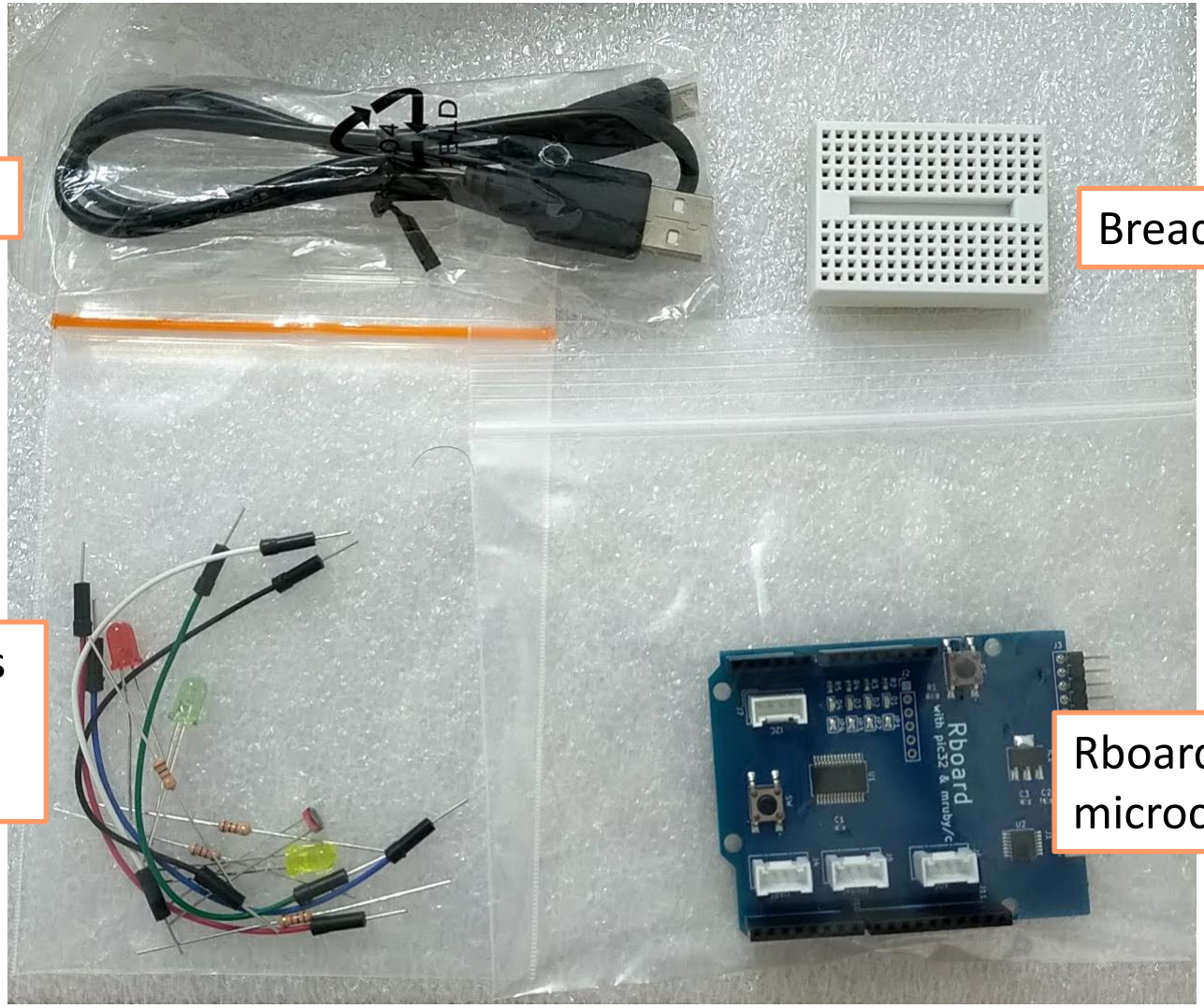
SAKURA Science Workshop

Kazuaki Tanaka

Kyushu Institute of Technology

Setup your environment

Items



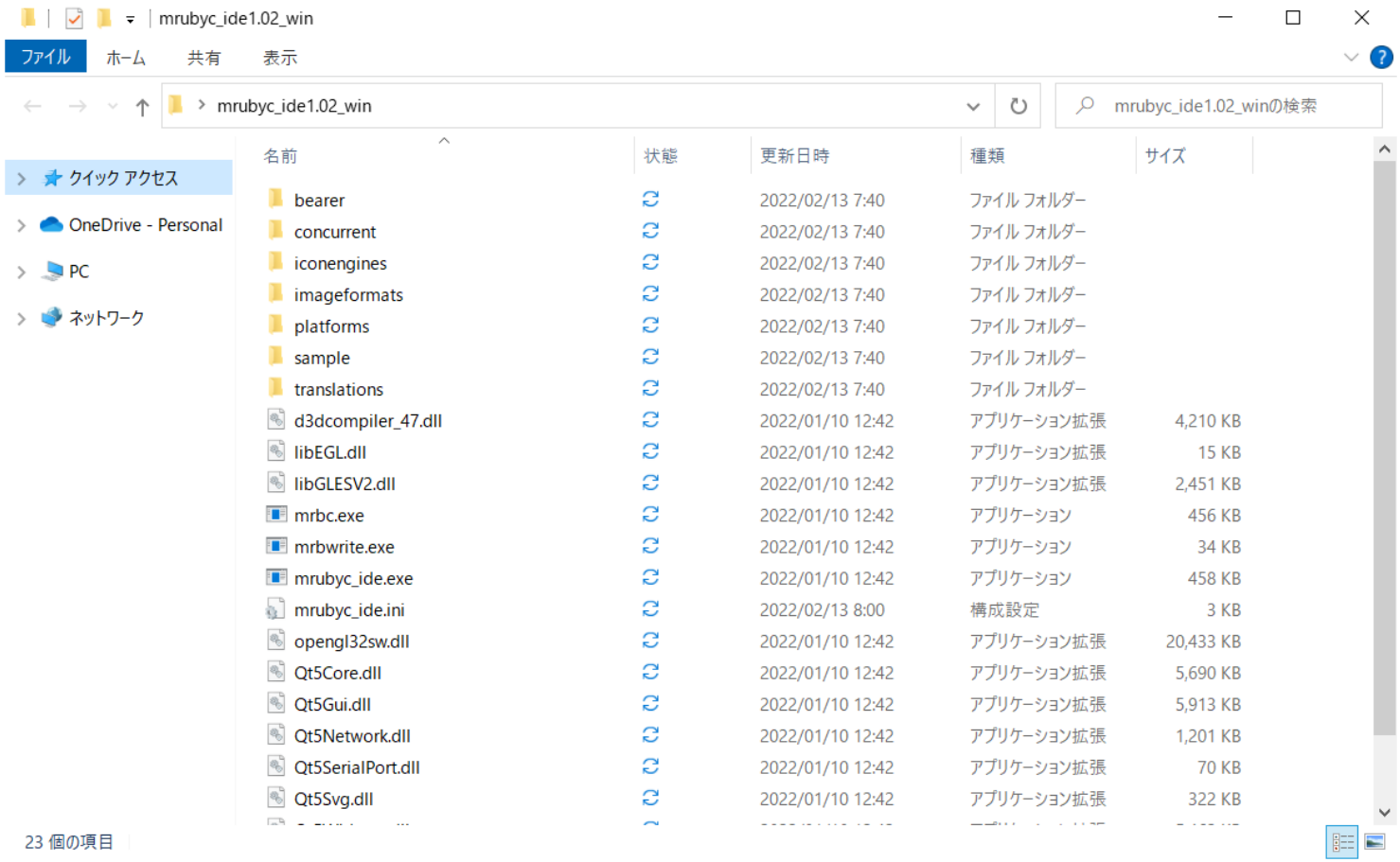
USB cable

