Nature Scanner

Hardware Design Description Document

v010
Prepared by:
Andrew Hredzak

Introduction

Purpose

The purpose of this NS DDD is to describe the design of the Nature Scanner device that incorporates a temp sensor, camera, gps and microphone into a durable portable waterproof form factor. This is a research prototype for and feasibility study for a device to create a digital twin of the world.

Scope

This document covers the different subsystems and provides linked documentation, relevant mathematical concepts. It also includes diagrams to give a high level overview of the entire system. It is important to understand what protocols were chosen and how the device is wired together. The DDD covers specific model numbers and versions to highlight the design of the device. There is also a BOM included to describe what is needed to build the NS and how much it costs.

Future Requirements

- What OS to use: free RTOS Contiki-NG ect..
- What chip to use esp32 esp32-s3, Atmel
- What sensors to use
- What communication protocols to use
- What Framework to use: arduino IDE, ESP IDF, bare metal programming, ASF (Advanced Software Framework)

Background information

Acronyms:

ARM: Advanced RISC Machine (Reduced Instruction Set Computing) – A family of computer processors based on a reduced instruction set architecture.

RISC: Reduced Instruction Set Computer – A type of microprocessor architecture that utilizes a small, highly optimized set of instructions.

SRAM: Static Random Access Memory – A type of memory that is faster and more reliable than

DRAM (Dynamic RAM), used for high-speed caches.

USB: Universal Serial Bus – A standard for communication between devices and a host controller (usually personal computers).

MAC: Media Access Control – A hardware identification number that uniquely identifies each device on a network.

CAN: Controller Area Network – A robust vehicle bus standard designed to allow microcontrollers and devices to communicate with each other in applications without a host computer.

MCI: MultiMedia Card Interface – A hardware interface used to connect flash memory cards (such as SD cards) to computers or other devices.

SDIO/SD/MMC: Secure Digital Input Output/Secure Digital/MultiMedia Card – Standards for memory cards used in portable devices.

NFC: NAND Flash Controller – Controls the interaction between the NAND flash memory (a type of non-volatile storage) and other components.

UART: Universal Asynchronous Receiver/Transmitter – A hardware communication protocol used for asynchronous serial communication.

TWI: Two-Wire Interface – Another name for the I²C protocol (Inter-Integrated Circuit), used for communication between microcontrollers and other peripherals.

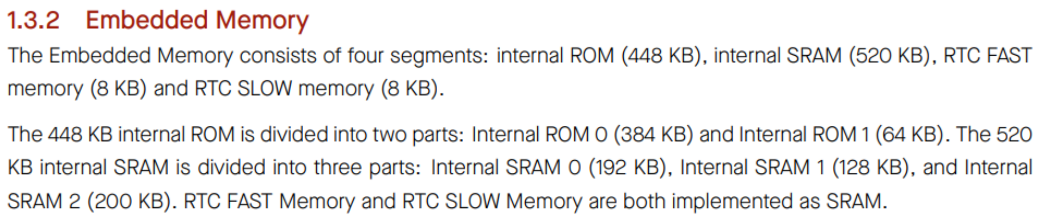
SPI: Serial Peripheral Interface – A synchronous serial communication interface used for short-distance communication, primarily in embedded systems.

HSPI: Hardware serial peripheral interface

VSPI: “very high speed???” serial peripheral interface

PWM: Pulse Width Modulation – A method used for controlling the amount of power delivered to a device by varying the width of the pulses in a pulse train.

RTC: Real-Time Clock – A clock that keeps track of the current time, even when the microcontroller is powered off.



1.3.2 Embedded Memory

The Embedded Memory consists of four segments: internal ROM (448 KB), internal SRAM (520 KB), RTC FAST memory (8 KB) and RTC SLOW memory (8 KB).

The 448 KB internal ROM is divided into two parts: Internal ROM 0 (384 KB) and Internal ROM 1 (64 KB). The 520 KB internal SRAM is divided into three parts: Internal SRAM 0 (192 KB), Internal SRAM 1 (128 KB), and Internal SRAM 2 (200 KB). RTC FAST Memory and RTC SLOW Memory are both implemented as SRAM.

*RTC context

RTT: Real-Time Timer – A timer used to track real-time events.

ADC: Analog-to-Digital Converter – A device that converts an analog signal (such as a voltage) into a digital number.

DAC: Digital-to-Analog Converter – A device that converts digital data (usually binary) into an analog signal (such as current or voltage).

PDC: Peripheral DMA Controller – Manages data transfer between peripherals and memory using direct memory access (DMA) channels.

DMA: Direct Memory Access – A feature that allows peripherals to communicate directly with system memory, bypassing the CPU, which improves performance.

AVR: Advanced Virtual RISC (Reduced Instruction Set Computer)

NL: new line

CR: carriage return

I2C: Inter-Integrated Circuit (I2C is a general-purpose interface for connecting microcontrollers to peripheral devices)

I2S: Inter-Integrated Circuit Sound

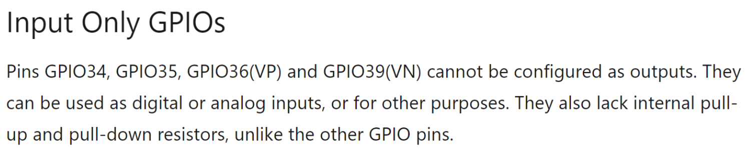
.h : header file(ex: freeRTOS.h)

.cpp: C++

NMEA: National Marine Electronics Association

I2S: I**nter-IC Sound**, sometimes also called Integrated Inter-IC Sound or IIS.
It is a specialized serial communication protocol designed for transmitting digital audio data between integrated circuits (ICs) within a device.

VP: voltage positive
VN: voltage negative



Input Only GPIOs

Pins GPIO34, GPIO35, GPIO36(VP) and GPIO39(VN) cannot be configured as outputs. They can be used as digital or analog inputs, or for other purposes. They also lack internal pull-up and pull-down resistors, unlike the other GPIO pins.

VP and VN example

Bluetooth LE SoC: bluetooth low energy system on chip

PSRAM: pseudo static random access memory

SHA: secure hash acronym
RSA: Rivest–Shamir–Adleman RSA is an asymmetric encryption algorithm widely used for secure data transmission.
AES: advanced encryption standard
RNG: random number generator
ULP: ultra low power

QFN: quad flat no leads. Flat no-leads packages such as quad-flat no-leads (QFN) and dual-flat no-leads (DFN) physically and electrically connect integrated circuits to printed circuit boards.

ESP: ESP stands for Espressif, which is the name of the company that designed the ESP family of chips. Espressif Systems is a semiconductor company that focuses on developing low-power, low-cost, and highly-integrated Wi-Fi and Bluetooth SoCs (System on Chips) and modules for wireless communication.

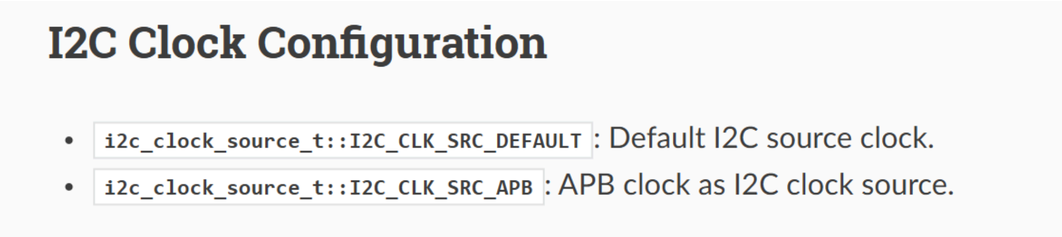
QSPI: quad serial peripheral interface

MMU: memory management unit
DMA: direct memory access

QEMU: Quick Emulator→ a free, open-source machine emulator and virtualizer that allows users to run operating systems and programs on different machines

.yml: YAML Ain't Markup Language (a recursive acronym, to distinguish its purpose as data-oriented, rather than document markup.)

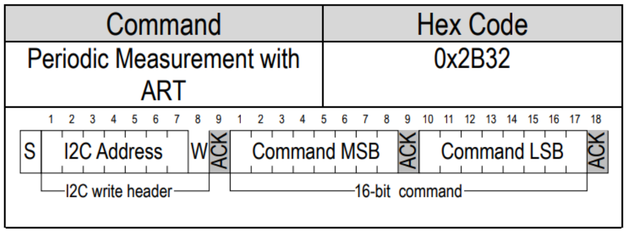
CS: chip select
APB: advanced peripheral clock



I2C Clock Configuration

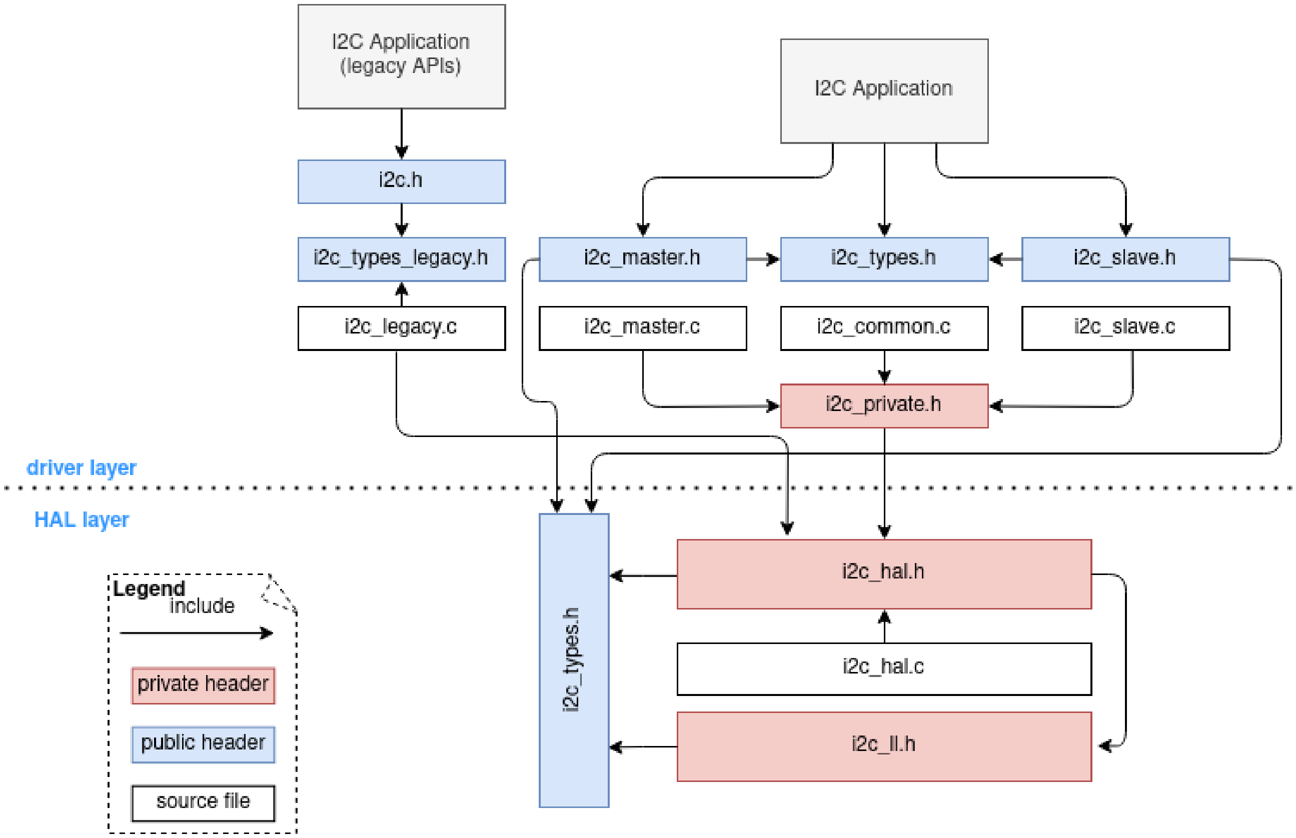
- `i2c_clock_source_t::I2C_CLK_SRC_DEFAULT` : Default I2C source clock.
- `i2c_clock_source_t::I2C_CLK_SRC_APB` : APB clock as I2C clock source.

ART: accelerated response time



Command	Hex Code
Periodic Measurement with ART	0x2B32

HAL: hardware abstraction layer



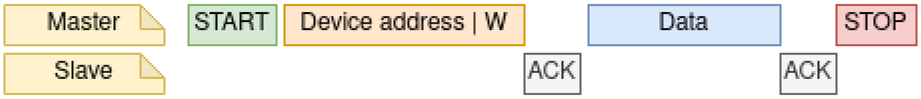
(REPL): Read-Eval-Print Loop
FHA: feed horn assembly
DDD: design description document

Communication Protocols

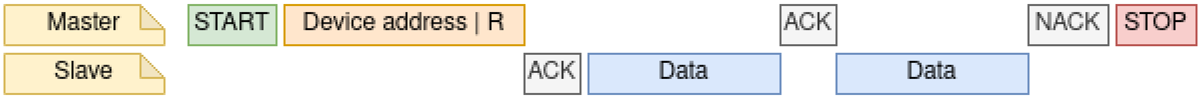
Protocol Category	Protocol Name	Key Features	Common Applications
Serial Communication	UART (Universal Asynchronous Receiver-Transmitter)	Simple, asynchronous, one bit at a time	Debugging, GPS modules, PC communication
	USART (Universal Synchronous/Asynchronous Receiver-Transmitter)	Supports synchronous and asynchronous modes	Higher data rates, more flexibility

Synchronous Serial	SPI (Serial Peripheral Interface)	High-speed, full-duplex, separate lines for TX/RX	Sensors, memory chips, displays
	I2C (Inter-Integrated Circuit)	Two-wire, multi-master, shared bus	Accelerometers, gyroscopes, real-time clocks
Other Important Protocols	1-Wire	Single-wire communication	Temperature sensors, unique identification
	CAN (Controller Area Network)	Robust, multi-master, for noisy environments	Automotive applications, industrial automation
	USB (Universal Serial Bus)	High-speed, widely used for peripherals	Programming, data logging, device communication
	Ethernet	Networking protocol for internet connectivity	Web servers, IoT devices, remote data acquisition
	Bluetooth	Wireless communication, short range	Wireless sensors, wearables, remote control
	Wi-Fi	Wireless communication, longer range	IoT devices, web servers, data logging
Specialized Protocols	MIDI (Musical Instrument Digital Interface)		Musical instruments and computers
	DMX512		Stage lighting and effects control
	Modbus		Industrial automation systems

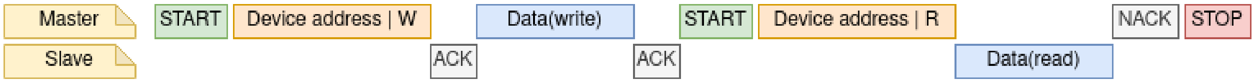
Inter-Integrated Circuit (I2C)



