

# μIPを用いたセンサネットワークの効率化

木下研究室 上甲 薫 200602835

# □ 背景


過去に本研究室において、セキュリティシステムの研究として、センサによる侵入者経路検知の研究が行われた。

この研究では、大量のデータを得る必要があった。そのために課題として、データ取得効率の向上が挙げられている。


そこで本研究では、μIPプロトコルスタックの実装による、高速データ取得システムを提案する。

# □ 目的

低コストで導入できる超高輝度LEDをセンサとし、検知対象の動作検知を可能にするシステムを考案。

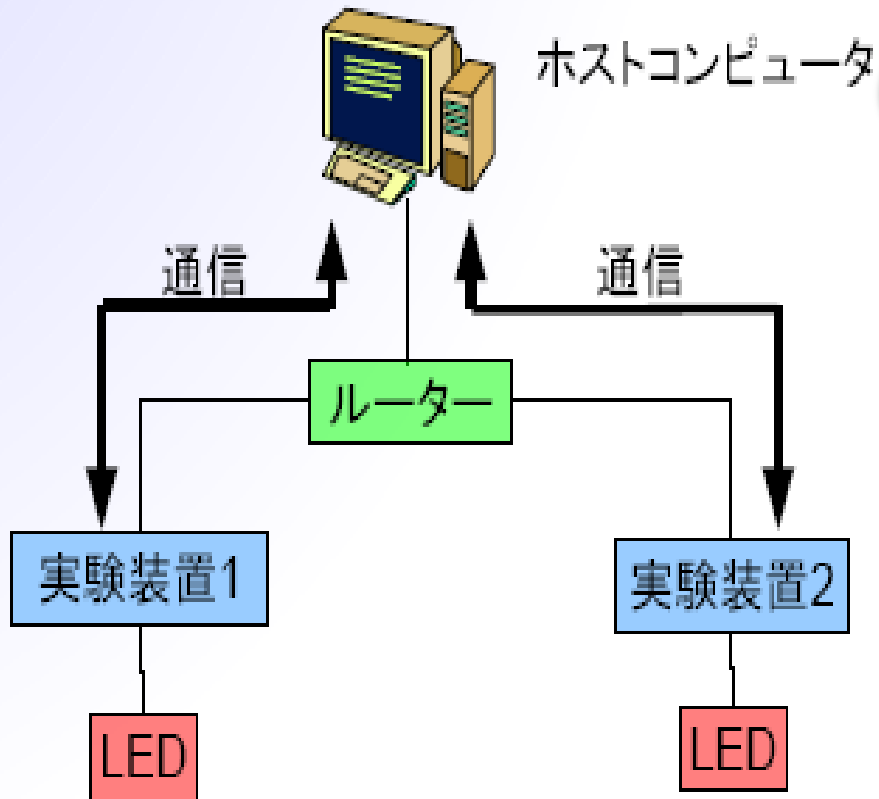


センサにより電圧の変化を感知し、感知したデータを、 $\mu$ IPを実装した装置によりTCP/IPで通信させシステムを構築する。



$\mu$ IPのソースコードのアプリケーション部を解析、変更し、センサのデータ取得スピードを上げることを目的としている。

# □ センサネットのメリット



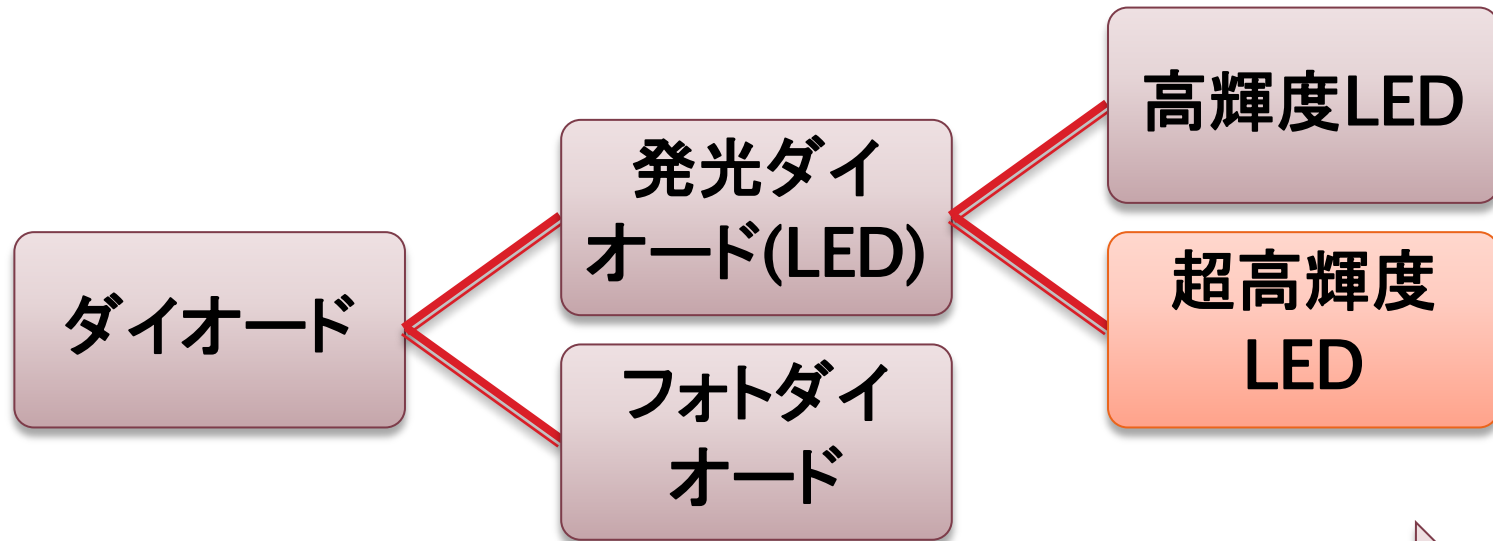
ネットワーク環境がある  
コンピュータならば、ど  
のコンピュータからでも  
視聴可能。

