

# Using Q editor macros to generate preprocessed Source Code without headers

Part 1: a basic implementation

# Steps involved

- “Instrument” source code with comments showing where the headers are
- Build from instrumented source and keep the preprocessed files
- Edit header lines out of preprocessed files
- (on Day 3) Generate a shadow directory tree

# Implementation (1/4)

- Create a new VCS branch to work in
- Change to the new branch
- Instrument the source (insert a ***cut here*** comment after the last `#include` in each file):

```
find . -name '*.c'|xargs q -oniu,cpp.qm^J^N2
```

# Implementation (2/4)

- Set up special build flags (*temps/* is an example):  
`CFLAGS='-P -C -dumpdir temps/ -save-temps' ./configure`  
*-P* suppresses line numbers; *-C* keeps comments  
(could use *-CC* to keep comments in macros)
- Clear the decks:  
`make -j$(($(nproc)+1)) clean;rm -rf temps; mkdir temps`

# Implementation (3/4)

- Do the build:

```
make -j$(($(nproc)+1))
```

- Remove unwanted assembler files:

```
rm temps/*.s
```

- You don't want binaries either:

```
make -j$(($(nproc)+1)) clean
```

# Implementation (4/4)

- Remove headers from .i files:

```
q -oiu, cpp.qm^J^N3 temps/* .i
```

If the last `#include` was `#ifdef`'d out, the ***cut here*** comment will be gone also. Need to edit out `#include` file contents manually later.

# Compare files

- Suggest using *tkdiff* to easily skip over uninteresting diffs
- cpp mangles white space a lot, so ignore whitespace and blank lines (e.g. **diff -Wb**)

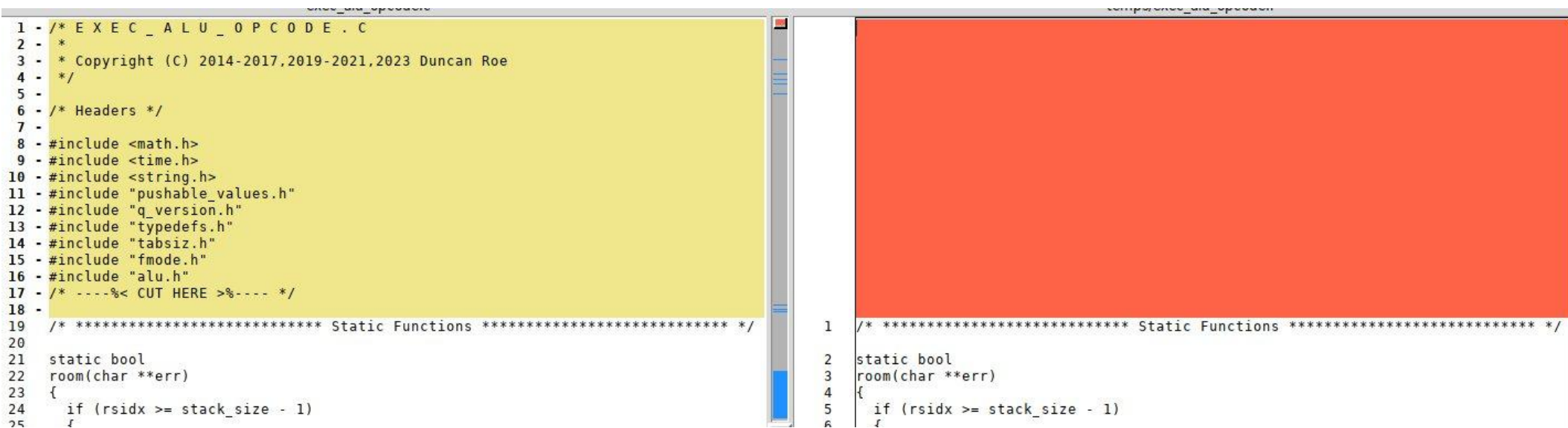
# Using Q editor macros to generate preprocessed Source Code without headers