I2C Master Read



I2C Master Read



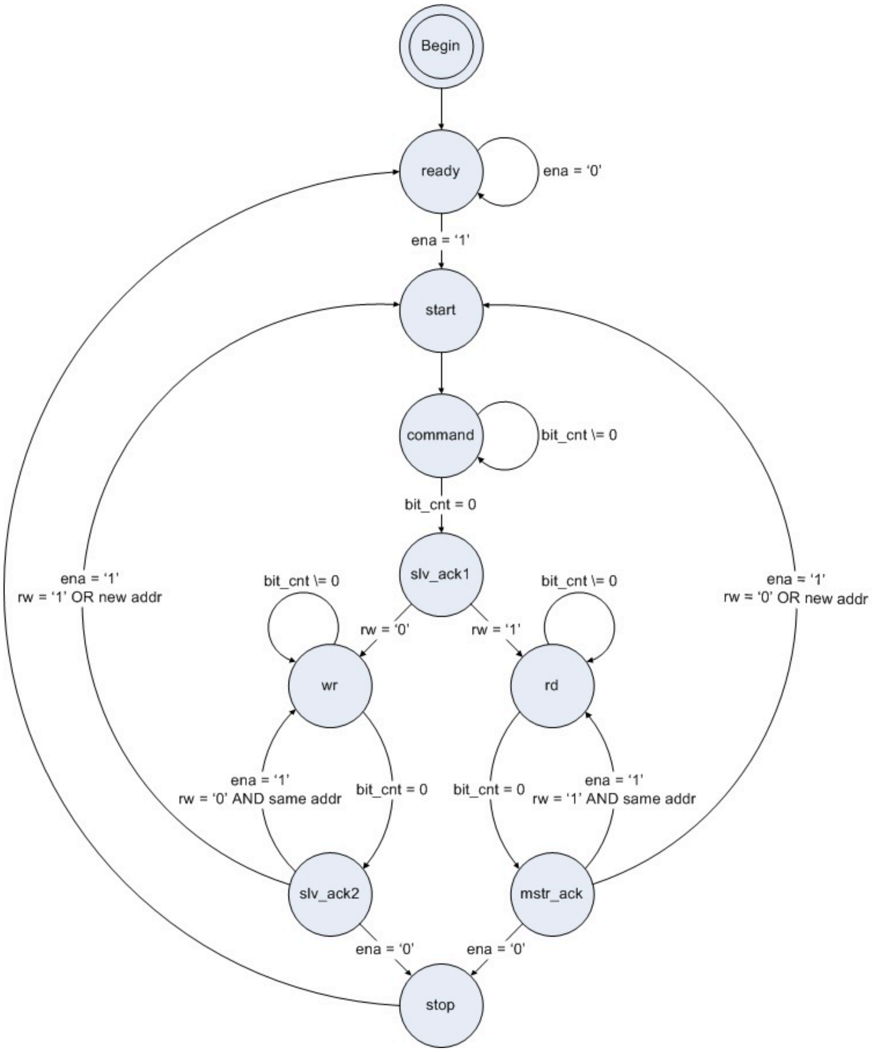
I2C Master Write and Read

```
I2c_master_transmit_receive
I2c_master_receive
i2c_master_transmit
```

START condition:

S (Start Condition)
I2C write header (7-bit I2C device address plus 0 as the write bit) and a 16-bit measurement command.
-It pulls the SDA pin low (ACK bit) after the falling edge of the 8th SCL clock to indicate the reception

W(Write Condition): 0 as the write bit, 1 as not write bit



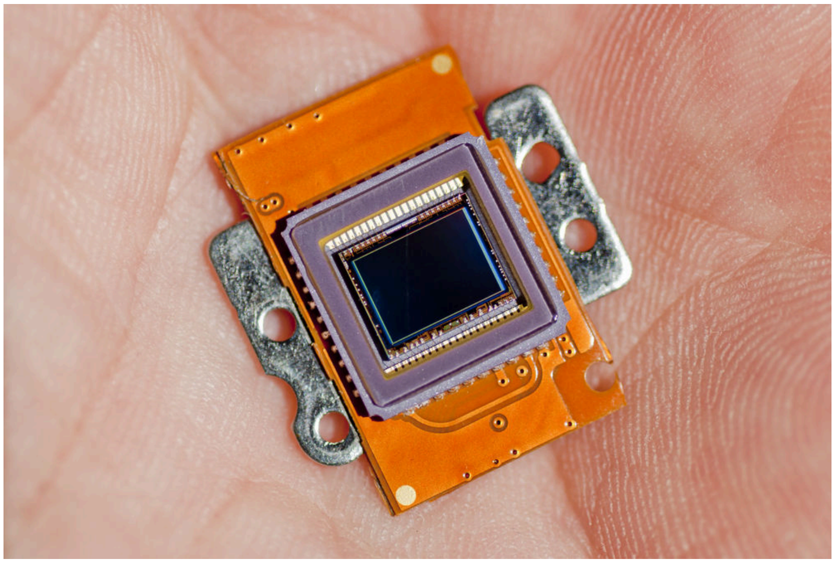
I2C Master State Machine

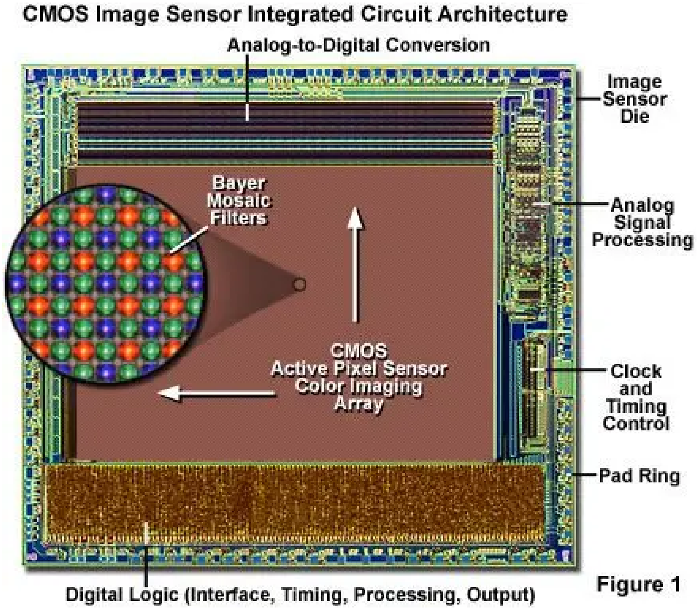
Common ESP IDF commands

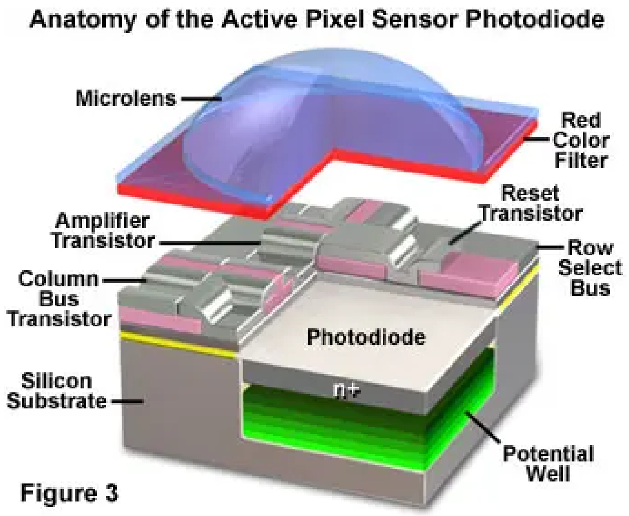
Command	Description	Example
Write Flash	Write a binary file (firmware) to flash memory.	python -m esptool --chip esp32 --port COM3 --baud 460800 write_flash 0x1000 firmware.bin
Erase Flash	Erase the entire flash memory on the ESP32.	python -m esptool --chip esp32 --port COM3 erase_flash
Read Flash	Read the contents of flash memory	python -m esptool --chip esp32 --port COM3 read_flash

	from the ESP32 and save it to a file.	0x1000 4096 firmware_dump.bin
Chip ID	Retrieve the unique chip ID from the ESP32.	python -m esptool --chip esp32 --port COM3 chip_id
Read MAC Address	Retrieve the ESP32's MAC address from its OTP (One-Time Programmable) memory.	python -m esptool --chip esp32 --port COM3 read_mac
Flash ID	Retrieve the manufacturer and device ID of the connected SPI flash.	python -m esptool --chip esp32 --port COM3 flash_id
Verify Flash	Verify that a binary file written to flash matches the file on your disk.	python -m esptool --chip esp32 --port COM3 verify_flash 0x1000 firmware.bin
Dump Memory	Dump the contents of an arbitrary memory location to disk.	python -m esptool --chip esp32 --port COM3 dump_mem 0x40000000 64
Run Program	Run the application code in flash memory after it has been uploaded.	python -m esptool --chip esp32 --port COM3 run
Image Info	Print detailed information about an application binary image (bootloader or application).	python -m esptool --chip esp32 --port COM3 image_info firmware.bin
Read Flash Status	Read the status register of the SPI flash (useful for debugging flash operations).	python -m esptool --chip esp32 --port COM3 read_flash_status
Make Image	Create an application image from binary files for flashing to ESP32.	python -m esptool --chip esp32 make_image -o output_image.bin
elf2image	Convert an ELF (Executable and Linkable Format) file into a flashable image.	python -m esptool --chip esp32 elf2image my_firmware.elf

CMOS Image Sensor

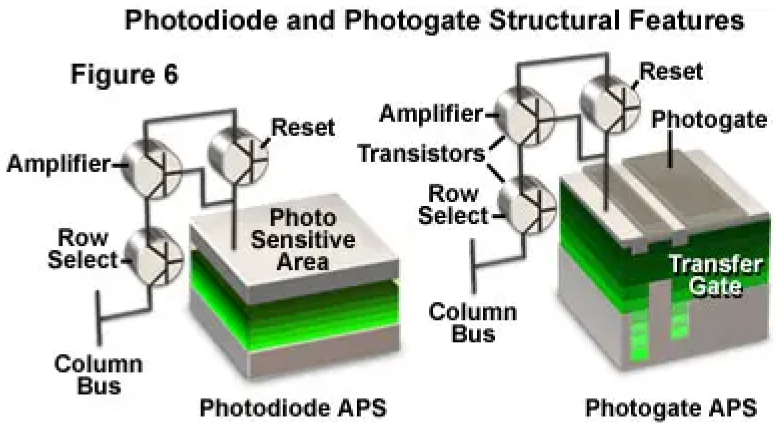



CMOS Image Sensor Integrated Circuit Architecture
Analog-to-Digital Conversion
Image Sensor Die
Bayer Mosaic Filters
Analog Signal Processing
CMOS Active Pixel Sensor Color Imaging Array
Clock and Timing Control
Pad Ring
Digital Logic (Interface, Timing, Processing, Output)
Figure 1



Anatomy of the Active Pixel Sensor Photodiode

Figure 3



Photodiode and Photogate Structural Features

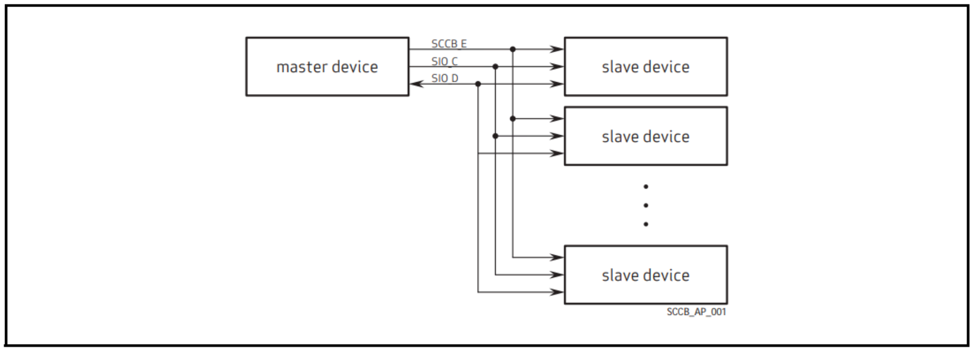
Figure 6

Serial Command Control Bus SCCB

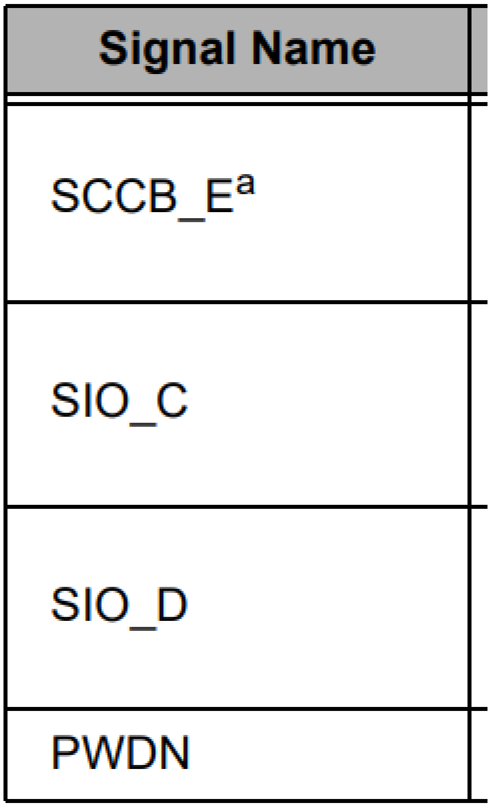
Serial Command Control Bus SCCB functional document

SCCB is a Omni Vision proprietary protocol

Figure 1-1 SCCB Functional Block Diagram



4 signal types:



Signal Name
SCCB_E[a]
SIO_C
SIO_D
PWDN

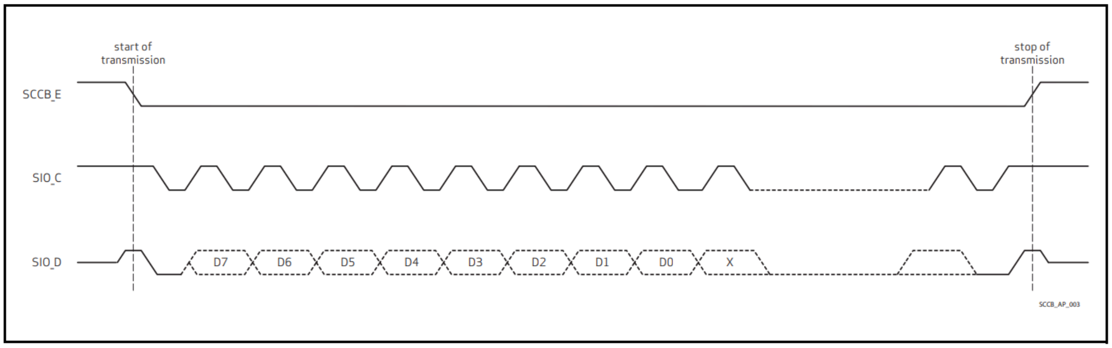
SCCB_E serial camera control bus enable. is the control enable signal. It can only be driven by the master. It is active LOW. logic of 1 indicates the bus is at IDLE

