# اینترنت اشیا (IOT)

**نیما قاسملو**
**معاون اداره کل سامانه های ملی و زیرساخت داده های مکانی**


**تیر ۱۴۰۳**

# اینترنت اشیا

- اینترنت اشیا (IoT) سیستمی از دستگاه‌های محاسباتی، ماشین‌های مکانیکی و دیجیتال، اشیا، حیوانات یا افراد است که با شناساگرهای یکتایی مشخص شده‌اند و توانایی انتقال داده در شبکه را بدون نیاز به تعاملات انسان با انسان یا انسان با کامپیوتر دارند.

- عبارت اینترنت اشیاء، نخستین بار در ۱۹۹۹ میلادی توسط کوین اشتون بریتانیایی معرفی شد. اشتون این مفهوم را در قالب دنیایی که در آن هر چیز و هر شی‌ای، دارای هویت دیجیتال باشد و کامپیوترها آن‌ها را کنترل و مدیریت نمایند، مطرح نمود.


# اینترنت اشیا

- اتصال دستگاه‌ها IoT :به بیش از میلیاردها دستگاه اجازه می‌دهد تا به اینترنت متصل شوند و اطلاعات را جمع‌آوری و به اشتراک بگذارند.

- حسگرهای هوشمند: با اضافه کردن حسگرها به دستگاه‌ها، IoT قابلیت تشخیص و جمع‌آوری اطلاعات محیطی را فراهم می‌کند، مانند دما، رطوبت، فشار و سایر پارامترها.

# نمونه‌های کاربردی اینترنت اشیا

- خانه هوشمند: با استفاده از IoT می‌توانید خانه خود را به یک خانه هوشمند تبدیل کنید، که شما را قادر می‌سازد بر روی امور مختلف مانند نورپردازی، سیستم‌های امنیتی، ترموستات و دستگاه‌های خانگی از راه دور کنترل داشته باشید.

- صنعت هوشمند: IoT در صنعت بهینه‌سازی فرآیندهای تولید و مدیریت دستگاه‌ها را امکان‌پذیر می‌کند، که منجر به افزایش بهره‌وری و کاهش هزینه‌ها می‌شود.


# نمونه‌های کاربردی اینترنت اشیا

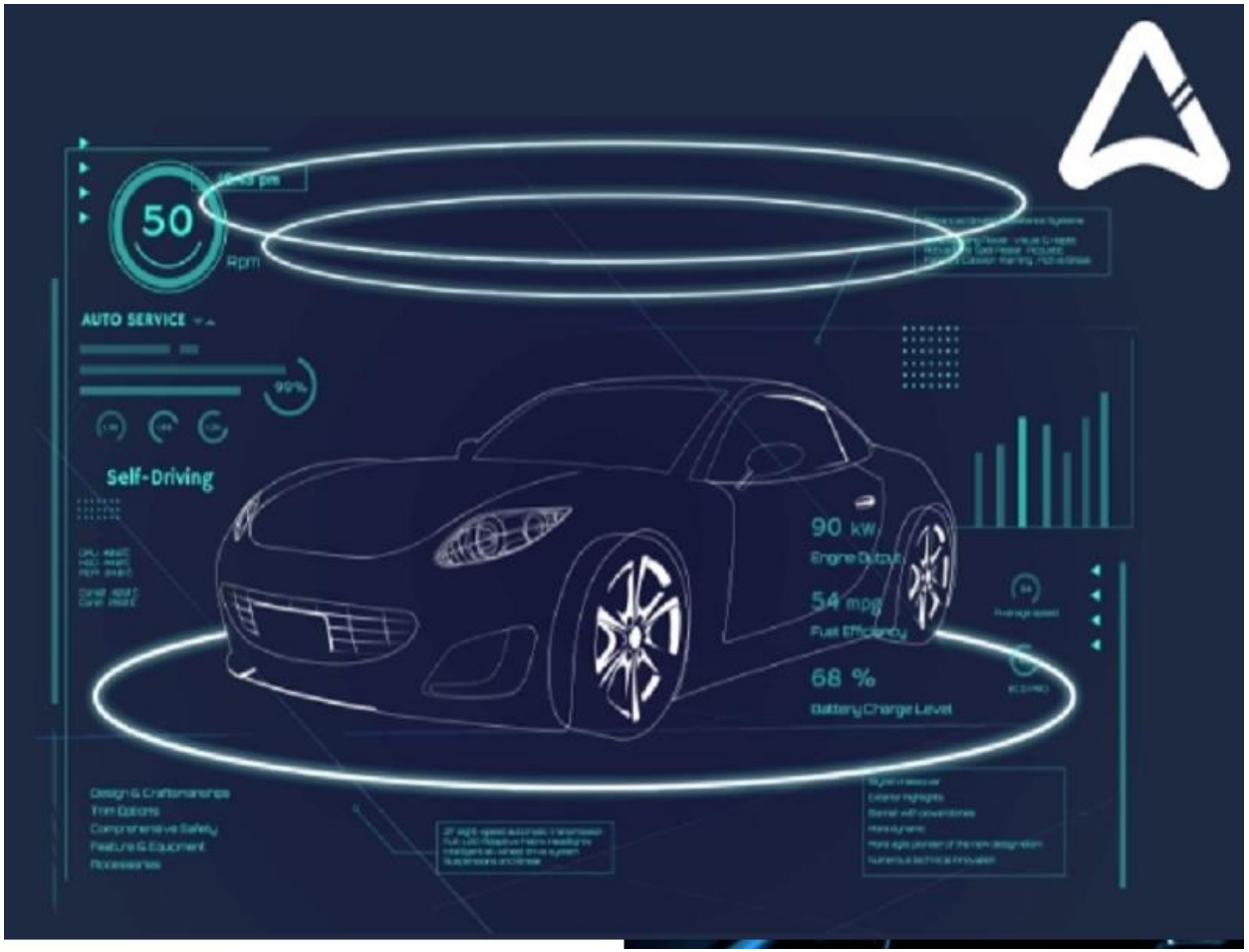
- شهر هوشمند: اینترنت اشیا با ادغام اطلاعات شهری و استفاده از حسگرها و دستگاه‌های متصل، شهرها را هوشمندتر می‌کند و خدماتی مانند پارکینگ هوشمند، مدیریت ترافیک، روشنایی هوشمند و مدیریت پسماند را بهبود می‌بخشد.
- سلامتی هوشمند: IoTدر حوزه سلامتی می‌تواند تجهیزاتی مانند ساعت‌های هوشمند، دستگاه‌های پوشیدنی و سیستم‌های ارتباطی برای پایش سلامتی فراهم کند و مدیریت بهتری برای مراقبت پزشکی فراهم آورد.

# نمونه‌های کاربردی اینترنت اشیا

- کشاورزی هوشمند: IoTدر کشاورزی می‌تواند با استفاده از حسگرها و دستگاه‌های متصل، بهبودی در مدیریت آبیاری، کنترل محیط رشد گیاهان و بهینه‌سازی عملکرد مزارع ایجاد کند.

- خودروهای هوشمند: IoTبه خودروها امکان ارتباط با شبکه‌های دیگر و سیستم‌های جاده‌ای را می‌دهد

# نمونه‌های کاربردی اینترنت اشیا



# دلیل استفاده از اینترنت اشیا

- اتصال و کنترل: با استفاده از اینترنت اشیا، می‌توانیم دستگاه‌ها و اشیا را به یکدیگر و به کاربران متصل کنیم. این امکان را به ما می‌دهد تا دستگاه‌ها را از راه دور کنترل کنیم و داده‌ها و اطلاعات را به صورت زنده جمع آوری کنیم.

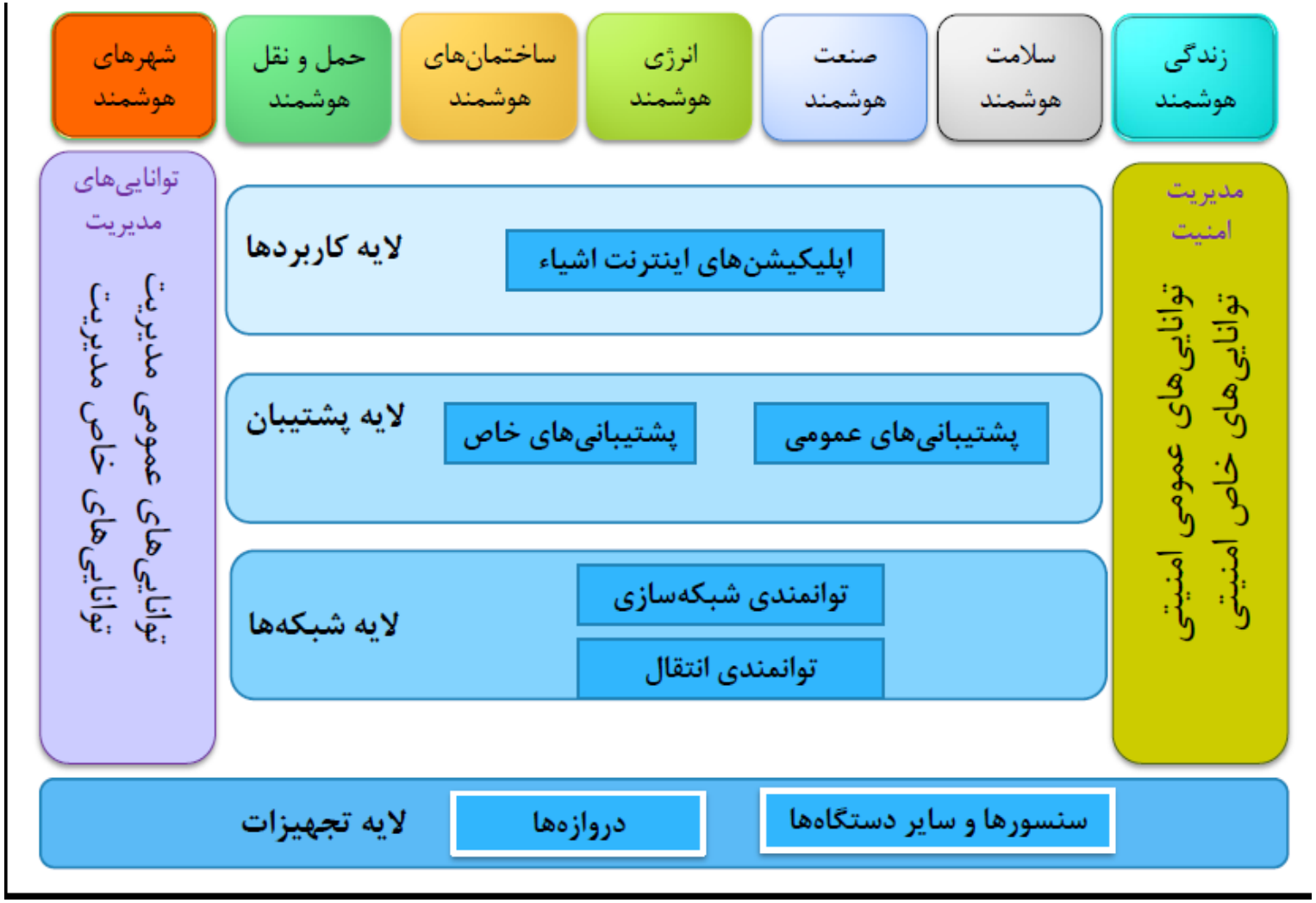
- جمع‌آوری داده‌ها و تحلیل: با اتصال اشیا به اینترنت، می‌توانیم داده‌های بزرگی را از سنسورها و دستگاه‌ها جمع آوری کنیم. این داده‌ها به ما امکان می‌دهد تا الگوها و روندهای مختلف را شناسایی کرده و تحلیل کنیم. این اطلاعات می‌توانند در بهبود فرآیندها، افزایش بهره‌وری و تصمیم‌گیری بهتر به کار گرفته شوند.

- هوشمندسازی و خودکارسازی: با استفاده از اینترنت اشیا، می‌توانیم اشیا را هوشمند کنیم و آنها را به طور خودکار براساس شرایط محیطی و داده‌های دریافتی عمل کنیم. این امکان را به ما می‌دهد تا فرآیندها را بهبود داده و هزینه‌ها و زمان مورد نیاز را کاهش دهیم.

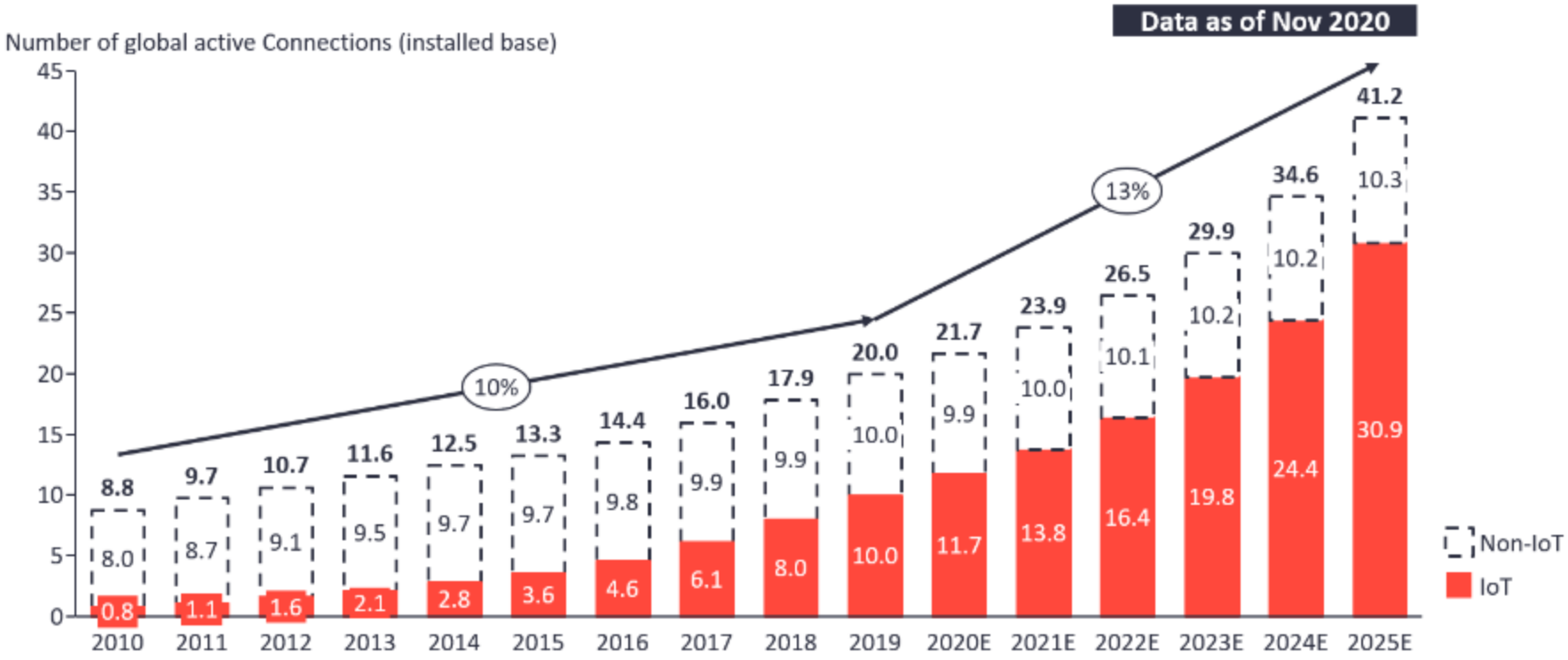
- ارتباطات ماشین به ماشین (M2M) : استفاده از اینترنت اشیا ارتباطات ماشین به ماشین را امکان‌پذیر می‌سازد. این به معنای این است که دستگاه‌ها و اشیا می‌توانند به صورت مستقل با یکدیگر ارتباط برقرار کنند و بدون نیاز به تداخل انسانی اطلاعات را تبادل کنند.

