

Und deren
Verheideung

Hemmisse in der Fließbandverarbeitung

Datenabhängigkeiten

Echte Datenabhängigkeits (Read After Write)

Ausgabeabhangigkeit (Write After Write)

Gegnabhangigkeit (Write After Read)

Vermeidung von Datenabhängigkeiten

Kontrollfließabhängigkeiten

Voraussetzer Sprung (Branch Delay Slot)

Statische Sprungvorhersage (Branch Prediction)

Dynamische Sprungvorhersage

Mehrdynamische Sprungvorhersage

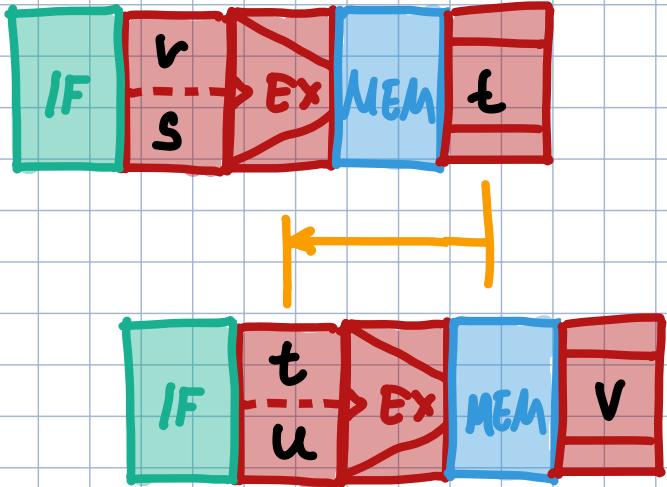
Datenflusshemmisse

Pipeline Hazards

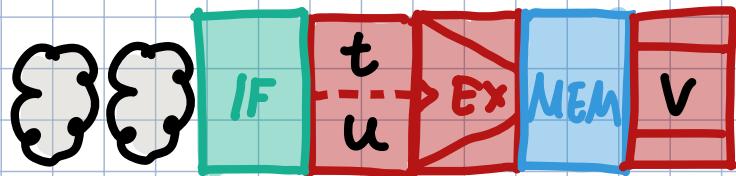
Echte Datumsabhängigkeit (RAW - Read After Write)

addl t -> s

addl v t ->



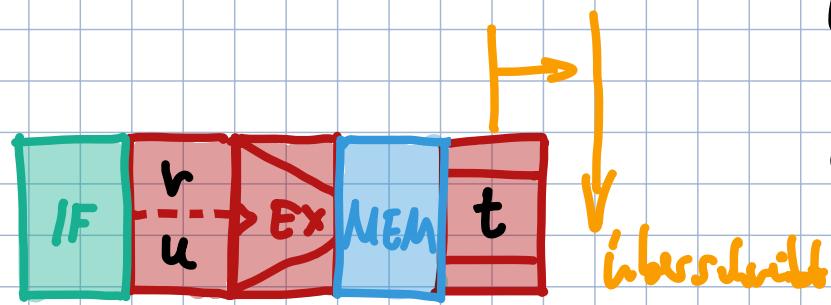
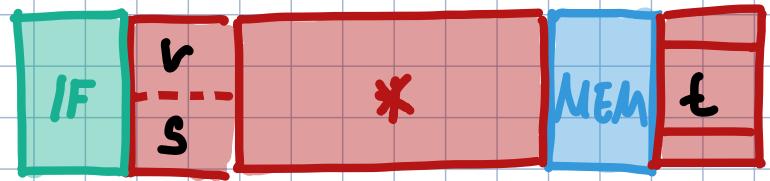
Das Zwischenschreiben in die Register erfolgt erst nachdem der Wert gemäß Datenfluss geben wurde sollte.



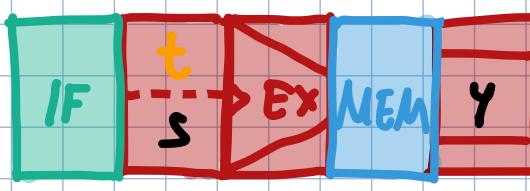
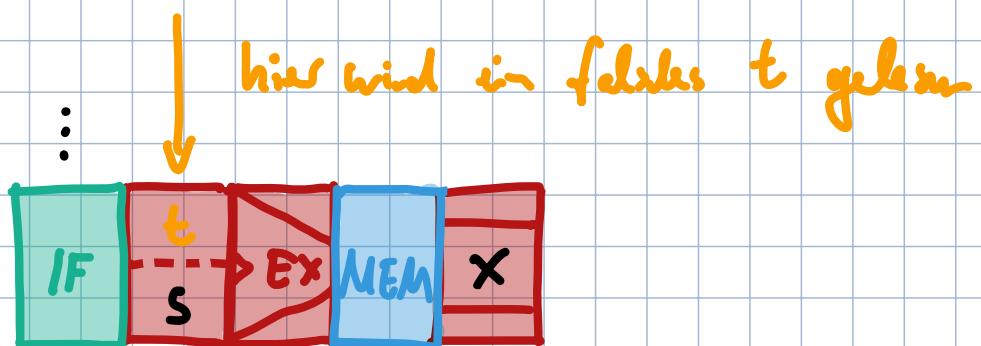
Compiler fügt Blasen (Bubbles) ein
durch NOP-Befehle

