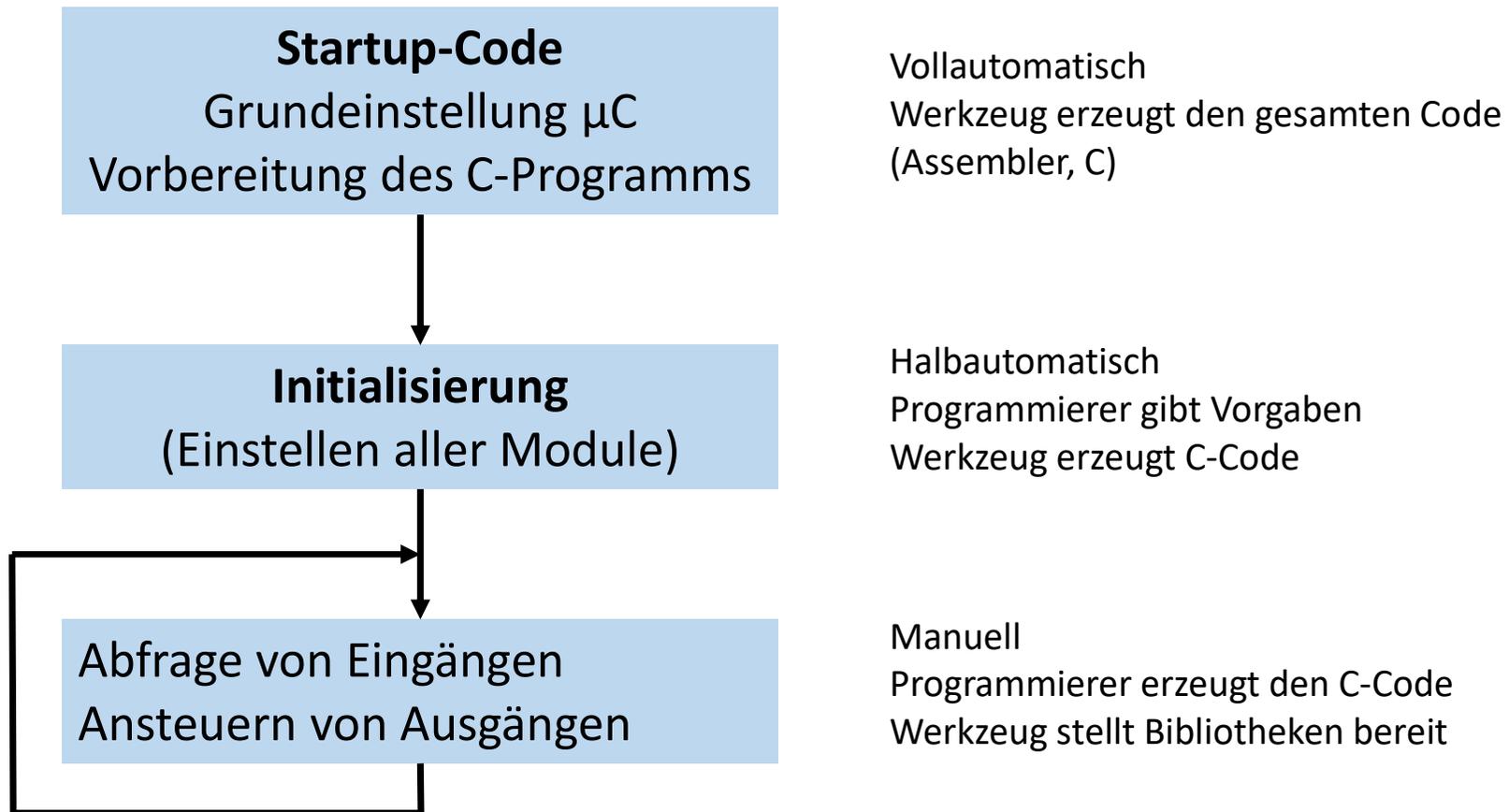


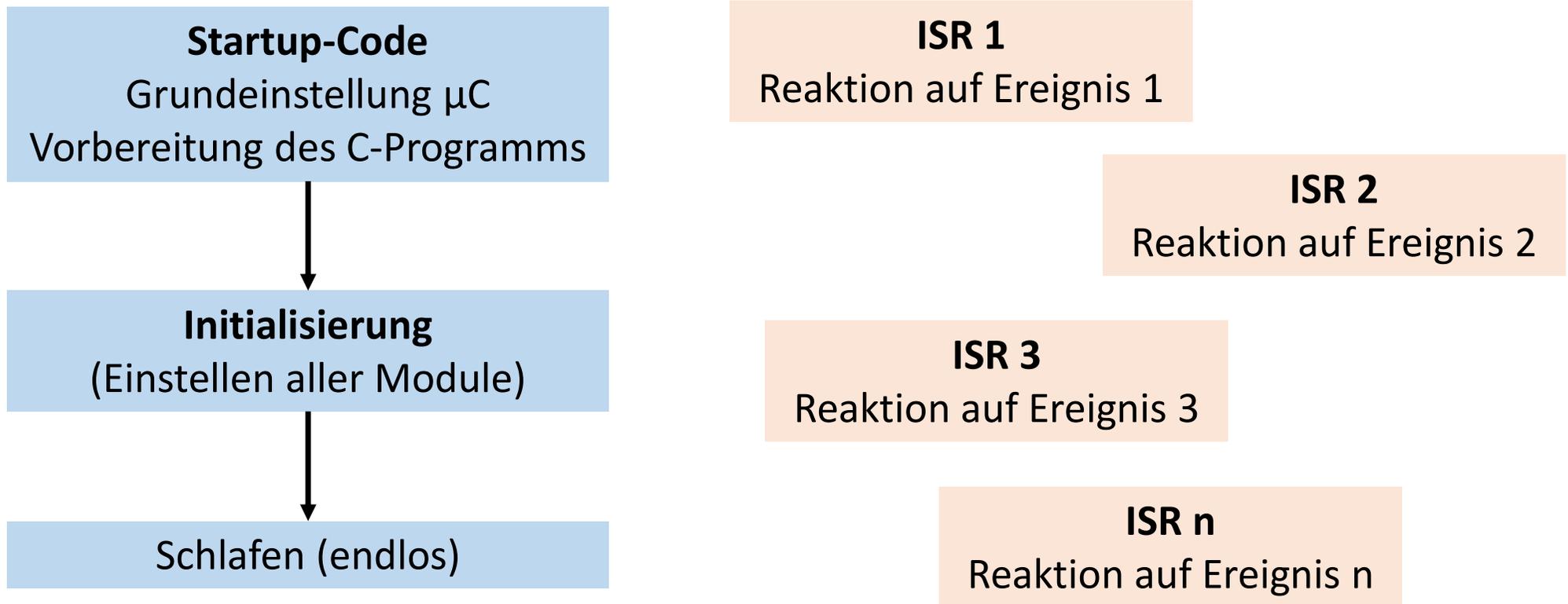
Aufbau eines μ C-Programms

(Version 1: Ohne Betriebssystem, ohne Interrupts)



Aufbau eines μ C-Programms

(Version 1: Ohne Betriebssystem, mit Interrupts)



Interrupts 1

- **Interrupt**

- stellt eine **Unterbrechung** des aktuellen Programmflusses dar
- nach der Unterbrechung wird das unterbrochene Programmfluss fortgesetzt
- **asynchron**: tritt zu einem **unvorhersehbaren Zeitpunkt** auf
- synchron: wird durch den Programmfluss selbst ausgelöst

- **Arbeitsmodell**

- Langlaufender, **zeitunkritischer** Anteil -> main()
- **Kurze**, aber **zeitkritische Aktionen als Reaktion** auf Ereignisse -> ISR()
- **Keine Verschwendung** von Energie und Rechenzeit durch ständiges Nachsehen, ob etwas zu tun ist (Polling)

Interrupts 2

- **Ablauf, Teil 1 (bei jedem Takt)**
 - Wenn Ereignis x eintritt, dann wird ein **Flag** x gesetzt
- **Ablauf, Teil 2 (vor jedem Befehl, für alle Flags)**
 1. Sammle alle **Flags** i, die gesetzt sind und deren **Maske** i offen ist
 2. Falls Menge leer: mache im Programm wie vorgesehen weiter
 3. Sonst: Bestimme das Flag j mit der höchsten Priorität
 4. Merke aktuellen Programmzustand
 5. Führe **ISR** j aus
- **Ablauf, Teil 3 (ISR x)**
 1. Setze Flag x zurück
 2. Bearbeite Ereignis
 3. Kehre zum unterbrochenen Programmfluss zurück

Interrupts 3

Flag

Ein Flag ist ein Ereignismerker.

Es wird automatisch beim Eintritt eines Ereignisses gesetzt.

Für welche Ereignisse es Flags gibt, wird vom Hersteller des μC bestimmt.