# تاریخچه اینترنت اشیا

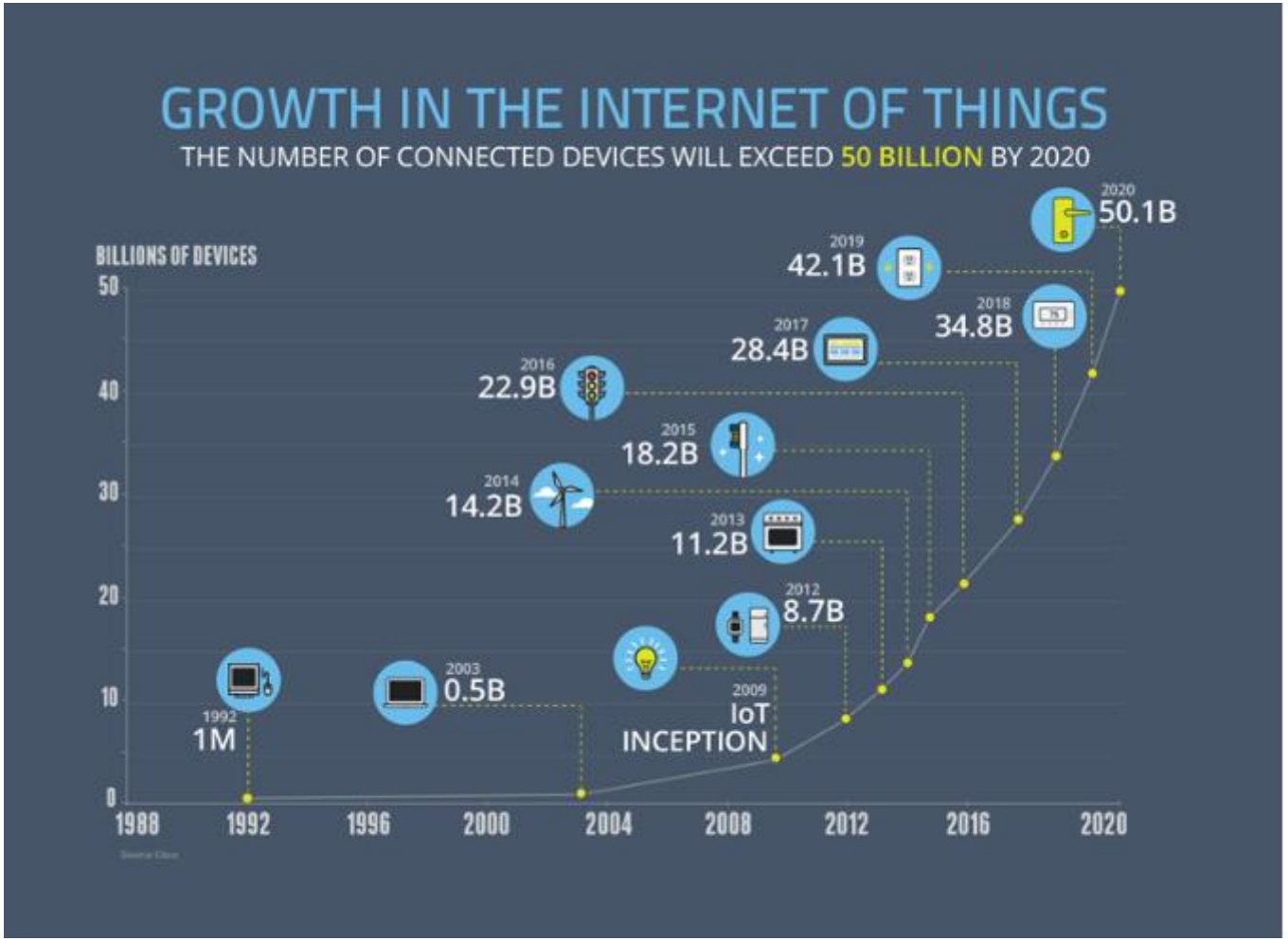
# معماری اینترنت اشیا



# چشم انداز اینترنت اشیا

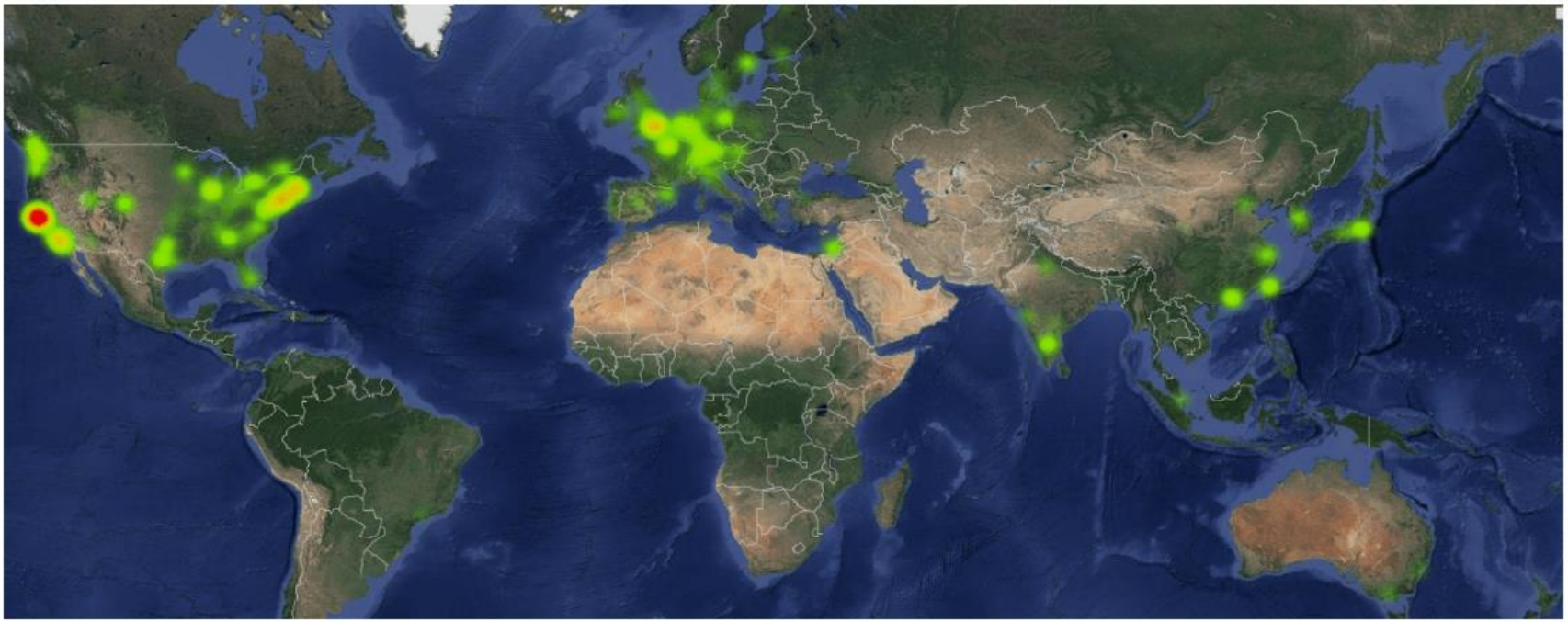


# چشم انداز اینترنت اشیا

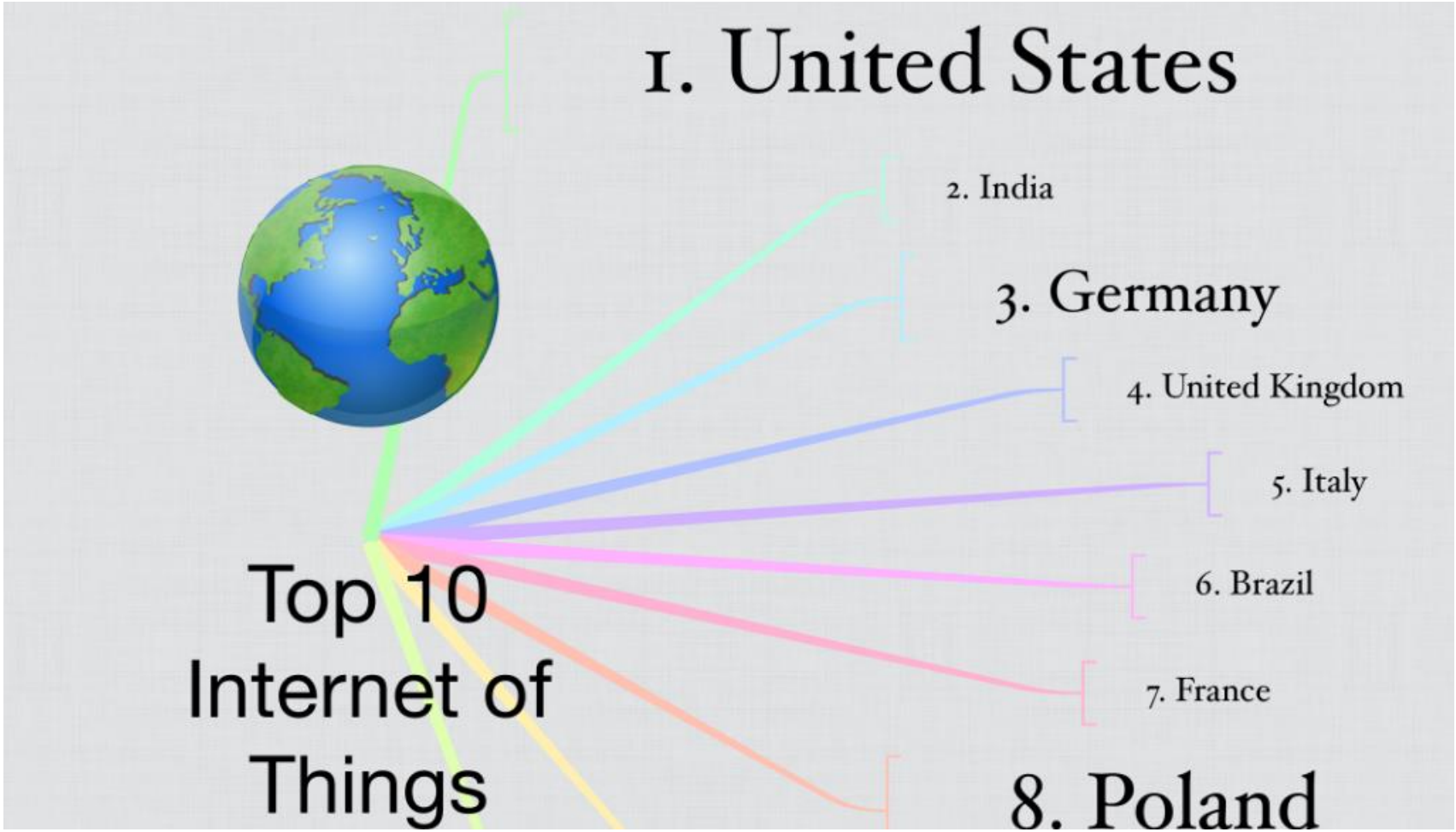


# چشم انداز اینترنت اشیا

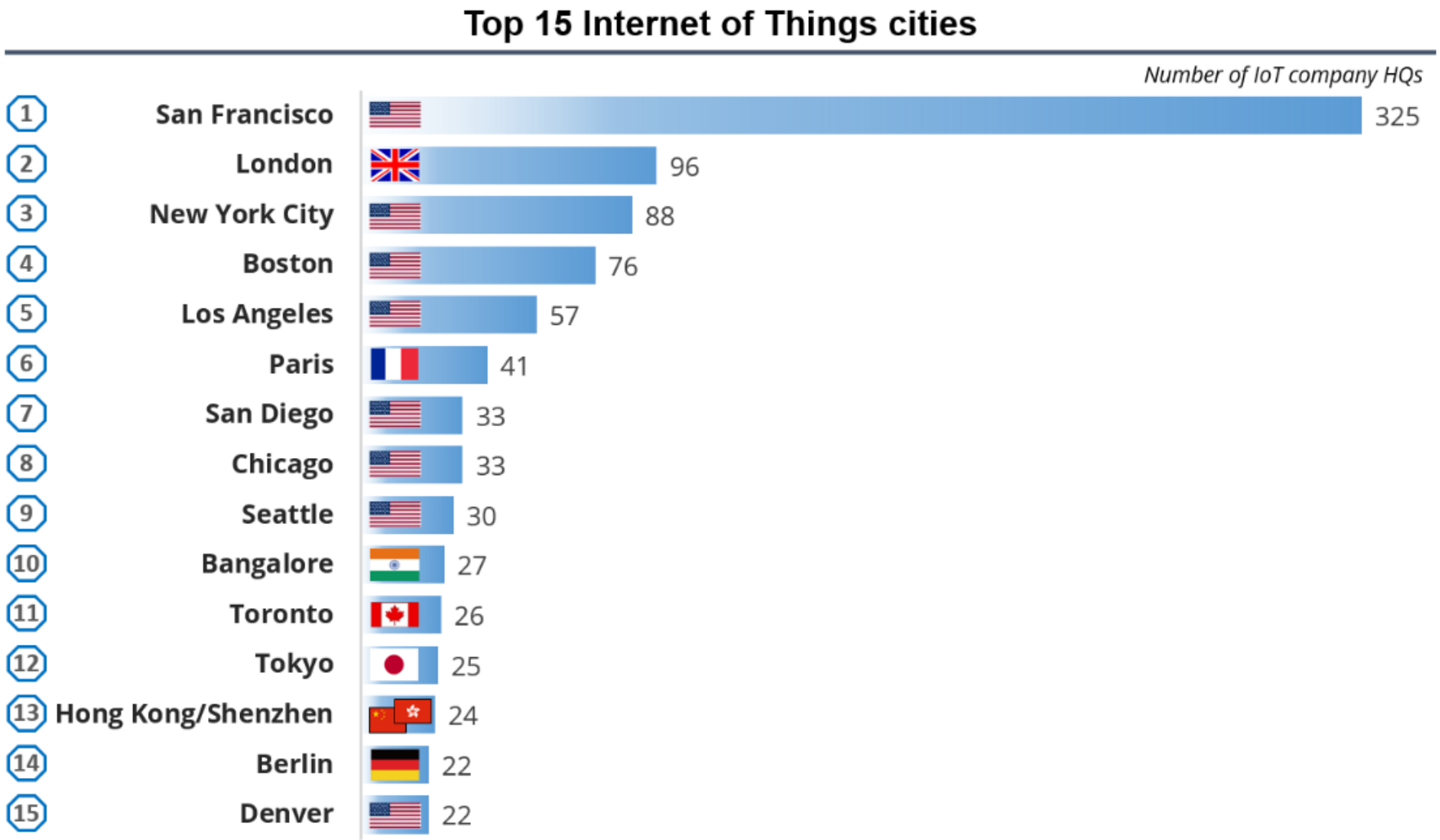
# چشم انداز اینترنت اشیا



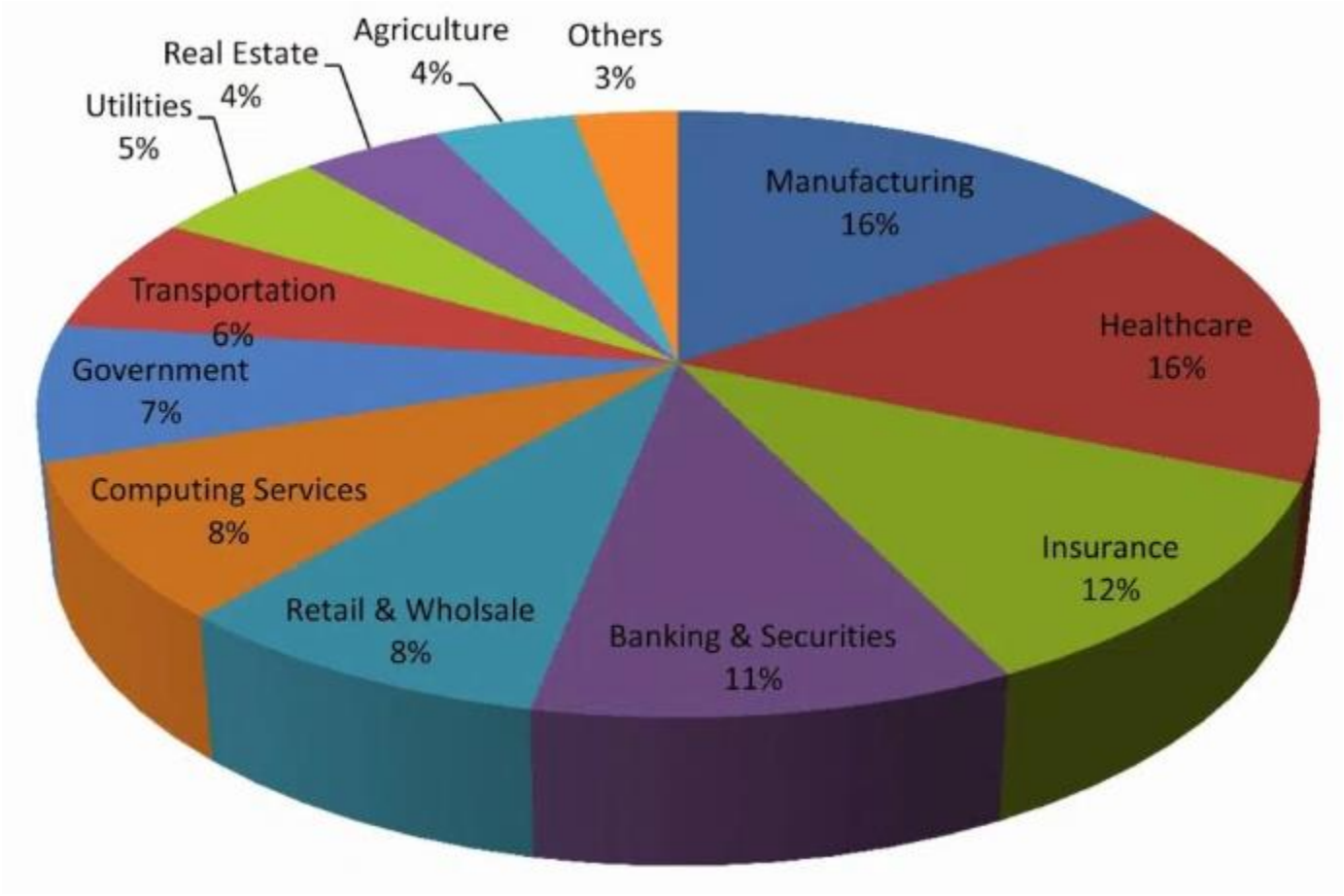
# چشم انداز اینترنت اشیا



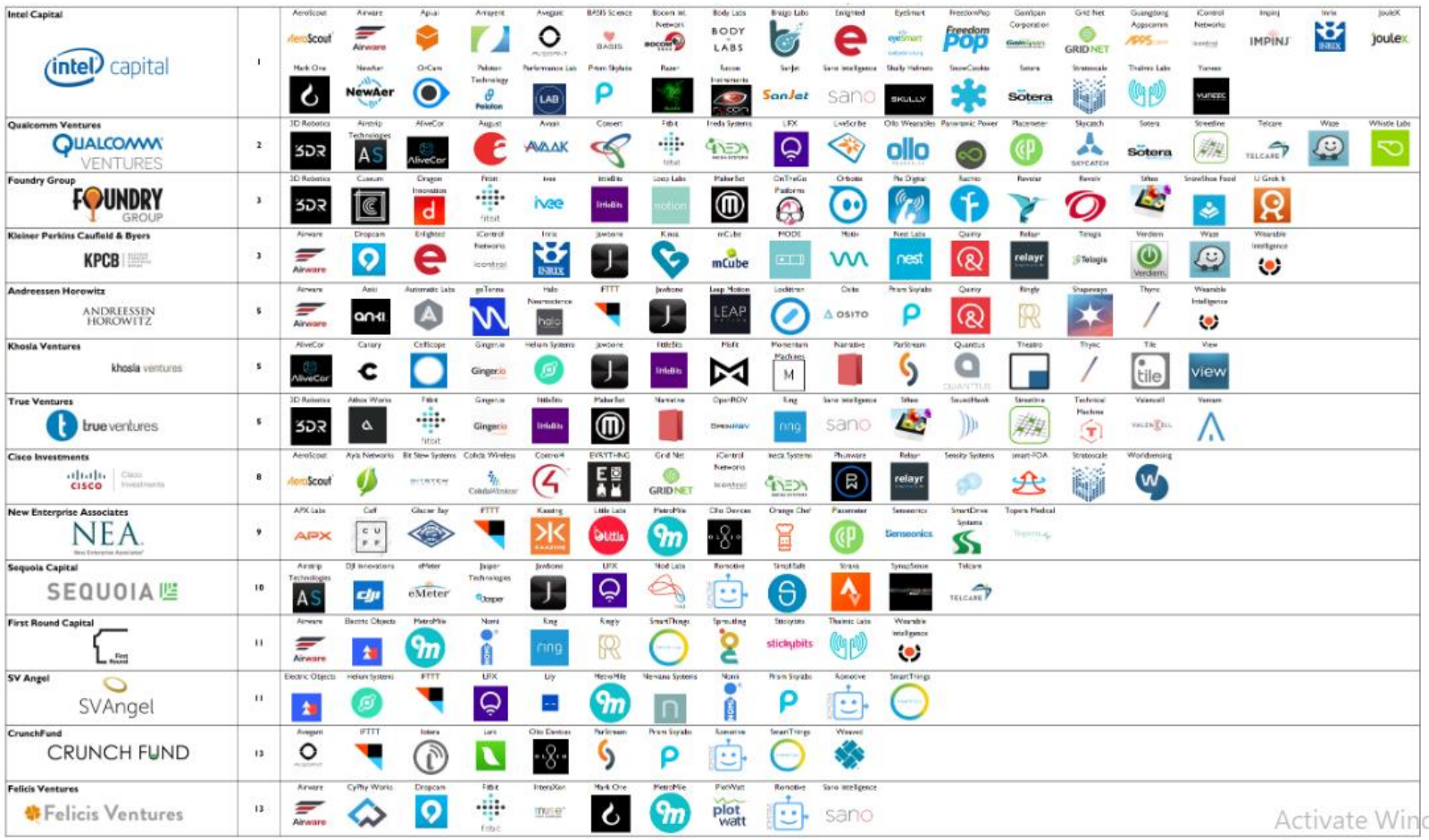
# چشم انداز اینترنت اشیا



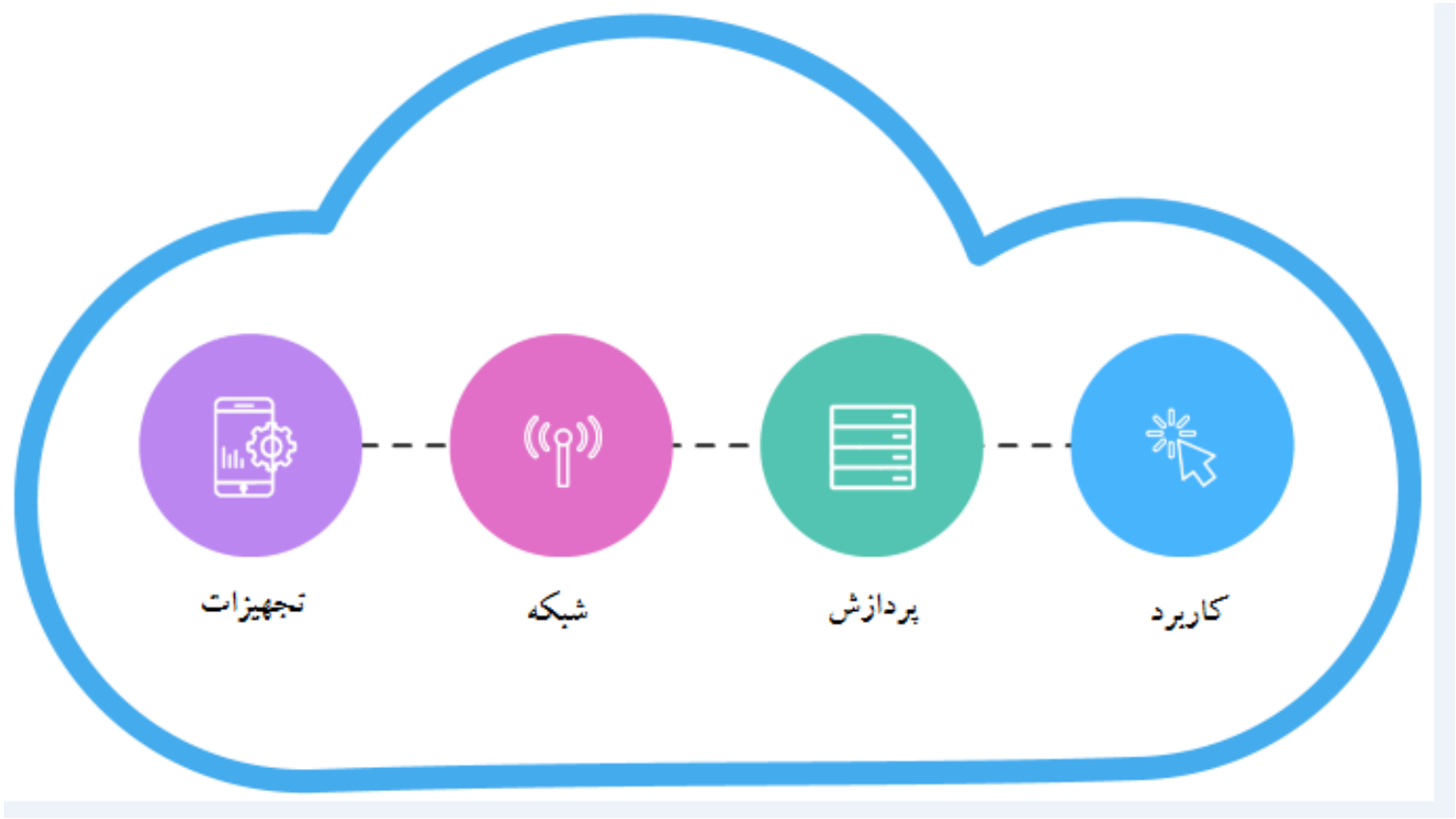
# چشم انداز اینترنت اشیا



# چشم انداز اینترنت اشیا

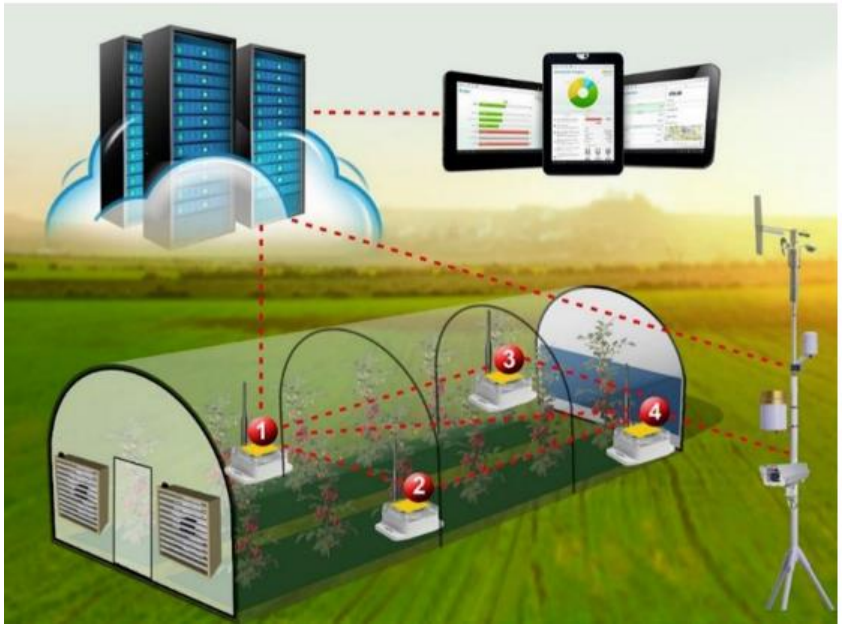


# لایه های اینترنت اشیا

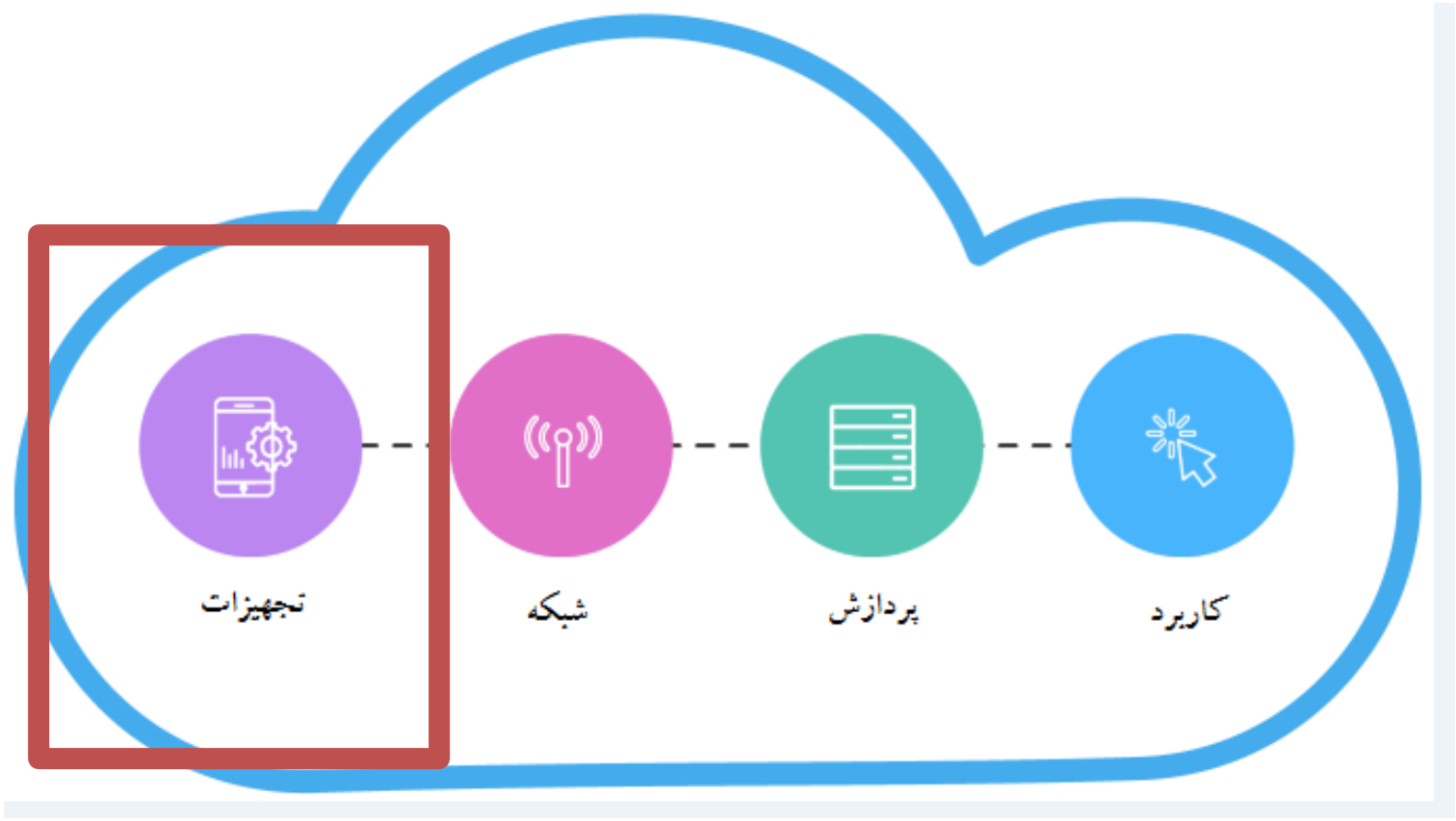


# چالش‌های اینترنت اشیا

- چالش‌های شبکه
- چالشهای کاربری( امنیت، تهیه نسخه پشتیبان)
- چالش‌های نظارتی و قانونگذاری
- تعامل‌پذیری
- تامین انرژی
- کیفیت ارائه خدمات
- داده‌های حجیم



# لایه های اینترنت اشیا



# سنسور

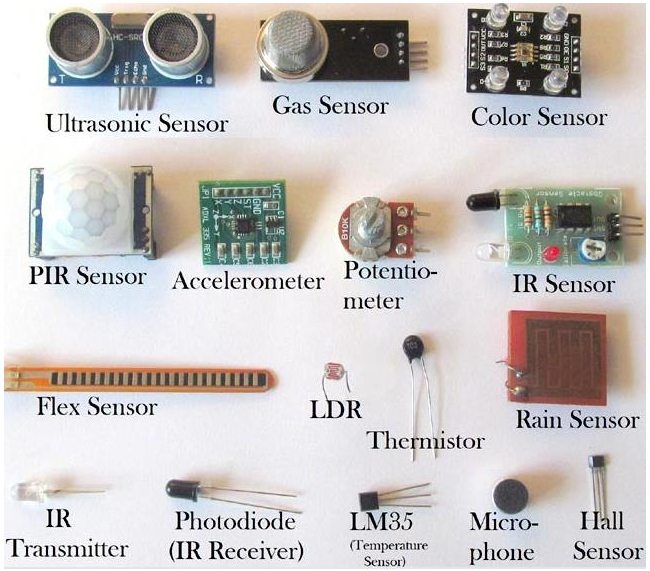
- سنسورها دستگاه‌هایی هستند که اطلاعات فیزیکی را مانند درجه حرارت، فشار، نور، صدا و وضعیت دیگر پارامترهای محیطی تشخیص می‌دهند و آنها را به یک سیستم الکترونیکی یا کامپیوتر منتقل می‌کنند.

- سنسورها به صورت معمول شامل یک عنصر حساسه و الکترونیک هستند.

عنصر حساسه می‌تواند از نوع مختلفی باشد، از جمله ترانزیستورها، فتودیودها، ترانزیستورهای نیمه هادی، پیزوالکتریک ها، سوئیچ‌های آلمانیوم و غیره.

قسمت الکترونیکی برای پردازش سیگنال‌ها و تقویت آنها استفاده می‌شود تا اطلاعات حاصل از سنسور به صورت مفهومی و قابل استفاده برای سیستم دیگری تبدیل شود.

- سنسورها برای کنترل، اندازه‌گیری، رصد و مانیتورینگ در بسیاری از صنایع و کاربردها استفاده می‌شوند



# کاربردهای سنسور

- صنعت خودرو: سنسورها در خودروها برای اندازه‌گیری دما، فشار لاستیک‌ها، سرعت، شتاب، دور موتور، سطح سوخت و سایر پارامترها استفاده می‌شوند.
