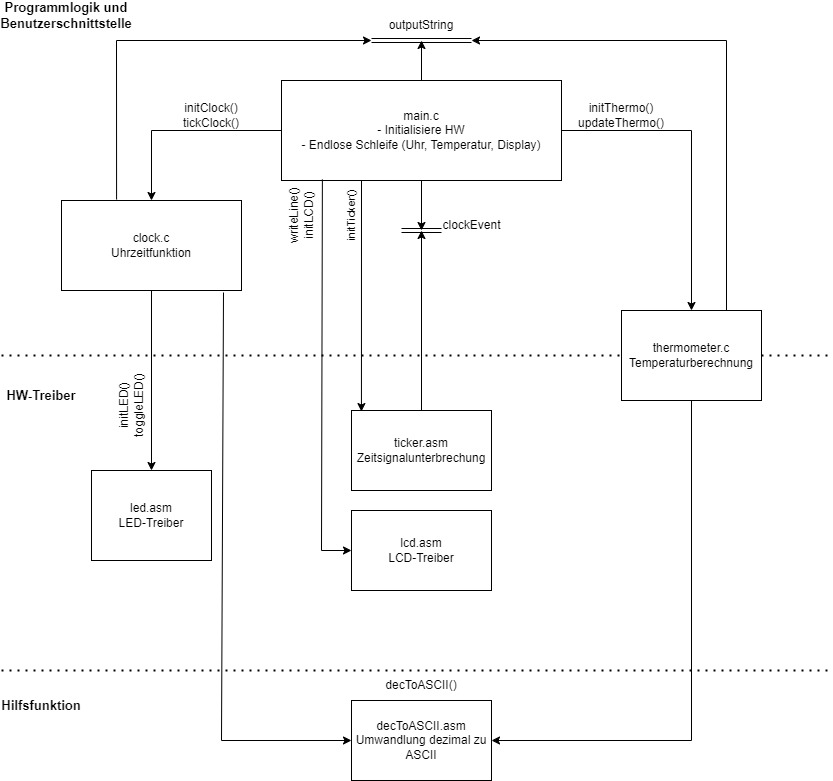
1. **Funktionale Voraussetzungen**
   1. **Anforderungen an die Uhr (Radio kontrollierte Uhr DCF77):**

* Die aktuelle Zeit muss im Format HH:MM:SS auf dem LCD Display dargestellt werden.
* Aktualisierung der Zeit jede Sekunde durch time Clock
* Aktualisierung der Zeit jede Minute durch DCF77
* Falls die Informationen der DCF77 fehlschlägt, soll Uhrzeit der time Clock verwendet werden
* Die LED an Ausgang B.0 muss einmal pro Sekunde durch time Clock umgeschalten werden
* Die LED an Ausgang B.1 muss an sein, wenn das DCF77 Signal niedrig ist und aus wenn das DCF77 Signal hoch ist. Wenn das komplette DCF Signal angekommen ist, soll auch diese LED einmal pro Sekunde umschalten
* Die LED an Ausgang B.2 muss an sein, wenn ein Fehler im Zeitformat durch Paritätsbitüberprüfung festgestellt wurde, solange bis gültige Informationen vorliegen
* Die LED an Ausgang B.3 muss an sein, wenn ein vollständiges und korrektes DCF77 Datum und Zeitinformation entschlüsselt wurde und aus sein wenn keine oder unvollständige Informationen angekommen sind
* Das DCF77 Signal ist über den Ausgang H.0 verbunden
  1. **Anforderungen an das LCD Display ( Darstellung ):**
* Die erste Zeile des LCD Displays wechselt jede 10 Sekunden zwischen „© IT W2023/24“ und „F.Fink T.Mencin “
* Die zweite Zeile wird folgendermaßen Dargestellt: Linksbündig „Uhrzeit“ Rechtsbündig „Temperatur in Grad Celsius“ wobei Zahlen die kleiner als 10 sind ohne führende Null dargestellt werden. Positive Werte sollen ohne Vorzeichen dargestellt werden
  1. **Sonstige Anforderungen:**
* Das Modul „ticker.asm“ aus Aufgabe 2.1 muss zur Taktgenerierung benutzt werden
* Für LCD und LED müssen die Module „LCD.asm“, „decToASCII.asm“ und „LED.asm“ benutzt werden
* Das Programm muss in verschiedene Module aufgeteilt werden. „Normaler Modus“ und „Setzen Modus“ müssen separat im Programm programmiert werden
* Die main() Funktion soll so kurz wie möglich sein und aus der Initialisierung der Hardware Komponenten bestehen, sowie in einer Schleife ausgeführt werden, welche die Uhr und die Temperatur jede Sekunde aufruft
* Die Unterbrechungsroutine soll sich nur mit der Unterbrechungshardware beschäftigen und nicht komplexe Routinen ausführen
* Die Kommunikation zwischen den Unterbrechungsroutinen und den anderen Teilen des Programms erfolgt über globale Variablen
* Um den Status der Knöpfe abzufragen, muss regelmäßiges Abfragen benutzt werden, keine Unterbrechungen

1. **Die Benutzerschnittstelle des Programms**
   1. **LCD Display**

* Erste Zeile: „© IT W2023/24“ oder „F.Fink T.Mencin “ (Wechsel alle 10s)
* Zweite Zeile: „HH:MM:SS“ (Linksbündig) „Temperatur in Grad Celsius“ (Rechtsbündig)
  1. **LEDs**
* LED0: Wechsel einmal pro Sekunde
* LED7: Aus wenn Modus „Normal“, An wenn Modus „Setzen“
  1. **Knöpfe**
* SW2: Wechsel zwischen Modus „Normal“ und „Setzen“
* SW3: Erhöhung der Stunden (Modus „Setzen“)
* SW4: Erhöhung der Minuten (Modus „Setzen“)
* SW5: Erhöhung der Sekunden (Modus „Setzen“)

