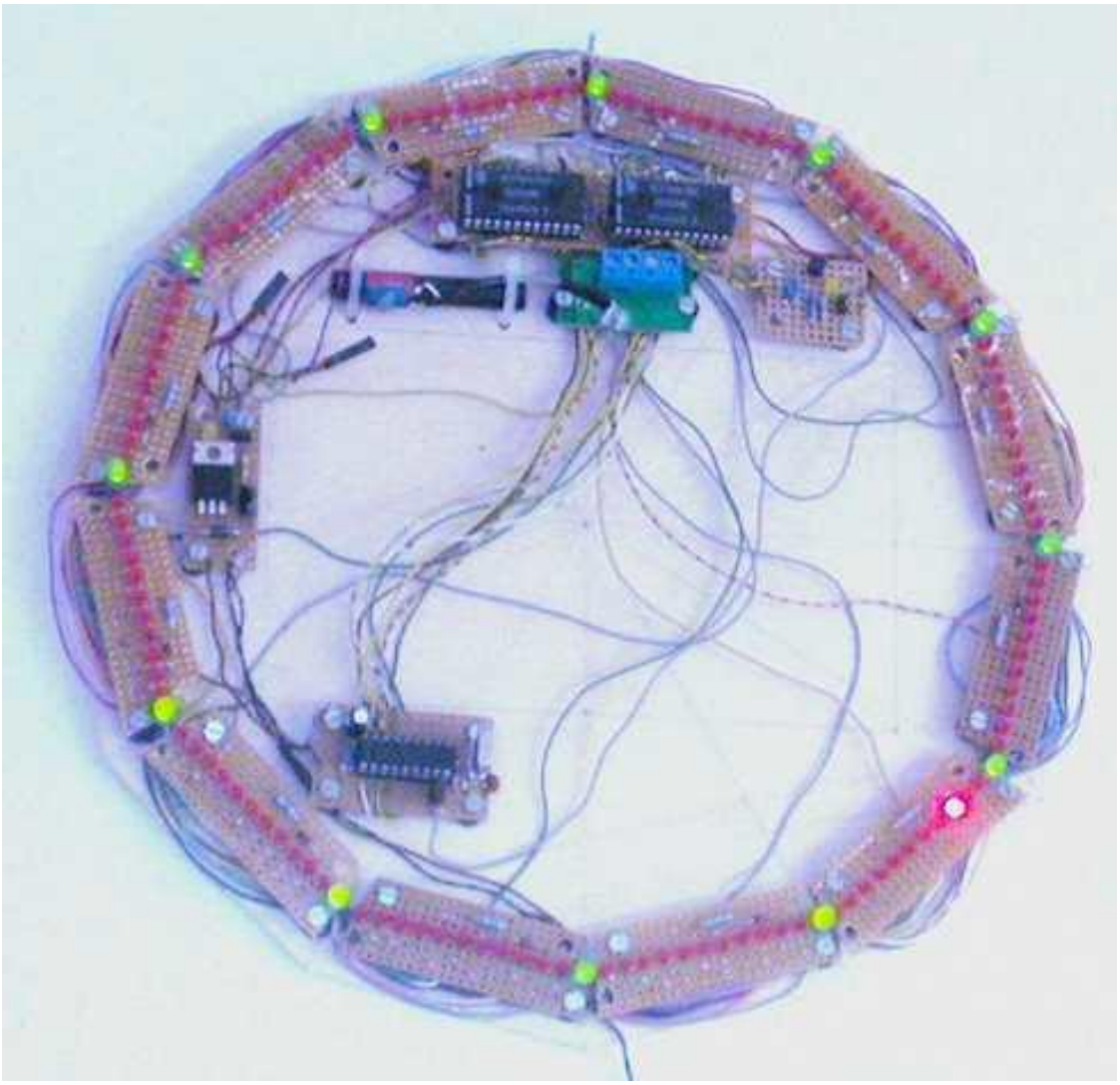
Analog/Digitale Wanduhr 1



Autor: Buchgeher Stefan

Letzte Bearbeitung: 15. Oktober 2006

Inhaltsverzeichnis

1. Einleitung

Eine etwas ungewöhnliche Uhr!

Ziel dieses Projektes war es, den DCF-Zeitzeichensender kennen zu lernen, und eine eigene Uhr zu entwickeln. Uhren gibt es in den unterschiedlichsten Variationen. Doch diese hebt sich ein bisschen hervor. Bei einer „normalen“ (analogen) Wanduhr zeigen 2 Zeiger die Stunden und Minuten an. Hier werden die beiden Zeiger nur durch eine leuchtende LED symbolisiert. Diese LED muss daher sowohl die Stunde als auch die Minute anzeigen. Die Farben der Leuchtdioden spielen hier aber keine Rolle. Für die Anzeige einer vollen Stunde sollte aber eine andere Farbe verwendet werden, als für die Anzeige einer Zeit die zwischen zwei vollen Stunden liegt. Ich verwendete hier 12 grüne Leuchtdioden (eine für jede volle Stunde) und 132 rote Leuchtdioden (je 11 für die Zeiten zwischen zwei vollen Stunden)
Diese Uhr sollte nicht als präzises Zeitmessgerät betrachtet werden, da es die Uhrzeit nur mit einer Genauigkeit von 5 Minuten anzeigt. Der Leuchtpunkt wandert nämlich nur alle 5 Minuten von einer Leuchtdiode zur nächsten Leuchtdiode.
Dank der DCF-Technik zeigt diese Uhr immer die korrekte Zeit an (hier aber, wie schon gesagt, nur mit einer Genauigkeit von 5 Minuten). Ein angenehmer Nebeneffekt ist, dass auch das (lästige) Zeitumstellen zweimal im Jahr entfällt.

Als Inspiration für diese Anzeigeart diente die im Elektronik-Monatsmagazin „Elektor“ in der Ausgabe 12/94 vorgestellte Uhr. Die Ansteuerung der Leuchtdioden wurde übernommen. Neu ist nur die Auswertung des DCF-Zeitzeichensenders mit einem Mikrocontroller.

Es steht aber nicht der praktische Nutzen, sondern die Programmierung und der Umgang mit Mikrocontrollern im Vordergrund dieses Projektes! Hier geht es hauptsächlich um die Auswertung (Dekodierung) des DCF-Signals. Dies ist auch der Grund weshalb ich alle erarbeiteten Unterlagen (Schaltpläne und Listing) zur freien Verfügung bereitstelle.
Da keine besonderen Anforderungen an den Mikrocontroller gestellt werden, könnte diese Aufgabe wohl auch von jeder anderen Mikrocontrollerfamilie durchgeführt werden. Ich habe mich aber für die PIC-Familie entschieden.

Ich hoffe, dass dieses Projekt ausreichend dokumentiert ist, so dass auch ein nicht Elektronik-Profi mit diesem Projekt zu Recht kommt.

Buchgeher Stefan

2. Grundlegendes zu DCF

Der Zeitzeichensender DCF77 befindet sich in Mainflingen, ca. 25 km südöstlich von Frankfurt am Main. Dieser Langwellensender hat, bedingt durch die niedrige Trägerfrequenz von 77,5 kHz eine Reichweite von ca. 1500 km bis 2000 km. (Bild 2.1)

Bild 2.1: Rechweite des DCF77-Senders (Quelle: www.dcf77.com/)

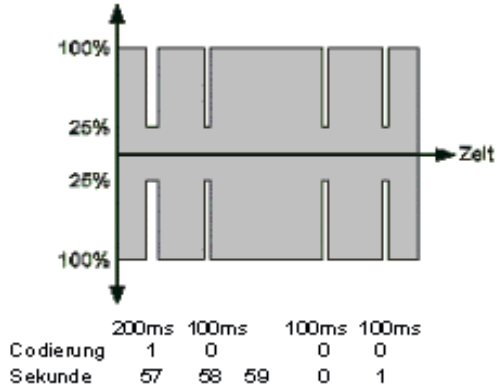


Bild 2.2: Amplitudenmodulation des 77,5kHz-Trägers

Der Dauerträger des DCF-Senders senkt im Sekundentakt für 100ms oder 200ms die Amplitude der Trägerfrequenz auf 25 % ab, was einer einfachen Amplitudenmodulation entspricht. Die Länge dieser so genannten Sekundenmarken überträgt in codierter Form das Zeittelegramm. Eine Absenkdauer des Trägers um 100ms (Toleranz: ±20ms) entspricht dabei einem logischen Low-Pegel, während ein logischer High-Pegel mit einer Absenkdauer von 200ms (Toleranz ±40ms) codiert ist. In jeder 59. Sekunde wird die Absenkung nicht vorgenommen, so dass damit eine eindeutige Zuordnung des Minutenanfangs möglich ist. Die neue Sekunde beginnt, mit Ausnahme der 59. Sekunde, jeweils mit dem Absenken des 77,5 kHz-Trägers.

Im jeweils einminütigem Zeittelegramm ist die Zeit (Stunde und Minute) der nächstfolgenden Minute sowie das komplette Datum und der jeweilige Wochentag codiert. Die folgende Tabelle zeigt die Bedeutung der 59 Bits, die pro Minute versandt werden.

Bit	Bedeutung	Wertigkeit	Bit	Bedeutung	Wertigkeit
0	Minutenbeginn Low		20	Telegrammbeginn High	
1	Reserve		21	Minuten Einer	1
2	Reserve		22	Minuten Einer	2
3	Reserve		23	Minuten Einer	4
4	Reserve		24	Minuten Einer	8
5	Reserve		25	Minuten Zehner	10
6	Reserve		26	Minuten Zehner	20
7	Reserve		27	Minuten Zehner	40
8	Reserve		28	Prüfbit 1	
9	Reserve		29	Stunden Einer	1
10	Reserve		30	Stunden Einer	2
11	Reserve		31	Stunden Einer	4
12	Reserve		32	Stunden Einer	8
13	Reserve		33	Stunden Zehner	10
14	Reserve		34	Stunden Zehner	20
15	Reserveantenne		35	Prüfbit 2	
16	Zeitumstellung Ankündigung		36	Kalendertag Einer	1
17	Zeitzonenbit 1		37	Kalendertag Einer	2
18	Zeitzonenbit 2		38	Kalendertag Einer	4
19	Schaltsekunde Ankündigung		39	Kalendertag Einer	8

Tabelle 2.1: Zeittelegramm

Bit	Bedeutung	Wertig-keit	Bit	Bedeutung	Wertig-keit
40	Kalendertag Zehner	10	50	Jahr Einer	1
41	Kalendertag Zehner	20	51	Jahr Einer	2
42	Wochentag	1	52	Jahr Einer	4
43	Wochentag	2	53	Jahr Einer	8
44	Wochentag	4	54	Jahr Zehner	10
45	Monat Einer	1	55	Jahr Zehner	20
46	Monat Einer	2	56	Jahr Zehner	40
47	Monat Einer	4	57	Jahr Zehner	80
48	Monat Einer	8	58	Prüfbit 3	
49	Monat Zehner	10	59	Keine Austastung	

Tabelle 2.1: Zeittelegramm (Fortsetzung)

