3. Šifriranje podatkov

Naredili bomo sintezo algoritma za šifriranje podatkov **Tiny Encryption Algorithm** in primerjali čas šifriranja bloka podatkov v strojni in v programski opremi. https://en.wikipedia.org/wiki/Tiny_Encryption_Algorithm

3.1 Algoritem za šifriranje v programski opremi

- Odpri projekt *meritev.sdk* v programskem orodju Xilinx SDK, ki vsebuje testni program za razvojno ploščo ZedBoard.
- Priklopi razvojno ploščo in v spodnjem zavihku **SDK Terminal** s klikom na + odpri serijski terminal, izberi vrata COM in nastavi Baud Rate: 115200.
- V oknu Project Explorer klikni na aplikacijo (test) in jo poženi na razvojni plošči **Run > Run As > Launch on Hardware**.
- Program izpiše na terminal: Hello World! in čas, ki ga porabi za izvedbo stavka za izpis.

```
#include <stdio.h>
#include "platform.h"
#include "xparameters.h"
#include "xtime_l.h"

void print(char *str);

int main()
{
    XTime tStart, tEnd;

    init_platform();

    XTime_GetTime(&tStart);
      print("Hello World\n\r");
    XTime_GetTime(&tEnd);

    printf("Output took %llu clock cycles.\n", 2*(tEnd - tStart));
    printf("Output took %.2f us.\n", 1.0 * (tEnd - tStart) / (COUNTS_PER_SECOND/1000000));

    cleanup_platform();
    return 0;
}
```

- V program helloworld.c dodaj funkcijo za šifriranje podatkov iz datoteke **encrypt.cpp** in kodo, ki izvede en blok šifriranja ter izmeri čas izvajanja algoritma

- Zapiši, koliko časa potrebuje procesor ARM, ki izvaja kodo s frekvenco 666 MHz za šifriranje bloka podatkov:

število iteracij ITER	**število ciklov**	**čas (µs)**
8		
32		

3.2 Sinteza vezja in optimizacija

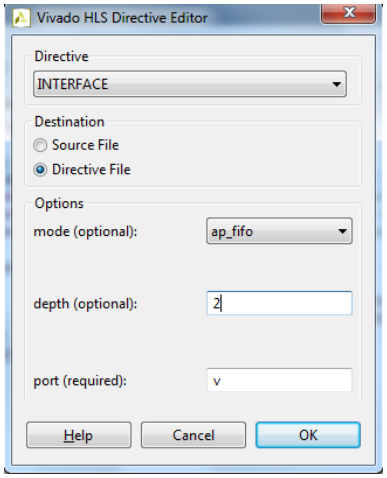
- Zaženi program Vivado HLS in izberi **Create New Project**, ki odpre pomočnika (wizzard) za izdelavo projekta. Določi ime projekta, npr. tea in lokacijo na disku
- V naslednjem oknu določi ime glavne funkcije (encrypt) in dodaj datoteko encrypt.cpp
- V tretjem oknu določi periodo ure 9 ns, kar ustreza frekvenci 111 MHz, in izberi razvojno ploščo ZedBoard.
- Poskusi izvedbe sinteze vezja ne uspe, ker program ne ve, koliko podatkov je potrebno prenesti preko kazalcev za posamezen izračun.

```
#define ITER 8
typedef unsigned long uint32_t;

void encrypt (uint32_t *v, const uint32_t k[4], uint32_t *w) {
       uint32_t v0=v[0], v1=v[1], sum=0, i;
       uint32_t delta=0x9e3779b9;
       uint32_t k0=k[0], k1=k[1], k2=k[2], k3=k[3];

encrypt_loop: for (i=0; i < ITER; i++) {
        sum += delta;
        v0 += ((v1<<4) + k0) ^ (v1 + sum) ^ ((v1>>5) + k1);
        v1 += ((v0<<4) + k2) ^ (v0 + sum) ^ ((v0>>5) + k3);
    }
    w[0]=v0; w[1]=v1;

}
```



- Na spremenljivki **v** in **w** dodaj direktivo HLS INTERFACE, določi vrsto vmesnika: ap_fifo in globino pomnilnika 2 besedi
- Na spremenljivko **k** pa dodaj direktivo HLS INTERFACE z vmesnikom: ap_memory in globino pomnilnika 4 besede
- Naredi sintezo vezja in zapiši rezultate v tabelo:

direktive	latenca / interval	DSP48E	FF	LUT
HLS INTERFACE				
HLS UNROLL				
HLS PIPELINE				

- Dodaj še direktivo za razvijanje zanke: HLS UNROLL ter izvedbo funkcije s cevljenjem: HLS PIPELINE ter vsakokrat posebej zabeleži rezultate visokonivojske sinteze vezja