ECHONET Lite 機器用通信ミドルウェア

TOPPERS/ECNLの使い方

2014年6月24日

Cores コアーズ株式会社
長島 宏明

ECHONET Lite とは

- 一般家電製品や設備機器向けホームネットワークの規格
  - エコーネットコンソーシアムで規定
  - http://www.echonet.gr.jp/
  - 創エネ、畜エネ、省エネのための通信プロトコル
  - マルチベンダで相互接続を実現したい
- ECHONET Lite規格書
  - 規格は一般に公開されている
  - http://www.echonet.gr.jp/spec/spec_v110_lite.htm

ECHONET Lite 通信ミドルウェアの概略

- ECHONET Lite規格書は５部とAPPENDIX
- 第２部「ECHONET Lite 通信ミドルウェア仕様」が実装対象
- アプリケーションソフトウェアでは、APPENDIX「ECHONET機器オブジェクト詳細規定」を参照する
- ECHONET Lite機器
  - 大きく分けて機器とコントローラがある
  - 本ミドルウェアでは機器を対象とする

ECHONET Lite規格の認証

- 開発した機器のECHONET Lite規格の適合確認
  - 神奈川工科大学のHEMS（ECHONET Lite）認証支援センターで確認
  - http://sh-center.org/
- 規格適合性認証
  - コンソーシアムに入会し、メーカーコードを取得
  - 規格認証認定機関で書面審査による自己認証

本ミドルウェアの対象範囲

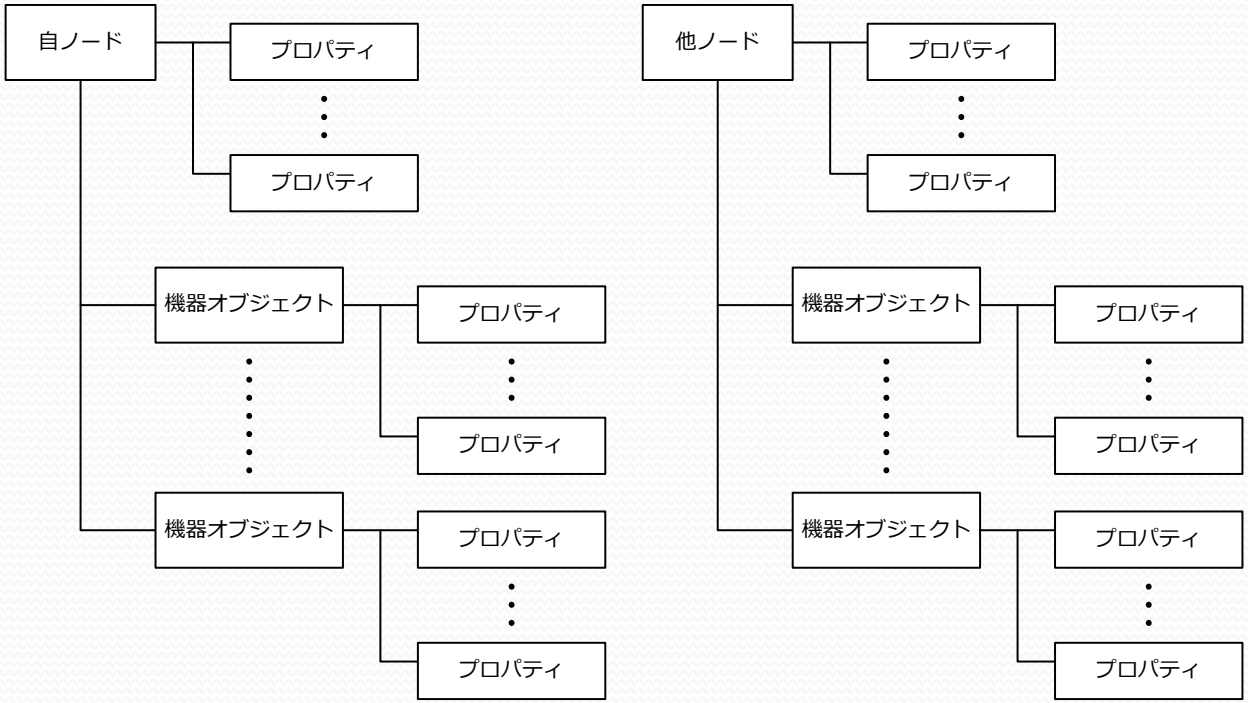
- 設計時に既知である制御や監視を行う機器向け
  - コントローラとしては機能不足
- 機器オブジェクトの定義
  - TOPPERS新世代カーネル用コンフィギュレータを使用し、静的APIで定義する
- ECHONET Liteの通信処理
  - 定義した機器オブジェクトの通信処理
  - プロパティ設定と取得コールバックの呼び出し
- IPv4のUDPによる通信処理
  - TINETを利用し、ECHONET用のUDP通信ポートを使用

アプリケーション作成にあたり

- ミドルウェアのソフトウェア構成
  - ノード・オブジェクト・プロパティの階層構造
  - 自ノード、他ノード
  - 他ノード同期型、非同期型
  - サービス処理タスク・UDP通信処理タスク・アプリケーションタスク
- ユーザーが作成するもの
  - ノード・オブジェクト・プロパティの定義
  - プロパティ設定・取得コールバックの作成
  - アプリケーションタスクの作成

