

Verilogと戯れる

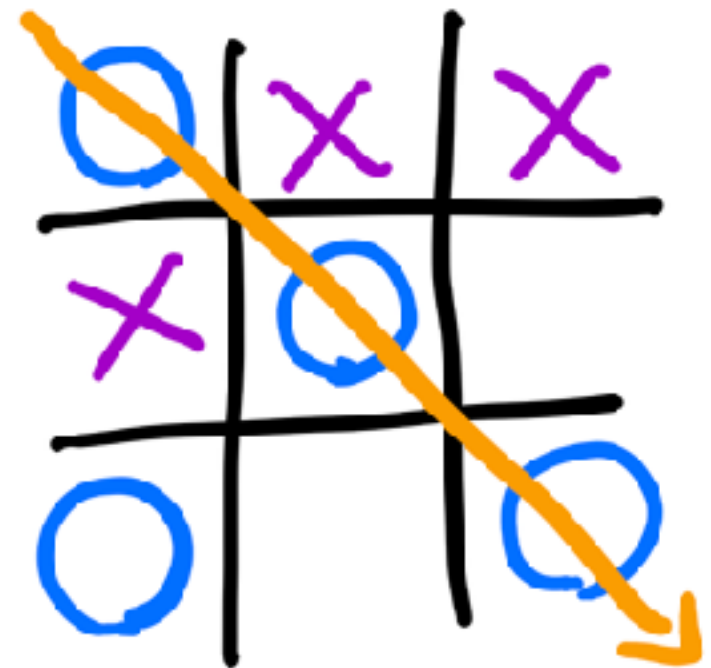
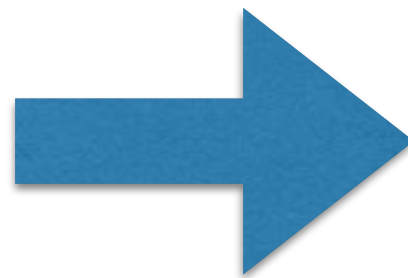
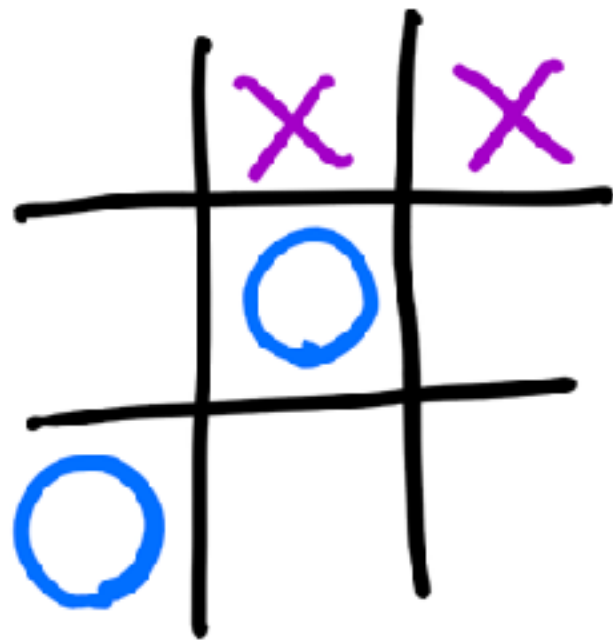
00-640726 桂 宏行

