





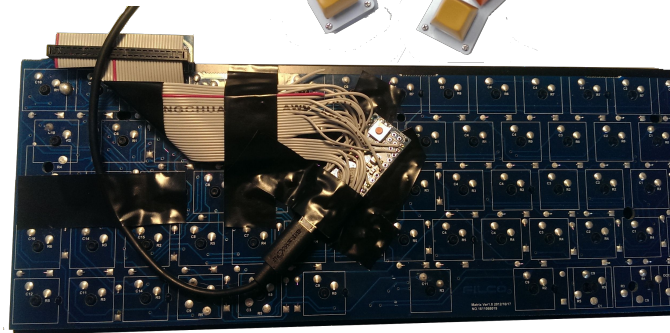
# Elixir keyboard!

ElixirConf EU 2020



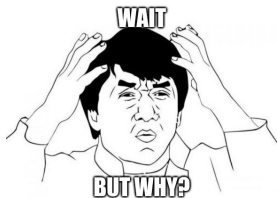
# Speaker

- André Albuquerque
  - @amalbuquerque 
- Software Engineer @ Hopin 
- (Mild) obsession with keyboards



# Outline

- Wait, but why?



- Nerves 101



- Baby steps



- A keyboard on top of the BEAM



- Pros & Cons



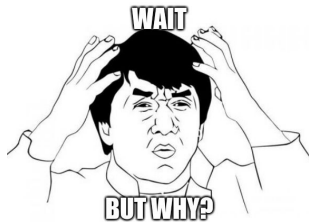
- Future work



- D E M O time



# Wait, but why?



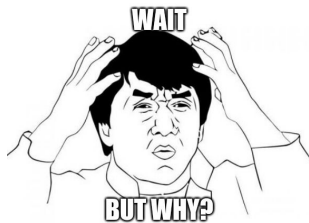
- Keyboards <3
- Custom firmwares based on C
  - TMK (and its notable QMK fork)