- پزشکی: در پزشکی، سنسورها برای اندازه‌گیری ضربان قلب، فشار خون، سطح گلوکز در خون، دما و سایر شاخص‌های بیماری و سلامتی استفاده می‌شوند.
- صنعت هوا و فضا: سنسورها در صنعت هوا و فضا برای اندازه‌گیری دما، فشار، شتاب، نور، رطوبت و سایر پارامترهای محیطی استفاده می‌شوند.
- صنعت الکترونیک: در صنعت الکترونیک، سنسورها جهت تشخیص حرکت، تشخیص اثر انگشت، تشخیص فاصله، تشخیص چهره و سایر کاربردها استفاده می‌شوند.
- کشاورزی هوشمند: سنسورها در کشاورزی هوشمند برای اندازه‌گیری رطوبت خاک، دما، نور، PHخاک و سایر عوامل محیطی استفاده می‌شوند تا بهینه‌سازی منابع آب و بهبود عملکرد محصولات کشاورزی کمک کنند.
- صنعت نفت و گاز: سنسورها در این صنعت برای اندازه‌گیری فشار، دما، جریان و سایر پارامترهای مربوط به استخراج، حمل و نقل و ذخیره‌سازی نفت و گاز استفاده می‌شوند.
- صنعت روباتیک: سنسورها در روباتیک برای تشخیص محیط، شناسایی اشیاء، تشخیص فاصله، تشخیص حرکت و کنترل حرکت روبات‌ها استفاده می‌شوند.

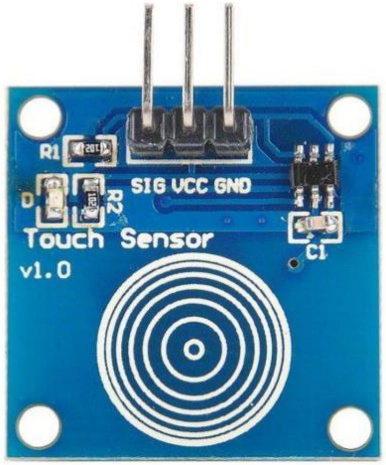
# قسمت‌های سنسور

- المان حسگر: این قسمت مسئول تبدیل یک پدیده فیزیکی (مثلاً دما، فشار، نور، شتاب و غیره) به یک سیگنال الکتریکی است. نوع المان حسگر متناسب با نوع و ویژگی‌های سنسور مختلف است.

- الکترونیک سنسور: این بخش شامل مدارها و قطعات الکترونیکی است که سیگنال الکتریکی تولید شده توسط المان حسگر را تقویت، فیلتر و پردازش می‌کند. این الکترونیک می‌تواند شامل تقویت کننده‌ها، فیلترها، آمپلی‌فایرها و قطعات مشابه باشد.

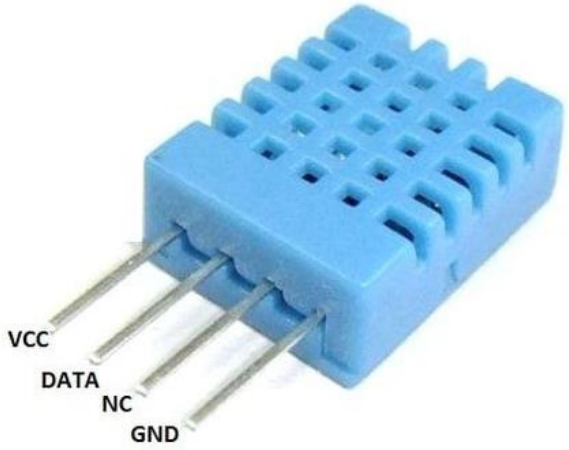
- رابط خروجی: این بخش مسئول انتقال سیگنال الکتریکی پردازش شده توسط الکترونیک سنسور به سیستم یا دستگاه استفاده‌کننده است. رابط خروجی ممکن است شامل پین‌ها، کانکتورها، رابط‌های سریال

- منبع تغذیه: سنسورها برای عملکرد نیاز به تغذیه الکتریکی دارند. منبع تغذیه شامل باتری، منبع تغذیه خارجی یا سیستم‌های تغذیه داخلی درون سنسور است.