Breadboard

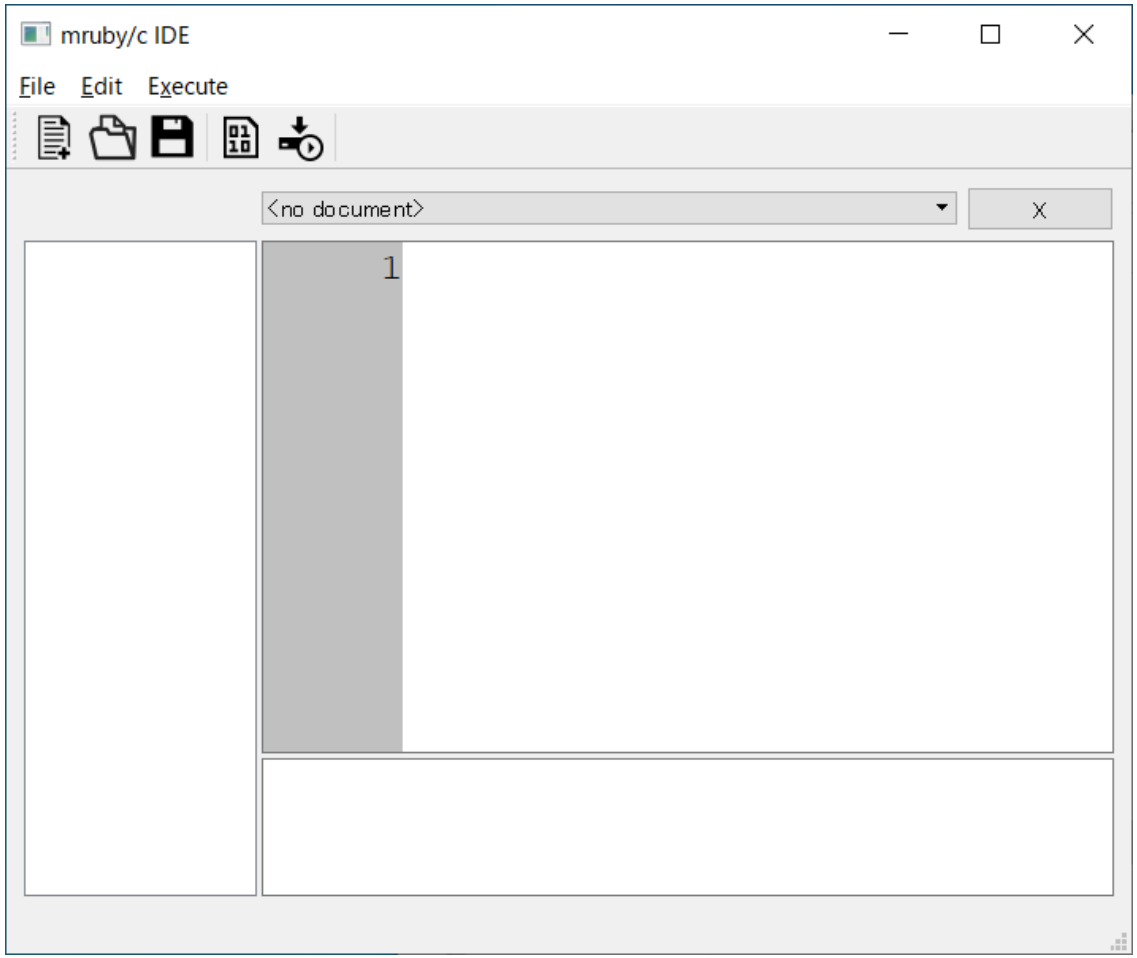
Jumper wires
LEDs
Registors

Rboard
microcontroller

Place extracted files on Desktop



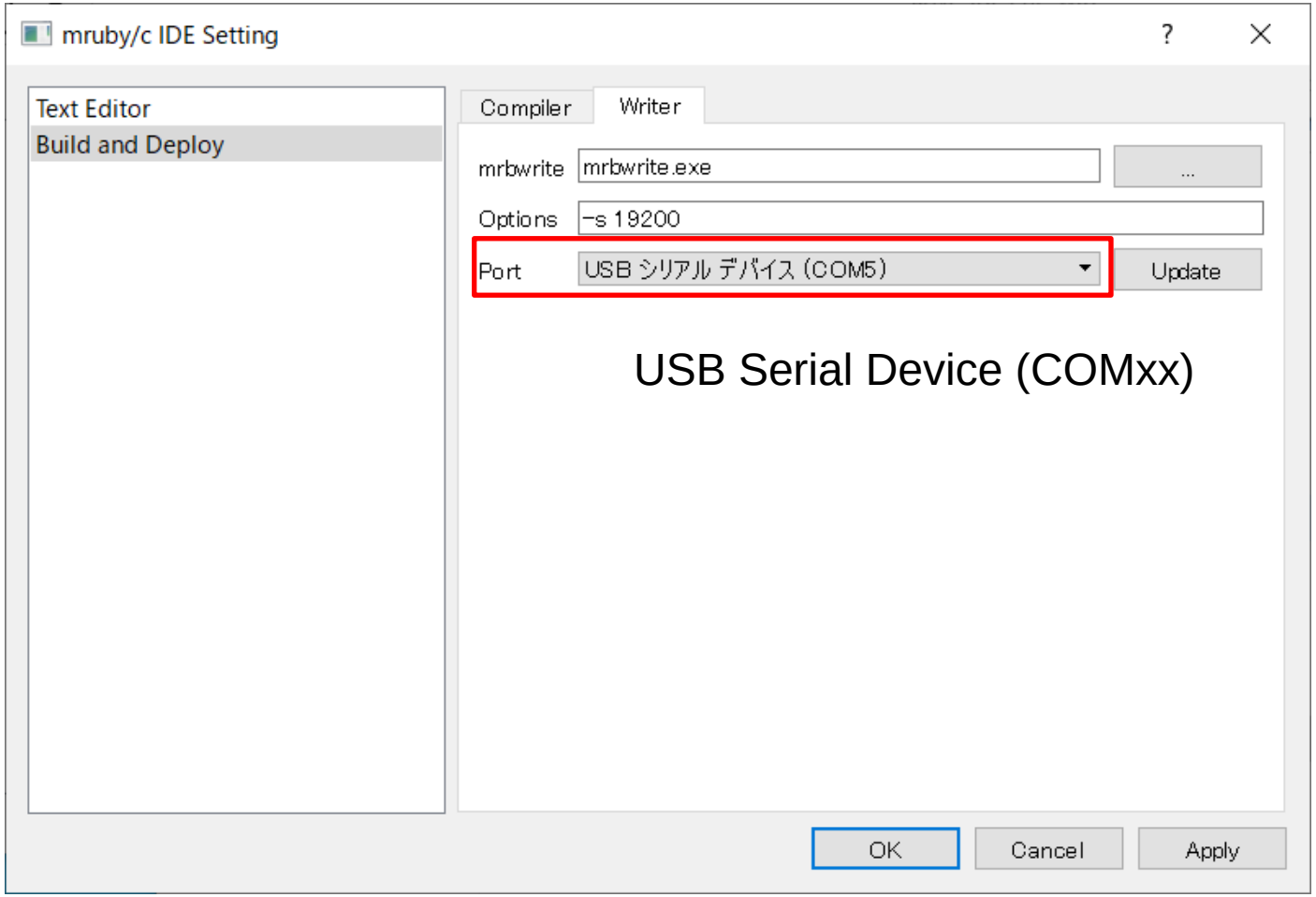
Launch “mrubyc_ide.exe” app



Settings

- Connect USB to Rboard microcontroller board
- [File] → [Settings],
  select [Build and Deploy]
  select [Writer]
- Choose “USB Serial Device (COMxx)” in Port

Port selection



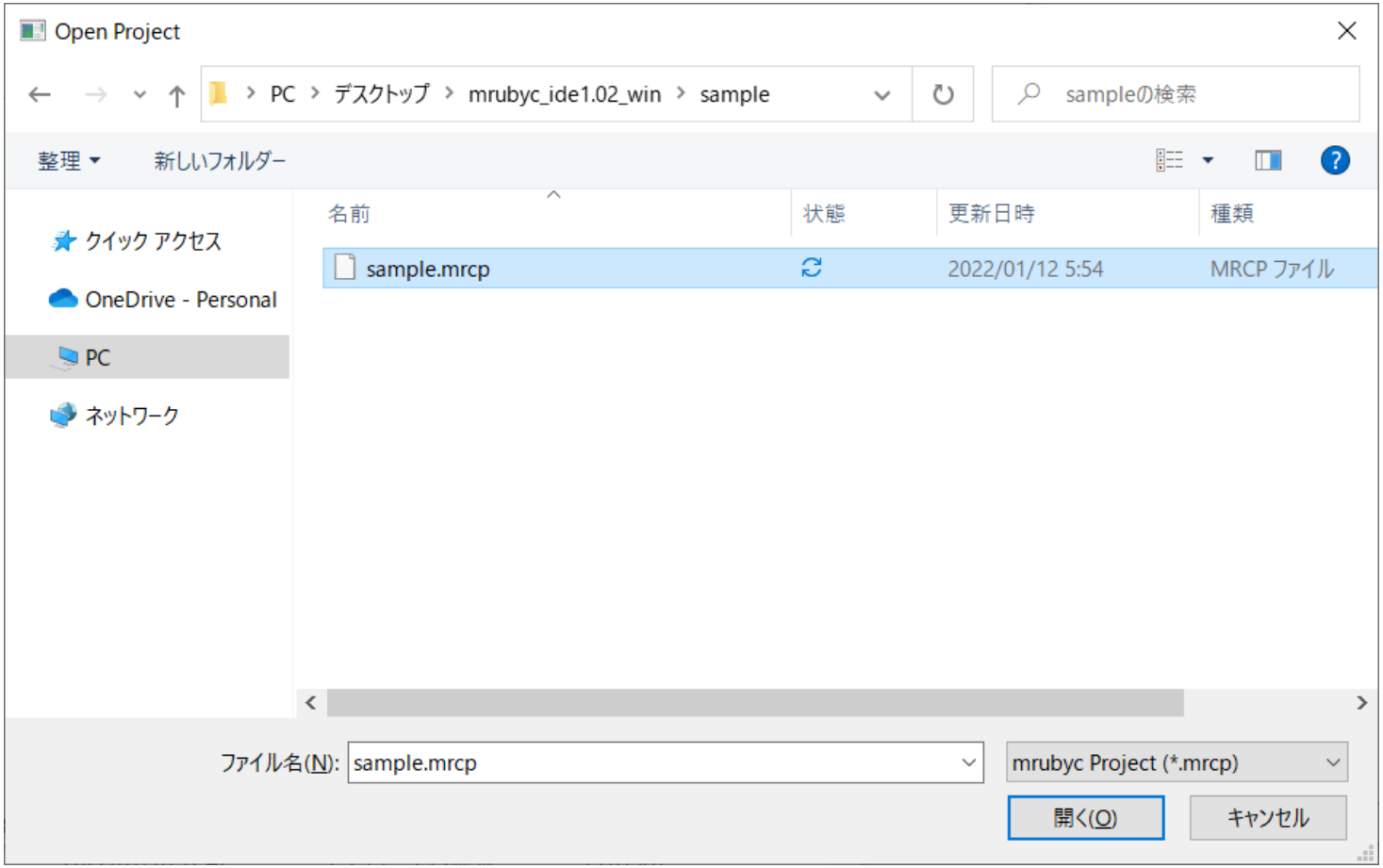
Check your environment

To CHECK your environment

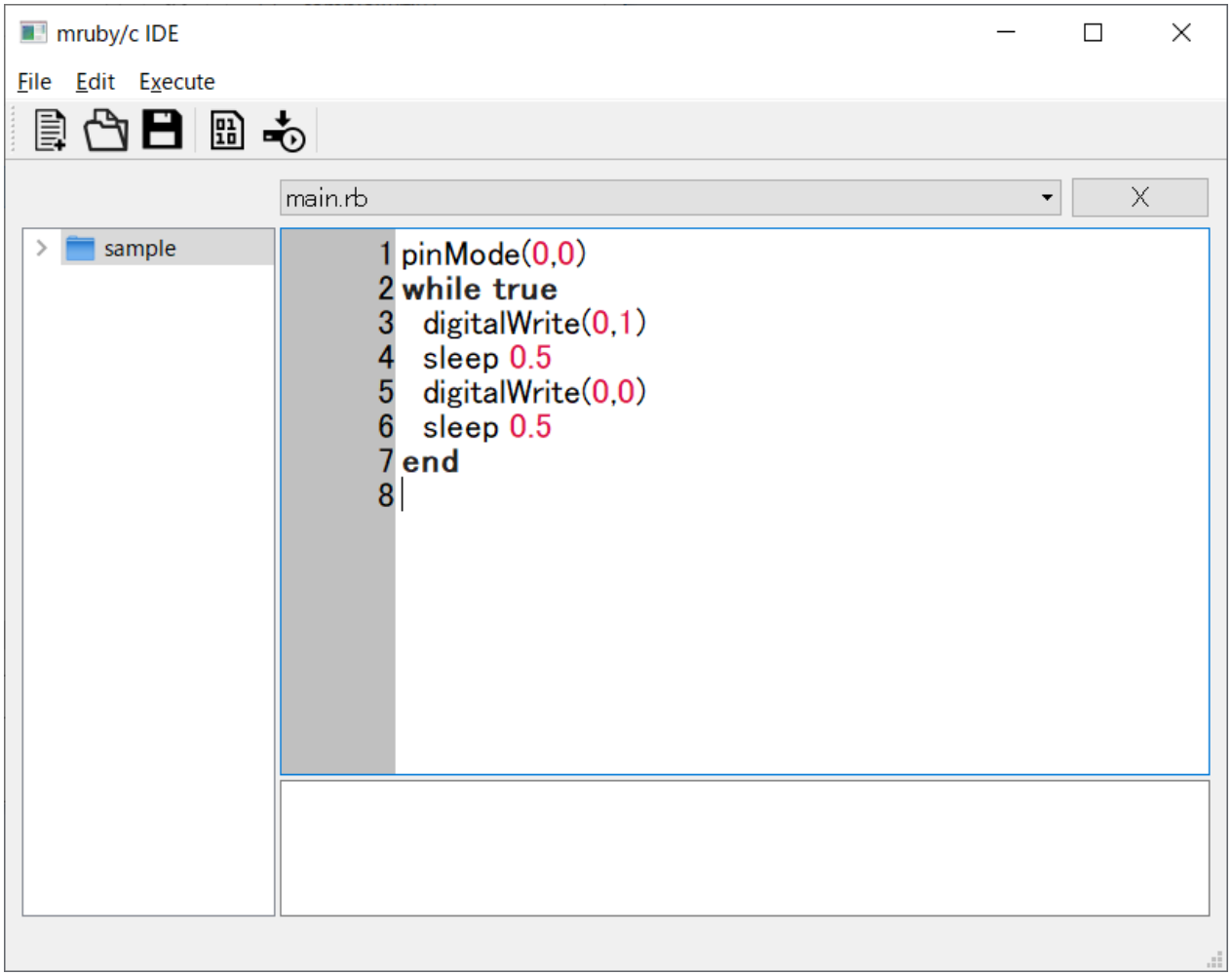
- Execute a quite simple program

- [File] → [Open]
  select “sample” and
  open “sample.mrcp”

