Arduino Library
MemoryMapLib

2011年11月27日

日本Androidの会・神戸支部

石井　康寛 Yasuhiro ISHII

ishii.yasuhiro@gmail.com

MemoryMap プロトコルとは?

JS Robotics社が用意された機器間通信向け汎用プロトコル。制御対象は、主にロボットを想定。
寺園氏から打診され、既存の神戸支部バイナリプロトコルと大差無いので移植工数もかからないだろうということで乗り換えする事にした。(AndyLibがこのプロトコルを使用するとのことであったため)

→結局は既存ソースが気に入らなくてフルスクラッチになってしまった。

MemoryMap プロトコルとは?

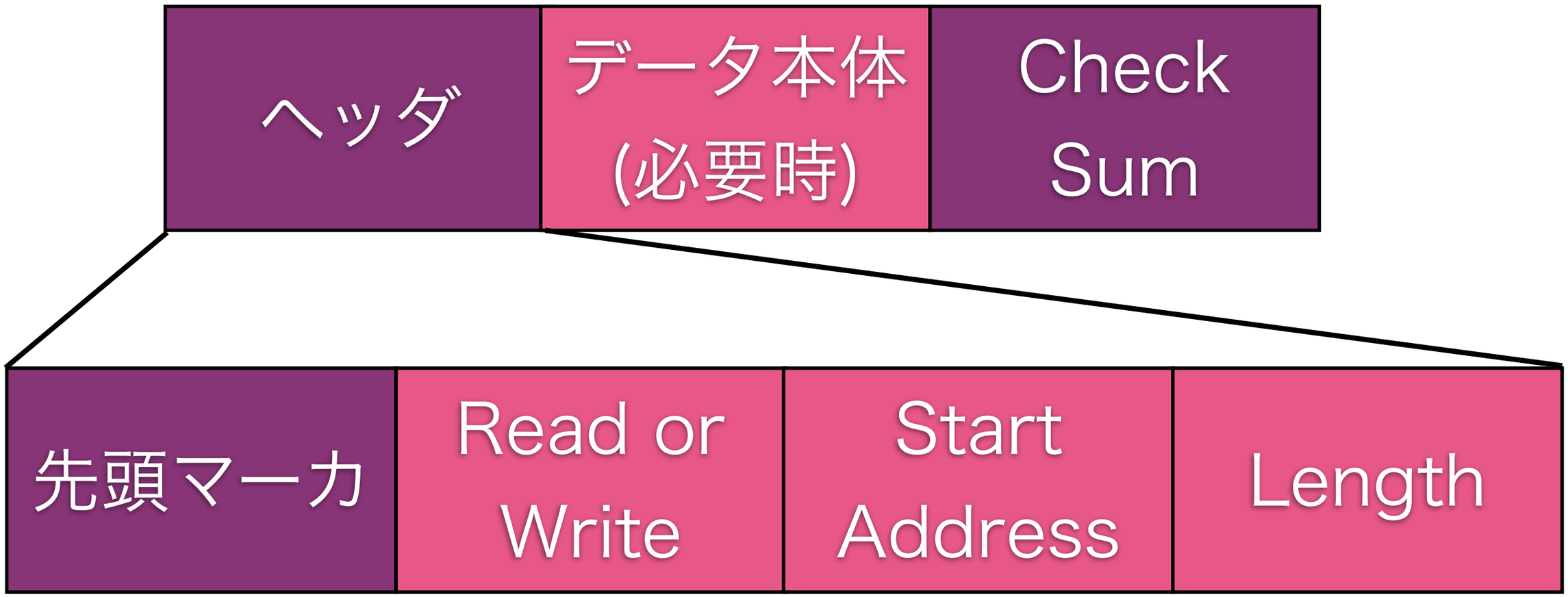
通信はホスト側から。

ex)Android、PC、リモコン等

ex)ロボット、マイコンボード等

MemoryMap プロトコルとは?

メッセージの基本形式



MemoryMap プロトコルとは?

アドレスに役割を割り当て(マッピング)する

アドレス	内容例
0	HWバージョン取得
1	モータ駆動R
2	モータ駆動L
3	ライト点滅速度
4	ビープ鳴動

機器間通信は、それぞれのアドレスに読み書きを実施するように行う。メモリーマッププロトコルの名称から、アドレスに割り当てた内容をメモリーと呼ぶことにする。

アドレスに機能をメモリーやレジスタのように定義する様がメモリーマップのようである。

MemoryMap プロトコルとは?

