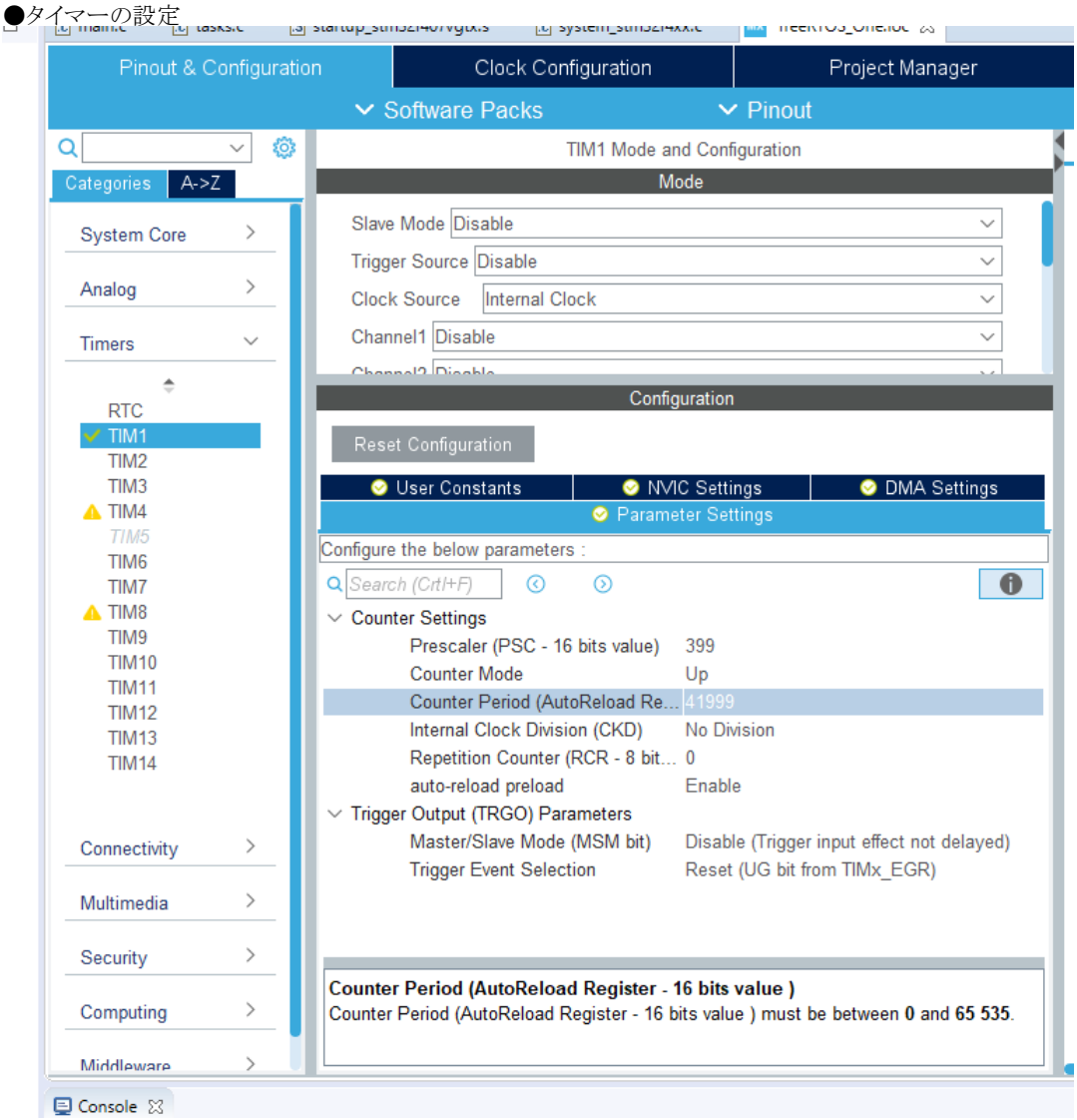
5. タイマー割り込み
〔STM32F4DISCOVERY/STM32CubeIDE で、freeRTOS を使う〕
2020年8月29日

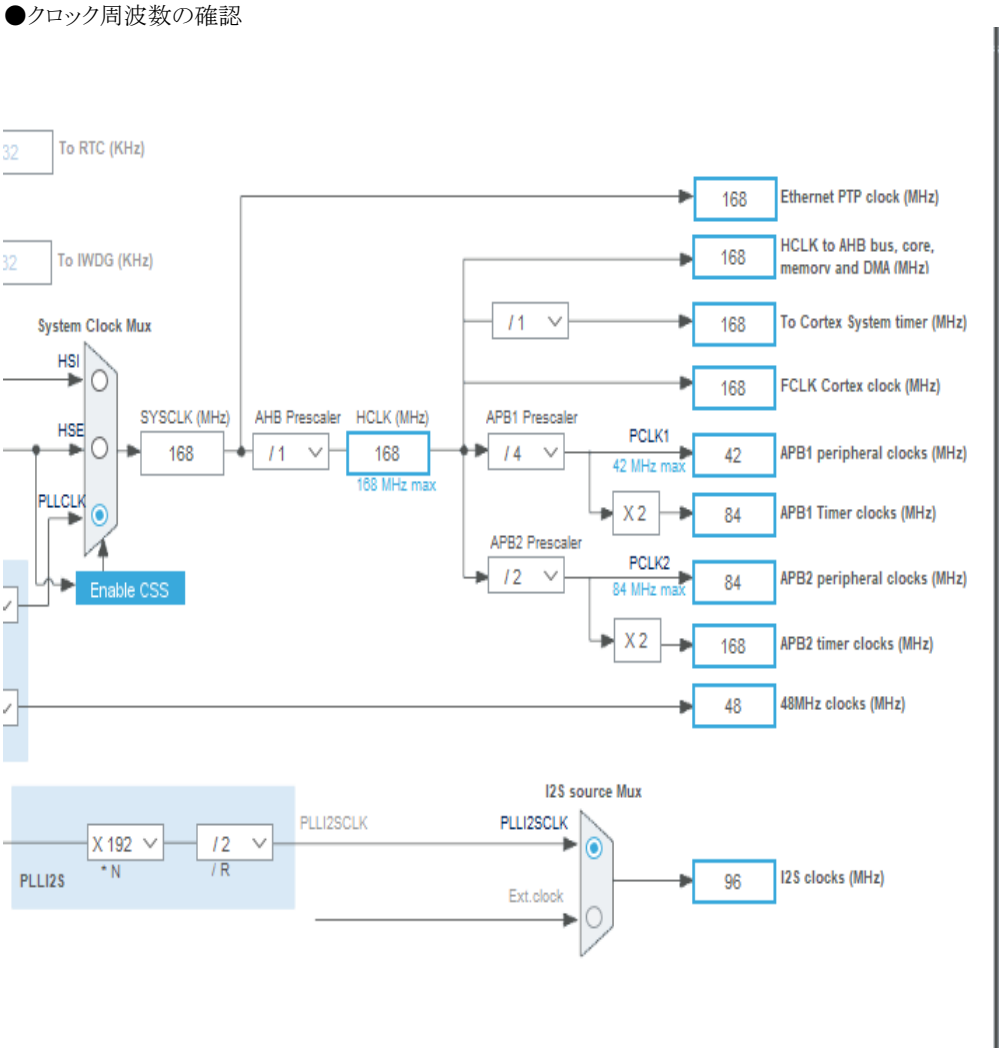
（骨子）
TIM1 を使う。
プリスケーラ × カウンタピリオド × 1/f = タイマー周期
400 × 42000 × 1/168M = 100ms
クロック周波数 f = 168MHz（後述参照）
APB1 : TIM2, 3, 4, 5
APB2 : TIM1, 9, 10, 11