Die Synchronisation des Sekundenzählers erfolgt mit Ausbleiben der Absenkung der 59. Sekunde. Die nächste Absenkung ist immer Low. Die Bits 1 bis 14 sind nicht belegt. Bit 15 zeigt durch einen High-Pegel an, dass zurzeit die Reserveantenne des DCF77-Senders aktiv ist. Im Normalfall ist dieses Bit auf „Low“ gesetzt. Bit 16 wird eine Stunde bevor die Zeitumstellung von Sommer- auf Winterzeit bzw. umgekehrt erfolgt auf „high“ gesetzt und mit der Zeitumstellung wieder zurückgesetzt. Die Zeitangaben beziehen sich auf die UTC-Zeit (**U**niversal **T**ime **C**oordinated). Bezogen auf die UTC-Zeit eilt die mitteleuropäische Zeit (MEZ) um eine Stunde vor, während die mitteleuropäische Sommerzeit (MESZ) um 2 Stunden voreilt. Diese Differenz wird in den Zeitzonenbits 17 und 18 ausgedrückt. Während der MEZ ist Bit 17 High und Bit 18 Low, während bei der Sommerzeit (MESZ) der Abstand zur UTC 2 Stunden beträgt und somit Bit 17 Low und Bit 18 High ist. Bit 19 kündigt eine bevorstehende Schaltsekunde an. Das eigentliche Zeit- und Datumstelegramm ist in den Bits 20 bis 58 codiert. Für die Einerstellen der Zeit und Datumsinformationen sind jeweils 4 Bit, während für die Zehnerstellen nur 2 oder 3 Bit erforderlich sind. Die Zahlendarstellung der Zeit- und Datumsinformation erfolgt im Binärformat (BCD-Code). Für die Jahreszahl werden nur die Einer- und Zehnerstelle übertragen. Die Bits 42 bis 44 geben in binärer Schreibweise den Wochentag an (der Wert 1 steht für den Montag, der Wert 7 für den Sonntag). Das Prüfbit 1 ergänzt die Bits 21 bis 27 auf gerade Parität, d.h. es werden die High-Bits 21 bis einschließlich 28 addiert, deren Ergebnis muss dann eine gerade Zahl ergeben. Das Prüfbit 2 ergänzt die Parität von Bit 29 bis 34, während Prüfbit 3 für die Parität der Bits 36 bis 57 zuständig ist. Diese Prüfbits sind ein erstes Überprüfungskriterium für ein DCF-Empfangsprogramm, welches damit zunächst auf einfache Weise die Konformität der empfangenen Daten überprüfen kann. Für eine fehlerfreie DCF-Decodierung sind allerdings noch weitere Maßnahmen notwendig. (Zum Beispiel ein Vergleich der soeben dekodierten Uhrzeit und des Datums mit der Uhrzeit und dem Datum welches mit der vorhergehenden Minute übertragen wurde und zusätzlich noch eine mitlaufende Softwareuhr).

In unregelmäßigen Zeitabständen muss eine Schaltsekunde eingefügt werden. Dies ist dadurch bedingt, dass sich die Erde nicht genau in 24 Stunden um sich selbst dreht. Auf die koordinierte Weltzeitskala UTC bezogen, wird diese Korrektur zum Ende der letzten Stunde des 31. Dezember oder 30. Juni vorgenommen. In Mitteleuropa muss die Schaltsekunde daher am 1. Januar um 1.00 Uhr MEZ oder am 1.Juli um 2.00 MESZ eingeschoben werden. Zu den genannten Zeiten werden daher 61 Sekunden gesendet.

3. Schaltungsbeschreibung

Die gesamte Schaltung lässt sich grob in vier Bereiche unterteilen:

- Mikrocontroller
- Anpass-Schaltung für das DCF-Empfangsmodul
- Ansteuerung der Leuchtdiodenmatrix (Ziffernblatt)
- Stromversorgung

Die Schaltungsbeschreibung erfolgt hier entsprechend diesen vier Bereichen.
Anhang A zeigt die Schaltpläne der gesamten Schaltung.

Mikrocontroller:
Der Mikrocontroller (IC1) zählt zu den wichtigsten Komponenten in diesem Projekt. Für diese Aufgabe wurde der Typ PIC16F84 der Fa. Microchip ausgewählt. Dieser verfügt über eine für dieses Projekt ausreichende Anzahl an I/O-Pins, über ausreichend Programmspeicher, Datenspeicher und auch über einen nichtflüchtigen Speicher[1]. Der größte Vorteil an diesem Mikrocontroller ist aber, dass er einen Flash-Programmspeicher besitzt und daher fast beliebig oft einfach neu programmiert werden kann.

Eigenschaften des Mikrocontroller PIC16F84:

- Flash-Programmspeicher: 1k x 14-bit-Wort
- Datenspeicher: 68 Bytes
- EEPROM-Datenspeicher: 64 Bytes
- I/O-Ports: 13
- Interruptquellen: 4
- Stapelspeicher: 8 Ebenen
- Timer: 1
- Taktfrequenz: 0 bis 20 MHz

Weitere Eigenschaften des PIC16F84:

- Direkte, indirekte und relative Adressierung
- Power-On-Reset (POR)
- Power-Up-Timer (PWRT)
- Interner Watchdog (WDT)
- In-Circuit Serial Programming™ (ICSP)
- Geringer Stromverbrauch

Für die Wanduhr wird jedoch nur ein Teil dieser Eigenschaften benötigt.

Grundbeschaltung des Mikrocontrollers:
Als Takterzeugung wird eine Standardapplikation bestehend aus einem Quarz (X1), zwei Keramikkondensatoren (C3, C4) und einem Widerstand (R14) verwendet. Bei X1 handelt es sich um einen 4,096 MHz-Quarz. Dieser recht „ungewöhnliche“ Wert ist notwendig, damit mit dem Controller eine genaue Uhr realisiert werden kann, welche auch ohne DCF-Synchronisierung oder bei einem gestörten DCF-Empfang eine genaue Zeit erzeugt.

[1] Unter einem nichtflüchtigen Speicher versteht man einen Speicher, welcher seine Werte beim Ausschalten des Gerätes beibehält. Diese Speicherart wird hauptsächlich für Geräteinstellungen verwendet

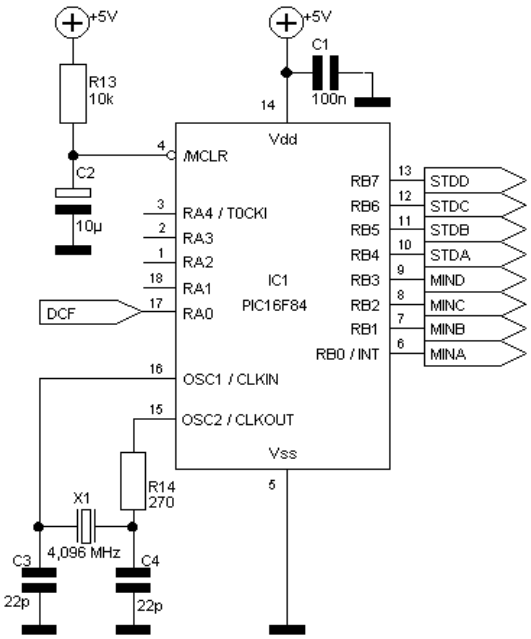


Bild 3.1: Beschaltung des Mikrocontroller

Zur Erzeugung des Reset wurde ebenfalls eine einfache Standardlösung bestehend aus einem Widerstand (R13) und einem Elektrolyt-Kondensator (C2) gewählt.

Der Kondensator C1 dient zur Entkoppelung der Betriebsspannung für den Mikrocontroller. Für diesen Koppelkondensator sollte ein Keramiktyp verwendet werden. Dieser muss möglichst nahe an diesen IC angebracht werden.

Wanduhrspezifische Beschaltung:

Das DCF-Empfangsmodul ist mit einer zusätzlichen Anpass-Schaltung am Portpin RA0 angeschlossen. Die restlichen 4 Pins des Port A sind für mögliche Erweiterungen vorgesehen.
Der gesamte Port B dient zur Ansteuerung der Leuchtdioden, also des Ziffernblatts.

Anpass-Schaltung für das DCF-Empfangsmodul:

Bild 3.2: DCF-Empfangsmodul

Eine weitere wichtige Komponente zur DCF-Dekodierung ist ein bei Conrad erhältliches DCF-Empfangsmodul (Bestell-Nr.: 641138). Dieses Modul enthält eine Empfangsantenne und einen Demodulator, so dass am Ausgang des Moduls das übertragene Zeittelegramm mit einem Mikrocontroller ausgewertet werden kann.

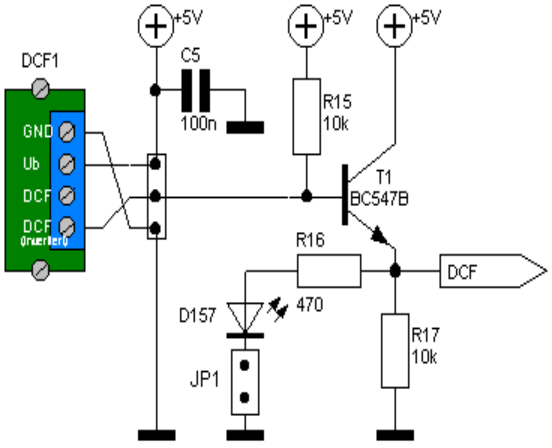


Bild 3.3: Anpass-Schaltung

Der Ausgangsstrom von nur einem Milliampere ist für die Kontroll-LED (D157) zuwenig. Die nach geschaltete Transistorstufe (T1) mit den Widerständen R15 und R17 gleicht diesen Nachteil aus. Der Widerstand R16 dient als Vorwiderstand für die Leuchtdiode (D157). Diese Leuchtdiode signalisiert den empfangenen Datenstrom, wenn der Jumper JP1 gesteckt ist. Bei einem korrekten Empfang blinkt diese Leuchtdiode im Sekundentakt. Dieses Blinken zeigt sozusagen den Ausgangspegel der Anpass-Schaltung an. Das Puls-Pausen-Verhältnis (Leuchtzeit/Dunkelzeit der Leuchtdiode D157) entspricht der im

Abschnitt 2 genannten Zeiten. Bei etwas Übung kann der Unterschied zwischen Low (100ms Absenkung des Trägers, LED ist 100ms dunkel) und High (200ms Absenkung des Trägers, LED ist 200ms dunkel) optisch erkannt werden. Dieser Jumper sollte allerdings nicht dauerhaft gesteckt werden, da dadurch die Spannungsquelle (z.B. ein Steckernetzteil) unnötig belastet wird.

Der Kondensator C5 dient zur Entkoppelung der Betriebsspannung für das DCF-Modul. Für diesen Koppelkondensator sollte ein Keramiktyp verwendet werden. Dieser muss möglichst nahe am DCF-Modul angebracht werden.

Ansteuerung der Leuchtdiodenmatrix (Ziffernblatt):