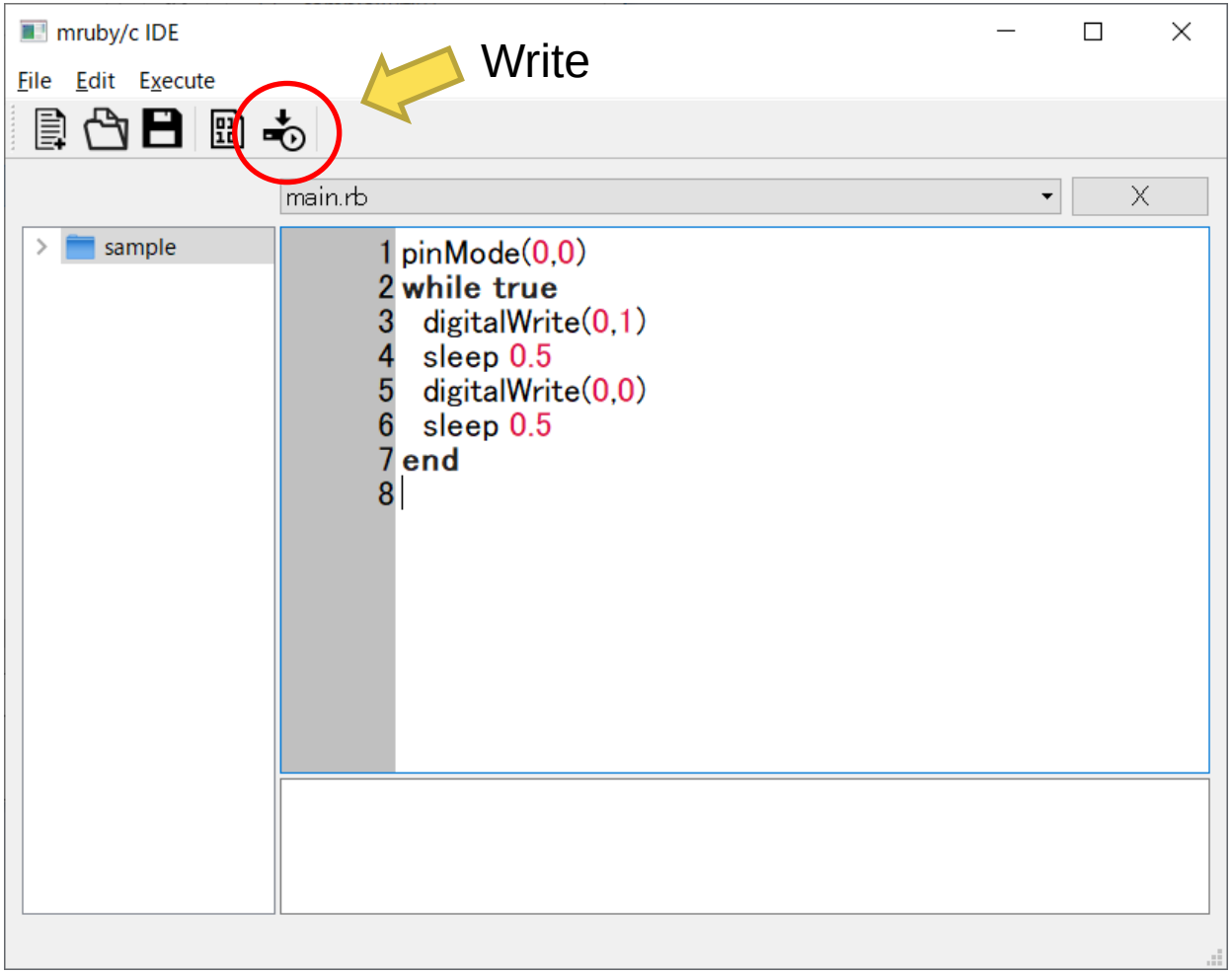
sample.mrcp


Open Project
PC › デスクトップ › mrubyc_ide1.02_win › sample
sampleの検索
整理
新しいフォルダー
クイック アクセス
OneDrive - Personal
PC
ネットワーク
名前
状態
更新日時
種類
sample.mrcp
2022/01/12 5:54
MRCP ファイル
ファイル名(N):
sample.mrcp
mrubyc Project (*.mrcp)
開く(O)
キャンセル

Sample program



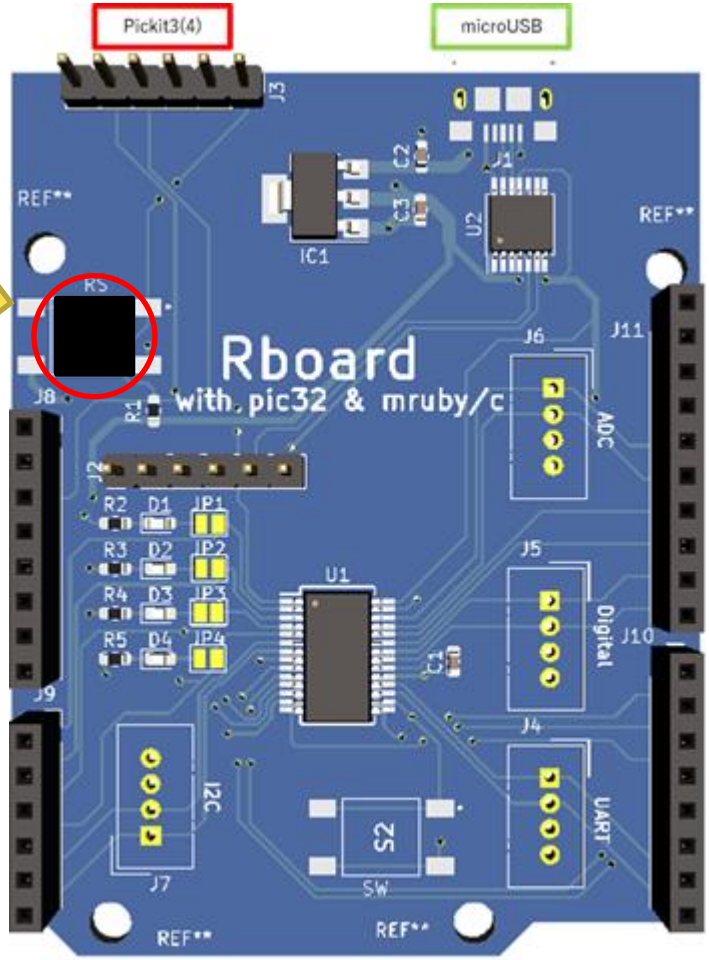
Execute


mruby/c IDE
File Edit Execute
Write
main.rb
sample
1 pinMode(0,0)
2 while true
3 digitalWrite(0,1)
4 sleep 0.5
5 digitalWrite(0,0)
6 sleep 0.5
7 end
8

