# Пердисловие

Данную лабу можно выполнить на основе любого Block design, поэтому я буду раскрывать её с помощью шестой работы. Не придавайте этому значения, и делайте на удобной. Предлагаю вам два варианта:

Для понимания процесса – параллельно спидрану читайте методу

Для сдачи – просто следуйте инструкции ниже

# Спидран

1. Запустите SDK и создайте New Application со следующими параметрами:
   * **Name:** <любое>
   * **OS Platform:** standalone *<- ahah, just like me…*
   * **Language:** C

<Click Next>

* + **Templates:** Hello World

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

рис. 1 Новый приложение

1. Теперь перейдите в папку src внутри него, и откройте **helloworld.c**, после чего сотрите код, и вставьте тот, что я привожу в конце спидрана. Нажмите Ctrl+S, чтобы автоматически выполнить build.

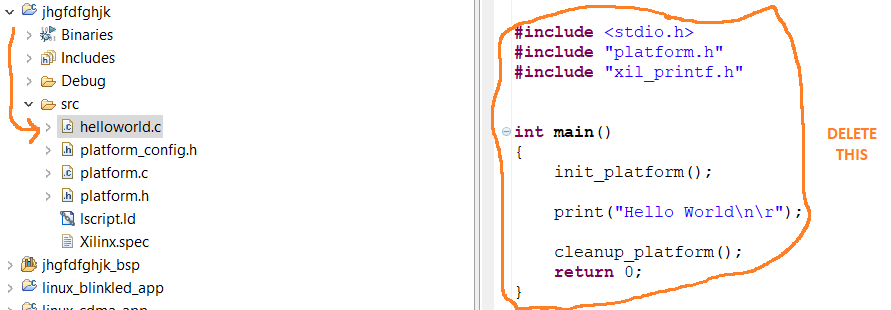
****

рис. 2 Как это было

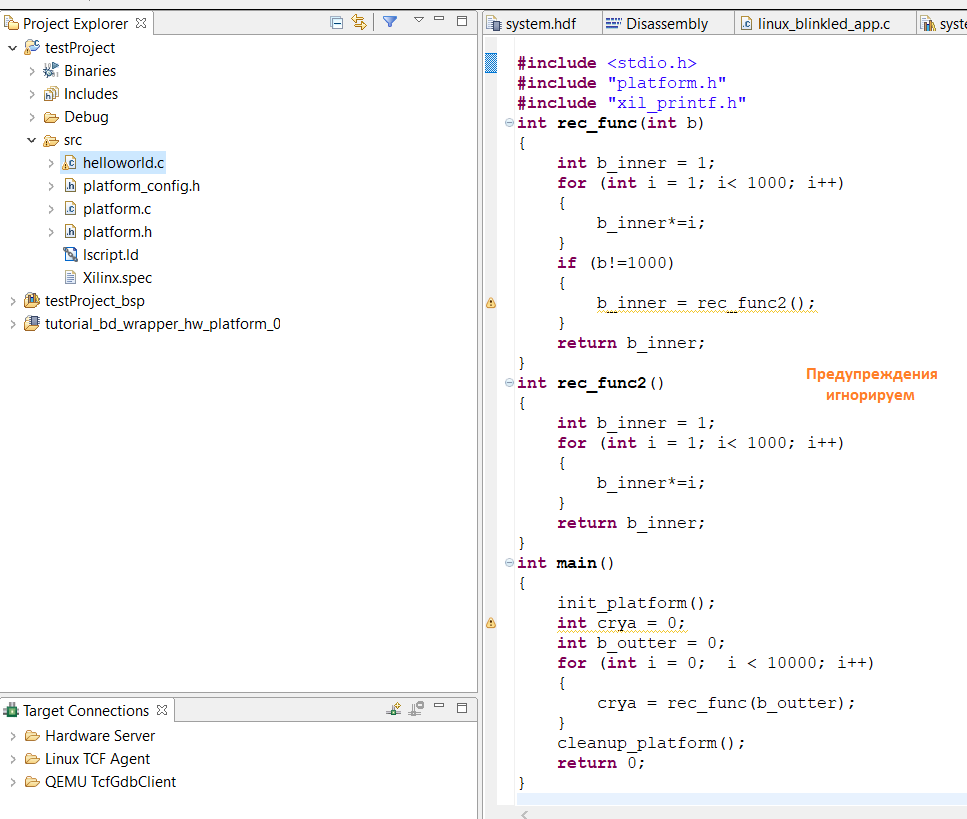
****

рис. 3 Как это стало

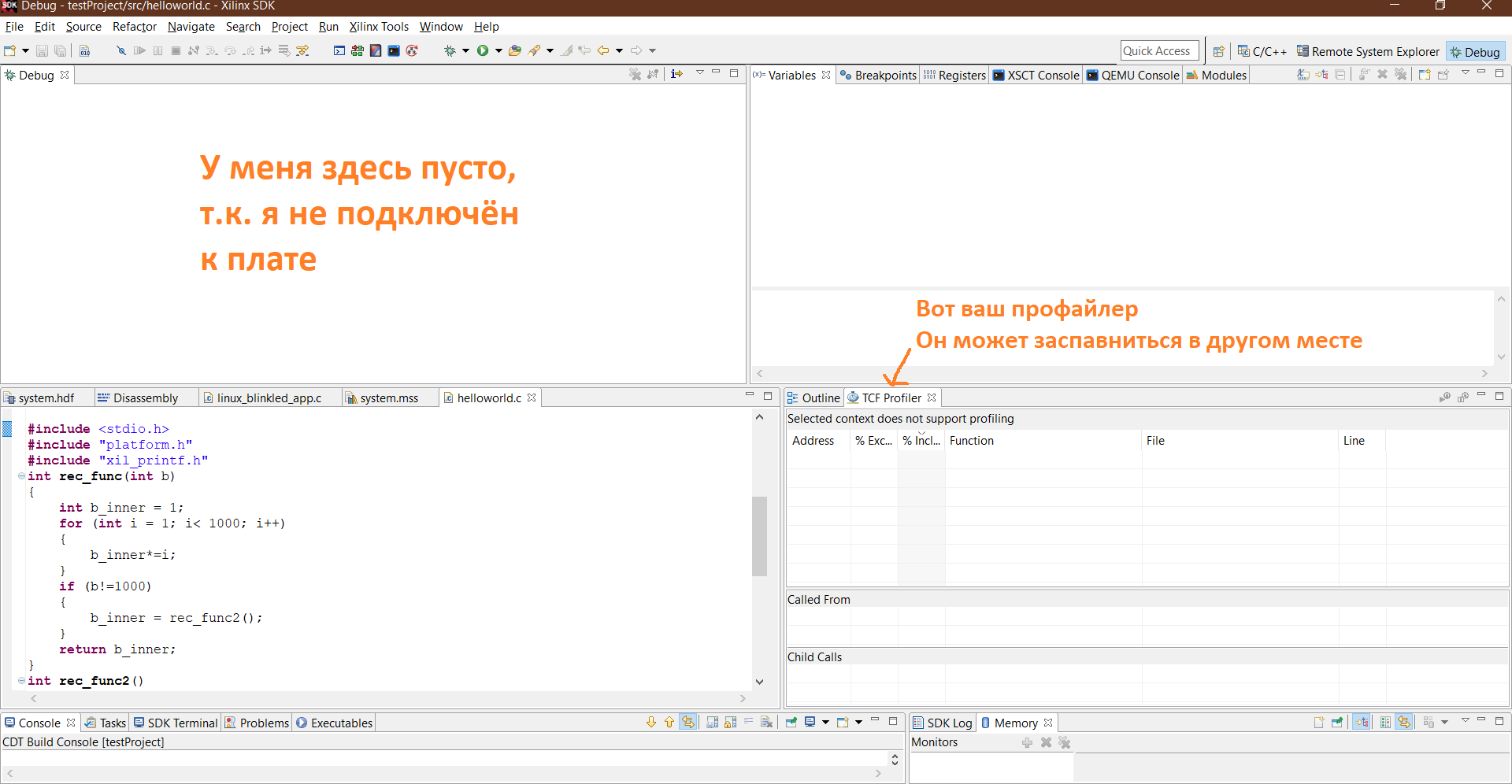
1. Откройте **Run** > **Debug As** > **Launch on Hardware (System Debugger)**, чтобы задебажить проект.

Изображение выглядит как текст, снимок экрана, программное обеспечение, веб-страница

Автоматически созданное описание

рис. 4 Deбаг

1. Откройте TLS Profiler (*Идите на*…: **Window** > **Show View** > **Other** > **Debug** > **TCF Profiler**)



1. В **TLS Profiler** нажмите кнопку , после чего в появившемся окошке нажмите **OK**. Вы получите следующую картину:

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

Изображение выглядит как текст, линия, Шрифт, число

Автоматически созданное описание

1. Здесь вам интересны 2-й и 3-й столбцы:
   * Столбец ***Exclusive*** показывает время выполнения вызова функции **без учета** ее вложенных функций.
   * Столбец ***Inclusive*** показывает время выполнения вызова функции **вместе с** ее вложенными функциями.

