Architektury systemów komputerowych 2016

Projekt nr 2

Symulator pamięci podręcznej

Celem implementacji tego projektu jest zrozumienie w jaki sposób procesor zarządza pamięcią podręczną na podstawie danych zbieranych w trakcie wykonania programu.

Na stronie przedmiotu udostępniono szkieletu kodu rozwiązania i przykładowe dane wejściowe. Wykonaj pomiary na dostarczonych śladach programów. Narzędzia służące do wygenerowania śladów są dostępne w katalogu *pintools*.

Na wejściu program ma akceptować rekordy reprezentujące dostępy do pamięci w porządku wykonania programu. Kolejne pola rekordu to: rodzaj operacji (odczyt lub zapis), 64-bitowy adres dostępu, rozmiar danych (1,2,4 lub 8). Zauważ, że symulator nie musi utrzymywać zawartości bloków.

```
w 7ffecf594948 8
r 7facb2f2ee70 4
```

Na wyjściu program ma wydrukować:

- w trybie gadatliwym (z opcją linii poleceń `--verbose`) dla każdego dostępu jedna linia, w której będzie para „`index:set`" dla trafienia lub „`miss`" dla chybienia; dla dwupoziomowej pamięci podręcznej odpowiednio z prefiksem „`L1:`" i „`L2:`"
- łączna ilość trafień i chybień dla poszczególnych poziomów pamięci podręcznej, ilość dostępów do pamięci operacyjnej.

Symulator musi akceptować z linii poleceń opis organizacji pamięci podręcznej, który dla każdego poziomu cache będzie zawierać informacje o:

- polityce zapisu: *write-back* i *write-allocate* lub *write-through* i *write-no-allocate*,
- wielkość bloku (potęga dwójki od 8 do 256): *block*,
- liczbę wierszy (potęga dwójki): *size*,
- liczność zbioru (od 1 do 16, 1 $\rightarrow$ mapowanie bezpośrednie): *associativity*,

Przykład wywołania symulatora z linii poleceń (pamięć podręczna `L2` nieaktywna):

```
./cache-sim --l1-write-back --l1-associativity=4 --l1-block=64 --l1-size=512 \
            < data0.in > data0.out
```

UWAGA! Obchodzi nas jedynie symulowanie zachowania metadanych (tj. znacznik i dodatkowe bity) w poszczególnych blokach – dlatego należy zdeklarować typ reprezentujący rekord metadanych. Pamięć podręczna będzie reprezentowana jako tablica (tablic) tychże rekordów.

Zadanie 1. Zaimplementuj pamięć podręczną z mapowaniem bezpośrednim i politykami zapisu *write-back* i *write-through*. Upewnij się, że dostępy do danych położonych na granicy dwóch bloków są prawidłowo obsługiwane.

Zadanie 2. Dodaj obsługę organizacji sekcyjno-skojarzeniowej z polityką wymiany *LRU*. Zbiory są uporządkowane, a bloki nie mogą zmieniać kolejności. Niech n będzie licznością zbioru. Algorytm ma być zrealizowany podobnie sieć sortująca o głębokości $O(log\, n)$. Można użyć co najwyżej $O(log\, n)$ dodatkowych bitów na blok.