# سنسور دما و رطوبت (DHT)

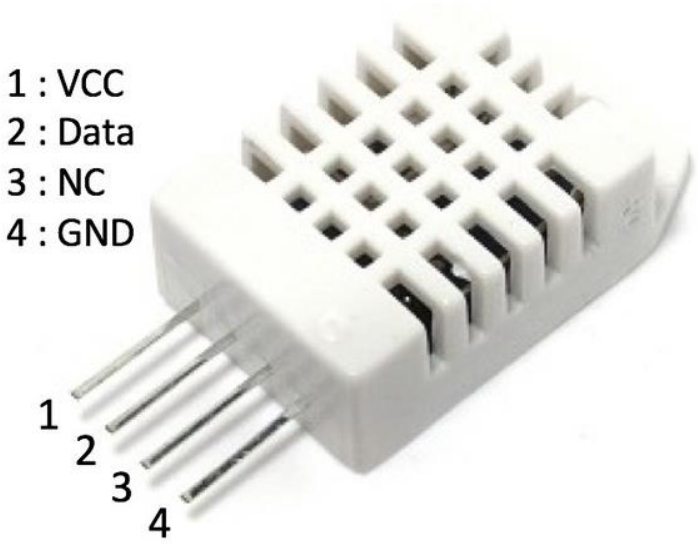
- این حسگرها بسیار ابتدایی هستند، اما برای علاقه‌مندانی که می‌خواهند اطلاعات اولیه را ثبت کنند عالی هستند. سنسورهای DHTاز دو قسمت یک حسگر رطوبت خازنی و یک ترمیستور ساخته شده اند . همچنین یک تراشه بسیار ابتدایی در داخل وجود دارد که مقداری تبدیل آنالوگ به دیجیتال را انجام می‌دهد و سیگنال دیجیتال را با دما و رطوبت بیرون می‌دهد. خواندن سیگنال دیجیتال با استفاده از هر میکروکنترلر نسبتاً آسان است.
- DHT11
  - هزینه فوق العاده کم
  - برق ۳ تا ۵ ولت و ورودی/خروجی
  - حداکثر استفاده از جریان ۲.۵ میلی آمپر در هنگام تبدیل (هنگام درخواست داده)
  - مناسب برای خوانش رطوبت ۲۰-۸۰٪ با دقت ۵٪
  - برای خوانش دمای ۰-۵۰ درجه سانتیگراد با دقت ۲ درجه سانتیگراد خوب است
  - نرخ نمونه برداری بیش از ۱ هرتز (هر ثانیه یک بار)
  - اندازه بدنه ۱۵.۵mm x 12mm x 5.5mm
  - 4پین با فاصله ۰.۱ اینچی



# سنسور دما و رطوبت (DHT)

## ❑ DHT22

- کم هزینه
- برق ۳ تا ۵ ولت و ورودی/خروجی
- حداکثر استفاده از جریان ۲.۵ میلی آمپر در هنگام تبدیل (هنگام درخواست داده)
- مناسب برای خوانش رطوبت ۰-۱۰۰٪ با دقت ۲-۵٪
- برای خوانش دمای -۴۰ تا ۸۰ درجه سانتیگراد با دقت ۰.۵ درجه سانتیگراد خوب است
- نرخ نمونه برداری بیش از ۰.۵ هرتز (هر ۲ ثانیه یک بار)
- اندازه بدنه ۱۵.۱mm x 25mm x 7.7mm
- 4پین با فاصله ۰.۱ اینچی




- پایه VCC : این پایه به ولتاژ مثبت تغذیه متصل می شود
- پایه DATA : این پایه خروجی مقادیر دما و رطوبت است که باید به میکروکنترلر متصل شود و همچنین با یک مقاومت به مثبت تغذیه نیز متصل می شود (پول آپ می شود)
- پایه NC : این پایه بدون استفاده است و لازم نیست به جایی متصل شود
- پایه GND : به منفی تغذیه متصل می شود

# ماژول حس‌گر دیجیتال سنجش شدت نور BH1750FVI

- ماژول BH1750یک برد مجهز به حس‌گر حساس به‌شدت نور است. این ماژول می‌تواند مستقیماً سیگنال دیجیتال در خروجی ایجاد کند. این ماژول برای تشخیص میزان نور محیط با دقت و رزولوشن بالا مناسب بوده و داده‌های خروجی آن به‌صورت لوکس متر است. همچنین این ماژول به‌راحتی به‌وسیله آردوینو قابل‌راه‌اندازی است.

ویژگی‌های ماژول :BH1750

- ارتباط I2C
- ولتاژ متغیر: ۴,۵ V~6Vیا ۳,۳
- محدوده وسیع و دقت بالا (۱ ~ ۶۵۵۳۵ lx)
- وابستگی کم به منبع نور (برای مثال لامپ رشته‌ای، هالوژن، LEDسفید و خورشیدی)
- نتیجه اندازه‌گیری قابل تنظیم با تأثیر پنجره نوری
- اندازه‌گیری تغییرات کم (۲۰٪+/-)
- سازگاری با آردوینو

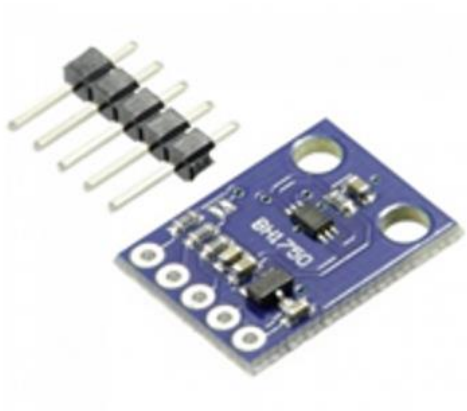
شب: ۰,۰۰۱~۰,۰۲

شب مهتابی: ۰,۰۲~۳۰۰

ابری در فضای بسته: ۵~۵۰

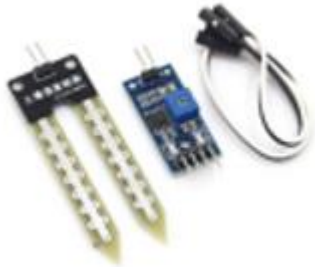
ابری در فضای باز: ۵۰~۸۰۰

آفتابی در فضای باز: ۱۰۰~۱۰۰۰



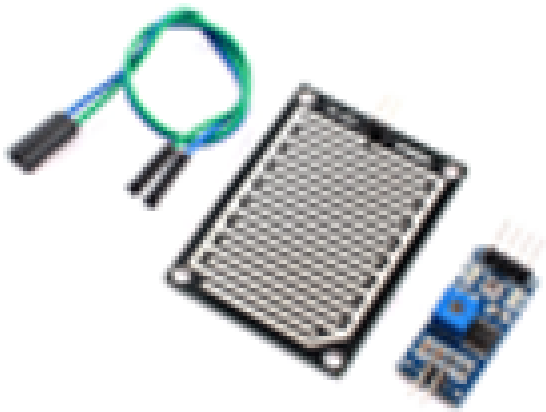
# حس‌گر رطوبت خاک YL-69

- حس‌گر رطوبت خاک ساخت YL-69با طراحی چنگال مانند خود به‌راحتی امکان داخل شدن در خاک را دارد. ولتاژ خروجی این حس‌گر با افزایش رطوبت خاک افزایش می‌یابد.
  - حساسیت قابل تنظیم از طریق پتانسیومتر آبی‌رنگ موجود بر روی برد
  - ولتاژ عملیاتی بین ۳.۳ تا ۵ ولت
  - دارای خروجی دیجیتال و آنالوگ
  - خروجی آنالوگ بین ۰ تا ۵ ولت
  - دمای نامی: ۲۰ درجه سانتی‌گراد
  - رطوبت: ۹۵٪ RH
  - طراحی فیزیکی مناسب برای نصب راحت و آسان
  - اندازه کوچک (۳ در ۱.۶ سانتیمتر)

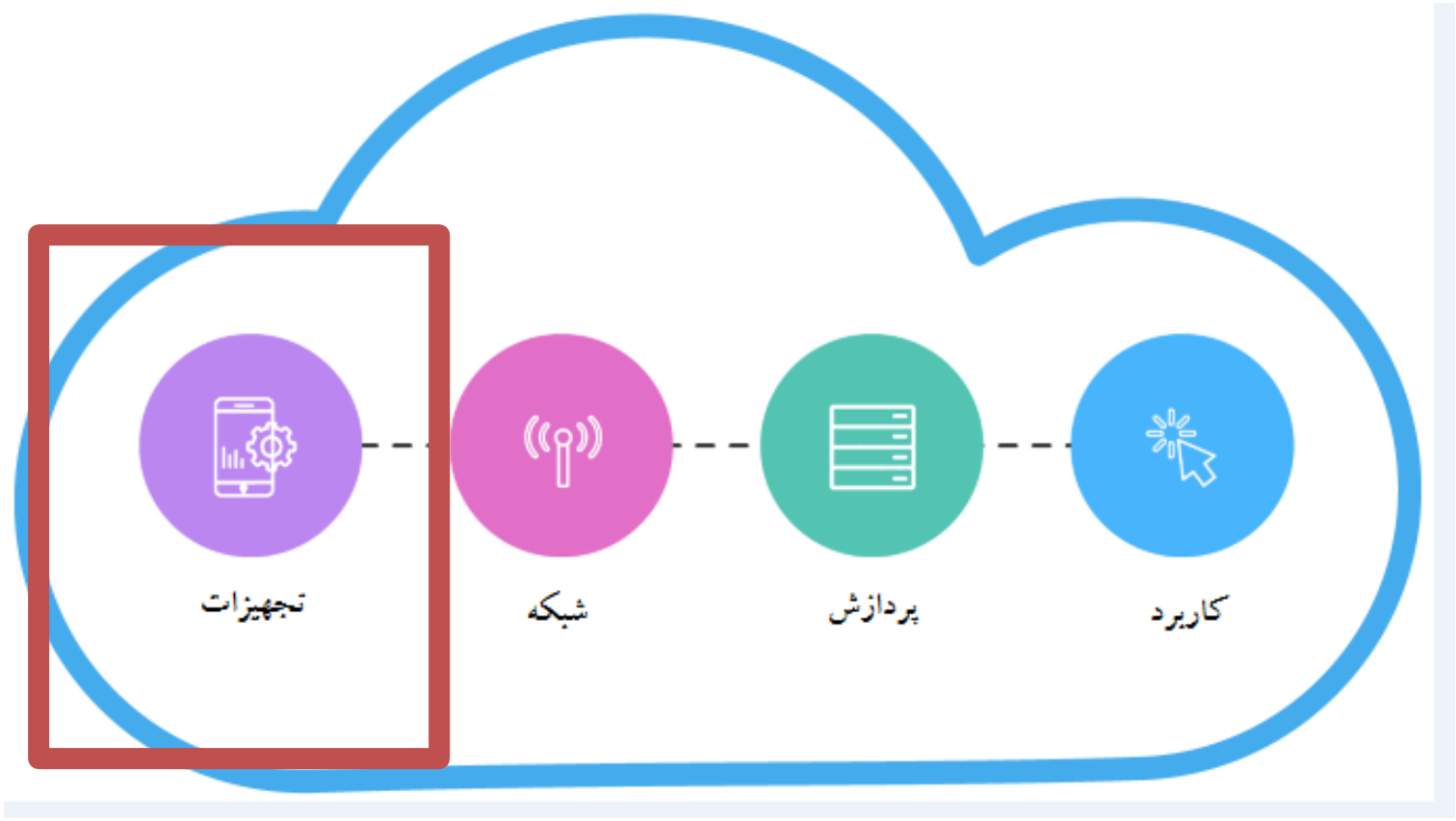


# حس‌گر اندازه‌گیری باران YL-83 RAIN

- ماژول حس‌گر تشخیص باران YL-83یک ابزار کاربردی و آسان برای تشخیص شدت باران است. لازم به ذکر که این ماژول از دو برد باران و کنترلر و سیم‌های رابط این دو برد تشکیل‌شده است؛ اما می‌توان به‌سادگی این دو برد را از یکدیگر جدا کرد. علاوه بر این می‌توان با کمک پتانسیومتر روی برد کنترلر، حساسیت خروجی دیجیتال ( (DOماژول را تنظیم کرد.
- مشخصات حس‌گر تشخیص باران:YL-83
  - دارای خروجی آنالوگ و دیجیتال به‌صورت مجزاست.
  - رله خروجی توانایی راه‌اندازی انواع موتور و آلارم DCیا ACتا جریان ۵ آمپر را دارا است.
  - حساسیت حس‌گر با استفاده از پتانسیومتر روی برد قابل تنظیم است
  - دارای دو LEDوضعیت برای نشان دادن اتصال تغذیه و تشخیص بارش باران



# لایه های اینترنت اشیا

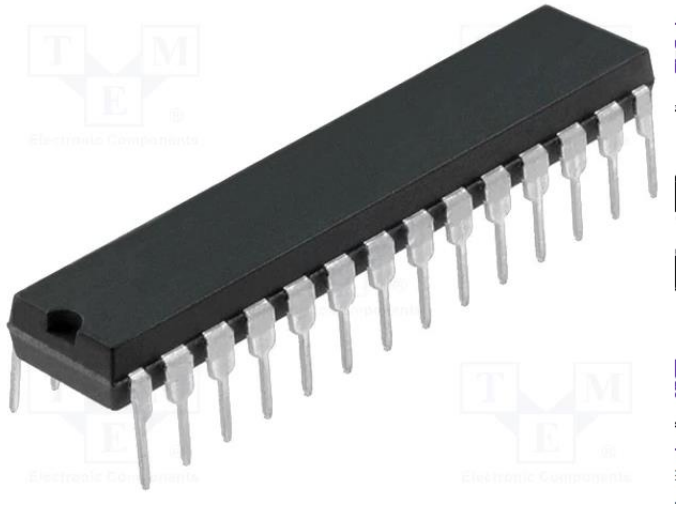


# میکروکنترلر

- میکروکنترلر یک نوع قطعه الکترونیکی است که تعبیه شده در سیستم‌ها و دستگاه‌های الکترونیکی استفاده می‌شود. این قطعه حاوی یک واحد پردازشگر (معمولاً یک پردازنده کوچک)، حافظه، وابستگی‌های ورودی/خروجی و سایر عناصر مورد نیاز برای کنترل و مدیریت سیستم است.

- میکروکنترلرها از طریق دریافت و پردازش سیگنال‌های ورودی و تولید سیگنال‌های خروجی، کارکرد و عملکرد دستگاه را کنترل می‌کنند. آنها به صورت یکپارچه، کوچک و با مصرف انرژی پایین طراحی شده‌اند و به عنوان "مغز" سیستم‌های الکترونیکی عمل می‌کنند.

- میکروکنترلرها در بسیاری از دستگاه‌ها مورد استفاده قرار می‌گیرند، از جمله تجهیزات خانگی (مثل لباسشویی، بخاری، ماکروویو و ... )، اسباب بازی‌های الکترونیکی، دستگاه‌های پزشکی، خودروها، تجهیزات صنعتی و بسیاری دیگر.

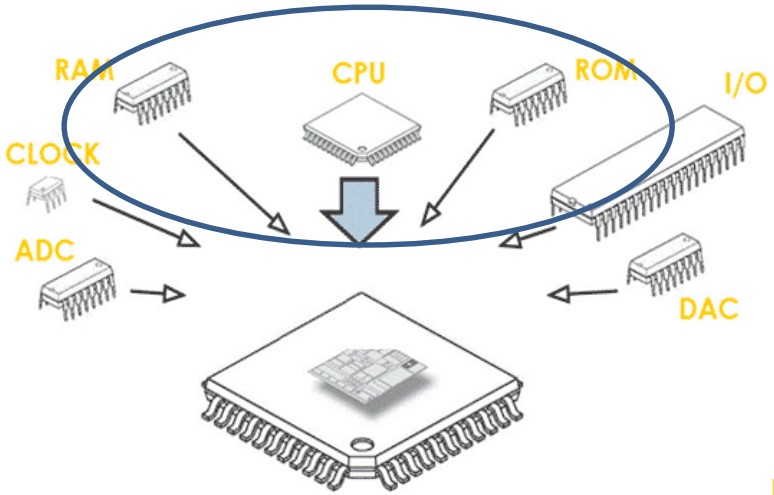



# اجزای میکروکنترلر

- یک میکروکنترلر شامل اجزای مختلفی است که به طور عمده برای کنترل و پردازش سیگنال‌ها و داده‌ها در سیستم‌های الکترونیکی استفاده می‌شود. اجزای اصلی یک میکروکنترلر عبارتند از:

- واحد پردازش مرکزی Central Processing Unit – CPU: این بخش مسئول اجرای عملیات‌ها و پردازش داده‌ها در میکروکنترلر است. این واحد شامل واحدهای حافظه، آرایه‌های رجیستر، واحد کنترل و واحد اجرا می‌شود.

- حافظه Memory حافظه در میکروکنترلر به منظور ذخیره سازی و بازیابی داده‌ها و برنامه‌ها استفاده می‌شود. حافظه در میکروکنترلر به صورت داخلی و خارجی وجود دارد.
  - ✓ حافظه داخلی شامل حافظه خواندن و نوشتن تصادفی Random Access Memory - RAM و حافظه فقط خواندن Read-Only Memory - ROM است.
  - ✓ حافظه خارجی می‌تواند شامل حافظه فلش یا حافظه اپراتورهای خارجی باشد که برای ذخیره‌سازی داده‌ها و برنامه‌ها استفاده می‌شود.



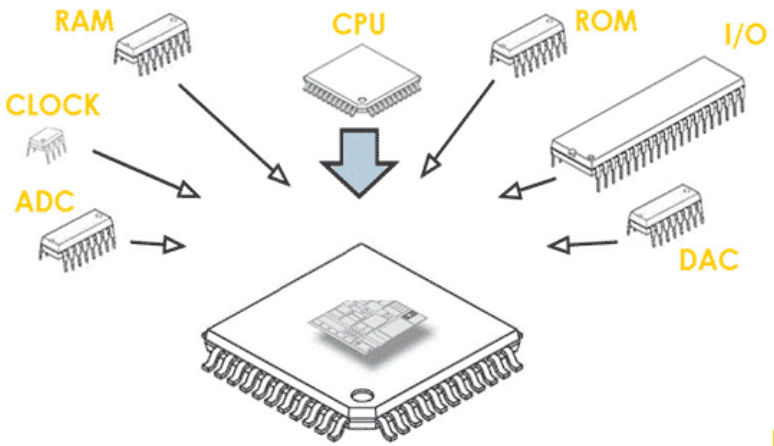
# اجزای میکروکنترلر

- رجیسترها Registers رجیسترها درون واحد پردازش مرکزی قرار دارند و برای ذخیره‌سازی داده‌های میانی و عملیات محاسباتی استفاده می‌شوند. رجیسترها سرعت بالا و دسترسی سریعی به داده‌ها را فراهم می‌کنند.

- واحد تایمر/شمارنده Timer/Counter این واحد برای شمارش زمان یا تعداد پالس‌ها در سیگنال‌های ورودی استفاده می‌شود. این واحد می‌تواند در برنامه‌های زمان‌بندی، کنترل فرآیندها و اندازه‌گیری‌های زمانی مورد استفاده قرار گیرد.

- پورت‌ها Ports پورت‌ها برای ارتباط با دستگاه‌های خارجی مورد استفاده قرار می‌گیرند. این پورت‌ها می‌توانند ورودی یا خروجی باشند و به عنوان رابط بین میکروکنترلر و سایر دستگاه‌ها عمل می‌کنند.

