Pairings pulled from map:

(0, 25, 'cpu0', 'cache0', 'wt')

(1, 29, 'cpu1', 'cache1', 'wt')

(2, 26, 'cpu0', 'cache0', 'rd')

(3, 32, 'cpu1', 'cache1', 'rd')

(4, 35, 'gfx', 'membus', 'upwt')

(5, 36, 'audio', 'membus', 'upwt')

(6, 37, 'usb', 'membus', 'uprd')

(7, 11, 'cache1', 'cache0', 'wt')

(8, 12, 'cache0', 'cache1', 'wt')

(9, 18, 'membus', 'cache0', 'wt')

(10, 27, 'membus', 'cache0', 'rd')

(13, 24, 'cache0', 'membus', 'wt')

(14, 28, 'cache1', 'membus', 'wt')

(15, 23, 'membus', 'mem', 'rd')

(16, 20, 'cache1', 'cache0', 'rd')

(17, 19, 'cache0', 'cache1', 'rd')

(21, 30, 'cache0', 'membus', 'rd')

(22, 31, 'cache1', 'membus', 'rd')

(33, 34, 'membus', 'mem', 'wt')

(38, 40, 'gfx', 'membus', 'uprd')

(39, 41, 'audio', 'membus', 'uprd')

(42, 43, 'uart', 'membus', 'uprd')

(44, 45, 'membus', 'usb', 'wt')

(46, 47, 'membus', 'uart', 'rd')

(48, 49, 'membus', 'audio', 'rd')

(50, 51, 'membus', 'uart', 'wt')

(52, 53, 'membus', 'usb', 'rd')

(54, 55, 'membus', 'gfx', 'rd')

(56, 57, 'membus', 'gfx', 'wt')

(58, 59, 'membus', 'audio', 'wt')

More specifically:

