input power supply for CPU.
VDD_SDIO connects to the output of an internal LDO, whose input is VDD3P3_RTC. When VDD_SDIO is connected to the same PCB net together with VDD3P3_RTC; the internal LDO is disabled automatically.

The internal LDO can be configured as 1.8V, or the same voltage as VDD3P3_RTC. It can be powered off via software to minimize the current of Flash/SRAM during the Deep-sleep mode.

Note:
It is required that the power supply of VDD3P3_RTC, VDD3P3_CPU and analog must be stable before the pin CHIP_PU is set at high level.

2.4 Strapping Pins

ESP32 has 6 strapping pins:

- MTDI/GPIO12: internal pull-down
- GPIO0: internal pull-up
- GPIO2: internal pull-down
- GPIO4: internal pull-down
- MTDO/GPIO15: internal pull-up
- GPIO5: internal pull-up

Software can read the value of these 6 bits from the register "GPIO_STRAPPING".

During the chip power-on reset, the latches of the strapping pins sample the voltage level as strapping bits of "0" or "1", and hold these bits until the chip is powered down or shut down. The strapping bits configure the device boot mode, the operating voltage of VDD_SDIO and other system initial settings.

Each strapping pin is connected with its internal pull-up/pull-down during the chip reset. Consequently, if a strapping pin is unconnected or the connected external circuit is high-impendence, the internal weak pull-up/pull-down will determine the default input level of the strapping pins.

To change the strapping bit values, users can apply the external pull-down/pull-up resistances, or apply the host MCU's GPIOs to control the voltage level of these pins when powering on ESP32.

After reset, the strapping pins work as the normal functions pins.

Refer to Table 2 for detailed boot modes configuration by strapping pins.

Table 2: Strapping Pins

<table>
<tr><th colspan="6">Voltage of Internal LDO (VDD_SDIO)</th></tr>
<tr><td>Pin</td><td>Default</td><td colspan="2">3.3V</td><td colspan="2">1.8V</td></tr>
<tr><td>MTDI</td><td>Pull-down</td><td colspan="2">0</td><td colspan="2">1</td></tr>
<tr><th colspan="6">Booting Mode</th></tr>
<tr><td>Pin</td><td>Default</td><td colspan="2">SPI Boot</td><td colspan="2">Download Boot</td></tr>
<tr><td>GPIO0</td><td>Pull-up</td><td colspan="2">1</td><td colspan="2">0</td></tr>
<tr><td>GPIO2</td><td>Pull-down</td><td colspan="2">Don't-care</td><td colspan="2">0</td></tr>
<tr><th colspan="6">Debugging Log on U0TXD During Booting</th></tr>
<tr><td>Pin</td><td>Default</td><td colspan="2">U0TXD Toggling</td><td colspan="2">U0TXD Silent</td></tr>
<tr><td>MTDO</td><td>Pull-up</td><td colspan="2">1</td><td colspan="2">0</td></tr>
<tr><th colspan="6">Timing of SDIO Slave</th></tr>
<tr><td>Pin</td><td>Default</td><td>Falling-edge Input Falling-edge Output</td><td>Falling-edge Input Rising-edge Output</td><td>Rising-edge Input Falling-edge Output</td><td>Rising-edge Input Rising-edge Output</td></tr>
<tr><td>MTDO</td><td>Pull-up</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td>GPIO5</td><td>Pull-up</td><td>0</td><td>1</td><td>0</td><td>1</td></tr>
</table>

Note:

Firmware can configure register bits to change the setting of "Voltage of Internal LDO (VDD_SDIO)" and "Timing of SDIO Slave" after booting.

3. Functional Description

This chapter describes the functions implemented in ESP32.

3.1 CPU and Memory

3.1.1 CPU

ESP32 contains two low-power Xtensa® 32-bit LX6 microprocessors with the following features.

- 7-stage pipeline to support the clock frequency of up to 240 MHz
- 16/24-bit Instruction Set provides high code-density
- Support Floating Point Unit
- Support DSP instructions, such as 32-bit Multiplier, 32-bit Divider, and 40-bit MAC
- Support 32 interrupt vectors from about 70 interrupt sources

The dual CPUs interface through:

- Xtensa RAM/ROM Interface for instruction and data
- Xtensa Local Memory Interface for fast peripheral register access
- Interrupt with external and internal sources
- JTAG interface for debugging

3.1.2 Internal Memory

ESP32's internal memory includes:

- 448 KBytes ROM for booting and core functions
- 520 KBytes on-chip SRAM for data and instruction
- 8 KBytes SRAM in RTC, which is called RTC SLOW Memory and can be used for co-processor accessing during the Deep-sleep mode
- 8 KBytes SRAM in RTC, which is called RTC FAST Memory and can be used for data storage and main CPU during RTC Boot from the Deep-sleep mode
- 1 Kbit of EFUSE, of which 256 bits are used for the system (MAC address and chip configuration) and the remaining 768 bits are reserved for customer applications, including Flash-Encryption and Chip-ID

3.1.3 External Flash and SRAM

ESP32 supports 4 x 16 MBytes of external QSPI Flash and SRAM with hardware encryption based on AES to protect developer's programs and data.

ESP32 accesses external QSPI Flash and SRAM by the high-speed caches

- Up to 16 MBytes of external Flash are memory mapped into the CPU code space, supporting 8-bit, 16-bit and 32-bit access. Code execution is supported.

- Up to 8 MBytes of external Flash/SRAM are memory mapped into the CPU data space, supporting 8-bit, 16-bit and 32-bit access. Data read is supported on the Flash and SRAM. Data write is supported on the SRAM.

3.1.4 Memory Map

The structure of address mapping is shown in Figure 3. The memory and peripherals mapping of ESP32 is shown in Table 3.

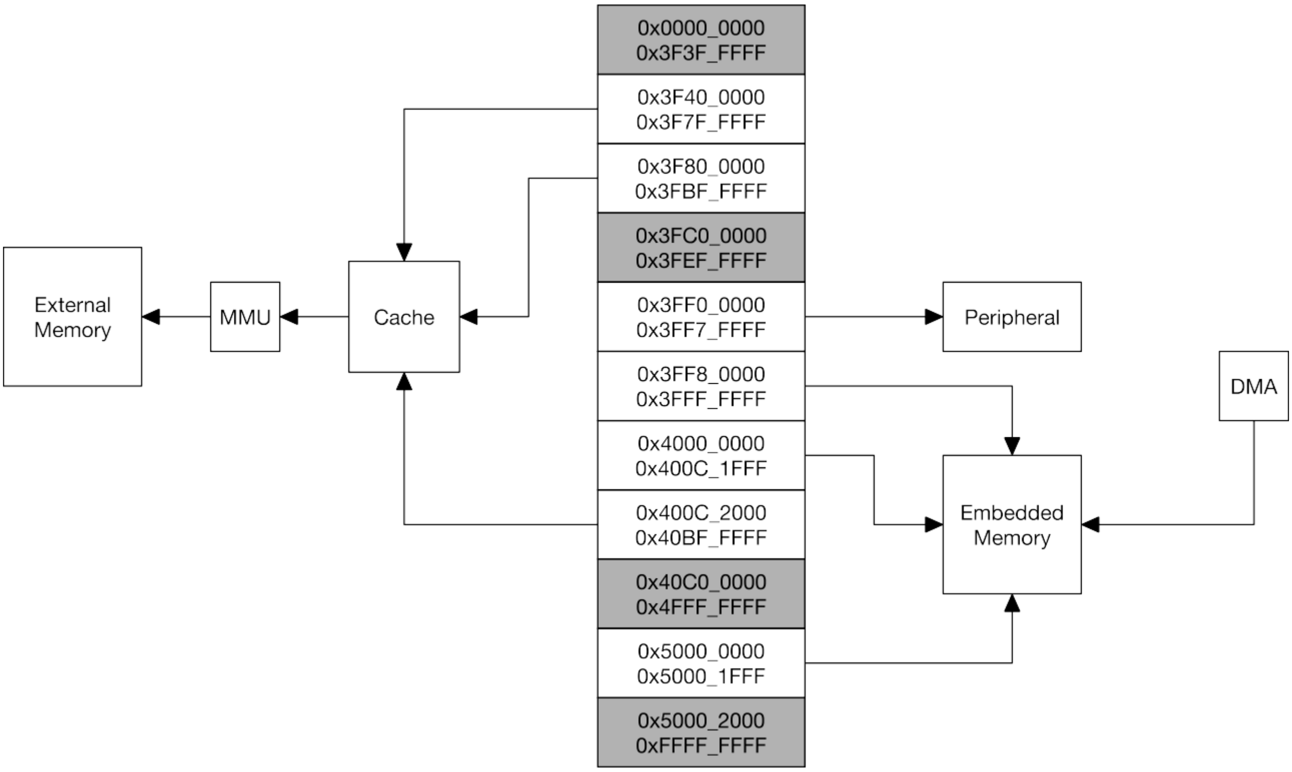


Figure 3: Address Mapping Structure

Table 3: Memory and Peripheral Mapping

Category	Target	Start Address	End Address	Size
Embedded Memory	Internal ROM 0	0x4000_0000	0x4005_FFFF	384 KB
	Internal ROM 1	0x3FF9_0000	0x3FF9_FFFF	64 KB
	Internal SRAM 0	0x4007_0000	0x4009_FFFF	192 KB
	Internal SRAM 1	0x3FFE_0000	0x3FFF_FFFF	128 KB
		0x400A_0000	0x400B_FFFF	
	Internal SRAM 2	0x3FFA_E000	0x3FFD_FFFF	200 KB
	RTC FAST Memory	0x3FF8_0000	0x3FF8_1FFF	8 KB
		0x400C_0000	0x400C_1FFF	
	RTC SLOW Memory	0x5000_0000	0x5000_1FFF	8 KB
External Memory	External Flash	0x3F40_0000	0x3F7F_FFFF	4 MB
		0x400C_2000	0x40BF_FFFF	11 MB
				248 KB
	External SRAM	0x3F80_0000	0x3FBF_FFFF	4 MB

Category	Target	Start Address	End Address	Size
Peripheral	DPort Register	0x3FF0_0000	0x3FF0_0FFF	4 KB
	AES Accelerator	0x3FF0_1000	0x3FF0_1FFF	4 KB
	RSA Accelerator	0x3FF0_2000	0x3FF0_2FFF	4 KB
	SHA Accelerator	0x3FF0_3000	0x3FF0_3FFF	4 KB
	Secure Boot	0x3FF0_4000	0x3FF0_4FFF	4 KB
	Cache MMU Table	0x3FF1_0000	0x3FF1_3FFF	16 KB
	PID Controller	0x3FF1_F000	0x3FF1_FFFF	4 KB
	UART0	0x3FF4_0000	0x3FF4_0FFF	4 KB
	SPI1	0x3FF4_2000	0x3FF4_2FFF	4 KB
	SPI0	0x3FF4_3000	0x3FF4_3FFF	4 KB
	GPIO	0x3FF4_4000	0x3FF4_4FFF	4 KB
	RTC	0x3FF4_8000	0x3FF4_8FFF	4 KB
	IO MUX	0x3FF4_9000	0x3FF4_9FFF	4 KB
	SDIO Slave	0x3FF4_B000	0x3FF4_BFFF	4 KB
	UDMA1	0x3FF4_C000	0x3FF4_CFFF	4 KB
	I2S0	0x3FF4_F000	0x3FF4_FFFF	4 KB
	UART1	0x3FF5_0000	0x3FF5_0FFF	4 KB
	I2C0	0x3FF5_3000	0x3FF5_3FFF	4 KB
	UDMA0	0x3FF5_4000	0x3FF5_4FFF	4 KB
	SDIO Slave	0x3FF5_5000	0x3FF5_5FFF	4 KB
	RMT	0x3FF5_6000	0x3FF5_6FFF	4 KB
	PCNT	0x3FF5_7000	0x3FF5_7FFF	4 KB
	SDIO Slave	0x3FF5_8000	0x3FF5_8FFF	4 KB
	LED PWM	0x3FF5_9000	0x3FF5_9FFF	4 KB
	Efuse Controller	0x3FF5_A000	0x3FF5_AFFF	4 KB
	Flash Encryption	0x3FF5_B000	0x3FF5_BFFF	4 KB
	PWM0	0x3FF5_E000	0x3FF5_EFFF	4 KB
	TIMG0	0x3FF5_F000	0x3FF5_FFFF	4 KB
	TIMG1	0x3FF6_0000	0x3FF6_0FFF	4 KB
	SPI2	0x3FF6_4000	0x3FF6_4FFF	4 KB
	SPI3	0x3FF6_5000	0x3FF6_5FFF	4 KB
	SYSCON	0x3FF6_6000	0x3FF6_6FFF	4 KB
	I2C1	0x3FF6_7000	0x3FF6_7FFF	4 KB
	SDMMC	0x3FF6_8000	0x3FF6_8FFF	4 KB
	EMAC	0x3FF6_9000	0x3FF6_AFFF	8 KB
	PWM1	0x3FF6_C000	0x3FF6_CFFF	4 KB
	I2S1	0x3FF6_D000	0x3FF6_DFFF	4 KB
	UART2	0x3FF6_E000	0x3FF6_EFFF	4 KB
	PWM2	0x3FF6_F000	0x3FF6_FFFF	4 KB
	PWM3	0x3FF7_0000	0x3FF7_0FFF	4 KB
	RNG	0x3FF7_5000	0x3FF7_5FFF	4 KB

3.2 Timers and Watchdogs

3.2.1 64-bit Timers

There are four general-purpose timers embedded in the ESP32. They are all 64-bit generic timers which are based on 16-bit prescalers and 64-bit auto-reload-capable up/downcounters.

The timers feature:

- A 16-bit clock prescaler, from 2 to 65536
- A 64-bit time-base counter
- Configurable up/down time-base counter: incrementing or decrmenting
- Halt and resume of time-base counter
- Auto-reload at alarming
- Software-controlled instant reload
- Level and edge interrupt generation

3.2.2 Watchdog Timers

The ESP32 has three watchdog timers: one in each of the two timer modules (called the Main Watchdog Timer, or MWDT) and one in the RTC module (called the RTC Watchdog Timer, or RWDT). These watchdog timers are intended to recover from an unforeseen fault, causing the application program to abandon its normal sequence. A watchdog timer has 4 stages. Each stage may take one of three or four actions on expiry of a programmed time period for this stage unless the watchdog is fed or disabled. The actions are: interrupt, CPU reset, and core reset, and system reset. Only the RWDT can trigger the system reset, and is able to reset the entire chip, including the RTC itself. A timeout value can be set for each stage individually.

During Flash boot the RWDT and the first MWDT start automatically in order to detect and recover from booting problems.

The ESP32 watchdogs have the following features:

- 4 stages, each can be configured or disabled separately
- Programmable time period for each stage
- One of 3 or 4 possible actions (interrupt, CPU reset, core reset, and system reset) on expiration of each stage
- 32-bit expiry counter
- Write protection, to prevent the RWDT and MWDT configuration from being inadvertently altered
- SPI Flash boot protection
  If the boot process from an SPI Flash does not complete within a predetermined time period, the watchdog will reboot the entire system.

3.3 System Clocks

3.3.1 CPU Clock

Upon reset, an external crystal clock source (2 MHz ~ 60 MHz), is selected as the default CPU clock. The external crystal clock source also connects to a PLL to generate a high frequency clock (typically 160 MHz).

In addition to this, ESP32 has an internal 8 MHz oscillator, of which the accuracy is guaranteed by design and is stable over temperature (within 1% accuracy). Hence, the application can then select from the external crystal clock source, the PLL clock or the internal 8 MHz oscillator. The selected clock source drives the CPU clock, directly or after division, depending on the application.

3.3.2 RTC Clock

The RTC clock has five possible sources:

- external low speed (32 kHz) crystal clock
- external crystal clock divided by 4
- internal RC oscillator (typically about 150 kHz and adjustable)
- internal 8 MHz oscillator
- internal 31.25 kHz clock (derived from the internal 8 MHz oscillator divided by 256)

When the chip is in the normal power mode and needs faster CPU accessing, the application can choose the external high speed crystal clock divided by 4 or the internal 8 MHz oscillator. When the chip operates in the low power mode, the application chooses the external low speed (32 kHz) crystal clock, the internal RC clock or the internal 31.25 kHz clock.

3.3.3 Audio PLL Clock

The audio clock is generated by the ultra low noise fractional-N PLL. The output frequency of the audio PLL is programmable, from 16 MHz to 128 MHz, given by the following formula:

$$f_{\text{out}} = \frac{f_{\text{xtal}} N_{\text{div}}}{M_{\text{div}} 2^{K_{\text{div}}}}$$

where f_{out} is the output frequency, f_{xtal} is the frequency of the crystal oscillator, and N_{div}, M_{div} and K_{div} are all integer values, configurable by registers.

3.4 Radio

The ESP32 radio consists of the following main blocks:

- 2.4 GHz receiver
- 2.4 GHz transmitter
- bias and regulators
- balun and transmit-receive switch
- clock generator

3.4.1 2.4 GHz Receiver

The 2.4 GHz receiver down-converts the 2.4 GHz RF signal to quadrature baseband signals and converts them to the digital domain with 2 high-resolution, high-speed ADCs. To adapt to varying signal channel conditions, RF filters, Automatic Gain Control (AGC), DC offset cancelation circuits and baseband filters are integrated within ESP32.

3.4.2 2.4 GHz Transmitter

The 2.4 GHz transmitter up-converts the quadrature baseband signals to the 2.4 GHz RF signal, and drives the antenna with a high powered Complementary Metal Oxide Semiconductor (CMOS) power amplifier. The use of digital calibration further improves the linearity of the power amplifier, enabling state-of-the-art performance of delivering +20.5 dBm of average power for 802.11b transmission and +17 dBm for 802.11n transmission.
Additional calibrations are integrated to cancel any imperfections of the radio, such as:

- Carrier leakage
- I/Q phase matching
- Baseband nonlinearities
- RF nonlinearities
- Antenna matching

These built-in calibration routines reduce the amount of time and required for product test and make test equipment unnecessary.

3.4.3 Clock Generator

The clock generator generates quadrature 2.4 GHz clock signals for the receiver and transmitter. All components of the clock generator are integrated on the chip, including all inductors, varactors, filters, regulators and dividers. The clock generator has built-in calibration and self test circuits. Quadrature clock phases and phase noise are optimized on-chip with patented calibration algorithms to ensure the best performance of the receiver and transmitter.

3.5 Wi-Fi

ESP32 implements TCP/IP, full 802.11 b/g/n/e/i WLAN MAC protocol, and Wi-Fi Direct specification. It supports Basic Service Set (BSS) STA and SoftAP operations under the Distributed Control Function (DCF) and P2P group operation compliant with the latest Wi-Fi P2P protocol.

Passive or active scanning, as well as the P2P discovery procedure are performed autonomously when initiated by appropriate commands. Power management is handled with minimum host interaction to minimize active duty period.

3.5.1 Wi-Fi Radio and Baseband

The ESP32 Wi-Fi Radio and Baseband support the following features:

- 802.11b and 802.11g data-rates
- 802.11n MCS0-7 in both 20 MHz and 40 MHz bandwidth
- 802.11n MCS32
- 802.11n 0.4 μS guard-interval
- Data-rate up to 150 Mbps
- Receiving STBC 2x1
- Up to 21 dBm transmitting power
- Adjustable transmitting power

- Antenna diversity and selection (software-managed hardware)

3.5.2 Wi-Fi MAC

The ESP32 Wi-Fi MAC applies low level protocol functions automatically as follows:

- Request To Send (RTS), Clear To Send (CTS) and Acknowledgement (ACK/BA)
- Fragmentation and defragmentation
- Aggregation AMPDU and AMSDU
- WMM, U-APSD
- 802.11 e: QoS for wireless multimedia technology
- CCMP (CBC-MAC, counter mode), TKIP (MIC, RC4), WAPI (SMS4), WEP (RC4) and CRC
- Frame encapsulation (802.11h/RFC 1042)
- Automatic beacon monitoring/scanning

3.5.3 Wi-Fi Firmware

The ESP32 Wi-Fi Firmware provides the following functions:

- Infrastructure BSS Station mode / P2P mode / softAP mode support
- P2P Discovery, P2P Group Owner, P2P Group Client and P2P Power Management
- WPA/WPA2-Enterprise and WPS driver
- Additional 802.11i security features such as pre-authentication and TSN
- Open interface for various upper layer authentication schemes over EAP such as TLS, PEAP, LEAP, SIM, AKA or customer specific
- Clock/power gating combined with 802.11-compliant power management dynamically adapted to current connection condition providing minimal power consumption
- Adaptive rate fallback algorithm sets the optimal transmission rate and transmit power based on actual Signal Noise Ratio (SNR) and packet loss information
- Automatic retransmission and response on MAC to avoid packet discarding on slow host environment

3.5.4 Packet Traffic Arbitration (PTA)

ESP32 has a configurable Packet Traffic Arbitration (PTA) that provides flexible and exact timing Bluetooth co-existence support. It is a combination of both Frequency Division Multiplexing (FDM) and Time Division Multiplexing (TDM), and coordinates the protocol stacks.

- It is preferable that Wi-Fi works in the 20 MHz bandwidth mode to decrease its interference with BT.
- BT applies AFH (Adaptive Frequency Hopping) to avoid using the channels within Wi-Fi bandwidth.
- Wi-Fi MAC limits the time duration of Wi-Fi packets, and does not transmit the long Wi-Fi packets by the lowest data-rates.
- Normally BT packets are of higher priority than normal Wi-Fi packets.
- Protect the critical Wi-Fi packets, including beacon transmission and receiving, ACK/BA transmission and receiving.

- Protect the highest BT packets, including inquiry response, page response, LMP data and response, park beacons, the last poll period, SCO/eSCO slots, and BLE event sequence.
- Wi-Fi MAC applies CTS-to-self packet to protect the time duration of BT transfer.
- In the P2P Group Own (GO) mode, Wi-Fi MAC applies a Notice of Absence (NoA) packet to disable Wi-Fi transfer to reserve time for BT.
- In the STA mode, Wi-Fi MAC applies a NULL packet with the Power-Save bit to disable WiFi transfer to reserve time for BT.

3.6 Bluetooth

ESP32 integrates Bluetooth link controller and Bluetooth baseband, which carry out the baseband protocols and other low-level link routines, such as modulation/demodulation, packets processing, bit stream processing, frequency hopping, etc.

3.6.1 Bluetooth Radio and Baseband

The ESP32 Bluetooth Radio and Baseband support the following features:

- Class-1, class-2 and class-3 transmit output powers and over 30 dB dynamic control range
- $\pi/4$ DQPSK and 8 DPSK modulation
- High performance in NZIF receiver sensitivity with over 98 dB dynamic range
- Class-1 operation without external PA
- Internal SRAM allows full speed data transfer, mixed voice and data, and full piconet operation
- Logic for forward error correction, header error control, access code correlation, CRC, demodulation, encryption bit stream generation, whitening and transmit pulse shaping
- ACL, SCO, eSCO and AFH
- A-law, μ-law and CVSD digital audio CODEC in PCM interface
- SBC audio CODEC
- Power management for low power applications
- SMP with 128-bit AES

3.6.2 Bluetooth Interface

- Provides UART HCI interface, up to 4 Mbps
- Provides SDIO / SPI HCI interface
- Provides I2C interface for the host to do configuration
- Provides PCM / I2S audio interface

3.6.3 Bluetooth Stack

The Bluetooth stack of ESP32 is compliant with Bluetooth v4.2 BR / EDR and BLE specification.

3.6.4 Bluetooth Link Controller

The link controller operates in three major states: standby, connection and sniff. It enables multi connection and other operations like inquiry, page, and secure simple pairing, and therefore enables Piconet and Scatternet. Below are the features:

- Classic Bluetooth
  - Device Discovery (inquiry and inquiry scan)
  - Connection establishment (page and page scan)
  - Multi connections
  - Asynchronous data reception and transmission
  - Synchronous links (SCO/eSCO)
  - Master/Slave Switch
  - Adaptive Frequency Hopping and Channel assessment
  - Broadcast encryption
  - Authentication and encryption
  - Secure Simple Pairing
  - Multi-point and scatternet management
  - Sniff mode
  - Connectionless Slave Broadcast (transmitter and receiver)
  - Enhanced power control
  - Ping
- Bluetooth Low Energy
  - Advertising
  - Scanning
  - Multiple connections
  - Asynchronous data reception and transmission
  - Adaptive Frequency Hopping and Channel assessment
  - Connection parameter update
  - Date Length Extension
  - Link Layer Encryption
  - LE Ping

3.7 RTC and Low-Power Management

With the advanced power management technologies, ESP32 can switch between different power modes (see Table 4).

- Power mode
  - Active mode: The chip radio is powered on. The chip can receive, transmit, or listen.
  - Modem-sleep mode: The CPU is operational and the clock is configurable. The Wi-Fi/Bluetooth baseband and radio are disabled.
  - Light-sleep mode: The CPU is paused. The RTC and ULP-coprocessor are running. Any wake-up events (MAC, host, RTC timer, or external interrupts) will wake up the chip.
  - Deep-sleep mode: Only RTC is powered on. Wi-Fi and Bluetooth connection data are stored in RTC memory. The ULP-coprocessor can work.
  - Hibernation mode: The internal 8MHz oscillator and ULP-coprocessor are disabled. The RTC recovery memory are power-down. Only one RTC timer on the slow clock and some RTC GPIOs are active. The RTC timer or the RTC GPIOs can wake up the chip from the Hibernation mode.
- Sleep Pattern
  - Association sleep pattern: The power mode switches between the active mode and Modem-sleep/Light-sleep mode during this sleep pattern. The CPU, Wi-Fi, Bluetooth, and radio are woken up at predetermined intervals to keep Wi-Fi/BT connections alive.
  - ULP sensor-monitored pattern: The main CPU is in the Deep-sleep mode. The ULP co-processor does sensor measurements and wakes up the main system, based on the measured data from sensors.

Table 4: Functionalities Depending on the Power Modes

Power mode	Active	Modem-sleep	Light-sleep	Deep-sleep	Hibernation
Sleep pattern	Association sleep pattern			ULP sensor-monitored pattern	-
CPU	ON	PAUSE	ON	OFF	OFF
Wi-Fi/BT baseband and radio	ON	OFF	OFF	OFF	OFF
RTC	ON	ON	ON	ON	OFF
ULP co-processor	ON	ON	ON	ON/OFF	OFF

The power consumption varies with different power modes/sleep patterns and work status of functional modules (see Table 5).

Table 5: Power Consumption by Power Modes

Power mode	Description	Power consumption
Active (RF working)	Wi-Fi Tx packet 13 dBm ~ 21 dBm	160 ~ 260 mA
	Wi-Fi / BT Tx packet 0 dBm	120 mA
	Wi-Fi / BT Rx and listening	80 ~ 90 mA
	Association sleep pattern (by Light-sleep)	0.9 mA@DTIM3, 1.2 mA@DTIM1
Modem-sleep	The CPU is powered on.	Max speed: 20 mA
		Normal speed: 5 ~ 10 mA
		Slow speed: 3 mA
Light-sleep	-	0.8 mA
Deep-sleep	The ULP co-processor is powered on.	0.15 mA
	ULP sensor-monitored pattern	25 μA @1% duty
	RTC timer + RTC memory	10 μA
Hibernation	RTC timer only	2.5 μA

Note:
For more information about RF power consumption, refer to Section 5.3 RF Power Consumption Specifications.

4. Peripheral Interface

4.1 General Purpose Input / Output Interface (GPIO)

ESP32 has 48 GPIO pins which can be assigned to various functions by programming the appropriate registers. There are several kinds of GPIOs: digital only GPIOs, analog enabled GPIOs, capacitive touch enabled GPIOs, etc. Analog enabled GPIOs can be configured as digital GPIOs. Capacitive touch enabled GPIOs can be configured as digital GPIOs.

Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance. When configured as an input, the input value can be read through the register. The input can also be set to edge-trigger or level-trigger to generate CPU interrupts. In short, the digital IO pins are bi-directional, non-inverting and tristate, including input and output buffer with tristate control. These pins can be multiplexed with other functions, such as the SDIO interface, UART, SI, etc. For low power operations, the GPIOs can be set to hold their states.

4.2 Analog-to-Digital Converter (ADC)

ESP32 integrates 12-bit SAR ADCs and supports measurements on 18 channels (analog enabled pins). Some of these pins can be used to build a programmable gain amplifier which is used for the measurement of small analog signals. The ULP-coprocessor in ESP32 is also designed to measure the voltages while operating in the sleep mode, to enable low power consumption; the CPU can be woken up by a threshold setting and/or via other triggers.

With the appropriate setting, the ADCs and the amplifier can be configured to measure voltages for a maximum of 18 pins.

4.3 Ultra Low Noise Analog Pre-Amplifier

ESP32 integrates an ultra low noise analog pre-amplifier that outputs to the ADC. The amplification ratio is given by the size of a pair of sampling capacitors that are placed off-chip. By using a larger capacitor, the sampling noise is reduced, but the settling time will be increased. The amplification ratio is also limited by the amplifier which peaks at about 60 dB gain.

4.4 Hall Sensor

ESP32 integrates a Hall sensor based on an N-carrier resistor. When the chip is in the magnetic field, the Hall sensor develops a small voltage laterally on the resistor, which can be directly measured by the ADC, or amplified by the ultra low noise analog pre-amplifier and then measured by the ADC.

4.5 Digital-to-Analog Converter (DAC)

Two 8-bit DAC channels can be used to convert two digital signals into two analog voltage signal outputs. The design structure is composed of integrated resistor strings and a buffer. This dual DAC supports power supply as input voltage reference and can drive other circuits. The dual channels support independent conversions.

4.6 Temperature Sensor

The temperature sensor generates a voltage that varies with temperature. The voltage is internally converted via an analog-to-digital converter into a digital code.

The temperature sensor has a range of -40°C to 125°C. As the offset of the temperature sensor varies from chip to chip due to process variation, together with the heat generated by the Wi-Fi circuitry itself (which affects measurements), the internal temperature sensor is only suitable for applications that detect temperature changes instead of absolute temperatures and for calibration purposes as well.

However, if the user calibrates the temperature sensor and uses the device in a minimally powered-on application, the results could be accurate enough.

4.7 Touch Sensor

ESP32 offers 10 capacitive sensing GPIOs which detect capacitive variations introduced by the GPIO's direct contact or close proximity with a finger or other objects. The low noise nature of the design and high sensitivity of the circuit allow relatively small pads to be used. Arrays of pads can also be used so that a larger area or more points can be detected. The 10 capacitive sensing GPIOs are listed in Table 6.

Table 6: Capacitive Sensing GPIOs Available on ESP32

Capacitive sensing signal name	Pin name
T0	GPIO4
T1	GPIO0
T2	GPIO2
T3	MTDO
T4	MTCK
T5	MTD1
T6	MTMS
T7	GPIO27
T8	32K_XN
T9	32K_XP

Note:
For more information about the touch sensor design and layout, refer to Appendix A Touch Sensor.

4.8 Ultra-Lower-Power Coprocessor

The ULP processor and RTC memory remains powered on during the Deep-sleep mode. Hence, the developer can store a program for the ULP processor in the RTC memory to access the peripheral devices, internal timers and internal sensors during the Deep-sleep mode. This is useful for designing applications where the CPU needs to be woken up by an external event, or timer, or a combination of these events, while maintaining minimal power consumption.

4.9 Ethernet MAC Interface

An IEEE-802.3-2008-compliant Media Access Controller (MAC) is provided for Ethernet LAN communications. ESP32 requires an external physical interface device (PHY) to connect to the physical LAN bus (twisted-pair, fiber, etc.). The PHY is connected to ESP32 through 17 signals of MII or 9 signals of RMII. With the Ethernet MAC (EMAC) interface, the following features are supported:

- 10 Mbps and 100 Mbps rates
- Dedicated DMA controller allowing high-speed transfer between the dedicated SRAM and Ethernet MAC
- Tagged MAC frame (VLAN support)
- Half-duplex (CSMA/CD) and full-duplex operation
- MAC control sublayer (control frames)
- 32-bit CRC generation and removal
- Several address filtering modes for physical and multicast address (multicast and group addresses)
- 32-bit status code for each transmitted or received frame
- Internal FIFOs to buffer transmit and receive frames. The transmit FIFO and the receive FIFO are both 512 words (32-bit)
- Hardware PTP (precision time protocol) in accordance with IEEE 1588 2008 (PTP V2)
- 25 MHz/50 MHz clock output

4.10 SD/SDIO/MMC Host Controller

An SD/SDIO/MMC host controller is available on ESP32 which supports the following features:

- Secure Digital memory (SD mem Version 3.0 and Version 3.01)
- Secure Digital I/O (SDIO Version 3.0)
- Consumer Electronics Advanced Transport Architecture (CE-ATA Version 1.1)
- Multimedia Cards (MMC Version 4.41, eMMC Version 4.5 and Version 4.51)

The controller allows clock output at up to 80 MHz and in three different data-bus modes: 1-bit, 4-bit and 8-bit. It supports two SD/SDIO/MMC4.41 cards in 4-bit data-bus mode. It also supports one SD card operating at 1.8 V level.

4.11 Universal Asynchronous Receiver Transmitter (UART)

ESP32 has three UART interfaces, i.e. UART0, UART1 and UART2, which provide asynchronous communication (RS232 and RS485) and IrDA support, and communicate at up to 5 Mbps. UART provides hardware management of the CTS and RTS signals and software flow control (XON and XOFF). All of the interfaces can be accessed by the DMA controller or directly by CPU.

4.12 I2C Interface

ESP32 has two I2C bus interfaces which can serve as I2C master or slave depending on the user's configuration. The I2C interfaces support:

- Standard mode (100 kbit/s)
- Fast mode (400 kbit/s)
- Up to 5 MHz, but constrained by SDA pull up strength
- 7-bit/10-bit addressing mode
- Dual addressing mode

Users can program command registers to control I2C interfaces to have more flexibility.

4.13 I2S Interface

Two standard I2S interfaces are available in ESP32. They can be operated in the master or slave mode, in full duplex and half-duplex communication modes, and can be configured to operate with an 8-/16-/32-/40-/48-bit resolution as input or output channels. BCK clock frequency from 10 kHz up to 40 MHz are supported. When one or both of the I2S interfaces are configured in the master mode, the master clock can be output to the external DAC/CODEC.

Both of the I2S interfaces have dedicated DMA controllers. PDM and BT PCM interfaces are supported.

4.14 Infrared Remote Controller

The infrared remote controller supports eight channels of infrared remote transmission and receiving. Through programming the pulse waveform, it supports various infrared protocols. Eight channels share a 512 x 32-bit block of memory to store the transmitting or receiving waveform.

4.15 Pulse Counter

The pulse counter captures pulse and counts pulse edges through seven modes. It has 8 channels; each channel captures four signals at a time. The four input signals include two pulse signals and two control signals. When the counter reaches a defined threshold, an interrupt is generated.

4.16 Pulse Width Modulation (PWM)

The Pulse Width Modulation (PWM) controller can be used for driving digital motors and smart lights. The controller consists of PWM timers, the PWM operator and a dedicated capture sub-module. Each timer provides timing in synchronus or independent form, and each PWM operator generates the waveform for one PWM channel. The dedicated capture sub-module can accurately capture external timing events.

4.17 LED PWM

The LED PWM controller can generate 16 independent channels of digital waveforms with the configurable periods and configurable duties.

The 16 channels of digital waveforms operate at 80 MHz APB clock, among which 8 channels have the option of using the 8 MHz oscillator clock. Each channel can select a 20-bit timer with configurable counting range and its accuracy of duty can be up to 16 bits with the 1 ms period.

The software can change the duty immediately. Moreover, each channel supports step-by-step duty increasing or decreasing automatically. It is useful for the LED RGB color gradient generator.

4.18 Serial Peripheral Interface (SPI)

ESP32 features three SPIs (SPI, HSPI and VSPI) in slave and master modes in 1-line full-duplex and 1/2/4-line half-duplex communication modes. These SPIs also support the following general-purpose SPI features:

- 4 timing modes of the SPI format transfer that depend on the polarity (POL) and the phase (PHA)
- up to 80 MHz and the divided clocks of 80 MHz
- up to 64-Byte FIFO

All SPIs can also be used to connect to the external Flash/SRAM and LCD. Each SPI can be served by DMA controllers.

4.19 Accelerator

ESP32 is equipped with hardware accelerators of general algorithms, such as AES (FIPS PUB 197), SHA (FIPS PUB 180-4), RSA, and ECC, which support independent arithmetic such as Big Integer Multiplication and Big Integer Modular Multiplication. The maximum operation length for RSA, ECC, Big Integer Multiply and Big Integer Modular Multiplication is 4096 bits.

The hardware accelerators greatly improve operation speed and reduce software complexity. They also support code encryption and dynamic decryption which ensures that codes in the Flash will not be stolen.

5. Electrical Characteristics

Note:
The specifications in this charpter are tested in general condition: V_{BAT} = 3.3V, T_A = 27°C, unless otherwise specified.

5.1 Absolute Maximum Ratings

Table 7: Absolute Maximum Ratings

Parameter	Symbol	Min	Max	Unit
Input low voltage	V_{IL}	-0.3	0.25×V_{IO}	V
Input high voltage	V_{IH}	0.75×V_{IO}	3.3	V
Input leakage current	I_{IL}	-	50	nA
Output low voltage	V_{OL}	-	0.1×V_{IO}	V
Output high voltage	V_{OH}	0.8×V_{IO}	-	V
Input pin capacitance	C_{pad}	-	2	pF
VDDIO	V_{IO}	1.8	3.3	V
Maximum drive capability	I_{MAX}	-	12	mA
Storage temperature range	T_{STR}	-40	150	°C

5.2 Recommended Operating Conditions

Table 8: Recommended Operating Conditions

Parameter	Symbol	Min	Typ	Max	Unit
Battery regulator supply voltage	V_{BAT}	2.8	3.3	3.6	V
I/O supply voltage	V_{IO}	1.8	3.3	3.6	V
Operating temperature range	T_{OPR}	-40	-	125	°C
CMOS low level input voltage	V_{IL}	0	-	0.3 x V_{IO}	V
CMOS high level input voltage	V_{IH}	0.7 x V_{IO}	-	V_{IO}	V
CMOS threshold voltage	V_{TH}	-	0.5 x V_{IO}	-	V

5.3 RF Power Consumption Specifications

The current consumption measurements are conducted with 3.0 V supply and 25°C ambient, at antenna port. All the transmitters' measurements are based on 90% duty cycle and continuous transmit mode.

Table 9: RF Power Consumption Specifications

Mode	Min	Typ	Max	Unit
Transmit 802.11b, DSSS 1 Mbps, POUT = +19.5 dBm	-	225	-	mA
Transmit 802.11b, CCK 11 Mbps, POUT = +18.5 dBm	-	205	-	mA
Transmit 802.11g, OFDM 54 Mbps, POUT = +16 dBm	-	160	-	mA
Transmit 802.11n, MCS7, POUT = +14 dBm	-	152	-	mA
Receive 802.11b, packet length = 1024 bytes, -80 dBm	-	85	-	mA
Receive 802.11g, packet length = 1024 bytes, -70 dBm	-	85	-	mA
Receive 802.11n, packet length = 1024 bytes, -65 dBm	-	80	-	mA
Receive 802.11n HT40, packet length = 1024 bytes, -65 dBm	-	80	-	mA

5.4 Wi-Fi Radio

Table 10: Wi-Fi Radio Characteristics

Description	Min	Typical	Max	Unit
Input frequency	2412	-	2484	MHz
Input impedance	-	50	-	Ω
Input reflection	-	-	-10	dB
Output power of PA for 72.2 Mbps	15.5	16.5	17.5	dBm
Output power of PA for 11b mode	19.5	20.5	21.5	dBm
DSSS, 1 Mbps	-	-98	-	dBm
CCK, 11 Mbps	-	-91	-	dBm
OFDM, 6 Mbps	-	-93	-	dBm
OFDM, 54 Mbps	-	-75	-	dBm
HT20, MCS0	-	-93	-	dBm
HT20, MCS7	-	-73	-	dBm
HT40, MCS0	-	-90	-	dBm
HT40, MCS7	-	-70	-	dBm
MCS32	-	-89	-	dBm
OFDM, 6 Mbps	-	37	-	dB
OFDM, 54 Mbps	-	21	-	dB
HT20, MCS0	-	37	-	dB
HT20, MCS7	-	20	-	dB

5.5 Bluetooth Radio

5.5.1 Receiver - Basic Data Rate

Table 11: Receiver Characteristics-Basic Data Rate

Parameter	Conditions	Min	Typ	Max	Unit
Sensitivity @0.1% BER	-	-	-98	-	dBm
Maximum received signal @0.1% BER	-	0	-	-	dBm
Co-channel C/I	-	-	+7	-	dB
Adjacent channel selectivity C/I	F = F0 + 1 MHz	-	-	-6	dB
	F = F0 - 1 MHz	-	-	-6	dB
	F = F0 + 2 MHz	-	-	-25	dB
	F = F0 - 2 MHz	-	-	-33	dB
	F = F0 + 3 MHz	-	-	-25	dB
	F = F0 - 3 MHz	-	-	-45	dB
Out-of-band blocking performance	30 MHz ~ 2000 MHz	-10	-	-	dBm
	2000 MHz ~ 2400 MHz	-27	-	-	dBm
	2500 MHz ~ 3000 MHz	-27	-	-	dBm
	3000 MHz ~ 12.5 GHz	-10	-	-	dBm
Intermodulation	-	-36	-	-	dBm

5.5.2 Transmitter - Basic Data Rate

Table 12: Transmitter Characteristics - Basic Data Rate

Parameter	Conditions	Min	Typ	Max	Unit
RF transmit power	-	-	+4	+4	dBm
RF power control range	-	-	25	-	dB
20 dB bandwidth	-	-	0.9	-	MHz
Adjacent channel transmit power	F = F0 + 1 MHz	-	-24	-	dBm
	F = F0 - 1 MHz	-	-16.1	-	dBm
	F = F0 + 2 MHz	-	-40.8	-	dBm
	F = F0 - 2 MHz	-	-35.6	-	dBm
	F = F0 + 3 MHz	-	-45.7	-	dBm
	F = F0 - 3 MHz	-	-40.2	-	dBm
	F = F0 + > 3 MHz	-	-45.6	-	dBm
	F = F0 - > 3 MHz	-	-44.6	-	dBm
Δ f1$_{avg}$	-	-	-	155	kHz
Δ f2$_{max}$	-	133.7	-	-	kHz
Δ f2$_{avg}$/Δ f1$_{avg}$	-	-	0.92	-	-
ICFT	-	-	-7	-	kHz
Drift rate	-	-	0.7	-	kHz/50 μs
Drift (1 slot packet)	-	-	6	-	kHz
Drift (5 slot packet)	-	-	6	-	kHz

5.5.3 Receiver - Enhanced Data Rate

Table 13: Receiver Characteristics - Enhanced Data Rate

Parameter	Conditions	Min	Typ	Max	Unit
$\pi/4$ DQPSK					
Sensitivity @0.01% BER	-	-	-98	-	dBm
Maximum received signal @0.1% BER	-	-	0	-	dBm
Co-channel C/I	-	-	11	-	dB
Adjacent channel selectivity C/I	F = F0 + 1 MHz	-	-7	-	dB
	F = F0 - 1 MHz	-	-7	-	dB
	F = F0 + 2 MHz	-	-25	-	dB
	F = F0 - 2 MHz	-	-35	-	dB
	F = F0 + 3 MHz	-	-25	-	dB
	F = F0 - 3 MHz	-	-45	-	dB
8DPSK					
Sensitivity @0.01% BER	-	-	-84	-	dBm
Maximum received signal @0.1% BER	-	0	-	-	dBm
C/I c-channel	-	-	18	-	dB
Adjacent channel selectivity C/I	F = F0 + 1 MHz	-	2	-	dB
	F = F0 - 1 MHz	-	2	-	dB
	F = F0 + 2 MHz	-	-25	-	dB
	F = F0 - 2 MHz	-	-25	-	dB
	F = F0 + 3 MHz	-	-25	-	dB
	F = F0 - 3 MHz	-	-38	-	dB

5.5.4 Transmitter - Enhanced Data Rate

Table 14: Transmitter Characteristics - Enhanced Data Rate

Parameter	Conditions	Min	Typ	Max	Unit
Maximum RF transmit power	-	-	+2	-	dBm
Relative transmit control	-	-	-1.5	-	dB
$\pi/4$ DQPSK max w0	-	-	-0.72	-	kHz
$\pi/4$ DQPSK max wi	-	-	-6	-	kHz
$\pi/4$ DQPSK max |wi + w0|	-	-	-7.42	-	kHz
8DPSK max w0	-	-	0.7	-	kHz
8DPSK max wi	-	-	-9.6	-	kHz
8DPSK max |wi + w0|	-	-	-10	-	kHz
$\pi/4$ DQPSK modulation accuracy	RMS DEVM	-	4.28	-	%
	99% DEVM	-	-	30	%
	Peak DEVM	-	13.3	-	%
8 DPSK modulation accuracy	RMS DEVM	-	5.8	-	%
	99% DEVM	-	-	20	%
	Peak DEVM	-	14	-	%

Parameter	Conditions	Min	Typ	Max	Unit
In-band spurious emissions	F = F0 + 1 MHz	-	-34	-	dBm
	F = F0 - 1 MHz	-	-40.2	-	dBm
	F = F0 + 2 MHz	-	-34	-	dBm
	F = F0 - 2 MHz	-	-36	-	dBm
	F = F0 + 3 MHz	-	-38	-	dBm
	F = F0 - 3 MHz	-	-40.3	-	dBm
	F = F0 +/- > 3 MHz	-	-	-41.5	dBm
EDR differential phase coding	-	-	100	-	%

5.6 Bluetooth LE Radio

5.6.1 Receiver

Table 15: Receiver Characteristics - BLE

Parameter	Conditions	Min	Typ	Max	Unit
Sensitivity @0.1% BER	-	-	-98	-	dBm
Maximum received signal @0.1% BER	-	0	-	-	dBm
Co-channel C/I	-	-	+10	-	dB
Adjacent channel selectivity C/I	F = F0 + 1 MHz	-	-5	-	dB
	F = F0 - 1 MHz	-	-5	-	dB
	F = F0 + 2 MHz	-	-25	-	dB
	F = F0 - 2 MHz	-	-35	-	dB
	F = F0 + 3 MHz	-	-25	-	dB
	F = F0 - 3 MHz	-	-45	-	dB
Out-of-band blocking performance	30 MHz ~ 2000 MHz	-10	-	-	dBm
	2000 MHz ~ 2400 MHz	-27	-	-	dBm
	2500 MHz ~ 3000 MHz	-27	-	-	dBm
	3000 MHz ~ 12.5 GHz	-10	-	-	dBm
Intermodulation	-	-36	-	-	dBm

5.6.2 Transmitter

Table 16: Transmitter Characteristics - BLE

Parameter	Conditions	Min	Typ	Max	Unit
RF transmit power	-	-	+7.5	+10	dBm
RF power control range	-	-	25	-	dB
Adjacent channel transmit power	F = F0 + 1 MHz	-	-14.6	-	dBm
	F = F0 - 1 MHz	-	-12.7	-	dBm
	F = F0 + 2 MHz	-	-44.3	-	dBm
	F = F0 - 2 MHz	-	-38.7	-	dBm
	F = F0 + 3 MHz	-	-49.2	-	dBm
	F = F0 - 3 MHz	-	-44.7	-	dBm
	F = F0 + > 3 MHz	-	-50	-	dBm

Parameter	Conditions	Min	Typ	Max	Unit
	F = F0 - > 3 MHz	-	-50	-	dBm
Δ f1$_{avg}$	-	-	-	265	kHz
Δ f2$_{max}$	-	247	-	-	kHz
Δ f2$_{avg}$/Δ f1$_{avg}$	-	-	-0.92	-	-
ICFT	-	-	-10	-	kHz
Drift rate	-	-	0.7	-	kHz/50 μs
Drift	-	-	2	-	kHz

6. Package Information

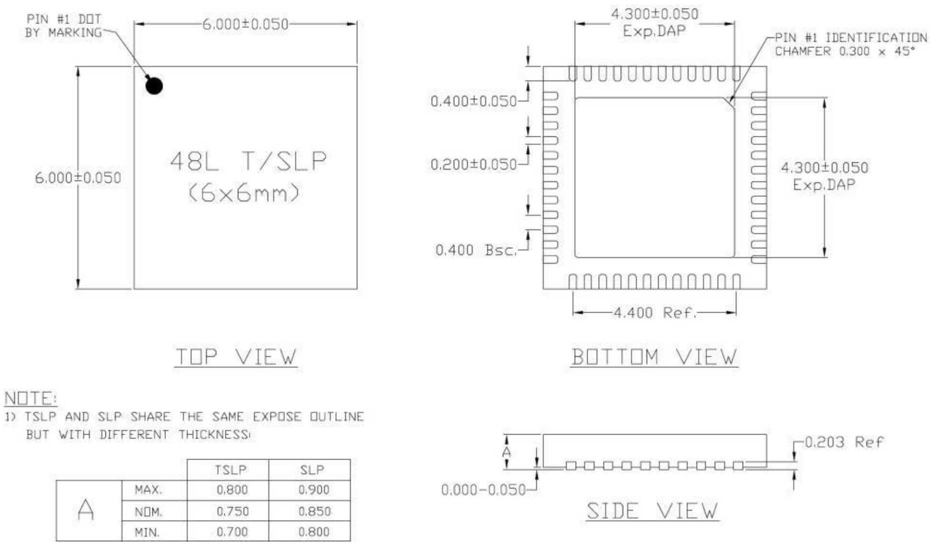


Figure 4: QFN48 (6x6 mm) Package

7. Supported Resources

7.1 Related Documentation

The following link provides related documents of ESP32.

- ESP32 Documentation
  All the available documentation and other resources of ESP32

7.2 Community Resources

The following links connect to ESP32 community resources.

- ESP32 Online Community
  An Engineer-to-Engineer (E2E) Community for ESP32 where you can ask questions, share knowledge, explore ideas and help solve problems with fellow engineers.
- ESP32 Github
  ESP32 development projects are freely distributed under Espressif's MIT license on Github. It is established to help developers get started with ESP32 and foster innovation and the growth of general knowledge about the hardware and software surrounding these devices.

Appendix A - Touch Sensor

A touch sensor system is built on a substrate which carries electrodes and relevant connections with a flat protective surface. When a user touches the surface, the capacitance variation is triggered, and a binary signal is generated to indicate whether the touch is valid.

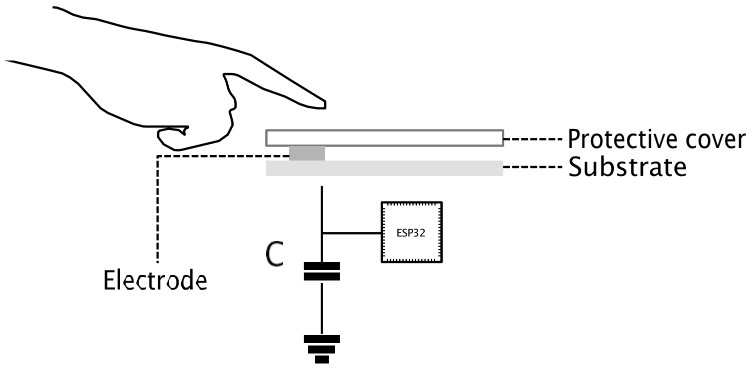


Figure 5: A Typical Touch Sensor Application

In order to prevent capacitive coupling and other electrical interference to the sensitivity of the touch sensor system, the following factors should be taken into account.

A.1. Electrode Pattern

The proper size and shape of an electrode helps improve system sensitivity. Round, oval, or shapes similar to a human fingertip is commonly applied. Large size or irregular shape might lead to incorrect responses from nearby electrodes.

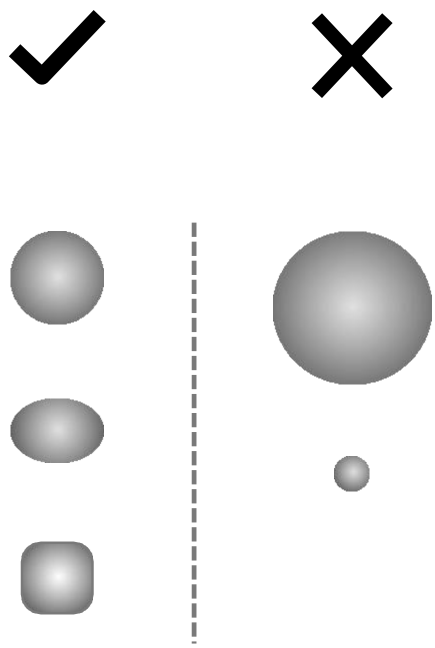


Figure 6: Electrode Pattern Requirements

Note:
The examples illustrated in Figure 6 are not of actual scale. It is suggested that users take a human fingertip as reference.

A.2. PCB Layout

The recommendations for correctly routing sensing tracks of electrodes are as follows:

- Close proximity between electrodes may lead to crosstalk between electrodes and false touch detections. The distance between electrodes should be at least twice the thickness of the panel used.
- The width of a sensor track creates parasitic capacitance, which could vary with manufacturing processes. The thinner the track is, the less capacitive coupling it generates. The track width should be kept as thin as possible and the length should not exceed 10cm to accommodate.
- We should avoid coupling between lines of high frequency signals. The sensing tracks should be routed parallel to each other on the same layer and the distance between the tracks should be at least twice the width of the track.
- When designing a touch sensor device, there should be no components adjacent to or underneath the electrodes.
- Do not ground the touch sensor device. It is preferable that no ground layer be placed under the device, unless there is a need to isolate it. Parasitic capacitance generated between the touch sensor device and the ground degrades sensitivity.

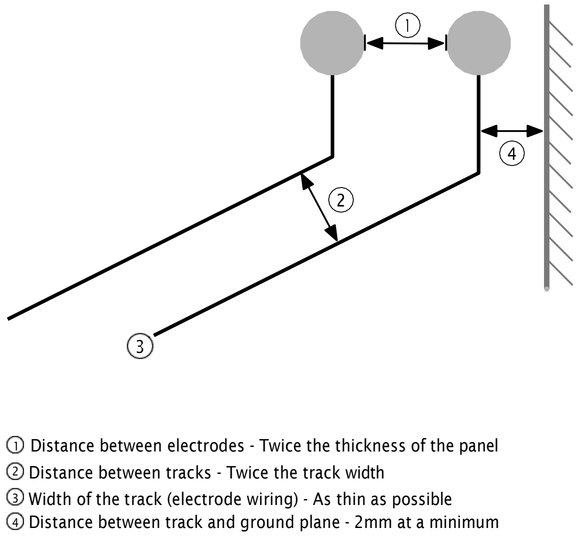


Figure 7: Sensor Track Routing Requirements

Appendix B - Code Examples

B.1. Input

```
>python esptool.py -p dev/tty8 -b 115200 write_Flash -c ESP32 -ff 40m -fm qio -fs 2MB
0x0 ~/Workspace/ESP32_BIN/boot.bin
0x04000 ~/Workspace/ESP32_BIN/drom0.bin
0x40000 ~/Workspace/ESP32_BIN/bin/irom0_Flash.bin
0xFC000 ~/Workspace/ESP32_BIN/blank.bin
0x1FC000 ~/Workspace/ESP32_BIN/esp_init_data_default.bin
```

B.2. Output

```
Connecting...
Erasing Flash...
Wrote 3072 bytes at 0x00000000 in 0.3 seconds (73.8 kbit/s)...
Erasing Flash...
Wrote 395264 bytes at 0x04000000 in 43.2 seconds (73.2 kbit/s)...
Erasing Flash...
Wrote 1024 bytes at 0x40000000 in 0.1 seconds (74.5 kbit/s)...
Erasing Flash...
Wrote 4096 bytes at 0xfc000000 in 0.4 seconds (73.5 kbit/s)...
Erasing Flash...
Wrote 4096 bytes at 0x1fc00000 in 0.5 seconds (73.8 kbit/s)...
Leaving...
```

Anexo 2
Arduino Mega

Product Reference Manual
SKU: A000067

Description

Arduino® Mega 2560 is an exemplary development board dedicated for building extensive applications as compared to other maker boards by Arduino. The board accommodates the ATmega2560 microcontroller, which operates at a frequency of 16 MHz. The board contains 54 digital input/output pins, 16 analog inputs, 4 UARTs (hardware serial ports), a USB connection, a power jack, an ICSP header, and a reset button.

Target Areas

3D Printing, Robotics, Maker

Features

- **ATmega2560 Processor**
  - Up to 16 MIPS Throughput at 16MHz
  - 256k bytes (of which 8k is used for the bootloader)
  - 4k bytes EEPROM
  - 8k bytes Internal SRAM
  - 32 × 8 General Purpose Working Registers
  - Real Time Counter with Separate Oscillator
  - Four 8-bit PWM Channels
  - Four Programmable Serial USART
  - Controller/Peripheral SPI Serial Interface
- **ATmega16U2**
  - Up to 16 MIPS Throughput at 16 MHz
  - 16k bytes ISP Flash Memory
  - 512 bytes EEPROM
  - 512 bytes SRAM
  - USART with SPI master only mode and hardware flow control (RTS/CTS)
  - Master/Slave SPI Serial Interface
- **Sleep Modes**
  - Idle
  - ADC Noise Reduction
  - Power-save
  - Power-down
  - Standby
  - Extended Standby
- **Power**
  - USB Connection
  - External AC/DC Adapter
- **I/O**
  - 54 Digital
  - 16 Analog
  - 15 PWM Output

Contents

1 The Board

Arduino® Mega 2560 is a successor board of Arduino Mega, it is dedicated to applications and projects that require large number of input output pins and the use cases which need high processing power. The Arduino® Mega 2560 comes with a much larger set of IOs when we compare it with traditional Uno board considering the form factor of both the boards.

1.1 Application Examples

- **Robotics**: Featuring the high processing capacitity, the Arduino Mega 2560 can handle the extensive robotic applications. It is compatible with the motor controller shield that enables it to control multiple motors at an instance, thus making it perfect of robotic applications. The large number of I/O pins can accommodate many robotic sensors as well.
- **3D Printing**: Algorithms play a significant role in implementation of 3D printers. Arduino Mega 2560 has the power to process these complex algorithms required for 3D printing. Additionally, the slight changes to the code is easily possible with the Arduino IDE and thus 3D printing programs can be customized according to user requirements.
- **Wi-Fi**: Integrating wireless functionality enhances the utility of the applications. Arduino Mega 2560 is compatible with WiFi shields hence allowing the wireless features for the applications in 3D printing and Robotics.

1.2 Accessories

1.3 Related Products

- Arduino® Uno Rev 3
- Arduino® Nano
- Arduino® DUE without headers

2 Ratings

2.1 Recommended Operating Conditions

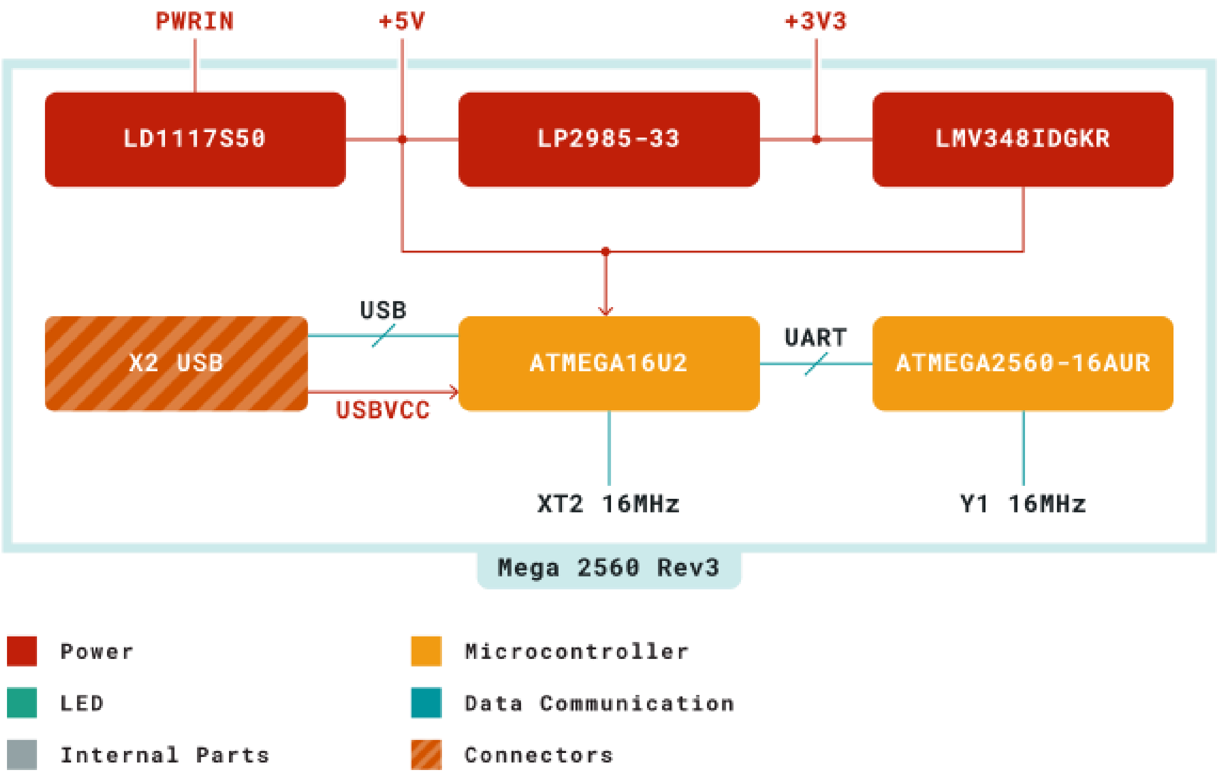
Symbol	Description	Min	Max
TOP	Operating temperature:	-40 °C	85 °C

2.2 Power Consumption

Symbol	Description	Min	Typ	Max	Unit
PWRIN	Input supply from power jack		TBC		mW
USB VCC	Input supply from USB		TBC		mW
VIN	Input from VIN pad		TBC		mW

3 Functional Overview

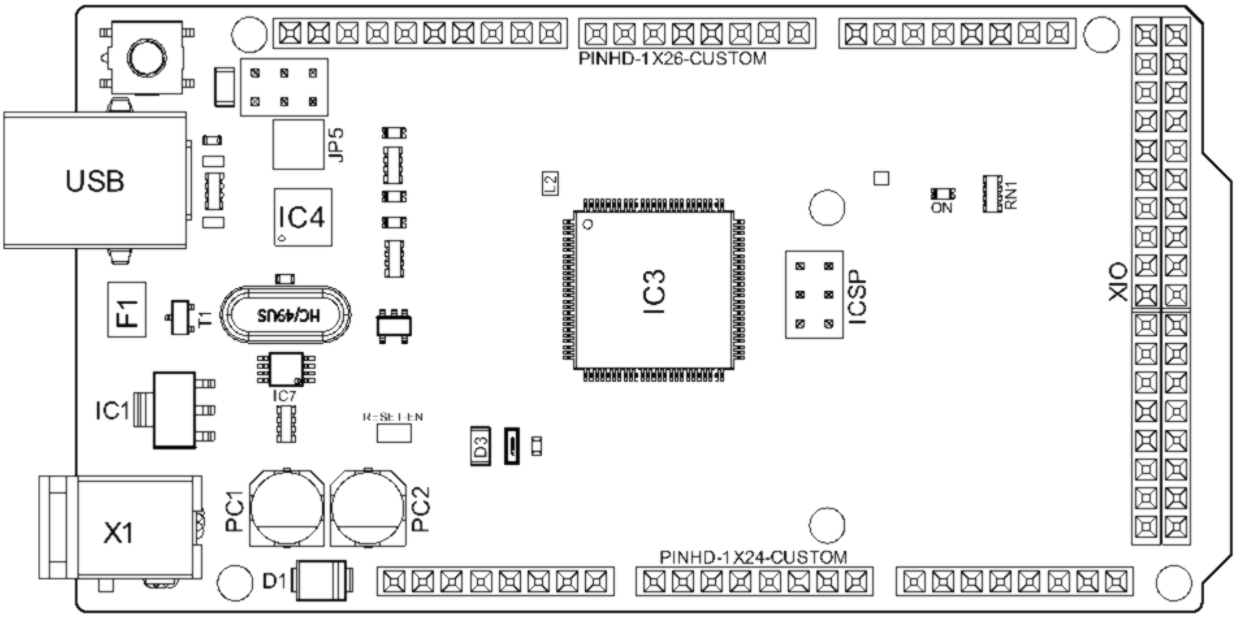
3.1 Block Diagram



Arduino MEGA Block Diagram

3.2 Board Topology

Front View



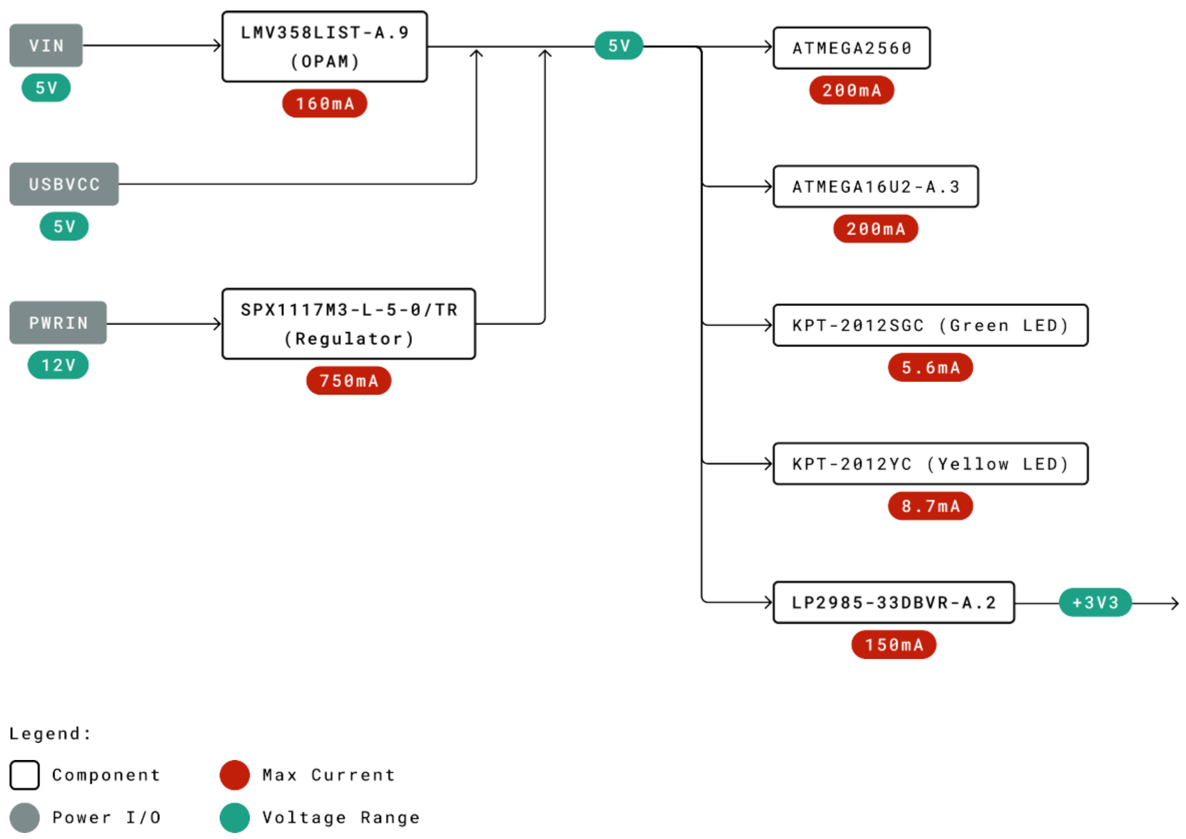
Arduino MEGA Top View

Ref.	Description	Ref.	Description
USB	USB B Connector	F1	Chip Capacitor
IC1	5V Linear Regulator	X1	Power Jack Connector
JP5	Plated Holes	IC4	ATmega16U2 chip
PC1	Electrolytic Alumninum Capacitor	PC2	Electrolytic Alumninum Capacitor
D1	General Purpose Rectifier	D3	General Purpose Diode
L2	Fixed Inductor	IC3	ATmega2560 chip
ICSP	Connector Header	ON	Green LED
RN1	Resistor Array	XIO	Connector

3.3 Processor

Primary processor of Arduino Mega 2560 Rev3 board is ATmega2560 chip which operates at a frequency of 16 MHz. It accommodates a large number of input and output lines which gives the provision of interfacing many external devices. At the same time the operations and processing is not slowed due to its significantly larger RAM than the other processors. The board also features a USB serial processor ATmega16U2 which acts an interface between the USB input signals and the main processor. This increases the flexibility of interfacing and connecting peripherals to the Arduino Mega 2560 Rev 3 board.

3.4 Power Tree



Power Tree

4 Board Operation

4.1 Getting Started - IDE

If you want to program your Arduino® MEGA 2560 while offline you need to install the Arduino® Desktop IDE **[1]** To connect the Arduino® MEGA 2560 to your computer, you'll need a Type-B USB cable. This also provides power to the board, as indicated by the LED.

4.2 Getting Started - Arduino Web Editor

All Arduino® boards, including this one, work out-of-the-box on the Arduino® Web Editor **[2]**, by just installing a simple plugin.

The Arduino® Web Editor is hosted online, therefore it will always be up-to-date with the latest features and support for all boards. Follow **[3]** to start coding on the browser and upload your sketches onto your board.