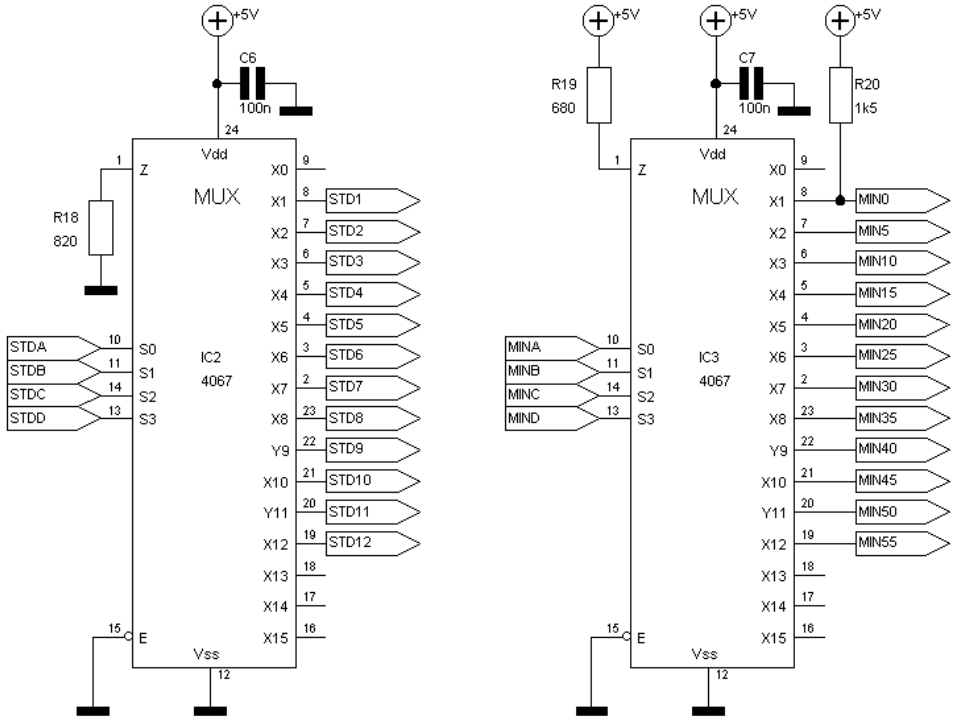


Bild 3.4: Ansteuerung der Leuchtdiodenmatrix

Für die Ansteuerung des aus insgesamt 144 Leuchtdioden bestehenden Ziffernblatts (Bild 3.5) dienen zwei 1-aus-16-Dekoder (IC2 und IC3). IC2 dient dabei zur Auswahl der Stunde, und mit IC3 wird die Minute ausgewählt.
Für die 1-aus-16-Dekoder wird der Type 4067 verwendet, wobei bei beiden nur die Ausgänge 1 bis 12 verwendet werden. Während der Synchronisierung wird auf den Ausgang 0 (Pin 9) geschaltet, so dass keine Leuchtdiode leuchtet. Nur bei der gültigen Uhrzeit leuchtet die durch diese beiden Dekoder ausgewählte Leuchtdiode.
Die Funktion der 1-aus-16-Dekoders ist sehr einfach. Der binäre Wert der 4 Eingänge S0 bis S3 bestimmt, welcher Ausgang X0 bis X15 mit dem Eingang Z (Pin 1) verbunden wird. Demnach schaltet IC3 den Widerstand R19 und das Pluspotential an den entsprechend ausgewählten Ausgang, während IC2 den Widerstand R18 und das Massepotential an den ausgewählten Ausgang legt. Als Folge wird eine der 144 Leuchtdioden über den „Vorwiderstand“ R19 auf Betriebsspannung gelegt und erhält auch gleichzeitig über einen weiteren „Vorwiderstand“ (R18) ein Massepotential und leuchtet, während die anderen Leuchtdioden dunkel bleiben.

Eine Besonderheit bilden die Leuchtdioden, die eine volle Stunde anzeigen (D1, D13, D25, D37 usw.). Diese 12 Leuchtdioden leuchten ständig (aber nur sehr schwach). Da sie mit ihren Kathoden über die relativ hohen Widerstände R1 bis R12 (10k) mit Masse und über R20 mit der Betriebsspannung verbunden sind. Zur vollen Stunde (Ausgang X1, Pin 8 von IC3 ist aktiv) wird R19 parallel zu R20 gelegt und R18 parallel zu einem der Widerstände R1 bis R12. Die Folge ist, dass die so ausgewählte Leuchtdiode stärker leuchtet als die anderen, die eine volle Stunde anzeigen. Mit diesem Schaltungstrick ist eine bessere Orientierung auf dem Ziffernblatt speziell in der Nacht möglich, da man sich an die schwach leuchtenden vollen Stunde orientieren kann.

Die beiden Kondensatoren C6 und C7 dienen zur Entkoppelung der Betriebsspannung für IC2 und IC3. Für diese Koppelkondensatoren sollten Keramiktypen verwendet werden. Diese sollten möglichst nahe an diese ICs angebracht werden.

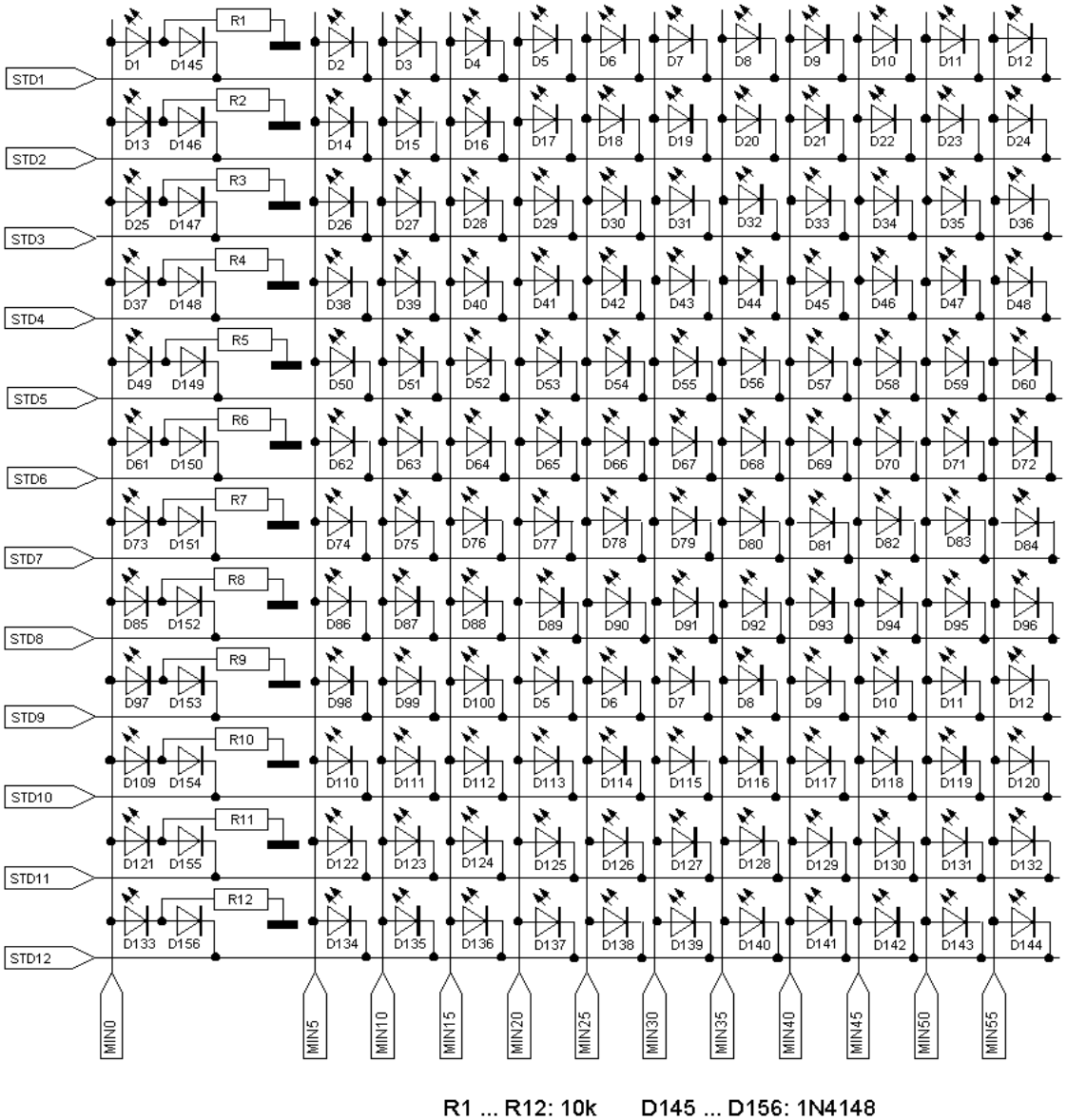


Bild 3.5: Leuchtdiodenmatrix

Stromversorgung:

Zur Stromversorgung gibt es nicht viel zu sagen. Festspannungsregler (IC4) vom Typ 7805 übernimmt mit den Kondensatoren C8 und C9 die Spannungsregelung. Als Spannungsquelle dient beispielsweise ein unstabilisiertes 9-V-Steckernetzteil.

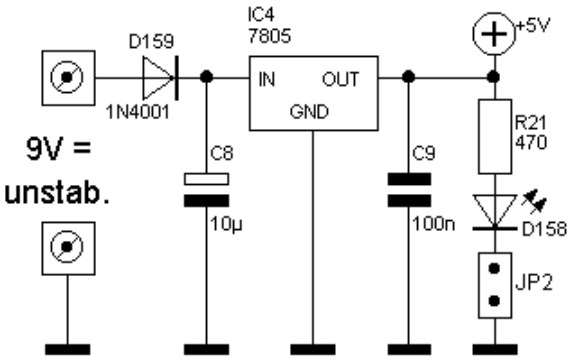


Bild 3.6: Stromversorgung

Bei gestecktem Jumper JP2 dient die Leuchtdiode D158 als Spannungskontrolle. Dieser Jumper sollte allerdings nicht dauerhaft gesteckt werden, da dadurch die Spannungsquelle unnötig zusätzlich belastet wird.

Die Diode D159 dient hier als Verpolungsschutz.

4. Softwarebeschreibung

Die Aufgabe der PIC-Software besteht bei der Wanduhr hauptsächlich aus der DCF-Dekodierung. Diese wiederum besteht aus mehreren Teilaufgaben:

- In kurzen Zeitabständen (ca. alle 4ms) das vom DCF-Empfangsmodul erzeugte Signal abtasten. Daraus entweder ein Low, High oder eine neue Minuten ermitteln. Kann aus den Abtastungen keines dieser drei Fälle ermittelt werden, so handelt es sich um einen Telegrammfehler.
- Die empfangenen Low- und High-Pegel zwischenspeichern
- Bei jeder neuen Minute aus den gespeicherten Werten entsprechend Tabelle 1 die Zeit- und Datumswerte zusammensetzen und in entsprechenden Registern sichern.
- Das soeben ermittelte Telegramm mit dem Telegramm der vorhergehenden Minute vergleichen. Die soeben ermittelte Minute muss dabei um 1 größer als die vorhergehende Minute sein, und die soeben ermittelten Werte für die Stunde, Tag, Monat, Jahr und Wochentag muss mit den Werten des vorhergehenden Telegramms übereinstimmen. ACHTUNG: Bei dieser einfachen Überprüfung wird der Übergang von z. B. 13:59 auf 14:00 als falsch gewertet, da sich hier die Stunde ändert, und auch die Minute um mehr als 1. Würden hier alle möglichen Übergänge berücksichtigt, dann würde dieses Unterprogramm noch umfangreicher als es ohnehin schon ist. Durch diese Vereinfachung dauert die Synchronisierung im schlimmsten Fall nur eine Minute länger.
- Parallel zur DCF-Uhr eine reine Softwareuhr erzeugen. Diese Softwareuhr ist gültig, wenn keine korrekte Zeit vom DCF-Empfangsmodul empfangen wird. Mit anderen Worten. Die Softwareuhr wird ständig mit einer gültig dekodierten „DCF-Zeit“ synchronisiert. Dies gilt natürlich auch für die Datumsinformationen.

Die zweite wichtige Aufgabe der PIC-Software ist die Anzeige der Uhrzeit. Für die Anzeige dient das gleichnamige Unterprogramm (ANZEIGE). Dieses Unterprogramm wird ebenfalls zyklisch alle Sekunde aufgerufen. Dazu wird die gleiche 1-Sekunden-Zeitbasis wie für die mitlaufende, innere Uhr verwendet.

4.1. Das DCF-Dekodierverfahren

Der PIC-I/O-Eingang, an dem das DCF-Empfangsmodul (mit einer Anpass-Schaltung) angeschlossen ist (hier bei der Wanduhr der Portpin RA0), wird zyklisch (ca. alle 4 ms) vom Unterprogramm DCFROUTINE abgefragt. Als Zeitbasis für den 4-ms-Takt dient der Timer-0-Interrupt. Der Aufruf des Unterprogramms DCFROUTINE erfolgt hier aber nicht von der ISR (Interrupt-Service-Routine), sondern vom Hauptprogramm. Die ISR setzt nur alle 4 ms ein Flag zur Kennzeichnung, dass das Hauptprogramm das Unterprogramm DCFROUTINE aufrufen soll.

Die Aufgabe des Unterprogramms DCFROUTINE besteht darin, aus dem vom DCF-Empfangsmodul empfangenen Datenstrom die Informationen LOW, HIGH und den Minutenwechsel (also das Ausbleiben der 59. Sekunde) gemäß dem im Abschnitt 2 (Grundlegendes zu DCF) gewonnen Erkenntnisse zu dekodieren. Für diese Informationen befinden sich im DCF-Statusregister (DCFSTATUS) entsprechende Flags. (Siehe Abschnitt 4.2. Benötigte Register, Konstanten, Portdefinition, Makros und Tabellen.)
Weiters beinhaltet das Register DCFSTATUS noch den Zustand des DCF-Eingangs 4 ms vor dem Aufruf des Unterprogramms DCFROUTINE. Mit Hilfe des aktuellen Zustands und des Zustands vor 4 ms kann nun erkannt werden, ob sich der Pegel am DCF-Eingang geändert hat.