Ausgabearbeitung (WAW - Write After Write)

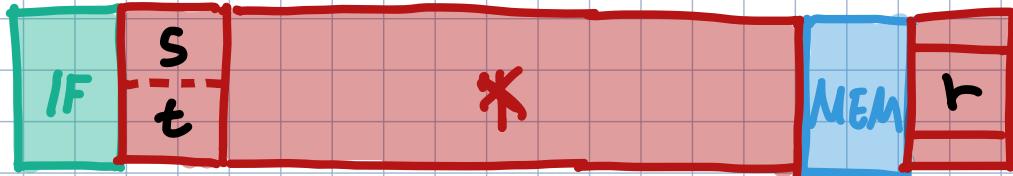
add t r s
ld t a



Ergebnis wird überschrieben bevor es genutzt werden kann.



Befehlsh鋍htigkeit (WAR - Write After Read)

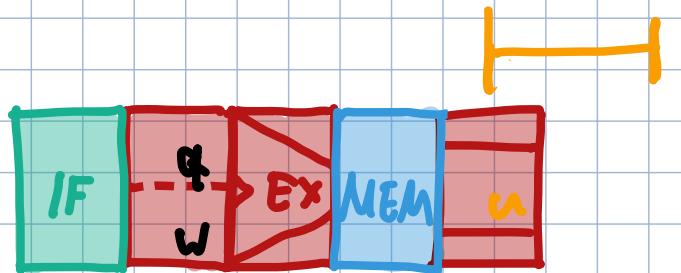
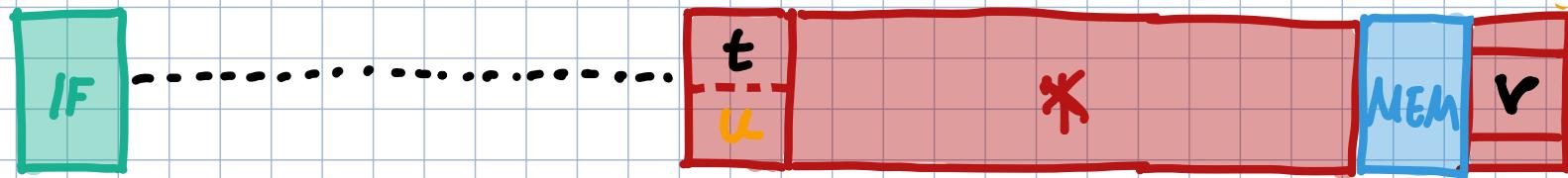


$$r = s \cdot t \quad \text{mul } r \ s \ t$$

$$v = t \cdot u \quad \text{mul } v \ t \ u$$

$$u = q + w \quad \text{add } u \ q \ w$$

wurde noch nicht gelesen

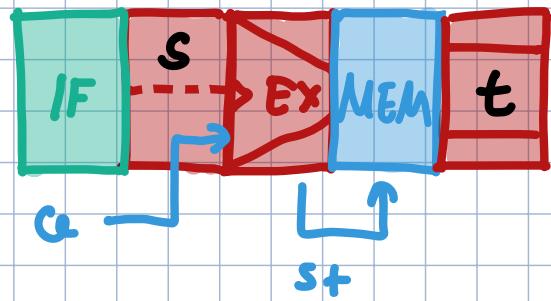


u wurde noch nicht gelesen

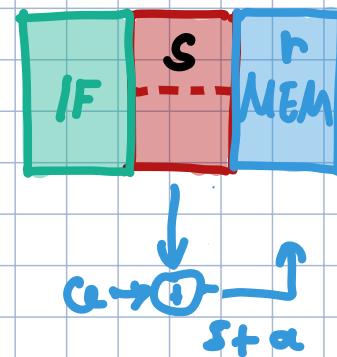
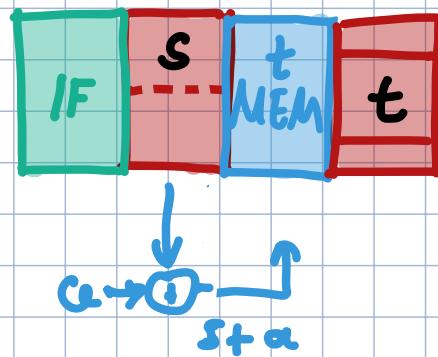
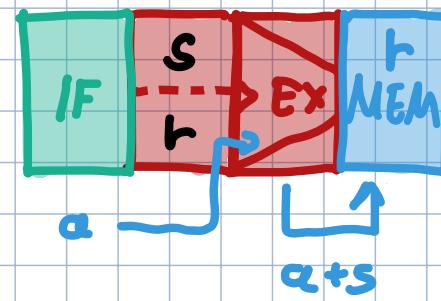
Diese Abh鋝igkeit ist aus der CDC 6000
bekannt (war da?)