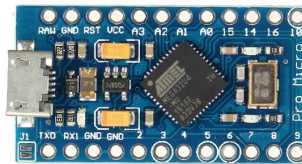
```
18 enum minila_layers {
19   _QWERTY, // Qwerty
20   _BASIC,  // Basic qwerty
21   _F1_LAYER, // Symbols
22   _F4_LAYER // Multimedia
23 };
24
25
26 enum custom_keycodes {
27   CTL_TAB = CTL_T(KC_TAB),
28   CTL_BSP = CTL_T(KC_BSPC),
29   TTO_FN1 = TT(_F1_LAYER),
30   TTO_FN4 = TT(_F4_LAYER),
31   SPC_FN1 = LT(_F1_LAYER, KC_SPC),
32   APP_FN4 = LT(_F4_LAYER, KC_APP),
33   SPC_LGUI = MT(MOD_LGUI, KC_SPC),
34   NKROTG = MAGIC_TOGGLE_NKRO
35 };
```

```
43 const uint16_t PROGMEM keymaps[][MATRIX_ROWS][MATRIX_COLS] = {
44   [_QWERTY] = LAYOUT(
45     KC_GRV,  KC_1,  KC_2,  KC_3,  KC_4,  KC_5,  KC_6,  KC_7,  KC_8,  KC_9,  KC_0,  KC_MINS, KC_EQL, KC_GRV, KC_BSPC,  \
46     KC_ESC,  KC_Q,  KC_W,  KC_E,  KC_R,  KC_T,  KC_Y,  KC_U,  KC_I,  KC_O,  KC_P,   KC_LBRC, KC_RBRC,   KC_BSLS,  \
47     CTL_TAB, KC_A,  KC_S,  KC_D,  KC_F,  KC_G,  KC_H,  KC_J,  KC_K,  KC_L,   KC_SCLN, KC_QUOT,   KC_ENT,   \
48     KC_LSPO, KC_Z,  KC_X,  KC_C,  KC_V,  KC_B,  KC_N,  KC_M,  KC_COMM, KC_DOT, KC_SLSH, KC_RSPC, KC_UP,  KC_DELETE, \
49     TTO_FN1, KC_LGUI, KC_LALT, SPC_LGUI, SPC_FN1, CTL_BSP, KC_RALT, APP_FN4, KC_LEFT, KC_DOWN, TTO_FN4  \
50   ),
51
52   [_F1_LAYER] = LAYOUT(
53     // ...
54   ),
55
56   [_F4_LAYER] = LAYOUT(
57     // ...
58   ),
59
60   [_FN5_LAYER] = LAYOUT(
61     // ...
62   ),
63
64   [_FN6_LAYER] = LAYOUT(
65     // ...
66   ),
67
68   [_FN7_LAYER] = LAYOUT(
69     // ...
70   ),
71
72   [_FN8_LAYER] = LAYOUT(
73     // ...
74   ),
75
76   [_FN9_LAYER] = LAYOUT(
77     // ...
78   ),
79
80   [_FN10_LAYER] = LAYOUT(
81     // ...
82   ),
83
84   [_FN11_LAYER] = LAYOUT(
85     // ...
86   ),
87
88   [_FN12_LAYER] = LAYOUT(
89     // ...
90   ),
91
92   [_FN13_LAYER] = LAYOUT(
93     // ...
94   ),
95
96   [_FN14_LAYER] = LAYOUT(
97     // ...
98   ),
99
100  [_FN15_LAYER] = LAYOUT(
101    // ...
102  ),
103
104  [_FN16_LAYER] = LAYOUT(
105    // ...
106  ),
107
108  [_FN17_LAYER] = LAYOUT(
109    // ...
110  ),
111
112  [_FN18_LAYER] = LAYOUT(
113    // ...
114  ),
115
116  [_FN19_LAYER] = LAYOUT(
117    // ...
118  ),
119
120  [_FN20_LAYER] = LAYOUT(
121    // ...
122  ),
123
124  [_FN21_LAYER] = LAYOUT(
125    // ...
126  ),
127
128  [_FN22_LAYER] = LAYOUT(
129    // ...
130  ),
131
132  [_FN23_LAYER] = LAYOUT(
133    // ...
134  ),
135
136  [_FN24_LAYER] = LAYOUT(
137    // ...
138  ),
139
140  [_FN25_LAYER] = LAYOUT(
141    // ...
142  ),
143
144  [_FN26_LAYER] = LAYOUT(
145    // ...
146  ),
147
148  [_FN27_LAYER] = LAYOUT(
149    // ...
150  ),
151
152  [_FN28_LAYER] = LAYOUT(
153    // ...
154  ),
155
156  [_FN29_LAYER] = LAYOUT(
157    // ...
158  ),
159
160  [_FN30_LAYER] = LAYOUT(
161    // ...
162  ),
163
164  [_FN31_LAYER] = LAYOUT(
165    // ...
166  ),
167
168  [_FN32_LAYER] = LAYOUT(
169    // ...
170  ),
171
172  [_FN33_LAYER] = LAYOUT(
173    // ...
174  ),
175
176  [_FN34_LAYER] = LAYOUT(
177    // ...
178  ),
179
180  [_FN35_LAYER] = LAYOUT(
181    // ...
182  ),
183
184  [_FN36_LAYER] = LAYOUT(
185    // ...
186  ),
187
188  [_FN37_LAYER] = LAYOUT(
189    // ...
190  ),
191
192  [_FN38_LAYER] = LAYOUT(
193    // ...
194  ),
195
196  [_FN39_LAYER] = LAYOUT(
197    // ...
198  ),
199
200  [_FN40_LAYER] = LAYOUT(
201    // ...
202  ),
203
204  [_FN41_LAYER] = LAYOUT(
205    // ...
206  ),
207
208  [_FN42_LAYER] = LAYOUT(
209    // ...
210  ),
211
212  [_FN43_LAYER] = LAYOUT(
213    // ...
214  ),
215
216  [_FN44_LAYER] = LAYOUT(
217    // ...
218  ),
219
220  [_FN45_LAYER] = LAYOUT(
221    // ...
222  ),
223
224  [_FN46_LAYER] = LAYOUT(
225    // ...
226  ),
227
228  [_FN47_LAYER] = LAYOUT(
229    // ...
230  ),
231
232  [_FN48_LAYER] = LAYOUT(
233    // ...
234  ),
235
236  [_FN49_LAYER] = LAYOUT(
237    // ...
238  ),
239
240  [_FN50_LAYER] = LAYOUT(
241    // ...
242  ),
243
244  [_FN51_LAYER] = LAYOUT(
245    // ...
246  ),
247
248  [_FN52_LAYER] = LAYOUT(
249    // ...
250  ),
251
252  [_FN53_LAYER] = LAYOUT(
253    // ...
254  ),
255
256  [_FN54_LAYER] = LAYOUT(
257    // ...
258  ),
259
260  [_FN55_LAYER] = LAYOUT(
261    // ...
262  ),
263
264  [_FN56_LAYER] = LAYOUT(
265    // ...
266  ),
267
268  [_FN57_LAYER] = LAYOUT(
269    // ...
270  ),
271
272  [_FN58_LAYER] = LAYOUT(
273    // ...
274  ),
275
276  [_FN59_LAYER] = LAYOUT(
277    // ...
278  ),
279
280  [_FN60_LAYER] = LAYOUT(
281    // ...
282  ),
283
284  [_FN61_LAYER] = LAYOUT(
285    // ...
286  ),
287
288  [_FN62_LAYER] = LAYOUT(
289    // ...
290  ),
291
292  [_FN63_LAYER] = LAYOUT(
293    // ...
294  ),
295
296  [_FN64_LAYER] = LAYOUT(
297    // ...
298  ),
299
300  [_FN65_LAYER] = LAYOUT(
301    // ...
302  ),
303
304  [_FN66_LAYER] = LAYOUT(
305    // ...
306  ),
307
308  [_FN67_LAYER] = LAYOUT(
309    // ...
310  ),
311
312  [_FN68_LAYER] = LAYOUT(
313    // ...
314  ),
315
316  [_FN69_LAYER] = LAYOUT(
317    // ...
318  ),
319
320  [_FN70_LAYER] = LAYOUT(
321    // ...
322  ),
323
324  [_FN71_LAYER] = LAYOUT(
325    // ...
326  ),
327
328  [_FN72_LAYER] = LAYOUT(
329    // ...
330  ),
331
332  [_FN73_LAYER] = LAYOUT(
333    // ...
334  ),
335
336  [_FN74_LAYER] = LAYOUT(
337    // ...
338  ),
339
340  [_FN75_LAYER] = LAYOUT(