7 : cache1:cache0:wt:req 11 : cache0:cache1:wt:resp

8 : cache0:cache1:wt:req 12 : cache1:cache0:wt:resp

9 : membus:cache0:wt:req 18 : cache0:membus:wt:resp

17 : cache0:cache1:rd:req 19 : cache1:cache0:rd:resp

16 : cache1:cache0:rd:req 20 : cache0:cache1:rd:resp

15 : membus:mem:rd:req 23 : mem:membus:rd:resp

13 : cache0:membus:wt:req 24 : membus:cache0:wt:resp

10 : membus:cache0:rd:req 27 : cache0:membus:rd:resp

14 : cache1:membus:wt:req 28 : membus:cache1:wt:resp

21 : cache0:membus:rd:req 30 : membus:cache0:rd:resp

22 : cache1:membus:rd:req 31 : membus:cache1:rd:resp

33 : membus:mem:wt:req 34 : mem:membus:wt:resp

44 : membus:usb:wt:req 45 : usb:membus:wt:resp

46 : membus:uart:rd:req 47 : uart:membus:rd:resp

48 : membus:audio:rd:req 49 : audio:membus:rd:resp

50 : membus:uart:wt:req 51 : uart:membus:wt:resp

52 : membus:usb:rd:req 53 : usb:membus:rd:resp

54 : membus:gfx:rd:req 55 : gfx:membus:rd:resp

56 : membus:gfx:wt:req 57 : gfx:membus:wt:resp

58 : membus:audio:wt:req 59 : audio:membus:wt:resp

#

0 : cpu0:cache0:wt:req25 : cache0:cpu0:wt:resp

2 : cpu0:cache0:rd:req 26 : cache0:cpu0:rd:resp

1 : cpu1:cache1:wt:req29 : cache1:cpu1:wt:resp

3 : cpu1:cache1:rd:req32 : cache1:cpu1:rd:resp

4 : gfx:membus:upwt:req 35 : membus:gfx:upwt:resp

5 : audio:membus:upwt:req 36 : membus:audio:upwt:resp

6 : usb:membus:uprd:req 37 : membus:usb:uprd:resp

38 : gfx:membus:uprd:req 40 : membus:gfx:uprd:resp

39 : audio:membus:uprd:req 41 : membus:audio:uprd:resp

42 : uart:membus:uprd:req 43 : membus:uart:uprd:resp

Code

```python
#Read in the msg file and extract the pairings in the data flow.
#1 and 2 are a pair if: src1 = dest2, dest1=src2. cmd1=cmd2. if type1 is resp, type2 must be req and vice versa.
def extract_pairs_from_msg_file(file_path):
    pair_indices = {}
    with open(file_path, 'r') as file:
        for line in file:
            line = line.strip()
            if line.startswith('#'):
                continue  # Ignore comments
            elif line:
                parts = [part.strip() for part in line.split(':')]
                if len(parts) == 5:
                    index, src, dest, cmd, type_ = parts
                    key = (src, dest, cmd, type_)
                    if key not in pair_indices:
                        pair_indices[key] = []
                    pair_indices[key].append(int(index))

    pairs = set()  # Avoid duplicates
    for key, indices in pair_indices.items():
        src, dest, cmd, type_ = key
        switched_key = (dest, src, cmd, "req" if type_ == "resp" else "resp")
        if switched_key in pair_indices:
            switched_indices = pair_indices[switched_key]
            for req_index in indices:
                for resp_index in switched_indices:
                    # Check if a pair with the same first two elements already exists in pairs
                    if not any(p[0] == min(req_index, resp_index) and p[1] == max(req_index, resp_index) for p in pairs):
                        # Include the src/dest for use later when outputting file names
                        pair = (min(req_index, resp_index), max(req_index, resp_index), src, dest, cmd)
                        pairs.add(pair)

    return sorted(pairs)  # Return sorted pairs
```

Trace List extracted from trace-small-5.txt

TRACE LIST trace-small-5.txt

[3, 16, 20, 22, 52, 53, 31, 32, 0, 8, 12, 13, 54, 55, 24, 25, 2, 0, 17, 19, 8, 12, 13, 15, 21, 52, 23, 53, 30, 24, 26, 25, 1, 0, 8, 29, 12, 13, 48, 49, 24, 25, 3, 1, 16, 7, 20, 11, 22, 29, 15, 23, 31, 32, 2, 17, 19, 21, 48, 49, 30, 26, 1, 7, 1, 7, 11, 11, 14, 15, 23, 28, 29, 14, 46, 47, 28, 29, 3, 16, 2, 17, 20, 19, 32, 21, 52, 53, 30, 26, 2, 17, 1, 19, 21, 7, 15, 11, 29, 23, 30, 26, 2, 17, 19, 26, 1, 7, 11, 14, 54, 55, 28, 29, 0, 8, 12, 13, 46, 2, 47, 24, 17, 19, 21, 25, 52, 53, 30, 26, 2, 17, 19, 21, 54, 55, 30, 26, 2, 17, 0, 8, 12, 13, 19, 21, 15, 15, 23, 30, 23, 26, 24, 25, 0, 8, 12, 25, 3, 16, 20, 32, 2, 17, 19, 21, 15, 23, 30, 26, 2, 17, 19, 21, 46, 47, 30, 26, 0, 8, 12, 25, 2, 17, 19, 21, 48, 49, 30, 26, 1, 7, 11, 14, 48, 49, 28, 29, 2, 17, 19, 21, 48, 49, 30, 26, 1, 7, 11, 14, 52, 53, 28, 29, 1, 7, 11, 14, 0, 8, 15, 12, 23, 13, 28, 29, 48, 49, 24, 25, 0, 8, 12, 13, 54, 55, 24, 25, 2, 17, 19, 21, 48, 49, 30, 26, 0, 8, 12, 13, 52, 53, 24, 25, 3, 0, 16, 20, 8, 22, 12, 15, 25, 23, 31, 32, 2, 3, 16, 20, 22, 52, 17, 53, 19, 21,

