Weiter mit Trace Cache (in 3.2.)
Tracecache:
Enthält Befehlssequenzen, die vorher abgelaufen sind
Sprungentscheidungen sind dort so abgelegt, wie sie in der Historie getroffen worden sind,
Wird bei erneutem Start, die sich im Trace Cache befindet, diese gefunden und dort abgelegt,
Maschinencode wird direkt abgearbeitet,
Sprungvorhersage integriert,
Abruch bei Abarbeitung aus dem Trace Cache nur notwendig, wenn eine Sprungvorhersage nicht eingetroffen ist. (F36,37)

Vorteile:

- Befehle des Traces müssen nicht aus dem Befehlsspeicher gelesen werden,
- Sprungvorhersage ist integriert und bezieht sich auf die Befehle nach dem Sprung,
- Bei mehreren Sprüngen erfolgt die Vorhersage der Zusammenhänge der Sprünge.

Erläuterung zum Ablauf (F37)

Problem: Cache ist deutlich kleiner als der Hauptspeicher.

➔ Er ist nach Anlauf immer vollständig belegt
➔ Eintragen neuer Werte geht nur durch Überschreiben (Ersetzen)
➔ Notwendig ist Ersetzungsstrategie (Auswahl der Werte, die überschrieben werden sollen)

Varianten:

- FIFO (First In First Out)
  Entfernen der ältesten Einträge
  Voraussetzung: Mechanismus , der das Alter ausweist.
  z.B. Zähler, der beim Schreiben des Eintrags gestartet wird
  Hardwareaufwand relativ hoch

- LRU (least recently used)
  Entfernen des Eintrags, auf den am längsten nicht mehr zugegriffen wurde
  Voraussetzung: Mechanismus, der den Zeitpunkt des letzten Zugriffs aufweist.
  z.B. Register, das den Zeitpunkt des letzten Zugriffs aufweist

Bei den nicht voll assotiativen Caches existiern Einschränkungen, die die Ersetzung einfacher machen.

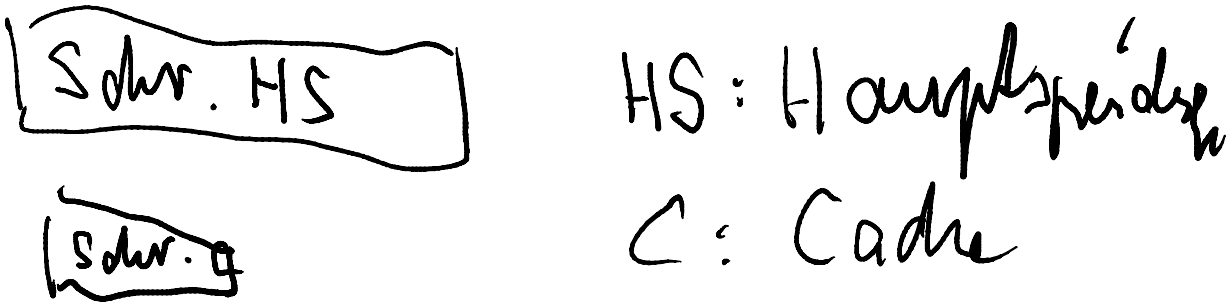
Direct Mapped Cache
- ➔ Je Index nur ein Eintrag, beim Überschreiben existiert keine Alternative, bei erneutem Auftreten eines Index mit anderem Tag -> Überschreiben

Zwei-Wege-Cache:
- ➔ LRU: 1bit zeigt von den zwei Werten auf den älteren

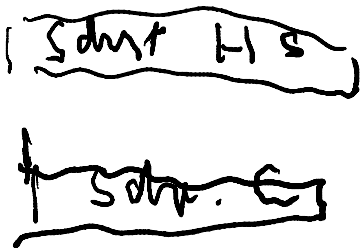
Weitere Strategien:
Zufällig
LFU (least frequently used)
- ➔ Entfernen des Eintrags, der am wenigsten benutzt wurde (Hardwareaufwand)

Weiteres Problem:
- Schreiben in den Cache, Wert ist enthalten bzw. wird durch Ersetzung erzeugt

Zeitablauf:



Write Back (F39 unten)



Write Throug (F39 oben) Zeitoptimalität wird nur beim lesen erreicht.