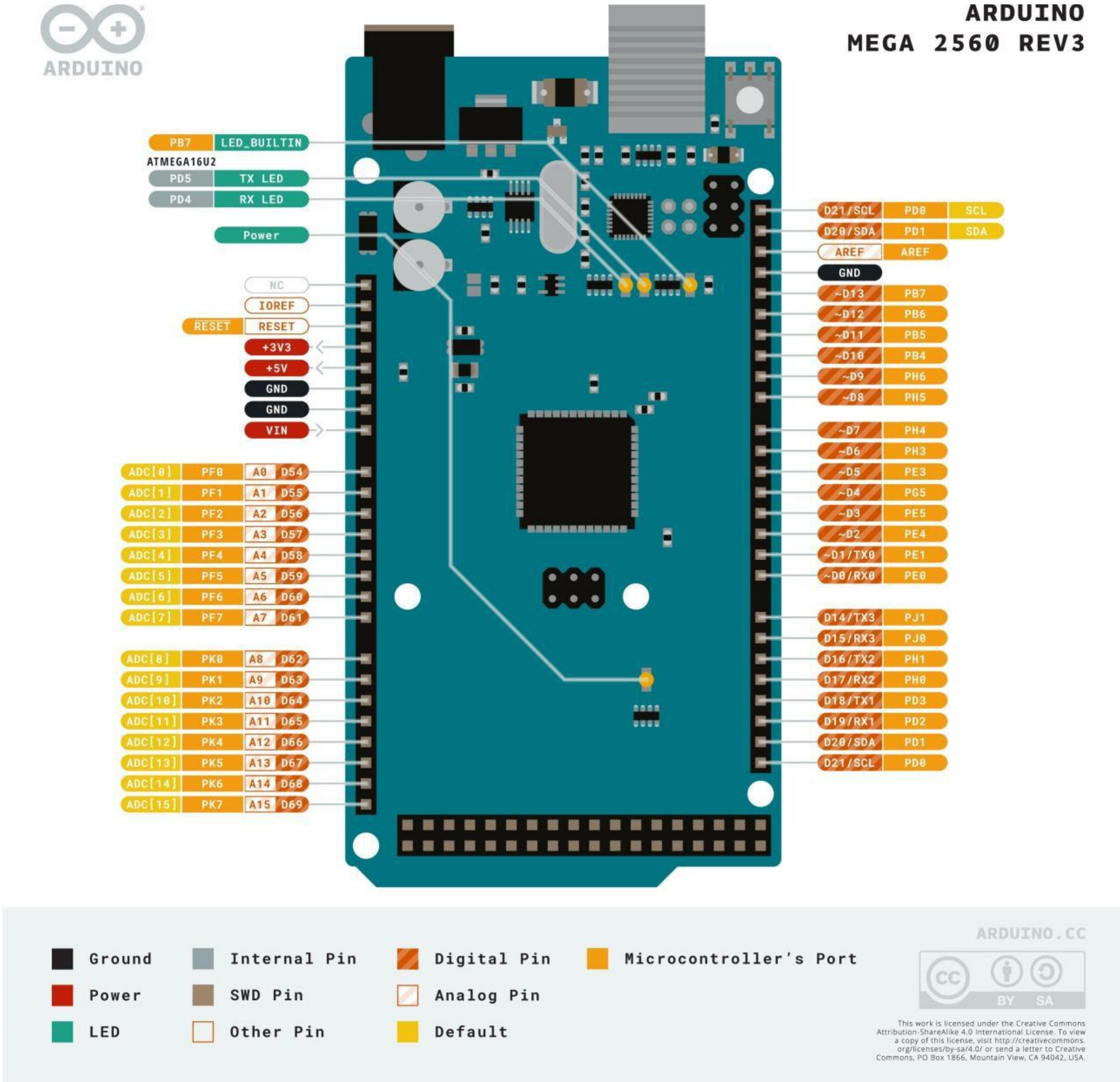
4.3 Sample Sketches

Sample sketches for the Arduino® MEGA 2560 can be found either in the "Examples" menu in the Arduino® IDE

4.4 Online Resources

Now that you have gone through the basics of what you can do with the board you can explore the endless possibilities it provides by checking exciting projects on ProjectHub **[5]**, the Arduino® Library Reference **[6]** and the online store **[7]** where you will be able to complement your board with sensors, actuators and more.

5 Connector Pinouts



Arduino Mega Pinout

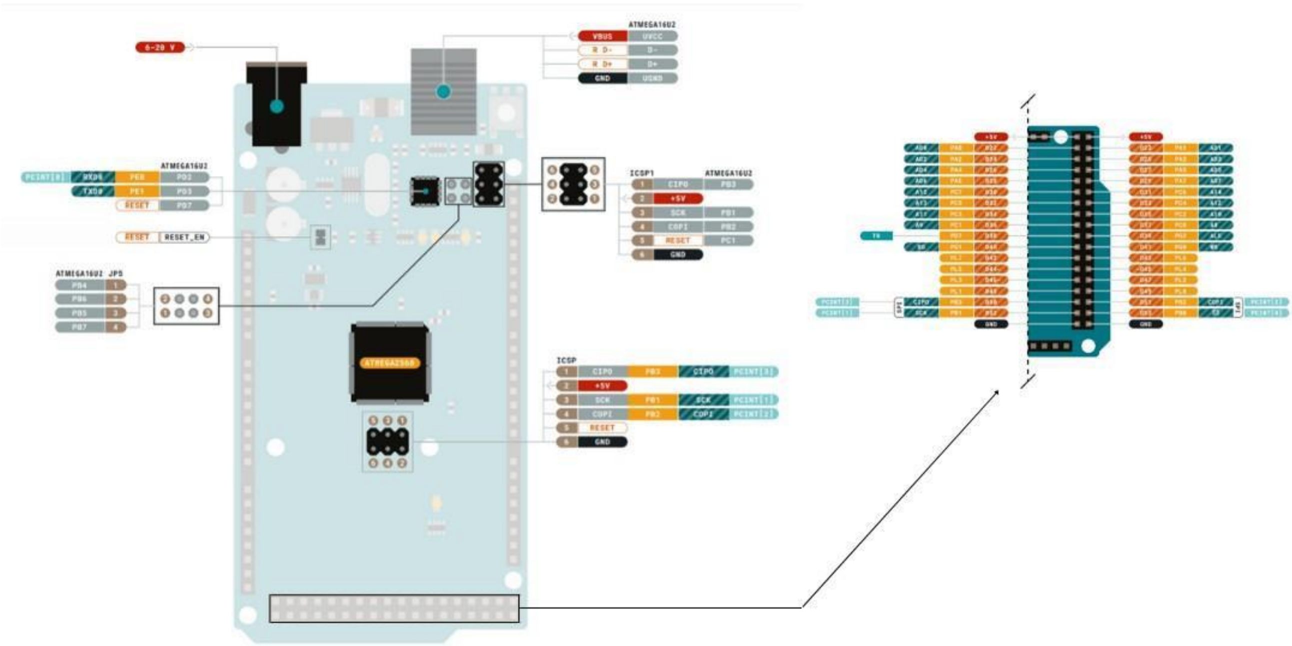
5.1 Analog

Pin	Function	Type	Description
1	NC	NC	Not Connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog	Analog input 0 /GPIO
10	A1	Analog	Analog input 1 /GPIO
11	A2	Analog	Analog input 2 /GPIO
12	A3	Analog	Analog input 3 /GPIO
13	A4	Analog	Analog input 4 /GPIO
14	A5	Analog	Analog input 5 /GPIO
15	A6	Analog	Analog input 6 /GPIO
16	A7	Analog	Analog input 7 /GPIO
17	A8	Analog	Analog input 8 /GPIO
18	A9	Analog	Analog input 9 /GPIO
19	A10	Analog	Analog input 10 /GPIO
20	A11	Analog	Analog input 11 /GPIO
21	A12	Analog	Analog input 12 /GPIO
22	A13	Analog	Analog input 13 /GPIO
23	A14	Analog	Analog input 14 /GPIO
24	A15	Analog	Analog input 15 /GPIO

5.2 Digital

Pin	Function	Type	Description
1	D21/SCL	Digital Input/I2C	Digital input 21/I2C Dataline
2	D20/SDA	Digital Input/I2C	Digital input 20/I2C Dataline
3	AREF	Digital	Analog Reference Voltage
4	GND	Power	Ground
5	D13	Digital/GPIO	Digital input 13/GPIO
6	D12	Digital/GPIO	Digital input 12/GPIO
7	D11	Digital/GPIO	Digital input 11/GPIO
8	D10	Digital/GPIO	Digital input 10/GPIO
9	D9	Digital/GPIO	Digital input 9/GPIO
10	D8	Digital/GPIO	Digital input 8/GPIO
11	D7	Digital/GPIO	Digital input 7/GPIO
12	D6	Digital/GPIO	Digital input 6/GPIO
13	D5	Digital/GPIO	Digital input 5/GPIO
14	D4	Digital/GPIO	Digital input 4/GPIO

Pin	Function	Type	Description
15	D3	Digital/GPIO	Digital input 3/GPIO
16	D2	Digital/GPIO	Digital input 2/GPIO
17	D1/TX0	Digital/GPIO	Digital input 1 /GPIO
18	D0/Tx1	Digital/GPIO	Digital input 0 /GPIO
19	D14	Digital/GPIO	Digital input 14 /GPIO
20	D15	Digital/GPIO	Digital input 15 /GPIO
21	D16	Digital/GPIO	Digital input 16 /GPIO
22	D17	Digital/GPIO	Digital input 17 /GPIO
23	D18	Digital/GPIO	Digital input 18 /GPIO
24	D19	Digital/GPIO	Digital input 19 /GPIO
25	D20	Digital/GPIO	Digital input 20 /GPIO
26	D21	Digital/GPIO	Digital input 21 /GPIO



Arduino Mega Pinout

5.3 ATMEGA16U2 JP5

Pin	Function	Type	Description
1	PB4	Internal	Serial Wire Debug
2	PB6	Internal	Serial Wire Debug
3	PB5	Internal	Serial Wire Debug
4	PB7	Internal	Serial Wire Debug

5.4 ATMEGA16U2 ICSP1

Pin	Function	Type	Description
1	CIPO	Internal	Controller In Peripheral Out
2	+5V	Internal	Power Supply of 5V
3	SCK	Internal	Serial Clock
4	COPI	Internal	Controller Out Peripheral In
5	RESET	Internal	Reset
6	GND	Internal	Ground

5.5 Digital Pins D22 - D53 LHS

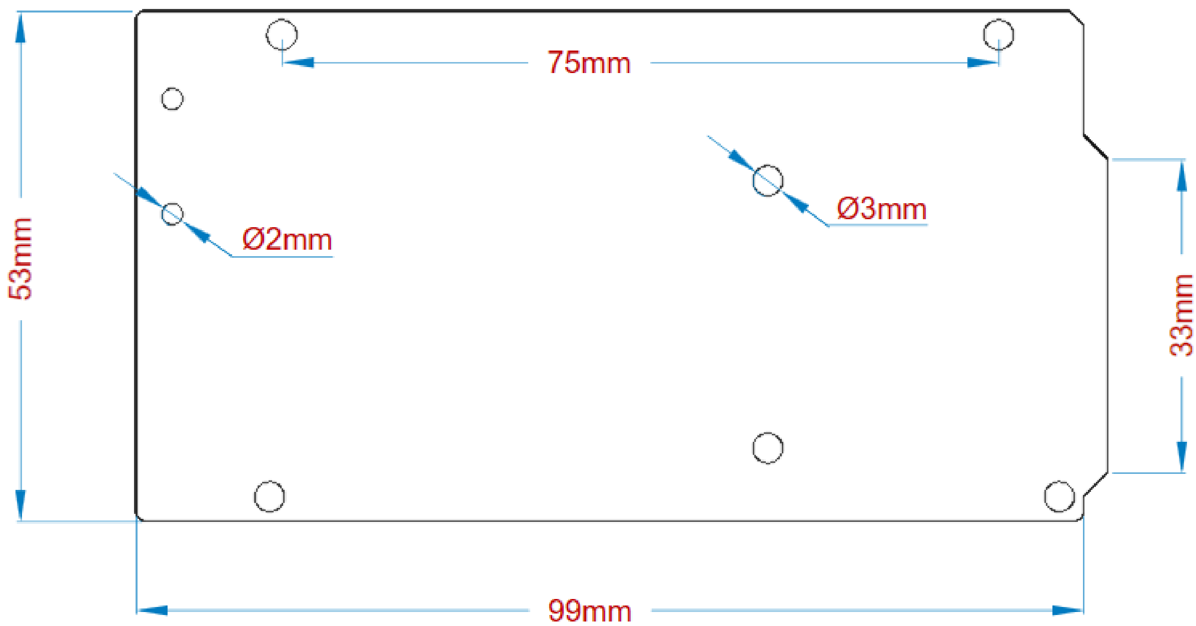
Pin	Function	Type	Description
1	+5V	Power	Power Supply of 5V
2	D22	Digital	Digital input 22/GPIO
3	D24	Digital	Digital input 24/GPIO
4	D26	Digital	Digital input 26/GPIO
5	D28	Digital	Digital input 28/GPIO
6	D30	Digital	Digital input 30/GPIO
7	D32	Digital	Digital input 32/GPIO
8	D34	Digital	Digital input 34/GPIO
9	D36	Digital	Digital input 36/GPIO
10	D38	Digital	Digital input 38/GPIO
11	D40	Digital	Digital input 40/GPIO
12	D42	Digital	Digital input 42/GPIO
13	D44	Digital	Digital input 44/GPIO
14	D46	Digital	Digital input 46/GPIO
15	D48	Digital	Digital input 48/GPIO
16	D50	Digital	Digital input 50/GPIO
17	D52	Digital	Digital input 52/GPIO
18	GND	Power	Ground

5.6 Digital Pins D22 - D53 RHS

Pin	Function	Type	Description
1	+5V	Power	Power Supply of 5V
2	D23	Digital	Digital input 23/GPIO
3	D25	Digital	Digital input 25/GPIO
4	D27	Digital	Digital input 27/GPIO
5	D29	Digital	Digital input 29/GPIO
6	D31	Digital	Digital input 31/GPIO
7	D33	Digital	Digital input 33/GPIO
8	D35	Digital	Digital input 35/GPIO
9	D37	Digital	Digital input 37/GPIO
10	D39	Digital	Digital input 39/GPIO
11	D41	Digital	Digital input 41/GPIO
12	D43	Digital	Digital input 43/GPIO
13	D45	Digital	Digital input 45/GPIO
14	D47	Digital	Digital input 47/GPIO
15	D49	Digital	Digital input 49/GPIO
16	D51	Digital	Digital input 51/GPIO
17	D53	Digital	Digital input 53/GPIO
18	GND	Power	Ground

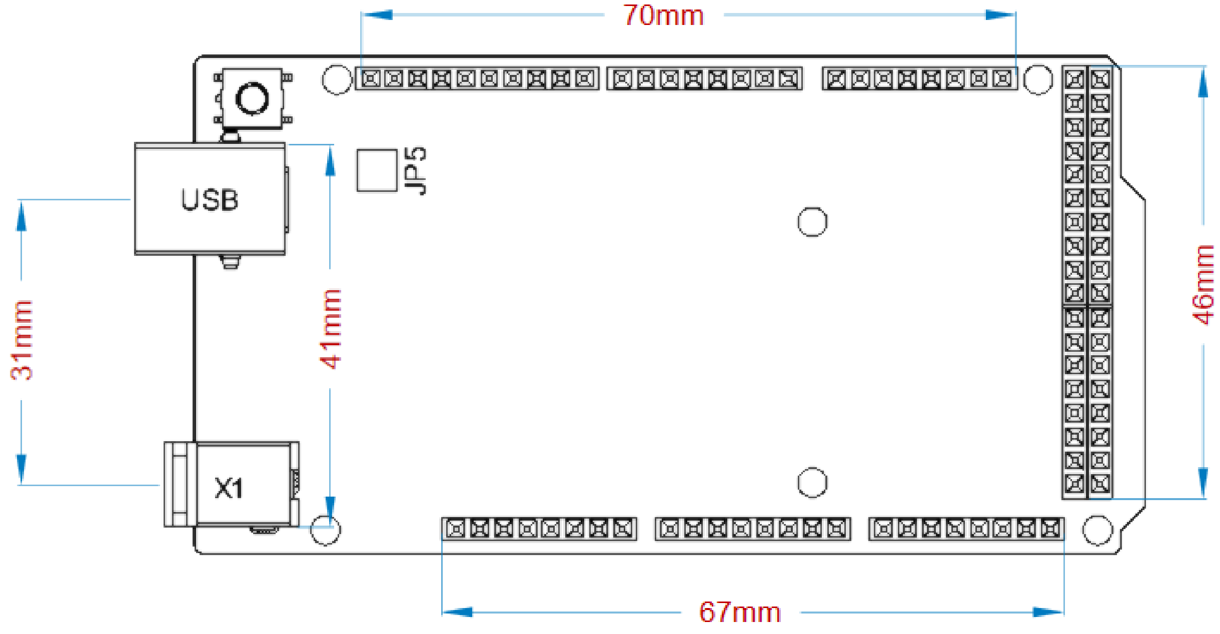
6 Mechanical Information

6.1 Board Outline



Arduino Mega Outline

6.2 Board Mount Holes



Arduino Mega Mount Holes

Certifications

7 Declaration of Conformity CE DoC (EU)

We declare under our sole responsibility that the products above are in conformity with the essential requirements of the following EU Directives and therefore qualify for free movement within markets comprising the European Union (EU) and European Economic Area (EEA).

8 Declaration of Conformity to EU RoHS & REACH 211 01/19/2021

Arduino boards are in compliance with RoHS 2 Directive 2011/65/EU of the European Parliament and RoHS 3 Directive 2015/863/EU of the Council of 4 June 2015 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

Substance	**Maximum Limit (ppm)**
Lead (Pb)	1000
Cadmium (Cd)	100
Mercury (Hg)	1000
Hexavalent Chromium (Cr6+)	1000
Poly Brominated Biphenyls (PBB)	1000
Poly Brominated Diphenyl ethers (PBDE)	1000
Bis(2-Ethylhexyl} phthalate (DEHP)	1000
Benzyl butyl phthalate (BBP)	1000
Dibutyl phthalate (DBP)	1000
Diisobutyl phthalate (DIBP)	1000

Exemptions : No exemptions are claimed.

Arduino Boards are fully compliant with the related requirements of European Union Regulation (EC) 1907 /2006 concerning the Registration, Evaluation, Authorization and Restriction of Chemicals (REACH). We declare none of the SVHCs (https://echa.europa.eu/web/guest/candidate-list-table), the Candidate List of Substances of Very High Concern for authorization currently released by ECHA, is present in all products (and also package) in quantities totaling in a concentration equal or above 0.1%. To the best of our knowledge, we also declare that our products do not contain any of the substances listed on the "Authorization List" (Annex XIV of the REACH regulations) and Substances of Very High Concern (SVHC) in any significant amounts as specified by the Annex XVII of Candidate list published by ECHA (European Chemical Agency) 1907 /2006/EC.

9 Conflict Minerals Declaration

As a global supplier of electronic and electrical components, Arduino is aware of our obligations with regards to laws and regulations regarding Conflict Minerals, specifically the Dodd-Frank Wall Street Reform and Consumer Protection Act, Section 1502. Arduino does not directly source or process conflict minerals such as Tin, Tantalum, Tungsten, or Gold. Conflict minerals are contained in our products in the form of solder, or as a component in metal alloys. As part of our reasonable due diligence Arduino has contacted component suppliers within our supply chain to verify their continued compliance with the regulations. Based on the information received thus far we declare that our products contain Conflict Minerals sourced from conflict-free areas.

10 FCC Caution

Any Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference

(2) this device must accept any interference received, including interference that may cause undesired operation.

FCC RF Radiation Exposure Statement:

1. This Transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
2. This equipment complies with RF radiation exposure limits set forth for an uncontrolled environment.
3. This equipment should be installed and operated with minimum distance 20cm between the radiator & your body.

English: User manuals for licence-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both. This device complies with Industry Canada licence-exempt RSS standard(s). Operation is subject to the following two conditions:

(1) this device may not cause interference

(2) this device must accept any interference, including interference that may cause undesired operation of the device.

French: Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

(1) l' appareil nedoit pas produire de brouillage

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

IC SAR Warning:

English This equipment should be installed and operated with minimum distance 20 cm between the radiator and your body.

French: Lors de l' installation et de l' exploitation de ce dispositif, la distance entre le radiateur et le corps est d 'au moins 20 cm.

Important: The operating temperature of the EUT can't exceed 85℃ and shouldn't be lower than -40℃.

Hereby, Arduino S.r.l. declares that this product is in compliance with essential requirements and other relevant provisions of Directive 201453/EU. This product is allowed to be used in all EU member states.

11 Company Information

Company name	**Arduino S.r.l.**
Company Address	Arduino SRL, Via Andrea Appiani 25, 20900 Monza MB, Italy

12 Reference Documentation

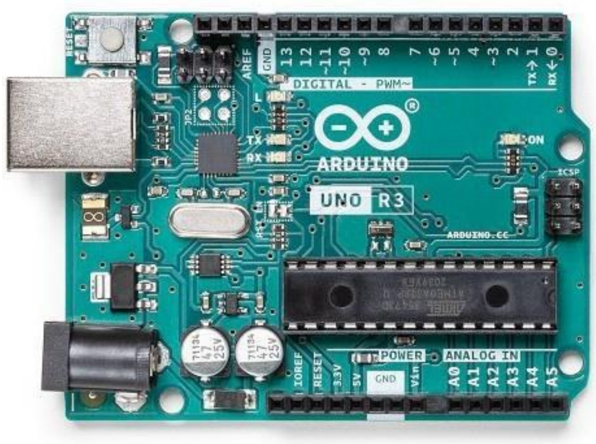
Ref	**Link**
Arduino IDE (Desktop)	https://www.arduino.cc/en/Main/Software
Arduino IDE (Cloud)	https://create.arduino.cc/editor
Cloud IDE Getting Started	https://create.arduino.cc/projecthub/Arduino_Genuino/getting-started-with-arduino-web-editor-4b3e4a
Arduino Pro Website	https://www.arduino.cc/pro
Project Hub	https://create.arduino.cc/projecthub?by=part&part_id=11332&sort=trending
Library Reference	https://www.arduino.cc/reference/en/libraries/
Online Store	https://store.arduino.cc/

13 Revision History

Date	**Revision**	**Changes**

Anexo 3
Arduino Uno

Product Reference Manual
SKU: A000066



Description

The Arduino UNO R3 is the perfect board to get familiar with electronics and coding. This versatile microcontroller is equipped with the well-known ATmega328P and the ATMega 16U2 Processor.
This board will give you a great first experience within the world of Arduino.

Target areas:

Maker, introduction, industries

Features

- **ATMega328P** Processor
  - **Memory**
    - AVR CPU at up to 16 MHz
    - 32KB Flash
    - 2KB SRAM
    - 1KB EEPROM
  - **Security**
    - Power On Reset (POR)
    - Brown Out Detection (BOD)
  - **Peripherals**
    - 2x 8-bit Timer/Counter with a dedicated period register and compare channels
    - 1x 16-bit Timer/Counter with a dedicated period register, input capture and compare channels
    - 1x USART with fractional baud rate generator and start-of-frame detection
    - 1x controller/peripheral Serial Peripheral Interface (SPI)
    - 1x Dual mode controller/peripheral I2C
    - 1x Analog Comparator (AC) with a scalable reference input
    - Watchdog Timer with separate on-chip oscillator
    - Six PWM channels
    - Interrupt and wake-up on pin change
  - **ATMega16U2 Processor**
    - 8-bit AVR® RISC-based microcontroller
  - **Memory**
    - 16 KB ISP Flash
    - 512B EEPROM
    - 512B SRAM
    - debugWIRE interface for on-chip debugging and programming
  - **Power**
    - 2.7-5.5 volts

CONTENTS

1 The Board

1.1 Application Examples

The UNO board is the flagship product of Arduino. Regardless if you are new to the world of electronics or will use the UNO as a tool for education purposes or industry-related tasks.

First entry to electronics: If this is your first project within coding and electronics, get started with our most used and documented board; Arduino UNO. It is equipped with the well-known ATmega328P processor, 14 digital input/output pins, 6 analog inputs, USB connections, ICSP header and reset button. This board includes everything you will need for a great first experience with Arduino.

Industry-standard development board: Using the Arduino UNO board in industries, there are a range of companies using the UNO board as the brain for their PLC's.

Education purposes: Although the UNO board has been with us for about ten years, it is still widely used for various education purposes and scientific projects. The board's high standard and top quality performance makes it a great resource to capture real time from sensors and to trigger complex laboratory equipment to mention a few examples.

1.2 Related Products

- Starter Kit
- Tinkerkit Braccio Robot
- Example

2 Ratings

2.1 Recommended Operating Conditions

Symbol	Description	Min	Max
	Conservative thermal limits for the whole board:	-40 °C (-40°F)	85 °C (185°F)

NOTE: In extreme temperatures, EEPROM, voltage regulator, and the crystal oscillator, might not work as expected due to the extreme temperature conditions

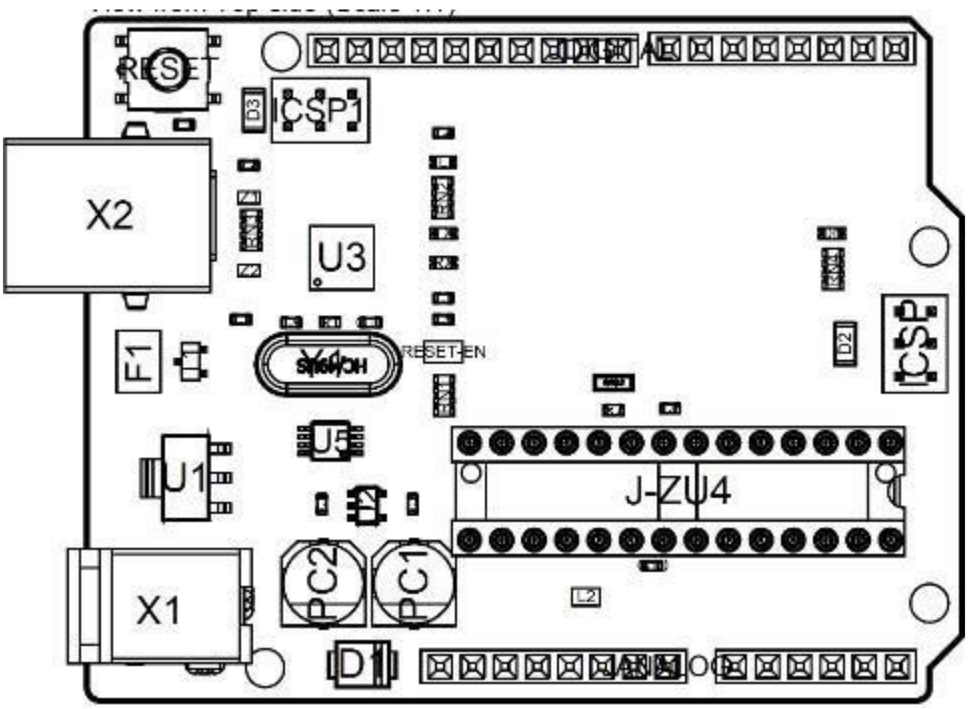
2.2 Power Consumption

Symbol	Description	Min	Typ	Max	Unit
VINMax	Maximum input voltage from VIN pad	6	-	20	V
VUSBMax	Maximum input voltage from USB connector		-	5.5	V
PMax	Maximum Power Consumption	-	-	xx	mA

3 Functional Overview

3.1 Board Topology

Top view



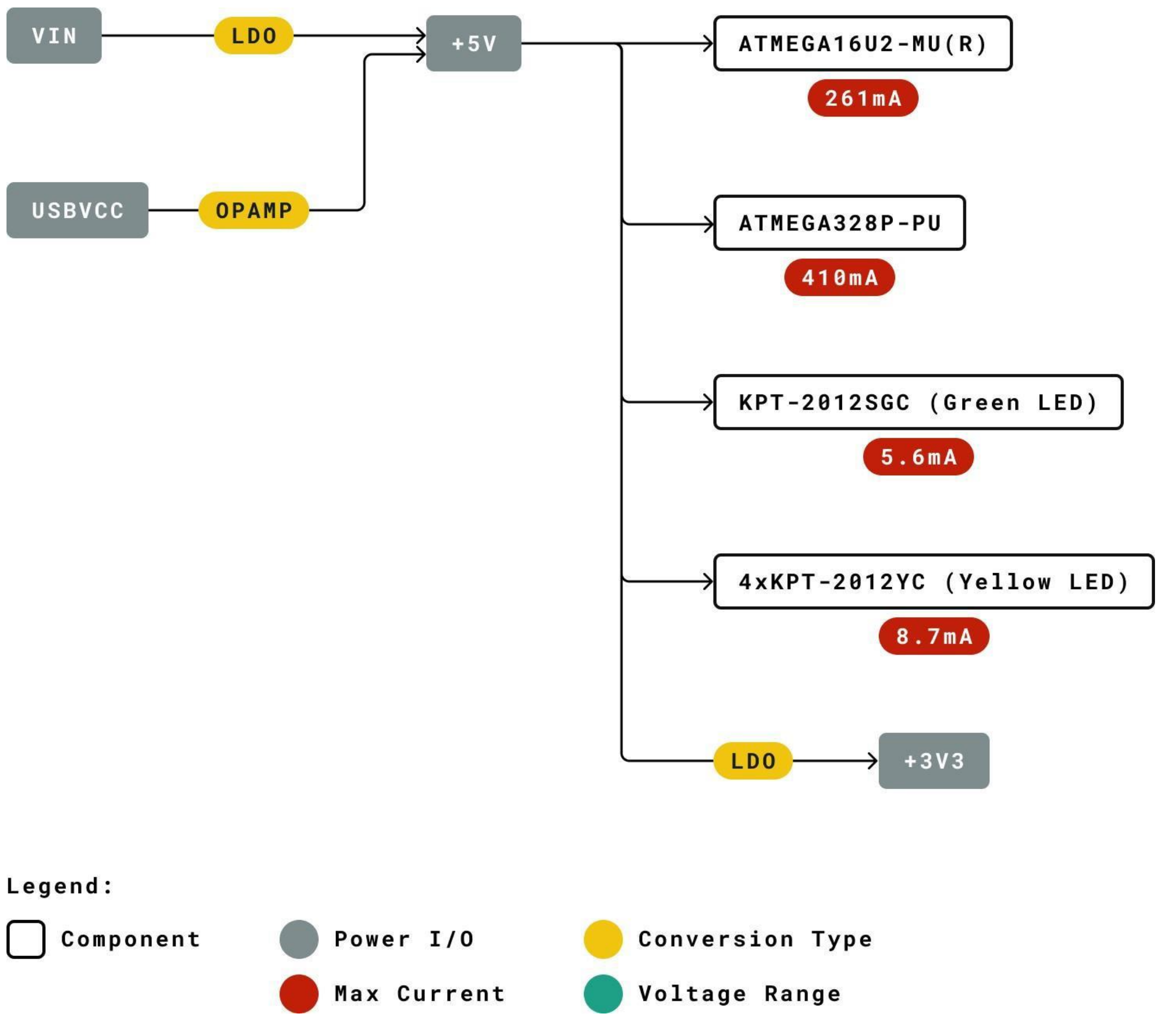
Board topology

Ref.	Description	Ref.	Description
X1	Power jack 2.1x5.5mm	U1	SPX1117M3-L-5 Regulator
X2	USB B Connector	U3	ATMEGA16U2 Module
PC1	EEE-1EA470WP 25V SMD Capacitor	U5	LMV358LIST-A.9 IC
PC2	EEE-1EA470WP 25V SMD Capacitor	F1	Chip Capacitor, High Density
D1	CGRA4007-G Rectifier	ICSP	Pin header connector (through hole 6)
J-ZU4	ATMEGA328P Module	ICSP1	Pin header connector (through hole 6)
Y1	ECS-160-20-4X-DU Oscillator		

3.2 Processor

The Main Processor is a ATmega328P running at up tp 20 MHz. Most of its pins are connected to the external headers, however some are reserved for internal communication with the USB Bridge coprocessor.

3.3 Power Tree



Power tree

4 Board Operation

4.1 Getting Started - IDE

If you want to program your Arduino UNO while offline you need to install the Arduino Desktop IDE [1] To connect the Arduino UNO to your computer, you'll need a Micro-B USB cable. This also provides power to the board, as indicated by the LED.

4.2 Getting Started - Arduino Web Editor

All Arduino boards, including this one, work out-of-the-box on the Arduino Web Editor [2], by just installing a simple plugin.

The Arduino Web Editor is hosted online, therefore it will always be up-to-date with the latest features and support for all boards. Follow **[3]** to start coding on the browser and upload your sketches onto your board.

4.3 Getting Started - Arduino IoT Cloud

All Arduino IoT enabled products are supported on Arduino IoT Cloud which allows you to Log, graph and analyze sensor data, trigger events, and automate your home or business.

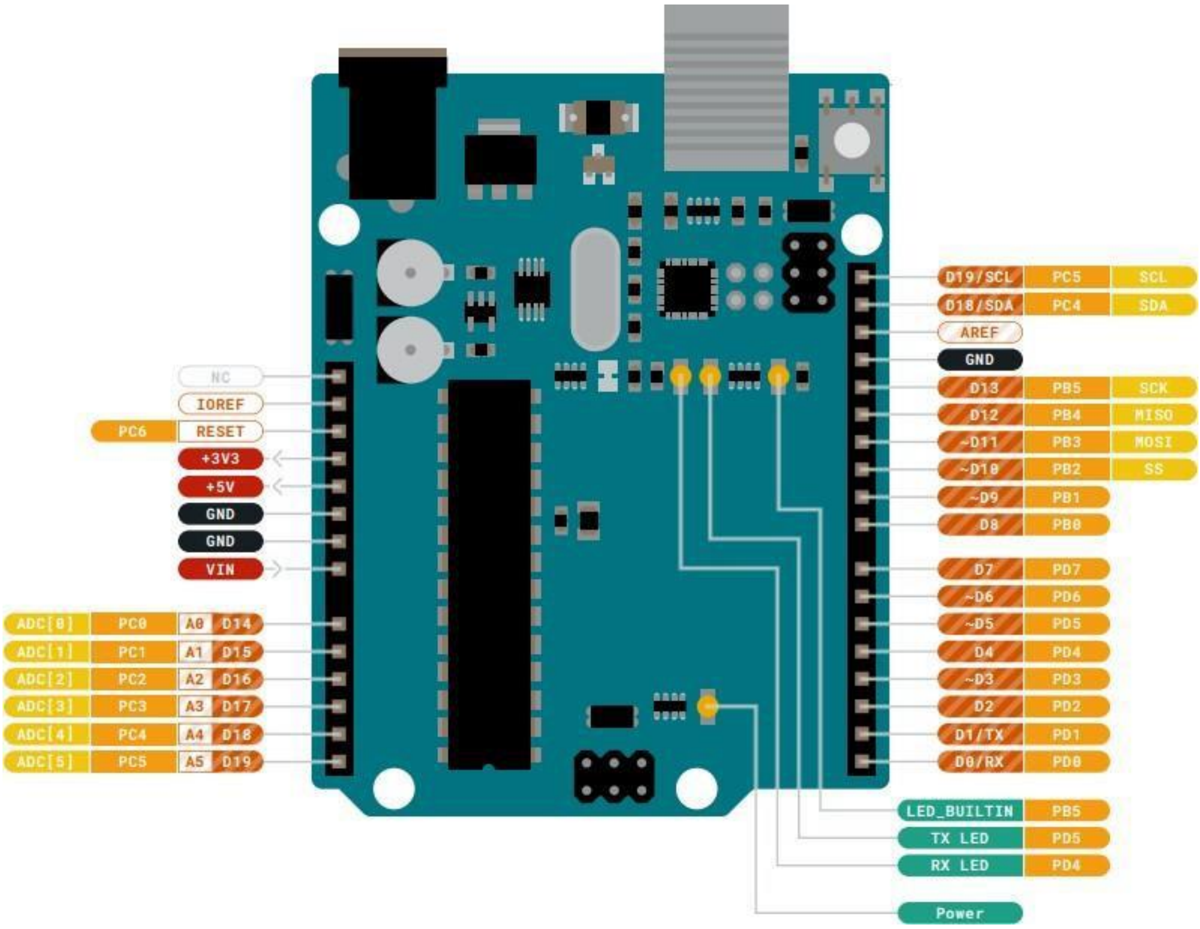
4.4 Sample Sketches

Sample sketches for the Arduino XXX can be found either in the "Examples" menu in the Arduino IDE or in the "Documentation" section of the Arduino Pro website [4]

4.5 Online Resources

Now that you have gone through the basics of what you can do with the board you can explore the endless possibilities it provides by checking exciting projects on ProjectHub **[5]**, the Arduino Library Reference **[6]** and the online store **[7]** where you will be able to complement your board with sensors, actuators and more

5 Connector Pinouts



Pinout

5.1 JANALOG

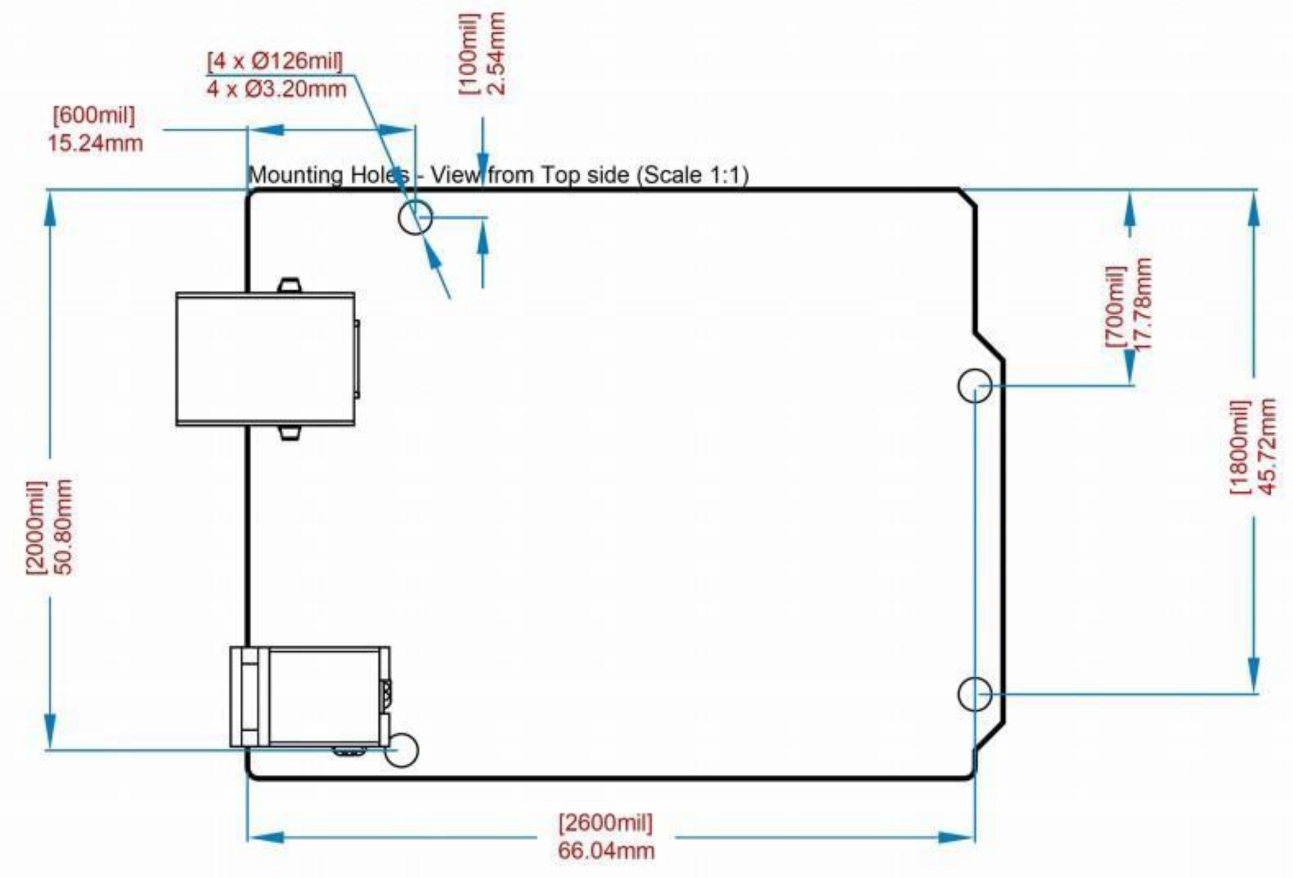
Pin	Function	Type	Description
1	NC	NC	Not connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog/GPIO	Analog input 0 /GPIO
10	A1	Analog/GPIO	Analog input 1 /GPIO
11	A2	Analog/GPIO	Analog input 2 /GPIO
12	A3	Analog/GPIO	Analog input 3 /GPIO
13	A4/SDA	Analog input/I2C	Analog input 4/I2C Data line
14	A5/SCL	Analog input/I2C	Analog input 5/I2C Clock line

5.2 JDIGITAL

Pin	Function	Type	Description
1	D0	Digital/GPIO	Digital pin 0/GPIO
2	D1	Digital/GPIO	Digital pin 1/GPIO
3	D2	Digital/GPIO	Digital pin 2/GPIO
4	D3	Digital/GPIO	Digital pin 3/GPIO
5	D4	Digital/GPIO	Digital pin 4/GPIO
6	D5	Digital/GPIO	Digital pin 5/GPIO
7	D6	Digital/GPIO	Digital pin 6/GPIO
8	D7	Digital/GPIO	Digital pin 7/GPIO
9	D8	Digital/GPIO	Digital pin 8/GPIO
10	D9	Digital/GPIO	Digital pin 9/GPIO
11	SS	Digital	SPI Chip Select
12	MOSI	Digital	SPI1 Main Out Secondary In
13	MISO	Digital	SPI Main In Secondary Out
14	SCK	Digital	SPI serial clock output
15	GND	Power	Ground
16	AREF	Digital	Analog reference voltage
17	A4/SD4	Digital	Analog input 4/I2C Data line (duplicated)
18	A5/SD5	Digital	Analog input 5/I2C Clock line (duplicated)

5.3 Mechanical Information

5.4 Board Outline & Mounting Holes



Board outline

6 Certifications

6.1 Declaration of Conformity CE DoC (EU)

We declare under our sole responsibility that the products above are in conformity with the essential requirements of the following EU Directives and therefore qualify for free movement within markets comprising the European Union (EU) and European Economic Area (EEA).

ROHS 2 Directive 2011/65/EU	
Conforms to:	EN50581:2012
Directive 2014/35/EU. (LVD)	
Conforms to:	EN 60950-1:2006/A11:2009/A1:2010/A12:2011/AC:2011
Directive 2004/40/EC & 2008/46/EC & 2013/35/EU, EMF	
Conforms to:	EN 62311:2008

6.2 Declaration of Conformity to EU RoHS & REACH 211 01/19/2021

Arduino boards are in compliance with RoHS 2 Directive 2011/65/EU of the European Parliament and RoHS 3 Directive 2015/863/EU of the Council of 4 June 2015 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

Substance	Maximum limit (ppm)
Lead (Pb)	1000
Cadmium (Cd)	100
Mercury (Hg)	1000
Hexavalent Chromium (Cr6+)	1000
Poly Brominated Biphenyls (PBB)	1000
Poly Brominated Diphenyl ethers (PBDE)	1000
Bis(2-Ethylhexyl} phthalate (DEHP)	1000
Benzyl butyl phthalate (BBP)	1000
Dibutyl phthalate (DBP)	1000
Diisobutyl phthalate (DIBP)	1000

Exemptions: No exemptions are claimed.

Arduino Boards are fully compliant with the related requirements of European Union Regulation (EC) 1907 /2006 concerning the Registration, Evaluation, Authorization and Restriction of Chemicals (REACH). We declare none of the SVHCs (https://echa.europa.eu/web/guest/candidate-list-table), the Candidate List of Substances of Very High Concern for authorization currently released by ECHA, is present in all products (and also package) in quantities totaling in a concentration equal or above 0.1%. To the best of our knowledge, we also declare that our products do not contain any of the substances listed on the "Authorization List" (Annex XIV of the REACH regulations) and Substances of Very High Concern (SVHC) in any significant amounts as specified by the Annex XVII of Candidate list published by ECHA (European Chemical Agency) 1907 /2006/EC.

6.3 Conflict Minerals Declaration

As a global supplier of electronic and electrical components, Arduino is aware of our obligations with regards to laws and regulations regarding Conflict Minerals, specifically the Dodd-Frank Wall Street Reform and Consumer Protection Act, Section 1502. Arduino does not directly source or process conflict minerals such as Tin, Tantalum, Tungsten, or Gold. Conflict minerals are contained in our products in the form of solder, or as a component in metal alloys. As part of our reasonable due diligence Arduino has contacted component suppliers within our supply chain to verify their continued compliance with the regulations. Based on the information received thus far we declare that our products contain Conflict Minerals sourced from conflict-free areas.

7 FCC Caution

Any Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference

(2) this device must accept any interference received, including interference that may cause undesired operation.

FCC RF Radiation Exposure Statement:

1. This Transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
2. This equipment complies with RF radiation exposure limits set forth for an uncontrolled environment.
3. This equipment should be installed and operated with minimum distance 20cm between the radiator & your body.

English: User manuals for license-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both. This device complies with Industry Canada license-exempt RSS standard(s). Operation is subject to the following two conditions:

(1) this device may not cause interference

(2) this device must accept any interference, including interference that may cause undesired operation of the device.

French: Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

(1) l' appareil nedoit pas produire de brouillage

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

IC SAR Warning:

English This equipment should be installed and operated with minimum distance 20 cm between the radiator and your body.

French: Lors de l' installation et de l' exploitation de ce dispositif, la distance entre le radiateur et le corps est d 'au moins 20 cm.

Important: The operating temperature of the EUT can't exceed 85℃ and shouldn't be lower than -40℃.

Hereby, Arduino S.r.l. declares that this product is in compliance with essential requirements and other relevant provisions of Directive 2014/53/EU. This product is allowed to be used in all EU member states.

8 Company Information

Company name	**Arduino S.r.l**
Company Address	Via Andrea Appiani 25 20900 MONZA Italy

9 Reference Documentation

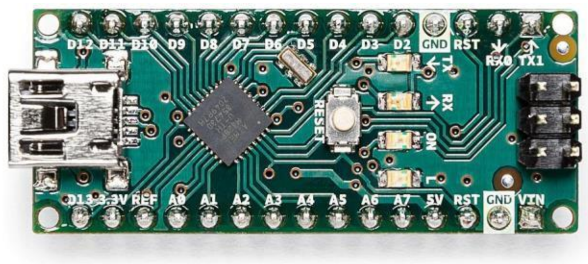
Reference	**Link**
Arduino IDE (Desktop)	https://www.arduino.cc/en/Main/Software
Arduino IDE (Cloud)	https://create.arduino.cc/editor
Cloud IDE Getting Started	https://create.arduino.cc/projecthub/Arduino_Genuino/getting-started-with-arduino-web-editor-4b3e4a
Arduino Pro Website	https://www.arduino.cc/pro
Project Hub	https://create.arduino.cc/projecthub?by=part&part_id=11332&sort=trending
Library Reference	https://www.arduino.cc/reference/en/
Online Store	https://store.arduino.cc/

10 Revision History

Date	**Revision**	**Changes**

Anexo 4
Arduino Nano

Product Reference Manual
SKU: A000005



Description

Arduino® Nano is an intelligent development board designed for building faster prototypes with the smallest dimension. Arduino Nano being the oldest member of the Nano family, provides enough interfaces for your breadboard-friendly applications. At the heart of the board is **ATmega328 microcontroller** clocked at a frequency of 16 MHz featuring more or less the same functionalities as the Arduino Duemilanove. The board offers 22 digital input/output pins, 8 analog pins, and a mini-USB port.

Target Areas

Maker, Security, Environmental, Robotics and Control Systems

Features

- **ATmega328** Microcontroller
  - High-performance low-power 8-bit processor
  - Achieve up to 16 MIPS for 16 MHz clock frequency
  - 32 kB of which 2 KB used by bootloader
  - 2 kB internal SRAM
  - 1 kB EEPROM
  - 32 x 8 General Purpose Working Registers
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
- **Power**
  - Mini-B USB connection
  - 6-20V unregulated external power supply (pin 30)
  - 5V regulated external power supply (pin 27)
- **Sleep Modes**
  - Idle
  - ADC Noise Reduction
  - Power-save
  - Power-down
  - Standby
  - Extended Standby
- **I/O**
  - 22 Digital
  - 8 Analog
  - 6 PWM Output

Contents

1 The Board

1.1 Application Examples

Arduino Nano is the first embedded microcontroller in the Nano series with minimum functionalities, designed for mini projects from the maker community. With a large number of input/output pins gives the advantage of utilizing several serial communications like UART, SPI and I2C. The hardware is compatible with Arduino IDE, Arduino CLI and web editor.

Security: The high-performance and low-power capabilities gives the chance to develop security based applications like access control systems using fingerprint sensors. The flexibility to interface sensors and external devices using serial communication has improved the scope of utility.

Environmental: The low-power feature of the microcontroller and the power supply options for the board has enhanced the ability to implement remote IoT projects related to environmental issues.

Robotics: Robotics has always been the favorite area of exploration for the Maker community and with this tiny embedded hardware you can now create complex and advanced robotic applications.

1.2 Accessories

1.3 Related Products

- Arduino Nano 33 BLE
- Arduino 33 IoT
- Arduino Micro

2 Ratings

2.1 Recommended Operating Conditions

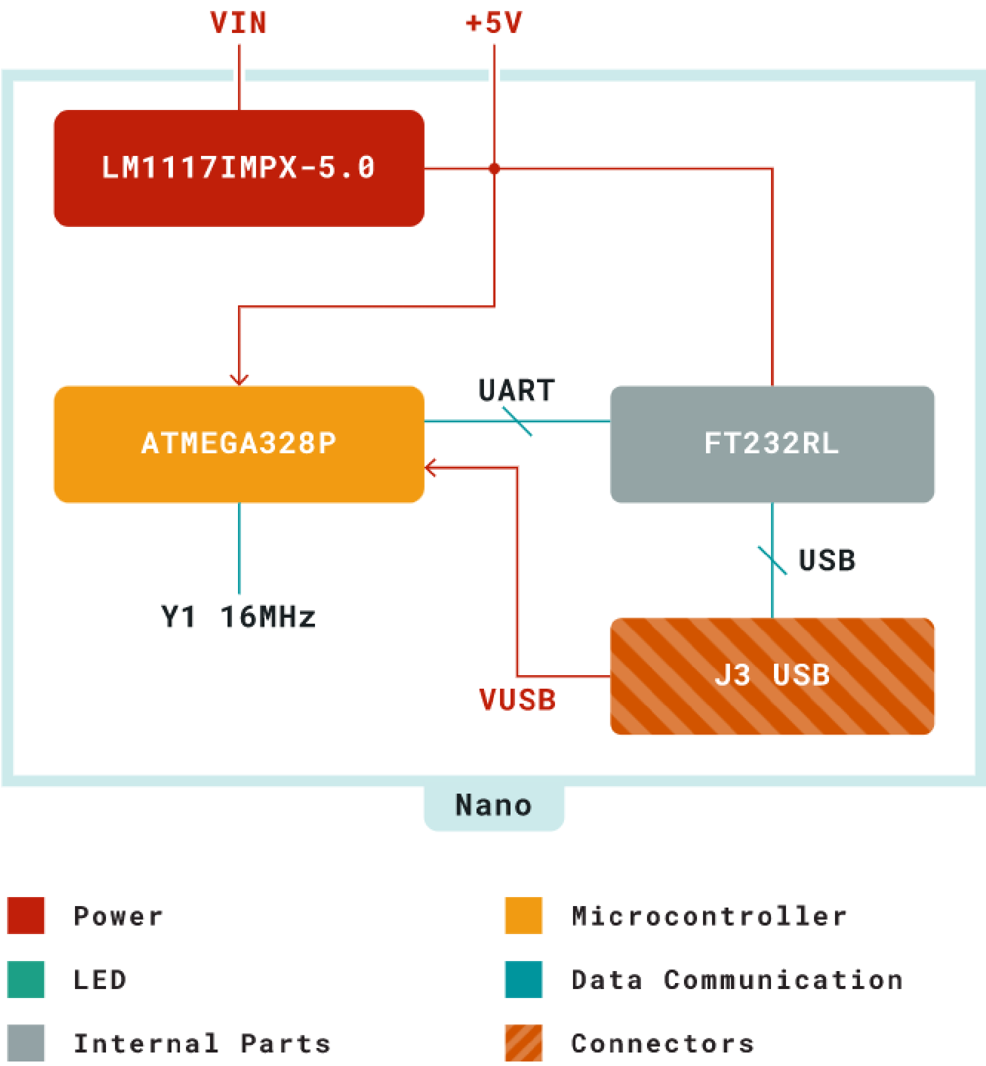
Symbol	Description	Min	Max
	Conservative thermal limits for the whole board:	-40 °C	85 °C

2.2 Power Consumption

Symbol	Description	Min	Typ	Max	Unit
USB VCC	Input supply from USB		TBC		mW
VIN	Input from VIN pad		TBC		mW

3 Functional Overview

3.1 Block Diagram

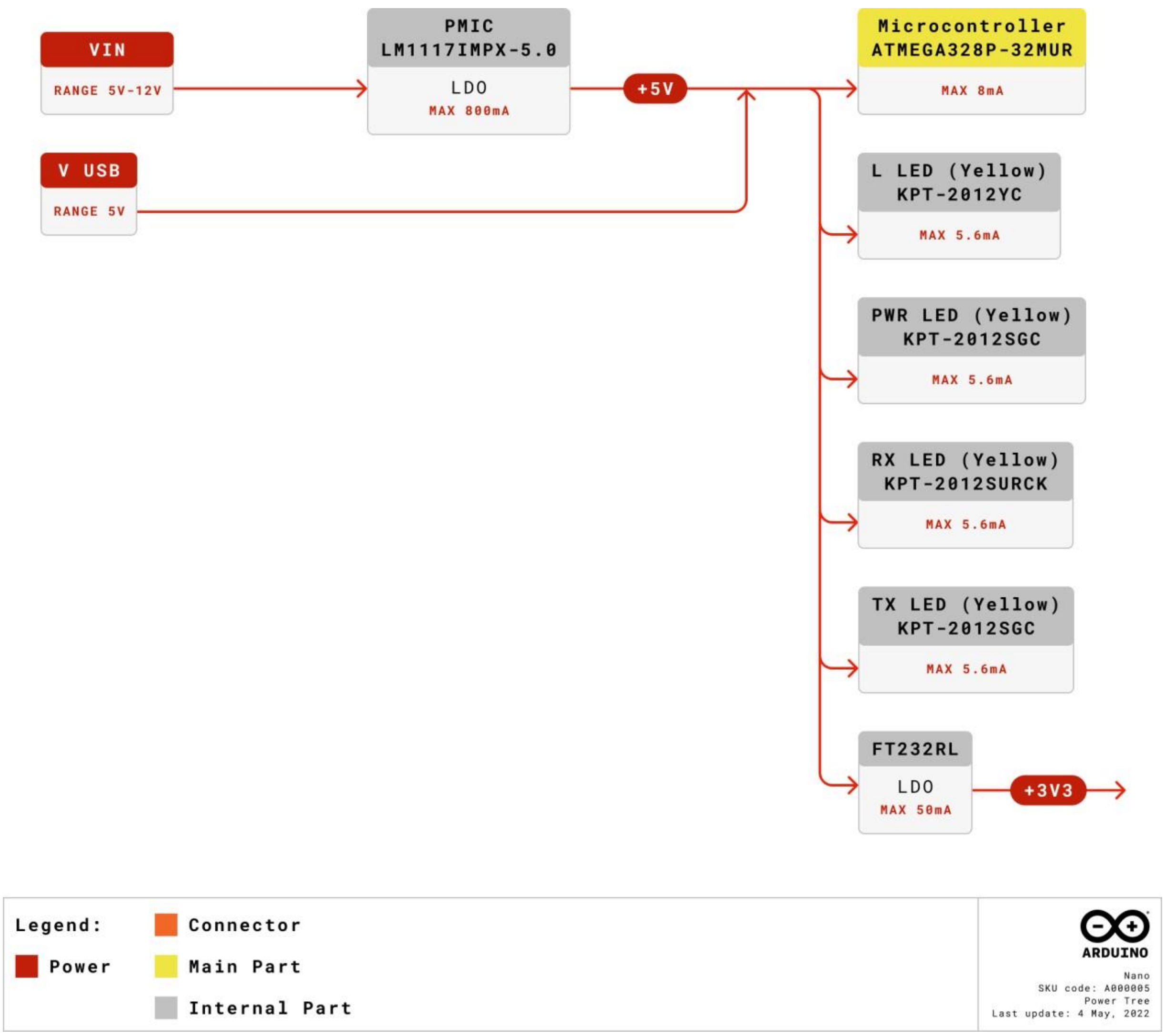


Block Diagram of Arduino Nano

3.2 Processor

The primary processor in the Arduino Nano v3.3 board is the high-performance and low-power 8-bit ATmega328 microcontroller that runs at a clock frequency of 16 MHz. The ability to interface external devices through serial communication supported by the chip with UART TTL (5V), I2C (TWI) and SPI. Arduino Nano can be programmed with Arduino software reducing the entry barriers for new users. Smallest dimension embedded hardware makes it a perfect choice for breadboard-friendly projects from the maker community.

3.3 Power Tree



Power Tree of Arduino Nano

The Arduino Nano can be powered by either the USB port or alternatively via VIN. The input supply of VIN is regulated by an LDO so the supply is limited to 5V for the optimal functioning of the board. There is also another regulator which limits the voltage to 3.3V for powrering the components with low voltage requirements.

4 Board Operation

4.1 Getting Started - IDE

If you want to program your Arduino® Nano while offline you need to install the Arduino® Desktop IDE **[1]** To connect the Arduino Uno to your computer, you'll need a Micro-B USB cable. This also provides power to the board, as indicated by the LED.

4.2 Getting Started - Arduino Web Editor

All Arduino® boards, including this one, work out-of-the-box on the Arduino Web Editor **[2]**, by just installing a simple plugin. The Arduino Web Editor is hosted online, therefore it will always be up-to-date with the latest features and support for all boards. Follow **[3]** to start coding on the browser and upload your sketches onto your board.

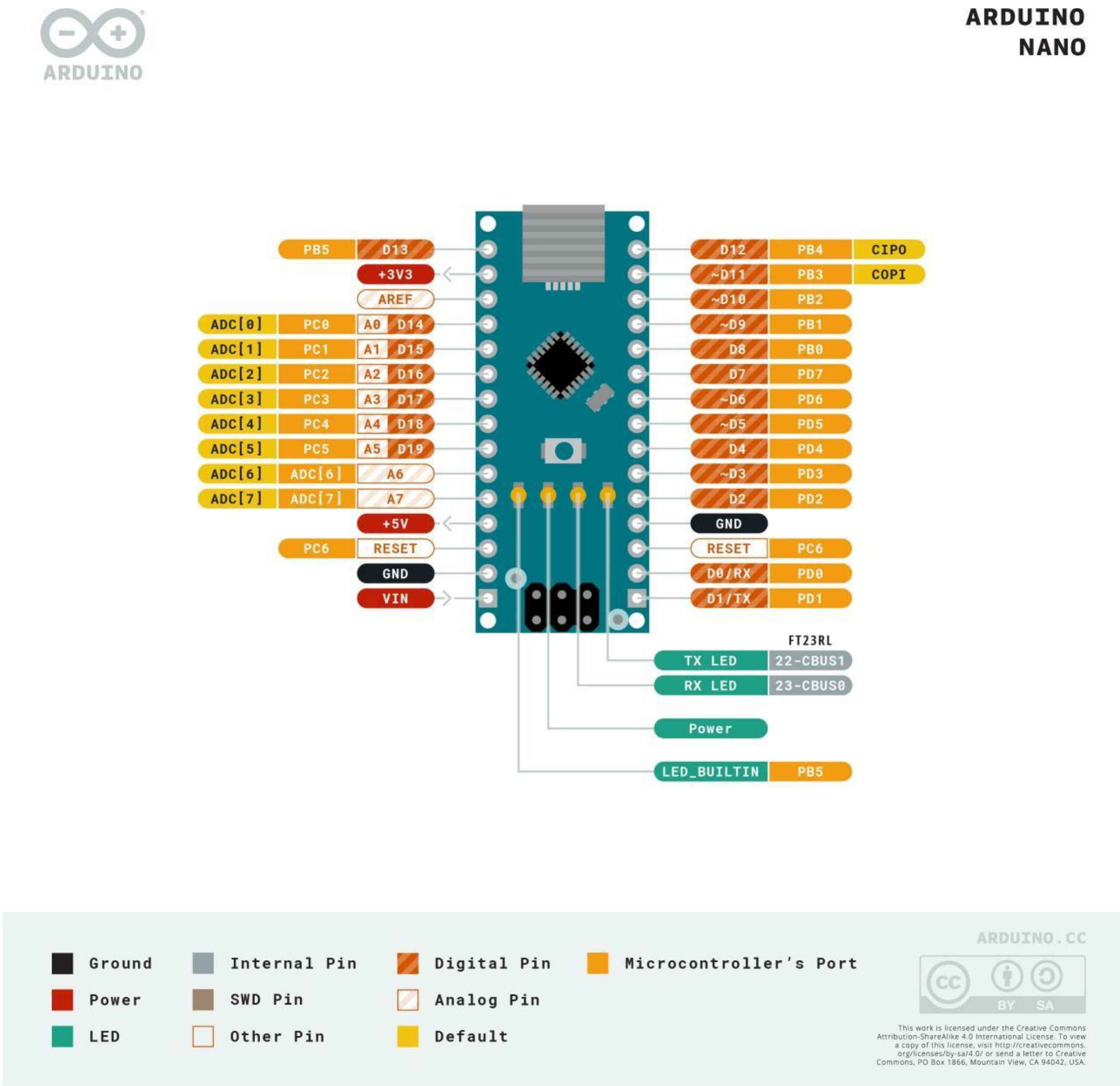
4.3 Sample Sketches

Sample sketches for the Arduino® can be found either in the "Examples" menu in the Arduino® IDE or in the "Documentation" section of the Arduino website **[4]**

4.4 Online Resources

Now that you have gone through the basics of what you can do with the board you can explore the endless possibilities it provides by checking exciting projects on ProjectHub **[5]**, the Arduino® Library Reference **[6]** and the online store **[7]** where you will be able to complement your board with sensors, actuators and more.

5 Connector Pinouts



Power Tree of Arduino Nano

5.1 Analog

Pin	Function	Type	Description
1	+3V3	Power	5V USB Power
2	A0	Analog	Analog input 0 /GPIO
3	A1	Analog	Analog input 1 /GPIO
4	A2	Analog	Analog input 2 /GPIO
5	A3	Analog	Analog input 3 /GPIO
6	A4	Analog	Analog input 4 /GPIO
7	A5	Analog	Analog input 5 /GPIO
8	A6	Analog	Analog input 6 /GPIO
9	A7	Analog	Analog input 7 /GPIO
10	+5V	Power	+5V Power Rail
11	Reset	Reset	Reset
12	GND	Power	Ground
12	VIN	Power	Voltage Input

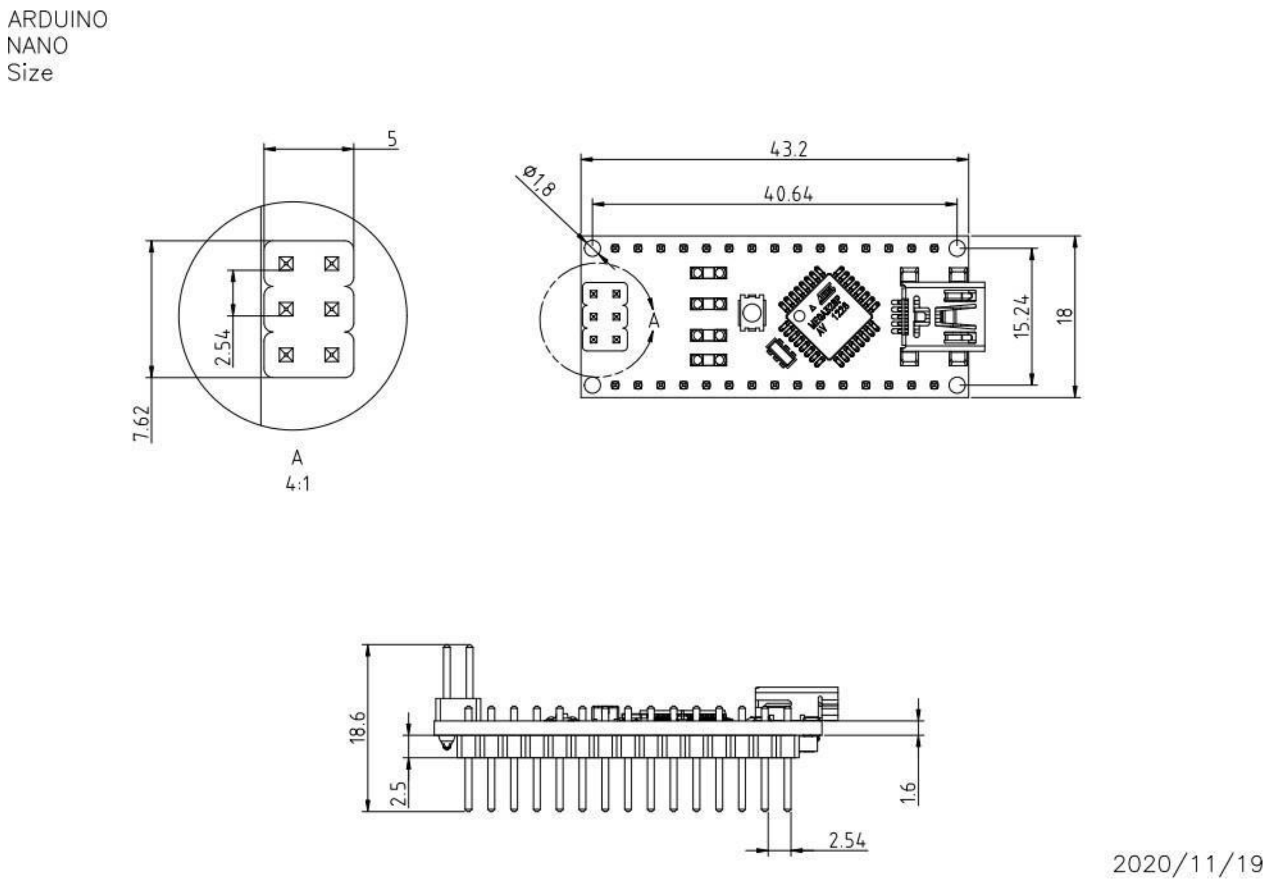
5.2 Digital

Pin	Function	Type	Description
1	D1/TX1	Digital	Digital Input 1 /GPIO
2	D0/RX0	Digital	Digital Input 0 /GPIO
3	D2	Digital	Digital Input 2 /GPIO
4	D3	Digital	Digital Input 3 /GPIO
5	D4	Digital	Digital Input 4 /GPIO
6	D5	Digital	Digital Input 5 /GPIO
7	D6	Digital	Digital Input 6 /GPIO
8	D7	Digital	Digital Input 7 /GPIO
9	D8	Digital	Digital Input 8 /GPIO
10	D9	Digital	Digital Input 9 /GPIO
11	D10	Digital	Digital Input 10 /GPIO
12	D11	Digital	Digital Input 11 /GPIO
13	D12	Digital	Digital Input 12 /GPIO
14	D13	Digital	Digital Input 13 /GPIO
15	Reset	Reset	Reset
16	GND	Power	Ground

5.3 ATmega328

Pin	Function	Type	Description
1	PB0	Internal	Serial Wire Debug
2	PB1	Internal	Serial Wire Debug
3	PB2	Internal	Serial Wire Debug
4	PB3	Internal	Serial Wire Debug
5	PB4	Internal	Serial Wire Debug
6	PB5	Internal	Serial Wire Debug

6 Mechanical Information



Mechanical dimensions of Arduino Nano

7 Certifications

7.1 Declaration of Conformity CE DoC (EU)

We declare under our sole responsibility that the products above are in conformity with the essential requirements of the following EU Directives and therefore qualify for free movement within markets comprising the European Union (EU) and European Economic Area (EEA).

7.2 Declaration of Conformity to EU RoHS & REACH 211 01/19/2021

Arduino boards are in compliance with RoHS 2 Directive 2011/65/EU of the European Parliament and RoHS 3 Directive 2015/863/EU of the Council of 4 June 2015 on the restriction of the use of certain hazardous substances in electrical and electronic equipment.

Substance	Maximum Limit (ppm)
Lead (Pb)	1000
Cadmium (Cd)	100
Mercury (Hg)	1000
Hexavalent Chromium (Cr6+)	1000
Poly Brominated Biphenyls (PBB)	1000
Poly Brominated Diphenyl ethers (PBDE)	1000
Bis(2-Ethylhexyl} phthalate (DEHP)	1000
Benzyl butyl phthalate (BBP)	1000
Dibutyl phthalate (DBP)	1000
Diisobutyl phthalate (DIBP)	1000

Exemptions : No exemptions are claimed.

Arduino Boards are fully compliant with the related requirements of European Union Regulation (EC) 1907 /2006 concerning the Registration, Evaluation, Authorization and Restriction of Chemicals (REACH). We declare none of the SVHCs (https://echa.europa.eu/web/guest/candidate-list-table), the Candidate List of Substances of Very High Concern for authorization currently released by ECHA, is present in all products (and also package) in quantities totaling in a concentration equal or above 0.1%. To the best of our knowledge, we also declare that our products do not contain any of the substances listed on the "Authorization List" (Annex XIV of the REACH regulations) and Substances of Very High Concern (SVHC) in any significant amounts as specified by the Annex XVII of Candidate list published by ECHA (European Chemical Agency) 1907 /2006/EC.

7.3 Conflict Minerals Declaration

As a global supplier of electronic and electrical components, Arduino is aware of our obligations with regards to laws and regulations regarding Conflict Minerals, specifically the Dodd-Frank Wall Street Reform and Consumer Protection Act, Section 1502. Arduino does not directly source or process conflict minerals such as Tin, Tantalum, Tungsten, or Gold. Conflict minerals are contained in our products in the form of solder, or as a component in metal alloys. As part of our reasonable due diligence Arduino has contacted component suppliers within our supply chain to verify their continued compliance with the regulations. Based on the information received thus far we declare that our products contain Conflict Minerals sourced from conflict-free areas.

7.4 FCC Caution

Any Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference

(2) this device must accept any interference received, including interference that may cause undesired operation.

FCC RF Radiation Exposure Statement:

1. This Transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
2. This equipment complies with RF radiation exposure limits set forth for an uncontrolled environment.
3. This equipment should be installed and operated with minimum distance 20cm between the radiator & your body.

English: User manuals for license-exempt radio apparatus shall contain the following or equivalent notice in a conspicuous location in the user manual or alternatively on the device or both. This device complies with Industry Canada license-exempt RSS standard(s). Operation is subject to the following two conditions:

(1) this device may not cause interference

(2) this device must accept any interference, including interference that may cause undesired operation of the device.

French: Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :

(1) l' appareil nedoit pas produire de brouillage

(2) l'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

IC SAR Warning:

English This equipment should be installed and operated with minimum distance 20 cm between the radiator and your body.

French: Lors de l' installation et de l' exploitation de ce dispositif, la distance entre le radiateur et le corps est d 'au moins 20 cm.

Important: The operating temperature of the EUT can't exceed 80°C and shouldn't be lower than -20°C.

Hereby, Arduino S.r.l. declares that this product is in compliance with essential requirements and other relevant provisions of Directive 2014/53/EU. This product is allowed to be used in all EU member states.

8 Company Information

Company name	**Arduino S.r.l.**
Company Address	Via Andrea Appiani 25, 20900 MONZA MB, Italy

9 Reference Documentation

Ref	**Link**
Arduino IDE (Desktop)	https://www.arduino.cc/en/software
Arduino IDE (Cloud)	https://create.arduino.cc/editor
Cloud IDE Getting Started	https://create.arduino.cc/projecthub/Arduino_Genuino/getting-started-with-arduino-web-editor-4b3e4a
Arduino Documentation	https://docs.arduino.cc/hardware/nano
Project Hub	https://create.arduino.cc/projecthub?by=part&part_id=11332&sort=trending
Library Reference	https://www.arduino.cc/reference/en/libraries/
Online Store	https://store.arduino.cc/

10 Revision History

Date	**Revision**	**Changes**
03/08/2022	2	Reference documentation links updates
12/04/2022	1	First Release

Anexo 5
Arduino Micro

Arduino Micro

A 000 053

Arduino Micro Front

Arduino Micro Rear

Overview

The Arduino Micro is a microcontroller board based on the ATmega32u4 (datasheet), developed in conjunction with Adafruit. It has 20 digital input/output pins (of which 7 can be used as PWM outputs and 12 as analog inputs), a 16 MHz crystal oscillator, a micro USB connection, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a micro USB cable to get started. It has a form factor that enables it to be easily placed on a breadboard.
The Micro is similar to the Arduino Leonardo in that the ATmega32u4 has built-in USB communication, eliminating the need for a secondary processor. This allows the Micro to appear to a connected computer as a mouse and keyboard, in addition to a virtual (CDC) serial / COM port. It also has other implications for the behavior of the board; these are detailed on the getting started page.