31, 32, 46, 47, 30, 26, 3, 16, 20, 22, 54, 55, 31, 32, 3, 16, 20, 32, 3, 1, 16, 20, 7, 22, 48, 49, 31, 11, 14, 32, 15, 23, 28, 29, 2, 17, 0, 8, 12, 13, 19, 21, 46, 15, 23, 47, 24, 30, 25, 26, 2, 17, 19, 21, 15, 23, 30, 26, 2, 26, 0, 8, 12, 3, 16, 25, 20, 22, 46, 47, 31, 32, 2, 2, 17, 17, 19, 19, 21, 52, 26, 53, 30, 26, 1, 7, 11, 14, 15, 23, 28, 29, 1, 7, 0, 11, 8, 12, 14, 13, 54, 54, 55, 55, 28, 24, 29, 25, 0, 25, 1, 7, 11, 14, 15, 23, 28, 29, 3, 0, 8, 16, 12, 25, 20, 22, 46, 47, 31, 32, 0, 8, 12, 13, 46, 47, 24, 25, 1, 7, 11, 14, 46, 47, 28, 29, 0, 25, 0, 8, 12, 13, 52, 53, 24, 25, 3, 16, 20, 22, 52, 53, 31, 32, 2, 17, 19, 0, 21, 8, 12, 46, 13, 15, 47, 30, 23, 26, 24, 25, 0, 8, 12, 13, 52, 53, 24, 25, 3, 32, 1, 7, 11, 14, 54, 55, 28, 29, 3, 32, 1, 2, 7, 11, 14, 48, 49, 17, 19, 21, 48, 28, 49, 29, 30, 26, 3, 16, 20, 32, 2, 17, 19, 21, 54, 55, 30, 26, 0, 3, 16, 8, 12, 20, 22, 13, 54, 55, 31, 46, 32, 47, 24, 25, 0, 25, 1, 7, 11, 14, 46, 47, 28, 29, 3, 32, 2, 26, 3, 16, 20, 22, 54, 55, 31, 32, 1, 7, 11, 14, 48, 49, 28, 29, 3, 16, 20, 22, 48, 49, 31, 32, 3, 16, 0, 20, 22, 52, 8, 53, 12, 31, 13, 32, 48, 49, 24, 25, 3, 0, 16, 20, 8, 22, 54, 12, 55, 13, 48, 31, 49, 32, 24, 25, 1, 7, 1, 7, 11, 11, 14, 14, 48, 46, 49, 28, 29, 47, 28, 29, 0, 8, 12, 13, 54, 55, 24, 25, 0, 8, 12, 13, 46, 47, 24, 25, 1, 7, 11, 14, 54, 55, 28, 29, 1, 7, 11, 14, 46, 47, 28, 29, 3, 1, 16, 20, 32, 7, 11, 14, 48, 49, 28, 29, 1, 7, 11, 29, 0, 25, 3, 16, 20, 22, 52, 53, 31, 32, 3, 16, 20, 22, 46, 47, 31, 32, 0, 8, 12, 13, 52, 53, 24, 25, 2, 2, 26, 17, 19, 21, 46, 47, 30, 26, 0, 2, 17, 8, 12, 13, 54, 55, 24, 19, 25, 26, 1, 7, 3, 11, 32, 14, 52, 53, 28, 29, 3, 16, 20, 22, 54, 55, 31, 32, 0, 25, 1, 7, 11, 14, 52, 53, 28, 29, 1, 7, 0, 11, 14, 54, 8, 55, 28, 12, 29, 13, 52, 53, 24, 25, 2, 17, 19, 21, 2, 26, 54, 55, 30, 26, 3, 32, 0, 2, 17, 19, 8, 21, 12, 52, 13, 53, 30, 26, 15, 23, 24, 25, 2, 1, 7, 11, 17, 14, 19, 21, 52, 46, 53, 47, 30, 26, 28, 29, 2, 17, 19, 26, 1, 0, 7, 8, 11, 12, 14, 52, 13, 15, 23, 24, 25, 53, 28, 29, 3, 16, 20, 22, 46, 47, 31, 32, 1, 7, 11, 29, 2, 26, 3, 16, 20, 22, 15, 23, 31, 32, 3, 2, 16, 20, 17, 22, 46, 19, 47, 31, 26, 32, 3, 0, 8, 12, 13, 16, 20, 22, 48, 48, 49, 49, 24, 25, 31, 32, 1, 7, 3, 11, 29, 16, 20, 22, 48, 49, 31, 32, 3, 16, 20, 22, 48, 49, 31, 32, 1, 3, 16, 29, 20, 22, 15, 23, 31, 32, 3, 16, 20, 22, 15, 23, 31, 32, 1, 29, 1, 29, 2, 17, 19, 21, 54, 55, 30, 26, 1, 29, 2, 17, 19, 21, 54, 55, 30, 26]

