

weiter vom 20. 6. 12

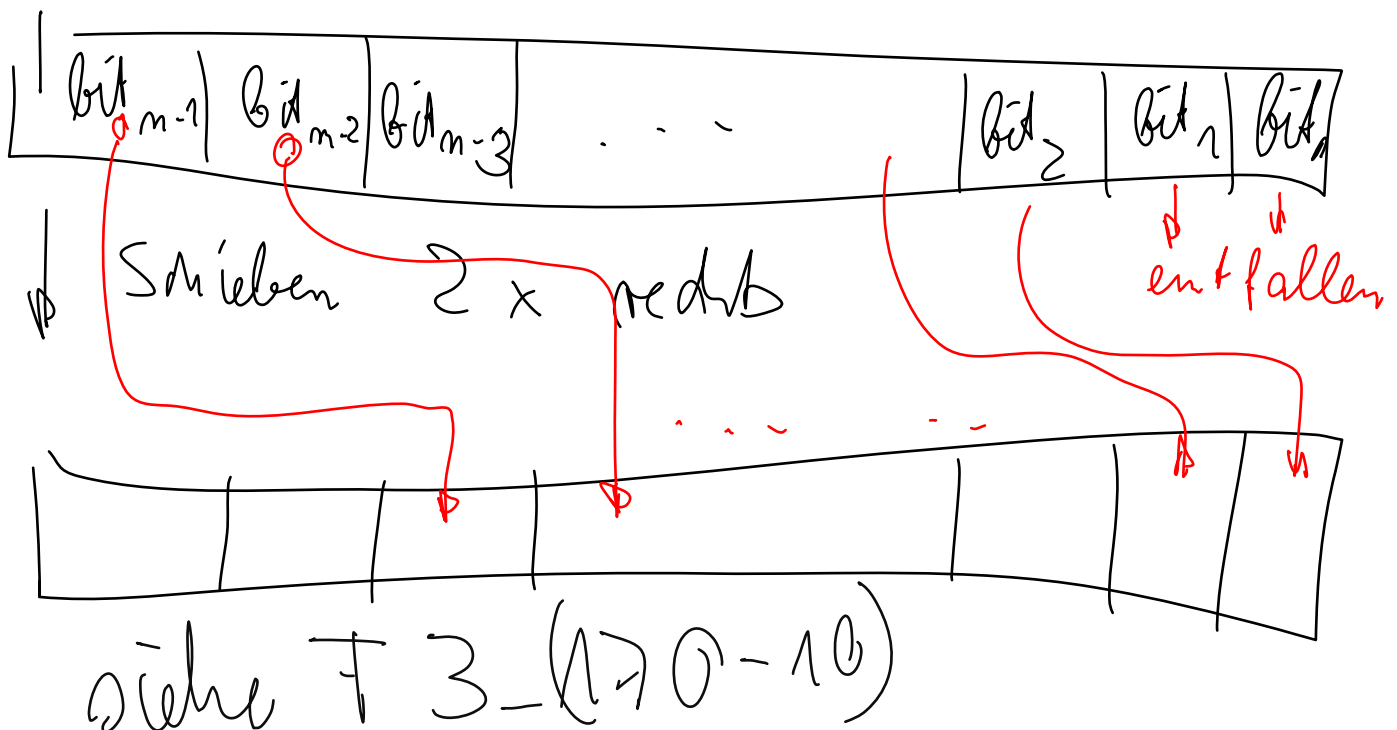
Bitmanipulation und Schieben

Einzelne bit im n-bit-Datenwort auf 0 oder 1 verändern
(Bitrücksetzen, Bitsetzen)

Schieben und rotieren

Veränderung von bit-Positionen im Datenwort

z.B. Schieben zweimal rechts:



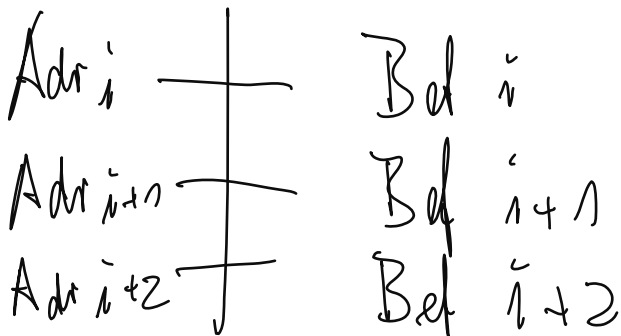
arithmetische Interpretation (Mult. /Div. mit Zweierpotenzen)

logische Interpretation: Vor- oder Nachbereiten von logischen Operationen (bit-Positionen so verändern, wie für die log. Op. notwendig)

