

第八項 PCとの連携・シリアル通信基礎

電子工作創作表現(2019/6/14)

パソコン<->Arduinoの通信

- 色々なソフトと連携する
- Processing, oF, Max/MSP, TouchDesigner...

USBシリアル通信

- 1バイトずつデータを送る、シンプルな方法
 - 親戚にRS-232CやRS-485がいる

「バイト」とは

- 2進数の0と1を8個並べたものが「1バイト」
- 0~255までの要素を表現できる
- 16進数では0x00~0xFFのように2桁で表現できる

バイトの列を送る

- 0-255までの数字を1個ずつ送るのがシリアル通信
- データの意味や区切りなど、自分でルール(仕様)を決めてやり取りする

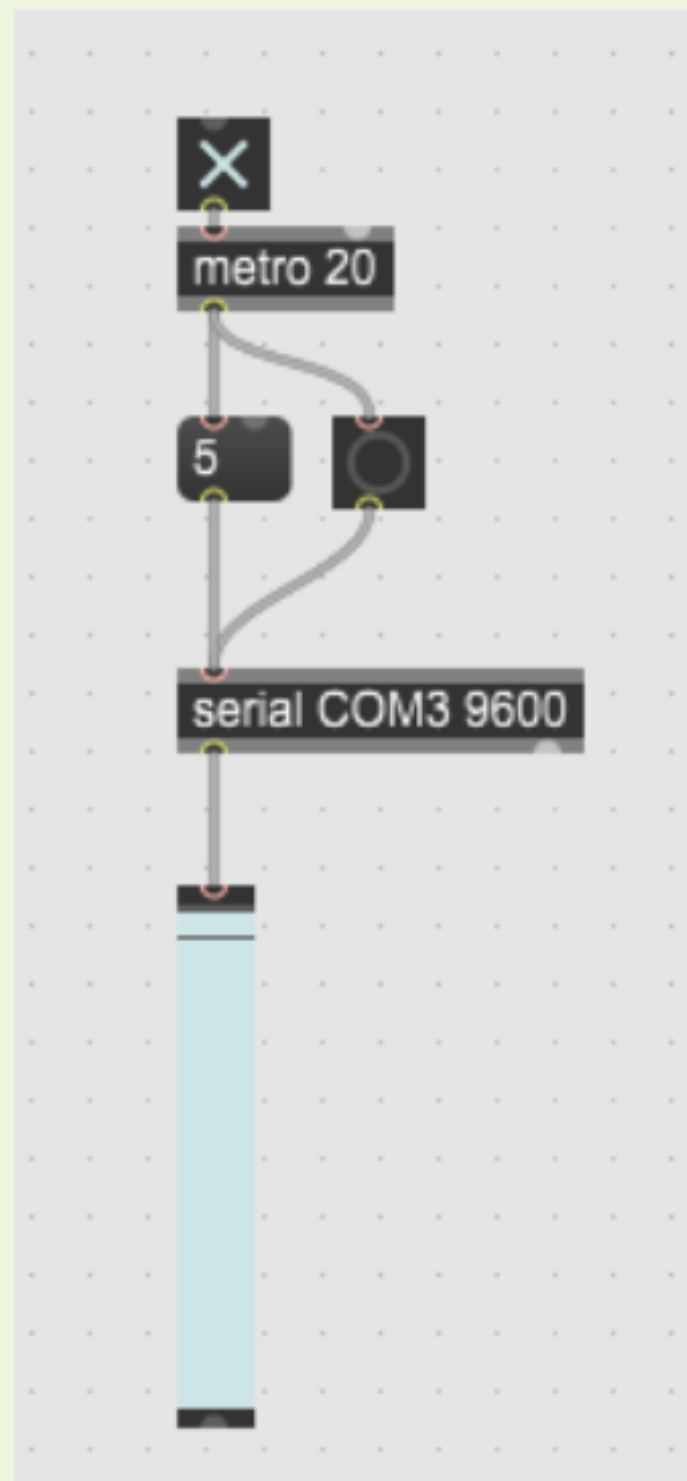
PCとの連携・基本編

- Arduino側と、PCソフト側のプログラムでルール(仕様)を決めながら書く
 - max/msp of Processing touchdesigner

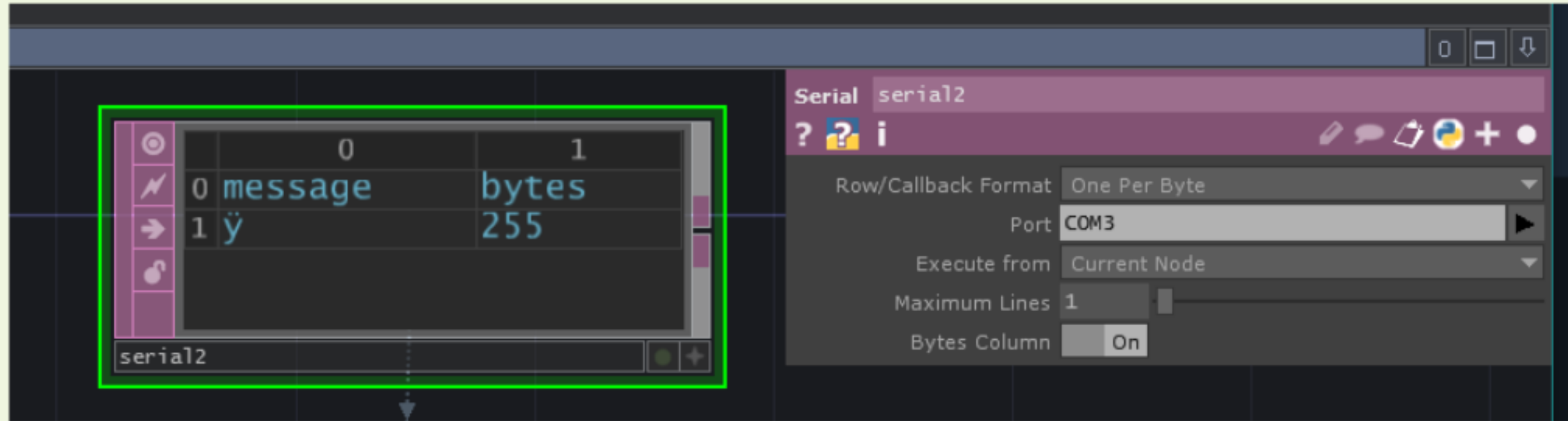
Arduinoシリアル送受信(基礎編)

- 指定したピン番号のアナログ値を返すサンプル

max/mspとの連携



touchdesignerとの連携



- serialDATを使う

- TDはアスキーコードとして使うのがデフォルト(後述)

execute DATからデータ送信

The image shows a Scratch script editor window. At the top, there is a custom block named "execute1" with a "Frame Start" toggle set to "On" and a "Pulse" button. Below this, a script editor is highlighted with a green border, containing the following Python code:

```
1 def onFrameStart(frame):  
2     op('serial2').sendBytes(5)  
3     return  
4  
5  
6
```

The script editor has a toolbar on the left with icons for a target, lightning bolt, arrow, and lock. The bottom of the script editor shows the name "execute1" and a plus sign icon.

CHOPで受け取る

The image shows a software interface with two main components. On the left, a block labeled 'serial12' is configured with two channels. Channel 0 is named 'message' and has a data type of 'bytes'. Channel 1 is named 'ÿ' and has a value of '255'. A dashed arrow points from the 'ÿ' channel to a block labeled 'constant1' below it. The 'constant1' block is highlighted with a green border and contains a slider for 'chan1' with a value of '255'. On the right, a detailed view of the 'Constant constant1' block is shown. It features a slider for 'chan1' with a value of '255'. Below the slider, the 'name0' is 'chan1' and the 'value0' is 'op('serial12')[1, 1]'. The 'Sample Rate' is set to '60'.