Part 2: dealing with included .c files



# Where we got to last time

Delete everything up to end of last #include



```
1 - /* EXEC_ALU_OPCODE.C
2 - *
3 - * Copyright (C) 2014-2017,2019-2021,2023 Duncan Roe
4 - */
5 -
6 - /* Headers */
7 -
8 - #include <math.h>
9 - #include <time.h>
10 - #include <string.h>
11 - #include "pushable_values.h"
12 - #include "q_version.h"
13 - #include "typedefs.h"
14 - #include "tabsiz.h"
15 - #include "fmode.h"
16 - #include "alu.h"
17 - /* -----%< CUT HERE >%---- */
18 -
19 /* ***** Static Functions ***** */
20
21 static bool
22 room(char **err)
23 {
24     if (rsidx >= stack_size - 1)
25     {
```

```
1 /* ***** Static Functions ***** */
2
3 static bool
4 room(char **err)
5 {
6     if (rsidx >= stack_size - 1)
```

(actually deletes 1 line extra, fixed later)

# Keep lines before 1<sup>st</sup> #include

take 1: Add a “KEEP TO HERE” comment

```
1 /* EXEC_ALU_OPCODE.C
2 *
3 * Copyright (C) 2014-2017,2019-2021,2023 Duncan Roe
4 */
5
6 /* Headers */
7
8 /* ----%< KEEP TO HERE >%---- */
9 - #include <math.h>
10 - #include <time.h>
11 - #include <string.h>
12 - #include "pushable_values.h"
13 - #include "q_version.h"
14 - #include "typedefs.h"
15 - #include "tabsiz.h"
16 - #include "fmode.h"
17 - #include "alu.h"
18 /* ----%< CUT HERE >%---- */
19
20 /* ***** Static Functions ***** */
21
```

```
390 + #define __STDC_IEC_559__ 1
391 + #define __STDC_IEC_559_COMPLEX__ 1
392 + /* wchar_t uses Unicode 10.0.0. Version 10.0 of the Unicode Standard is
393 + synchronized with ISO/IEC 10646:2017, fifth edition, plus
394 + the following additions from Amendment 1 to the fifth edition:
395 + - 56 emoji characters
396 + - 285 hentaigana
397 + - 3 additional Zanabazar Square characters */
398 + #define __STDC_ISO_10646__ 201706L
399 /* EXEC_ALU_OPCODE.C
400 *
401 * Copyright (C) 2014-2017,2019-2021,2023 Duncan Roe
402 */
403
404 /* Headers */
405
406 /* ----%< KEEP TO HERE >%---- */
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
```

# Keep lines before 1<sup>st</sup> #include

## take 2: Insert a "CODE STARTS" comment

```
1 /* -----%< CODE STARTS >%---- */
2 /* EXEC_ALU_OPCODE.C
3 *
4 * Copyright (C) 2014-2017,2019-2021,2023 Duncan Roe
5 */
6
7 /* Headers */
8
9 /* -----%< KEEP TO HERE >%---- */
10 -#include <math.h>
11 -#include <time.h>
12 -#include <string.h>
13 -#include "pushable_values.h"
14 -#include "q_version.h"
15 -#include "typedefs.h"
16 -#include "tabsiz.h"
17 -#include "fmode.h"
18 -#include "alu.h"
19 /* -----%< CUT HERE >%---- */
20
21 /* ***** Static Functions ***** */
22
```

```
1 /* -----%< CODE STARTS >%---- */
2 /* EXEC_ALU_OPCODE.C
3 *
4 * Copyright (C) 2014-2017,2019-2021,2023 Duncan Roe
5 */
6
7 /* Headers */
8
9 /* -----%< KEEP TO HERE >%---- */
10
11
12
13
14
15
16
17
18
19
20 /* -----%< CUT HERE >%---- */
21
22 /* ***** Static Functions ***** */
```

looks good

# Original Experts Exchange question

TROUBLESHOOTING QUESTION



phoffric  
1/8/2023 - Sun

## Linux Ubuntu: Remove MACROS in C-Code by modifying makefile

I would like to modify a makefile in order to remove the macros in hard to read c-code in a liquid-dsp application. I tried this, but no luck:

<https://stackoverflow.com/questions/3742822/preprocessor-output>

For every .c file I would like to get a corresponding pre-processor file. One of the methods seemed to work except that the standard include header files were present, and the file had line numbers that I do not want.

slightly further on...

```
$ git clone https://github.com/jgaeddert/liquid-dsp.git
```

Did that. First problem: if the last `#include` is `#ifdef`'d out, so is immediately following "CUT HERE" comment.

Also discovered that some .c files `#include` other .c files

# Need better marker comments

- 3 components:
  - 1 Token*, to easily find next marker. Must not occur in original source
  - 2 Marker type*, single word (no spaces) e.g. **KEEP2HERE**
  - 3 Source Path*, originally so **diff** doesn't get false matches. Turns out to be useful for other reasons as well.

Example:

```
/* >%- - - - CODE_STARTS exec_alu_opcode.c */
```

# Invite Manual Edit

You see this

```
Determine what is last included line; enter "d ta - <that line>; key ^N5 (you should see  
a keep2here line followed by a cut_here line), enter q to continue
```

```
Type Q to continue macro 1507; FQ to abandon  
Noted screen dimensions 88 x 98
```

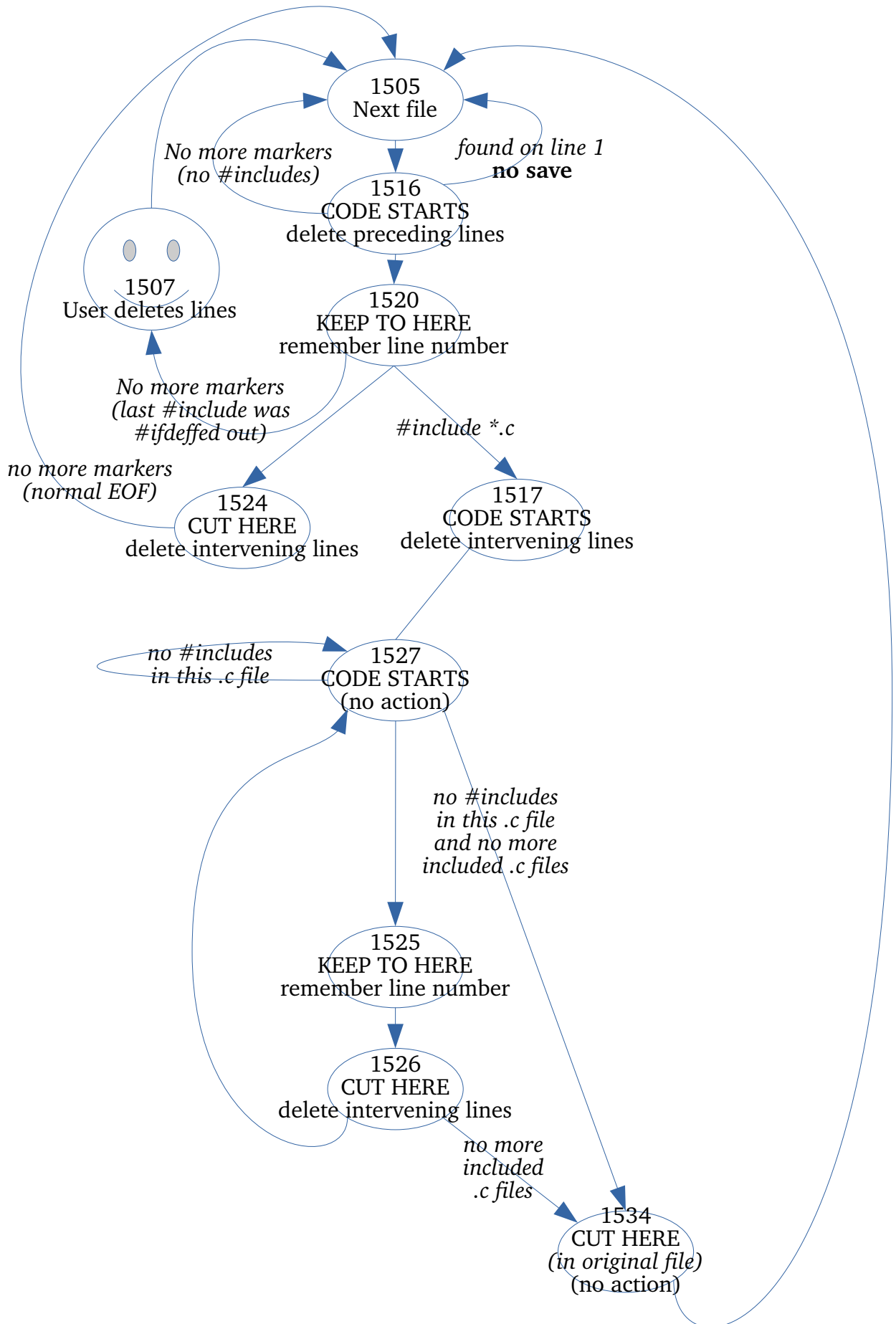
```
>
```

(quick live demo)

# Need more logic to deal with included `.c` files

- Detect already-processed files (for development)
- May get `CUT_HERE` or `CODE_STARTS` after `KEEP2HERE`
- And so on ...

# ^ N3:- process .i files state diagram (sort-of)





# Challenges from phoffric

- *All user .h files get their macros expanded to form .ih files.*
- *A mirror folder structure matching the liquid-dsp folder structure is defined and the .i and the .ih files are just named .c and .h (but having no macros).*

# Challenges from phoffric (continued)

- *Difficult (and negates some of the above goals): Create separate sibling folders if shared files (.h or proto) take on different content as a result of the macros being expanded differently as a result of some earlier macro expansion.*