Hilfband - Mechanismus : Besonderheit bei Load/Store mit Zwischenwerten

Load: $ldv \quad t \leftarrow s$



Store: $str \quad r \leftarrow s \alpha$

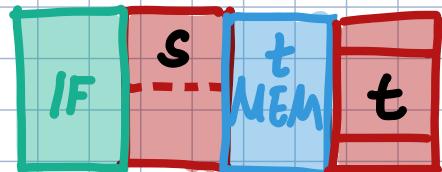


Separate Addressing

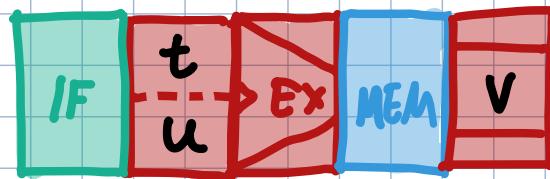
Flipboard - Heanise : Besonderheit bei Load/Store mit Zwischenwerten

Load :

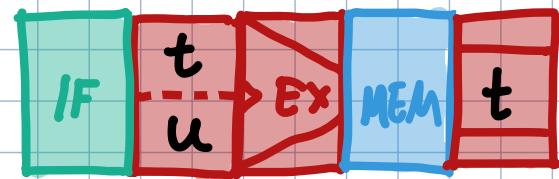
ldv t s a



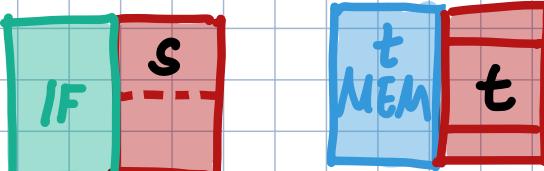
add v fu a



add f fu a

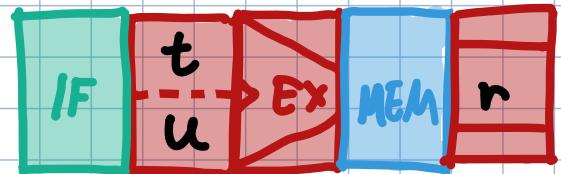


ldv t s a

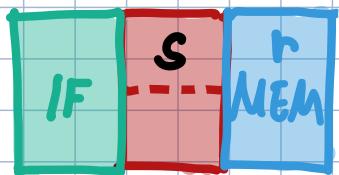


Store: stw \$s, \$r, a

add r ba

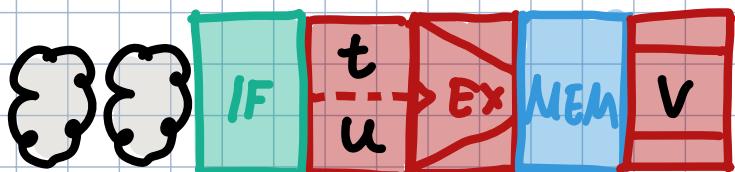


stw r sa



Vermeiden der Abhängigkeiten: Compiles

1) Ersatz von NOP-Befehl (Bubbles)



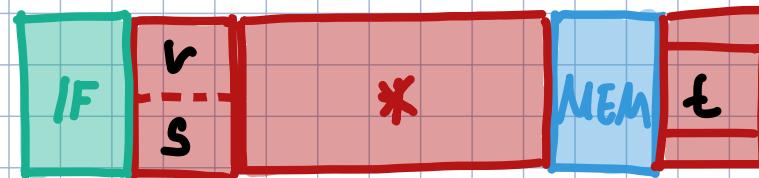
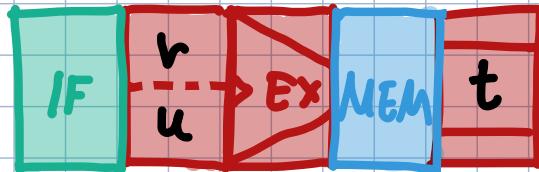
2) Umsortieren von Code