メモリーへの書き込み

メモリー書き込みは、Start AddressとLengthを指定して連続転送する

ex)コマンド「アドレス1から3バイトWrite、値は2Fh,50h,60h」

アドレス	内容
0	HWバージョン取得
1	モータ駆動R
2	モータ駆動L
3	ライト点滅速度
4	ビープ鳴動

Write

← 2Fh

← 50h

← 60h

連続するアドレスをまとめてライトできる。

MemoryMap プロトコルとは?

メモリーからの読み出し

メモリー読み出しは、Start AddressとLengthを指定して連続転送する

ex)コマンド「アドレス0から1バイトRead」

アドレス	内容
0	HWバージョン取得
1	モータ駆動R
2	モータ駆動L
3	ライト点滅速度
4	ビープ鳴動

Read → 0x01h

連続するアドレスをまとめてリードできる。

Arduino Library

MemoryMapLib

MemoryMapLibとは?

前述のプロトコルを簡単に実現する為のArduino向けライブラリ

ArduinoがMemoryMapプロトコルの送受信を行う。現在はUARTとのやりとりを想定しているが、その他のやりとりにも対応容易。

UARTを使用している一般に流通しているBluetoothモジュールは直結可能。

MemoryMapLib使い方

Arduino IDEへの登録

MemoryMapLib.cppとMemoryMap.hを
以下のパスにコピー(デフォルトインストールの場合)

- Mac

/Applications/Arduino.app/Contents/Resources/Java/libraries/MemoryMapLib

- Win

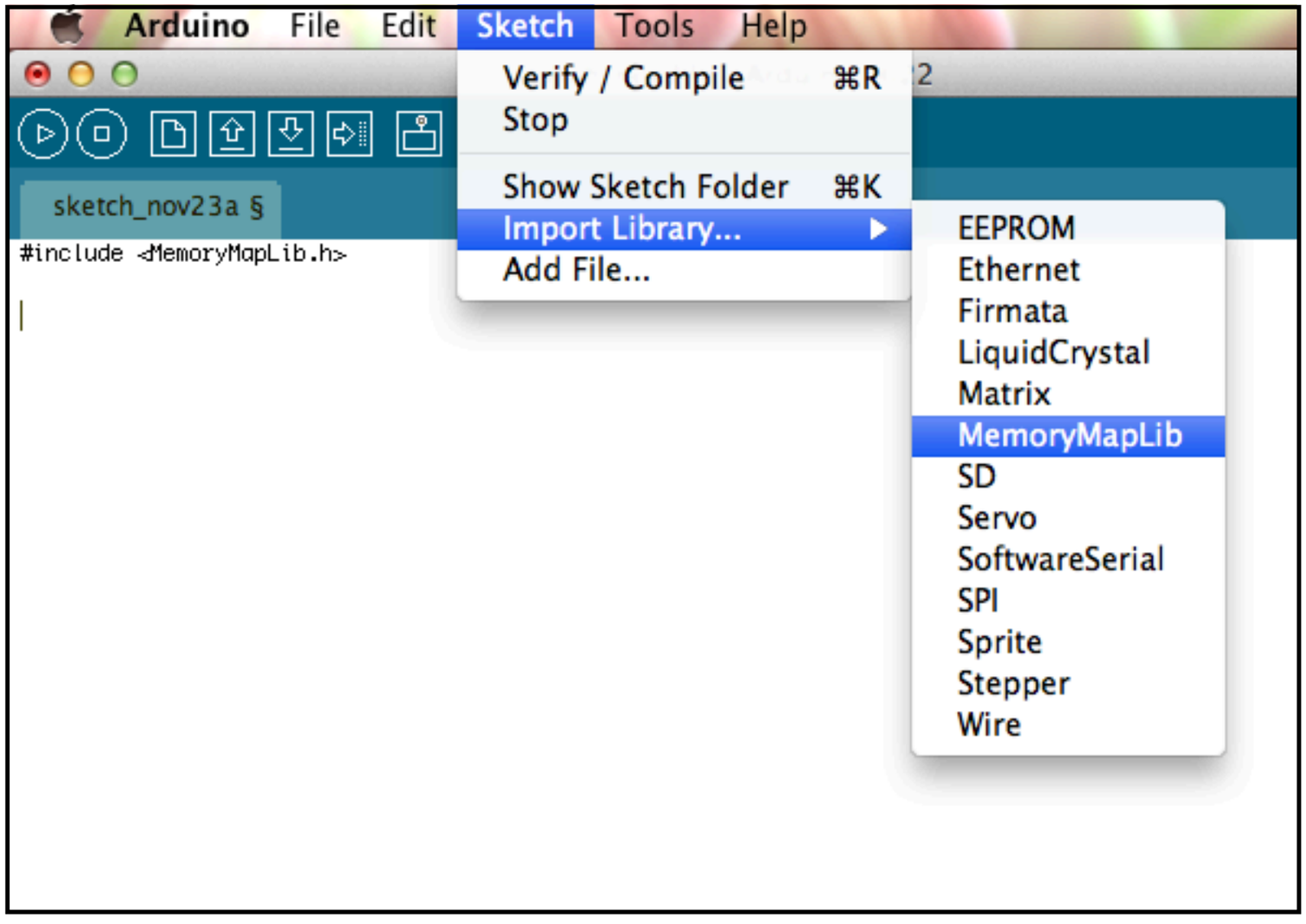
c:¥Program Files¥....

