22. 6. 12

weiter zu Erweiterung der Grundstruktur/ Stackadressierung und Stack

F3_110

spezielles Operandenregister Stack Pointer (Stapel Zeiger)

Bsp. 2. Schreiben, danach 2 mal Lesen

Stack (Speicher) im Speicherraum des Prozessors

1. Schreiben: SP:=Sp-1, Operand schreiben mit Adresse aus SP (Quelle: im Prozessor)
2. Schreiben SP:=SP-1, Op. schr. m. Adr. aus SP

1. Lesen Op. lesen m. Adr. aus SP, SP:=SP+1 (Beim Lesen nach ! dem Speicherzugriff)
2. Lesen: Op. le. m. Adr. aus SP, SP:=Sp-1

SP-bezogene Schreib- und Leseoperationen sollen „symmetrisch“ sein, d.h. im Mittel sind Anzahl Schreib- und Leseoperationen gleich.

Mehrere parallele ALE’s (F3_120)

Gleichzeitig mehrere Operationen auf Operanden ausführen.

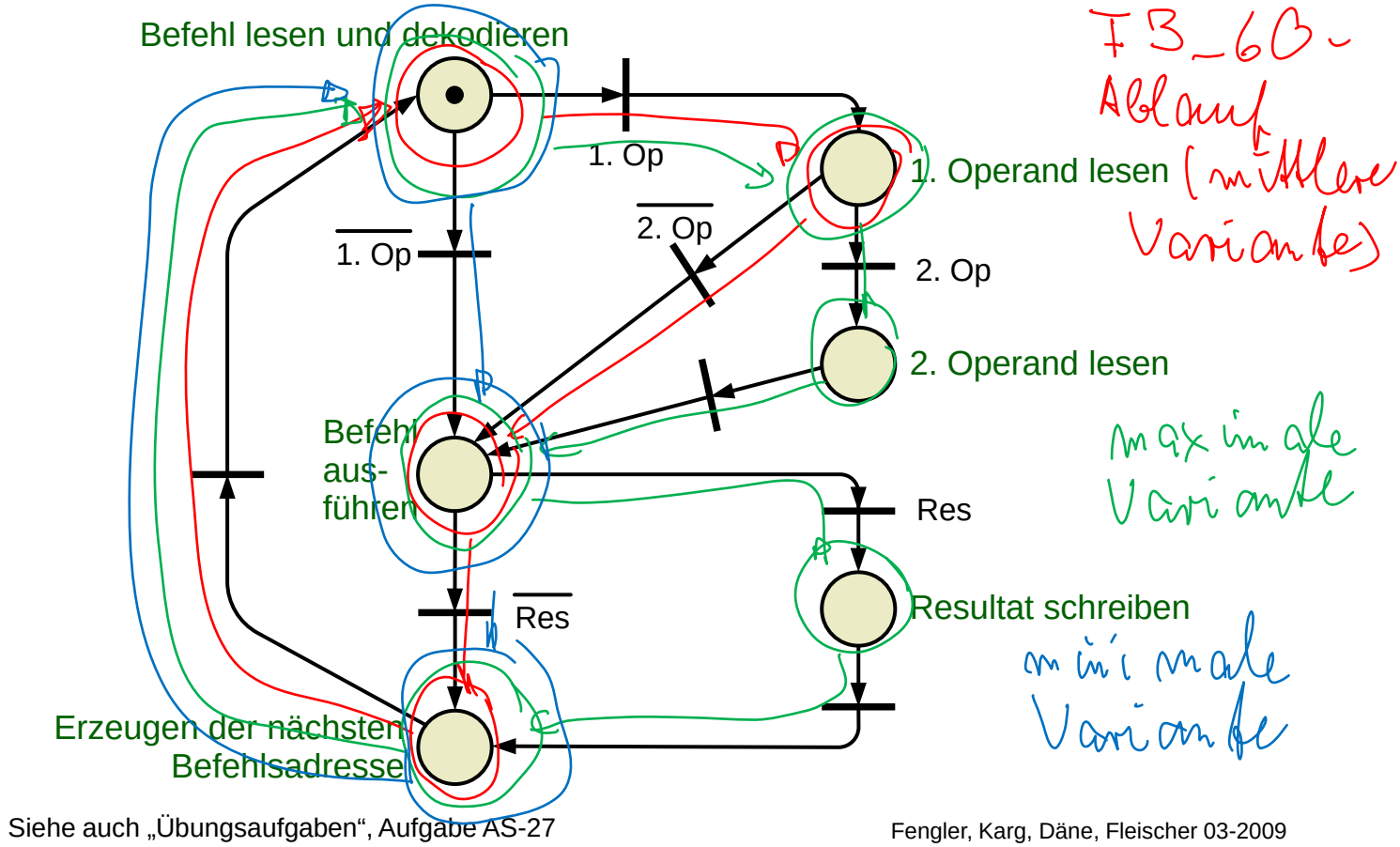
spezialisiert: die ALE’s haben unterschiedliche Funktionen (z.B. Festkomma- oder Gleitkommaarithmetik) -> kaum höherer Logikaufwand

- ➔ Parallelarbeit führt zur Leistungssteigerung (mehr Befehle werden in der selben Zeit abgearbeitet)
- ➔ evtl. auch gleiche ALE’s parallel

Verallgemeinerter Befehlsablauf (F3_130)

- ➔ durch Erweiterungen der Grundstruktur möglich
- ➔ F3_60 ist Spezialfall davon

Verallgemeinerter Befehlsablauf



Siehe auch „Übungsaufgaben“, Aufgabe AS-27

Fengler, Karg, Däne, Fleischer 03-2009

Zeitverbrauch (1. Näherung, je Phase (1 PN-Platz) gleiche Zeit tph):

min. Variante: 3 * tph
mittlere Var.: 4 * tph
max. Variante: 6* tph

3.3.4. Befehlsübersicht

➔ Maschinenbefehle: Befehle, die der Prozessor direkt dekodieren und ausführen kann

➔ sprachliches Äquivalent: Assemblerbefehle (1 Ass.bef. entspricht 1 Maschinenbef.)