SIO_C serial input output control. single directional active HIGH control signal

SIO_D serial input output data is a bidirectional data signal

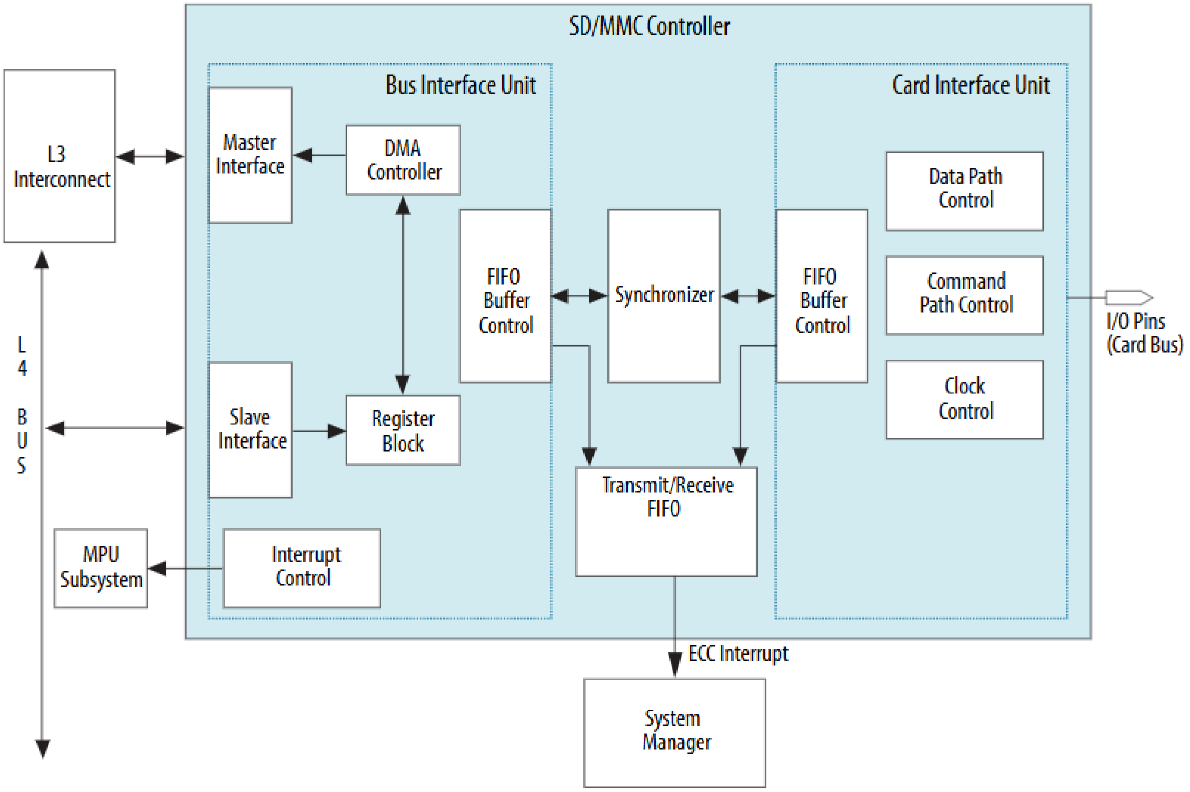
PWDN power down input or output

Figure 3-1 3-WIre Data Transmission Timing Diagram

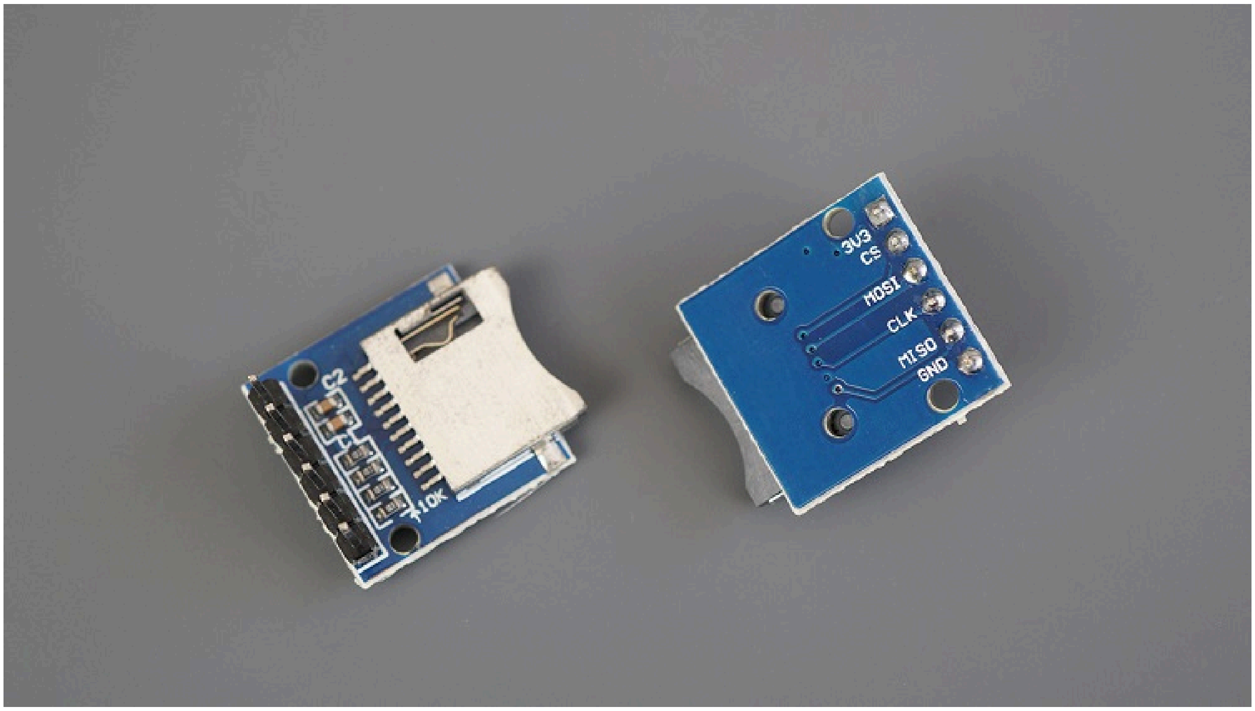


SD/MMC Controller

Figure 49. SD/MMC Controller Connectivity

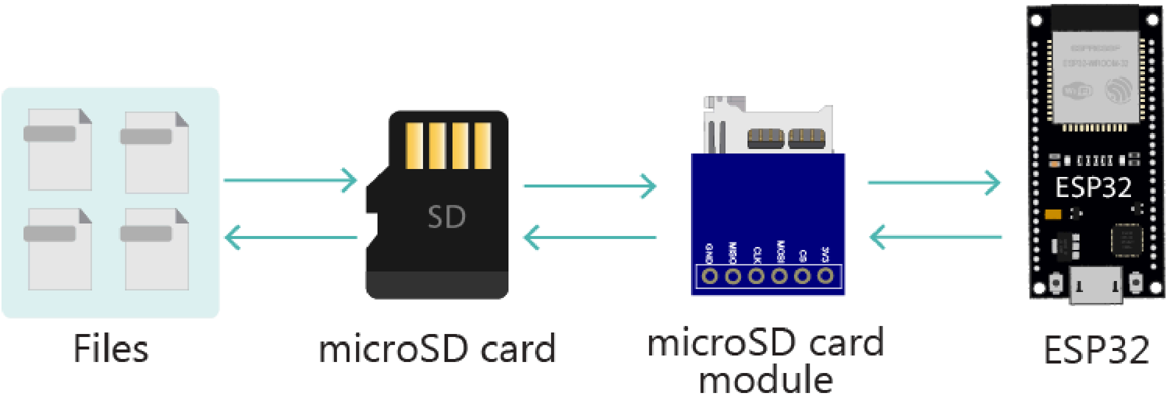


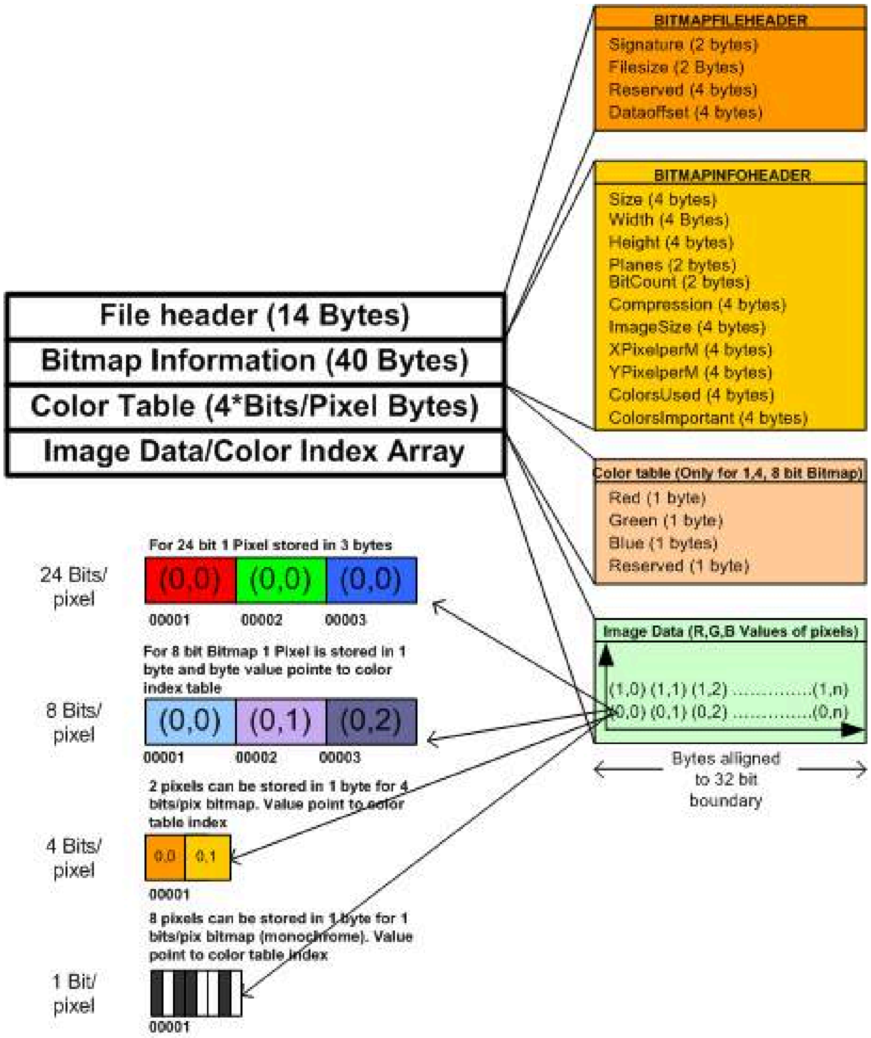
SD/MMC controller



MicroSD Card Module

communicates using SPI communication protocol

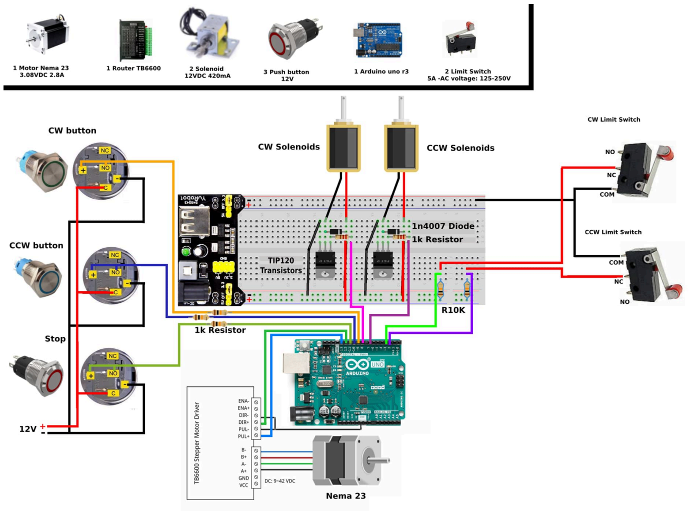


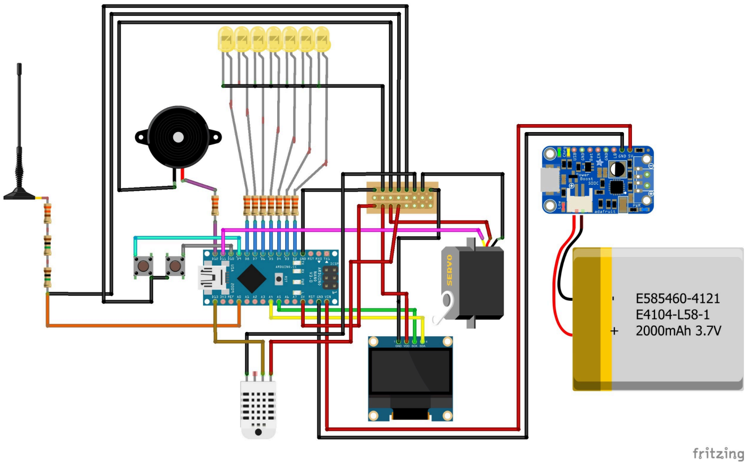

File header (14 Bytes)
Bitmap Information (40 Bytes)
Color Table (4*Bits/Pixel Bytes)
Image Data/Color Index Array
BITMAPFILEHEADER
Signature (2 bytes)
Filesize (2 Bytes)
Reserved (4 bytes)
Dataoffset (4 bytes)
BITMAPINFOHEADER
Size (4 bytes)
Width (4 Bytes)
Height (4 bytes)
Planes (2 bytes)
BitCount (2 bytes)
Compression (4 bytes)
ImageSize (4 bytes)
XPixelperM (4 bytes)
YPixelperM (4 bytes)
ColorsUsed (4 bytes)
ColorsImportant (4 bytes)
Color table (Only for 1,4, 8 bit Bitmap)
Red (1 byte)
Green (1 byte)
Blue (1 bytes)
Reserved (1 byte)
Image Data (R,G,B Values of pixels)
(1,0) (1,1) (1,2) ....(1,n)
(0,0) (0,1) (0,2) ....(0,n)
Bytes alligned to 32 bit boundary
24 Bits/ pixel
For 24 bit 1 Pixel stored in 3 bytes
(0,0)
(0,0)
(0,0)
00001
00002
00003
8 Bits/ pixel
For 8 bit Bitmap 1 Pixel is stored in 1 byte and byte value pointe to color index table
(0,0)
(0,1)
(0,2)
00001
00002
00003
4 Bits/ pixel
2 pixels can be stored in 1 byte for 4 bits/pix bitmap. Value point to color table index
0,0
0,1
00001
1 Bit/ pixel
8 pixels can be stored in 1 byte for 1 bits/pix bitmap (monochrome). Value point to color table index
00001

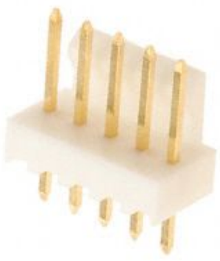
Bitmap BMP architecture

fb: frame buffer
BMP: Bitmap. It's a raster graphics image file format used to store bitmap digital images
DIB: Device Independent Bitmap
VFS: Virtual File System A layer that abstracts file system operations, enabling support for file systems like FAT.
FAT: File Allocation Table A file system format used for storage devices like SD cards.
SDMMC: Secure Digital MultiMedia Card A protocol and controller for interfacing with SD cards or MultiMedia Cards.
SDSPI: Secure Digital Serial Peripheral Interface A method to interface with SD cards using the SPI protocol.
SDIO: secure digital input output
NVS: non volatile storage
POSIX stands for Portable Operating System Interface

