ARTSAT

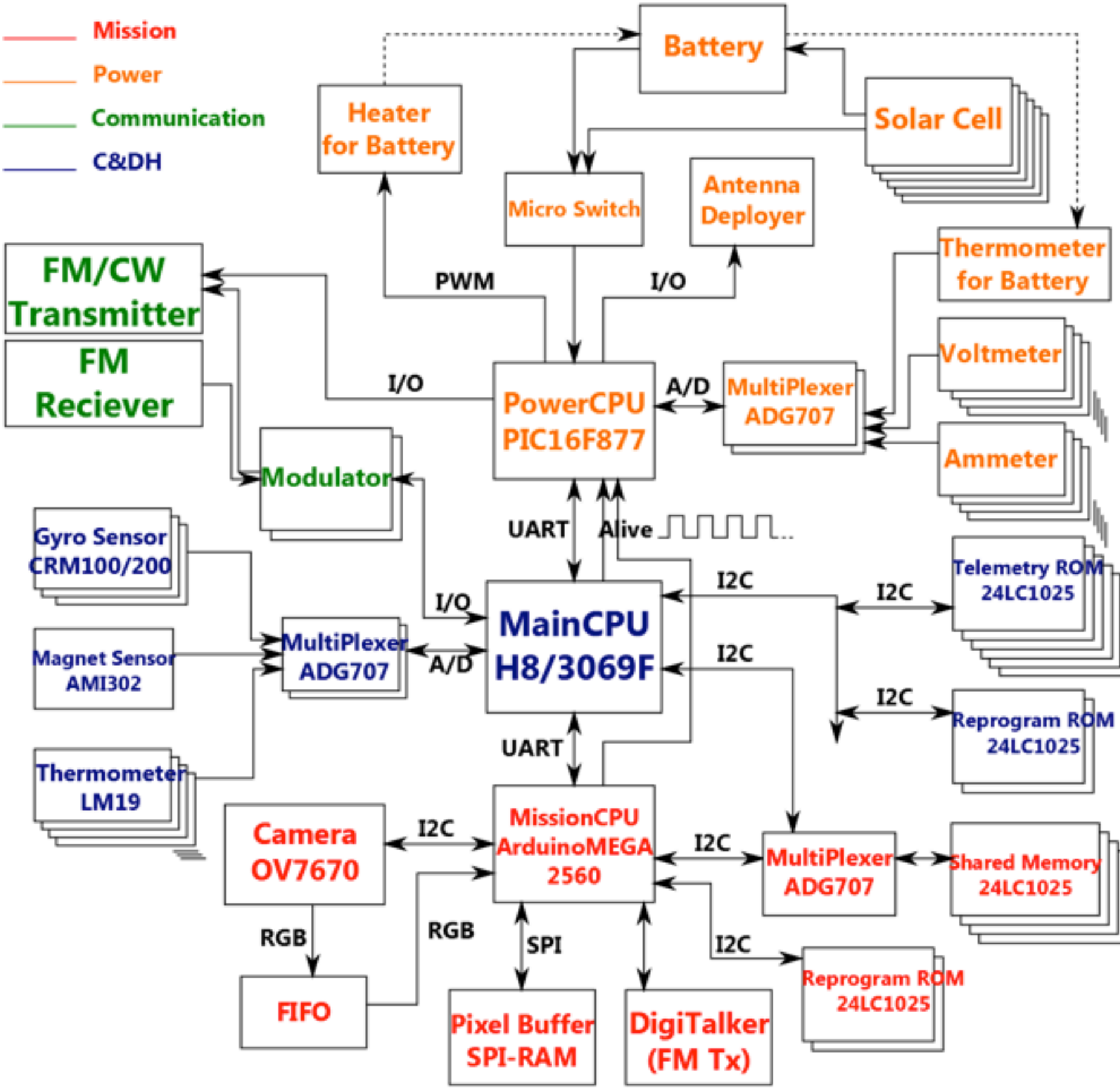
Arduino互換ミッションOBC用のソフトウェア開発

堀口淳史・橋本論・中澤賢人・久保田晃弘

芸術衛星

ARTSAT1：INVADER

芸術衛星INVADERのフライトモデル

