آشنایی با پروتکل‌های ارتباطی و سنسورها

صدف صادقیان (۸۱۰۱۹۵۴۱۹) ، یاسمن جعفری (۸۱۰۱۹۵۳۷۶) ، فرزاد حبیبی (۸۱۰۱۹۵۳۸۳)

پاسخ سوالات

۱)

زمانی که دو master می‌خواهند همزمان از خط SDA داده بخوانند و یا داده بفرستند. برای حل این مشکل هر master قبل از انتقال دیتا باید چک کند که خط SDA، high است یا low است. اگر مقدار خط low است یعنی کنترل bus در اختیار یک master دیگر است و master باید برای فرستادن داده صبر کند.اگر SDA line، high است، می‌تواند داده را بفرستد و امن است.

۲)

- Master شرط شروع (start condition) را به تمام slave ها با عوض کردن SDA line از high voltage به low voltage قبل از عوض کردن SCL line از high به low می‌فرستد.
- master به تمام slave ها آدرس ۷ یا ۱۰ بیتی متعلق به slave ای که می‌خواهد با آن ارتباط برقرار را به همراه read/write bit می‌فرستد.
- هر slave آدرس فرستاده شده توسط master را با آدرس خودش مقایسه می‌کند. اگر آدرس ها با هم match شدند slave یک بیت ACK را با صفر کردن SDA line به اندازه یک بیت می‌فرستد و در غیر این صورت می‌گذارد SDA line high بماند.
- master یک فریم از داده را می‌فرستد یا می‌گیرد.
- بعد از اینکه هر فریم از داده منتقل شد، device ای که در حال گرفتن داده است یک بیت ACK دیگر به فرستنده برای اطلاع دادن دریافت موفق فریم داده، به فرستنده می‌فرستد.
- برای متوقف کردن انتقال داده، master یک شرط خاتمه(stop condition) به slave توسط high کردن SCL و سپس high کردن SDA می‌فرستد.

۳)

Baud rate سرعت انتقال داده روی خط سریال را مشخص می‌کند و برحسب bit-per-seconds است. این مقدار مشخص می‌کند transmitter باید چه مقدار serial line را high یا low نگه دارد و reciever با چه سرعتی از line ش نمونه برداری می‌کند.یک مقدار استاندارد ۹۶۰۰ است و هر چه این سرعت بیشتر شود، سرعت انتقال داده بیشتر خواهد شد.

این موضوع مهم است که baud rate بین دو دستگاهی که با هم ارتباط برقرار می‌کنند یکسان باشد چرا که اگر دو دستگاه با سرعت یکسان صحبت نکنند، داده ممکن است از دست برود و یا اشتباه تفسیر شود.

۴)

مزایای I2C:

- به تعداد کمی pin یا سیگنال نیاز است. (حتی در زمانی که تعداد زیادی device روی bus داریم.)
- به راحتی می‌تواند با دستگاه‌های slave مختلف، وفق داده شود.
- می‌تواند چندین دستگاه master را support کند.
- برای داشتن error handling بهتری داشته باشد، از ACK/NACK functionality استفاده می‌کند.

معایب I2C:

- اضافه شدن پیچیدگی به لایه سطح پایین سخت‌افزار و firmware
- سربار پروتکل که باعث کاهش throughput می‌شود.
- نیاز به pull-up resistor ها دارد که باعث ایجاد محدودیت روی سرعت clock، مصرف بالای توان و همچنین گرفته شدن فضای باارزش PCB در سیستم‌هایی با فضای محدود می‌شوند.

مزایای UART:

- تنها از ۲ تا سیم استفاده می‌کند.
- بدون نیاز به Clock signal
- یک بیت parity دارد که می‌تواند به error checking کمک کند.
- ساختار packet داده می‌تواند در صورت set up شدن ۲ طرف برای آن، عوض شود.
- به خوبی document شده است و به صورت گسترده استفاده شده است.

معایب UART:

- سایز data frame محدود به حداکثر ۹ بیت است.
- نمی‌تواند چندین سیستم slave و master را به صورت همزمان support کند.
- Baud rate هر UART باید در محدوده 10% بقیه باشد.

درواقع، استفاده از I2C زمانی مناسب است که شبکه‌ای پیچیده، متنوع و بزرگ برای ارتباط دستگاه‌ها داریم. UART interface ها معمولا برای ارتباطات point-to-point استفاده می‌شوند زیرا در آن‌ها هیچ راه استانداردی برای آدرس‌دهی به دستگاه‌های مختلف یا به اشتراک‌گذاری پین‌ها در آن وجود ندارد.