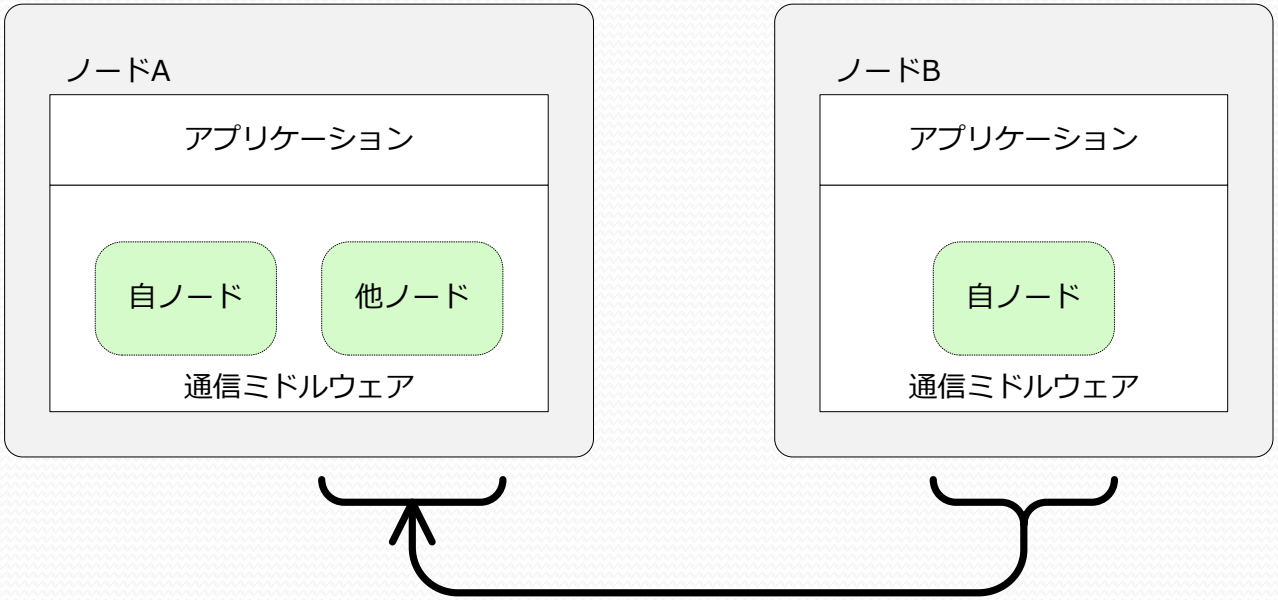
階層構造

- ノードの配下にいくつかのオブジェクトを持つ
- オブジェクトにいくつかのプロパティを持つ



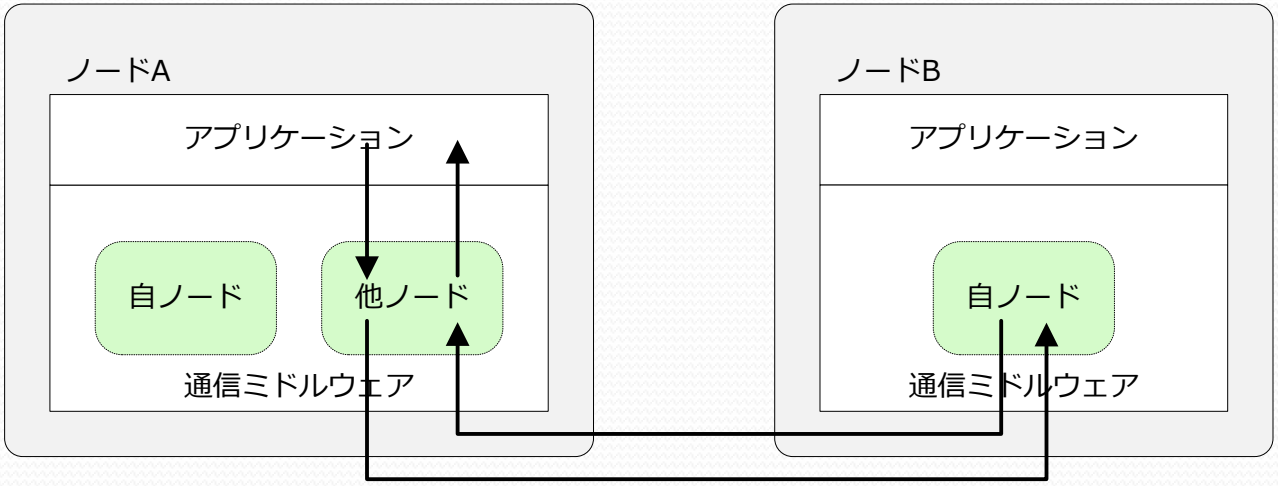
自ノード、他ノード

- 自機器にある機能は自ノード配下のオブジェクト
- 他機器にある機能は他ノード配下のオブジェクト
- アプリケーションからは等しく見える



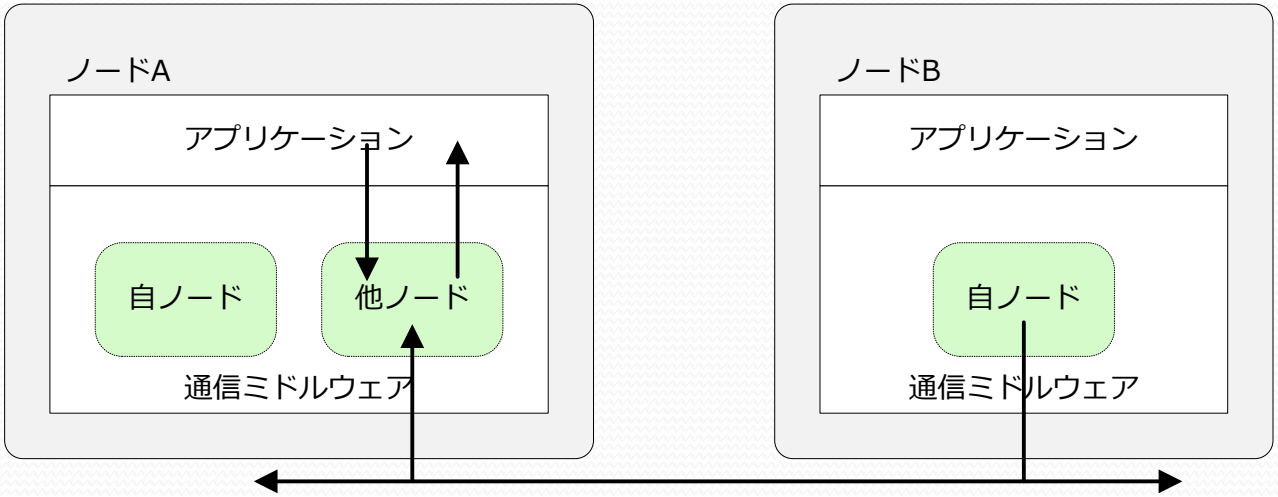
他ノード同期型

- アプリケーションからのプロパティ要求で、他機器との通信を行う



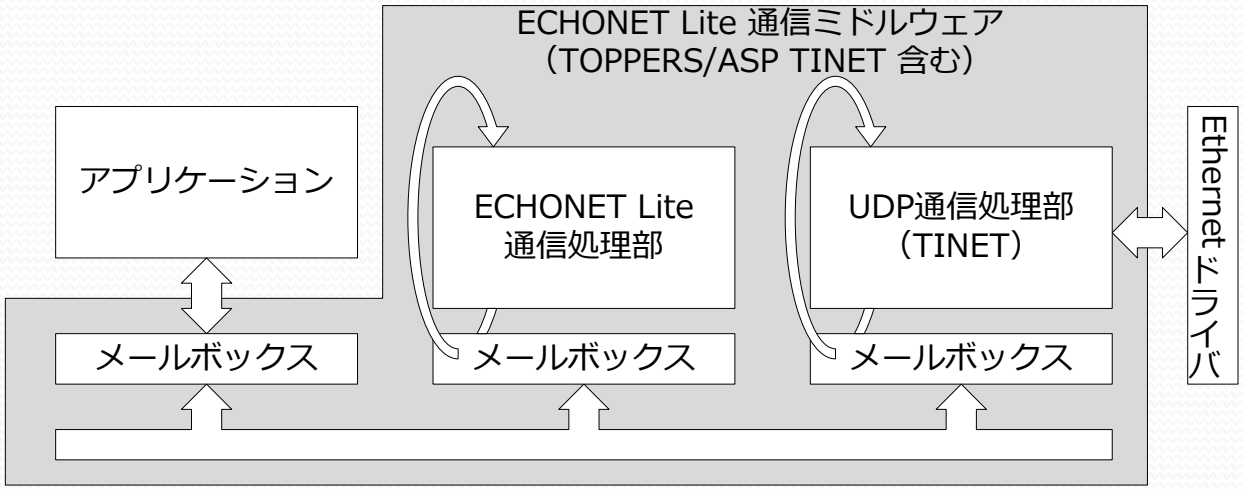
他ノード非同期型

- アプリケーションからの要求には、自機器内で持っているプロパティ値を返す
- 他機器からの通知されたプロパティ値を保存してく



タスク構成

- ECHONET Liteの電文はサービス処理タスク
- 下位レイヤーとの中継ぎはUDP通信処理タスク
- タスク間通信はメールボックス



アプリケーション作成

アプリケーションの作成方法

機器オブジェクトの定義

- 規格書のAPPENDIXから、目的の機器オブジェクト選び、対応するクラスグループコード・クラスコードなどで、静的APIのECN_CRE_EOBJを使用し機器オブジェクトを定義する

３．３．２４　一般照明クラス規定

クラスグループコード：0x02
クラスコード：0x90
インスタンスコード：0x01～0x7F

```
/*
 * 一般照明オブジェクト
 */
ECN_CRE_EOBJ (GENERAL_LIGHTING_EOBJ, { EOBJ_DEVICE, LOCAL_NODE_EOBJ, 0,
EOJ_X1_AMENITY, EOJ_X2_GENERAL_LIGHTING_CLASS, EOJ_X3_GENERAL_LIGHTING_CLASS_1 });
```

プロパティの定義

- APPENDIXのプロパティ定義から、必須のものと選択したものを静的APIのECN_DEF_EPRPで定義する
- プロパティ値の設定と取得の際にミドルウェアから呼ばれる、コールバック関数を定義する

プロパティ名称	EPC	プロパティ内容 値域	データ型	データサイズ	アクセスルール	状態変化時アナウンス	
動作状態	0x80	・・・	unsigned char	1 Byte	Get/Set	○	