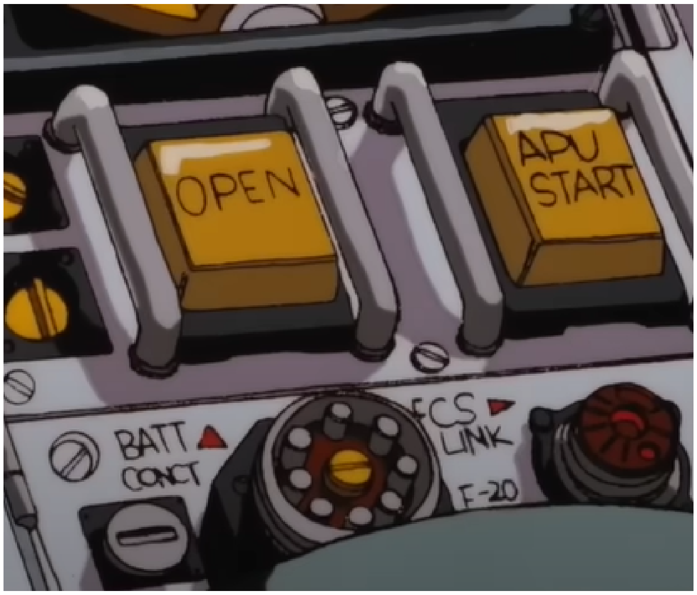
Wiring Diagram examples





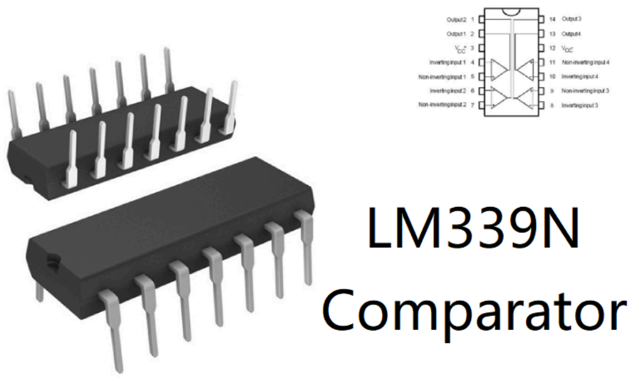


Molex connector

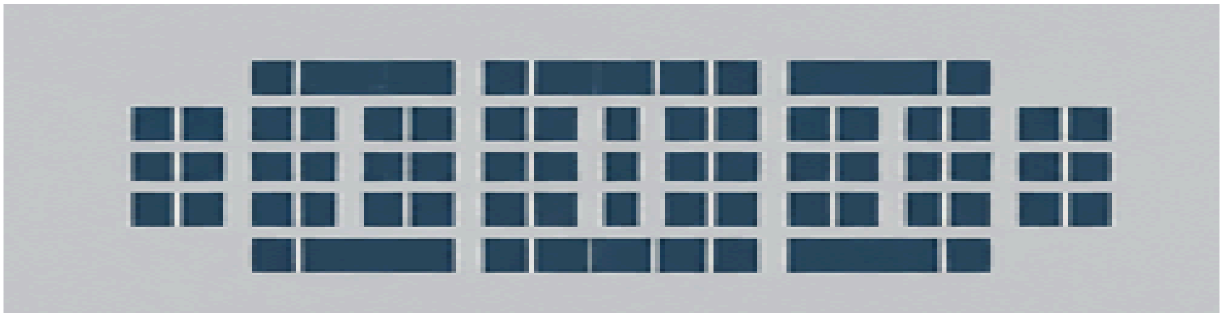


Practice

comparator
TI
lm339n
[Data sheet]



BOM

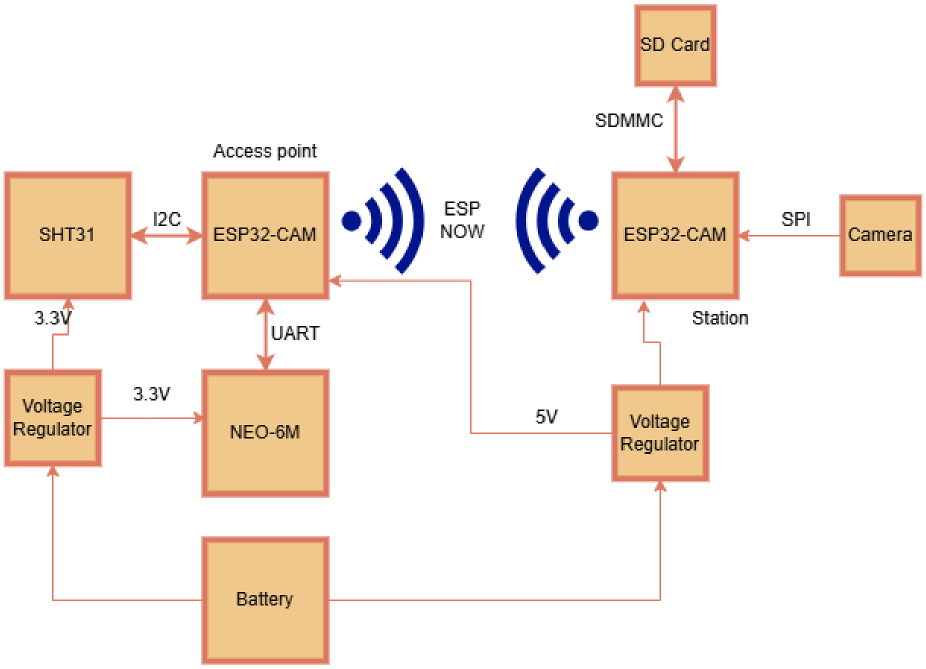


BOM

Category	Component	Description	Approximate Cost (USD)	Purchase
microcontroller	ESP32-CAM dev board x2	dual-core 32-bit LX6 CPU, 2MP OV2640 cam module, microSD slot, wifi, bluetooth	$14.00	Purchased
sensor	SHT31-DIS-B Sensor	High-precision humidity and temperature sensor, I2C interface	$12.88	Purchased
sensor	Ublox NEO-M6 GPS Module	GPS module with 1-2 meter accuracy, EEPROM supports multiple satellite systems	$9.00	Purchased
Encloser	ABS Plastic IP65 Enclosure	Waterproof, dustproof project enclosure (5.9 x 3.9 x 2.8 inch)	$13.99	Purchased
serial interface	HiLetgo Serial adapter	FT232RL Mini USB to TTL Serial Converter Adapter Module	$6.49	Purchased
memory	SD card	SanDisk 32GB 2-Pack Ultra MicroSDHC UHS-I Memory Card (2x32GB)	$13.56	Purchased
PCB	PCB	Prototype PCB Solderable Breadboard(5 Pack + 1 Mini Board, Red)	$8.49	Purchased
power	voltage regulator	AMS1117-3.3V Buck Module LDO 800MA	$0.60	Purchased
power	voltage regulator	5v Regulator Module Mini Voltage	$1.10	Purchased

		Reducer DC 4.5-24V 12V 24V to 5V 3A		
power	battery	7.4V Lipo Battery 600mAh 2S 30C Rechargeable Lithium Polymer Batteries	$17.00	Purchased
Total:			**$83.11**	

System Architecture



System architecture block diagram

ESP32 SoC Microcontroller

References:

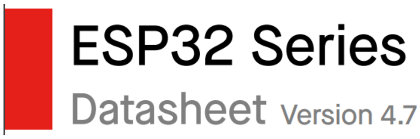
ESP Hardware Design Guidelines

ESPRESSIF

 ESP-IDF Programming Guide

ESPRESSIF

ESPRESSIF SDK + DEMOS


ESP32 Series

Datasheet Version 4.7

ESP32

Technical Reference Manual Version 5.2

ESP-EDF programming

idf.py location on my machine:
C:\Espressif\frameworks\esp-idf-v5.3.1\tools

SPI Controller (SPI)

from *esp32_technical_reference_manual_en.pdf*

127 / 744

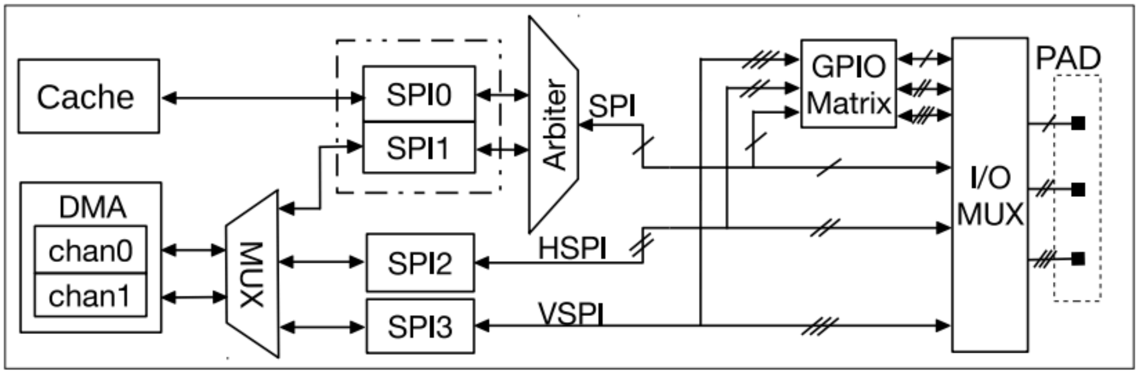


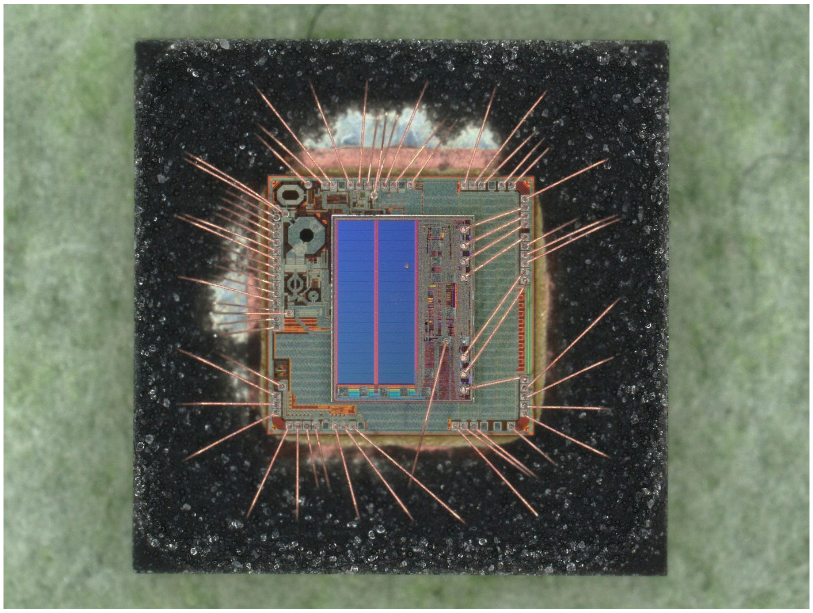
Figure 7-1. SPI Architecture

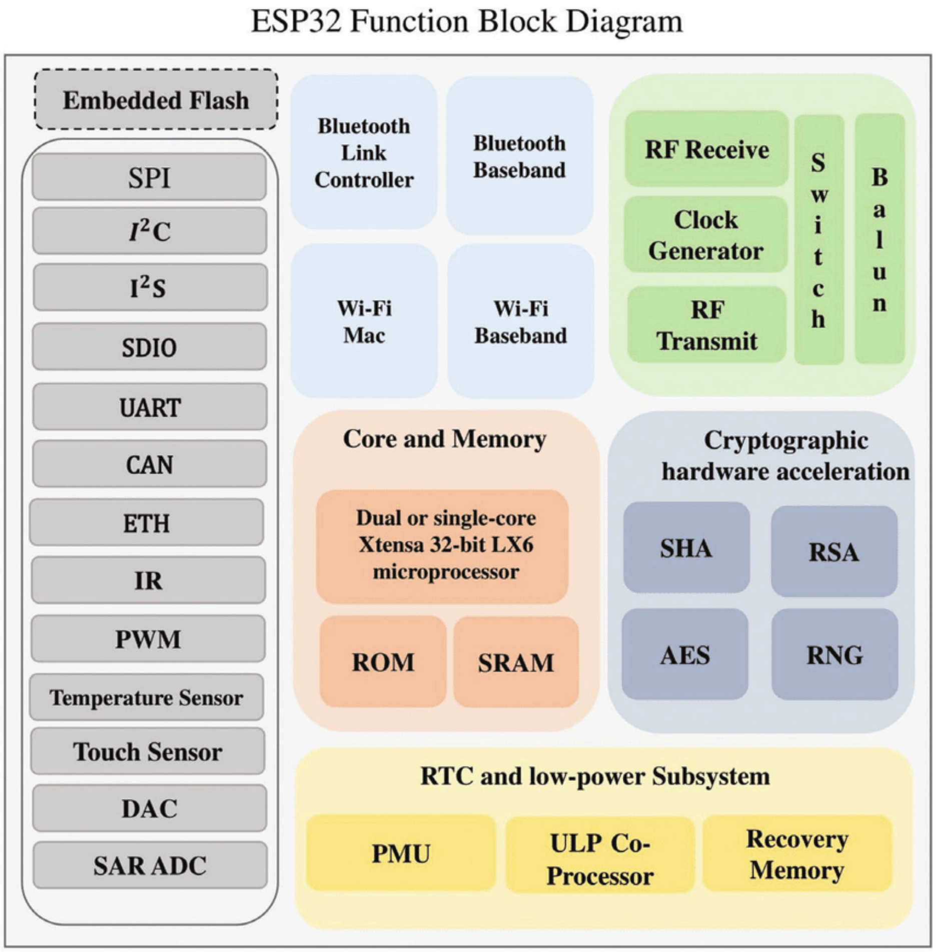
General Purpose Input/Output Interface (GPIO)

ESP32 has 34 GPIO pins which can be assigned various functions by programming the appropriate registers.

There are several kinds of GPIOs:
digital-only, analog-enabled, capacitive-touch-enabled, etc. Analog-enabled
GPIOs and Capacitive-touch-enabled GPIOs can be configured as digital GPIOs

ESP32 chip





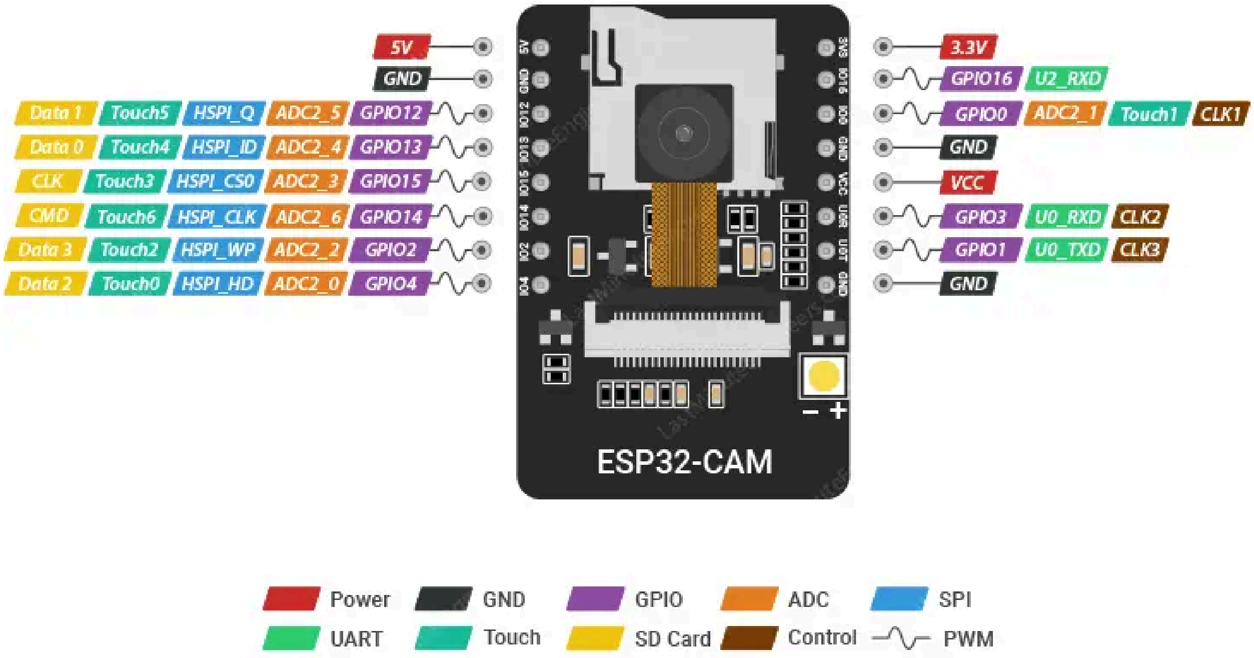
ESP32 Functional Block Diagram

ESP32S Series: These chips generally focus on **security**. They often include features like:

From ESP32-CAM module

- Secure boot
- Flash encryption
- Cryptographic acceleration (for faster encryption/decryption)
- Examples: ESP32-S2, ESP32-S3

Pinout


5V
GND
Data 1 Touch5 HSPI_Q ADC2_5 GPIO12
Data 0 Touch4 HSPI_ID ADC2_4 GPIO13
CLK Touch3 HSPI_CS0 ADC2_3 GPIO15
CMD Touch6 HSPI_CLK ADC2_6 GPIO14
Data 3 Touch2 HSPI_WP ADC2_2 GPIO2
Data 2 Touch0 HSPI_HD ADC2_0 GPIO4
5V GND IO12 IO13 IO15 IO14 IO2 IO4
3V3 IO16 IO0 GND VCC U0R U0T GND
3.3V
GPIO16 U2_RXD
GPIO0 ADC2_1 Touch1 CLK1
GND
VCC
GPIO3 U0_RXD CLK2
GPIO1 U0_TXD CLK3
GND
− +
ESP32-CAM
Power
GND
GPIO
ADC
SPI
UART
Touch
SD Card
Control
PWM

ESP32-CAM pinout

ESP-32S datasheet

ESP-32S Datasheet

ESP32- CAM- SN001

Specs

ESP-32S

*from flashing output:
Chip is ESP32-D0WDQ6-V3 (revision v3.1)
Features: WiFi, BT, Dual Core, 240MHz, VRef calibration in efuse, Coding Scheme None
Crystal is 40MHz
MAC: a0:dd:6c:77:bd:84

Vender: Hosyond

ESP32- CAM- SN002

……….