➔ Prozessoren haben unterschiedliche Maschinenbefehlssätze. Gründe: unterschiedliche Einsatzgebiete, Hersteller, Verträglichkeitsforderungen (Kompatibilität) …, evtl. anwendungsbezogen variabel (siehe auch RA2 später)

hier Übersicht, in der Übung Ausschnitte aus einem konkreten Befehlssatz (siehe Arbeitsblätter usw.)

Übersicht siehe F3_140

hier nur allgemeine Besonderheiten, konkret sie Arb.bl. + Übungsstoff

3.4.1. Datentransfer

- Vor- und Nachbereitung von Operationen durch Ortsveränderung von Operanden
- Ziel: wo ist der Operand nach dem Bef., Quelle: wo ist der O. vor dem Bef.
- i.a. Kopieren Quelle überschreibt Ziel, Quelle bleibt erhalten
- verschiedene Kombinationsmöglichkeiten von Ziel und Quelle:

(im Weiteren immer alle Varianten von Befehlen einschließlich der Existenz von Befehlen ist Prozessortyp-abhängig)

Register (ORi, OAi, SP, Basis-Reg…, PSR)

Datenworte im Speicher (adressiert durch OA, SP, im Befehl direkt usw.)

Ein-Ausgabeschnittstellen

3.4.2. Arithmetik-Logik

Operandenbeeinflussung bzw. Op.auswertung in den Maschinendatentypen (s. RO, welche existieren)

max. 2 E-Op. max 1 A-Op., PSR nach Operation

Arithmetische Grundoperationen:

- Addition, Subtraktion
- Multiplikation, Division
- weitere Operationen als Teilprogramme (z.B. in verfügbarer Bibliothek)
- das gilt auch für mehr als zwei E-Operanden, mehr als 1 A-Operand

Logische Grundoperationen:

- UND
- ODER
- Exklusiv ODER (Antivalenz)
- Negation

Datenformat: F3_150

➔ Operation erfolgt immer auf n bit (n Anzahl der bit im Datenwort, gleich bit-Position (letzteres gilt auch für Ergebnis)

➔ Einzelbitoperationen sind Prozessorabhöngig evtl vorhanden (Test Einzelbit (biti -> negiert z-bit vom PSR), Setzen, Rücksetzen von Einzelbit auf 1 bzw. 0)

Vergleich: Subtraktion ohne Ergebnis aber PSR-Belegung

mit Z-bit und S-bit (zereo, sign): Test auf gleich, kleiner, größer, kleiner-gleich, größer-gleich)

Konvertierungen zischen verschiedenen Zahlenformaten

Schieben und rotieren von bit-Positionen im Datenwort: Schieben-Bsp. 3_160, Rotieren Bsp.3_170

möglich nach rechts, links, m bit, mit m<=n (n Breite Datenwort)

- ➔ logische Interpretation: Vorbereitung von logischen Operationen, oder deren Nachbereitung
- ➔ arith. Interpretation Multiplikation (schieben links) und Division (schieben rechts mit 2 hoch m (bei schieben um m Positionen)

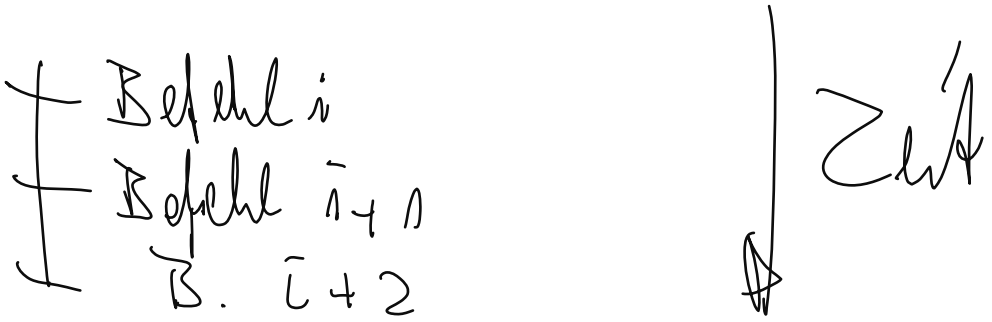
Bsp. 3_170

- ➔ im Unterschied zu Schieben (dort entfallen heraus geschobene bit) werden die herausgeschobenen bit wieder auf der anderen Seite eingeschoben (keine arithmetische Interpretation möglich)

3.4.4. Programmtransferbefehle

bisher EZNB: BA:=BA+1

- ➔ damit: rein lineare Befehlsfolgen:



für normal komplexe Algorithmen unbedingt notwendig:

Verzweigung (evtl in Abhängigkeit von Bedingungen) und Zusammenführung (F3_180 links) -> Sprungbefehle notwendig

nächste Woche: weiter mit Sprung- Unterprogrammbefehlen, Interrupt.