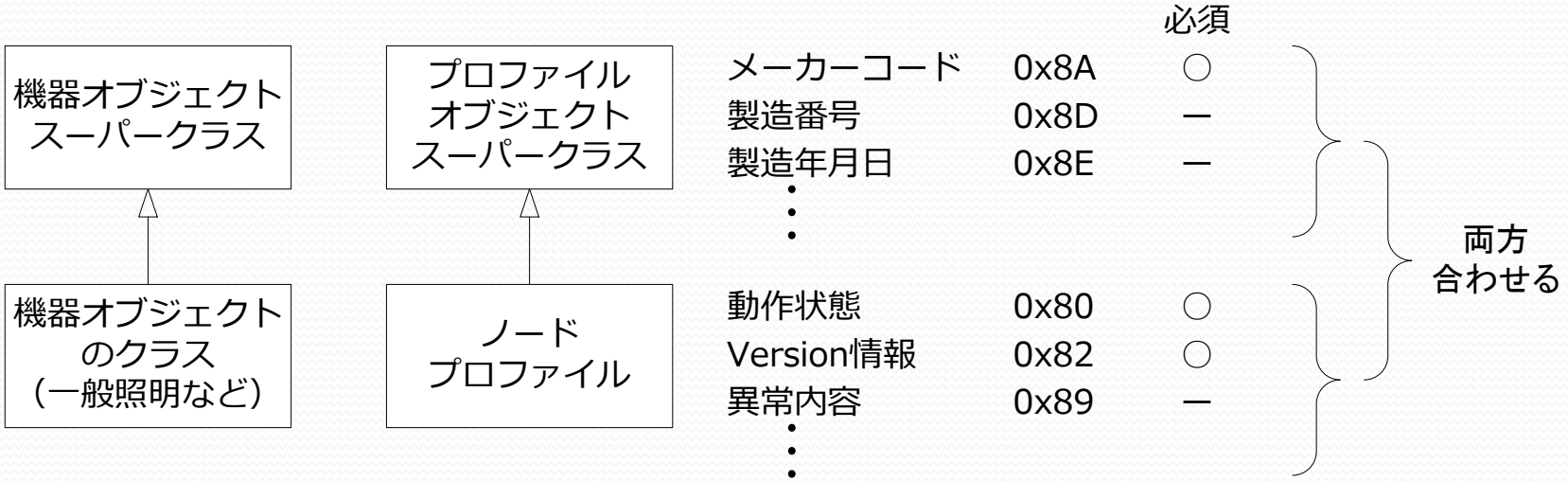
```
/* 動作状態 */
ECN_DEF_EPRP (GENERAL_LIGHTING_EOBJ, { 0x80, EPC_RULE_SET | EPC_RULE_GET |
EPC_ANNOUNCE, 1, (intptr_t)&epc_data[0], (EPRP_SETTER *)data_prop_set, (EPRP_GETTER
*)data_prop_get });
```

プロパティの定義

- プロパティ構成は継承される
- コンフィギュレータが自動生成するプロパティ
- プロパティのコールバック関数の定義
  - ポートを直接操作する方法
  - ポート状態を自ノードへ書き込みする方法

プロパティ構成の継承について

- 機器オブジェクトは、機器オブジェクトスーパークラスから、ノードプロファイルは、プロファイルオブジェクトスーパークラスから継承
- 継承元と先の両方のプロパティが必要、ただし必須でないものは任意に定義

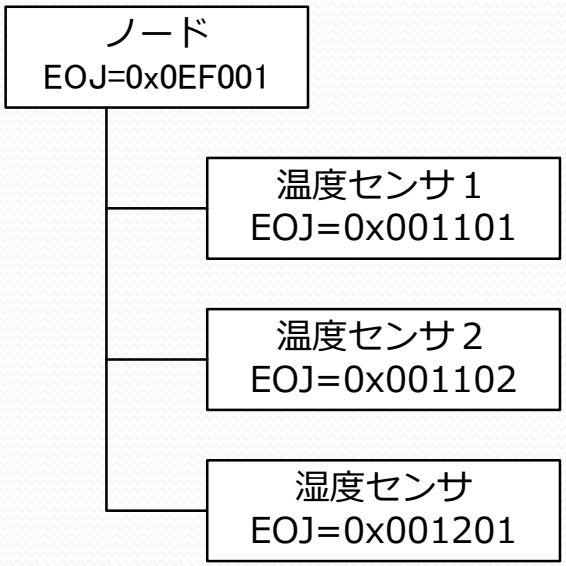


自動生成するプロパティ

- ノードプロファイル
  - インスタンス数（0xD3）
  - クラス数（0xD4）
  - インスタンスリスト通知（0xD5）
  - インスタンスリストS（0xD6）
  - クラスリストS（0xD7）
- オブジェクト
  - 状態アナウンスプロパティマップ（0x9D）
  - Setプロパティマップ（0x9E）
  - Getプロパティマップ（0x9F）
- ユーザーが定義する必要はない

インスタンスリストの生成

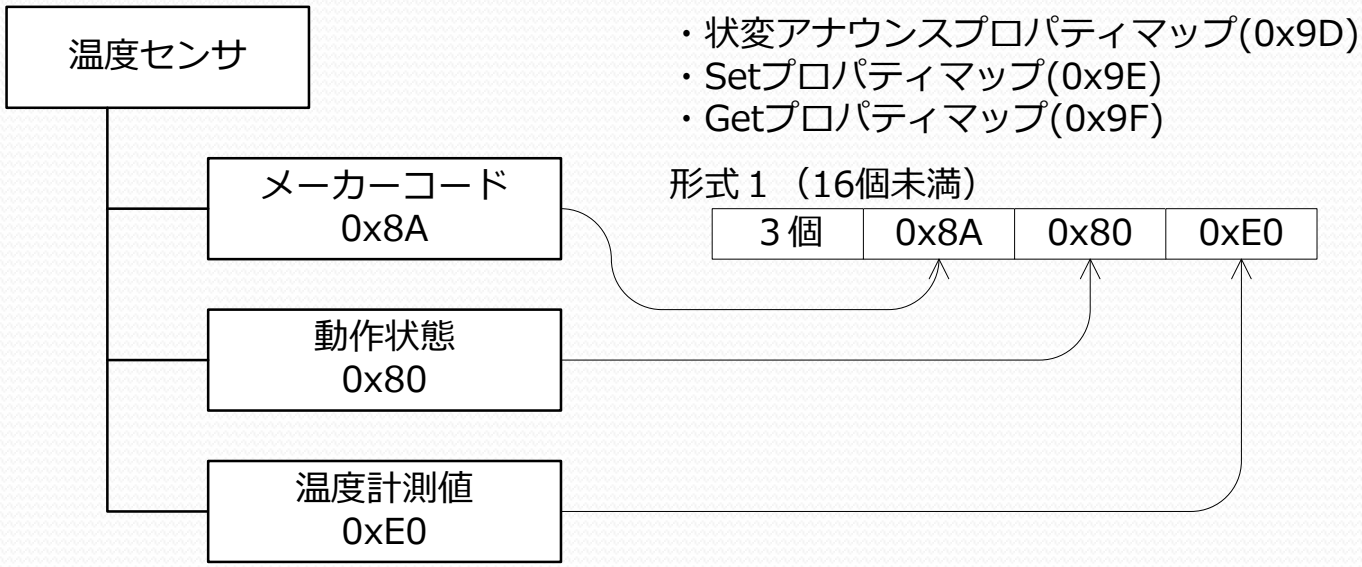
- ノードの自動生成プロパティ
  - 自ノードのクラス、オブジェクトの情報



・自ノードインスタンス数(0xD3)
：0x000003（温度センサ×２、湿度センサ×１）
・自ノードクラス数(0xD4)
：0x0003（ノードプロファイル、温度センサ、湿度センサ）
・インスタンスリスト通知(0xD5)
：0x030011010011020012 01
・自ノードインスタンスリストS(0xD6)
：0x030011010011020012 01
・自ノードクラスリストS(0xD7)
：0x0200110012

