mruby machine

An Operating System for Microcontroller

HASUMI Hitoshi @hasumikin

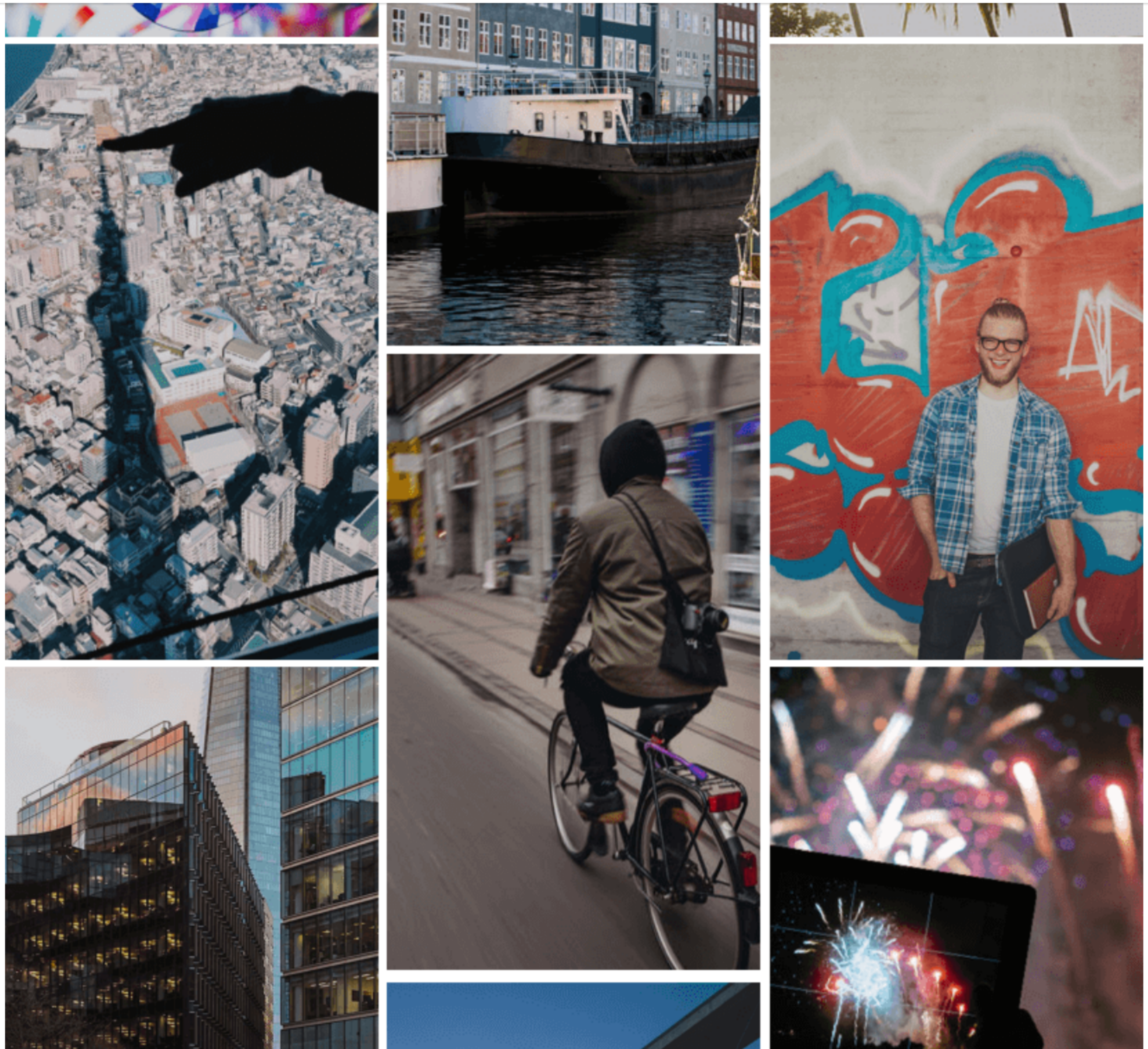
September 4-5, 2020

Me

- HASUMI Hitoshi（羽角 均）
  @hasumikin（ハスミキン）
- From Ruby City Matsue,
  a holy place of 🍶（熱燗）
- Developer on Rails
  （レールに乗った開発者）
- Coffee ☕
  Curry 🍛
  Soba (NoEmojiError)