Summary

Microcontroller	ATmega32u4
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	20
PWM Channels	7
Analog Input Channels	12
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega32u4) of which 4 KB used by bootloader
SRAM	2.5 KB (ATmega32u4)
EEPROM	1 KB (ATmega32u4)
Clock Speed	16 MHz

Schematic & Reference Design

EAGLE files: arduino-micro-reference-design.zip
Schematic: arduino-micro-schematic-rev3b.pdf

Power

The Arduino Micro can be powered via the micro USB connection or with an external power supply. The power source is selected automatically.
External (non-USB) power can come either from a DC power supply or battery. Leads from a battery or DC power supply can be connected to the Gnd and Vin pins.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.
The power pins are as follows:

- **VI.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- ≑ Ground pins.

Memory

The ATmega32u4 has 32 KB (with 4 KB used for the bootloader). It also has 2.5 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

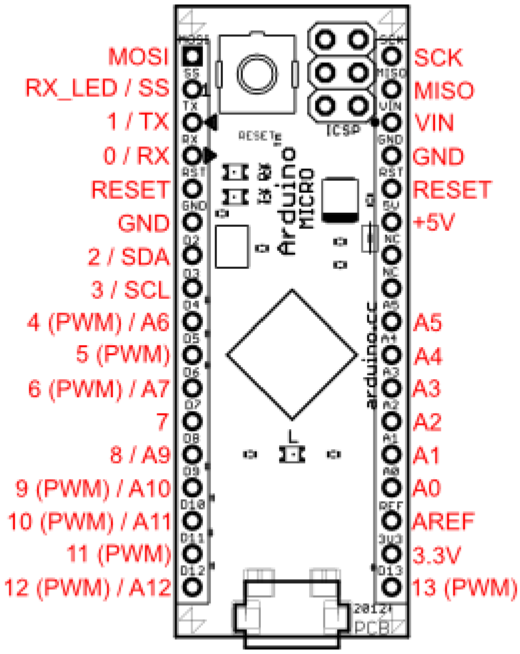
Each of the 20 digital i/o pins on the Micro can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data using the ATmega32U4 hardware serial capability. Note that on the Micro, the **Serial** class refers to USB (CDC) communication; for TTL serial on pins 0 and 1, use the **Serial1** class.
- **TWI: 2 (SDA) and 3 (SCL).** Support TWI communication using the Wire library.
- **External Interrupts: 0(RX), 1(TX), 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, 11, and 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: on the ICSP header.** These pins support SPI communication using the SPI library. Note that the SPI pins are not connected to any of the digital I/O pins as they are on the Arduino Uno, they are only available on the ICSP connector and on the nearby pins labelled MISO, MOSI and SCK.
- **RX_LED/SS** This is an additional pin with respect to the Leonardo. It is connected to the RX_LED that indicates the activity of transmission during USB communication, but is can also used as slave select pin (SS) in SPI communication.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- **Analog Inputs: A0-A5, A6 - A11 (on digital pins 4, 6, 8, 9, 10, and 12).** The Micro has a total of 12 analog inputs, pins from A0 to A5 are labelled directly on the pins and the other ones that you can access in code using the constants from A6 trough A11 are shared respectively on digital pins 4, 6, 8, 9, 10, and 12. All of which can also be used as digital I/O. Each analog input provide 10 bits of resolution (i.e. 1024 different values). By default the analog inputs measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.

There are a couple of other pins on the board:

AREF. Reference voltage for the analog inputs. Used with analogReference().
Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Pinout



Pin Mapping of the Arduino Micro displays the complete functioning for all the pins, to use them as in the Leonardo.
See also the mapping between Arduino pins and ATmega32u4 ports.

Communication

The Micro has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega32U4 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). The 32U4 also allows for serial (CDC) communication over USB and appears as a virtual com port to software on the computer. The chip also acts as a full speed USB 2.0 device, using standard USB COM drivers. On Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB connection to the computer (but not for serial communication on pins 0 and 1).
A SoftwareSerial library allows for serial communication on any of the Micro's digital pins.
The ATmega32U4 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.
The Micro appears as a generic keyboard and mouse, and can be programmed to control these input devices using the Keyboard and Mouse classes.

Programming

The Micro can be programmed with the Arduino software (download). Select "Arduino Micro from the **Tools > Board** menu. For details, see the reference and tutorials.
The ATmega32U4 on the Arduino Micro comes pre-burned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the AVR109 protocol.
You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

Automatic (Software) Reset and Bootloader Initiation

Rather than requiring a physical press of the reset button before an upload, the Micro is designed in a way that allows it to be reset by software running on a connected computer. The reset is triggered when the Micro's virtual (CDC) serial / COM port is opened at 1200 baud and then closed. When this happens, the processor will reset, breaking the USB connection to the computer (meaning that the virtual serial / COM port will disappear). After the processor resets, the bootloader starts, remaining active for about 8 seconds. The bootloader can also be initiated by pressing the reset button on the Micro. Note that when the board first powers up, it will jump straight to the user sketch, if present, rather than initiating the bootloader. Because of the way the Micro handles reset it's best to let the Arduino software try to initiate the reset before uploading, especially if you are in the habit of pressing the reset button before uploading on other boards. If the software can't reset the board you can always start the bootloader by pressing the reset button on the board.

USB Overcurrent Protection

The Micro has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

The maximum length and width of the Micro PCB are 4.8cm and 1.77cm respectively, with the USB connector extending beyond the former dimension. The layout allows for easy placement on a solderless breadboard..

Anexo 6
Motor XD-37GB555

XD-37GB555 High Torque DC Gear Motor

XD-37GB555 is a high quality DC gear motor available in a wide range of RPM configurations, ideal for linear motion control, DIY project and robotics application.

SKU: FAM1084

Specifications:

- Motor type: XD-37GB555.
- Operating voltage: 12V.
- Free-run speed: 600RPM@12V.
- Free-run current: 1.0A@12V.
- Stall current: 3A@12V.
- Rated Torque: 1Kg.cm.
- Gear ratio: 1:10.
- Gear Type: All Metal.
- Shaft Diameter: Ø6mm D-Shape.
- Gear Box size L: 19 mm.
- Weight: 300g.

Mechanical Dimension:

Unit: mm

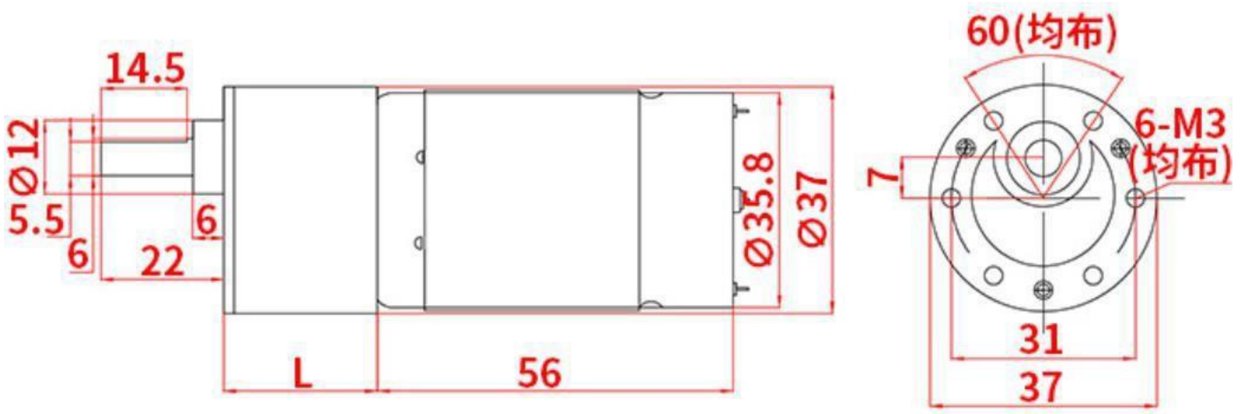


Table-1:

Model No	SKU	RPM (No Load)	Gear Ratio	Gear Box Length (L)	Rated Torque
XD-37GB555-600	FAM1084	600@12V	1:10	19.0	1Kg.cm

Application Example:

Application Note: Useful Motor/Torque Equations

Force (Newtons)
F = m x a
m = mass (kg)
a = acceleration (m/s2)

Motor Torque (Newton-meters)
T = F x d
F = force (Newtons)
d = moment arm (meters)

Power (Watts)
P = I x V
I = current (amps)
V = voltage (volts)

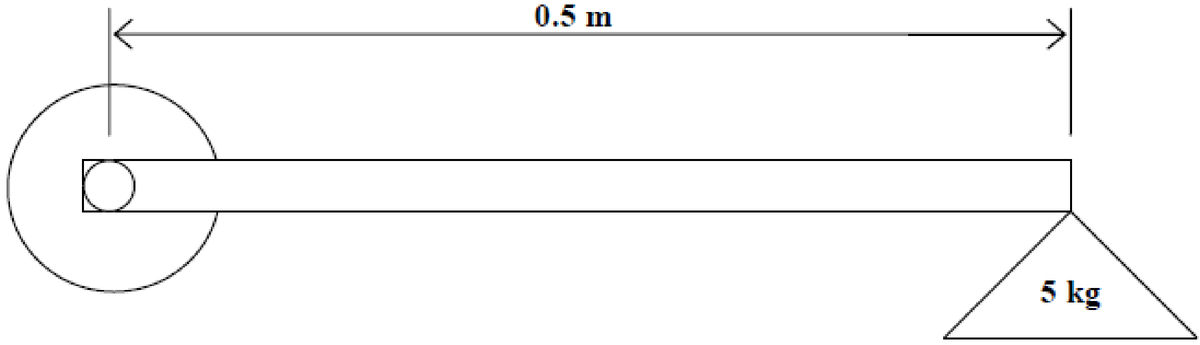
P = T x ω
T = torque (Newton-meters)
ω = angular velocity (radian/second)

Unit Conversions
Length (1 in = 0.0254 m)
Velocity (1 RPM = 0.105 rad/sec)
Torque (1 in-lb = 0.112985 N-m)
Power (1 HP = 745.7 W)

Example 1

Determine if the following motor can be used to lift a 5-kg load using a 0.5-m lever arm.

Merkle-Korff Gearmotor specifications
Stall Torque = 40 in-lb
Stall Current = 3.5 amps



Solution

Convert Stall Torque from in-lb to N-m

1 in-lb = 0.112985 N-m

40 in-lb = 40 x 0.112985 N-m = 4.5194 N-m

Calculate the Force required to lift the 5-kg load

F = m x a = 5 kg x 9.81 m/s2 = 49.05 N

Calculate the Torque required to lift the Force with the lever arm

T = F x d = 49.05 N x 0.5 m = 24.525 N-m

We cannot perform the lift with this set-up, because the stall torque is smaller than the torque required for the lift. We must either shorten the length of the lever arm, or we must choose another motor with a higher stall torque to perform this operation.

Example 2

Using the same motor as in Example 1 with a 12-V power supply:
a) Calculate the power used by the motor to rotate a 5-kg load at 50 RPM using a 3-inch lever arm.
b) Calculate the current draw from the battery to perform this operation.

Solution

Convert inches to meters:
1 in = 0.0254 m
3 in = 0.0762 m

Calculate the Force required to lift the 5-kg load:
F = m x a = 5 kg x 9.81 m/s2 = 49.05 N

Calculate the Torque required for this operation:
T = F x d = 49.05 N x 0.0762 m = 3.738 N-m

> Note- This toque is lower than the motor's stall torque, so this operation is possible using the specified motor, mass, and lever arm

Convert RPM to radians/second:
1 RPM x 2π rad/rev x 1 min/60 sec = 0.105 rad/sec
ω = 50 rev/min x 0.105 rad/sec/RPM = 5.25 rad/sec

Calculate the Power required for this operation:

> P = T x ω = 3.738 N-m x 5.25 rad/sec = 19.622 W

Calculate the Current draw from the battery (use the supply voltage in this calculation):

> I = P/V = 19.622 W/12 V = 1.635 Amps

Note- This current is smaller than the maximum allowable current draw of the motor.

Example 3
Determine the motor torque necessary to power the robot drive wheels.

Solution
The following approach is merely one way to solve this problem. Several exist.
Assume the robot will be powered by two powered drive wheels and supported by two freely rotating caster wheels. Robot weight is denoted by W and for this simple example we'll assume the weight is distributed evenly over all 4 wheels, as shown in Figure 1 below.

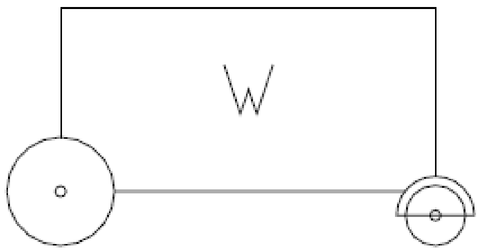


Figure 1

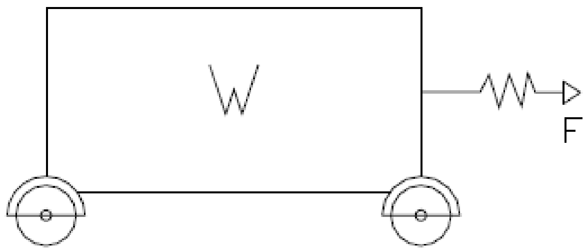


Figure 2

Thinking logically about the problem, we could model the robot as having 4 of the identical caster wheels (Figure 2) and the force required to propel the robot is simply the force needed to start the robot moving (this could be measured empirically with a force scale). The problem is we haven't yet built the robot so testing it in this manner is not an option. We need to calculate the force (and hence motor torque) required to move the robot **before** we build anything.
Looking closer at the caster wheel we can see the actual friction that must be overcome to put the robot in motion. Fw is the friction force between the wheel and the floor and Fa is the friction force between the wheel and the axle. Tw and Ta are the respective torques between the wheel and floor and the wheel and axle.

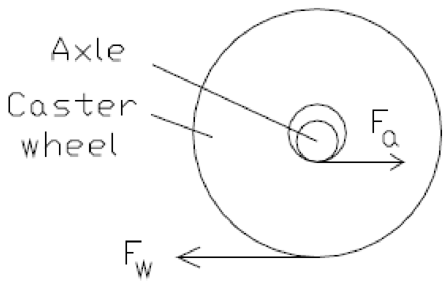


Figure 3

Fa = W/2 * µa
Ta = Fa * Ra
Fw = W/2 * µw
Tw = Fw * Rw
Tw is the *maximum* torque the wheel can transmit to the ground before it slips.

Our goal is to find a realistic range for Tm, the motor torque.
As calculated above, Tw would be the *maximum* amount of torque the motor could transfer to the ground before the wheel begins to slip (ie Tm, max).
Typically, we desire µw > µa, so the wheel does not slip/slide across the floor, but rather rolls. We can easily look up the µa value for the axle/wheel materials in contact. Knowing µa and the weight of the vehicle, Fa can be computed. This is the *minimum* amount of force we would have to provide at the wheel/axle interface to overcome the friction between the two. To relate the computed axle force Fa to the *minimum* amount of

wheel torque required to move the robot, we would use the “virtual radius” of the wheel/axle combination, which is computed as follows:

Rv = Rw – Ra

This is the fictitious radius about which Fa would act to rotate the wheel about the tangent point in contact with the ground at any instant, as shown in Figure 4 below.

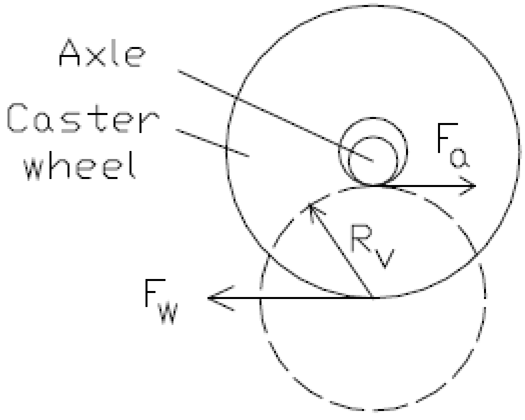


Figure 4

Therefore our equation for the *minimum* amount of torque the motor must transfer to the ground before the wheel begins to roll (thus causing the robot to move) would be:

Tm (min = Fa * Rv = Fa * (Rw – Ra)

In summation, Tm, min ≤ Tm ≤ Tm, max or alternatively, Fa * (Rw – Ra) ≤ Tm ≤ Fw * Rw

Appendix: Motor Data Calculation:

It is very important to measure different electrical and mechanical parameters of your motor and calculate unknown values using the following helpful formulas. We will use the International System of Units (SI). This is modern metric system that is officially accepted in electrical engineering in the USA.

One of the most important laws of physics is the fundamental Ohm's Law. It states that current through the conductor is directly proportional to applied voltage and is expressed as:

$$\mathbf{I = V / R}$$

where I – current, measured in amperes (A);
V – applied voltage, measured in volts (V);
R – resistance, measured in ohms (Ω).

This formula could be used in many cases. You may calculate the resistance of your motor by measuring the consumed current and applied voltage. For any given resistance (in the motors it is basically the resistance of the coil) this formula explains that the current can be controlled by applied voltage.

The consumed electrical power of the motor is defined by the following formula:

$$\mathbf{P_{in} = I * V}$$

where P_{in} – input power, measured in watts (W);
I – current, measured in amperes (A);
V – applied voltage, measured in volts (V).

Motors supposed to do some work and two important values define how powerful the motor is. It is motor speed and torque – the turning force of the motor. Output mechanical power of the motor could be calculated by using the following formula:

$$\mathbf{P_{out} = \tau * \omega}$$

where P_{out} – output power, measured in watts (W);
τ – torque, measured in Newton meters (N•m);
ω – angular speed, measured in radians per second (rad/s).

It is easy to calculate angular speed if you know rotational speed of the motor in rpm:

$$\mathbf{\omega = rpm * 2\pi / 60}$$

where ω – angular speed, measured in radians per second (rad/s);
rpm – rotational speed in revolutions per minute;
π – mathematical constant pi (3.14).
60 – number of seconds in a minute.

If the motor has 100% efficiency all electrical power is converted to mechanical energy. However such motors do not exist. Even precision made small industrial motors such as one we use as a generator in generator kit have maximum efficiency of 50-60%. Motors built from our kits usually have maximum efficiency of about 15% (see *Experiments* section on how we estimated this).

Don't be disappointed with 15% maximum efficiency. All our kits are intended for education and not designed for real applications. This efficiency is not bad at all – it is actually much better than most of other self made designs on Internet can provide. The motors have enough torque and speed to do all kinds of experiments and calculations.

Measuring the torque of the motor is a challenging task. It requires special expensive equipment. Therefore we suggest calculating it.

Efficiency of the motor is calculated as mechanical output power divided by electrical input power:

$$\mathbf{E = P_{out} / P_{in}}$$

therefore

$$\mathbf{P_{out} = P_{in} * E}$$

after substitution we get

$$\boldsymbol{\tau * \omega = I * V * E}$$

$$\boldsymbol{\tau * rpm * 2\pi / 60 = I * V * E}$$

and the formula for calculating torque will be

$$\boldsymbol{\tau = (I * V * E *60) / (rpm * 2\pi)}$$

Connect the motor to the load. Using the motor from generator kit is the best way to do it. Why do you need to connect the motor to the load? Well, if there is no load – there is no torque.

Measure current, voltage and rpm. Now you can calculate the torque for this load at this speed assuming that you know efficiency of the motor.

Our estimated 15% efficiency represents maximum efficiency of the motor which occurs only at a certain speed. Efficiency may be anywhere between zero and the maximum; in our example below 1000 rpm may not be the optimal speed so the for the sake of calculations you may use 10% efficiency (E = 0.1).

Example: speed is 1000 rpm, voltage is 6 Volts, and current is 220 mA (0.22 A):

$$\boldsymbol{\tau = (0.22 * 6 * 0.1 * 60) / (1000 * 2 * 3.14) = 0.00126\ N{\bullet}m}$$

As the result is small usually it is expressed in milliNewton meters (mN•m). There is 1000 mN•m in 1 N•m, so the calculated torque is 1.26 mN•m. It could be also converted further to still common gram force centimeters (g-cm) by multiplying the result by 10.2, i.e. the torque is 12.86 g-cm.

In our example input electrical power of the motor is 0.22 A x 6 V = 1.32 W, output mechanical power is 1000 rpm x 2 x 3.14 x 0.00126 N•m /60 = 0.132 W.

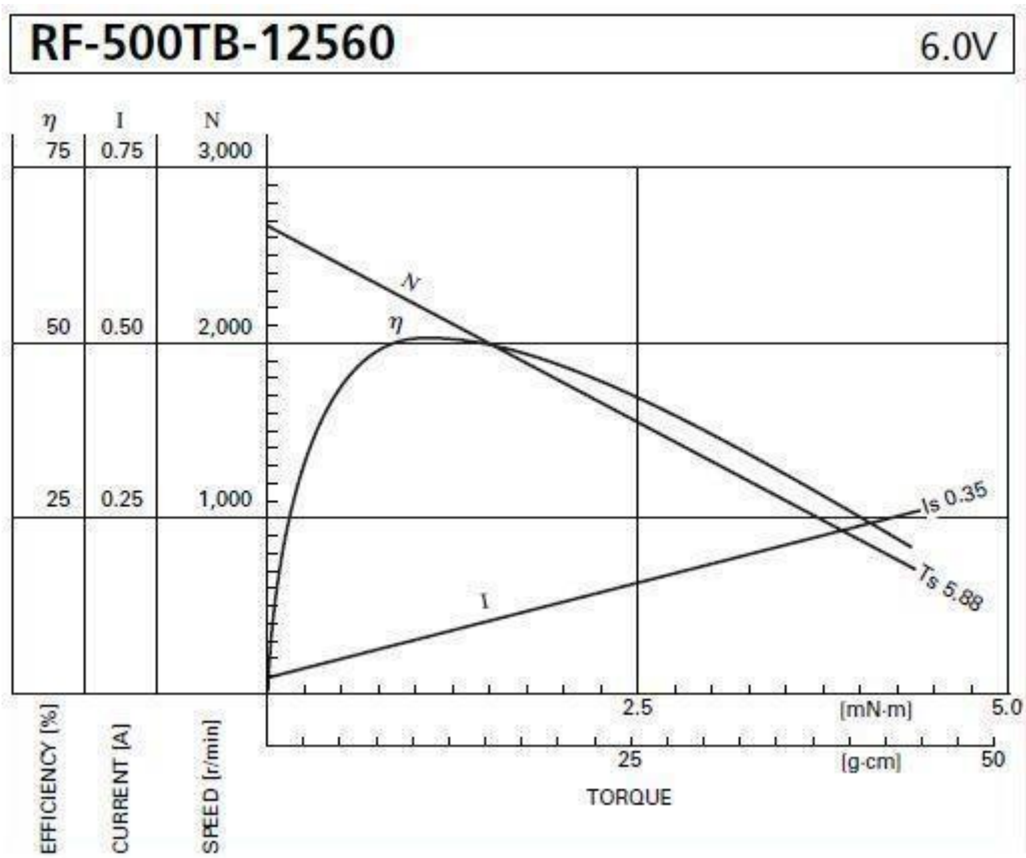
Motor torque changes with the speed. At no load you have maximum speed and zero torque. Load adds mechanical resistance. The motor starts to consume more current to overcome this resistance and the speed decreases. If you increase the load at some point motor stops (this is called stall). When it occurs the torque is at maximum and it is called stall torque. While it is hard to measure stall torque without special tools you can find this value by plotting speed-torque graph. You need to take at least two measurements with different loads to find the stall torque.

How accurate is the torque calculation? While voltage, current and speed could be accurately measured, efficiency of the motor may not be correct. It depends on the accuracy of your assembly, sensor position, friction, alignment of the motor and generator axles etc.

Speed, torque, power and efficiency of the motors are not constant values. Usually the manufacturer provides the following data in a table like this one (sample data from one of the motors used in generator kit):

MODEL	VOLTAGE		NO LOAD		AT MAXIMUM EFFICIENCY					STALL		
	OPERATING RANGE	NOMINAL	SPEED	CURRENT	SPEED	CURRENT	TORQUE		OUTPUT	TORQUE		CURRENT
			r/min	A	r/min	A	mN·m	g·cm	W	mN·m	g·cm	A
RF-500TB-12560	1.5~12.0	6V CONSTANT	2700	0.020	2180	0.084	1.13	11.6	0.26	5.88	60	0.35

Also the manufacturers usually provide power curves for the motor at nominal voltage:



These curves are generated by plotting motor speed, consumed current, and efficiency as functions of the motor torque. Sometimes there might be also a curve representing mechanical output power.

As you can see from the graph speed and current are linear functions of torque so you might need only two measurements to draw these graphs. Efficiency and power will need more data. Usually for small motors maximum power is at 50% of stall torque (approximately 50% of no load speed). Maximum efficiency may be 10-30% of motor stall torque (70-90% of no load speed).

While it is technically better to follow the same format and create similar curves for your motor it is not absolutely necessary for a good science project. You may take all measurements, calculate unknown values and plot the graphs where for example speed and torque are represented as functions of applied voltage or current etc.

Simple formulas and calculations described here are essential for calculating most common motor parameters. However this is a simplified approach that does not take into consideration many factors. If you want to extend your research further – see Links section and search the Internet. There is tons of information with more complex calculations.

Web Resources:

Motor power calculation: https://simplemotor.com/calculations/

Motors, Fans and Accessories Selection

40x40x10 mm DC Brushless Cooling Fan

Ultra quiet powerful brushless DC fan, quiet sleeve-bearing design. Specialized design, professional made, stable performance. Operating Temperature: -10 C to +60C. Long Life Expectancy.

EMH-1071	**GDT4010S12B**	**RM 6.50**

GA12-N20 Geared Mini DC Motor

This is a DC Mini Metal Gear Motor, ideal for making robots. Light weight, high torque and low RPM. Fine craftsmanship, durable, not easy to wear. Widely used on boat, model car, robotic, home appliances, linear motion control.

EMH-1176	**GA12-N20**	**RM 18.50**

30x30x10 mm DC Brushless Cooling Fan

Ultra quiet powerful brushless DC fan, quiet sleeve-baring design. Specialized design, professional made, stable performance. Operating Temperature: -10 C to +60C. Long Life Expectancy.

EMH-1070	**GDT3010S12B**	**RM 7.50**

Nema23 Bipolar/Unipolar Stepper Motor 1.0A

A stepper motor to satisfy all your 3D-Printer, robotics, Linear Motion projects needs! This 6-wire uni-polar/bipolar stepper motor has 1.8° per step for smooth motion and a nice holding torque.

EMH-1179	**23HS2610**	**RM 110.00**

1.2A Nema 17 Stepper Motor

A stepper motor to satisfy all your 3D-Printer, robotics, Linear Motion projects needs! This 4-wire bipolar stepper has 1.8° per step for smooth motion and a nice holding torque.

EMH-1016	**42HS40-1204D**	**RM 44.50**

1.7 A Nema 17 Stepper Motor

A stepper motor to satisfy all your 3D-Printer, robotics, Linear Motion projects needs! This 4-wire bipolar stepper has 1.8 ° per step for smooth motion and a nice holding torque.

EMH-1181	**17HS-4401SD**	**RM 47.00**

SG90 Tower Pro Gear Micro Servo Motor

Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction). Good for beginners who want to make stuff move without building a motor controller with feedback & gear box.

EMH-1140	**TPSG90S**	**RM 7.40**

Nema-17 Planetary Geared Stepper Motor

This high precision NEMA17 Stepper motor has an integrated Planetary Gearbox with 1:5.18 gear ratio, the resolution can reach 0.35°step angle.

EMH-1173	**42BYGP40P**	**RM 185.00**

Web Resources:

- 68mm High Grip Rubber Wheel for Robotics Car
- Hex Motor Shaft Coupler for Robotic Wheel
- Right Angle Bracket for JGB37 Gear Motor

HandsOn Technology provides a multimedia and interactive platform for everyone interested in electronics. From beginner to diehard, from student to lecturer. Information, education, inspiration and entertainment. Analog and digital, practical and theoretical; software and hardware.

open source hardware

Hands*On* Technology support Open Source Hardware (OSHW) Development Platform.

Learn : Design : Share

www.handsontec.com

The Face behind our product quality…

In a world of constant change and continuous technological development, a new or replacement product is never far away – and they all need to be tested.
Many vendors simply import and sell wihtout checks and this cannot be the ultimate interests of anyone, particularly the customer. Every part sell on Handsotec is fully tested. So when buying from Handsontec products range, you can be confident you're getting outstanding quality and value.

We keep adding the new parts so that you can get rolling on your next project.

Breakout Boards & Modules

Connectors

Electro-Mechanical Parts

Engineering Material

Mechanical Hardware

P

Electronics Components

Power Supply

Arduino Board & Shield

Tools & Accessory

Anexo 7
Driver DRV8871

DRV8871 3.6-A Brushed DC Motor Driver With Internal Current Sense (PWM Control)

1 Features

- H-Bridge Motor Driver
  - Drives One DC Motor, One Winding of a Stepper Motor, or Other Loads
- Wide 6.5-V to 45-V Operating Voltage
- 565-mΩ Typical $R_{DS(on)}$ (HS + LS)
- 3.6-A Peak Current Drive
- PWM Control Interface
- Current Regulation Without a Sense Resistor
- Low-Power Sleep Mode
- Small Package and Footprint
  - 8-Pin HSOP With PowerPAD™
  - 4.9 × 6 mm
- **Integrated Protection Features**
  - VM Undervoltage Lockout (UVLO)
  - Overcurrent Protection (OCP)
  - Thermal Shutdown (TSD)
  - Automatic Fault Recovery

2 Applications

- Printers
- Appliances
- Industrial Equipment
- Other Mechatronic Applications

3 Description

The DRV8871 device is a brushed-DC motor driver for printers, appliances, industrial equipment, and other small machines. Two logic inputs control the H-bridge driver, which consists of four N-channel MOSFETs that can control motors bidirectionally with up to 3.6-A peak current. The inputs can be pulse-width modulated (PWM) to control motor speed, using a choice of current-decay modes. Setting both inputs low enters a low-power sleep mode.

The DRV8871 device has advanced current-regulation circuitry that does not use an analog voltage reference or external sense resistor. This novel solution uses a standard low-cost, low-power resistor to set the current threshold. The ability to limit current to a known level can significantly reduce the system power requirements and bulk capacitance needed to maintain stable voltage, especially for motor startup and stall conditions.

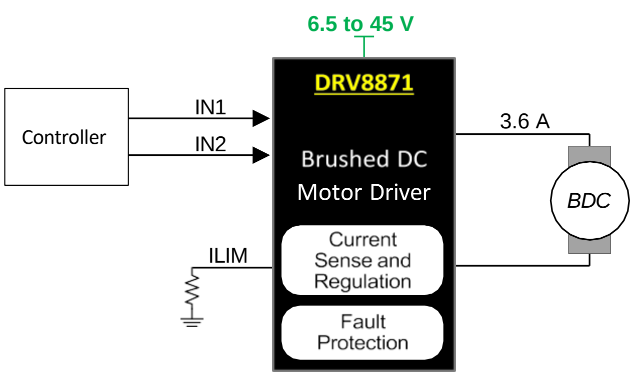
The device is fully protected from faults and short circuits, including undervoltage (UVLO), overcurrent (OCP), and overtemperature (TSD). When the fault condition is removed, the device automatically resumes normal operation.

Device Information (1)

PART NUMBER	PACKAGE	BODY SIZE (NOM)
DRV8871	HSOP (8)	4.90 mm × 6.00 mm

(1) For all available packages, see the orderable addendum at the end of the data sheet.

Simplified Schematic



Peak Current Regulation

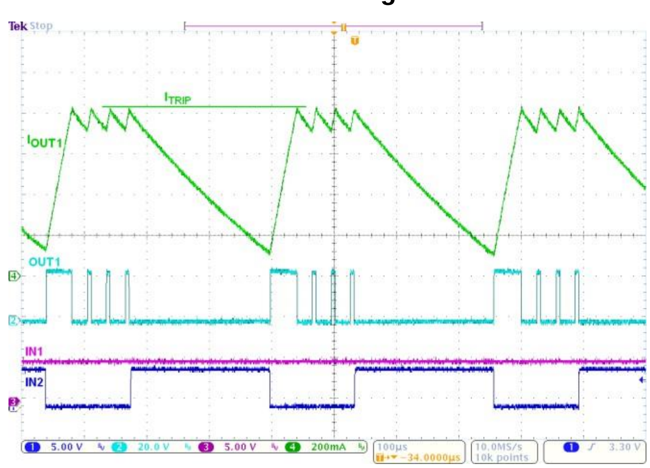


Table of Contents

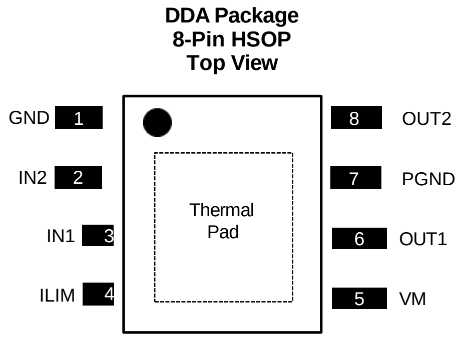
4 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision A (January 2016) to Revision B	Page
• Deleted the power supply voltage ramp rate (VM) parameter from the *Absolute Maximum Ratings* table	3
• Added the output current parameter to the *Absolute Maximum Ratings* table	3
• Added the *Receiving Notification of Documentation Updates* section	17

Changes from Original (August 2015) to Revision A	Page
• Updated the f_{PWM} max value and added a note	4
• Removed the redundant T_A condition and added f_{PWM} = 24 kHz	5
• Added more information to clarify how the max RMS current varies for different applications	12

5 Pin Configuration and Functions



Pin Functions

PIN		TYPE	DESCRIPTION	
NAME	NO.			
GND	1	PWR	Logic ground	Connect to board ground
ILIM	4	I	Current limit control	Connect a resistor to ground to set the current chopping threshold
IN1	3	I	Logic inputs	Controls the H-bridge output. Has internal pulldowns (see Table 1).
IN2	2			
OUT1	6	O	H-bridge output	Connect directly to the motor or other inductive load.
OUT2	8			
PGND	7	PWR	High-current ground path	Connect to board ground.
VM	5	PWR	6.5-V to 45-V power supply	Connect a 0.1-µF bypass capacitor to ground, as well as sufficient bulk capacitance, rated for the VM voltage.
PAD	—	—	Thermal pad	Connect to board ground. For good thermal dissipation, use large ground planes on multiple layers, and multiple nearby vias connecting those planes.

6 Specifications

6.1 Absolute Maximum Ratings

over operating free-air temperature range (unless otherwise noted)(1)

	MIN	MAX	UNIT
Power supply voltage (VM)	–0.3	50	V
Logic input voltage (IN1, IN2)	–0.3	7	V
Continuous phase node pin voltage (OUT1, OUT2)	–0.7	VM + 0.7	V
Output current (100% duty cycle)	0	3.5	A
Operating junction temperature, T_J	–40	150	°C
Storage temperature, T_{stg}	–65	150	°C

(1) Stresses beyond those listed under *Absolute Maximum Ratings* may cause permanent damage to the device. These are stress ratings only, which do not imply functional operation of the device at these or any other conditions beyond those indicated under *Recommended Operating Conditions*. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

6.2 ESD Ratings

			VALUE	UNIT
$V_{(ESD)}$	Electrostatic discharge	Human-body model (HBM), per ANSI/ESDA/JEDEC JS-001(1)	±6000	V
		Charged-device model (CDM), per JEDEC specification JESD22-C101(2)	±750	

(1) JEDEC document JEP155 states that 500-V HBM allows safe manufacturing with a standard ESD control process.
(2) JEDEC document JEP157 states that 250-V CDM allows safe manufacturing with a standard ESD control process.

6.3 Recommended Operating Conditions

over operating free-air temperature range (unless otherwise noted)

		MIN	MAX	UNIT
VM	Power supply voltage	6.5	45	V
V_I	Logic input voltage (IN1, IN2)	0	5.5	V
f_{PWM}	Logic input PWM frequency (IN1, IN2)	0	200(1)	kHz
I_{peak}	Peak output current(2)	0	3.6	A
T_A	Operating ambient temperature(2)	–40	125	°C

(1) The voltages applied to the inputs should have at least 800 ns of pulse width to ensure detection. Typical devices require at least 400 ns. If the PWM frequency is 200 kHz, the usable duty cycle range is 16% to 84%.
(2) Power dissipation and thermal limits must be observed

6.4 Thermal Information

THERMAL METRIC(1)		DRV8871 DDA (HSOP) 8 PINS	UNIT
$R_{\theta JA}$	Junction-to-ambient thermal resistance	41.1	°C/W
$R_{\theta JC(top)}$	Junction-to-case (top) thermal resistance	53.1	°C/W
$R_{\theta JB}$	Junction-to-board thermal resistance	23.1	°C/W
ψ_{JT}	Junction-to-top characterization parameter	8.2	°C/W
ψ_{JB}	Junction-to-board characterization parameter	23	°C/W
$R_{\theta JC(bot)}$	Junction-to-case (bottom) thermal resistance	2.7	°C/W

(1) For more information about traditional and new thermal metrics, see the *Semiconductor and IC Package Thermal Metrics* application report (SPRA953).

6.5 Electrical Characteristics

T_A = 25°C, over recommended operating conditions (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
POWER SUPPLY (VM)						
VM	VM operating voltage		6.5		45	V
I_{VM}	VM operating supply current	VM = 12 V		3	10	mA
$I_{VMSLEEP}$	VM sleep current	VM = 12 V			10	µA
t_{ON} (1)	Turn-on time	VM > V_{UVLO} with IN1 or IN2 high		40	50	µs
LOGIC-LEVEL INPUTS (IN1, IN2)						
V_{IL}	Input logic low voltage				0.5	V
V_{IH}	Input logic high voltage		1.5			V
V_{HYS}	Input logic hysteresis			0.5		V
I_{IL}	Input logic low current	V_{IN} = 0 V	–1		1	µA
I_{IH}	Input logic high current	V_{IN} = 3.3 V		33	100	µA
R_{PD}	Pulldown resistance	To GND		100		kΩ
t_{PD}	Propagation delay	INx to OUTx change (see Figure 6)		0.7	1	µs
t_{sleep}	Time to sleep	Inputs low to sleep		1	1.5	ms
MOTOR DRIVER OUTPUTS (OUT1, OUT2)						
$R_{DS(ON)}$	High-side FET on resistance	VM = 24 V, I = 1 A, f_{PWM} = 25 kHz		307	360	mΩ
$R_{DS(ON)}$	Low-side FET on resistance	VM = 24 V, I = 1 A, f_{PWM} = 25 kHz		258	320	mΩ
t_{DEAD}	Output dead time			220		ns
V_d	Body diode forward voltage	I_{OUT} = 1 A		0.8	1	V
CURRENT REGULATION						
V_{ILIM}	Constant for calculating current regulation (see Equation 1)	I_{OUT} = 1 A	59	64	69	kV
t_{OFF}	PWM off-time			25		µs
t_{BLANK}	PWM blanking time			2		µs
PROTECTION CIRCUITS						
V_{UVLO}	VM undervoltage lockout	VM falls until UVLO triggers		6.1	6.4	V
		VM rises until operation recovers		6.3	6.5	
$V_{UV,HYS}$	VM undervoltage hysteresis	Rising to falling threshold	100	180		mV
I_{OCP}	Overcurrent protection trip level		3.7	4.5	6.4	A
t_{OCP}	Overcurrent deglitch time			1.5		µs
t_{RETRY}	Overcurrent retry time			3		ms
T_{SD}	Thermal shutdown temperature		150	175		°C
T_{HYS}	Thermal shutdown hysteresis			40		°C

(1) t_{ON} applies when the device initially powers up, and when it exits sleep mode.

6.6 Typical Characteristics

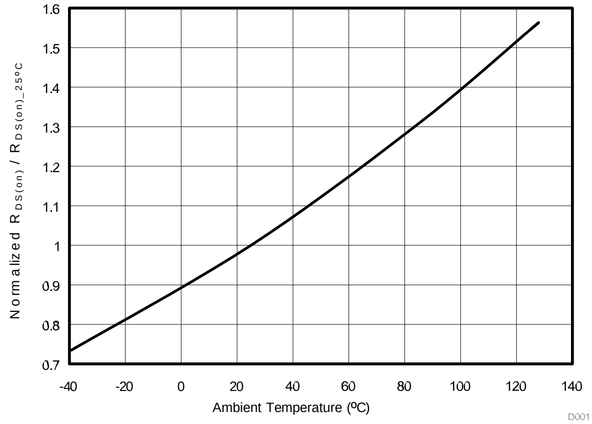


Figure 1. $R_{DS(on)}$ vs Temperature

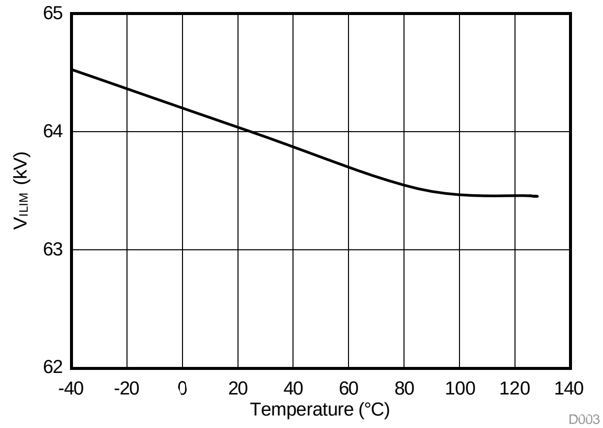


Figure 2. V_{ILIM} vs Temperature

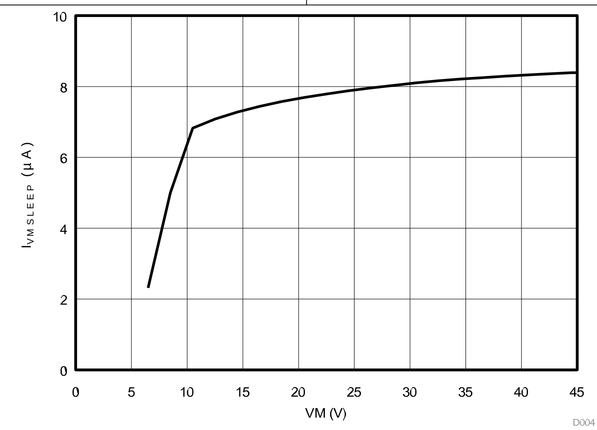


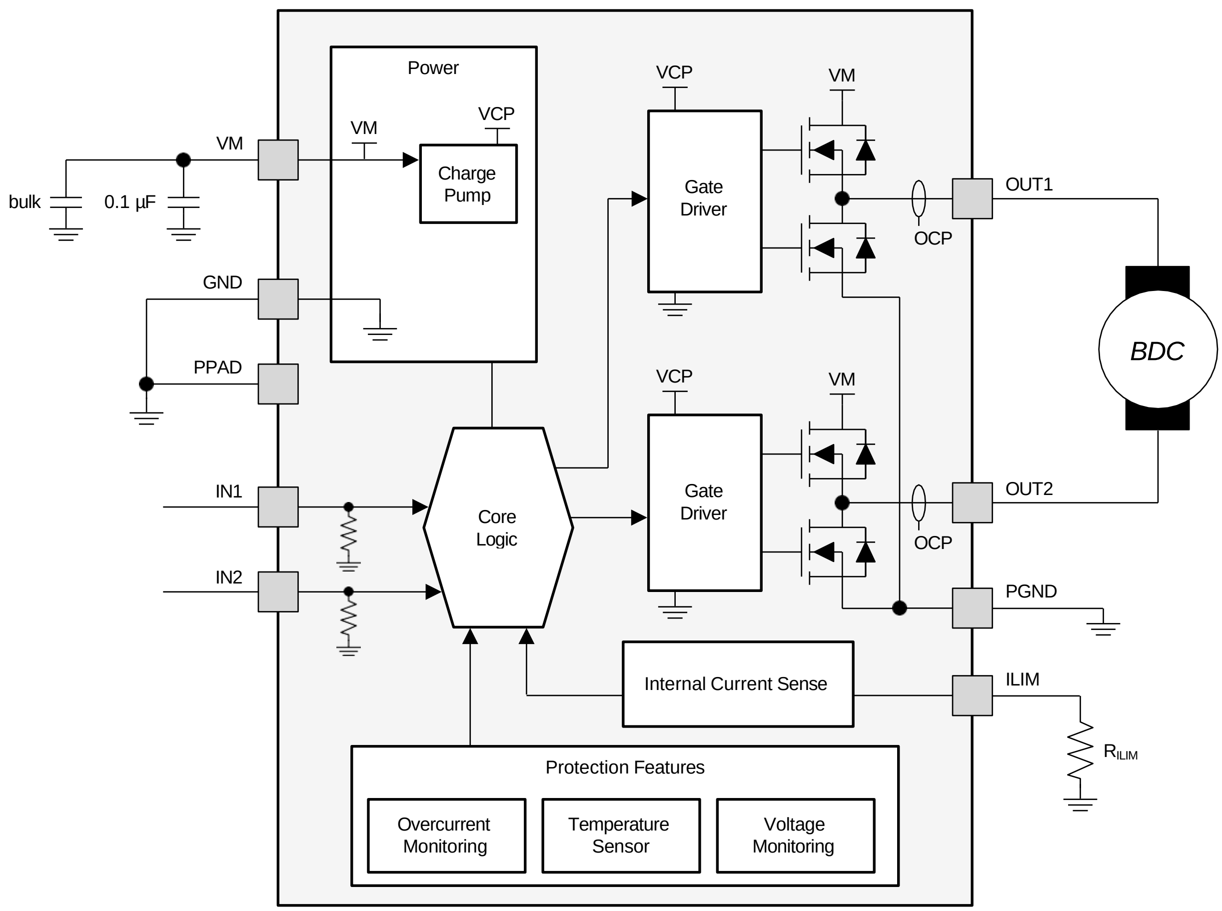
Figure 3. $I_{VMSLEEP}$ vs VM at 25°C

7 Detailed Description

7.1 Overview

The DRV8871 device is an optimized 8-pin device for driving brushed DC motors with 6.5 to 45 V and up to 3.6-A peak current. The integrated current regulation restricts motor current to a predefined maximum. Two logic inputs control the H-bridge driver, which consists of four N-channel MOSFETs that have a typical $R_{ds(on)}$ of 565 mΩ (including one high-side and one low-side FET). A single power input, VM, serves as both device power and the motor winding bias voltage. The integrated charge pump of the device boosts VM internally and fully enhances the high-side FETs. Motor speed can be controlled with pulse-width modulation, at frequencies between 0 to 100 kHz. The device has an integrated sleep mode that is entered by bringing both inputs low. An assortment of protection features prevent the device from being damaged if a system fault occurs.

7.2 Functional Block Diagram



7.3 Feature Description

7.3.1 Bridge Control

The DRV8871 output consists of four N-channel MOSFETs that are designed to drive high current. They are controlled by the two logic inputs IN1 and IN2, according to Table 1.

Table 1. H-Bridge Control

IN1	IN2	OUT1	OUT2	DESCRIPTION
0	0	High-Z	High-Z	Coast; H-bridge disabled to High-Z (sleep entered after 1 ms)
0	1	L	H	Reverse (Current OUT2 → OUT1)
1	0	H	L	Forward (Current OUT1 → OUT2)
1	1	L	L	Brake; low-side slow decay

The inputs can be set to static voltages for 100% duty cycle drive, or they can be pulse-width modulated (PWM) for variable motor speed. When using PWM, it typically works best to switch between driving and braking. For example, to drive a motor forward with 50% of its max RPM, IN1 = 1 and IN2 = 0 during the driving period, and IN1 = 1 and IN2 = 1 during the other period. Alternatively, the coast mode (IN1 = 0, IN2 = 0) for *fast current decay* is also available. The input pins can be powered before VM is applied.

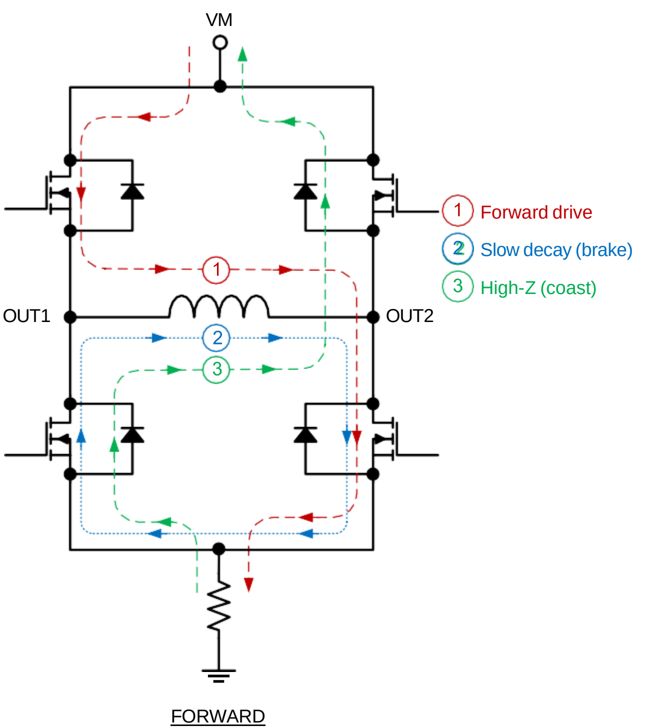


Figure 4. H-Bridge Current Paths

7.3.2 Sleep Mode

When IN1 and IN2 are both low for time t_{SLEEP} (typically 1 ms), the DRV8871 device enters a low-power sleep mode, where the outputs remain High-Z and the device uses $I_{VMSLEEP}$ (microamps) of current. If the device is powered up while both inputs are low, sleep mode is immediately entered. After IN1 or IN2 are high for at least 5 µs, the device will be operational 50 µs (t_{ON}) later.

7.3.3 Current Regulation

The DRV8871 device limits the output current based on a standard resistor attached to pin ILIM, according to this equation:

$$I_{TRIP}\ (A) = \frac{V_{ILIM}\ (kV)}{R_{ILIM}\ (k\Omega)} = \frac{64\ (kV)}{R_{ILIM}\ (k\Omega)} \tag{1}$$

For example, if R_{ILIM} = 32 kΩ, the DRV8871 device limits motor current to 2 A no matter how much load torque is applied. The minimum allowed R_{ILIM} is 15 kΩ. System designers should always understand the min and max I_{TRIP}, based on the R_{ILIM} resistor component tolerance and the DRV8871 specified V_{ILIM} range.

When I_{TRIP} has been reached, the device enforces slow current decay by enabling both low-side FETs, and it does this for time t_{OFF} (typically 25 µs).

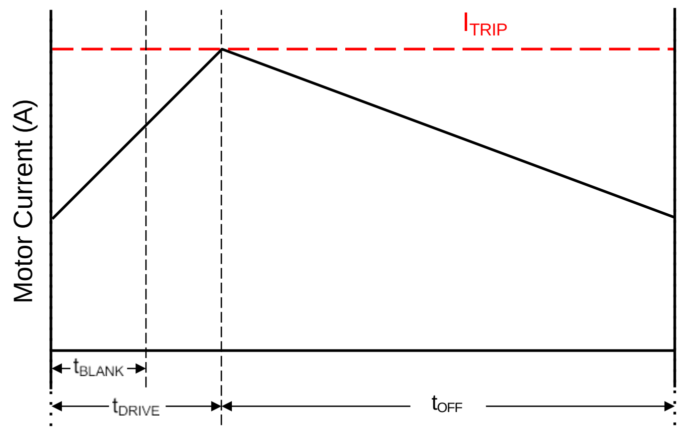


Figure 5. Current Regulation Time Periods

After t_{OFF} has elapsed, the output is re-enabled according to the two inputs INx. The drive time (t_{DRIVE}) until reaching another I_{TRIP} event heavily depends on the VM voltage, the motor's back-EMF, and the motor's inductance.

7.3.4 Dead Time

When an output changes from driving high to driving low, or driving low to driving high, dead time is automatically inserted to prevent shoot-through. t_{DEAD} is the time in the middle when the output is High-Z. If the output pin is measured during t_{DEAD}, the voltage will depend on the direction of current. If current is leaving the pin, the voltage will be a diode drop below ground. If current is entering the pin, the voltage will be a diode drop above VM. This diode is the body diode of the high-side or low-side FET.

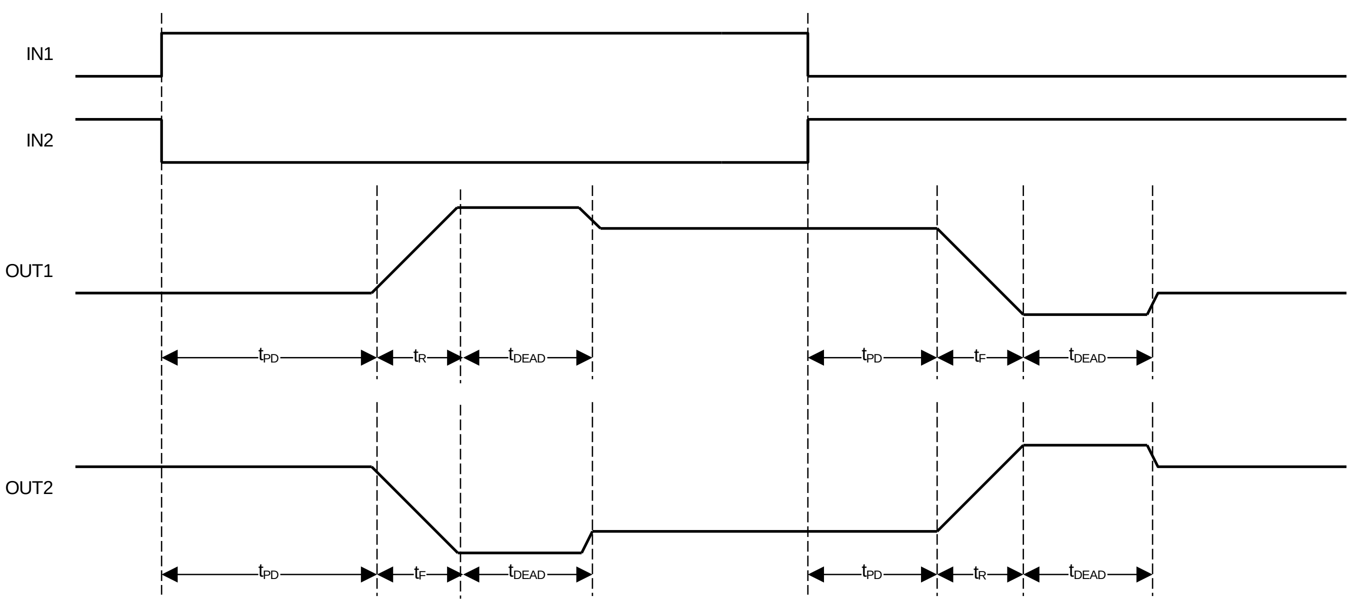


Figure 6. Propagation Delay Time

7.3.5 Protection Circuits

The DRV8871 device is fully protected against VM undervoltage, overcurrent, and overtemperature events.

7.3.5.1 VM Undervoltage Lockout (UVLO)

If at any time the voltage on the VM pin falls below the undervoltage lockout threshold voltage, all FETs in the H-bridge will be disabled. Operation will resume when VM rises above the UVLO threshold.

7.3.5.2 Overcurrent Protection (OCP)

If the output current exceeds the OCP threshold I_{OCP} for longer than t_{OCP}, all FETs in the H-bridge are disabled for a duration of t_{RETRY}. After that, the H-bridge will be re-enabled according to the state of the INx pins. If the overcurrent fault is still present, the cycle repeats; otherwise normal device operation resumes.

7.3.5.3 Thermal Shutdown (TSD)

If the die temperature exceeds safe limits, all FETs in the H-bridge will be disabled. After the die temperature has fallen to a safe level, operation automatically resumes.

Table 2. Protection Functionality

FAULT	CONDITION	H-BRIDGE STATUS	RECOVERY
VM undervoltage lockout (UVLO)	VM < V_{UVLO}	Disabled	VM > V_{UVLO}
Overcurrent (OCP)	$I_{OUT} > I_{OCP}$	Disabled	t_{RETRY}
Thermal Shutdown (TSD)	T_J > 150°C	Disabled	$T_J < T_{SD} – T_{HYS}$

7.4 Device Functional Modes

The DRV8871 device can be used in multiple ways to drive a brushed DC motor.

7.4.1 PWM With Current Regulation

This scheme uses all of the device capabilities. I_{TRIP} is set above the normal operating current, and high enough to achieve an adequate spin-up time, but low enough to constrain current to a desired level. Motor speed is controlled by the duty cycle of one of the inputs, while the other input is static. Brake/slow decay is typically used during the off-time.

7.4.2 PWM Without Current Regulation

If current regulation is not needed, a 15-kΩ to 18-kΩ resistor should be used on pin ILIM. This mode provides the highest possible peak current: up to 3.6 A for a few hundred milliseconds (depending on PCB characteristics and the ambient temperature). If current exceeds 3.6 A, the device might reach overcurrent protection (OCP) or overtemperature shutdown (TSD). If that happens, the device disables and protects itself for about 3 ms (t_{RETRY}) and then resumes normal operation.

7.4.3 Static Inputs With Current Regulation

IN1 and IN2 can be set high and low for 100% duty cycle drive, and I_{TRIP} can be used to control the current, speed, and torque capability of the motor.

7.4.4 VM Control

In some systems it is desirable to vary VM as a means of changing motor speed. See *Motor Voltage* for more information.

8 Application and Implementation

NOTE

Information in the following applications sections is not part of the TI component specification, and TI does not warrant its accuracy or completeness. TI's customers are responsible for determining suitability of components for their purposes. Customers should validate and test their design implementation to confirm system functionality.

8.1 Application Information

The DRV8871 device is typically used to drive one brushed DC motor.

8.2 Typical Application

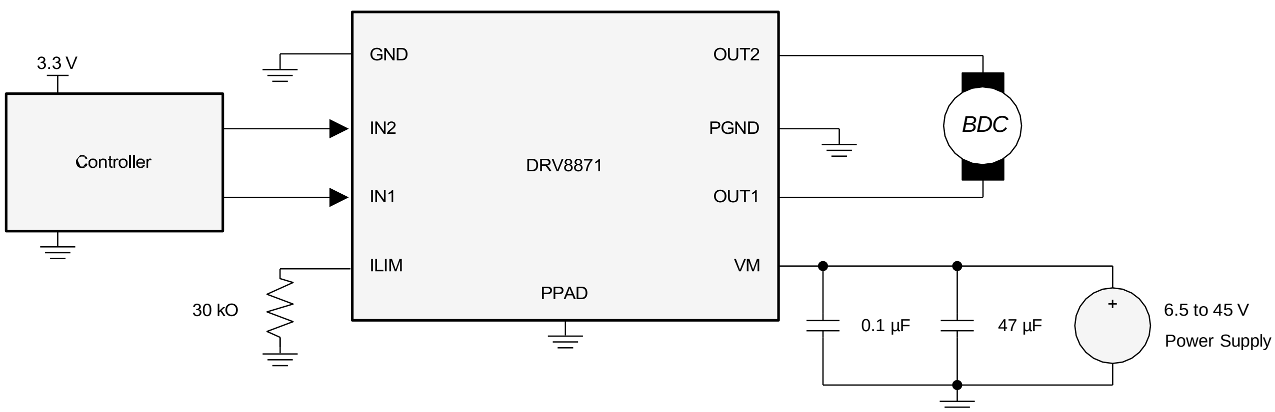


Figure 7. Typical Connections

8.2.1 Design Requirements

Table 3 lists the design parameters.

Table 3. Design Parameters

DESIGN PARAMETER	REFERENCE	EXAMPLE VALUE
Motor voltage	V_M	24 V
Motor RMS current	I_{RMS}	0.8 A
Motor startup current	I_{START}	2 A
Motor current trip point	I_{TRIP}	2.1 A
ILIM resistance	R_{ILIM}	30 kΩ
PWM frequency	f_{PWM}	5 kHz

8.2.2 Detailed Design Procedure

8.2.2.1 Motor Voltage

The motor voltage to use will depend on the ratings of the motor selected and the desired RPM. A higher voltage spins a brushed DC motor faster with the same PWM duty cycle applied to the power FETs. A higher voltage also increases the rate of current change through the inductive motor windings.

8.2.2.2 Drive Current

The current path is through the high-side sourcing DMOS power driver, motor winding, and low-side sinking DMOS power driver. Power dissipation losses in one source and sink DMOS power driver are shown in the following equation.

$$P_D = I^2 \left(R_{DS(on)Source} + R_{DS(on)Sink} \right) \quad (2)$$

The DRV8871 device has been measured to be capable of 2-A RMS current at 25°C on standard FR-4 PCBs. The max RMS current varies based on the PCB design, ambient temperature, and PWM frequency. Typically, switching the inputs at 200 kHz compared to 20 kHz causes 20% more power loss in heat.

8.2.3 Application Curves

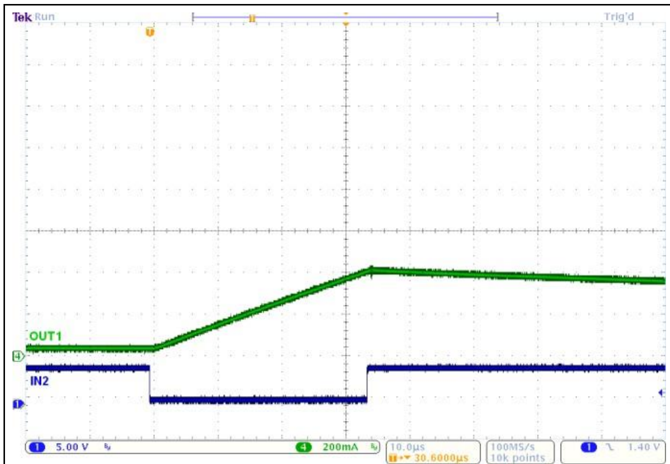


Figure 8. Current Ramp With a 2-Ω, 1 mH, RL Load and VM = 12 V

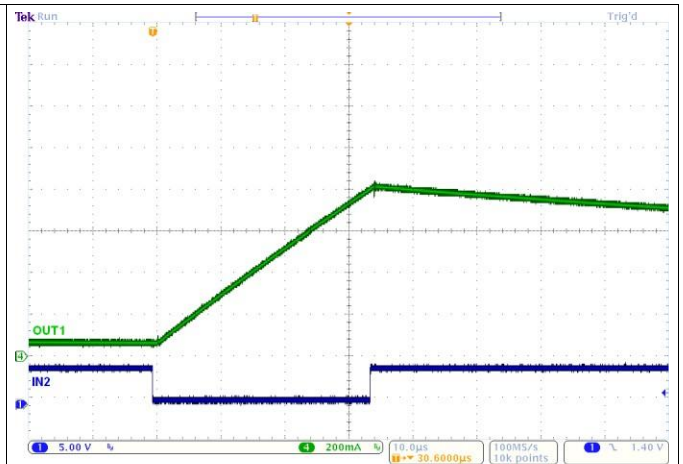


Figure 9. Current Ramp With a 2-Ω, 1 mH, RL Load and VM = 24 V

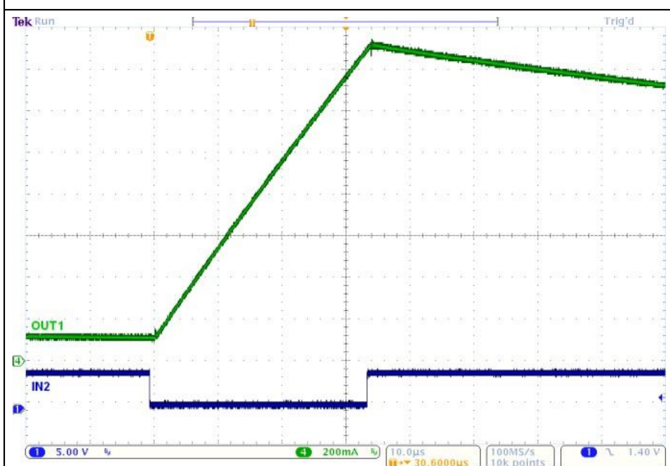


Figure 10. Current Ramp With a 2-Ω, 1 mH, RL Load and VM = 45 V

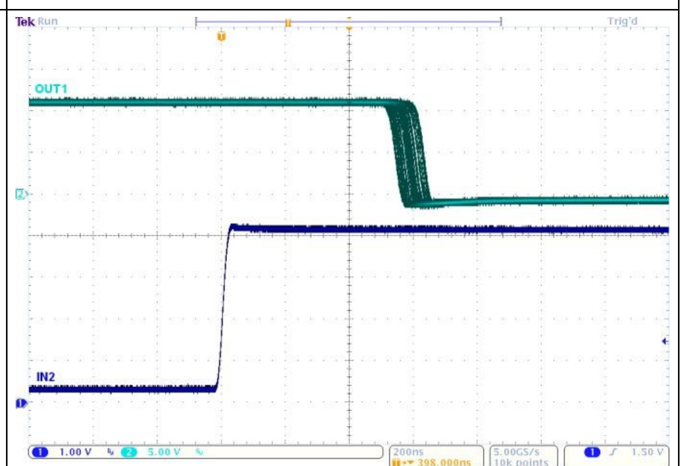


Figure 11. t_{PD}

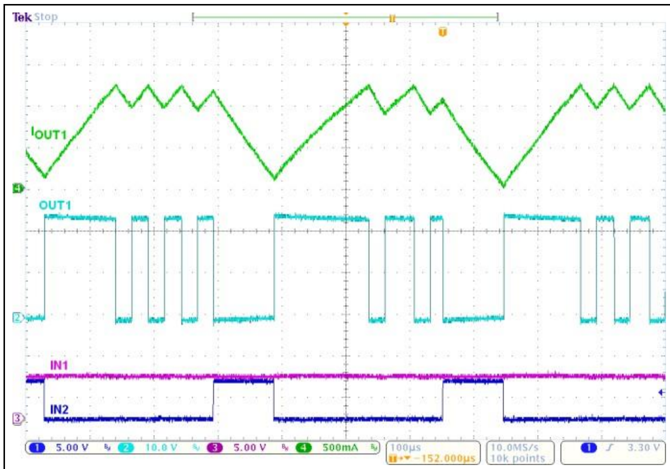


Figure 12. Current Regulation With R_{ILIM} = 50.5 kΩ

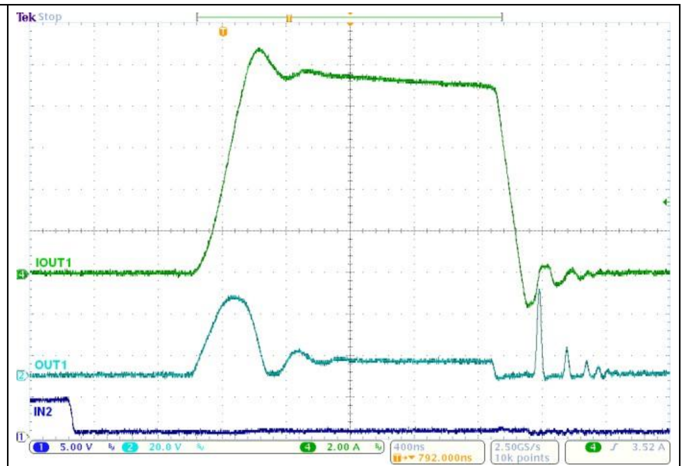


Figure 13. OCP With 45 V and the Outputs Shorted Together

9 Power Supply Recommendations

9.1 Bulk Capacitance

Having appropriate local bulk capacitance is an important factor in motor drive system design. In general, having have more bulk capacitance is beneficial, while the disadvantages are increased cost and physical size.

The amount of local capacitance needed depends on a variety of factors, including:

- The highest current required by the motor system
- The power supply's capacitance and ability to source current
- The amount of parasitic inductance between the power supply and motor system
- The acceptable voltage ripple
- The type of motor used (brushed DC, brushless DC, stepper)
- The motor braking method

The inductance between the power supply and motor drive system will limit the rate current can change from the power supply. If the local bulk capacitance is too small, the system reponds to excessive current demands or dumps from the motor with a change in voltage. When adequate bulk capacitance is used, the motor voltage remains stable and high current can be quickly supplied.

The data sheet generally provides a recommended value, but system-level testing is required to determine the appropriate sized bulk capacitor.

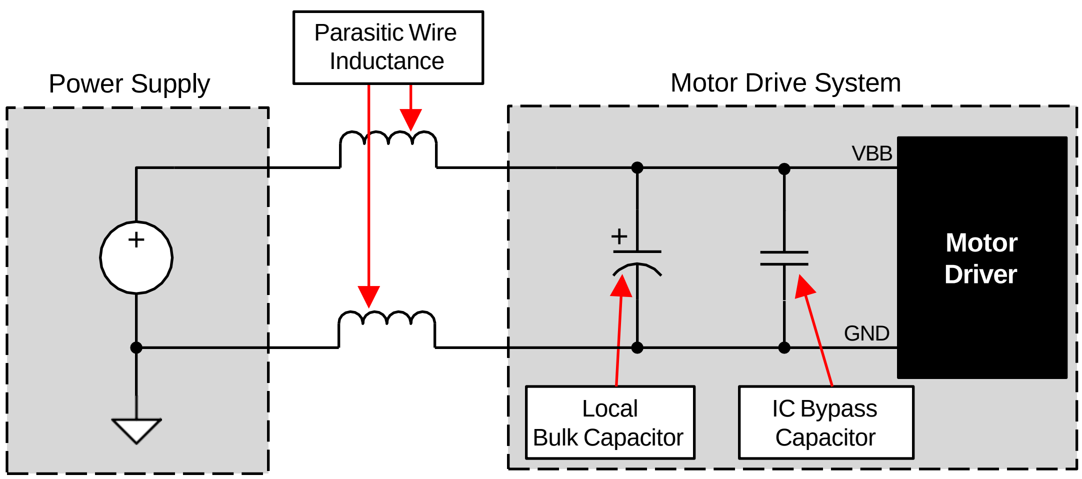


Figure 14. Example Setup of Motor Drive System With External Power Supply

The voltage rating for bulk capacitors should be higher than the operating voltage, to provide margin for cases when the motor transfers energy to the supply.

10 Layout

10.1 Layout Guidelines

The bulk capacitor should be placed to minimize the distance of the high-current path through the motor driver device. The connecting metal trace widths should be as wide as possible, and numerous vias should be used when connecting PCB layers. These practices minimize inductance and allow the bulk capacitor to deliver high current.

Small-value capacitors should be ceramic, and placed closely to device pins.

The high-current device outputs should use wide metal traces.