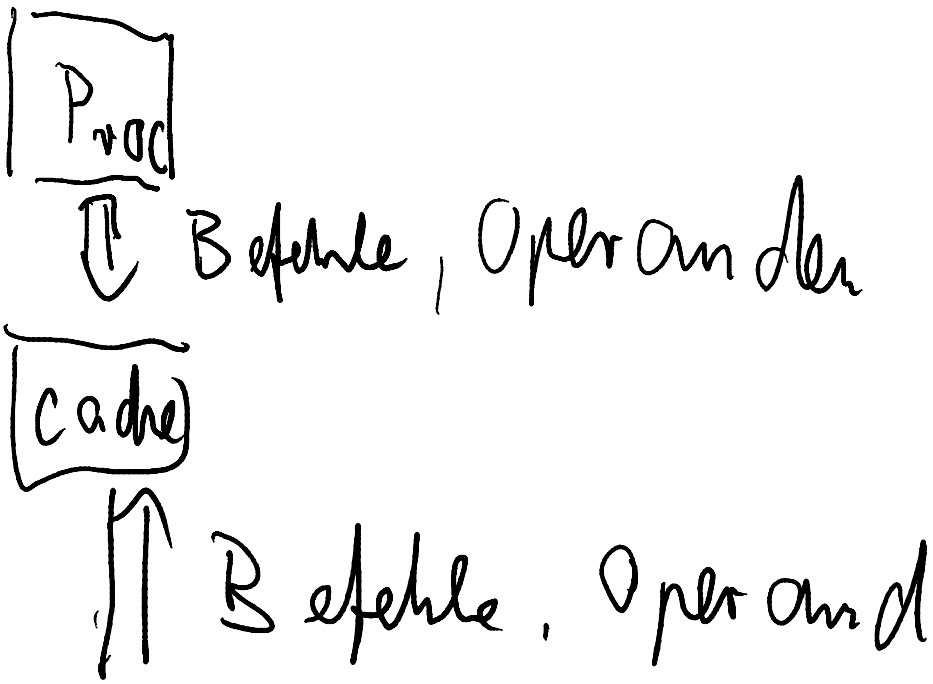
Einschätzung der Effektivität durch Trefferrate:
Treffer (Hit): Wert wird beim Zugriff im Cache gefunden.
0 <= T <=1 Ziel T geht gegen 1, real rund 0,7

Abhängig von:

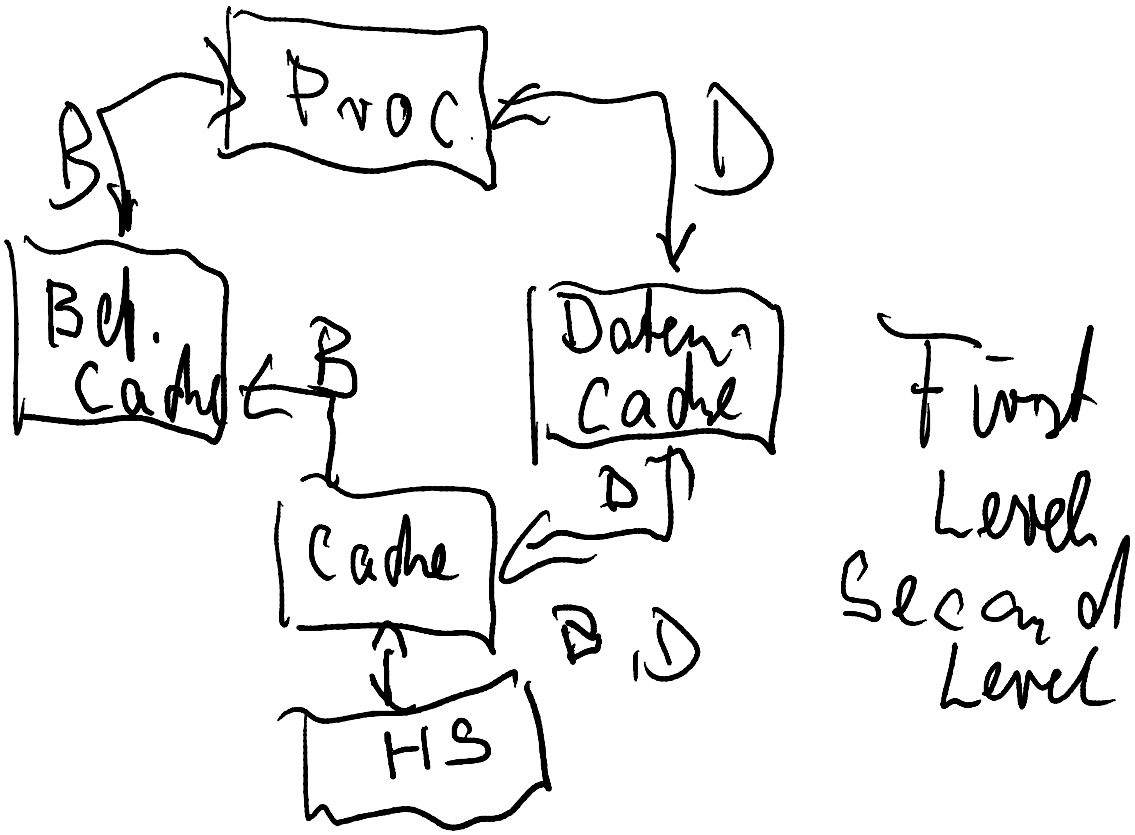
- Cache-Größe
- Verwendeten Prinzipien
- Cache-Architektur
- Aktuelle Programmstruktur (z.B. viele Schleifen benutzen mehrhach hintereinander den gleichen Befehlscode)

Architekturvarianten für Cache-Strukturen:

Princeton:



Mischform Harward, Princeton (häufigste derzeitigste)



Merkmale:
2 Wege für Befehle und Daten zu 2 getrennten First-Lvel- Caches, die zumeist auf dem Prozessorchip angeordnet sind, dadurch Verdopplung der Verbindungen geringerer Aufwand
Danach für Daten und Befehle gemeinsamer Speicher und Hauptspeicher.