つくったもの

- ・ Tic Tac Toe
- ・ （それに付随してVerilogを補助するパーサ）

Tic Tac Toe

- ・ 3x3のマス目に交互に丸ばつを入れていくゲーム
- ・ 縦横斜めどれかに一列並べられた方が勝ち

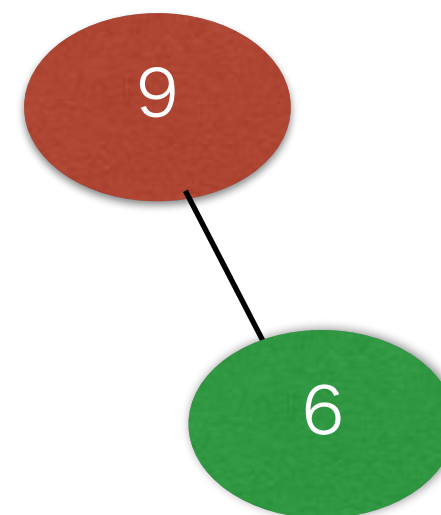
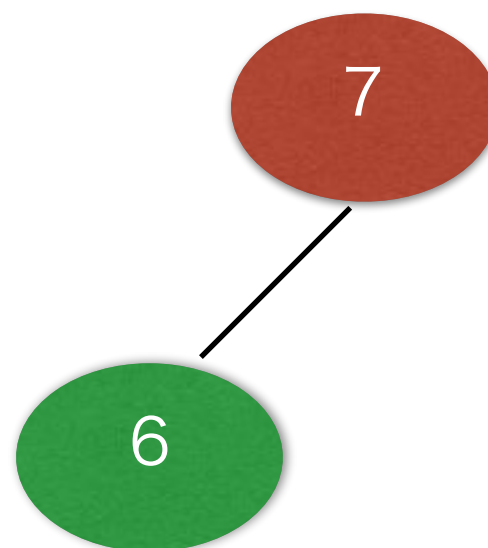
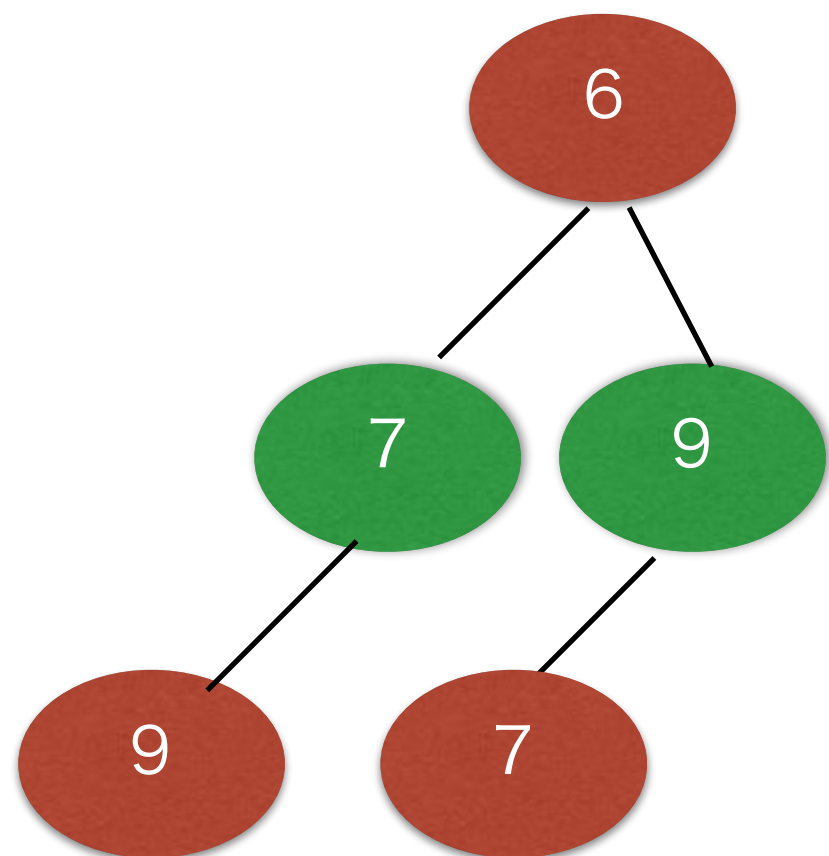