۵)

سخت‌افزار آردوینو ارتباطات سریال را بر روی پین‌های ۰ و ۱ ساپورت می‌کند. پورت‌های سریال اصلی آردوینو توسط قطعه‌ای سخت‌افزاری ایجاد می‌شوند که به اسم UART معروف است. کتابخانه‌های نرم‌افزاری‌ای وجود دارد که این ساپورت را با استفاده از نرم‌افزار فراهم می‌آورند. به این صورت که به طور معمول کار با آن‌ها مشابه با پورت‌های سخت‌افزاری اصلی می‌باشد. با این تفاوت که در ابتدا باید دو عدد پین دیجیتال را به عنوان پین‌های RX , TX معرفی کنیم.

از محدودیت‌های سریال نرم‌افزاری می‌توان به موارد زیر اشاره کرد.

- در صورتی که از چندین سریال نرم‌افزاری استفاده کنیم در یک زمان تنها یکی از آن‌ها می‌تواند اطلاعات را دریافت کند.
- در بعضی از ورژن‌های آردوینو، با توجه به کتابخانه‌ی استفاده شده ممکن است که همه‌ی پین‌های دیجیتال از این امکان نتوانند بهره‌مند شوند.
- در بعضی از کتابخانه‌ها، با توجه به ورژن‌های مختلف کتابخانه‌ها ممکن است به سرعت یک سریال سخت‌افزاری نتوانیم دست یابیم.
- هر کدام از کتابخانه‌ها ممکن است مزیت‌هایی نسبت به دیگر کتابخانه‌ها داشته باشند که و از آنجایی که سربار نرم‌افزار وجود دارد نتوان به اندازه‌ی سریال سخت‌افزاری از مزیت‌های آن استفاده کنیم.

۶)

این سنسور از موج‌های با فرکانس بالا استفاده می‌کند به این شکل که موج را می‌فرستد ودر صورتی که مانعی باشد، موج منعکس شده و به سنسور برمی‌گردد.سنسور فاصله زمان طی شده را تا از وقتی موج فرستاده می‌شود تا وقتی که موج بازگردانده شده را دریافت کند، اندازه‌گیری می‌کند و با استفاده از این زمان و سرعت صوت فاصله را محاسبه می‌کند.

Distance = (time * speed) / 2

سنسور SRF04 بازه بین ۳ سانتی متر تا ۳ متر را می‌تواند اندازه‌گیری کند و دقت این سنسور ۳ تا ۴ سانتی‌متر است.

(۷

در صورتی که بخواهیم از ۲ تا سنسور دو ماژول فاصله‌سنج داشته باشیم، ممکن است از آنجا که هر دو با استفاده از اشعه‌های مادون قرمز کار می‌کنند، در کار یکدیگر تداخل ایجاد کنند. بنابراین هنگام کار با آن‌ها باید مطمئن شد که در جهات مختلف باشند و یا یک switching algorithm روی arduino داشته باشیم که بین این سنسورها switch کند و چک کند که وقتی یکی روشن است، دیگری حتما خاموش باشد.