Das Hauptprogramm prüft nun ständig die Flags DCFNEUESEK (wird zusätzlich zu DCFLOW bzw. DCFHIGH gesetzt, wenn eine gültige Sekunde empfangen wurde) und DCFNEUEMIN, und ruft die zugehörigen Unterprogramme *DCFUPSEKUNDE* bzw. *DCFUPMINUTE* auf. Die Aufgabe des Unterprogramms DCFUPSEKUNDE ist es, die empfangenen Low- und High-Pegel in den Register DCFTELEGRAMM1 bis DCFTELEGRAMM8 zwischenzuspeichern.
Die Aufgabe des Unterprogramms DCFUPMINUTE ist dagegen etwas umfangreicher:
Zunächst die Zeit und das Datum aus den in den Registern DCFTELEGRAMM1 bis DCFTELEGRAMM8 zwischengespeicherten Werten entsprechend Tabelle 1 zusammensetzen. Anschließend die soeben gewonnen Zeit- und Datumsinformationen mit den Zeit- und Datumsinformationen der vorhergehenden Minute vergleichen. Die Zeit- und Datumswerte der soeben empfangenen Minute sind nur dann gültig, wenn sich die Minute um maximal 1 von der vorhergehenden Minute unterscheidet, während sich die restlichen Zeit- und Datumsinformationen mit denen der vorhergehenden Minute nicht unterscheiden. Dies ist nur eine einfache Überprüfung, die aber manche richtigen Telegramme als falsch auswertet. (Siehe Abschnitt 4.5.4)

Parallel zur Dekodierung des DCF-Eingangs erzeugt das Unterprogramm *INNEREUHR* eine reine Softwareuhr. Für diesen Zweck ist eine genaue 1-Sekunden-Zeitbasis notwendig. Diese Zeitbasis wird von der schon erwähnten Timer-0-ISR (Interrupt Service Routine) zusätzlich zur 4ms-Zeitbasis erzeugt. Auch für diese Zeitbasis wird in der ISR ein entsprechendes Flag gesetzt, und das Hauptprogramm ruft das Unterprogramm INNEREUHR auf, wenn dieses Flag gesetzt ist. Diese zusätzliche Softwareuhr mag auf dem ersten Blick unnötig erscheinen. Es hat sich aber gezeigt, dass ein DCF-Empfang oft auch über eine längere Zeit nicht möglich ist. In diesem Fall würde die Uhr „stehen“. Diese Zeit wird mit einer zusätzlichen Softwareuhr so überbrückt, dass der Betrachter der Uhr davon nichts bemerkt.

4.2. Benötigte Register, Konstanten, Portdefinition, Makros und Tabellen

Register:
Für die Wanduhr sind neben einigen internen Register (SFR, **S**pezielle **F**unktions-**R**egister) noch eine ganze Menge eigener Register notwendig, wobei der Großteil für die DCF-Dekodierung notwendig sind:

- ISR_STAT_TEMP: Zwischenspeicher des Statusregister der ISR
- ISR_w_TEMP: Zwischenspeicher des Arbeitsregister der ISR
- FLAGSISRHP: beinhaltet Botschaftsflags ISR -> HP, hier bei der Wanduhr die Flags für die 4-ms-Zeitbasis (Flag FLAG4MSEK) und für 1-Sekunden-Zeitbasis (Flag FLAG1SEK)
- ZAEHLERISR1SEK: Zählregister für 1-Sekunden-Zeitbasis
- DCFSTATUS: Statusregister der DCF-Dekodierung. Dieses Register beinhaltet folgende Bits:

Bit	Flagname	Funktion
0	DCFPORTNEU	Akt. Zustand DCF-Porteingang
1	DCFPORTALT	Vorhergehender Zustand am DCF-Porteingang
2	DCFNEUESEK	gesetzt, wenn neue Sekunde begonnen
3	DCFNEUEMIN	gesetzt, wenn neue Minute begonnen
4	DCFLOW	gesetzt, wenn Low-Signal erkannt
5	DCFHIGH	gesetzt, wenn High-Signal erkannt
6	DCFFEHLER	gesetzt, wenn ein Fehler erkannt
7	DCFSYNC	gesetzt, wenn Uhr mit DCF synchronisiert

Tabelle 4.1: Zusammensetzung von DCFSTATUS

- DCFPULS: Zählregister zur Ermittlung der Pulsdauer bei der DCF-Dekodierung
- DCFPAUSE: Zählregister zur Ermittlung der Pausedauer bei der DCF-Dekodierung.
- DCFTELEGRAMM1 – DCFTELEGRAMM8: In diese Register wird das empfangene, neue DCF-Telegramm im Sekundentakt eingelesen. Beginnt eine neue Minute, so werden diese Register ausgewertet und in die entsprechenden Register kopiert, z.B. in das Register DCFMINBCD.
- DCFMINBCD: Beinhaltet die dekodierte, aktuelle Minute im BCD-Format.
- DCFSTDBCD: Beinhaltet die dekodierte, aktuelle Stunde im BCD-Format.
- DCFTAGBCD: Beinhaltet den dekodierten, aktuellen Tag im BCD-Format.
- DCFMONBCD: Beinhaltet den dekodierten, aktuellen Monat im BCD-Format.
- DCFJAHRBCD: Beinhaltet das dekodierte, aktuelle Jahr im BCD-Format, wobei nur die Einer- und die Zehnerstelle im DCF-Telegramm enthalten sind. Die Hunderter- und die Tausenderstelle befinden sich nicht im DCF-Telegramm
- DCFWOCHENTAG: Beinhaltet den dekodierten, aktuellen Wochentag, wobei folgende Tabelle gilt:

Wert	Bedeutung
1	Montag
2	Dienstag
3	Mittwoch
4	Donnerstag
5	Freitag
6	Samstag
7	Sonntag

Tabelle 4.2: Wochentag

- DCFZUSATZINFOS: Beinhaltet weitere, folgende Informationen, die im DCF-Telegramm enthalten sind:

Bit	Flagname	Funktion
0	DCFRESANT	Ist dieses Bit gesetzt so wurde die Reserveantenne zum Versenden des DCF-Telegramms verwendet
1	DCFSOMWIN	Ist dieses Bit gesetzt, so kündigt es eine Umstellung zwischen Sommer- und Winterzeit an. Dieses Bit wird eine Stunde vor der Zeitumstellung gesetzt. Ab der Zeitumstellung enthält es wieder den Wert 0
2	DCFSOMMER	Während der Sommerzeit ist dieses Bit gesetzt
3	DCFWINTER	Während der Winterzeit ist dieses Bit gesetzt
4	DCFSCHALTSEK	Ist dieses Bit gesetzt, so kündigt es eine Schaltsekunde an

Tabelle 4.3: Zusammensetzung von DCFZUSATZINFOS

- DCFMINALT: Beinhaltet die empfangen Minute des DCF-Telegramms, welches mit der vorangegangenen Minute empfangen wurde. Diese wird zur Überprüfung des neuen DCF-Telegramms verwendet.
- DCFSTDALT: Beinhaltet die empfangen Stunde des DCF-Telegramms, welches mit der vorangegangenen Minute empfangen wurde. Diese wird zur Überprüfung des neuen DCF-Telegramms verwendet.
- DCFTAGALT: Beinhaltet den empfangenen Tag des DCF-Telegramms, welcher mit der vorangegangenen Minute empfangen wurde. Dieser wird zur Überprüfung des neuen DCF-Telegramms verwendet.
- DCFMONALT: Beinhaltet den empfangen Monat des DCF-Telegramms, welcher mit der vorangegangenen Minute empfangen wurde. Dieser wird zur Überprüfung des neuen DCF-Telegramms verwendet.
- DCFJAHRALT: Beinhaltet das empfangen Jahr des DCF-Telegramms, welches mit der vorangegangenen Minute empfangen wurde. Dieses wird zur Überprüfung des neuen DCF-Telegramms verwendet.
- DCFWTAGALT: Beinhaltet den empfangen Wochentag des DCF-Telegramms, welcher mit der vorangegangenen Minute empfangen wurde. Dieser wird zur Überprüfung des neuen DCF-Telegramms verwendet.
- UHRSEKUNDE: Sekundenzähler der inneren quarzgesteuerten Uhr. (wird mit jedem gültigen DCF-dekod. Wert synchronisiert)
- UHRMINUTE: Minutenzähler der inneren quarzgesteuerten Uhr. (wird mit jedem gültigen DCF-dekod. Wert synchronisiert)
- UHRSTUNDE: Stundenzähler der inneren quarzgesteuerten Uhr. (wird mit jedem gültigen DCF-dekod. Wert synchronisiert)
- DATUMTAG: Tageszähler der inneren quarzgesteuerten Uhr. (wird mit jedem gültigen DCF-dekod. Wert synchronisiert)
- DATUMMONAT: Monatszähler der inneren quarzgesteuerten Uhr. (wird mit jedem gültigen DCF-dekod. Wert synchronisiert)
- DATUMJAHR: Jahreszähler der inneren quarzgesteuerten Uhr. (wird mit jedem gültigen DCF-dekod. Wert synchronisiert)
- DATUMWTAG: Wochentagszähler der inneren quarzgesteuerten Uhr. (wird mit jedem gültigen DCF-dekod. Wert synchronisiert)
- TEMP1, TEMP2: Hilfsregister

Konstanten:

Die Konstante *KONSTISR1SEK* gibt die Anzahl der notwendigen ISR-Aufrufe für die 1-Sekunden Zeitbasis an.

Hier wird die ISR alle 4ms aufgerufen, daher ergibt sich für diese Konstanten der Wert 250 (250 x 4ms = 1000ms = 1 Sekunde).

Bei einem LOW dauert die Absenkung des Trägers zwischen 80ms und 120ms, für ein High dauert die Absenkung des Trägers zwischen 160ms und 240ms (siehe auch Abschnitt 2. Grundlegendes zu DCF). Die Konstanten *DCFLOWMIN*, *DCFLOWMAX*, *DCFHIGHMIN* und *DCFHIGHMAX* dienen zur Ermittlung eines LOW oder eines HIGH.
Das Unterprogramm DCFROUTINE wird alle 4ms aufgerufen, daher ergeben sich für die Konstanten folgende Werte:

- DCFLOWMIN: 20 (20 x 4ms = 80ms)
- DCFLOWMAX: 30 (30 x 4ms = 120ms)
- DCFHIGHMIN: 40 (40 x 4ms = 160ms)
- DCFHIGHMAX: 60 (60 x 4ms = 240ms)

Portdefinition:
Im Allgemeinen wird bei jeder Anwendung der Eingangspin für die DCF-Dekodierung an einem anderen Portpin verwendet. Damit dies in der Software nur an einer Stelle berücksichtigt werden muss befindet sich in der Software eine Portdefinition für den DCF-Eingang. Diese besteht aus den folgenden 3 Parametern:

- DCFINPORT: Dieser Parameter gibt den Port an (z.B. Port A, Port B,...).
- DCFINTRIS: Dieser Parameter ist für die Initialisierung des Verwendeten Ports zuständig. Für die DCF-Dekodierung muss der verwendete Portpin als Eingang definiert werden.
- DCFIN: Dieser Parameter gibt den Portpin des Verwendeten Ports an (z.B. 0, 1, 2, usw.).