実装した内容

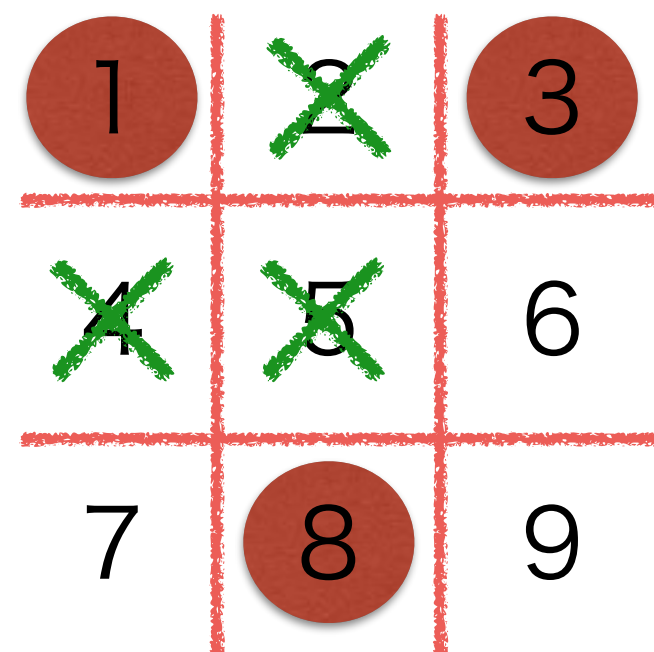
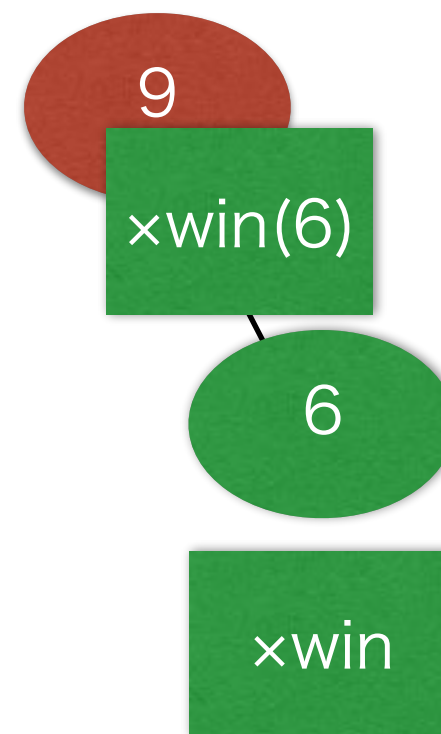
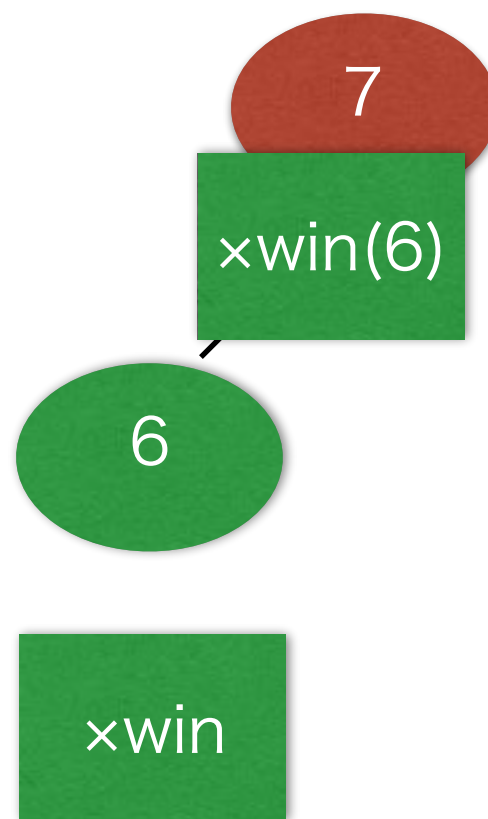
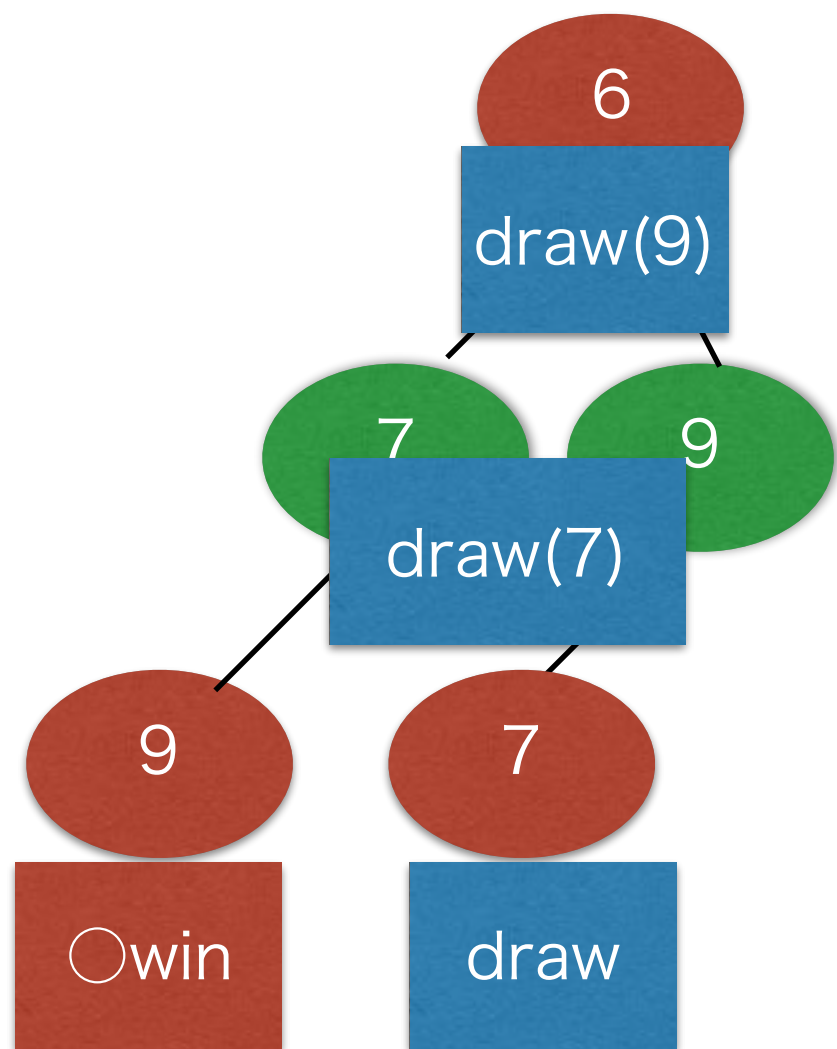
- ・ ゲーム実装
 - ・ 盤面表示
 - ・ 勝敗判定
- ・ 対戦CPU
 - ・ 乱択（っぽいAI）
 - ・ 最善手を打ってくるAI
- ・ その他ゲームらしい機能
 - ・ ゲームクリア～って出たりとか