ESP32-C Series: This line emphasizes **low-cost and connectivity**. They tend to have:

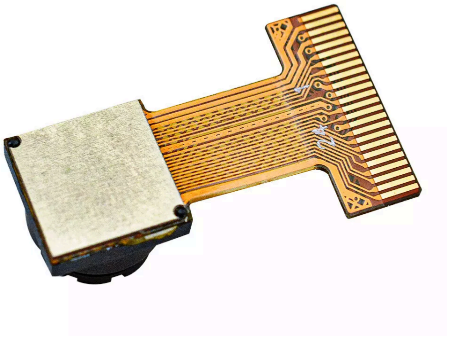
- Lower power consumption
- Wi-Fi and sometimes Bluetooth capabilities
- Optimized for cost-sensitive IoT applications
- Example: ESP32-C3

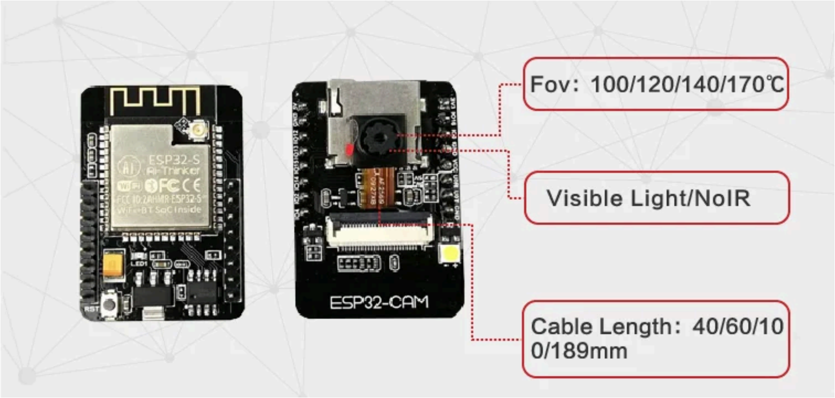
ESP32-H Series: These chips are designed for low-power applications that use the **IEEE 802.15.4** standard. This standard is commonly used for things like:

- Industrial wireless sensor networks
- Smart home devices
- Mesh networks (where devices connect to each other directly)

ESP32 Camera

OV02640-VL9A
OV2640 Color CMOS UXGA (2.0 Megapixel)





SPECS:
Resolution: 2 Megapixels (1600 x 1200 UXGA).
Optical Format: 1/4 inch.
Pixel Size: 2.2 µm x 2.2 µm.
Shutter Type: Electronic Rolling Shutter.
Output Formats: YUV(422/420), YCbCr422, RGB565/555, 8-bit compressed data, 8/10-bit Raw RGB data.

Performance:
Maximum Frame Rate:
UXGA/SXGA: 15 fps.
SVGA: 30 fps.
CIF: 60 fps.
Sensitivity: 0.6 V/Lux-sec.
S/N Ratio: 40 dB.
Dynamic Range: 50 dB.

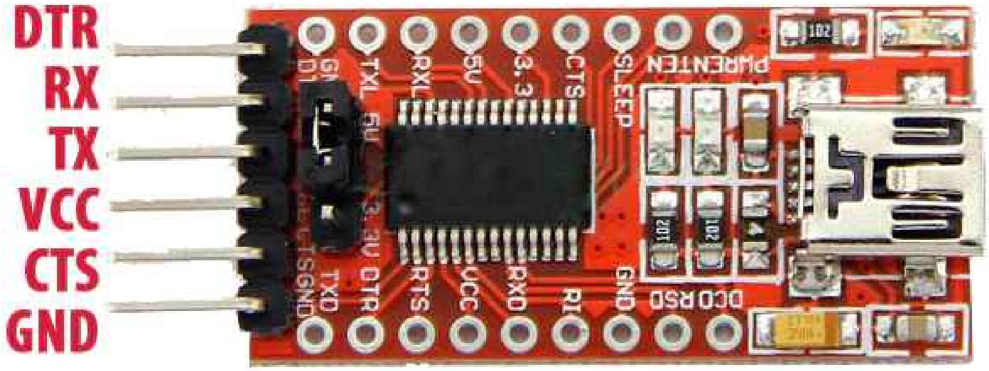
Power:
Core: 1.3V DC ± 5%.
Analog: 2.5~3.0V DC.
I/O: 1.7V to 3.3V.

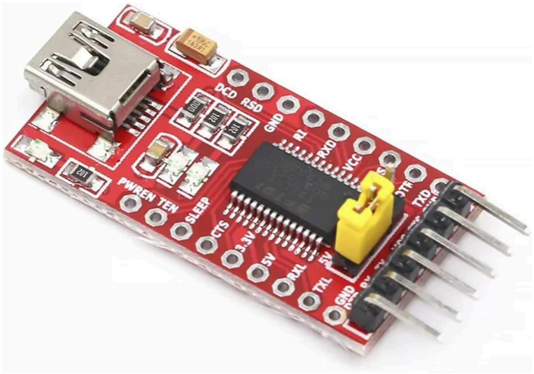
Power Consumption:
YUV mode full res & framerate: 125mW.
Compressed mode full Res & framerate: 140mW.
Active current consumption: 41.66 mA
Standby: 600µA.

Getting Started With ESP32-CAM

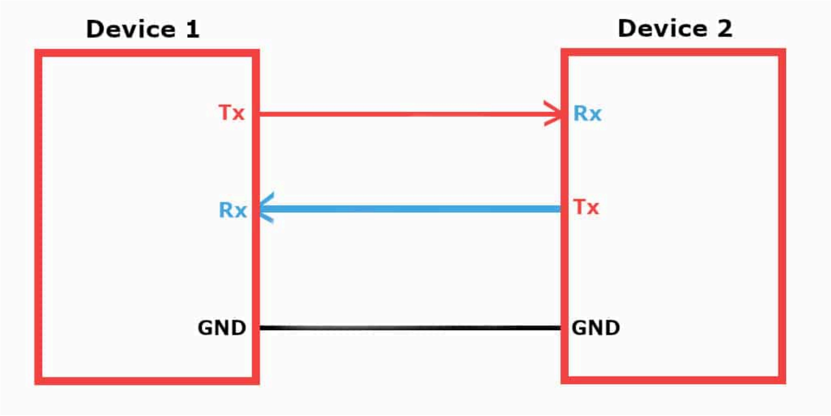
Since many users have reported problems when powering the device with 3.3V, it is advised that the ESP32-CAM always be powered via the **5V pin**

HiLetgo FT232R USB UART IC Serial adapter



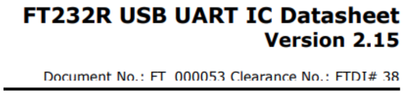


Acronym	Full Name	Description
DTR	Data Terminal Ready	Indicates the connected device is ready to receive data.
RX	Receive Data	Pin for receiving data.
TX	Transmit Data	Pin for transmitting data.
VCC	Voltage Common Collector	Power supply pin (usually 3.3V or 5V).
CTS	Clear To Send	Indicates the connected device is ready to receive data (flow control).
GND	Ground	Ground connection.



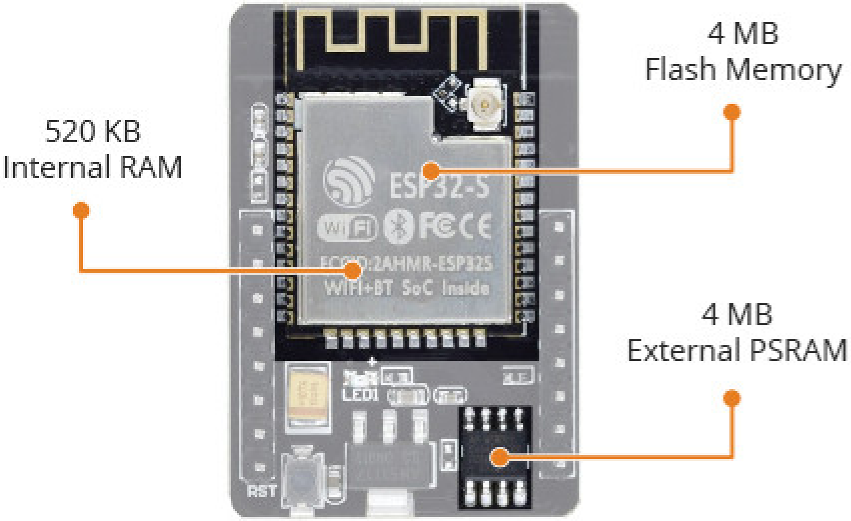
resources

Website link

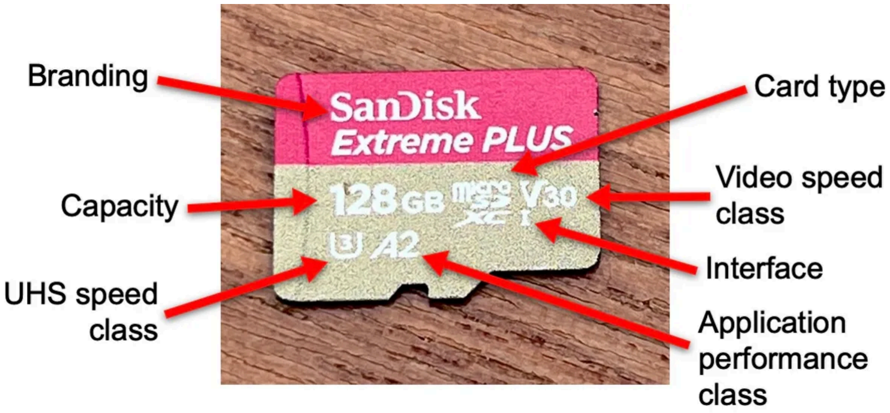

FT232R USB UART IC Datasheet
Version 2.15

Document No.: FT 000053 Clearance No.: FTDI# 38

Memory



Memory modules

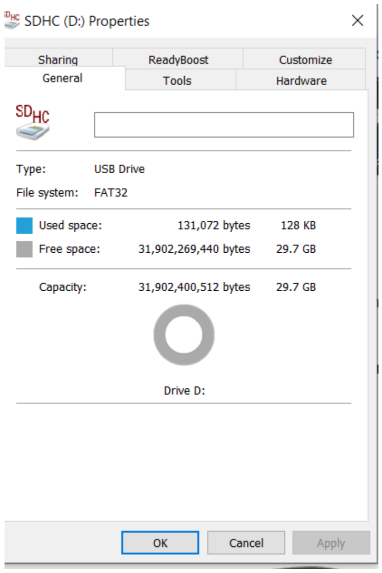


Micro SD card

SD: secure digital

microSD card for data storage. Specifically, it supports microSD cards formatted in **FAT16** or **FAT32**

"TF" stands for **TransFlash**, which is the original name for **microSD** cards.



SDHC (D:) Properties

Type: USB Drive

File system: FAT32

Used space:	131,072 bytes	128 KB
Free space:	31,902,269,440 bytes	29.7 GB
Capacity:	31,902,400,512 bytes	29.7 GB

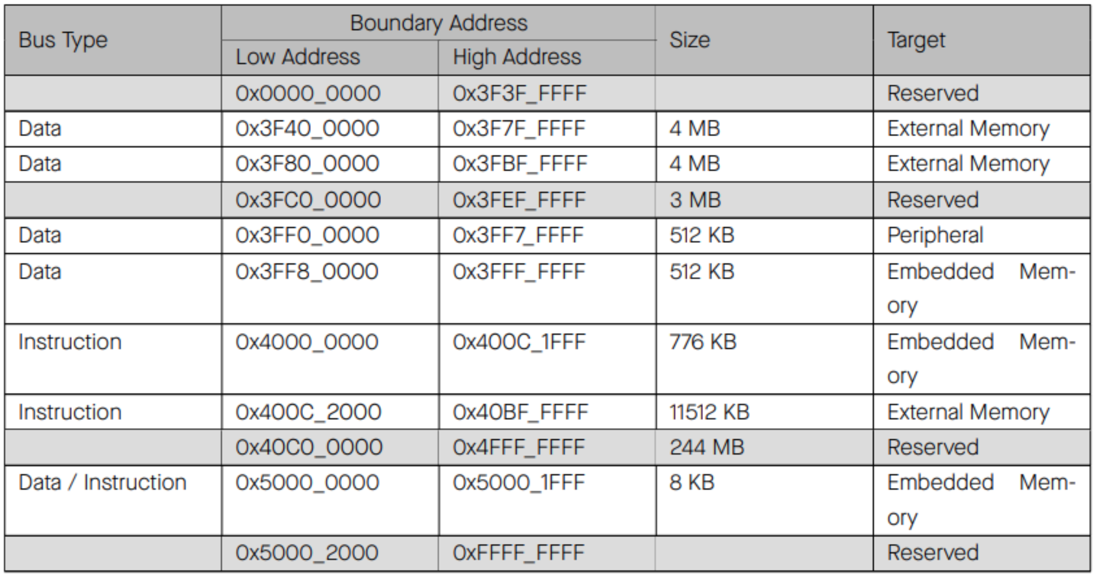
Drive D:

FAT32

ESP32 Address Mapping

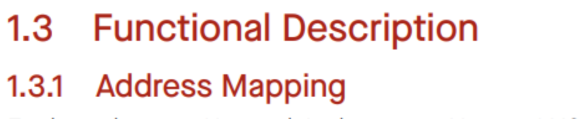
Each of the two Harvard Architecture Xtensa LX6 CPUs has 4 GB (32-bit) address space. Address spaces are symmetric between the two CPUs.