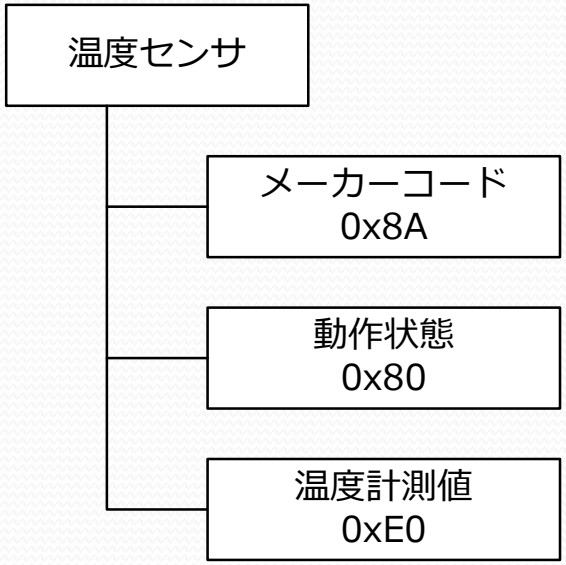
プロパティマップの生成

- オブジェクトの自動生成プロパティ
  - 形式１（プロパティ数が１６未満の場合）



プロパティマップの生成

- オブジェクトの自動生成プロパティ
  - 形式２（プロパティ数が１６以上の場合）



形式２（16個以上）

個数	16Byteのビットマップ

ビットマップ
プロパティありで対応Bitが1

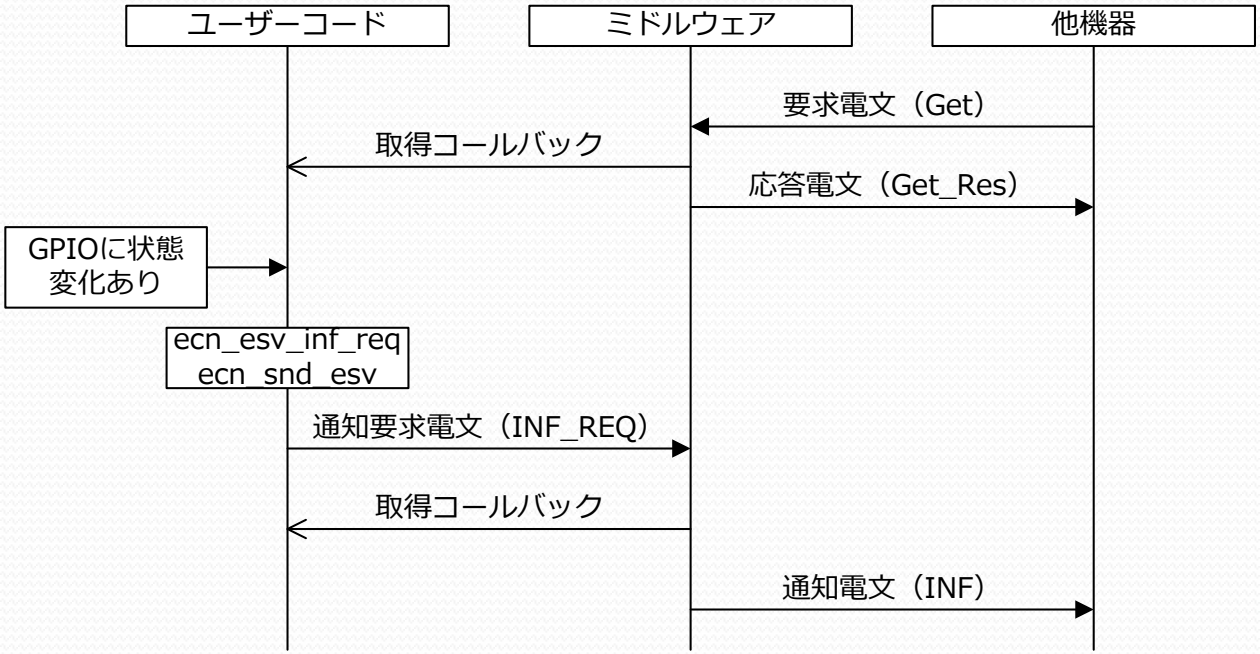
	Bit7	Bit6	Bit5	・・・	Bit0
2Byte目	0xF0	0xE0	0xD0	・・・	0x80
3Byte目	0xF1	0xE1	0xD1	・・・	0x81
・ ・ ・	・ ・ ・	・ ・ ・	・ ・ ・		・ ・ ・
17Byte目	0xFF	0xEF	0xDF	・・・	0x8F

コールバック関数の定義

- プロパティの設定や取得は、ミドルウェアからのコールバック関数の呼び出しで行う
- プロパティを保存する領域は、ユーザーが用意する
- プロパティの設定をポートへ直接書き込み、プロパティの取得をポートから直接読み出す場合などは、保存領域を用意しなくても良い

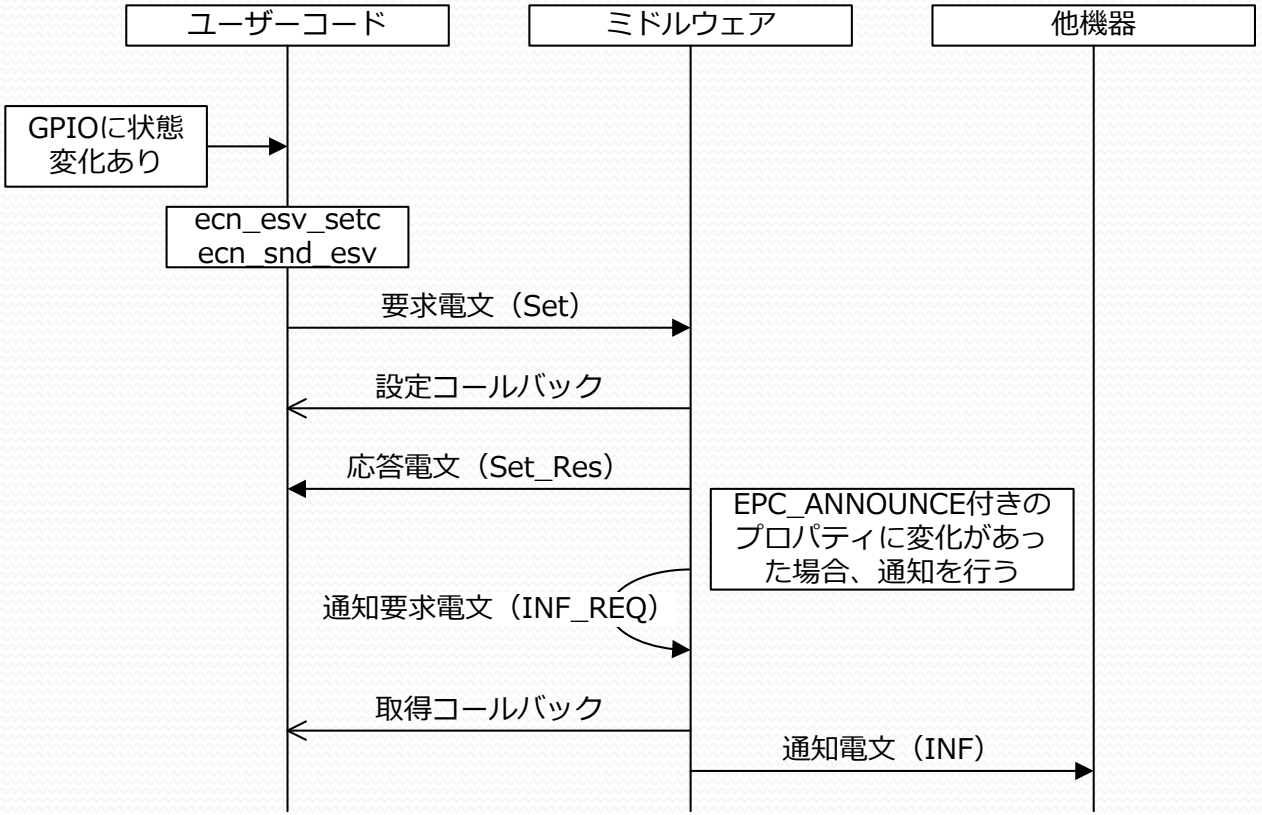
ポートを直接操作する場合

- アプリケーションタスクはポートの状態を監視し、変化があった時は、通知要求で通知する



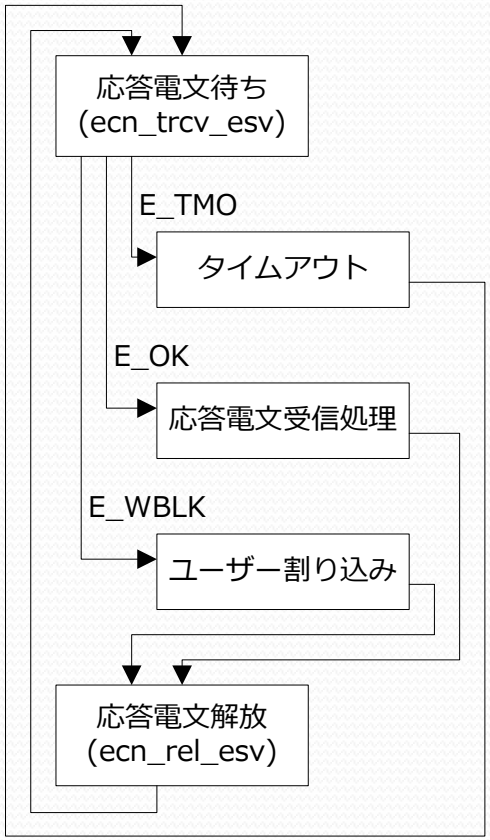
自ノードへ書き込みする場合

- アプリケーションタスクはポートの状態を監視し、変化があった時は、書き込み要求で書き込む



アプリケーション

- ユーザーが定義するタスク
  - ミドルウェアの起動
  - 応答電文受信を待つループ
  - 必要に応じてポートの監視
- サービス電文の送信
- サービス電文待ちの割り込み



