明治大学　理工学部

2024年度

卒業論文

カレントミラーを組み合わせた折り返し型アナログ乗算回路の出力範囲を拡大する回路構成

学生番号：1512201217

電気電子生命学科

小島　光

指導教員：和田　和千

実家住所：〒124-0006　東京都葛飾区堀切8-8-3-306

電話番号：070-3614-1189

概要

フォトニックリザバの学習・計算の際に必要となるアナログ的な積和演算を可能とする乗算回路の構成を提案する。KCL を利用することで、多数の信号の和を電流的に実現することができるが、信号線を共有するため和の出力振幅は単体の乗算回路と共有される。即ち、積和演算回路の出力振幅は乗算回路単体の出力振幅と等しくなり、和をとる信号の数が多くなればなるほど乗算回路一つあたりの入力範囲が限られてしまうため信号対雑音比 (S/N 比) の劣化が懸念される。そこで、従来のギルバート乗算回路を応用した乗算回路を提案しギルバート乗算回路同様、アナログ信号の乗算ができることを、小信号解析を用いて確認した。また、乗算器が動作する条件として、MOSFET が遮断しない条件を使い出力範囲を示した。そして、今後集積化を行い出力範囲拡大ができていることを確認するための素子値を設計し、パッケージでの測定を踏まえたシミュレーションを行った。

目 次

第1章　序論

現在、機械学習の分野ではディープラーニングが盛んに研究され、とくに時系列データを扱うためには各ノードが次の層への結合のみを持つ順伝搬型 (feed-forword) ではなく以前の層への結合も持つリカレントニューラルネット (RNN) と呼ばれる方式が使われている。しかし、一般なディープラーニングにおいて、現実的な精度を得るためには多くの隠れ層を必要とするがその学習は必ずしも現実的な計算量では終わらない。これは多数の隠れ層を持つニューラルネットワーク (NN) では、ノード間の結合が非線形な活性化関数で定義されるため、教師データに対する数値解析的なフィッティングしかできないことに起因する。そこで、学習効率を上げる方法として RNN を物理現象によって再現したリザバが登場した。

このリザバからの複数の出力を積和演算することで特徴量を抽出することができる。特に、学習するのはリザバではなく積和演算の重みのみであるため学習コストを低減させることができる。今回、光で入出力を行うフォトニックリザバにおいては高速な画像認識に用いることができる可能性が報告されている。しかしながら、現状では出力の光を光のまま積和演算することは難しいためフォトダイオードによって電気に変換し積和演算・並びに学習を行うことが検討されている。ここで、積和演算が画像認識速度のボトルネックにならないよう高速な処理が必要なため今回はアナログ信号の積和演算を行うギルバート乗算回路を使用することが本学で検討されていた。ギルバート乗算回路を用いる積和演算では図**??**に示すよう、出力信号に対してギルバート乗算回路で重みづけを行い、出力信号をまとめることで電流的に和をとる。即ち、各乗算器の出力振幅が積和演算回路の出力振幅の上限となるため、積和演算を行う信号の数が増えるとその分入力範囲が限られる。これにより信号対雑音比 (S/N 比)

の劣化が懸念される。

本研究では、S/N 比向上のため乗算回路単体の出力振幅を拡大する構成を提案する。本論の構成は以下のとおりである。まず 2 章で検討していたギルバート乗算回路構成を示し、小信号解析を行うことでその特性を確認する。さらにその現実的な出力範囲を導出する。3 章において提案する回路構成についての基本的な方針を示し、小信号解析において提案回路がギルバート乗算回路同様、アナログ信号の乗算が可能であることを明らかにする。また、その出力範囲を導出しギルバート乗算回路に対し出力範囲を広げられることを示し、シミュレーション上でも確認する。最後に、4 章で結論並びに今後の展望について述べる。

謝辞

本研究を遂行するにあたり，大変手厚く御指導頂いた本学電気電子生命学科和田和千准教授に深く感謝する．併せて，本論文の執筆にあたり有益なる御助言を頂いた同学科通信伝送グループの井家上哲史教授，関根かをり教授，中村守里也准教授に深く感謝する．また，日頃の研究において，議論を通じて多くの御助言を頂いた波動信号処理回路研究室諸氏に厚く感謝する．

2024 年 2 月日
明治大学　理工学部　電気電子生命学科　4 年
波動信号処理回路研究室
小島　光

付 録A　実装回路図ならびにPCBレイアウト図

実装した送受信回路の回路図ならびにPCBレイアウト図を次頁以降に添付する.

