FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Dokumentace k projektu

# Měření výpočetních režií modulem Data Watchpoint and Trace (DWT)

Mikroprocesorové a vestavěné systémy

15. prosince 2020 Mirka Kolaříková(xkolar76)

# 1 Zadání projektu

Cílem projektu je demonstrovat možnosti měření výpočetních režií pomocí modulu Data Watchpoint and Trace (DWT) dostupného na mikrokontroléru Kinetis K60 s jádrem ARM Cortex-M4 z desky platformy FITkit 3.

# 2 Problematika

Data Watchpoint and Trace jednotka obsahující komparátory, které mimo jiné umožní počítání počtů cyklů CPU, počtů cyklů CPU strávených ve spánku a počet cyklů strávených obsluhou přerušení.

Tato počítaní cyklů provedeme na určitých částech kódu opakovaně a získáme maximální, minimální a průměrný počet cyklů.

Bohužel kvůli pandemii nebylo možné vybavení FITkit3 zajistit a měření na něm provést.

## 2.1 Využité registry

V tomto projektu se pracuje s následujícími registry:

**Debug Exception and Monitor Control Register, DEMCR**

Viz obrázek 1. TRCENA bit (24. bit) registru slouží k povolení DWT funkcí.

**Control register, DWT_CTRL**

Viz obrázek 2. V tomto registru se povolí čítače cyklů. K povolení čítače cyklů slouží CYCCNTENA bit (0. bit). K povolení a inicializaci čítače cyklů strávených ve spánku slouží SLEEPEVTENA bit (19. bit). K povolení a inicializaci čítače cyklů při přerušení je EXCEVTENA bit (18. bit).

**Cycle Count register, DWT_CYCCNT**

Viz obrázek 3. V tomto registru nastavíme nebo zjistíme provedený počet cyklů.

**Sleep Count register, DWT_SLEEPCNT**

Viz obrázek 4. V tomto registru zjistíme provedený počet cyklů strávených ve spánku.

**Exception Overhead Count register, DWT_EXCCNT**

Viz obrázek 5. V tomto registru zjistíme provedený počet cyklů strávených obsluhou přerušení.

# 3 Popis řešení

Implementace se nachází v souboru `main.c`.

## 3.1 Inicializace MCU

Ve funkci `MCUInit()` provedeme nastavení hodin a vypnutí watchdogu.

## 3.2 Povolení DWT

K povolení funkcí DWT je potřeba nastavit **TRCENA** bit **DEMCR** registru na **1**. **DEMCR** se nachází na adrese **0xE000EDFC**. DWT se povolí pomocí makra `InitCycleCounter()`.

```
#define DEMCR                (*((volatile uint32_t*)0xE000EDFC))
#define DEMCR_TRCENA_BIT     (1UL<<24)

#define InitCycleCounter() \
  DEMCR |= DEMCR_TRCENA_BIT
```

## 3.3 Obsluha čítačů cyklů

Control register se nachází na adrese **0xE0001000**. Nastavením **CYCCNTENA** bitu Control registu na **1** povolíme generování paketů čítače cyklů. Nastavením tohoto bitu na **0** generování deaktivujeme. K aktivaci a dekativaci slouží makra `EnableCycleCounter()` a `DisableCycleCounter()`. Počet provedených cyklů se nachází v registru **DWT_CYCCNT**, který je na adrese **0xE0001004**.

Nastavením **SLEEPEVTENA** bitu na **1** povolíme a inicializujeme počítaní cyklů strávených ve spánku. Na to slouží `EnableSleepCycleCounter()`. Počet těchto cyklů se nachází v registru **DWT_SLEEPCNT**, který je na adrese **0xE0001010**.

Obdobně nastavením bitu **EXCEVTENA** na **1** povolíme a inicializujeme počítání cyklů strávených obsluhou přerušení. Počet se nacházi v registru **DWT_EXCCNT**, který je na adrese **0xE000100C**.