Table 1-1. Address Mapping



Bus Type	Boundary Address		Size	Target
	Low Address	High Address		
	0x0000_0000	0x3F3F_FFFF		Reserved
Data	0x3F40_0000	0x3F7F_FFFF	4 MB	External Memory
Data	0x3F80_0000	0x3FBF_FFFF	4 MB	External Memory
	0x3FC0_0000	0x3FEF_FFFF	3 MB	Reserved
Data	0x3FF0_0000	0x3FF7_FFFF	512 KB	Peripheral
Data	0x3FF8_0000	0x3FFF_FFFF	512 KB	Embedded Memory
Instruction	0x4000_0000	0x400C_1FFF	776 KB	Embedded Memory
Instruction	0x400C_2000	0x40BF_FFFF	11512 KB	External Memory
	0x40C0_0000	0x4FFF_FFFF	244 MB	Reserved
Data / Instruction	0x5000_0000	0x5000_1FFF	8 KB	Embedded Memory
	0x5000_2000	0xFFFF_FFFF		Reserved

- External Memory
- Peripheral
- Embedded Memory
- External Memory



1.3 Functional Description

1.3.1 Address Mapping

esp32_technical_reference_manual_en.pdf

I2C space on ESP32

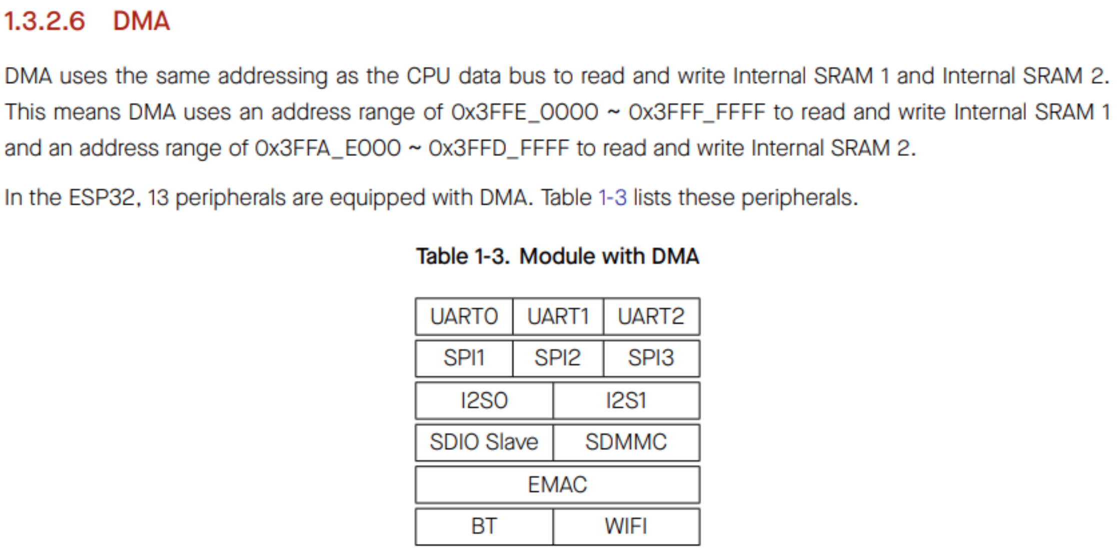
Addressing in Bytes:

- Each memory address represents 1 byte (8 bits) of data. So, address `0x3FF6_7001` holds a single byte of data.
- For example, if the value `0x12` is stored at `0x3FF6_7001`, it only occupies one byte (8 bits).

32-bit Access:

- The CPU can access 32-bit (4-byte) data by reading from a group of four consecutive addresses.
- For instance, if you want to read a 32-bit word starting at address `0x3FF6_7000`, the CPU will read bytes from `0x3FF6_7000`, `0x3FF6_7001`, `0x3FF6_7002`, and `0x3FF6_7003`.
- These four bytes together make up a 32-bit word, such as `0x12345678`, with each byte representing part of the word:
  - `0x3FF6_7000` stores `0x78`
  - `0x3FF6_7001` stores `0x56`
  - `0x3FF6_7002` stores `0x34`
  - `0x3FF6_7003` stores `0x12`

```
i2c-tools> i2cdump -c 0x44 -s 4
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f    0123456789abcdef
00: 00 00 81 ff 00 00 81 ff 00 00 81 ff 00 00 81 ff    ..?...?...?...?.
10: 00 00 81 ff 00 00 81 ff 00 00 81 ff 00 00 81 ff    ..?...?...?...?.
20: 00 00 81 ff 00 00 81 ff 00 00 81 ff 00 00 81 ff    ..?...?...?...?.
30: 00 00 81 ff 00 00 81 ff 00 00 81 ff 00 00 81 ff    ..?...?...?...?.
40: 00 00 81 ff 00 00 81 ff 00 00 81 ff 00 00 81 ff    ..?...?...?...?.
50: 00 00 81 ff 00 00 81 ff 00 00 81 ff 00 00 81 ff    ..?...?...?...?.
60: 00 00 81 ff 00 00 81 ff 00 00 81 ff 00 00 81 ff    ..?...?...?...?.
70: 00 00 81 ff 00 00 81 ff 00 00 81 ff 00 00 81 ff    ..?...?...?...?.
```



1.3.2.6 DMA

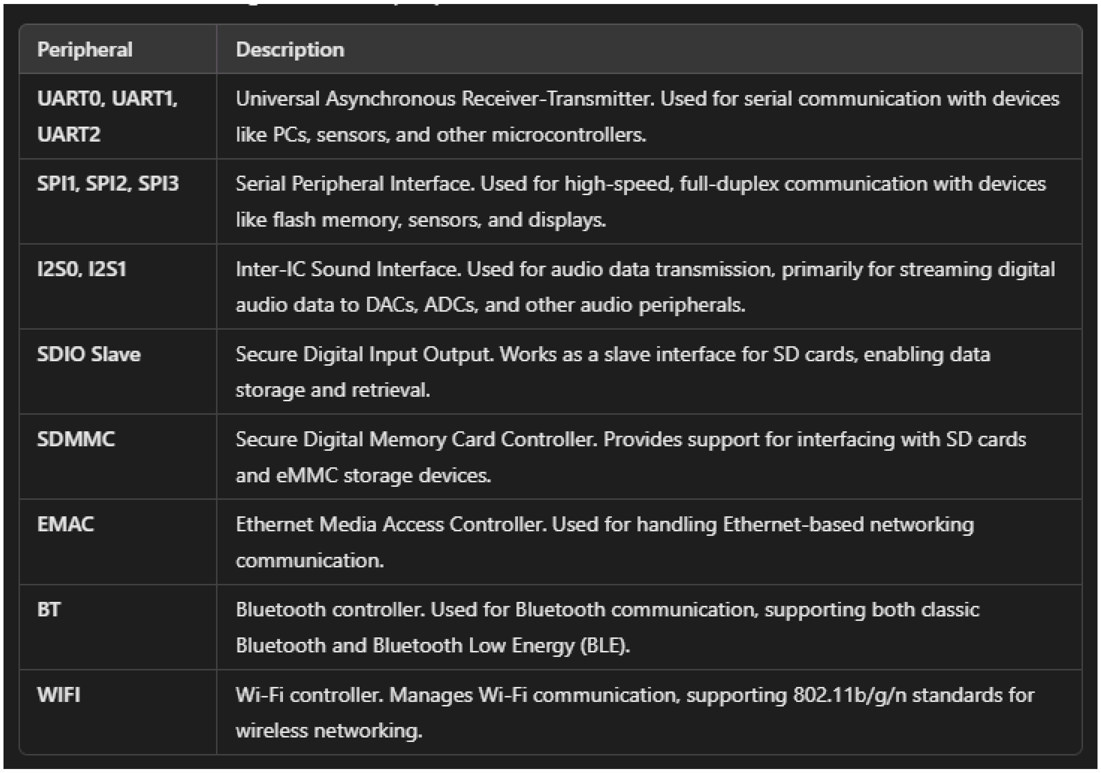
DMA uses the same addressing as the CPU data bus to read and write Internal SRAM 1 and Internal SRAM 2. This means DMA uses an address range of 0x3FFE_0000 ~ 0x3FFF_FFFF to read and write Internal SRAM 1 and an address range of 0x3FFA_E000 ~ 0x3FFD_FFFF to read and write Internal SRAM 2.

In the ESP32, 13 peripherals are equipped with DMA. Table 1-3 lists these peripherals.

Table 1-3. Module with DMA

UART0	UART1	UART2
SPI1	SPI2	SPI3
I2S0	I2S1	
SDIO Slave	SDMMC	
EMAC		
BT	WIFI	

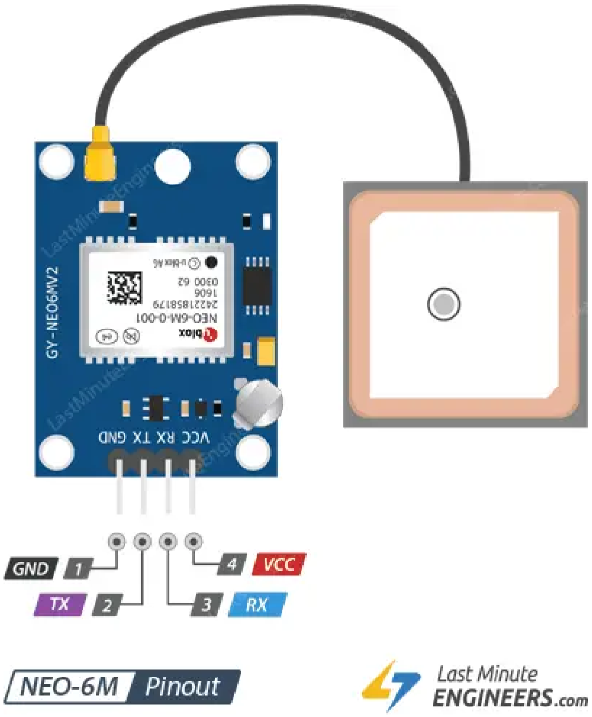
Direct Memory Access



Peripheral	Description
UART0, UART1, UART2	Universal Asynchronous Receiver-Transmitter. Used for serial communication with devices like PCs, sensors, and other microcontrollers.
SPI1, SPI2, SPI3	Serial Peripheral Interface. Used for high-speed, full-duplex communication with devices like flash memory, sensors, and displays.
I2S0, I2S1	Inter-IC Sound Interface. Used for audio data transmission, primarily for streaming digital audio data to DACs, ADCs, and other audio peripherals.
SDIO Slave	Secure Digital Input Output. Works as a slave interface for SD cards, enabling data storage and retrieval.
SDMMC	Secure Digital Memory Card Controller. Provides support for interfacing with SD cards and eMMC storage devices.
EMAC	Ethernet Media Access Controller. Used for handling Ethernet-based networking communication.
BT	Bluetooth controller. Used for Bluetooth communication, supporting both classic Bluetooth and Bluetooth Low Energy (BLE).
WIFI	Wi-Fi controller. Manages Wi-Fi communication, supporting 802.11b/g/n standards for wireless networking.

Ublox NEO-M8N GPS Module

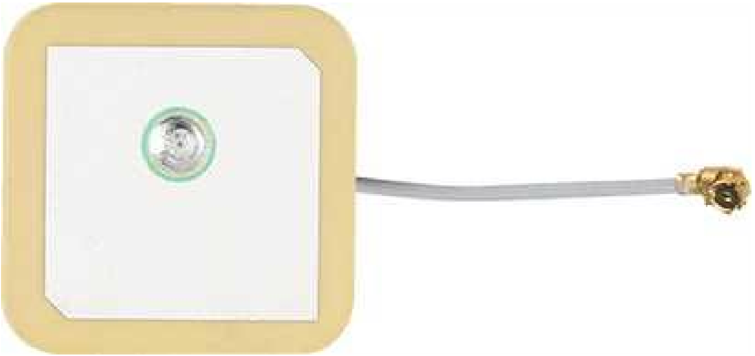
Good ref link



Parameter	Value
Material	ABS Plastic
Relative Permittivity	3
Thickness	6.35 mm
Signal Frequency	1575.42 MHz (GPS L1)
Estimated Attenuation	< 3 dB
Acceptable Loss	3 dB
Impact on Performance	Minimal

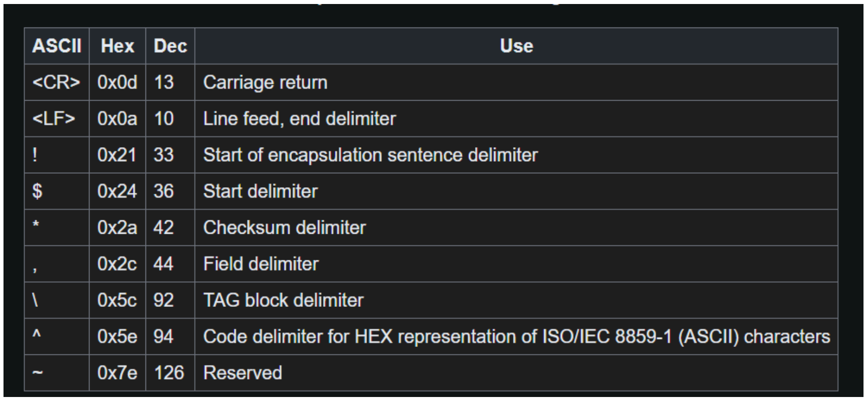
Antenna

The module comes with a -161 dBm sensitivity patch antenna for receiving radio signals from GPS satellites.



NMEA GPS Messaging Protocol

NMEA 0183 sentences



ASCII	Hex	Dec	Use
<CR>	0x0d	13	Carriage return
<LF>	0x0a	10	Line feed, end delimiter
!	0x21	33	Start of encapsulation sentence delimiter
$	0x24	36	Start delimiter
*	0x2a	42	Checksum delimiter
,	0x2c	44	Field delimiter
\	0x5c	92	TAG block delimiter
^	0x5e	94	Code delimiter for HEX representation of ISO/IEC 8859-1 (ASCII) characters
~	0x7e	126	Reserved

NMEA sentences are ASCII strings, where each character (including commas, dollar signs, and the asterisk) is represented by 1 byte in memory.

example:

$GPRMC,030742.00,A,2232.7830,N,11404.58520,E,0.356,,070314,,A*77

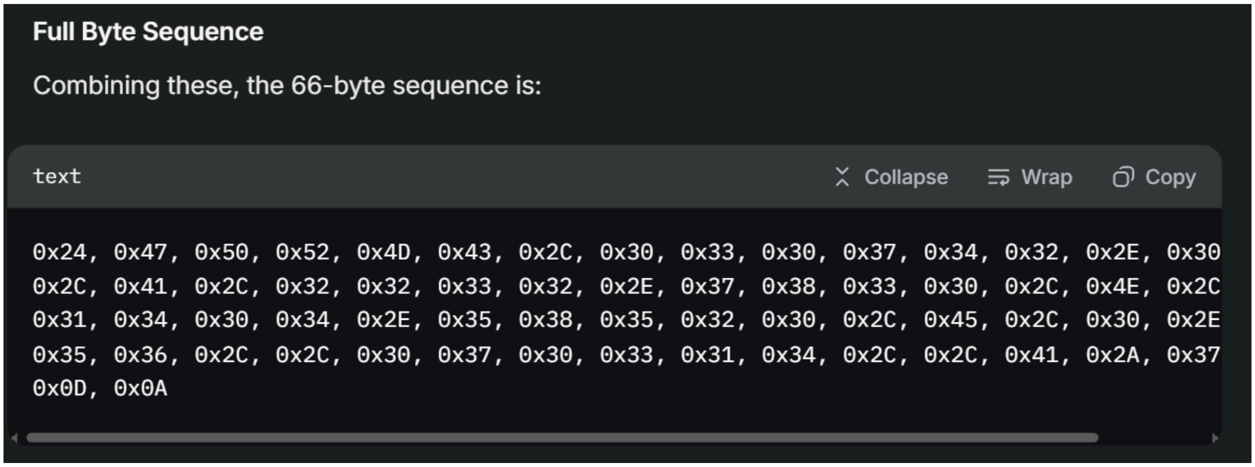
About 66 byes long including carriage characters /n

ASCII Values

Here’s the breakdown of the sentence into ASCII values (in hexadecimal for clarity):

- $GPRMC → 0x24, 0x47, 0x50, 0x52, 0x4D, 0x43
- , → 0x2C
- 030742.00 → 0x30, 0x33, 0x30, 0x37, 0x34, 0x32, 0x2E, 0x30, 0x30