341    // ...
342  ),
343
344  [_FN76_LAYER] = LAYOUT(
345    // ...
346  ),
347
348  [_FN77_LAYER] = LAYOUT(
349    // ...
350  ),
351
352  [_FN78_LAYER] = LAYOUT(
353    // ...
354  ),
355
356  [_FN79_LAYER] = LAYOUT(
357    // ...
358  ),
359
360  [_FN80_LAYER] = LAYOUT(
361    // ...
362  ),
363
364  [_FN81_LAYER] = LAYOUT(
365    // ...
366  ),
367
368  [_FN82_LAYER] = LAYOUT(
369    // ...
370  ),
371
372  [_FN83_LAYER] = LAYOUT(
373    // ...
374  ),
375
376  [_FN84_LAYER] = LAYOUT(
377    // ...
378  ),
379
380  [_FN85_LAYER] = LAYOUT(
381    // ...
382  ),
383
384  [_FN86_LAYER] = LAYOUT(
385    // ...
386  ),
387
388  [_FN87_LAYER] = LAYOUT(
389    // ...
390  ),
391
392  [_FN88_LAYER] = LAYOUT(
393    // ...
394  ),
395
396  [_FN89_LAYER] = LAYOUT(
397    // ...
398  ),
399
400  [_FN90_LAYER] = LAYOUT(
401    // ...
402  ),
403
404  [_FN91_LAYER] = LAYOUT(
405    // ...
406  ),
407
408  [_FN92_LAYER] = LAYOUT(
409    // ...
410  ),
411
412  [_FN93_LAYER] = LAYOUT(
413    // ...
414  ),
415
416  [_FN94_LAYER] = LAYOUT(
417    // ...
418  ),
419
420  [_FN95_LAYER] = LAYOUT(
421    // ...
422  ),
423
424  [_FN96_LAYER] = LAYOUT(
425    // ...
426  ),
427
428  [_FN97_LAYER] = LAYOUT(
429    // ...
430  ),
431
432  [_FN98_LAYER] = LAYOUT(
433    // ...
434  ),
435
436  [_FN99_LAYER] = LAYOUT(
437    // ...
438  ),
439
440  [_FN100_LAYER] = LAYOUT(
441    // ...
442  ),
443
444  [_FN101_LAYER] = LAYOUT(
445    // ...
446  ),
447
448  [_FN102_LAYER] = LAYOUT(
449    // ...
450  ),
451
452  [_FN103_LAYER] = LAYOUT(
453    // ...
454  ),
455
456  [_FN104_LAYER] = LAYOUT(
457    // ...
458  ),
459
460  [_FN105_LAYER] = LAYOUT(
461    // ...
462  ),
463
464  [_FN106_LAYER] = LAYOUT(
465    // ...
466  ),
467
468  [_FN107_LAYER] = LAYOUT(
469    // ...
470  ),
471
472  [_FN108_LAYER] = LAYOUT(
473    // ...
474  ),
475
476  [_FN109_LAYER] = LAYOUT(
477    // ...
478  ),
479
480  [_FN110_LAYER] = LAYOUT(
481    // ...
482  ),
483
484  [_FN111_LAYER] = LAYOUT(
485    // ...
486  ),
487
488  [_FN112_LAYER] = LAYOUT(
489    // ...
490  ),
491
492  [_FN113_LAYER] = LAYOUT(
493    // ...
494  ),
495
496  [_FN114_LAYER] = LAYOUT(
497    // ...
498  ),
499
500  [_FN115_LAYER] = LAYOUT(
501    // ...
502  ),
503
504  [_FN116_LAYER] = LAYOUT(
505    // ...
506  ),
507
508  [_FN117_LAYER] = LAYOUT(
509    // ...
510  ),
511
512  [_FN118_LAYER] = LAYOUT(
513    // ...
514  ),
515
516  [_FN119_LAYER] = LAYOUT(
517    // ...
518  ),
519
520  [_FN120_LAYER] = LAYOUT(
521    // ...
522  ),
523
524  [_FN121_LAYER] = LAYOUT(
525    // ...
526  ),
527
528  [_FN122_LAYER] = LAYOUT(
529    // ...
530  ),
531
532  [_FN123_LAYER] = LAYOUT(
533    // ...
534  ),
535
536  [_FN124_LAYER] = LAYOUT(
537    // ...
538  ),
539
540  [_FN125_LAYER] = LAYOUT(
541    // ...
542  ),
543
544  [_FN126_LAYER] = LAYOUT(
545    // ...
546  ),
547
548  [_FN127_LAYER] = LAYOUT(
549    // ...
550  ),
551
552  [_FN128_LAYER] = LAYOUT(
553    // ...
554  ),
555
556  [_FN129_LAYER] = LAYOUT(
557    // ...
558  ),
559
560  [_FN130_LAYER] = LAYOUT(
561    // ...
562  ),
563
564  [_FN131_LAYER] = LAYOUT(
565    // ...
566  ),
567
568  [_FN132_LAYER] = LAYOUT(
569    // ...
570  ),
571
572  [_FN133_LAYER] = LAYOUT(
573    // ...
574  ),
575
576  [_FN134_LAYER] = LAYOUT(
577    // ...
578  ),
579
580  [_FN135_LAYER] = LAYOUT(
581    // ...
582  ),
583
584  [_FN136_LAYER] = LAYOUT(
585    // ...
586  ),
587
588  [_FN137_LAYER] = LAYOUT(
589    // ...
590  ),
591
592  [_FN138_LAYER] = LAYOUT(
593    // ...
594  ),
595
596  [_FN139_LAYER] = LAYOUT(
597    // ...
598  ),
599
600  [_FN140_LAYER] = LAYOUT(
601    // ...
602  ),
603
604  [_FN141_LAYER] = LAYOUT(
605    // ...
606  ),
607
608  [_FN142_LAYER] = LAYOUT(
609    // ...
610  ),
611
612  [_FN143_LAYER] = LAYOUT(
613    // ...
614  ),
615
616  [_FN144_LAYER] = LAYOUT(
617    // ...
618  ),
619
620  [_FN145_LAYER] = LAYOUT(
621    // ...
622  ),
623
624  [_FN146_LAYER] = LAYOUT(
625    // ...
626  ),
627
628  [_FN147_LAYER] = LAYOUT(
629    // ...
630  ),
631
632  [_FN148_LAYER] = LAYOUT(
633    // ...
634  ),
635
636  [_FN149_LAYER] = LAYOUT(
637    // ...
638  ),
639
640  [_FN150_LAYER] = LAYOUT(
641    // ...
642  ),
643
644  [_FN151_LAYER] = LAYOUT(
645    // ...
646  ),
647
648  [_FN152_LAYER] = LAYOUT(
649    // ...
650  ),
651
652  [_FN153_LAYER] = LAYOUT(
653    // ...
654  ),
655
656  [_FN154_LAYER] = LAYOUT(
657    // ...
658  ),
659
660  [_FN155_LAYER] = LAYOUT(
661    // ...
662  ),
663
664  [_FN156_LAYER] = LAYOUT(
665    // ...
666  ),
667
668  [_FN157_LAYER] = LAYOUT(
669    // ...
670  ),
671
672  [_FN158_LAYER] = LAYOUT(
673    // ...
674  ),
675
676  [_FN159_LAYER] = LAYOUT(
677    // ...
678  ),
679
680  [_FN160_LAYER] = LAYOUT(
681    // ...
682  ),
683
684  [_FN161_LAYER] = LAYOUT(
685    // ...
686  ),
687
688  [_FN162_LAYER] = LAYOUT(
689    // ...
690  ),
691
692  [_FN163_LAYER] = LAYOUT(
693    // ...
694  ),
695
696  [_FN164_LAYER] = LAYOUT(
697    // ...
698  ),
699
700  [_FN165_LAYER] = LAYOUT(
701    // ...
702  ),
703
704  [_FN166_LAYER] = LAYOUT(
705    // ...
706  ),
707
708  [_FN167_LAYER] = LAYOUT(
709    // ...