Ukázka definice maker pro počítaní cyklů a cyklů strávených ve spánku:

```
#define DWT_CONTROL             (*((volatile uint32_t*)0xE0001000))
#define DWT_CYCCNT              (*((volatile uint32_t*)0xE0001004))
#define DWT_SLEEPCNT            (*((volatile uint32_t*)0xE0001010))
#define DWT_CYCCNTENA_BIT       (1UL<<0)
#define DWT_SLEEPEVTENA_BIT     (1UL<<19)

#define EnableCycleCounter() \
  DWT_CONTROL |= DWT_CYCCNTENA_BIT

#define DisableCycleCounter() \
  DWT_CONTROL &= ~DWT_CYCCNTENA_BIT

#define GetCycleCounter() \
  DWT_CYCCNT

#define EnableSleepCycleCounter() \
  DWT_CONTROL |= DWT_SLEEPEVTENA_BIT

#define GetSleepCycleCounter() \
  DWT_SLEEPCNT
```

### 3.4 Měření výpočetní režie

Měří se výpočetní režie pro konstrukce typu sekvence, selekce a iterace. Poté se měří i režie pro Selection sort [1] a Insertion sort[4] na poli o pěti prvcích.

Před každým měřením určité konstrukce se ve funkci `initStatsVar()` inicializují proměnné, které uchovávají celkový, minimální, maximální a průměrný počet cyklů.

Poté ve smyčce opakovaně probíhá měření režie tohoto kódu a aktualizují se hodnoty s minimálním, maximálním a celkovým počtem cyklů. Následně se vypočítá i průměrný počet vykonaných cyklů.

Vypočítáne statistiky se potom vytisknout do termínálu pomocí funkce `printStats()`.

Příklad měření výpočetní režie Selection sortu při 10 opakování:

```
#define ITERATIONS_COUNT            10

initStatsVar();
for(int i = 0; i < ITERATIONS_COUNT; i++){
        startCounting();
        int arr[] = {64, 25, 12, 22, 11};
    int n = sizeof(arr)/sizeof(arr[0]);
    selectionSort(arr, n);
        finishCounting();
}
cyclesMean = cyclesTotal / ITERATIONS_COUNT;
sleepCyclesMean = sleepCyclesTotal / ITERATIONS_COUNT;
printStats("SELECTION_SORT");
```

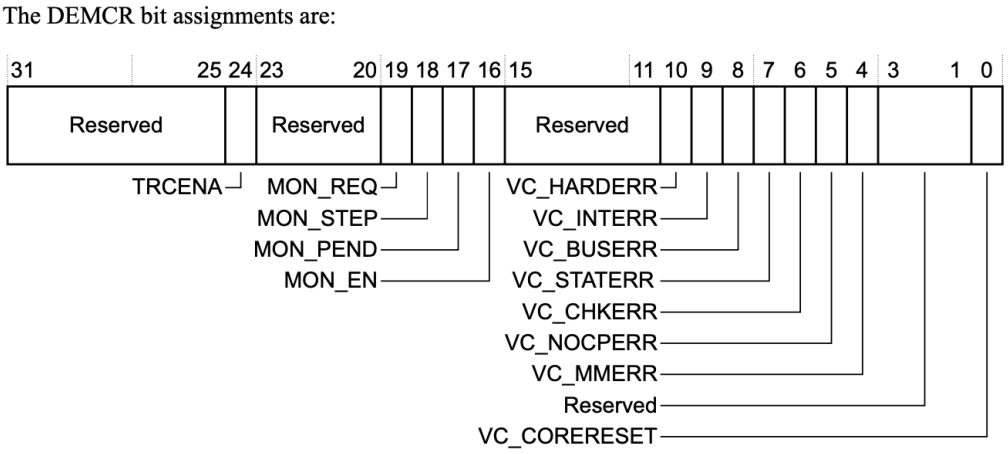
# 4 Závěr

Projekt bohužel nebylo možné bez FITkitu vyzkoušet, což je škoda, protože výsledné statistiky by učitě byly velmi zajímavé.

