# Running the ELMO Examples

Two examples are given with the release of ELMO. One of these performs a fixed vs random test on an implementation of AES from the MBed TLS suite. The other performs a fixed vs random test on one round of a masked implementation of AES analysis using the mask flow analysis.

These examples use the libopencm3 library and the arm gcc toolchain to compile the source code into a Thumb binary to be run by ELMO. The libopencm3 library will need to be linked in the Makefile.rules file for it to compile.

# Example 1: AES DPA Trace Generation

**Example found in Examples/DPATraces/MBedAES. To run this example, the defines FIXEDVSRANDOM and MASKFLOW should be excluded from the ELMO build.**

This example generates 200 traces for an implementation of AES taken from the Mbed TLS suite. The inputs are random and the key is fixed to 0,1,2,3,…,15. The randdata() function will generate a byte of random data and store it in the input address given. The file where this data is stored is specified in elmodefines.h but by default is in the output folder listed as randdata.txt. The output ciphertexts are also printed in a similar way using the printbyte() function.

The example should be compiled using the make file. Once the binary has been generated this can run by ELMO using the command:

**./elmo  Examples/DPATraces/MBedAES/MBedAES.bin**

This should generate the 200 traces. The inputs (stored in randdata.txt) can be used along with these traces to perform a DPA style attack.

The assembly instructions executed for the program can be found in the folder output/asmtraces and the nonprolifed indexes in the folder output/asmtraces.

# Example 2: AES Fixed vs Random

**Example found in Examples/FixedvsRandom/MBedAES. To run this example, the define FIXEDVSRANDOM should be included and MASKFLOW should be excluded from the ELMO build.**