710  ),
711
712  [_FN168_LAYER] = LAYOUT(
713    // ...
714  ),
715
716  [_FN169_LAYER] = LAYOUT(
717    // ...
718  ),
719
720  [_FN170_LAYER] = LAYOUT(
721    // ...
722  ),
723
724  [_FN171_LAYER] = LAYOUT(
725    // ...
726  ),
727
728  [_FN172_LAYER] = LAYOUT(
729    // ...
730  ),
731
732  [_FN173_LAYER] = LAYOUT(
733    // ...
734  ),
735
736  [_FN174_LAYER] = LAYOUT(
737    // ...
738  ),
739
740  [_FN175_LAYER] = LAYOUT(
741    // ...
742  ),
743
744  [_FN176_LAYER] = LAYOUT(
745    // ...
746  ),
747
748  [_FN177_LAYER] = LAYOUT(
749    // ...
750  ),
751
752  [_FN178_LAYER] = LAYOUT(
753    // ...
754  ),
755
756  [_FN179_LAYER] = LAYOUT(
757    // ...
758  ),
759
760  [_FN180_LAYER] = LAYOUT(
761    // ...
762  ),
763
764  [_FN181_LAYER] = LAYOUT(
765    // ...
766  ),
767
768  [_FN182_LAYER] = LAYOUT(
769    // ...
770  ),
771
772  [_FN183_LAYER] = LAYOUT(
773    // ...
774  ),
775
776  [_FN184_LAYER] = LAYOUT(
777    // ...
778  ),
779
780  [_FN185_LAYER] = LAYOUT(
781    // ...
782  ),
783
784  [_FN186_LAYER] = LAYOUT(
785    // ...
786  ),
787
788  [_FN187_LAYER] = LAYOUT(
789    // ...
790  ),
791
792  [_FN188_LAYER] = LAYOUT(
793    // ...
794  ),
795
796  [_FN189_LAYER] = LAYOUT(
797    // ...
798  ),
799
800  [_FN190_LAYER] = LAYOUT(
801    // ...
802  ),
803
804  [_FN191_LAYER] = LAYOUT(
805    // ...
806  ),
807
808  [_FN192_LAYER] = LAYOUT(
809    // ...
810  ),
811
812  [_FN193_LAYER] = LAYOUT(
813    // ...
814  ),
815
816  [_FN194_LAYER] = LAYOUT(
817    // ...
818  ),
819
820  [_FN195_LAYER] = LAYOUT(
821    // ...
822  ),
823
824  [_FN196_LAYER] = LAYOUT(
825    // ...
826  ),
827
828  [_FN197_LAYER] = LAYOUT(
829    // ...
830  ),
831
832  [_FN198_LAYER] = LAYOUT(
833    // ...
834  ),
835
836  [_FN199_LAYER] = LAYOUT(
837    // ...
838  ),
839
840  [_FN200_LAYER] = LAYOUT(
841    // ...
842  ),
843
844  [_FN201_LAYER] = LAYOUT(
845    // ...
846  ),
847
848  [_FN202_LAYER] = LAYOUT(
849    // ...
850  ),
851
852  [_FN203_LAYER] = LAYOUT(
853    // ...
854  ),
855
856  [_FN204_LAYER] = LAYOUT(
857    // ...
858  ),
859
860  [_FN205_LAYER] = LAYOUT(
861    // ...
862  ),
863
864  [_FN206_LAYER] = LAYOUT(
865    // ...
866  ),
867
868  [_FN207_LAYER] = LAYOUT(
869    // ...
870  ),
871
872  [_FN208_LAYER] = LAYOUT(
873    // ...
874  ),
875
876  [_FN209_LAYER] = LAYOUT(
877    // ...
878  ),
879
880  [_FN210_LAYER] = LAYOUT(
881    // ...
882  ),
883
884  [_FN211_LAYER] = LAYOUT(
885    // ...
886  ),
887
888  [_FN212_LAYER] = LAYOUT(
889    // ...
890  ),
891
892  [_FN213_LAYER] = LAYOUT(
893    // ...
894  ),
895
896  [_FN214_LAYER] = LAYOUT(
897    // ...
898  ),
899
900  [_FN215_LAYER] = LAYOUT(
901    // ...
902  ),
903
904  [_FN216_LAYER] = LAYOUT(
905    // ...
906  ),
907
908  [_FN217_LAYER] = LAYOUT(
909    // ...
910  ),
911
912  [_FN218_LAYER] = LAYOUT(
913    // ...
914  ),
915
916  [_FN219_LAYER] = LAYOUT(
917    // ...
918  ),
919
920  [_FN220_LAYER] = LAYOUT(
921    // ...
922  ),
923
924  [_FN221_LAYER] = LAYOUT(
925    // ...
926  ),
927
928  [_FN222_LAYER] = LAYOUT(
929    // ...
930  ),
931
932  [_FN223_LAYER] = LAYOUT(
933    // ...
934  ),
935
936  [_FN224_LAYER] = LAYOUT(
937    // ...
938  ),
939
940  [_FN225_LAYER] = LAYOUT(
941    // ...
942  ),
943
944  [_FN226_LAYER] = LAYOUT(
945    // ...
946  ),
947
948  [_FN227_LAYER] = LAYOUT(
949    // ...
950  ),
951
952  [_FN228_LAYER] = LAYOUT(
953    // ...
954  ),
955
956  [_FN229_LAYER] = LAYOUT(
957    // ...
958  ),
959
960  [_FN230_LAYER] = LAYOUT(
961    // ...
962  ),
963
964  [_FN231_LAYER] = LAYOUT(
965    // ...
966  ),
967
968  [_FN232_LAYER] = LAYOUT(
969    // ...
970  ),
971
972  [_FN233_LAYER] = LAYOUT(
973    // ...
974  ),
975
976  [_FN234_LAYER] = LAYOUT(
977    // ...
978  ),
979
980  [_FN235_LAYER] = LAYOUT(
981    // ...
982  ),
983
984  [_FN236_LAYER] = LAYOUT(
985    // ...
986  ),
987
988  [_FN237_LAYER] = LAYOUT(
989    // ...
990  ),
991
992  [_FN238_LAYER] = LAYOUT(
993    // ...
994  ),
995
996  [_FN239_LAYER] = LAYOUT(
997    // ...
998  ),
999
1000  [_FN240_LAYER] = LAYOUT(
1001    // ...
1002  ),
1003
1004  [_FN241_LAYER] = LAYOUT(
1005    // ...
1006  ),
1007
1008  [_FN242_LAYER] = LAYOUT(
1009    // ...
1010  ),
1011
1012  [_FN243_LAYER] = LAYOUT(
1013    // ...
1014  ),
1015
1016  [_FN244_LAYER] = LAYOUT(
1017    // ...
1018  ),
1019
1020  [_FN245_LAYER] = LAYOUT(
1021    // ...
1022  ),
1023
1024  [_FN246_LAYER] = LAYOUT(
1025    // ...
1026  ),
1027
1028  [_FN247_LAYER] = LAYOUT(
1029    // ...
1030  ),
1031
1032  [_FN248_LAYER] = LAYOUT(
1033    // ...
1034  ),
1035
1036  [_FN249_LAYER] = LAYOUT(
1037    // ...
1038  ),
1039
1040  [_FN250_LAYER] = LAYOUT(
1041    // ...
1042  ),
1043
1044  [_FN251_LAYER] = LAYOUT(
1045    // ...
1046  ),
1047
1048  [_FN252_LAYER] = LAYOUT(
1049    // ...
1050  ),
1051
1052  [_FN253_LAYER] = LAYOUT(
1053    // ...
1054  ),
1055
1056  [_FN254_LAYER] = LAYOUT(
1057    // ...
1058  ),
1059
1060  [_FN255_LAYER] = LAYOUT(
1061    // ...
1062  ),
1063
1064  [_FN256_LAYER] = LAYOUT(
1065    // ...
1066  ),
1067
1068  [_FN257_LAYER] = LAYOUT(
1069    // ...
1070  ),
1071
1072  [_FN258_LAYER] = LAYOUT(