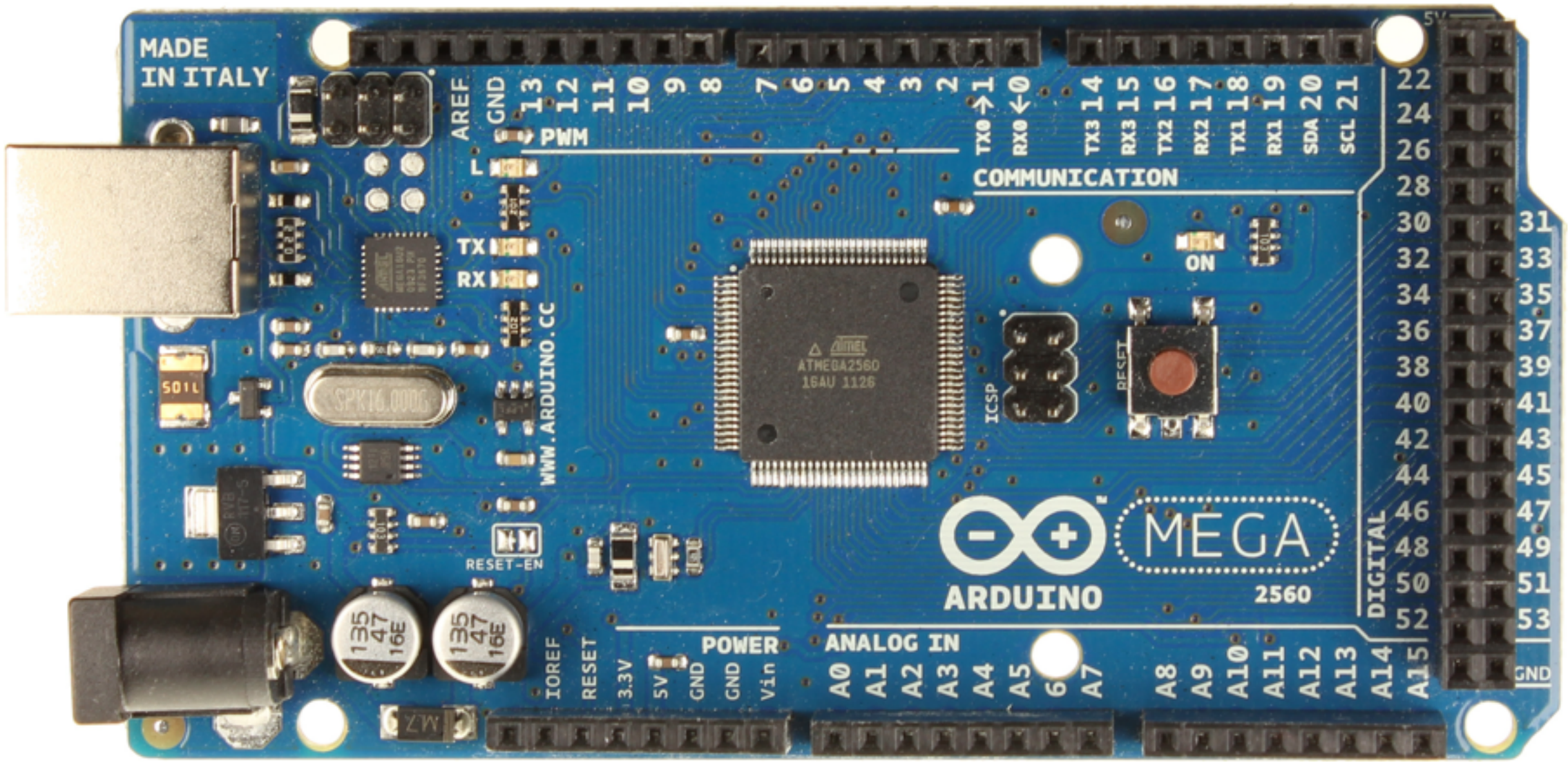


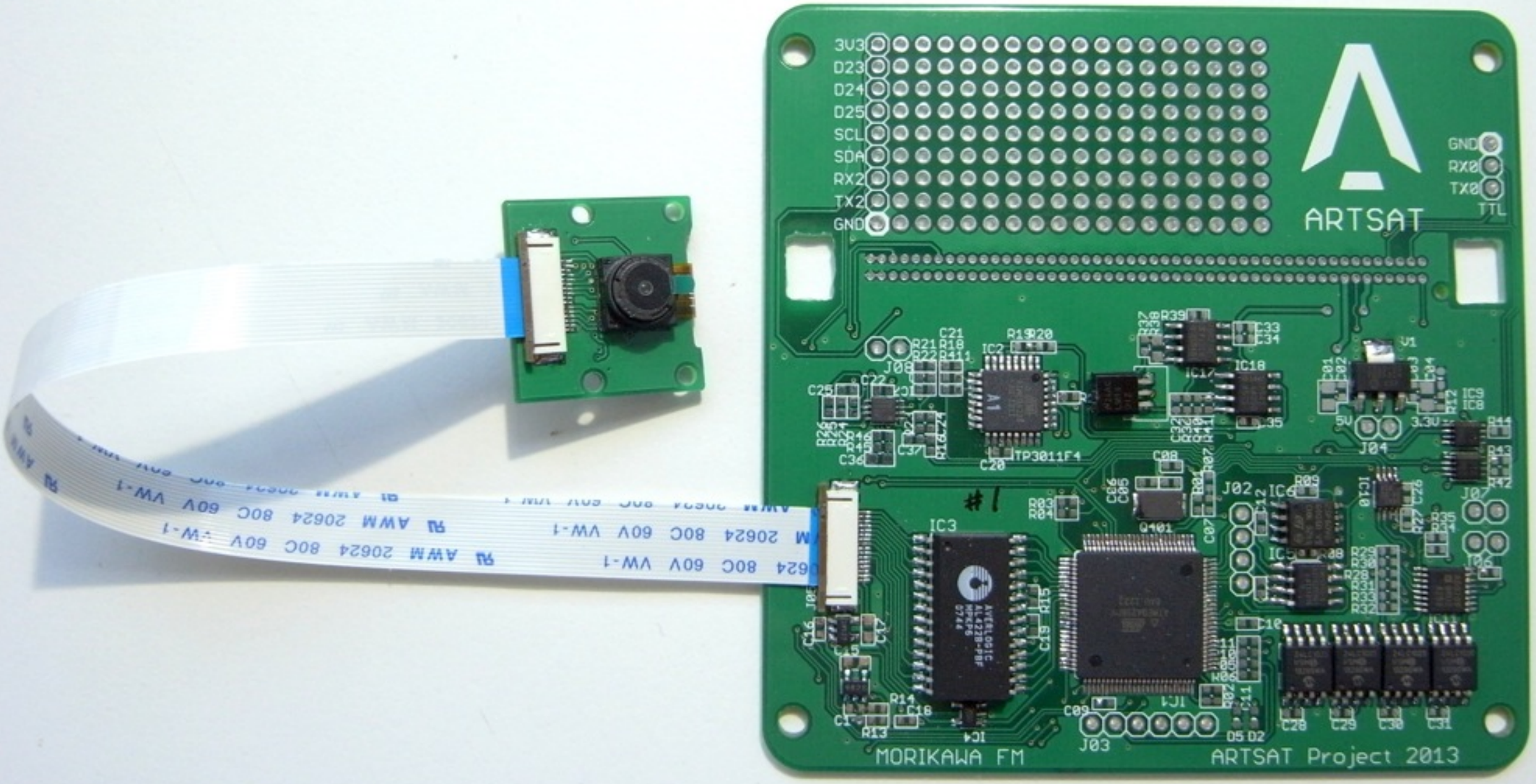
Morikawa

極限環境における遠隔創造

遠隔創造

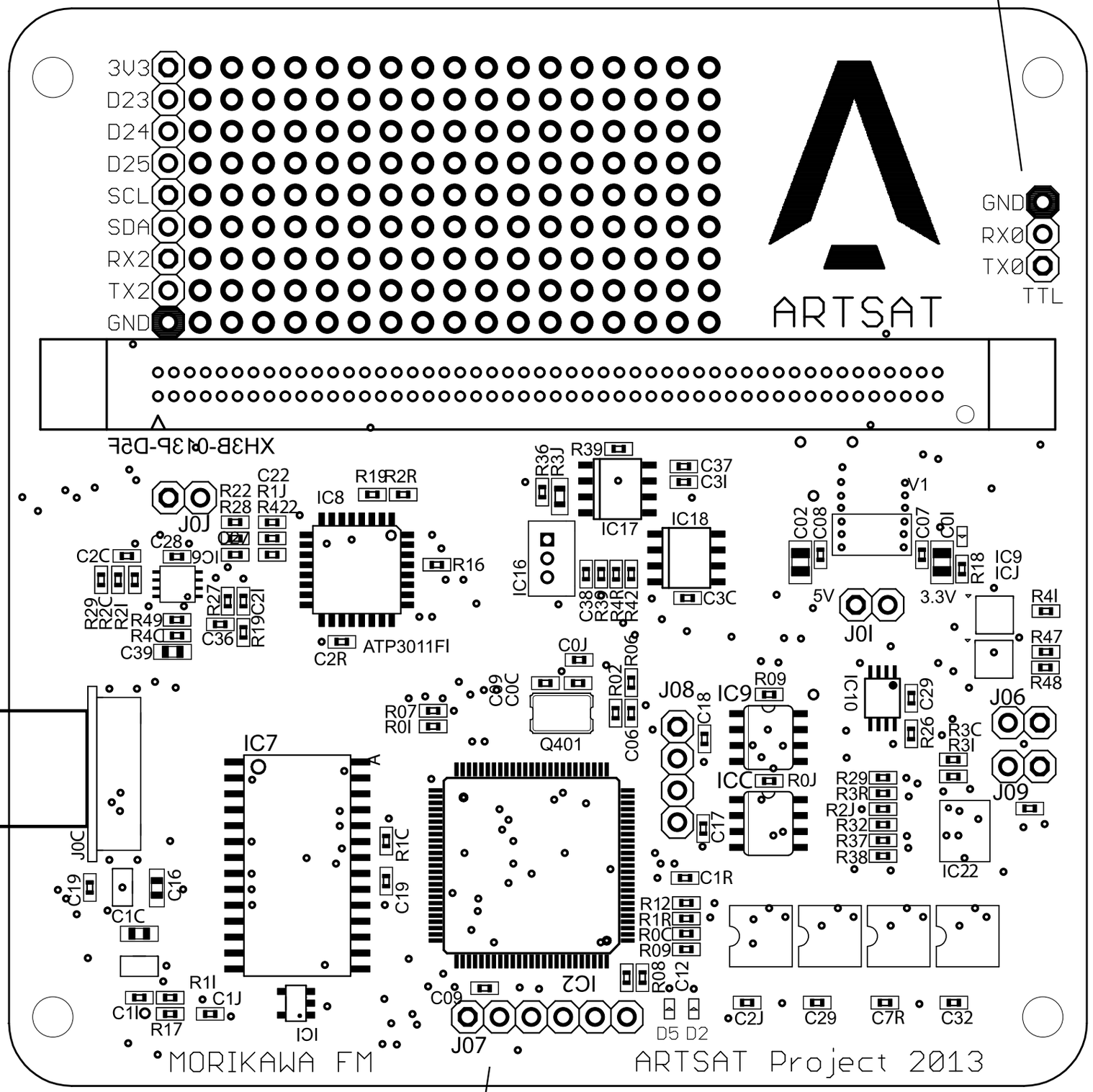
Software Art on Orbit





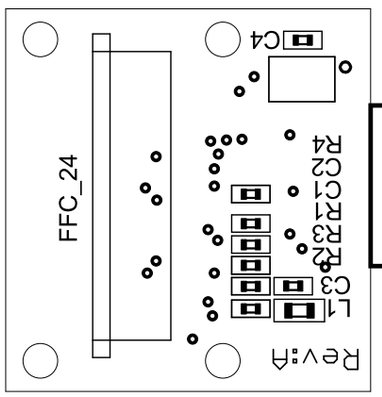
H8との通信ライン
(確認用)

ユニバーサル基板エリア(試作領域+余剰IO)



書き込みピンヘッダ(FTDI配列、5V給電OK)