# Using Q editor macros to generate preprocessed Source Code without headers

Part 3: create a shadow tree to diff against

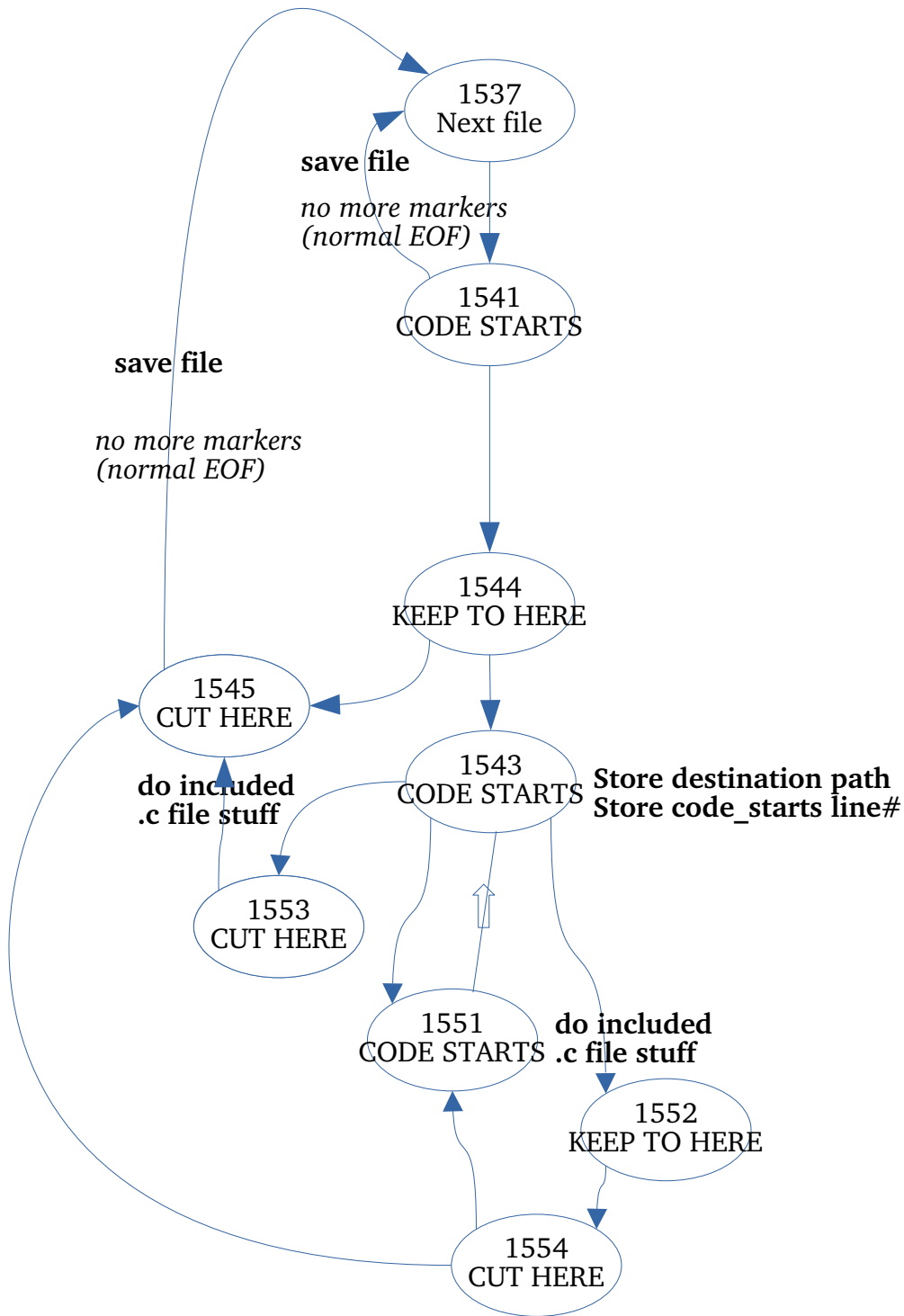
# Make a “Master Shadow” tree

- New macro `^N+` creates the shadow tree:

```
rm -rf shadow; q -oiu, cpp.qm^J^N+ temps/*.i
```

- Each included `.c` file (***proto.c file***) is written out to a file named `<proto.c file>.<basename including file .c>`. Proto.c file is removed from including file.
- With this naming scheme, can create a shadow tree with any combination of expanded proto.c files.

# ^ N+ :- make shadow directory from .i files



## Included .c file stuff

If included file already exists, rename it with backup suffix  
 Write out included file and delete it from including file  
 If we made a backup, compare and delete if same else ask user to take action

# It's all shell scripting from now on

- Eventual plan is to have proto.c files as symlinks so can tell from `ls -l` where they came from.
- But first, check for proto.c files that are only `#included` once. These can be safely **mv**'d into place.