Unterschied Rotieren zu Schieben (Bsp. 3_170): entfallende bit-Positionen werden auf der anderen Seite wieder eingeschoben, keine arithmetische Interpretation möglich

Bisher Erzeugen nächste Befehlsadresse (ENBA) durch
 $BA := BA + 1$

damit sind rein lineare sequentielle Befehlsfolgen möglich



für normale Algorithmen sind Abweichungen von dieser linearen sequentiellen Folge notwendig:

- ENBA nicht durch $+1$
- Sprünge, Unterprogrammbeefehle, Interruptmechanismus

Sprünge:

1. Möglichkeit: Weiterarbeit mit $BA := BA + 1$
2. Möglichkeit: Weiterarbeit mit $BA :=$ vorgegebener Adresse

unbedingt: immer 2. Möglichkeit

bedingt: 1. Möglichkeit bei nicht erfüllter Bedingung

2 . Möglichkeit bei erfüllter Bedingung

Möglichkeiten für vorgegebene Adresse:

1. fester Wert (evtl. Basisadressiert) (absolut)
2. Wert:= aktueller Wert + Distanz (relativ)
3. Wert:= Inhalt eines Registers (eventuell speziell berechnet) (indirekt)

Kombinationen möglich:

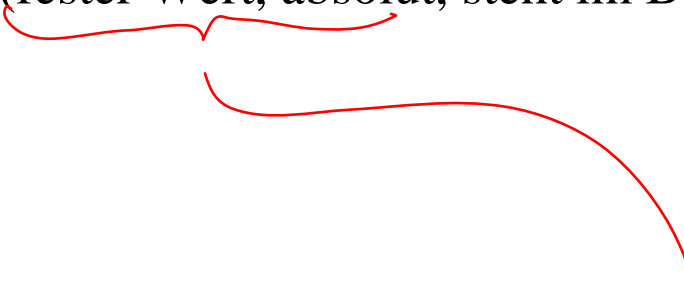
z.B. bedingt, relativ


Bedingungen sind typ. PSR-bits

Unterprogramm: mehrfach nutzbares Teilprogramm, wobei die Rückkehr nach Ende dieses Teilprogramms auf den Befehl erfolgt, der nach dem aufrufenden Befehl im Speicher steht. (F3_180 rechts)

➔ Speicherung der „Rückkehradresse“ -> im Stack

Ablauf UP-Aufruf

1. BL CALL (UP-Ruf-Befehl)
 2. OL mit $BA:=BA+1$ (fester Wert, absolut, steht im Befehlscode nach dem CALL)
 3. $BA:=BA+1$
- 

4. Stackschreiben Inhalt von BA (Rückkehradresse) mit der SP-Manipulation nach LIFO-Prinzip
 5. ENBA mit gelesenen festen absolutem Wert
- 

Ablauf UP-Rückkehr

1. BL RET (Return, UP-Rückkehrbefehl)
2. Stacklesen in ENBA mit SP-Manipulation nach LIFO-Prinzip

können wie die Sprünge in Zusammenhang mit Bedingungen und Varianten zur Adresswertvorgabe auftreten.

Verschachtelter Aufruf von UP möglich:

d.h im UP wieder ein UP aufrufen (aufgrund des LIFO-Stackprinzips (F3_190))