ユーザー定義タスク

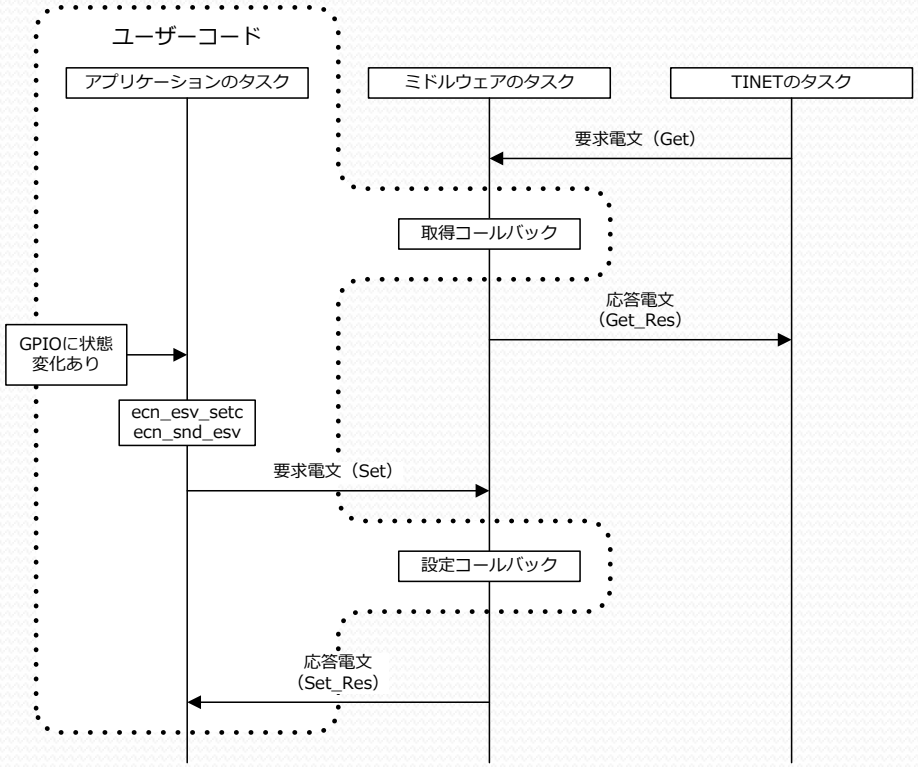
- ミドルウェア起動
  - タスク起動：act_tsk(ECN_UDP_TASK)
- 応答電文受信待ち
  - 永久待ち：ecn_rcv_esv
  - ポーリング：ecn_prcv_esv
  - タイムアウト付：ecn_trcv_esv
- ポートの監視など
  - タイムアウトでポート監視などを行う

サービス電文の送信

- ECHONET Lite要求電文の作成
  - プロパティ値書き込み要求 Get：ecn_esv_get
  - プロパティ値読み出し要求（応答不要） SetI：ecn_esv_seti
  - プロパティ値読み出し要求（応答要） SetC：ecn_esv_setc
  - その他にecn_esv_inf_req、ecn_esv_set_getなど
- プロパティの追加
  - 読み出しではecn_add_epc
  - 書き込みではecn_add_edt
- 電文送信
  - 上記で作成した電文を送信：ecn_snd_esv

サービス電文待ちの割り込み

- プロパティ設定・取得コールバックは、ミドルウェアのタスクコンテキストで呼ばれる



サービス電文待ちの割り込み

- ユーザー定義タスクの電文受信以外での待ち解除
  - コールバック関数内：ecn_brk_wai
  - ユーザー定義タスク内：ecn_get_brk_dat

サンプル アプリケーション

サンプルアプリケーションの説明

ハードウェア

- TK-850/JH3-E+NET
  - テセラ・テクノロジー製
  - CPU：V850ES/JH3-E
  - 動作クロック：48MHz
  - 内蔵ROM：512KByte
  - 内蔵RAM：61KByte
  - 内蔵データRAM：64KByte
  - Ethernetインターフェイス付き
  - EZ Emulator付きで、ボード単体でデバッグ可能

開発環境

- 統合環境
  - Renesas CubeSuite+
- ツールチェイン
  - CA850（V850用コンパイラ）
  - C言語のみのコンパイラ（C++に未対応）
  - StdLibあり
- 無償版を使用
  - 実行領域のサイズに256KByteまでの制限あり
  - サンプルプログラムは80KByte程度
    （ASP+TINET+ECLN+アプリケーション）

コードサイズ詳細

- サンプルアプリのコードサイズ（参考）

割り込みベクター	1,940	Byte
ROMデータ	12,870	Byte
実行プログラム	79,564	Byte
Ethernet用バッファ	12,496	Byte
RAMデータ	44,876	Byte

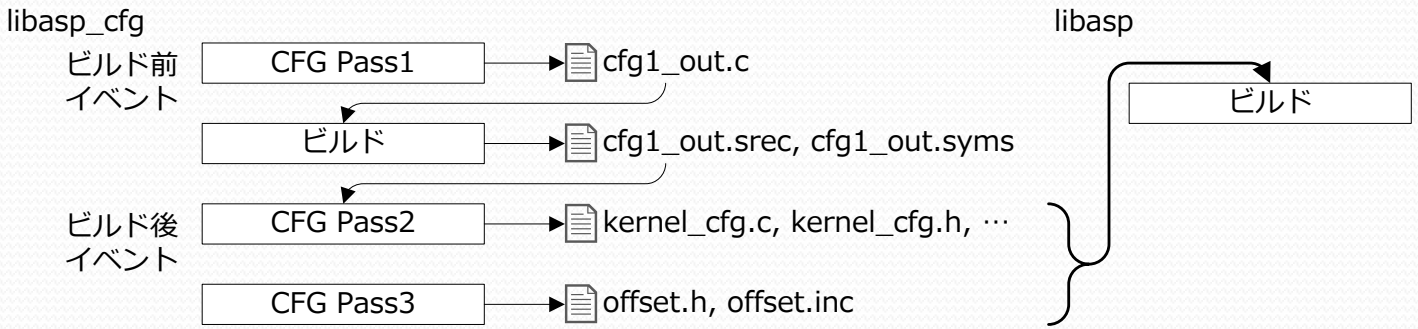
	ROMデータ	実行プログラム	RAMデータ	
TOPPERS/ASP	4,590	39,214	5,182	Byte
TINET	1,380	19,180	900	Byte
TOPPERS/ECNL	1,784	15,720	1,912	Byte
サンプル	5,116	4,680	36,876	Byte