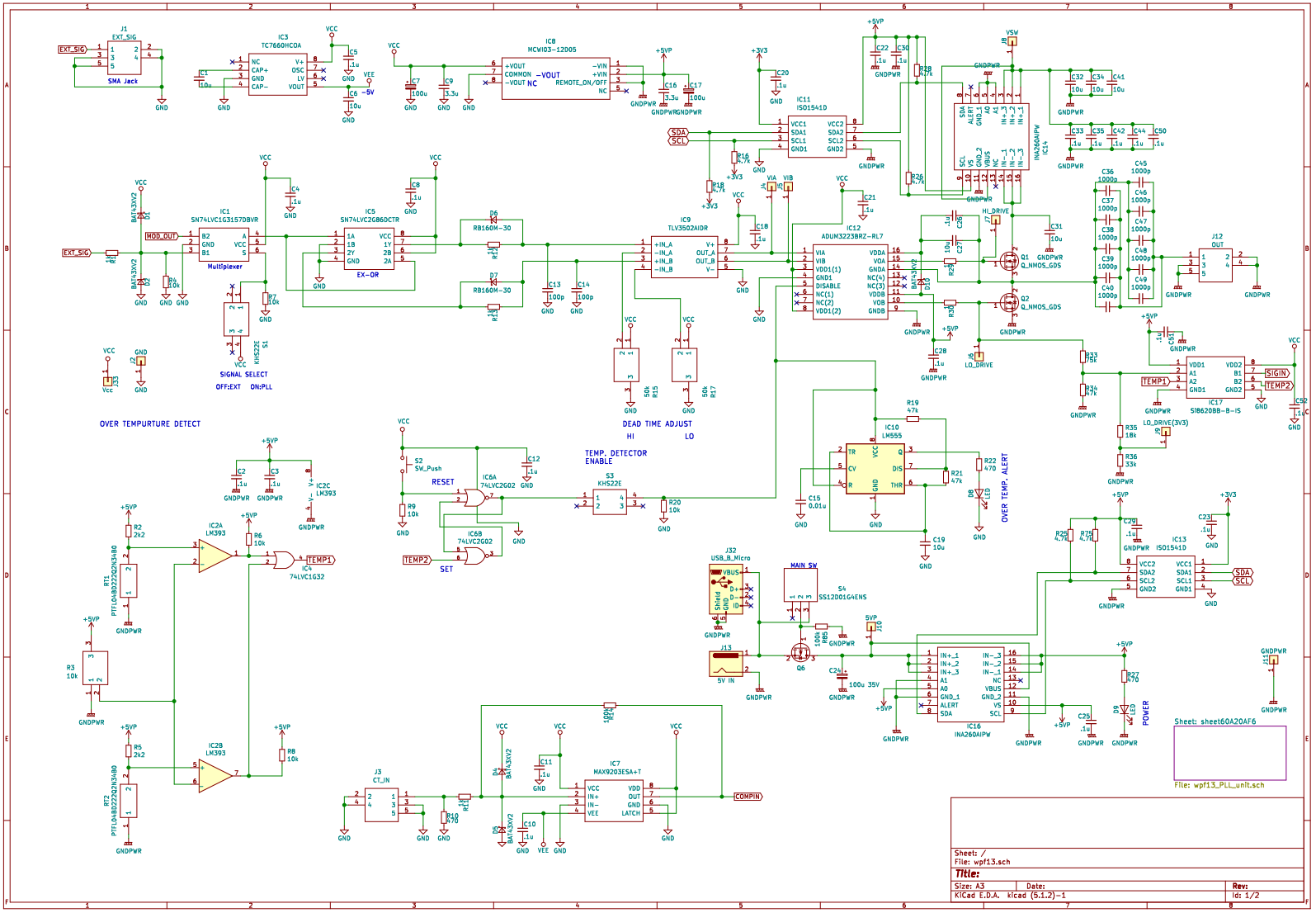


図 A.1: 送信側回路図 (1/2)

