Флаги.

После измерения периода сигнала происходит инициализация TIM14. Обновление и запуск счета включают TIM\_SR\_UIF и происходит прерывание. В прерывании запускается АЦП, потом DMA и программа обработки данных.

То есть никто не ждет пока TIM14 досчитает до UIF. При своем следующем обновлении опять запускается АЦП & e.t.

Однако сигнал может случиться шустрым и мы можем не успеть посчитать данные до следующего обновления. Можем не успеть сделать это несколько раз.

Для анализа этой ситуации придуман счетный семафор xCountingSemaphoreADCSpeedNormal, который увеличивается в TIM14\_IRQHandler, а уменьшается в vTaskDMA1\_Channel1\_IRQ (void \*argument) – псевдообработчике прерывания DMA1\_Channel1\_IRQHandler, для переключения контекста (управления) на которую используется бинарный семафор xBinarySemaphoreDMA1\_Channel1.

То есть xCountingSemaphoreADCSpeedNormal ничего не переключает, а только считает. Это счетный флаг. По замыслу его значение должно примерно подсказывать во сколько раз нужно уменьшить частоту, чтобы успеть правильно измерить амплитуду.

Однако, поскольку TIM14 не является «думающим счетчиком» разрешим ему запускать АЦП & e.t. только при uxSemaphoreGetCount(xCountingSemaphoreADCSpeedNormal) == 1.

Поскольку значение семафора увеличивается в TIM14\_IRQHandler, а функции, которая бы его обнуляла я не нашел, то будем этот семафор удалять, а потом заново создавать перед очередной инициализации TIM14. Его значение нам критично только по выходу из прерывания DMA1\_Channel1\_IRQHandler.

Достаточно интересная ситуация происходит, если период сигнала меньше количества симплов. Период делится на их количество и TIM14->ARR становится равным нулю. В этом случае счетчик считает до 0xffff. Такой ситуации лучше избежать.