Es muss ggf. manuell (im Programm) zurückgesetzt werden.

Maske

Eine Maske ist eine Schranke für ein oder mehrere Flags

Ist die Schranke geschlossen, passiert weiter nichts.

Ist die Schranke offen, wird die zum Flag gehörende ISR aufgerufen.

ISR Interrupt Service Routine

Ist eine ganz normale Funktion, die aber nicht im normalen Programmablauf von einer anderen Funktion aufgerufen wird.

Zweck: Bearbeitung eines Ereignisses

Soll zeitlich (vom Beginn bis zum Ende) so kurz wie möglich gehalten werden

Interrupts 4

Priorität

Legt fest, welche ISR ausgeführt wird, wenn mehrere Flags mit offener Maske gleichzeitig gesetzt sind.

Prioritäten können fest vom Hersteller vorgegeben sein oder vom Programmierer zur Laufzeit festgelegt werden. Mischformen sind üblich.

Vektor

Adresse der ISR x , die zum Flag x gehört. Wird heute automatisch vom Werkzeug (Linker) richtig bestimmt und ins Programm eingetragen. Kein Eingriff nötig

Nested Interrupts (Verschachtelung)

Eine gerade laufende ISR wird von einer anderen ISR mit noch höherer Priorität unterbrochen. Wer wen unterbrechen kann, wird über Masken und/oder Prioritätsebenen bestimmt. Das kann der Programmierer zur Laufzeit festlegen

Interrupts 5

- **Prioritäten**

- 1. Wer kann wen unterbrechen?**

- durch Masken gesteuert (Nutzer muss das machen)
- Prioritätsebenen (typisch 4-8 Ebenen, Zuweisung durch Nutzer)

- 2. Wer kommt bei Gleichzeitigkeit zuerst dran?**

- Feste Einstellung (Hersteller)
- Programmierbar (ggf. begrenzt, z.B. 4-8 Stufen, Zuweisung durch Nutzer)