カメラが破壊・盗難され  
ても、情報はネットワー  
ク上に残る。



センサネットを導入する  
ことにより、防犯性が高  
まる。

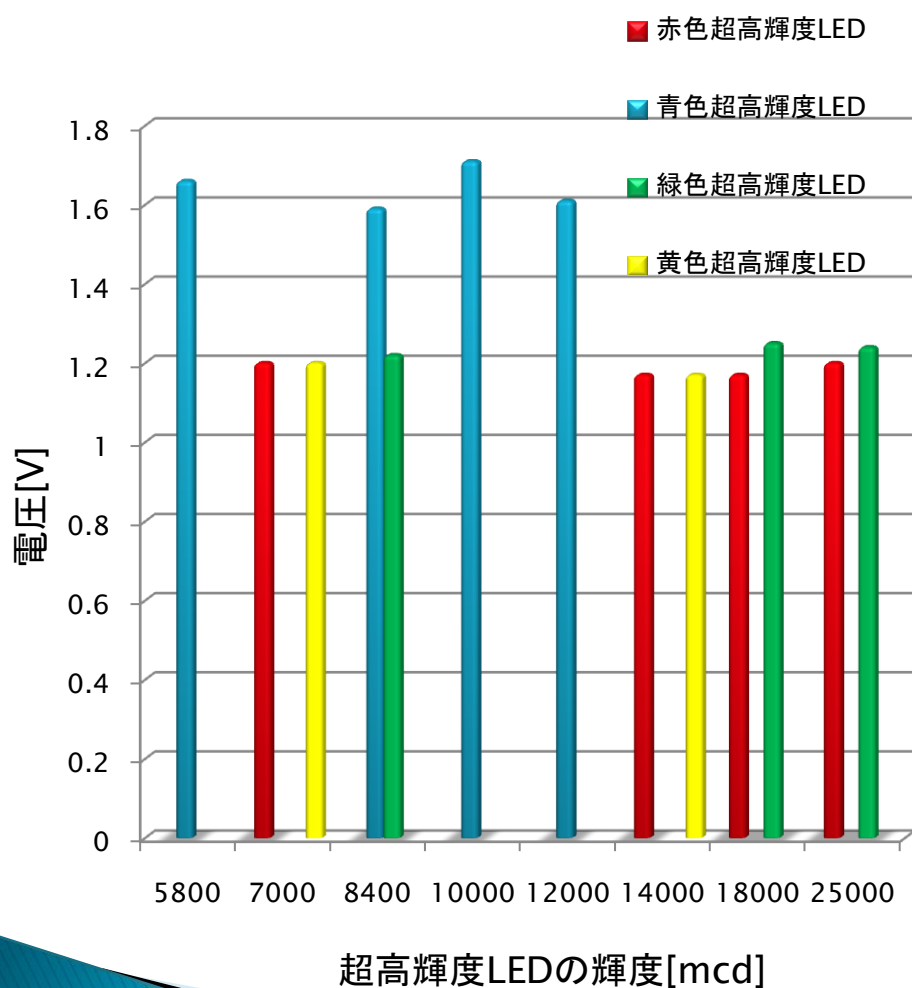
# □ 超高輝度LEDを用いる理由



超高輝度LEDと  
フォトダイオード  
を比較すると

