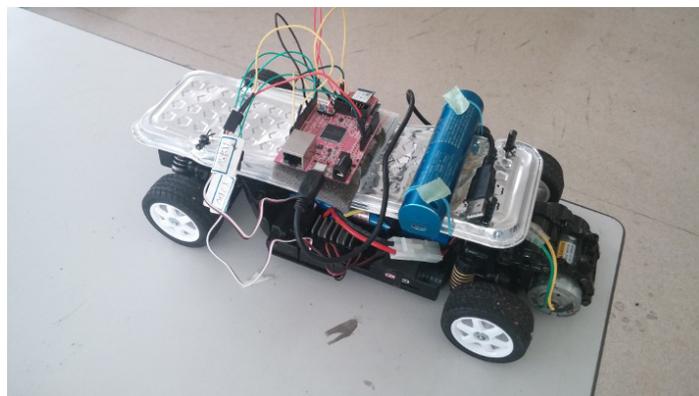


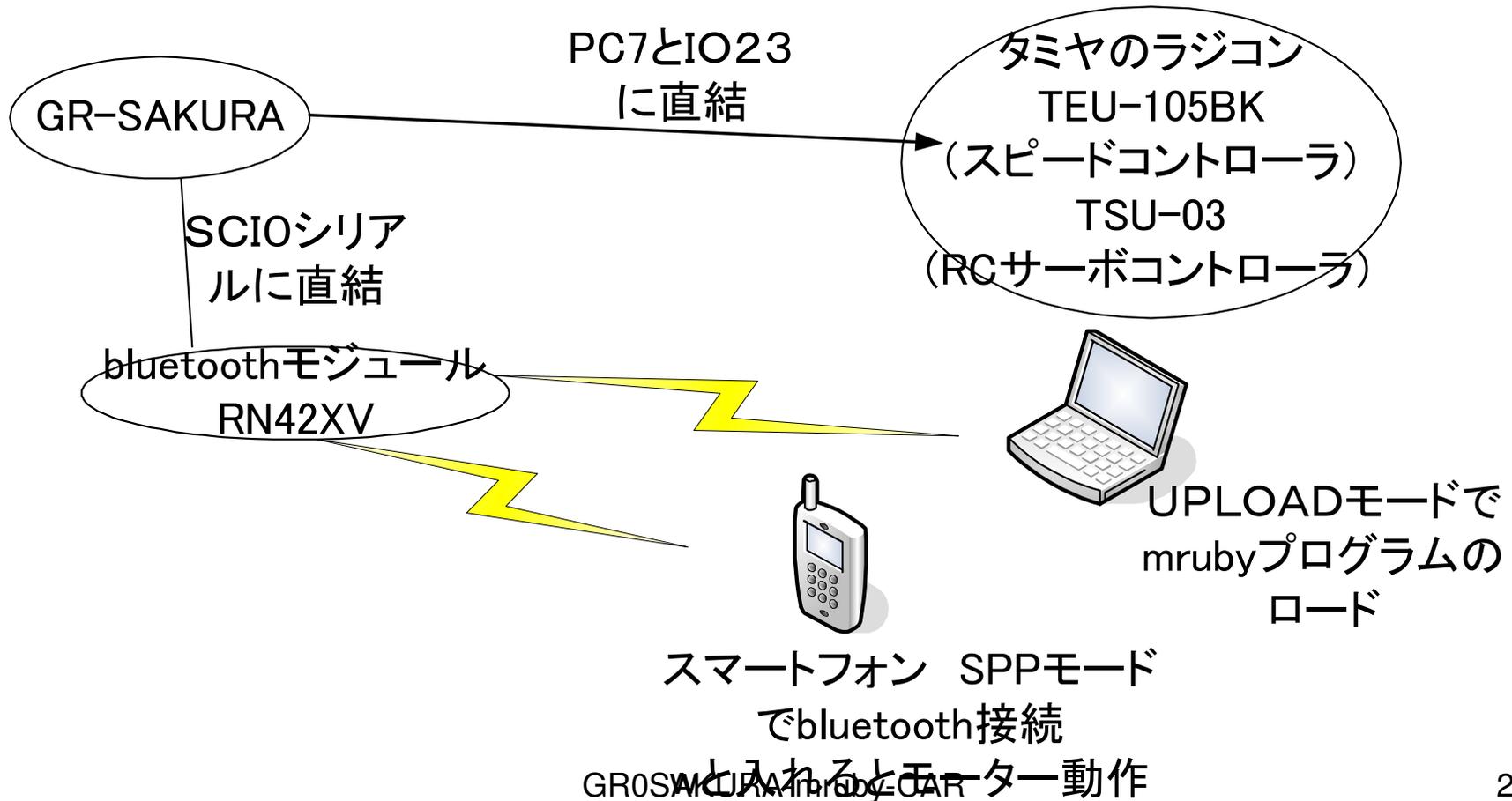
GR-SAKURAで動く mrubyスマホリモコンカー (タミヤ製ラジコン改造)



