Tugas Post-Day SEKURO 2023: Day 3

Rizmi Ahmad Raihan (16523196)

A. Pengantar

Layaknya komunikasi manusia, komunikasi antar perangkat elektronik harus dilakukan dalam sebuah bahasa yang dapat dipahami oleh pihak dalam komunikasi. Dalam elektronik, bahasa yang digunakan disebut sebagai protokol komunikasi (*communication protocol*). Terdapat banyak jenis protokol. Beberapa protokol yang marak digunakan adalah SPI, I^2C, dan UART.

Terdapat dua tipe komunikasi data; paralel dan serial. SPI, I^2C dan UART termasuk dalam komunikasi serial. Berbeda dengan komunikasi pararel, komunikasi serial hanya menggunakan satu jalur komunikasi (*communication line*) sehingga bit data ditransmisikan satu persatu. Komunikasi serial mungkin tidak lebih cepat dibandingkan komunikasi pararel, tetapi komunikasi serial lebih sederhana dan lebih murah untuk diterapkan terutama untuk komunikasi komponen elektronik (sensor, motor, dsb.) dengan mikrokontroler.

Komunikasi serial menggunakan dua metode; sinkronus dan asinkronus.

1. Asinkronus
    a. Mengirimkan satu bit setiap waktu
    b. Tidak membutuhkan sinyal waktu (*clock signal*)
    c. contoh: UART (Universal Asynchronous Reciever Transmitter).
2. Sinkronus
    a. Mengirimkan sebuah blok data setiap waktu.
    b. Membutuhkan sinyal waktu.
    c. Contoh: SPI (Serial Peripheral Interface), I2C (Inter Integrated Circuit).

Selain itu, terdapat jenis-jenis transmisi data; simplex, half duplex, dan full duplex.

1. Simplex
   Transmisi data dilakukan hanya satu arah. Hanya satu arah dari TX (transmitter) ke RX (reciever).
2. Half-duplex
   Transmisi data dapat dilakukan dalam dua arah, tetapi hanya satu arah di saat yang sama. Hanya satu arah dari TX ke RX **atau** dari RX ke TX dalam waktu yang sama.
3. Full-duplex
   Transmisi data dapat dilakukan dalam dua arah **pada saat yang sama**. Terdapat dua TX dan dua RX, masing-masing berpasangan sehingga menghasilkan komunikasi RX-TX dan TX-RX.

B. Baud Rate

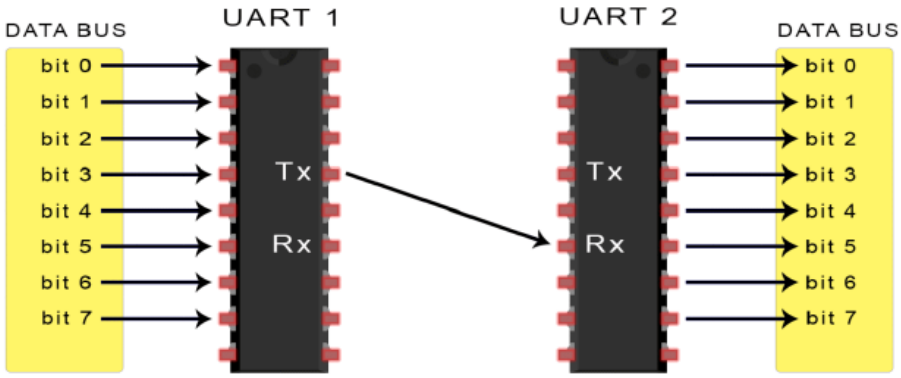
Kecepatan transfer data dalam komunikasi serial diukur dalam satuan bit per detik (*bits per second*, bps) yang dikenal sebagai Baud Rate. 9600 baud bermakna perangkat dapat melakukan transfer data dengan maksimum secepat 9600 bit per detik.

Sebagai contoh, dalam pengiriman data melalui USB menggunakan Serial, Arduino dapat mengirimkan/menerima data dengan Baud Rate 300 s.d. 115200 baud.