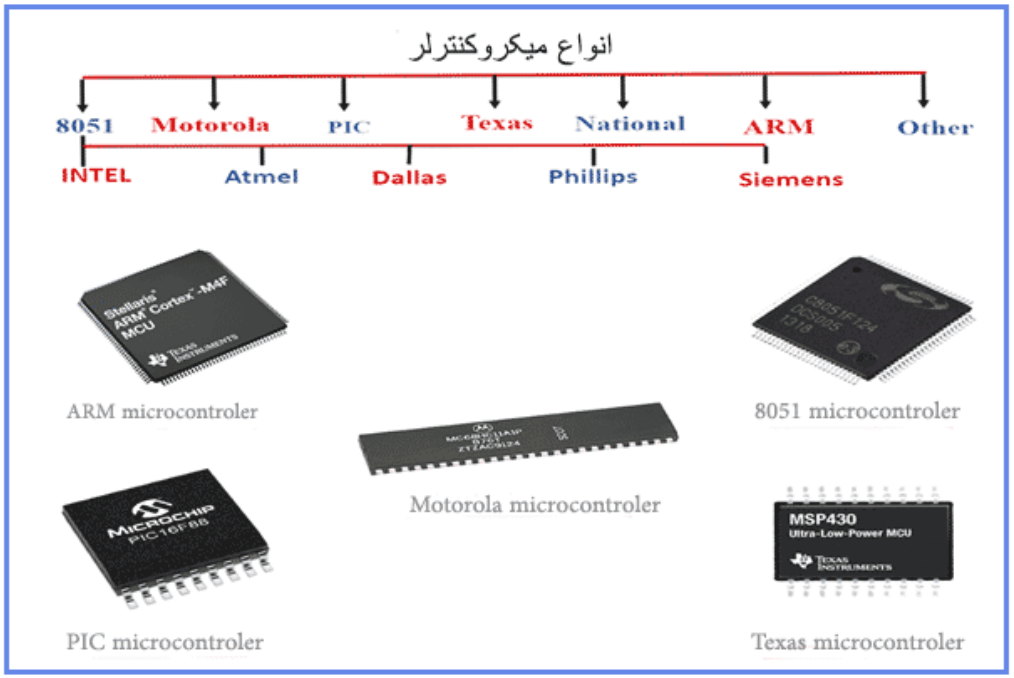
- مبدل آنالوگ به دیجیتال Analog-to-Digital Converter - ADC این واحد برای تبدیل سیگنال‌های آنالوگ به داده‌های دیجیتال استفاده می‌شود. این عملکرد بسیار مهم است زیرا بسیاری از سنسورها و ورودی‌های دستگاه‌ها سیگنال‌های آنالوگ را تولید می‌کنند



# انواع میکروکنترلر

- اغلب میکروکنترلرها ویژگی های مشترک زیادی دارند زیرا همه ی آنها دارای یک حافظه درایو، پایه های ورودی و خروجی و توان مصرفی کم هستند. اما در جزئیاتی مانند تعداد پایه ها ، ابعاد، قیمت تمام شده و ... نیز با هم متفاوت اند.

- شرکت های مختلفی اقدام به طراحی و تولید انواع میکروکنترلر ها بسته به نیاز مشتری نموده اند. از تولید کنندگان اصلی میکروکنترلر ها می توان به شرکت های ATMELو ARMاشاره نمود.




- تقسیم بندی میکروکنترلرها
  - اندازه بیت
  - حافظه
  - ساختار حافظه

# انواع میکروکنترلر

- تقسیم بندی میکروکنترلر ها بر اساس اندازه بیت
  - ۸ بیتی
  - ۱۶ بیتی
  - ۳۲ بیتی

- یک میکروکنترلر ۸ بیتی به معنای این است که اندازه داده‌ها و عملیاتی که میکروکنترلر می‌تواند انجام دهد، به صورت ۸ بیت است. بیت به معنای یک واحد اطلاعات دیجیتال است که می‌تواند دو حالت ۰ و ۱ را نشان دهد.

- در یک میکروکنترلر ۸ بیتی، طول داده‌ها و رجیسترها واحدهای ۸ بیتی است، به عبارت دیگر هر بار می‌توانید یک عدد ۸ بیتی (۲۵۶ حالت مختلف) را در آن ذخیره کنید

# انواع میکروکنترلر

- فرق بین یک میکروکنترلر ۸ بیتی و ۱۶ بیتی در اصل با نظریه اطلاعات و ظرفیت پردازش آن‌ها مرتبط است.

- دقت: یک میکروکنترلر ۱۶ بیتی نسبت به یک میکروکنترلر ۸ بیتی، قابلیت پردازش دقت بالاتری را دارد. این به این معنی است که میکروکنترلر ۱۶ بیتی می‌تواند با دقت بیشتری عملیات ریاضی پیچیده را انجام دهد و اعداد بزرگتری را نمایش دهد.
- حافظه: یک میکروکنترلر ۱۶ بیتی عموماً دارای حافظه بزرگتری نسبت به یک میکروکنترلر ۸ بیتی است. این به این معنی است که میکروکنترلر ۱۶ بیتی می‌تواند حجم بیشتری از داده‌ها و برنامه‌ها را ذخیره کند و به دسترسی به آنها با سرعت بیشتری دست پیدا کند.
- قابلیت پردازش: یک میکروکنترلر ۱۶ بیتی قابلیت پردازش بیشتری نسبت به یک میکروکنترلر ۸ بیتی دارد. این به این معنی است که میکروکنترلر ۱۶ بیتی می‌تواند عملیات‌های ریاضی پیچیده‌تر و عملکردهای پیشرفته‌تر را انجام دهد.
- سرعت: معمولاً میکروکنترلرهای ۸ بیتی سرعت عملیاتی بالاتری نسبت به میکروکنترلرهای ۱۶ بیتی دارند. این به معنی این است که میکروکنترلرهای ۸ بیتی به صورت عمومی در برنامه‌هایی که نیاز به سرعت بالای پردازش دارند، مورد استفاده قرار می‌گیرند.

# انواع میکروکنترلر

- انواع میکرو کنترلرها از نظر نوع کارکردی و مدار

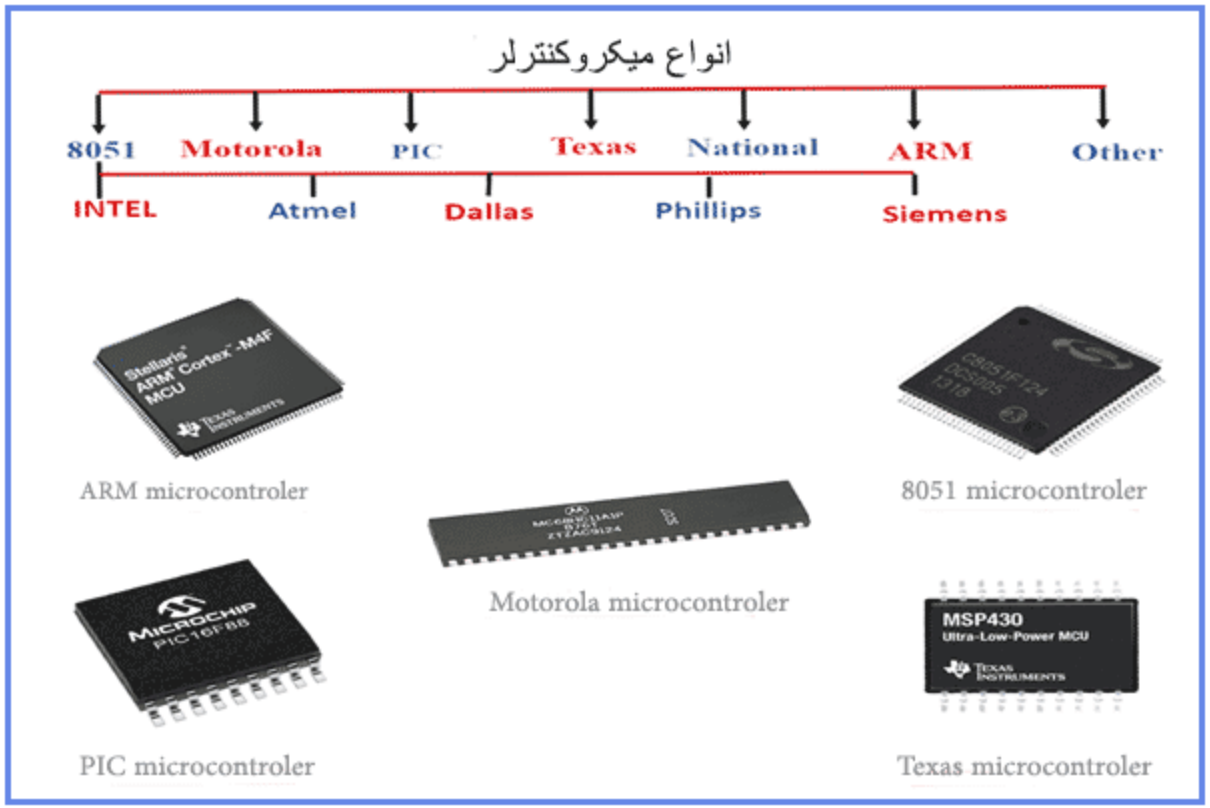
میکروکنترلرهای AVR

میکروکنترلرهای ARM

میکروکنترلر های x-mega

میکروکنترلرهای PIC

میکروکنترلرهای ۸۰۵۱



# انواع میکروکنترلر

## ❑میکروکنترلر AVR

- میکروکنترلر AVRکه با نام Advanced Virtual RISCنیز شناخته می شود، یک میکروکنترلر تراشه انفرادی ۸ بیتی RISCمعماری هاروارد است. در سال ۱۹۶۶ توسط Atmelاختراع شد

## ❑زبان برنامه نویسی میکروکنترلر AVR

- استفاده از دستور های برنامه نویسی اسمبلی یکی از راه های برنامه نویسی میکروکنترلر AVRبوده که به دلیل پیچیدگی، مورد استقبال برنامه نویسان قرار نگرفته است.

- استفاده از زبان های برنامه نویسی سطح بالا مانند C، JAVAو Basicکه هر خط کد از این برنامه ها معادل چند خط کد در زبان اسمبلی می باشد نیز برای برنامه نویسی میکروکنترلر AVRممکن می باشد.

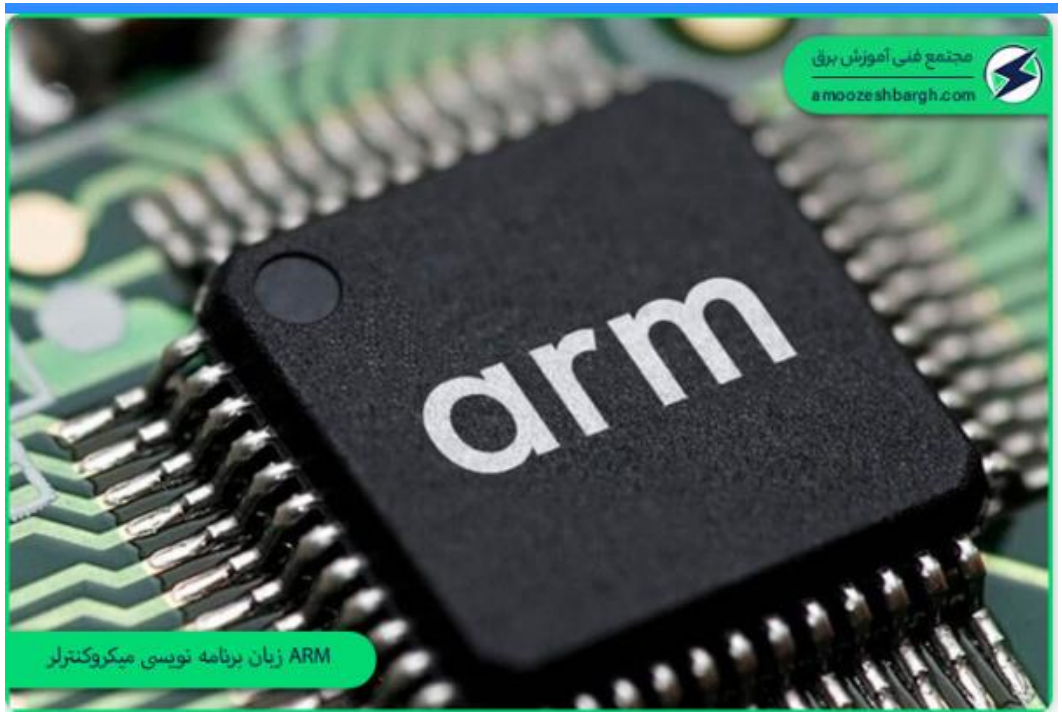
# انواع میکروکنترلر

## ❑میکروکنترلر ARM

- میکروکنترلر های ARMدر قطعات الکترونیکی ساده که به استفاده از CPUجدا نیاز ندارند مانند فلش مموری، دوربین فیلم برداری، کنترل تلویزیون و مودم ها مورد استفاده قرار می گیرند. میکروکنترلر های ARMاغلب در سیستم های نهفته و قابل حمل مانند تلفن های هوشمند،تبلت ها، لپ تاپ ها و ساعت های هوشمند کاربرد دارند.

## ❑زبان برنامه نویسی میکروکنترلر ARM

- میکروکنترلر ARMمانند دیگر میکروکنترلر ها از زبان ها سطح بالا مانند Cپشتیبانی می کنند.



# آردوینو

- برد های آردوینو چند سالی است که بسیار مورد توجه قرار گرفته اند. آردوینو در بین دانشجو ها ، دانش‌آموزان ، مهندسان و ... محبوب است و برای پروژه های متفاوت استفاده میشود.

- آردوینو در واقع میکروکنترلر نیست. بلکه آردوینو برد هایی هستند که از میکروکنترلرهای متفاوت به عنوان مغز استفاده میکنند و امکانات زیادی را در اختیار کاربران قرار میدهد.

- پلتفرم آردوینو به منظور تولید سریع و ساده پروژه‌های سخت افزاری تعاملی و ساخت وسایلی که با محیط تعامل داشته باشند طراحی شدهاست، البته بردهای آردوینو اهداف آموزشی را نیز دنبال میکنند.

# آردوینو

- اغلب برد های آردوینو دارای میکروکنترلر مرکزی AVRهستند و برخی از آن ها نیز از میکروکنترلر های ARMاستفاده میکنند.
- علاوه بر این، پلتفرم آردوینو یک نرم افزار IDE Arduinoدارد که به منظور برنامه‌نویسی برای بردهای آردوینو طراحی شده است
- IDE Arduinoامکان برنامه نویسی به زبان Cو C++را روی برد آردوینو فراهم میکند.

# آردوینو

- آردوینو مبتنی بر سخت افزار و نرم افزار است. برد های آردوینو می توانند ورودی ها را بخوانند (مثال اطلاعات سنسور ها و ).. و آن را به یک خروجی تبدیل می کنند مثال موتور را فعال کنند ، ال ای دی را روشن کنند یا ...

- شما با برنامه نویسی آردوینو که در در نرم افزار Arduinoانجام میشود میتوانید به آردوینو بگویید که چه کاری انجام دهد.

# آردوینو

- برد آردوینو UNOکه پرکاربردترین برد آردوینو و برد پایه آردوینو در اکثر آموزش های آردوینو است بر پایه میکروکنترلر ATmega328ساخته شدهاست.
- همچنین دارای رابط یو اس بی جهت بارگذاری برنامه و ارتباط با کامپیوتر است. ۶ پین ورودی آنالوگ و همچنین ۱۴ پین ورودی/خروجی دیجیتال دارد که شما را قادر می‌سازند تا برد آردوینو را به قطعات، سنسورها، بردها و ماژولهای دیگری متصل کنید.
- تعداد ورودی خروجیهای آنالوگ و دیجیتال در مدلهای مختلف بردهای آردوینو با توجه به میکروکنترلر اصلی استفاده شده بر روی برد متفاوت است

# دلایل استفاده از آردوینو

- سهولت استفاده: بردهای آردوینو برای کاربرانی که تازه وارد دنیای الکترونیک و برنامه‌نویسی هستند، بسیار مناسب هستند. زیرا آنها دارای یک محیط توسعه یکپارچه ( IDE)ساده و قابل فهم هستند که برنامه‌نویسی آنها را بسیار آسان می‌کند.
- ۲پشتیبانی گسترده: آردوینو دارای یک جامعه بزرگ و فعالی از کاربران و توسعه‌دهندگان است. این جامعه از طریق انجمن‌ها، وبسایت‌ها، ویدئوها و منابع آموزشی مختلف، به کاربران در حل مشکلات و یادگیری بیشتر در مورد آردوینو کمک می‌کند.
- قابلیت توسعه: بردهای آردوینو از پروتکل‌ها و رابط‌های مختلفی پشتیبانی می‌کنند. این قابلیت به کاربران اجازه می‌دهد با انواع سنسورها، ماژول‌ها و دستگاه‌های جانبی مختلف ارتباط برقرار کنند و پروژه‌های خود را گسترش دهند.
- هزینه مناسب: بردهای آردوینو به صورت عمومی قیمت مناسبی دارند و در دسترس برای بسیاری از افراد قرار دارند. این امر به کاربران اجازه می‌دهد تا با هزینه کمتری پروژه‌های الکترونیکی خود را شروع کنند.
- سازگاری: بردهای آردوینو با نرم‌افزارها و سخت‌افزارهای مختلف سازگاری بالایی دارند.

# تاریخچه آردوینو

- ایده در سال ۲۰۰۳ میلادی در انستیتو طراحی تعاملی ایورئا در کشور ایتالیا شکل گرفت.
- ایده عبارت بود از ساخت وسیله‌ای ساده و کم‌هزینه برای انجام پروژه‌های دیجیتال دانشجویان

- سه فرد کلیدی در به ثمر نشاندن این ایده نقش داشتند:
- فرناندو باراگان، ماسیمو بانزی، و کیسی ریس.
- باراگان یکی از دانشجویان انستیتو ایورئا بود که تصمیم گرفت پایان‌نامه کارشناسی‌ارشد خود را در این زمینه اجراء نماید.
- بانزی و ریس نیز استادان راهنمای پایان‌نامه باراگان بودند. تا آن زمان هنوز اسمی از آردوینو در میان نبود.

# تاریخچه آردوینو

- نتیجه پایان‌نامه باراگان بسیار موفقیت‌آمیز بود و منجر به ایجاد سخت‌افزار و نرم‌افزاری شد که وایرینگ نام گرفت.
- سخت‌افزار وایرینگ ویژگی های مورد نظر را نسبت به سایر نمونه‌های موجود در بازار آن زمان داشت یعنی ساده و کم‌هزینه بود.
- نرمافزار وایرینگ نیز بر مبنای یکی از زبانهای برنامه‌نویسی موجود به نام پراسسینگ تهیه شده بود.
- پس از اتمام پایان‌نامه، بانزی درصدد کاهش هزینه‌های سختافزار وایرینگ برآمد و در سال ۲۰۰۵ میلادی با همکاری دیوید کوآرتلس و دیوید ملیس (که بهترتیب کارمند و دانشجوی انستیتو ایورئا بودند)، به توسعه پروژه وایرینگ پرداخت و نام آن را به آردوینو تغییر داد. این نام جدید برگرفته از نام کافهای به نام آردوین در شهر ایورئا بود که اکثر جلسات گروه در آنجا تشکیل می‌شد.