カメラモジュール固定基板

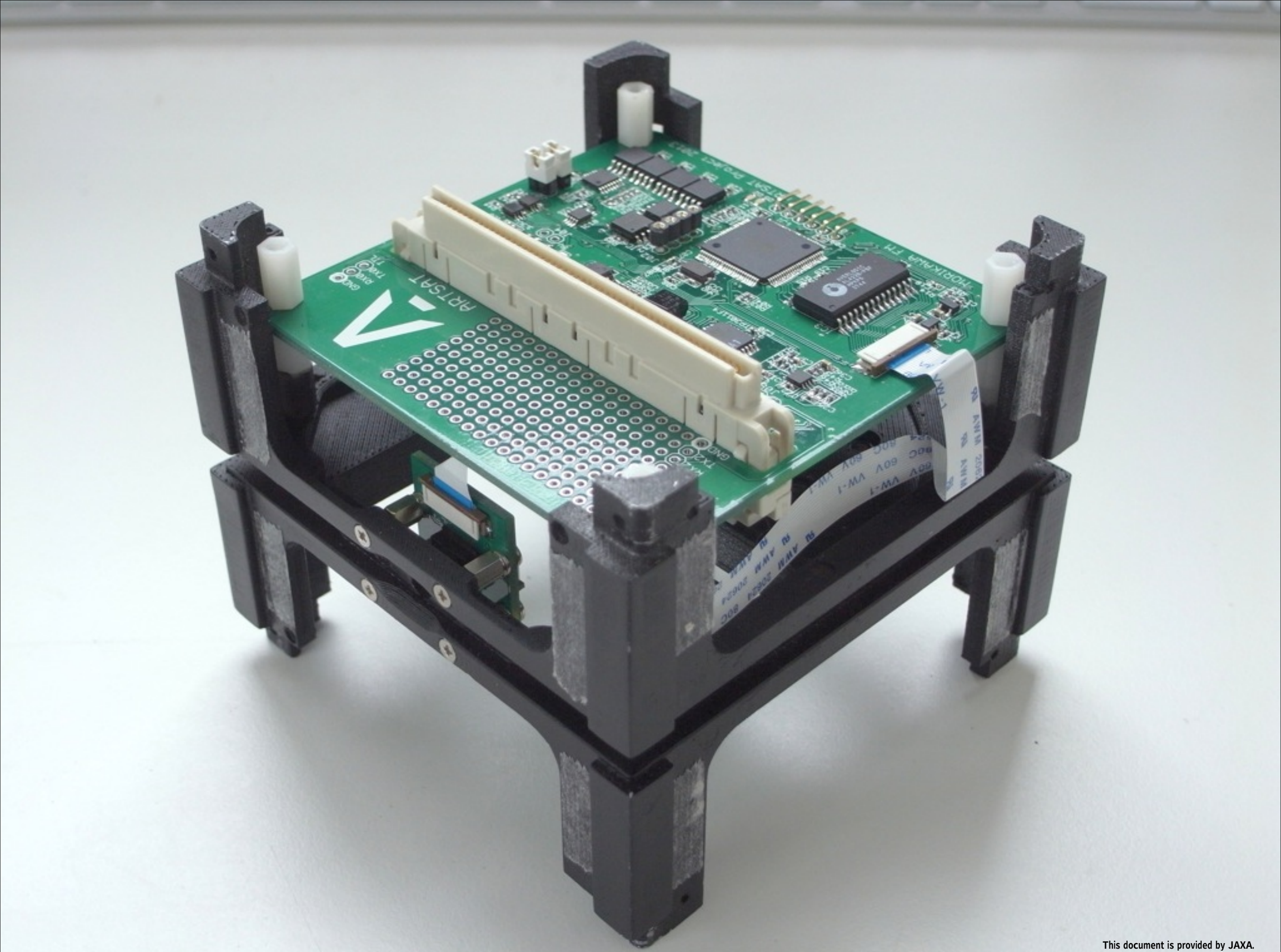


FFC Cable

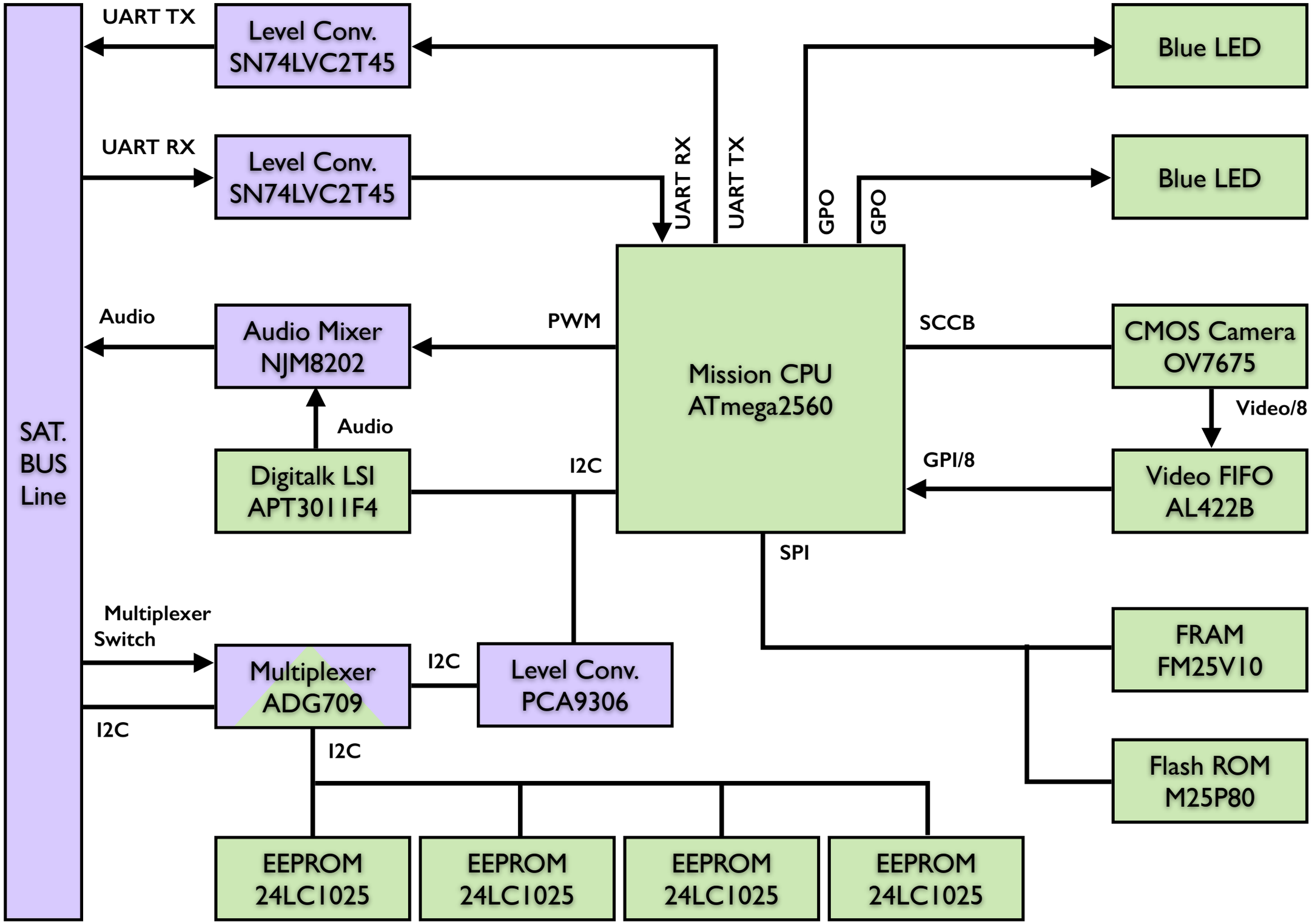
16P(0.5mm)