- Linux

?

MemoryMapLib使い方

Arduino IDEでSketch→Import Libraryする



MemoryMapクラスが使えるようになる。

MemoryMapLib使い方

MemoryMapLibインスタンシエーション

```
MemoryMap m;
```

MemoryMapLib使い方

MemoryMapLibが提供するAPI

MemoryMap::setStreamInterface()

MemoryMap::registerMapAddressJob()

MemoryMap::registerMapAddressVar()

MemoryMap::removeMapAddress()

MemoryMap::poll()

MemoryMap::registerMapAddressBlock()

MemoryMapLib使い方

アドレスにマッピング

特定のアドレス毎に、R or Wアクセスに対する動作を定義する。 変数値更新または対応関数呼び出しを選択する。

使用するAPI

registerMapAddressJob(unsigned char address,unsigned char RWOP,void (*func)(unsigned char,unsigned char,unsigned char*))

registerMapAddressVar(unsigned char address,unsigned char RWOP,unsigned char* ptr)

removeMapAddress(unsigned char address)

MemoryMapLib使い方

アドレスに変数をマッピング

アドレス20hに変数valを設定。書き込みのみ定義。

```
unsigned char val;


m.registerMapAddressVar(
0x20,
(OPERATION_WRITE),
&val);
```

読み出しのみの場合は、OPERATION_READ。
読み書き対応の場合は
OPERATION_WRITE|OPERATION_READ。

MemoryMapLib使い方

アドレスに関数をマッピング

マッピングする関数はvoid func(unsigned char,unsigned char,unsigned char*)形式

```
void function0x21(unsigned char RWOP,unsigned char address,unsigned
char* value)
{
    if(RWOP == OPERATION_WRITE){
        other_stuff = *value
    } else {
        *value = 0x20;
    }
}
```

```
m.registerMapAddressJob(
    0x21,
    (OPERATION_WRITE|OPERATION_READ),
    &function0x21);
```

アドレス21hに関数 function0x21を設定。

読み書き対応。

MemoryMapLib使い方
シリアルI/Fと紐付ける

```
Serial.begin(9600);   // シリアル(UART)初期化。9600bps

m.setStreamInterface(&Serial); // MemoryMapプロトコルが使用するI/Fに設定
```

Arduino MEGAでは複数のUART(Serial1、Serial2、…)があるので、必要なUARTで登録すれば良い(はず)