With offices in 26 cities around the world, we know your market

Tokyo, Osaka, Matsue, Chengdu, Qingdao, Shanghai, Beijing, Hanoi, Danang, Singapore, Dhaka, Cebu, Manila, Dubai, Copenhagen, Arhus, London, Manchester, Amsterdam, Prague, Bangkok, Berlin, New York,



Chapter 1

Background

Motivation 😲 MicroPython

Generally, we keep minimal configuration of MicroPython under ***80K*** *of ARM Thumb2 code (which includes* ***compiler*** *and* ***interactive prompt****, ...). That means a Cortex-M microcontroller with* ***128KB*** *of* ***flash*** *can host a minimal version together with some hardware drivers.*

[cited from `https://github.com/micropython/micropython/wiki/FAQ']

Motivation 😐 MicroPython

*Regarding **RAM** usage, (...) **8KB** is minimal amount to run simple scripts. As Python is interpreted high-level language, the more memory you have, the more capable applications you can run. The reference MicroPython board, PyBoard, has **128KB** of RAM.*

[cited from `https://github.com/micropython/micropython/wiki/FAQ']

Motivation 😕 mruby and mruby/c

- Flashing mruby application into 128KB ROM and running it on 128KB RAM are difficult
- mruby/c runs on 64KB RAM but it doesn't have mruby compiler
  - You have to compile Ruby script into VM code before embedding

Motivation 😖

MicroPython:
"Overwhelming, we are!"

「圧倒的じゃないか、我が軍は」

Motivation 🤔

Would it be great if we have a smaller mruby compiler?

Target Microcontroller

- PSoC5LP: Reference board of mruby/c
  - Arm Cortex-M3 32bit processor
  - 64KB RAM
  - 256KB ROM
- It will be comparable to MicroPython if Ruby runs on PSoC5LP with compiler