C. UART

a. Pengantar

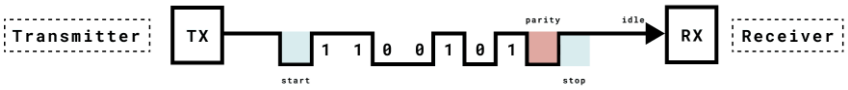
UART (Universal Asynchronous Reciever Transmitter) adalah salah satu jenis protokol komunikasi antarperangkat (serial) yang paling sering digunakan. UART adalah protokol yang digunakan oleh Arduino untuk berkomunikasi dengan komputer. UART memungkinkan pengaturan baud rate dan format data. Dalam Arduino, komunikasi melalui UART dilakukan dengan Serial.



Gambar I: Ilustrasi Komunikasi dengan UART. Sirkuit UART terhubung langsung dengan bus data

b. Cara Kerja

UART bekerja dengan mentransmisikan data sebagai bit-bit yang terdiri dari bit mulai (*start bit*), bit data (*data bit*), dan bit pariti (*parity bit*). UART bekerja secara seri (bit dikirimkan satu-persatu; tidak secara pararel). UART mengirimkan data secara asinkronus yang berarti data yang dikirimkan melalui protokol UART tidak memiliki *clock signal* yang berguna untuk menyinkronkan transmisi data. UART mengandalkan baud rate yang diatur dalam perangkat *reciever* dan *transmitter* sebagai pengganti sinyal clock. Oleh sebab itu, dalam komunikasi UART, perangkat *reciever* dan *transmitter* harus memiliki baud rate yang sama.



Gambar II: Paket data dalam protokol UART.

Data yang ditransmisikan melalui protokol UART disusun dalam "paket". Setiap paket mengandung 1 bit mulai, 5 s.d. 9 bit daya, bit paritas opsional, dan 1 atau 2 bit berhenti (*stop bits*). Penerima data (*reciever*) akan menghapus bit mulai, bit paritas, dan bit berhenti dari paket data.

Bit yang ditransmisikan melalui protokol UART berfungsi sebagai berikut:

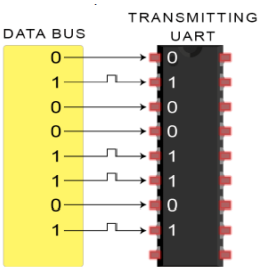
1. Bit mulai (*start bit*)
   Bit yang ditransmisikan di awal sinyal UART; digunakan untuk menandakan awal transmisi data dan mempersiapkan penerima untuk transfer data. Dalam UART, bit mulai selalu bernilai 0 (0 tidak berarti 0 V, tetapi berada dalam rentang tegangan lain yang bukan 1).
2. Bit data (*data bits*)
   Berisi informasi yang disampaikan dari pengirim ke penerima. Bit data mengandung informasi sesungguhnya yang dikirimkan dari pengirim ke penerima.

Jumlah bit data protokol UART dapat bervariasi. Namun, konfigurasi bit data yang paling sering digunakan untuk protokol UART dalah 8-bit. Konfigurasi lain yang digunakan adalah 7-bit dan 6-bit.

3. Bit paritas (*parity bit*)
   Protokol UART mungkin mengandung bit paritas yang berfungsi sebagai *error-checking* data yang dikirimkan. Paritas dapat di-set sebagai ganjil atau genap sehingga memastikan jumlah bit bernilai 1 ganjil atau genap. Apabila jumlah bit bernilai 1 tidak sesuai dengan bit paritas, penerima dapat mengetahui keberadaan error. Namun, penerima tidak dapat mengetahui letak error.
4. Bit berhenti (*stop bit*)
   Menandakan akhir transmisi. Biasanya, digunakan satu bit untuk menandakan komunikasi selesai.

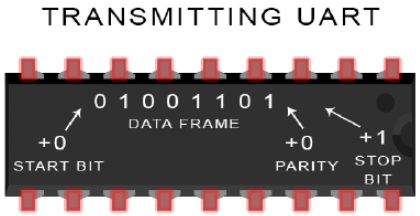
c. Tahap-tahap Transmisi

1. UART pengirim menerima data dari bus data



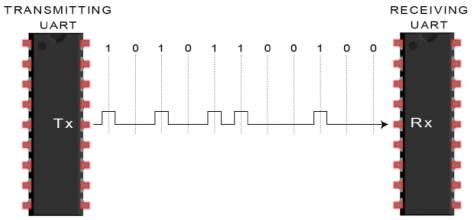
Gambar III: data dikirim dari data bus.