# Commands to make *shadow1/*

*shadow1/* has a number of proto.c files in place because they never change (except perhaps with different ./configure options, but we're not going there). Will use *shadow1/* as a template to build individual shadow trees.

```
find shadow -type f ! -name '*.c'|rev|cut -d. -f2-|cut -d/ -f1|rev|sort -u >p1
cat p1|glb -v '\.c$'|rev|cut -d. -f2-|cut -d/ -f1|rev|sort -u>p1a
cat p1|glb '\.c$' >p2
cat p1a >>p2
(cd shadow; for i in $(cat ../p2);do if [ $(l $i|wc -l) -eq 1 ]; then echo $i;fi;done) >u
rm -rf shadow1
cp -a shadow shadow1
(cd shadow1; for i in $(cat ../u); do j=$(l $i); (cd $(dirname $j); mv -iv $(basename $j) $i) done)
```

# Create sample tree *sy1/*

- Function to create individual .proto.c files:

```
vm(){ (while [ $# -gt 0 ]; do (cd $(dirname $1);ln -s $(basename $1) $(  
  (basename $(echo ${1/.c//}|rev|cut -d/ -f2-|rev).c)); shift; done); }
```

VM() can be used with wildcards, e.g.

```
vm src/buffer/src/*.proto.c.bufferf
```

which picks up *cbuffer.proto.c*, *wdelay.proto.c* & *window.proto.c*.