Demo

Chapter 2

Terms and Tools

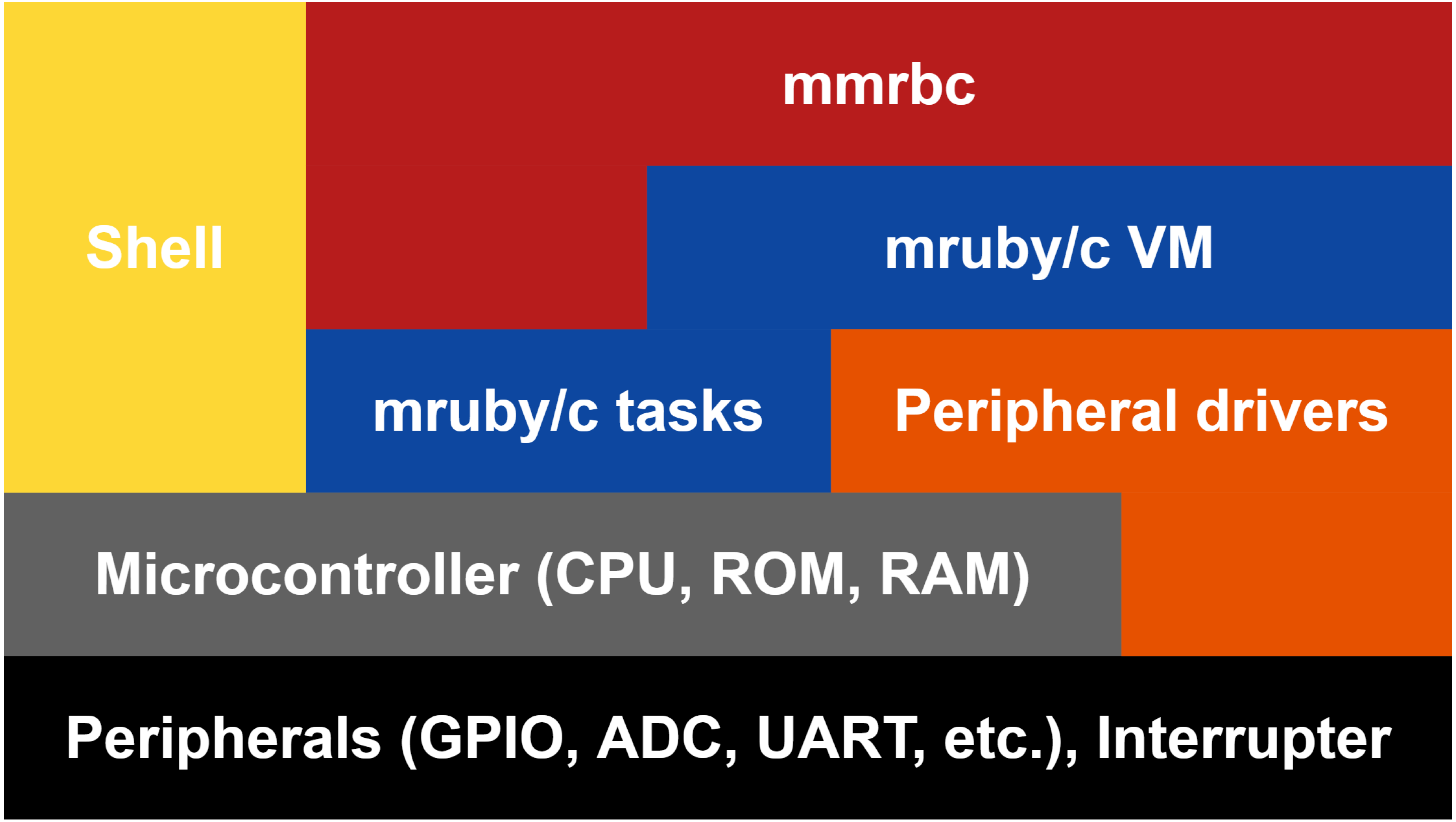
Terms

- mruby
  - A Ruby implementation for general embedded purpose
- mrbc
  - The original mruby compiler bundled with mruby
- mruby/c
  - Another implementation of mruby VM for one-chip microcontroller
  - It doesn't have compiler

Terms

- mmrbc (mini mruby compiler)
  - Another mruby compiler for one-chip microcontroller
- mmruby (mini mruby) = mmrbc + mruby/c
- mruby machine
  - A microcontroller embedded with integration of mmruby, CUI shell and peripheral drivers

mruby machine


mmrbc
Shell
mruby/c VM
mruby/c tasks
Peripheral drivers
Microcontroller (CPU, ROM, RAM)
Peripherals (GPIO, ADC, UART, etc.), Interrupter

Cross compilation tools for bare metal Arm

- arm-none-eabi-gcc, arm-none-eabi-ar, arm-none-eabi-ld, arm-none-eabi-strip, etc.
- What is "arm-none-eabi-" ?
  - arm: Arm architecture
  - none: Without OS (Bare metal)
  - eabi: Embedded Application Binary Interface

Three tool-chains in my laptop (Intel 64)

1. (x86_64-linux-gnu-)gcc
   Usual compilation in daily development
2. arm-linux-gnueabi-gcc
   Binary runs on QEMU (machine emulator)
3. arm-none-eabi-gcc (--with-newlib)
   To make a final library archive (libmmrbc.a) for bare metal Arm

Docker image

(('tag:center'))

https://hub.docker.com/r/hasumikin/arm-none-eabi-tools

C Library

- glibc (GNU C Library)
- Newlib
  - Another C library especially for embedded system
  - Some implementation is missed. eg) regex.c
- Newlib-nano
  - Even smaller version of Newlib for Arm processor
- Yet other libraries
  - EGLIBC, Clibc, sglibc, etc.