2. UART pengirim menambahkan bit mulai, bit paritas, dan bit berhenti.



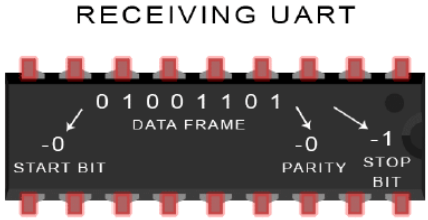
Gambar IV: UART pengirim menambahkan bit mulai, bit paritas, dan bit berhenti

3. Paket data dikirim dari UART pengirim ke UART penerima. UART penerima menerima data berdasarkan baud rate yang sudah diatur.



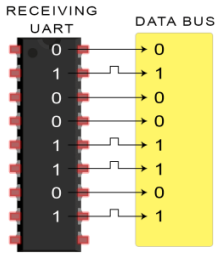
Gambar V: Transfer data

4. UART penerima membuat bit mulai, bit paritas, dan bit henti.



Gambar VI: UART penerima membuang bit-bit selain bit data

5. UART penerima mengubah data ke format serial dan mengirimkannya ke bus data penerima



Gambar VII: UART penerima mengubah format.

d. Kelebihan dan Kekurangan
   1. Kelebihan
      a. Tidak membutuhkan sinyal clock
      b. Mempunyai bit paritas untuk mengecek error
      c. Marak digunakan
   2. Kekurangan
      a. Ukuran data yang dikirim terbatas 9 bit
      b. Tidak mendukung banyak *slave* atau *master*

D. SPI (Serial Peripheral Interface)

a. Pengantar

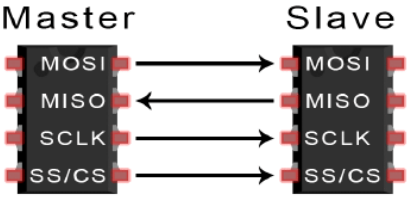
SPI adalah protokol komunikasi sinktonus, *full-duplex*. SPI digunakan di banyak perangkat; pembaca SD Card, pembaca RFID, hingga transmitter/reciever *wireless* 2.4 GHz. Semua perangkat tersebut menggunakan protokol SPI untuk berkomunikasi dengan mikrokontroler.

Keuntungan dari menggunaan SPI adalah data dapat dikirimkan tanpa interupsi. Jumlah bit berapapun dapat dikirim atau diterima secara kontinu. Hal ini berbeda dengan protokol I^2C dan UART yang mengirimkan data sebagai paket-paket data. Bit mulai (*start bit*) dan bit berhenti (*stop bit*) memisahkan paket-paket data sehingga transmisi data terinterupsi dalam transmisi.

Perangkat yang terhubung melalui protokol SPI berada dalam hubungan *master-slave* (atau dalam terminologi Arduino yang diperbaharui; *controller-peripheral*). Perangkat master adalah perangkat kontrol (biasanya berupa mikrokontroler), sementara perangkat *slave* adalah perangkat *input/output* (contoh: sensor, display, memory chip). Perangkat slave menerima instruksi dari master. Konfigurasi paling sederhana komunikasi SPI adalah sistem master dan slave tunggal.

Melalui protokol SPI, sebuah master dapat mengonrol lebih dari satu slave.

b. Cara Kerja



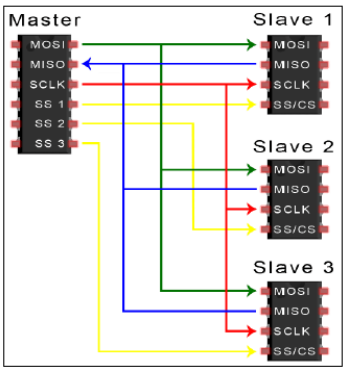
Gambar IIX: Konfigurasi sederhana protokol SPI

SPI bekerja secara sinkronus yang berarti SPI menggunakan jalur yang berbeda untuk sinyal clock. Sinyak klock adalah sinyal yang memberi tahu reciever (*slave*) waktu untuk membaca data sehingga menyinkronkan pengiriman dan penerimaan data. Ketika slave menerima sinyal clock yang sesuai, slave akan membaca data (membaca bit selanjutnya).