MemoryMapLib使い方
周期処理から呼び出し

```
loop(){
    m.poll();
}
```

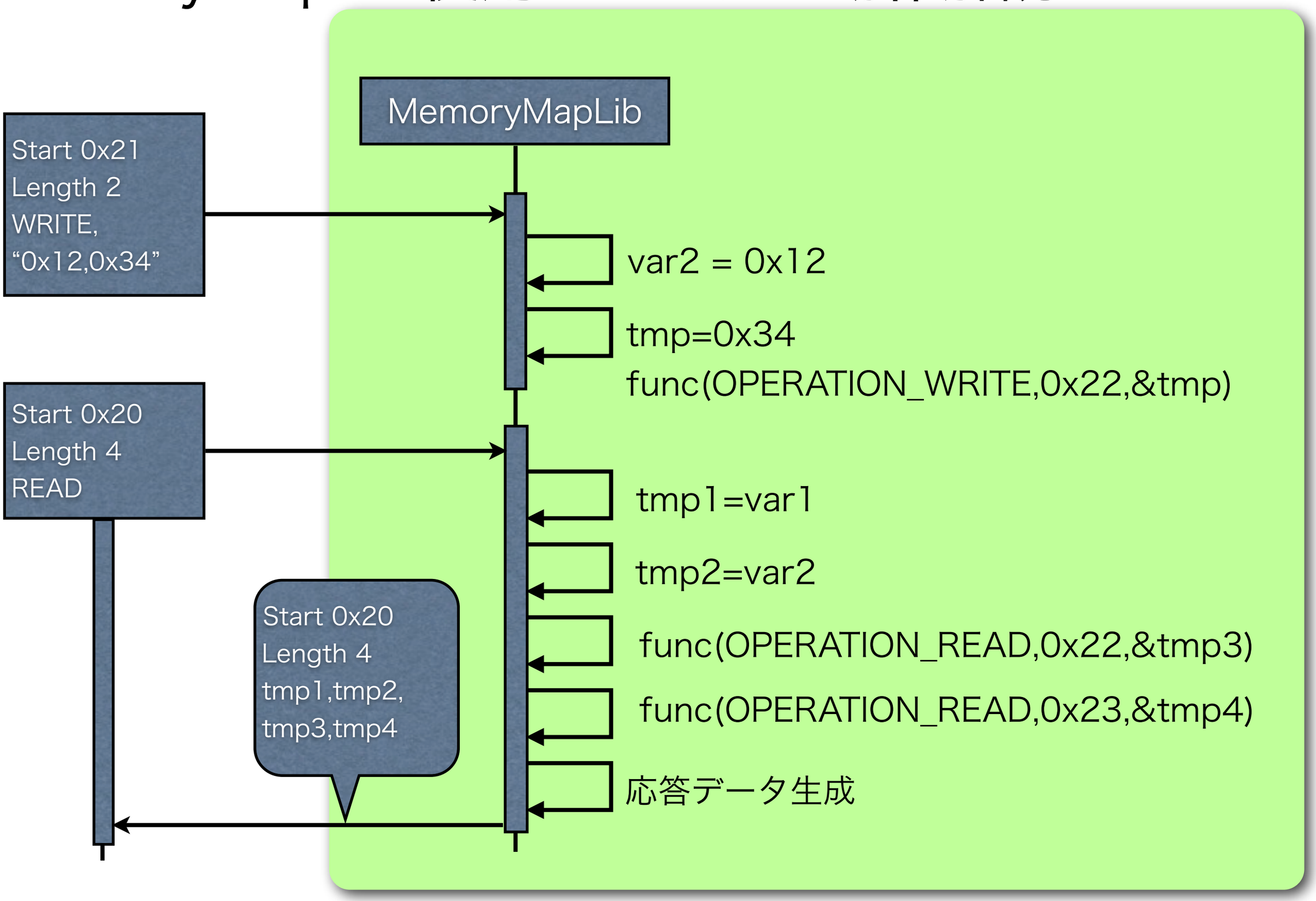
registerMapAddressJobで定義された通信があれば設定した関数をコールバックする。
registerMapAddressVarで定義された通信があれば、設定した変数に対応した処理を行う。