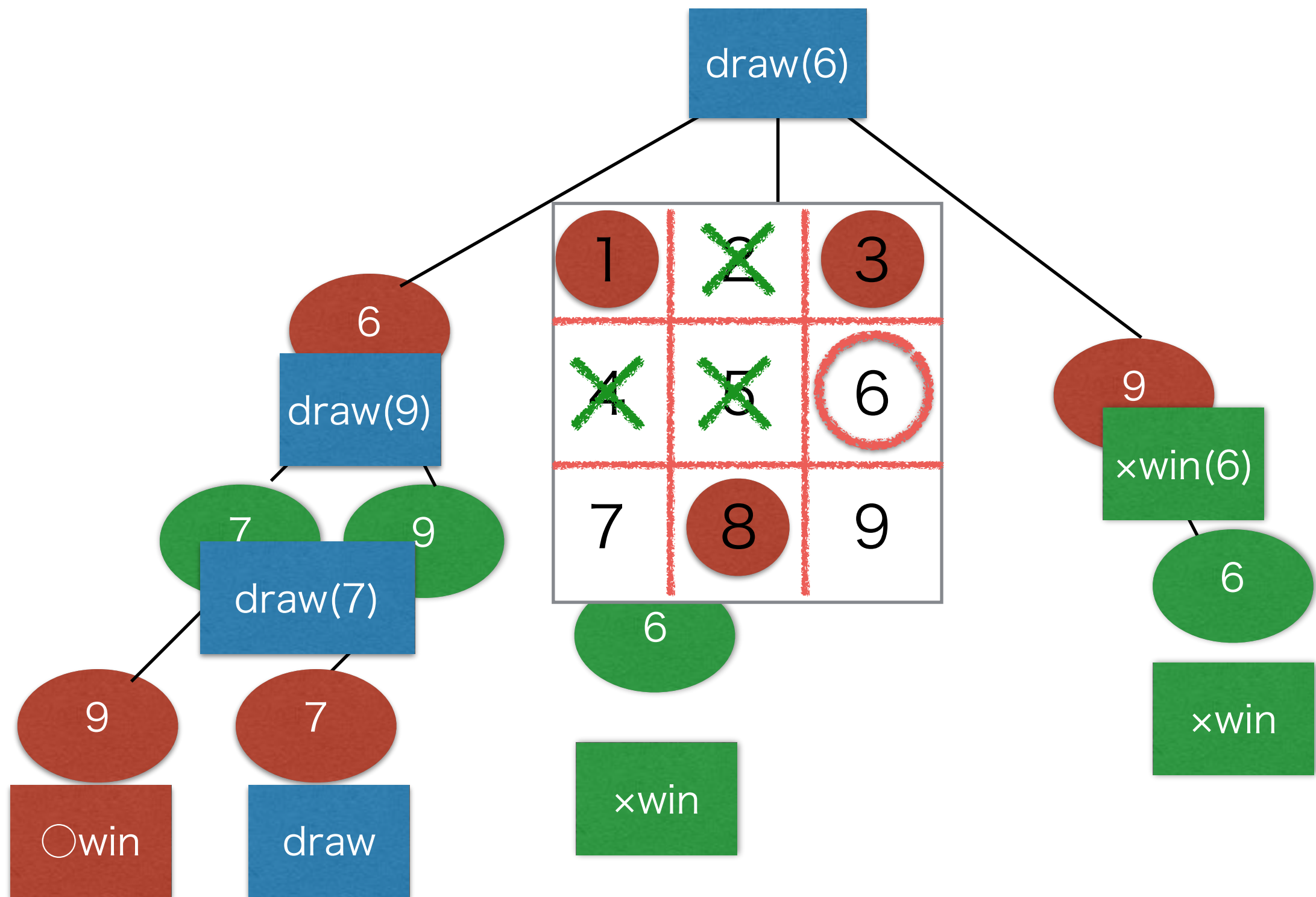
AIについて

- ・ 盤面が3x3なので 3^9 で全探索可能
- ・ 引き分け以上を狙うAIを深さ優先探索で書いた
- ・ (再帰とか回路ではできないのでちょっとめんどくさかった)