mul $t \leftarrow a$

ld $t \leftarrow q$

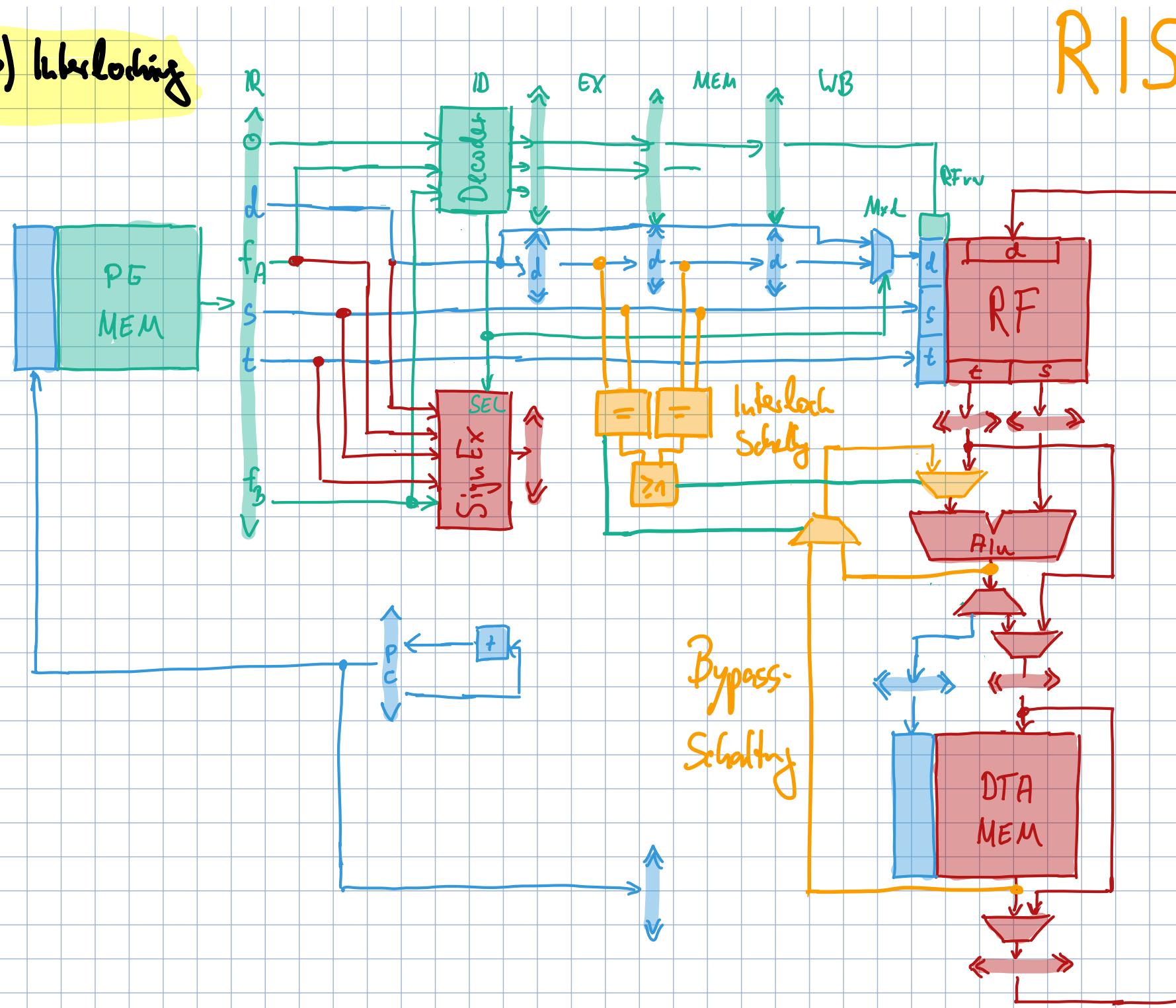
ld $t \leftarrow a$

mul $t \leftarrow b$



RISC V

3) Interlocking



 Sprungverhessage
 (Branch Prediction)

Kontrollfluss innerhalb Programms

Programmfluss

BB₁

add \$r, \$s, \$t

bne \$r, \$s, a

BB₂

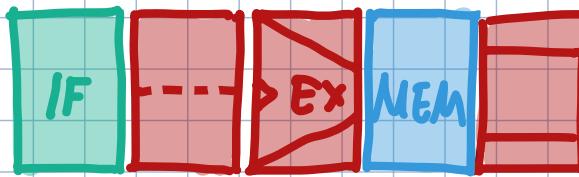
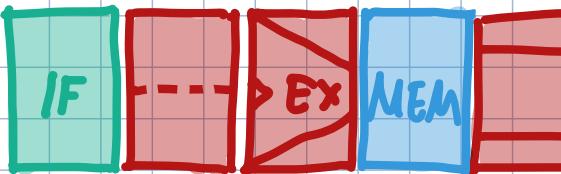
add \$r, \$s, \$t

sub \$u, \$v, \$w

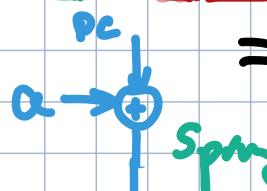
a mul \$u, \$v, \$x

add ...

BB = Basis Block



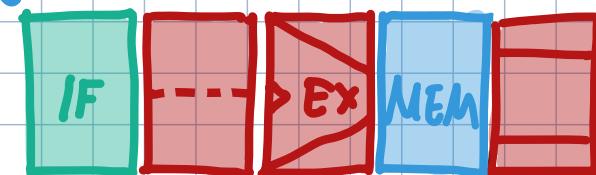
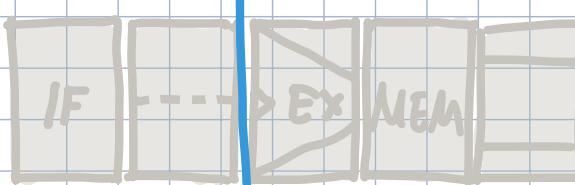
bne \$r, \$s, a



