Logic synthesis for software circuits

Pierre-Evariste Dagand (dagand@irif.fr)

I am looking for a function

```
int sbox3(int x)
```

such that

```
sbox3(0)  = 8   sbox3(4)  = 3
sbox3(1)  = 6   sbox3(5)  = 12
sbox3(2)  = 7   sbox3(6)  = 10
sbox3(3)  = 9   sbox3(7)  = 15

sbox3(8)  = 13  sbox3(12) = 0
sbox3(9)  = 1   sbox3(13) = 11
sbox3(10) = 14  sbox3(14) = 5
sbox3(11) = 4   sbox3(15) = 2
```

with the added constraint that this function is forbidden to read from main memory. Typically, `x` would be a secret (e.g., a cryptographic key) and an access to memory depending on the value of `x` would reveal sensitive information that can be used by an attacker to retrieve the secret.

To side-step this issue, cryptographers usually resort to writing what is essentially a combinational circuit: the function `sbox3` is implemented by a branch-free, memory-less sequence of logical operations (and, or, xor, negation) that implements the truth table given above.

This project aims at implementing a brute-force search algorithm producing the most efficient implementation of such a truth table, for a given computer architecture & instruction set (eg., supporting various SIMD instruction sets) and at a predictable compute budget. While doing this, we will fortunately be standing on the shoulders of giants:

- "Speeding up Serpent", Dag Arne Osvik (AES Candidate Conference 2000)
- "Optimizing bitslice DES S-box expressions"[1], OpenWall/John the Ripper

[1] https://openwall.info/wiki/sbox-opt/des