1	2	3
4	5	6
7	8	9





AIについて

- ・ Pythonでdfsをかくと20行
- ・ しかし、Verilogで再帰やforは使えないので、毎クロックに処理を展開する
- ・ 比較的めんどくさい

AIについて

```
def dfs(l, turn):  
    w = check(l)  
    if w != 0:  
        return (-1, w)  
    draw = -1  
    for i in range(9):  
        if l[i] == 0:  
            t = put(l, i, turn)  
            ret, w = dfs(l, t)  
            l[i] = 0  
            if w == turn:  
                return (i, w)  
            elif w == 3:  
                draw = i  
    return (draw, turn ^ 0b11 if draw == -1 else 3)
```

```
def dfs(l, turn):
    w = check(l)
    if w != 0:
        return (-
draw = -1
for i in range
    if l[i] ==
        t = p
    ret, w
    l[i] =
    if w ==
    elif w
    d
return (draw,
```

```

1  (Cont) = 170) begin
2  total = 170;
3  if (cont = 0) begin
4  if (put[0] = 0) begin
5  if (C = 0) begin
6  res, flag = 170;
7  end
8  if (for x, y, i in each "00000", str(C), str(C) + 1) for x in range(2) for y in range(2)) {
9  if (board[0][0] = 170) begin
10  res, put = 0, 0;
11  end
12  }
13  end
14  end else begin
15  res, put = put[0];
16  end
17  end else begin
18  res = put[0] - 0;
19  C = C - 0;
20  res, put = res, put + 0;
21  put[0] = 0;
22  end
23  end else begin
24  total = total + 170;
25  if (total = 0) begin
26  if (C = 0) begin
27  end
28  if (for x in each "00000", str(C), str(C) + 1) for x in range(2) {
29  if (board[0][0] = 170 && board[0][1] = 170 && board[0][2] = 170) begin
30  res = 0;
31  end else if (board[0][0] = 170 && board[0][1] = 170 && board[0][2] = 170) begin
32  res = 0;
33  end else if (board[0][0] = 170 && board[0][1] = 170 && board[0][2] = 170) begin
34  res = 0;
35  end else if (board[0][0] = 170 && board[0][1] = 170 && board[0][2] = 170) begin
36  res = 0;
37  end
38  end
39  if (board[0] = board[1] && board[1] = board[2]) begin
40  if (board[0] = 170) begin
41  res = 0;
42  end else if (board[0] = 170) begin
43  res = 0;
44  end
45  end else if (board[0] = board[1] && board[1] = board[2]) begin
46  if (board[1] = 170) begin
47  res = 0;
48  end else if (board[1] = 170) begin
49  res = 0;
50  end
51  end
52  if (C = 1)
53  if (for x, y in each "00000", str(C), str(C) + 1) for x in range(2) for y in range(2) {
54  if (board[0][0] = 170)
55  }
56  } begin
57  res = 0;
58  end else begin
59  res = 0;
60  end
61  end
62  end else begin
63  res = 0;
64  if (current_player = 0 && res = 0) begin
65  current_player = 1;
66  C = C - 0;
67  put[0] = 0;
68  if (draw[0] = 0)
69  res, put = res, put + 0;
70  end else if (current_player = 0 && C = 0) begin
71  res, put = put[0];
72  end else if (C = 0) begin
73  if (draw[0] = 0) begin
74  current_player = 0;
75  end else begin
76  current_player = 0;
77  end
78  end
79  put[0] = draw[0];
80  draw[0] = 0;
81  total = 170;
82  end else begin
83  if (current_player = 0) begin
84  if (C = 0) begin
85  end
86  if (for x, y, i in each "00000", str(C), str(C) + 1) for x in range(2) for y in range(2) {
87  if (C = 0 && board[0][0] = 170) begin
88  if (res = 0) begin
89  board[0][0] = 170;
90  end else begin
91  board[0][0] = 170;
92  end
93  end
94  put[0] = 0;
95  res, put = res, put + 0;
96  C = C - 0;
97  C = 0;
98  end
99  end
100  if (C = 1)
101  if (for x, y, i in each "00000", str(C), str(C) + 1) for x in range(2) for y in range(2) {
102  if (C = 0 && board[0][0] = 170) begin
103  if (C = 0 && board[0][0] = 170) begin
104  end
105  end
106  if (current_player = 0) begin
107  put[0] = 0;
108  draw[0] = 0;
109  res = put[0] - 0;
110  C = C - 0;
111  res, put = res, put + 0;
112  end else if (current_player = 0) begin
113  current_player = 0;
114  end
115  end
116  put[0] = draw[0];
117  draw[0] = 0;
118  total = 170;
119  end
120  end
121  end
122  end
123  end
124  end
125  end
126  end
127  end
128  end
129  end
130  end
131  end
132  end
133  end
134  end
135  end
136  end
137  end
138  end
139  end
140  end
141  end
142  end
143  end
144  end
145  end
146  end
147  end
148  end
149  end
150  end
151  end
152  end
153  end
154  end
155  end
156  end
157  end
158  end
159  end
160  end
161  end
162  end
163  end
164  end
165  end
166  end
167  end
168  end
169  end
170  end
171  end
172  end
173  end
174  end
175  end
176  end
177  end
178  end
179  end
180  end
181  end
182  end
183  end
184  end
185  end
186  end
187  end
188  end
189  end
190  end
191  end
192  end
193  end
194  end
195  end
196  end
197  end
198  end
199  end
200  end
201  end
202  end
203  end
204  end
205  end
206  end
207  end
208  end
209  end
210  end
211  end
212  end
213  end
214  end
215  end
216  end
217  end
218  end
219  end
220  end
221  end
222  end
223  end
224  end
225  end
226  end
227  end
228  end
229  end
230  end
231  end
232  end
233  end
234  end
235  end
236  end
237  end
238  end
239  end
240  end
241  end
242  end
243  end
244  end
245  end
246  end
247  end
248  end
249  end
250  end
251  end
252  end
253  end
254  end
255  end
256  end
257  end
258  end
259  end
260  end
261  end
262  end
263  end
264  end
265  end
266  end
267  end
268  end
269  end
270  end
271  end
272  end
273  end
274  end
275  end
276  end
277  end
278  end
279  end
280  end
281  end
282  end
283  end
284  end
285  end
286  end
287  end
288  end
289  end
290  end
291  end
292  end
293  end
294  end
295  end
296  end
297  end
298  end
299  end
300  end
301  end
302  end
303  end
304  end
305  end
306  end
307  end
308  end
309  end
310  end
311  end
312  end
313  end
314  end
315  end
316  end
317  end
318  end
319  end
320  end
321  end
322  end
323  end
324  end
325  end
326  end
327  end
328  end
329  end
330  end
331  end
332  end
333  end
334  end
335  end
336  end
337  end
338  end
339  end
340  end
341  end
342  end
343  end
344  end
345  end
346  end
347  end
348  end
349  end
350  end
351  end
352  end
353  end
354  end
355  end