Sprung habe aus

nach Entscheidung
gefahrt wird

Pipeline Stall

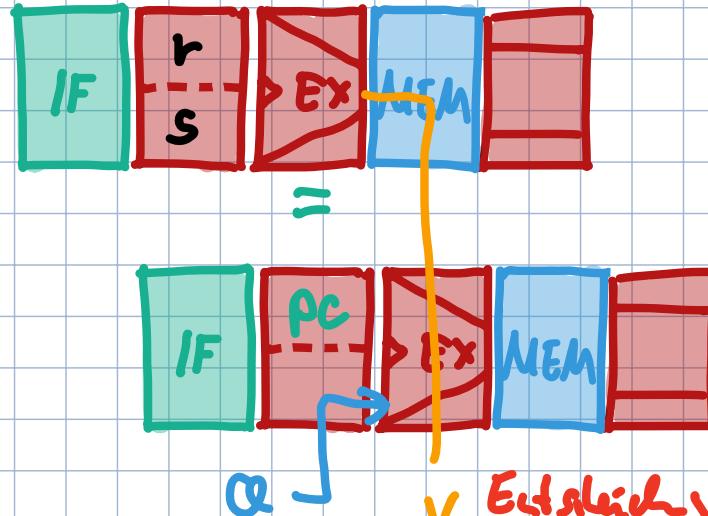


} Before
and in
Memory

Vorläufiger Sprung (Branch Delay Slot)

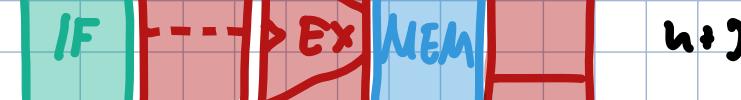
n add \$r, \$s, \$t
 n+1 cmp \$r, \$s
 n+2 bne a
 n+3 add \$r, \$s, \$t
 n+4 sub \$u, \$v, \$w
 n+5 add \$r, \$s, \$t
 a mul \$u, \$v, \$x
 add ...

Einspan des Adresses zu Berechnung des Sprungs



n+1

n+2



n+3

Nachfolgender Befehl wird - immer ausgeführt!



↑ Laden des nächsten Befehls

Sprungvorhersage (Branch Prediction)

Statische Sprungvorhersage (Static Branch Prediction)

Compiler sortiert Befehle in der richtigen Reihenfolge

Vorteil: Kein Aufruf in Hardware

Nachteil: Keine hohe Datendependenzberücksichtigung

Dynamische Sprungvorhersage

Für jede Adresse jeweils eines Sprungbefehls wird ein Zustand gespeichert

Taken 1 Not Taken 0

Vorteil: Berücksichtigt Datendependenz

Nachteil: Aufruf des Branch Predictor Buffers

Modell des Kontrollflusses

Basisblock - Graph

BB - Graph

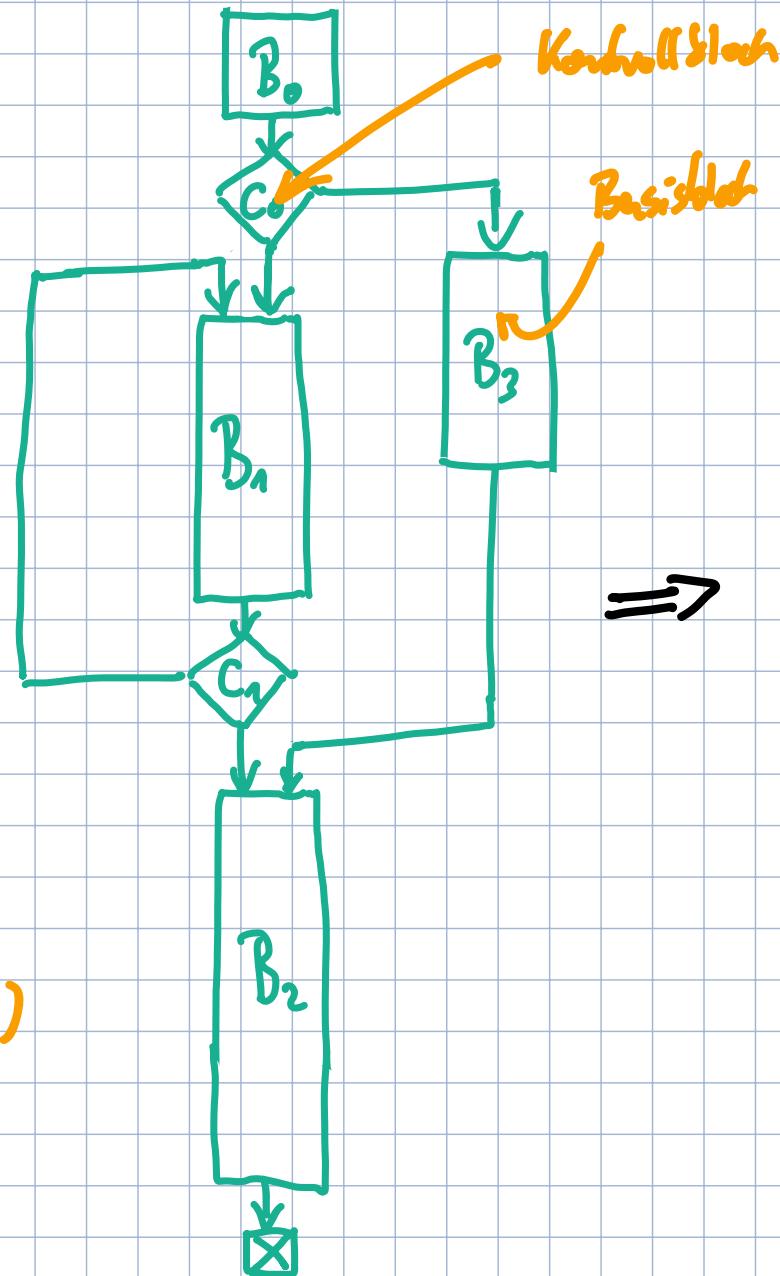
Def: Ein Basisblock ist ein Programmsegment (Folge von Befehlen), die keine Sprünge enthalten.