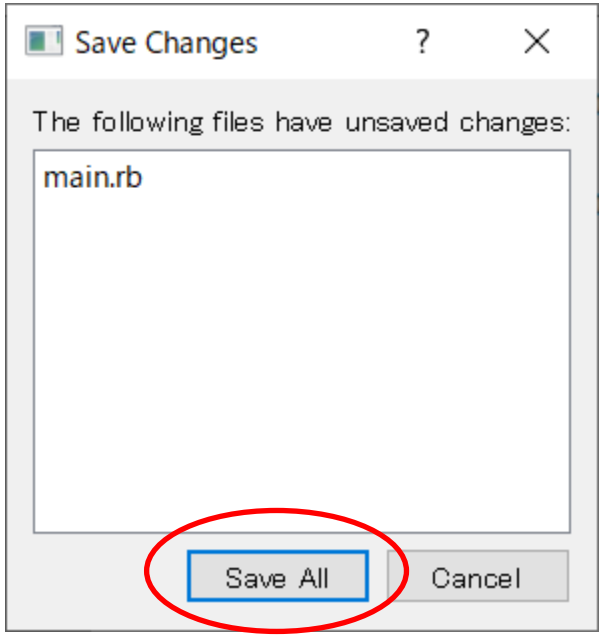
Executing procedure

- Click “Write”
  - IDE will wait for preparation
- Press “RS” button on Rboard
  - Reset button, then accept external program
  - After writing, starts your program

RS button
(Reset)



If you see SAVE dialog window,



Save your program

About this sample program

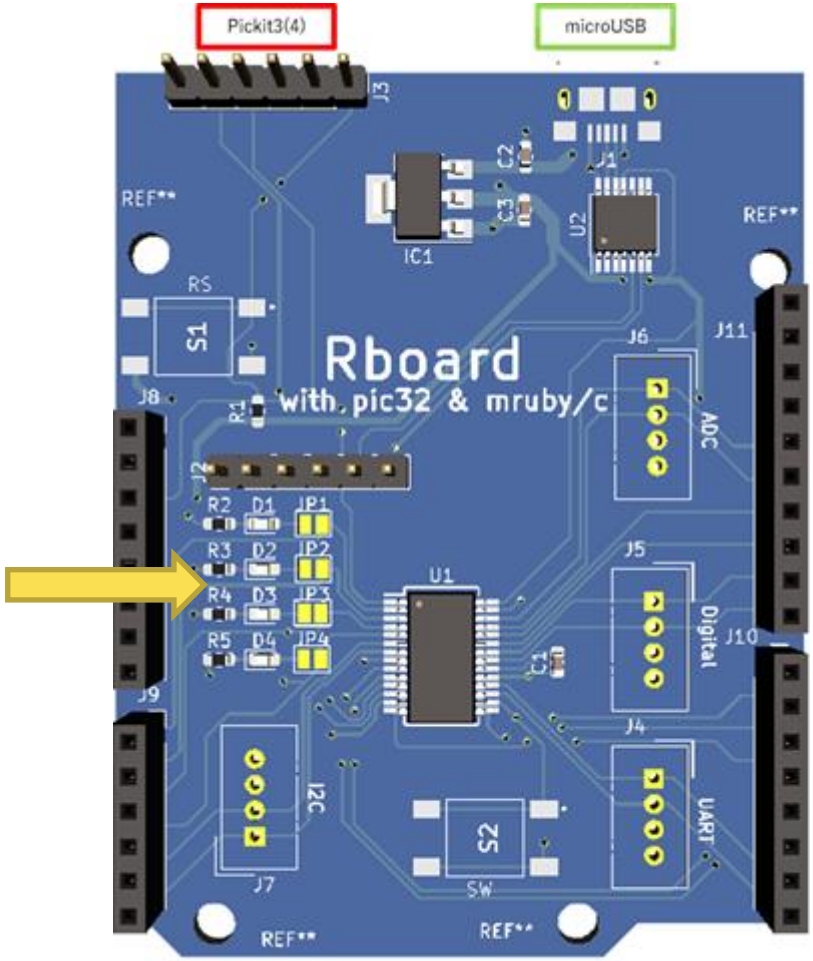
```
pinMode(0,0)
while true
  digitalWrite(0,1)
  sleep(0.5)
  digitalWrite(0,0)
  sleep(0.5)
end
```

Repeat forever

- digitalWrite <pin>, <0or1>
  - Output voltage to PIN(or LED)
- sleep <time>
  - Wait for specified seconds

LED pins

D1: Pin 0
D2: Pin 1
D3: Pin 5
D4: Pin 6



Exercises

Data structure

(sample2)

```
ary = [0, 1, 5, 6]

while true
  ary.each do |led|
    digitalWrite(led, 1)
    sleep(0.9)
    digitalWrite(led, 0)
  end
end
```

Array data structure

For each elements in ary
Each element is assigned to led

Function

(sample3)

```
def blink(led)
  digitalWrite(led, 1)
  sleep(0.1)
  digitalWrite(led, 0)
end

while true
  blink(0)
  blink(1)
  blink(5)
  blink(6)
  blink(5)
  blink(1)
end
```