356  end
357  end
358  end
359  end
360  end
361  end
362  end
363  end
364  end
365  end
366  end
367  end
368  end
369  end
370  end
371  end
372  end
373  end
374  end
375  end
376  end
377  end
378  end
379  end
380  end
381  end
382  end
383  end
384  end
385  end
386  end
387  end
388  end
389  end
390  end
391  end
392  end
393  end
394  end
395  end
396  end
397  end
398  end
399  end
400  end
401  end
402  end
403  end
404  end
405  end
406  end
407  end
408  end
409  end
410  end
411  end
412  end
413  end
414  end
415  end
416  end
417  end
418  end
419  end
420  end
421  end
422  end
423  end
424  end
425  end
426  end
427  end
428  end
429  end
430  end
431  end
432  end
433  end
434  end
435  end
436  end
437  end
438  end
439  end
440  end
441  end
442  end
443  end
444  end
445  end
446  end
447  end
448  end
449  end
450  end
451  end
452  end
453  end
454  end
455  end
456  end
457  end
458  end
459  end
460  end
461  end
462  end
463  end
464  end
465  end
466  end
467  end
468  end
469  end
470  end
471  end
472  end
473  end
474  end
475  end
476  end
477  end
478  end
479  end
480  end
481  end
482  end
483  end
484  end
485  end
486  end
487  end
488  end
489  end
490  end
491  end
492  end
493  end
494  end
495  end
496  end
497  end
498  end
499  end
500  end
501  end
502  end
503  end
504  end
505  end
506  end
507  end
508  end
509  end
510  end
511  end
512  end
513  end
514  end
515  end
516  end
517  end
518  end
519  end
520  end
521  end
522  end
523  end
524  end
525  end
526  end
527  end
528  end
529  end
530  end
531  end
532  end
533  end
534  end
535  end
536  end
537  end
538  end
539  end
540  end
541  end
542  end
543  end
544  end
545  end
546  end
547  end
548  end
549  end
550  end
551  end
552  end
553  end
554  end
555  end
556  end
557  end
558  end
559  end
560  end
561  end
562  end
563  end
564  end
565  end
566  end
567  end
568  end
569  end
570  end
571  end
572  end
573  end
574  end
575  end
576  end
577  end
578  end
579  end
580  end
581  end
582  end
583  end
584  end
585  end
586  end
587  end
588  end
589  end
590  end
591  end
592  end
593  end
594  end
595  end
596  end
597  end
598  end
599  end
600  end
601  end
602  end
603  end
604  end
605  end
606  end
607  end
608  end
609  end
610  end
611  end
612  end
613  end
614  end
615  end
616  end
617  end
618  end
619  end
620  end
621  end
622  end
623  end
624  end
625  end
626  end
627  end
628  end
629  end
630  end
631  end
632  end
633  end
634  end
635  end
636  end
637  end
638  end
639  end
640  end
641  end
642  end
643  end
644  end
645  end
646  end
647  end
648  end
649  end
650  end
651  end
652  end
653  end
654  end
655  end
656  end
657  end
658  end
659  end
660  end
661  end
662  end
663  end
664  end
665  end
666  end
667  end
668  end
669  end
670  end
671  end
672  end
673  end
674  end
675  end
676  end
677  end
678  end
679  end
680  end
681  end
682  end
683  end
684  end
685  end
686  end
687  end
688  end
689  end
690  end
691  end
692  end
693  end
694  end
695  end
696  end
697  end
698  end
699  end
700  end
701  end
702  end
703  end
704  end
705  end
706  end
707  end
708  end
709  end
710  end
711  end
712  end
713  end
714  end
715  end
716  end
717  end
718  end
719  end
720  end
721  end
722  end
723  end
724  end
725  end
726  end
727  end
728  end
729  end
730  end
731  end
732  end
733  end
734  end
735  end
736  end
737  end
738  end
739  end
740  end
741  end
742  end
743  end
744  end
745  end
746  end
747  end
748  end
749  end
750  end
751  end
752  end
753  end
754  end
755  end
756  end
757  end
758  end
759  end
760  end
761  end
762  end
763  end
764  end
765  end
766  end
767  end
768  end
769  end
770  end
771  end
772  end
773  end
774  end
775  end
776  end
777  end
778  end
779  end
780  end
781  end
782  end
783  end
784  end
785  end
786  end
787  end
788  end
789  end
790  end
791  end
792  end
793  end
794  end
795  end
796  end
797  end
798  end
799  end
800  end
801  end
802  end
803  end
804  end
805  end
806  end
807  end
808  end
809  end
810  end
811  end
812  end
813  end
814  end
815  end
816  end
817  end
818  end
819  end
820  end
821  end
822  end
823  end
824  end
825  end
826  end
827  end
828  end
829  end
830  end
831  end
832  end
833  end
834  end
835  end
836  end
837  end
838  end
839  end
840  end
841  end
842  end
843  end
844  end
845  end
846  end
847  end
848  end
849  end
850  end
851  end
852  end
853  end
854  end
855  end
856  end
857  end
858  end
859  end
860  end
861  end
862  end
863  end
864  end
865  end
866  end
867  end
868  end
869  end
870  end
871  end
872  end
873  end
874  end
875  end
876  end
877  end
878  end
879  end
880  end
881  end
882  end
883  end
884  end
885  end
886  end
887  end
888  end
889  end
890  end
891  end
892  end
893  end
894  end
895  end
896  end
897  end
898  end
899  end
900  end
901  end
902  end
903  end
904  end
905  end
906  end
907  end
908  end
909  end
910  end
911  end
912  end
913  end
914  end
915  end
916  end
917  end
918  end
919  end
920  end
921  end
922  end
923  end
924  end
925  end
926  end
927  end
928  end
929  end
930  end
931  end
932  end
933  end
934  end
935  end
936  end
937  end
938  end
939  end
940  end
941  end
942  end
943  end
944  end
945  end
946  end
947  end
948  end
949  end
950  end
951  end
952  end
953  end
954  end
955  end
956  end
957  end
958  end
959  end
960  end
961  end
962  end
963  end
964  end
965  end
966  end
967  end
968  end
969  end
970  end
971  end
972  end
973  end
974  end
975  end
976  end
977  end
978  end
979  end
980  end
981  end
982  end
983  end
984  end
985  end
986  end
987  end
988  end
989  end
990  end
991  end
992  end
993  end
994  end
995  end
996  end
997  end
998  end
999  end
1000  end