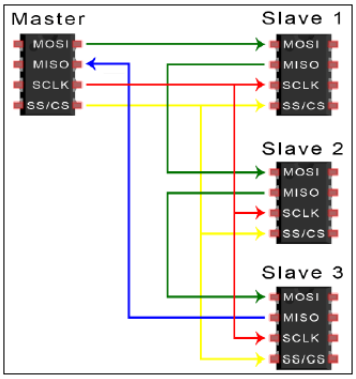
SPI menggunakan beberapa pin, yaitu:

1. MOSI (Master Output/Slave Input) juga dikenal sebagai (SDO; Serial Data Out)
   Pin untuk master untuk mengirimkan data kepada slave.
2. MISO (Master Input/Slave Output) juga dikenal sebagai (SDI; Serial Data In)
   Pin untuk slave menerima data dari master.
3. SCLK (Clock)
   Pin untuk sinyal clock
4. SS/CS (Slave Select/Chip Select)
   Pin memilih slave untuk master (master dapat menangani lebih dari satu slave dengan SPI).

SPI dapat diatur sedemikian rupa sehingga dapat menangani lebih dari satu slave. Pengkabelan SPI untuk lebih dari satu slave adalah sebagai berikut:

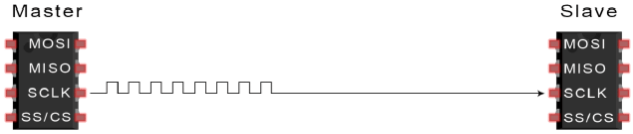


Gambar IX: Konfigurasi SPI untuk dengan master memiliki banyak SS/CS



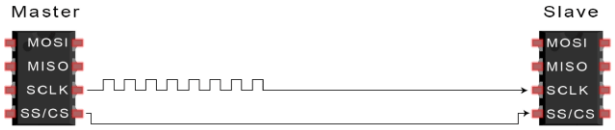
Gambar X: Konfigurasi SPI untuk master dengan SS/CS tunggal

c. Tahap Transmisi
   a. Master mengirimkan sinyal clock



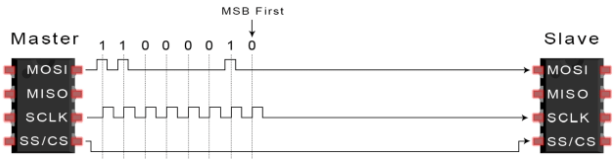
Gambar XI: Transmisi UART I

b. Master mengirimkan sinyal voltase rendah di pin SS/CS yang mengaktifkan slave tertentu.



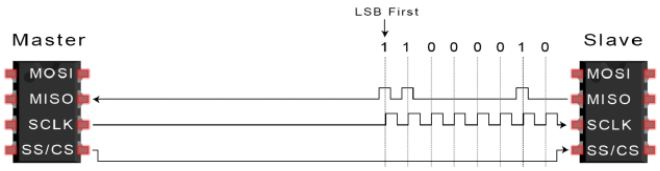
Gambar XII: Transmisi UART II

c. Master mengirimkan data bit per bit melalui MOSI. Slave membaca bit tersebut.



Gambar XIII: Transmisi UART III

d. Apabila respon harus dikirim dari slave atau slave harus mengirimkan data ka master, slave akan mengirimkan sinyal melalui MOSI



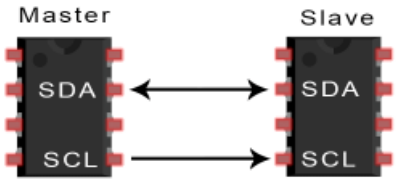
Gambar XIV: Transmisi UART IV

d. Kelebihan/Kekurangan
    a. Kelebihan
        i. Tidak ada bit mulai/behenti sehingga data dapat dikirimkan secara kontinu
        ii. Tidak ada sistem *slave addressing* seperti I2C
        iii. Laju transfer data lebih tinggi daripada I2C
        iv. MISO dan MOSI terpisah sehingga pengiriman dan penerimaan data dapat terhajai bersamaan.
    b. Kekurangan
        i. Menggunakan 4 kabel
        ii. Tidak ada konfirmasi data terkirim/tidak (I2C mempunyai ini)
        iii. Tidak ada paritas
        iv. Hanya dapat digunakan untuk satu master.