- ・ 照明用のLEDがそのまま測定用として使える。
- ・ 超高輝度LEDの方が入力光に対する電圧の変化が大きいため、感度が高く、本研究に適している。

# □ 超高輝度LEDの選択



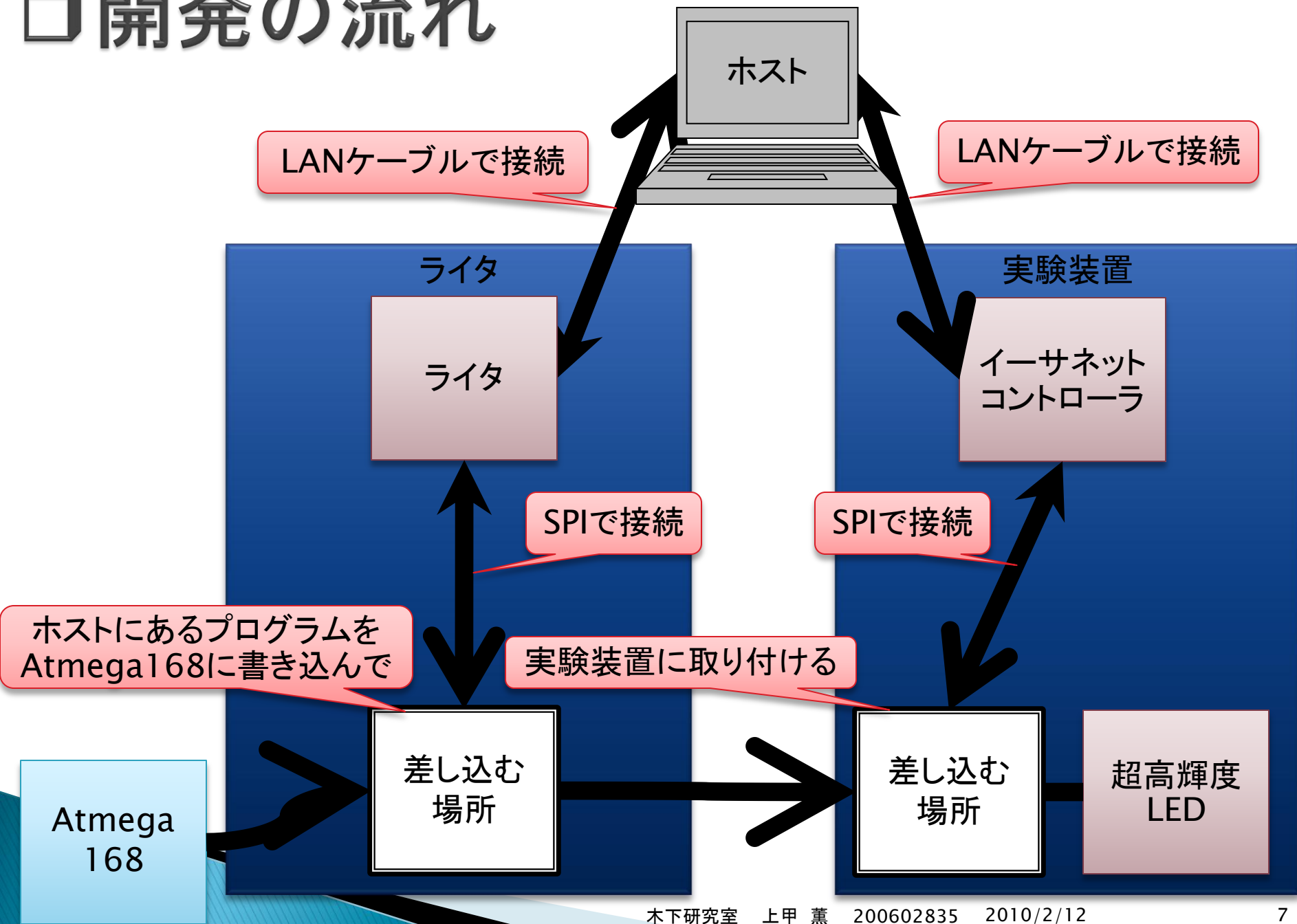
使用する超高輝度LEDを選択するため、本研究室にある、超高輝度LEDの評価を行った。