The device thermal pad should be soldered to the PCB top-layer ground plane. Multiple vias should be used to connect to a large bottom-layer ground plane. The use of large metal planes and multiple vias help dissipate the $I^2 \times R_{DS(on)}$ heat that is generated in the device.

10.2 Layout Example

Recommended layout and component placement is shown in Figure 15

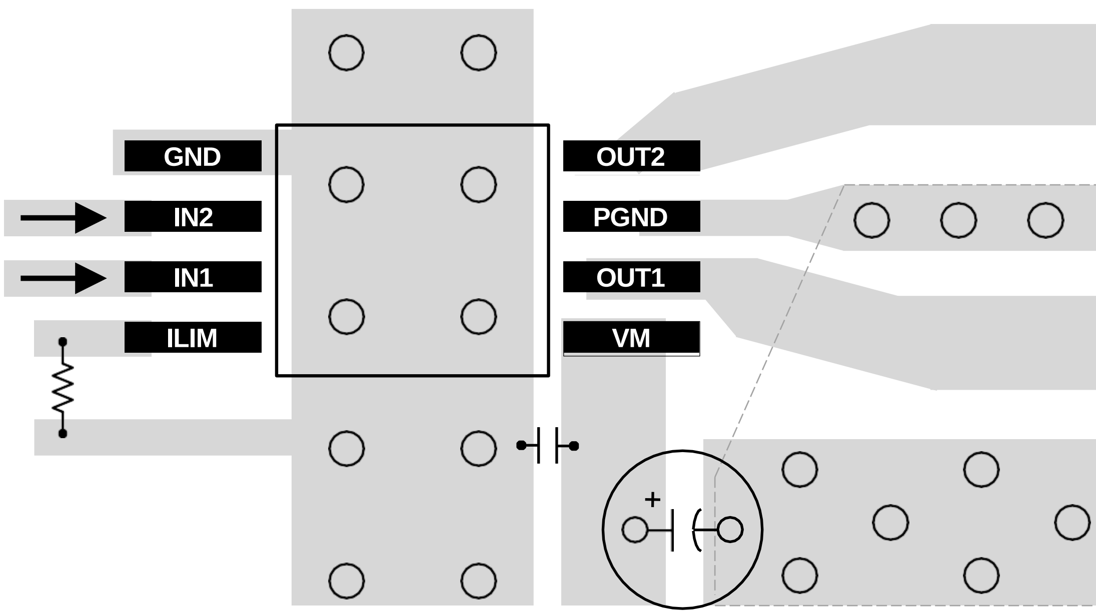


Figure 15. Layout Recommendation

10.3 Thermal Considerations

The DRV8871 device has thermal shutdown (TSD) as described in the *Thermal Shutdown (TSD)* section. If the die temperature exceeds approximately 175°C, the device is disabled until the temperature drops below the temperature hysteresis level.

Any tendency of the device to enter TSD is an indication of either excessive power dissipation, insufficient heatsinking, or too high of an ambient temperature.

10.4 Power Dissipation

Power dissipation in the DRV8871 device is dominated by the power dissipated in the output FET resistance, $R_{DS(on)}$. Use the equation in the *Drive Current* section to calculate the estimated average power dissipation when driving a load.

Note that at startup, the current is much higher than normal running current; this peak current and its duration must be also be considered.

Power Dissipation (continued)

The maximum amount of power that can be dissipated in the device is dependent on ambient temperature and heatsinking.

NOTE

$R_{DS(on)}$ increases with temperature, so as the device heats, the power dissipation increases. This fact must be taken into consideration when sizing the heatsink.

The power dissipation of the DRV8871 device is a function of RMS motor current and the FET resistance ($R_{DS(ON)}$) of each output.

$$\text{Power} :: I_{RMS}^2 \times \left(\text{High-side } R_{DS(ON)} + \text{ Low-side } R_{DS(ON)}\right) \tag{3}$$

For this example, the ambient temperature is 58°C, and the junction temperature reaches 80°C. At 58°C, the sum of $R_{DS(ON)}$ is about 0.72 Ω. With an example motor current of 0.8 A, the dissipated power in the form of heat will be 0.8 A^2 × 0.72 Ω = 0.46 W.

The temperature that the DRV8871 device reaches depends on the thermal resistance to the air and PCB. It is important to solder the device PowerPAD to the PCB ground plane, with vias to the top and bottom board layers, in order dissipate heat into the PCB and reduce the device temperature. In the example used here, the DRV8871 device had an effective thermal resistance $R_{\theta JA}$ of 48°C/W, and:

$$T_J = T_A + (P_D \times R_{\theta JA}) = 58°C + (0.46\ W \times 48°C/W) = 80°C \tag{4}$$

10.4.1 Heatsinking

The PowerPAD package uses an exposed pad to remove heat from the device. For proper operation, this pad must be thermally connected to copper on the PCB to dissipate heat. On a multi-layer PCB with a ground plane, this connection can be accomplished by adding a number of vias to connect the thermal pad to the ground plane.

On PCBs without internal planes, a copper area can be added on either side of the PCB to dissipate heat. If the copper area is on the opposite side of the PCB from the device, thermal vias are used to transfer the heat between top and bottom layers.

For details about how to design the PCB, refer to *PowerPAD™ Thermally Enhanced Package* (SLMA002) and *PowerPAD Made Easy™* (SLMA004), available at www.ti.com. In general, the more copper area that can be provided, the more power can be dissipated.

11 Device and Documentation Support

11.1 Documentation Support

11.1.1 Related Documentation

For related documentation, see the following:

- *Current Recirculation and Decay Modes*
- *Calculating Motor Driver Power Dissipation*
- *DRV8871 Evaluation Module*
- *PowerPAD™ Thermally Enhanced Package*
- *PowerPAD™ Made Easy*
- *Understanding Motor Driver Current Ratings*

11.2 Receiving Notification of Documentation Updates

To receive notification of documentation updates, navigate to the device product folder on ti.com. In the upper right corner, click on *Alert me* to register and receive a weekly digest of any product information that has changed. For change details, review the revision history included in any revised document.

11.3 Community Resources

The following links connect to TI community resources. Linked contents are provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

TI E2E™ Online Community ***TI's Engineer-to-Engineer (E2E) Community.*** Created to foster collaboration among engineers. At e2e.ti.com, you can ask questions, share knowledge, explore ideas and help solve problems with fellow engineers.

Design Support ***TI's Design Support*** Quickly find helpful E2E forums along with design support tools and contact information for technical support.

11.4 Trademarks

PowerPAD, E2E are trademarks of Texas Instruments.
All other trademarks are the property of their respective owners.

11.5 Electrostatic Discharge Caution

These devices have limited built-in ESD protection. The leads should be shorted together or the device placed in conductive foam during storage or handling to prevent electrostatic damage to the MOS gates.

11.6 Glossary

SLYZ022 — *TI Glossary*.

This glossary lists and explains terms, acronyms, and definitions.

12 Mechanical, Packaging, and Orderable Information

The following pages include mechanical, packaging, and orderable information. This information is the most current data available for the designated devices. This data is subject to change without notice and revision of this document. For browser-based versions of this data sheet, refer to the left-hand navigation.

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead finish/ Ball material (6)	MSL Peak Temp (3)	Op Temp (°C)	Device Marking (4/5)	Samples
DRV8871DDA	ACTIVE	SO PowerPAD	DDA	8	75	RoHS & Green	NIPDAUAG	Level-2-260C-1 YEAR	-40 to 125	8871	Samples
DRV8871DDAR	ACTIVE	SO PowerPAD	DDA	8	2500	RoHS & Green	NIPDAUAG	Level-2-260C-1 YEAR	-40 to 125	8871	Samples

(1) The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

(2) **RoHS:** TI defines "RoHS" to mean semiconductor products that are compliant with the current EU RoHS requirements for all 10 RoHS substances, including the requirement that RoHS substance do not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, "RoHS" products are suitable for use in specified lead-free processes. TI may reference these types of products as "Pb-Free".
RoHS Exempt: TI defines "RoHS Exempt" to mean products that contain lead but are compliant with EU RoHS pursuant to a specific EU RoHS exemption.
Green: TI defines "Green" to mean the content of Chlorine (Cl) and Bromine (Br) based flame retardants meet JS709B low halogen requirements of <=1000ppm threshold. Antimony trioxide based flame retardants must also meet the <=1000ppm threshold requirement.

(3) MSL, Peak Temp. - The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

(4) There may be additional marking, which relates to the logo, the lot trace code information, or the environmental category on the device.

(5) Multiple Device Markings will be inside parentheses. Only one Device Marking contained in parentheses and separated by a "~" will appear on a device. If a line is indented then it is a continuation of the previous line and the two combined represent the entire Device Marking for that device.

(6) Lead finish/Ball material - Orderable Devices may have multiple material finish options. Finish options are separated by a vertical ruled line. Lead finish/Ball material values may wrap to two lines if the finish value exceeds the maximum column width.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

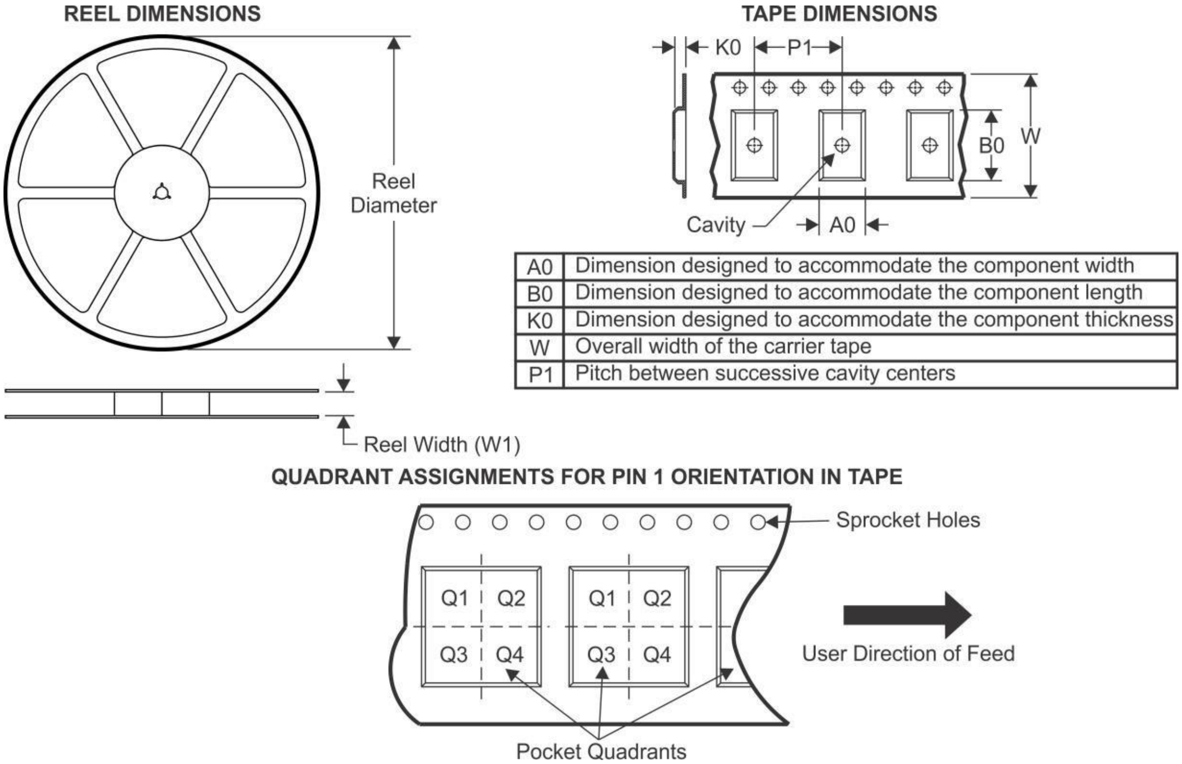
OTHER QUALIFIED VERSIONS OF DRV8871 :

- Automotive: DRV8871-Q1

NOTE: Qualified Version Definitions:

- Automotive - Q100 devices qualified for high-reliability automotive applications targeting zero defects

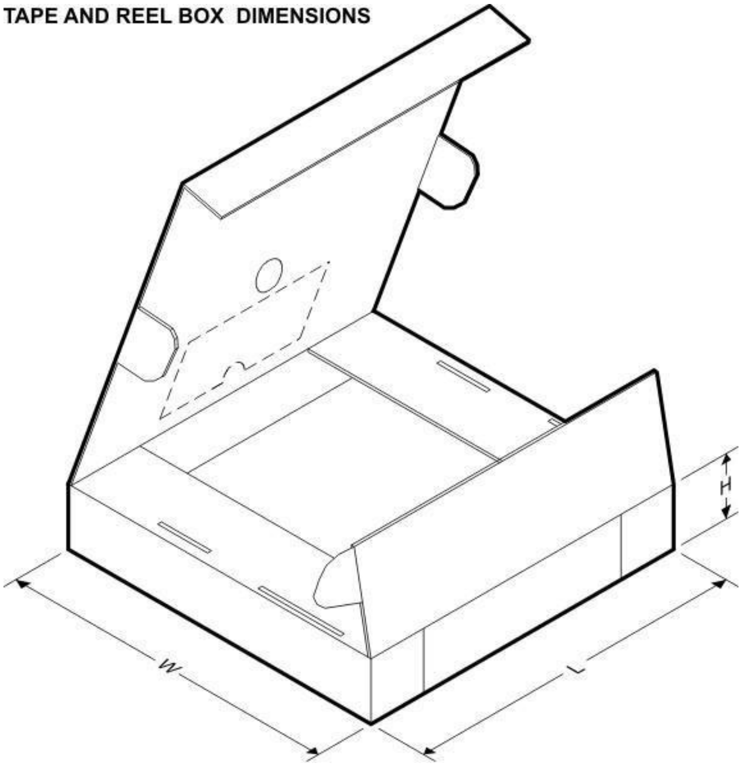
TAPE AND REEL INFORMATION



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Reel Diameter (mm)	Reel Width W1 (mm)	A0 (mm)	B0 (mm)	K0 (mm)	P1 (mm)	W (mm)	Pin1 Quadrant
DRV8871DDAR	SO Power PAD	DDA	8	2500	330.0	12.8	6.4	5.2	2.1	8.0	12.0	Q1

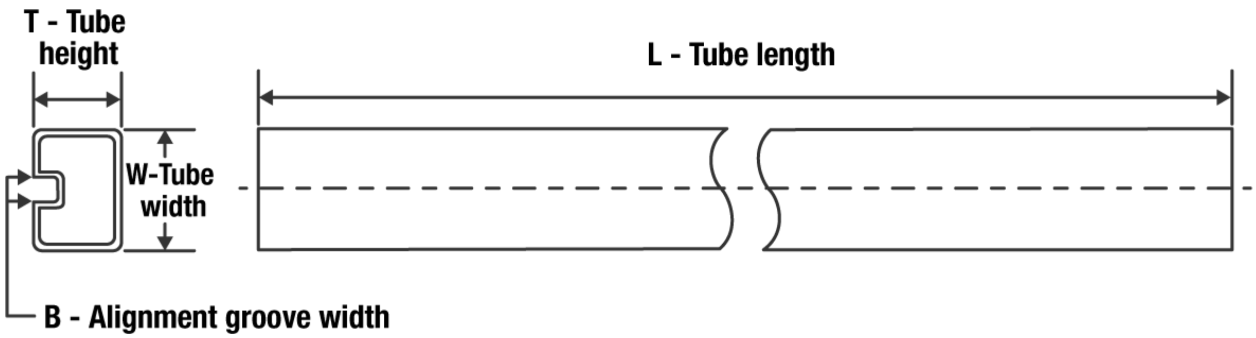
TAPE AND REEL BOX DIMENSIONS



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Length (mm)	Width (mm)	Height (mm)
DRV8871DDAR	SO PowerPAD	DDA	8	2500	366.0	364.0	50.0

TUBE

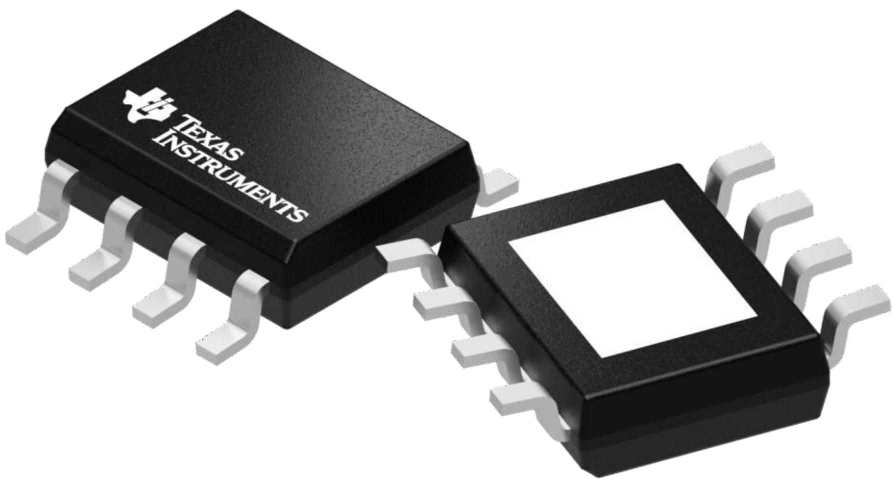


*All dimensions are nominal

Device	Package Name	Package Type	Pins	SPQ	L (mm)	W (mm)	T (µm)	B (mm)
DRV8871DDA	DDA	HSOIC	8	75	517	7.87	635	4.25

DDA 8 PowerPAD™ SOIC - 1.7 mm max height

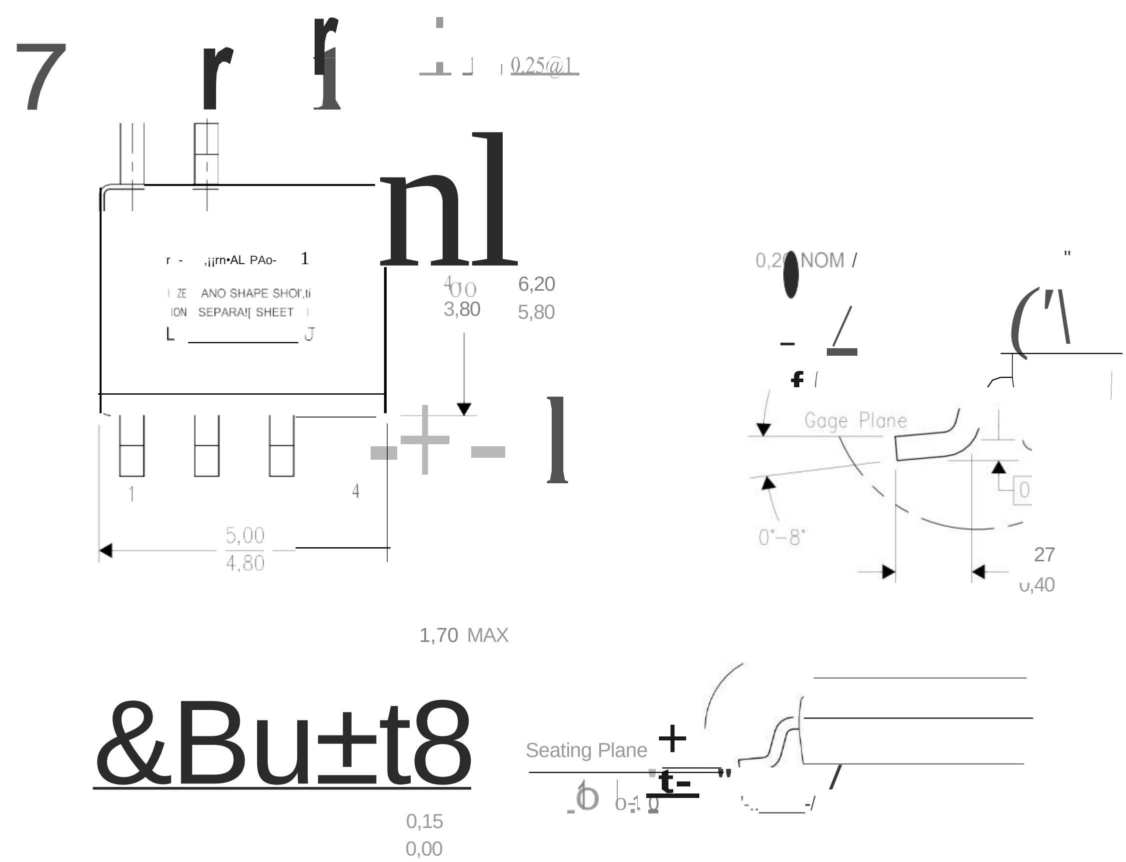
PLASTIC SMALL OUTLINE



Images above are just a representation of the package family, actual package may vary.
Refer to the product data sheet for package details.

4202561/G

ODA (R-PDSO-G8) PowerPAD TM PLASTIC SMALL-OUTLINE



4202561/F 12/11

NOTES:
A. All linear dimensions are in millimeters. Dimensioning and tolerancing per ASME Y14.5-1994.
B. This drawing is subject to change without notice.
C. Body dimensions do no\ include mold flash or protrusion not to exceed O,15.
D. This package is designed to be soldered to a thermal pad on the board. Refer to Technical Brief, PowerPad Thermally Enhanced Package, Texas lnstruments Literature No. SLMA002 for information regarding recommended board layout. This document is available a\ www.ti.com <http://www.ti.com>.
E. See the additional figure in the Product Data Sheet for details regarding the exposed thermal pad features and dimensions.
F. This package complies to JEDEC MS-012 variation BA

PowerPAD is a trademark of Texas lnstruments,

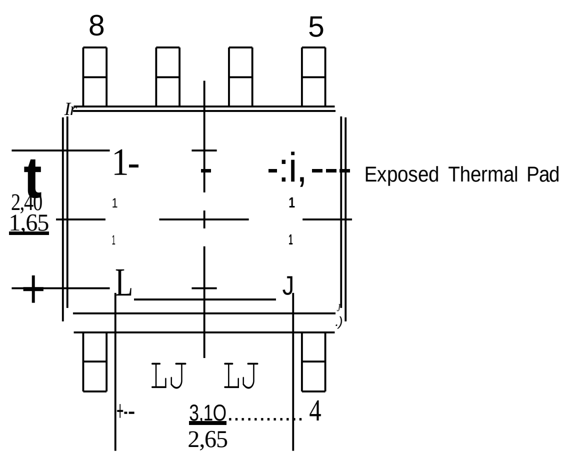
DDA (R-PDSO-G8) PowerPAD™ PLASTIC SMALL OUTLINE

THERMAL INFORMATION

This PowerPAD™ package incorporates an exposed thermal pad that is designed to be attached to a printed circuit board (PCB). The thermal pad must be soldered directly to the PCB. After soldering, the PCB can be used as a heatsink. In addition, through the use of thermal vias, the thermal pad can be attached directly to the appropriate copper plane shown in the electrical schematic for the device, or alternatively, can be attached to a special heatsink structure designed into the PCB. This design optimizes the heat transfer from the integrated circuit (IC).

For additional information on the PowerPAD package and how to take advantage of its heat dissipating abilities, refer to Technical Brief, PowerPAD Thermally Enhanced Package, Texas Instruments Literature No. SLMA002 and Application Brief, PowerPAD Made Easy, Texas Instruments Literature No. SLMA004. Both documents are available at www.ti.com.

The exposed thermal pad dimensions for this package are shown in the following illustration.



Top View

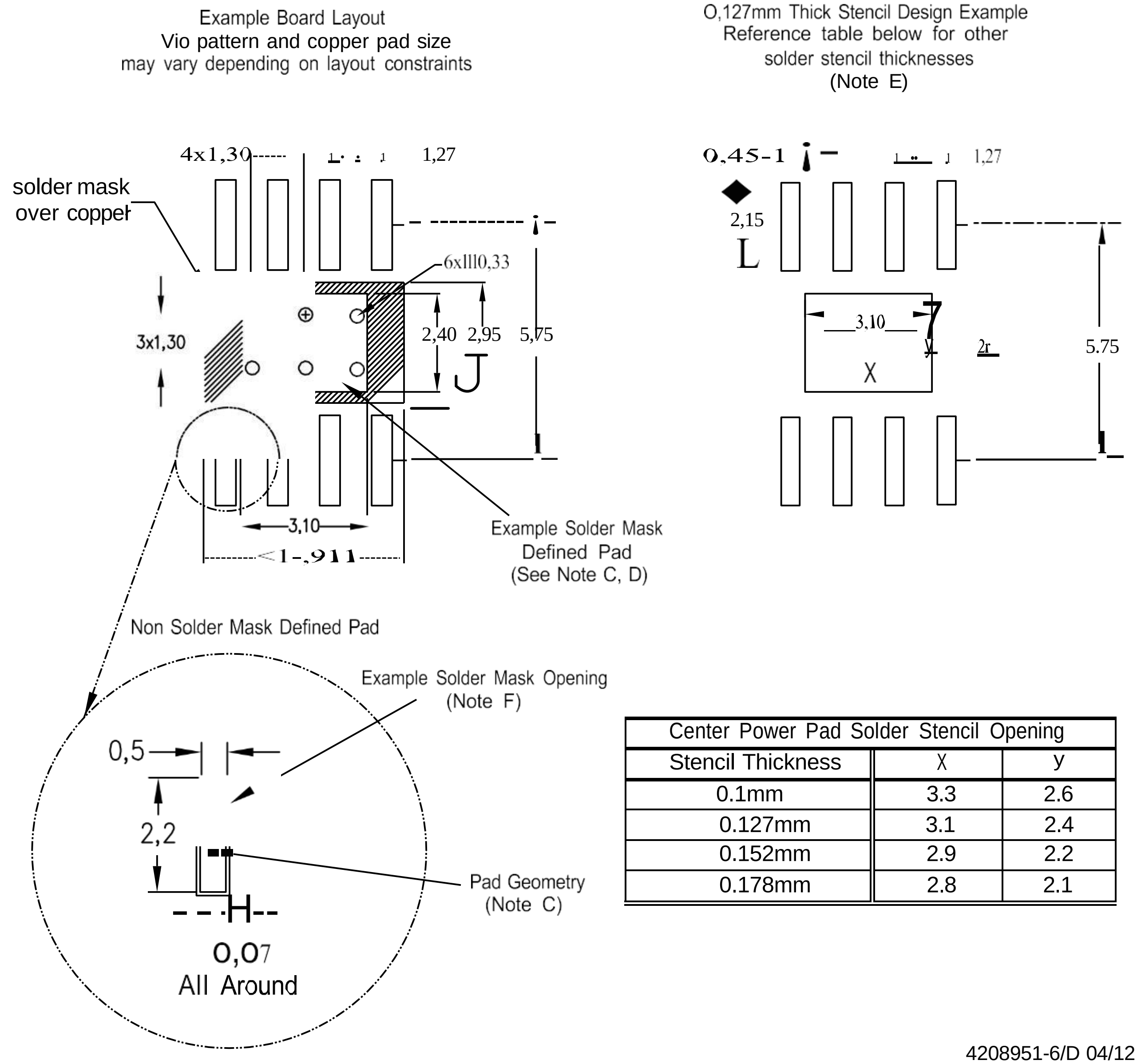
Exposed Thermal Pad Dimensions

4206322-6/L 05/12

NOTE: A. All linear dimensions are in millimeters

PowerPAD is a trademark of Texas Instruments

ODA (R-PDSO-GS) PowerPAD™ PLASTIC SMALL OUTLINE



Center Power Pad Solder Stencil Opening		
Stencil Thickness	X	y
0.1mm	3.3	2.6
0.127mm	3.1	2.4
0.152mm	2.9	2.2
0.178mm	2.8	2.1

4208951-6/D 04/12

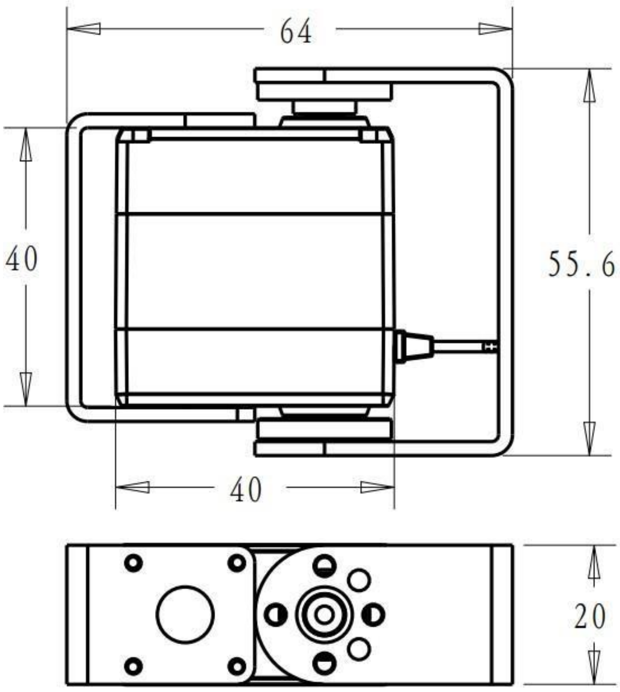
NOTES:
- A. All linear dimensions are in millimeters.
- B. This drawing is subjed to change without notice.
- C. Publication IPC-7351 is recommended for alternate designs.
- D. This package is designed to be soldered to a thermal pad on the board. Refer to Technical Brief, PowerPad Thermally Enhanced Package, Texas Instruments Literature No. SLMA002, SLMA004, and also the Produd Data Sheets for specific thermal information, vio requirements, and recommended board layout. These documents are available at www.ti.com <http://www.ti.com>. Publication IPC-7351 is recommended for alternate designs.
- E. Laser cutting apertures with trapezoidal walls and also rounding corners **will** offer better paste release. Customers should contad their board assembly site for stencil design recommendations. Example stencil design based on a 50% volumetric metal load solder paste. Refer to IPC-7525 for other stencil recommendations.
- F. Customers should contad their board fabrication site for solder mask tolerances between and around signal pads.

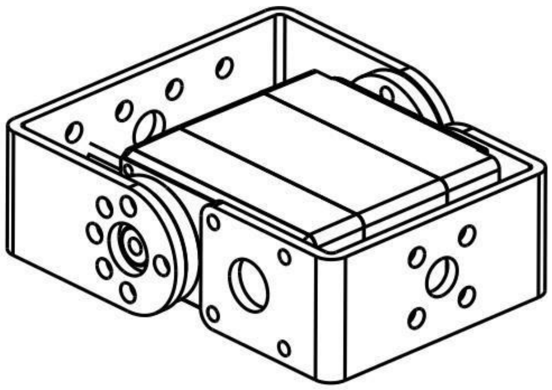
Anexo 8
Servomotor

东莞市达盛舵机科技有限公司
Dongguan City Dsservo Technology Co.Ltd

产品规格书（Product datasheet）

产品型号（Product Name）：RDS3225
产品描述（Product Description）：6V 25kg Robot Digital Servo
产 品 图（Drawing）





1. 使用环境条件 Apply Environmental Condition

No.	Item	Specification
1-1	存储温度 Storage Temperature Range	-30℃～80℃
1-2	运行温度 Operating Temperature Range	-15℃～70℃
1-3	工作电压范围 Operating Voltage Range	4.8-6.8V

2. 机械特性 Mechanical Specification

No.	Item	Specification
2-1	尺寸 Size	40*20*40mm
2-2	重量 Weight	60g
2-3	齿轮比 Gear ratio	373
2-4	轴承 Bearing	Double bearing
2-5	舵机线 Connector wire	300±5mm
2-6	马达 Motor	3-pole(s)
2-7	防水性能 Waterproof performance	No

更多的舵机规格书和 3D 模型，请到网站下载(For more servo datasheet and 3D files, please go to the website to download) www.dsservo.com

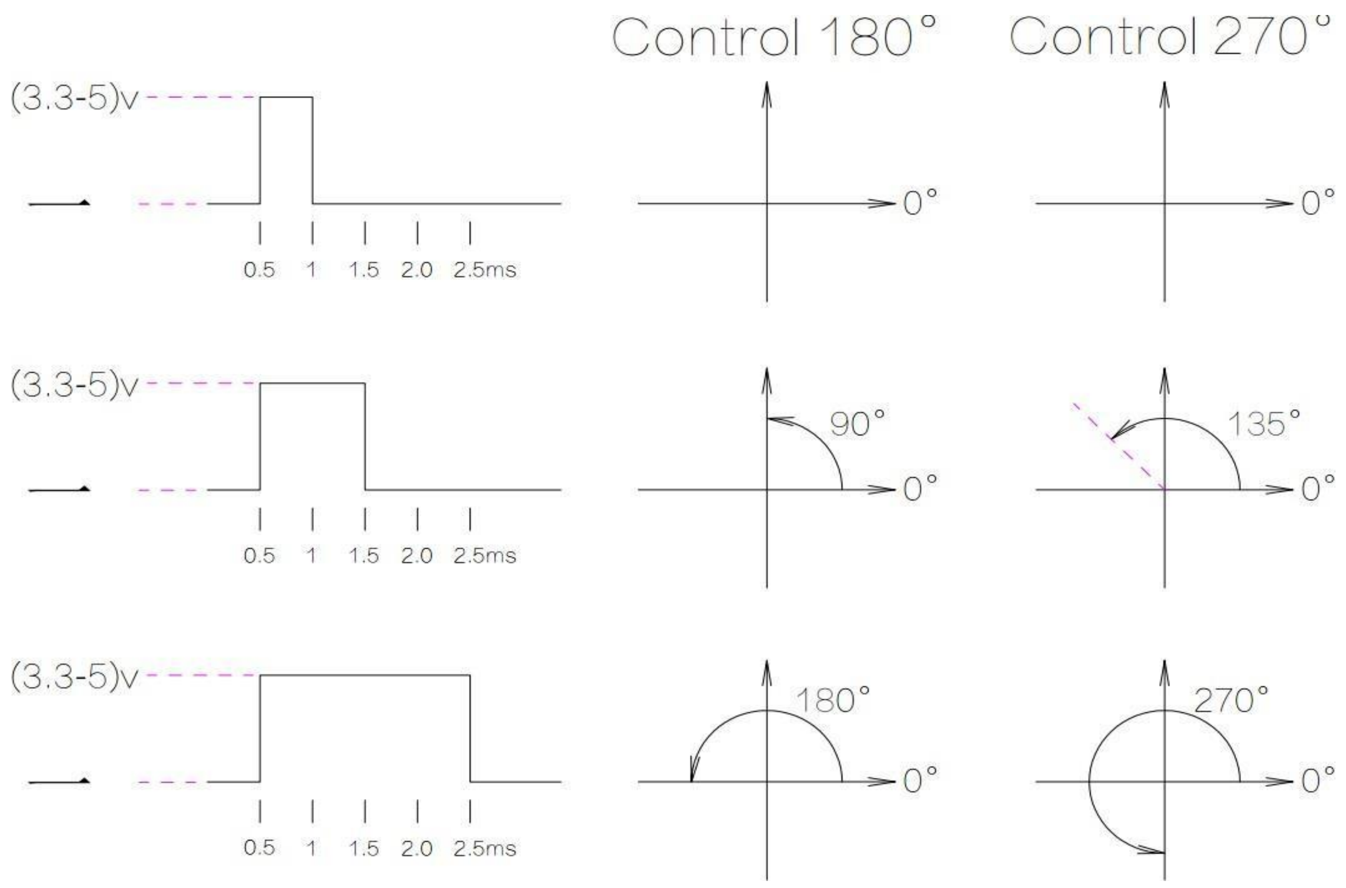
3. 电气特性 Electrical Specification

No.	工作电压 Operating Voltage	5V	6.8V
3-1	待机电流 Idle current(at stopped)	4mA	5mA
3-2	空载转速 Operating speed (at no load)	0.19 sec/60°	0.17sec/60°
3-3	堵转扭矩 Stall torque (at locked)	24 kg-cm	28 kg-cm
3-4	堵转电流 Stall current (at locked)	1.9A	2.3A

4. 控制特性 Control Specification

No.	Item	Specification
4-1	驱动方式 Control System	PWM(Pulse width modification)
4-2	脉宽范围 Pulse width range	500 ~ 2500μsec
4-3	中点位置 Neutral position	1500μsec
4-4	控制角度 Running degree	180° or 270° (when 500 ~ 2500 μ sec)
4-5	控制精度 Dead band width	3 μsec
4-6	控制频率 Operating frequency	50-330Hz
4-7	旋转方向 Rotating direction	Counterclockwise (when 500 ~ 2500 μsec)

5. 关于 PWM 控制说明 About PWM Control



更多的舵机规格书和 3D 模型，请到网站下载(For more servo datasheet and 3D files, please go to the website to download) www.dsservo.com

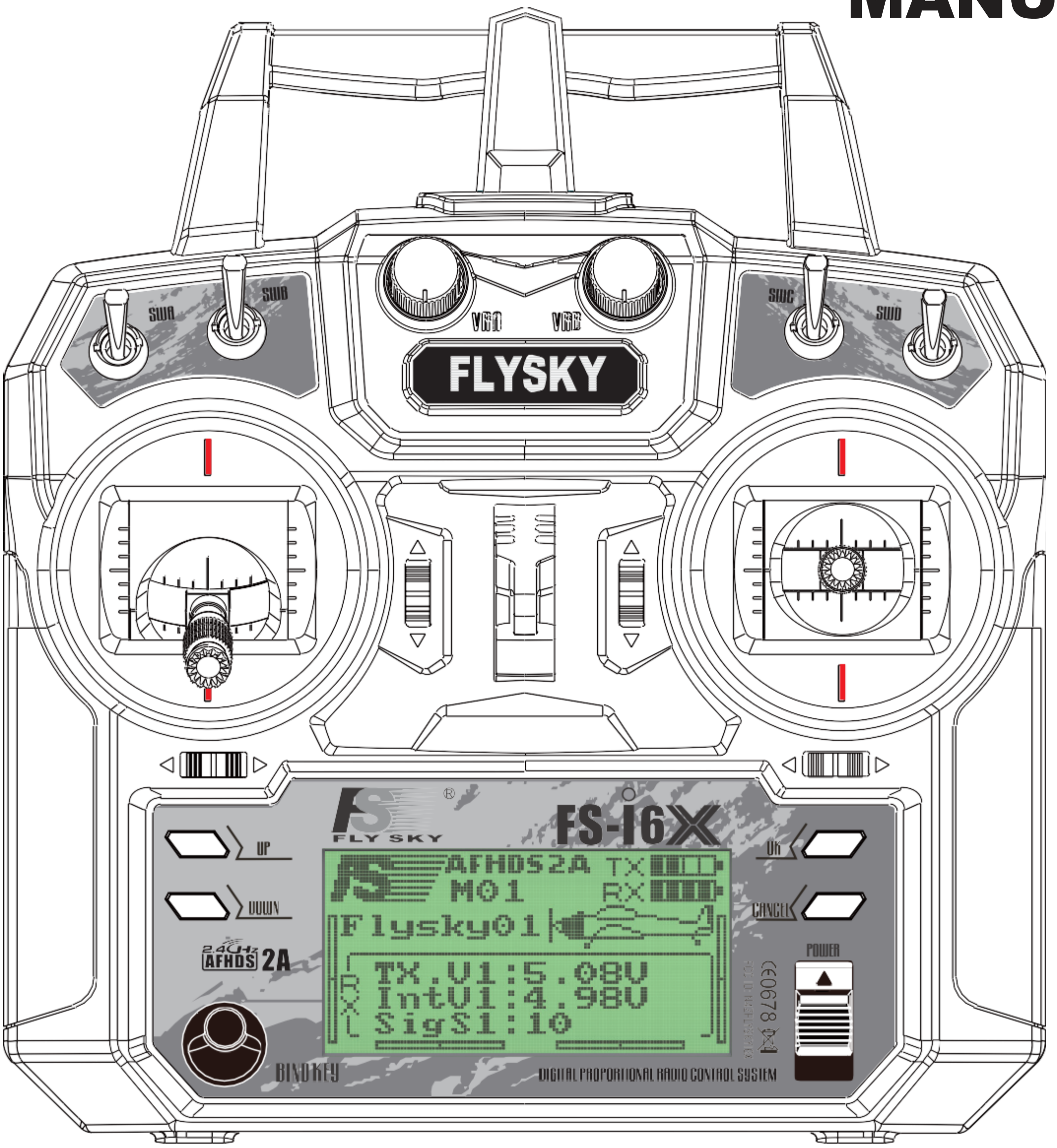
Anexo 9
Flysky FS-i6x

FLY SKY

FS-i6X

Digital Proportional Radio Control System

INSTRUCTION MANUAL



2.4GHz AFHDS 2A

Digital Proportional Radio Control System

WARNING:
This product is only for 15 years old or above

15+

CE0678
FCC ID:N4ZFLYSKYI6X

Thank you for purchasing our product, an ideal radio system for beginners or experienced users alike.

Read this manual carefully before operation in order to ensure your safety, and the safety of others or the safe operation of your system.

If you encounter any problem during use, refer to this manual first. If the problem persists, contact your local dealer or visit our service and support website for help:

http://www.flysky-cn.com

Table of Contents

1. Safety

1.1 Safety Symbols

Pay close attention to the following symbols and their meanings. Failure to follow these warnings could cause damage, injury or death.

Danger	• Not following these instructions may lead to serious injuries or death.
Warning	• Not following these instructions may lead to major injuries.
Attention	• Not following these instructions may lead to minor injuries.

1.2 Safety Guide

Prohibited | Mandatory

- **Do not use the product at night or in bad weather like rain or thunderstorm. It can cause erratic operation or loss of control.**
- **Do not use the product when visibility is limited.**
- **Do not use the product on rain or snow days. Any exposure to moisture (water or snow) may cause erratic operation or loss of control.**
- **Interference may cause loss of control. To ensure the safety of you and others, do not operate in the following places:**
  - Near any site where other radio control activity may occur
  - Near power lines or communication broadcasting antennas
  - Near people or roads
  - On any pond when passenger boats are present
- **Do not use this product when you are tired, uncomfortable, or under the influence of alcohol or drugs. Doing so may cause serious injury to yourself or others.**
- **The 2.4GHz radio band is limited to line of sight. Always keep your model in sight as a large object can block the RF signal and lead to loss of control.**
- **Never grip the transmitter antenna during operation. It significantly degrades signal quality and strength and may cause loss of control.**
- **Do not touch any part of the model that may generate heat during operation, or immediately after use. The engine, motor or speed control, may be very hot and can cause serious burns.**

- **Misuse of this product may lead to serious injury or death. To ensure the safety of you and your equipment, read this manual and follow the instructions.**
- **Make sure the product is properly installed in your model. Failure to do so may result in serious injury.**
- **Make sure to disconnect the receiver battery before turning off the transmitter. Failure to do so may lead to unintended operation and cause an accident.**
- **Ensure that all motors operate in the correct direction. If not, adjust the direction first.**
- **Make sure the model flies within a certain distance. Otherwise, it could cause loss of control.**

2. Introduction

The FS-i6X transmitter and FS-iA6 receiver constitute a 6-channel 2.4GHz AFHDS 2A digital proportional computerized R/C system. It is compatible with fixed-wing and helicopters.

2.1 System Features

The AFHDS 2A (Automatic Frequency Hopping Digital System Second Generation) developed and patented by FLYSKY is specially developed for all radio control models. Offering superior protection against interference while maintaining lower power consumption and high reliable receiver sensitivity, FLYSKY's AFHDS technology is considered to be one of the leaders in the RC market today.

Bidirectional Communication
Capable of sending and receiving data, each transmitter is capable of receiving data from temperature, altitude and many other types of sensors, servo calibration and i-BUS Support.

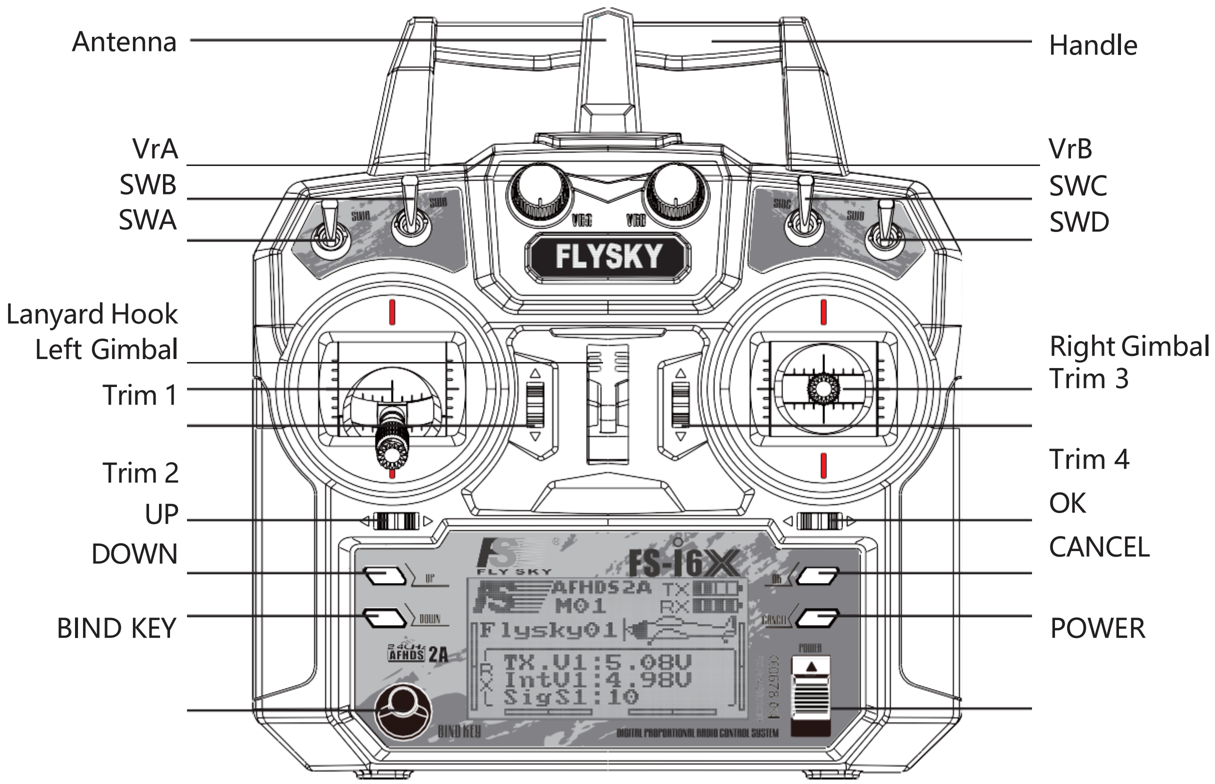
Multi-channel Hopping Frequency
This systems bandwidth ranges from 2.408GHz to 2.475GHz. This band is divided in 135 channels. Each transmitter hops between 16 channels (32 for Japanese and Korean versions) in order to reduce interference from other transmitters.

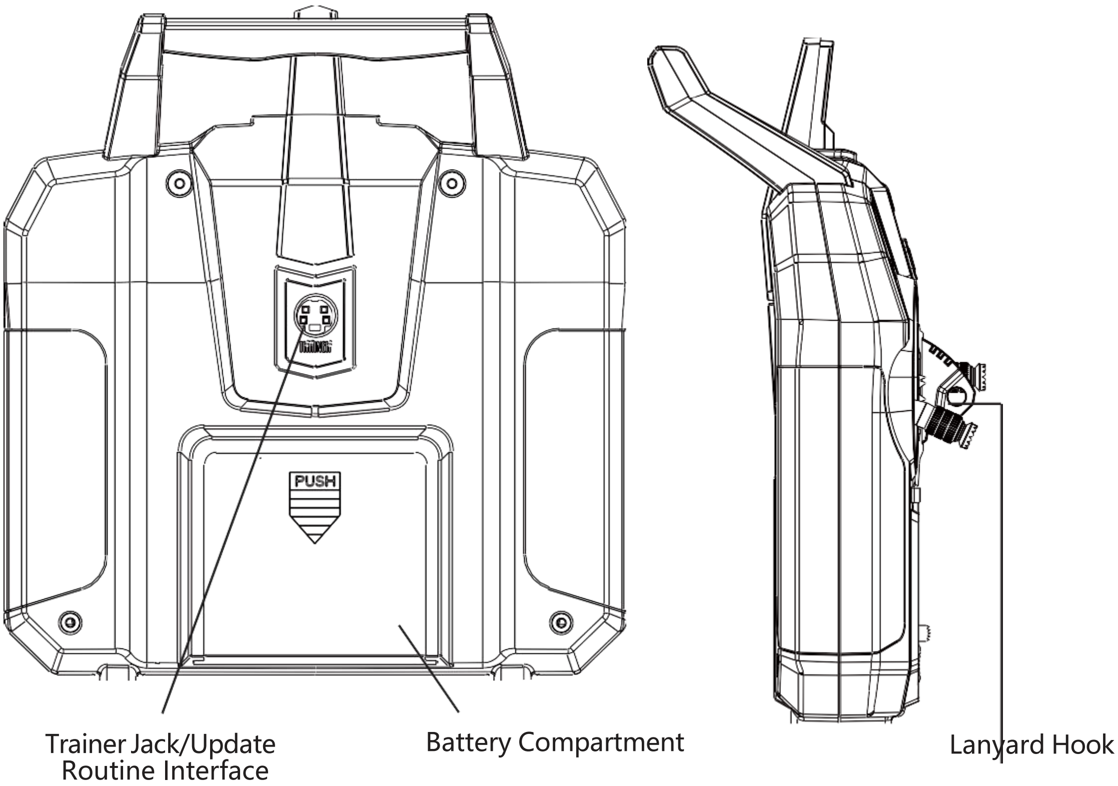
Omni-directional Gain Antenna
The high efficiency Omni-directional high gain antenna cuts down on interference, while using less power and maintaining a strong reliable connection.

Unique ID Recognition System
Each transmitter and receiver has it's own unique ID. Once the transmitter and receiver have been paired, they will only communicate with each other, preventing other systems accidentally connecting to or interfering with the systems operation.

Low Power Consumption
The system is built using highly sensitive low power consumption components, maintaining high receiver sensitivity, while consuming as little as one tenth the power of a standard FM system, dramatically extending battery life.

2.2 Transmitter Overview





2.2.1 Transmitter Antenna

⚠ **Warning**	• **For best signal quality, make sure that the antenna is at about a 90 degree angle to the model. Do not point the antenna directly at the receiver.**
⚠ **Danger**	• **Never grip the transmitter antenna during operation. It significantly degrades the RF signal quality and strength and may cause loss of control.**

2.2.2 Battery Indicator

The status indicator is used to indicate the power and status of the transmitter and receiver. If a receiver is not connected or bound to the transmitter no battery status will be displayed for the receiver.

2.2.3 Trims

There are 4 trims affecting stick functionality, one for ailerons (Channel 1), elevator (Channel 2), throttle (Channel 3) and rudder(Channel4). Each time a trim is toggled, the trim will move one step. It is possible to make quicker trim adjustments by holding the trim in the desired direction. When the trim position reaches the middle, the transmitter beeps in a higher tone.

2.3 Receiver Overview

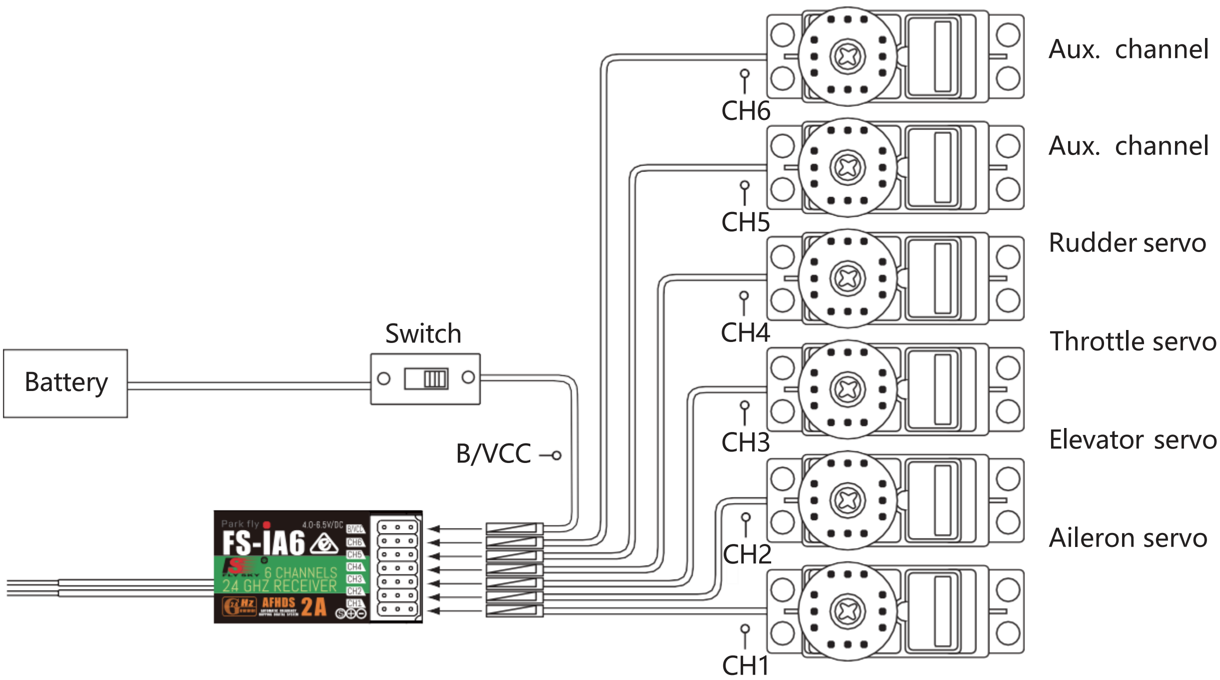
2.3.1 Receiver Antenna

⚠ **Attention**	• **For best signal quality, ensure that the receiver is mounted away from motors or metal parts.**

2.3.2 Connectors

The connectors are used to connect the parts of model and the receiver.

- CH1 to CH6: used to connect the servos, power or other parts.
- B/VCC: used to connect the bind cable for binding, and the power cable during normal operation.



3. Getting Started

Before operation, install the battery and connect the system as instructed below.

3.1 Transmitter Battery Installation

⚠ **Danger** • **Only use specified battery.**

⚠ **Danger** • **Do not open, disassemble, or attempt to repair the battery.**

⚠ **Danger** • **Do not crush/puncture the battery, or short the external contacts.**

⚠ **Danger** • **Do not expose to excessive heat or liquids.**

⚠ **Danger** • **Do not drop the battery or expose to strong shocks or vibrations.**

⚠ **Danger** • **Always store the battery in a cool, dry place.**

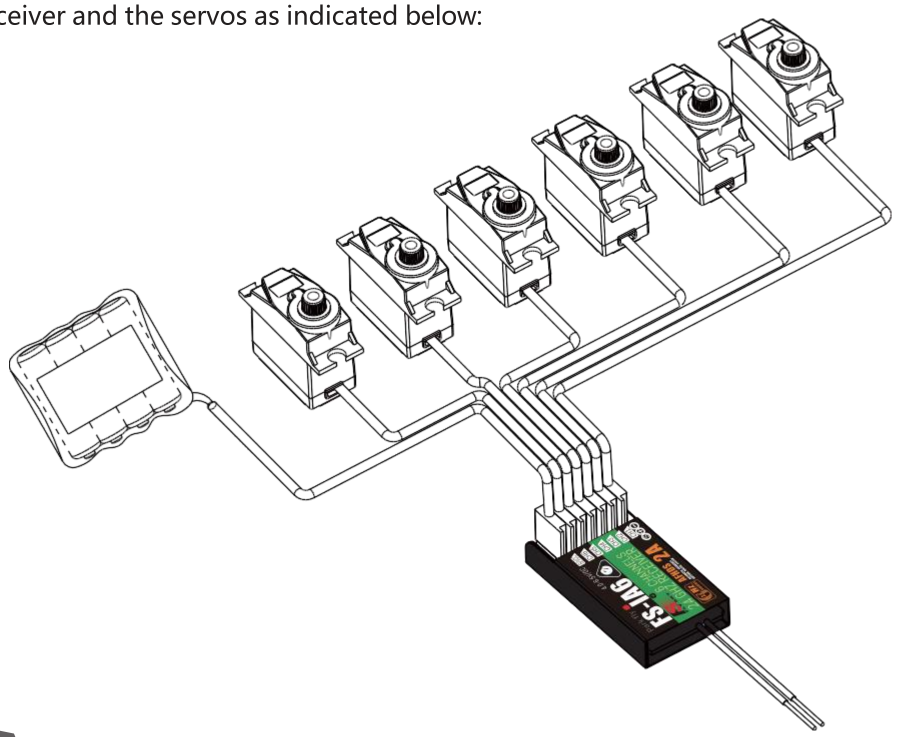
⚠ **Danger** • **Do not use the battery if damaged.**

Follow the steps to install the transmitter battery:

1. Open the battery compartment.
2. Insert 4 fully-charged AA batteries into the compartment. Make sure that the batteries makes good contact with the battery compartments' contacts, with the correct polarity.
3. Replace the battery compartment cover.

3.2 Connecting the Receiver and Servos

Connect the receiver and the servos as indicated below:



4. Operation Instructions

After setting up, follow the instructions below to operate the system.

4.1 Power On

Follow the steps below to turn on the system:

1. Check the system and make sure that:
    - The batteries are fully charged and installed properly.
    - The receiver is off and correctly installed.
2. Toggle the power switch to its upward position.
3. Connect the receiver power supply to the **B/VCC** port on the receiver.

The system is now powered on. Operate with caution,or serious injury could result.

4.2 Binding

The transmitter and receiver have been pre-bound before delivery. If you are using another transmitter or receiver, follow the steps below to bind the transmitter and receiver:

1. Connect the supplied bind cable to the **B/VCC** port on the receiver.
2. Insert power into any other port.
3. Hold the bind key while powering on the transmitter to enter bind mode.
4. Remove the power and bind cable from the receiver. Then connect the power cable to the **B/VCC** port.
5. Check the servos' operation. If anything does not work as expected, restart this procedure from the beginning.

4.3 Pre-use Check

Before operation, perform the following steps to check the system:

1. Check to make sure that all servos and motors are working as expected.
2. Check operating distance: one operator holds the transmitter, and another one moves the model away from the transmitter. Check the model and mark the distance from where the model starts to lose control.

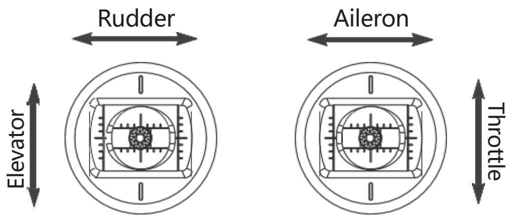
⚠ **Danger**	•	**Stop operation if any abnormal activity is observed.**
⚠ **Danger**	•	**Make sure the model does not go out of range.**
⚠ **Attention**	•	**Sources of interference may affect signal quality.**

4.4 Changing Stick Modes

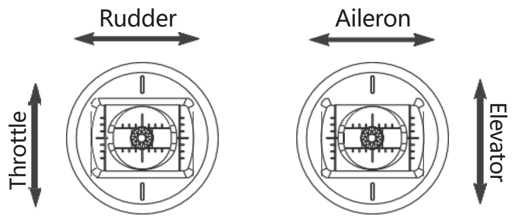
Usually the stick with the self centering feature on both axes will be mapped to the Elevator, while the other to the Throttle.

The functions of the sticks in respective modes are shown below:

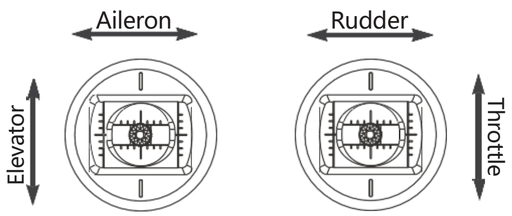
Mode 1



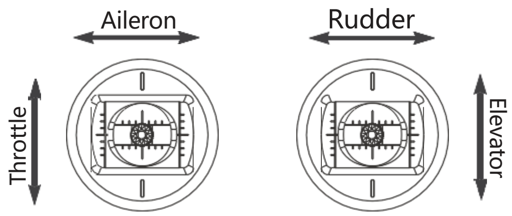
Mode 2



Mode 3

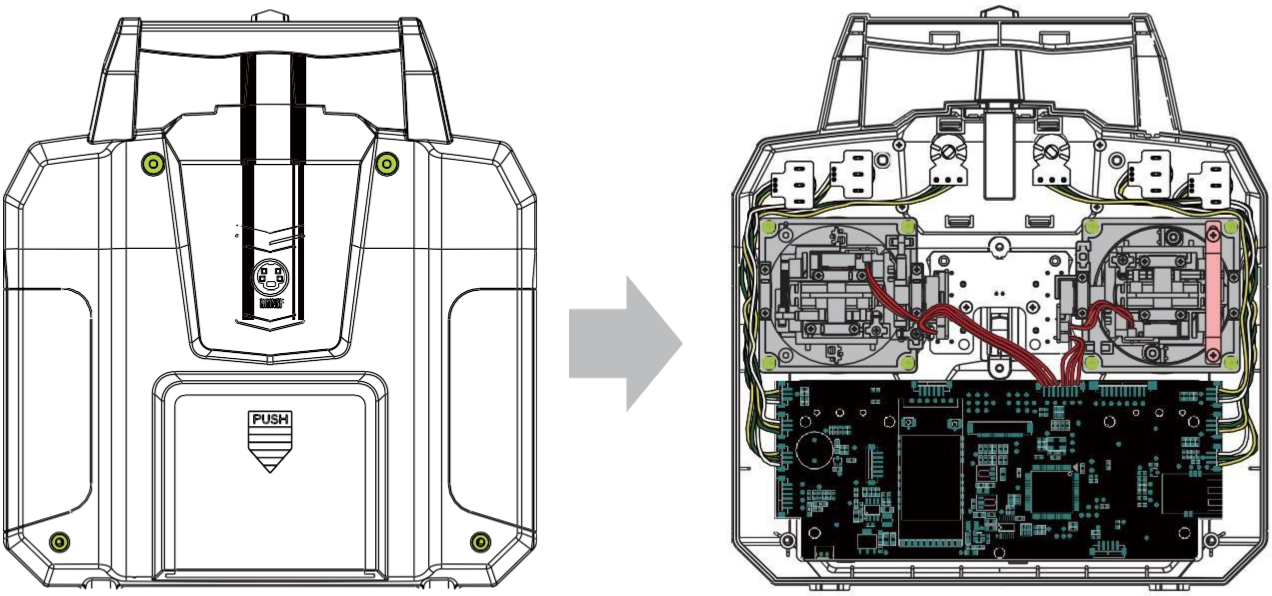


Mode 4



When switching between modes one and two it is nessesary to reverse the gimbals positions to ensure that throttle is on the correct side. To switch the sticks:

1. Take the battery out from the transmitter, Loosen the four screws that hold the rear cover shown in green on left .
2. Carefully take the back off the transmitter and disconnect the cables connected to it.
3. Unscrew the screws around the gimbals, marked in green in the picture on right.
4. Switch the gimbals to the opposite side. Make sure the gimbals have been rotated 180 degrees so that the wires are facing towards the middle of the system.
5. Reconnect the wires connecting the back to the front, then reattach the back and tighten the screws.



6. Turn the transmitter, go to Main Menu, select "System Setup" and navigate to "Sticks mode" then make sure the correct stick mode is selected. From the main menu enter "System Setup" and select "Display" and move the joystick to make sure that the channel moves in the correct direction.

4.5 Power Off

Follow the steps below to turn off the system:

1. Disconnect the receiver power.
2. Toggle the transmitter's power switch to its low position.

⚠ **Danger**

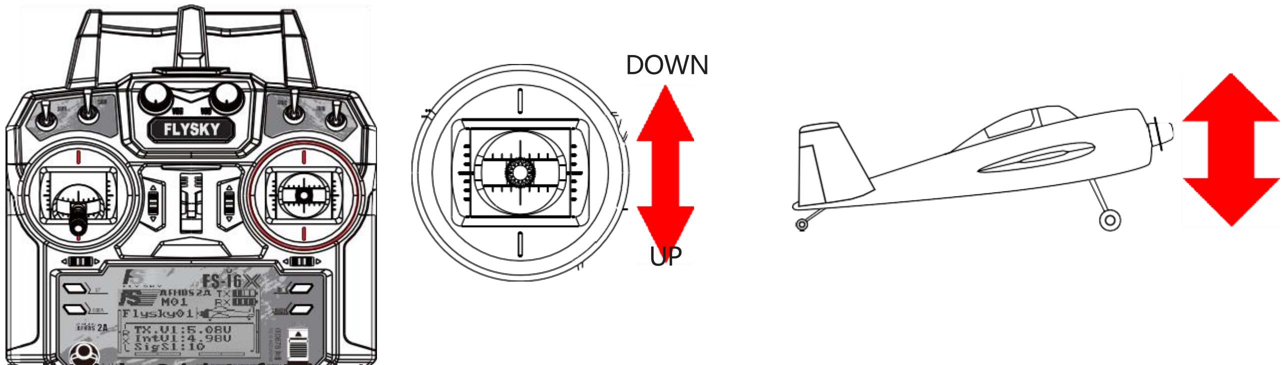
- **Make sure to disconnect the receiver power before turning off the transmitter. Failure to do so may lead to damage or serious injury.**

5. Function Descriptions

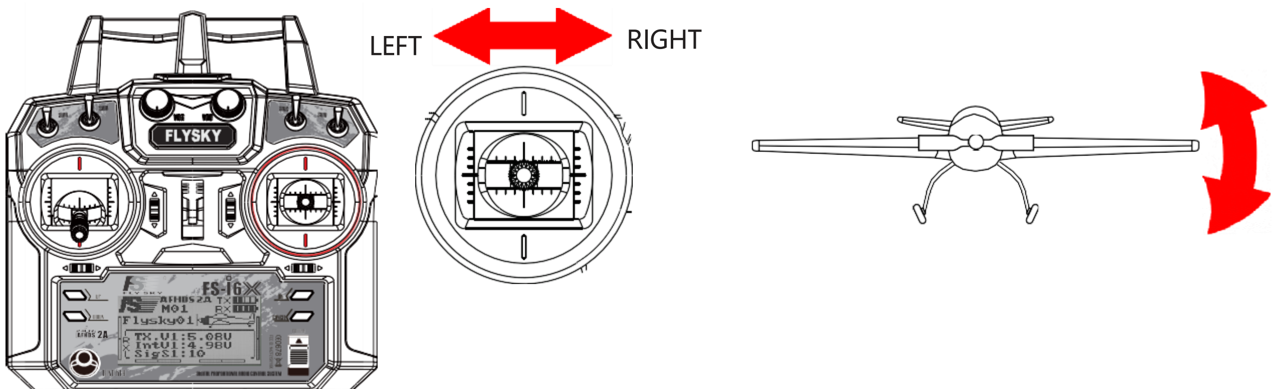
5.1 Flight Controls (Default Mode 2)

The sticks are used for controlling the aircraft, each stick has 2 functions. The right stick controls pitch and roll, the left stick controls throttle and yaw.

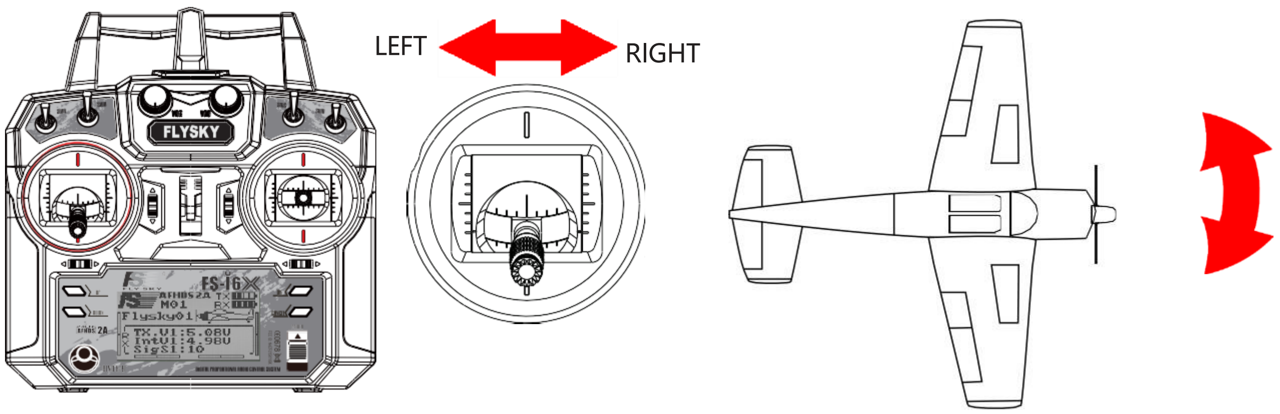
Pitch (Right Stick Up/Down)



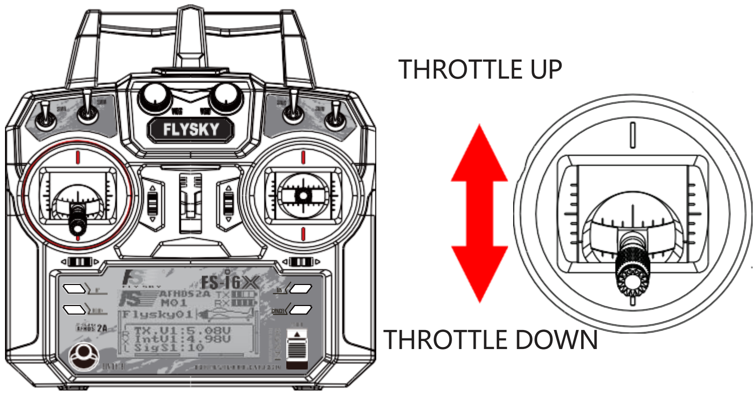
Roll (Right Stick Left/Right)

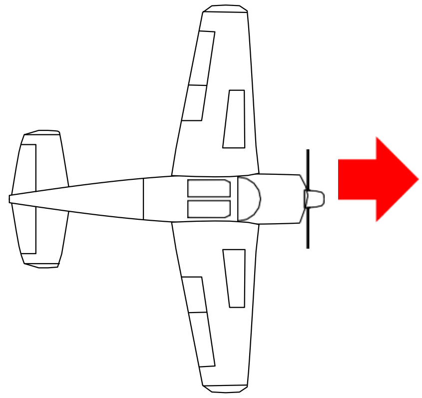


Yaw (Left Stick Left/Right)



Throttle (Left Stick Up/Down)





5.2 Reverse Function

The reverse function changes a channels direction of movement in relation to its input. For example, if a servo has to be mounted upside down due to space restrictions within a model, this function can be used to correct its movement so that it matches up with the user controls.

Setup:

1. To change between normal and press the "OK" key until the desired channel is selected, then use the "UP" and "DOWN" keys to change setting.

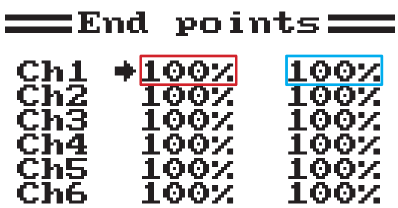
Nor = Normal, Rev = Reverse.

2. Hold the "CANCEL" key to save and return to the previous menu.
3. To return to default settings press and hold the "OK" key for 3 seconds.Press and hold the"CANCEL"key to save.

5.3 End Points

The end points function changes the range of movement available to a channel. This can be used to prevent damage to a model when a servo moves too far, potentially leading to damage to pushrods etc.