Chapter 3

mmrbc

Repo

https://github.com/hasumikin/mmruby

Features of mmrbc

- Portable and Less dependent
  - Doesn't depend on mruby nor mruby/c
  - (To the contrary, mrbc depends on mruby itself)
  - Depends on only standard C library including Newlib(-nano)
  - Can be integrated with both mruby and mruby/c

Features of mmrbc

- Small (by linking to Newlib-nano)
  - Fits in less than 256KB ROM (including mruby/c VM and some application code)
  - Runs enough on less than 128KB RAM

Features of mmrbc

- **Lemon** as a parser generator instead of Yacc/Bison
  - Originally a parser generator of SQLite

Yacc/Bison

- LALR(1) parser generator, typical choice of parser generator
- Yacc (Barkley Yacc)
  - Non-reentrant
  - Parser-Calls-Lexer style (CRuby and mruby use it)
- Bison (Gnu Bison. Modernized version of Yacc)
  - Reentrant, Thread-Safe and Lexer-Calls-Parser style
  - (You can also select old Yacc style)

Lemon

- Reentrant
- Lexer-Calls-Parser style

Parser-Calls-Lexer style (Yacc)

```
yyparse(p);

/* in y.tab.c (generated by Yacc) */
int yyparse(parser_state *p) {
  yytoken = yylex(yylval, p);
  /* makes AST according as you defined */
  ...
  goto yyhogefuga;
  ...
}

/* yylex() is called by parser  */
int yylex(void *lval, parser_state *p) {
  # Find next token and return token type
  return tokenType;
}

GenerateVMCode(p->ast);
```

Lexer-Calls-Parser style (Lemon)

```
/* in your own Lexer */
while (hasMoreToken(rubyScript)) {
  token = getToken(rubyScript);
  Parse(parser, tokenType, token);
}
Parse(parser, 0, "");

/* in parse.c (generated by Lemon) */
void Parse(void *yyp, int yymajor, ...) {
  /* makes AST according as you defined */
  ...
}

GenerateVMCode(p->ast);
```

Lemon

- LALR(1) parser generator as well
- Reentrant
- Lexer-Calls-Parser style
- Generated parser is, I'm still not sure though,
  - Faster than Yacc/Bison (according to its document)
  - Smaller code (according to someone)
- Grammar syntax of parse.y is different from Yacc/-Bison

Lemon

https://sqlite.org/src/doc/trunk/doc/lemon.html

Chapter 4:

How smaller than mrbc?

RAM consumption

- mruby's branch
  - `mruby3` as of August 14th

RAM consumption

- To make things simple, both mrbc and mmrbc are
  - built for x86_64 architecture
  - using `malloc()` and `free()` of glibc
    - (mmrbc usually uses `mrbc_alloc()` to manage heap area)
  - Linux file system to load Ruby script
    - (microcontroller doesn't have file system)
- As a result, we can simply compare with **valgrind**

RAM consumption

- Analysis command looks like:

```
valgrind \
  --tool=massif \
  --stacks=yes \
  path/to/(mrbc|mmrbc) path/to/script.rb
```