Informace o DWT jsem hledala v oficiálních manuálech [3] [2]. Také mi pomohly další zdroje [6] [5].

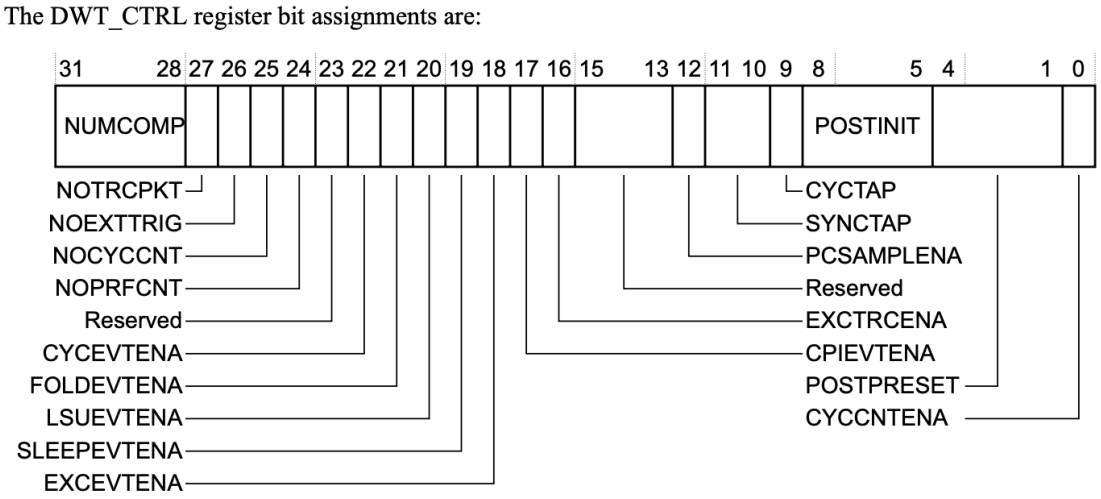
Kromě ukázky kódu s přerušením se mi podařilo implementovat vše potřebné.

# A DEMCR registr




Obrázek 1: DEMCR registr

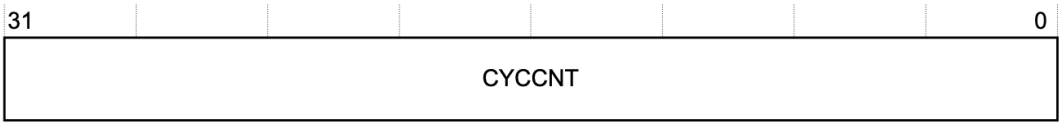
# B Control registr




Obrázek 2: Control registr

# C Cycle counter

The DWT_CYCCNT register bit assignments are:




Obrázek 3: Cycle Count registr

# D Sleep cycle counter

The DWT_SLEEPCNT register bit assignments are:


Obrázek 4: Sleep Cycle Count registr

# E Exception cycles counter

The DWT_SLEEPCNT register bit assignments are:


Obrázek 5: Exception cycles counter registr

# Použité zdroje


[1] Selection Sort. [online], květen 2020. Dostupné z: https://www.geeksforgeeks.org/selection-sort/

[2] *ARM®v7-M Architecture Reference Manual*. prosinec 2014.

[3] *ARM® Cortex®-M4 Processor Technical Reference Manual*. únor 2015.

[4] Insertion Sort. [online], červenec 2020. Dostupné z: https://www.geeksforgeeks.org/insertion-sort/

[5] Chernobay, V.: DWT - Data Watchpoint and Trace unit. [online]. Dostupné z: https://arm-stm.blogspot.com/2014/05/dwt-data-watchpoint-and-trace-unit.html

[6] Styger, E.: Cycle Counting on ARM Cortex-M with DWT. [online], leden 2017. Dostupné z: https://mcuoneclipse.com/2017/01/30/cycle-counting-on-arm-cortex-m-with-dwt/