The left box is the low end point, the right box is the high end point, marked below as low being red and bule being high.

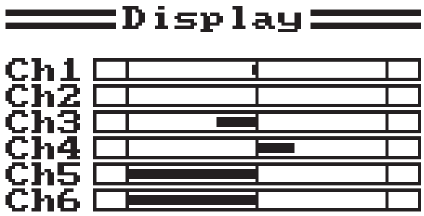


Setup:

1. Press the "OK" to change channels.
2. Move the channel using its stick or knob to select the low or high side.
3. Use the "UP" and "DOWN" keys to increase or decrease the value.
4. Hold the "CANCEL" key to save and return to the previous menu.
5. To return to default settings press and hold the "OK" key for 3 seconds.Press and hold the"CANCEL"key to save.

5.4 Display

This function displays the model's channel output in real time.



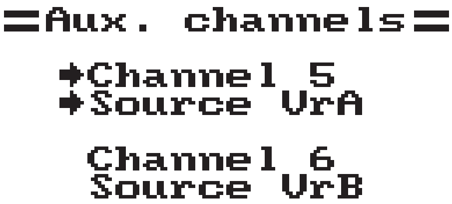
⚠ **Warning**	• **Make sure the model engine is powered off while the test function is activated. If powered on, it will rev up and cause unexpected results.**
⚠ **Danger**	• **Make sure the model does not go out of range.**

Setup:

1. Hold the "OK" key to enable channel scrub mode. In this mode the channels will sweep though their entire range of motion.
2. Press "CANCEL" key to exit.

5.5 Aux Channels

The auxiliary channels function can be used to assign switches to extra channels to control additional part of a model such as landing gear or lights.

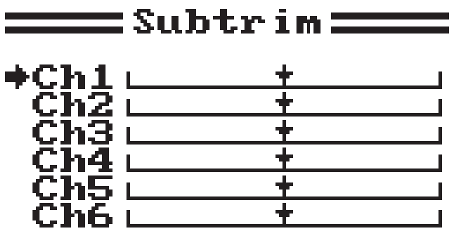


Setup:

1. Press the "OK" to change channels.
2. Use the "UP" and "DOWN" keys to select a source (Switch ,Knob or None).
3. Hold the "CANCEL" key to save and return to the previous menu.

5.6 Subtrim

Subtrim changes the center point of the channel. For example, if a models rudder is slightly out of alignment, the subtrim could be used to fix this.



Setup:

1. Press the "OK" to change channels.
2. Use the "UP" and "DOWN" keys to change the subtrim position.
3. Hold the "CANCEL" key to save and return to the previous menu.
4. To return to default settings press and hold the "OK" key for 3 seconds. until the channel returns to the center.Press and hold the "CANCEL" key to save.

5.7 Dual rate/exp.

The dual rate/exp. function only applies to channels 1、2、4.

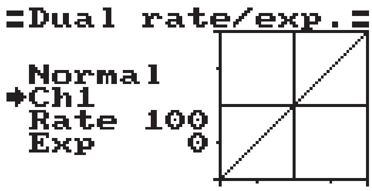
[Dual Rate]: Dual Rate reduces or increases the difference between the highest and lowest possible value, for example if applied to the rudder, (set to a throw of 10cm) before changing the settings, when you move your stick to 1/2 you would get 5cm rudder movement, if you move the stick 1/4 of the way, the rudder will move 2.5cm, so at 100% there is a direct, linear relationship of stick movement and surface movement.

If a setting of 50% is entered then moving the stick all the way in one direction will only give 1/2 of the surface movement and 1/2 stick movement will only produce 1/4 surface movement, this has the effect of reducing how responsive the rudder is when the stick is moved, effectively reducing the range of movement available to the servo. This function is usually assigned to a condition so that it can be turned on and off during flight.

[Exp. (Exponential)]: Exponential changes the relationship between stick movement and surface movement by creating a curve, when in use the stick movement and surface movement are no longer linear so the stick has a different response in different at different positions. For example this is useful when needing less reaction during a take-off but more reaction when in the air.

Setup:

1. Press the "OK" to change between settings.
2. Use the "UP" and "DOWN" keys to change the channel/ rate/exp depending on the selected setting.
3. Hold the "CANCEL" key to save and return to the previous menu.
4. To return a setting to default, press and hold the "OK" key for 3 seconds. Press and hold the "CANCEL" key to save.



5.8 Throttle Curve

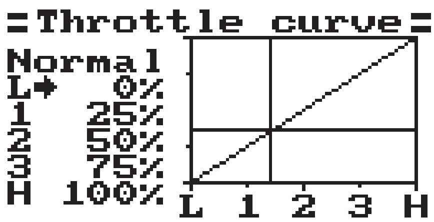
This function enables the user to adjust the ratio between stick and servo movement using a linear line or non-linear curves.

This is useful when wanting to change how the throttle reacts at between different stick positions, for example having a smaller throttle change when the stick is between 0-30%, then a larger throttle change between 30% and 100%. If your models throttle is not linear, it is also possible to use this function to create a more linear movement.

This function uses 5 points to change the throttle curve, L being the low and H being the high.

Setup:

1. Press the "OK" to change between points.
2. Use the "UP" and "DOWN" keys to change point position.
3. Hold the "CANCEL" key to save and return to the previous menu.
4. To return a setting to default, press and hold the "OK" key for 3 seconds. Press and hold the "CANCEL" key to save.



5.9 Mixes

This function is used to create a mix between channels. For example if at low throttle some automated flap movement was desired then it is possible to create a mix to do this. This system can have up to 3 different mixes.

Setup:

1. Use the "UP" and "DOWN" keys to select a mix.
2. Use the "OK" key to change between settings.
3. Select a master channel, this channel will control the slave channel.
4. Select a slave channel to be controlled by the master.

5. Set the positive and negative mix, this setting controls how much the slave channel will move in relation to the masters movement, if set to 50% the slave will move half the amount of the master.
6. Set the offset, the offset changes the center of the slave channel in relation to the master.
7. Hold the "CANCEL" key to save and return to the previous menu.
8. To return a setting to default, press and hold the "OK" key for 3 seconds. Press and hold the "CANCEL" key to save.

```
Mixes
➜Mix#1
 Mix is      Off
 Master      Ch1
 Slave       Ch2
 Pos. mix    50%
 Neg. mix    50%
 Offset       0%
```

5.10 Elevon

The elevon function is used for planes that combine the elevons an ailerons together.

Setup:

1. Use the "UP" and "DOWN" to turn the function on and off.
2. Use the "OK" key to change between settings.
3. Use the "UP" and "DOWN" keys to change the percentage.
4. To return a setting to default, press and hold the "OK" key for 3 seconds. Press and hold the "CANCEL" key to save.

```
Elevon
➜Elevon   On
  Ch1 50%
  Ch2 50%
Ch1 <= Ch2+Ch1
Ch2 <= Ch2-Ch1
```

5.11 V Tail

The V Tail function is used for planes that use a v tail configuration.

Setup:

1. Use the "UP" and "DOWN" to turn the function on and off.
2. Use the "OK" key to change between settings.
3. Use the "UP" and "DOWN" keys to change the percentage.
4. To return a setting to default, press and hold the "OK" key for 3 seconds. Press and hold the "CANCEL" key to save.

```
V tail
➜V tail   On
  Ch2 50%
  Ch4 50%
Ch2 <= Ch2-Ch4
Ch4 <= Ch2+Ch4
```

5.12 Assign Switches

This function enables you to assign switches to Fly mode, Idel mode, and Throttle hold.

Setup:

1. Use the "OK" key to change between settings.
2. Use the "UP" and "DOWN" keys to change switch assignment.
3. Press and hold the "CANCEL" key to save.

- Switches must be turned on in the [7.13 Aux Switches] function to be avaliable for assignment.

```
:Assign switches:
➜Fly mode    SwA
   Normal
 Idle mode   SwB
   Normal
 Thro. hold  SwD
   Off
```

5.13 Throttle Hold

This function is used with gas powered models in order to stop stalls when not in use.

Setup:

1. Use the "OK" key to change between settings.
2. Use the "UP" and "DOWN" keys to turn the function on or off and increase and decrease the hold percentage.
3. To return a setting to default, press and hold the "OK" key for 3 seconds. Press and hold the "CANCEL" key to save.

- This function will not work unless assigned to a switch. The switch can be used to enable or disable the function. Please refer to [5.12 Assign Switches] for more details on how to assign a switch to a function.
- Switches must be turned on in the [7.13 Aux Switches] function to be avaliable for assignment.

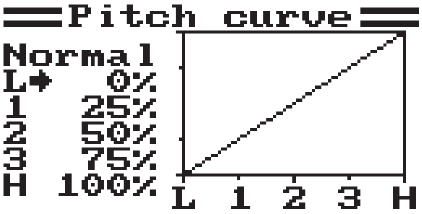
```
=Throttle hold=

 Hold   On
 Value  50%

    Inactive
```

Note • **This function must be assigned to a switch in the Switches assign funciton.**

6. Helicopter Functions

6.1 Pitch Curve

The pitch curve function is for programing the response of the helicopters blades collective pitch, which controls the amount of lift the helicopter has. This functions output is shown on the graph, with points along the bottom (L,1,2,3,H), and collective pitch up the side (0-100%). When the throttle stick is moved its position will be shown in real time.



Setup:

1. Use the "OK" key to cycle between points.
2. Use the "UP" and "DOWN" keys to change the percentage (All changes are shown in real time in the graph)
3. To return to default settings press and hold the "OK" key for 3 seconds, press and hold the "CANCEL" key to save.

6.2 Swashplate Mix

The swashplate mix function sets the relative movement between each servo controlling movement of the swash plate controlling aileron, elevator and pitch.

Setup:

1. Press the "OK" key to cycle through aileron, elevator and pitch.
2. Use the "UP" and 'DOWN" keys to change the percentage.
3. Press and hold the "CANCEL" key to save and exit.
4. To return to default settings press and hold the "OK" key until the currently selected perameter returns to 50%,and press and hold the "CANCEL" key to save.

6.3 Gyroscope

The gyroscope function uses a gyroscope to correct for torque produced by changes in engine speed, pitch and wind etc., which can cause issues with yaw control. If not corrected each of these variables could cause the RC helicopter to spin, sometimes quite violently.

This function has 2 settings, Gryro (On/Off) and Value (%). The Mode shows the state of the Idle up function (This function needs to be assigned to a switch).

Setup:

1. Use the "OK" key to cycle between "Gyro" and "Value", select "Gyro" and press the "UP" or "DOWN" key to toggle on or off.
2. Select "Value" and up the "UP" and "DOWN" arrow keys to change the percentage.
3. To return to default settings press and hold the "OK" key until the currently selected perameter returns to 50%, and press and hold the "CANCEL" key to save.

7. System

7.1 Model Select

Use this function to select stored models, use the "UP" and "DOWN" keys to choose a model and press and hold the "CANCEL" key to save and exit. The system can store up to 20 models.

7.2 Model Name

This function renames the currently selected model.

Setup:

1. Use the "UP" and "DOWN" keys to select a letter or number, then press the "OK" key to confirm.
2. To save press and hold the "CANCEL" key.

To return to default press and hold the "OK" key for 3 seconds,press and hold the "CANCLE" key to save.

7.3 Type Select

This function changes the type of the currently selected model, including airplane and helicopter with different types of swashplates.

Helicopter swash plate type	Available functions
Swash 140°	Pitch Curve，Swashplate Mix，Gyroscope
Swash 120°	Pitch Curve，Swashplate Mix，Gyroscope
Swash 90°	Pitch Curve，Swashplate Mix，Gyroscope
Variable pitch	Pitch Curve，Gyroscope
Fixed pitch	Gyroscope

Setup:

1. To change the model type press the "UP" and "DOWN" keys to select the model type, then press and hold the "CANCEL" key to save and exit.

7.4 Model Copy

This function copies the one model to another model slot.

Setup:

1. Use the "UP" and "DOWN" keys to select the model you want to copy.
2. Use the "OK" key to and use the "UP" and "DOWN" keys to select the slot to copy the model.
3. Press and hold the "OK" key to confirm, the system will display a prompt asking "Are you sure", use the "UP' or "DOWN" key to select Yes and press "OK" again to confirm.

7.5 Model Reset

This function resets the current model to the default settings.

Setup:

1. Use the "UP' annd "DOWN" keys to select a model. Press the "OK' key to confirm,.
2. The system will display a prompt asking "Are you sure", use the "UP' or "DOWN" key to select Yes and press "OK" key again to confirm.

7.6 Trainer Mode

Trainer mode is used to take control of a slave system when a switch is in the off position. This function will only work when two systems are linked via the trainer lead.

Setup (This function must be assigned to a switch and will only be inactive when the switch is on):

1. Use the "UP" and "DOWN" keys turn the function on and off.
2. Use the "OK" key to and use the "UP" and "DOWN" keys to select a switch.
3. Press and hold the "CANCEL" key to save and exit.

7.7 Student Mode

Student mode is used when another system is connected as a master (Trainer), when this mode is active all settings will be bypassed and the system will only function through the master.

Setup:

1. To enable the function press "OK" then select Yes, The system will return to the previous menu.
2. To exit student mode repeat this process.

7.8 Sticks Mode

There are 4 available stick modes, each stick mode changes the stick functions.

For example when using stick mode 2 the left stick controls throttle on the vertical axis and rudder in the horizontal axis, however in stick mode 3, the vertical axis controls elevator and the horizontal axis controls aileron. These modes are largely down to personal preference.

Setup:

1. Use the "UP" and "DOWN" keys to select a stick mode,.
2. Press and hold the "CANCEL" key to save and exit.
3. To return to default settings press and hold the "OK" key for 3 seconds,press and hold the "CANCLE" key to save.

7.9 LCD Brightness

Setup:

1. Use the "UP" and "DOWN" keys.
2. Press and hold "CANCEL" to save and exit.
3. To return to default settings press and hold the "OK" key for 3 seconds,press and hold the "CANCLE" key to save.

7.10 Firmware Ver.

This function displays the current firmware version.

7.11 Firmware Update

This function updates the firmware using a USB to PS/2 connection lead.

Setup:

1. First download the update from our website,http://www.flysky-cn.com.
2. Connect the system to the computer via the supplied cable and press "OK" while in this function.
3. Wait for windows to recognise the system.

4. Then open the update on the computer and press update.
5. Once the update is finished cycle the system power.

7.12 Factory Reset

This function resets the entire system back to its factory settings.

To reset press "OK",then use the "UP" and "DOWN" keys to select Yes and press 'OK" again.

7.13 Aux Switches

This function both activates and deactivates switches/knobs as well as changing the amount of active channels the system will use. This is usually done when a new switch or knob has been.

Setup:

1. Use the "OK" key to cycle through the selection of switches and knobs.
2. Use the "UP" and "DOWN" keys to turn the selected switch/knob on or off.
3. Keep pressing the "OK" key until "Ch" is selected.
4. Use the "UP" and "DOWN" keys to change the amount of active channels to match your current configuration.
5. To return to default settings press and hold the "OK" key for 3 seconds,press and hold the "CANCLE" key to save.

8 RX Setup

8.1 RF Standard

This menu allows you to change the communication protocol for the transmitter. The available protocols are:

RF Protocol	Receiver
AFHDS	R9B,R6B,R6C,GR3E,GR3F
AFHDS 2A	A3, A6,X6, iA4B, iA6, iA6B, iA10, iA10B

To Switching Between AFHDS 2A and AFHDS:

1. First navagate to the system menu by pressing and holding the "OK" key until the main menu opens, select "System Setup" by pressing the "OK" key again.
2. Use the "DOWN" key to navagate to "RX setup" and press the "OK" key again to enter, then press the "OK" key one more time to select RF Standard.
3. The system will display a prompt asking if you are sure, use the "UP" or "DOWN" key to select yes and press "OK".
4. Then use the "UP" or "DOWN" key to select the desired RF standard and press and hold the "CANCEL" key for until the system returns to the previous menu to save.
5. Use the "UP" and "DOWN" keys to select a mode then press and hold the "CANCEL" key to save and exit.

8.2 RX Battery

This function is used to change the battery monitor settings. This function can be switch to an external or internal sensor.

There are 4 settings:

[External sensor/ Internal Sensor]: The system has its own voltage sensor however it is possible to change to an external sensor.

[Low]: Sets the low battery voltage, see your batteries user manual to set this setting.

[Alarm]: Sets the voltage level at which the system will allert the user if the battery gets too low.

[High]: Sets the voltage for the battery if it is full.

⚠ **Note**

- **These settings affect how the system shows battery levels, if the high and low are incorrect the systems battery display will not be reliable.**

8.3 Failsafe

This function is used to protect the models and users if the receiver loses signal and therefore is no longer controllable.

All channels are listed in the failsafe menu. [**Off**] means that in case of a loss of signal, the corresponding servo will keep its last received position. If it displays a percentage, the servo will instead move to the selected position.

Setup:

1. Use the "UP" and "DOWN" to choose a channel and press "OK" to enter its failsafe settings.
2. Use the "UP" and "DOWN" to turn the failsafe on or off.
3. Move the channels control surface to the desired position and hold the "CANCEL" key to confirm and exit.

4. To return to default settings press and hold the "OK" key for 3 seconds,use the "UP" and "DOWN" keys to select Yes. press and hold the "CANCLE" key to save.

You can set the failsafe position for all channels with the[All channels] button at once. To do so,

1. Move all your channels to the desired position.
2. Select [All channels].

- Once the failsafe has been set, a percentage will be displayed.

8.4 Sensors List

This function displays all connected sensors and thier outputs.

8.5 Choose Sensors

This function changes which sensors will be displayed on the home screen. The home screen can display up to 3 sensors.

Setup:

1. To add a sensor to the home screen, use the "OK" key to change sensor slot, then use the "UP" and "DWON" keys to select a sensor.
2. To return to default settings press and hold the "OK" key for 3 seconds, Press and hold the "CANCEL" key to save and exit.

8.6 Speed and Distance

This function is for setting up speed and distance sensors.

Speed Sensor

If a sensors is connected, us the "UP" and "DOWN" arrow keys to select the desired senor then press and hold the "CANCEL" key to save.

Rotation Length

Measure the distance from the center of the prop to the distance sensor. Then use the "UP" and "DOWN" arrow keys to enter the length. Press and hold the "CANCEL" key to save.

Reset Odometer 1 + 2

These settings return the odometer to 0. To reset select one, odometer 1 or 2, then press "OK". The system will display a prompt, select yes.

Reset odometer 1

Resets odometer 1 to 0. Odometer 1 records the distance traveled during a session. Note that restarting the system will also reset odometer 1.

Reset odometer 2

Resets odometer 2 to 0. Odometer 2 records the total distance traveled since last reset. This means that the distance over several sessions will be added together.

8.7 ASL Pressure

The set ASL (Above Sea Level) function is used to calibrate an altitude sensor. When an altitude sensor is connected, change the [Air pressure] setting until the altitude is at 0m.

Setup:

1. Make sure that your TX and RX are bound and turned on.
2. Set your model on the ground.
3. Use the "UP" and "DOWN" keys to change the hPa value. If the system is showing a positive altitude, reduce the hPa value until the altitude reaches 0m. If the system is showing a negative altitude increase the hPa value until it reaches 0m.
4. To return to default settings press and hold the "OK" key for 3 seconds,press and hold the "CANCLE" key to save.

Note: Make sure that your model is at ground level during this process.

8.8 Output Mode

PPM is capable of transferring all channels through one physical output. When [**RX PPM output**] is checked:

- When [PWM] is selected the receiver will output channels 1-6 via channel 1-6.
- When [PPM] is selected the receiver will output a standard PPM signal via the PPM interface.

To turn the function on press the "UP" or "DOWN" keys to turn the function on then press and hold the "CANCEL" key to save and exit.

8.9 i-BUS Setup

This function is used to set up the i-BUS module. The i-BUS module can be used to add servos to your model that may be too far away from the receiver.

Setup:

1. Use the "UP" and "DOWN" keys to choose a channel and press "OK".
2. Press the button on the i-BUS module that corresponds to the desired output, the system will then return to the previous menu.
3. After setting up the desired channels press and hold the "CANCEL" key to save and exit.

8.10 Servos Freq

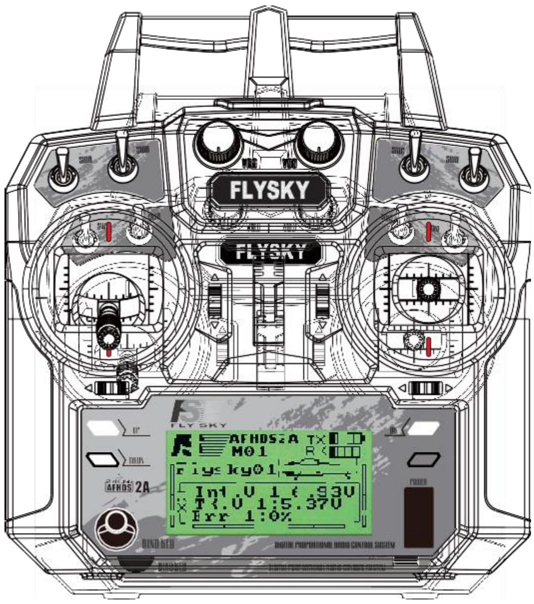
This function sets the frequency that the receiver outputs to the servos. Check your servos usermanual to find the correct setting.

9. System Customization

The FS-i6X's switches and knobs can be moved to other channels. Or if using receivers with more channels, the system can be expanded with extra switches or knobs.

By default, from left to right, the knobs are channels 5 and 6, the switches are7, 8, 9 and 10.

FS-A6/FS-iA6B/FS-iA6	6CH
FS-iA10B	6-10CH



9.1 Switching Channel Assignments

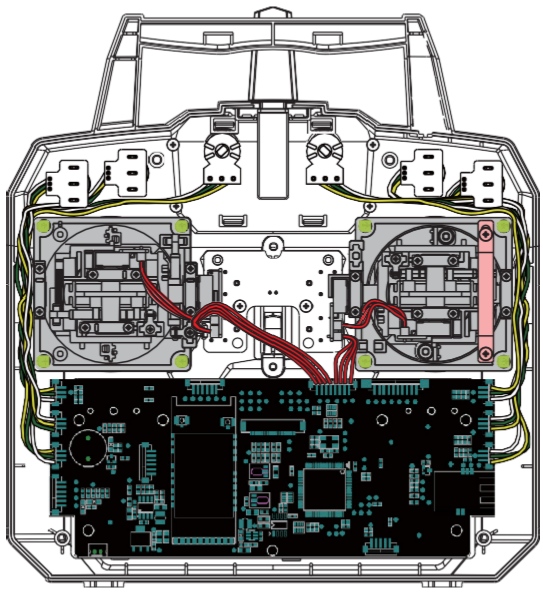
To change a switch or knobs channel, the system must be taken apart. The first step is taking the back cover off.

1. Remove any batteries from the system and replace the battery cover.
2. Remove the screws marked in green.

Note	• **Make sure the screw driver you are using is not too big or too small. Failure to do so could damage the head of the screw.**

3. Carefully pry the front and back apart, this may take some force.

Note	• **Don't pull the pieces too far apart, doing so could damage cables attacking the front and back together.**



4. Carefully disconnect the cables connecting the front to the back.

Note	• **Make sure you keep the screws in a safe place.**
Note	• **Make sure that the wires are fitted along beside each of the gimbals as show right.**
Note	• **Make sure that all switches are installed with the correct orientation shown right.**

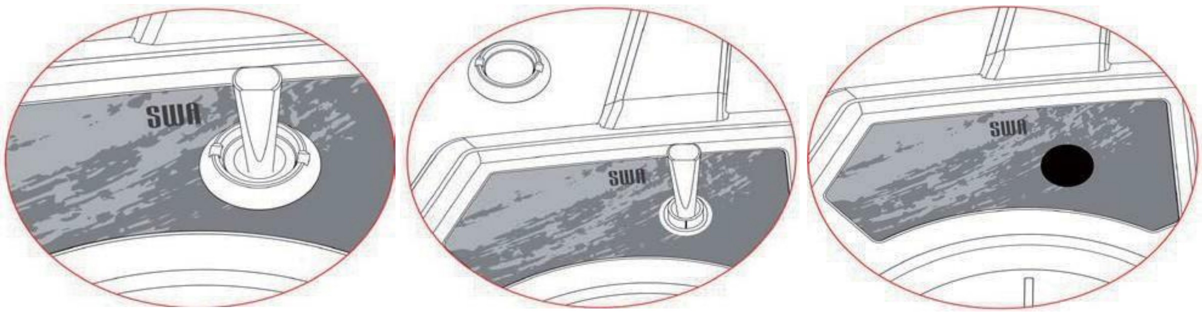
5. On the circuit board each channel is labled, making it easy to find the correct channel. Follow the cables leading from each connector to identify which switch or knob goes to each channel.
6. Carefully remove the desired connectors from the board.

⚠ Note • **Do not pull on the wires themselves, doing so may damage the connector or wire.**

7. Replace the desired switch/knob connectors into the corrsponding channel slot.

Setup:

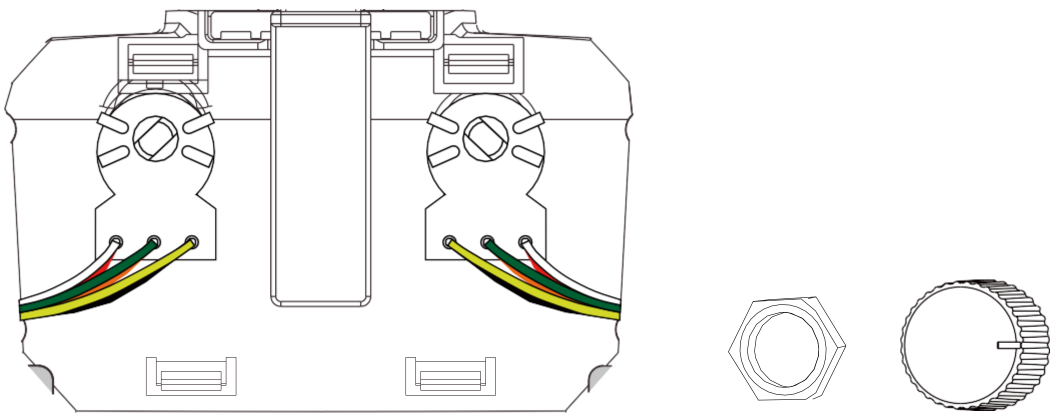
1. Take the transmitter apart following the above instructions.
2. Remove the toggles connector from the circuit board.
3. Unscrew the plate holding the toggle in place on the front of the transmitter.



Removing a knob:

Remove the pot cover by slowly pulling on it, it should come off without much effort.

1. Remove the 4 screws located on the back of the system and remove the back cover.
2. Follow the knobs wire and disconnect it form the board.
3. Gently remove the knob cap by pulling it up.
4. Remove the nut holding the knob in place.
5. Remove knob.



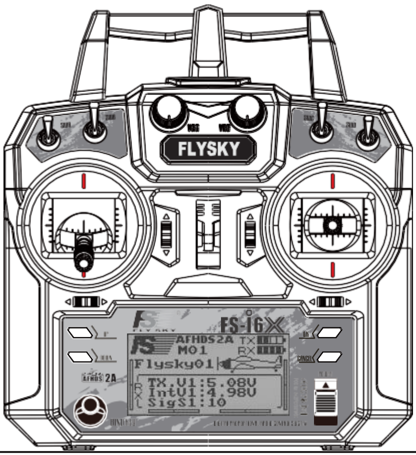
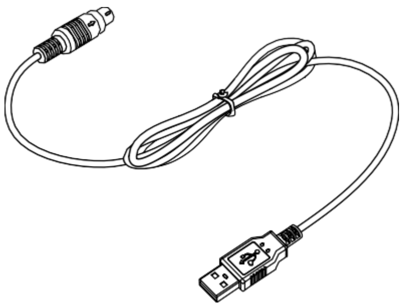
8. Put the back cover back in place, and squeeze the handle until the two pieces click together.
9. Replace the cover screws.

9.2 Activate Switch/Knob

Open the system menu, navigate to "Aux Switches" and press the "OK" key. Use the "OK" key to change switch/knob, then use the "UP" or "DOWN" keys to turn the switch on.

The switch will now be available in the "Assign Switches" menu.

10. Package Contents

4-10 Channel 2.4GHz Transmitter (FS-i6X)	
2.4GHz Receiver (FS-iA6 (6 CH))	
User Manual （CD)	
PS/2 to USB Update Cable	

11 Product Specification

11.1 Transmitter specification (FS-i6X)

Channels	6-10 (Default 6)
Model Type	Fixed-Wing/Glider/HElicopter
RF Range	2.408-2.475GHz
RF power	< 20dBm
RF Channel	135
Bandwidth	500KHz
2.4GHz System	AFHDS 2A / AFDHS
Modulation Type	GFSK
Stick Resolution	4096
Low Voltage Warning	< 4.2V
DSC port	PS/2 Port PPM
Chargeable	No
Antenna Length	26mm(Dual Antenna)
Weight	392 g
Power	6V DC 1.5AA*4
Display	STNTransflective Display ,LCD128x64 Lattice，VA 73x 39mm , LCD with white backlight
Size	174x89x190mm
On-line Update	Yes
Color	Black
Certificate	CE0678，FCC ID:N4ZFLYSKYI6X

11.2 Receiver specification (FS-iA6)

Channels	6
Model Type	Fixed-Wing/Glider/HElicopter
RF Range	2.408-2.475GHz
RF Channel	135
RF Receiver Sensitivity	- 105dBm
Bandwidth	500KHz
2.4GHz System	AFHDS 2A
Modulation Type	GFSK
Power	4.0~6.5V DC
Antenna Length	26mm(Dual Antenna)
Weight	7g
Size	40.4x21.1x15mm
i-BUS Port	No
Data Acquisition Port	No
Color	Black
Certificate	CE0678，FCC ID:N4ZFLYSKYIA6

Appendix 1 FCC Statement

This equipment has been tested and found to comply with the limits for a Class B digital device pursuant to part 15 of theFCC rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or televison reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

To assure continued compliance, any changes or modifications not expressly approved by the party responsible for compliance could void the user' s authority to operate this equipment. (Example use only shielded interface cables when connecting to computer or peripheral devices).

This equipment complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions:

(1) This device may not cause harmful interference, and

(2) This device must accept any interference received, including interference that may cause undesired operation.

Caution!

The manufacturer is not responsible for any radio or TV interference caused by unauthorized modifications to this equipment. Such modifications could void the user authority to operate the equipment.

Digital Proportional Radio Control System

CE 0678 FCC ID:N4ZFLYSKYI6X

http://www.flysky-cn.com

Edition: 2016-08-17

Anexo 1
Sensor HC-SR04

Ultrasonic Ranging Module HC - SR04

Product features:

Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact measurement function, the ranging accuracy can reach to 3mm. The modules includes ultrasonic transmitters, receiver and control circuit. The basic principle of work:
(1) Using IO trigger for at least 10us high level signal,
(2) The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
(3) IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning.
Test distance = (high level time×velocity of sound (340M/S) / 2,

Wire connecting direct as following:

- 5V Supply
- Trigger Pulse Input
- Echo Pulse Output
- 0V Ground

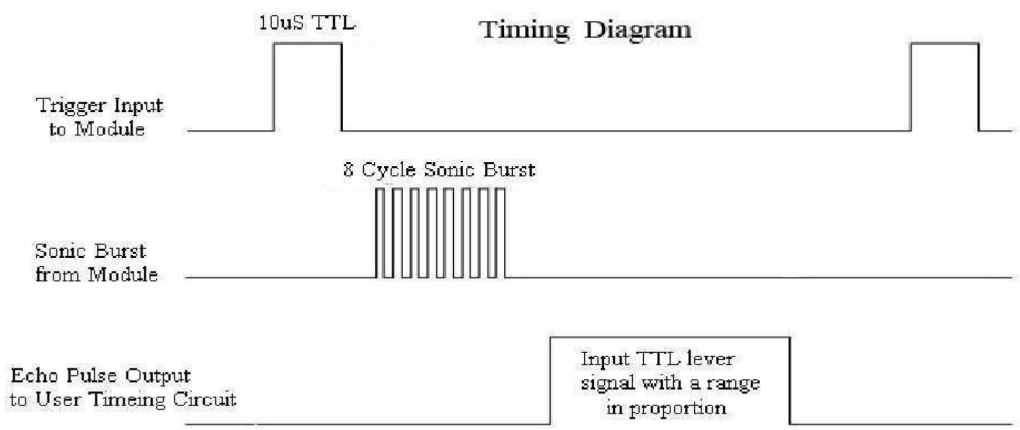
Electric Parameter

Working Voltage	**DC 5 V**
Working Current	**15mA**
Working Frequency	**40Hz**
Max Range	**4m**
Min Range	**2cm**
MeasuringAngle	**15 degree**
Trigger Input Signal	**10uS TTL pulse**
Echo Output Signal	**Input TTL lever signal and the range in proportion**
Dimension	**45*20*15mm**

Vcc Trig Echo GND

Timing diagram

The Timing diagram is shown below. You only need to supply a short 10uS pulse to the trigger input to start the ranging, and then the module will send out an 8 cycle burst of ultrasound at 40 kHz and raise its echo. The Echo is a distance object that is pulse width and the range in proportion .You can calculate the range through the time interval between sending trigger signal and receiving echo signal. Formula: uS / 58 = centimeters or uS / 148 =inch; or: the range = high level time * velocity (340M/S) / 2; we suggest to use over 60ms measurement cycle, in order to prevent trigger signal to the echo signal.



Attention:

- The module is not suggested to connect directly to electric, if connected electric, the GND terminal should be connected the module first, otherwise, it will affect the normal work of the module.
- When tested objects, the range of area is not less than 0.5 square meters and the plane requests as smooth as possible, otherwise ,it will affect the results of measuring.

Anexo 11
Sensor ultrasonidos DFROBOT

Introduction

URM37 VS.O is a powerful ultrasonic sensor mudule with bulit-in temperature compensation to ensure accurate distance measurement in the scene of temperature-changing applications. It has rich interface and offers various output: analog output, switch, serial (TTL and RS232 level optional), PWM and so on. The module can be used to measure the rotation angle of the servo. Connected with an externa! servo, it changes into a spatial ultrasonic scanner. URM37 has been on the market for many years and plays an important role in various fields, and we are constantly optimizing and improving it. The mechanical size, pin interface and communication commands of this version (VS.O) are compatible with older versions. Based on the old version, the following improvements have been made:

- The ranae has been increased from 5-SOOcm to 2-800cm

- The ranging performance is very stable at the voltage range of 3.3V~S.5V.

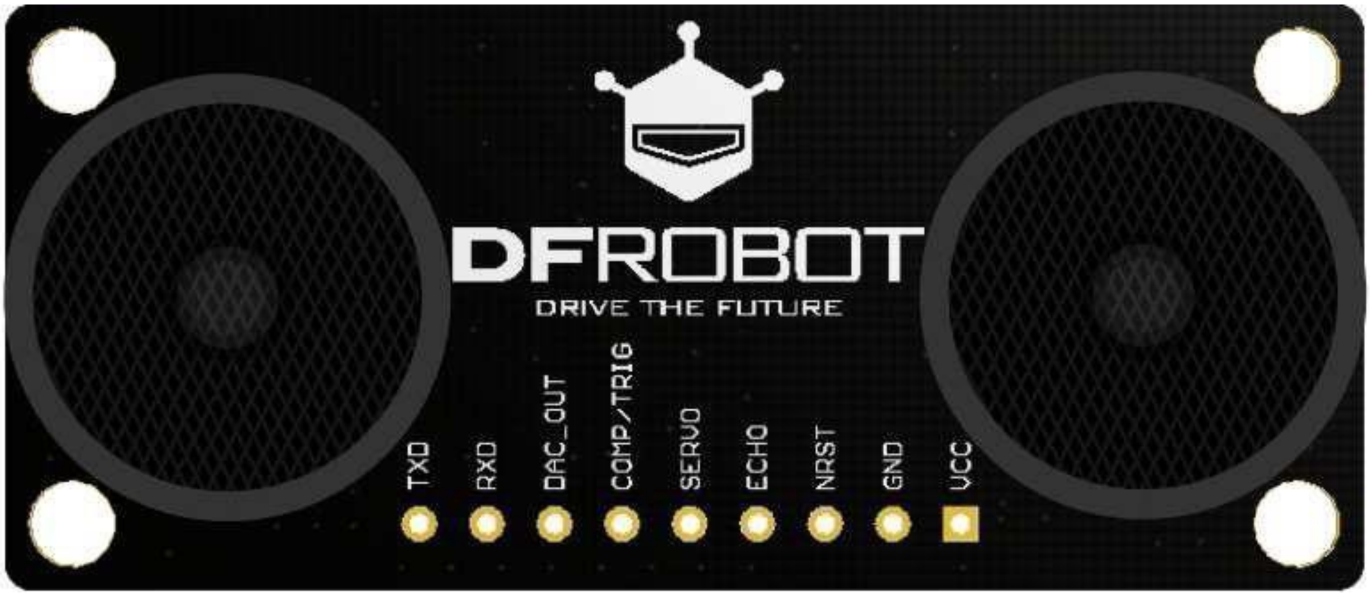
Specification

- Operating Voltage: 3.3V ~ 5.SV
- Operating Current: 20mA
- Working temperature: -10ºC ~ 70º(
- Detecting range: 2cm-800cm(ultimate range 1000cm)
- Resolution: 1cm
- Accuracy:1%
- Measuring Period: < 100ms (Max)
- Dimensions: 22mm x 51 mm
- Weight: about 25g

Technical Descriptions

- Out of the use of a better ranging method, the measurement distance is further and more stable. if there is a need for customization, please contact the company.
- The module uses RS232 serial port for higher reliability, and the data can be collected through the computer serial port, which is very convenient to write communication programs.
- Serial level selected from the skipped stitches to button, user can easily select RS232 or TTL-level output level output by pressing the settings(after reboot).
- The measured distance can be output via PWM, which eases the use process of the module.
- Pre-set a comparative value for the module, under the mode of automatic measurement, if the measured distance value is smaller than the pre-set value, the pin COMP/Trig will output a low level. In this way, this module can be used asan ultrasonic proximity switch.
- The module is equipped with the funciton of servo controlling. Under the mode of non-automatic measurement, it can combine with a servo into a 180º measuring module to sean the obstacles at the range of O~ 180º.
- The module has a 123 bytes of EEPROM to memory whose values are kept when the board is turned off.
- The built-in temperature compensation circuit of the module is able to increase the accurary of the measurement.
- The module has a bulit-in temperature measurement component to read the environmental temperature with a resolution of 0.1ºC.
- Power reverse protection
- Automatic measurement of time interval can be modified.
- Analog voltage output, voltage and the measured distance is proportional.

Pinout



Num	Label	Description
1	VCC	Power input (3.3V-5.SV)
2	GND	Ground
3	NRST	Reset
4	ECHO	Measured distance presented by the Data Output 0-25000US by PWM pulse width, 1 CM/ S0US representative
5	SERVO	Servo Control Pin
6	COMP/TRIG	COMP: On/OFF mode, when the detected distance is smaller than a pre-set value, this pin pulls low./TRIG: PWM mode trigger input
7	DAC_OUT	Analog voltage output; the voltage is proportional to the distance
8	RXD	Asynchronous communication module data receiving pin: RS232/TTL level
9	TXD	Asynchronous communication module data receiving pin: RS232/TTL level

Tutorial

The functions of URM37 VS.0 are so powerful. Now, let us get known about the basic functions of the module. There are three measurement modes:

1. PWM triggered measurement mode

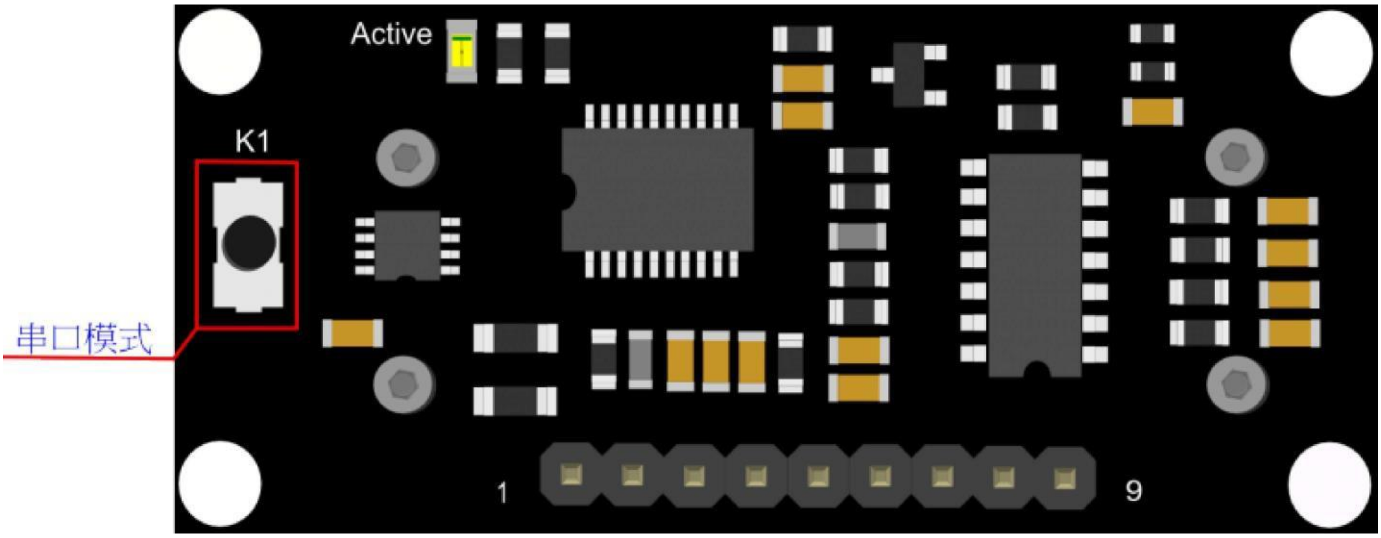
2. Automatically measure mode
3. Serial passive measurement

Then it also supports:

- Simulation volume output (proportional with measurement distance, 6.8mV/cm)
- Temperature read
- Serial level choose(TTL or RS232 level)
- Interna! EEPROM without losing data
- Serial EEPROM data read

The products have been conducted a set of rigorous tests by us, when you get your purchase, you can do sorne setting according to your demands, firstly, you may have to set the serial port-level (or RS232 TTL level), then we can access to the module through the serial port, then set the range mode (0x02 writes data on the interna! EEPROM address), after that, you can access to ultrasound module through MCU or PC.

To begin with this Ultrasonic Sensor, there is a software could help to make ita lot of easier.And there are sorne paramenters you may want to reverse to meet more situations.



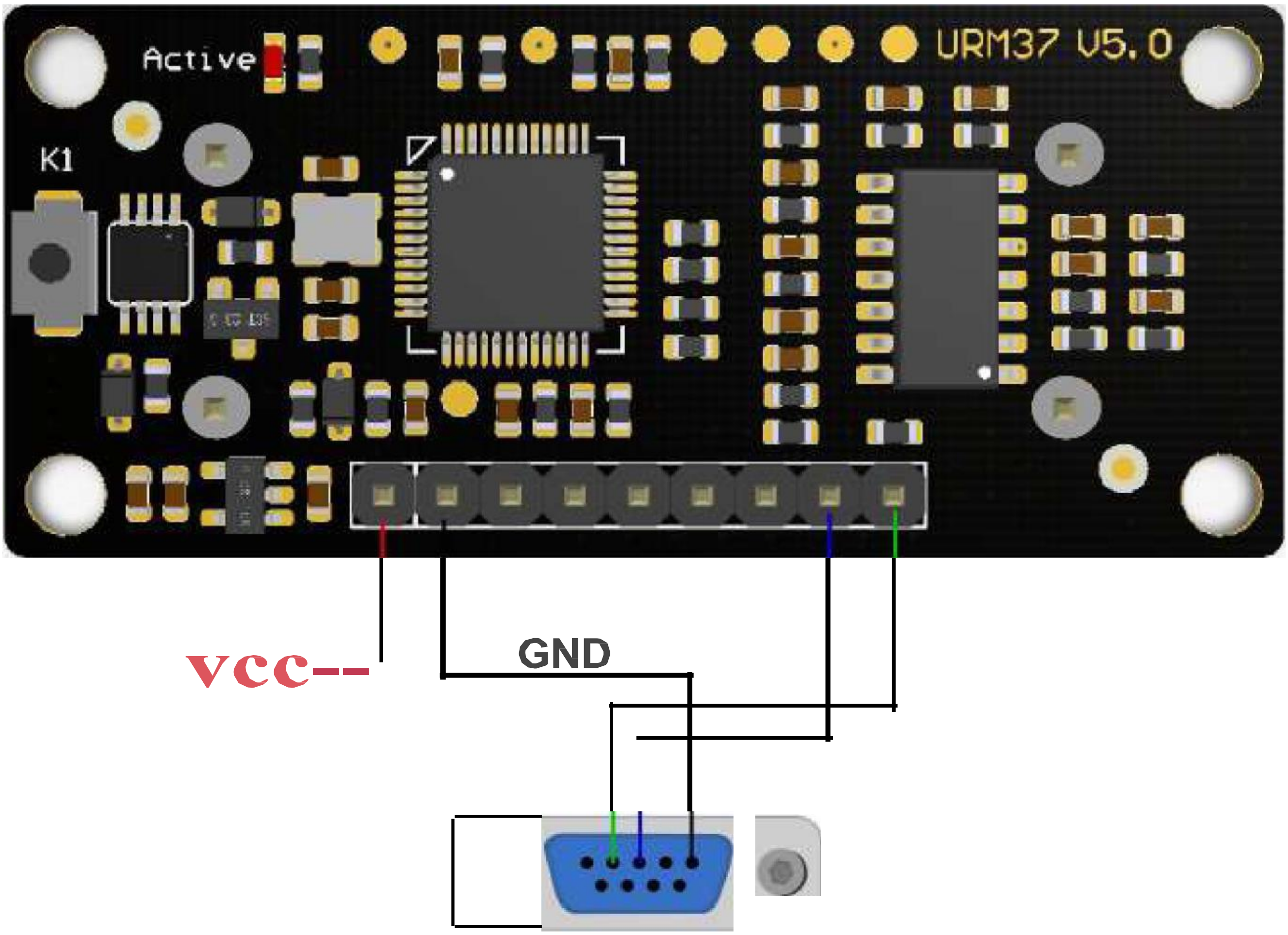
Button for RS232/TTL Choosing

The first basic step to communicate with the model is to choose the serial level_TTL(default) or RS232. We step forward over the last version3.2 which by jumper, now we could do it by **pressing the only one button** on the board for 1 second, after the light turn off from state-on, release the botton. Repower again, the indicator appears to flash like **once long and once short** -present **TTL** level output, **once long and twice short** flash presenting **RS232** level.

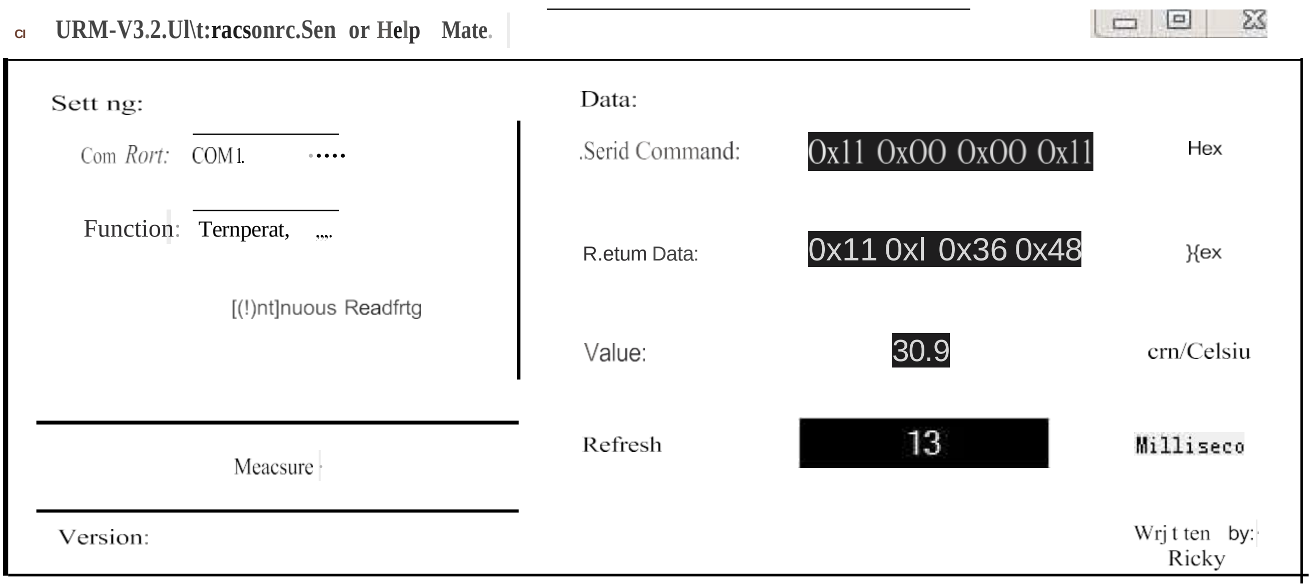
⚠ Do not connect the sensor to TTL MCU when the output mode is set to RS232, doing so will permanently damage the unit.

Test on Software

This feature is only available for Rev2 and after. If there are no Jumpers, or no button on the back of the sensor, the sensor should be Rev1 and hence not supporting this feature.



Power on the sensor, read the blink of the LED(active) to get the serial level(see above), wire according to the above picture. After this, you can use our "URMV3.2HelpMate (https://www.dfrobot.com/image/data/SEN0001/URMV3.2He1pMate.rar)" to test the module.



The usage of the software is very simple: ensure that there is no other software on the computer occupying the serial port, and then run Mate.exe, select the COM Port, and choose the parameter [illegible] measure the temperature and the distance.

Other Setting address in EEPROM

Here, we are talking about the meaning of the data in EEPROM several addresses.(For more details, can be found in the Serial control protocol (https://www.dfrobot.com/wiki/index.php?title=URM37_V3.2_Ultrasonic_Sensor_SKU:SEN0001_ == =Serial_control_protocol) part)

Address	Meaning
0x00	larger than set distance
0x01	less than set distance
0x02	measure mode(write 0xaa present automatically measure mode, other data except 0xaa present PWM Passive measurement mode)
0x03	Serial level mode(TTL/RS232)(write 0xO0 present TTL mode, 0x01 present RS232 mode, other data will be modified to 0x00)
0x04	automatically measure time span(minimum value: 70ms; maximum value: 255ms; default value: 1O0ms. Witting format is 8-bit 16 binary,and its unit is ms. e.g. Write 6E means 110ms)
0x0S	measure sensitivity(the range of writing data is 0x0a-0xc8(equivalent to decimal 10-200), the smaller the value, the higher the sensivity. The default value is 0x0a, and the sensitivity is the highest.)

The factory default settings

- Serial TTL level
- Measure mode: PWM trigger
- Comparison of distance: O
- Automatically measure interval time:25ms
- Interna! EEPROM Data are all 0x00
- the EEPROM address are unavailable: 0x00~0x04, please do not try to modify the data.

Three Measure Modes

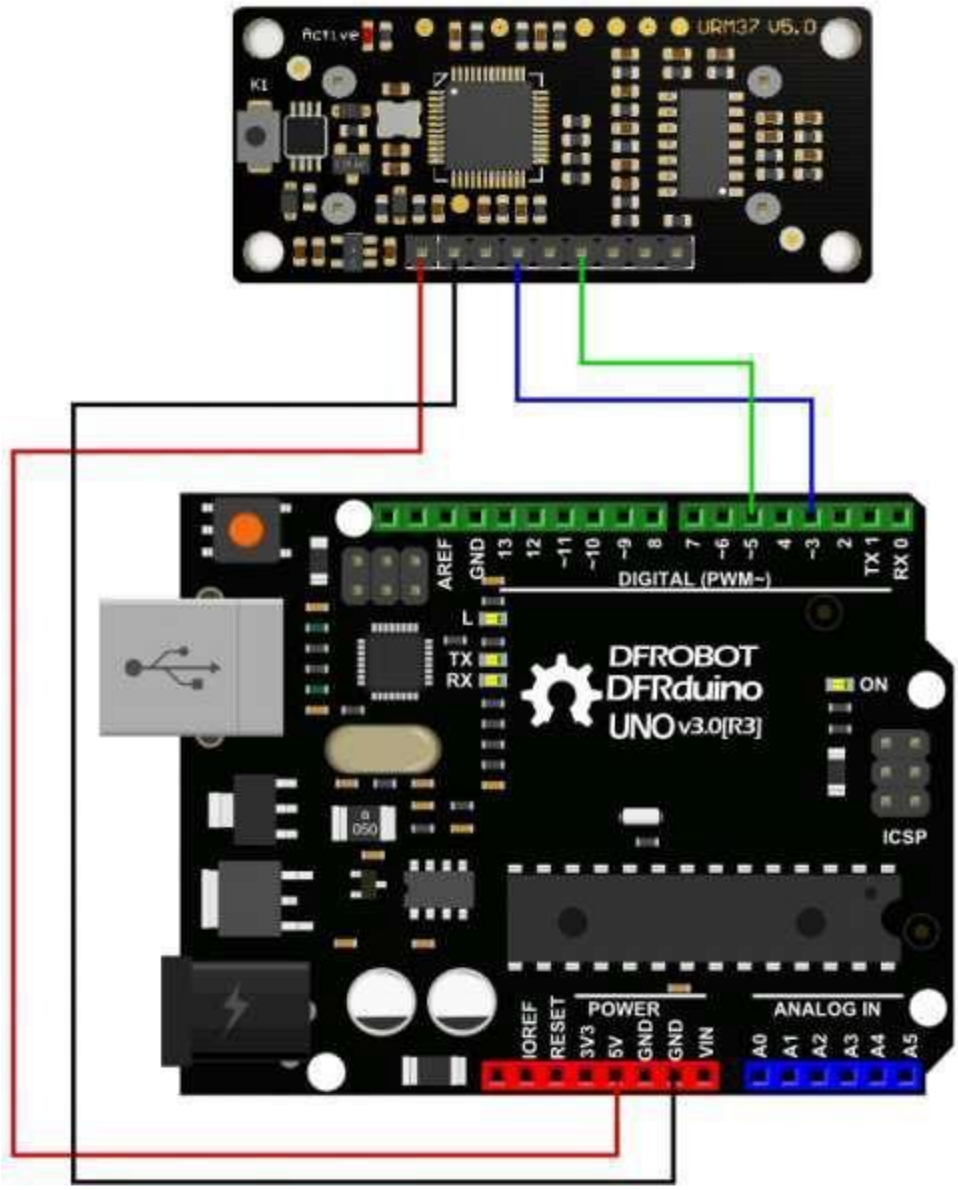
PWM trigger mode

PWM Output in Trigger Mode

In trigger mode, pin COMP/TRIG produces a trigger pulse signal of low level, starting distance measurement operation once. At the same time, this low level pulse width represents the fJdldllleLel IUI LUIILIUllll!:J u,e dll!:Jle UI u,e ::,ervu::, IULdLIUII. IOU Ue!:Jlee ::,fJIIL IIILU '+O dll!:Jle controlling parameters, which means each parameter is equal to 4 degree. The parameter ranges from O to 45 and every S0US pulses representa angle controlling parameter. When send out the trigger pulse, the MOTO pin of the module will produce servo controlling pulse to alter the

rotating degree of the servo. And then the detected distance will be output in the form of low level pulse via PWM from the ECHO pin. Every S0US pulses represent 1 centimeter. In this way, we can read the distance. The measurement is invalid if it returns a pulse of S0000US.

Upload the code below to your arduino board, wire the devices together, then you can realize the distance measurement.



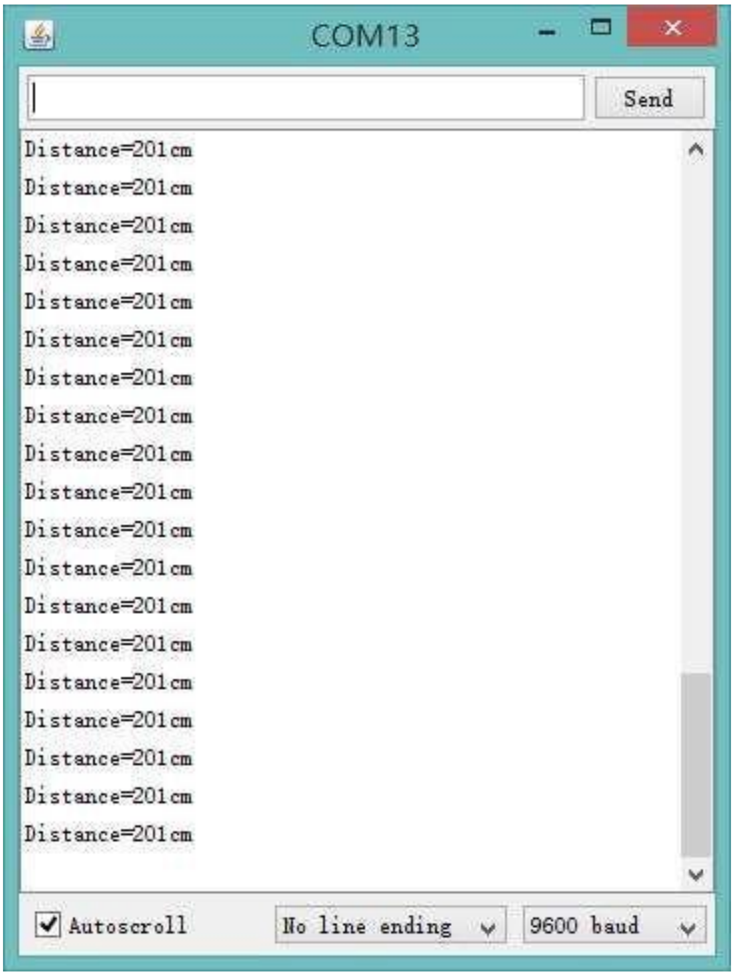
Demo code

```
// # Editor     roker
// # Date       05.03.2018

// # Product name: URM V5.0 ultrasonic sensor
// # Product SKU  SEN0001
// # Version     1.0

// # Description:
// # The Sketch for scanning 180 degree area 3-500cm detecting range
// # The sketch for using the URM37 PWM trigger pin mode from DFRobot
// #   and writes the values to the serialport
// # Connection:
// #       Vcc (Arduino)  -> Pin 1 VCC (URM V5.0)
// #       GND (Arduino)   -> Pin 2 GND (URM V5.0)
// #       Pin 3 (Arduino)   -> Pin 4 ECHO (URM V5.0)
// #       Pin 5 (Arduino)   -> Pin 6 COMP/TRIG (URM V5.0)

// # Working Mode: PWM trigger pin  mode.

int URECHO = 3;         // PWM Output 0-25000US,Every 50US represent 1cm
int URTRIG = 5;         // trigger pin

unsigned int DistanceMeasured = 0;

void setup()
{
  //Serial initialization
  Serial.begin(9600);                        // Sets the baud rate to 9600
  pinMode(URTRIG, OUTPUT);                   // A low pull on pin COMP/TRIG
  digitalWrite(URTRIG, HIGH);                // Set to HIGH
  pinMode(URECHO, INPUT);                    // Sending Enable PWM mode command
  delay(500);
  Serial.println("Init the sensor");

}
void loop()
{
  Serial.print("Distance=");
  digitalWrite(URTRIG, LOW);
  digitalWrite(URTRIG, HIGH);

  unsigned long LowlevelTime = pulsein(URECHO, LOW) ;
  if (LowlevelTime >= 50000)              // the reading is invalid.
  {
    Serial.println("Invalid");
  }
  else
  {
    DistanceMeasured = LowlevelTime / 50;  // everv 50us low level stands for 1cm
```

```
    Serial.print(DistanceMeasured);
    Serial.println("cm");
  }


  delay(200);
}
```

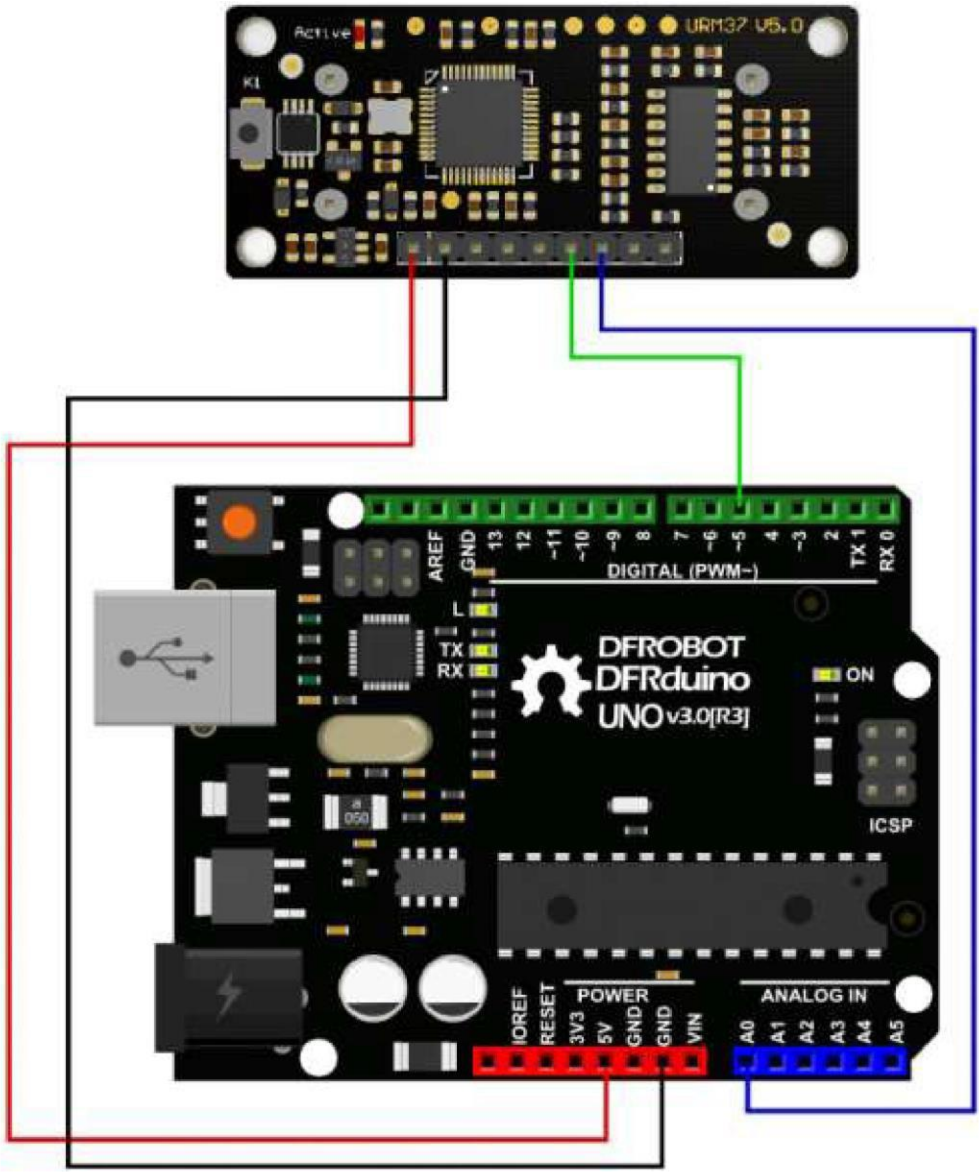
Result

Arduino will send the distance to master computer through serial port. The baud rate should be set to 9600.



Analog Voltage Output in Trigger Mode

Once we are able to implement the most basic measurements, we can use more of the features on our module, such as the analog voltage output function mentioned above. The output voltage is proportional to the measured distance with the proportion of 4.12SmV / cm. when exceeded the measurement range, the output voltage is 3.3V at full voltage. By reversing the code"#define Measure 1 "to "#define Measure O", we can read the distance by analog voltage. Upload the cierno code to Arduino, and connect the ultrasonic sensor with Arduino as the way shown below.


Active
URM37 V5.0
K1
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX 1
RX 0
DIGITAL (PWM~)
L
TX
RX
DFROBOT
DFRduino
UNO v3.0[R3]
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5

Demo Code

```
// # Editor     roker
// # Date       05.03.2018

// # Product name: URM V5.0 ultrasonic sensor
// # Product SKU  SEN0001
// # Version      1.0

// # Description:
// # The Sketch for scanning 180 degree area 3-500cm detecting range
// # The sketch for using the URM37 PWM trigger pin mode from DFRobot
// #   and writes the values to the serialport
// # Connection:
// #       Vcc (Arduino)  -> Pin 1 VCC (URM V5.0)
// #       GND (Arduino)  -> Pin 2 GND (URM V5.0)
// #       Pin 5 (Arduino) -> Pin 6 COMP/TRIG (URM V5.0)
// #       Pin A0 (Arduino) -> Pin 7 DAC (URM V5.0)

int URTRIG = 5;         // trigger pin
int sensorPin = A0;     // select the input pin for the potentiometer
int sensorValue = 0;    // variable to store the value coming from the sensor

unsigned int DistanceMeasured = 0;

void setup()
{
  //Serial initialization
  Serial.begin(9600);                        // Sets the baud rate to 9600
  pinMode(URTRIG, OUTPUT);                   // A low pull on pin COMP/TRIG
  digitalWrite(URTRIG, HIGH);                // Set to HIGH
  delay(500);
  Serial.println("Init the sensor");

}
void loop()
{
  Serial.print("Distance=");
  digitalWrite(URTRIG, LOW);
  digitalWrite(URTRIG, HIGH);
  delay(200);
  sensorValue = analogRead(sensorPin);

  sensorValue = sensorValue * 1.1; //  (sensorValue * 5000 / 1024) / 4.125 = sensorValue
  Serial.print(sensorValue);
  Serial.println("cm");

}
```

Note: the error of the distance got by calculating output analog voltage is bigger than that of the distance by other ways.

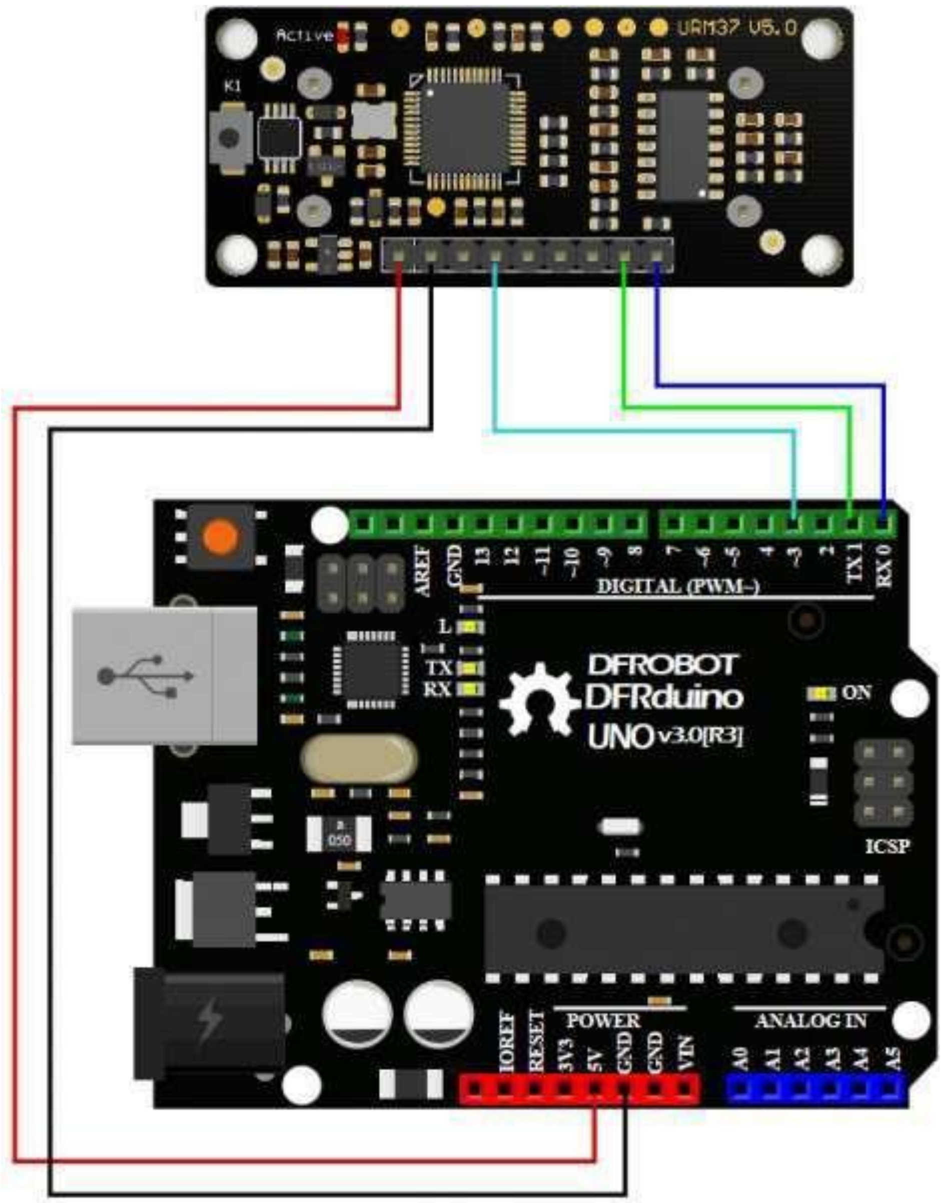
Auto Measure Mode

By means of the computer software or MCU Module, write 0xAA to 0x02 address to switch to automatic measurement mode. Writting a 8-bit 16 binary data to 0x04 address to reverse the measure time interval. This module measures distance automatically every 25 ms (Settable), then compare the data with the set value, if equal to or less than the set value, COMP/TRIG pin output low. In addition, in every measure, the PWM Terminal will read the distance as a low level pulse, S0uS represents **1** cm.

Tips: if you have set the Compare value, you could use this module as a Ultrasonic Switch.

Download the sample code to the Arduino board, then wire as shown modules and Arduino connected on ultrasonic distance measurement can be achieved.

Note:download first befare connect the Arduino TX/RX, otherwise it will fail.



Demo Code

```
// # Editor     roker
// # Date       05.03.2018

// # Product name: URM VS.0 ultrasonic sensor
// # Product SKU  SEN0001
// # Version     1.0

// # Description:
// # The sketch for using the URM37 autonomous   mode from DFRobot
// #   and writes the values to the serialport


// # Connection:
// #       Vcc (Arduino)      -> Pin 1 VCC (URM  VS.0)
// #       GND (Arduino)      -> Pin 2 GND (URM  VS.0)
// #       Pin 3 (Arduino)    -> Pin 4 ECHO (URM  VS.0)
// #        Pin TX1 (Arduino)  -> Pin 8 RXD (URM  VS.0)
// #        Pin RX0 (Arduino)  -> Pin 9 TXD (URM  VS.0)
// # Working Mode:  Automatic measurement model.

int URECHO = 3; // PWM Output 0-25000US, Every 50US represent 1cm

unsigned int Distance = 0;
uint8_t AutomaticModelCmd[4] = {0x44, 0x02, 0xaa, 0xf0}; // distance measure command

void setup()
{
  Serial.begin(9600);          // Serial initialization
  delay(5000);                  // wait for sensor setup
  AutomaticModelSetup();        //Automatic measurement model set
}

void loop()
{
  AutomaticMeasurement();
  delay(100);
}


void AutomaticModelSetup(void)
{
  pinMode(URECHO, INPUT);
  for (int i = 0; i < 4; i++)
  {
    Serial.write(AutomaticModelCmd[i]);// Sending Automatic measurement model command
  }
}
```

```
void AutomaticMeasurement(void)
 {
   unsigned long DistanceMeasured = pulsein(URECHO, LOW);
   if (DistanceMeasured >= 50000) // the reading is invalid.
   {
    Serial.print("Invalid");
   }
   else
   {
     Distance = DistanceMeasured / 50;  // every 50us low level stands far 1cm
     Serial.print("Distance=");
     Serial.print(Distance);
     Serial.println("cm");
   }
 }
```

Result

Arduino sends the distance information to the computer through serial port.