W 160mm

外形: 21x22mm



Morikawa System Diagram



サポートする周辺機器

1. EEPROM(ATmega 2560 内蔵)
2. 共有メモリ(EEPROM 24LC1025×4)
3. 一時メモリ(SRAM FM25V10×1)
4. 一時メモリ(Flash M25P80×1)
5. オンボードLED×2

サポートする周辺機器

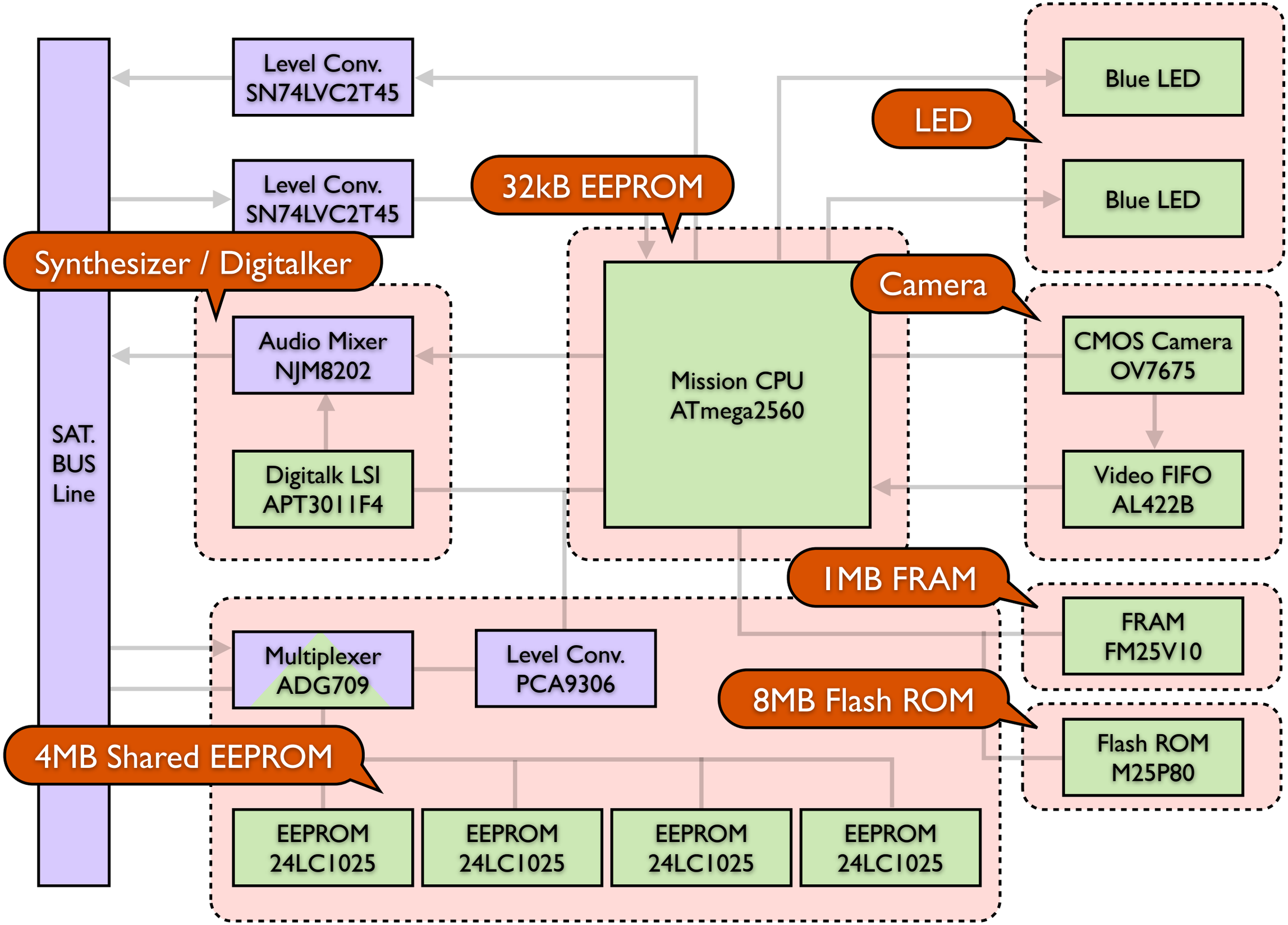
6. 音色発音 (ATmega 2560 内蔵)

7. モールス発音 (ATmega 2560 内蔵)

8. 音声合成 (AquesTalk ATP3011F4)