1073    // ...
1074  ),
1075
1076  [_FN259_LAYER] = LAYOUT(
1077    // ...
1078  ),
1079
1080  [_FN260_LAYER] = LAYOUT(
1081    // ...
1082  ),
1083
1084  [_FN261_LAYER] = LAYOUT(
1085    // ...
1086  ),
1087
1088  [_FN262_LAYER] = LAYOUT(
1089    // ...
1090  ),
1091
1092  [_FN263_LAYER] = LAYOUT(
1093    // ...
1094  ),
1095
1096  [_FN264_LAYER] = LAYOUT(
1097    // ...
1098  ),
1099
1100  [_FN265_LAYER] = LAYOUT(
1101    // ...
1102  ),
1103
1104  [_FN266_LAYER] = LAYOUT(
1105    // ...
1106  ),
1107
1108  [_FN267_LAYER] = LAYOUT(
1109    // ...
1110  ),
1111
1112  [_FN268_LAYER] = LAYOUT(
1113    // ...
1114  ),
1115
1116  [_FN269_LAYER] = LAYOUT(
1117    // ...
1118  ),
1119
1120  [_FN270_LAYER] = LAYOUT(
1121    // ...
1122  ),
1123
1124  [_FN271_LAYER] = LAYOUT(
1125    // ...
1126  ),
1127
1128  [_FN272_LAYER] = LAYOUT(
1129    // ...
1130  ),
1131
1132  [_FN273_LAYER] = LAYOUT(
1133    // ...
1134  ),
1135
1136  [_FN274_LAYER] = LAYOUT(
1137    // ...
1138  ),
1139
1140  [_FN275_LAYER] = LAYOUT(
1141    // ...
1142  ),
1143
1144  [_FN276_LAYER] = LAYOUT(
1145    // ...
1146  ),
1147
1148  [_FN277_LAYER] = LAYOUT(
1149    // ...
1150  ),
1151
1152  [_FN278_LAYER] = LAYOUT(
1153    // ...
1154  ),
1155
1156  [_FN279_LAYER] = LAYOUT(
1157    // ...
1158  ),
1159
1160  [_FN280_LAYER] = LAYOUT(
1161    // ...
1162  ),
1163
1164  [_FN281_LAYER] = LAYOUT(
1165    // ...
1166  ),
1167
1168  [_FN282_LAYER] = LAYOUT(
1169    // ...
1170  ),
1171
1172  [_FN283_LAYER] = LAYOUT(
1173    // ...
1174  ),
1175
1176  [_FN284_LAYER] = LAYOUT(
1177    // ...
1178  ),
1179
1180  [_FN285_LAYER] = LAYOUT(
1181    // ...
1182  ),
1183
1184  [_FN286_LAYER] = LAYOUT(
1185    // ...
1186  ),
1187
1188  [_FN287_LAYER] = LAYOUT(
1189    // ...
1190  ),
1191
1192  [_FN288_LAYER] = LAYOUT(
1193    // ...
1194  ),
1195
1196  [_FN289_LAYER] = LAYOUT(
1197    // ...
1198  ),
1199
1200  [_FN290_LAYER] = LAYOUT(
1201    // ...
1202  ),
1203
1204  [_FN291_LAYER] = LAYOUT(
1205    // ...
1206  ),
1207
1208  [_FN292_LAYER] = LAYOUT(
1209    // ...
1210  ),
1211
1212  [_FN293_LAYER] = LAYOUT(
1213    // ...
1214  ),
1215
1216  [_FN294_LAYER] = LAYOUT(
1217    // ...
1218  ),
1219
1220  [_FN295_LAYER] = LAYOUT(
1221    // ...
1222  ),
1223
1224  [_FN296_LAYER] = LAYOUT(
1225    // ...
1226  ),
1227
1228  [_FN297_LAYER] = LAYOUT(
1229    // ...
1230  ),
1231
1232  [_FN298_LAYER] = LAYOUT(
1233    // ...
1234  ),
1235
1236  [_FN299_LAYER] = LAYOUT(
1237    // ...
1238  ),
1239
1240  [_FN300_LAYER] = LAYOUT(
1241    // ...
1242  ),
1243
1244  [_FN301_LAYER] = LAYOUT(
1245    // ...
1246  ),
1247
1248  [_FN302_LAYER] = LAYOUT(
1249    // ...
1250  ),
1251
1252  [_FN303_LAYER] = LAYOUT(
1253    // ...
1254  ),
1255
1256  [_FN304_LAYER] = LAYOUT(
1257    // ...
1258  ),
1259
1260  [_FN305_LAYER] = LAYOUT(
1261    // ...
1262  ),
1263
1264  [_FN306_LAYER] = LAYOUT(
1265    // ...
1266  ),
1267
1268  [_FN307_LAYER] = LAYOUT(
1269    // ...
1270  ),
1271
1272  [_FN308_LAYER] = LAYOUT(
1273    // ...
1274  ),
1275
1276  [_FN309_LAYER] = LAYOUT(
1277    // ...
1278  ),
1279
1280  [_FN310_LAYER] = LAYOUT(
1281    // ...
1282  ),
1283
1284  [_FN311_LAYER] = LAYOUT(
1285    // ...
1286  ),
1287
1288  [_FN312_LAYER] = LAYOUT(
1289    // ...
1290  ),
1291
1292  [_FN313_LAYER] = LAYOUT(
1293    // ...
1294  ),
1295
1296  [_FN314_LAYER] = LAYOUT(
1297    // ...
1298  ),
1299
1300  [_FN315_LAYER] = LAYOUT(
1301    // ...
1302  ),
1303
1304  [_FN316_LAYER] = LAYOUT(
1305    // ...
1306  ),
1307
1308  [_FN317_LAYER] = LAYOUT(
1309    // ...
1310  ),
1311
1312  [_FN318_LAYER] = LAYOUT(
1313    // ...
1314  ),
1315
1316  [_FN319_LAYER] = LAYOUT(
1317    // ...
1318  ),
1319
1320  [_FN320_LAYER] = LAYOUT(
1321    // ...
1322  ),
1323
1324  [_FN321_LAYER] = LAYOUT(
1325    // ...
1326  ),
1327
1328  [_FN322_LAYER] = LAYOUT(
1329    // ...
1330  ),
1331
1332  [_FN323_LAYER] = LAYOUT(
1333    // ...
1334  ),
1335
1336  [_FN324_LAYER] = LAYOUT(
1337    // ...
1338  ),
1339
1340  [_FN325_LAYER] = LAYOUT(
1341    // ...
1342  ),
1343
1344  [_FN326_LAYER] = LAYOUT(
1345    // ...
1346  ),
1347
1348  [_FN327_LAYER] = LAYOUT(
1349    // ...
1350  ),
1351
1352  [_FN328_LAYER] = LAYOUT(
1353    // ...
1354  ),
1355
1356  [_FN329_LAYER] = LAYOUT(
1357    // ...
1358  ),
1359
1360  [_FN330_LAYER] = LAYOUT(
1361    // ...
1362  ),
1363
1364  [_FN331_LAYER] = LAYOUT(
1365    // ...
1366  ),
1367
1368  [_FN332_LAYER] = LAYOUT(
1369    // ...
1370  ),
1371
1372  [_FN333_LAYER] = LAYOUT(
1373    // ...
1374  ),
1375
1376  [_FN334_LAYER] = LAYOUT(
1377    // ...
1378  ),
1379
1380  [_FN335_LAYER] = LAYOUT(
1381    // ...
1382  ),
1383
1384  [_FN336_LAYER] = LAYOUT(
1385    // ...
1386  ),
1387
1388  [_FN337_LAYER] = LAYOUT(
1389    // ...
1390  ),
1391
1392  [_FN338_LAYER] = LAYOUT(
1393    // ...
1394  ),
1395
1396  [_FN339_LAYER] = LAYOUT(
1397    // ...
1398  ),
1399
1400  [_FN340_LAYER] = LAYOUT(
1401    // ...
1402  ),
1403
1404  [_FN341_LAYER] = LAYOUT(
1405    // ...
1406  ),
1407
1408  [_FN342_LAYER] = LAYOUT(
1409    // ...
1410  ),
1411
1412  [_FN343_LAYER] = LAYOUT(
1413    // ...
1414  ),
1415
1416
```