COM13

Serial Passive Mode

In this mode, actually, as long as you wire the module TX & RX with the MCU, justas we did in the test on software (https://www.dfrobot.com/wiki/index.php?title=URM37_V4.0_Ultrasonic_Sensor_SKU:SEN0001_#2_Test_on_Software),you are using this mode.By serial, you have all authority to access to the sensor such as: ultrasonic distance measurement, temperature measurement, the distance changes, automatic measurement intervals set, serial port set(RS232 or TTL, reboot to take effect). e.g.

1. Read the temperature data command: 0x11 0x00 0x00 0x11
2. Read the distance data command: 0x22 0x00 0x00 0x22
3. Read EEPROM data command: 0x33 0x00 0x00 0x33
4. Write EEPROM data command: 0x44 0x02 0x00 0x46

Download the code below to your uno board(if you use the leonardo,please modify the code for the serial problem, help on arduino.cc), then wire the TX/RX,SV,GND.Follow test on software (https://www.dfrobot.com/wiki/index.php?title=URM37_V4.0_Ultrasonic_Sensor_SKU:SEN0001_#2_Test_on_Software).Here,we use the sensor to read the tempreture.

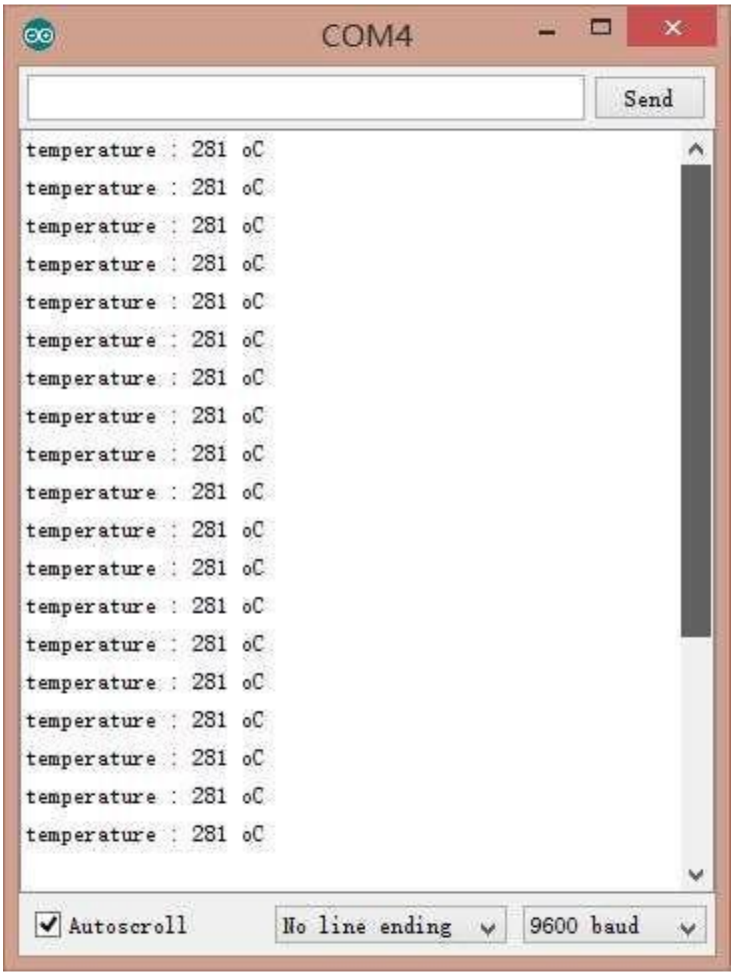
Demo Code

```
// # Editor     roker
// # Date     05.03.2018

// # Product name: URM VS.0 ultrasonic sensor
// # Product SKU  SEN0001
// # Version     1.0

// # Description:
// # The sketch for using the URM37 Serial  mode from DFRobot
// #   and writes the values to the serialport

// # Connection:
// #       Vcc (Arduino)     -> Pin 1 VCC (URM VS.0)
// #       GND (Arduino)     -> Pin 2 GND (URM VS.0)
// #       Pin TXl (Arduino)  -> Pin 8 RXD (URM VS.0)
// #       Pin RX0 (Arduino)  -> Pin 9 TXD (URM VS.0)
// # Working Mode: Serial  Mode.

uint8_t EnTempCmd[4] = {0x11, 0x00, 0x00, 0x11}; // temperature measure command
uint8_t TempData[4];
unsigned int TempValue =0;
void setup()
{
  Serial.begin(9600);
  delay(100);
  Serial.println("Init the sensor");
}
void loop()
{
  SerialCmd();
  delay(200);
}
void SerialCmd()
{
  int i;
  for (i = 0; i < 4; i++) {
    Serial.write(EnTempCmd[i]);
  }
  while (Serial.available() > 0) // if received data
  {
    for (i = 0; i < 4; i++) {
      TempData[i] = Serial.read();
    }
    TempValue = TempData[l] << 8;
    TempValue = TempValue + TempData[2];
    Serial. print("temperature : ");
    Serial.print(TempValue, DEC);
    Serial.println(" oC");
  ]
```

```
}
```

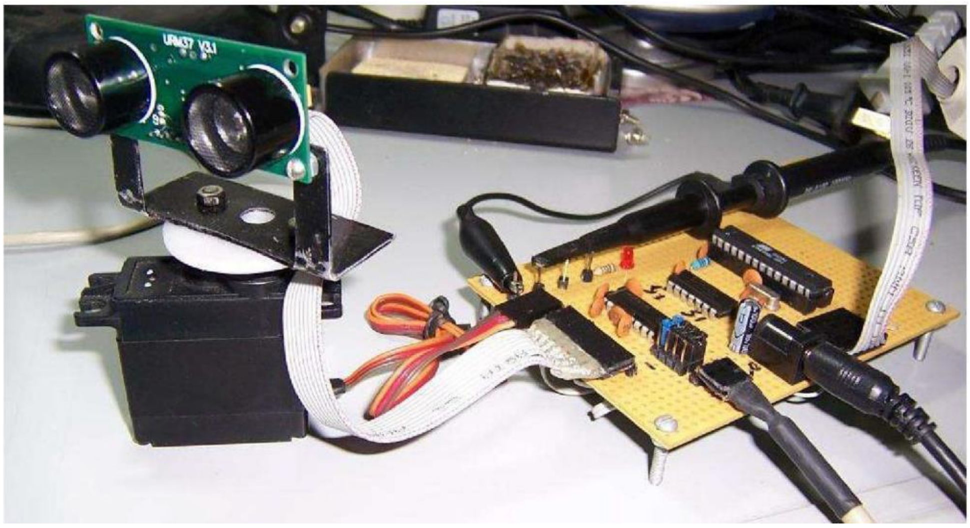
Result

This temperature was magnified 10 times, in the test, actual tempreture is 28.1 degrees Celsius.



Servo Rotation Reference Table

DEC	0	1	2	3	4	5	6	7	8	9	10	1
HEX	0	01	02	03	04	05	06	07	08	09	0A	0
Degree	0	6	12	18	24	29	35	41	47	53	59	6
DEC	16	17	18	19	20	21	22	23	24	25	26	2
HEX	10	11	12	13	14	15	16	17	18	19	1A	1
Degree	94	100	106	112	117	123	129	135	141	147	153	1
DEC	32	33	34	35	36	37	38	39	40	41	42	4
HEX	20	21	22	23	24	25	26	27	28	29	2A	2
Degree	188	194	200	206	211	217	223	229	235	241	247	2



Arduino Sketch

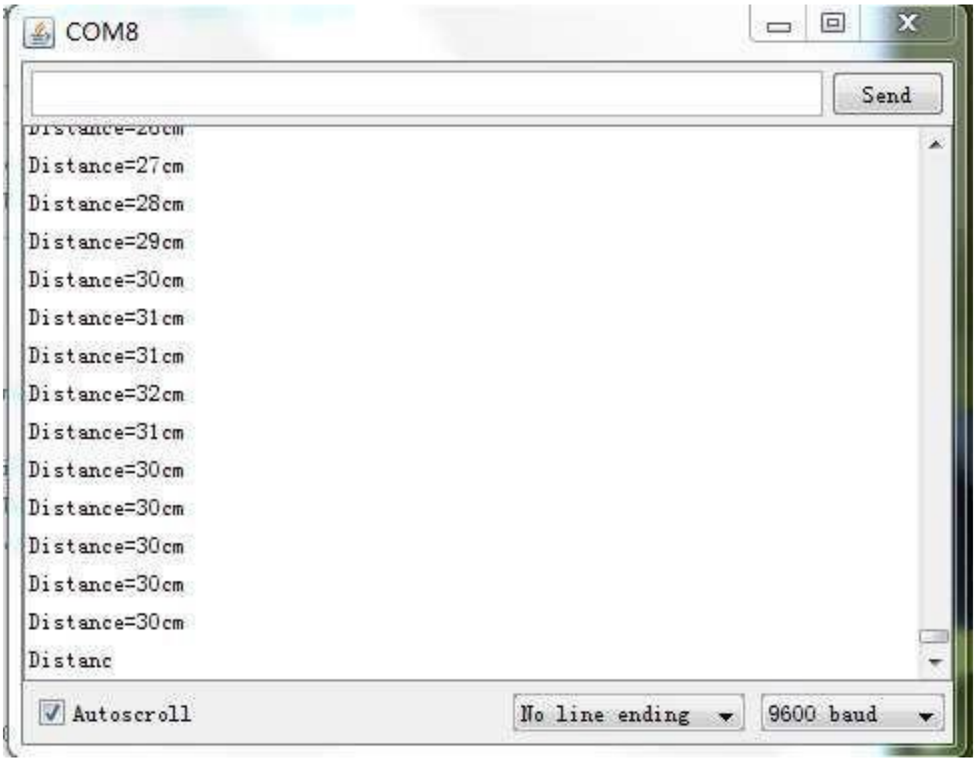
NOTE: Please put the sensor jumpers to TTL mode. See above far a picture indicating TTL mode.

---

cpp

```
// # Editor    Jiang from DFRobot
// # Data      24.07.2012

// # Product name:ultrasonic scanner Kit
// # Product SKU:SEN0001
// # Version  0.2

// # Description:
// # The Sketch for scanning 180 degree area 4-500cm detecting range

// # Connection:
// #       Pin 1 VCC (URM V3.2) -> VCC (Arduino)
// #       Pin 2 GND (URM V3.2) -> GND (Arduino)
// #       Pin 4 PWM (URM V3.2) -> Pin 3 (Arduino)
// #       Pin 6 COMP/TRIG (URM V3.2) -> Pin 5 (Arduino)
// # Pin mode: PWM
// # Working Mode: PWM passive control mode.
// # If it is your first time to use it,please make sure the two jumpers to the right hanc
// # side of the device are set to TTL mode. You'll also find a secondary jumper on
// # the left hand side, you must break this connection or you may damage your device.

#include <Servo.h>                                 // Include Servo library
Servo myservo;                                     // create servo object to control a se

int pos=0;                                         // variable to store the servo positic
int URPWM=3;                                       // PWM Output 0-25000us,every 50us re;
int URTRIG=S;                                      // PWM trigger pin
boolean up=true;                                   // create a boolean variable
unsigned long time;                                // create a time variable
unsigned long urmTimer = 0;                         // timer for managing the sensor reac

unsigned int Distance=0;
uint8_t EnPwmCmd[4]={0x44,0x22,0xbb,0x01};         // distance measure command

void setup(){                                     // Serial initialization
  Serial.begin(9600);                             // Sets the baud rate to 9600
  myservo.attach(9);                               // Pin 9 to control servo
  PWM_Mode_Setup();
}

void loop(){
   if(millis()-time>=20){                          // interval 0.02 seconds
     time=millis();                                // get the current time of programme
     if(up){                                       // judge the condition
      if(pos>=0 && pos<=179){
        pos=pos 1;                                 // in steps of 1 degree
        mvservo.write(pos);                        // tell servo to go to position in var
```

```
      }
      if(pos>179)  up= false;                         // assign the variable again
    }
     else {
      if(pos>=1 && pos<=180){
        pos=pos-1;
        myservo.write(pos);
      }
      if(pos<1)  up=true;
    }
   }

   if(millis()-urmTimer>50){
     urmTimer=millis();
     PWM_Mode();
   }
 }

 void PWM_Mode_Setup(){
  pinMode(URTRIG,OUTPUT);                            // A low pull on pin COMP/TRIG
  digitalWrite(URTRIG,HIGH);                         // Set to HIGH

  pinMode(URPWM, INPUT);                             // Sending Enable PWM mode command

  for(int i=0;i<4;i  ){
      Serial.write(EnPwmCmd[i]);
   }
}

void PWM_Mode(){                                     // a low pull on pin COMP/TRIG  trig{
    digitalWrite(URTRIG, LOW);
    digitalWrite(URTRIG, HIGH);                      // reading Pin PWM will output pulse

    unsigned  long  DistanceMeasured=pulsein(URPWM,LOW);

    if(DistanceMeasured==50000){                     // the reading is invalid.
      Serial.print("Invalid");
   }
    else{
      Distance=DistanceMeasured/50;                  // every 50us low level stands far le
   }
  Serial.print("Distance=");
  Serial.print(Distance);
  Serial.println("cm");
}
```



Protocol

Serial setting: Port rate: 9600; Parity: none; Stop bit: 1

Command: Control command consists of four bits, command dataO data1 sum. Sum=Low 8 bit of the sum of command dataO data1.

Command Format	Function	Description
Ox11 + NC +NC+ Sum (Sample: Ox11 OxOO 0xOO Ox11)	Enable 16 bit temperature reading	Reading the temperature, the return data format will be: Ox11 High(temperature) Low(temperature) SUM; If the temperature is above O, the first four bits of High will be all O. If the temperature is below O, the first four bits of High will be all 1; The last 4 bits of High together with the Low bits stands for 12bits temperature. The resolution is 0.1. When the reading is invalid, it returns Ox11 OxFF OxFF SUM
Ox22 + Degree + NC+SUM (Sample: Ox22 0x00 OxOO Ox22	Enable 16 bit distance reading	The degree in the command is used to control a servo motor to rotate corresponding degree; Degree: 0-46 stands for 0-270 degrees, for example, 3 stands for 18 degrees; Return data format will be: Ox22 + High(distance) + Low(distance) SUM. When the reading is invalid, it returns Ox22 OxFF OxFF

Coml),and Format	Function	Description
		SUM
0x33 + Add + NC +SUM	Enable interna! EEPROM reading	Return data will be 0x33 + Add +Data+ SUM.
0x44+ Add+ Data+ SUM (Sample: 0x44 0x02 0xbb 0x01) Enable PWM mode	Enable interna! EEPROM writing	Written data can only from 0-255. Address Ox00-0x02 is used to configure the mode. 0x00 - threshold distance (Low) 0x01 - threshold distance (High) 0x02 - Operation Mode (0xaa for autonomous mode) (0xbb for PWM passive control mode);The return data format will be: 0x44 + Add +Data+ SUM

NOTE: NC stands for any data, SUM stands for sum, Add stands for address.

1. PWN_ON must be set to High to enable sensor.

Examples: Function to calculate the temperature:

```
IF(HightByte>=0xF0)
{
Temperature= ((HightByte-0xF0)*256-LowByte)/10
 }
Else
{
Temperature= ((HightByte)*256-LowByte)/10
}
```

Trouble shooting

1. If you have connected sensor to the Arduino, but unable to use it, please first check the current serial port-level mode, it may be in TTL level, while our module works in RS232 levels.

2. The ultrasonic attenuation violently in the air (inversely proportional to the d^2(distance)), besides, barrier surface reflection of the sound is affected by many factors (such as barrier

shape, onentatIon and texture) the Intluence ot u trasonIc dIstance measurement Is theretore limited.

3. The far testing distance is a wall, close test can be a pen. Analyte based on the use of the environment and quality of different measurement may result in inconsistent with the data provided.

4. The mentioned servo above is a ordinary model on the market, can be rotated 180 degrees. If you use a special steering servo, it may draw the user's attention to control the timing in a different way.

5. If you are experiencing technical issues, please ask on our **forum** (https://www.dfrobot.com/forum/) or send us **email,** we will answer your questions as soon as possible.

   More question and cool idea, visit DFRobot Forum (https://www.dfrobot.com/index.php?route=DFblog/blogs)

More

- Arduino Library from milesburton(IDE 0023 and below) (http://milesburton.com/URM37_Ultrasonic_Distance_Measurement_Library)
- Old version_URM37 V3.2 (https://www.dfrobot.com/wiki/index.php?title=URM37_V3.2_Ultrasonic_Sensor_SKU:SEN000 **1**_#Resources)

Get **URM37 VS.O Ultrasonic Sensor** (https://www.dfrobot.com/product-53.html) from DFRobot Store or **DFRobot Distributor.** (https://www.dfrobot.com/index.php?route=information/d istributorslogo)

Anexo 12
IMU

ICM-20948

World's Lowest Power 9-Axis MEMS MotionTracking™ Device

GENERAL DESCRIPTION

The ICM-20948 is the world's lowest power 9-axis MotionTracking device that is ideally suited for Smartphones, Tablets, Wearable Sensors, and IoT applications.

- 3-axis gyroscope, 3-axis accelerometer, 3-axis compass, and a Digital Motion Processor™ (DMP™) in a 3 mm x 3 mm x 1 mm (24-pin QFN) package
- DMP offloads computation of motion processing algorithms from the host processor, improving system power performance
- Software drivers are fully compliant with Google's latest Android release
- EIS FSYNC support

ICM-20948 supports an auxiliary I²C interface to external sensors, on-chip 16-bit ADCs, programmable digital filters, an embedded temperature sensor, and programmable interrupts. The device features an operating voltage range down to 1.71V. Communication ports include I²C and high speed SPI at 7 MHz.

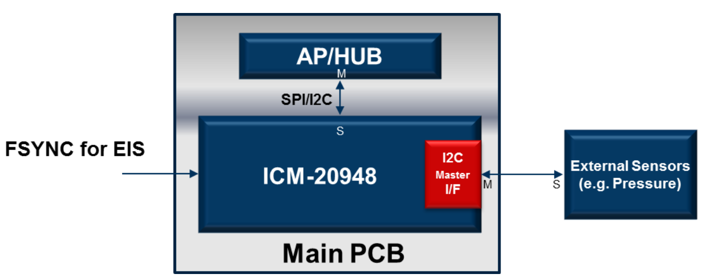
Note: ICM-20948 VDDIO range is 1.71V to 1.95V, different than the MPU-9250 9-axis device.

ORDERING INFORMATION

PART	TEMP RANGE	PACKAGE
ICM-20948†	−40°C to +85°C (Compass: −30°C to +85°C)	24-Pin QFN

†Denotes RoHS and Green-Compliant Package

BLOCK DIAGRAM



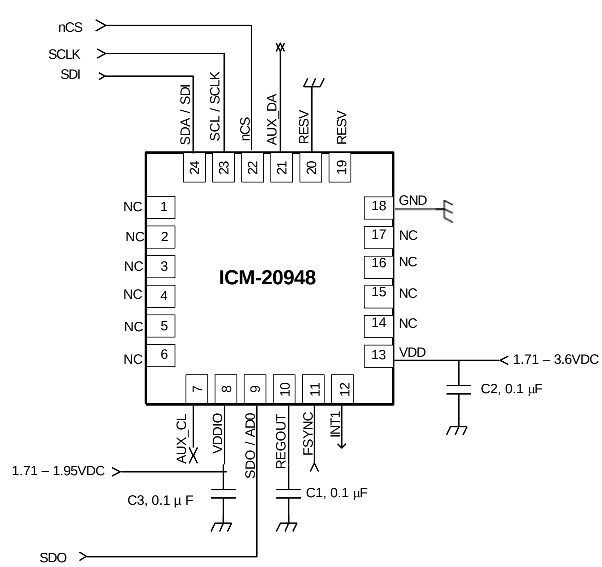
APPLICATIONS

- Smartphones and Tablets
- Wearable Sensors
- IoT Applications

FEATURES

- Lowest Power 9-Axis Device at 2.5 mW
- 3-Axis Gyroscope with Programmable FSR of ±250 dps, ±500 dps, ±1000 dps, and ±2000 dps
- 3-Axis Accelerometer with Programmable FSR of ±2*g*, ±4*g*, ±8*g*, and ±16*g*
- 3-Axis Compass with a wide range to ±4900 µT
- Onboard Digital Motion Processor (DMP)
- Android support
- Auxiliary I²C interface for external sensors
- On-Chip 16-bit ADCs and Programmable Filters
- 7 MHz SPI or 400 kHz Fast Mode I²C
- Digital-output temperature sensor
- VDD operating range of 1.71V to 3.6V
- MEMS structure hermetically sealed and bonded at wafer level
- RoHS and Green compliant

TYPICAL OPERATING CIRCUIT



LIST OF FIGURES

LIST OF TABLES

1 GENERAL DESCRIPTION

1.1 PURPOSE AND SCOPE

This document is a preliminary data sheet, providing a description, specifications, and design related information on the ICM-20948 MotionTracking device.

For references to register map and descriptions of individual registers, please refer to the ICM-20948 Register Map and Register Descriptions document.

1.2 PRODUCT OVERVIEW

The ICM-20948 is a multi-chip module (MCM) consisting of two dies integrated into a single QFN package. One die houses a 3-axis gyroscope, a 3-axis accelerometer, and a Digital Motion Processor™ (DMP). The other die houses the AK09916 3-axis magnetometer from Asahi Kasei Microdevices Corporation. The ICM-20948 is a 9-axis MotionTracking device all in a small 3x3x1mm QFN package. The device supports the following features:

- FIFO of size 4kBytes (FIFO size will vary depending on DMP feature-set)
- Runtime Calibration
- Enhanced FSYNC functionality to improve timing for applications like EIS

ICM-20948 devices, with their 9-axis integration, on-chip DMP, and run-time calibration firmware, enable manufacturers to eliminate the costly and complex selection, qualification, and system level integration of discrete devices, guaranteeing optimal motion performance for consumers.

The gyroscope has a programmable full-scale range of ±250 dps, ±500 dps, ±1000 dps, and ±2000 dps. The accelerometer has a user-programmable accelerometer full-scale range of ±2*g*, ±4*g*, ±8*g*, and ±16*g*. Factory-calibrated initial sensitivity of both sensors reduces production-line calibration requirements.

Other key features include on-chip 16-bit ADCs, programmable digital filters, an embedded temperature sensor, and programmable interrupts. The device features I^2C and SPI serial interfaces, a VDD operating range of 1.71V to 3.6V, and a separate digital IO supply, VDDIO from 1.71V to 1.95V.

Communication with all registers of the device is performed using I^2C at up to 100 kHz (standard-mode) or up to 400 kHz (fast-mode), or SPI at up to 7 MHz.

By leveraging its patented and volume-proven CMOS-MEMS fabrication platform, which integrates MEMS wafers with companion CMOS electronics through wafer-level bonding, InvenSense has driven the package size down to a footprint and thickness of 3 mm x 3 mm x 1 mm (24-pin QFN), to provide a very small yet high-performance, low-cost package. The device provides high robustness by supporting 20,000*g* shock reliability.

1.3 APPLICATIONS

- Smartphones and Tablets
- Wearable Sensors
- IoT Applications
- Drones

2 FEATURES

2.1 GYROSCOPE FEATURES

The triple-axis MEMS gyroscope in the ICM-20948 includes the following features:

- Digital-output X-, Y-, and Z-axis angular rate sensors (gyroscopes) with a user-programmable full-scale range of ±250 dps, ±500 dps, ±1000 dps, and ±2000 dps, and integrated 16-bit ADCs
- User-selectable ODR; User-selectable low pass filters
- Self-test

2.2 ACCELEROMETER FEATURES

The triple-axis MEMS accelerometer in ICM-20948 includes the following features:

- Digital-output X-, Y-, and Z-axis accelerometer with a programmable full scale range of ±2g, ±4g, ±8g, and ±16*g*, and integrated 16-bit ADCs
- User-selectable ODR; User-selectable low pass filters
- Wake-on-motion interrupt for low power operation of applications processor
- Self-test

2.3 MAGNETOMETER FEATURES

The triple-axis MEMS magnetometer in ICM-20948 includes a wide range of features:

- 3-axis silicon monolithic Hall-effect magnetic sensor with magnetic concentrator
- Wide dynamic measurement range and high resolution with lower current consumption.
- Output data resolution of 16-bits
- Full scale measurement range is ±4900 µT
- Self-test function with internal magnetic source to confirm magnetic sensor operation on end products

2.4 DMP FEATURES

The DMP in ICM-20948 includes the following capabilities:

- Offloads computation of motion processing algorithms from the host processor. The DMP can be used to minimize power, simplify timing, simplify the software architecture, and save valuable MIPS on the host processor for use in applications.
- The DMP enables ultra-low power run-time and background calibration of the accelerometer, gyroscope, and compass, maintaining optimal performance of the sensor data for both physical and virtual sensors generated through sensor fusion. This enables the best user experience for all sensor enabled applications for the lifetime of the device.
- DMP features simplify the software architecture resulting in quicker time to market.
- DMP features are OS, Platform, and Architecture independent, supporting virtually any AP, MCU, or other embedded architecture.

2.5 ADDITIONAL FEATURES

The ICM-20948 includes the following additional features:

- I^2C at up to 100 kHz (standard-mode) or up to 400 kHz (fast-mode) or SPI at up to 7 MHz for communication with registers
- Auxiliary master I^2C bus for reading data from external sensors (e.g. magnetometer)
- Digital-output temperature sensor
- 20,000*g* shock tolerant
- MEMS structure hermetically sealed and bonded at wafer level
- RoHS and Green compliant

3 ELECTRICAL CHARACTERISTICS

3.1 GYROSCOPE SPECIFICATIONS

Typical Operating Circuit of section 4.2, VDD = 1.8V, VDDIO = 1.8V, T_A=25°C, unless otherwise noted.

NOTE: All specifications apply to Low-Power Mode and Low-Noise Mode, unless noted otherwise

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
GYROSCOPE SENSITIVITY						
Full-Scale Range	GYRO_FS_SEL=0		±250		dps	1
	GYRO_FS_SEL=1		±500		dps	1
	GYRO_FS_SEL=2		±1000		dps	1
	GYRO_FS_SEL=3		±2000		dps	1
Gyroscope ADC Word Length			16		bits	1
Sensitivity Scale Factor	GYRO_FS_SEL=0		131		LSB/(dps)	1
	GYRO_FS_SEL=1		65.5		LSB/(dps)	1
	GYRO_FS_SEL=2		32.8		LSB/(dps)	1
	GYRO_FS_SEL=3		16.4		LSB/(dps)	1
Sensitivity Scale Factor Tolerance	25°C		±1.5		%	2
Sensitivity Scale Factor Variation Over Temperature	-40°C to +85°C		±3		%	2
Nonlinearity	Best fit straight line; 25°C		±0.1		%	2, 3
Cross-Axis Sensitivity			±2		%	2, 3
ZERO-RATE OUTPUT (ZRO)						
Initial ZRO Tolerance	25°C (Component-level)		±5		dps	2
ZRO Variation Over Temperature	-40°C to +85°C		±0.05		dps/°C	2
GYROSCOPE NOISE PERFORMANCE (GYRO_FS_SEL=0)						
Noise Spectral Density	Based on Noise Bandwidth = 10 Hz		0.015		dps/√Hz	2
GYROSCOPE MECHANICAL FREQUENCIES		25	27	29	kHz	2
LOW PASS FILTER RESPONSE	Programmable Range	5.7		197	Hz	1, 3
GYROSCOPE START-UP TIME	From Full-Chip Sleep mode		35		ms	2, 3
OUTPUT DATA RATE	Low-Power Mode	4.4		562.5	Hz	1
	Low-Noise Mode GYRO_FCHOICE=1; GYRO_DLPFCFG=x	4.4		1.125k	Hz	
	Low-Noise Mode GYRO_FCHOICE=0; GYRO_DLPFCFG=x			9k	Hz	

Table 1. Gyroscope Specifications

NOTES:

1. Guaranteed by design.
2. Derived from validation or characterization of parts, not guaranteed in production.
3. Low-noise mode specification.

3.2 ACCELEROMETER SPECIFICATIONS

Typical Operating Circuit of section 4.2, VDD = 1.8V, VDDIO = 1.8V, T_A=25°C, unless otherwise noted.

NOTES: All specifications apply to Low-Power Mode and Low-Noise Mode, unless noted otherwise

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
ACCELEROMETER SENSITIVITY						
Full-Scale Range	ACCEL_FS=0		±2		*G*	1
	ACCEL_FS=1		±4		*G*	1
	ACCEL_FS=2		±8		*G*	1
	ACCEL_FS=3		±16		*G*	1
ADC Word Length	Output in two's complement format		16		Bits	1
Sensitivity Scale Factor	ACCEL_FS=0		16,384		LSB/*g*	1
	ACCEL_FS=1		8,192		LSB/*g*	1
	ACCEL_FS=2		4,096		LSB/*g*	1
	ACCEL_FS=3		2,048		LSB/*g*	1
Initial Tolerance	Component-level		±0.5		%	2
Sensitivity Change vs. Temperature	-40°C to +85°C ACCEL_FS=0		±0.026		%/ºC	2
Nonlinearity	Best Fit Straight Line		±0.5		%	2, 3
Cross-Axis Sensitivity			±2		%	2, 3
ZERO-G OUTPUT						
Initial Tolerance	Component-level, all axes		±25		m*g*	2
Initial Tolerance	Board-level, all axes		±50		m*g*	2
Zero-G Level Change vs. Temperature	0°C to +85°C		±0.80		m*g*/°C	2
ACCELEROMETER NOISE PERFORMANCE						
Noise Spectral Density	Based on Noise Bandwidth = 10 Hz		230		µ*g*/√Hz	2
LOW PASS FILTER RESPONSE	Programmable Range	5.7		246	Hz	1, 3
ACCELEROMETER STARTUP TIME	From Sleep mode		20		ms	2, 3
	From Cold Start, 1 ms V_{DD} ramp		30		ms	2, 3
OUTPUT DATA RATE	Low-Power Mode	0.27		562.5	Hz	1
	Low-Noise Mode ACCEL_FCHOICE=1; ACCEL_DLPFCFG=x	4.5		1.125k	Hz	
	Low-Noise Mode ACCEL_FCHOICE=0; ACCEL_DLPFCFG=x			4.5k	Hz	

Table 2. Accelerometer Specifications

NOTES:

1. Guaranteed by design.
2. Derived from validation or characterization of parts, not guaranteed in production.
3. Low-noise mode specification.

3.3 MAGNETOMETER SPECIFICATIONS

Typical Operating Circuit of section 4.2, VDD = 1.8V, VDDIO = 1.8V, T_A=25°C, unless otherwise noted.

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
MAGNETOMETER SENSITIVITY						
Full-Scale Range			±4900		µT	1
Output Resolution			16		bits	1
Sensitivity Scale Factor			0.15		µT / LSB	1
ZERO-FIELD OUTPUT						
Initial Calibration Tolerance		-2000		+2000	LSB	2
OTHER						
Output Data Rate				100	Hz	1

Table 3. Magnetometer Specifications

NOTES:

1. Guaranteed by design.
2. Derived from validation or characterization of parts, not guaranteed in production.

3.4 ELECTRICAL SPECIFICATIONS

D.C. Electrical Characteristics

Typical Operating Circuit of section 4.2, VDD = 1.8V, VDDIO = 1.8V, T_A=25°C, unless otherwise noted.

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
SUPPLY VOLTAGES						
VDD		1.71	1.8	3.6	V	1
VDDIO		1.71	1.8	1.95	V	1
SUPPLY CURRENTS						
9-Axis (DMP disabled)	Low-Noise Mode; Compass in Continuous Mode		3.11		mA	2
Gyroscope Only (DMP, Barometer & Accelerometer disabled)	Low-Power Mode, 102.3 Hz update rate, 1x averaging filter		1.23		mA	2
Accelerometer Only (DMP, Barometer & Gyroscope disabled)	Low-Power Mode, 102.3 Hz update rate, 1x averaging filter		68.9		µA	2
Magnetometer Only (DMP, Accelerometer & Gyroscope disabled)	8 Hz update rate		90		µA	2
Full-Chip Sleep Mode			8		µA	2
TEMPERATURE RANGE						
Specified Temperature Range	Performance parameters are not applicable beyond Specified Temperature Range	-40		+85	°C	1, 3

Table 4. D.C. Electrical Characteristics

NOTES:

1. Guaranteed by design.
2. Derived from validation or characterization of parts, not guaranteed in production.
3. Barometer Specified Temperature Range is -30°C to +85°C

A.C. Electrical Characteristics

Typical Operating Circuit of section 4.2, VDD = 1.8V, VDDIO = 1.8V, T_A=25°C, unless otherwise noted.

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
SUPPLIES						
Supply Ramp Time (T_{RAMP})	Monotonic ramp. Ramp rate is 10% to 90% of the final value.	0.01	20	100	ms	1
TEMPERATURE SENSOR						
Operating Range	Ambient	-40		85	°C	1
Sensitivity	Untrimmed		333.87		LSB/°C	
Room Temp Offset	21°C		0		LSB	
POWER-ON RESET						
Supply Ramp Time (T_{RAMP})	Valid power-on RESET	0.01	20	100	ms	1
Start-up time for register read/write	From power-up		11	100	ms	1
I²C ADDRESS	AD0 = 0 AD0 = 1		1101000 1101001			
DIGITAL INPUTS (FSYNC, AD0, SCLK, SDI, CS)						
V_{IH}, High Level Input Voltage		0.7*VDDIO			V	1
V_{IL}, Low Level Input Voltage				0.3*VDDIO	V	
C_I, Input Capacitance			< 10		pF	
DIGITAL OUTPUT (SDO, INT)						
V_{OH}, High Level Output Voltage	R_{LOAD}=1 MΩ;	0.9*VDDIO			V	1
V_{OL1}, LOW-Level Output Voltage	R_{LOAD}=1 MΩ;			0.1*VDDIO	V	
$V_{OL.INT1}$, INT Low-Level Output Voltage	OPEN=1, 0.3 mA sink Current			0.1	V	
Output Leakage Current	OPEN=1		100		nA	
t_{INT}, INT Pulse Width	LATCH_INT_EN=0		50		µs	
I²C I/O (SCL, SDA)						
V_{IL}, LOW Level Input Voltage		-0.5V		0.3*VDDIO	V	1
V_{IH}, HIGH-Level Input Voltage		0.7*VDDIO		VDDIO + 0.5V	V	
V_{hys}, Hysteresis			0.1*VDDIO		V	
V_{OL}, LOW-Level Output Voltage	3 mA sink current	0		0.4	V	
I_{OL}, LOW-Level Output Current	V_{OL}=0.4V V_{OL}=0.6V		3 6		mA mA	
Output Leakage Current			100		nA	
t_{of}, Output Fall Time from V_{IHmax} to V_{ILmax}	C_b bus capacitance in pf	20+0.1C_b		250	ns	
AUXILLIARY I/O (AUX_CL, AUX_DA)						
V_{IL}, LOW-Level Input Voltage		-0.5V		0.3*VDDIO	V	1
V_{IH}, HIGH-Level Input Voltage		0.7* VDDIO		VDDIO + 0.5V	V	
V_{hys}, Hysteresis			0.1* VDDIO		V	
V_{OL1}, LOW-Level Output Voltage	VDDIO > 2V; 1 mA sink current	0		0.4	V	
V_{OL3}, LOW-Level Output Voltage	VDDIO < 2V; 1 mA sink current	0		0.2* VDDIO	V	
I_{OL}, LOW-Level Output Current	V_{OL} = 0.4V V_{OL} = 0.6V		3 6		mA mA	
Output Leakage Current			100		nA	
t_{of}, Output Fall Time from V_{IHmax} to V_{ILmax}	C_b bus capacitance in pF	20+0.1C_b		250	ns	

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
INTERNAL CLOCK SOURCE						
Clock Frequency Initial Tolerance	Accelerometer Only Mode	-5		+5	%	1
	Gyroscope or 6-Axis Mode WITHOUT Timebase Correction	-9		+9	%	1
	Gyroscope or 6-Axis Mode WITH Timebase Correction	-1		+1		
Frequency Variation over Temperature	Accelerometer Only Mode	-10		+10	%	1
	Gyroscope or 6-Axis Mode		±1		%	1

Table 5. A.C. Electrical Characteristics

NOTES:

1. Derived from validation or characterization of parts, not guaranteed in production.

Other Electrical Specifications

Typical Operating Circuit of section 4.2, VDD = 1.8V, VDDIO = 1.8V, T_A=25°C, unless otherwise noted.

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS	NOTES
SERIAL INTERFACE						
SPI Operating Frequency, All Registers Read/Write	Low Speed Characterization		100 ±10%		kHz	
	High Speed Characterization		7 ±10%		MHz	
I^2C Operating Frequency	All registers, Fast-mode			400	kHz	
	All registers, Standard-mode			100	kHz	

Table 6. Other Electrical Specifications

NOTES:

1. Derived from validation or characterization of parts, not guaranteed in production.

3.5 I²C TIMING CHARACTERIZATION

Typical Operating Circuit of section 4.2, VDD = 1.8V, VDDIO = 1.8V, T_A=25°C, unless otherwise noted.

PARAMETERS	CONDITIONS	MIN	TYPICAL	MAX	UNITS	NOTES
I²C TIMING	**I²C FAST-MODE**					
f_{SCL}, SCL Clock Frequency				400	kHz	1, 2
$t_{HD.STA}$, (Repeated) START Condition Hold Time		0.6			µs	1, 2
t_{LOW}, SCL Low Period		1.3			µs	1, 2
t_{HIGH}, SCL High Period		0.6			µs	1, 2
$t_{SU.STA}$, Repeated START Condition Setup Time		0.6			µs	1, 2
$t_{HD.DAT}$, SDA Data Hold Time		0			µs	1, 2
$t_{SU.DAT}$, SDA Data Setup Time		100			ns	1, 2
t_r, SDA and SCL Rise Time	C_b bus cap. from 10 to 400 pF	20+0.1C_b		300	ns	1, 2
t_f, SDA and SCL Fall Time	C_b bus cap. from 10 to 400 pF	20+0.1C_b		300	ns	1, 2
$t_{SU.STO}$, STOP Condition Setup Time		0.6			µs	1, 2
t_{BUF}, Bus Free Time Between STOP and START Condition		1.3			µs	1, 2
C_b, Capacitive Load for each Bus Line			< 400		pF	1, 2
$t_{VD.DAT}$, Data Valid Time				0.9	µs	1, 2
$t_{VD.ACK}$, Data Valid Acknowledge Time				0.9	µs	1, 2

Table 7. I²C Timing Characteristics

NOTES:

1. Timing Characteristics apply to both Primary and Auxiliary I²C Bus.
2. Based on characterization of 5 parts over temperature and voltage as mounted on evaluation board or in sockets.

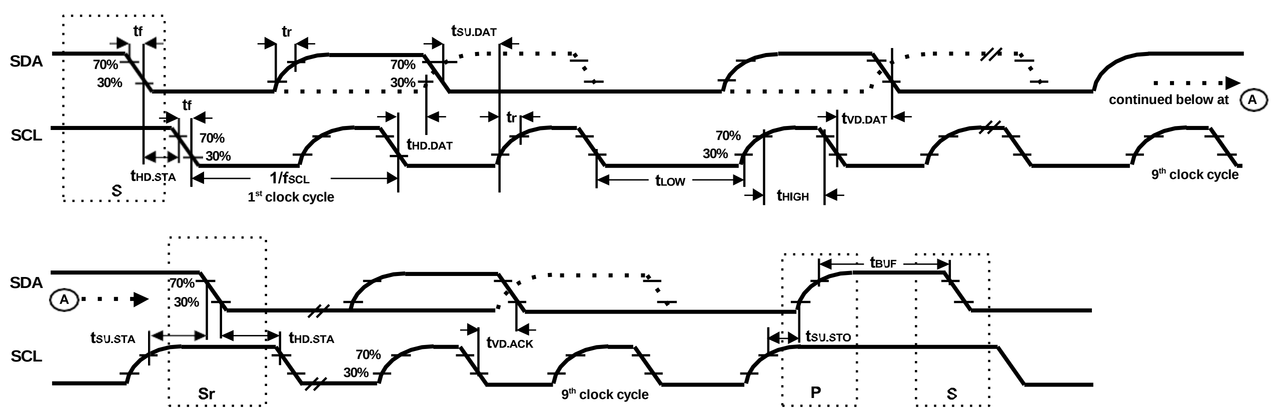


Figure 1. I²C Bus Timing Diagram

3.6 SPI TIMING CHARACTERIZATION

Typical Operating Circuit of section 4.2, VDD = 1.8V, VDDIO = 1.8V, T_A=25°C, unless otherwise noted.

PARAMETERS	CONDITIONS	MIN	TYPICAL	MAX	UNITS	NOTES
SPI TIMING						
f_{SCLK}, SCLK Clock Frequency				7	MHz	
t_{LOW}, SCLK Low Period		64			ns	
t_{HIGH}, SCLK High Period		64			ns	
$t_{SU.CS}$, CS Setup Time		8			ns	
$t_{HD.CS}$, CS Hold Time		500			ns	
$t_{SU.SDI}$, SDI Setup Time		5			ns	
$t_{HD.SDI}$, SDI Hold Time		7			ns	
$t_{VD.SDO}$, SDO Valid Time	C_{load} = 20 pF			59	ns	
$t_{HD.SDO}$, SDO Hold Time	C_{load} = 20 pF	6			ns	
$t_{DIS.SDO}$, SDO Output Disable Time				50	ns	

Table 8. SPI Timing Characteristics (7 MHz)

NOTES:

1. Based on characterization of 5 parts over temperature and voltage as mounted on evaluation board or in sockets

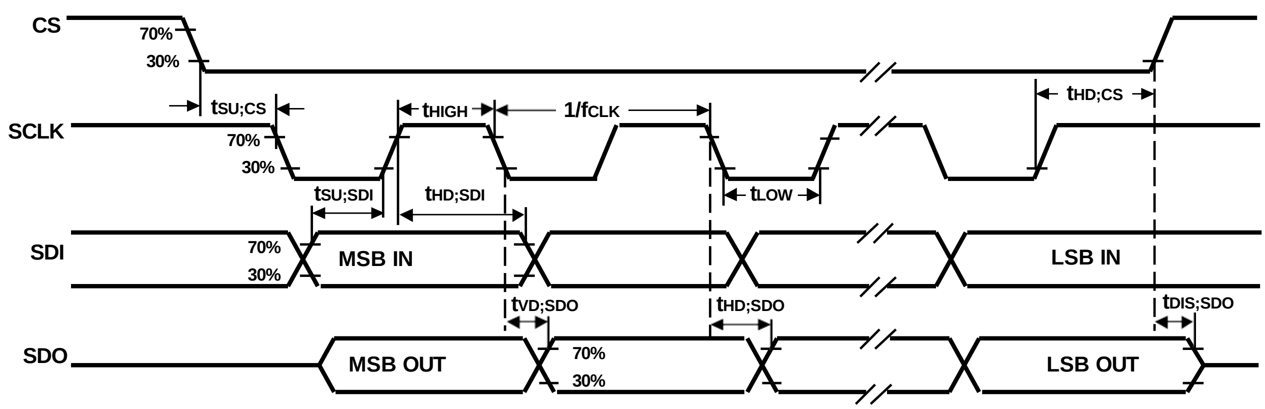


Figure 2. SPI Bus Timing Diagram

3.7 ABSOLUTE MAXIMUM RATINGS

Stress above those listed as "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these conditions is not implied. Exposure to the absolute maximum ratings conditions for extended periods may affect device reliability.

PARAMETER	RATING
Supply Voltage, VDD	-0.5V to +4V
Supply Voltage, VDDIO	-0.3V to +2.5V
REGOUT	-0.5V to 2V
Input Voltage Level (AUX_DA, AD0, FSYNC, INT, SCL, SDA)	-0.5V to VDD + 0.5V
Acceleration (Any Axis, unpowered)	20,000*g* for 0.2 ms
Operating Temperature Range	-40°C to +105°C (Compass: -30°C to +85°C)
Storage Temperature Range	-40°C to +125°C
Electrostatic Discharge (ESD) Protection	2kV (HBM); 200V (MM)
Latch-up	JEDEC Class II (2),125°C ±100 mA

Table 9. Absolute Maximum Ratings

4 APPLICATIONS INFORMATION

4.1 PIN OUT DIAGRAM AND SIGNAL DESCRIPTION

PIN NUMBER	PIN NAME	PIN DESCRIPTION
7	AUX_CL	I^2C Master serial clock, for connecting to external sensors
8	VDDIO	Digital I/O supply voltage
9	AD0 / SDO	I^2C Slave Address LSB (AD0); SPI serial data output (SDO)
10	REGOUT	Regulator filter capacitor connection
11	FSYNC	Frame synchronization digital input. Connect to GND if unused
12	INT1	Interrupt 1
13	VDD	Power supply voltage
18	GND	Power supply ground
19	RESV	Reserved. Do not connect.
20	RESV	Reserved. Connect to GND.
21	AUX_DA	I^2C master serial data, for connecting to external sensors
22	nCS	Chip select (SPI mode only)
23	SCL / SCLK	I^2C serial clock (SCL); SPI serial clock (SCLK)
24	SDA / SDI	I^2C serial data (SDA); SPI serial data input (SDI)
1 – 6, 14 - 17	NC	Do not connect

Table 10. Signal Descriptions

NOTE: Power up with SCL/SCLK and nCS pins held low is not a supported use case. In case this power up approach is used, software reset is required using the PWR_MGMT_1 register, prior to initialization.

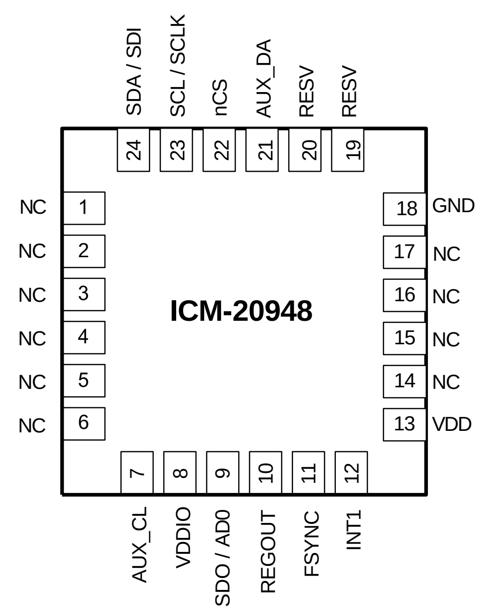
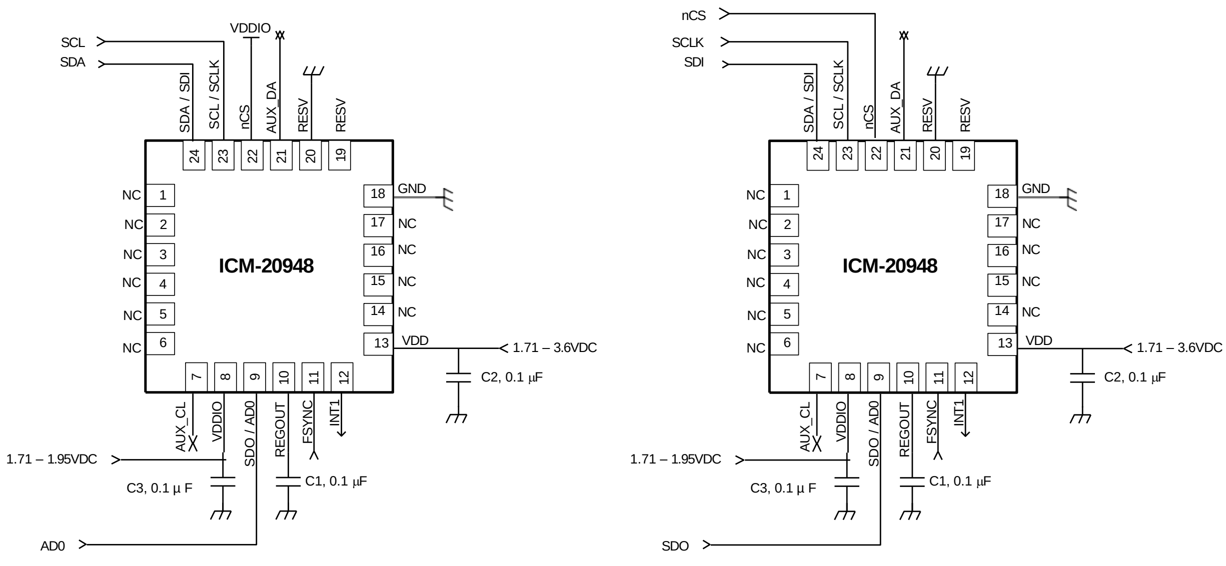


Figure 3. Pin out Diagram for ICM-20948 3 mm x 3 mm x 1 mm QFN

4.2 TYPICAL OPERATING CIRCUIT



(a) (b)

Figure 4. ICM-20948 Application Schematic (a) I²C operation (b) SPI operation

Note that the INT pin should be connected to a GPIO pin on the system processor that is capable of waking the system processor from suspend mode.

I²C lines are open drain and pullup resistors (e.g. 10 kΩ) are required.

4.3 BILL OF MATERIALS FOR EXTERNAL COMPONENTS

COMPONENT	LABEL	SPECIFICATION	QUANTITY
Regulator Filter Capacitor	C1	Ceramic, X7R, 0.1 µF ±10%, 2V	1
VDD Bypass Capacitor	C2	Ceramic, X7R, 0.1 µF ±10%, 4V	1
VDDIO Bypass Capacitor	C3	Ceramic, X7R, 0.1 µF ±10%, 4V	1

Table 11. Bill of Materials

4.4 EXPOSED DIE PAD PRECAUTIONS

InvenSense products have very low active and standby current consumption. The exposed die pad is not required for heat sinking, and should not be soldered to the PCB. Failure to adhere to this rule can induce performance changes due to package thermo-mechanical stress. There is no electrical connection between the pad and the CMOS.

4.5 BLOCK DIAGRAM

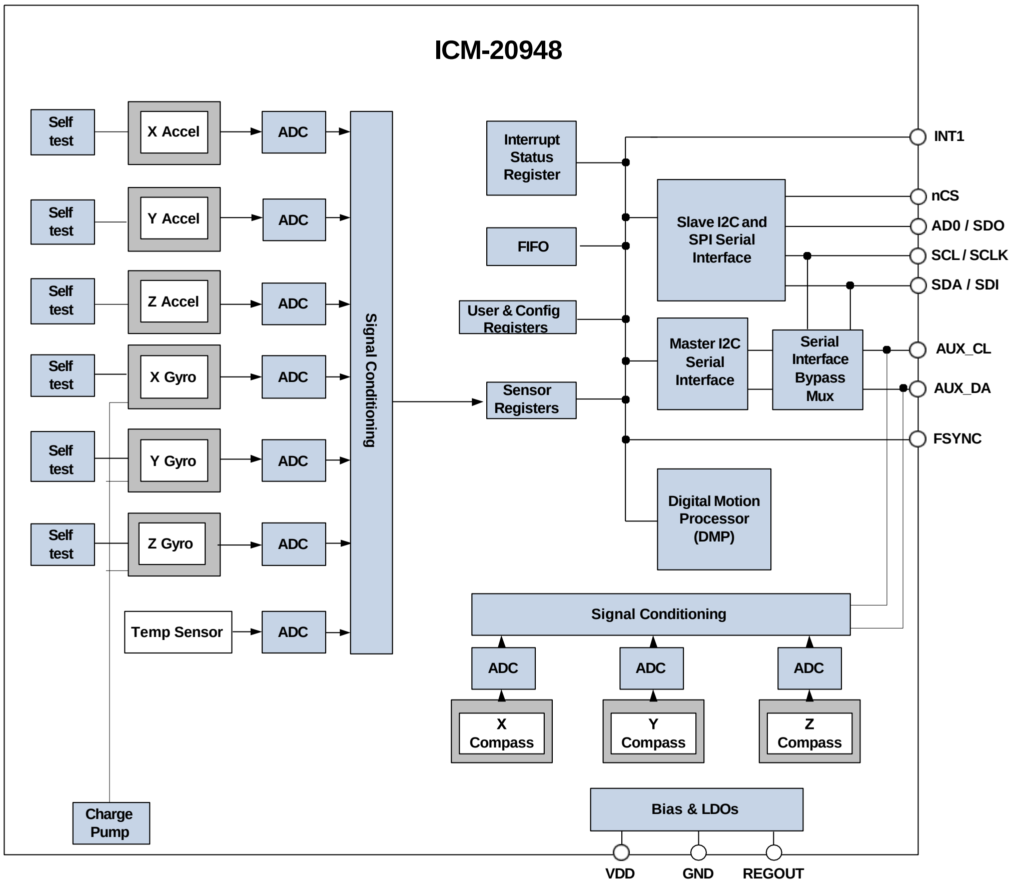


Figure 5. ICM-20948 Block Diagram

4.6 OVERVIEW

The ICM-20948 is comprised of the following key blocks and functions:

- Three-axis MEMS rate gyroscope sensor with 16-bit ADCs and signal conditioning
- Three-axis MEMS accelerometer sensor with 16-bit ADCs and signal conditioning
- Three-axis MEMS magnetometer sensor with 16-bit ADCs and signal conditioning
- Digital Motion Processor (DMP) engine
- Primary I^{2}C and SPI serial communications interfaces
- Auxiliary I^{2}C serial interface
- Gyroscope, Accelerometer, and Magnetometer Self-Test
- Clocking
- Sensor Data Registers
- FIFO
- FSYNC
- Interrupts
- Digital-Output Temperature Sensor
- Bias and LDOs
- Charge Pump
- Power Modes

4.7 THREE-AXIS MEMS GYROSCOPE WITH 16-BIT ADCS AND SIGNAL CONDITIONING

The ICM-20948 consists of three independent vibratory MEMS rate gyroscopes, which detect rotation about the X-, Y-, and Z-Axes. When the gyros are rotated about any of the sense axes, the Coriolis Effect causes a vibration that is detected by a capacitive pickoff. The resulting signal is amplified, demodulated, and filtered to produce a voltage that is proportional to the angular rate. This voltage is digitized using individual on-chip 16-bit Analog-to-Digital Converters (ADCs) to sample each axis. The full-scale range of the gyro sensors may be digitally programmed to ±250, ±500, ±1000, or ±2000 degrees per second (dps).

4.8 THREE-AXIS MEMS ACCELEROMETER WITH 16-BIT ADCS AND SIGNAL CONDITIONING

The ICM-20948's 3-Axis accelerometer uses separate proof masses for each axis. Acceleration along a particular axis induces displacement on the corresponding proof mass, and capacitive sensors detect the displacement differentially. The ICM-20948's architecture reduces the accelerometers' susceptibility to fabrication variations as well as to thermal drift. When the device is placed on a flat surface, it will measure 0*g* on the X- and Y-axes and +1*g* on the Z-axis. The accelerometers' scale factor is calibrated at the factory and is nominally independent of supply voltage. Each sensor has a dedicated sigma-delta ADC for providing digital outputs. The full scale range of the digital output can be adjusted to ±2*g*, ±4*g*, ±8*g*, or ±16*g*.

4.9 THREE-AXIS MEMS MAGNETOMETER WITH 16-BIT ADCS AND SIGNAL CONDITIONING

The 3-axis magnetometer uses highly sensitive Hall sensor technology. The magnetometer portion of the IC incorporates magnetic sensors for detecting terrestrial magnetism in the X-, Y-, and Z-Axes, a sensor driving circuit, a signal amplifier chain, and an arithmetic circuit for processing the signal from each sensor. Each ADC has a 16-bit resolution and a full scale range of ±4900 µT.

4.10 DIGITAL MOTION PROCESSOR

The embedded Digital Motion Processor (DMP) within the ICM-20948 offloads computation of motion processing algorithms from the host processor. The DMP acquires data from accelerometers, gyroscopes, and additional third party sensors such as magnetometers, and processes the data. The resulting data can be read from the FIFO. The DMP has access to the external pins, which can be used for generating interrupts.

The purpose of the DMP is to offload both timing requirements and processing power from the host processor. Typically, motion processing algorithms should be run at a high rate, often around 200 Hz, in order to provide accurate results with low latency. This is required even if the application updates at a much lower rate; for example, a low power user interface may update as slowly as 5 Hz, but the motion processing should still run at 200 Hz. The DMP can be used to minimize power, simplify timing, simplify the software architecture, and save valuable MIPS on the host processor for use in applications.

4.11 PRIMARY I^{2}C AND SPI SERIAL COMMUNICATIONS INTERFACES

The ICM-20948 communicates to a system processor using either a SPI or an I^{2}C serial interface. The ICM-20948 always acts as a slave when communicating to the system processor. The LSB of the of the I^{2}C slave address is set by pin 1 (AD0).

ICM-20948 Solution Using I^{2}C Interface

In Figure 6, the system processor is an I^{2}C master to the ICM-20948. In addition, the ICM-20948 is an I^{2}C master to the optional external sensor. The ICM-20948 has limited capabilities as an I^{2}C Master, and depends on the system processor to manage the initial configuration of any auxiliary sensors. The ICM-20948 has an interface bypass multiplexer, which connects the system processor I^{2}C bus pins 23 and 24 (SCL and SDA) directly to the auxiliary sensor I^{2}C bus pins 7 and 21 (AUX_CL and AUX_DA).

Once the auxiliary sensors have been configured by the system processor, the interface bypass multiplexer should be disabled so that the ICM-20948 auxiliary I²C master can take control of the sensor I²C bus and gather data from the auxiliary sensors.

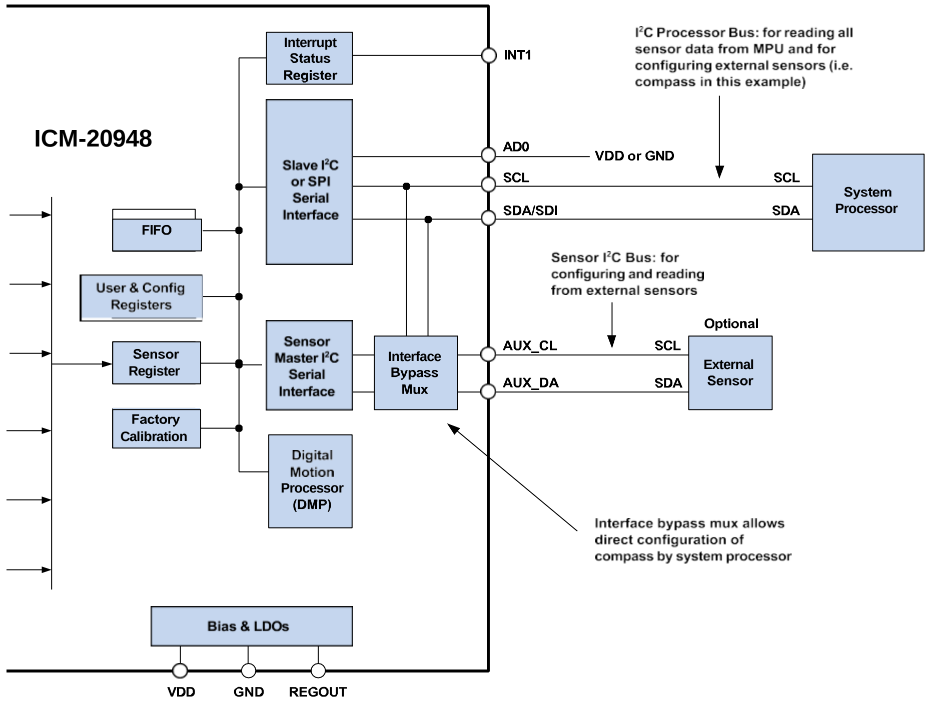


Figure 6. ICM-20948 Solution Using I²C Interface

ICM-20948 Solution Using SPI Interface

In Figure 7, the system processor is an SPI master to the ICM-20948. Pins 9, 22, 23, and 24 are used to support the SDO, nCS, SCLK, and SDI signals for SPI communications. Because these SPI pins are shared with the I²C slave pins (9, 23 and 24), the system processor cannot access the auxiliary I²C bus through the interface bypass multiplexer, which connects the processor I²C interface pins to the sensor I²C interface pins. Since the ICM-20948 has limited capabilities as an I²C Master, and depends on the system processor to manage the initial configuration of any auxiliary sensors, another method must be used for programming the sensors on the auxiliary sensor I²C bus pins 7 and 21 (AUX_CL and AUX_DA).

When using SPI communications between the ICM-20948 and the system processor, configuration of devices on the auxiliary I²C sensor bus can be achieved by using I²C Slaves 0-4 to perform read and write transactions on any device and register on the auxiliary I²C bus. The I²C Slave 4 interface can be used to perform only single byte read and write transactions. Once the external sensors have been configured, the ICM-20948 can perform single or multi-byte reads using the sensor I²C bus. The read results from the Slave 0-3 controllers can be written to the FIFO buffer as well as to the external sensor registers.

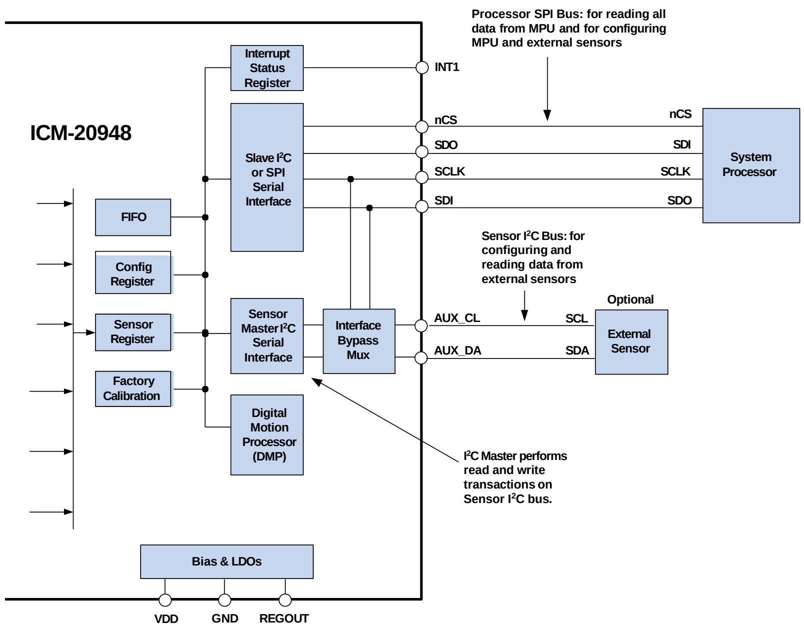


Figure 7. ICM-20948 Solution Using SPI Interface

4.12 AUXILIARY I²C SERIAL INTERFACE

The ICM-20948 has an auxiliary I²C bus for communicating to external sensors. This bus has two operating modes:

- I²C Master Mode: The ICM-20948 acts as a master to any external sensors connected to the auxiliary I²C bus
- Pass-Through Mode: The ICM-20948 directly connects the primary and auxiliary I²C buses together, allowing the system processor to directly communicate with any external sensors.

Auxiliary I²C Bus Modes of Operation:

- I²C Master Mode: Allows the ICM-20948 to directly access the data registers of external sensors. In this mode, the ICM-20948 directly obtains data from auxiliary sensors without intervention from the system applications processor. The I²C Master can be configured to read up to 24 bytes from up to 4 auxiliary sensors. A fifth sensor can be configured to work single byte read/write mode.
- Pass-Through Mode: Allows an external system processor to act as master and directly communicate to the external sensors connected to the auxiliary I²C bus pins (AUX_DA and AUX_CL). In this mode, the auxiliary I²C bus control logic of the ICM-20948 is disabled, and the auxiliary I²C pins AUX_CL and AUX_DA (pins 7 and 21) are connected to the main I²C bus (Pins 23 and 24) through analog switches internally. Pass-Through mode is useful for configuring the external sensors.

4.13 SELF-TEST

Self-test allows for the testing of the mechanical and electrical portions of the sensors. The self-test for each measurement axis can be activated by means of the gyroscope and accelerometer self-test registers.

When the self-test is activated, the electronics cause the sensors to be actuated and produce an output signal. The output signal is used to observe the self-test response.

The self-test response is defined as follows:

SELF-TEST RESPONSE = SENSOR OUTPUT WITH SELF-TEST ENABLED – SENSOR OUTPUT WITHOUT SELF-TEST ENABLED

The self-test response for each gyroscope axis is defined in the gyroscope specification table, while that for each accelerometer axis is defined in the accelerometer specification table.

When the value of the self-test response is within the specified min/max limits, the part has passed self-test. When the self-test response exceeds the min/max values, the part is deemed to have failed self-test. It is recommended to use InvenSense MotionApps software for executing self-test.

4.14 CLOCKING

The internal system clock sources include: (1) an internal relaxation oscillator, and (2) a PLL with MEMS gyroscope oscillator as the reference clock. With the recommended clock selection setting (CLKSEL = 1), the best clock source for optimum sensor performance and power consumption will be automatically selected based on the power mode. Specifically, the internal relaxation oscillator will be selected when operating in accelerometer only mode, while the PLL will be selected whenever gyroscope is on, which includes gyroscope and 6-axis modes.

As clock accuracy is critical to the preciseness of distance and angle calculations performed by DMP, it should be noted that the internal relaxation oscillator and PLL show different performances in some aspects. The internal relaxation oscillator is trimmed to have a consistent operating frequency at room temperature, while the PLL clock frequency varies from part to part. The PLL frequency deviation from the nominal value in percentage is captured in register TIMEBASE_CORRECTION_PLL (detailed in section 12.5), and users can factor it in during distance and angle calculations to not sacrifice accuracy. Other than that, PLL has better frequency stability and lower frequency variation over temperature than the internal relaxation oscillator.

4.15 SENSOR DATA REGISTERS

The sensor data registers contain the latest gyro, accelerometer, auxiliary sensor, and temperature measurement data. They are read-only registers, and are accessed via the serial interface. Data from these registers may be read anytime.

4.16 FIFO

The ICM-20948 contains a FIFO of size 4kBytes (FIFO size will vary depending on DMP feature-set) that is accessible via the Serial Interface. The FIFO configuration register determines which data is written into the FIFO. Possible choices include gyro data, accelerometer data, temperature readings, auxiliary sensor readings, and FSYNC input.

A FIFO counter keeps track of how many bytes of valid data are contained in the FIFO. The FIFO register supports burst reads. The interrupt function may be used to determine when new data is available.

For further information regarding the FIFO, please refer to the Section 7.

4.17 FSYNC

The FSYNC pin can be used from an external interrupt source to wake up the device from sleep. It is particularly useful in EIS applications to synchronize the gyroscope ODR with external inputs from an imaging sensor. Connecting the VSYNC or HSYNC pin of the image sensor subsystem to FSYNC on ICM-20948 allows timing synchronization between the two otherwise unconnected subsystems.

An FSYNC_ODR delay time register is used to capture the delay between an FSYNC pulse and the very next gyroscope data ready pulse.

4.18 INTERRUPTS

Interrupt functionality is configured via the Interrupt Configuration register. Items that are configurable include the INT pin configuration, the interrupt latching and clearing method, and triggers for the interrupt. Section 5 provides a summary of interrupt sources. The interrupt status can be read from the Interrupt Status register.

For further information regarding interrupts, please refer to Section 7.

4.19 DIGITAL-OUTPUT TEMPERATURE SENSOR

An on-chip temperature sensor and ADC are used to measure the ICM-20948 die temperature. The readings from the ADC can be read from the FIFO or the Sensor Data registers.

4.20 BIAS AND LDOS

The bias and LDO section generates the internal supply and the reference voltages and currents required by the ICM-20948. Its two inputs are an unregulated VDD and a VDDIO logic reference supply voltage. The LDO output is bypassed by a capacitor at REGOUT. For further details on the capacitor, please refer to the Bill of Materials for External Components.

4.21 CHARGE PUMP

An on-chip charge pump generates the high voltage required for the MEMS oscillators.

4.22 POWER MODES

Table 12 lists the user-accessible power modes for ICM-20948.

MODE	NAME	GYRO	ACCEL	MAGNETOMETER	DMP
1	Sleep Mode	Off	Off	Off	Off
2	Low-Power Accelerometer Mode	Off	Duty-Cycled	Off	On or Off
3	Low-Noise Accelerometer Mode	Off	On	Off	On or Off
4	Gyroscope Mode	On	Off	Off	On or Off
5	Magnetometer Mode	Off	Off	On	On or Off
6	Accel + Gyro Mode	On	On	Off	On or Off
7	Accel + Magnetometer Mode	Off	On	On	On or Off
8	9-Axis Mode	On	On	On	On or Off

Table 12. Power Modes for ICM-20948

5 PROGRAMMABLE INTERRUPTS

The ICM-20948 has a programmable interrupt system which can generate an interrupt signal on the INT pin. Status flags indicate the source of an interrupt. Interrupt sources may be enabled and disabled individually. Table 13 lists the interrupt sources.

INTERRUPT SOURCE
DMP Interrupt
Wake on Motion Interrupt
PLL RDY Interrupt
I2C Master Interrupt
Raw Data Ready Interrupt
FIFO Overflow Interrupt
FIFO Watermark Interrupt

Table 13. Interrupt Sources

6 DIGITAL INTERFACE

6.1 I²C AND SPI SERIAL INTERFACES

The internal registers and memory of the ICM-20948 can be accessed using either I²C at 400 kHz or SPI at 7 MHz. SPI operates in four-wire mode.

PIN NUMBER	PIN NAME	PIN DESCRIPTION
9	AD0 / SDO	I²C Slave Address LSB (AD0); SPI serial data output (SDO)
22	nCS	Chip select (SPI mode only)
23	SCL / SCLK	I²C serial clock (SCL); SPI serial clock (SCLK)
24	SDA / SDI	I²C serial data (SDA); SPI serial data input (SDI)

Table 14. Serial Interface

NOTE: To prevent switching into I²C mode when using SPI, the I²C interface should be disabled by setting the *I2C_IF_DIS* configuration bit. Setting this bit should be performed immediately after waiting for the time specified by the "Start-Up Time for Register Read/Write" in Section 6.3.

For further information regarding the *I2C_IF_DIS* bit, please refer to Section 7.

6.2 I²C INTERFACE

I²C is a two-wire interface comprised of the signals serial data (SDA) and serial clock (SCL). In general, the lines are open-drain and bi-directional. In a generalized I²C interface implementation, attached devices can be a master or a slave. The master device puts the slave address on the bus, and the slave device with the matching address acknowledges the master.

The ICM-20948 always operates as a slave device when communicating to the system processor, which thus acts as the master. SDA and SCL lines typically need pull-up resistors to VDD. The maximum bus speed is 400 kHz.

The slave address of the ICM-20948 is b110100X which is 7 bits long. The LSB bit of the 7-bit address is determined by the logic level on pin AD0. This allows two ICM-20948s to be connected to the same I²C bus. When used in this configuration, the address of the one of the devices should be b1101000 (pin AD0 is logic low) and the address of the other should be b1101001 (pin AD0 is logic high).

6.3 I²C COMMUNICATIONS PROTOCOL

START (S) and STOP (P) Conditions

Communication on the I²C bus starts when the master puts the START condition (S) on the bus, which is defined as a HIGH-to-LOW transition of the SDA line while SCL line is HIGH (see figure below). The bus is considered to be busy until the master puts a STOP condition (P) on the bus, which is defined as a LOW to HIGH transition on the SDA line while SCL is HIGH (see figure below).

Additionally, the bus remains busy if a repeated START (Sr) is generated instead of a STOP condition.

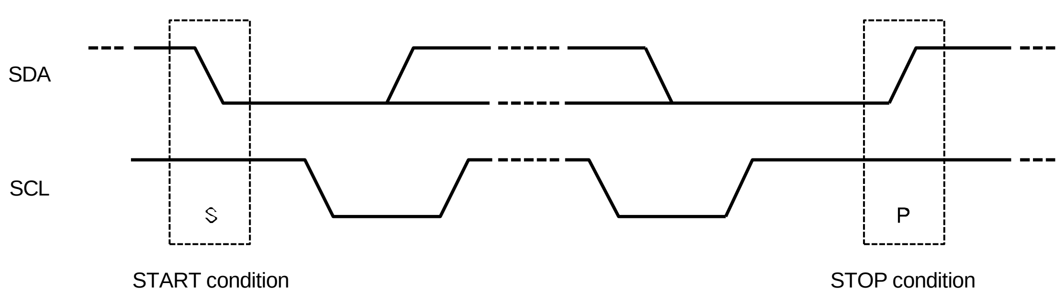


Figure 8. START and STOP Conditions

Data Format / Acknowledge

I^{2}C data bytes are defined to be 8-bits long. There is no restriction to the number of bytes transmitted per data transfer. Each byte transferred must be followed by an acknowledge (ACK) signal. The clock for the acknowledge signal is generated by the master, while the receiver generates the actual acknowledge signal by pulling down SDA and holding it low during the HIGH portion of the acknowledge clock pulse.

If a slave is busy and cannot transmit or receive another byte of data until some other task has been performed, it can hold SCL LOW, thus forcing the master into a wait state. Normal data transfer resumes when the slave is ready, and releases the clock line (refer to the following figure).

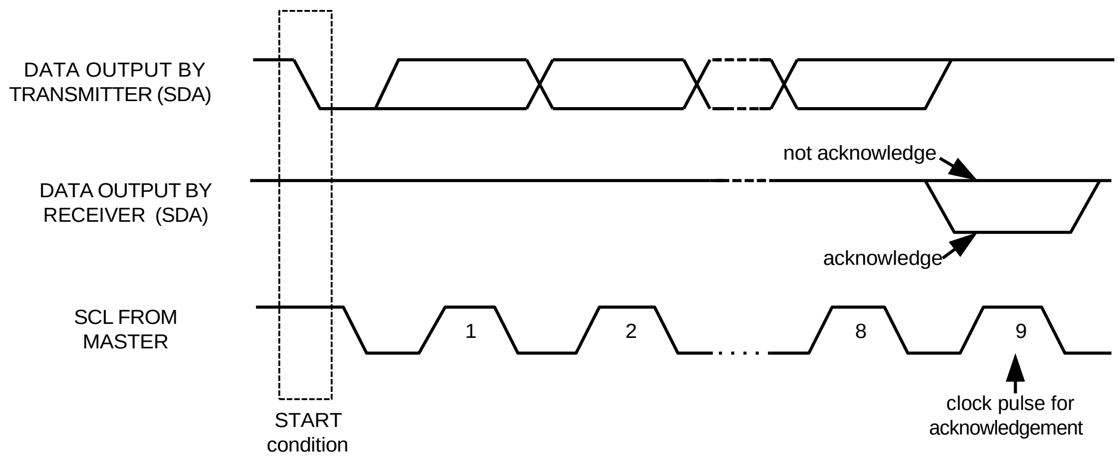


Figure 9. Acknowledge on the I^{2}C Bus

Communications

After beginning communications with the START condition (S), the master sends a 7-bit slave address followed by an 8th bit, the read/write bit. The read/write bit indicates whether the master is receiving data from or is writing to the slave device. Then, the master releases the SDA line and waits for the acknowledge signal (ACK) from the slave device. Each byte transferred must be followed by an acknowledge bit. To acknowledge, the slave device pulls the SDA line LOW and keeps it LOW for the high period of the SCL line. Data transmission is always terminated by the master with a STOP condition (P), thus freeing the communications line. However, the master can generate a repeated START condition (Sr), and address another slave without first generating a STOP condition (P). A LOW to HIGH transition on the SDA line while SCL is HIGH defines the stop condition. All SDA changes should take place when SCL is low, with the exception of start and stop conditions.

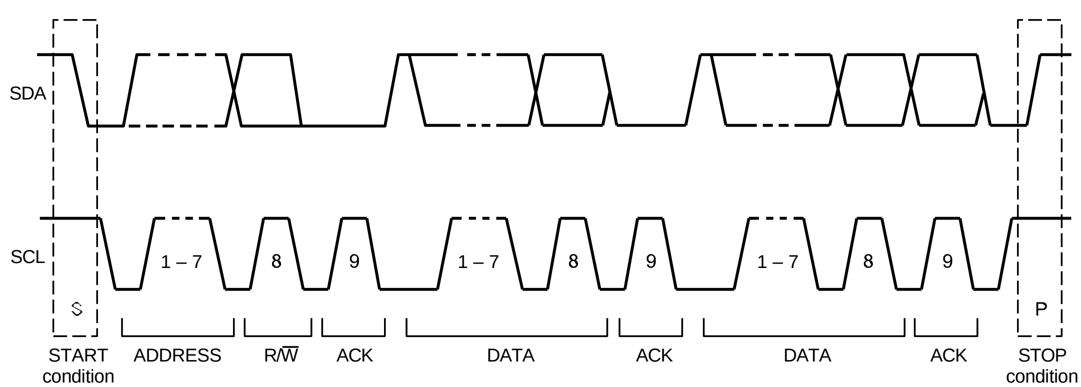


Figure 10. Complete I^{2}C Data Transfer

To write the internal ICM-20948 registers, the master transmits the start condition (S), followed by the I²C address and the write bit (0). At the 9th clock cycle (when the clock is high), the ICM-20948 acknowledges the transfer. Then the master puts the register address (RA) on the bus. After the ICM-20948 acknowledges the reception of the register address, the master puts the register data onto the bus. This is followed by the ACK signal, and data transfer may be concluded by the stop condition (P). To write multiple bytes after the last ACK signal, the master can continue outputting data rather than transmitting a stop signal. In this case, the ICM-20948 automatically increments the register address and loads the data to the appropriate register. The following figures show single and two-byte write sequences.

Single-Byte Write Sequence

Master	S	AD+W		RA		DATA		P
Slave			ACK		ACK		ACK	

Burst Write Sequence

Master	S	AD+W		RA		DATA		DATA		P
Slave			ACK		ACK		ACK		ACK	

To read the internal ICM-20948 registers, the master sends a start condition, followed by the I²C address and a write bit, and then the register address that is going to be read. Upon receiving the ACK signal from the ICM-20948, the master transmits a start signal followed by the slave address and read bit. As a result, the ICM-20948 sends an ACK signal and the data. The communication ends with a not acknowledge (NACK) signal and a stop bit from master. The NACK condition is defined such that the SDA line remains high at the 9th clock cycle. The following figures show single and two-byte read sequences.

Single-Byte Read Sequence

Master	S	AD+W		RA		S	AD+R			NACK	P
Slave			ACK		ACK			ACK	DATA		

Burst Read Sequence

Master	S	AD+W		RA		S	AD+R			ACK		NACK	P
Slave			ACK		ACK			ACK	DATA		DATA		

6.4 I²C TERMS

SIGNAL	DESCRIPTION
S	Start Condition: SDA goes from high to low while SCL is high
AD	Slave I²C address
W	Write bit (0)
R	Read bit (1)
ACK	Acknowledge: SDA line is low while the SCL line is high at the 9th clock cycle
NACK	Not-Acknowledge: SDA line stays high at the 9th clock cycle
RA	ICM-20948 internal register address
DATA	Transmit or received data
P	Stop condition: SDA going from low to high while SCL is high

Table 15. I²C Terms

6.5 SPI INTERFACE

SPI is a 4-wire synchronous serial interface that uses two control lines and two data lines. The ICM-20948 always operates as a Slave device during standard Master-Slave SPI operation.

With respect to the Master, the Serial Clock output (SCLK), the Serial Data Output (SDO) and the Serial Data Input (SDI) are shared among the Slave devices. Each SPI slave device requires its own Chip Select (CS) line from the master.

CS goes low (active) at the start of transmission and goes back high (inactive) at the end. Only one CS line is active at a time, ensuring that only one slave is selected at any given time. The CS lines of the non-selected slave devices are held high, causing their SDO lines to remain in a high-impedance (high-z) state so that they do not interfere with any active devices.

SPI Operational Features

1. Data is delivered MSB first and LSB last
2. Data is latched on the rising edge of SCLK
3. Data should be transitioned on the falling edge of SCLK
4. The maximum frequency of SCLK is 7MHz
5. SPI read and write operations are completed in 16 or more clock cycles (two or more bytes). The first byte contains the SPI Address, and the following byte(s) contain(s) the SPI data. The first bit of the first byte contains the Read/Write bit and indicates the Read (1) or Write (0) operation. The following 7 bits contain the Register Address. In cases of multiple-byte Read/Writes, data is two or more bytes:

SPI Address format

MSB							**LSB**
R/W	A6	A5	A4	A3	A2	A1	A0

SPI Data format

MSB							**LSB**
D7	D6	D5	D4	D3	D2	D1	D0

6. Supports Single or Burst Read/Writes.

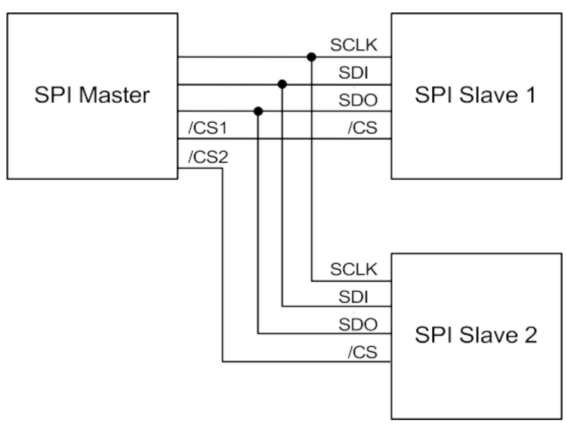


Figure 11. Typical SPI Master / Slave Configuration

7 REGISTER MAP FOR GYROSCOPE AND ACCELEROMETER

The following table lists the register map for the ICM-20948, for user banks 0, 1, 2, 3.

7.1 USER BANK 0 REGISTER MAP

ADDR (HEX)	ADDR (DEC.)	REGISTER NAME	SERIAL I/F	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
00	0	WHO_AM_I	R	WHO_AM_I[7:0]							
03	3	USER_CTRL	R/W	DMP_EN	FIFO_EN	I2C_MST_EN	I2C_IF_DIS	DMP_RST	SRAM_RST	I2C_MST_RST	-
05	5	LP_CONFIG	R/W		I2C_MST_CYCLE	ACCEL_CYCLE	GYRO_CYCLE	-			
06	6	PWR_MGMT_1	R/W	DEVICE_RESET	SLEEP	LP_EN	-	TEMP_DIS	CLKSEL[2:0]		
07	7	PWR_MGMT_2	R/W	-		DISABLE_ACCEL			DISABLE_GYRO		
0F	15	INT_PIN_CFG	R/W	INT1_ACTL	INT1_OPEN	INT1_LATCH_INT_EN	INT_ANYRD_2CLEAR	ACTL_FSYNC	FSYNC_INT_MODE_EN	BYPASS_EN	-
10	16	INT_ENABLE	R/W	REG_WOF_EN	-			WOM_INT_EN	PLL_RDY_EN	DMP_INT1_EN	I2C_MST_INT_EN
11	17	INT_ENABLE_1	R/W	-							RAW_DATA_0_RDY_EN
12	18	INT_ENABLE_2	R/W	-			FIFO_OVERFLOW_EN[4:0]				
13	19	INT_ENABLE_3	R/W	-			FIFO_WM_EN[4:0]				
17	23	I2C_MST_STATUS	R/C	PASS_THROUGH	I2C_SLV4_DONE	I2C_LOST_ARB	I2C_SLV4_NACK	I2C_SLV3_NACK	I2C_SLV2_NACK	I2C_SLV1_NACK	I2C_SLV0_NACK
19	25	INT_STATUS	R/C	-				WOM_INT	PLL_RDY_INT	DMP_INT1	I2C_MST_INT
1A	26	INT_STATUS_1	R/C	-							RAW_DATA_0_RDY_INT
1B	27	INT_STATUS_2	R/C	-			FIFO_OVERFLOW_INT[4:0]				
1C	28	INT_STATUS_3	R/C	-			FIFO_WM_INT[4:0]				
28	40	DELAY_TIMEH	R	DELAY_TIMEH[7:0]							
29	41	DELAY_TIMEL	R	DELAY_TIMEL[7:0]							
2D	45	ACCEL_XOUT_H	R	ACCEL_XOUT_H[7:0]							
2E	46	ACCEL_XOUT_L	R	ACCEL_XOUT_L[7:0]							
2F	47	ACCEL_YOUT_H	R	ACCEL_YOUT_H[7:0]							
30	48	ACCEL_YOUT_L	R	ACCEL_YOUT_L[7:0]							
31	49	ACCEL_ZOUT_H	R	ACCEL_ZOUT_H[7:0]							
32	50	ACCEL_ZOUT_L	R	ACCEL_ZOUT_L[7:0]							
33	51	GYRO_XOUT_H	R	GYRO_XOUT_H[7:0]							
34	52	GYRO_XOUT_L	R	GYRO_XOUT_L[7:0]							
35	53	GYRO_YOUT_H	R	GYRO_YOUT_H[7:0]							
36	54	GYRO_YOUT_L	R	GYRO_YOUT_L[7:0]							
37	55	GYRO_ZOUT_H	R	GYRO_ZOUT_H[7:0]							
38	56	GYRO_ZOUT_L	R	GYRO_ZOUT_L[7:0]							
39	57	TEMP_OUT_H	R	TEMP_OUT_H[7:0]							
3A	58	TEMP_OUT_L	R	TEMP_OUT_L[7:0]							
3B	59	EXT_SLV_SENS_DATA_00	R	EXT_SLV_SENS_DATA_00[7:0]							
3C	60	EXT_SLV_SENS_DATA_01	R	EXT_SLV_SENS_DATA_01[7:0]							
3D	61	EXT_SLV_SENS_DATA_02	R	EXT_SLV_SENS_DATA_02[7:0]							
3E	62	EXT_SLV_SENS_DATA_03	R	EXT_SLV_SENS_DATA_03[7:0]							
3F	63	EXT_SLV_SENS_DATA_04	R	EXT_SLV_SENS_DATA_04[7:0]							
40	64	EXT_SLV_SENS_DATA_05	R	EXT_SLV_SENS_DATA_05[7:0]							
41	65	EXT_SLV_SENS_DATA_06	R	EXT_SLV_SENS_DATA_06[7:0]							
42	66	EXT_SLV_SENS_DATA_07	R	EXT_SLV_SENS_DATA_07[7:0]							
43	67	EXT_SLV_SENS_DATA_08	R	EXT_SLV_SENS_DATA_08[7:0]							

ADDR (HEX)	ADDR (DEC.)	REGISTER NAME	SERIAL I/F	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
44	68	EXT_SLV_SENS_DATA_09	R	EXT_SLV_SENS_DATA_09[7:0]							
45	69	EXT_SLV_SENS_DATA_10	R	EXT_SLV_SENS_DATA_10[7:0]							
46	70	EXT_SLV_SENS_DATA_11	R	EXT_SLV_SENS_DATA_11[7:0]							
47	71	EXT_SLV_SENS_DATA_12	R	EXT_SLV_SENS_DATA_12[7:0]							
48	72	EXT_SLV_SENS_DATA_13	R	EXT_SLV_SENS_DATA_13[7:0]							
49	73	EXT_SLV_SENS_DATA_14	R	EXT_SLV_SENS_DATA_14[7:0]							
4A	74	EXT_SLV_SENS_DATA_15	R	EXT_SLV_SENS_DATA_15[7:0]							
4B	75	EXT_SLV_SENS_DATA_16	R	EXT_SLV_SENS_DATA_16[7:0]							
4C	76	EXT_SLV_SENS_DATA_17	R	EXT_SLV_SENS_DATA_17[7:0]							
4D	77	EXT_SLV_SENS_DATA_18	R	EXT_SLV_SENS_DATA_18[7:0]							
4E	78	EXT_SLV_SENS_DATA_19	R	EXT_SLV_SENS_DATA_19[7:0]							
4F	79	EXT_SLV_SENS_DATA_20	R	EXT_SLV_SENS_DATA_20[7:0]							
50	80	EXT_SLV_SENS_DATA_21	R	EXT_SLV_SENS_DATA_21[7:0]							
51	81	EXT_SLV_SENS_DATA_22	R	EXT_SLV_SENS_DATA_22[7:0]							
52	82	EXT_SLV_SENS_DATA_23	R	EXT_SLV_SENS_DATA_23[7:0]							
66	102	FIFO_EN_1	R/W	-				SLV_3_FIFO_EN	SLV_2_FIFO_EN	SLV_1_FIFO_EN	SLV_0_FIFO_EN
67	103	FIFO_EN_2	R/W	-			ACCEL_FIFO_EN	GYRO_Z_FIFO_EN	GYRO_Y_FIFO_EN	GYRO_X_FIFO_EN	TEMP_FIFO_EN
68	104	FIFO_RST	R/W	-			FIFO_RESET[4:0]				
69	105	FIFO_MODE	R/W	-			FIFO_MODE[4:0]				
70	112	FIFO_COUNTH	R	-			FIFO_CNT[12:8]				
71	113	FIFO_COUNTL	R	FIFO_CNT[7:0]							
72	114	FIFO_R_W	R/W	FIFO_R_W[7:0]							
74	116	DATA_RDY_STATUS	R/C	WOF_STATUS	-			RAW_DATA_RDY[3:0]			
76	118	FIFO_CFG	R/W	-							FIFO_CFG
7F	127	REG_BANK_SEL	R/W	-		USER_BANK[1:0]		-			

7.2 USER BANK 1 REGISTER MAP

Addr (Hex)	Addr (Dec.)	Register Name	Serial I/F	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
02	2	SELF_TEST_X_GYRO	R/W	XG_ST_DATA[7:0]							
03	3	SELF_TEST_Y_GYRO	R/W	YG_ST_DATA[7:0]							
04	4	SELF_TEST_Z_GYRO	R/W	ZG_ST_DATA[7:0]							
0E	14	SELF_TEST_X_ACCEL	R/W	XA_ST_DATA[7:0]							
0F	15	SELF_TEST_Y_ACCEL	R/W	YA_ST_DATA[7:0]							
10	16	SELF_TEST_Z_ACCEL	R/W	ZA_ST_DATA[7:0]							
14	20	XA_OFFS_H	R/W	XA_OFFS[14:7]							
15	21	XA_OFFS_L	R/W	XA_OFFS[6:0]							-
17	23	YA_OFFS_H	R/W	YA_OFFS[14:7]							
18	24	YA_OFFS_L	R/W	YA_OFFS[6:0]							-
1A	26	ZA_OFFS_H	R/W	ZA_OFFS[14:7]							
1B	27	ZA_OFFS_L	R/W	ZA_OFFS[6:0]							-

Addr (Hex)	Addr (Dec.)	Register Name	Serial I/F	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
28	40	TIMEBASE_CORRECTION_PLL	R/W	TBC_PLL[7:0]							
7F	127	REG_BANK_SEL	R/W	-		USER_BANK[1:0]		-			

7.3 USER BANK 2 REGISTER MAP

ADDR (HEX)	ADDR (DEC.)	REGISTER NAME	SERIAL I/F	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
00	0	GYRO_SMPLRT_DIV	R/W	GYRO_SMPLRT_DIV[7:0]							
01	1	GYRO_CONFIG_1	R/W	-		GYRO_DLPFCFG[2:0]			GYRO_FS_SEL[1:0]		GYRO_FCHOICE
02	2	GYRO_CONFIG_2	R/W	-		XGYRO_CTEN	YGYRO_CTEN	ZGYRO_CTEN	GYRO_AVGCFG[2:0]		
03	3	XG_OFFS_USRH	R/W	X_OFFS_USER[15:8]							
04	4	XG_OFFS_USRL	R/W	X_OFFS_USER[7:0]							
05	5	YG_OFFS_USRH	R/W	Y_OFFS_USER[15:8]							
06	6	YG_OFFS_USRL	R/W	Y_OFFS_USER[7:0]							
07	7	ZG_OFFS_USRH	R/W	Z_OFFS_USER[15:8]							
08	8	ZG_OFFS_USRL	R/W	Z_OFFS_USER[7:0]							
09	9	ODR_ALIGN_EN	R/W	-							ODR_ALIGN_EN
10	16	ACCEL_SMPLRT_DIV_1	R/W	-				ACCEL_SMPLRT_DIV[11:8]			
11	17	ACCEL_SMPLRT_DIV_2	R/W	ACCEL_SMPLRT_DIV[7:0]							
12	18	ACCEL_INTEL_CTRL	R/W	-						ACCEL_INTEL_EN	ACCEL_INTEL_MODE_INT
13	19	ACCEL_WOM_THR	R/W	WOM_THRESHOLD[7:0]							
14	20	ACCEL_CONFIG	R/W	-		ACCEL_DLPFCFG[2:0]			ACCEL_FS_SEL[1:0]		ACCEL_FCHOICE
15	21	ACCEL_CONFIG_2	R/W	-			AX_ST_EN_REG	AY_ST_EN_REG	AZ_ST_EN_REG	DEC3_CFG[1:0]	
52	82	FSYNC_CONFIG	R/W	DELAY_TIME_EN	-	WOF_DEGLITCH_EN	WOF_EDGE_INT	EXT_SYNC_SET[3:0]			
53	83	TEMP_CONFIG	R/W	-					TEMP_DLPFCFG[2:0]		
54	84	MOD_CTRL_USR	R/W	-							REG_LP_DMP_EN
7F	127	REG_BANK_SEL	R/W	-		USER_BANK[1:0]		-			

7.4 USER BANK 3 REGISTER MAP

ADDR (HEX)	ADDR (DEC.)	REGISTER NAME	SERIAL I/F	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
00	0	I2C_MST_ODR_CONFIG	R/W	-				I2C_MST_ODR_CONFIG[3:0]			
01	1	I2C_MST_CTRL	R/W	MULT_MST_EN	-		I2C_MST_P_NSR	I2C_MST_CLK[3:0]			
02	2	I2C_MST_DELAY_CTRL	R/W	DELAY_ES_SHADOW	-		I2C_SLV4_DELAY_EN	I2C_SLV3_DELAY_EN	I2C_SLV2_DELAY_EN	I2C_SLV1_DELAY_EN	I2C_SLV0_DELAY_EN
03	3	I2C_SLV0_ADDR	R/W	I2C_SLV0_RNW	I2C_ID_0[6:0]						
04	4	I2C_SLV0_REG	R/W	I2C_SLV0_REG[7:0]							
05	5	I2C_SLV0_CTRL	R/W	I2C_SLV0_EN	I2C_SLV0_BYTE_SW	I2C_SLV0_REG_DIS	I2C_SLV0_GRP	I2C_SLV0_LENG[3:0]			
06	6	I2C_SLV0_DO	R/W	I2C_SLV0_DO[7:0]							
07	7	I2C_SLV1_ADDR	R/W	I2C_SLV1_RNW	I2C_ID_1[6:0]						
08	8	I2C_SLV1_REG	R/W	I2C_SLV1_REG[7:0]							
09	9	I2C_SLV1_CTRL	R/W	I2C_SLV1_EN	I2C_SLV1_BYTE_SW	I2C_SLV1_REG_DIS	I2C_SLV1_GRP	I2C_SLV1_LENG[3:0]			

ADDR (HEX)	ADDR (DEC.)	REGISTER NAME	SERIAL I/F	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
0A	10	I2C_SLV1_DO	R/W	I2C_SLV1_DO[7:0]							
0B	11	I2C_SLV2_ADDR	R/W	I2C_SLV2_RNW	I2C_ID_2[6:0]						
0C	12	I2C_SLV2_REG	R/W	I2C_SLV2_REG[7:0]							
0D	13	I2C_SLV2_CTRL	R/W	I2C_SLV2_EN	I2C_SLV2_BYTE_SW	I2C_SLV2_REG_DIS	I2C_SLV2_GRP	I2C_SLV2_LENG[3:0]			
0E	14	I2C_SLV2_DO	R/W	I2C_SLV2_DO[7:0]							
0F	15	I2C_SLV3_ADDR	R/W	I2C_SLV3_RNW	I2C_ID_3[6:0]						
10	16	I2C_SLV3_REG	R/W	I2C_SLV3_REG[7:0]							
11	17	I2C_SLV3_CTRL	R/W	I2C_SLV3_EN	I2C_SLV3_BYTE_SW	I2C_SLV3_REG_DIS	I2C_SLV3_GRP	I2C_SLV3_LENG[3:0]			
12	18	I2C_SLV3_DO	R/W	I2C_SLV3_DO[7:0]							
13	19	I2C_SLV4_ADDR	R/W	I2C_SLV4_RNW	I2C_ID_4[6:0]						
14	20	I2C_SLV4_REG	R/W	I2C_SLV4_REG[7:0]							
15	21	I2C_SLV4_CTRL	R/W	I2C_SLV4_EN	I2C_SLV4_BYTE_SW	I2C_SLV4_REG_DIS	I2C_SLV4_DLY[4:0]				
16	22	I2C_SLV4_DO	R/W	I2C_SLV4_DO[7:0]							
17	23	I2C_SLV4_DI	R	I2C_SLV4_DI[7:0]							
7F	127	REG_BANK_SEL	R/W	-		USER_BANK[1:0]		-			

8 USER BANK 0 REGISTER DESCRIPTIONS

This section describes the function and contents of the User Bank 0 Register Map within the ICM-20948.

NOTE: The device will come up in sleep mode upon power-up.

8.1 WHO_AM_I

Name: WHO_AM_I
Address: 0 (00h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0xEA

BIT	NAME	FUNCTION
7:0	WHO_AM_I[7:0]	Register to indicate to user which device is being accessed. The value for ICM-20948 is 0xEA.

8.2 USER_CTRL

Name: USER_CTRL
Address: 3 (03h)
Type: USR0
Bank: 0
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	DMP_EN	1 – Enables DMP features. 0 – DMP features are disabled after the current processing round has completed.
6	FIFO_EN	1 – Enable FIFO operation mode. 0 – Disable FIFO access from serial interface. To disable FIFO writes by DMA, use FIFO_EN register. To disable possible FIFO writes from DMP, disable the DMP.
5	I2C_MST_EN	1 – Enable the I^{2}C Master I/F module; pins ES_DA and ES_SCL are isolated from pins SDA/SDI and SCL/ SCLK. 0 – Disable I^{2}C Master I/F module; pins ES_DA and ES_SCL are logically driven by pins SDA/SDI and SCL/ SCLK.
4	I2C_IF_DIS	1 – Reset I^{2}C Slave module and put the serial interface in SPI mode only.
3	DMP_RST	1 – Reset DMP module. Reset is asynchronous. This bit auto clears after one clock cycle of the internal 20 MHz clock.
2	SRAM_RST	1 – Reset SRAM module. Reset is asynchronous. This bit auto clears after one clock cycle of the internal 20 MHz clock.
1	I2C_MST_RST	1 – Reset I^{2}C Master module. Reset is asynchronous. This bit auto clears after one clock cycle of the internal 20 MHz clock. **NOTE**: This bit should only be set when the I^{2}C master has hung. If this bit is set during an active I^{2}C master transaction, the I^{2}C slave will hang, which will require the host to reset the slave.
0	-	Reserved.

8.3 LP_CONFIG

Name: LP_CONFIG **Address: 5 (05h)** **Type: USR0** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x40**		
BIT	**NAME**	**FUNCTION**
7	-	Reserved.
6	I2C_MST_CYCLE	1 - Operate I^{2}C master in duty cycled mode. ODR is determined by I2C_MST_ODR_CONFIG register. 0 – Disable I^{2}C master duty cycled mode.
5	ACCEL_CYCLE	1 – Operate ACCEL in duty cycled mode. ODR is determined by ACCEL_SMPLRT_DIV register. 0 – Disable ACCEL duty cycled mode.
4	GYRO_CYCLE	1 – Operate GYRO in duty cycled mode. ODR is determined by GYRO_SMPLRT_DIV register. 0 – Disable GYRO duty cycled mode.
3:0	-	Reserved.

8.4 PWR_MGMT_1

Name: PWR_MGMT_1 **Address: 6 (06h)** **Type: USR0** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x41**		
BIT	**NAME**	**FUNCTION**
7	DEVICE_RESET	1 – Reset the internal registers and restores the default settings. Write a 1 to set the reset, the bit will auto clear.
6	SLEEP	When set, the chip is set to sleep mode (in sleep mode all analog is powered off). Clearing the bit wakes the chip from sleep mode.
5	LP_EN	The LP_EN only affects the digital circuitry, it helps to reduce the digital current when sensors are in LP mode. Please note that the sensors themselves are set in LP mode by the LP_CONFIG register settings. Sensors in LP mode, and use of LP_EN bit together help to reduce overall current. The bit settings are: 1: Turn on low power feature. 0: Turn off low power feature. LP_EN has no effect when the sensors are in low-noise mode.
4	-	Reserved.
3	TEMP_DIS	When set to 1, this bit disables the temperature sensor.
2:0	CLKSEL[2:0]	**Code: Clock Source** 0: Internal 20 MHz oscillator 1-5: Auto selects the best available clock source – PLL if ready, else use the Internal oscillator 6: Internal 20 MHz oscillator 7: Stops the clock and keeps timing generator in reset **NOTE**: CLKSEL[2:0] should be set to 1~5 to achieve full gyroscope performance.

8.5 PWR_MGMT_2

Name: PWR_MGMT_2
Address: 7 (07h)
Type: USR0
Bank: 0
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:6	-	Reserved.
5:3	DISABLE_ACCEL	Only the following values are applicable: 111 – Accelerometer (all axes) disabled. 000 – Accelerometer (all axes) on.
2:0	DISABLE_GYRO	Only the following values are applicable: 111 – Gyroscope (all axes) disabled. 000 – Gyroscope (all axes) on.

8.6 INT_PIN_CFG

Name: INT_PIN_CFG
Address: 15 (0Fh)
Type: USR0
Bank: 0
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	INT1_ACTL	1 – The logic level for INT1 pin is active low. 0 – The logic level for INT1 pin is active high.
6	INT1_OPEN	1 – INT1 pin is configured as open drain. 0 – INT1 pin is configured as push-pull.
5	INT1_LATCH__EN	1 – INT1 pin level held until interrupt status is cleared. 0 – INT1 pin indicates interrupt pulse is width 50 µs.
4	INT_ANYRD_2CLEAR	1 – Interrupt status in INT_STATUS is cleared (set to 0) if any read operation is performed. 0 – Interrupt status in INT_STATUS is cleared (set to 0) only by reading INT_STATUS register. This bit only affects the interrupt status bits that are contained in the register INT_STATUS, and the corresponding hardware interrupt. This bit does not affect the interrupt status bits that are contained in registers INT_STATUS_1, INT_STATUS_2, INT_STATUS_3, and the corresponding hardware interrupt.
3	ACTL_FSYNC	1 – The logic level for the FSYNC pin as an interrupt to the ICM-20948 is active low. 0 – The logic level for the FSYNC pin as an interrupt to the ICM-20948 is active high.
2	FSYNC_INT_MODE_EN	1 – This enables the FSYNC pin to be used as an interrupt. A transition to the active level described by the ACTL_FSYNC bit will cause an interrupt. The status of the interrupt is read in the I²C Master Status register PASS_THROUGH bit. 0 – This disables the FSYNC pin from causing an interrupt.
1	BYPASS_EN	When asserted, the I2C_MASTER interface pins (ES_CL and ES_DA) will go into 'bypass mode' when the I²C master interface is disabled.
0	-	Reserved.

8.7 INT_ENABLE

Name: INT_ENABLE **Address: 16 (10h)** **Type: USR0** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7	REG_WOF_EN	1 – Enable wake on FSYNC interrupt. 0 – Function is disabled.
6:4	-	Reserved.
3	WOM_INT_EN	1 – Enable interrupt for wake on motion to propagate to interrupt pin 1. 0 – Function is disabled.
2	PLL_RDY_EN	1 – Enable PLL RDY interrupt (PLL RDY means PLL is running and in use as the clock source for the system) to propagate to interrupt pin 1. 0 – Function is disabled.
1	DMP_INT1_EN	1 – Enable DMP interrupt to propagate to interrupt pin 1. 0 – Function is disabled.
0	I2C_MST_INT_EN	1 – Enable I^2C master interrupt to propagate to interrupt pin 1. 0 – Function is disabled.

8.8 INT_ENABLE_1

Name: INT_ENABLE_1 **Address: 17 (11h)** **Type: USR0** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:1	-	Reserved.
0	RAW_DATA_0_RDY_EN	1 – Enable raw data ready interrupt from any sensor to propagate to interrupt pin 1. 0 – Function is disabled.

8.9 INT_ENABLE_2

Name: INT_ENABLE_2 **Address: 18 (12h)** **Type: USR0** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:5	-	Reserved.
4:0	FIFO_OVERFLOW_EN[4:0]	1 – Enable interrupt for FIFO overflow to propagate to interrupt pin 1. 0 – Function is disabled.

8.10 INT_ENABLE_3

Name: INT_ENABLE_3
Address: 19 (13h)
Type: USR0
Bank: 0
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:5	-	Reserved.
4:0	FIFO_WM_EN[4:0]	1 – Enable interrupt for FIFO watermark to propagate to interrupt pin 1. 0 – Function is disabled.

8.11 I2C_MST_STATUS

Name: I2C_MST_STATUS
Address: 23 (17h)
Type: USR0
Bank: 0
Serial IF: R/C
Reset Value: 0x00

BIT	NAME	FUNCTION
7	PASS_THROUGH	Status of FSYNC interrupt – used as a way to pass an external interrupt through this chip to the host. If enabled in the INT_PIN_CFG register by asserting bit FSYNC_INT_MODE_EN, this will cause an interrupt. A read of this register clears all status bits in this register.
6	I2C_SLV4_DONE	Asserted when I²C slave 4's transfer is complete, will cause an interrupt if bit I2C_MST_INT_EN in the INT_ENABLE register is asserted, and if the SLV4_DONE_INT_EN bit is asserted in the I2C_SLV4_CTRL register.
5	I2C_LOST_ARB	Asserted when I²C slave loses arbitration of the I²C bus, will cause an interrupt if bit I2C_MST_INT_EN in the INT_ENABLE register is asserted.
4	I2C_SLV4_NACK	Asserted when slave 4 receives a NACK, will cause an interrupt if bit I2C_MST_INT_EN in the INT_ENABLE register is asserted.
3	I2C_SLV3_NACK	Asserted when slave 3 receives a NACK, will cause an interrupt if bit I2C_MST_INT_EN in the INT_ENABLE register is asserted.
2	I2C_SLV2_NACK	Asserted when slave 2 receives a NACK, will cause an interrupt if bit I2C_MST_INT_EN in the INT_ENABLE register is asserted.
1	I2C_SLV1_NACK	Asserted when slave 1 receives a NACK, will cause an interrupt if bit I2C_MST_INT_EN in the INT_ENABLE register is asserted.
0	I2C_SLV0_NACK	Asserted when slave 0 receives a NACK, will cause an interrupt if bit I2C_MST_INT_EN in the INT_ENABLE register is asserted.

8.12 INT_STATUS

Name: INT_STATUS
Address: 25 (19h)
Type: USR0
Bank: 0
Serial IF: R/C
Reset Value: 0x00

BIT	NAME	FUNCTION
7:4	-	Reserved.
3	WOM_INT	1 – Wake on motion interrupt occurred.
2	PLL_RDY_INT	1 – Indicates that the PLL has been enabled and is ready (delay of 4 ms ensures lock).
1	DMP_INT1	1 – Indicates the DMP has generated INT1 interrupt.
0	I2C_MST_INT	1 – Indicates I²C master has generated an interrupt.

8.13 INT_STATUS_1

Name: INT_STATUS_1
Address: 26 (1Ah)
Type: USR0
Bank: 0
Serial IF: R/C
Reset Value: 0x00

BIT	NAME	FUNCTION
7:1	-	Reserved.
0	RAW_DATA_0_RDY_INT	1 – Sensor Register Raw Data, from all sensors, is updated and ready to be read.

8.14 INT_STATUS_2

Name: INT_STATUS_2
Address: 27 (1Bh)
Type: USR0
Bank: 0
Serial IF: R/C
Reset Value: 0x00

BIT	NAME	FUNCTION
7:5	-	Reserved.
4:0	FIFO_OVERFLOW_INT[4:0]	1 – FIFO Overflow interrupt occurred.

8.15 INT_STATUS_3

Name: INT_STATUS_3
Address: 28 (1Ch)
Type: USR0
Bank: 0
Serial IF: R/C
Reset Value: 0x00

BIT	NAME	FUNCTION
7:5	-	Reserved.
4:0	FIFO_WM_INT[4:0]	1 – Watermark interrupt for FIFO occurred.

8.16 DELAY_TIMEH

Name: DELAY_TIMEH
Address: 40 (28h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	DELAY_TIMEH[7:0]	High-byte of delay time between FSYNC event and the 1st gyro ODR event (after the FSYNC event). Reading DELAY_TIMEH will lock DELAY_TIMEH and DELAY_TIMEL from the next update. Reading DELAY_TIMEL will unlock DELAY_TIMEH and DELAY_TIMEL to take the next update due to an FSYNC event.

8.17 DELAY_TIMEL

Name: DELAY_TIMEL Address: 41 (29h) Type: USR0 Bank: 0 Serial IF: R Reset Value: 0x00		
BIT	**NAME**	**FUNCTION**
7:0	DELAY_TIMEL[7:0]	Low-byte of delay time between FSYNC event and the 1st gyro ODR event (after the FSYNC event). Reading DELAY_TIMEH will lock DELAY_TIMEH and DELAY_TIMEL from the next update. Reading DELAY_TIMEL will unlock DELAY_TIMEH and DELAY_TIMEL to take the next update due to an FSYNC event. Delay time in µs = (DELAY_TIMEH * 256 + DELAY_TIMEL) * 0.9645

8.18 ACCEL_XOUT_H

Name: ACCEL_XOUT_H Address: 45 (2Dh) Type: USR0 Bank: 0 Serial IF: R Reset Value: 0x00		
BIT	**NAME**	**FUNCTION**
7:0	ACCEL_XOUT_H[7:0]	High Byte of Accelerometer X-axis data.

8.19 ACCEL_XOUT_L

Name: ACCEL_XOUT_L Address: 46 (2Eh) Type: USR0 Bank: 0 Serial IF: R Reset Value: 0x00		
BIT	**NAME**	**FUNCTION**
7:0	ACCEL_XOUT_L[7:0]	Low Byte of Accelerometer X-axis data. To convert the output of the accelerometer to acceleration measurement use the formula below: X_acceleration = ACCEL_XOUT/Accel_Sensitivity

8.20 ACCEL_YOUT_H

Name: ACCEL_YOUT_H Address: 47 (2Fh) Type: USR0 Bank: 0 Serial IF: R Reset Value: 0x00		
BIT	**NAME**	**FUNCTION**
7:0	ACCEL_YOUT_H[7:0]	High Byte of Accelerometer Y-axis data.

8.21 ACCEL_YOUT_L

Name: ACCEL_YOUT_L
Address: 48 (30h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	ACCEL_YOUT_L[7:0]	Low Byte of Accelerometer Y-axis data. To convert the output of the accelerometer to acceleration measurement use the formula below: Y_acceleration = ACCEL_YOUT/Accel_Sensitivity

8.22 ACCEL_ZOUT_H

Name: ACCEL_ZOUT_H
Address: 49 (31h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	ACCEL_ZOUT_H[7:0]	High Byte of Accelerometer Z-axis data.

8.23 ACCEL_ZOUT_L

Name: ACCEL_ZOUT_L
Address: 50 (32h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	ACCEL_ZOUT_L[7:0]	Low Byte of Accelerometer Z-axis data. To convert the output of the accelerometer to acceleration measurement use the formula below: Z_acceleration = ACCEL_ZOUT/Accel_Sensitivity

8.24 GYRO_XOUT_H

Name: GYRO_XOUT_H
Address: 51 (33h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	GYRO_XOUT_H[7:0]	High Byte of Gyroscope X-axis data.

8.25 GYRO_XOUT_L

Name: GYRO_XOUT_L
Address: 52 (34h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	GYRO_XOUT_L[7:0]	Low Byte of Gyroscope X-axis data. To convert the output of the gyroscope to angular rate measurement use the formula below: X_angular_rate = GYRO_XOUT/Gyro_Sensitivity

8.26 GYRO_YOUT_H

Name: GYRO_YOUT_H
Address: 53 (35h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	GYRO_YOUT_H[7:0]	High Byte of Gyroscope Y-axis data.

8.27 GYRO_YOUT_L

Name: GYRO_YOUT_L
Address: 54 (36h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	GYRO_YOUT_L[7:0]	Low Byte of Gyroscope Y-axis data. To convert the output of the gyroscope to angular rate measurement use the formula below: Y_angular_rate = GYRO_YOUT/Gyro_Sensitivity

8.28 GYRO_ZOUT_H

Name: GYRO_ZOUT_H
Address: 55 (37h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	GYRO_ZOUT_H[7:0]	High Byte of Gyroscope Z-axis data.

8.29 GYRO_ZOUT_L

Name: GYRO_ZOUT_L
Address: 56 (38h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	GYRO_ZOUT_L[7:0]	Low Byte of Gyroscope Z-axis data. To convert the output of the gyroscope to angular rate measurement use the formula below: Z_angular_rate = GYRO_ZOUT/Gyro_Sensitivity

8.30 TEMP_OUT_H

Name: TEMP_OUT_H
Address: 57 (39h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	TEMP_OUT_H[7:0]	High Byte of Temp sensor data.

8.31 TEMP_OUT_L

Name: TEMP_OUT_L
Address: 58 (3Ah)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	TEMP_OUT_L[7:0]	Low Byte of Temp sensor data. To convert the output of the temperature sensor to degrees C use the following formula: TEMP_degC = ((TEMP_OUT – RoomTemp_Offset)/Temp_Sensitivity) + 21degC

8.32 EXT_SLV_SENS_DATA_00

Name: EXT_SLV_SENS_DATA_00
Address: 59 (3Bh)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_00[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.33 EXT_SLV_SENS_DATA_01

Name: EXT_SLV_SENS_DATA_01
Address: 60 (3Ch)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_01[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.34 EXT_SLV_SENS_DATA_02

Name: EXT_SLV_SENS_DATA_02
Address: 61 (3Dh)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_02[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.35 EXT_SLV_SENS_DATA_03

Name: EXT_SLV_SENS_DATA_03
Address: 62 (3Eh)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_03[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.36 EXT_SLV_SENS_DATA_04

Name: EXT_SLV_SENS_DATA_04
Address: 63 (3Fh)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_04[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.37 EXT_SLV_SENS_DATA_05

Name: EXT_SLV_SENS_DATA_05
Address: 64 (40h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_05[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.38 EXT_SLV_SENS_DATA_06

Name: EXT_SLV_SENS_DATA_06
Address: 65 (41h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_06[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.39 EXT_SLV_SENS_DATA_07

Name: EXT_SLV_SENS_DATA_07
Address: 66 (42h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_07[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.40 EXT_SLV_SENS_DATA_08

Name: EXT_SLV_SENS_DATA_08
Address: 67 (43h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_08[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.41 EXT_SLV_SENS_DATA_09

Name: EXT_SLV_SENS_DATA_09
Address: 68 (44h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_09[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.42 EXT_SLV_SENS_DATA_10

Name: EXT_SLV_SENS_DATA_10
Address: 69 (45h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_10[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.43 EXT_SLV_SENS_DATA_11

Name: EXT_SLV_SENS_DATA_11
Address: 70 (46h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_11[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.44 EXT_SLV_SENS_DATA_12

Name: EXT_SLV_SENS_DATA_12
Address: 71 (47h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_12[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.45 EXT_SLV_SENS_DATA_13

Name: EXT_SLV_SENS_DATA_13
Address: 72 (48h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_13[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.46 EXT_SLV_SENS_DATA_14

Name: EXT_SLV_SENS_DATA_14
Address: 73 (49h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_14[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.47 EXT_SLV_SENS_DATA_15

Name: EXT_SLV_SENS_DATA_15
Address: 74 (4Ah)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_15[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.48 EXT_SLV_SENS_DATA_16

Name: EXT_SLV_SENS_DATA_16
Address: 75 (4Bh)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_16[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.49 EXT_SLV_SENS_DATA_17

Name: EXT_SLV_SENS_DATA_17
Address: 76 (4Ch)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_17[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.50 EXT_SLV_SENS_DATA_18

Name: EXT_SLV_SENS_DATA_18
Address: 77 (4Dh)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_18[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.51 EXT_SLV_SENS_DATA_19

Name: EXT_SLV_SENS_DATA_19
Address: 78 (4Eh)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_19[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.52 EXT_SLV_SENS_DATA_20

Name: EXT_SLV_SENS_DATA_20
Address: 79 (4Fh)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_20[7:0]	Sensor data read from external I^2C devices via the I^2C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.53 EXT_SLV_SENS_DATA_21

Name: EXT_SLV_SENS_DATA_21
Address: 80 (50h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_21[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.54 EXT_SLV_SENS_DATA_22

Name: EXT_SLV_SENS_DATA_22
Address: 81 (51h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_22[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.55 EXT_SLV_SENS_DATA_23

Name: EXT_SLV_SENS_DATA_23
Address: 82 (52h)
Type: USR0
Bank: 0
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	EXT_SLV_SENS_DATA_23[7:0]	Sensor data read from external I^{2}C devices via the I^{2}C master interface. The data stored is controlled by the I2C_SLV(0-4)_ADDR, I2C_SLV(0-4)_REG, and I2C_SLV(0-4)_CTRL registers.

8.56 FIFO_EN_1

Name: FIFO_EN_1
Address: 102 (66h)
Type: USR0
Bank: 0
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:4	-	Reserved.
3	SLV_3_FIFO_EN	1 – Write EXT_SENS_DATA registers associated to SLV_3 (as determined by I2C_SLV2_CTRL, I2C_SLV1_CTRL, and I2C_SL20_CTRL) to the FIFO at the sample rate; 0 – Function is disabled.
2	SLV_2_FIFO_EN	1 – Write EXT_SENS_DATA registers associated to SLV_2 (as determined by I2C_SLV0_CTRL, I2C_SLV1_CTRL, and I2C_SL20_CTRL) to the FIFO at the sample rate; 0 – Function is disabled.
1	SLV_1_FIFO_EN	1 – Write EXT_SENS_DATA registers associated to SLV_1 (as determined by I2C_SLV0_CTRL and I2C_SLV1_CTRL) to the FIFO at the sample rate; 0 – Function is disabled.
0	SLV_0_FIFO_EN	1 – Write EXT_SENS_DATA registers associated to SLV_0 (as determined by I2C_SLV0_CTRL) to the FIFO at the sample rate; 0 – Function is disabled.

8.57 FIFO_EN_2

Name: FIFO_EN_2
Address: 103 (67h)
Type: USR0
Bank: 0
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:5	-	Reserved.
4	ACCEL_FIFO_EN	1 – Write ACCEL_XOUT_H, ACCEL_XOUT_L, ACCEL_YOUT_H, ACCEL_YOUT_L, ACCEL_ZOUT_H, and ACCEL_ZOUT_L to the FIFO at the sample rate; 0 – Function is disabled.
3	GYRO_Z_FIFO_EN	1 – Write GYRO_ZOUT_H and GYRO_ZOUT_L to the FIFO at the sample rate. 0 – Function is disabled.
2	GYRO_Y_FIFO_EN	1 – Write GYRO_YOUT_H and GYRO_YOUT_L to the FIFO at the sample rate. 0 – Function is disabled.
1	GYRO_X_FIFO_EN	1 – Write GYRO_XOUT_H and GYRO_XOUT_L to the FIFO at the sample rate. 0 – Function is disabled.
0	TEMP_FIFO_EN	1 – Write TEMP_OUT_H and TEMP_OUT_L to the FIFO at the sample rate. 0 – Function is disabled.

8.58 FIFO_RST

Name: FIFO_RST **Address: 104 (68h)** **Type: USR0** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:5	-	Reserved.
4:0	FIFO_RESET[4:0]	S/W FIFO reset. Assert and hold to set FIFO size to 0. Assert and de-assert to reset FIFO.

8.59 FIFO_MODE

Name: FIFO_MODE **Address: 105 (69h)** **Type: USR0** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:5	-	Reserved.
4:0	FIFO_MODE[4:0]	0 – Stream. 1 – Snapshot. When set to '1', when the FIFO is full, additional writes will not be written to FIFO. When set to '0', when the FIFO is full, additional writes will be written to the FIFO, replacing the oldest data.

8.60 FIFO_COUNTH

Name: FIFO_COUNTH **Address: 112 (70h)** **Type: USR0** **Bank: 0** **Serial IF: R** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:5	-	Reserved.
4:0	FIFO_CNT[12:8]	High Bits, count indicates the number of written bytes in the FIFO. Reading this byte latches the data for both FIFO_COUNTH, and FIFO_COUNTL.

8.61 FIFO_COUNTL

Name: FIFO_COUNTL **Address: 113 (71h)** **Type: USR0** **Bank: 0** **Serial IF: R** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:0	FIFO_CNT[7:0]	Low bits, count indicates the number of written bytes in the FIFO.

8.62 FIFO_R_W

Name: FIFO_R_W **Address: 114 (72h)** **Type: USR0** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:0	FIFO_R_W[7:0]	Reading from or writing to this register actually reads/writes the FIFO. For example, to write a byte to the FIFO, write the desired byte value to FIFO_R_W[7:0]. To read a byte from the FIFO, perform a register read operation and access the result in FIFO_R_W[7:0].

8.63 DATA_RDY_STATUS

Name: DATA_RDY_STATUS **Address: 116 (74h)** **Type: USR0** **Bank: 0** **Serial IF: R/C** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7	WOF_STATUS	Wake on FSYNC interrupt status. Cleared on read.
6:4	-	Reserved.
3:0	RAW_DATA_RDY[3:0]	Data from sensors is copied to FIFO or SRAM. Set when sequence controller kicks off on a sensor data load. Only bit 0 is relevant in a single FIFO configuration. Cleared on read.

8.64 FIFO_CFG

Name: FIFO_CFG **Address: 118 (76h)** **Type: USR0** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:1	-	Reserved.
0	FIFO_CFG	This bit should be set to 1 if interrupt status for each sensor is required.

8.65 REG_BANK_SEL

Name: REG_BANK_SEL **Address: 127 (7Fh)** **Type: ALL** **Bank: 0** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:6	-	Reserved.
5:4	USER_BANK[1:0]	Use the following values in this bit-field to select a USER BANK. 0: Select USER BANK 0. 1: Select USER BANK 1. 2: Select USER BANK 2. 3: Select USER BANK 3.
3:0	-	Reserved.

9 USR BANK 1 REGISTER DESCRIPTIONS

This section describes the function and contents of the User Bank 1 Register Map within the ICM-20948.

NOTE: The device will come up in sleep mode upon power-up.

9.1 SELF_TEST_X_GYRO

Name: SELF_TEST_X_GYRO
Address: 2 (02h)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	XG_ST_DATA[7:0]	The value in this register indicates the self-test output generated during manufacturing tests. This value is to be used to check against subsequent self-test outputs performed by the end user.

9.2 SELF_TEST_Y_GYRO

Name: SELF_TEST_Y_GYRO
Address: 3 (03h)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	YG_ST_DATA[7:0]	The value in this register indicates the self-test output generated during manufacturing tests. This value is to be used to check against subsequent self-test outputs performed by the end user.

9.3 SELF_TEST_Z_GYRO

Name: SELF_TEST_Z_GYRO
Address: 4 (04h)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	ZG_ST_DATA[7:0]	The value in this register indicates the self-test output generated during manufacturing tests. This value is to be used to check against subsequent self-test outputs performed by the end user.

9.4 SELF_TEST_X_ACCEL

Name: SELF_TEST_X_ACCEL
Address: 14 (0Eh)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	XA_ST_DATA[7:0]	Contains self-test data for the X Accelerometer.

9.5 SELF_TEST_Y_ACCEL

Name: SELF_TEST_Y_ACCEL
Address: 15 (0Fh)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	YA_ST_DATA[7:0]	Contains self-test data for the Y Accelerometer.

9.6 SELF_TEST_Z_ACCEL

Name: SELF_TEST_Z_ACCEL
Address: 16 (10h)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	ZA_ST_DATA[7:0]	Contains self-test data for the Z Accelerometer.

9.7 XA_OFFS_H

Name: XA_OFFS_H
Address: 20 (14h)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: Trimmed on a per-part basis for optimal performance

BIT	NAME	FUNCTION
7:0	XA_OFFS[14:7]	Upper bits of the X accelerometer offset cancellation. +/- 16g Offset cancellation in all Full Scale modes, 15 bit 0.98-mg steps.

9.8 XA_OFFS_L

Name: XA_OFFS_L
Address: 21 (15h)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: Trimmed on a per-part basis for optimal performance

BIT	NAME	FUNCTION
7:1	XA_OFFS[6:0]	Lower bits of the X accelerometer offset cancellation. +/- 16g Offset cancellation in all Full Scale modes, 15 bit 0.98-mg steps.
0	-	Reserved.

9.9 YA_OFFS_H

Name: YA_OFFS_H
Address: 23 (17h)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: Trimmed on a per-part basis for optimal performance

BIT	NAME	FUNCTION
7:0	YA_OFFS[14:7]	Upper bits of the Y accelerometer offset cancellation. +/- 16g Offset cancellation in all Full Scale modes, 15 bit 0.98-mg steps.

9.10 YA_OFFS_L

Name: YA_OFFS_L
Address: 24 (18h)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: Trimmed on a per-part basis for optimal performance

BIT	NAME	FUNCTION
7:1	YA_OFFS[6:0]	Lower bits of the Y accelerometer offset cancellation. +/- 16g Offset cancellation in all Full Scale modes, 15 bit 0.98-mg steps.
0	-	Reserved .

9.11 ZA_OFFS_H

Name: ZA_OFFS_H
Address: 26 (1Ah)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: Trimmed on a per-part basis for optimal performance

BIT	NAME	FUNCTION
7:0	ZA_OFFS[14:7]	Upper bits of the Z accelerometer offset cancellation. +/- 16g Offset cancellation in all Full Scale modes, 15 bit 0.98-mg steps.

9.12 ZA_OFFS_L

Name: ZA_OFFS_L
Address: 27 (1Bh)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: Trimmed on a per-part basis for optimal performance

BIT	NAME	FUNCTION
7:1	ZA_OFFS[6:0]	Lower bits of the Z accelerometer offset cancellation. +/- 16g Offset cancellation in all Full Scale modes, 15 bit 0.98-mg steps.
0	-	Reserved.

9.13 TIMEBASE_CORRECTION_PLL

Name: TIMEBASE_CORRECTION_PLL
Address: 40 (28h)
Type: USR1
Bank: 1
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	TBC_PLL[7:0]	System PLL clock period error (signed, [-10%, +10%]).

9.14 REG_BANK_SEL

Name: REG_BANK_SEL
Address: 127 (7Fh)
Type:
Bank: 1
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:6	-	Reserved.
5:4	USER_BANK[1:0]	Use the following values in this bit-field to select a USER BANK. 0: Select USER BANK 0. 1: Select USER BANK 1. 2: Select USER BANK 2. 3: Select USER BANK 3.
3:0	-	Reserved.

10 USR BANK 2 REGISTER MAP

This section describes the function and contents of the User Bank 2 Register Map within the ICM-20948.

NOTE: The device will come up in sleep mode upon power-up.

10.1 GYRO_SMPLRT_DIV

Name: GYRO_SMPLRT_DIV
Address: 0 (00h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	GYRO_SMPLRT_DIV[7:0]	Gyro sample rate divider. Divides the internal sample rate to generate the sample rate that controls sensor data output rate, FIFO sample rate, and DMP sequence rate. **NOTE:** This register is only effective when FCHOICE = 1'b1 (FCHOICE_B register bit is 1'b0), and (0 < DLPF_CFG < 7). ODR is computed as follows: 1.1 kHz/(1+GYRO_SMPLRT_DIV[7:0])

10.2 GYRO_CONFIG_1

Name: GYRO_CONFIG_1
Address: 1 (01h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x01

BIT	NAME	FUNCTION
7:6	-	Reserved.
5:3	GYRO_DLPFCFG[2:0]	Gyro low pass filter configuration as shown in Table 16.
2:1	GYRO_FS_SEL[1:0]	Gyro Full Scale Select: 00 = ±250 dps 01= ±500 dps 10 = ±1000 dps 11 = ±2000 dps
0	GYRO_FCHOICE	0 – Bypass gyro DLPF. 1 – Enable gyro DLPF.

The gyroscope DLPF is configured by GYRO_DLPFCFG, when GYRO_FCHOICE = 1. The gyroscope data is filtered according to the value of GYRO_DLPFCFG and GYRO_FCHOICE as shown in Table 16.

GYRO_FCHOICE	GYRO_DLPFCFG	OUTPUT		
		3DB BW [HZ]	NBW [HZ]	RATE [HZ]
0	x	12106	12316	9000
1	0	196.6	229.8	1125/(1+GYRO_SMPLRT_DIV)Hz where GYRO_SMPLRT_DIV is 0, 1, 2,…255
1	1	151.8	187.6	1125/(1+GYRO_SMPLRT_DIV)Hz where GYRO_SMPLRT_DIV is 0, 1, 2,…255
1	2	119.5	154.3	1125/(1+GYRO_SMPLRT_DIV)Hz where GYRO_SMPLRT_DIV is 0, 1, 2,…255
1	3	51.2	73.3	1125/(1+GYRO_SMPLRT_DIV)Hz where GYRO_SMPLRT_DIV is 0, 1, 2,…255
1	4	23.9	35.9	1125/(1+GYRO_SMPLRT_DIV)Hz where GYRO_SMPLRT_DIV is 0, 1, 2,…255
1	5	11.6	17.8	1125/(1+GYRO_SMPLRT_DIV)Hz where GYRO_SMPLRT_DIV is 0, 1, 2,…255
1	6	5.7	8.9	1125/(1+GYRO_SMPLRT_DIV)Hz where GYRO_SMPLRT_DIV is 0, 1, 2,…255
1	7	361.4	376.5	1125/(1+GYRO_SMPLRT_DIV)Hz where GYRO_SMPLRT_DIV is 0, 1, 2,…255

Table 16. Gyroscope Configuration 1

10.3 GYRO_CONFIG_2

Name: GYRO_CONFIG_2
Address: 2 (02h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:6	-	Reserved.
5	XGYRO_CTEN	X Gyro self-test enable.
4	YGYRO_CTEN	Y Gyro self-test enable.
3	ZGYRO_CTEN	Z Gyro self-test enable.
2:0	GYRO_AVGCFG[2:0]	Averaging filter configuration settings for low-power mode. 0: 1x averaging. 1: 2x averaging. 2: 4x averaging. 3: 8x averaging. 4: 16x averaging. 5: 32x averaging. 6: 64x averaging. 7: 128x averaging.

Table 17 lists the gyroscope filter bandwidths available in the low-power mode of operation. In the low-power mode of operation, the gyroscope is duty-cycled.

	AVERAGES	1X	2X	4X	8X	16X	32X	64X	128X
	GYRO_FCHOICE	1	1	1	1	1	1	1	1
	GYRO_AVGCFG	0	1	2	3	4	5	6	7
	TON [MS]	1.15	1.59	2.48	4.26	7.82	14.93	29.15	57.59
	NBW [HZ]	773.5	469.8	257.8	134.8	68.9	34.8	17.5	8.8
	RMS NOISE [DPS-RMS] TYP (BASED ON GYROSCOPE NOISE: 0.015 DPS/√HZ)	0.42	0.33	0.24	0.17	0.12	0.09	0.06	0.04
GYRO_SMPLRT_DIV	ODR [HZ]	CURRENT CONSUMPTION [MA] TYP							
255	4.4	1.04	1.05	1.05	1.06	1.09	1.14	1.24	1.45
64	17.3	1.07	1.08	1.10	1.15	1.25	1.45	1.85	N/A
63	17.6	1.07	1.08	1.11	1.16	1.26	1.46	1.87	
32	34.1	1.10	1.12	1.17	1.27	1.47	1.86	N/A	
31	35.2	1.10	1.13	1.18	1.28	1.48	1.89		
22	48.9	1.13	1.16	1.23	1.37	1.66	2.22		
16	66.2	1.16	1.21	1.30	1.49	1.88	N/A		
15	70.3	1.17	1.22	1.32	1.52	1.93			
10	102.3	1.23	1.30	1.45	1.74	2.34			
8	125.0	1.27	1.36	1.54	1.90	N/A			
7	140.6	1.30	1.40	1.60	2.01				
5	187.5	1.38	1.52	1.79	2.33				
4	225.0	1.45	1.62	1.94	N/A				
3	281.3	1.56	1.76	2.17					
2	375.0	1.74	2.00	N/A					
1	562.5	2.09	N/A						

Table 17. Gyroscope Configuration 2

NOTE: Ton is the ON time for motion measurement when the gyroscope is in duty cycle mode.

10.4 XG_OFFS_USRH

Name: XG_OFFS_USRH
Address: 3 (03h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	X_OFFS_USER[15:8]	Upper byte of X gyro offset cancellation. Step size: 0.0305 dps/LSB.

10.5 XG_OFFS_USRL

Name: XG_OFFS_USRL
Address: 4 (04h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	X_OFFS_USER[7:0]	Lower byte of X gyro offset cancellation. Step size: 0.0305 dps/LSB.

10.6 YG_OFFS_USRH

Name: YG_OFFS_USRH
Address: 5 (05h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	Y_OFFS_USER[15:8]	Upper byte of Y gyro offset cancellation. Step size: 0.0305 dps/LSB.

10.7 YG_OFFS_USRL

Name: YG_OFFS_USRL
Address: 6 (06h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	Y_OFFS_USER[7:0]	Lower byte of Y gyro offset cancellation. Step size: 0.0305 dps/LSB.

10.8 ZG_OFFS_USRH

Name: ZG_OFFS_USRH
Address: 7 (07h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	Z_OFFS_USER[15:8]	Upper byte of Z gyro offset cancellation. Step size: 0.0305 dps/LSB.

10.9 ZG_OFFS_USRL

Name: ZG_OFFS_USRL
Address: 8 (08h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	Z_OFFS_USER[7:0]	Lower byte of Z gyro offset cancellation. Step size: 0.0305 dps/LSB.

10.10 ODR_ALIGN_EN

Name: ODR_ALIGN_EN
Address: 9 (09h)
Type: USR2
Bank: 2
OTP: No
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:1	-	Reserved.
0	ODR_ALIGN_EN	0: Disables ODR start-time alignment. 1: Enables ODR start-time alignment when any of the following registers is written (with the same value or with different values): GYRO_SMPLRT_DIV, ACCEL_SMPLRT_DIV_1, ACCEL_SMPLRT_DIV_2, I2C_MST_ODR_CONFIG.

10.11 ACCEL_SMPLRT_DIV_1

Name: ACCEL_SMPLRT_DIV_1
Address: 16 (10h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:4	-	Reserved.
3:0	ACCEL_SMPLRT_DIV[11:8]	MSB for ACCEL sample rate div.

10.12 ACCEL_SMPLRT_DIV_2

Name: ACCEL_SMPLRT_DIV_2
Address: 17 (11h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	ACCEL_SMPLRT_DIV[7:0]	LSB for ACCEL sample rate div. ODR is computed as follows: 1.125 kHz/(1+ACCEL_SMPLRT_DIV[11:0])

10.13 ACCEL_INTEL_CTRL

Name: ACCEL_INTEL_CTRL
Address: 18 (12h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:2	-	Reserved.
1	ACCEL_INTEL_EN	Enable the WOM logic.
0	ACCEL_INTEL_MODE_INT	Selects WOM algorithm. 1 = Compare the current sample with the previous sample. 0 = Initial sample is stored, all future samples are compared to the initial sample.

10.14 ACCEL_WOM_THR

Name: ACCEL_WOM_THR
Address: 19 (13h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	WOM_THRESHOLD[7:0]	This register holds the threshold value for the Wake on Motion Interrupt for ACCEL x/y/z axes. LSB = 4 mg. Range is 0 mg to 1020 mg.

10.15 ACCEL_CONFIG

Name: ACCEL_CONFIG
Address: 20 (14h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x01

BIT	NAME	FUNCTION
7:6	-	Reserved.
5:3	ACCEL_DLPFCFG[2:0]	Accelerometer low pass filter configuration as shown in Table 18.
2:1	ACCEL_FS_SEL[1:0]	Accelerometer Full Scale Select: 00: ±2g 01: ±4g 10: ±8g 11: ±16g
0	ACCEL_FCHOICE	0: Bypass accel DLPF. 1: Enable accel DLPF.

ACCEL_FCHOICE	ACCEL_DLPFCFG	OUTPUT		
		3DB BW [HZ]	NBW [HZ]	RATE [HZ]
0	x	1209	1248	4500
1	0	246.0	265.0	1125/(1+ACCEL_SMPLRT_DIV)Hz where ACCEL_SMPLRT_DIV is 0, 1, 2,…4095
1	1	246.0	265.0	1125/(1+ACCEL_SMPLRT_DIV)Hz where ACCEL_SMPLRT_DIV is 0, 1, 2,…4095
1	2	111.4	136.0	1125/(1+ACCEL_SMPLRT_DIV)Hz where ACCEL_SMPLRT_DIV is 0, 1, 2,…4095
1	3	50.4	68.8	1125/(1+ACCEL_SMPLRT_DIV)Hz where ACCEL_SMPLRT_DIV is 0, 1, 2,…4095
1	4	23.9	34.4	1125/(1+ACCEL_SMPLRT_DIV)Hz where ACCEL_SMPLRT_DIV is 0, 1, 2,…4095
1	5	11.5	17.0	1125/(1+ACCEL_SMPLRT_DIV)Hz where ACCEL_SMPLRT_DIV is 0, 1, 2,…4095
1	6	5.7	8.3	1125/(1+ACCEL_SMPLRT_DIV)Hz where ACCEL_SMPLRT_DIV is 0, 1, 2,…4095
1	7	473	499	1125/(1+ACCEL_SMPLRT_DIV)Hz where ACCEL_SMPLRT_DIV is 0, 1, 2,…4095

Table 18. Accelerator Configuration

The data rate out of the DLPF filter block can be further reduced by a factor of 1.125 kHz/(1+ACCEL_SMPLRT_DIV[11:0]) where ACCEL_SMPLRT_DIV is a 12-bit integer.

10.16 ACCEL_CONFIG_2

Name: ACCEL_CONFIG_2
Address: 21 (15h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:5	-	Reserved.
4	AX_ST_EN_REG	X Accel self-test enable.
3	AY_ST_EN_REG	Y Accel self-test enable.
2	AZ_ST_EN_REG	Z Accel self-test enable.
1:0	DEC3_CFG[1:0]	Controls the number of samples averaged in the accelerometer decimator: 0: Average 1 or 4 samples depending on ACCEL_FCHOICE (see Table 19). 1: Average 8 samples. 2: Average 16 samples. 3: Average 32 samples.

Table 19 lists the accelerometer filter bandwidths available in the low-power mode of operation. In the low-power mode of operation, the accelerometer is duty-cycled.

	AVERAGES	1X	4X	8X	16X	32X
	ACCEL_FCHOICE	0	1	1	1	1
	ACCEL_DLPFCFG	x	7	7	7	7
	DEC3_CFG	0	0	1	2	3
	TON (MS)	0.821	1.488	2.377	4.154	7.71
	NBW (HZ)	1237.5	496.8	264.8	136.5	69.2
	RMS NOISE [MG-RMS] TYP (BASED ON ACCELEROMETER NOISE: 230µG/√HZ)	8.1	5.1	3.7	2.7	1.9
ACCEL_SMPLRT_DIV	ODR [HZ]	CURRENT CONSUMPTION [µA] TYP				
4095	0.27	6.2	6.3	6.5	6.9	7.6
2044	0.55	6.3	6.6	7.0	7.7	9.2
1022	1.1	6.7	7.2	8.0	9.4	12.3
513	2.2	7.3	8.4	9.9	12.8	18.6
255	4.4	8.7	10.9	13.8	19.7	31.4
127	8.8	11.4	15.8	21.6	33.3	56.7
63	17.6	16.8	25.6	37.3	60.7	107.5
31	35.2	27.6	45.2	68.6	115.3	208.9
22	48.9	36.1	60.5	93.0	158.1	288.3
15	70.3	49.2	84.3	131.1	224.7	411.9
10	102.3	68.9	119.9	188.0	324.1	596.3
7	140.6	92.4	162.7	256.3	443.3	N/A

5	187.5	121.2	214.9	
3	281.3	178.9	319.3	N/A
1	562.5	351.7	N/A	

Table 19. Accelerator Configuration 2

NOTE: Ton is the ON time for motion measurement when the accelerometer is in duty cycle mode.

10.17 FSYNC_CONFIG

Name: FSYNC_CONFIG
Address: 82 (52h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	DELAY_TIME_EN	0: Disables delay time measurement between FSYNC event and the first ODR event (after FSYNC event). 1: Enables delay time measurement between FSYNC event and the first ODR event (after FSYNC event).
6	-	Reserved.
5	WOF_DEGLITCH_EN	Enable digital deglitching of FSYNC input for Wake on FSYNC.
4	WOF_EDGE_INT	0: FSYNC is a level interrupt for Wake on FSYNC. 1: FSYNC is an edge interrupt for Wake on FSYNC. ACTL_FSYNC is used to set the polarity of the interrupt.
3:0	EXT_SYNC_SET[3:0]	Enables the FSYNC pin data to be sampled. EXT_SYNC_SET FSYNC bit location. 0: Function disabled. 1: TEMP_OUT_L[0]. 2: GYRO_XOUT_L[0]. 3: GYRO_YOUT_L[0]. 4: GYRO_ZOUT_L[0]. 5: ACCEL_XOUT_L[0]. 6: ACCEL_YOUT_L[0]. 7: ACCEL_ZOUT_L[0].