- Showing the result:

```
ms_print massif.out.xxxx | less
```

RAM consumption: Case 1

Ruby code:

```ruby
# hello_world.rb
puts "Hello World!"
```

Output:

```
Hello World!
```

RAM consumption: Case 1 / mrbc

```
--------------------------------------------------------------------------------
Command:            ../mruby/build/host/bin/mrbc ./hello_world.rb
Massif arguments:   --stacks=yes
ms_print arguments: massif.out.4087
--------------------------------------------------------------------------------
    KB
157.8^                                                                       #
     |                                                                       #
     |                                                                       #
     |                                                                       #
     |                                                                       #
     |                                                                      @#
     |                                                                      @#
     |                                                                      @#
     |                                                                    @@@#
     |                                               @@:::::::::::@:::::::@ @#
     |                                       @:::::::@ :::: :: :: @: :: ::@ @#
     |                            :::::@:::::@: :: ::@ :::: :: :: @: :: ::@ @#
     |       ::::::::::::::::::::::: : @:: ::@: :: ::@ :::: :: :: @: :: ::@ @#
     |     ::::: ::: ::: ::: : : ::: : @:: ::@: :: ::@ :::: :: :: @: :: ::@ @#
     |     : ::: ::: ::: ::: : : ::: : @:: ::@: :: ::@ :::: :: :: @: :: ::@ @#
     |     : ::: ::: ::: ::: : : ::: : @:: ::@: :: ::@ :::: :: :: @: :: ::@ @#
     |     : ::: ::: ::: ::: : : ::: : @:: ::@: :: ::@ :::: :: :: @: :: ::@ @#
     |     : ::: ::: ::: ::: : : ::: : @:: ::@: :: ::@ :::: :: :: @: :: ::@ @#
     |     : ::: ::: ::: ::: : : ::: : @:: ::@: :: ::@ :::: :: :: @: :: ::@ @#
     |     : ::: ::: ::: ::: : : ::: : @:: ::@: :: ::@ :::: :: :: @: :: ::@ @#
   0 +----------------------------------------------------------------------->Mi
     0                                                                   3.235
```

RAM consumption: Case 1 / mmrbc

```
--------------------------------------------------------------------------------
Command:            ../mmruby/build/host-production/bin/mmrbc ./hello_world.rb
Massif arguments:   --stacks=yes
ms_print arguments: massif.out.1652
--------------------------------------------------------------------------------
    KB
11.17^                                                                #
     |                                                               @#
     |                                                              @@#  ::
     |                                                              @@#:::::
     |                                            @     @  :@: : :: @@#:::::
     |                                            @  :::@: :@::: :::@@#:::::
     |                                            @  :::@: :@:::@:::@@#:::::
     |                                            @:::::@:::@:::@:::@@#:::::
     |                                            @: :::@:::@:::@:::@@#:::::
     |      ::                                    @: :::@:::@:::@:::@@#::::::
     |      ::                                    @: :::@:::@:::@:::@@#::::::
     |      :: :                                  @: :::@:::@:::@:::@@#::::::
     |      :: :                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::       ::::::::::::::: @:::::@::: @: :::@:::@:::@:::@@#::::::
     |     ::::::::::::: :: : :::::::::@:::::@::: @: :::@:::@:::@:::@@#:::::@
   0 +----------------------------------------------------------------------->ki
     0                                                                   224.5
```

RAM consumption: Case 1 / mmrbc 👏 8888

```
--------------------------------------------------------------------------------
Command:            ../mmruby/build/host-production/bin/mmrbc ./hello_world.rb
Massif arguments:   --stacks=yes
ms_print arguments: massif.out.1652
--------------------------------------------------------------------------------
    KB
11.17^                                                                #
     |                                                               @#
     |                                                              @@#  ::
     |                                                              @@#:::::
     |                                            @     @  :@: : :: @@#:::::
     |                                            @  :::@: :@::  :::@@#:::::
     |                                            @  :::@: :@:::@:::@@#:::::
     |                                            @:::::@:::@:::@:::@@#:::::
     |                                            @: :::@:::@:::@:::@@#:::::
     |      ::                                    @: :::@:::@:::@:::@@#::::::
     |      ::                                    @: :::@:::@:::@:::@@#::::::
     |      :: :                                  @: :::@:::@:::@:::@@#::::::
     |      :: :                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::                                  @: :::@:::@:::@:::@@#::::::
     |      ::::       ::::::::::::::: @:::::@::: @: :::@:::@:::@:::@@#::::::
     |     ::::::::::::: :: : :::::::::@:::::@::: @: :::@:::@:::@:::@@#:::::@
   0 +----------------------------------------------------------------------->ki
     0                                                                   224.5
```