# انواع بردهای آردوینو





| Characteristics | Arduino UNO | Arduino Nano | Arduino Mega | Arduino Due |
|---|---|---|---|---|
| Microcontroller | ATmega328 | ATmega328 | ATmega1280 | AT91SAM3X8E |
| Clock frequency | 16 MHz | 16 MHz | 16 MHz | 84 MHz |
| Analog inputs | 8 | 8 | 16 | 12 |
| Analog output | 0 | 0 | 0 | 2 |
| Digital input/output | 22 | 22 | 54 | 54 |
| Analog signals range | 0 to 5 V | 0 to 5 V | 0 to 5 V | 0 to 3.3 V |
| Analog signals resolution | 10 bit/4.88 mV | 10 bit/4.88 mV | 10 bit/4.88 mV | 12 bit/0.806 mV |

# آردوینو UNO

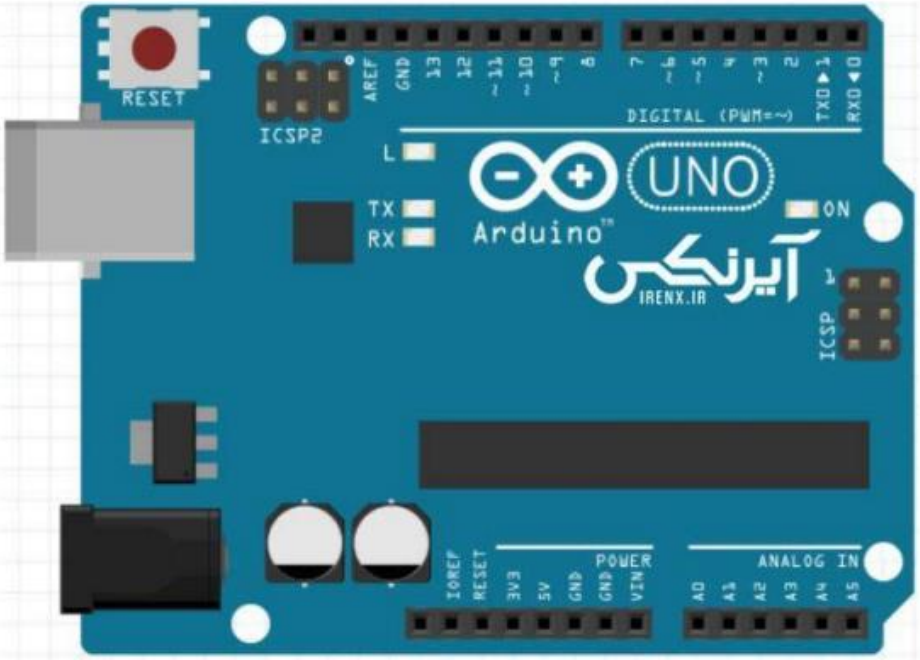
- Uno Arduinoیک برد مبتنی بر میکروکنترلر ATmega328Pاست.
- این برد مجهز به مجموعه ای از پین های ورودی / خروجی دیجیتال و آنالوگ است.
- بردUno Arduino دارای ۱۴ پین ورودی / خروجی دیجیتال است.
- بردUno Arduino دارای ۶ پین ورودی آنالوگ است.
- این برد با استفاده از نرم افزار IDE Arduinoاز طریق کابل USB نوع B قابل برنامه ریزی است.

# آردوینو UNO

- کلمه unoبه زبان ایتالیایی به معنی “یک” است و به منظور انتشار اولیه نرم افزار آردوینو انتخاب شده است.
- برد Uno اولین سری از سری برد های Arduino مبتنی بر USB است. میکروکنترلر ATmega328موجود در برد با یک بوت لودر از قبل برنامه ریزی شده است که اجازه می‌دهد بدون استفاده از پروگرامر سخت افزاری خارجی برنامه ریزی شود.
- برد آردوینو Unoرا میتوان از طریق کابل USBیا باتری ۹ ولتی خارجی تغذیه کرد.
- این برد ولتاژ بین ۷ تا ۲۰ را میتواند به عنوان ورودی استفاده کند زیرا دارای یک رگوالتور داخلی برای تبدیل ولتاژ است.

# آردوینو UNO

- یک ال ای دی داخلی است که به پین دیجیتال شماره ۱۳ متصل است. هنگامی که در کدLow باشد، ال ای دی خاموش باشد و هنگامی که High باشد ال ای دی روشن می‌شود
- VINولتاژ ورودی به برد آردوینو است که هنگامی که از یک منبع تغذیه خارجی استفاده کنیم، استفاده می‌شود. شما می‌توانید از این پین برای تغذیه استفاده کنید یا از USBو جک تغذیه.



# نرم افزار آردوینو

- لینک دانلود

https://d.irenx.ir/arduino-ide_2.1.0_irenx.ir.zip

# انواع داده در آردوینو

- نواع داده در آردوینو ، نوع داده در زبان Cبه منظور تعریف نوع متغیرها یا نوع توابع به کار می رود. نوع داده یک متغیر تعیین کننده میزان ظرفیت ذخیره سازی یک متغیر و چگونگی ذخیره سازی بیت ها در آن است.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| void | Boolean | char | Unsigned char | byte | int | Unsigned int | word |
| long | Unsigned long | short | float | double | array | String-char array | String-object |

# انواع داده در آردوینو

## Void

- کلیدواژه void فقط در تعریف توابع مورد استفاده قرار می‌گیرد. این کلمه نشان دهنده این است که تابع فراخوانی شده به تابعی که فراخوانی را انجام می دهد داده ای ارائه نمی دهد یا مقدار بازگشتی ندارد.

```
Void Loop ( )
Example
{
 // rest of the code
}
```

# انواع داده در آردوینو

## Boolean

- این نوع داد فقط میتواند مقدار صحیح یا غلط را نگه داری کند. هر متغیر از نوع Booleanیک بایت از حافظه را اشغال می کند.

```
Example
boolean val = false ;
boolean state = true ;
```

## Char

- این نوع داده یک بایت از حافظه را اشغال می کند و شامل یک کاراکتر است. کاراکترهای تکی برای ذخیره سازی باید به عنوان مثال به صورت A نوشته شوند و برای ذخیره سازی رشته ها به عنوان مثال ABC نوشته می شود. اگر چه باید توجه شود که کاراکترها نیز مطابق کد ASCIIبه صورت عدد ذخیره سازی می شوند.
- یک نکته در این مورد این است که میتوانیم روی کاراکترها عملیات حسابی (جمع، تفریق و...) انجام دهیم چون به صورت عدد ذخیره می شوند. به عنوان مثال میتوانیم بنویسیم ' A'+1که معادل ۶۶ است زیرا کد اسکی معادل Aبرابر ۶۵ است.

# انواع داده در آردوینو

Ascii Chart

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NUL 0 | SOH 1 | STX 2 | ETX 3 | EOT 4 | ENQ 5 | ACK 6 | BEL 7 | BS 8 | HT 9 | LF 10 | VT 11 | FF 12 | CR 13 | SO 14 | SI 15 |
| 1 | DLE 16 | DC1 17 | DC2 18 | DC3 19 | DC4 20 | NAK 21 | SYN 22 | ETB 23 | CAN 24 | EM 25 | SUB 26 | ESC 27 | FS 28 | GS 29 | RS 30 | US 31 |
| 2 | SPC 32 | ! 33 | " 34 | # 35 | $ 36 | % 37 | & 38 | ' 39 | ( 40 | ) 41 | * 42 | + 43 | , 44 | - 45 | . 46 | / 47 |
| 3 | 0 48 | 1 49 | 2 50 | 3 51 | 4 52 | 5 53 | 6 54 | 7 55 | 8 56 | 9 57 | : 58 | ; 59 | < 60 | = 61 | > 62 | ? 63 |
| 4 | @ 64 | A 65 | B 66 | C 67 | D 68 | E 69 | F 70 | G 71 | H 72 | I 73 | J 74 | K 75 | L 76 | M 77 | N 78 | O 79 |
| 5 | P 80 | Q 81 | R 82 | S 83 | T 84 | U 85 | V 86 | W 87 | X 88 | Y 89 | Z 90 | [ 91 | \ 92 | ] 93 | ^ 94 | _ 95 |
| 6 | ` 96 | a 97 | b 98 | c 99 | d 100 | e 101 | f 102 | g 103 | h 104 | i 105 | j 106 | k 107 | l 108 | m 109 | n 110 | o 111 |
| 7 | p 112 | q 113 | r 114 | s 115 | t 116 | u 117 | v 118 | w 119 | x 120 | y 121 | z 122 | { 123 | \| 124 | } 125 | ~ 126 | DEL 127 |
| 8 | € 128 | 129 | ‚ 130 | ƒ 131 | „ 132 | … 133 | † 134 | ‡ 135 | ˆ 136 | ‰ 137 | Š 138 | ‹ 139 | Œ 140 | 141 | Ž 142 | 143 |
| 9 | 144 | ‘ 145 | ’ 146 | “ 147 | ” 148 | • 149 | – 150 | — 151 | ˜ 152 | ™ 153 | š 154 | › 155 | œ 156 | 157 | ž 158 | Ÿ 159 |
| A | 160 | ¡ 161 | ¢ 162 | £ 163 | ¤ 164 | ¥ 165 | ¦ 166 | § 167 | ¨ 168 | © 169 | ª 170 | « 171 | ¬ 172 | - 173 | ® 174 | ¯ 175 |
| B | ° 176 | ± 177 | ² 178 | ³ 179 | ´ 180 | µ 181 | ¶ 182 | · 183 | ¸ 184 | ¹ 185 | º 186 | » 187 | ¼ 188 | ½ 189 | ¾ 190 | ¿ 191 |
| C | À 192 | Á 193 | Â 194 | Ã 195 | Ä 196 | Å 197 | Æ 198 | Ç 199 | È 200 | É 201 | Ê 202 | Ë 203 | Ì 204 | Í 205 | Î 206 | Ï 207 |
| D | Ð 208 | Ñ 209 | Ò 210 | Ó 211 | Ô 212 | Õ 213 | Ö 214 | × 215 | Ø 216 | Ù 217 | Ú 218 | Û 219 | Ü 220 | Ý 221 | Þ 222 | ß 223 |
| E | à 224 | á 225 | â 226 | ã 227 | ä 228 | å 229 | æ 230 | ç 231 | è 232 | é 233 | ê 234 | ë 235 | ì 236 | í 237 | î 238 | ï 239 |
| F | ð 240 | ñ 241 | ò 242 | ó 243 | ô 244 | õ 245 | ö 246 | ÷ 247 | ø 248 | ù 249 | ú 250 | û 251 | ü 252 | ý 253 | þ 254 | ÿ 255 |

# انواع داده در آردوینو

## Unsigned char

این نوع از داده یک بایت از حافظه را اشغال می کند و میتواند از عدد صفر تا ۲۵۵ را ذخیره سازی کند.

```
Example
Unsigned Char chr_y = 121 ;
```

## Byte

این نوع داده نیز ۸ بیت را اشغال کرده و از عدد صفر تا ۲۵۵ را ذخیره سازی می کند.

```
byte m = 25 ;
```

## Int

از این نوع داده برای ذخیره سازی اعداد صحیح استفاده می شود. نوع intیک عدد دو بایتی (۱۶ بیت) را ذخیره سازی می کند که معادل است با ۳۲۷۶۸ – تا ۳۲۷۶۷

**ظرفیت نوع intممکن است در بردهای مختلف متفاوت باشد.** به عنوان مثال در Arduino Dueاین نوع از دادع معادل ۳۲ بیت یا ۴ بایت است که معادل ذخیره سازی از عدد ۲،۱۴۷،۴۸۳،۶۴۸- تا ۲،۱۴۷،۴۸۳،۶۴۷ می باشد.

# انواع داده در آردوینو

**Unsigned int**

این نوع از داده مشابه نوع intاست با این تفاوت که فقط اعداد صحیح مثبت را ذخیره سازی می کند که شامل صفر تا ۶۵،۵۳۵ می شود. در برد Dueاین محدوده ۴ بایت است و از صفر تا ۴،۲۹۴،۹۶۷،۲۹۵ (۳۲^۲ – ۱) را شامل می شود.

**Long**

این نوع داده شامل ۳۲ بیت از عدد ۲،۱۴۷،۴۸۳،۶۴۸- تا ۲،۱۴۷،۴۸۳،۶۴۷ می شود.

**Unsigned long**

این نوع داده شامل ۳۲ بیت بوده و برخلاف نوع استاندارد ( longاعداد منفی را ذخیره سازی نمی کند و محدوده ذخیره سازی آن از صفر تا ۴،۲۹۴،۹۶۷،۲۹۵ می باشد.

# انواع داده در آردوینو

**Short**

این نوع داده ۱۶ بیتی است. در تمام بردهای آردوینو (مبتنی بر ATMegaو ARM) این نوع داده ۱۶ بیتی (۲ بایتی) است که شامل محدوده –۳۲،۷۶۸ تا ۳۲۷۸۸ می باشد.

**Float**

این نوع برای ذخیره سازی اعداد اعشار استفاده می شود. از نوع اعشار معمولا برای ذخیره سازی مقادیر آنالوگ و پیوسته استفاده می شود زیرا دارای دقت بالاتری نسبت به نوع صحیح می باشد. محدوده اعداد این نوع عبارت است از ۳,۴۰۲۸۲۳۵E+38 - تا مقدار ۳,۴۰۲۸۲۳۵E+38 و در حافظه ۴ بایت ذخیره سازی می شود.

**Double**

در برد Unoو دیگر بردهای مبتنی برد میکروهای ATMEGA، این نوع دارای ظرفیت ۲ برابر نسبت به floatاست و ۴ بایت را اشغال می کند ولی دقت آن مشابه floatاست. در برد Arduino Dueاین نوع دارای ۸ بایت (۶۴ بیت) دقت می باشد.

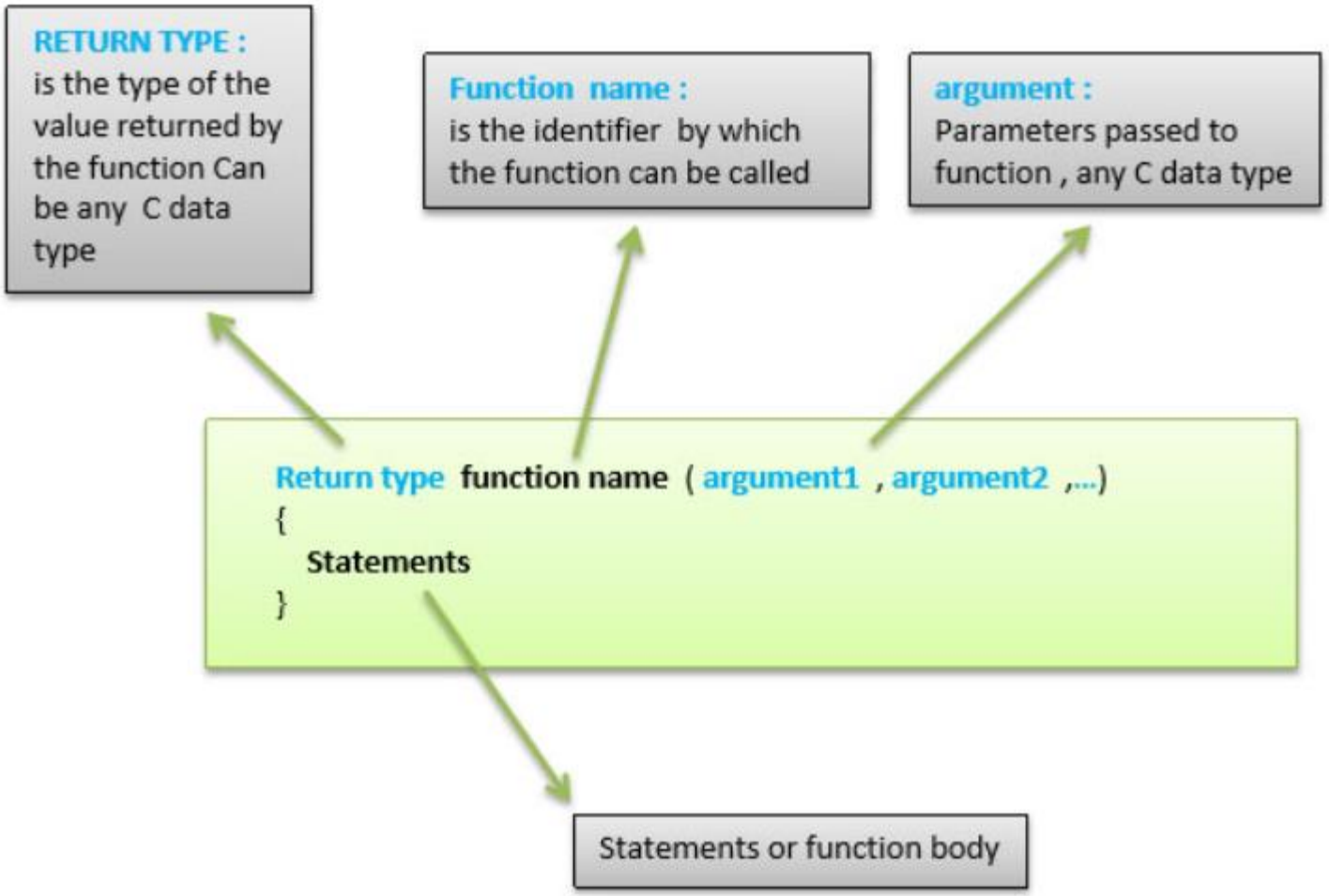
# توابع مورد استفاده در آردوینو

توابع به ما اجازه می دهند که برنامه را به شکلی ساختارمند به بخش های مختلفی تقسیم بندی کنیم که هر بخش وظایفی مشخص را اجرا کنند. دلیل عمده استفاده از توابع استفاده از یک قطعه کد به تعداد زیاد در برنامه های دیگر می باشد.