```

```
draw == -1 else 3)
```

大変だったポイント

- ・ ハードウェア的な問題
- ・ デバッグの問題
- ・ Verilogを書くのがきつい問題

ハードウェア的な問題

ハードウェア的な問題

次のようなコードを考えます

ハードウェア的な問題

次のよう

```
2 x は2bitの値
3
4 if (x == 2'b00) begin
5     黄色の円を表示
6 end
7 else if (x == 2'b01) begin
8     緑色の円を表示
9 end
10 else if (x == 2'b10) begin
11     青色の円を表示
12 end
13 else if (x == 2'b11) begin
14     白色の円を表示
15 end
16 else begin
17     赤色の円を表示
18 end
```

ハードウェア的な問題

次のよう

```
2 x は2bitの値
3
4 if (x == 2'b00) begin
5     黄色の円を表示
6 end
7 else if (x == 2'b01) begin
8     緑色の円を表示
9 end
10 else if (x == 2'b10) begin
11     青色の円を表示
12 end
13 else if (x == 2'b11) begin
14     白色の円を表示
15 end
16 else begin
17     赤色の円を表示
18 end
```

冷静に考えて、このelseって
起こらなくない？

ハードウェア的な問題

次のよう

```
2 x は2bitの値
3
4 if (x == 2'b00) begin
5     黄色の円を表示
6 end
7 else if (x == 2'b01) begin
8     緑色の円を表示
9 end
10 else if (x == 2'b10) begin
11     青色の円を表示
12 end
13 else if (x == 2'b11) begin
14     白色の円を表示
15 end
16 else begin
17     赤色の
18 end
```

これが起こる。

のelseって
ない？

ハードウェア的な問題

次のよう

```
2 x は2bitの値
```

```
3
```

```
4 if (x == 2'b00) begin
```

驚くべきは、if文の中身が（挙動上）中途半端に実行されているようだったこと

```
12 end
```

```
13 else if (x == 2'b11) begin
```

```
14   白色の円を表示
```

```
15 end
```

```
16 else begin
```

```
17   赤色の
```

```
18 end
```