RAM consumption: Case 2

Ruby code:

```ruby
# idiotic_script.rb
ary = Array.new(3)
ary[0] = {a: 123e2}
ary[0][:key] = "string"
ary[1] = %w(abc ABC Hello)
ary[2] = 0x1f
ary[3] = !true
puts "my name is #{self.class}, result is #{ary[2] * ary[0][:a]}"
p ary
```

RAM consumption: Case 2

Output:

```
my name is Object, result is 381300
[{:a=>12300, :key=>"string"}, ["abc", "ABC", "Hello"], 31, false]
```

RAM consumption: Case 2 / mrbc

```
--------------------------------------------------------------------------------
Command:            ../mruby/build/host/bin/mrbc ./idiotic_script.rb
Massif arguments:   --stacks=yes
ms_print arguments: massif.out.4133
--------------------------------------------------------------------------------
    KB
162.9^                                                                     ##
     |                                                                     #
     |                                                                    @#
     |                                                                    @#
     |                                                                    @#
     |                                                                    @#
     |                                                                  @:@#
     |                                                                  @:@#
     |                                                                  @:@# :
     |                                                    :::@@::::::@@@@:@# :
     |                                     :::::::@::::::::::@ :: :::@  @:@# :
     |                          ::::@@:::::@ :: ::@:: ::: :::@ :: :::@  @:@# :
     |               ::::::::::::: :@ :::: @ :: ::@:: ::: :::@ :: :::@  @:@# :
     |     ::::::::::::: :: :: ::: :@ :::: @ :: ::@:: ::: :::@ :: :::@  @:@# :
     |     :: :::: ::::: :: :: ::: :@ :::: @ :: ::@:: ::: :::@ :: :::@  @:@# :
     |     :: :::: ::::: :: :: ::: :@ :::: @ :: ::@:: ::: :::@ :: :::@  @:@# :
     |     :: :::: ::::: :: :: ::: :@ :::: @ :: ::@:: ::: :::@ :: :::@  @:@# :
     |     :: :::: ::::: :: :: ::: :@ :::: @ :: ::@:: ::: :::@ :: :::@  @:@# :
     |     :: :::: ::::: :: :: ::: :@ :::: @ :: ::@:: ::: :::@ :: :::@  @:@# :
     |     :: :::: ::::: :: :: ::: :@ :::: @ :: ::@:: ::: :::@ :: :::@  @:@# :
   0 +----------------------------------------------------------------------->Mi
     0                                                                   3.470
```

RAM consumption: Case 2 / mmrbc

```
--------------------------------------------------------------------------------
Command:            ../mmruby/build/host-production/bin/mmrbc idiotic_script.rb
Massif arguments:   --stacks=yes
ms_print arguments: massif.out.13414
--------------------------------------------------------------------------------
    KB
53.20^                                                                 #
     |                                                                 #
     |                                                                @#  :::
     |                                                                @#  : :
     |                                                                @#  : :
     |                                                               @@#  : :
     |                                                               @@# :: :
     |                                                               @@#::: :
     |                                                               @@#::: :
     |                                                           @@::@@#::: :
     |                                                           @@: @@#::: :
     |                                               @@:@:::@::::@@: @@#::: ::
     |                                         @::@::@ :@:: @:: :@@: @@#::: ::
     |                                 @:::@@@:@::@: @ :@:: @:: :@@: @@#::: ::
     |                           @@:@::@:: @@ :@::@: @ :@:: @:: :@@: @@#::: ::
     |                       ::@@@ :@: @:: @@ :@::@: @ :@:: @:: :@@: @@#::: ::
     |             :::::: :::::@ @ :@: @:: @@ :@::@: @ :@:: @:: :@@: @@#::: ::
     |      :::::@@:: :::::: ::@ @ :@: @:: @@ :@::@: @ :@:: @:: :@@: @@#::: ::
     |      : :: @ :: :::::: ::@ @ :@: @:: @@ :@::@: @ :@:: @:: :@@: @@#::: ::
     |      : :: @ :: :::::: ::@ @ :@: @:: @@ :@::@: @ :@:: @:: :@@: @@#::: ::
   0 +----------------------------------------------------------------------->Mi
     0                                                                    1.616
```