センサ部分の超高輝度LEDを、測定する度に変更し、それぞれの超高輝度LEDで生じる電圧を測定。



青色超高輝度LEDが最も高い電圧値を示した。取得できる電圧値の最大値は3Vなので、本研究では、青色超高輝度LEDを二つ使用することにした。

# □ 開発の流れ





# □内容



パソコンと装置  
をSPI(上の写真)  
で接続し、実行  
させる。

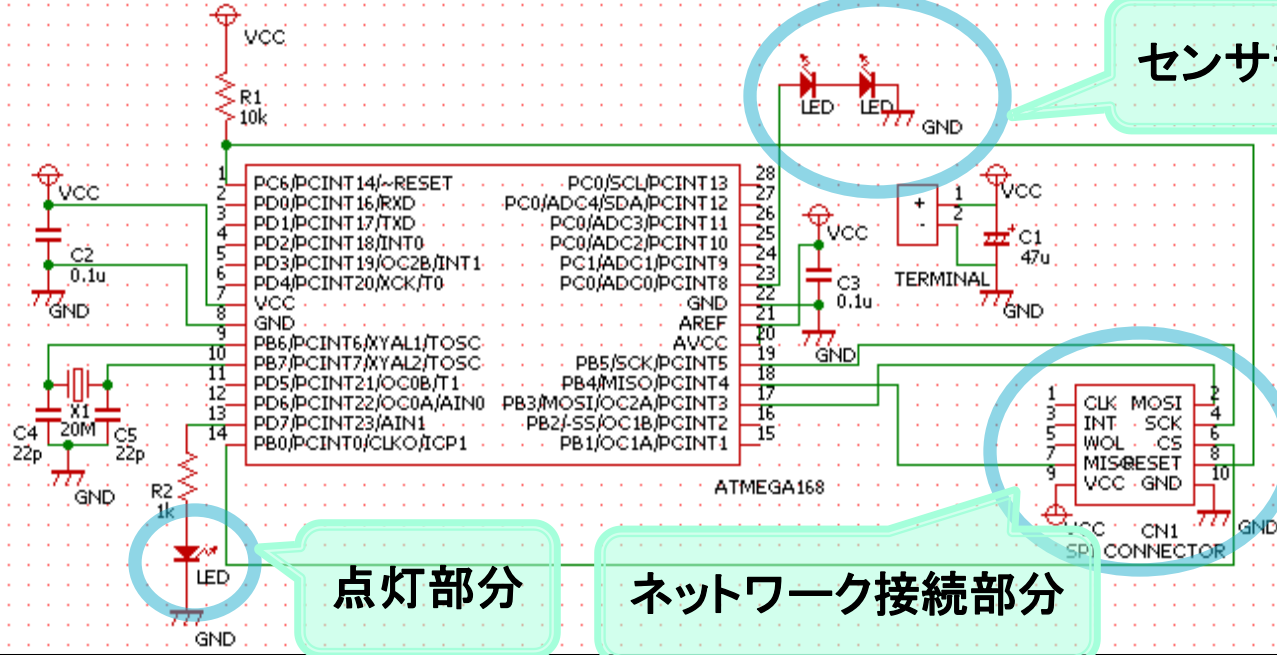
Atmega168(右  
の写真)にプログ  
ラムを書き込み、  
回路を作る。