Channel	Message	Bytes
0	message	bytes
1	ÿ	255

serial12

constant1

Constant constant1

chan1 255

name0 chan1

value0 op('serial12')[1, 1]

Sample Rate 60

グラフィックのパラメータにしてみる

The image shows a visual programming interface with several interconnected nodes:

- execute1**: A code block containing the following Python code:

```
1 def onFrameStart(frame):  
2   op('serial2').sendBytes(5)  
3   return  
4  
5
```
- serial2**: A node with a table interface:

	0	1
0	message	bytes
1	k	107
- constant1**: A node with a value of 107 and a label 'chan1'.
- express1**: A node with a value of 0.535 and a label 'chan1'.
- null1**: A node with a value of 0.535 and a label 'chan1'.
- noise1**: A node displaying a colorful, noisy texture.

On the right side, the **Noise** node's properties are visible:

- Transform Order: Scale Rotate Translate
- Rotate Order: Rx Ry Rz
- Translate: 0, 0, 0.535
- Rotate: 0, 0, 0
- Scale: 1, 1, 1
- Pivot: 0, 0, 0
- Translate 4D: 0
- Scale 4D: 1

The **noise1** node's transform properties are set to use the output of the **express1** node for the Z-axis translation, as indicated by the expression `op('null1')['chan1']` in the 'Translate' row.

processingとの連携

- **processing.serial**ライブラリを使用
- **serialEvent**メソッドが呼び出される

openFrameworksとの連携

- C++なので、Arduino側と読み書きの構造は近い
- 今日紹介した中で処理速度は最速

USBシリアルでの「決めごと」

- 通信速度(9600/38400/115200bpsなど)

 - データ長・パリティビット・ストップビット

- 8ビット長・パリティ無し・1ストップビット(SERIAL_8N1)がArduinoデフォルト

補足:通信速度

- 9600だと60fpsの世界では1フレーム160バイト
- 115200で1フレーム1920バイトなので無難?

ここまでがシリアル通信基本形

- 生のバイナリ(数字)データでシンプルにやりとり
- 複数種類のデータや大きい数字など複雑にしづらい

前期課題

- デバイスを使った作品プランを提出
- 「1種類のインプットと、それに対応した1種類のアウトプットを持つもの」
- その組み合わせがどういう意味を持つか、どういう意味を持たせられるか考える