auch:

Basisblockgraph

Kontrollflusgraph (CFG)

Kontrollflusgraph



Programm im Spieler:

Statische Sprungvorhersage:

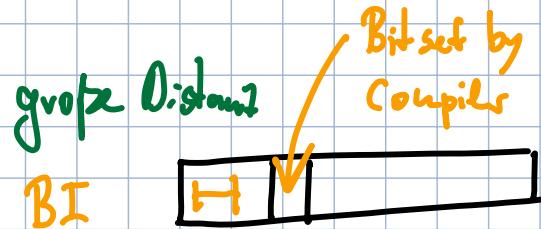
i) Programm wird linear ausgeführt

Vorteil: Einfach

Nachteil: Schleife ineffektiv

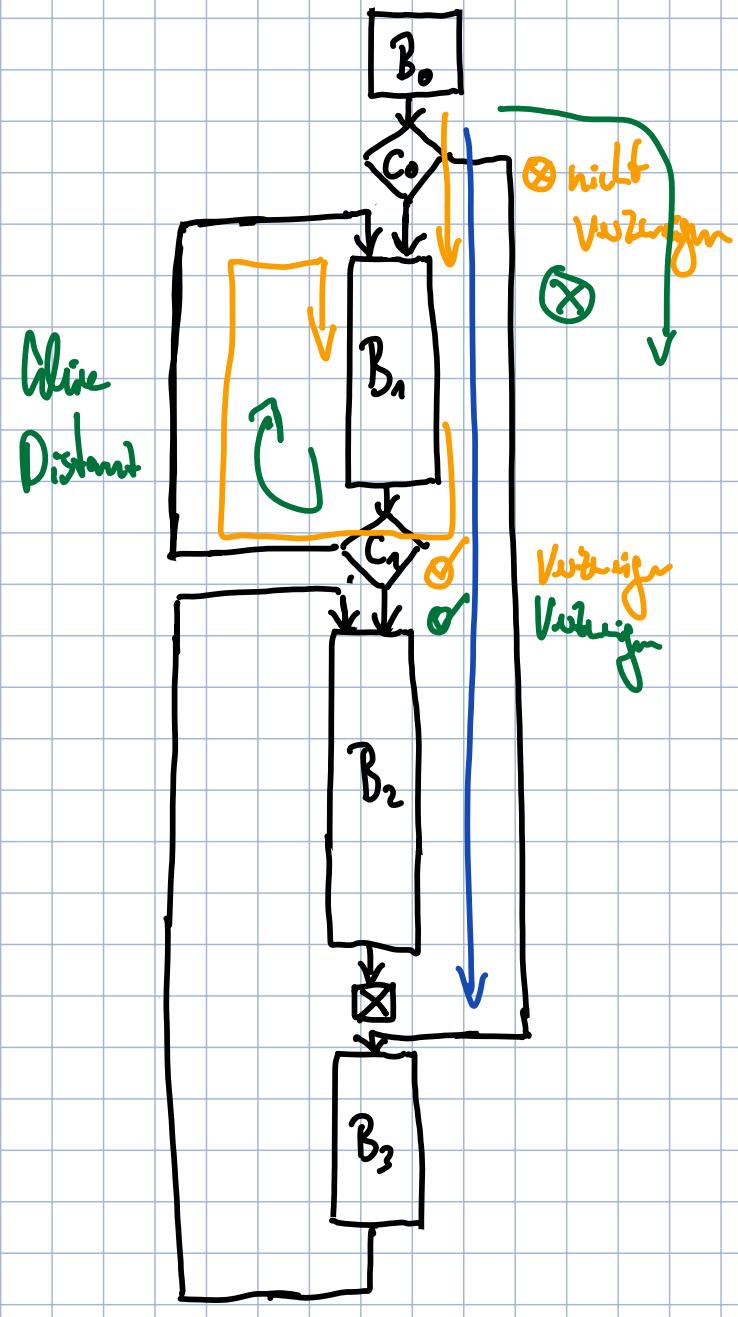
ii) Compiler bestimmt ob Verzweigt

wird oder nicht:



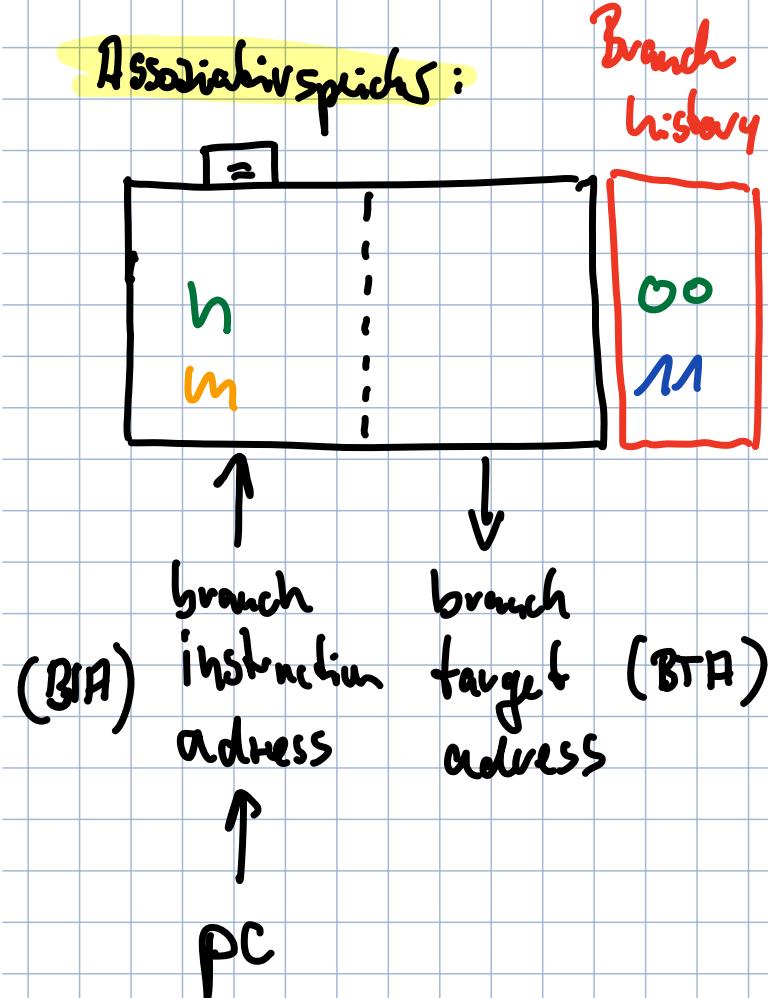
iii) Bestimmung der Distanz zum
Sprungziel

$|d| < \varepsilon$ Schleife!
Spring



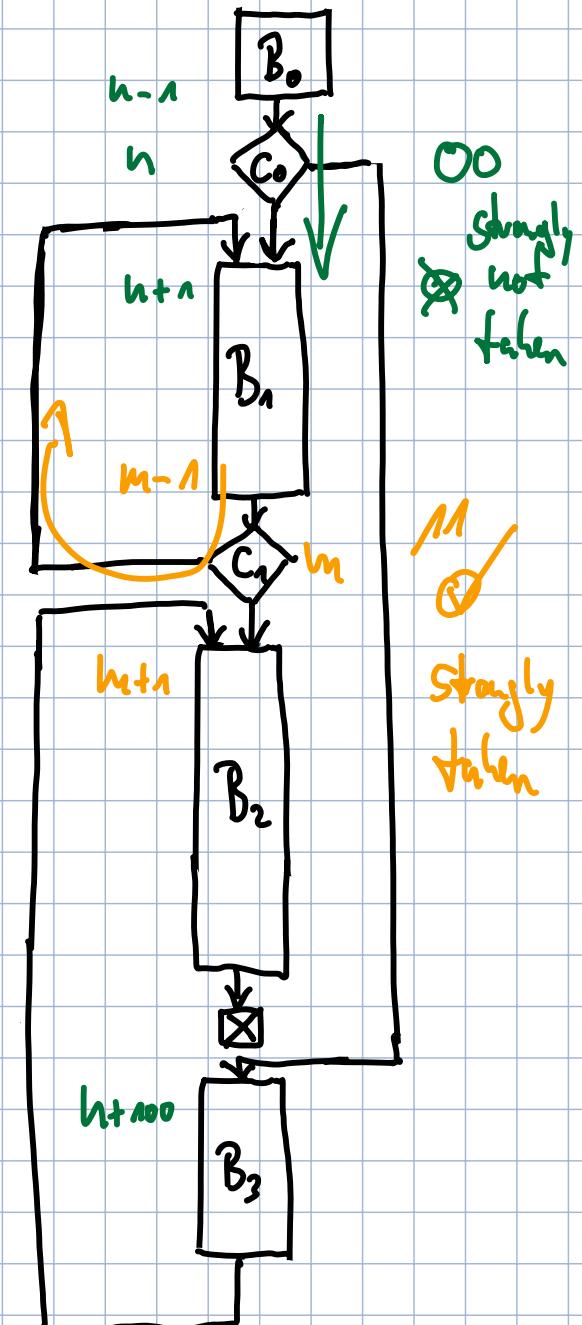
Dynamische Sprungvorhersage (Dynamic Branch Prediction)

Branch Target Buffer:



Mögliche Bushände:

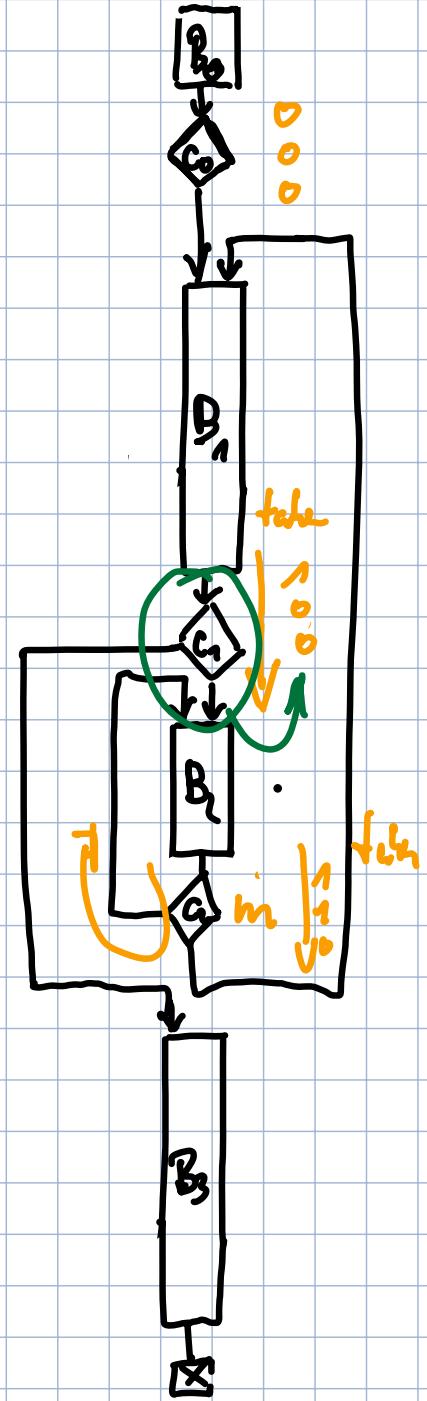
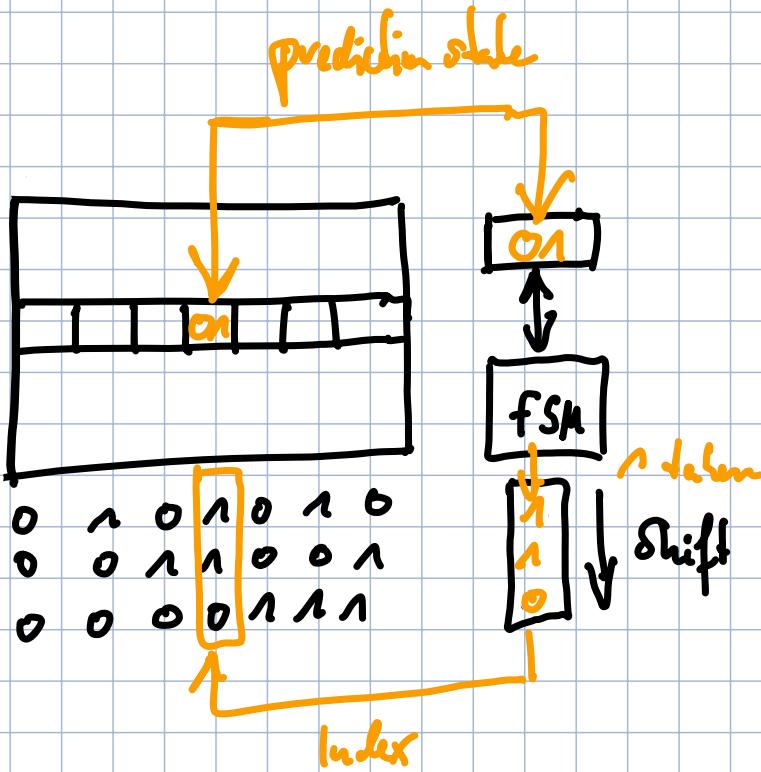
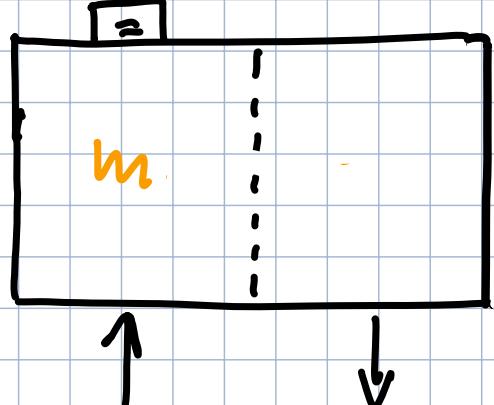
- 00 Strongly not taken
- 01 not taken
- 10 taken
- 11 Strongly taken



Hotdynamicische Sprungvorlage (95% Treffergenauigkeit)

(Correlated Branch Prediction)

Assoziativspeicher:

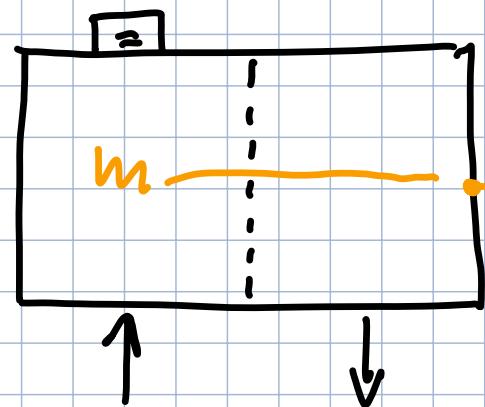


Bereitsrichtig des Programmflusses:

In den Slides vergessen nach die negativen
Treffer durch 'n' zu geprägt.

Hybridische Sprungvorlage (95% Trefferrate)

Assozierungspräzise:



Pattern history Table:

