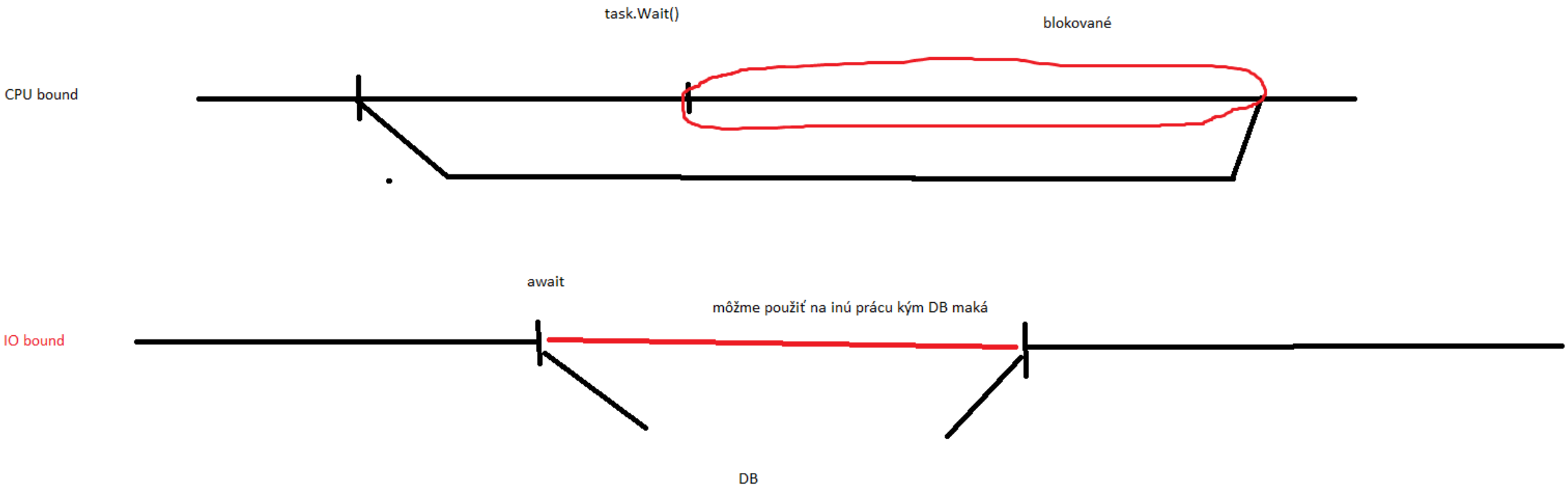
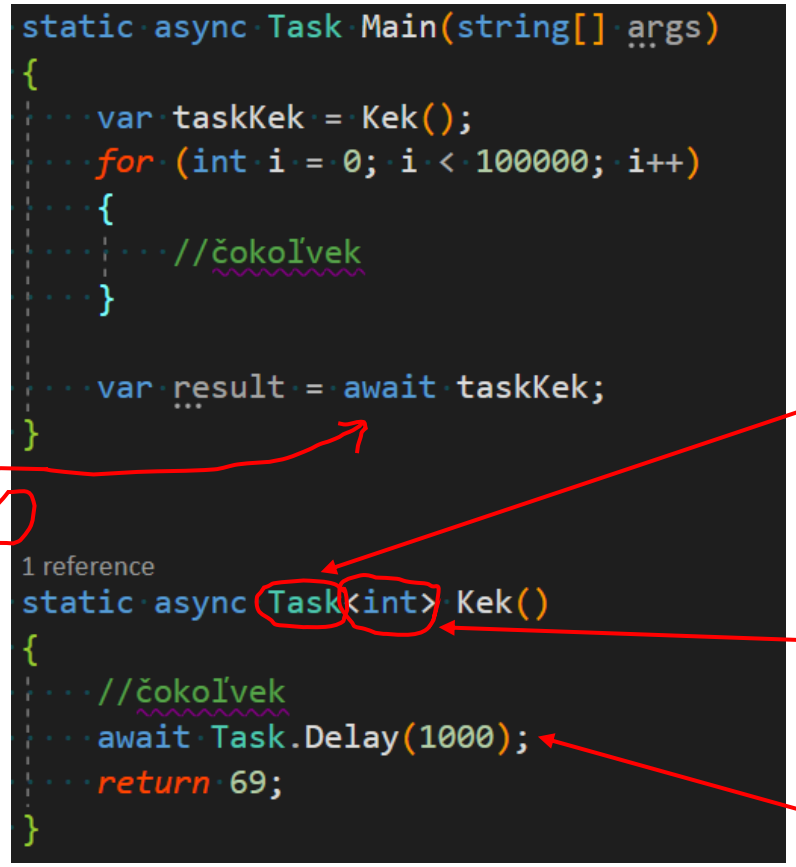
Cviko11 recap



Async používame práve pre IO operácie (súbory, sieť…)

Štandardne môžte očakávať, že na pozadí sa nevytvára nové vlákno



Po zavolaní async metódy, sa metóda zavolá a vykonáva klasicky synchrónne (modifikátor *async* nám len povolí použiť *await,* nič viac nerobí)

Vždy návratová hodnota musí byť Task (používa sa nato aby sme sa vedeli vrátiť späť do metódy po dobehnutí async volania)

Ak chceme aby metóda niečo reálne vrátila tak použijeme parametrizovaný Task

Po zavolaní *await* sa predá riadenie o metódu vyššie (obdobne ako pri *yield return*) a pokračuje sa v jej vykonávaní kým sa znova nenarazí na *await* (až toto robí danú metódu asynchrónnou)

await taktiež “rozbalí” Task a vráti nám jeho “vnútro” (vtomto prípade int)

[Ako to funguje]

[Await automat, ktorý beží na pozadí]

```
private static void Exercise05_WhatWillBeTheNextStepAfterThrowStatement()
{
    try
    {
        Exercise05ThrowExceptionAsync();
    }
    catch (InvalidOperationException ex)
    {
        Console.WriteLine(ex.Message);
    }
}

1 reference
private static async void Exercise05ThrowExceptionAsync()
{
    throw new InvalidOperationException("Exercise03 has thrown an exception!");
}
```

Nepoužívať *async void* kým nemusíte (zvyčajne nevyhnutné len pre event handlery)

Metódy s takouto hlavičkou nevieme awaitnuť a vtomto prípade nechytíme výnimku a celé sa to zosype

Nato aby sme výnimku chytili, treba metódu aj *awaitnuť*