RAM consumption: Summary

Ruby script	mrbc	mmrbc
hello_world.rb	157.8KB	11.17KB
idiotic_script.rb	162.9KB	53.20KB

Note: These results are larger than possible results on 32bit architecture because of pointer size

Idiotic script

```ruby
# idiotic_script.rb
ary = Array.new(3)
ary[0] = {a: 123e2}
ary[0][:key] = "string"
ary[1] = %w(abc ABC Hello)
ary[2] = 0x1f
ary[3] = !true
puts "my name is #{self.class}, result is #{ary[2] * ary[0][:a]}"
p ary

=>
my name is Object, result is 381300
[{:a=>12300, :key=>"string"}, ["abc", "ABC", "Hello"], 31, false]
```

Syntax that mmrbc can compile

Syntax ✅ implemented

- lvar, $gvar, @ivar, CONSTANT
- assignment, fcall, method chain
- Literals: Array, Hash, String, Fixnum, Float, etc.
- Keywords: true, false, nil, return, self, etc.
- +, -, *, /, %, **, +=, -=, *=, /=, %=, &=, |=, ^=, **=, <<=, >>=
- !, ~, +, -, not, &, |, ^, <<, >>
- Syntax sugar: `%w(a b c)`
- Interpolation: `"The answer is #{get_result(@attr)}!"`

Syntax 🛠 under construction

- if then elif else end, unless
- case when else then
- while, until, break, next, redo, retry, for in
- [0,1].each do |var| end
- def method_name(arg); end
- class MyClass < ParentClass; end
- rescue, ensure
- ..., etc.

Roadmap

https://github.com/hasumikin/mmruby/issues/6

Code size (to estimate ROM consumption)

```
% ls -l
(...)
.rwxrwxrwx 609k hasumi 17 Aug  7:03 mruby_machine_PSoC5LP.a
.rwxrwxrwx 903k hasumi 17 Aug  7:03 mruby_machine_PSoC5LP.elf
.rwxrwxrwx 650k hasumi 17 Aug  7:03 mruby_machine_PSoC5LP.hex
.rwxrwxrwx 425k hasumi 17 Aug  7:03 mruby_machine_PSoC5LP.map

% size mruby_machine_PSoC5LP.elf
   text    data     bss     dec     hex filename
 136080     320   63009  199409   30af1 mruby_machine_PSoC5LP.elf
```

Code size (to estimate ROM consumption)

```
% ls -l
(...)
.rwxrwxrwx 609k hasumi 17 Aug  7:03 mruby_machine_PSoC5LP.a
.rwxrwxrwx 903k hasumi 17 Aug  7:03 mruby_machine_PSoC5LP.elf
.rwxrwxrwx 650k hasumi 17 Aug  7:03 mruby_machine_PSoC5LP.hex
.rwxrwxrwx 425k hasumi 17 Aug  7:03 mruby_machine_PSoC5LP.map

% size mruby_machine_PSoC5LP.elf
   text	   data	    bss	    dec	    hex	filename
 136080	    320	  63009	 199409	  30af1	mruby_machine_PSoC5LP.elf
```

136_080 + 320 = 136_400 ≒ **133.2KB** 👍

Summary

- mmrbc is a mini mruby compiler
  - VM code runs on both mruby VM and mruby/c VM
  - Still incomplete regarding syntax and memory efficiency
- mruby machine
  = mmrbc + mruby/c + shell + dirivers
  - Runs on PSoC5LP { RAM: 64KB, ROM: 256KB }

Thank you!