100[ms]に一度以上コールすること。

MemoryMapLib使用スケッチと動作解説

```
MemoryMap mMemoryMap;
unsigned char var1,var2;
void setup(){
    mMemoryMap.registerMapAddressVar(
        0x20,OPERATION_READ,&var1);
    mMemoryMap.registerMapAddressVar(
        0x21,OPERATION_WRITE|OPERATION_READ,&var2);
    mMemoryMap.registerMapJob(
        0x22,OPERATION_WRITE|OPERATION_READ,&func);
    mMemoryMap.registerMapJob(
        0x23,OPERATION_READ,&func);
    Serial.begin(9600);
    mMemoryMap.setStreamInterface(&Serial);
}
void loop(){
    mMemoryMap.poll();
    if(var1 == 0x55) ...
}
void func(unsigned char RWOP,unsigned char addr,unsigned char* value){
    if(RWOP & OPERATION_READ){
        *value = 0xc3;
    } else {
        if(*value != 0x00){
            LED_ON();
        }
    }
}
```

MemoryMapLib使用スケッチと動作解説

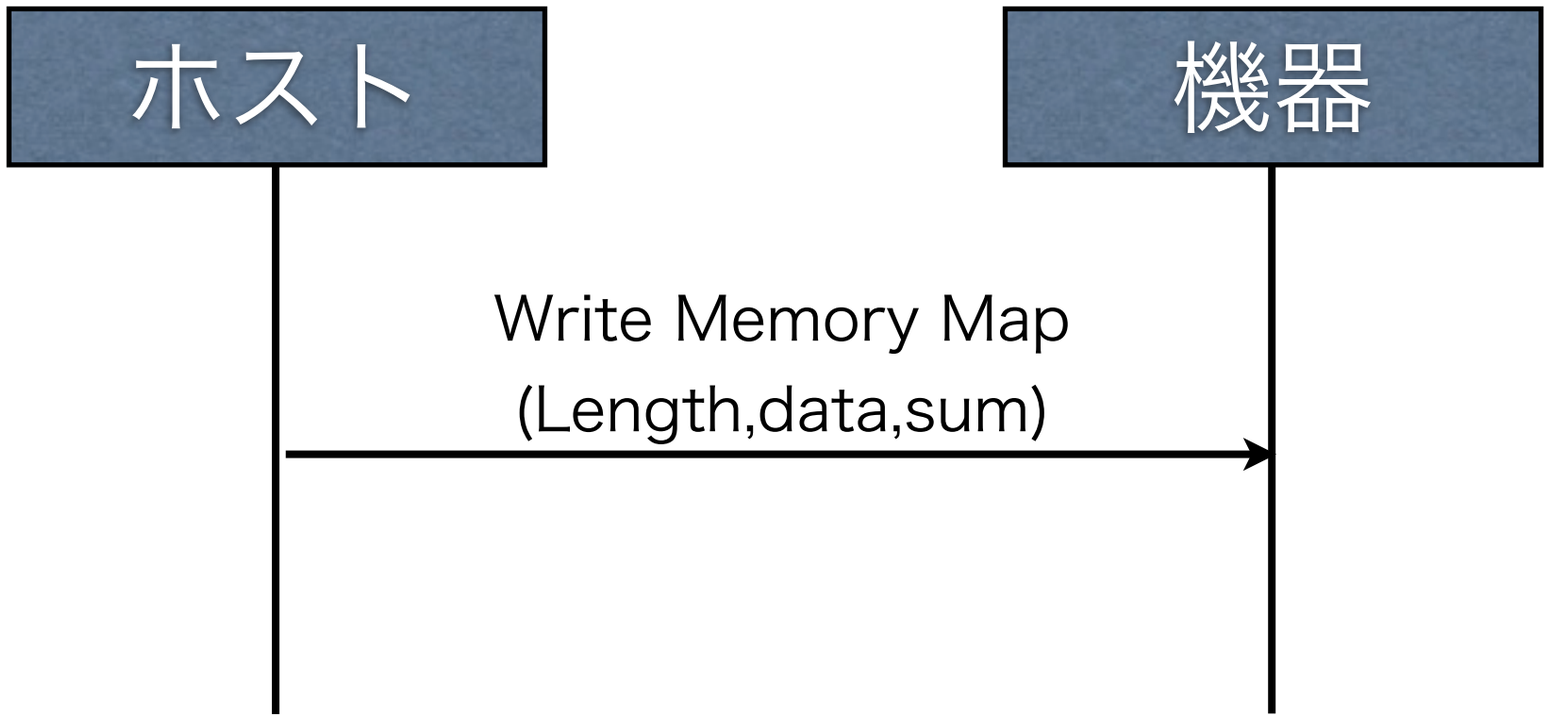


付録

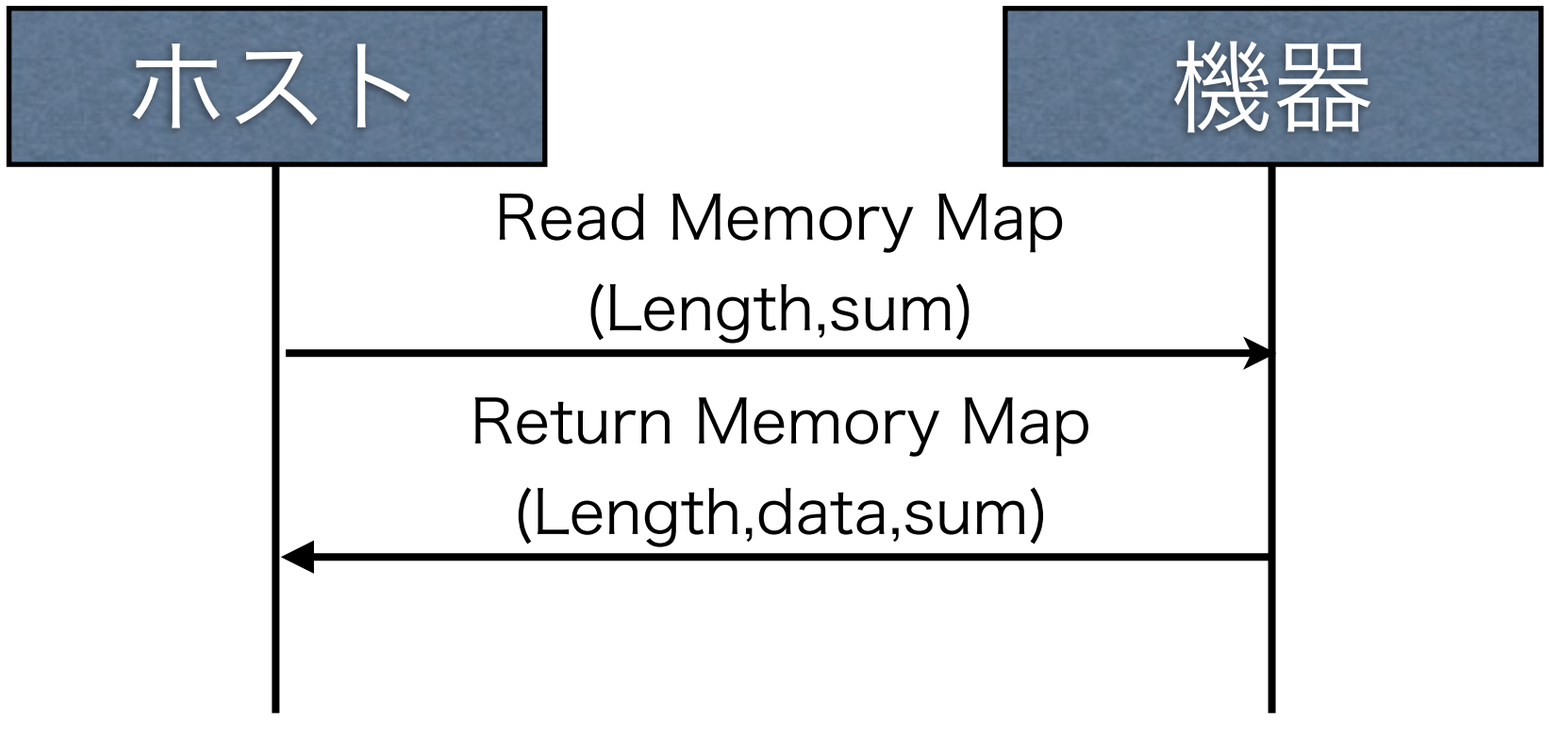
プロトコル内容について

プロトコル基本シーケンス

ホスト→機器データ書き込み(Write)