アライブビジョンソフトウェア株式会社
高橋和浩
TOPPERSプロジェクト特別会員

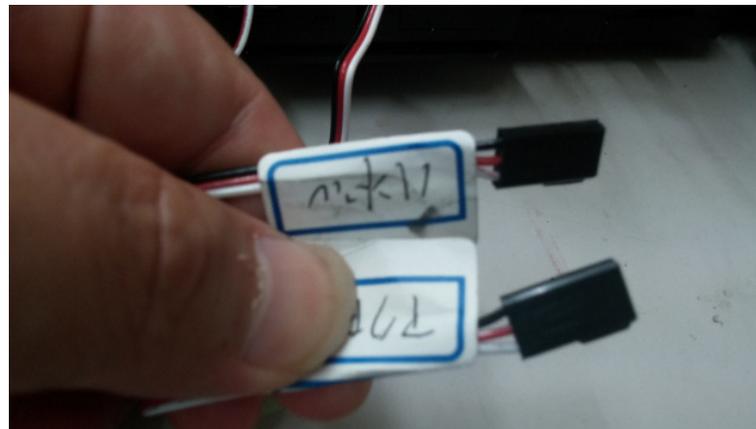
全体構成

mrubyOS環境
マルチスレッドmruby環境
(TOPPERS/SSP最少セットカーネル改造版)