プロジェクトの構成と作成

- フォルダ構成
  - CubeSuite+のプロジェクトをcsp配下に置いた

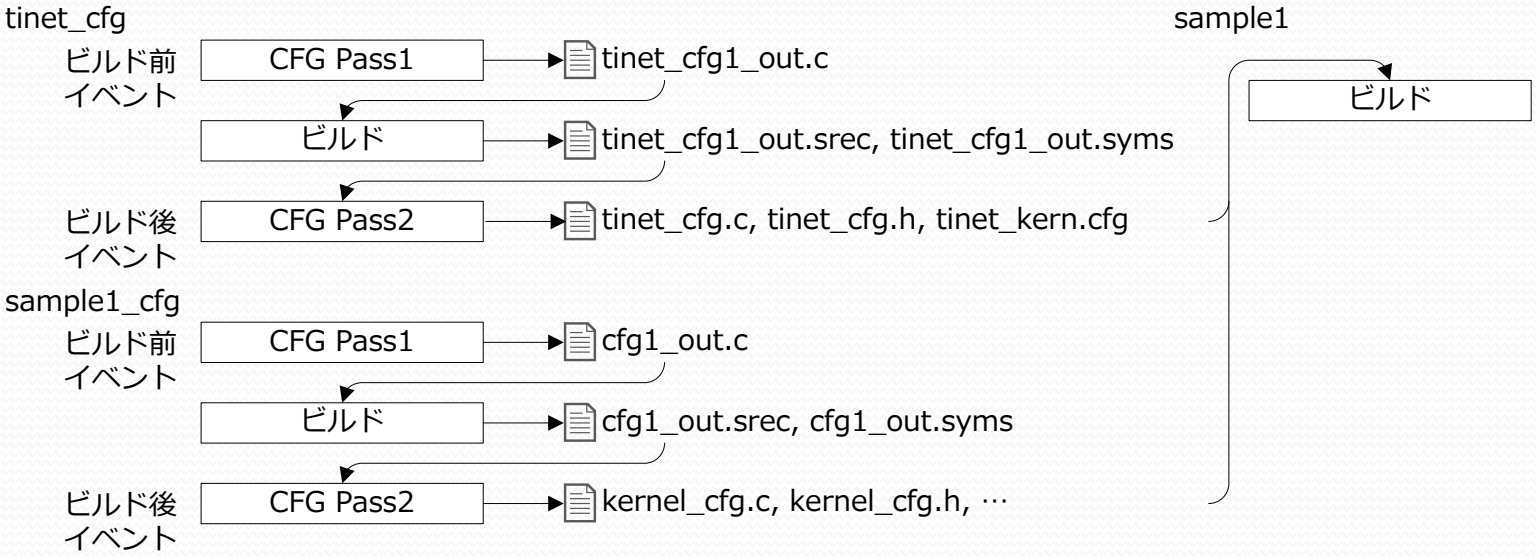
コンフィギュレーション

- ASPライブラリ
  - 複数のサンプルプログラムから再利用できるようASPをライブラリ化
  - コンフィギュレーション用のサブプロジェクトを作成
  - ビルド・イベントでコンフィギュレーションを実行



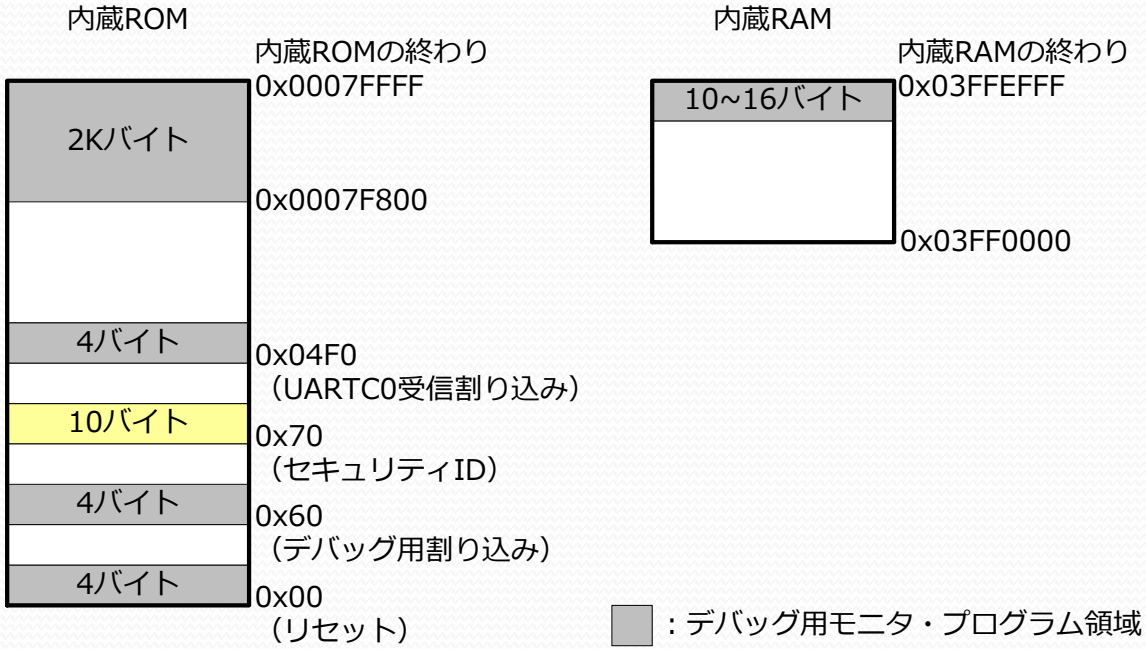
コンフィギュレーション

- アプリケーション
  - ASPライブラリと同じようにサブプロジェクトを作成
  - TINETは２回必要なため、１回目用のサブプロジェクトも作成



EZ Emulatorのための変更

- デバッグ用モニタ・プログラム領域
  - 必要な領域を空けておき、レジスタ設定も必要



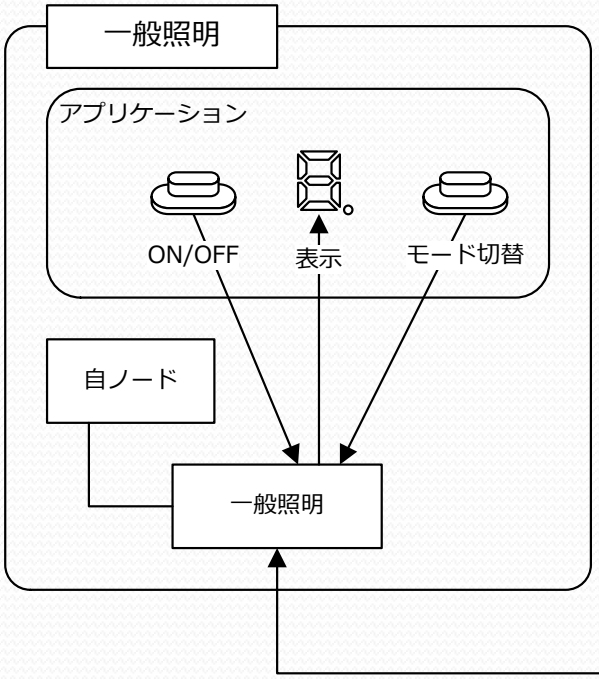
参照元> QB-MINI2 プログラミング機能付きオンチップ・デバッグ・エミュレータ ユーザーズマニュアル

サンプルアプリケーション

- サンプルアプリケーションとして、連携するものを作成
- 一般照明、コントローラ
  - コントローラで一般照明のON/OFF
- 一般照明、人体検知センサ
  - 人体検知センサで一般照明のON/OFF
- 家庭用エアコン、温度センサ
  - 家庭用エアコンが温度センサを監視
- 電気ポット、ブザー
  - 電気ポットが操作や状態を通知し、ブザーが鳴動