- void setup()
- void loop()
- pinMode(پین, OUTPUT);
- digitalWrite(پین,HIGH);
- delay(1000);
- digitalRead(پین);
- if(شرط)
- else

- #include <LiquidCrystal.h>
- LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
- lcd.begin(16, 2);
- lcd.setCursor(ستون, ردیف);
- lcd.print("متن مورد نظر");
- lcd.clear();

# توابع مورد استفاده در آردوینو



# توابع مورد استفاده در آردوینو

در یک برنامه آردوینو دو تابع ضروری است : تابع setup()و تابع loop() بقیه توابع در خارج از محدوده این دو تابع تعریف می‌شوند.

```
int sum_func (int x, int y) // function declaration {
 int z = 0;
 z = x+y ;
 return z; // return the value
}

void setup () {
 Statements // group of statements
}

Void loop () {
 int result = 0 ;
 result = Sum_func (5,6) ; // function call
}
```

# توابع مورد استفاده در آردوینو

Setup()

این تابع setup() با شروع یک پروژه فراخوانی می شود. از آن برای مقداردهی اولیه متغیرها، حالت‌های پین، شروع استفاده از کتابخانه‌ها و غیره استفاده کنید. این تابع setup() فقط یک بار، پس از هر بار روشن کردن یا تنظیم مجدد برد آردوینو اجرا می‌شود.

```
int buttonPin = 3;

void setup() {
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}

void loop() {
  // ...
}
```

# توابع مورد استفاده در آردوینو

loop()

پس از ایجاد یک تابع setup()، که مقادیر اولیه را مقداردهی اولیه و تنظیم می کند، loop()تابع دقیقاً همان کاری را انجام می دهد که نامش نشان می دهد و به طور متوالی حلقه می زند و به برنامه شما اجازه می دهد تغییر کند و پاسخ دهد. از آن برای کنترل فعال برد آردوینو استفاده کنید.

```
// loop checks the button pin each time,
// and will send serial if it is pressed
void loop() {
  if (digitalRead(buttonPin) == HIGH) {
    Serial.write('H');
  }
  else {
    Serial.write('L');
  }

  delay(1000);
}
```

# توابع مورد استفاده در آردوینو

pinMode()

pinMode()در آردوینو پین مشخص‌شده را تنظیم می‌کند که به‌عنوان ورودی input یا خروجی outputعمل کند.

pinMode(pin, mode)

```
int ledPin = 13;
void setup()
{
  pinMode(ledPin, OUTPUT);
// پین دیجیتال را به‌عنوان خروجی تنظیم (ست) می‌کند
}
```

# توابع مورد استفاده در آردوینو

digitalWrite()

اگر پین به عنوان OUTPUT با pinMode()پیکربندی شده باشد، ولتاژ آن روی مقدار مربوطه تنظیم می شود: ۵ ولت (یا ۳.۳ ولت در بردهای ۳.۳ ولت) برای HIGH، 0ولت (زمین) برای Low

digitalRead ()

در آردوینو مقدار یک پین دیجیتال مشخص‌شده، که HIGHیا LOWاست را می‌خواند.

digitalRead(pin)

**پارامترها**

pin:عدد پین دیجیتالی که می‌خواهید بخوانید .int

**خروجی**

HIGHیا LOW

# توابع مورد استفاده در آردوینو

digitalRead

```
//  ال ای دی به پین دیجیتال شماره‌ی سیزده متصل شده
int inPin = 7;
//کلید فشاری (پوش باتن ) به پین دیجیتال شماره‌ی هفت متصل شده
int val = 0;
// متغیری برای ذخیره مقدار خوانده‌شده

void setup()
{
  pinMode(ledPin, OUTPUT);
  //پین دیجیتال شماره‌ی سیزده را به‌عنوان خروجی تنظیم می‌کند
  pinMode(inPin, INPUT);
 // پین دیجیتال شماره‌ی هفت را به‌عنوان ورودی تنظیم می‌کند
}

void loop()
{
  val = digitalRead(inPin);
// پین ورودی (این پوت) را می‌خواند
  digitalWrite(ledPin, val);
 // ال ای دی را به مقدار دکمه تنظیم می‌کند
}
```

# توابع مورد استفاده در آردوینو

## دستور analogRead در آردوینو تبدیل ADC

این دستور مقدار ولتاژ را در پین های آنالوگ ورودی میخواند و آنها را به عدد تبدیل میکند.

این دستور برای تبدیل آنالوگ به دیجیتال در برد آردوینو استفاده میشود. برد های آردوینو دارای مبدل های آنالوگ به دیجیتال (ADC) چند کاناله هستند. اکثر برد های آردوینو دارای مبدل ۱۰ بیتی هستند و ولتاژ عملیاتی آنها ۵ ولت است. این بدان معنا است که ولتاژ ۰ تا ۵ را به عددی بین ۰ تا ۱۰۲۳ تبدیل میکنند.

```
int analogPin = A3; // پایه میانی پتانسیومتر به این پین متصل میشود.
int val = 0;  // ایجاد متغیر برای قرار دادن مقدار تبدیل شده

void setup() {
  Serial.begin(9600);           //  شروع ارتباط سریال
}

void loop() {
  val = analogRead(analogPin);  // خواندن مقدار در پین ورودی آنالوگ
  Serial.println(val);          // نمایش مقدار بدست آمده در سریال مانیتور
}
```

# توابع مورد استفاده در آردوینو

**دستور analogWrite در آردوینو**

با این دستور مقدار آنالوگ مورد نظر خود را روی یک پین ایجاد میکنیم. از این دستور برای روشن کردن یک LEDدر شدت روشنایی های مختلف، حرکت موتور با سرعت متفاوت و ... استفاده میشود.

```
for(int i = ۰; i<۲۵۶; i++)  // نوشتن دستور for
{
  analogWrite(led, i); // استفاده از آنالوگ برای افزایش و کاهش پایه
  delay(delaytime);
  Serial.println(i);
}
```

# توابع زمانی مورد استفاده در آردوینو

**در آردوینو ۴ تابع برای بحث زمان وجود دارد که در ادامه در مورد آنها و کاربرد هریک توضیح خواهیم داد.**

delay استفاده از این تابع بسیار ساده است. کافی است یک عدد صحیح به عنوان ورودی به تابع داده شود که در واقع میزان تاخیر برحسب میلی ثانیه خواهد بود.

۲- delayMicroseconds : استفاده از این تابع نیز مشابه delay()است. کافی است یک عدد صحیح به عنوان ورودی به تابع داده شود که در واقع میزان تاخیر برحسب میکرو ثانیه خواهد بود.

۳- milis از این تابع برای محاسبه مدت زمان گذشته شده از زمان اجرای برنامه برحسب میلی ثانیه استفاده می شود.

۴- microsاز این تابع برای محاسبه مدت زمان گذشته شده از زمان اجرای برنامه برحسب میکرو ثانیه استفاده می شود. نکته مهمی که باید توجه داشته باشید این است که این تابع در صورتی که مدت زمان از ۷۰ دقیقه بگذرد سرریز کرده و مجددا به صفر بازمیگردد.

# توابع زمانی مورد استفاده در آردوینو

**تابع : delay**

همان طور که پیش تر گفته شد استفاده از تابع delayبسیار ساده است. کافی است یک عدد یا متغیر صحیح به ورودی تابع برحسب میلی ثانیه بدهید. برنامه بر روی این تابع تا مدت زمان تعیین شده توسط این عدد منتظر می ماند و پس از گذر زمان تعیین شده خطوط بعدی برنامه اجرا خواهد شد. البته در نظر داشته باشید که استفاده از این تابع برای ایجاد تاخیر گزینه مناسبی نیست زیرا یک عامل ایجاد کننده مانع در برابر فرایند اجرای برنامه است.

```
delay (ms) ;
```

# ساختارهای کنترلی مورد استفاده در آردوینو

**شرط نویسی برای سنسورها در واقع مسیر به آن ها میدهد. دستورات برنامه نویسی مهمی مثل if else, switch case, for از جمله کاربردی ترین ساختارهای کنترلی است.**

**ساختار تکرار for:**

تصور کنید میخواهید ۵۰ عدد را از ورودی دریافت و با استفاده از دستورات، یک عمل خاصی مانند جمع یا تفریق را روی آن ها انجام دهیم! یعنی ۵۰ بار باید تکرار کنیم! پس اصلا منطقی نیست! برای اینکار میتوانیم از ساختار تکرار for استفاده کنیم. ساختار تکرار for برای ساخت حلقه است و معمولا در حالتی که تعداد دفعات تکرار حلقه از قبل مشخص باشد، از آن استفاده می‌کنیم.

```
For    ( مقدار اولیه;شرط حلقه; گام حرکت ) {

نوشتن دستور اول;

نوشتن دستور دوم;

نوشتن دستورn;

}
```

```
for  (i = ۱; i<۱۰; i++)
```

- متغیر شمارنده تعداد دفعات تکرار حلقه را کنترل می‌کند.
- مقدار اولیه در برنامه در هر بار اجرای دستورات ، مقداری به آن اضافه می‌شود.
- این افزایش مقدار توسط گام حرکت مشخص می‌شود.
- گام حرکت میتواند عدد مثبت، منفی و اعشاری باشد.
- شرط حلقه مشخص می‌کند که دستورات داخل حلقه تا چه زمانی باید اجرا شود.
- اگر شرطی که مینویسیم دارای ارزش درستی True باشد، دستورات اجرا می‌شود.
- در غیر اینصورت کنترل برنامه از حلقه تکرار خارج می‌شود.

# ساختارهای کنترلی مورد استفاده در آردوینو

**شرط نویسی برای سنسورها در واقع مسیر به آن ها میدهد. دستورات برنامه نویسی مهمی مثل if else, switch case, for از جمله کاربردی ترین ساختارهای کنترلی است.**

```
if (condition) {
Block of statements;
}
```

**ساختار شرطی if:**

دستور شرطی if از مهمترین دستورات کنترلی می باشد و شامل دستور یا دستورات شرطی داخل پرانتزمی باشد. اگر دستورات شرطی داخل پرانتز برابر با True شد (درست بود) عبارات یا دستورات درون بلوک شرط اجرا خواهند شد و در صورتی که شرط برقرار نباشد از جرای دستورات دخل بلوک شرط صرف نظر خواهد شد.

دستور شرطی if را به دو صورت می توان نوشت:

در صورتی که با درسط بودن شرط بخواهیم فقط یک عبارت یا دستور اجرا شود به صورت زیر استفاده می کنیم.

```
Void loop () {
/* check the boolean condition */
if (A > B) /* if condition is true then execute the following statement*/
A++;
/* check the boolean condition */
If ( ( A < B ) && ( B != 0 )) /* if condition is true then execute the following statement*/ {
A += B;
B--;
}
```

# ساختارهای کنترلی مورد استفاده در آردوینو

**ساختار if / else در آردوینو:**

**ساختار if / else در آردوینو به نسبت if ساده، با چک کردن چند شرطِ همراه با هم، قدرت بیشتری در کنترل جریان کد به ما می‌دهد. برای مثال در چک کردن یک ورودی آنالوگ، هنگامی که ورودی کمتر از ۵۰۰ است،کار خاصی قابل انجام است و کار دیگر، هنگامی صورت می گیرد که ورودی بیشتر یا مساوی ۵۰۰ باشد. کد این مثال به شکل زیر خواهد بود:**

```
if (pinFiveInput < 500)
{
  // action A
}
else
{
  // action B
}
```

```
if (pinFiveInput < 500)
{
  // do Thing A
}
else if (pinFiveInput >= 1000)
{
  // do Thing B
}
else
{
  // do Thing C
}
```

# ساختارهای کنترلی مورد استفاده در آردوینو

**switch case در آردوینو:**

**switch case در آردوینو همانند ساختار if، جریان برنامه را کنترل می‌کند. ساختار سوییچ این کار را با امکان مشخص کردن کدهای متفاوتی که باید در شرایط مختلف اجرا شوند، انجام می‌دهد. به‌طور خاص، ساختار سوییچ، مقدار متغیر را با مقداری که در قسمت کیس مشخص شده، مقایسه می‌کند. وقتی‌که مقدار بررسی‌شده یک کیس با مقدار متغیر برابر شد؛ کد آن کیس اجرا می‌شود.**

```
switch (var) {
    case 1:
      //do something when var equals 1
      break;
    case 2:
      //do something when var equals 2
      break;
    default:
      // if nothing else matches, do the default
      // default is optional
    break;
  }
```

```
switch (var) {
    case 1:
      {
      //do something when var equals 1
      int a = 0;
      .......
      .......
      }
      break;
    default:
      // if nothing else matches, do the default
      // default is optional
    break;
  }
```

# ساختارهای کنترلی مورد استفاده در آردوینو

**حلقه while و do while در برنامه نویسی آردوینو**

**حلقه whileبه طور مداوم و بی نهایت تکرار میشود تا زمانی که عبارت داخل پرانتز () نادرست شود. چیزی باید تغییر کند تا حلقه متوقف شود، در غیر این صورت حلقه whileدائما تکرار میشود. چیزی که تغییر میکند میتواند یک متغیر باشد که افزایش یافته یا یک شرایط خارجی (مثل سنسور)**

**دقت کنید که شرط درون پرانتز هر وقت صحیح باشد حلقه اجرا میشود و با نادرست شدن شرط، حلقه متوقف میشود.**

```
while (condition) { // شرط در پرانتز نوشته میشود
  // کد هایی که میخواهید اجرا شوند
}
---------------------------------------- مثال
var = 0;
while (var < 200) { // تا زمانی که متغیر کمتر از 200 باشد حلقه اجرا میشود
var++; // اضافه کردن به متغیر
// اجرای حلقه 200 بار
}
```

# ساختارهای کنترلی مورد استفاده در آردوینو

**حلقه while و do while در برنامه نویسی آردوینو**

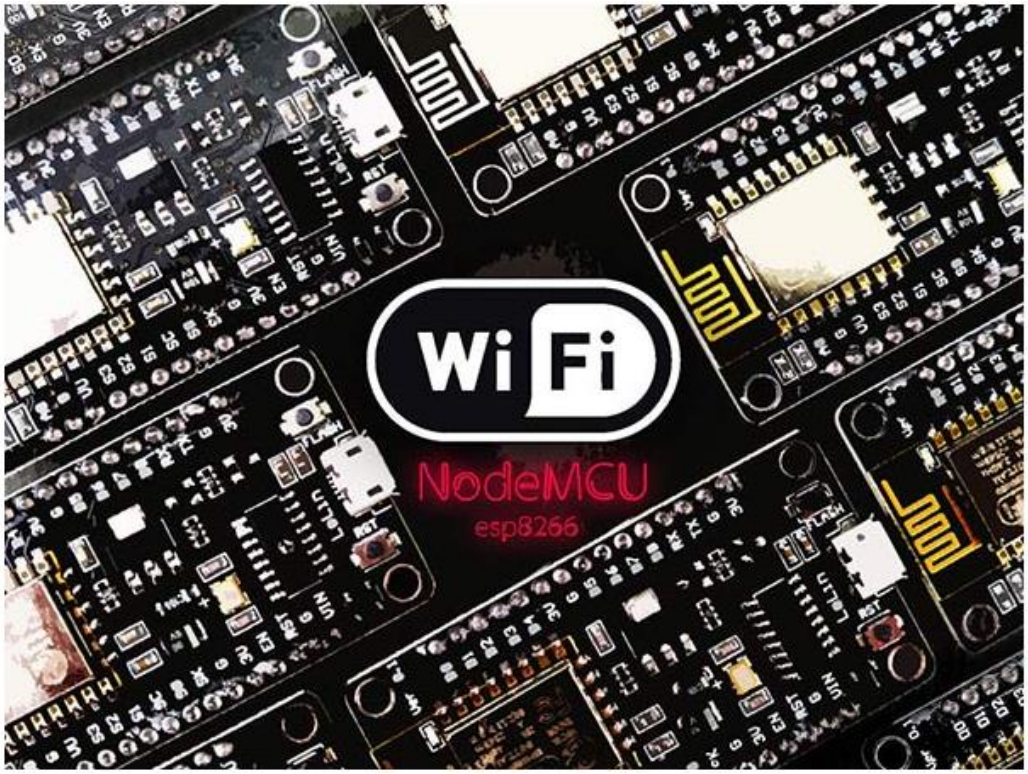
**حلقه do... while به همان روال حلقه while کار می کند ، با این تفاوت که شرط در انتهای حلقه آزمایش می شود ، بنابراین حلقه do همیشه حداقل یک بار اجرا می شود.**

```
do {
  // کد هایی که میخواهید اجرا کنید
} while (condition);
---------- مثال
int x = 0; // ایجاد یک متغیر
do {
delay(50); // انتظار برای چک شدن سنسور
x = readSensors(); // ریختن مقدار سنسور در متغیر ایکس
} while (x < 100); // اگر مقدار سنسور زیر 100 باشد دوباره حلقه اجرا میشود
```

# برد NodeMCU ESP8266

# NodeMCU

**Nodemcuیک پلتفروم متن باز مبتنی بر ESP8266می باشد که قابلیت اتصال اشیا بهم و انتقال اطلاعات از طریق Wifiرا فراهم می سازد بعلاوه با فراهم آوردن برخی کاربردهای مهم میکروکنترلرها از جمله ADC و ... می تواند به تنهایی بسیاری از نیازهای پروژه را برطرف سازد.**



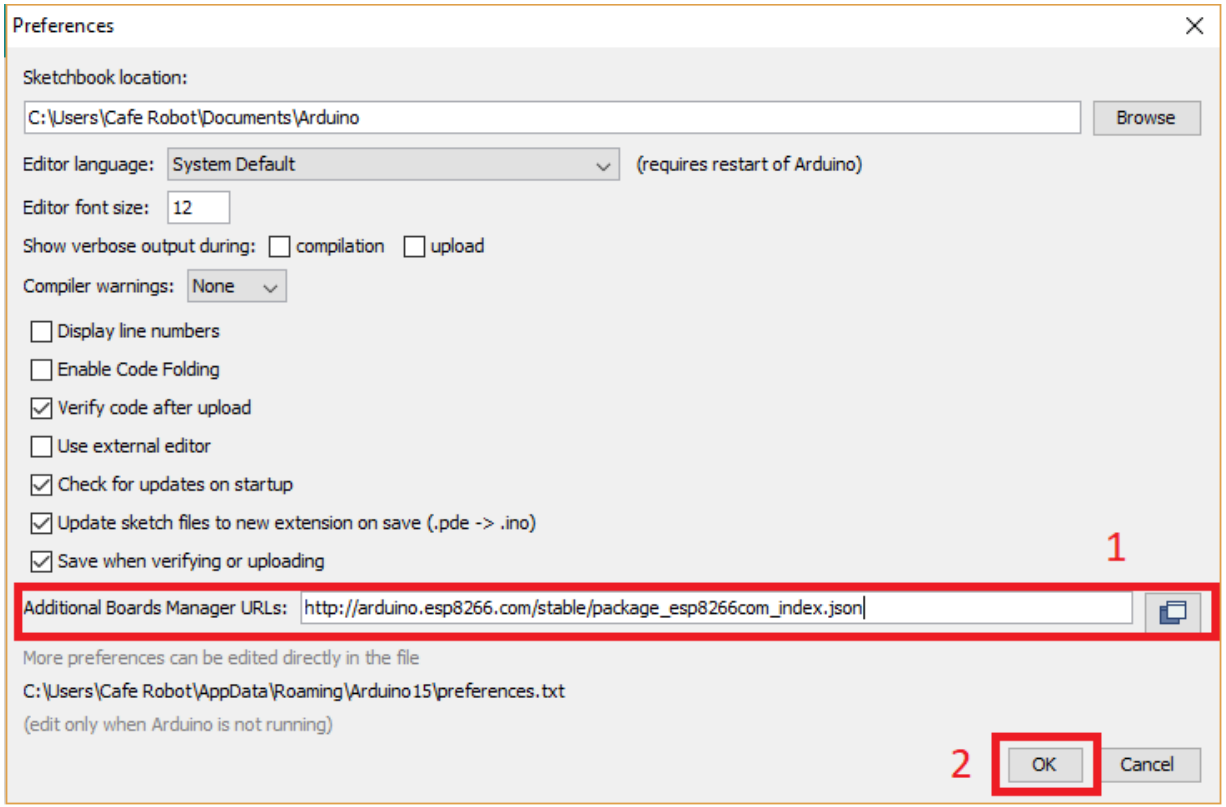
# NodeMCU

- **مزایای استفاده :**
- راحتی استفاده
- قابلیت برنامه نویسی با زبان برنامه نویسی Arduino IDE
- قابلیت استفاده بصورت access point یا station
- قابلیت استفاده در کاربردهای Event-driven API
- دارای آنتن داخلی
- دارای ۱۳ پایه