- , → 0x2C
- A → 0x41
- , → 0x2C
- 2232.7830 → 0x32, 0x32, 0x33, 0x32, 0x2E, 0x37, 0x38, 0x33, 0x30
- , → 0x2C
- N → 0x4E
- , → 0x2C
- 11404.58520 → 0x31, 0x31, 0x34, 0x30, 0x34, 0x2E, 0x35, 0x38, 0x35, 0x32, 0x30
- , → 0x2C
- E → 0x45
- , → 0x2C
- 0.356 → 0x30, 0x2E, 0x33, 0x35, 0x36
- , → 0x2C
- , → 0x2C (empty field)
- 070314 → 0x30, 0x37, 0x30, 0x33, 0x31, 0x34
- , → 0x2C
- , → 0x2C (empty field)
- A → 0x41
- *77 → 0x2A, 0x37, 0x37
- \r\n → 0x0D, 0x0A

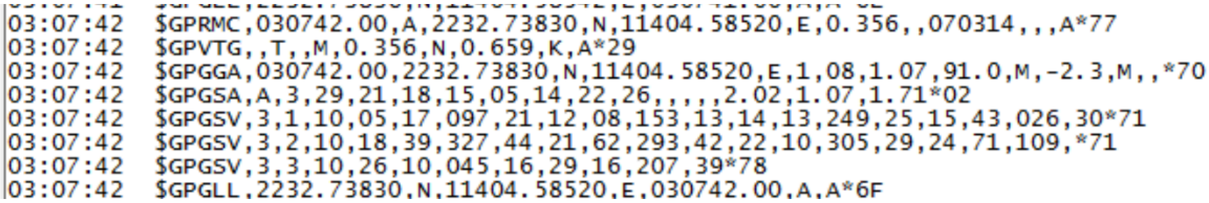


Full Byte Sequence

Combining these, the 66-byte sequence is:

```
0x24, 0x47, 0x50, 0x52, 0x4D, 0x43, 0x2C, 0x30, 0x33, 0x30, 0x37, 0x34, 0x32, 0x2E, 0x30
0x2C, 0x41, 0x2C, 0x32, 0x32, 0x33, 0x32, 0x2E, 0x37, 0x38, 0x33, 0x30, 0x2C, 0x4E, 0x2C
0x31, 0x34, 0x30, 0x34, 0x2E, 0x35, 0x38, 0x35, 0x32, 0x30, 0x2C, 0x45, 0x2C, 0x30, 0x2E
0x35, 0x36, 0x2C, 0x2C, 0x30, 0x37, 0x30, 0x33, 0x31, 0x34, 0x2C, 0x2C, 0x41, 0x2A, 0x37
0x0D, 0x0A
```

example:

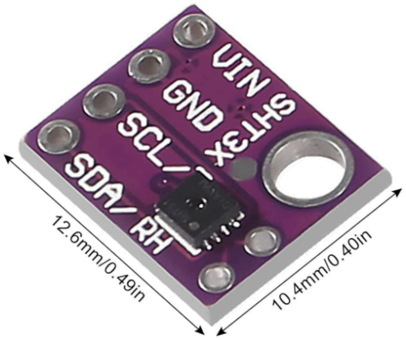


```
03:07:42   $GPRMC,030742.00,A,2232.73830,N,11404.58520,E,0.356,,070314,,,A*77
03:07:42   $GPVTG,,T,,M,0.356,N,0.659,K,A*29
03:07:42   $GPGGA,030742.00,2232.73830,N,11404.58520,E,1,08,1.07,91.0,M,-2.3,M,,*70
03:07:42   $GPGSA,A,3,29,21,18,15,05,14,22,26,,,,,2.02,1.07,1.71*02
03:07:42   $GPGSV,3,1,10,05,17,097,21,12,08,153,13,14,13,249,25,15,43,026,30*71
03:07:42   $GPGSV,3,2,10,18,39,327,44,21,62,293,42,22,10,305,29,24,71,109,*71
03:07:42   $GPGSV,3,3,10,26,10,045,16,29,16,207,39*78
03:07:42   $GPGLL,2232.73830,N,11404.58520,E,030742.00,A,A*6F
```

The number of bytes depends on the NMEA sentence:

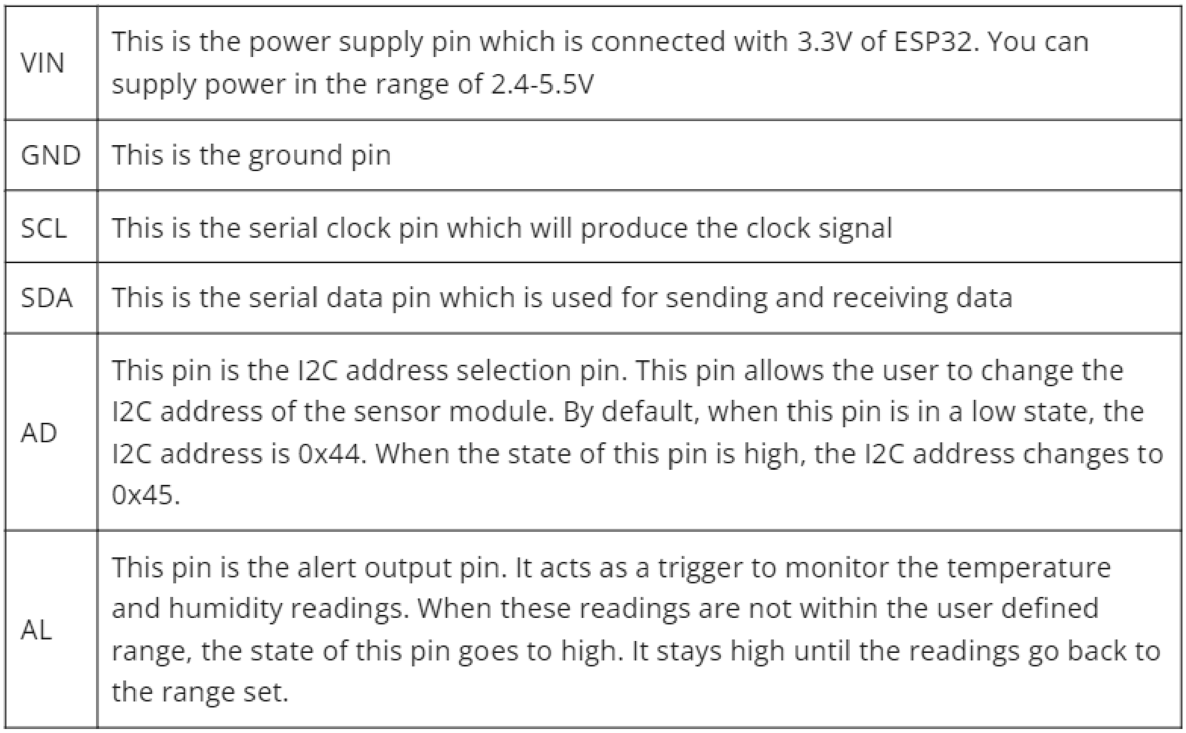
- $GPRMC: 66 bytes (including \r\n).
- $GPVTG: 35 bytes.
- $GPGGA: 69 bytes.
- $GPGSA: 47 bytes.
- $GPGSV: 54 bytes.
- $GPGLL: 42 bytes.

TEMP and Humidity Sensor



SHT31-DIS-B Sensor
SHT: Sensirion Temperature/Humidity sensors
High-precision humidity and temperature sensor, I2C interface

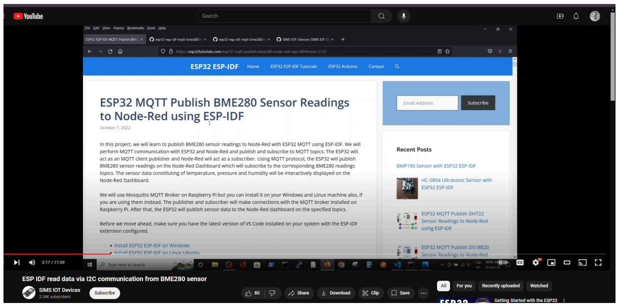
Pinout Description



VIN	This is the power supply pin which is connected with 3.3V of ESP32. You can supply power in the range of 2.4-5.5V
GND	This is the ground pin
SCL	This is the serial clock pin which will produce the clock signal
SDA	This is the serial data pin which is used for sending and receiving data
AD	This pin is the I2C address selection pin. This pin allows the user to change the I2C address of the sensor module. By default, when this pin is in a low state, the I2C address is 0x44. When the state of this pin is high, the I2C address changes to 0x45.
AL	This pin is the alert output pin. It acts as a trigger to monitor the temperature and humidity readings. When these readings are not within the user defined range, the state of this pin goes to high. It stays high until the readings go back to the range set.

SDA/RH: serial data relative humidity

SHTX tutorial


YouTube
Search
File Edit View History Bookmarks Tools Help
ESP32 ESP-IDF MQTT: Publish BME
esp32-esp-idf-mqtt-bme280/
esp32-esp-idf-mqtt-bme280/
SIMS-IOT-Devices (SIMS IOT D
https://esp32tutorials.com/esp32-mqtt-publish-bme280-node-red-esp-idf/#more-2125
ESP32 ESP-IDF
Home
ESP32 ESP-IDF Tutorials
ESP32 Arduino
Contact
Email Address
Subscribe
Recent Posts
BMP180 Sensor with ESP32 ESP-IDF
HC-SR04 Ultrasonic Sensor with ESP32 ESP-IDF
ESP32 MQTT Publish DHT22 Sensor Readings to Node-Red using ESP-IDF
ESP32 MQTT Publish DS18B20 Sensor Readings to Node-Red
2:17 / 11:59
Type here to search
SIMS IOT Devices
2.34K subscribers
Subscribe
80
Share
Download
Clip
Save
All
For you
Recently uploaded
Watched
Getting Started with the ESP32

ESP32 MQTT Publish BME280 Sensor Readings to Node-Red using ESP-IDF

October 7, 2022

In this project, we will learn to publish BME280 sensor readings to Node-Red with ESP32 MQTT using ESP-IDF. We will perform MQTT communication with ESP32 and Node-Red and publish and subscribe to MQTT topics. The ESP32 will act as an MQTT client publisher and Node-Red will act as a subscriber. Using MQTT protocol, the ESP32 will publish BME280 sensor readings on the Node-Red Dashboard which will subscribe to the corresponding BME280 readings topics. The sensor data constituting of temperature, pressure and humidity will be interactively displayed on the Node-Red Dashboard.

We will use Mosquitto MQTT Broker on Raspberry Pi but you can install it on your Windows and Linux machine also, if you are using them instead. The publisher and subscriber will make connections with the MQTT broker installed on Raspberry Pi. After that, the ESP32 will publish sensor data to the Node-Red dashboard on the specified topics.

Before we move ahead, make sure you have the latest version of VS Code installed on your system with the ESP-IDF extension configured.

- Install ESP32 ESP-IDF on Windows
- Install ESP32 ESP-IDF on Linux Ubuntu

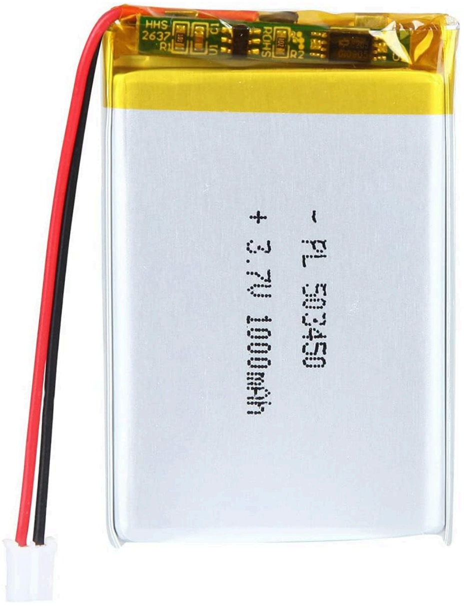
ESP IDF read data via I2C communication from BME280 sensor

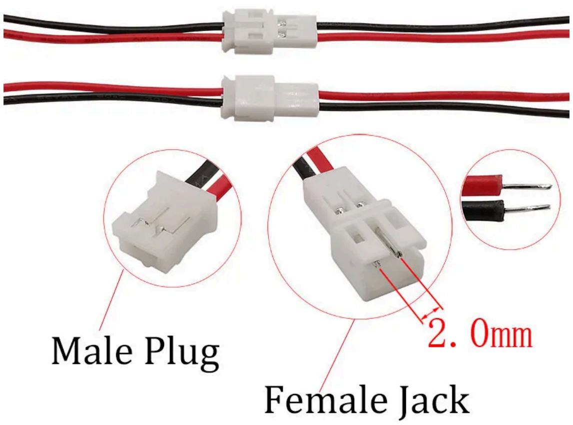
ESP IDF read data via I2C communication from BME280 sensor

SHT31 Temperature and Humidity Sensor with ESP32 ESP-IDF link

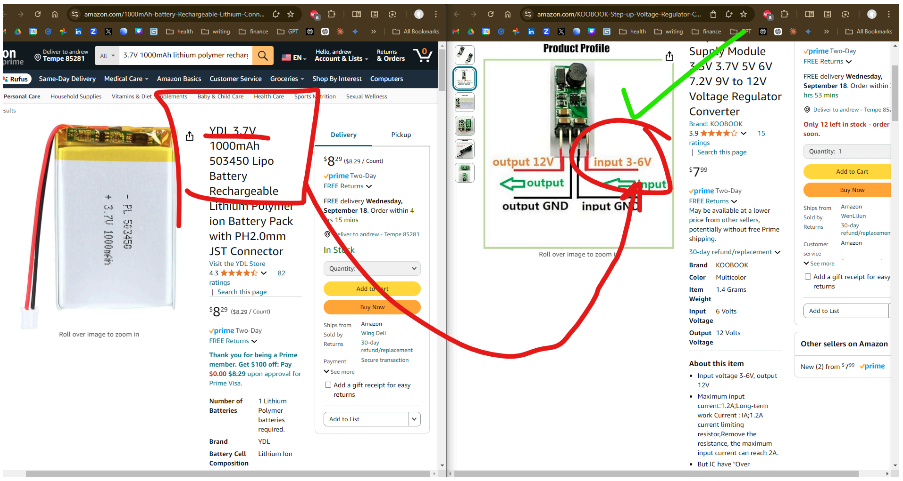
Microphone

Power





JST-PH2.0 connector



YDL 3.7V 1000mAh 503450 Lipo Battery Rechargeable Lithium Polymer ion Battery Pack with PH2.0mm JST Connector

Visit the YDL Store

4.3 ★★★★☆ 82 ratings | Search this page

$8.29 ($8.29 / Count)

✓prime Two-Day

FREE Returns

Thank you for being a Prime member. Get $100 off: Pay $0.00 ~~$8.29~~ upon approval for Prime Visa.

Number of Batteries	1 Lithium Polymer batteries required.
Brand	YDL
Battery Cell Composition	Lithium Ion

$8.29 ($8.29 / Count)

✓prime Two-Day

FREE Returns

FREE delivery **Wednesday, September 18**. Order within 4 hrs 15 mins

Deliver to andrew - Tempe 85281

In Stock

Ships from	Amazon
Sold by	Wing Deli
Returns	30-day refund/replacement
Payment	Secure transaction

□ Add a gift receipt for easy returns

Supply Module 3.5V 3.7V 5V 6V 7.2V 9V to 12V Voltage Regulator Converter

Brand: KOOBOOK

3.9 ★★★★☆ 15 ratings | Search this page

$7.99

✓prime Two-Day

FREE Returns

May be available at a lower price from other sellers, potentially without free Prime shipping.