# Wait, but why?



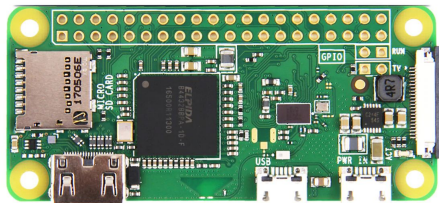
- Building the keyboard firmware with C made tweaks more difficult
  1. Update the keymap
  2. Compile the C code with the right toolchain
  3. Burn the firmware to the controller



- Nerves seemed so much fun



- Actively under development
- Bringing Elixir to the IoT world
- Ran on the Raspberry Pi Zero
  - 28 IO pins (!!!)





NERVES




elixir



# Nerves 101



- The Raspberry Pi Zero has an ARM architecture - **target**
  - My laptop has a x86 architecture - **host**
- The right **toolchain** lets me compile code on the **host** that runs on the **target**
  - Our Elixir application needs to be compiled to run on the **target** architecture
- The BEAM needs an Operating System on the **target** to run - **system**
  - Nerves provides a streamlined Linux (Buildroot-based) that will run on the **target** 
- The **firmware image** contains:
  - The Buildroot-based Linux
  - Our application compiled for the **target**
  - Everything needed for the **target** system to boot and start our application



Baby steps





# Baby steps (1/4)



- `mix nerves.new elixir_kee`
- `mix deps.get`
- `MIX_TARGET=rpi0 mix firmware && mix firmware.burn`

```
~/projs/personal/elixir_kee
```

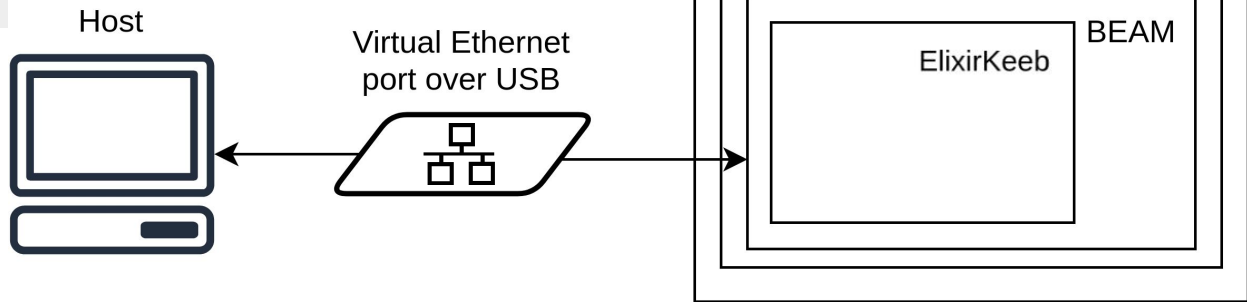
```
> ssh nerves.local
```

```
Interactive Elixir (1.8.1) - press Ctrl+C to exit (type h() ENTER for help)
```

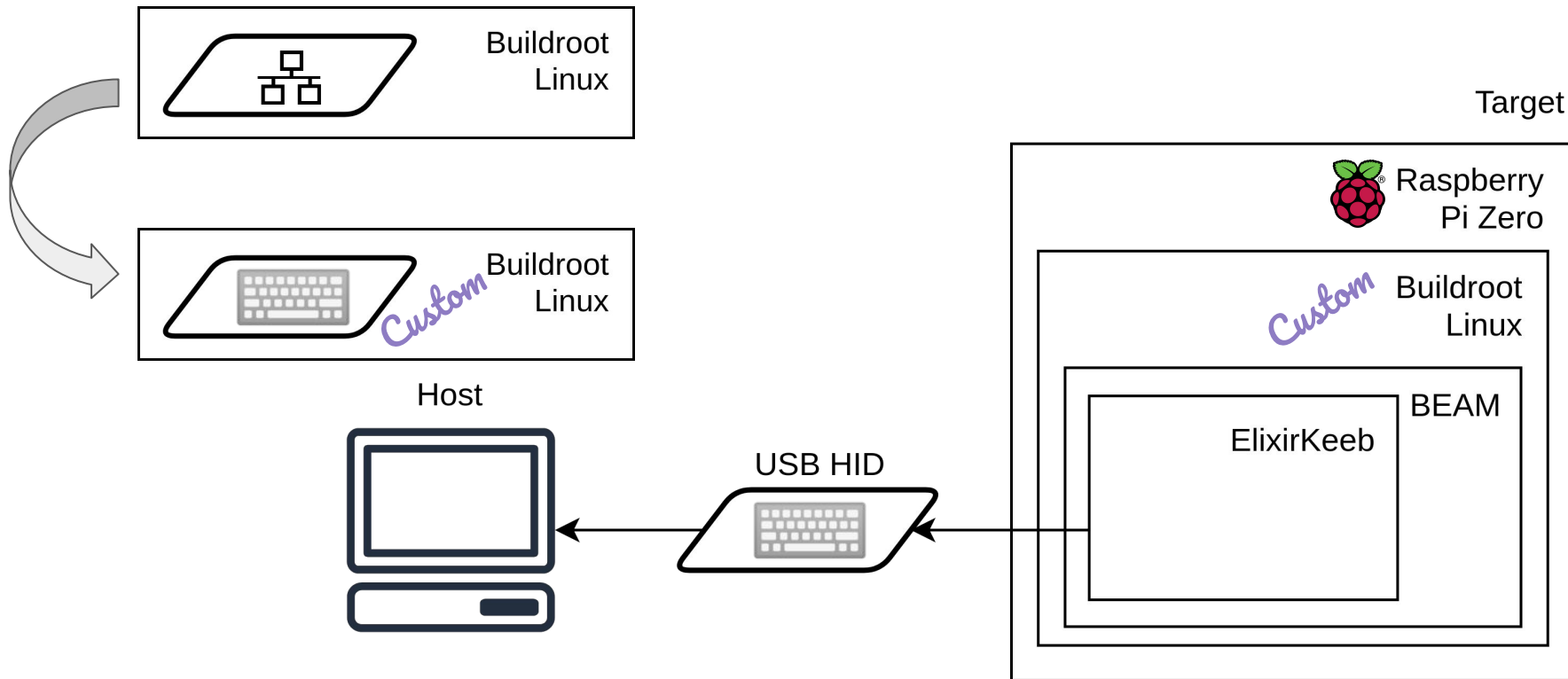
```
Toolshed imported. Run h(Toolshed) for more info
```

```
RingLogger is collecting log messages from Elixir and Linux. [...]
```

```
iex(elixir_kee@nerves.local)1> _
```