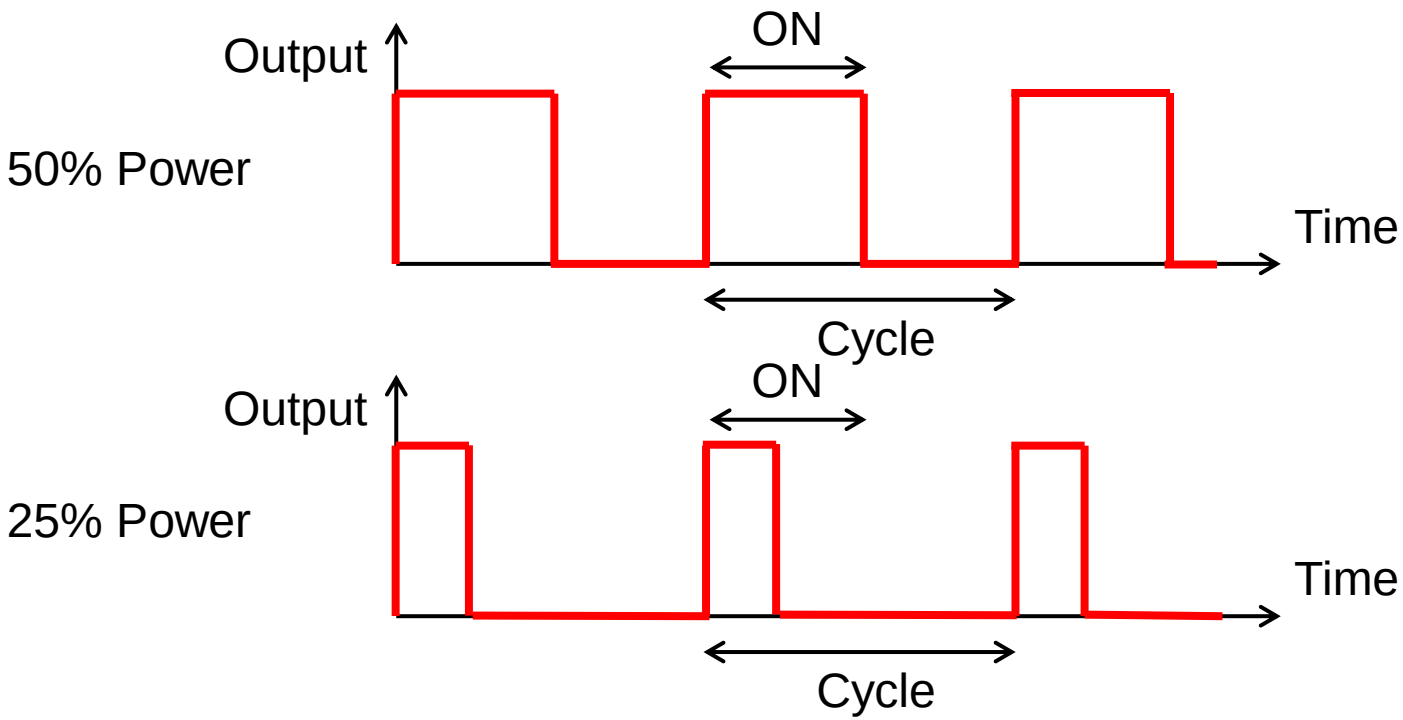
Define function “blink”
parameter: led

Call function “blink”
with an argument

PWM control

Pulse Width Modulation

- Control output power by digital on/off.
  - Total power = ON duration / Cycle



PWM

(sample4)

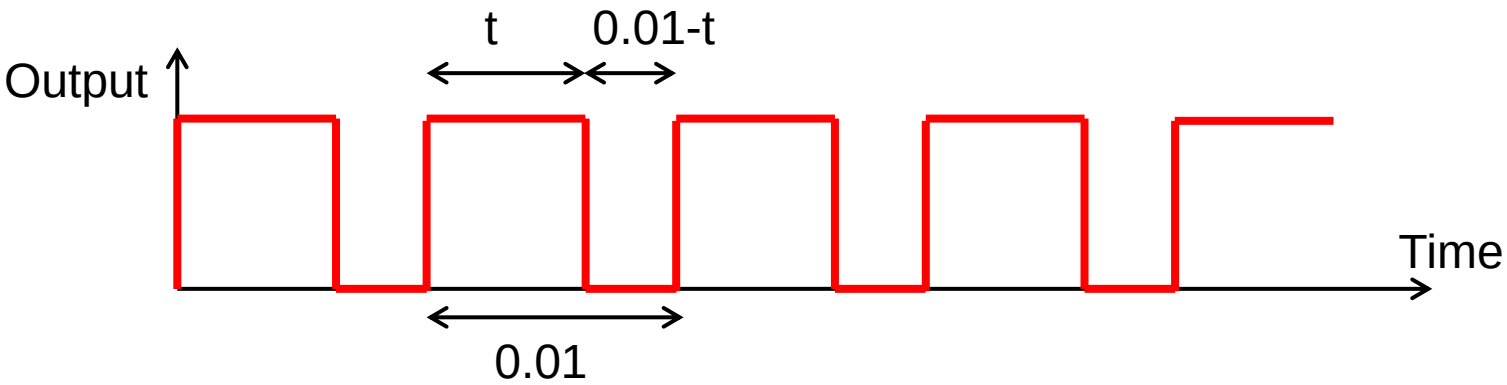
```
def pwm(t)
  100.times do
    digitalWrite(0,1)
    sleep(t)
    digitalWrite(0,0)
    sleep(0.01-t)
  end
end

while true
  pwm(0.001)
  pwm(0.002)
  pwm(0.004)
  pwm(0.009)
end
```

Repeat 100 times

PWM

- Control brightness by highspeed switching




```
digitalWrite(0, 1)
sleep(t)
digitalWrite(0, 0)
sleep(0.01-t)
```

ON time = t
OFF time = 0.01-t

$$\text{Duty ratio} = \frac{t}{0.01}$$

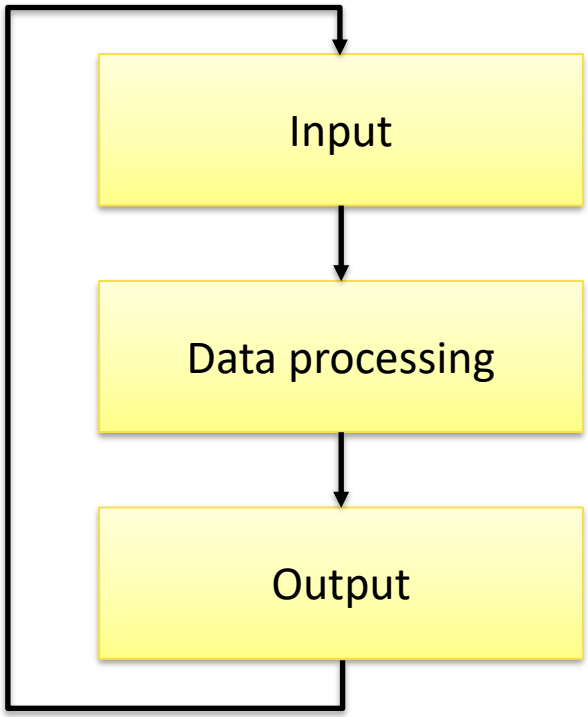
t = 0.001: 10%
t = 0.009: 90%

Multi programming

Multi porogramming

- Execute programs concurrently
- Useful feature in IoT programming
  - Input, Data processing, Output