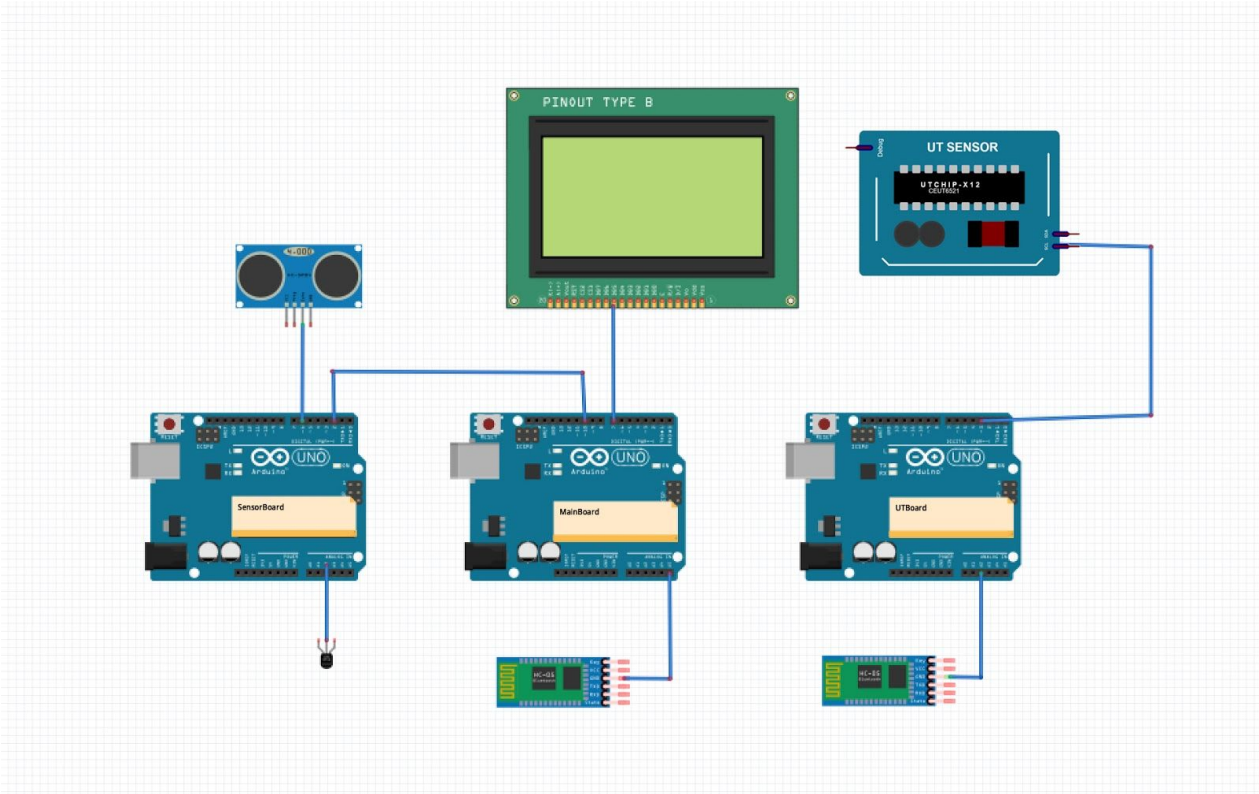
(۸

خیر اتصال چند سنسور مختلف با I2C مشکلی ایجاد نمی‌کند چرا که سنسورهای مختلف آدرس‌های متفاوتی دارند و زمانی که در ارتباط I2C می‌خواهد ارتباطی ایجاد شود چون device ها از طریق آدرس شناخته می‌شوند مشکلی پیش نمی‌آید.

چون دو سنسور یکسان آدرس‌های یکسانی دارند برای داشتن دو سنسور مشابه باید طوری پیاده‌سازی کنیم که یکی از آنها زمانی که روی خط مقدار high است یکی از آنها روشن است و داده را valid می‌بیند و زمانی که آن آدرس صدا می‌شود، پاسخ می‌دهد و زمانی که low است دیگری روشن است.

اما در صورت داشتن سه سنسور مشابه مشکل پیدا خواهیم کرد چرا که در این حالت هر سه آدرس یکسان دارند و همچنین از روشی همانند روشی که برای دو تا استفاده کردیم، نمی‌توانیم استفاده کنیم چون فقط دو مقدار می‌تواند روی خط باشد.

(۹



```
mainBoard {
        Initialization for Bluetooth Serial port , LCD and Software Serial Port
        while(true)
        {
                If ( is data available from bluetooth )
                        Save it in utBoard buffer;
                        If ( buffer is full )
                                Show utBoard buffer on LCD;
                                Clean utBoard buffer;
                If ( is data available from software serial port )
                        Save it in sensorBoard buffer;
                        If ( reached end of message)
                                Show sensorBoard buffer on LCD;
                                Clean sensorBoard buffer;
        }

}


sensorBoard {
        Initialization for Serial Port and temperature sensor instantiation
        while(true)
        {
                Read temperature value from sensor;
                Read distance value from sensor;
                Construct message for sending;
                Send message to main board;
                Wait for 100 ms;
        }

}
```

```
utBoard {
        Initialization for Bluetooth Serial port and I2C port and turn on UT sensor;
        while(true)
        {
                Read the first value from sensor and store (x value);
                Read the second value from sensor and store (y value);
                Calculate the sigma of  x^2 + y^2 and store;
                Send the previous result via bluetooth to main board;
                Wait for 100 ms;
        }
}
```