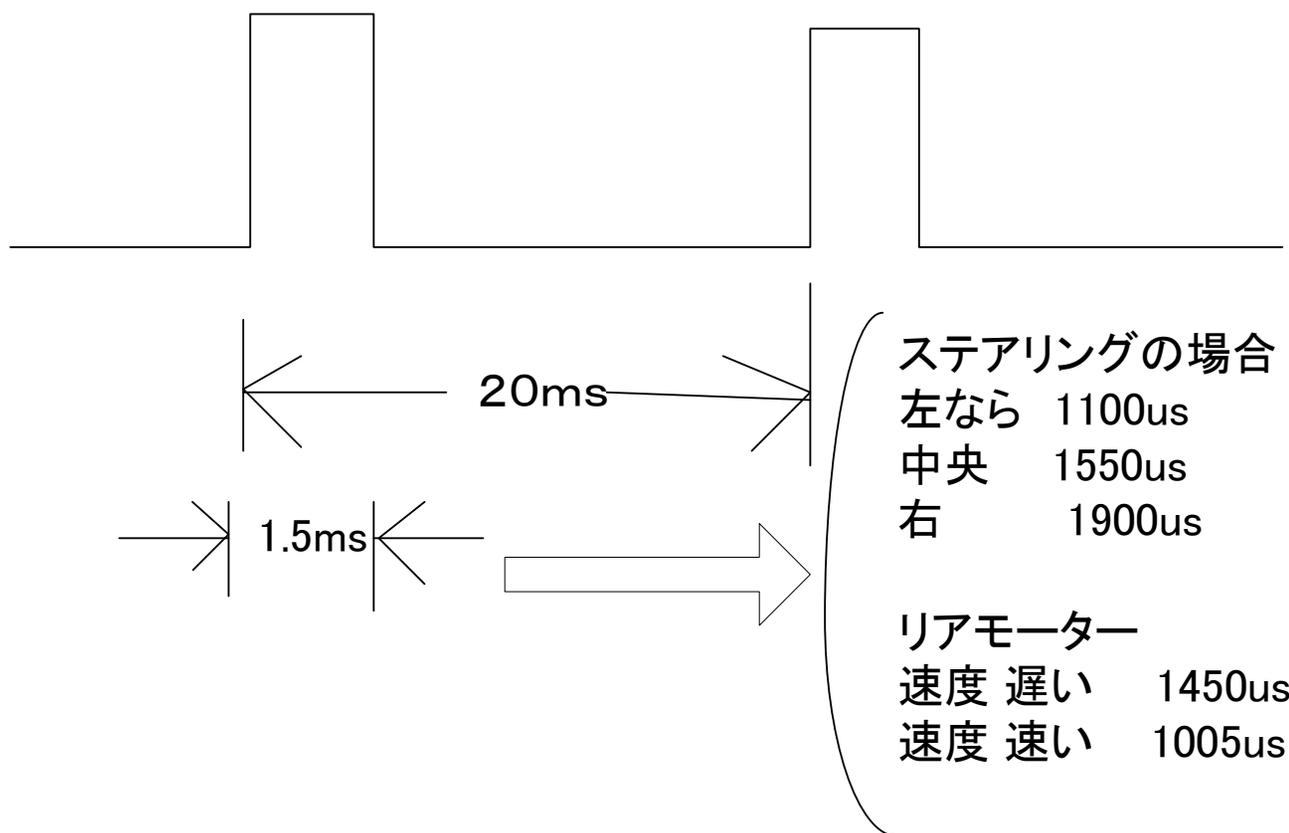
ラジコンカーの説明 接続編

- 工作が苦手な人必見。
- 完成済み1/10スケールのタミヤ製なら、フタバソケットケーブルのひっこ抜くだけ



ラジコンカーの説明 制御編

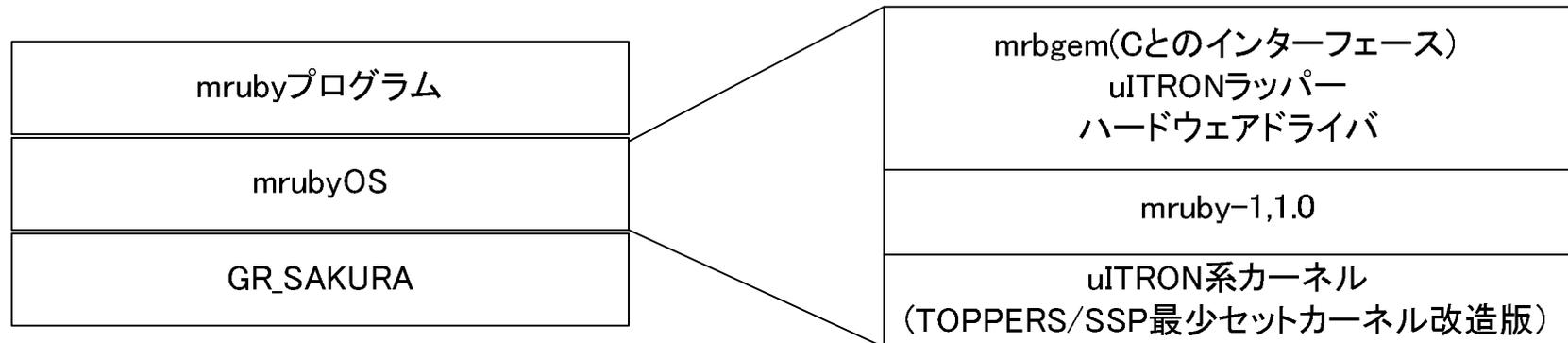
RCサーボの制御方法



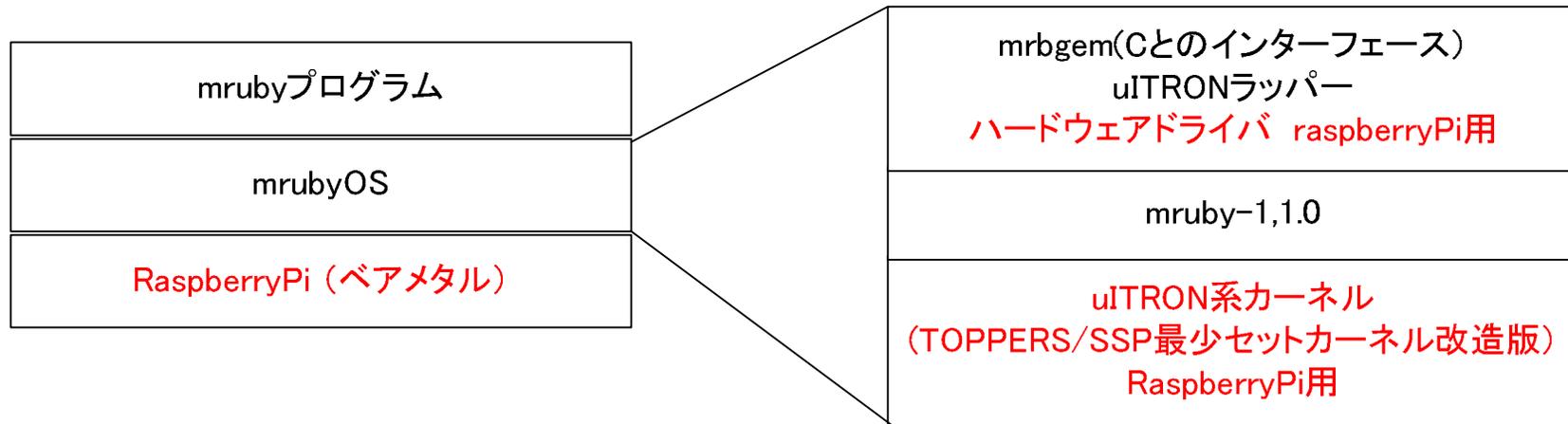
今回の特徴

- mrubyOS (仮称) を搭載しており、アプリケーションはmruby言語をマルチスレッド環境で動作します
- アプリケーションをUPLOADして実行
- UPLOADせず、リンク済みのmrubyの実行も可能

ソフトウェア構成



ソフトウェア構成 raspberryPiの場合



uITRON系カーネル/ハードウェアドライバのRaspberryPi版は
すでにあります。

元々、RaspberryPiのラジコンカーを作っていましたが、今日のために
GR-SAKURA版を用意しました

Youtube「ドリフト raspberry pi」で検索
GR0SAKURA-mruby-CAR

デモ

ソース

```
class Myalarm < Ssp_alarm
  def initialize(id,gpio)
    super("alarm",id)
    @gpio = gpio
  end
  def alarm(n)
    @gpio.off
  end
end
class Gpiocyclic < Ssp_cyclic
  def initialize(id,alarm,gpio,led)
    super("cyclic",id)
    @alarm = alarm
    @gpio = gpio
    @led = led
  end
  def start(tout)
    @tout = tout
    super
    @led.on
  end
end
```

```

def change_tout(value)                                #moterは 100から141
  @tout += value
  p "tout = "+ @tout.to_s
end
def stop
  super
  @alarm.stop
  @led.off
end
def cyclic(n)
  @gpio.on
  @alarm.istart(@tout)
end
end
class MainThread < Ssp_thread
  def initialize(id)
    super("thread",id)
  end
  def thread(n)
    puts "start mainThread"
    led1  = Rx_gpio.new(Rx_gpio::LED1)                #accel
    accel = Rx_gpio.new(Rx_gpio::IO2)
    led1.off
    accel_timer = Myalarm.new(1,accel)
    accel_loop  = Gpiocyclic.new(1,accel_timer,accel,led1)
    stear       = Rx_rcservo.new()
  end
end