9. CMOS カメラ (OV7670 + AL422B)

Morikawa Peripherals



MorikawaSDK

- Arduino 互換ライブラリ
- Arduino IDE で開発可能
- C++ で構成されたシンプルな API

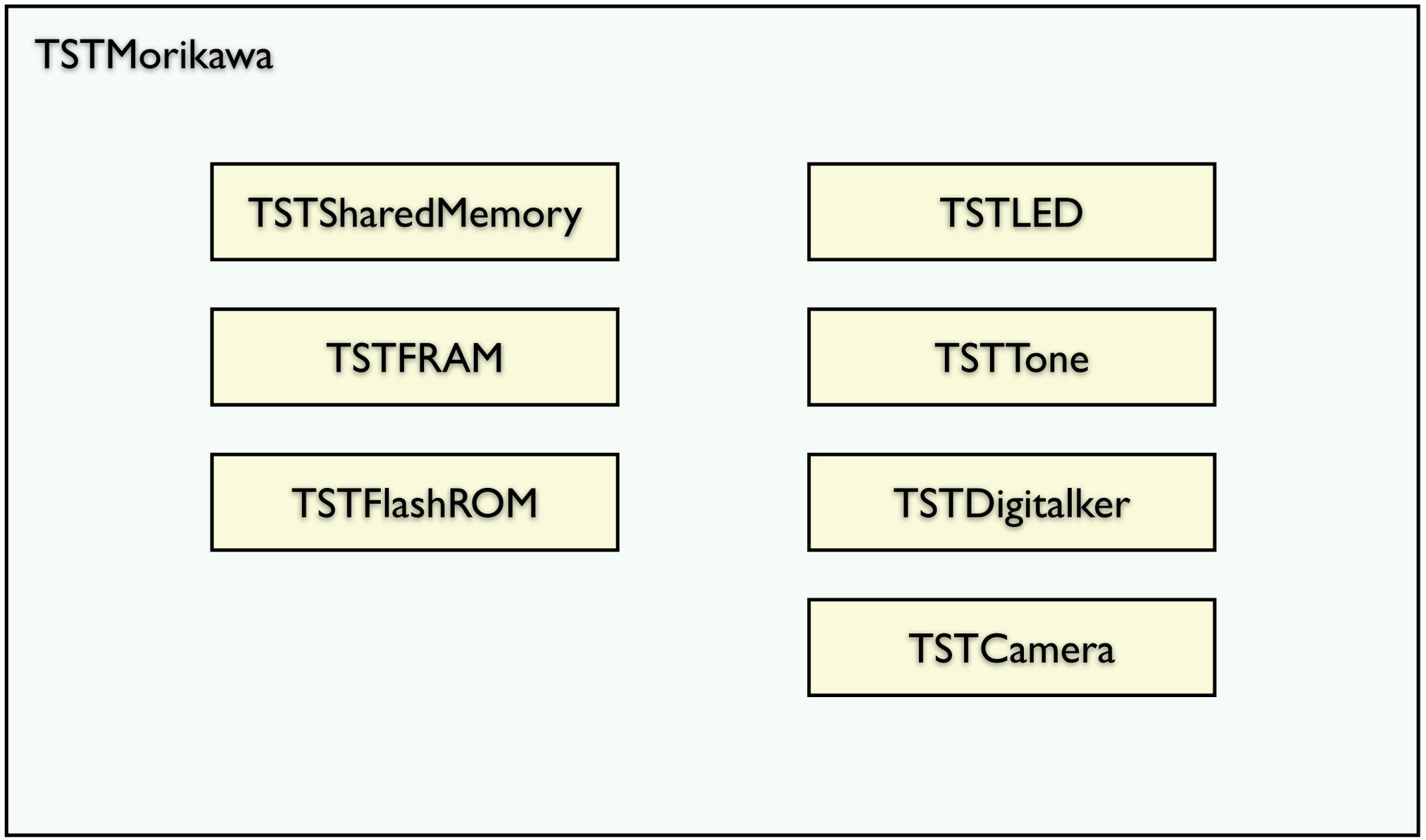
ライブラリ設計の方針

- Arduino用ライブラリの設計方針に準拠
- インターフェースを１つのクラスにまとめた理解しやすい設計
- 最小機能のAPIの組み合わせで最大の可能性を提供
- 放射線耐性を持たせるための重要変数の３冗長化
- メインCPUからの強制シャットダウンには可能な限り安全なタイミングで処理を中止

クラス解説

クラス名	機能
TSTMorikawa	全てのペリフェラル機器の管理・メイン CPU との通信等
TSTSharedMemory	共有メモリの読み書き
TSTFRAM	FRAM の読み書き
TSTFlashROM	Flash ROM の読み書き
TSTLED	LED の駆動
TSTTone	PWM を利用した音色の生成
TSTDigitalker	音声合成 LSI の操作
TSTCamera	CMOS カメラの操作・撮影データの読み込み
TSTCriticalSection	割り込み禁止コード領域の管理
TSTTrinity	プリミティブ型変数の３冗長化
TSTSCCB	SCCB バスの操作