コントローラ、一般照明

- ECHONETコンソーシアムで提供しているSDK
  - SSNGをコントローラとして使用し、動作状態のON/OFFや、点灯モード設定（B6）を取得、設定

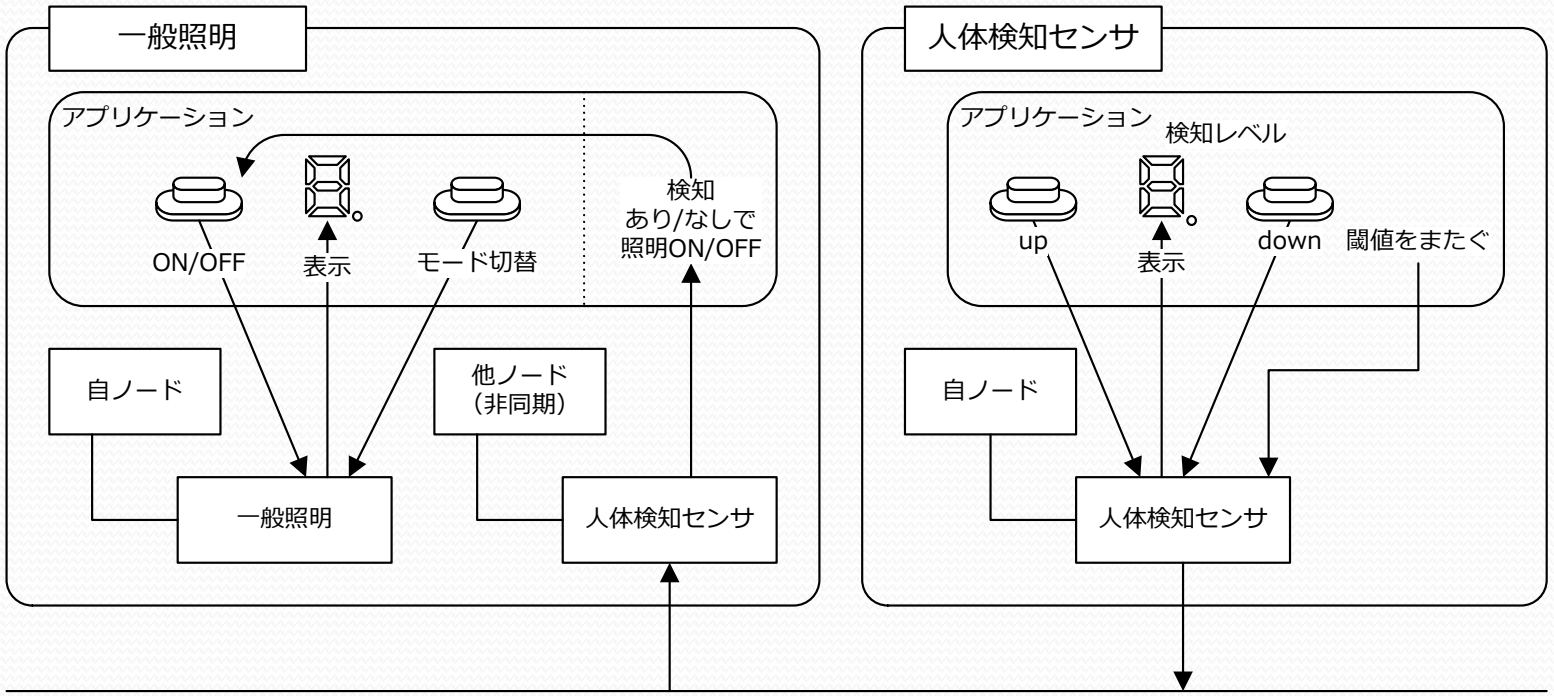


コントローラ、一般照明

- 一般照明
  - 他機器として定義を持たないコントローラからの、プロパティ操作の受付
  - 動作状況プロパティ設定コールバックで７セグの表示をON/OFF
  - 点灯モード設定プロパティ設定コールバックで７セグの表示を点灯モードに対応する「A」「b」「c」「E」に

一般照明、人体検知センサ

- 人体検知センサはボタン２つで検知レベルを上下させ、閾値を跨いだら一般照明をON/OFF

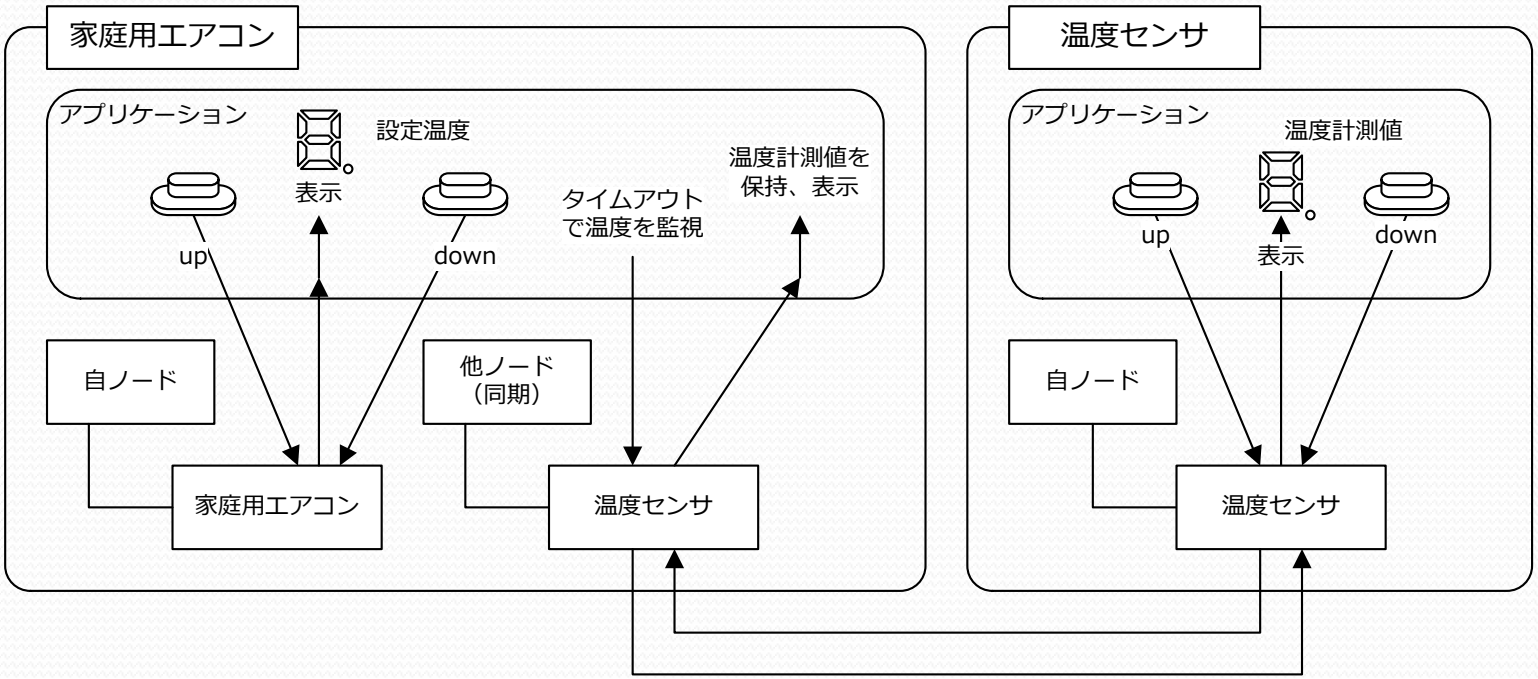


一般照明、人体検知センサ

- 一般照明
  - 動作状況プロパティ設定コールバックで７セグの表示をON/OFF
- 人体検知センサ
  - アプリケーションタスクで、ボタンを定期的に監視
  - ボタン操作で、自ノードの検知レベルをSetC要求電文で設定
  - 検知レベルプロパティ設定コールバックで、閾値を跨いだ時に、SetC要求電文で一般照明をON/OFF