PS: значения в диаграмме могут отличаться, суть от этого не меняется.

# Сдача лабы

Если вы выполнили шаги 1–6, всё готово для сдачи. Выберите козла отпущения (*рекомендованное имя:* Максим), который будет тыкать в экран и пламенно вещать о том, как вы добились результата. Текст для него приведён ниже:

*Мы создали новый проект на базе helloworld’a, и вставили в него свой код, после чего сделали «убрать жуков» и в процессе отладки запустили TLS Profiler, позволяющий отслеживать время выполнения функций.*

*Обратите внимание на строчки с названиями* ***rec\_func*** *и* ***rec\_func2 (****далее* ***f1, f2):****В первом столбце функция* ***f1*** *занимает немного времени, а во втором намного больше. Дело в том, что* ***f1*** *вызывает* ***f2*** *внутри себя. Столбец* ***Exclusive*** *не учитывает время её выполнение, пока работает* ***f2****, в то время как столбец* ***Inclusive*** *считает и то и другое. Очевидно, что функция* ***f2*** *имеет одно и то же значение времени, т. к. не имеет в себе вложенных функций.*

*Самый простой способ повлиять на диаграммы, исследуя их поведение – изменить число итераций в циклах функций, и заново запустить профайлер*

# Код для hell\_world.c

**#include** <stdio.h>

**#include** "platform.h"

**#include** "xil\_printf.h"

**int** **rec\_func**(**int** b)

{

**int** b\_inner = 1;

**for** (**int** i = 1; i< 1000; i++)

{

b\_inner\*=i;

}

**if** (b!=1000)

{

b\_inner = rec\_func2();

}

**return** b\_inner;

}

**int** **rec\_func2**()

{

**int** b\_inner = 1;

**for** (**int** i = 1; i< 1000; i++)

{

b\_inner\*=i;

}

**return** b\_inner;

}

**int** **main**()

{

init\_platform();

**int** crya = 0;

**int** b\_outter = 0;

**for** (**int** i = 0; i < 10000; i++)

{

crya = rec\_func(b\_outter);

}

cleanup\_platform();

**return** 0;

}

*Когда подступит сессия – а она обязательно подступит – я обязательно сдохну. Так успейте же… помочь мне с курсачом по мат.логике…*