光の遮断を行い、  
取得した電圧の  
値の変化を見る。

```
user@your-6faa821ee4 ~
$ telnet 192.168.1.12 46791
Trying 192.168.1.12...
Connected to 192.168.1.12.
Escape character is '^]'.
255
255
170
197
222
204
50
0
25
255
255
255
254
255
255
255
```





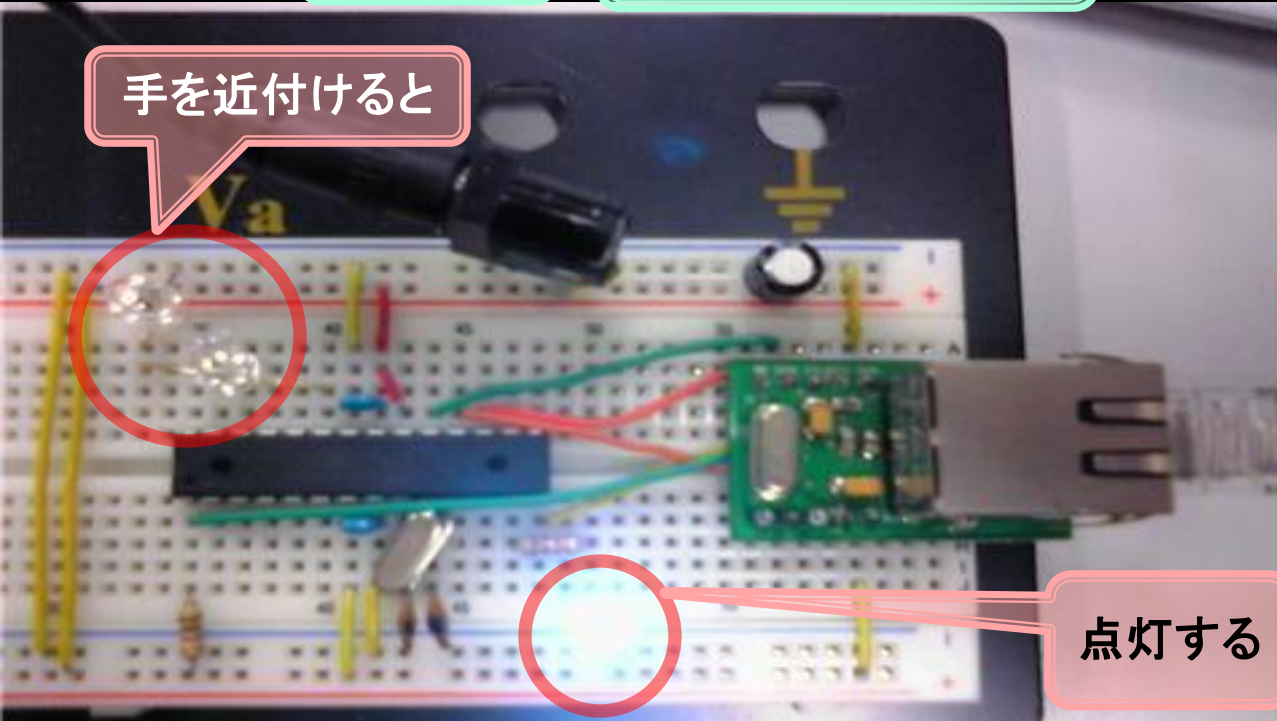
センサ部分

点灯部分

ネットワーク接続部分

```

user@your-6faa821ee4 ~
$ telnet 192.168.1.12 46791
Trying 192.168.1.12...
Connected to 192.168.1.12.
Escape character is '^]'.
255
255
170
197
222
204
50
0
25
255
255
255
254
255
255
255
  
```



手を近づけると

点灯する

光を遮断している部分  
1秒で表示される範囲

# □ポーリングによるバッファリング

```
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
```

- ・□の中が一度に表示され、一度の表示には0.2秒かかる。
- ・□の中の数字は、すべて同じデータ。

以上の通り、電圧の取得・送信は0.2秒に1回行われる。このスピードを上げるために、データのバッファリング処理を行った。

μIPのアプリケーション部を書き換え、データをバッファリングし、データがある程度溜まったら、まとめて送信する。



1つのデータを複数に増やして送信するという結果になってしまい、想定されていたシステムとはならなかった。

# □ 割り込み処理を行う理由

ポーリングによるバッファリングではうまくいかなかった理由として、ポーリング処理では、送信処理が間に合わないためと考えられる。

そこで割り込み処理を行うが、 $\mu$ IPではTCP/IPスタックの処理も割り込みにより行っているため、新しく電圧取得・送信割り込みを設定すると、多重割り込みとなる。

しかし、多重割り込みになると、お互いの処理のタイミングに問題が発生し、また、Atmega168の性能では、多重割り込みの信頼性を保証することができない。