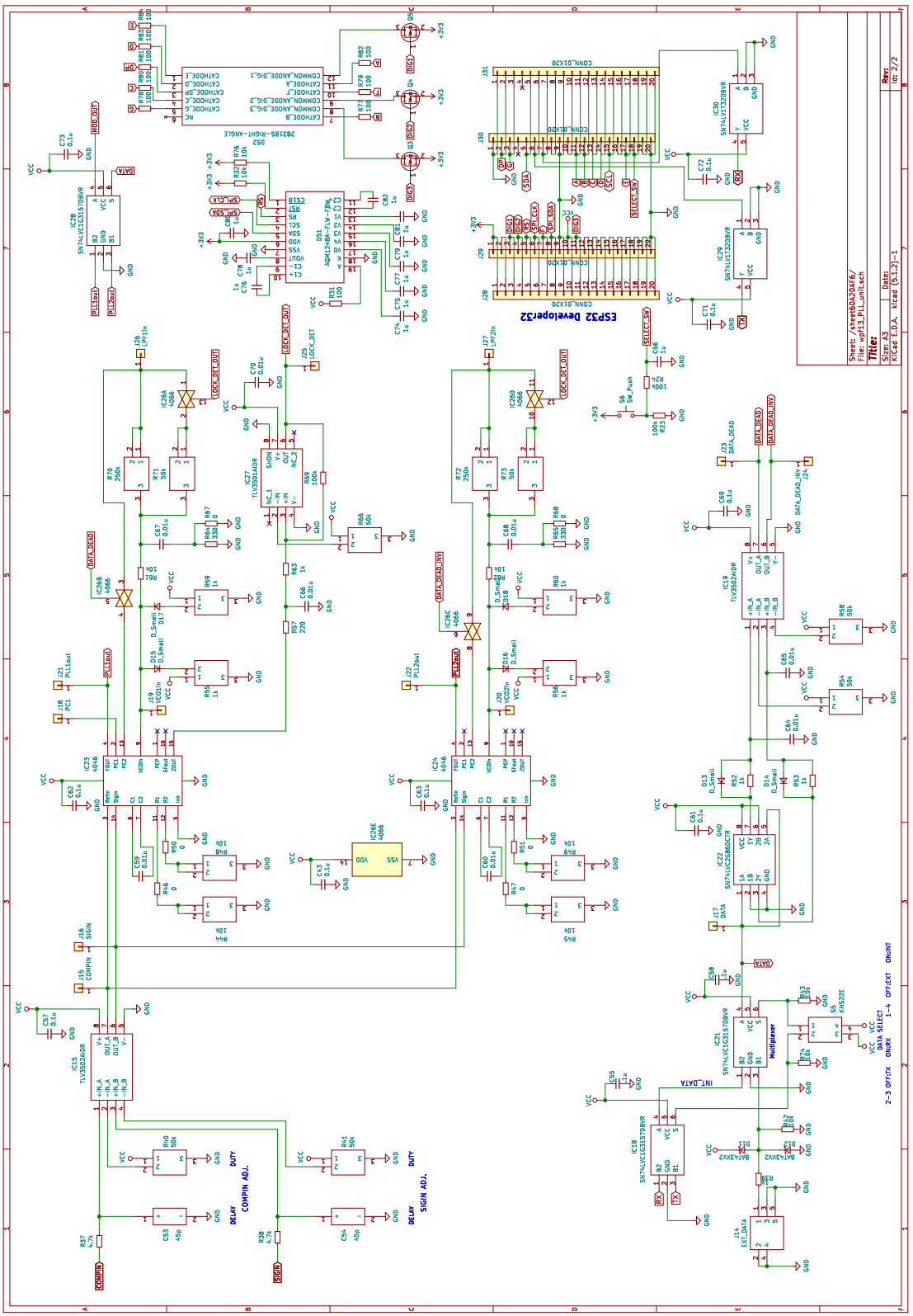


図 A.2: 送信側回路図 (2/2)

