

第九項 PCとの連携・シリアル通信の応用

電子工作創作表現(2019/6/21)

先週の「シリアル通信」前回まで

- PCから数字を1つ送る
- Arduinoから来る数字を受信する

シンプルに数字を送るだけ

- 複数のデータはどう区別する？
- 255より大きい値を送るには？

基本的な文字をやり取りする「アスキーコード」

- 簡単な文字を128種類におさめた「ASCII CODE」
- 英数字記号などの簡単な「文字」が送れる

1バイトを1文字として扱う

- 例:111:110:107:97:110 で「onkan」
- 数字も文字として登録されていて「0」は48から始まる
- 文字としてのデータが「アスキー」数字としてのデータが「バイナリ」

アスキーデータのやり取り

- 送るデータの自由度は上がる
- ただしプログラムは煩雑になる

シリアルモニタはアスキーベース

- ArduinoIDEの「シリアルモニタ」はアスキーベース
- 受信したデータを出力して、送信もできる

Arduinoでデータを受信する

- 文字を入れる変数「String」を使う
- 「<on 2>」「<off 2>」というコマンドでLEDをスイッチする例

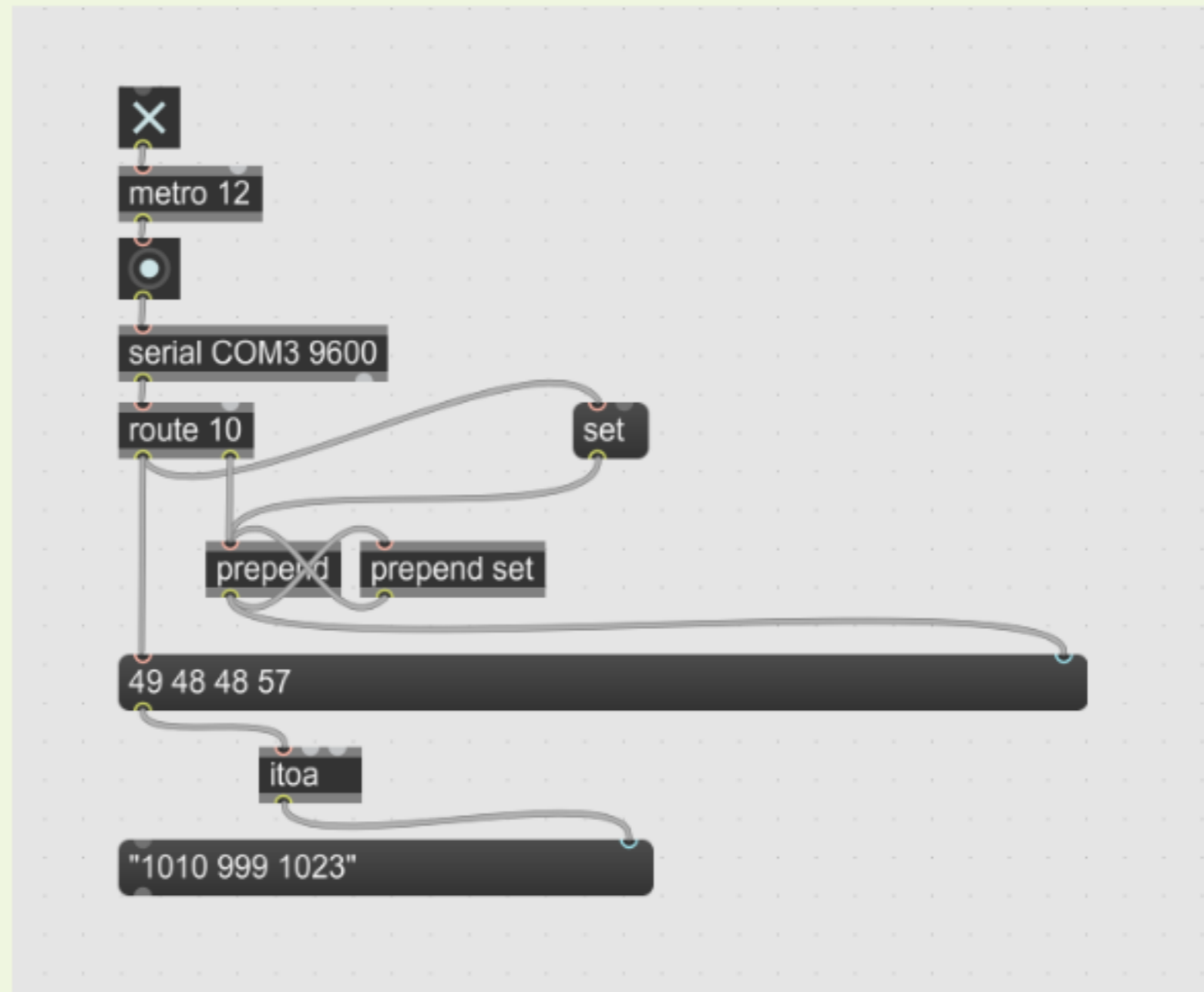
Arduinoでデータを送信する

- **Serial.print() もしくは Serial.println()**
 - **数字は自動的に変換してくれる**

表記できない文字の表記

- 「\ (バックslash)」の後にアルファベットで、タブや改行などを表現
- 「エスケープ文字」と言って特殊な文字を表記する時に使う

max/mspで受け取る



touchdesignerで受け取る

The screenshot displays the TouchDesigner interface with three main components:

- serial1**: A serial port component with a table view showing received data. The table has 11 rows (0-10) and 4 columns. The first column is labeled 'message'. The data is as follows:

Row	message
0	message
1	1006 987 1023
2	1006 987 1023
3	1006 987 1023
4	1005 986 1023
5	1005 985 1023
6	1005 984 1021
7	1005 984 1023
8	1005 985 1023
9	1005 985 1023
10	1005 985 1023
- serial1.Serial**: A configuration panel for the serial port. Settings include: Active: Off, Row/Callback Format: One Per Line, Port: COM3, Baud Rate: 9600, Data Bits: 8, Parity: None, Stop Bits: 2, DTR: Enable, RTS: Disable.
- constant1**: A constant component with a table view showing the processed data. The table has 3 rows and 2 columns. The first column is labeled 'chan1' and the second is labeled 'chan2'. The data is as follows:

chan1	chan2
1005	chan1
985	chan2
1023	chan3

Below the components is a code editor window titled **serial1_callbacks** containing the following Python code:

```
1 # me - this DAT
2 #
3 # dat - the DAT that received the data
4 # rowIndex - the row number the data was placed into
5 # message - an ascii representation of the data
6 # Unprintable characters and unicode characters will
7 # not be preserved. Use the 'bytes' parameter to get
8 # the raw bytes that were sent.
9 # bytes - byte array of the data received
10
11 def onReceive(dat, rowIndex, message, bytes):
12     list = message.split(' ')
13     op('constant1').par.value0 = list[0]
14     op('constant1').par.value1 = list[1]
15     op('constant1').par.value2 = list[2]
16
17     return
18
19
```

Processingで受け取る

- **Arduinoに近い書き方**