# معرفی NodeMCU به آردوینو

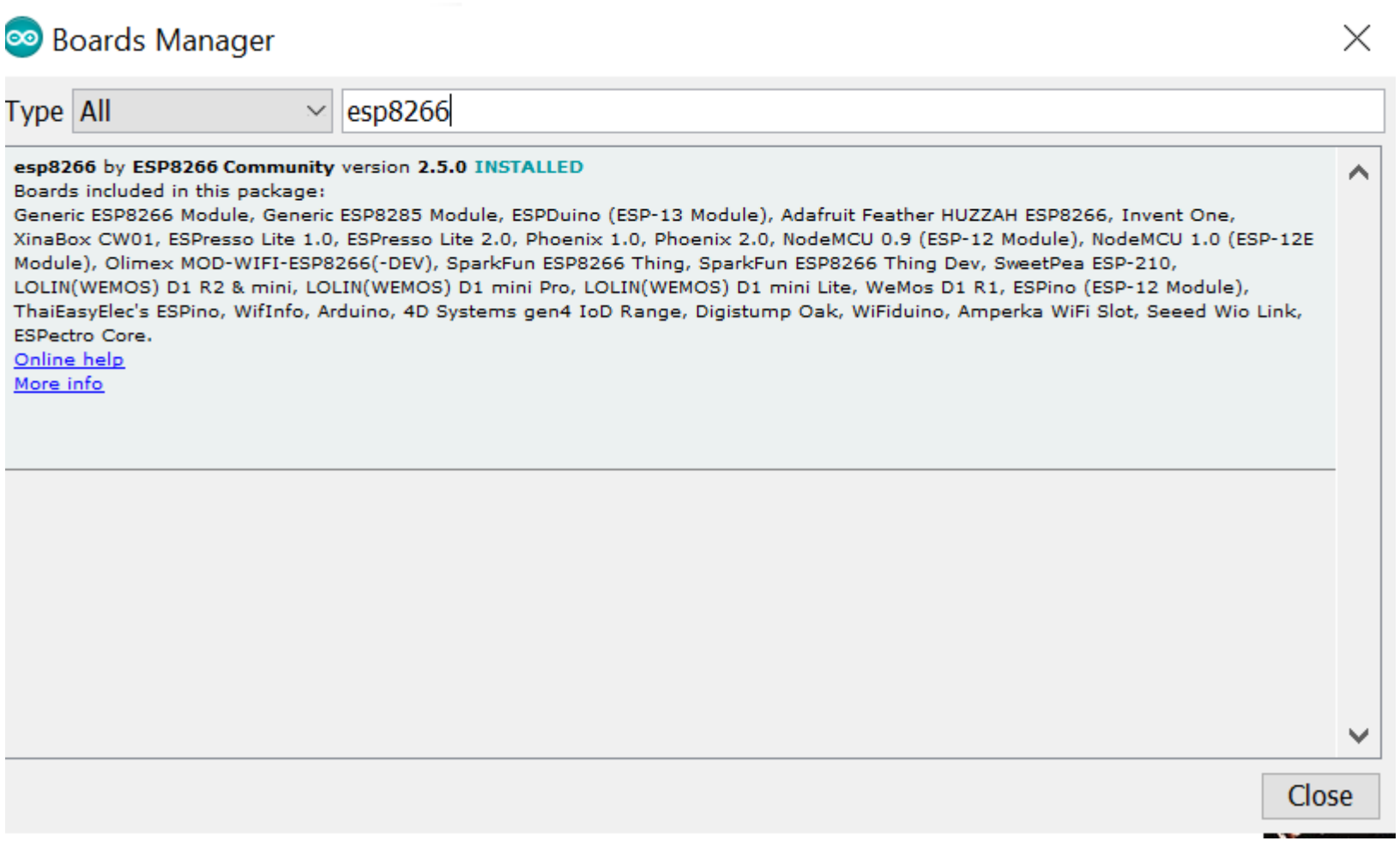
- برای استفاده از Arduino IDE به منظور پروگرم کردن Nodemcu ابتدا باید این بورد را به نرم افزار معرفی کنید. برای این کار کد زیر را کپی کرده و گام پیش رو را انجام دهید:

http://arduino.esp8266.com/stable/package_esp8266com_index.json



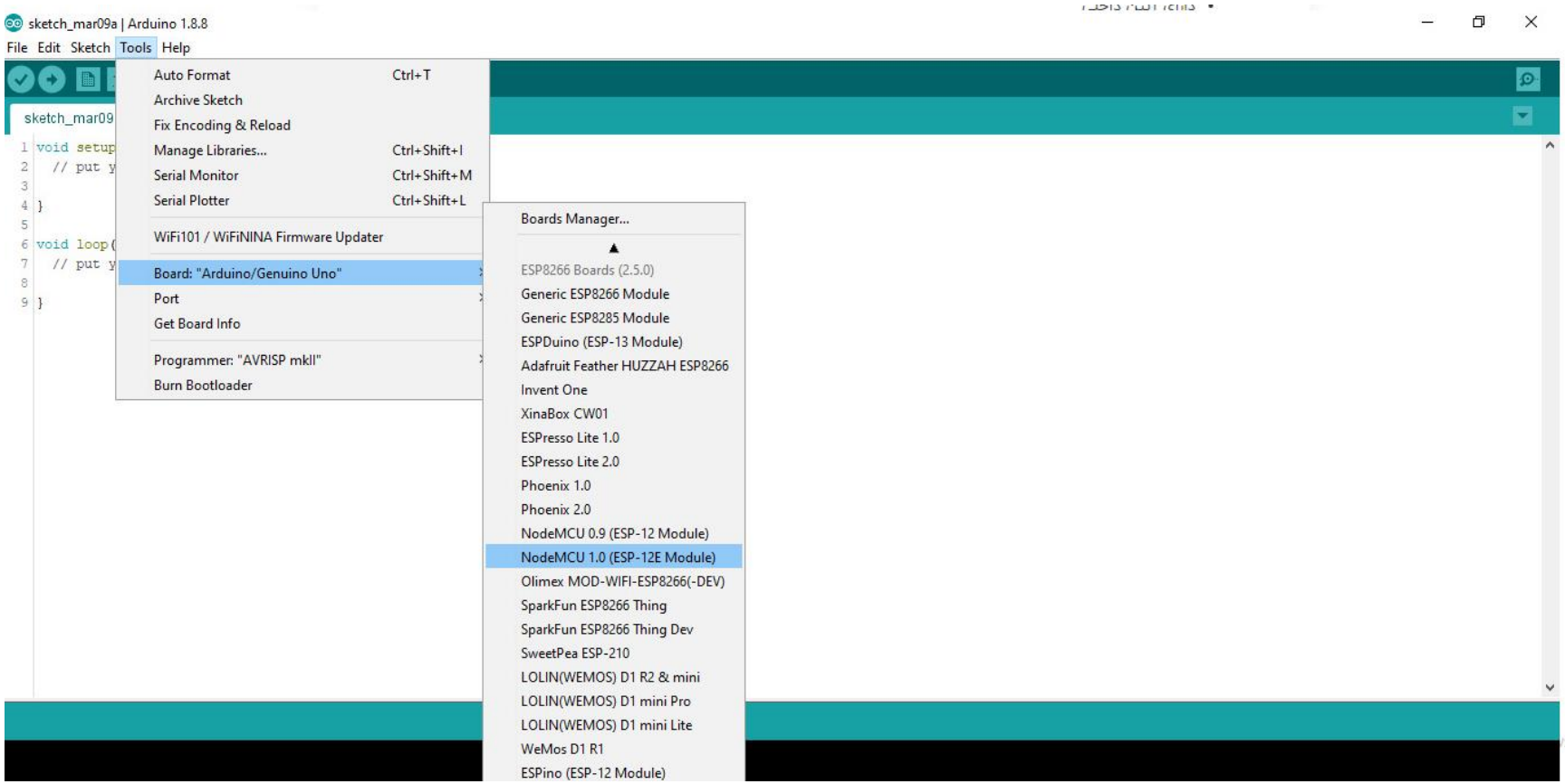
# معرفی NodeMCU به آردوینو

- **گام دوم)** به منوی Toolsبخش Boards>Boards Managerرفته و کلمه Esp8266را در تب موجود سرچ کنید و بوردهای Esp8266را نصب کنید، پس از پایان مراحل نصب لیبل نصب شده برروی بوردهای Esp8266ظاهر می

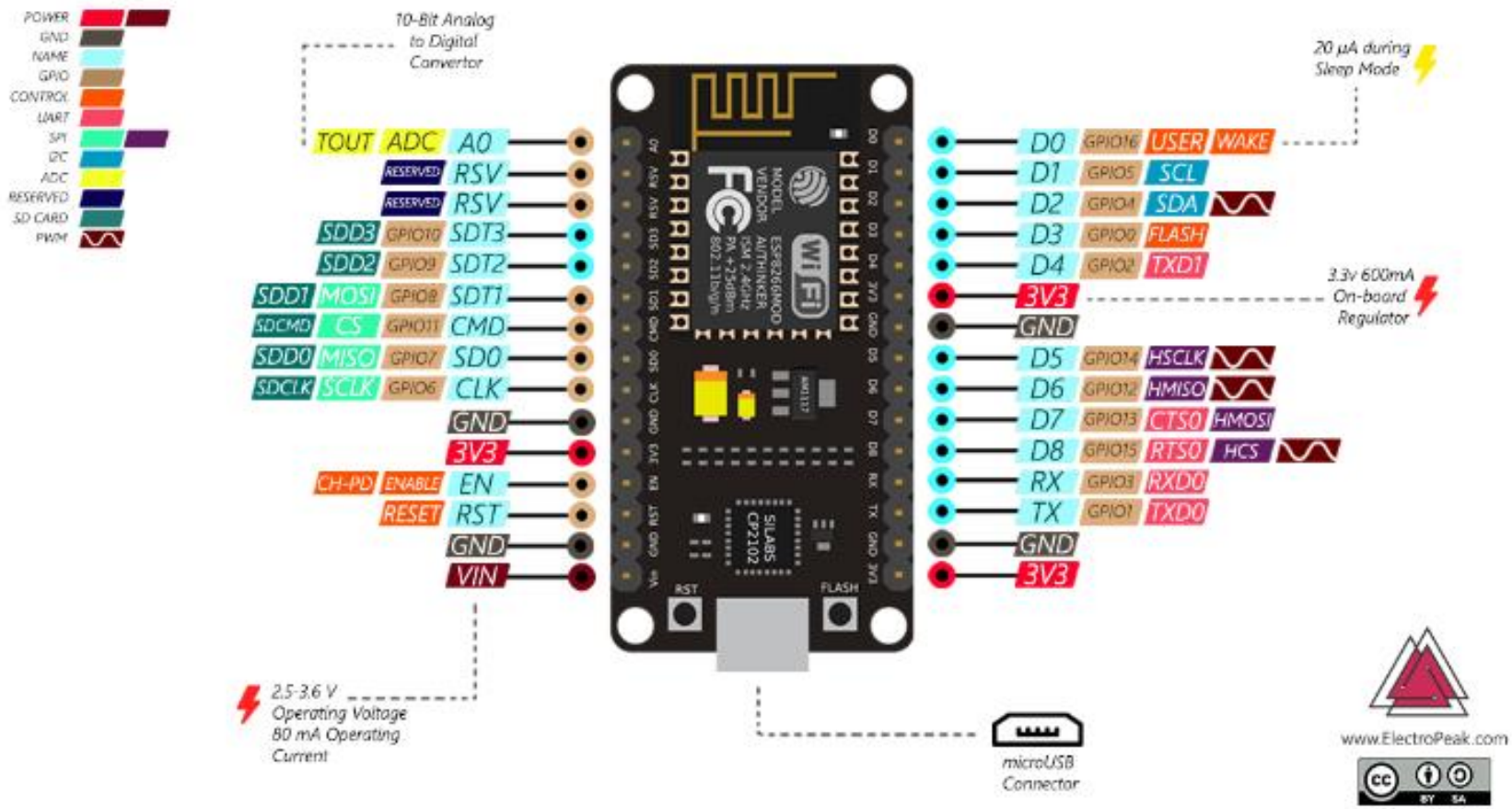


# معرفی NodeMCU به آردوینو

- پس از پایان این دو گام انواع بوردهای مبتنی بر Esp8266 از جمله Nodemcu به لیست بوردهای Arduino IDE شما اضافه می شود و با انتخاب آن می توانید برروی آن کد مدنظر خود را پروگرم کنید.



# شمای پایه های Nodemcu



# OGC Sensor Service

- **OGC Sensor Servicesیک مجموعه استانداردهای تعریف شده توسط کارگروه تطبیق مکانی باز (OGC) است که به کشف، دسترسی و تبادل داده‌ها و سرویس‌های مرتبط با سنسور کمک می‌کند. این استانداردها یک چارچوب برای مدیریت و ارتباط اطلاعات جمع‌آوری شده توسط انواعی از سنسورها، مانند ایستگاه‌های هواشناسی، دستگاه‌های اینترنت اشیاء (IoT)، پهپادها و ماهواره‌ها، فراهم می‌کند.**

- Sensor Observation Service (SOS): این استاندارد به تعریف یک رابط برای پرس‌وجو و دسترسی به مشاهدات سنسور می‌پردازد. این امکان را به کاربران می‌دهد تا داده‌های سنسورهای زمان‌واقعی یا تاریخی را همراه با اطلاعات موقعیت مکانی و زمانی، مانند مقادیر دما، رطوبت یا کیفیت هوا، بازیابی کنند.

- Sensor Planning Service(SPS): SPS با ارائه دستورات برای پردازش و کنترل سنسورها، فرآیند تعامل با سنسورها را تسهیل می‌کند. این استاندارد به کاربران امکان می‌دهد عملیات خاص سنسورها را مشخص و درخواست کنند، مانند گرفتن تصویر یا جمع‌آوری اندازه‌گیری‌های خاص در زمان و مکان مشخص.

- Sensor Web Enablement (SWE): SWE مجموعه‌ای از استانداردها هستند که جنبه‌های مختلفی از مدل‌سازی، رمزگذاری و ارتباط داده‌های سنسور را شامل می‌شوند. آنها شامل مؤلفه‌هایی مانند SensorML برای توصیف سنسورها و قابلیت‌های آنها، Sensor Markup Language (SensorML) برای رمزگذاری فرادادههای سنسور، و Observations and Measurements (O&M) برای نمایش مشاهدات سنسورها هستند.

# Sensor Observation Service

- **سرویس مشاهده حسگر ( (SOSیک پروتکل استاندارد برای دسترسی، پرس و جو و بازیابی داده های حسگر از طریق اینترنت است. این یک جزء کلیدی از مجموعه استانداردهای Sensor Web Enablement (SWE)است که توسط کنسرسیوم فضایی باز ( (OGCتوسعه یافته است.**

- SOSبه کاربران این امکان را می دهد که مشاهدات حسگر را در زمان واقعی یا بایگانی شده از انواع مختلف حسگرها مانند ایستگاه های هواشناسی، سنسورهای کیفیت هوا، سنسورهای کیفیت آب و غیره درخواست و دریافت کنند. این مشاهدات می تواند شامل پارامترهایی مانند دما، رطوبت، فشار، سطوح آلودگی و هر مقدار قابل اندازه گیری دیگری باشد که سنسور برای گرفتن آن طراحی شده است.
- SOSاز معماری سرویس گیرنده–سرور پیروی می کند، که در آن مشتری پرس و جوهایی را برای درخواست داده های حسگر خاص به سرور ارسال می کند. سپس سرور درخواست را پردازش می کند و مشاهدات درخواستی را در قالب استاندارد شده، مانند فرمت مشاهدات و اندازه گیری SensorMLیا OGC (O&M)برمی گرداند.

# Sensor Observation Service

## ❑ توابع SOS:

- ❖ GetCapabilities:قابلیت های سرور SOS، مانند نسخه های پشتیبانی شده، پدیده های مشاهده شده و رویه های موجود را بازیابی می کند.
- ❖ DescribeSensor: فراداده های مربوط به یک حسگر خاص، از جمله قابلیت ها، مکان و پارامترهای مشاهده شده آن را بازیابی می کند.
- ❖ GetObservation: مشاهدات حسگر را بر اساس معیارهای مشخص شده، مانند محدوده زمانی، منطقه جغرافیایی، یا شناسه های حسگر خاص بازیابی می کند.
- ❖ InsertObservation: اجازه درج مشاهدات حسگر جدید در پایگاه داده SOSرا می دهد.
- ❖ RegisterSensor: یک حسگر جدید را با سرور SOSثبت می کند و متادیتا درباره حسگر و قابلیت های آن ارائه می دهد.

# نرم‌افزارهای SOS

- 52°North SOS: این یک پیاده سازی منبع باز استاندارد SOS است که توسط ۵۲° North Initiativeتوسعه یافته است. این به زبان جاوا نوشته شده است و اجرای سرور SOSقوی و قابل تنظیم را ارائه می دهد.

- istSOS :
- یکی دیگر از اجرای SOSمنبع باز است که در پایتون نوشته شده است. این یک راه حل سبک و انعطاف پذیر برای مدیریت و دسترسی به مشاهدات حسگر ارائه می دهد.

# با تشکر از توجه شما