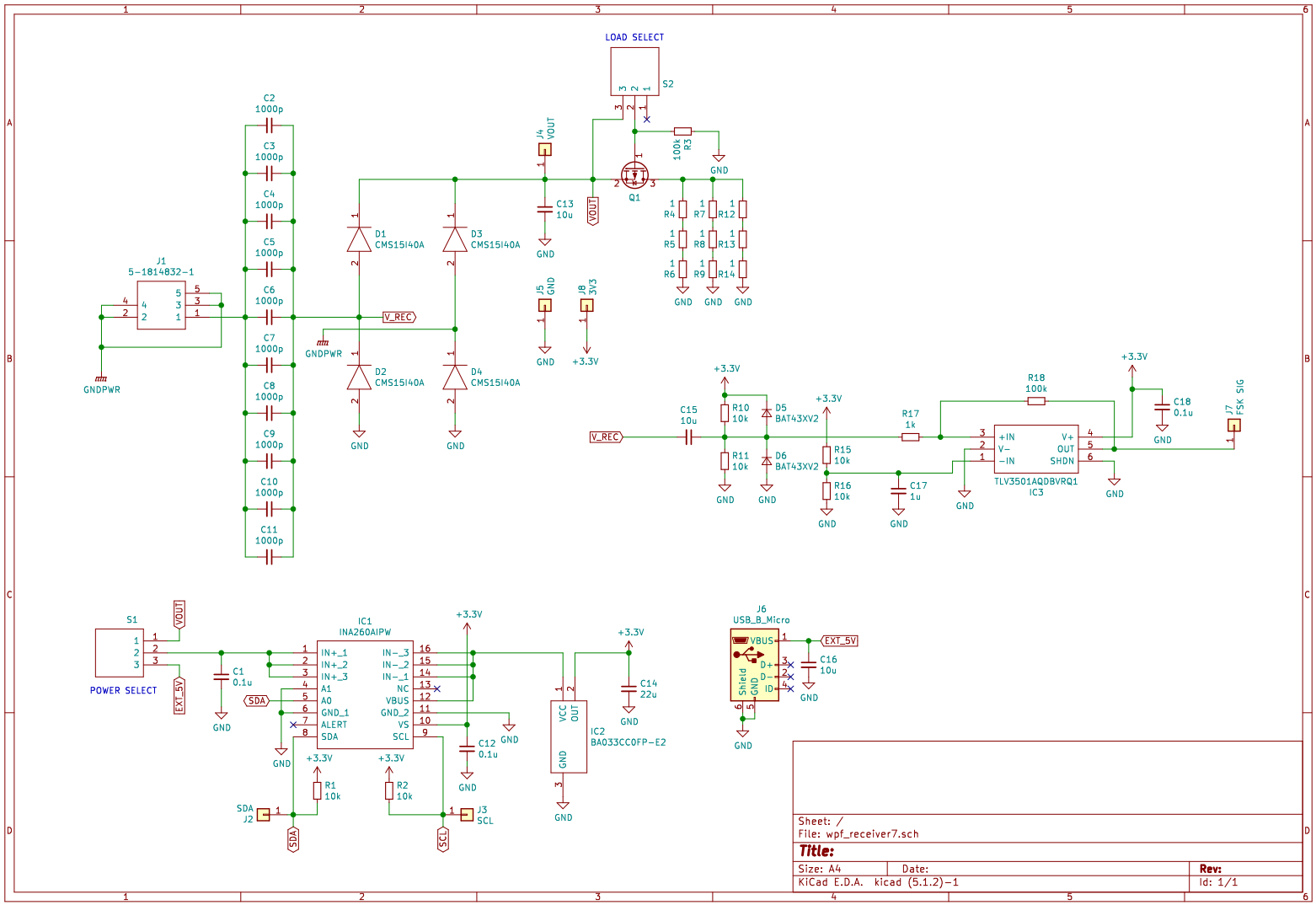
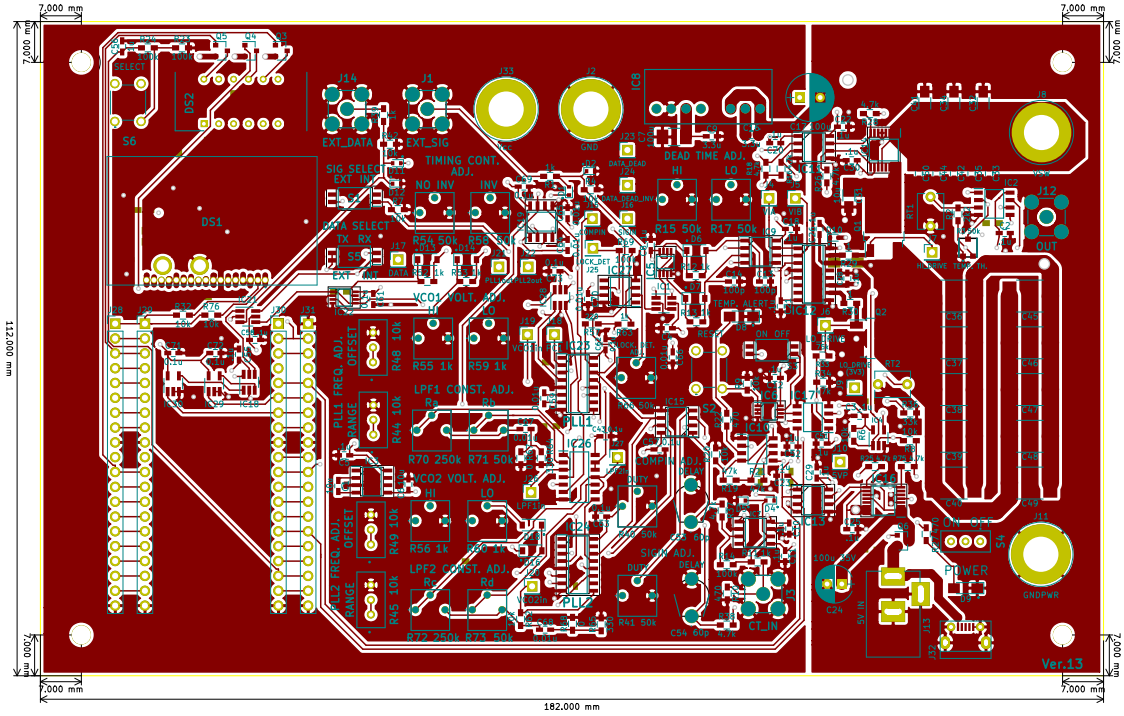