Single IoT program



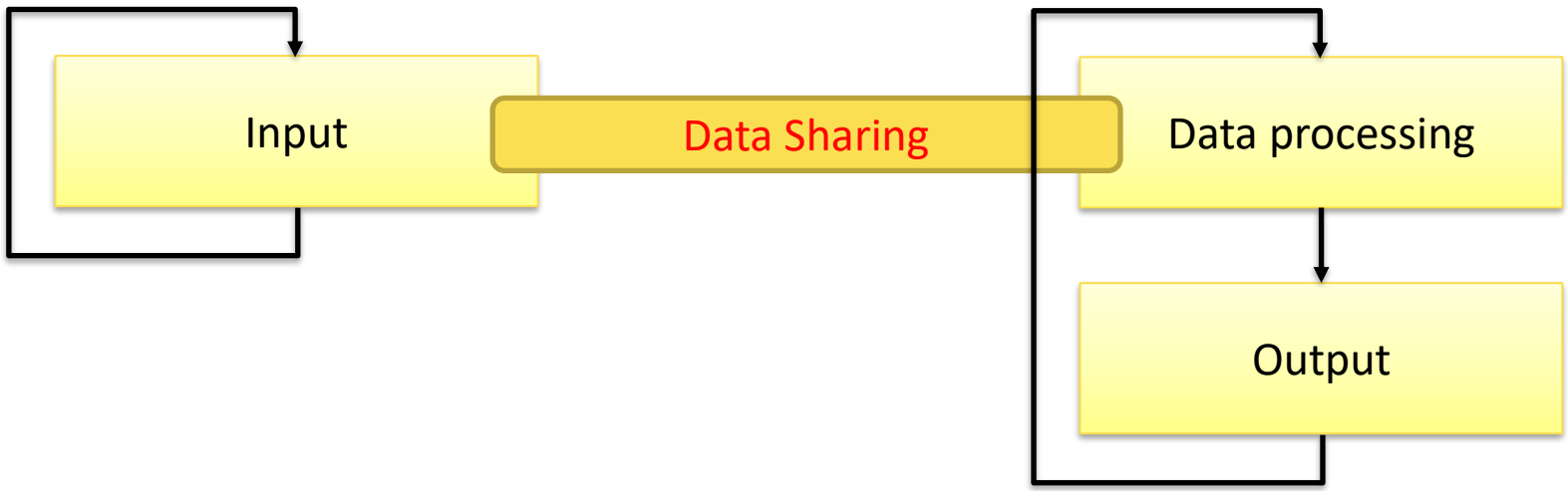
Simple idea, but,

Execution timing is different.

Input timing should depend on sensors, human interface, and so on.

Output timing is usually fixed.

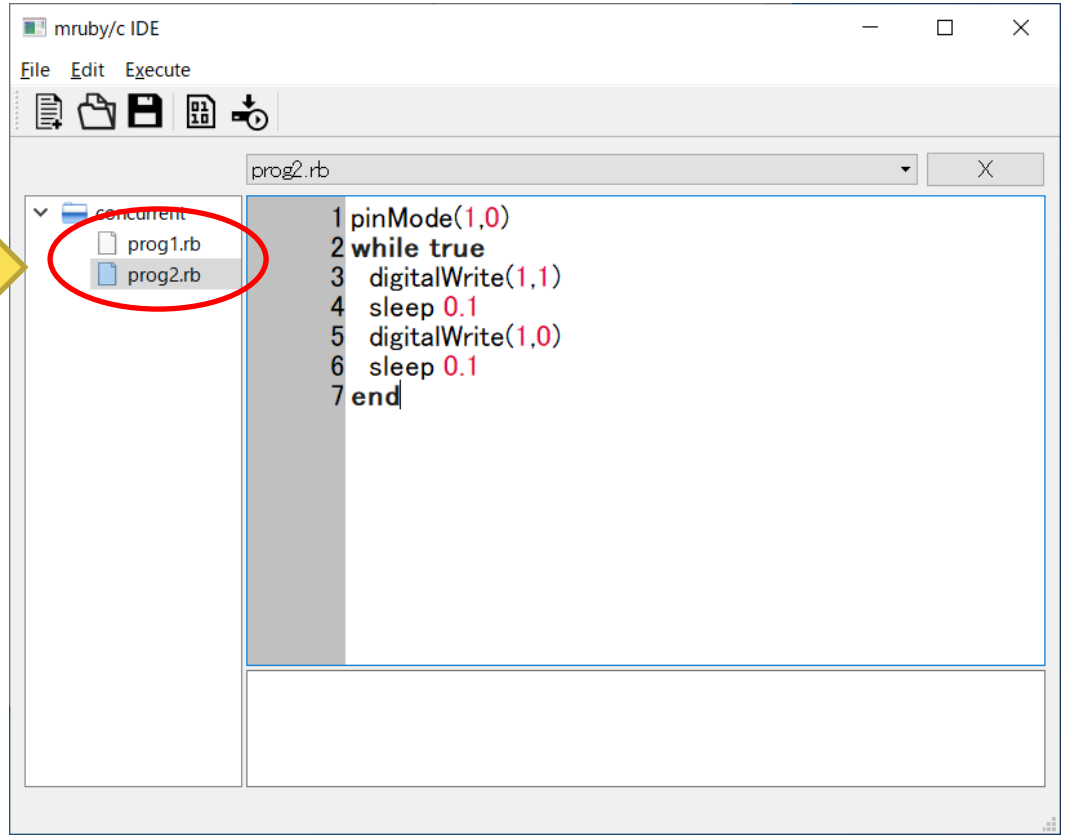
Multi IoT programming



Concurrent execution

- Open “concurrent”

Programs



User interface

```
$s = 1
pinMode(12,1)
while true
  $s = digitalRead(12)
  sleep(0.01)
end
```

Pin 12 is button “SW” on Rborad

```
pinMode(0,0)
while true
  if $s==0 then
    digitalWrite(0,1)
  else
    digitalWrite(0,0)
  end
  sleep(0.01)
end
```

Variable $s is shared between two programs

More complex UI

```
$s = 1
pinMode(12,1)
while true
  if digitalRead(12) == 0 then
    $s = 1-$s
    sleep(0.2)
    while digitalRead(12) == 0
      sleep(0.01)
    end
  end
  sleep(0.01)
end
```

```
pinMode(0,0)
while true
  if $s==0 then
    digitalWrite(0,1)
  else
    digitalWrite(0,0)
  end
  sleep(0.01)
end
```

Day 1 Closing