クラス所有関係

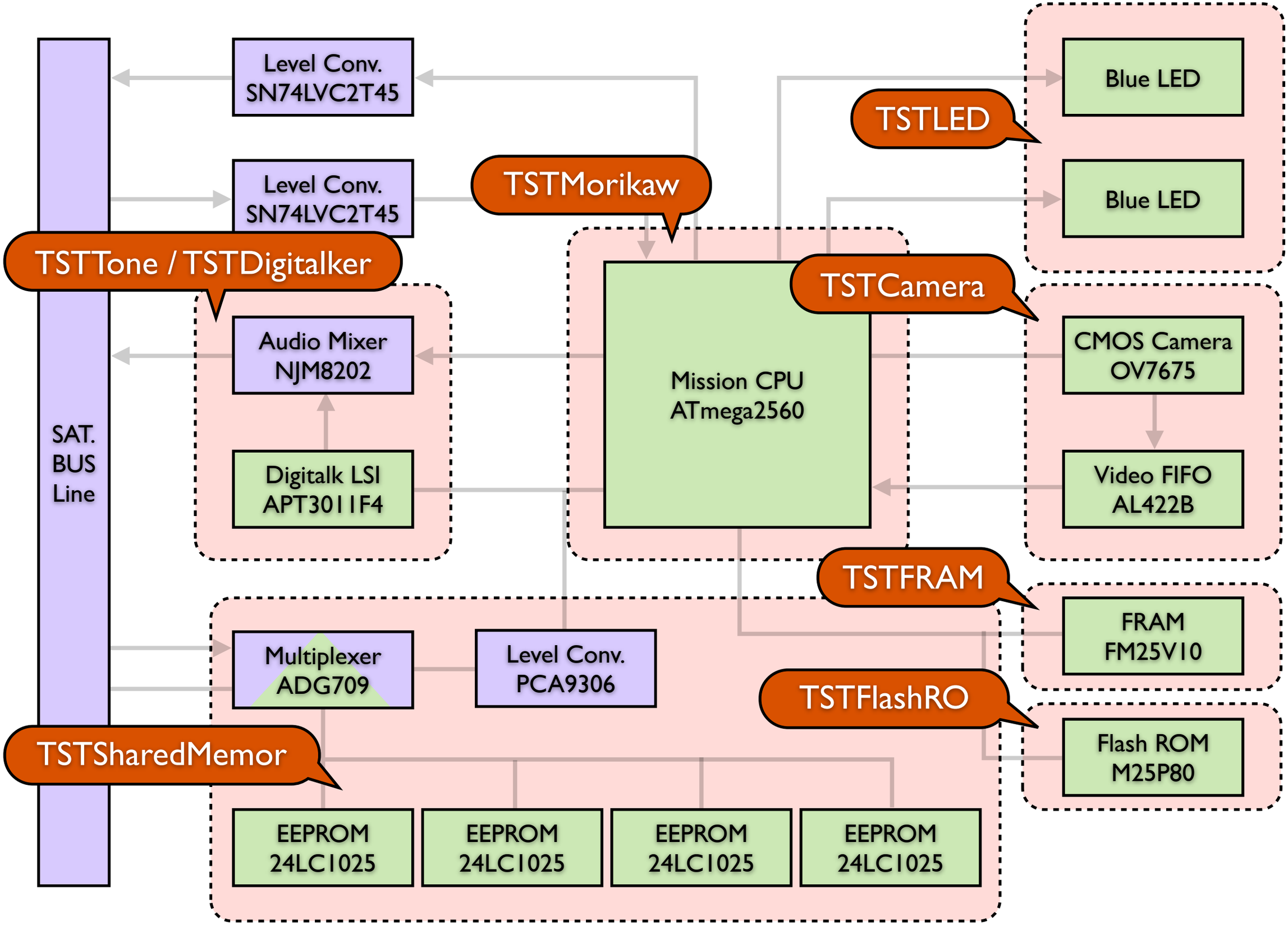


TSTCriticalSection

TSTTrinity

TSTSCCB

Peripheral Driver Classes



テレメトリ取得機能

1 秒毎にメイン CPU から受信されるテレメトリ

分類	詳細
時刻	OBC 時刻
電圧	バス・電池・太陽電池
電流	バス・電池・太陽電池・太陽電池（個別）・電源基板 メイン基板・ミッション基板・アンテナ・電池ヒーター FM 送信機・FM 受信機・CW 送信機
温度	電池（1 ~ 3）・太陽電池（個別）・電源基板 メイン基板・ミッション基板 FM / CW 送信機・FM 受信機
角速度	X 軸・Y 軸・Z 軸
磁気	X 軸・Y 軸・Z 軸

API 例 - 基本関連

戻り値	関数	引数
unsigned long	getBootTime	(void) const
unsigned char	getBootCount	(void) const
unsigned char	getBootMode	(void) const
bool	isValid	(void) const
bool	hasAbnormalShutdown	(void) const
TSTError	setup	(void)
void	cleanup	(void)
void	shutdown	(void)
void	loop	(void)

API 例 - テレメトリ関連

戻り値	関数	引数
TSTError	getTelemetryTime	(TimeType type, unsigned long* result) const
TSTError	getTelemetryVoltage	(VoltageType type, unsigned char* result) const
TSTError	getTelemetryVoltage	(VoltageType type, double* result) const
TSTError	getTelemetryCurrent	(CurrentType type, unsigned char* result) const
TSTError	getTelemetryCurrent	(CurrentType type, double* result) const
TSTError	getTelemetryTemperature	(TemperatureType type, unsigned char* result) const
TSTError	getTelemetryTemperature	(TemperatureType type, double* result) const
TSTError	getTelemetryGyro	(GyroType type, unsigned char* result) const
TSTError	getTelemetryGyro	(GyroType type, double* result) const
TSTError	getTelemetryMagnet	(MagnetType type, unsigned char* result) const
TSTError	getTelemetryMagnet	(MagnetType type, double* result) const
bool	hasTelemetryUpdate	(void) const

API 例 - FRAM 関連

戻り値	関数	引数
unsigned long	getSizeFRAM	(void)
unsigned int	getPageSizeFRAM	(void)
unsigned long	getSectorSizeFRAM	(void)
bool	isValidFRAM	(void) const
TSTError	writeFRAM	(unsigned long address, void const* data, unsigned int size, unsigned int* result = NULL)
TSTError	writeFRAMPGM	(unsigned long address, void const PROGMEM* data, unsigned int size, unsigned int* result = NULL)
TSTError	readFRAM	(unsigned long address, void* data, unsigned int size, unsigned int* result = NULL)
TSTError	formatFRAM	(void)

API 例 - 音色関連

戻り値	関数	引数
TSTError	setNoteBPM	(int param = -1)
int	getNoteBPM	(void) const
bool	isValidTone	(void) const
TSTError	playFrequency	(FrequencyType frequency, unsigned long duration)
TSTError	playFrequency	(FrequencySequence const* sequence, int length = -1)
TSTError	playFrequencyPGM	(FrequencySequence const PROGMEM* sequence, int length = -1)
TSTError	playNote	(NoteType note, DurationType duration, QualifierType qualifier = QUALIFIER_NONE)
TSTError	playNote	(NoteSequence const* sequence, int length = -1)
TSTError	playNotePGM	(NoteSequence const PROGMEM* sequence, int length = -1)

MorikawaSDK を用いたスケッチ

- MorikawaSDK.hをインクルード
- TSTMorikawaクラスのインスタンスである Morikawa オブジェクトを利用
- setup()関数内で Morikawa.setup() を呼び出す
- loop()関数内で Morikawa.loop() を呼び出す
- タスク終了後 Morikawa.shutdown() を呼び出し低消費電力モードに移行

Example - Hello Space

```
#include <MorikawaSDK.h>

void setup(void)
{
    if (Morikawa.setup() == TSTERROR_OK) {

        // use debug serial
        Serial.begin(9600);
        Serial.println(F("Morikawa says \"hello space\"."));
    }
    else {
        Morikawa.shutdown();
    }
    return;
}

void loop(void)
{
    Morikawa.loop();

    // use morse player
    Morikawa.playMorsePGM(NOTE_C6, PSTR("hello space"));
    delay(1000);

    // use voice speaker
    Morikawa.speakPhrasePGM(PSTR("konnnichi'wa uchu-"));
    delay(1000);
    return;
}
```

Morikawa App

1. Hello, space!
2. Speech Text
3. Code Text
4. Play Melody
5. Invader Bot (Space Eliza)
6. Invader Music
7. Invader Cam
8. INVADER VM
9. Telemetry Dump