ゆえに、電圧取得・送信処理の割り込み処理は、TCP/IPの処理と同期して行い、互いの処理に影響がないようなプログラミングを考える必要がある。



# □割り込みによるバッファリング

```
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
142 157 157 153 153 142 143 200 200 205 205 188 188 177 167 160 160 135 135 127
113 186 186 108 108 113 113 0 26 34 34 41 41 36 36 33 24 18 18 18
18 21 21 25 31 38 38 39 39 41 41 41 40 66 66 41 41 57 57 231
252 255 255 255 255 255 255 238 218 191 191 192 192 168 168 164 160 146 146 151
151 159 159 158 153 128 128 148 148 146 146 150 136 83 83 87 87 71 67 155
155 203 203 152 152 155 155 173 173 173 173 212 212 174 174 52 52 47
47 53 53 16 16 51 51 51 51 54 46 21 21 16 16 45 45 39 33
33 40 40 118 118 162 190 191 191 179 179 134 134 117 108 86 86 106 106
105 255 255 255 255 255 255 255 255 255 255 255 212 212 197 197 149 149 134 126 125
125 122 122 124 124 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
135 135 134 116 116 135 135 155 155 191 214 213 213 240 240 255 255 255 255 255
255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255 255
255 253 252 242 242 252 252 249 249 247 225 231 231 235 235 241 241 239 241 239
239 233 233 208 208 207 162 220 220 130 130 99 99 63 39 62 62 21 21 5
```