```

###-----

```
while 1 do
width = 0
  p "please hit any key!"
  command = Rx_getc()
  p "getc = " + command.to_s
  case command

  when 'w' then                               #accel start
    accel_loop.start(140)
  when 's' then                               #accel stop
    accel_loop.stop
  when '1' then                               # Speed down
    accel_loop.change_tout(1)
  when '3' then                               # speed up
    accel_loop.change_tout(-1)

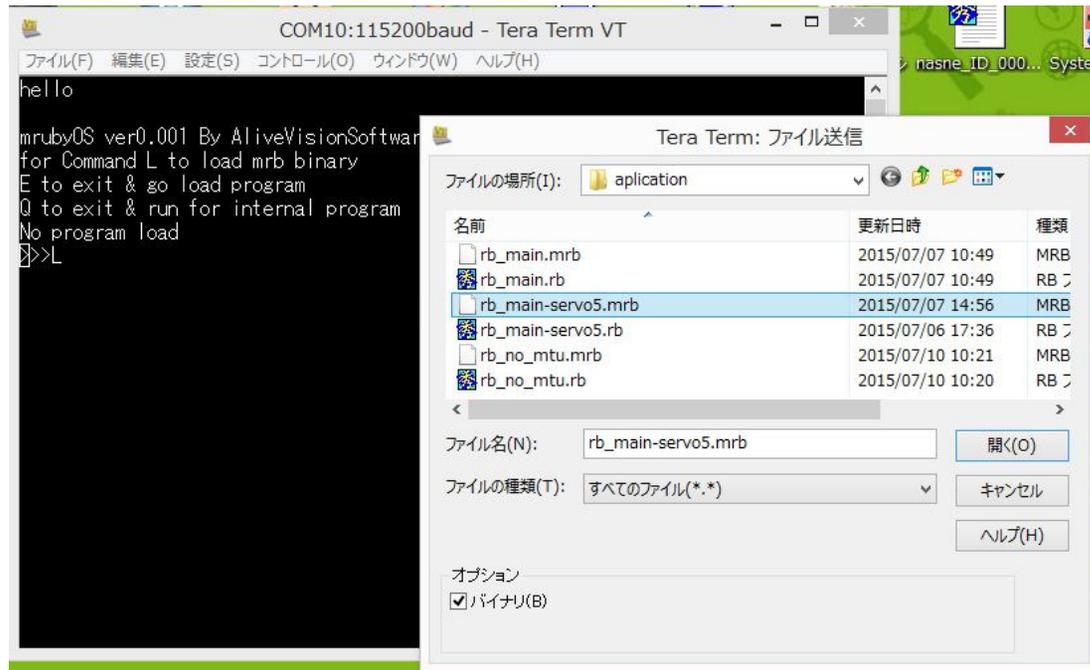
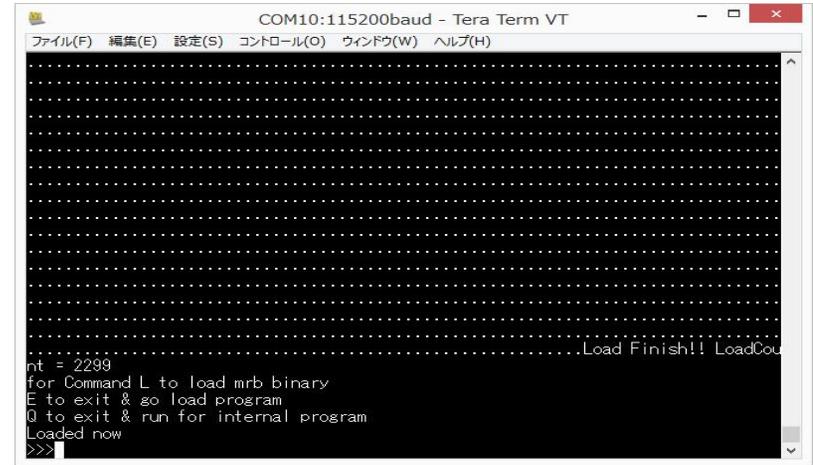
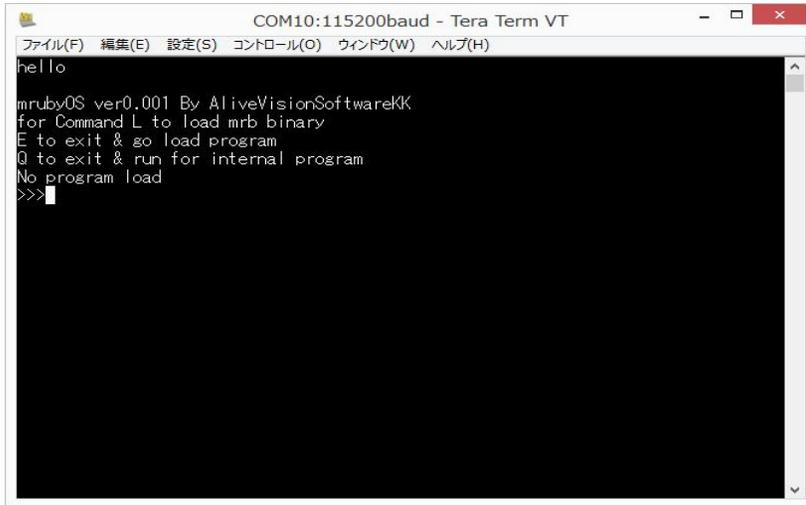
  when '4' then                               # 左
    width = 1050
  when '5' then                               # 中
    width = 1550
  when '6' then                               # 右
    width = 1900
  end
  if width != 0 then
    p "width =" + width.to_s
    stear.Start(width)
  end
  GC.start
end
```

###-----

```
    end
end
GC.interval_ratio = 20
GC.step_ratio = 200
led0 = Rx_gpio.new(Rx_gpio::LED0)
led0.on #動作中
thread_main = MainThread.new(5)
puts "mruby setup finish!"
thread_main.act()
```

mrubyのコンパイル & UPLOAD & 実行

- C:¥>mrbc rb_main-servo5.rb
- 上記の場合 rb_main-servo5.mrbができるのでこれをUPLOADします。
- UPLOADモードで Lコマンドで送信
- Eコマンドで実行



スマホ操作

- BluetoothSPP 無料版を使います。
- キーボードモードでキーをボタンに割り当て可能です。 シリアル接続と同じです。

スマホ画面例