Code

```
#Read in the trace file into a list
def read_trace_file(file_path):
    numbers = []
    with open(file_path, 'r') as file:
        for line in file:
            parts = line.strip().split()
            for part in parts:
                if part == '-1':
                    continue
                elif part == '-2':
                    return numbers
                else:
                    numbers.append(int(part))
    return numbers
```

Extracted sequences for trace-small-5.txt

[0, 25, 0, 25]

[1, 29, 1, 29, 1, 1, 29, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 1, 29, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29, 1, 29]

[2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 2, 26, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 2, 26, 26, 2, 26, 2, 2, 26, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26, 2, 26]

[3, 32, 3, 32]

[7, 11, 7, 7, 11, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 7, 11, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11, 7, 11]

[8, 12, 8, 12]

[13, 24, 13, 24]

[14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 14, 28, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28, 14, 28]

[15, 23, 15, 23, 15, 23, 15, 23, 15, 15, 23, 23, 15, 23]

[16, 20, 16, 20]

[17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 17, 19, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19, 17, 19]

[21, 30, 21, 30]

[22, 31, 22, 31]

[46, 47, 46, 47]

[48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 48, 49, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49, 48, 49]

[52, 53, 52, 53]

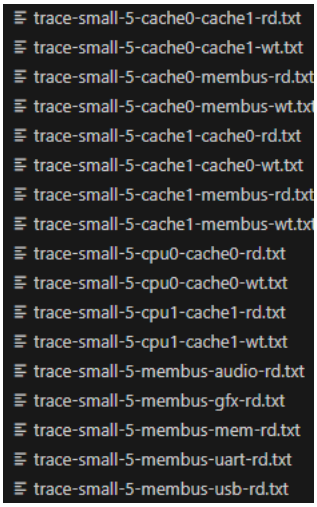
[54, 55, 54, 55, 54, 55, 54, 55, 54, 55, 54, 54, 55, 55, 54, 55]

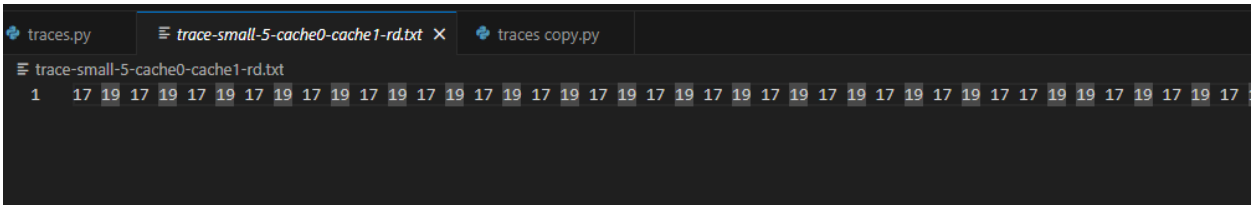
Code

```python
#extract the sequences and output into file names based on the src/dest
def extract_sequences(trace, pairs, name):
    pair_indices = {pair: [] for pair in pairs}
    for i, num in enumerate(trace):
        for pair in pairs:
            if num in pair:
                pair_indices[pair].append(i)

    sequences = []
    for pair in pairs:
        src_index, dest_index = pair[0], pair[1]
        src, dest, cmd = pair[2], pair[3], pair[4]
        filename = f"{name}-{src}-{dest}-{cmd}.txt"
        subsequence = [trace[i] for i in pair_indices.get(pair, [])]  # Get subsequence for the pair
        if subsequence:
            with open(filename, "w") as file:
                file.write(" ".join(map(str, subsequence)))
            sequences.append(subsequence)

    return sequences
```

(only iterate through trace list once)

File output





(17, 19, 'cache0', 'cache1', 'rd')