# Baby steps (2/4)



# Baby steps (3/4)



```
3 config :nerves_network,  
4   regulatory_domain: "EU"  
5  
6 config :nerves_network, :default,  
7   wlan0: [  
8     networks: [  
9       [  
10        ssid: System.get_env("NERVES_SSID"),  
11        psk: System.get_env("NERVES_PSK"),  
12        key_mgmt: :WPA-PSK  
13      ]  
14    ]  
15  ]
```

Host



*nerves\_network*



USB HID



Target



Raspberry  
Pi Zero

*Custom*

Buildroot  
Linux

BEAM

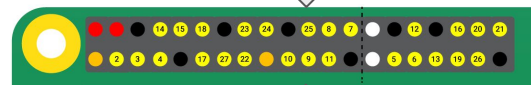
ElixirKeep

NERVES\_SSID=... NERVES\_PSK=... MIX\_TARGET=rpi0 mix firmware

# Baby steps (4/4)



*circuits\_gpio*



Target



Raspberry  
Pi Zero

*Custom*

Buildroot  
Linux

BEAM

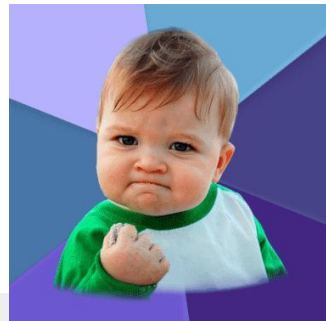
ElixirKeeb

USB HID

Host



# Baby steps

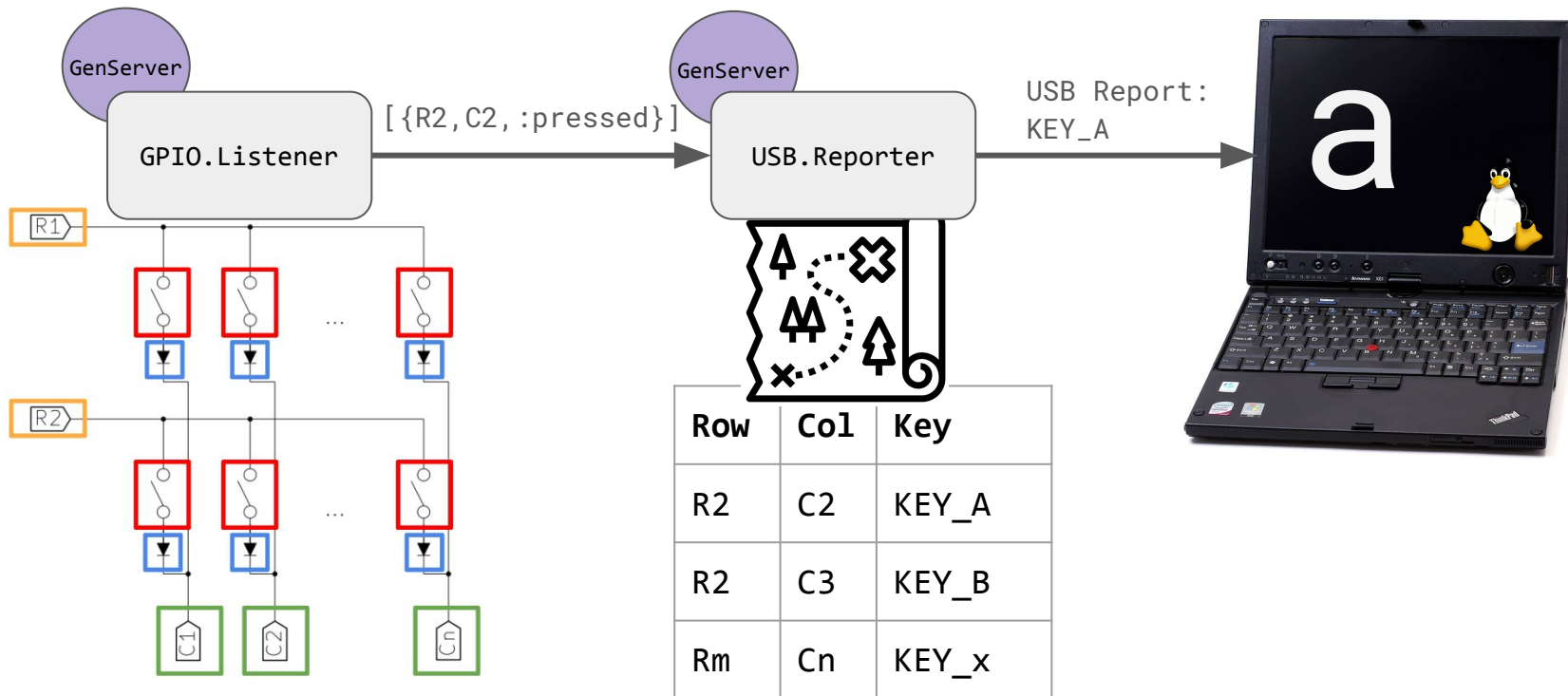


```
1 :elixir_keeB
2 |> Nerves.mix_new() # baby step 1/4
3 |> custom_build_root_system(fn system -> # baby step 2/4
4   system
5   |> disable_ethernet_port_over_usb()
6   |> enable_usb_hid()
7 end)
8 |> configure_nerves_network() # baby step 3/4
9 |> use_circuits_gpio() # baby step 4/4
  |> tada!()
```