- Function to compare created tree with original:

```
difcpp(){ find -D exec src -type d \( -name tests -o -name bench \) -prune -o  
-name '*.c' -exec diff -wB {} $1/{} \; 2>&1|glb -v "^DebugExec: process"|k; }
```



# Commands to populate *sy1/*

```
rm -rf sy1; cp -a shadow1 sy1; cd sy1
vm src/agc/src/agc.proto.c.agc_crcf
vm src/fft/src/*.proto.c.spgramcf
vm src/filter/src/*.c.filter_rrrf
vm src/framing/src/*sync.proto.c.*sync_cccf
vm src/buffer/src/*.proto.c.buffercf
vm src/equalization/src/*.proto.c.equalizer_rrrf
vm src/multichannel/src/firpfbch.proto.c.firpfbch_crcf
vm src/matrix/src/*.c.matrixc
vm src/math/src/poly.*.proto.c.polyf
vm src/quantization/src/quantizer.proto.c.quantizercf
vm src/matrix/src/smatrix.proto.c.smatrixi
vm src/vector/src/vector_add.proto.c.vectorcf_add.port
vm src/vector/src/vector_mul.proto.c.vectorf_mul.port
vm src/vector/src/vector_norm.proto.c.vectorcf_norm.port
vm src/vector/src
cd ..
```

# Compare w/original source

- There is a *DebugExec* line between each file

```
15:23:36$ difcpp syl
DebugExec: launching process (argc=4): 'diff' '-wB' 'src/sequence/src/msequence.c' 'sy1/src/sequence/src/msequence.c'
29,34d26
< #include <stdio.h>
< #include <stdlib.h>
< #include <string.h>
< #include <math.h>
<
< #include "liquid.internal.h"
36,39d27
<
< #define LIQUID_MIN_MSEQUENCE_M 2
< #define LIQUID_MAX_MSEQUENCE_M 15
<
74,76c61,62
<     if (_m > LIQUID_MAX_MSEQUENCE_M || _m < LIQUID_MIN_MSEQUENCE_M)
<         return liquid_error_config("msequence_create(), m not in range");
<
---
>     if (_m > 15 || _m < 2)
>         return liquid_error_config_f1("src/sequence/src/msequence.c", 75, "msequence_create(), m not in range");
109,110c88
<         return liquid_error_config("msequence_create_genpoly(), invalid generator polynomial: 0x%x", _g);
<
---
>         return liquid_error_config_f1("src/sequence/src/msequence.c", 109, "msequence_create_genpoly(), invalid generator polynomial: 0x%x", _g);
123,125c99,100
<     if (_m > LIQUID_MAX_MSEQUENCE_M || _m < LIQUID_MIN_MSEQUENCE_M)
<         return liquid_error_config("msequence_create(), m not in range");
<
---
>     if (_m > 15 || _m < 2)
>         return liquid_error_config_f1("src/sequence/src/msequence.c", 124, "msequence_create(), m not in range");
DebugExec: launching process (argc=4): 'diff' '-wB' 'src/sequence/src/bsequence.c' 'sy1/src/sequence/src/bsequence.c'
```