10.18 TEMP_CONFIG

Name: TEMP_CONFIG
Address: 83 (53h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
2:0	TEMP_DLPFCFG[2:0]	Low pass filter configuration for temperature sensor as shown in the table below:

TEMP_DLPCFG<2:0>	TEMP SENSOR	
	NBW (HZ)	RATE (KHZ)
0	7932.0	9
1	217.9	1.125
2	123.5	1.125
3	65.9	1.125
4	34.1	1.125
5	17.3	1.125
6	8.8	Rate (kHz)
7	7932.0	9

10.19 MOD_CTRL_USR

Name: MOD_CTRL_USR
Address: 84 (54h)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x03

BIT	NAME	FUNCTION
7:1	-	Reserved.
0	REG_LP_DMP_EN	Enable turning on DMP in Low Power Accelerometer mode.

10.20 REG_BANK_SEL

Name: REG_BANK_SEL
Address: 127 (7Fh)
Type: USR2
Bank: 2
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:6	-	Reserved.
5:4	USER_BANK[1:0]	Use the following values in this bit-field to select a USER BANK. 0: Select USER BANK 0. 1: Select USER BANK 1. 2: Select USER BANK 2. 3: Select USER BANK 3.
3:0	-	Reserved.

11 USR BANK 3 REGISTER MAP

This section describes the function and contents of the User Bank 3 Register Map within the ICM-20948.

NOTE: The device will come up in sleep mode upon power-up.

11.1 I2C_MST_ODR_CONFIG

Name: I2C_MST_ODR_CONFIG
Address: 0 (00h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:4	-	Reserved
3:0	I2C_MST_ODR_CONFIG[3:0]	ODR configuration for external sensor when gyroscope and accelerometer are disabled. ODR is computed as follows: 1.1 kHz/(2^((odr_config[3:0]))) When gyroscope is enabled, all sensors (including I2C_MASTER) use the gyroscope ODR. If gyroscope is disabled, then all sensors (including I2C_MASTER) use the accelerometer ODR.

11.2 I2C_MST_CTRL

Name: I2C_MST_CTRL
Address: 1 (01h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	MULT_MST_EN	Enables multi-master capability. When disabled, clocking to the I2C_MST_IF can be disabled when not in use and the logic to detect lost arbitration is disabled.
6:5	-	Reserved.
4	I2C_MST_P_NSR	This bit controls the I^2C Master's transition from one slave read to the next slave read. 0 - There is a restart between reads. 1 - There is a stop between reads.
3:0	I2C_MST_CLK[3:0]	Sets I^2C master clock frequency as shown in Table 23.

11.3 I2C_MST_DELAY_CTRL

Name: I2C_MST_DELAY_CTRL
Address: 2 (02h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	DELAY_ES_SHADOW	Delays shadowing of external sensor data until all data is received.
6:5	-	Reserved.
4	I2C_SLV4_DELAY_EN	When enabled, slave 4 will only be accessed 1/(1+I2C_SLC4_DLY) samples as determined by I2C_MST_ODR_CONFIG.
3	I2C_SLV3_DELAY_EN	When enabled, slave 3 will only be accessed 1/(1+I2C_SLC4_DLY) samples as determined by I2C_MST_ODR_CONFIG.
2	I2C_SLV2_DELAY_EN	When enabled, slave 2 will only be accessed 1/(1+I2C_SLC4_DLY) samples as determined by I2C_MST_ODR_CONFIG.
1	I2C_SLV1_DELAY_EN	When enabled, slave 1 will only be accessed 1/(1+I2C_SLC4_DLY) samples as determined by I2C_MST_ODR_CONFIG.
0	I2C_SLV0_DELAY_EN	When enabled, slave 0 will only be accessed 1/(1+I2C_SLC4_DLY) samples as determined by I2C_MST_ODR_CONFIG.

11.4 I2C_SLV0_ADDR

Name: I2C_SLV0_ADDR
Address: 3 (03h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	I2C_SLV0_RNW	1 – Transfer is a read. 0 – Transfer is a write.
6:0	I2C_ID_0[6:0]	Physical address of I^2C slave 0.

11.5 I2C_SLV0_REG

Name: I2C_SLV0_REG
Address: 4 (04h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	I2C_SLV0_REG[7:0]	I^2C slave 0 register address from where to begin data transfer.

11.6 I2C_SLV0_CTRL

Name: I2C_SLV0_CTRL
Address: 5 (05h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	I2C_SLV0_EN	1 – Enable reading data from this slave at the sample rate and storing data at the first available EXT_SENS_DATA register, which is always EXT_SENS_DATA_00 for I²C slave 0. 0 – Function is disabled for this slave.
6	I2C_SLV0_BYTE_SW	1 – Swap bytes when reading both the low and high byte of a word. Note there is nothing to swap after reading the first byte if I2C_SLV0_REG[0] = 1, or if the last byte read has a register address lsb = 0. For example, if I2C_SLV0_REG = 0x1, and I2C_SLV0_LENG = 0x4: 1) The first byte read from address 0x1 will be stored at EXT_SENS_DATA_00, 2) the second and third bytes will be read and swapped, so the data read from address 0x2 will be stored at EXT_SENS_DATA_02, and the data read from address 0x3 will be stored at EXT_SENS_DATA_01, 3) The last byte read from address 0x4 will be stored at EXT_SENS_DATA_03. 0 – No swapping occurs; bytes are written in order read.
5	I2C_SLV0_REG_DIS	When set, the transaction does not write a register value, it will only read data, or write data.
4	I2C_SLV0_GRP	External sensor data typically comes in as groups of two bytes. This bit is used to determine if the groups are from the slave's register address 0 and 1, 2 and 3, etc.., or if the groups are address 1 and 2, 3 and 4, etc. 0 indicates slave register addresses 0 and 1 are grouped together (odd numbered register ends the group). 1 indicates slave register addresses 1 and 2 are grouped together (even numbered register ends the group). This allows byte swapping of registers that are grouped starting at any address.
3:0	I2C_SLV0_LENG[3:0]	Number of bytes to be read from I²C slave 0.

11.7 I2C_SLV0_DO

Name: I2C_SLV0_DO
Address: 6 (06h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	I2C_SLV0_DO[7:0]	Data out when slave 0 is set to write.

11.8 I2C_SLV1_ADDR

Name: I2C_SLV1_ADDR
Address: 7 (07h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	I2C_SLV1_RNW	1 – Transfer is a read. 0 – Transfer is a write.
6:0	I2C_ID_1[6:0]	Physical address of I²C slave 1.

11.9 I2C_SLV1_REG

Name: I2C_SLV1_REG
Address: 8 (08h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	I2C_SLV1_REG[7:0]	I^2C slave 1 register address from where to begin data transfer.

11.10 I2C_SLV1_CTRL

Name: I2C_SLV1_CTRL
Address: 9 (09h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	I2C_SLV1_EN	1 – Enable reading data from this slave at the sample rate and storing data at the first available EXT_SENS_DATA register as determined by I2C_SLV0_EN and I2C_SLV0_LENG. 0 – Function is disabled for this slave.
6	I2C_SLV1_BYTE_SW	1 – Swap bytes when reading both the low and high byte of a word. Note there is nothing to swap after reading the first byte if I2C_SLV1_REG[0] = 1, or if the last byte read has a register address lsb = 0. For example, if I2C_SLV0_EN = 0x1, and I2C_SLV0_LENG = 0x3 (to show swap has to do with I2C slave address not EXT_SENS_DATA address), and if I2C_SLV1_REG = 0x1, and I2C_SLV1_LENG = 0x4: 1) The first byte read from address 0x1 will be stored at EXT_SENS_DATA_03 (slave 0's data will be in EXT_SENS_DATA_00, EXT_SENS_DATA_01, and EXT_SENS_DATA_02), 2) the second and third bytes will be read and swapped, so the data read from address 0x2 will be stored at EXT_SENS_DATA_04, and the data read from address 0x3 will be stored at EXT_SENS_DATA_05, 3) The last byte read from address 0x4 will be stored at EXT_SENS_DATA_06. 0 – No swapping occurs, bytes are written in order read.
5	I2C_SLV1_REG_DIS	When set, the transaction does not write a register value, it will only read data, or write data.
4	I2C_SLV1_GRP	External sensor data typically comes in as groups of two bytes. This bit is used to determine if the groups are from the slave's register address 0 and 1, 2 and 3, etc.., or if the groups are address 1 and 2, 3 and 4, etc. 0 indicates slave register addresses 0 and 1 are grouped together (odd numbered register ends the group). 1 indicates slave register addresses 1 and 2 are grouped together (even numbered register ends the group). This allows byte swapping of registers that are grouped starting at any address.
3:0	I2C_SLV1_LENG[3:0]	Number of bytes to be read from I^2C slave 1.

11.11 I2C_SLV1_DO

Name: I2C_SLV1_DO **Address: 10 (0Ah)** **Type: USR3** **Bank: 3** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:0	I2C_SLV1_DO[7:0]	Data out when slave 1 is set to write.

11.12 I2C_SLV2_ADDR

Name: I2C_SLV2_ADDR **Address: 11 (0Bh)** **Type: USR3** **Bank: 3** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7	I2C_SLV2_RNW	1 – Transfer is a read. 0 – Transfer is a write.
6:0	I2C_ID_2[6:0]	Physical address of I^{2}C slave 2.

11.13 I2C_SLV2_REG

Name: I2C_SLV2_REG **Address: 12 (0Ch)** **Type: USR3** **Bank: 3** **Serial IF: R/W** **Reset Value: 0x00**		
BIT	**NAME**	**FUNCTION**
7:0	I2C_SLV2_REG[7:0]	I^{2}C slave 2 register address from where to begin data transfer.

11.14 I2C_SLV2_CTRL

Name: I2C_SLV2_CTRL
Address: 13 (0Dh)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	I2C_SLV2_EN	1 – Enable reading data from this slave at the sample rate and storing data at the first available EXT_SENS_DATA register as determined by I2C_SLV0_EN, I2C_SLV0_LENG, I2C_SLV1_EN and I2C_SLV1_LENG. 0 – Function is disabled for this slave.
6	I2C_SLV2_BYTE_SW	1 – Swap bytes when reading both the low and high byte of a word. Note there is nothing to swap after reading the first byte if I2C_SLV2_REG[0] = 1, or if the last byte read has a register address lsb = 0. See I2C_SLV1_CTRL for an example. 0 – No swapping occurs, bytes are written in order read.
5	I2C_SLV2_REG_DIS	When set, the transaction does not write a register value, it will only read data, or write data.
4	I2C_SLV2_GRP	External sensor data typically comes in as groups of two bytes. This bit is used to determine if the groups are from the slave's register address 0 and 1, 2 and 3, etc.., or if the groups are address 1 and 2, 3 and 4, etc. 0 indicates slave register addresses 0 and 1 are grouped together (odd numbered register ends the group). 1 indicates slave register addresses 1 and 2 are grouped together (even numbered register ends the group). This allows byte swapping of registers that are grouped starting at any address.
3:0	I2C_SLV2_LENG[3:0]	Number of bytes to be read from I²C slave 2.

11.15 I2C_SLV2_DO

Name: I2C_SLV2_DO
Address: 14 (0Eh)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	I2C_SLV2_DO[7:0]	Data out when slave 2 is set to write.

11.16 I2C_SLV3_ADDR

Name: I2C_SLV3_ADDR
Address: 15 (0Fh)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	I2C_SLV3_RNW	1 – Transfer is a read. 0 – Transfer is a write.
6:0	I2C_ID_3[6:0]	Physical address of I²C slave 3.

11.17 I2C_SLV3_REG

Name: I2C_SLV3_REG
Address: 16 (10h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	I2C_SLV3_REG[7:0]	I^2C slave 3 register address from where to begin data transfer.

11.18 I2C_SLV3_CTRL

Name: I2C_SLV3_CTRL
Address: 17 (11h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	I2C_SLV3_EN	1 – Enable reading data from this slave at the sample rate and storing data at the first available EXT_SENS_DATA register as determined by I2C_SLV0_EN, I2C_SLV0_LENG, I2C_SLV1_EN, I2C_SLV1_LENG, I2C_SLV2_EN and I2C_SLV2_LENG. 0 – Function is disabled for this slave.
6	I2C_SLV3_BYTE_SW	1 – Swap bytes when reading both the low and high byte of a word. Note there is nothing to swap after reading the first byte if I2C_SLV3_REG[0] = 1, or if the last byte read has a register address lsb = 0. See I2C_SLV1_CTRL for an example. 0 – No swapping occurs, bytes are written in order read.
5	I2C_SLV3_REG_DIS	When set, the transaction does not write a register value, it will only read data, or write data.
4	I2C_SLV3_GRP	External sensor data typically comes in as groups of two bytes. This bit is used to determine if the groups are from the slave's register address 0 and 1, 2 and 3, etc.., or if the groups are address 1 and 2, 3 and 4, etc. 0 indicates slave register addresses 0 and 1 are grouped together (odd numbered register ends the group). 1 indicates slave register addresses 1 and 2 are grouped together (even numbered register ends the group). This allows byte swapping of registers that are grouped starting at any address.
3:0	I2C_SLV3_LENG[3:0]	Number of bytes to be read from I^2C slave 3.

11.19 I2C_SLV3_DO

Name: I2C_SLV3_DO
Address: 18 (12h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	I2C_SLV3_DO[7:0]	Data out when slave 3 is set to write.

11.20 I2C_SLV4_ADDR

Name: I2C_SLV4_ADDR
Address: 19 (13h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	I2C_SLV4_RNW	1 – Transfer is a read. 0 – Transfer is a write.
6:0	I2C_ID_4[6:0]	Physical address of I²C slave 4.

NOTE: The I²C Slave 4 interface can be used to perform only single byte read and write transactions.

11.21 I2C_SLV4_REG

Name: I2C_SLV4_REG
Address: 20 (14h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	I2C_SLV4_REG[7:0]	I²C slave 4 register address from where to begin data transfer.

11.22 I2C_SLV4_CTRL

Name: I2C_SLV4_CTRL
Address: 21 (15h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7	I2C_SLV4_EN	1 – Enable data transfer with this slave at the sample rate. If read command, store data in I2C_SLV4_DI register, if write command, write data stored in I2C_SLV4_DO register. Bit is cleared when a single transfer is complete. Be sure to write I2C_SLV4_DO first. 0 – Function is disabled for this slave.
6	I2C_SLV4_INT_EN	1 – Enables the completion of the I2C slave 4 data transfer to cause an interrupt. 0 – Completion of the I2C slave 4 data transfer will not cause an interrupt.
5	I2C_SLV4_REG_DIS	When set, the transaction does not write a register value, it will only read data, or write data.
4:0	I2C_SLV4_DLY[4:0]	When enabled via the I2C_MST_DELAY_CTRL, those slaves will only be enabled every1/(1+I2C_SLV4_DLY) samples as determined by I2C_MST_ODR_CONFIG.

11.23 I2C_SLV4_DO

Name: I2C_SLV4_DO
Address: 22 (16h)
Type: USR3
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	I2C_SLV4_DO[7:0]	Data out when slave 4 is set to write.

11.24 I2C_SLV4_DI

Name: I2C_SLV4_DI
Address: 23 (17h)
Type: USR3
Bank: 3
Serial IF: R
Reset Value: 0x00

BIT	NAME	FUNCTION
7:0	I2C_SLV4_DI[7:0]	Data read from I^2C Slave 4.

11.25 REG_BANK_SEL

Name: REG_BANK_SEL
Address: 127 (7Fh)
Type:
Bank: 3
Serial IF: R/W
Reset Value: 0x00

BIT	NAME	FUNCTION
7:6	-	Reserved.
5:4	USER_BANK[1:0]	Use the following values in this bit-field to select a USER BANK. 0: Select USER BANK 0. 1: Select USER BANK 1. 2: Select USER BANK 2. 3: Select USER BANK 3.
3:0	-	Reserved.

12 REGISTER MAP FOR MAGNETOMETER

The register map for the ICM-20948's Magnetometer (AK09916) section is listed below.

NAME	ADDRESS	READ/WRITE	DESCRIPTION	BIT WIDTH	EXPLANATION
WIA2	01H	READ	Device ID	8	
ST1	10H	READ	Status 1	8	Data status
HXL	11H	READ	Measurement data	8	X-axis data
HXH	12H			8	
HYL	13H			8	Y-axis data
HYH	14H			8	
HZL	15H			8	Z-axis data
HZH	16H			8	
ST2	18H	READ	Status 2	8	Data status
CNTL2	31H	READ/ WRITE	Control 2	8	Control Settings
CNTL3	32H	READ/ WRITE	Control 3	8	Control Settings
TS1	33H	READ/ WRITE	Test	8	DO NOT ACCESS
TS2	34H	READ/ WRITE	Test	8	DO NOT ACCESS

Table 20. Register Table for Magnetometer

Addresses 00h to 18h, 30h to 32h are compliant with automatic increment function of serial interface respectively. In other modes, read data is not correct. When the address is in 00h to 18h, the address is incremented 00h ➔ 01h ➔ 02h ➔ 03h ➔ 10h ➔ 11h ➔... ➔ 18h, and the address goes back to 00h after 18h. When the address is in 30h to 32h, the address goes back to 30h after 32h.

12.1 REGISTER MAP DESCRIPTION

ADDR	REGISTER NAME	D7	D6	D5	D4	D3	D2	D1	D0
READ-ONLY REGISTER									
01H	WIA2	0	0	0	0	1	0	0	1
10H	ST1	0	0	0	0	0	0	DOR	DRDY
11H	HXL	HX7	HX6	HX5	HX4	HX3	HX2	HX1	HX0
12H	HXH	HX15	HX14	HX13	HX12	HX11	HX10	HX9	HX8
13H	HYL	HY7	HY6	HY5	HY4	HY3	HY2	HY1	HY0
14H	HYH	HY15	HY14	HY13	HY12	HY11	HY10	HY9	HY8
15H	HZL	HZ7	HZ6	HZ5	HZ4	HZ3	HZ2	HZ1	HZ0
16H	HZH	HZ15	HZ14	HZ13	HZ12	HZ11	HZ10	HZ9	HZ8
18H	ST2	0	RSV30	RSV29	RSV28	HOFL	0	0	0
WRITE/READ REGISTER									
31H	CNTL2	0	0	0	MODE4	MODE3	MODE2	MODE1	MODE0
32H	CNTL3	0	0	0	0	0	0	0	SRST
33H	TS1	-	-	-	-	-	-	-	-
34H	TS2	-	-	-	-	-	-	-	-

Table 21. Register Map for Magnetometer

When VDD is turned ON, POR function works and all registers of AK09916 are initialized.

TS1 and TS2 are test registers for shipment test. Do not access these registers.

13 DETAILED DESCRIPTIONS FOR MAGNETOMETER REGISTERS

This section details each register within the ICM-20948 Magnetometer section.

13.1 WIA: DEVICE ID

ADDR	REGISTER NAME	D7	D6	D5	D4	D3	D2	D1	D0
READ-ONLY REGISTER									
01H	WIA	0	0	0	0	1	0	0	1

Device ID of AK09916. It is described in one byte and fixed value.

09H: fixed

13.2 ST1: STATUS 1

ADDR	REGISTER NAME	D7	D6	D5	D4	D3	D2	D1	D0
READ-ONLY REGISTER									
10H	ST1	0	0	0	0	0	0	DOR	DRDY
Reset		0	0	0	0	0	0	0	0

DRDY: Data Ready

“0”: Normal

“1”: Data is ready

DRDY bit turns to “1” when data is ready in Single measurement mode, Continuous measurement mode 1, 2, 3, 4 or Self-test mode. It returns to “0” when any one of ST2 register or measurement data register (HXL to TMPS) is read.

DOR: Data Overrun

“0”: Normal

“1”: Data overrun

DOR bit turns to “1” when data has been skipped in Continuous measurement mode 1, 2, 3, 4. It returns to “0” when any one of ST2 register or measurement data register (HXL to TMPS) is read.

13.3 HXL TO HZH: MEASUREMENT DATA

ADDR	REGISTER NAME	D7	D6	D5	D4	D3	D2	D1	D0
READ-ONLY REGISTER									
11H	HXL	HX7	HX6	HX5	HX4	HX3	HX2	HX1	HX0
12H	HXH	HX15	HX14	HX13	HX12	HX11	HX10	HX9	HX8
13H	HYL	HY7	HY6	HY5	HY4	HY3	HY2	HY1	HY0
14H	HYH	HY15	HY14	HY13	HY12	HY11	HY10	HY9	HY8
15H	HZL	HZ7	HZ6	HZ5	HZ4	HZ3	HZ2	HZ1	HZ0
16H	HZH	HZ15	HZ14	HZ13	HZ12	HZ11	HZ10	HZ9	HZ8
Reset		0	0	0	0	0	0	0	0

Measurement data of magnetic sensor X-axis/Y-axis/Z-axis

HXL[7:0]: X-axis measurement data lower 8bit

HXH[15:8]: X-axis measurement data higher 8bit

HYL[7:0]: Y-axis measurement data lower 8bit

HYH[15:8]: Y-axis measurement data higher 8bit

HZL[7:0]: Z-axis measurement data lower 8bit

HZH[15:8]: Z-axis measurement data higher 8bit

Measurement data is stored in two's complement and Little Endian format. Measurement range of each axis is from --32752 to 32752 in 16-bit output.

MEASUREMENT DATA (EACH AXIS) [15:0]			MAGNETIC FLUX DENSITY [µT]
TWO'S COMPLEMENT	HEX	DECIMAL	
0111 1111 1111 0000	7FF0	32752	4912(max.)
|	|	|	|
0000 0000 0000 0001	0001	1	0.15
0000 0000 0000 0000	0000	0	0
1111 1111 1111 1111	FFFF	-1	-0.15
|	|	|	|
1000 0000 0001 0000	8010	-32752	-4912(min.)

Table 22. Magnetometer Measurement Data Format

13.4 ST2: STATUS 2

ADDR	REGISTER NAME	D7	D6	D5	D4	D3	D2	D1	D0
READ-ONLY REGISTER									
18H	ST2	0	RSV30	RSV29	RSV28	HOFL	0	0	0
Reset		0	0	0	0	0	0	0	0

ST2[6:4] bits: Reserved register for AKM.

HOFL: Magnetic sensor overflow

"0": Normal

"1": Magnetic sensor overflow occurred

In Single measurement mode, Continuous measurement mode 1, 2, 3, 4, and Self-test mode, magnetic sensor may overflow even though measurement data register is not saturated. In this case, measurement data is not correct and HOFL bit turns to "1". When measurement data register is updated, HOFL bit is updated.

ST2 register has a role as data reading end register, also. When any of measurement data register (HXL to TMPS) is read in Continuous measurement mode 1, 2, 3, 4, it means data reading start and taken as data reading until ST2 register is read. Therefore, when any of measurement data is read, be sure to read ST2 register at the end.

13.5 CNTL2: CONTROL 2

ADDR	REGISTER NAME	D7	D6	D5	D4	D3	D2	D1	D0
READ/WRITE REGISTER									
31H	CNTL2	0	0	0	MPDE4	MODE3	MODE2	MODE1	MODE0
Reset		0	0	0	0	0	0	0	0

MODE[4:0] bits: Operation mode setting

"00000": Power-down mode

"00001": Single measurement mode

"00010": Continuous measurement mode 1

"00100": Continuous measurement mode 2

"00110": Continuous measurement mode 3

"01000": Continuous measurement mode 4

"10000": Self-test mode

Other code settings are prohibited

When each mode is set, AK09916 transits to the set mode.

13.6 CNTL3: CONTROL 3

ADDR	REGISTER NAME	D7	D6	D5	D4	D3	D2	D1	D0
READ/WRITE REGISTER									
32H	CNTL3	0	0	0	0	0	0	0	SRST
Reset		0	0	0	0	0	0	0	0

SRST: Soft reset

"0": Normal

"1": Reset

When "1" is set, all registers are initialized. After reset, SRST bit turns to "0" automatically.

13.7 TS1, TS2: TEST 1, 2

ADDR	REGISTER NAME	D7	D6	D5	D4	D3	D2	D1	D0
READ/WRITE REGISTER									
33H	TS1	-	-	-	-	-	-	-	-
34H	TS2	-	-	-	-	-	-	-	-
Reset		0	0	0	0	0	0	0	0

TS1 and TS2 registers are test registers for shipment test. Do not use these registers.

14 USE NOTES

14.1 GYROSCOPE MODE TRANSITION

When gyroscope is transitioning from low-power to low-noise mode, several unsettled output samples will be observed at the gyroscope output due to filter switching and settling. The number of unsettled gyroscope output samples depends on the filter and ODR settings.

14.2 POWER MANAGEMENT 1 REGISTER SETTING

CLKSEL[2:0] has to be set to 001 to achieve the datasheet performance.

14.3 DMP MEMORY ACCESS

Reading/writing DMP memory and FIFO through I^2C in a multithreaded environment can cause wrong data being read. To avoid the issue, one may use SPI instead of I^2C, or use I^2C with mutexes.

14.4 TIME BASE CORRECTION

The system clock frequency at room temperature in gyroscope mode and 6-Axis mode varies from part to part, and the clock rates specified in datasheet are the nominal values. The percentage of frequency deviation from the nominal values for each part is logged in register TIMEBASE_CORRECTION_PLL, and the range of the code is ±10% with each LSB representing a step of 0.079%. For example, if on one part TIMEBASE_CORRECTION_PLL = 0x0C = d’12, it means the clock frequency in gyroscope mode and 6-Axis mode is ~0.94% faster than the nominal value.

When operating in accelerometer-only mode, the system clock frequency at room temperature is the nominal frequency over parts, and it is independent of the value stored in TIMEBASE_CORRECTION_PLL register.

14.5 I^2C MASTER CLOCK FREQUENCY

I^2C master clock frequency can be set by register I2C_MST_CLK as shown in Table 23. Due to temperature variation and part to part variation of system clock frequency in different power modes, I2C_MST_CLK should be set such that in all conditions the clock frequency will not exceed what a slave device can support. To achieve a targeted clock frequency of 400 kHz, MAX, it is recommended to set I2C_MST_CLK = 7 (345.6 kHz / 46.67% duty cycle).

I2C_MST_CLK	NOMINAL CLK FREQUENCY [KHZ]	DUTY CYCLE
0	370.29	50.00%
1	-	-
2	370.29	50.00%
3	432.00	50.00%
4	370.29	42.86%
5	370.29	50.00%
6	345.60	40.00%
7	345.60	46.67%
8	304.94	47.06%
9	432.00	50.00%
10	432.00	41.67%
11	432.00	41.67%
12	471.27	45.45%
13	432.00	50.00%

I2C_MST_CLK	NOMINAL CLK FREQUENCY [KHZ]	DUTY CYCLE
14	345.60	46.67%
15	345.60	46.67%

Table 23. I²C Master Clock Frequency

14.6 CLOCKING

The internal system clock sources include: (1) an internal relaxation oscillator, and (2) a PLL with MEMS gyroscope oscillator as the reference clock. With the recommended clock selection setting (CLKSEL = 1), the best clock source for optimum sensor performance and power consumption will be automatically selected based on the power mode. Specifically, the internal relaxation oscillator will be selected when operating in accelerometer only mode, while the PLL will be selected whenever gyroscope is on, which includes gyroscope and 6-axis modes.

As clock accuracy is critical to the preciseness of distance and angle calculations performed by DMP, it should be noted that the internal relaxation oscillator and PLL show different performances in some aspects. The internal relaxation oscillator is trimmed to have a consistent operating frequency at room temperature, while the PLL clock frequency varies from part to part. The PLL frequency deviation from the nominal value in percentage is captured in register TIMEBASE_CORRECTION_PLL, and users can factor it in during distance and angle calculations to not sacrifice accuracy. Other than that, PLL has better frequency stability and lower frequency variation over temperature than the internal relaxation oscillator.

14.7 LP_EN BIT-FIELD USAGE

The LP_EN bit-field (User Bank 0, PWR_MGMT_1 register, bit [5] helps to reduce the digital current. The recommended setting for this bit-field is 1 to achieve the lowest possible current. However, when LP_EN is set to 1, user may not be able to write to the following registers. If it is desired to write to registers in this list, it is recommended to first set LP_EN=0, write the desired register(s), then set LP_EN=1 again:

- USER BANK 0: All registers except LP_CONFIG, PWR_MGMT_1, PWR_MGMT_2, INT_PIN_CFG, INT_ENABLE, FIFO_COUNTH, FIFO_COUNTL, FIFO_R_W, FIFO_CFG, REG_BANK_SEL
- USER BANK 1: All registers except REG_BANK_SEL
- USER BANK 2: All registers except REG_BANK_SEL
- USER BANK 3: All registers except REG_BANK_SEL

14.8 REGISTER ACCESS USING SPI INTERFACE

Using the SPI interface, when the AP/user disables the gyroscope sensor (User Bank 0, PWR_MGMT_2 register, bits [2:0]=111) as part of a sequence of register read or write commands, the AP/user will be required to subsequently wait 22 µs prior to any of the following operations:

(1) Writing to any of the following registers:

- USER BANK 0: All registers except LP_CONFIG, PWR_MGMT_1, PWR_MGMT_2, INT_PIN_CFG, INT_ENABLE, FIFO_COUNTH, FIFO_COUNTL, FIFO_R_W, FIFO_CFG, REG_BANK_SEL
- USER BANK 1: All registers except REG_BANK_SEL
- USER BANK 2: All registers except REG_BANK_SEL
- USER BANK 3: All registers except REG_BANK_SEL

(2) Reading data from FIFO
(3) Reading from memory

15 ORIENTATION OF AXES

Figure 12 and Figure 13 show the orientation of the axes of sensitivity and the polarity of rotation. Note the pin 1 identifier (•) in the figures.

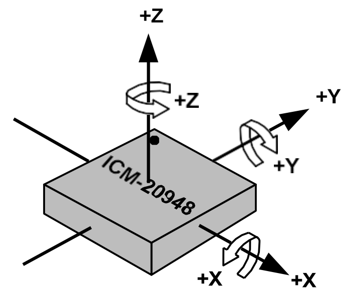


Figure 12. Orientation of Axes of Sensitivity and Polarity of Rotation

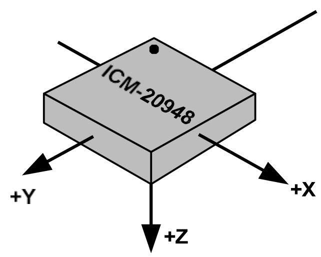


Figure 13. Orientation of Axes of Sensitivity for Magnetometer

16 PACKAGE DIMENSIONS

This section provides package dimensions for the device. Information for the 24 Lead QFN 3.0x3.0x0.9 package is in Figure 14 and Table 24

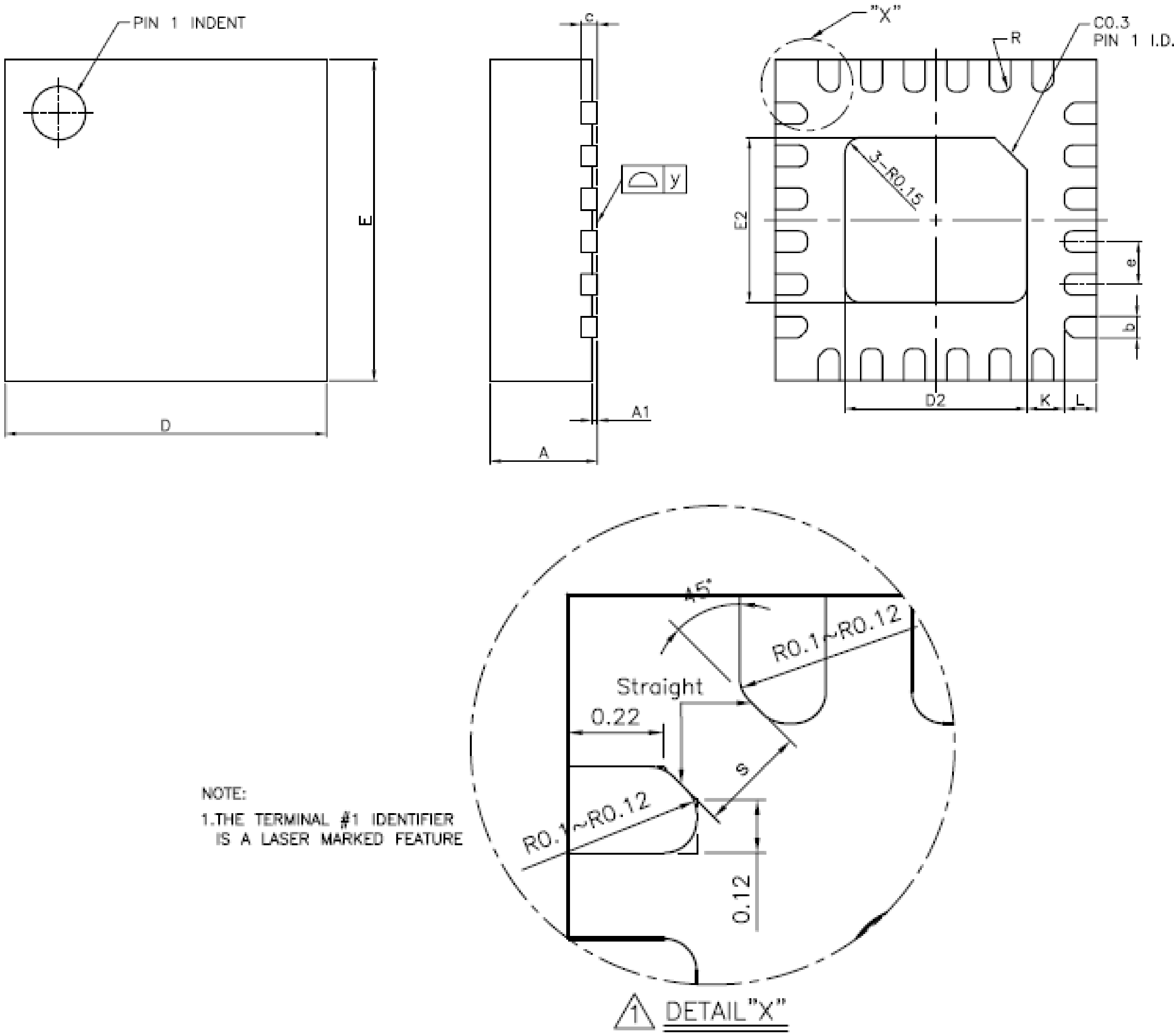


Figure 14. Package Dimensions

SYMBOLS	DIMENSIONS IN MILLIMETERS		
	MIN.	NOM.	MAX.
A	0.95	1.00	1.05
A1	0.00	0.02	0.05
b	0.15	0.20	0.25
c	---	0.15 REF.	---
D	2.90	3.00	3.10
D2	1.65	1.70	1.75
E	2.90	3.00	3.10
E2	1.49	1.54	1.59
e	---	0.40	---
K	---	0.35 REF.	---
L	0.25	0.30	0.35
R	0.075	REF.	---
s	---	0.25 REF.	---
y	0.00	---	0.075

Table 24. Package Dimensions

17 PART NUMBER PART MARKINGS

The part number part markings for ICM-20948 devices are summarized below:

PART NUMBER	PART NUMBER PART MARKING
ICM-20948	I2948

Table 25. Part Number Part Markings

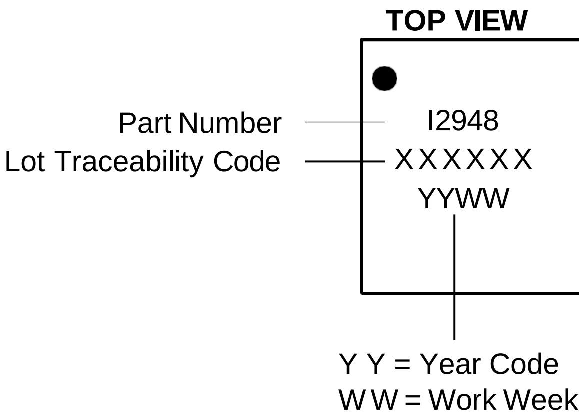


Figure 15. Part Number Part Markings

18 REFERENCES

Please refer to "InvenSense MEMS Handling Application Note (AN-IVS-0002A-00)" for the following information:

- Manufacturing Recommendations
  - Assembly Guidelines and Recommendations
  - PCB Design Guidelines and Recommendations
  - MEMS Handling Instructions
  - ESD Considerations
  - Reflow Specification
  - Storage Specifications
  - Package Marking Specification
  - Tape & Reel Specification
  - Reel & Pizza Box Label
  - Packaging
  - Representative Shipping Carton Label
- Compliance
  - Environmental Compliance
  - DRC Compliance
  - Compliance Declaration Disclaimer

19 DOCUMENT INFORMATION

19.1 REVISION HISTORY

REVISION DATE	REVISION	DESCRIPTION
12/07/2016	1.0	Initial Release
1/17/2017	1.1	Formatting fix
04/06/2017	1.2	Updated Section 4
06/02/2017	1.3	Updated Sections 3, 4
07/01/2021	1.4	Updated FIFO size information (Sections 1.2, 4.16); Updated Tables 4, 9
09/02/2021	1.5	Updated step size information (Sections 9.7 to 9.12, 10.4 to 10.9); Updated Noise values (Tables 17, 19)

COMPLIANCE DECLARATION DISCLAIMER

InvenSense believes the environmental and other compliance information given in this document to be correct but cannot guarantee accuracy or completeness. Conformity documents substantiating the specifications and component characteristics are on file. InvenSense subcontracts manufacturing and the information contained herein is based on data received from vendors and suppliers, which has not been validated by InvenSense.

Anexo 13
Código

src\main.cpp

```cpp
// Código para el proyecto del rover Asturiosity
// Alba González Fernández

//*DECLARACIÓN DE LIBRERÍAS
#include <Arduino.h>

#include <Servo.h>         // Librería para funcionamiento de servomotores
#include <ServoEasing.hpp> // proporciona una forma fácil de crear movimientos de servo suaves y personalizados.
// Esta librería utiliza algoritmos de interpolación para calcular el movimiento de los servos de manera precisa y suave,
// permitiendo a los usuarios crear patrones de movimiento específicos como ondas sinusoidales, ondas cuadradas, rampas, escalones, etc.
// Además, la librería ofrece la capacidad de personalizar la curva de movimiento para adaptarse a diferentes requisitos.
#include <IBusBM.h> // Librería para comunicación con el mando y el receptor
// #include <AccelStepper.h>
// #include <SoftwareSerial.h>
#include <Adafruit_NeoPixel.h> // Librería para la utilización de la tira LED

#include <Adafruit_ICM20X.h>
#include <Adafruit_ICM20948.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>

//* DECLARACIÓN DE VARIABLES

// Variables de los motores DC
#define motorW1_IN1 7
#define motorW1_IN2 6
#define motorW2_IN1 5
#define motorW2_IN2 4
#define motorW3_IN1 3
#define motorW3_IN2 2
#define motorW4_IN1 10
#define motorW4_IN2 13
#define motorW5_IN1 8
#define motorW5_IN2 9
#define motorW6_IN1 12
#define motorW6_IN2 11

// Declaraciónd de servomotores
ServoEasing servoW1;
ServoEasing servoW3;
ServoEasing servoW4;
ServoEasing servoW6;

// Declaración de libreria IBus
IBusBM IBus;
IBusBM IBusSensor;

// Declaración de variables de control de velocidad y giro
int angle = 0;                          // posición del servo en grados
int ch0, ch1, ch2, ch3, ch4, ch5 = 0; // canales del mando
int servo1Angle = 90;                   // En el instante primero de encendido los 4 servos se colocan a 90º
int servo3Angle = 90;
```

```
int servo4Angle = 90;
int servo6Angle = 90;
int s = 0; // velocidad del rover
int r = 0; // radio de giro

float s1, s2, s3 = 0;                                                      // declaración de las 3 velocidades distintas que puede adquirir el rover
float s1PWM, s2PWM, s3PWM = 0;                                             // Velocidades de los PWM escaladas
float thetaInnerFront, thetaInnerBack, thetaOuterFront, thetaOuterBack = 0; // Variables para el cáculo de giro de los servomotores

// Punto de referencia para el giro o radio de giro
float d1 = 271; // distancia in mm
float d2 = 278;
float d3 = 301;
float d4 = 304;

int valor_0; // Valor en el que se inicializa el mando
int valor_1; // Valor que cambia cuando detecta mando conectado
bool flag = true;
#define STOP_SIGNAL_FRONT 38 // Pin digital que recibe datos del sensor delantero
#define STOP_SIGNAL_BACK 34  // Pin digital que recibe datos del sensor posterior
int STOP_MOTOR_FRONT;        // Variable de parada detectado para el sensor delantero
int STOP_MOTOR_BACK;         // Variable de parada detectado para el sensor posterior
bool one_time_front = true;
bool one_time_back = true;

#define PIN_LED 40 // Pin digital para la señal de la tira LED
#define NUM_LEDS 7 // Número de LEDs de la tira
bool ledRojo_front = true;
bool ledRojo_back = true;

// Variable de estado para cada uno de los movimientos posibles
bool motor_stop = false;
bool motor_forward = false;
bool motor_backward = false;
bool motor_Forward_right = false;
bool motor_Backward_right = false;
bool motor_Backward_left = false;
bool motor_Forward_left = false;
bool motor_signal_front = false;
bool motor_signal_back = false;

// Crea un objeto de la clase Adafruit_NeoPixel
Adafruit_NeoPixel strip = Adafruit_NeoPixel(NUM_LEDS, PIN_LED, NEO_GRB + NEO_KHZ800);

// IMU

Adafruit_ICM20948 icm;
uint16_t measurement_delay_us = 65535; // Delay between measurements for testing
// For SPI mode, we need a CS pin
#define ICM_CS 43
// For software-SPI mode we need SCK/MOSI/MISO pins
#define ICM_SCK 21
#define ICM_MISO 42
#define ICM_MOSI 20

bool ledAzul = false;
int tiempo_ledAzul;

```

```cpp
sensors_event_t accel;
sensors_event_t gyro;
sensors_event_t mag;
sensors_event_t temp;

//* DECLARACIÓN DE FUNCIONES

// Calcular las 3 velocidades que puede tener el rover segun el radio de giro
void calculateMotorsSpeed(int s, int s1, int s2, int s3)
{
    //  si no se realiza ningún giro, todos los motores tienen la misma velocidad
    if (IBus.readChannel(0) > 1450 && IBus.readChannel(0) < 1550)
    {
        s1 = s2 = s3 = s;
    }
    // cuando se produce giro, la velocidad de los motores depende del radio de giro
    else
    {
        // Los motores que esten por el exterior del giro tendrán mayor radio de giro y por tanto máxima velocidad
        // Debido a la geometría del rover, los tres motores con mayor radio de giro tendrán casi la misma velocidad, la diferencia es solo 1% por lo que se asume misma velocidad

        s1 = s;
        // Las ruedas de delante y de detrás internas al giro tendrán el menor radio de giro y menores velocidades que las externar al radio de giro

        s2 = s * sqrt(pow(d3, 2) + pow((r - d1), 2)) / (r + d4);
        // La el motor en la posición intermedia es el más próximo al radio de giro y por lo tanto tendr´la menor velocidad

        s3 = s * (r - d4) / (r + d4);
    }

    // Valor de la velocidad de 0 a 100% a valor PWM desde 0 a 255
    s1PWM = map(round(s1), 0, 100, 0, 255);
    s2PWM = map(round(s2), 0, 100, 0, 255);
    s3PWM = map(round(s3), 0, 100, 0, 255);
}

//* Cálculo angulo de los servos
void calculateServoAngle()
{
    // Calcular el ángulo para cada servomotor para el radio de giro "r"

    thetaInnerFront = round((atan((d3 / (r + d1)))) * 180 / PI);
    thetaInnerBack = round((atan((d2 / (r + d1)))) * 180 / PI);
    thetaOuterFront = round((atan((d3 / (r - d1)))) * 180 / PI);
    thetaOuterBack = round((atan((d2 / (r - d1)))) * 180 / PI);
}

//* Función para la parada de motores
void motorStop()
{
    // DC Motors
    // Motor Wheel 1 - Left Front
    digitalWrite(motorW1_IN1, LOW); // PWM value
    digitalWrite(motorW1_IN2, LOW); // Forward
    // Motor Wheel 2 - Left Middle
    digitalWrite(motorW2_IN1, LOW);
```

```
    digitalWrite(motorW2_IN2, LOW);
    // Motor Wheel 3 - Left Back
    digitalWrite(motorW3_IN1, LOW);
    digitalWrite(motorW3_IN2, LOW);
    // right side motors move in opposite direction
    // Motor Wheel 4 - Right Front
    digitalWrite(motorW4_IN1, LOW);
    digitalWrite(motorW4_IN2, LOW);
    // Motor Wheel 5 - Right Middle
    digitalWrite(motorW5_IN1, LOW);
    digitalWrite(motorW5_IN2, LOW);
    // Motor Wheel 6 - Right Back
    digitalWrite(motorW6_IN1, LOW);
    digitalWrite(motorW6_IN2, LOW);
    // motor_stop = true;
}

//* Función de movimiento hacia delante y recto

void motorForward()
{
    // Motor Wheel 1 - Left Front
    analogWrite(motorW1_IN1, s1PWM); // all wheels move at the same speed
    digitalWrite(motorW1_IN2, LOW);  // Forward
    // Motor Wheel 2 - Left Middle
    analogWrite(motorW2_IN1, s1PWM);
    digitalWrite(motorW2_IN2, LOW);
    // Motor Wheel 3 - Left Back
    analogWrite(motorW3_IN1, s1PWM);
    digitalWrite(motorW3_IN2, LOW);
    // right side motors move in opposite direction
    // Motor Wheel 4 - Right Front
    digitalWrite(motorW4_IN1, LOW);
    analogWrite(motorW4_IN2, s1PWM);
    // Motor Wheel 5 - Right Middle
    digitalWrite(motorW5_IN1, LOW);
    analogWrite(motorW5_IN2, s1PWM);
    // Motor Wheel 6 - Right Back
    digitalWrite(motorW6_IN1, LOW);
    analogWrite(motorW6_IN2, s1PWM);
    // motor_forward = true;
}

//* Función de movimiento hacia atrás y recto
void motorBackward()
{
    // Motor Wheel 1 - Left Front
    digitalWrite(motorW1_IN1, LOW);  // all wheels move at the same speed
    analogWrite(motorW1_IN2, s1PWM); // Forward
    // Motor Wheel 2 - Left Middle
    digitalWrite(motorW2_IN1, LOW);
    analogWrite(motorW2_IN2, s1PWM);
    // Motor Wheel 3 - Left Back
    digitalWrite(motorW3_IN1, LOW);
    analogWrite(motorW3_IN2, s1PWM);
    // right side motors move in opposite direction
    // Motor Wheel 4 - Right Front
    analogWrite(motorW4_IN1, s1PWM);
    digitalWrite(motorW4_IN2, LOW);
    // Motor Wheel 5 - Right Middle
```

```
    analogWrite(motorW5_IN1, s1PWM);
    digitalWrite(motorW5_IN2, LOW);
    // Motor Wheel 6 - Right Back
    analogWrite(motorW6_IN1, s1PWM);
    digitalWrite(motorW6_IN2, LOW);
    // motor_backward = true;
}

//* Función de movimiento hacia delante y giro a la derecha
void motorForward_right()
{
    // Move forward right
    // Motor Wheel 1 - Left Front
    analogWrite(motorW1_IN1, s1PWM); // Outer wheels running at speed1 - max speed
    digitalWrite(motorW1_IN2, LOW);
    // Motor Wheel 2 - Left Middle
    analogWrite(motorW2_IN1, s1PWM);
    digitalWrite(motorW2_IN2, LOW);
    // Motor Wheel 3 - Left Back
    analogWrite(motorW3_IN1, s1PWM);
    digitalWrite(motorW3_IN2, LOW);
    // right side motors move in opposite direction
    // Motor Wheel 4 - Right Front
    digitalWrite(motorW4_IN1, LOW);
    analogWrite(motorW4_IN2, s2PWM); // Inner front wheel running at speed2 - lower speed
    // Motor Wheel 5 - Right Middle
    digitalWrite(motorW5_IN1, LOW);
    analogWrite(motorW5_IN2, s3PWM); // Inner middle wheel running at speed3 - lowest speed
    // Motor Wheel 6 - Right Back
    digitalWrite(motorW6_IN1, LOW);
    analogWrite(motorW6_IN2, s2PWM); // Inner back wheel running at speed2 - lower speed
    // motor_Forward_right = true;
}

//* Función de movimiento hacia atrás y giro a la derecha
void motorBackward_right()
{
    // Motor Wheel 1 - Left Front
    digitalWrite(motorW1_IN1, LOW); // Outer wheels running at speed1 - max speed
    analogWrite(motorW1_IN2, s1PWM);
    // Motor Wheel 2 - Left Middle
    digitalWrite(motorW2_IN1, LOW);
    analogWrite(motorW2_IN2, s1PWM);
    // Motor Wheel 3 - Left Back
    digitalWrite(motorW3_IN1, LOW);
    analogWrite(motorW3_IN2, s1PWM);
    // right side motors move in opposite direction
    // Motor Wheel 4 - Right Front
    analogWrite(motorW4_IN1, s2PWM);
    digitalWrite(motorW4_IN2, LOW); // Inner front wheel running at speed2 - lower speed
    // Motor Wheel 5 - Right Middle
    analogWrite(motorW5_IN1, s3PWM);
    digitalWrite(motorW5_IN2, LOW); // Inner middle wheel running at speed3 - lowest speed
    // Motor Wheel 6 - Right Back
    analogWrite(motorW6_IN1, s2PWM);
    digitalWrite(motorW6_IN2, LOW); // Inner back wheel running at speed2 - lower speed
    // motor_Backward_right = true;
}

```

```cpp
//* Función de movimiento hacia delante y giro a la izquierda
void motorForward_left()
{
    // Move forward
    // Motor Wheel 1 - Left Front
    analogWrite(motorW1_IN1, s2PWM); // PWM value
    digitalWrite(motorW1_IN2, LOW);  // Forward
    // Motor Wheel 2 - Left Middle
    analogWrite(motorW2_IN1, s3PWM);
    digitalWrite(motorW2_IN2, LOW);
    // Motor Wheel 3 - Left Back
    analogWrite(motorW3_IN1, s2PWM);
    digitalWrite(motorW3_IN2, LOW);
    // Motor Wheel 4 - Right Front
    // right side motors move in opposite direction
    digitalWrite(motorW4_IN1, LOW);
    analogWrite(motorW4_IN2, s1PWM);
    // Motor Wheel 5 - Right Middle
    digitalWrite(motorW5_IN1, LOW);
    analogWrite(motorW5_IN2, s1PWM);
    // Motor Wheel 6 - Right Back
    digitalWrite(motorW6_IN1, LOW);
    analogWrite(motorW6_IN2, s1PWM);
    // motor_Forward_left = true;
}

//* Función de movimiento hacia atrás y giro a la izquierda
void motorBackward_left()
{
    // Move backward
    // Motor Wheel 1 - Left Front
    digitalWrite(motorW1_IN1, LOW);  // PWM value
    analogWrite(motorW1_IN2, s2PWM); // Forward
    // Motor Wheel 2 - Left Middle
    digitalWrite(motorW2_IN1, LOW);
    analogWrite(motorW2_IN2, s3PWM);
    // Motor Wheel 3 - Left Back
    digitalWrite(motorW3_IN1, LOW);
    analogWrite(motorW3_IN2, s2PWM);
    // Motor Wheel 4 - Right Front
    // right side motors move in opposite direction
    analogWrite(motorW4_IN1, s1PWM);
    digitalWrite(motorW4_IN2, LOW);
    // Motor Wheel 5 - Right Middle
    analogWrite(motorW5_IN1, s1PWM);
    digitalWrite(motorW5_IN2, LOW);
    // Motor Wheel 6 - Right Back
    analogWrite(motorW6_IN1, s1PWM);
    digitalWrite(motorW6_IN2, LOW);
    // motor_Backward_left = true;
}

// Función para establecer el color de todos los LEDs en la tira
void setColor(int red, int green, int blue)
{
    for (int i = 0; i < NUM_LEDS; i++)
    {
        strip.setPixelColor(i, strip.Color(red, green, blue));
    }
    strip.show(); // Actualiza los LEDs con el nuevo color
```

```
}

void IMU()
{

    if (((accel.acceleration.y) < -8.5) || ((accel.acceleration.y) > 8.5) ||
((mag.magnetic.x) > 28) || ((mag.magnetic.x) < -12))
    {
        if (!ledAzul)
        {
            tiempo_ledAzul = millis();  // Guardar el tiempo actual
            setColor(0, 0, 255);        // Establece el color azul
            Serial.println("LED AZUL"); // Establece el color azul
            ledAzul = true;
        }
    }
    else
    {
        if (ledAzul)
        {
            setColor(0, 0, 0);                   // Apagar leds
            Serial.println("LED AZUL APAGADO"); // Establece el color azul// Apagar la
tira de LEDs después de la duración especificada
            ledAzul = false;
        }
    }

    if (ledAzul && millis() - tiempo_ledAzul >= 1000)
    {
        setColor(0, 0, 0);                   // Apagar leds
        Serial.println("LED AZUL APAGADO"); // Establece el color azul// Apagar la tira de
LEDs después de la duración especificada
        ledAzul = false;
    }
}

void setup()
{
    Serial.begin(115200);

    IBus.begin(Serial1, IBUSBM_NOTIMER);       // Servo IBUS
    IBusSensor.begin(Serial2, IBUSBM_NOTIMER); // Sensor IBUS
    IBusSensor.addSensor(IBUSS_INTV);          // sensor de voltaje

    // DC Motors PARADOS
    // Motor Wheel 1 - Left Front
    digitalWrite(motorW1_IN1, LOW); // PWM value
    digitalWrite(motorW1_IN2, LOW); // Forward
    // Motor Wheel 2 - Left Middle
    digitalWrite(motorW2_IN1, LOW);
    digitalWrite(motorW2_IN2, LOW);
    // Motor Wheel 3 - Left Back
    digitalWrite(motorW3_IN1, LOW);
    digitalWrite(motorW3_IN2, LOW);
    // right side motors move in opposite direction
    // Motor Wheel 4 - Right Front
    digitalWrite(motorW4_IN1, LOW);
    digitalWrite(motorW4_IN2, LOW);
    // Motor Wheel 5 - Right Middle
    digitalWrite(motorW5_IN1, LOW);
    digitalWrite(motorW5_IN2, LOW);
```

```cpp
    // Motor Wheel 6 - Right Back
    digitalWrite(motorW6_IN1, LOW);
    digitalWrite(motorW6_IN2, LOW);

    // pines para servomotores
    servoW1.attach(22);
    servoW3.attach(23);
    servoW4.attach(24);
    servoW6.attach(25);

    // Inicializar los servomotres a 90 grados
    servoW1.write(90);
    servoW3.write(90);
    servoW4.write(90);
    servoW6.write(90);

    // Incializar la velocidad de giro de los servomotores
    servoW1.setSpeed(550);
    servoW3.setSpeed(550);
    servoW4.setSpeed(550);
    servoW6.setSpeed(550);

    // Inicializar los pines digitales de la señal de los sensores como entrada
    pinMode(STOP_SIGNAL_FRONT, INPUT);
    pinMode(STOP_SIGNAL_BACK, INPUT);

    // Inicializa la tira de LEDs RGB
    strip.begin();
    strip.show(); // Apaga todos los LEDs al inicio

    // Try to initialize!
    if (!icm.begin_I2C())
    {
        // if (!icm.begin_SPI(ICM_CS)) {
        // if (!icm.begin_SPI(ICM_CS, ICM_SCK, ICM_MISO, ICM_MOSI)) {

        Serial.println("Failed to find ICM20948 chip");
        while (1)
        {
            delay(10);
        }
    }
    Serial.println("ICM20948 Found!");
    icm.setAccelRange(ICM20948_ACCEL_RANGE_16_G);
    Serial.print("Accelerometer range set to: ");
    switch (icm.getAccelRange())
    {
    case ICM20948_ACCEL_RANGE_2_G:
        // Serial.println("+-2G");
        break;
    case ICM20948_ACCEL_RANGE_4_G:
        // Serial.println("+-4G");
        break;
    case ICM20948_ACCEL_RANGE_8_G:
        // Serial.println("+-8G");
        break;
    case ICM20948_ACCEL_RANGE_16_G:
        // Serial.println("+-16G");
        break;
    }
```

```
    Serial.println("OK");

    icm.setGyroRange(ICM20948_GYRO_RANGE_2000_DPS);
    Serial.print("Gyro range set to: ");
    switch (icm.getGyroRange())
    {
    case ICM20948_GYRO_RANGE_250_DPS:
        // Serial.println("250 degrees/s");
        break;
    case ICM20948_GYRO_RANGE_500_DPS:
        // Serial.println("500 degrees/s");
        break;
    case ICM20948_GYRO_RANGE_1000_DPS:
        // Serial.println("1000 degrees/s");
        break;
    case ICM20948_GYRO_RANGE_2000_DPS:
        // Serial.println("2000 degrees/s");
        break;
    }

    //  icm.setAccelRateDivisor(4095);
    uint16_t accel_divisor = icm.getAccelRateDivisor();
    float accel_rate = 1125 / (1.0 + accel_divisor);

    // Serial.print("Accelerometer data rate divisor set to: ");
    Serial.println(accel_divisor);
    // Serial.print("Accelerometer data rate (Hz) is approximately: ");
    Serial.println(accel_rate);

    //  icm.setGyroRateDivisor(255);
    uint8_t gyro_divisor = icm.getGyroRateDivisor();
    float gyro_rate = 1100 / (1.0 + gyro_divisor);

    Serial.print("Gyro data rate divisor set to: ");
    Serial.println(gyro_divisor);
    Serial.print("Gyro data rate (Hz) is approximately: ");
    Serial.println(gyro_rate);

    icm.setMagDataRate(AK09916_MAG_DATARATE_10_HZ);
    Serial.print("Magnetometer data rate set to: ");
    switch (icm.getMagDataRate())
    {
    case AK09916_MAG_DATARATE_SHUTDOWN:
        // Serial.println("Shutdown");
        break;
    case AK09916_MAG_DATARATE_SINGLE:
        // Serial.println("Single/One shot");
        break;
    case AK09916_MAG_DATARATE_10_HZ:
        // Serial.println("10 Hz");
        break;
    case AK09916_MAG_DATARATE_20_HZ:
        // Serial.println("20 Hz");
        break;
    case AK09916_MAG_DATARATE_50_HZ:
        // Serial.println("50 Hz");
        break;
    case AK09916_MAG_DATARATE_100_HZ:
        // Serial.println("100 Hz");
        break;
```

```
    }
    // Serial.println();
}

void loop()
{

    //* Declaración de los canales en los que se leerán los datos que se reciben del trasmisor RC.
    IBus.loop();
    ch0 = IBus.readChannel(0);  // Channel 1 Girar
    ch1 = IBus.readChannel(1);  // Channel 2 Iniciar mando
    ch2 = IBus.readChannel(2);  // Channel 3 Speed
    ch3 = IBus.readChannel(3);  // Channel 4  NO
    ch4 = IBus.readChannel(4);  // Channel 5 sENSORES
    ch5 = IBus.readChannel(5);  // Channel 6 Direction

    // ch1 = 0; // Le damos valor 0 a los canales que no usamos para que si cambian los valores en el mando no cree problemas en el código
    ch3 = 0;

    // //*MOTORES PARADOS INICIALMENTE HASTA RECIBIR SEÑAL DEL MANDO
    // valor_0 = IBus.readChannel(1);
    while (flag)
    {
        IBus.loop();
        valor_0 = IBus.readChannel(1);
        valor_1 = IBus.readChannel(1);
        motorStop();

        if (valor_1 == valor_0)
        {
            valor_1 = IBus.readChannel(1);
            Serial.println(valor_1);
            Serial.println("Conectar mando y mover canal derecho");
            motorStop();
        }

        if (valor_1 > 1800)
        {
            Serial.println("Mando conectado");
            flag = false;
            break;
        }
    }

    //* Convirtiendo datos
    // Girando hacia la derecha
    if (IBus.readChannel(0) > 1550)
    {
        r = map(IBus.readChannel(0), 1550, 2000, 1400, 600); // radio de giro de 1400mm a 600mm
    }
    // Girando hacia la izquierda
    else if (IBus.readChannel(0) < 1450)
    {
        r = map(IBus.readChannel(0), 1450, 1000, 1400, 600); // radio de giro de 600mm a 1400mm
    }
    // Velocidad en % de 0 a 100
    s = map(IBus.readChannel(2), 1000, 2000, 0, 100); // rover speed from 0% to 100%
```

```
    calculateMotorsSpeed(s, s1, s2, s3);
    calculateServoAngle();

    //* RECIBIR STOP FRONT
    // Leer datos del sensor delantero
    // Serial.println("lectura señal stop front:");
    STOP_MOTOR_FRONT = digitalRead(STOP_SIGNAL_FRONT);
    // Serial.println(STOP_MOTOR_FRONT);

    if (IBus.readChannel(4) < 1600 && ((motor_forward || motor_Forward_left ||
motor_Forward_right) == true))
    {
        if (one_time_front)
        {
            STOP_MOTOR_FRONT = digitalRead(STOP_SIGNAL_FRONT);
            one_time_front = false;
        }
        while (STOP_MOTOR_FRONT == HIGH) // Cuando reciba 1 que comience el modo de ir
hacia atrás 2 seg y pare motores
        {
            IBus.loop();
            if (ledRojo_front)
            {
                int start_Time_front;
                int start_Time2_front;
                setColor(255, 0, 0); // Establece el color rojo
                if (!motor_signal_front)
                {
                    // motorBackward_sensor();
                    start_Time_front = millis(); // Guardamos el tiempo en milisegundos
                    start_Time2_front = start_Time_front;
                    // Serial.println("front");
                }
                // Serial.println("start_Time_front");
                // Serial.println(start_Time_front);

                while (start_Time2_front - start_Time_front < 1500)
                {
                    start_Time2_front = millis();
                    if (!motor_signal_front)
                    {
                        motorBackward();
                        // Serial.println("motor 5 segundos hacia atras");
                    }
                    // Serial.println("esperando");
                }
                motorStop();

                ledRojo_front = false;

                motor_signal_front = false;
            }

            if (IBus.readChannel(4) > 1700)
            {

                STOP_MOTOR_FRONT = LOW;
                setColor(0, 0, 0); // Apaga los LEDs
                ledRojo_front = true;
```

```cpp
                one_time_front = true;
                // Serial.println("desconexion sensores");
                break;
            }
        }
    }

    //* RECIBIR STOP BACK
    // Leer datos del sensor posterior
    // Serial.println("lectura señal stop back:");
    STOP_MOTOR_BACK = digitalRead(STOP_SIGNAL_BACK);
    Serial.println(STOP_MOTOR_BACK);
    if (IBus.readChannel(4) < 1600 && ((motor_backward || motor_Backward_left || motor_Backward_right) == true))
    {
        if (one_time_back)
        {
            STOP_MOTOR_BACK = digitalRead(STOP_SIGNAL_BACK);

            one_time_back = false;
        }
        while (STOP_MOTOR_BACK == HIGH) // Cuando reciba 1 que comience el  modo de ir hacia atrás 2 seg y pare motores
        {
            IBus.loop();
            if (ledRojo_back)
            {
                int start_Time_front;
                int start_Time2_front;
                setColor(255, 0, 0); // Establece el color rojo
                if (!motor_signal_back)
                {
                    // motorBackward_sensor();
                    start_Time_front = millis(); // Guardamos el tiempo en milisegundos
                    start_Time2_front = start_Time_front;
                    // Serial.println("back");
                }
                // Serial.println("start_Time_back");
                // Serial.println(start_Time_front);

                while (start_Time2_front - start_Time_front < 1500)
                {
                    start_Time2_front = millis();
                    if (!motor_signal_back)
                    {
                        motorForward();
                        // Serial.println("motor 5 segundos hacia atras");
                    }
                    // Serial.println("esperando");
                }
                motorStop();
            }

            ledRojo_back = false;

            motor_signal_back = false;

            if (IBus.readChannel(4) > 1700)
            {

```

```cpp
                STOP_MOTOR_BACK = LOW;
                setColor(0, 0, 0); // Apaga los LEDs
                ledRojo_back = true;
                one_time_back = true;
                // Serial.println("desconexion sensores");
                break;
            }
        }
    }

    //* Giro hacia la derecha

    if (IBus.readChannel(0) > 1550)
    {
        // Servomotores
        // Ruedas externas
        servoW1.startEaseTo(97 + thetaInnerFront); // front wheel steer right
        servoW3.startEaseTo(97 - thetaInnerBack);  // back wheel steer left for  overall steering to the right of the rover
        // Ruedas internas
        servoW4.startEaseTo(94 + thetaOuterFront);
        servoW6.startEaseTo(96 - thetaOuterBack);

        // Motores DC hacia delante con giro a la derecha
        if (IBus.readChannel(5) < 1400)
        {
            motorForward_right();
            motor_Forward_right = true;
            motor_Backward_right = false;
        }
        // Motores DC hacia atrás con giro a la derecha
        else if (IBus.readChannel(5) > 1600)
        {
            motorBackward_right();
            motor_Backward_right = true;
            motor_Forward_right = false;
        }
        // Parada de motores DC
        else if (IBus.readChannel(5) > 1400 || IBus.readChannel(5) < 1600) // &&: y ; || ; ó ; ==: igual
        {
            motorStop();
            motor_stop = true;
        }
    }

    //* Giro a la izquierda
    else if (IBus.readChannel(0) < 1450)
    {
        // Servomotores
        servoW1.startEaseTo(97 - thetaOuterFront);
        servoW3.startEaseTo(97 + thetaOuterBack);
        servoW4.startEaseTo(94 - thetaInnerFront);
        servoW6.startEaseTo(96 + thetaInnerBack);

        // Motores DC hacia adelante con giro a la izquierda
        if (IBus.readChannel(5) < 1400)
        {
            motorForward_left();
            motor_Forward_left = true;
```

```cpp
            motor_Backward_left = false;
        }
        // Motores DC hacia atrás con giro a la izquierda
        else if (IBus.readChannel(5) > 1600)
        {
            motorBackward_left();
            motor_Backward_left = true;
            motor_Forward_left = false;
        }
        // Parada de motores DC
        else if (IBus.readChannel(5) > 1400 || IBus.readChannel(5)  < 1600)
        {
            motorStop();
            motor_stop = true;
        }
    }
    //* Movimiento recto lineal
    else
    {
        servoW1.startEaseTo(97);
        servoW3.startEaseTo(97);
        servoW4.startEaseTo(94);
        servoW6.startEaseTo(96);

        // Motores DC hacia adelante
        if (IBus.readChannel(5) < 1400)
        {
            motorForward();
            motor_forward = true;
            motor_backward = false;
        }
        // Motores DC hacia atrás
        else if (IBus.readChannel(5) > 1600)
        {
            motorBackward();
            motor_backward = true;
            motor_forward = false;
        }
        // Parda de motores DC
        else if (IBus.readChannel(5) > 1400 || IBus.readChannel(5) < 1600)
        {
            motorStop();
            motor_stop = true;
        }
    }
    // //* IMU

    sensors_event_t accel;
    sensors_event_t gyro;
    sensors_event_t mag;
    sensors_event_t temp;
    icm.getEvent(&accel, &gyro, &temp, &mag);
    IMU();

    //* Monitor de de voltaje de la batería
    int sensorValue = analogRead(A0);
    float voltage = sensorValue * (6.00 / 1023.00) * 3.02; // COnvierte los  valores leídos de 6V para los 14.8 V
    //  Envío del valor de voltaje de batería al transmisor
    IBusSensor.loop();
```

```
    IBusSensor.setSensorMeasurement(1, voltage * 100);
}

```

src\main.cpp

```cpp
// Código de configuración de los sensores de ultrasonidos
// Subsistema modular de precolisión del proyecto rover Asturiosity
// Alba González Fernández

#include <Arduino.h>
#include <NewPing.h> //Librería para optimizar el funcionamiento de los sensores de ultrasonidos

//* Definición de variables
#define TRIGGER_PIN_1 8 // Pines de los sensores
#define ECHO_PIN_1 9
#define TRIGGER_PIN_2 5
#define ECHO_PIN_2 6
// Variables globales
bool flag1 = true;
bool flag2 = true;
bool flag_90_front = true;
bool flag_90_back = true;
bool flag = true;
bool flag_90 = true;

// Variables que definen los pines de envío de datos
#define STOP_SIGNAL_FRONT 3
#define STOP_SIGNAL_BACK 2

// Definición de los pines para la libreria
NewPing sonar1(TRIGGER_PIN_1, ECHO_PIN_1);
NewPing sonar2(TRIGGER_PIN_2, ECHO_PIN_2);

int distance_front;
int distance_back;

void setup()
{
    Serial.begin(115200);
    // Se definen los pines como salida
    pinMode(STOP_SIGNAL_FRONT, OUTPUT);
    pinMode(STOP_SIGNAL_BACK, OUTPUT);
}

void loop()
{
    distance_front = sonar1.ping_cm(); // Mide la distancia con el sensor 1 y guarda el valor en el arreglo
    distance_back = sonar2.ping_cm();  // Mide la distancia con el sensor 2 y guarda el valor en el arreglo
    sonar1.ping_cm();
    sonar2.ping_cm();

    // Imprimir lectura de distancias
    Serial.print("distancia delante: ");
    Serial.print(sonar1.ping_cm());
    Serial.print("cm");
    Serial.println();

    Serial.print("distancia atras: ");
    Serial.print(sonar2.ping_cm());
```

```
    Serial.print("cm");
    Serial.println();
    // delay(1000);

    bool flag1 = true;
    bool flag2 = true;
    bool flag_90_front = true;
    bool flag_90_back = true;
    flag = true;
    bool flag_90 = true;

    sonar1.ping_cm();
    sonar2.ping_cm();

    //* Función para enviar señal de stop si las distancias son menores a las distancias definidas para peligro de colisión en el sensor 1 delantero

    while ((sonar1.ping_cm() <= 90))
    {
        while (flag)
        {
            if (flag1)
            {
                digitalWrite(STOP_SIGNAL_FRONT, HIGH);
                int front = digitalRead(STOP_SIGNAL_FRONT);
                Serial.println(front);
                flag1 = false;
            }

            if ((sonar1.ping_cm() > 90))
            {
                digitalWrite(STOP_SIGNAL_FRONT, LOW);
                int front = digitalRead(STOP_SIGNAL_FRONT);
                Serial.println(front);

                flag = false;
            }
        }
    }
    flag = true;

    //* Función para enviar señal de stop si las distancias son menores a las distancias definidas para peligro de colisión en el sensor 2 trasero

    while ((sonar2.ping_cm() <= 110))
    {
        while (flag)
        {
            if (flag2)
            {
                digitalWrite(STOP_SIGNAL_BACK, HIGH);
                int back = digitalRead(STOP_SIGNAL_BACK);
                Serial.println(back);
                flag2 = false;
            }

            if ((sonar2.ping_cm() > 110))
            {
                digitalWrite(STOP_SIGNAL_BACK, LOW);
                int back = digitalRead(STOP_SIGNAL_BACK);
```

```
                Serial.println(back);

                flag = false;
            }
        }
    }
}

```