MorikawaApp | HelloSpace | SelfTest

```
#include <MorikawaSDK.h>

#define REGISTER_APP(param)\
    g_setup = &param##_setup;\
    g_loop = &param##_loop;

static void(*g_setup)(void) = NULL;
static void(*g_loop)(void) = NULL;

void setup(void)
{
    if (Morikawa.setup() == TSTERROR_OK) {
        switch (Morikawa.getBootMode()) {
            case 1:
                REGISTER_APP(HelloSpace)
                break;
            case 2:
                break;
            case 0:
            default:
                REGISTER_APP(SelfTest)
                break;
        }
        (*g_setup)();
    }
    else {
        Morikawa.shutdown();
    }
    return;
}

void loop(void)
{
    Morikawa.loop();
    (*g_loop)();
    return;
}
```

MorikawaApp | Arduino 1.0.5

MorikawaApp | HelloSpace | SelfTest

```
static char const PROGMEM g_morse[] = "hello space";
static char const PROGMEM g_speak[] = "konnnichi'wa";

void HelloSpace_setup(void)
{
    // use debug serial
    Serial.begin(9600);
    Serial.println("Morikawa says \"hello space\".");
    return;
}

void HelloSpace_loop(void)
{
    // use morse player
    Morikawa.playMorsePGM(NOTE_C6, g_morse);
    delay(1000);

    // use voice speaker
    Morikawa.speakPhrasePGM(g_speak);
    delay(1000);
    return;
}
```

Done compiling.

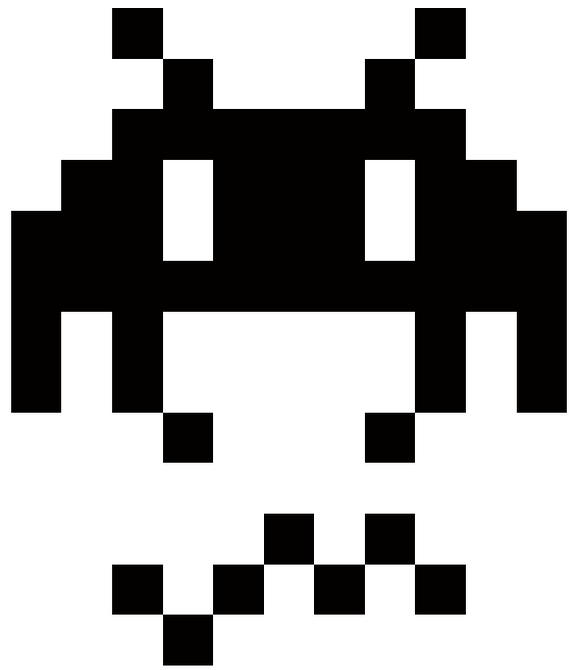
```
Binary sketch size: 13,600 bytes (of a 258,048 byte maximum)
```

8 Mega Pro 2560V 3.3V on /dev/tty.usbmodemfd131

Morikawaのリプログラム

軌道上でのプログラム書き換え

InvaderVM



レジスタ
- 0: REG_ERRN
- 1: REG_FUNC
- 2: REG_RETV
- 3: REG_HCUR
- 4: REG_HEAP
- 5〜15: REG_ARG0〜A

コード
- VM_END
- VM_NOP
- VM_SET
- VM_SETC

Morikawaアプリとしての VM

README.md

iva

InvaderVM Assembler

example

```
# load string into the heap
CLR,   HCUR
TXT,   11,   <NUM VAL=3>

# initialize registers
SETC,  FUNC, speakPhrase

# speakPhrase() argument
SETC,  ARG0, 11

# counter
SETC,  ARG1, 3

# 1000 millisec
SETI,  ARG2, 1000

# while (ARG1 > 0)
begin:
CLR,   HCUR
CALL

WAIT,  ARG2

SETC,  HCUR, 9
DEC,   ARG1
DEC,   HEAP
JMPIF, ARG1, begin:
# end while

CLR,   HCUR
TXT,   18,   konnnichi'wa uchu-

SETC,  ARG0, 18
CALL

END
```

```
00000020  03 08 1e 07 03 03 09 0d  06 0d 04 0a 06 f1 05 03  |................|
00000030  1f 12 00 00 00 6b 6f 6e  6e 6e 69 63 68 69 27 77  |.....konnnichi'w|
00000040  61 20 75 63 68 75 2d 03  05 12 08 00              |a uchu-.....|
```

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP

Rubyで記述されたVMアセンブラ

ARTSAT

TOKYO, JAPAN http://artsat.jp/

Find a repository…

iva

Ruby ★0 ⑂0

Invader VM Assembler

Updated 18 days ago

MorikawaApp

Arduino ★2 ⑂1

The mission applications for INVADER.

Updated 19 days ago

MorikawaSDK

C++ ★0 ⑂0

An arduino base application programming environment for satellite's mission.

Updated a month ago

ground_station

C++ ★0 ⑂0

Ground Station software for ARTSAT under development.

Updated 3 months ago

ofxARTSATAPI

C ★8 ⑂0

An openFrameworks addon to support accessing to the satellite.

Updated 4 months ago

Members 2 >

toolbits
HORIGUCHI Junshi

h2so5
h2so5

http://github.com/ARTSAT

2014.2.28 03:07～

H-IIA 23号機

GPM相乗り打ち上げ

INVADER衛星　主要運用イベント計画

実施時期	イベント名	内容
初期運用	データ送受信確立の確認	a) CW音の確認とCW音の解析
		b) CW音が聞こえた ∧ 日照パスであれば、アンテナ展開停止コマンドの送信（同時に送受信リンクの確認となる）
		c)（以降、コマンドを打つのは日照パスのみ）コマンドによるHouse Keepingデータの取得
		d) 電力収支の成立確認
3日目以降 (軌道情報が 安定してから)	ミッション運用1	a) コマンドによりミッションデータ（API配信用テレメトリデータ）の取得 b) 衛星botの運用（衛星搭載人工無能の動作）
3月以降 (ミッション運用の電力収支成立性の確認後)	ミッション運用2	a) デジトーカーによる音声通信 b) ミッションカメラの駆動と画像取得 c) VMによる軌道上プログラム書き換え試験
再突入時（9月T.B.D）	ミッション運用3	a) 衛星のCWを遺言モードに変更し、アマチュア無線家と共同の受信コンテスト

ARTSAT