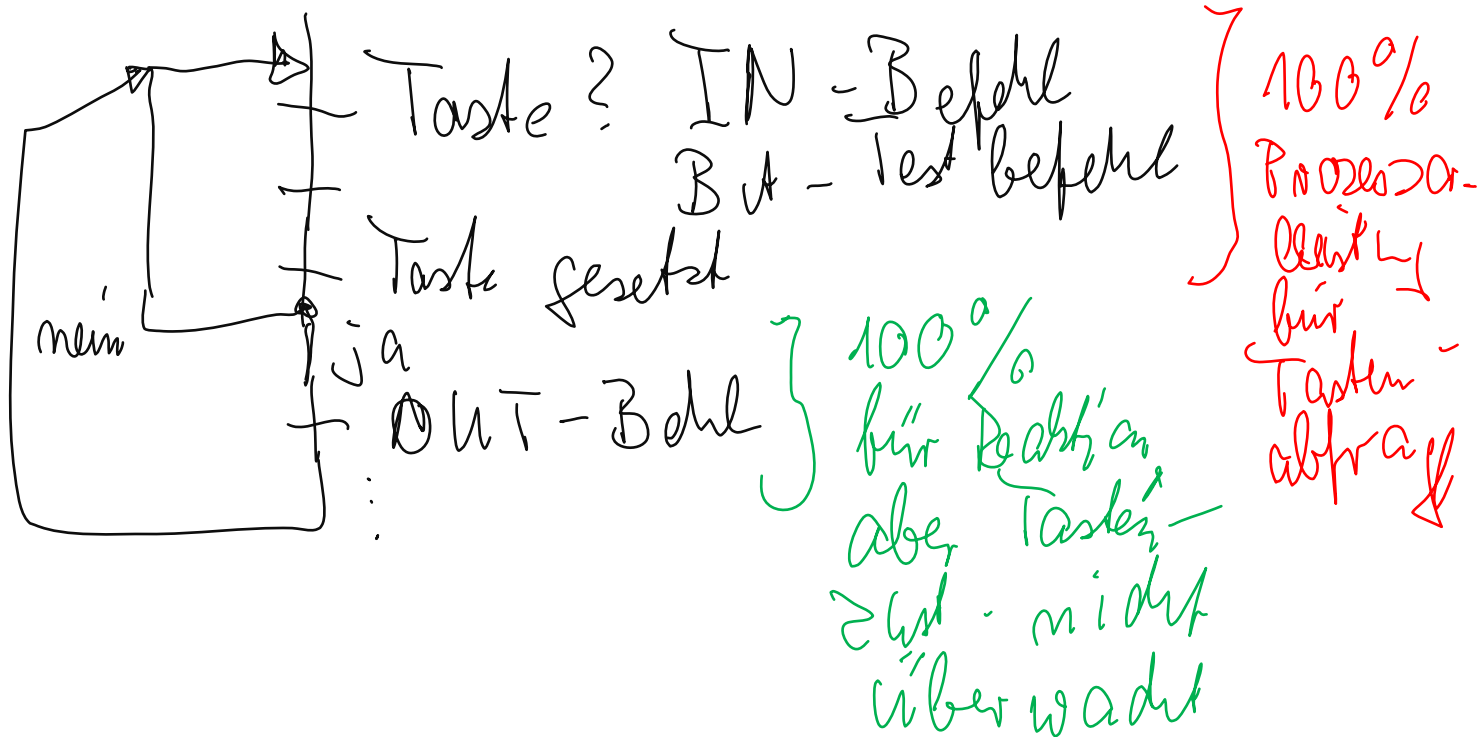
Interrupt (Programmunterbrechung):

➔ programmtechnische Reaktion auf ein zeitlich nicht vorhersehbares ext. Ereignis.

Bsp.:

Taste -> Betätigung -> Ausgabe *(Reaktion auf Taste)*

Mit bedingten Sprüngen:



Effektiver mit Interrupt (F3_200):

Prozessor arbeitet solange in einem Hauptprogramm, bis die Taste betätigt wird (z.B. längere Berechnung)

dabei wird die Taste nicht ständig abgefragt!

Tastenbetätigung führt zu:

Stopp Hauptprogramm nach aktuellem Befehl -> Abarbeitung eines der Taste zugeordneten Interruptprogramms

Am Ende des Interruptprogramms Rückkehr in das Hauptprogramm an die Stelle nach dem dort aktuell ausgeführten letzten Befehls

Ablauf dazu: F3_207

Interruptcontroller: Zusatzhardware zur (erweiterten) Prozessorgrundstruktur

Eingänge für auf binare Signale abgebildete Ereignisse (z.B. Taste)

Ereignis: ①

Mitteilung an Prozessor: es existiert ein Ereignis (noch nicht, welches) ②, ③

nach Bef. k vom Hauptprogramm – Prozessor fordert vom IC an, welches Ereignis (welcher Interrupt) ④

IC liefert Int-Nr. der Taste (Int. i) ⑤

Prozessor liest aus einer Tabelle im Speicher (die enthält die Startadressen aller den Interrupteingängen zugeordneten Ereignisreaktionsprogramme) die Startadresse für Reaktion auf die Taste ⑥, ⑦

Rückkehradresse (Adr. von Bef. k+1 des Hauptprogramms) in den Stack (zusätzlich aktuelle PSR-Belegung im Hauptprogramm) ⑧

Sprung auf die Startadresse ⑧

Abarbeiten des Reaktionsprogramms ⑨

Rückkehradresse (zusätzlich die PSR-Belegung) aus dem Stack und Sprung auf die Rückkehradresse ⑩

Warum PSR-Belegung bei Innterrupt in den Stack?

- ➔ F3_210
- ➔ im Hauptprogram ist akt. Befehl (Bef.k) ein Befehl, der das PSR setzt
- ➔ danach Interruptprogramm -> verändert PSR
- ➔ ohne PSR-Speicherung würde der Bef. k+1 im Hauptprogramm, der hier das PSR auswertet mit einem ungültigen PSR-Wert arbeiten

weiter mit Kap. 4. Speicher