●タイマーの設定



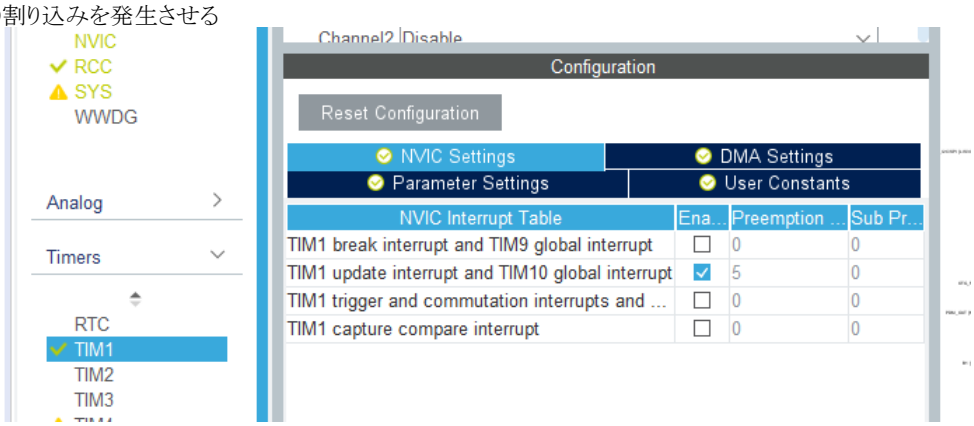

※Clock Source = Internal Clock。 Prescaler、Counter Period には1小さい値を設定する。

●クロック周波数の確認




※APB2=168MHz --> TIM1

●割り込みを発生させる



■実装（100ms 周期のタイマー割り込みの中からシグナルイベントを発生させ、タスクで受けて LED をトグルする）

●初期化（要コーディング）

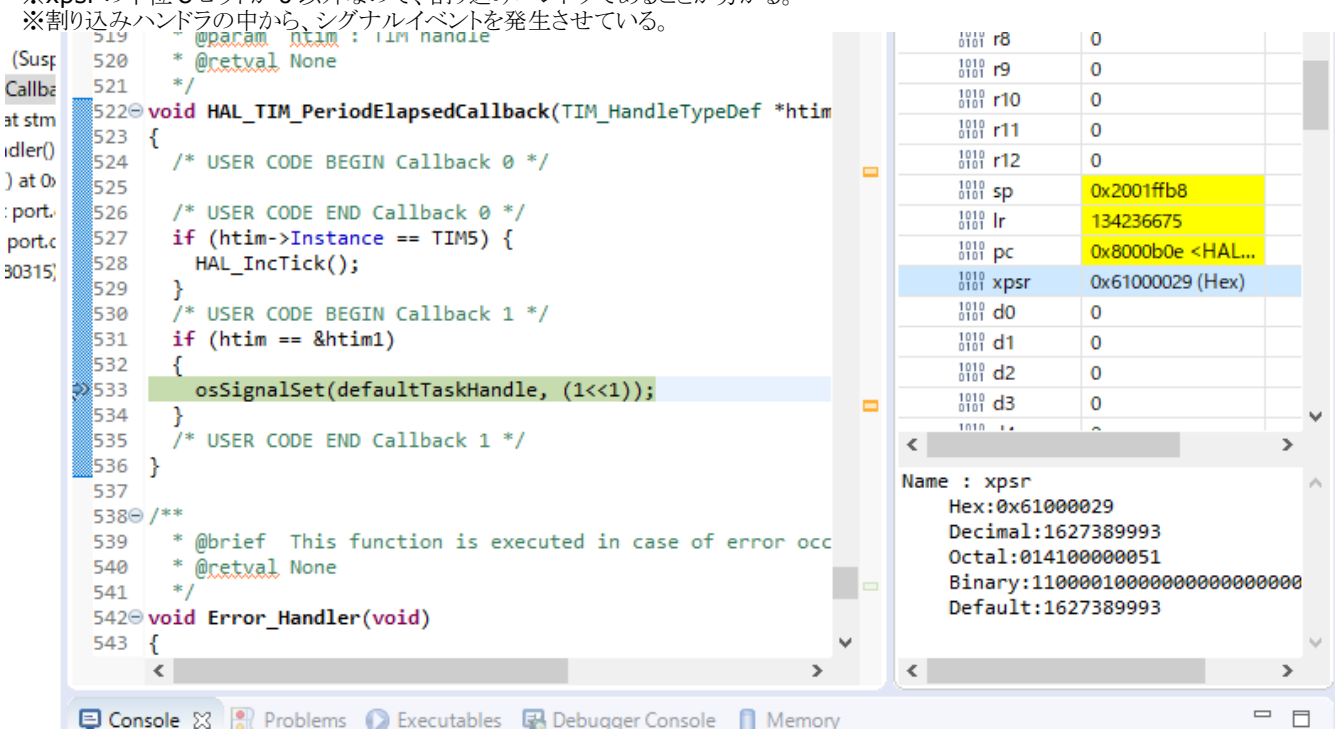
```
HAL_TIM_Base_Start_IT(&htim1);
```

●割り込みハンドラ（TIM 共通）

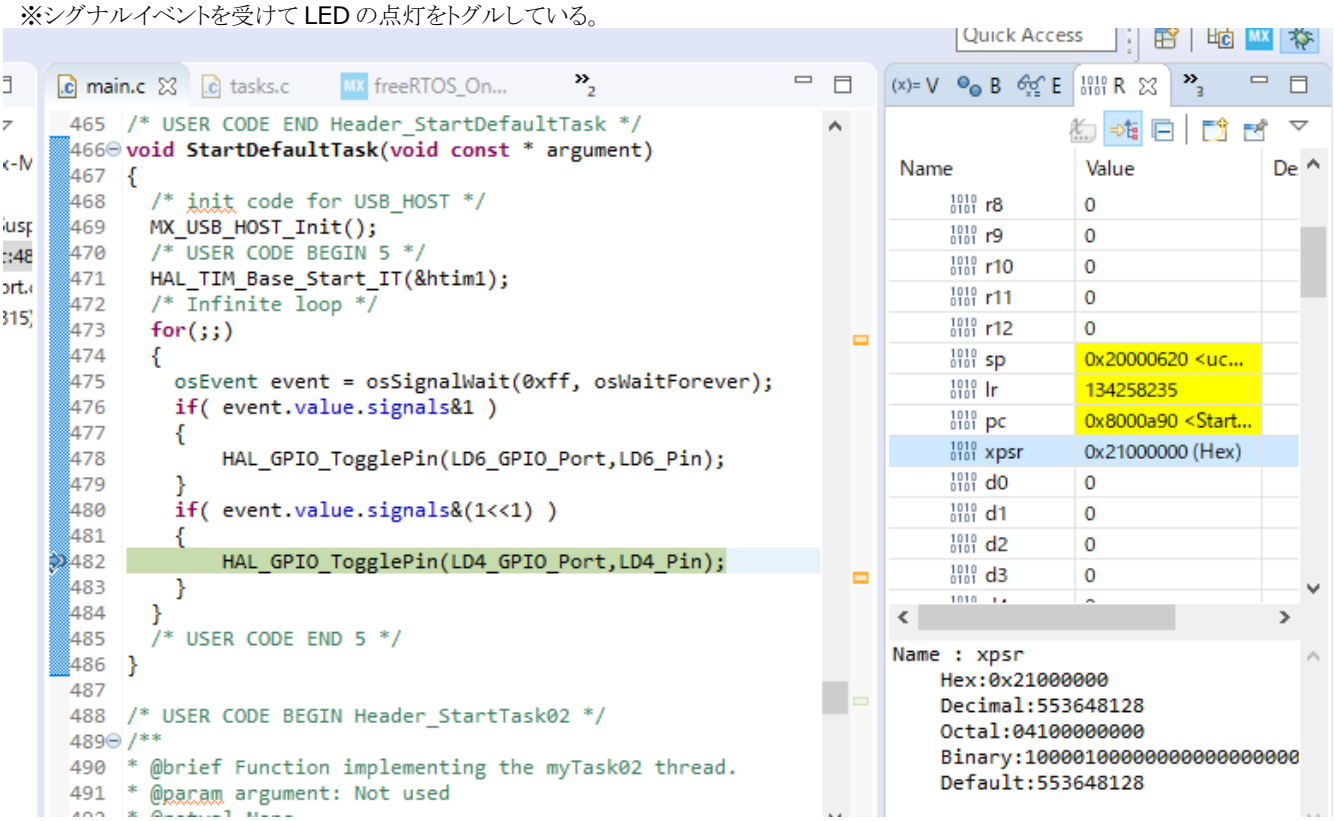
```
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
  /* USER CODE BEGIN Callback 0 */
  /* USER CODE END Callback 0 */

  /* USER CODE BEGIN Callback 1 */
  if (htim == &htim1)
  {
    osSignalSet(defaultTaskHandle, (1<<1));
  }
  /* USER CODE END Callback 1 */
}
```

■割り込みハンドラが本当に割り込みの中で呼ばれているか確認する。

●割り込みハンドラにブレークをかけて、xpsr の値を調べる。
※xpsr の下位 8 ビットが 0 以外なので、割り込みハンドラであることが分かる。
※割り込みハンドラの中から、シグナルイベントを発生させている。



●シグナルイベントを受けるタスク側にブレークをかけて、xpsr の値を調べる。
※xpsr の下位 8 ビットが 0 なので、タスク(割り込みハンドラではない)ことが分かる。
※シグナルイベントを受けて LED の点灯をトグルしている。




■参考URL

STM32CubeIDE で FreeRTOS を使ってみる
https://moons.link/post-117/

(コンフィグレーターを使ってみる　タイマー割り込み編)
https://moons.link/post-50/