家庭用エアコン、温度センサ

- 家庭用エアコンは定期的に温度センサを監視
- 温度センサは２つのボタンで温度を上げ下げ

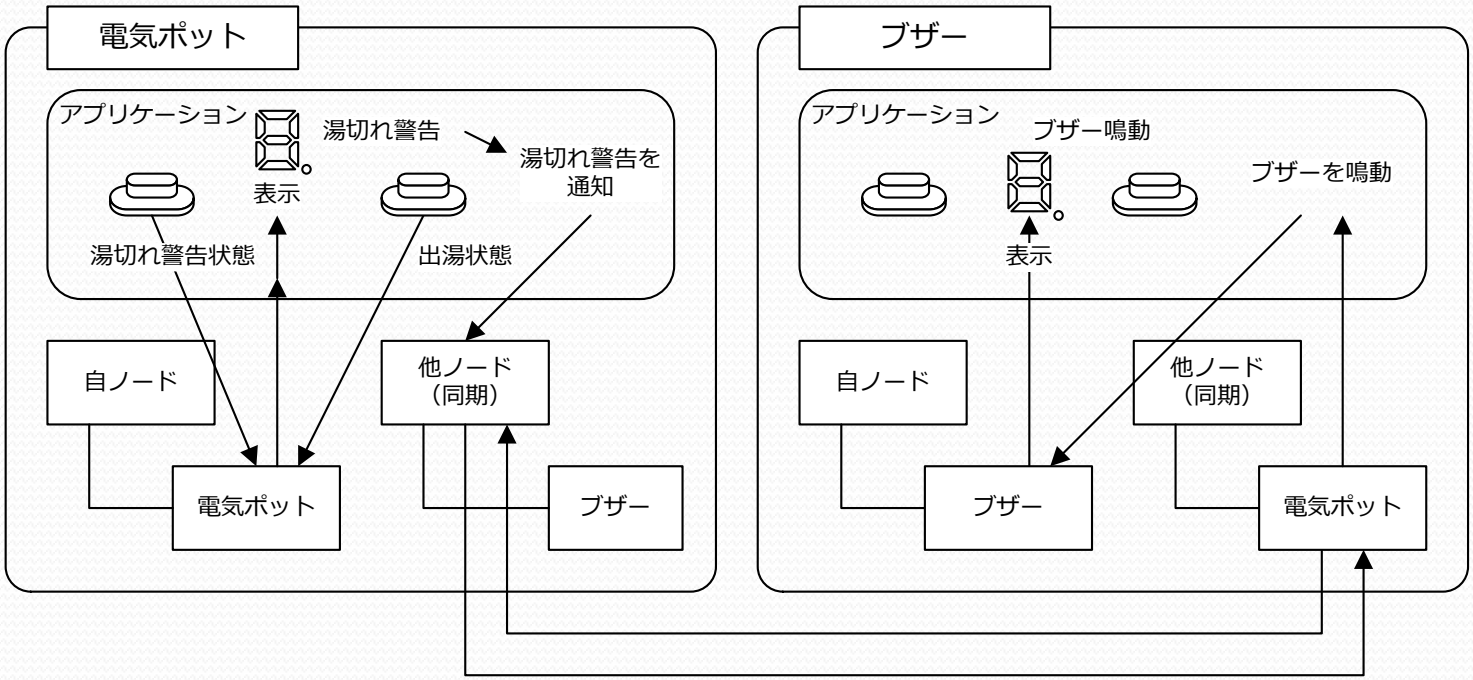


家庭用エアコン、温度センサ

- 家庭用エアコン
  - アプリケーションタスクを一定間隔でタイムアウトさせ、Get要求電文で、温度センサから温度計測値プロパティを読み出し
- 温度センサ
  - アプリケーションタスクで、ボタンを定期的に監視
  - ボタン操作で、自ノードの温度計測値プロパティをSetGet要求電文で設定し、応答電文で設定値を確認
  - 温度計測値プロパティ設定コールバックで７セグの表示内容変更

電気ポット、ブザー

- 電気ポットはボタンを押すと湯切れ警告状態になり、通知電文を送信、ブザーで受け取ってブザー鳴動

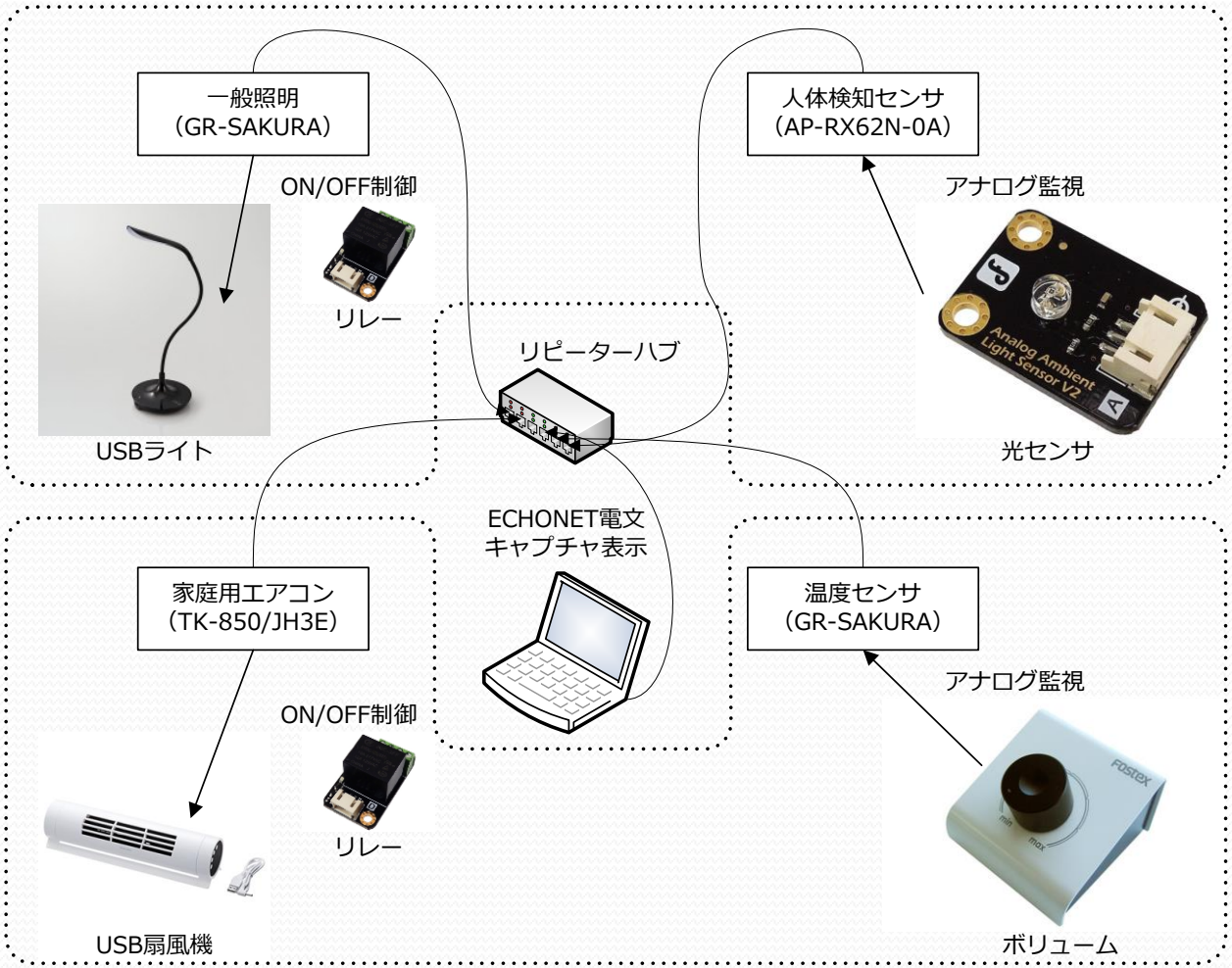


電気ポット、ブザー

- 電気ポット
  - アプリケーションタスクで、ボタンを定期的に監視
  - ボタン操作で、自ノードの湯切れ警告状態プロパティをSetI要求電文で設定
  - 湯切れ警告状態プロパティ設定コールバックでInfoC要求電文でブザーに通知
- ブザー
  - 他ノード（電気ポット）の湯切れ警告状態プロパティ設定コールバックで、ブザー鳴動状態を７セグで表示

ESEC2014

- 展示物



今後必要になりそうなもの

- IPv6への対応
  - 920MHz帯の無線や電力線通信（IEEE802.15.4）
  - TTC JJ-300.10規格（Wi-SUN、ZigBee）
  - TTC JJ-300.11規格（PLC）
  - IPv6と6LoWPANが規格に含まれている
    （暗号化に関する規格も含まれる）
- 関連プロトコルの追加
  - IGMP/MLDへの対応（マルチキャストの管理）
  - DHCPへの対応（IPアドレス自動取得）
- TCPへの対応（規格書1.10で追加）

課題、展望

- APPENDIXバージョンの複数対応
  - コンフィギュレータでの必須プロパティチェック
- コンフィギュレータのエラー表示
  - cfgファイルの記述ミスを修正しやすいような表示
- 動的に増減する機器への対応（コントローラ）
  - オブジェクトやプロパティなどを運用時に組み替える必要がある
  - mrubyやNETMFなどのオブジェクトとして対応する

ありがとうございました

Cores