

# Výukové prostredie na simuláciu SIMD algoritmov

Jozef Číž

Univerzita Komenského v Bratislave  
Fakulta matematiky, fyziky a informatiky

*ciz5@uniba.sk*

31. augusta 2022

- SIMD (Single Instruction Multiple Data)
- WT formalizmus
  - Vysokoúrovňový programovací jazyk
  - Pohodlný návrh a analýza algoritmov
- WT\* platforma
  - Existuje kompilátor a virtuálny stroj
  - Problém s užívateľským rozhraním
- Cieľ – vytvoriť užívateľské rozhranie

# Príklad kódu

```
// pole obsahujúce 0 alebo 1
input int pole[_];

output int existuje = 0;

for(int i = 0; i < pole.size; ++i)
    if(pole[i])
        existuje = 1;
```

# Príklad kódu

```
// môžeme zapisovať do tej istej
// premennej ak ide o rovnakú hodnotu
#mode cCRCW

// pole obsahujúce 0 alebo 1
input int pole[_];

output int existuje = 0;

pardo(i: pole.size)
    if(pole[i])
        existuje = 1;
```

- Jednoduchosť
- Podpora platforiem
  - Web
  - Natívne na rôznych operačných systémoch
- Ladič
  - Dynamické body prerušenia
  - Krokovanie – dnu, cez, von
  - Zobrazovanie hodnôt premenných
  - Podmienené body prerušenia

# Virtuálny stroj

- Paralelný stroj sa líši od sériového
  - Kontrola prístupu do pamäte
  - Kaktusový zásobník
  - ...
- Zdrojový kód sa kompiluje, čím vzniknú inštrukcie
- Virtuálny stroj vykonáva zaradom inštrukcie
- Jednoduchý spôsob, ako rozširovať funkcionality a ovplyvňovať beh stroja je pridávanie nových typov inštrukcií

- Dynamické body prerušenia
  - Inštrukcia `NOOP` pred každým príkazom
  - V prípade potreby nahradená inštrukciou `BREAK`
- Krokovanie
  - Inštrukcie `STEP_IN` a `STEP_OUT` okolo každého príkazu
- Zobrazovanie hodnôt premenných
  - Relatívne adresy každej premennej zapamätané pri kompilácii

# Podmienené body prerušenia

- Podmienka je kód, teda postupnosť inštrukcií
- Inštrukčný kód je pridaný na koniec
- Výsledkom je číselná hodnota
- Ak je v aspoň jednom vlákne nenulová, program sa zastaví
- Avšak
  - Zapamätanie výslednej hodnoty nie je priamočiare
  - Vygenerovanie inštrukčného kódu je náročné



# Vývojové prostredie

- Dear ImGui
- Dokovanie
- Notifikácie
- Dialóg pre výber súborov
- Moduly
- Efektívne a nenáročné

# Vývojové prostredie

The screenshot displays a C++ development environment with three main panels:

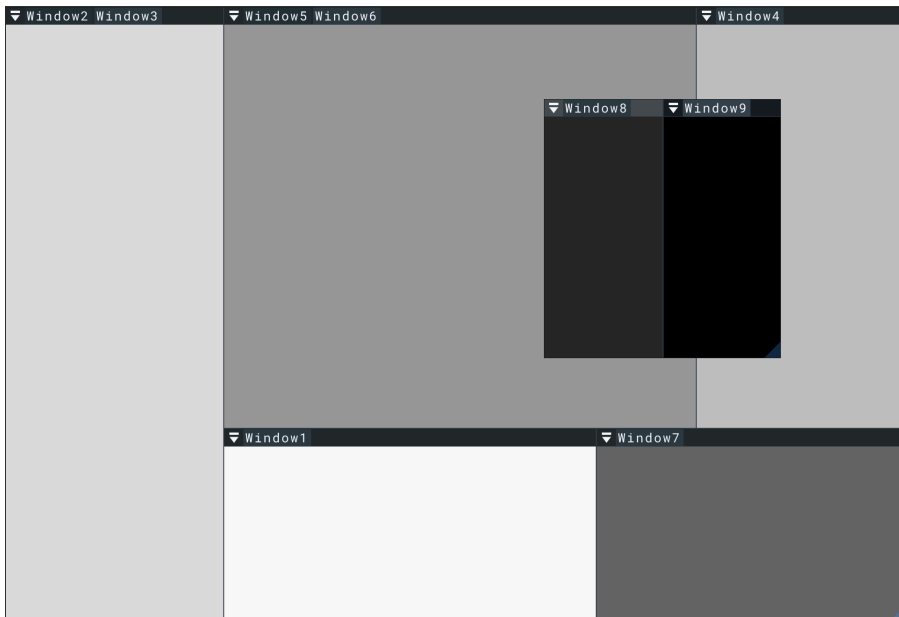
- FileTree:** Shows a project structure with folders like .cache, CMakeFiles, and src, and files like main.cpp and test.wt.
- main.cpp test.wt:** Contains the following code:

```
7/15 12 lines | Ins | | C++ | test.wt
1 input int x;
2 input int A[];
3 output int sum;
4
5 pardo(i: A.size) {
6     A[i] += x;
7     A[i] += 1;
8 }
9
10 for(int i = 0; i < A.size; ++i) {
11     sum += A[i];
12 }
```
- Dear ImGui Demo Variable Viewer Debug:** Shows the state of global variables:

Name	Type	Value
x	int	5
A	int	[5 10 15 20 25]
sum	int	0

Below the code editor, the **Program Analyzer** panel shows compiler output and a breakpoint set at line 6 of test.wt. The **Threads** panel shows a single thread with watchpoints for variables x, A, and sum.

# Dokovanie



# Moduly vývojového prostredia

- Editor
- Vstup programu, výpis programu a kompilátora
- Súborový strom
- Ovládač ladiča
- Zobrazovač premenných
- ...

# Editory

```
1 // Zep
2
3 input int x;
4 input int A[_];
5 output int sum;
6
7 pardo(i: A.size) {
8     A[i] *= x;
9     A[i] += 1;
10 }
11
12 for(int i = 0; i < A.size; ++i) {
13     sum += A[i];
14 }
```

Standard INSERT

```
File Edit View
1/22 14 lines | Ins | * | C++ | test.wt
1 // InquiColorTextEdit
2
3 input int x;
4 input int A[_];
5 output int sum;
6
7 pardo(i: A.size) {
8     A[i] *= x;
9     A[i] += 1;
10 }
11
12 for(int i = 0; i < A.size; ++i) {
13     sum += A[i];
14 }
```

# Moduly ladiča

▼ Debug Control

Set source test.wt

Compile Run Destroy

Continue  Trace

Step Over Into Out

Breakpoint:  Stop on BP

Add Remove Remove All

0097>test.wt:7(7:2:7:10)

test.wt	File
6	- + Line
return i > 3;	Condition

Add

▼ Variable Viewer

W= 129 T= 37 W/T= 3.49

▼ Globals

Name	Type	Value
x	int	5
A	int	[5 10 15 20]
sum	int	0

Thr: 0 - +

▼ Call stack

Size: 1

▼ 0: (null) (base=0 ret\_ins=96)

Name	Type	Value
▼ locals		
▼ parent		
i	int	0
▼ parent		
x	int	5
A	int	[5 10 15 20]
sum	int	0

▼ Threads

Watch ▶ tid=4 | par=3 | i=0

Watch ▼ tid=5 | par=3 | i=1

Name	Type	Value
▼ locals		
▼ parent		
i	int	1
▼ parent		
x	int	5
A	int	[5 10 15 20]
sum	int	0