Achtung: Wird für DCFINPORT der Port A verwendet, so muss für DCFINTRIS das zum Port A zugehörige TRIS-Register definiert werden. Bei der hier beschriebenen Wanduhr ergeben sich aufgrund der Hardwarebeschaltung folgende Definitionen.

```
DCFINPORT   equ   PORTA
DCFINTRIS   equ   TRISA
DCFIN       equ   0
```

Makros:

```
;Umschalten zu Registerbank 0
bank0       MACRO
            bcf   STAT,RP0
            ENDM

;Umschalten zu Registerbank 1
bank1       MACRO
            bsf   STAT,RP0
            ENDM
```

Tabellen:
Die bei der Wanduhr verwendeten Tabellen dienen zur Anzeige der Uhrzeit:

- Mit Hilfe dieser Tabelle kann eine Division des w-Register durch 5 sehr einfach realisiert werden

```
TABDIV5         addwf PC,f
                dt    .0,.0,.0,.0,.0
                dt    .1,.1,.1,.1,.1
                dt    .2,.2,.2,.2,.2
                dt    .3,.3,.3,.3,.3
```

```
dt    .4,.4,.4,.4,.4
dt    .5,.5,.5,.5,.5
dt    .6,.6,.6,.6,.6
dt    .7,.7,.7,.7,.7
dt    .8,.8,.8,.8,.8
dt    .9,.9,.9,.9,.9
dt    .10,.10,.10,.10,.10
dt    .11,.11,.11,.11,.11
dt    .0,.0,.0,.0
```

- Die Tabelle TABSTUNDEN wandelt den Stundenwert von 24 nach 12 um.

Hier die Tabelle:

```
TABSTUNDEN    addwf PC,f
              dt    .12
              dt    .1,.2,.3,.4,.5,.6,.7,.8,.9,.10,.11,.12
              dt    .1,.2,.3,.4,.5,.6,.7,.8,.9,.10,.11
              dt    .0,.0,.0,.0,.0,.0,.0,.0,.0
```

4.3. Hauptprogramm

Aufgaben des Hauptprogramms:
Zuerst wird der Mikrocontroller initialisiert. Diese Tätigkeit wird von einem Unterprogramm (INIT) ausgeführt. Danach wird der Timer0-Interrupt und der globale Interrupt freigegeben.[2] Dafür ist das Register INTCON zuständig. Je nach benötigten Interrupts werden die entsprechenden Freigabebits (im Englischen: Enable) gesetzt. Wird ein Interrupt verwendet so muss zusätzlich zum verwendeten Interrupt auch die globale Interruptfreigabe GIE (**G**eneral **I**nterrupt **E**nable) gesetzt werden. Er ist sozusagen der Hauptschalter, der Interrupts ermöglicht. Das INTCON-Register kann einfach mit den Befehlen `movlw b‘10100000‘` und `movwf INTCON` beschrieben werden. Der Timer0-Interrupt ist jetzt eingeschaltet. Er sorgt hier bei der Wanduhr für eine 4-ms-Zeitbasis und für eine1-Sekunden-Zeitbasis.

Nun befindet sich die Software in einer Endlosschleife. Diese Schleife besitzt die Aufgabe ständig die so genannten Botschaftsflags abzufragen. Ist eines dieser Botschaftsflags gesetzt, so muss vom Hauptprogramm eine bestimmte Aufgabe ausgeführt werden. Diese Aufgaben sind in Form von Unterprogrammen vorhanden. Zwei dieser Botschaftsflags werden in der Timer-ISR gesetzt. Und zwar wird alle 4ms ein Flag gesetzt. Das zweite Flag von der ISR wird jede Sekunde gesetzt.

Hier Zusammenfassend die Tätigkeiten in der Endlosschleife, welche durch die Botschaftsflags ausgelöst werden:

- Tätigkeiten und/oder Unterprogramme, die alle 4ms durchgeführt werden müssen
  - Unterprogramm DCFROUTINE aufrufen (siehe Abschnitt 4.5.2)
- Tätigkeiten und/oder Unterprogramme, die jede Sekunde durchgeführt werden müssen
  - Unterprogramm INNEREUHR aufrufen (siehe Abschnitt 4.5.5)
  - Unterprogramm ANZEIGE aufrufen (siehe Abschnitt 4.5.6)
- Bei einem Minutenwechsel (Flag DCFNEUEMIN ist gesetzt) das Unterprogramm DCFUPMINUTE abarbeiten (siehe Abschnitt 4.5.4)

[2] Der PIC16F84 besitzt 4 Interruptquellen. Bei der Wanduhr wird aber nur der Timer0-Interrupt benötigt.

- Bei einem Sekundenwechsel (Flag DCFNEUESEK ist gesetzt) das Unterprogramm DCFUPSEKUNDE abarbeiten (siehe Abschnitt 4.5.3)

Hier das Hauptprogramm:

```
Beginn      call  INIT              ;Initialisierungen
            movlw b'10100000'       ;TMR0 freigeben durch
            movwf INTCON            ;Setzen von GIE und T0IE im Register
                                    ;  INTCON

HPSCHLEIFE  btfsc FLAGSISRHP,FLAG4MSEK
            goto  HPWEITER1

            btfsc FLAGSISRHP,FLAG1SEK
            goto  HPWEITER2

            btfsc DCFSTATUS,DCFNEUEMIN ;Wenn Anforderungsflag DCFNEUEMIN
                                    ;  gesetzt
            call  DCFUPMINUTE       ;Unterprogramm DCFUPMINUTE ausfuehren
            btfsc DCFSTATUS,DCFNEUESEK ;Wenn Anforderungsflag DCFNEUESEK
                                    ;  gesetzt
            call  DCFUPSEKUNDE      ;Unterprogramm DCFUPSEKUNDE ausfuehren

            goto  HPSCHLEIFE

HPWEITER1   call  DCFROUTINE        ;Taetigkeiten, die alle 4 ms durchgefuehrt
                                    ;  werden muessen

            bcf   FLAGSISRHP,FLAG4MSEK ;Anforderungsflag (fuer 4ms-
                                    ;  Aktivitaeten) zuruecksetzen
            goto  HPSCHLEIFE

HPWEITER2   call  INNEREUHR         ;Taetigkeiten, die jede Sekunde
                                    ;  durchgefuehrt werden muessen
            call  ANZEIGE

            bcf   FLAGSISRHP,FLAG1SEK ;Anforderungsflag (fuer 1-Sek.-
                                    ;  Aktivitaeten) zuruecksetzen
            goto  HPSCHLEIFE
```

4.4. ISR (Timer 0)

Eine ISR (**I**nterrupt **S**ervice **R**outine) ist im Prinzip ein Unterprogramm, welches aber im Gegensatz zu normalen Unterprogrammen, „unvorhergesehen“ aufgerufen wird. Hier, beim Timer 0-Interrupt jedes Mal, wenn der Timer 0 überläuft, also von 255 auf 0 wechselt. Würde zum Beispiel ein RB-Interrupt verwendet werden, so würde bei jeder Pegeländerung von RB4 bis RB7 ein Interrupt auftreten und die entsprechende ISR wird ausgeführt. Eine ISR sollte daher so kurz wie möglich sein.

Ein weiterer wichtiger Punkt bei einer ISR ist, dass das w-Register (Working- oder Arbeitsregister) und das STATUS-Register in andere Registern zwischengespeichert werden müssen, falls diese in der ISR ihren Registerinhalt verändern. Der Grund dafür ist, dass eine ISR eben unvorhergesehen aufgerufen wird, und die angesprochenen Register unter Umständen zu diesem Zeitpunkten gerade benötigte Werte enthalten. Nach Ausführung der ISR springt diese zwar wieder genau an die Stelle zurück, wo sie war, bevor der Interrupt auftauchte, aber mit einem möglicherweise falschen Wert im w-Register (bzw. STATUS-Register). Das Zwischenspeichern des w-Register bzw. des

STATUS-Registers wird häufig auch als PUSH bezeichnet. Das Widerherstellen von w-Register und STATUS-Register nennt man POP.

Woher weis das Programm, dass ein Interrupt aufgerufen werden muss? Dazu gibt es für jede Interruptquelle ein Kontroll-Flag. Dies wird vom Controller gesetzt wenn dieser Interrupt auftritt. (Vorausgesetzt, dass diese Interruptquelle freigegeben ist). Damit aber die ISR nicht ständig aufgerufen wird, muss dieses Bit in der ISR wieder gelöscht werden.

Nun aber zur wanduhrspezifischen Timer 0-ISR. Diese hat lediglich die Aufgabe eine Zeitbasis für 4ms und eine Zeitbasis für 1 Sekunde zu erzeugen. Eine Zeitbasis für 4ms ist hier besonders einfach zu erzeugen, da ja diese ISR ohnehin schon zu genau diesen Zeitabständen aufgerufen wird. Es ist also nur notwendig ein Flag zu setzen. Dieses Flag trägt hier den Namen FLAG4MSEK und befindet sich im Register FLAGSISRHP. Die Generierung für die 1-Sekunden-Zeitbasis ist dagegen schon etwas „umfangreicher“. Damit eine Zeit von einer Sekunde entsteht muss die ISR 250-mal aufgerufen werden (250 x 4ms = 1000ms = 1 Sekunde). Bei jedem ISR-Aufruf muss also ein Zählregister um 1 vermindert werden. Besitzt es danach den Wert 0, so ist eine Sekunde vergangen. Nun wird das Botschaftsflag FLAG1SEK im Register FLAGSISRHP gesetzt, und das Zählregister muss mit dem Wert 250 neu geladen werden. Der Wert 250 wird hier durch die Konstante KONSTISR1SEK ersetzt.

Die ISR wird, wie schon mehrmals erwähnt, alle 4ms aufgerufen Diese 4ms ergeben sich folgendermaßen: TMR0 wird mit dem Wert 0 geladen – es dauert also 256 Taktzyklen bis das Register wieder den Wert 0 besitzt, der Vorteiler besitzt den Wert 16 (vgl. Unterprogramm INIT, Abschnitt. 4.5.1). Der Taktzyklus ergibt sich aus dem verwendeten Quarz (X1). Dieser ist bei der PIC-Familie wie folgt definiert:

$$\frac{4}{f_{Quarz}}$$

Daraus ergibt sich folgender Zusammenhang:

$$ISRAUFRUF \quad [\mu s] = \frac{4 \bullet 256 \bullet 16}{f_{Quarz}\ [MHz]} = \frac{4 \bullet 256 \bullet 16}{4{,}096} = 4000$$

Also ein ISR-Aufruf alle 4000µs, was gleichbedeutend mit 4ms ist.