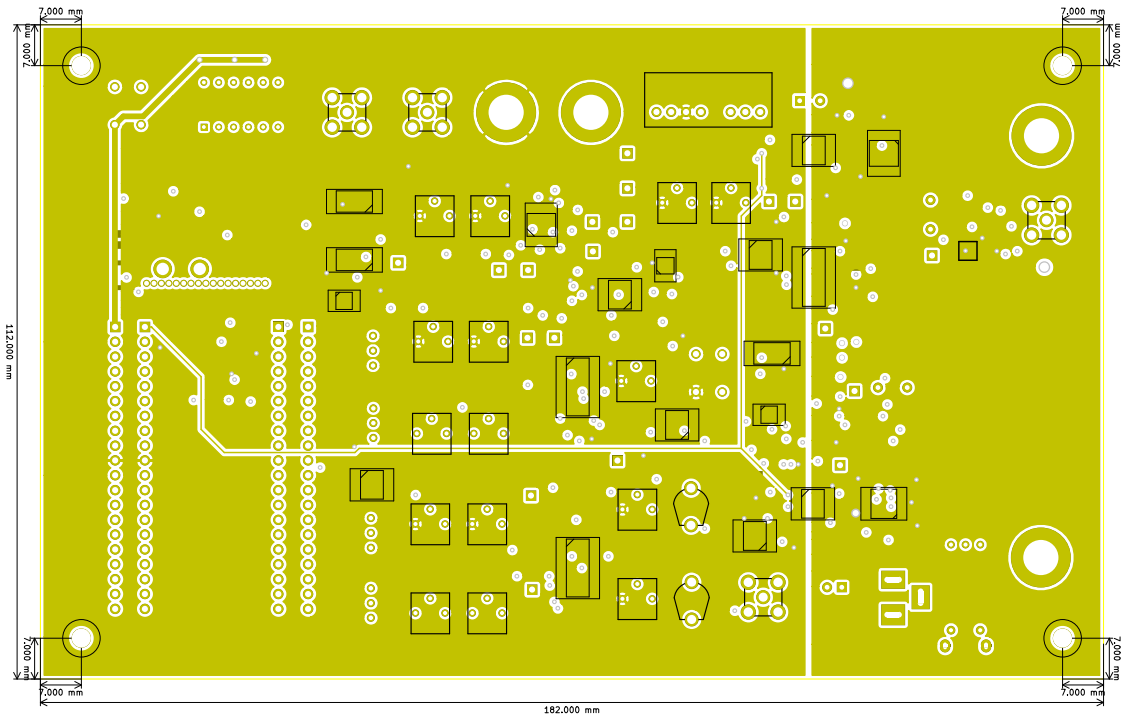