1. **Modulüberblick**

****

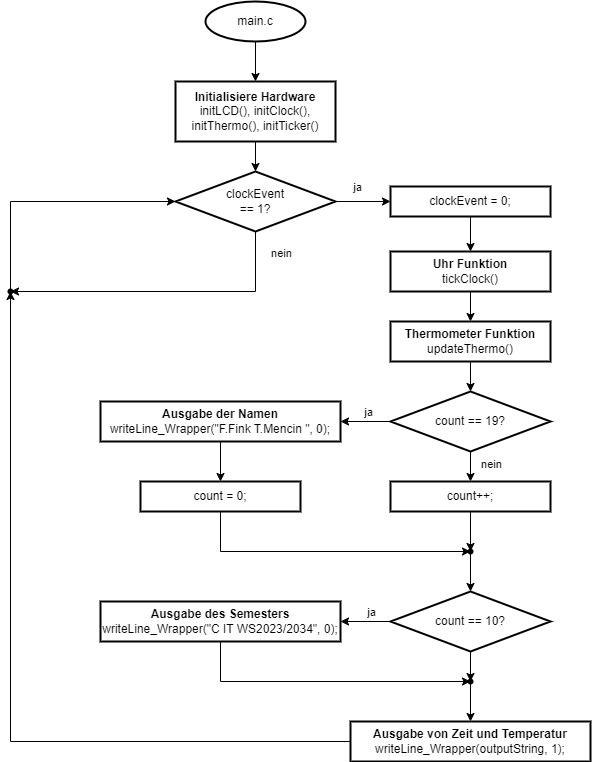
1. **Datenstruktur**
   1. **Globale Variablen**

|  |  |  |  |
| --- | --- | --- | --- |
| **Modul** | **Variablenname** | **C Datentyp** | **Zweck** |
| main [ticker] | clockEvent | unsigned char (0...1) | Regelmäßiges Anzeigen, das eine Sekunde rum ist |
| main | count | unsigned char (0...19) | Wechsel der ersten Zeile alle 10 Sekunden |
| wrappers [main, thermometer, clock] | outputString | char [17] | Speicherung der zweiten Ausgabezeile |
| clock | time.hour | char (struct) (0...23) | Stunden |
| clock | time.minute | char (struct) (0...59) | Minuten |
| clock | time.second | char (struct) (0...59) | Sekunden |
| clock | clockMode | enum (NORMAL,SET) | Speicherung des Uhrmodus |
| dcf77 | dataBits | char[59] | Speicherung der erhaltenen dcf77 Daten Bits |
| dcf77 | bitCount | unsigned char  (0...1) | Speicherung des Decoder Bits |
| clock | timeZone | unsigned char (0...1) | Speicherung der Zeitzone (EU,US) |
| clock | clockEvent | CLOCKEVENT | Speicherung des letzten Clock Events |
| dcf77 | decoderState | enum  {DATAINVALID, WAITFORPOSEDGE, WAITFORNEGEDGE} | Speicherung des Decoder Status |

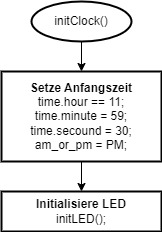
* 1. **Hardware Ressourcen**

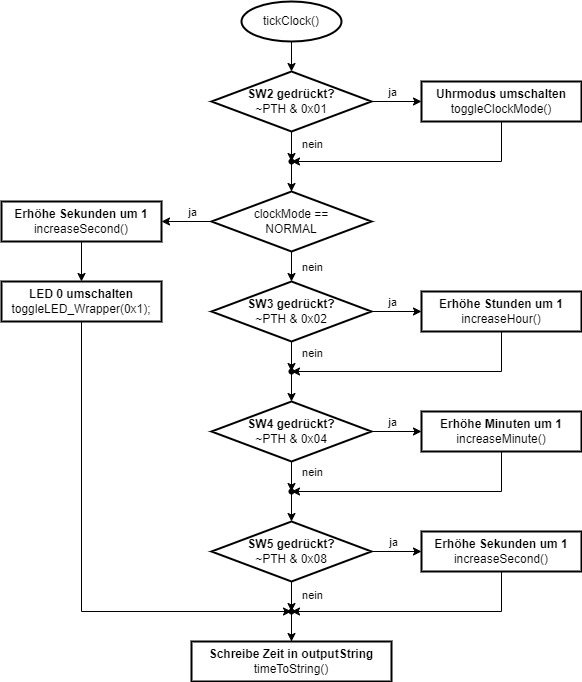
|  |  |  |
| --- | --- | --- |
| **Module** | **HCS12 or Dragon12 HW resource** | **Purpose** |
| LCD  [Main] | LCD Anzeige | Erste Zeile: „© IT W2023/24“ oder „F.Fink T.Mencin “  Zweite Zeile: Zeit & Temperatur |
| LED  [Clock] | LEDs | Statussignale |
| Clock | Knöpfe SW2-SW5 | Setzen der Zeit/Modus |
| Ticker  [Main] | Timer  Kanal 4 | 10 ms Ticker |

1. **Flussdiagram für alle Module**
   1. **Main**

****

* 1. **Clock**

****



* 1. **Thermometer**