Eine ISR muss mit dem Befehl *retfie* beendet werden.[3]

Hier die gesamte ISR:

```
ISR
PUSH        movwf ISR_w_TEMP        ;w-Register retten
            swapf STAT,w            ;Statusregister
            movwf ISR_STAT_TEMP     ;  retten

            ;Beginn der eigentlichen ISR-Routine
```

[3] Beim Aufruf der ISR wird automatisch das Bit GIE gelöscht, damit während der Ausführung der ISR kein weiterer Interrupt ausgelöst werden kann, was bei mehreren freigegebenen Interruptquellen durchaus möglich sein kann. Mit dem Befehl *retfie* wird zunächst an die Stelle im Programm zurückgesprungen, wo sich die Programmabarbeitung befand bevor die ISR aufgerufen wurde, und die verwendeten Interrupts werden wieder freigegeben (hier bei der Wanduhr der Timer0-Interrupt)

```
            bsf   FLAGSISRHP,FLAG4MSEK ;Botschaftsflag setzen

            decfsz ZAEHLERISR1SEK,f ;Zaehlregister fuer 1-Sekunden-Zeitbasis
                                    ;  um 1 vermindern

            goto  ISRWEITER1
            bsf   FLAGSISRHP,FLAG1SEK ;Botschaftsflag setzen
            movlw KONSTISR1SEK      ;Zaehlregister fuer den Sekundentakt mit
            movwf ZAEHLERISR1SEK    ;  der Konstanten KONSTISR1SEK laden

ISRWEITER1
            ;Ende der eigentlichen ISR-Routine
ISRFERTIG   bcf   INTCON,T0IF       ;T0-Interruptflag loeschen

POP         swapf ISR_STAT_TEMP,w   ;Status-Register
            movwf STAT              ;  und
            swapf ISR_w_TEMP,f      ;  w-Register
            swapf ISR_w_TEMP,w      ;  wieder herstellen

            retfie
```

4.5. Unterprogramme

Die insgesamt sieben Unterprogramme lassen sich folgendermaßen einteilen:

- Unterprogramm zur Initialisierung des Mikrocontroller (INIT)
- Unterprogramme zur DCF-Dekodierung (DCFROUTINE, DCFUPSEKUNDE und DCFUPMINUTE)
- Unterprogramm, für den Fall dass kein DCF-Empfang möglich ist (INNEREUHR)
- Unterprogramm zur Ausgabe der Uhrzeit (ANZEIGE)
- Unterprogramm für Berechnungen, Umwandlungen und dergleichen (BCDBIN2)

4.5.1. INIT

Dieses Unterprogramm dient zur Initialisierung des Mikrocontrollers. Der Portpin an dem der DCF-Empfänger angeschlossen ist muss als Eingang definiert werden, während die Portpins zur Anzeige als Ausgang definiert werden.
In der Initialisierungsroutine muss für den erste Aufruf des Unterprogramms DCFROUTINE der aktuelle Zustand dieses Portpins gelesen, und im Register DCFSTATUS kopiert werden.
Da das Unterprogramm DCFROUTINE zyklisch (alle 4ms) aufgerufen wird, ist eine entsprechende Zeitbasis notwendig. Diese wird mit Hilfe eines Timer-Interrupts erzeugt. Für die Definition der Zeitbasis ist hier das mikrocontrollerinterne Funktions-Register OPTREG (in der Registerbank 1) zuständig. Damit bei einer PIC-Taktfrequenz von 4,096MHz eine Zeitbasis von 4ms erzeugt wird, muss das Register OPTREG mit dem binären Wert b‘00000011‘ geladen werden. Das Zählregister für dies Zeitbasis (Funktions-Register TMR0, in Registerseite 0) muss gelöscht werden.

Weiters müssen einige Register vorbelegt (gelöscht) werden.

Hier das Unterprogramm:

```
INIT        clrf  TMR0             ;Timer0 auf 0 voreinstellen

            bank1                  ;Registerseite 1
```

```
movlw b'00000011'       ;interner Takt, Vorteiler = 16 an TMR0
movwf OPTREG
clrf  TRISA             ;Port A: Ausgaenge (unbenutzt)

clrf  TRISB             ;Port B: Ausgaenge

bsf   DCFINTRIS,DCFIN   ;DCFIN als Eingang definieren
bank0                   ;Registerseite 0

movlw KONSTISR1SEK      ;Zaehlregister fuer den Sekundentakt mit
movwf ZAEHLERISR1SEK    ;  der Konstanten KONSTISR1SEK laden

clrf  DCFSTATUS         ;DCF-Statusregister (DCFSTATUS)
                        ;  initialisieren
bsf   DCFSTATUS,DCFFEHLER ;  Fehlerflag setzen, alle anderen Flags
                        ;  loeschen

btfsc DCFINPORT,DCFIN   ;Pegel von DCFIN einlesen und im Flag
bsf   DCFSTATUS,DCFPORTNEU ;  DCFPORTNEU des Register DCFSTATUS
                        ;  sichern

clrf  PORTB             ;Anzeige (Ziffernblatt loeschen)

clrf  UHRSEKUNDE        ;Sekundenzaehler initialisieren
clrf  UHRMINUTE         ;Minutenzaehler initialisieren
clrf  UHRSTUNDE         ;Stundenzaehler initialisieren
clrf  DATUMWTAG         ;Wochentagzaehler initialisieren
clrf  DATUMTAG          ;Tageszaehler initialisieren
clrf  DATUMMONAT        ;Monatszaehler initialisieren
clrf  DATUMJAHR         ;Jahreszaehler initialisieren
movlw .99
movwf DCFMINALT         ;Vergleichsregister initialisieren
movwf DCFSTDALT         ;Vergleichsregister initialisieren
movwf DCFTAGALT         ;Vergleichsregister initialisieren
movwf DCFMONALT         ;Vergleichsregister initialisieren
movwf DCFJAHRALT        ;Vergleichsregister initialisieren
movwf DCFWTAGALT        ;Vergleichsregister initialisieren
return
```

4.5.2. DCFROUTINE

Das Unterprogramm DCFROUTINE ist für die DCF-Dekodierung die wichtigste Komponente.

Diese Routine wird ca. alle 4 ms aufgerufen

Aufgabe und Vorgehensweise:
Die Aufgabe des Unterprogramms DCFROUTINE besteht darin aus den Abtastungen herauszufinden, ob ein Low, ein High oder ein Minutenwechsel gesendet wurde. Dazu wird bei jedem Aufruf des Unterprogramms DCFROUTINE entweder das Zählregister DCFPULS oder das Zählregister DCFPAUSE um 1 erhöht (inkrementiert), wenn sich der Pegel vom DCF-Eingang zum vorhergehenden Aufruf dieses Unterprogramms nicht verändert hat. Ist der DCF-Eingang Low (also logisch 0), so wird das Zählregister DCFPULS um 1 erhöht, ist der DCF-Eingang High (also logisch 1), so wird das Zählregister DCFPAUSE um 1 erhöht. Beide Zählregister können maximal den Wert 255 aufnehmen, was einer Zeit von 1020 ms (oder 1,02 Sekunden) entspricht (=255 * 4ms).

Unterscheidet sich der aktuelle Pegel des DCF-Eingangs mit dem Pegel vom vorhergehenden Aufruf, so spricht man von einer Flanke, wobei hier zwischen einer fallenden und einer steigenden Flanke unterschieden werden muss. Bei einer fallenden Flanke (der aktuelle Pegel des DCF-Eingangs ist logisch 0, während der Pegel beim vorhergehenden Aufruf noch logisch 1 war), erfolgt zunächst eine Auswertung des Zählregisters DCFPULS. Beinhaltet dieses Zählregister einen Wert zwischen den Konstanten DCFLOWMIN und DCFLOWMAX, so wurde ein LOW des DCF-Telegramms ermittelt. Im DCF-Statusregister (DCFSTATUS) werden daher die Flags DCFLOW und DCFNEUESEK gesetzt. Das Flag DCFNEUESEK dient als Zeichen einer neu empfangenen und gültigen Sekunde. Anschließend wird das Zählregister DCFPULS gelöscht. Beinhaltet das Zählregister DCFPULS einen Wert zwischen den Konstanten DCFHIGHMIN und DCFHIGHMAX, so wurde ein HIGH des DCF-Telegramms ermittelt. Im DCF-Statusregister (DCFSTATUS) werden daher die Flags DCFHIGH und DCFNEUESEK gesetzt. Das Flag DCFNEUESEK dient auch hier als Zeichen einer neu empfangenen und gültigen Sekunde, und auch das Zählregister DCFPULS wird anschließend gelöscht. Beinhaltet das Zählregister DCFPULS einen Wert der weder zwischen den Konstanten DCFLOWMIN und DCFLOWMAX noch zwischen den Konstanten DCFHIGHMIN und DCFHIGHMAX liegt, so handelt es sich um einen Übertragungsfehler. In diesem Fall wird das Fehlerflag (DCFFEHLER) im DCF-Statusregister (DCFSTATUS) gesetzt und auch das Zählregister DCFPULS wird anschließend gelöscht. Das Flag DCFNEUESEK wird hier aber nicht gesetzt, da es sich in diesem Fall um keine gültige Sekunde handelt. Das Fehlerflag wird erst bei der Erkennung einer neuen Minute wieder gelöscht. Achtung: Solange das Fehlerflag gesetzt ist erfolgt **keine** Auswertung des Zählregisters DCFPULS. Dieses Zählregister wird aber dennoch bei jeder fallenden Flanke zurückgesetzt.
Bei einer steigenden Flanke (der aktuelle Pegel des DCF-Eingangs ist logisch 1, während der Pegel beim vorhergehenden Aufruf noch logisch 1 war) wird das Zählregister DCFPAUSE gelöscht.
Entsprechend dem im Abschnitt 2 beschriebenen Protokoll erfolgt in der 59. Sekunde keine Absenkung des Trägers. Da es also in der 59. Sekunde keine fallende Flanke gibt, gibt es auch keine steigende Flanke. Das Zählregister DCFPAUSE „läuft über“, da es nur einen Zählbereich von 0 bis 255 besitzt. Dieser Überlauf wird hier ausgenützt. Zur Bestimmung einer neuen Minute bzw. zur Bestimmung des Telegrammbeginns. Ist das Fehlerflag gesetzt, so wird es nun gelöscht. War es nicht gesetzt, so wird das Flag DCFNEUEMIN im DCF-Statusregister (DCFSTATUS) gesetzt.

Das folgende Flussdiagramm soll das soeben beschriebene und umfangreiche Unterprogramm verdeutlichen. In den grau hinterlegten Bereichen erfolgt das Setzen oder Rücksetzen der zur weiteren Verwendung notwendigen Übergabeflags im DCF-Statusregister (DCFSTATUS). Die Unterprogramme DCFUPSEKUNDE und DCFUPMINUTE greifen auf diese Flags zu.

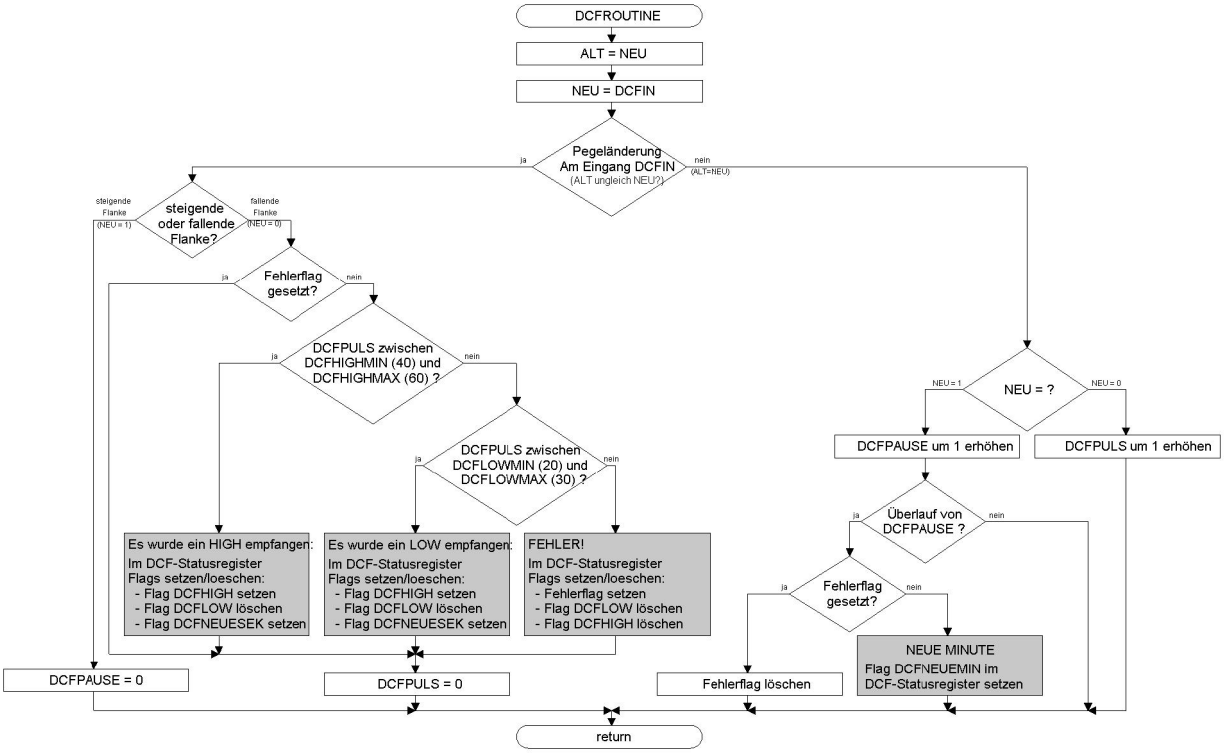


Bild 4.1: Flussdiagramm (DCFROUTINE)