LOAD SELECT
S2
J4
VOUT
Q1
R3
100k
GND
C2
1000p
C3
1000p
C4
1000p
C5
1000p
C6
1000p
C7
1000p
C8
1000p
C9
1000p
C10
1000p
C11
1000p
J1
5-1814832-1
GNDPWR
D1
CMS15I40A
D2
CMS15I40A
D3
CMS15I40A
D4
CMS15I40A
V_REC
C13
10u
J5
GND
J8
3V3
+3.3V
R4
R5
R6
R7
R8
R9
R12
R13
R14
C15
10u
R10
10k
R11
10k
D5
BAT43XV2
D6
BAT43XV2
R15
10k
R16
10k
R17
1k
R18
100k
C17
1u
C18
0.1u
J7
FSK SIG
TLV3501AQDBVRQ1
IC3
+IN
V-
-IN
V+
OUT
SHDN
S1
POWER SELECT
EXT_5V
C1
0.1u
IC1
INA260AIPW
SDA
A1
A0
GND_1
ALERT
IN+_1
IN+_2
IN+_3
IN-_3
IN-_2
IN-_1
NC
VBUS
GND_2
VS
SCL
C12
0.1u
R1
10k
R2
10k
J2
J3
IC2
BA033CC0FP-E2
VCC
OUT
C14
22u
J6
USB_B_Micro
VBUS
D+
D-
ID
Shield
C16
10u
Sheet: /
File: wpf_receiver7.sch
Title:
Size: A4
Date:
Rev:
KiCad E.D.A. kicad (5.1.2)-1
Id: 1/1

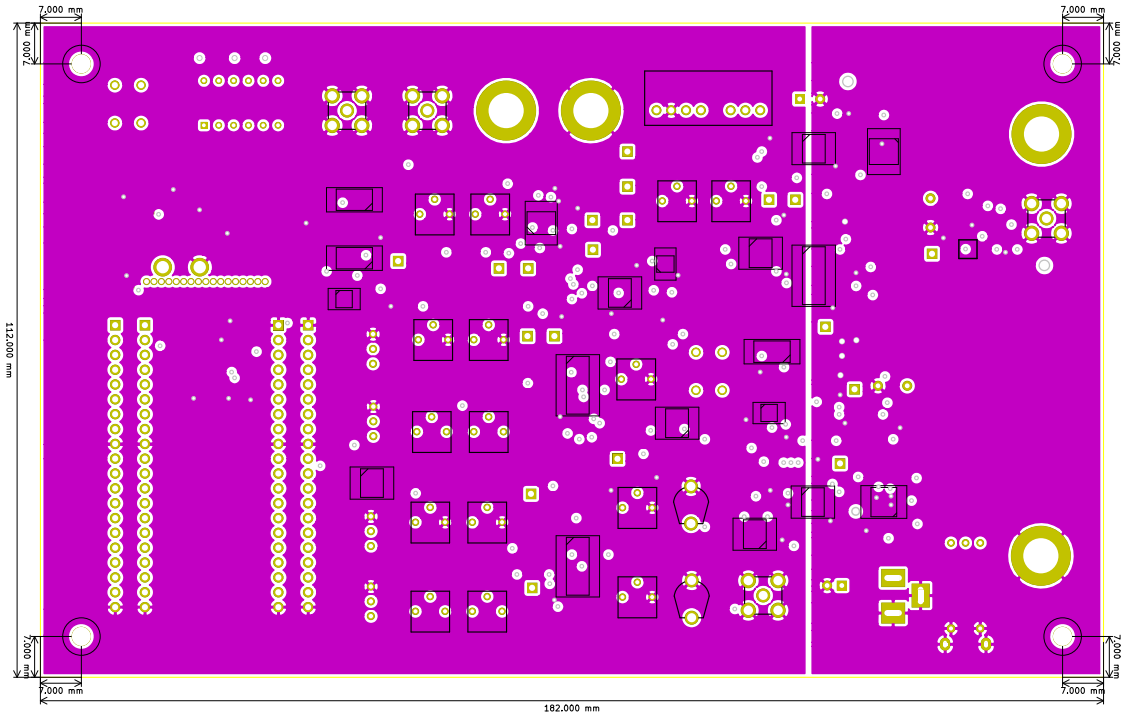
図 A.3: 受信側回路図


7.000 mm
112.000 mm
182.000 mm
J14
J1
J33
J2
EXT_DATA
EXT_SIG
GND
TIMING CONT. ADJ.
DEAD TIME ADJ.
HI
LO
SIG SELECT
EXT IN
NO_INV
INV
DATA SELECT
TX_RX
EXT
VCO1 VOLT. ADJ.
R55 1k
R59 1k
R48 10k
R44 10k
LPF1 CONST. ADJ.
R70 250k
R71 50k
VCO2 VOLT. ADJ.
R56 1k
R49 10k
R45 10k
LPF2 CONST. ADJ.
R72 250k
R73 50k
R41 50k
R15 50k
R17 50k
J12
OUT
J11
POWER
GNDPWR
5V
C24
5V_IN
CT_IN
Ver13