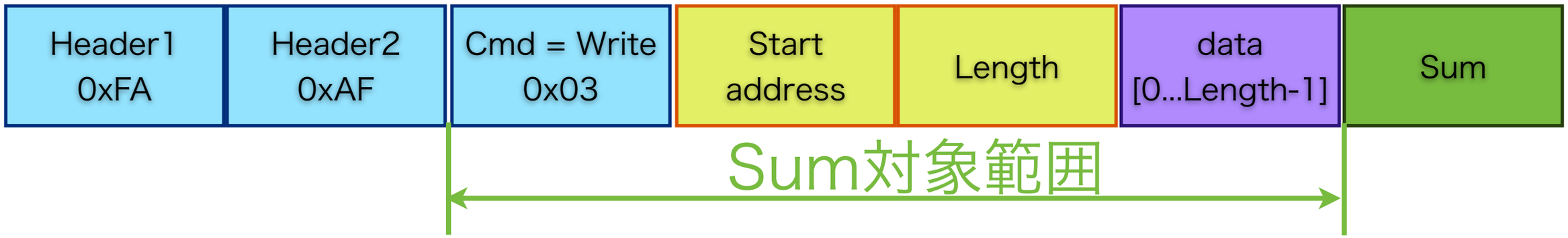


ホスト→機器データ問い合わせ・応答(Read)

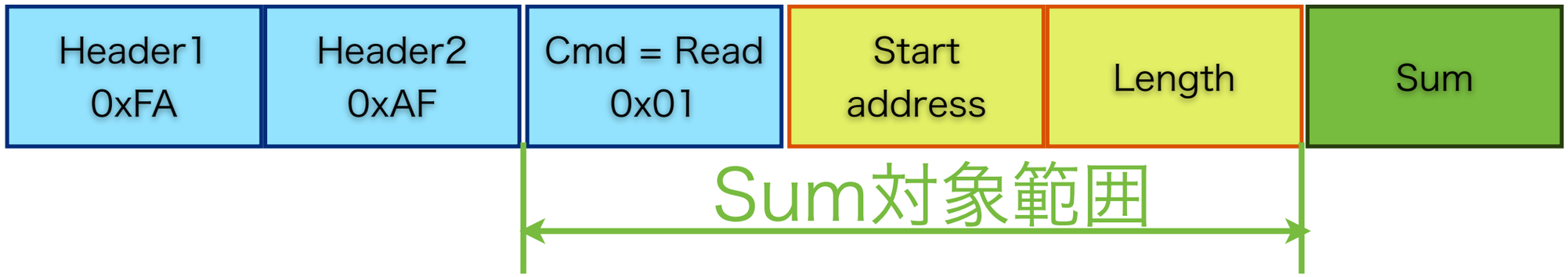


メッセージ形式

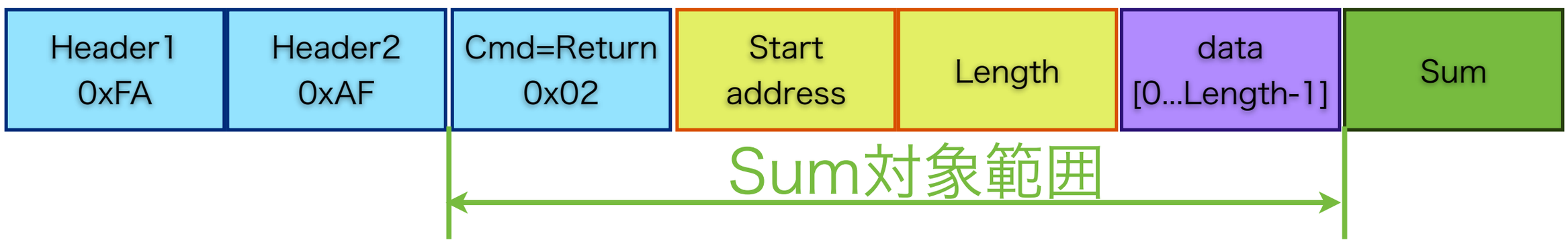
Write Memory Map



Read Memory Map



Return Memory Map



コマンド形式

Sum値の算出方法

Sum対象範囲のデータを1バイト毎にXOR
していく

Header1 0xFA	Header2 0xAF	Cmd = Write 0x03	Address = 0x1C	Length = 1	data[0] 0x1B	Sum = 0x05

Sum = 0x03 XOR 0x1c XOR 0x01 XOR 0x 1b

= 0x05

参考文献

- JS Robotics社メモリーマップ資料

改訂履歴

2011/11/29 初版