E. I2C
    a. Pengantar

    I2C merupakan singkatan dari *Inter-Integrated Circuit*. Protokol I2C dapat menghubungkan beberapa slave dengan beberapa master. Hal ini sangat berguna apabila beberapa mikrokontroler digunakan untuk berbagai keluaran/masukan.

    Protokol I2C menggunakan dua kabel untuk transfer data antar-perangkat. SDA (Serial Data Line) dan SCL (Serial Clock Line). Oleh sebab itu, protokol I2C menggunakan jumlah pin yang belih sedikit dibandingkan komunikasi pararel sehingga mengurangi biaya produksi, ukuran alat, dan konsumsi daya.



Gambar XV: Ilustrasi Komunikasi dengan Protokol I2C

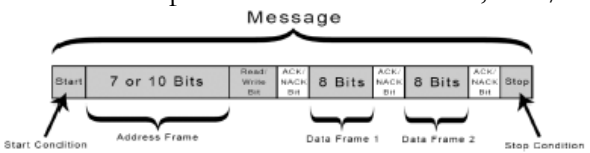
b. Cara Kerja

I2C bekerja secara serial (data ditransmisikan bit-per-bit dalam satu kabel [kabel SDA]). Seperti SPI, I2C bersifat sinkronus sehingga bit output harus disinkronkan dengan sinyal clock yang dikirimkan master (dalam I2C, sinyal clock selalu dikontrol oleh master).

I2C menggunakan dua pin, yaitu:

a. SDA (Serial Data)
   Digunakan untuk pengiriman data antara slave dan master.
b. SCL (Serial Clock)
   Digunakan untuk pengiriman sinyal clock.

Dengan I2C, data ditransmisikan dalam bentuk *pesan*. Pesan dipecah-pecah menjadi frame data. Setiap pesan memiliki frame alamat yang berisi alamat bit slave yang dituju. Pesan yang dikirimkan memiliki pesan mulai dan berhenti, read/write, dsb.,



Gambar XVI: Pesan yang dikirimkan melalui I2C.

- Start Bit
  Bit yang berfungsi untuk menandakan dimulainya pengiriman data. Jalur SDA berganti dari tegangan tinggi ke tegangan rendah sebelum jalur SCL berganti dari tegangan tinggi ke rendah.
- Address
  Frame yang menandakan perangkat yang dituju.
- Read/Write
  Frame yang menyatakan slave menerima/memberikan data kepada master.
- ACK/NACK
  Aknowledge/no-acknowledge bit. Jika frame address/data frame diterima oleh slave, sebuah bit ACK dikirimkan kepada master dari slave.
- Data frame
  Data yang dikirimkan
- Stop Bit
  Jalur SDA berganti dari mode tegangan rendah ke mode tegangn tinggi setelah jalur SCL berpindah dari rendah ke tinggi.

c. Tahap Pengiriman

- Master mengirimkan start bit kepada setiap slave yang terhubung dengan mengubah jalur SDA dari tegangan tinggi ke tegangan rendah sebelum mengganti SCL dari tegangan tinggi ke rendah
- Master mengirimkan ke masing0masing slave 7 atau 10 bit address (alamat) tujuan komunikasi master bersamaan dengan bit read/write.

- Setiap slave membandingkan address (alamat) yang dikirmkan dengan addressnya sendiri. Jika address sesuai, slave mengirimkan ACK dengan mengubah SDA menjadi low (rendah, 0) sebanyak satu bit.
- Master mengirimkan/menerima frame data
- ACK dikirimkan oleh perangkat yang menerima data untuk menandakan pengiriman berhasil.
- Kondisi stop dikirimkan dengan mengubah SCL ke kondusu HIGH sebelum mengubah SDA menjadi HIGH.

F. CAN-BUS

CAN bus adalah sistem yang dikembangkan oleh BOSCH sebagi sistem *broadcast* multi-master dengan kecepatan maksimum 1 Mbps. CAN tidak dapat mengirimkan data berukuran besar dari titik ke titik. Dalam jaringan CAN, banyak pesan pendek berisi data tempeture atau RPM disiarkan di seluruh jaringan sehingga menyediakan konsistensi data untuk setiap titik dalam sistem. CAN digunakan untuk komunikasi dalam kendaraan.