# Baby steps



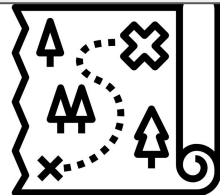
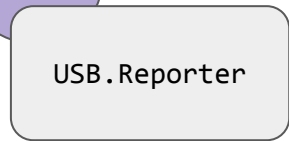
# A keyboard on top of the BEAM



# A keyboard on top of the BEAM



- DSL to define the keymap
- Multiple layers
  - Toggle or lock
- Macros
  - Pre-defined macros
  - Record macros on-the-fly



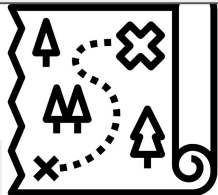


# A keyboard on top of the BEAM



GenServer

USB.Reporter



- DSL to define the keymap
- Multiple layers

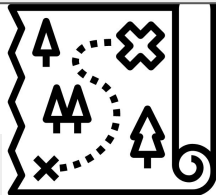
```
23 defmodule ElixirKeeb.CanonTypewriter.Layout do
24   use ElixirKeeb.Layout, matrix: ElixirKeeb.CanonTypewriter.Matrix
25
26   @layouts [
27     [ # layer 0
28       [:kc_escape, :kc_1, :kc_2, :kc_3, :kc_4, :kc_5, :kc_6, :kc_7, :kc_8, :kc_9, :kc_0, :kc_equal, :kc_slash, :kc_delete, :kc_bspace],
29       [toggle_layer(1), :kc_q, :kc_w, :kc_e, :kc_r, :kc_t, :kc_y, :kc_u, :kc_i, :kc_o, :kc_p, :kc_lbracket, :kc_rbracket, lock_layer(1)],
30       [:kc_lctrl, :kc_a, :kc_s, :kc_d, :kc_f, :kc_g, :kc_h, :kc_j, :kc_k, :kc_l, :kc_scolon, :kc_quote, :kc_lgui, :kc_enter],
31       [:kc_lshift, :kc_z, :kc_x, :kc_c, :kc_v, :kc_b, :kc_n, :kc_m, :kc_comma, :kc_dot, :kc_grave, :kc_lshift],
32       [:kc_lalt, :kc_space, :kc_ralt]
33     ],
34     [ # layer 1
35       [____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____],
36       [____, :kc_1, :kc_2, :kc_3, :kc_4, :kc_5, :kc_6, :kc_7, :kc_8, :kc_9, :kc_0, :kc_tab, :kc_bslash, _____],
37       [____, m(0), m(1), m(2), m(3), m(4), record(0), replay(0), _____, _____, _____, _____, _____, _____, _____],
38       [____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____, _____],
39       [____, :kc_x, _____]
40     ]
41   ]
42 end
```

# A keyboard on top of the BEAM



GenServer

USB.Reporter



## - Macros

- Pre-defined macros
- Record macros on-the-fly

```
23 defmodule ElixirKeeb.CanonTypewriter.Layout do
24   use ElixirKeeb.Layout, matrix: ElixirKeeb.CanonTypewriter.Matrix
25
26   @macros [
27     [:kc_a, :kc_b, :kc_c], # macro 0
28     "xyz" |> String.graphemes(), # macro 1
29     [
30       {"lshift", :pressed}, "e", {"lshift", :released}, "l", "i", "x", "i", "r", {"lshift", :pressed}, "1", {"lshift", :released},
31     ], # macro 2
32     "Hello, world!", # macro 3
33     &ElixirKeeb.CanonTypewriter.Macros.my_macro/1, # macro 4
34   ]
35
36   @layouts [
37     # layer 0 ...
38     [ # layer 1
39       # ...
40       [ :__, m(0), m(1), m(2), m(3), m(4), record(0), replay(0), :__, :__, :__, :__, :__, :__ ],
41       # ...
42     ]
43   ]
```

```
17 defmodule ElixirKeeb.CanonTypewriter.Macros do
18   def my_macro(state) do
19     {"Hello from macro function!", state}
20   end
21 end
```

# Pros & cons



**+** To iterate, I can **redefine the modules on-the-fly** inside an IEx shell on the Pi Zero

**+** Easily update the keyboard firmware

- Burn firmware to microSD card only **once**
- Further firmware changes can be **done via SSH**

**+** Being able to use a **highly expressive** language like Elixir to do embedded projects is pure bliss

**+** Listener and Reporter as **supervised** processes

**—** Raspberry Pi Zero is 10x **more expensive** than the Pro Micro controller

- ~10EUR vs ~1EUR

**—** **Slow** boot time

**—** More **power hungry**

- I don't think the ConfigFS device acts on the host suspend/resume instructions

**—** **Bigger** footprint

- Trickier to use in smaller keyboards

# Future work



- Implement a tap or toggle mechanism
  - Keep Control pressed → send Control
  - Tap Control → send Escape
- Proper power management
  - Honor suspend/resume instructions from the host
- Improve boot time
  - Look at streamlining even more the Buildroot system
- Support for other input devices
  - Trackball, mouse

# D E M O time

- Simple keyboard
  - Multiple layers
- Matrix scan latency and Matrix to USB latency
- Macros
  - Pre-defined macros
  - Recording macros
- Tweaking the layout on-the-fly

# Thank you!

Questions?



[amalbuquerque/elixir-keep](https://github.com/amalbuquerque/elixir-keep)

[amalbuquerque/elixir\\_keep\\_ui](https://github.com/amalbuquerque/elixir_keep_ui)

# Resources

- [amalbuquerque/elixir-keeb](#)
- [amalbuquerque/elixir\\_keeb\\_ui](#)
- [elixir-circuits/circuits\\_gpio: Use GPIOs from Elixir](#)
- [nerves-project/usb\\_gadget: Configure USB Gadget devices](#)
- [mindok/context: Charting and graphing library for Elixir](#)
- [Buildroot - Making Embedded Linux Easy](#)
- [Nerves Platform](#)
- [tmk/tmk\\_keyboard: Keyboard firmwares for Atmel AVR and Cortex-M](#)
- [qmk/qmk\\_firmware: Open-source keyboard firmware for Atmel AVR and Arm USB families](#)