Hier das Unterprogramm:

```
DCFROUTINE  bcf   DCFSTATUS,DCFPORTALT    ;DCFSTATUS,DCFPORTNEU ->
                                          ;  DCFSTATUS,DCFPORTALT
            btfsc DCFSTATUS,DCFPORTNEU
            bsf   DCFSTATUS,DCFPORTALT

            movf  DCFINPORT,w
            movwf TEMP1
            bcf   DCFSTATUS,DCFPORTNEU    ;DCFIN -> DCFSTATUS,DCFPORTNEU,
                                          ;  TEMP1,DCFPORTALT
            btfsc TEMP1,DCFIN
            bsf   DCFSTATUS,DCFPORTNEU
            clrf  TEMP2
            btfsc TEMP1,DCFIN
            bsf   TEMP2,DCFPORTALT

            movf  DCFSTATUS,w
            xorwf TEMP2,f

            btfss TEMP2,DCFPORTALT        ;Hat sich am Pin DCFIN der Pegel
                                          ;  geaendert?
            goto  DCFINCPULSPAUSE         ;nein: Puls- bzw. Pause-Zaehler um 1
                                          ;  erhoehen
            btfss DCFSTATUS,DCFPORTNEU    ;ja: -> steigende oder fallende
                                          ;  Flanke?
            goto  DCFHLFLANKE             ;fallende Flanke
DCFLHFLANKE clrf  DCFPAUSE                ;steigende Flanke: Zaehlregister
                                          ;  DCFPAUSE
            goto  DCFROUTINEFERTIG        ;  loeschen und UP verlassen

            ;Fallende Flanke
DCFHLFLANKE btfss DCFSTATUS,DCFFEHLER     ;Fehlerflag gesetzt:
            goto    DCFPLAUSIBEL          ;nein: Ermittlung, ob die ermittelte
                                          ;  Pulszeit einem HIGH oder einem
                                          ;  LOW entspricht
```

```
                                        ;  (Plausibilitaetspruefung)
            goto  DCFPULSLOESCHEN       ;ja: Puls-Zaehler loeschen

DCFPLAUSIBEL ;Pruefen, ob ein High empfangen wurde
DCFCHECKHIGH
            movlw DCFHIGHMIN            ;unteren High-Grenzwert laden
            subwf DCFPULS,w
            btfss STAT,C                ;Grenzwert < DCFPULS ?
            goto  DCFCHECKLOW           ;nein: kein HIGH-Pegel, pruefen, ob
                                        ;  LOW
            movlw DCFHIGHMAX            ;ja: DCFPULS mit oberen HIGH-
                                        ; Grenzwert vergleichen
            subwf DCFPULS,w
            btfsc STAT,C                ;DCFPULS < Grenzwert ?
            goto  DCFCHECKLOW           ;nein: es handelt sich um kein HIGH-
                                        ;  Signal, pruefen, ob ein LOW
                                        ;  empfangen wurde
            bcf   DCFSTATUS,DCFLOW      ;ja: Es wurde ein High-Signal
                                        ;  empfangen, das Flag DCFLOW
                                        ;  loeschen,
            bsf   DCFSTATUS,DCFHIGH     ;  das Flag DCFHIGH setzen,
            bsf   DCFSTATUS,DCFNEUESEK  ;  das Flag DCFNEUESEK setzen
            goto  DCFPULSLOESCHEN       ;  und Puls-Zaehler loeschen

            ;Pruefen, ob ein Low empfangen wurde
DCFCHECKLOW movlw DCFLOWMIN             ;unteren Low-Grenzwert laden
            subwf DCFPULS,w
            btfss STAT,C                ;Grenzwert < DCFPULS ?
            goto  DCFSETFEHLER          ;nein: kein LOW-Pegel,
                                        ;  Empfangsfehler
            movlw DCFLOWMAX             ;ja: DCFPULS mit oberen LOW-
                                        ;  Grenzwert vergleichen
            subwf DCFPULS,w
            btfsc STAT,C                ;DCFPULS < Grenzwert ?
            goto  DCFSETFEHLER          ;nein: es handelt sich um kein LOW-
                                        ;  Signal -> Empfangsfehler
            bcf   DCFSTATUS,DCFHIGH     ;ja: Es wurde ein LOW-Signal
                                        ;  empfangen, das Flag DCFLOW
                                        ;  setzen,
            bsf   DCFSTATUS,DCFLOW      ;  das Flag DCFHIGH loeschen,
            bsf   DCFSTATUS,DCFNEUESEK  ;  das Flag DCFNEUESEK setzen
            goto  DCFPULSLOESCHEN       ;  und den Puls-Zaehler loeschen

            ;Empfangsfehler
DCFSETFEHLER
            bsf   DCFSTATUS,DCFFEHLER   ;Bei einem Empfangsfehler
            bcf   DCFSTATUS,DCFHIGH     ;  Fehlerflag (DCFFEHLER) setzen,
            bcf   DCFSTATUS,DCFLOW      ;  die Flags DCFLOW und DCFHIGH
                                        ;  loeschen,
            goto  DCFPULSLOESCHEN       ;  und den Puls-Zaehler loeschen

            ;Puls- oder Pausen-Zaehler erhoehen
DCFINCPULSPAUSE
            btfsc DCFSTATUS,DCFPORTNEU
            goto  DCFINCPAUSE

DCFINCPULS  incf  DCFPULS,f             ;Puls-Zaehler um 1 erhoehen
            goto  DCFROUTINEFERTIG

DCFINCPAUSE incfsz DCFPAUSE,f           ;Pause-Zaehler erhoehen und auf
                                        ;  Ueberlauf pruefen
            goto  DCFROUTINEFERTIG      ;kein Ueberlauf: Unterprogramm
                                        ;  verlassen
```

```
            btfsc DCFSTATUS,DCFFEHLER     ;Ueberlauf: NEUE MINUTE: wenn
                                          ;  Fehlerflag
            goto  DCFROUTINEW1

            bsf   DCFSTATUS,DCFNEUEMIN    ;nicht gesetzt, Flag DCFNEUEMIN
                                          ;  setzen
            goto  DCFROUTINEFERTIG        ;  und Unterprogramm verlassen

DCFROUTINEW1
            bcf   DCFSTATUS,DCFFEHLER     ;Fehlerflag gesetzt: Fehlerflag
                                          ;  loeschen
            goto  DCFROUTINEFERTIG        ;  und Unterprogramm verlassen (Flag
                                          ;  DCFNEUEMIN wird in diesem Fall
                                          ;  nicht gesetzt)

            ;Puls-Zaehler loeschen
DCFPULSLOESCHEN
            clrf  DCFPULS

DCFROUTINEFERTIG
            return
```

Anmerkung:
Die temporären Register TEMP1 und TEMP2 werden hier nur als Hilfsregister benötigt. Sie können daher auch in anderen Unterprogrammen verwendet werden.

4.5.3. DCFUPSEKUNDE

Aufgaben:
- Je nachdem ob das Bit DCFLOW oder das Bit DCFHIGH gesetzt ist dieses Bit dem Telegramm (Register DCFTELEGRAMM1 bis DCFTELEGRAMM8) hinzufügen
- Anforderungsflag (Flag DCFNEUESEK) zurücksetzen

Vorgehensweise:
Ist das Flag DCFLOW (im Register DCFSTATUS) gesetzt das Carryflag (im SFR STAT) löschen, ist aber das Flag DCFHIGH (ebenfalls im Register DCFSTATUS) gesetzt das Carryflag setzten. Dieses Carryflag nun dem Telegramm hinzufügen. Dieser Vorgang erfolgt mit einem so genannten Schiebebefehl. Dabei werden alle Bits des Registers DCFTELEGRAMM1 auf die nächst höhere Position verschoben. (Bit 6 wandert ins Bit 7, Bit 5 wandert ins Bit 6 usw. Bit 0 wandert ins Bit 1). Der Inhalt vom Carryflag wandert ins Bit 0. Der Inhalt von Bit 7 wird ins Carryflag geschoben. Bei den Registern DCFTELEGRAMM2 bis DCFTELEGRAMM6 erfolgt derselbe Vorgang. (Der Inhalt von Bit 7 des Registers DCFTELEGRAMM1 wird ins Carry geschoben, und das Carry aber weiter in das Bit 0 des Registers DCFTELEGRAMM2)

Anmerkung:
Die Flags DCFLOW und DCFHIGH können nie gleichzeitig gesetzt sein. Es ist nur möglich, dass entweder nur DCFLOW oder nur DCFHIGH gesetzt ist, aber nie beide gleichzeitig.

Hier das Unterprogramm:

```
DCFUPSEKUNDE
            btfsc DCFSTATUS,DCFLOW  ;Ist das Flag DCFLOW im Register DCFSTATUS
                                    ;  gesetzt?
```

```
            goto  DCFSCHIEBELOW
            btfsc DCFSTATUS,DCFHIGH ;nein: Wenn das FLAG DCFHIGH im Register
DCFSCHIEBEHIGH
            bsf   STAT,C            ;  DCFSTATUS gesetzt ist, Dem Telegramm
            goto  DCFSCHIEBE        ;  ein Low mit Hilfe des Carry hinzufuegen
DCFSCHIEBELOW
            bcf   STAT,C            ;ja: Dem Telegramm ein High mit Hilfe des
                                    ;  Carry hinzufuegen
DCFSCHIEBE  rlf   DCFTELEGRAMM1,f
            rlf   DCFTELEGRAMM2,f
            rlf   DCFTELEGRAMM3,f
            rlf   DCFTELEGRAMM4,f
            rlf   DCFTELEGRAMM5,f
            rlf   DCFTELEGRAMM6,f
;           rlf   DCFTELEGRAMM7,f
;           rlf   DCFTELEGRAMM8,f

            bcf   DCFSTATUS,DCFNEUESEK ;Anforderungsflag loeschen
            return
```

4.5.4. DCFUPMINUTE

Aufgaben:

- Fehlende 59. Sekunde "nachholen" (Unterprogramm DCFUPSEKUNDE aufrufen)
- Datum, Uhrzeit und die Zusatzinformationen aus den Registern DCFTELEGRAMM1 bis DCFTELEGRAMM8 zusammensetzen
- Das soeben ermittelte Telegramm mit dem Telegramm der vorhergehenden Minute vergleichen. Die soeben ermittelte Minute muss dabei um 1 größer als die vorhergehende Minute sein, und die soeben ermittelten Werte für die Stunde, Tag, Monat, Jahr und Wochentag muss mit den Werten des vorhergehenden Telegramms übereinstimmen. ACHTUNG: Bei dieser einfachen Überprüfung wird der Übergang von z. B. 13:59 auf 14:00 als falsch gewertet, da sich hier die Stunde ändert, und auch die Minute um mehr als 1. Würden hier alle möglichen Übergänge berücksichtigt, dann würde dieses Unterprogramm noch umfangreicher als es ohnehin schon ist. Durch diese Vereinfachung dauert die Synchronisierung im schlimmsten Fall nur eine Minute länger.
- Das alte Datum bzw. die alte Uhrzeit (von der vorhergehenden Minute) durch das neue Datum bzw. Uhrzeit ersetzen. Dies ist für die Überprüfung des nächsten Telegramms notwendig.
- Wenn das neu empfangene Datum bzw. die neu empfangene Uhrzeit gültig ist, die BCD-kodierten Datums- bzw. Uhrzeitkomponenten in eine binäre Form umwandeln und damit die mitlaufende Uhr synchronisieren. Das Flag DCFSYNC im Register DCFSTATUS setzen. Dieses gesetzte Flag kennzeichnet, dass die mitlaufende Uhr mit der DCF-Uhr synchronisiert ist.
- Anforderungsflag (Flag DCFNEUEMIN im Register DCFSTATUS) zurücksetzen