30-day refund/replacement

Brand	KOOBOOK
Color	Multicolor
Item Weight	1.4 Grams
Input Voltage	6 Volts
Output Voltage	12 Volts

About this item

- Input voltage 3-6V, output 12V
- Maximum input current:1.2A;Long-term work Current : IA;1.2A current limiting resistor,Remove the resistance, the maximum input current can reach 2A.
- But IC have "Over

✓prime Two-Day

FREE Returns

FREE delivery **Wednesday, September 18.** Order within hrs 53 mins

Deliver to andrew - Tempe 852

Only 12 left in stock - order soon.

Ships from	Amazon
Sold by	WenLiJun
Returns	30-day refund/replacement
Customer service	Amazon

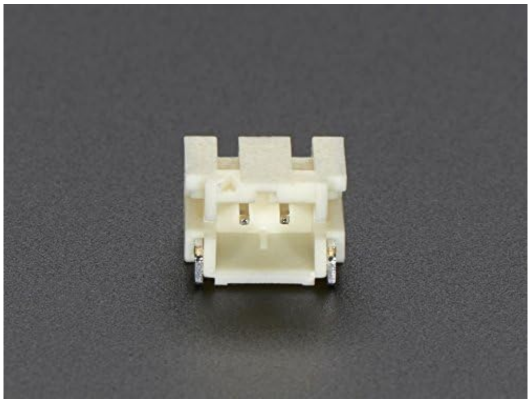
□ Add a gift receipt for easy returns

Other sellers on Amazon

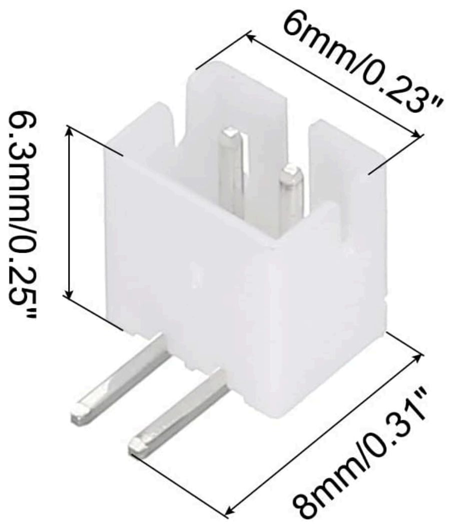
New (2) from $7.99 ✓prime

Solder a JST-PH2.0 connector

Adafruit JST-PH 2-Pin SMT Right Angle Connector [ADA1769]



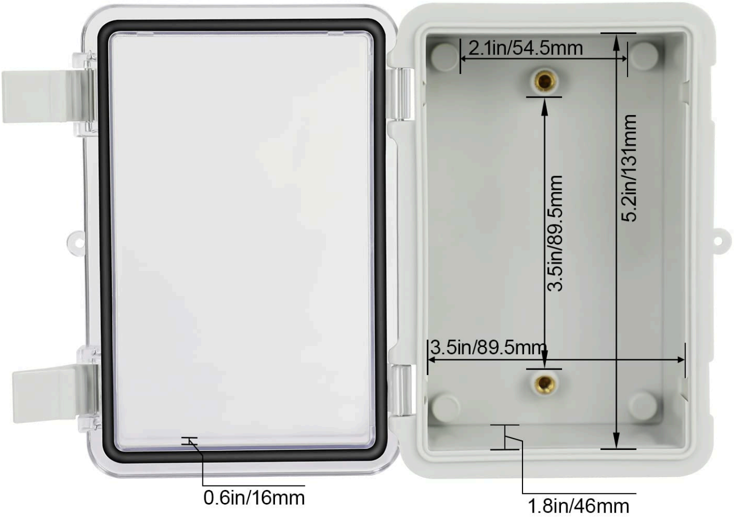
maybeeeee????????-->No ❌



Enclosure

ABS Plastic Dustproof Waterproof IP65 Junction Box Hinged Shell Universal Electrical Project Enclosure Gray, with PC Transparent Clear Cover

Internal Size (Allowable Error : ±2mm)



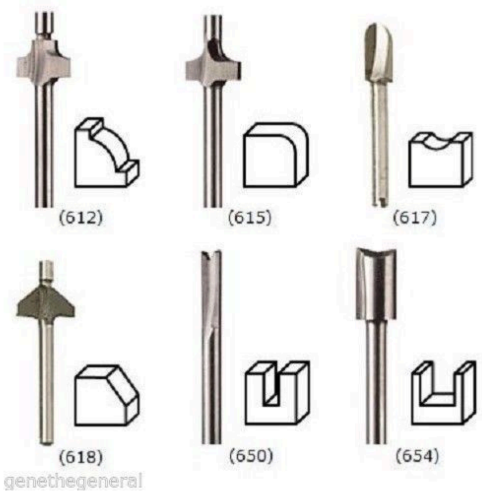
Acrylonitrile Butadiene Styrene, or ABS, is an opaque thermoplastic. It is an amorphous polymer comprised of three monomers, acrylonitrile, butadiene and styrene.

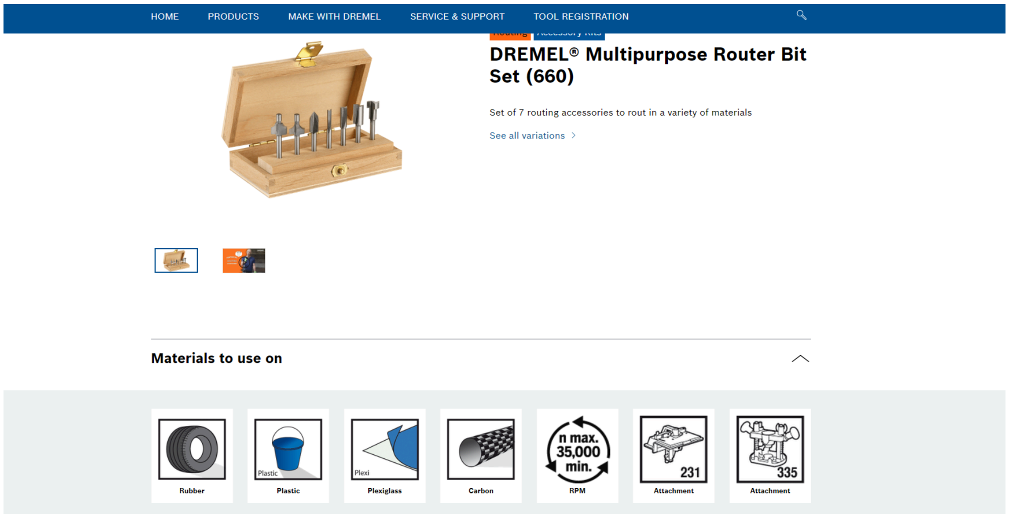
IP65 is an international standard rating that indicates the level of protection an enclosure provides against dust and water ingress.

IP stands for Ingress Protection.
The first digit (6) refers to dust protection:
6 signifies dust tight, meaning no dust can enter the enclosure.

The second digit (5) refers to water protection:
5 indicates protection against water jets from any direction. It can withstand low-pressure water jets, but not complete submersion.

Routing Material



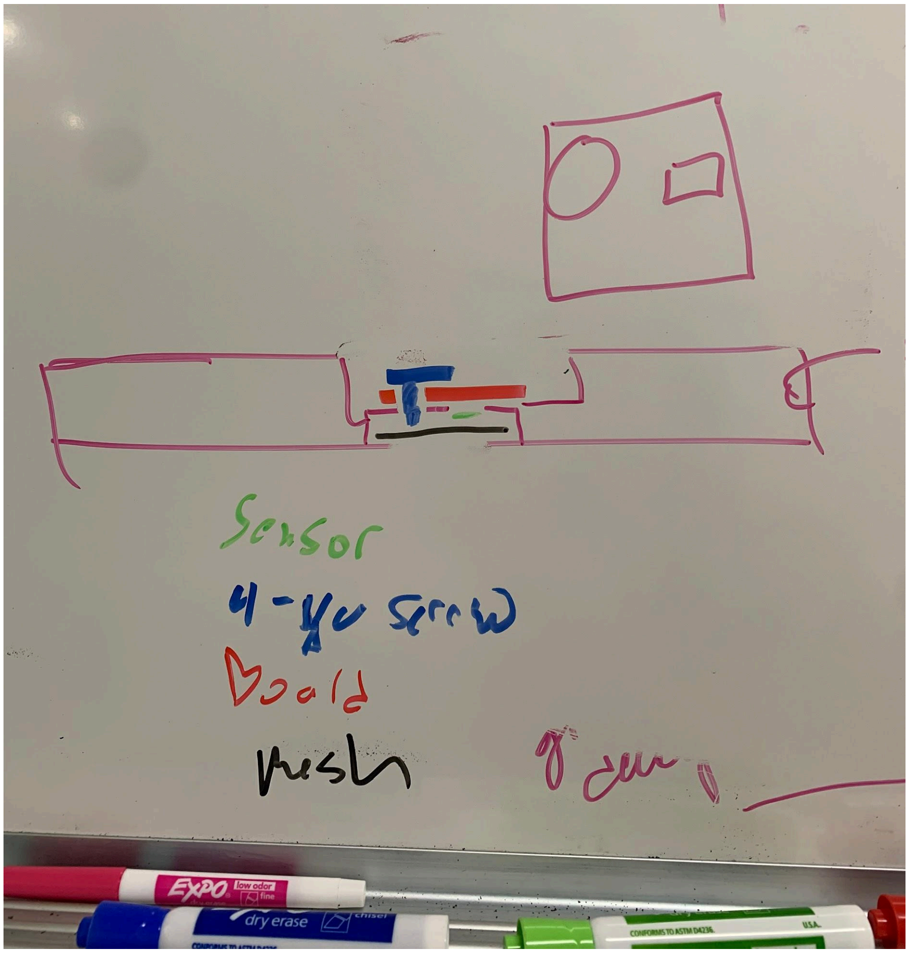


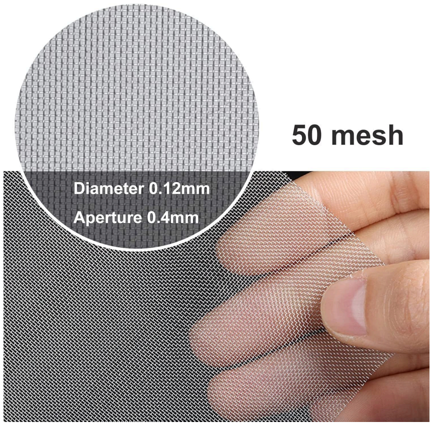
DREMEL® Multipurpose Router Bit Set (660)

Set of 7 routing accessories to rout in a variety of materials

See all variations ›

Materials to use on

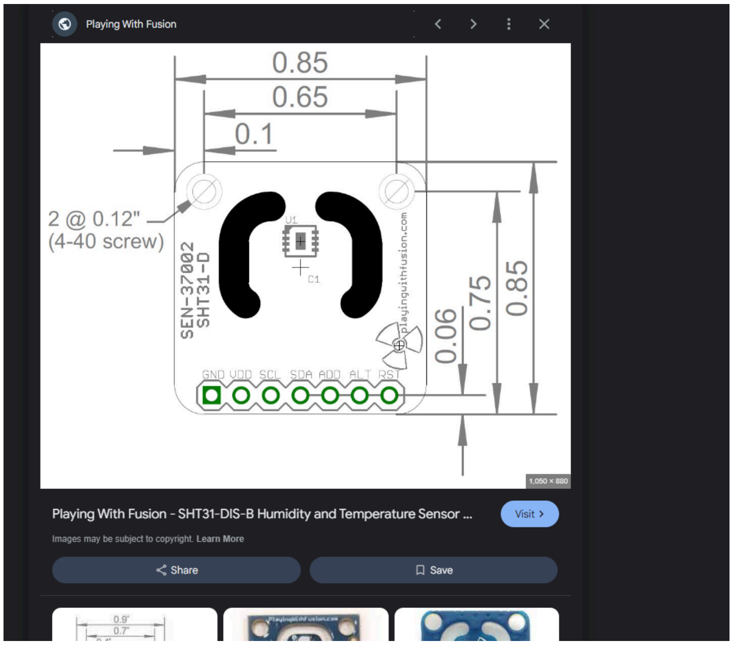




Stainless Steel Woven Wire Mesh

304 Stainless Steel Wire Mesh

50 Mesh - 0.4mm Hole Wire Metal Mesh Sheet

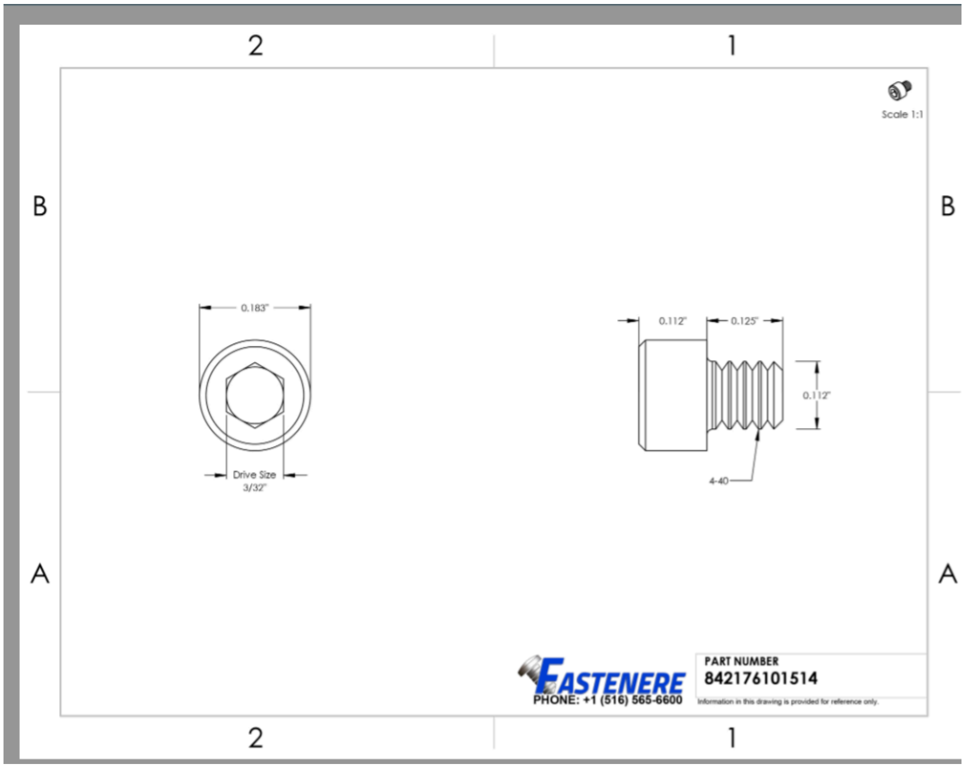


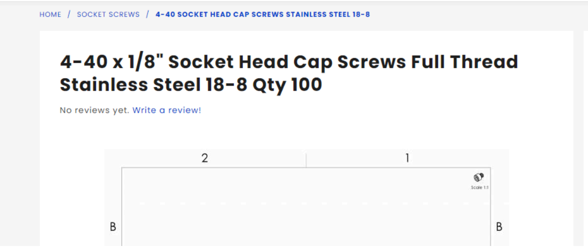
Playing With Fusion

Playing With Fusion - SHT31-DIS-B Humidity and Temperature Sensor ...

Images may be subject to copyright. Learn More

I think that is a screw




HOME / SOCKET SCREWS / 4-40 SOCKET HEAD CAP SCREWS STAINLESS STEEL 18-8

4-40 x 1/8" Socket Head Cap Screws Full Thread Stainless Steel 18-8 Qty 100

No reviews yet. Write a review!