# Make sy2/ with 1 changed vm

```
1 ! rm -rf sy1; cp -a shadow1 sy1; cd sy1
2 vm src/agc/src/agc.proto.c.agc_crcf
3 vm src/fft/src/*.proto.c.spgramcf
4 vm src/filter/src/*.c.filter_rrrf
5 vm src/framing/src/*sync.proto.c.*sync_cccf
6 ! vm src/buffer/src/*.proto.c.buffercf
7 vm src/equalization/src/*.proto.c.equalizer_rrrf
8 vm src/multichannel/src/firpfbch.proto.c.firpfbch_crcf
9 vm src/matrix/src/*.c.matrixc
10 vm src/math/src/poly/*.proto.c.polyf
11 vm src/quantization/src/quantizer.proto.c.quantizercf
12 vm src/matrix/src/smatrix.proto.c.smatrixi
13 vm src/vector/src/vector_add.proto.c.vectorcf_add.port
14 vm src/vector/src/vector_mul.proto.c.vectorcf_mul.port
15 vm src/vector/src/vector_norm.proto.c.vectorcf_norm.port
16 vm src/vector/src/vector_trig.proto.c.vectorcf_trig.port
17 cd ..
```

```
1 ! rm -rf sy2; cp -a shadow1 sy2; cd sy2
2 vm src/agc/src/agc.proto.c.agc_crcf
3 vm src/fft/src/*.proto.c.spgramcf
4 vm src/filter/src/*.c.filter_rrrf
5 vm src/framing/src/*sync.proto.c.*sync_cccf
6 ! vm src/buffer/src/*.proto.c.bufferf
7 vm src/equalization/src/*.proto.c.equalizer_rrrf
8 vm src/multichannel/src/firpfbch.proto.c.firpfbch_crcf
9 vm src/matrix/src/*.c.matrixc
10 vm src/math/src/poly/*.proto.c.polyf
11 vm src/quantization/src/quantizer.proto.c.quantizercf
12 vm src/matrix/src/smatrix.proto.c.smatrixi
13 vm src/vector/src/vector_add.proto.c.vectorcf_add.port
14 vm src/vector/src/vector_mul.proto.c.vectorcf_mul.port
15 vm src/vector/src/vector_norm.proto.c.vectorcf_norm.port
16 vm src/vector/src/vector_trig.proto.c.vectorcf_trig.port
17 cd ..
```

# Compare sy1/ and sy2/

- *cbuffer.proto.c*, *wdelay.proto.c* & *window.proto.c* are changed.

```
11:39:14$ diffdir -wB sy1 sy2
diff -wB sy1/src/buffer/src/cbuffer.proto.c sy2/src/buffer/src/cbuffer.proto.c
29c29
< int cbuffercf_linearize(cbuffercf _q);
---
> int cbufferf_linearize(cbufferf _q);
31c31
< struct cbuffercf_s {
---
> struct cbufferf_s {
33c33
<     float _Complex * v;
---
>     float * v;
48c48
< cbuffercf cbuffercf_create(unsigned int _max_size)
---
> cbufferf cbufferf_create(unsigned int _max_size)
51c51
<     cbuffercf q = cbuffercf_create_max(_max_size, _max_size);
---
>     cbufferf q = cbufferf_create_max(_max_size, _max_size);
58c58
< cbuffercf cbuffercf_create_max(unsigned int _max_size,
---
> cbufferf cbufferf create_max(unsigned int _max size,
```

# Resources

- [cpp.qm](#) (Q Macro file)
- [liquid-dsp](#) (the project with included .c files)

```
glb(){ grep -E --line-buffered "$@"; }
l(){ find . -depth \( -name "*" "$1" "*" -o -name ".*" "$1" "*" \) -print; }
k(){ less "$@"; }
diffdir(){ opts=""; while [ $(echo -- "$1"|cut -c4) = '-' ]; do opts="$opts $1"; shift; done;
  if [ -z "$1" -o -z "$2" ]; then echo "Usage:- $(basename "$0") [diff opts] <dir1> <dir to be compared to
dir1>"; return 1; fi
  find "$1" -type d -exec sh -c "diff $opts \"{}\" \"\$(echo \"{}\" | sed s?^\ "$1\"?\"$2\"?)\" \" \" \; 2>&1|
glb -v '^Common subdirectories: ' ; }
```