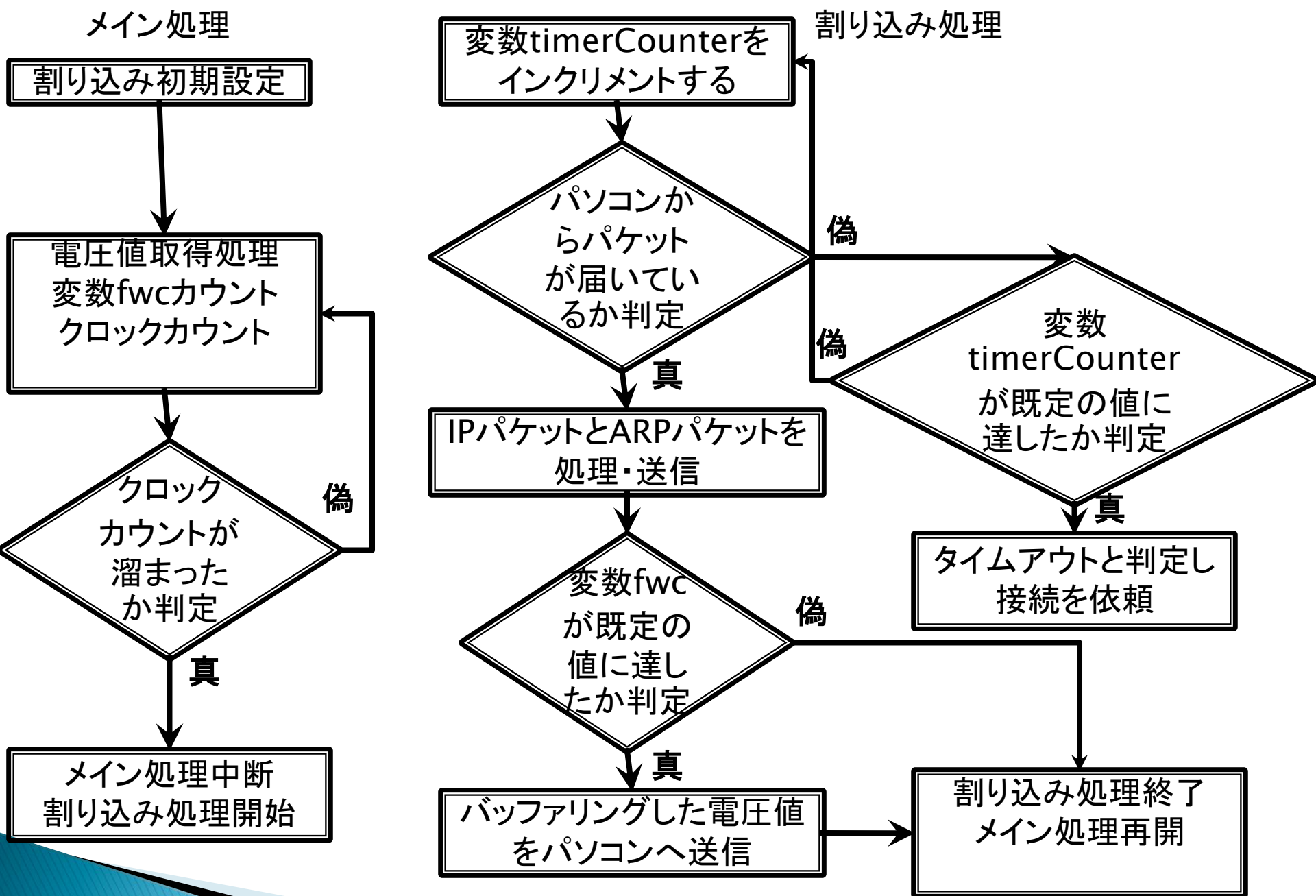
□の中が一度に表示され、一度の表示には4秒かかる。  
□の中の数字は、すべて違うデータ。

次に割り込み処理を用いて、データのバッファリングを行った。μIPのアプリケーション部だけでなく、TCP/IP処理部分も書き換えた。

電圧値の取得をメイン処理で行い、データの送信は割り込み処理の中で、ポーリング処理によりバッファを監視し、送信する。



実験を行った結果、20個まとめて送信させる場合、4秒待たないと表示されなかったもので、0.2秒で1個送信している場合と変化がなかった。



# □ 結論

本研究では、センサネットのデータ取得効率向上のため、 $\mu$ IPを解析し、割り込み処理によるバッファリングを行った。

しかし $\mu$ IPでは、TCP/IP処理を割り込み処理により実現しているため、電圧の取得・送信の優先度が下がってしまう。

今後はより $\mu$ IPの解析を進め、各処理のタイミングを操作し、同期させていく必要がある。