これが起こる。

のelseって
ない？

ハードウェア的な問題

- ・ ハードウェアの並列性を無視してブロッキング代入をすると途中で値が変化したりしてしまう
- ・ ボタンのチャタリング除去をすると治った

デバッグの問題

デバッグの問題

- ・ 基本的に、Verilogでコードを書き、
- ・ FPGAに乗せて挙動を確認

デバッグの問題

- ・ 基本的に、Verilogでコードを書き、

- ・ FPGAに乗せて挙動を確認
この繰り返しで実装

デバッグの問題

- ・ Pythonで動くコードをVerilogに焼き直したら動かない＞＜
- ・ しかも、どのパラメタが壊れているのかわからない

デバッグの問題

- ・ printfデバッグがしたい
- ・ が、printfは当然できないので、

デバッグの問題

- ・ printfデバッグがしたい

- ・ が

```
if (param < 0 || param > 10) begin  
    画面を赤くする  
end
```

- ・ このようにおかしい値を検知して気合いで直していた

デバッグの問題

- ・ printfデバッグがしたい

- ・ が

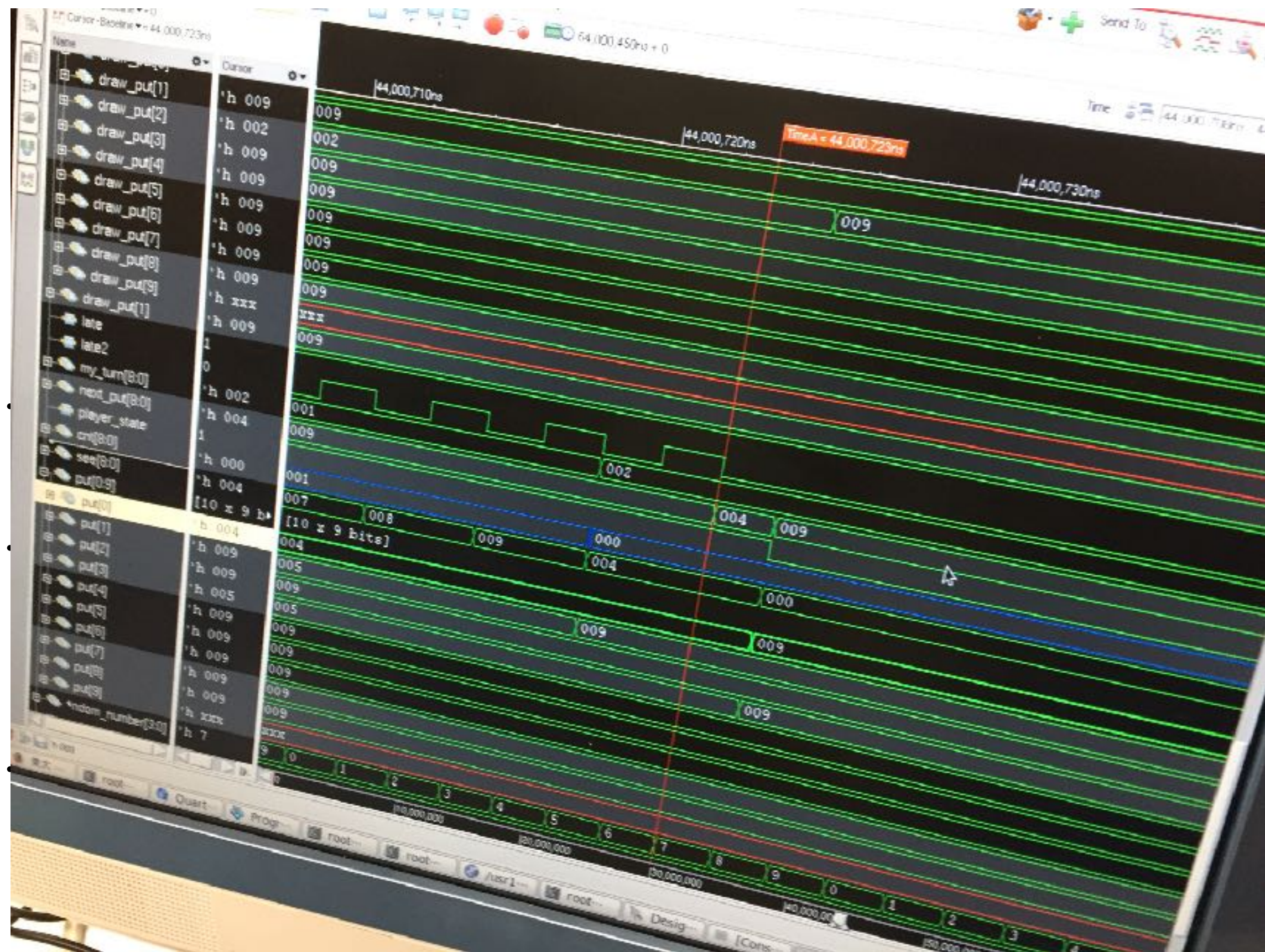
```
if (param < 0 || param > 10) begin  
    画面を赤くする  
end
```

- ・ このようにおかしい値
た

しかし、深さ優先探索の部分のコードが
どうも動かない

デバッグの問題

- ・ シミュレータを変数の値を可視化して解決
- ・ (TAの方にシミュレータを実行するコードを書い
てもらった)
- ・ これにより実質的なprintfデバッグができた



特定のクロックでの
レジスタの値がわかる



Verilogを書くのがきつい問題

Verilogを書くのがきつい問題

回路に組み込むときには、いくつか制限があり、以下の欲求があった

- ・ forは使いたい
- ・ 定数を1箇所に管理したい
- ・ コンパイル前に解決できる定数を解決したい

Verilogを書くのがきつい問題

回路に組み込むときには、いくつか制限があり、以下の欲求がある

勢いで言語拡張を書いていた

- ・ forは使いたくない
- ・ 定数を1箇所に管理したい
- ・ コンパイル前に解決できる定数を解決したい

Verilogを書くのがきつい問題

実装した機能

- ・ forによるループ展開
- ・ 定数の埋め込みと内部でPythonによるスニペット実行
- ・ importによるファイル分割
- ・ 特定の場所に画像を埋め込む機能

Verilogを書くのがきつい問題

- ・ forによるループ展開

回路を書く時にforは非推奨らしい（展開してくれるコンパイラもあるようだが、そうでないものも存在するため）

するとコピペコーディングをしまくることになる

これはつらい

Verilogを書くのがきつい問題

- ・ forによるループ展開

回路を書く時にforは非推奨らしい（展開してくれるコンパイラもあるようだが、そうでないものも存在するため）

するとコピペコーディング

forによる展開を自動生成
できるようにした

これはつらい

- ・ forによるループ展開

具体的には、次のような感じ

```
[[for i in 1, 2, 3 {  
    if (x == {{i}}) begin  
        // hogehoge  
    end  
}]
```

・ forに

具体的には、次のような感

```
if (x == 1'd1) begin  
    // hogehoge  
end  
if (x == 2'd2) begin  
    // hogehoge  
end  
if (x == 2'd3) begin  
    // hogehoge  
end
```

```
// [Compile] 1502037502.728922
```

まとめ

- ・ VerilogでTic Tac Toeをハードウェアを設計した
- ・ 作ったもの自体は簡単だが、FPGA上で正しく動作させるのはソフトウェアで行うのと比べて難しかった