0111

ダイアグラム

自動的に生成された説明

現在の関係図である.

ラズパイにはEnddeviceと光センサが接続されている.

1. 光センサのセンサ値をラズパイを介して, Enddeviceが取得する.
2. センサ値とEnddeviceが取得したデータをラズパイで表示し、確認する.
3. Enddeviceはセンサ値をCoordinatorへ送信する.
4. Coordinatorで受信したセンサ値を表示し, 確認する.

【前回】

16進数のセンサ値をEnddeviceからCoordinatorへ送信し, Coordinator側で受信したセンサ値を16進数として表示することができていた.

【今回】

Coordinator側で受信したセンサ値を16進数ではなく, 10進数で表示するようにする.

そこで, Coordinatorのコードに、受信した16進数を10進数に変換するビット演算を追加した.

以下はCoordinator側の受信したデータを処理するapp\_coordinator\_endpoint.cである.

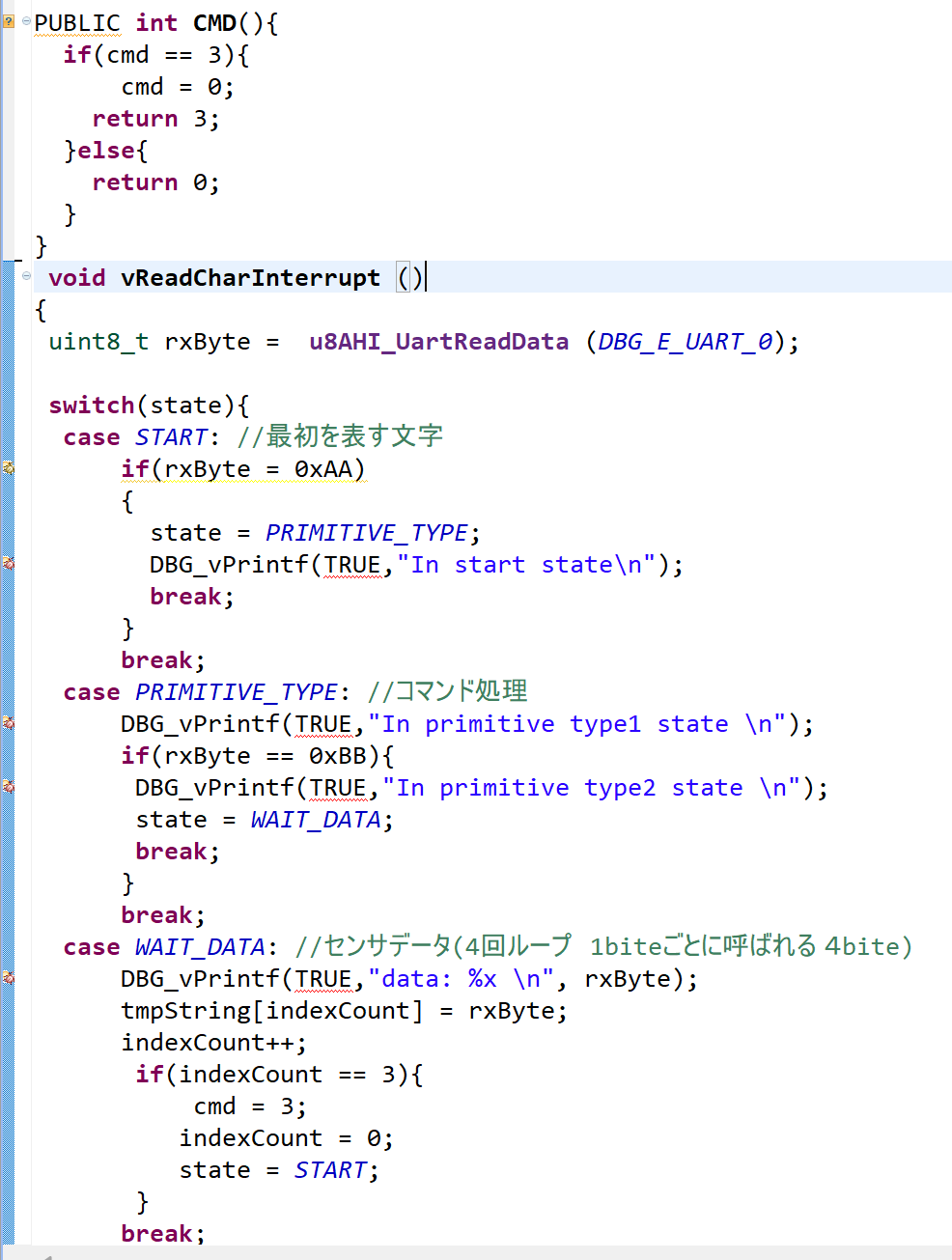
テキスト

自動的に生成された説明

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

以下はEnddeviceとCoordinatorどちらも共有するutils.cである.



以下はラズパイとCoordinator側の実行結果である.

テキスト

自動的に生成された説明テーブル

自動的に生成された説明

上記の結果から, 受信した16進数を表すreceived data16とラズパイのセンサ値が一致していない. 期待表示内容は

received data16 : 00dbf9

received data10：56313

しかし, この「56313」もラズパイに表示してあるセンサ値と一致していない.

〇decimal = (decimal << 4) | currentDigit;　について

16進数の各桁を10進数に変換している.

具体的には、(decimal << 4) は decimalのビットを左に4回シフトし, これにより16進数の各桁を10進数に変換している. そして, | currentDigit は現在の16進数の桁を加えている.

例えば, 1A を変換する場合だと

1. 最初に decimalは 0 である.

2. '1' を処理すると, (0 << 4) | 1 となり, decimal は 1 になる.

3. 'A' を処理すると, (1 << 4) | 10 となり, decimal は 26 になる.

これにより, 16進数 1A は 10進数の 26 に変換できる.

アルベルト先生からは,

16進数を受け取るときは, int8\_tの配列 → 1つまたは複数のint32\_tの値

の変換が必要とのことだったが, 実装できなかったため, 上記のコードで試した.

再度検討すべき.