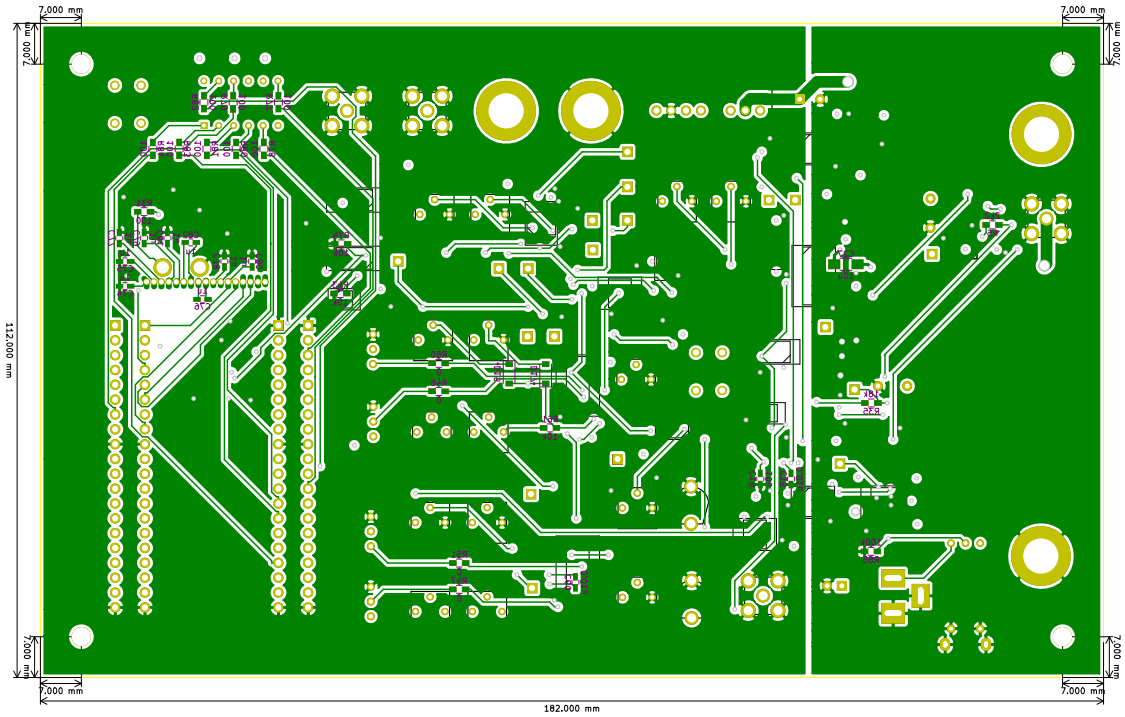
(a) 表面


7.000 mm
112.000 mm
182.000 mm

(b) 内層 1 面

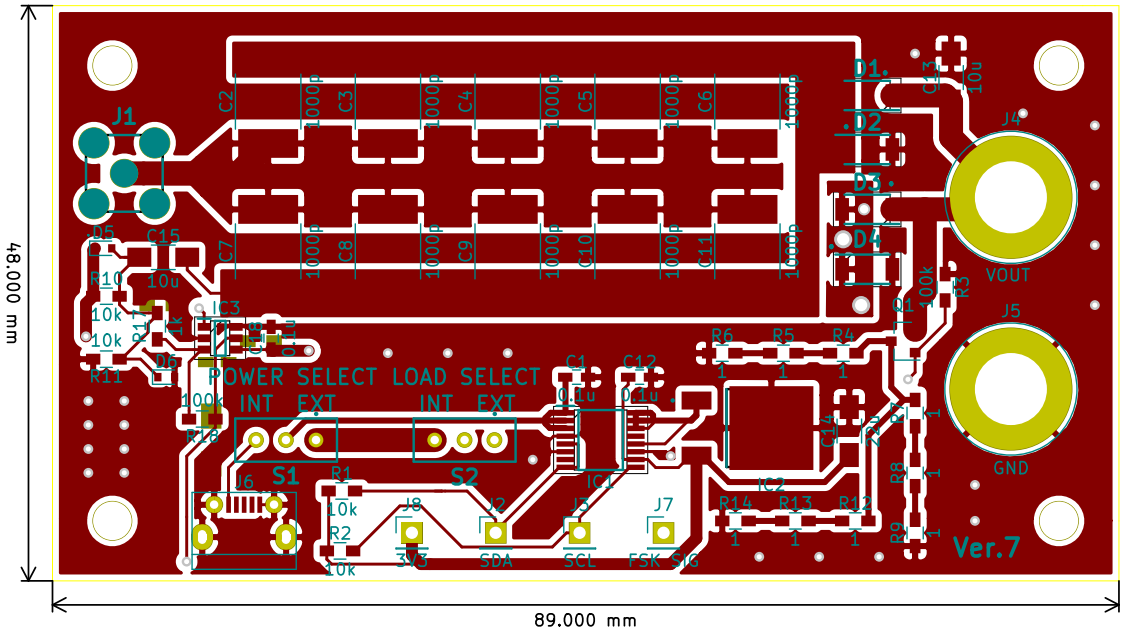


(c) 内層 2 面

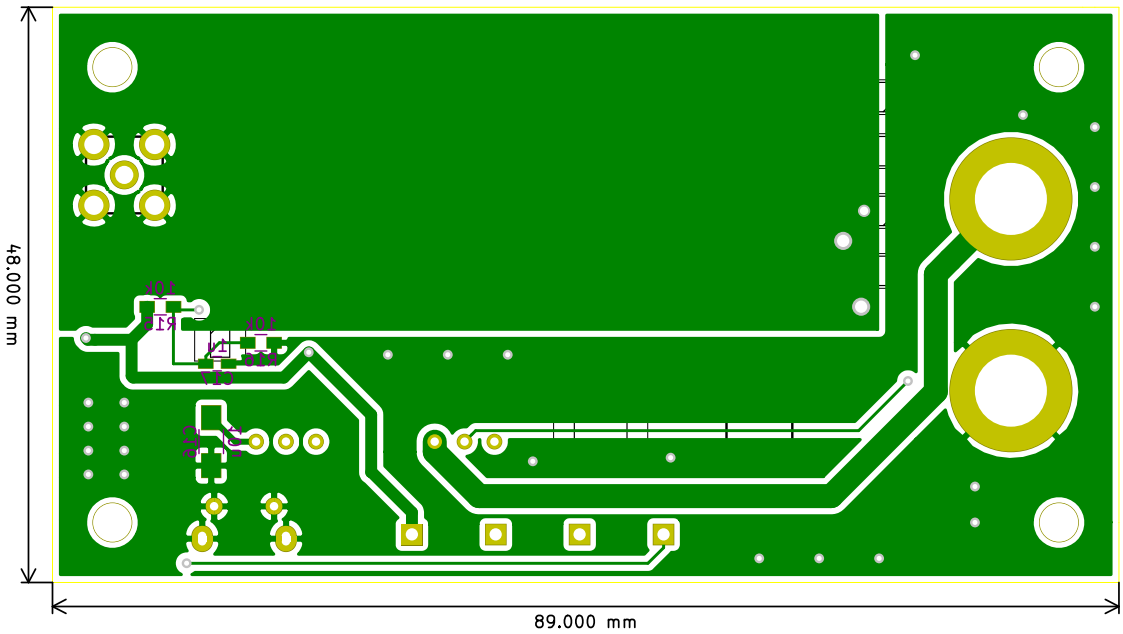


(d) 裏面

図 A.4: 送信側 PCB レイアウト図



(a) 表面



(b) 裏面

図 A.5: 受信側 PCB レイアウト図

付 録B　FSK復調器のVerilog-HDLソースコード

FPGA を用いたFSK 復調器の Verilog-HDL ソースコードならびに論理シミュレーション用テストベンチを以下に示す.

Listing B.1: FSK 復調器

```
1
2  module counter7_for_thesis(
3      input clk,
4      input serial_datain,
5      input sw0,
6      input sw1,
7      input sw2,
8      input sw3,
9      input btn0,
10     input extfsksig, //b10
11
12     output serial_dataout,
13     output led0,
14     output b1,
15     output b2,
16     output b3,
17     output b4,
18     output b7,
19     output b8,
20     output b9
21         );
22
23
24 parameter low_freq=598; /*で指定 kHz*/
```

```
25 parameter high_freq=979;
26 parameter integer cnt_fspace=(1/(low_freq*2*0.000008));
27 parameter integer cnt_fmark=(1/(high_freq*2*0.000008));
28 parameter integer threshold=167;
29
30 reg [19:0] cntdata=20'd0;
31 reg [31:0] cntfspace=32'd0;
32 reg [31:0] cntfmark=32'd0;
33 reg [31:0] demodcnt_1p=32'd0;
34 reg [31:0] demodcnt_2p=32'd0;
35 reg [31:0] demodcnt_p=32'd0;
36 reg [31:0] difference_p=32'd0;
37 reg [31:0] bps=32'd0;
38 reg [31:0] rstpls_p=32'b0;
39 reg [63:0] resetcounter=64'd0;
40
41 reg fmark=1'b0; /* high freq*/
42 reg fspace=1'b0; /*low freq*/
43 reg fsksig=1'b0;
44 reg fdata=1'b0;
45 reg datachange=1'b0;
46 reg demodout=1'b0;
47 reg cntrst_p=1'b0;
48 reg resetflag=1'b0;
49 reg sigselect=1'b0;
50 reg recoverflag=1'b0;
51
52 initial @(negedge demodout) recoverflag<=1'b1;
53
54 always @(posedge clk)begin
55
56
57 /* set rate of pseudo data */
58 ////////////////////////////////////////////////////
59
60          if(sw3==0 & sw2==0 & sw1==0& sw0==0) bps<='d300;
61     else if(sw3==0 & sw2==0 & sw1==0& sw0==1) bps<='d600;
62     else if(sw3==0 & sw2==0 & sw1==1& sw0==0) bps<='d1200;
63     else if(sw3==0 & sw2==0 & sw1==1& sw0==1) bps<='d2400;
64     else if(sw3==0 & sw2==1 & sw1==0& sw0==0) bps<='d4800;
65     else if(sw3==0 & sw2==1 & sw1==0& sw0==1) bps<='d9600;
66     else if(sw3==0 & sw2==1 & sw1==1& sw0==0) bps<='d14400;
```

```
67      else if(sw3==0 & sw2==1 & sw1==1& sw0==1) bps<='d19200;
68      else if(sw3==1 & sw2==0 & sw1==0& sw0==0) bps<='d38400;
69      else if(sw3==1 & sw2==0 & sw1==0& sw0==1) bps<='d57600;
70      else if(sw3==1 & sw2==0 & sw1==1& sw0==0) bps<='d115200;
71      else if(sw3==1 & sw2==0 & sw1==1& sw0==1) bps<='d230400;
72      else if(sw3==1 & sw2==1 & sw1==0& sw0==0) bps<='d460800;
73      else if(sw3==1 & sw2==1 & sw1==0& sw0==1) bps<='d921600;
74      else bps<='d115200;
75
76  ////////////////////////////////////////////////////
77
78
79  /*genarate pseudo FSK signal*/
80  ////////////////////////////////////////////////////
81      if(cntfspace == cnt_fspace) begin
82          cntfspace<=32'd0; // reset cnt //
83          fspace<=~fspace;
84   end
85
86  else begin
87      cntfspace<=cntfspace+32'd1;
88   end
89
90  ////////////////////////////////////////////
91
92       if(cntfmark == cnt_fmark) begin
93          cntfmark<=32'd0; // reset cnt //
94          fmark<=~fmark;
95   end
96
97  else begin
98      cntfmark<=cntfmark+32'd1;
99   end
100
101 ////////////////////////////////////////////
102
103       if(cntdata == 'd125000000/bps) begin
104         cntdata<=20'd0; // reset cnt //
105        fdata<=~fdata;
106  end
107
108 else begin
```

```
109     cntdata<=cntdata+20'd1;
110  end
111
112 ////////////////////////////////////////////////////
113
114
115 /*choose internal(pseudo) or external FSK signal*/
116 ////////////////////////////////////////////////////
117
118  if(btn0==1) sigselect<=~sigselect;
119
120  if(sigselect==0) begin
121
122     if(fdata==0) begin
123     fsksig<=fspace;
124     end
125      else begin
126     fsksig<=fmark;
127     end
128  end
129
130  if(sigselect==1) fsksig<=extfsksig;
131
132 ////////////////////////////////////////////////////
133
134
135 /*generate counter-reset pulse*/
136 ////////////////////////////////////////////////////
137
138 if(fsksig==1'b1) begin
139     rstpls_p<=rstpls_p+1'b1;
140         if(rstpls_p<1'b1) cntrst_p<=1'b1;
141         else if (rstpls_p==1'b1) begin
142         rstpls_p<=1'b1;
143         cntrst_p<=1'b0;
144         end
145
146 end
147
148 if(fsksig==1'b0) rstpls_p<=1'b0;
149 ////////////////////////////////////////////////////
150
```

```
151
152 /*set counter*/
153 ////////////////////////////////////////////////////
154
155   if(fsksig==1'b0 && cntrst_p==1'b0) demodcnt_2p<=demodcnt_2p
        +32'd1;
156  else if (fsksig==1'b1 && cntrst_p==1'b0) demodcnt_1p<=
       demodcnt_1p+32'd1;
157  else begin
158     demodcnt_1p<=32'd0;
159     demodcnt_2p<=32'd0;
160  end
161
162  demodcnt_p<=demodcnt_1p+demodcnt_2p;
163
164 ////////////////////////////////////////////////////
165
166
167 /*set normally HIGH*/
168 ////////////////////////////////////////////////////
169 if(cntrst_p==1'b1) begin
170     if(demodcnt_1p>demodcnt_2p) difference_p<=demodcnt_1p-
          demodcnt_2p;
171     if(demodcnt_2p>demodcnt_1p) difference_p<=demodcnt_2p-
          demodcnt_1p;
172     if(demodcnt_1p==demodcnt_2p) difference_p<=0;
173 end
174
175 if(difference_p>'d10) datachange<=1'b1;
176 else datachange<=1'b0;
177
178 ////////////////////////////////////////////
179
180 if(datachange==0)begin // set normally high
181     resetcounter<=resetcounter+32'd1;
182
183     if(resetcounter>'d125000000) resetflag<=1'b1; else
          resetflag<=1'b0;
184 end
185
186 if(datachange==1) resetcounter<=32'd0;
187
```

```
188 ////////////////////////////////////////////////////
189
190
191 /* data detection */
192 ////////////////////////////////////////////////////
193
194 if(cntrst_p==1'b1) begin
195         if(demodcnt_p>threshold) demodout<=1'b0;
196         if(demodcnt_p<threshold) demodout<=1'b1;
197     end
198
199    else demodout<=demodout;
200
201 ////////////////////////////////////////////////////
202  end
203
204 assign b1=fdata;
205 assign b2=fmark;
206 assign b3=fspace;
207 assign b4=fsksig;
208 assign b7=demodout;
209 assign b8=cntrst_p;
210 assign b9=datachange;
211 assign led0=sigselect;
212 assign serial_dataout=demodout;
213 endmodule
```

Listing B.2: テストベンチ

```
1
2 `timescale 1ns / 1ps
3
4 module counter7_tb;
5
6     reg clk;
7     reg rnd;
8
9     wire sw0=0;
10    wire sw1=0;
11    wire sw2=0;
12    wire sw3=1;
13    wire btn0=0;
14    wire extfsksig=0;
```

```
15
16      wire serial_dataout;
17      wire led0;
18      wire b1;
19      wire b2;
20      wire b3;
21      wire b4;
22      wire b7;
23      wire b8;
24      wire b9;
25
26      initial clk<=0;
27
28      always #4 clk=~clk;
29
30      counter7_for_thesis uut(
31      .clk(clk),
32      .serial_datain(rnd),
33
34      .sw0(sw0),
35      .sw1(sw1),
36      .sw2(sw2),
37      .sw3(sw3),
38      .btn0(btn0),
39      .extfsksig(extfsksig),
40
41      .serial_dataout(serial_dataout),
42      .led0(led0),
43      .b1(b1),
44      .b2(b2),
45      .b3(b3),
46      .b4(b4),
47      .b7(b7),
48      .b8(b8),
49      .b9(b9));
50  endmodule
```