Hier das Unterprogramm:

```
DCFUPMINUTE call  DCFUPSEKUNDE      ;Fehlende Sekunde nachholen

            clrf  DCFJAHRBCD        ;Jahr zusammensetzen (BCD-Format)
            btfsc DCFTELEGRAMM1,1
            bsf   DCFJAHRBCD,7
            btfsc DCFTELEGRAMM1,2
            bsf   DCFJAHRBCD,6
            btfsc DCFTELEGRAMM1,3
```

```
bsf   DCFJAHRBCD,5
btfsc DCFTELEGRAMM1,4
bsf   DCFJAHRBCD,4
btfsc DCFTELEGRAMM1,5
bsf   DCFJAHRBCD,3
btfsc DCFTELEGRAMM1,6
bsf   DCFJAHRBCD,2
btfsc DCFTELEGRAMM1,7
bsf   DCFJAHRBCD,1
btfsc DCFTELEGRAMM2,0
bsf   DCFJAHRBCD,0

clrf  DCFMONBCD         ;Monat zusammensetzen (BCD-Format)
btfsc DCFTELEGRAMM2,1
bsf   DCFMONBCD,4
btfsc DCFTELEGRAMM2,2
bsf   DCFMONBCD,3
btfsc DCFTELEGRAMM2,3
bsf   DCFMONBCD,2
btfsc DCFTELEGRAMM2,4
bsf   DCFMONBCD,1
btfsc DCFTELEGRAMM2,5
bsf   DCFMONBCD,0

clrf  DCFWOCHENTAG      ;Wochentag zusammensetzen (BCD-Format)
btfsc DCFTELEGRAMM2,6
bsf   DCFWOCHENTAG,2
btfsc DCFTELEGRAMM2,7
bsf   DCFWOCHENTAG,1
btfsc DCFTELEGRAMM3,0
bsf   DCFWOCHENTAG,0

clrf  DCFTAGBCD         ;Tag zusammensetzen (BCD-Format)
btfsc DCFTELEGRAMM3,1
bsf   DCFTAGBCD,5
btfsc DCFTELEGRAMM3,2
bsf   DCFTAGBCD,4
btfsc DCFTELEGRAMM3,3
bsf   DCFTAGBCD,3
btfsc DCFTELEGRAMM3,4
bsf   DCFTAGBCD,2
btfsc DCFTELEGRAMM3,5
bsf   DCFTAGBCD,1
btfsc DCFTELEGRAMM3,6
bsf   DCFTAGBCD,0

clrf  DCFSTDBCD         ;Stunde zusammensetzen (BCD-Format)
btfsc DCFTELEGRAMM4,0
bsf   DCFSTDBCD,5
btfsc DCFTELEGRAMM4,1
bsf   DCFSTDBCD,4
btfsc DCFTELEGRAMM4,2
bsf   DCFSTDBCD,3
btfsc DCFTELEGRAMM4,3
bsf   DCFSTDBCD,2
btfsc DCFTELEGRAMM4,4
bsf   DCFSTDBCD,1
btfsc DCFTELEGRAMM4,5
bsf   DCFSTDBCD,0

clrf  DCFMINBCD         ;Minute zusammensetzen (BCD-Format)
btfsc DCFTELEGRAMM4,7
bsf   DCFMINBCD,6
```

```
                btfsc DCFTELEGRAMM5,0
                bsf   DCFMINBCD,5
                btfsc DCFTELEGRAMM5,1
                bsf   DCFMINBCD,4
                btfsc DCFTELEGRAMM5,2
                bsf   DCFMINBCD,3
                btfsc DCFTELEGRAMM5,3
                bsf   DCFMINBCD,2
                btfsc DCFTELEGRAMM5,4
                bsf   DCFMINBCD,1
                btfsc DCFTELEGRAMM5,5
                bsf   DCFMINBCD,0

;               clrf  DCFZUSATZINFOS          ;Zusaetzliche Informationen
;               btfsc DCFTELEGRAMM5,7         ;  zusaetzliche Schaltsekunde
;               bsf   DCFZUSATZINFOS,DCFSCHALTSEK
;               btfsc DCFTELEGRAMM6,0         ;  Sommerzeit
;               bsf   DCFZUSATZINFOS,DCFSOMMER
;               btfsc DCFTELEGRAMM6,1         ;  Winterzeit
;               bsf   DCFZUSATZINFOS,DCFWINTER
;               btfsc DCFTELEGRAMM6,2         ;  Vorankuendigung: Wechsel
;               bsf   DCFZUSATZINFOS,DCFSOMWIN ;   Sommerzeit <-> Winterzeit
;               btfsc DCFTELEGRAMM6,3         ;  Reserveantenne
;               bsf   DCFZUSATZINFOS,DCFRESANT

                ;Neues Telegramm mit dem alten Vergleichen
                bcf   DCFSTATUS,DCFFEHLER     ;zuerst Fehlerflag loeschen

                movf  DCFMINALT,w             ;Minuten pruefen
                subwf DCFMINBCD,w
                movwf TEMP1                   ;TEMP1 = DCFMINBCD - DCFMINALT
                decfsz TEMP1,w                ;TEMP1 - 1 = 0 ?
                bsf   DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf  DCFSTDALT,w             ;Stunden pruefen
                subwf DCFSTDBCD,w
                btfss STAT,Z                  ;DCFSTDBCD - DCFSTDALT = 0?
                bsf   DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf  DCFTAGALT,w             ;Tag pruefen
                subwf DCFTAGBCD,w
                btfss STAT,Z                  ;DCFTAGBCD - DCFTAGBCD = 0?
                bsf   DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf  DCFMONALT,w             ;Monat pruefen
                subwf DCFMONBCD,w
                btfss STAT,Z                  ;DCFMONBCD - DCFMONALT = 0?
                bsf   DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf  DCFJAHRALT,w            ;Jahr pruefen
                subwf DCFJAHRBCD,w
                btfss STAT,Z                  ;DCFJAHRBCD - DCFJAHRALT = 0?
                bsf   DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf  DCFWTAGALT,w            ;Wochentag pruefen
                subwf DCFWOCHENTAG,w
                btfss STAT,Z                  ;DCFWOCHENTAG - DCFWTAGALT = 0?
                bsf   DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                ;Altes Telegramm durch das neue Telegramm ersetzen
                movf  DCFMINBCD,w
                movwf DCFMINALT
```

```
            movf  DCFSTDBCD,w
            movwf DCFSTDALT

            movf  DCFTAGBCD,w
            movwf DCFTAGALT

            movf  DCFMONBCD,w
            movwf DCFMONALT

            movf  DCFJAHRBCD,w
            movwf DCFJAHRALT

            movf  DCFWOCHENTAG,w
            movwf DCFWTAGALT

            btfsc DCFSTATUS,DCFFEHLER     ;DCF-Telegramm korrekt?
            goto  DCFUPMINUTEENDE         ;nein: UP beenden
            movf  DCFSTDBCD,w             ;ja: DCFSTDBCD
            call  BCDBIN2                 ;von BCD nach binaer umwandeln
            movwf UHRSTUNDE               ;und die Stunde der mitlaufenden Uhr
                                          ;ueberschreiben

            movf  DCFMINBCD,w             ;DCFMINBCD
            call  BCDBIN2                 ;von BCD nach binaer umwandeln
            movwf UHRMINUTE               ;und die Minute der mitlaufenden Uhr
                                          ;ueberschreiben

            clrf  UHRSEKUNDE              ;Sekunde loeschen

            movf  DCFTAGBCD,w             ;DCFTAGBCD
            call  BCDBIN2                 ;von BCD nach binaer umwandeln
            movwf DATUMTAG                ;und den Tag des mitlaufenden Datums
                                          ;ueberschreiben

            movf  DCFMONBCD,w             ;DCFMONBCD
            call  BCDBIN2                 ;von BCD nach binaer umwandeln
            movwf DATUMMONAT              ;und den Monat des mitlaufenden
                                          ;Datums ueberschreiben

            movf  DCFJAHRBCD,w            ;DCFJAHRBCD
            call  BCDBIN2                 ;von BCD nach binaer umwandeln
            movwf DATUMJAHR               ;und das Jahr des mitlaufenden
                                          ;Datums ueberschreiben

            movf  DCFWOCHENTAG,w          ;DCFWOCHENTAG
            movwf DATUMWTAG

            bsf   DCFSTATUS,DCFSYNC

DCFUPMINUTEENDE   bcf   DCFSTATUS,DCFFEHLER
            bcf   DCFSTATUS,DCFNEUEMIN    ;Anforderungsflag loeschen

            return
```

Anmerkung:
Das temporäre Register TEMP1 wird hier nur als Hilfsregister benötigt, und kann daher auch in anderen Unterprogrammen verwendet werden.

4.5.5. INNEREUHR

Aufgabe:
Für den Fall dass kein gültiges DCF-Telegramm empfangen werden kann sorgt dieses Unterprogramm dafür, dass die Uhrzeit trotzdem jede Sekunde aktualisiert wird. Ist für eine längere Zeit kein korrekter DCF-Empfang möglich, so würden die Minuten, Stunden, Tage usw. stehen bleiben, da ja im Unterprogramm DCFUPMINUTE nur gültige DCF-Telegramme übernommen werden.
Dieses Unterprogramm wird also parallel zu den Unterprogrammen für die DCF-Dekodierung aufgerufen.

Vorgehensweise:
Die Sekunden (Register UHRSEKUNDE) um 1 erhöhen. Beinhalte dieses Register nun den Wert 60, so beginnt eine neue Minute. Daher die Sekunden löschen und die Minuten (Register UHRMINUTE) um 1 erhöhen. Beinhaltet dieses Register nun den Wert 60, so beginnt eine neue Stunde. Daher die Minuten löschen und die Stunden (Register UHRSTUNDE) um 1 erhöhen. Beinhaltet dieses Register nun den Wert 24, so beginnt ein neuer Tag. Daher die Stunden löschen. Ist das Mitzählen des Datums notwendig, so muss nun ein Register für den Tag (z.B. DATUMTAG) um 1 erhöht werden und dieses Überprüft werden, wobei diese Prüfung nun nicht mehr so einfach ist, da ja jeder Monat unterschiedlich viele Tage besitzt. Erschwerend kommt auch noch hinzu, dass auch die Schaltjahre miteinbezogen werden müssen!

Achtung:
Dieses Unterprogramm muss daher jede Sekunde aufgerufen werden

Hier das Unterprogramm:

```
INNEREUHR   incf  UHRSEKUNDE,f      ;Sekundenzähler um 1 erhöhen
            movf  UHRSEKUNDE,w
            sublw .60
            btfss STAT,Z            ;Pruefen, ob Sekunde=60
            goto  INNERERUHRFERTIG
            clrf  UHRSEKUNDE        ;Wenn Sekunde=60, Sekunde loeschen
            incf  UHRMINUTE,f       ;  und Minute um 1 erhöhen
            movf  UHRMINUTE,w
            sublw .60
            btfss STAT,Z            ;Pruefen, ob Minute=60
            goto  INNERERUHRFERTIG
            clrf  UHRMINUTE         ;Wenn Minute=60, Minute loeschen
            incf  UHRSTUNDE,f       ;  und Stunde erhoehen
            movf  UHRSTUNDE,w
            sublw .24
            btfsc STAT,Z            ;Pruefen, ob Stunde=24
            clrf  UHRSTUNDE         ;Wenn Stunde=24, Stunden loeschen
INNERERUHRFERTIG
            return
```

4.5.6. ANZEIGE

Dieses Unterprogramm wird vom Hauptprogramm jede Sekunde aufgerufen.

Aufgabe:
Wenn die mitlaufende Uhr mit der DCF-Uhr synchronisiert ist (Flag DCFSYNC im Register DCFSTATUS gesetzt) die Uhrzeit wie folgt ausgeben. Andenfalls dieses Unterprogramm vorzeitig beenden:

- Die Stunden von "24-nach-12" umwandeln und um 1 erhoehen. In das höherwertige Nibble von TEMP2 kopieren
- Die Minuten durch 5 dividieren, um 1 erhoehen und in das niederwertige Nibble von TEMP2 kopieren
- TEMP2 am Port B ausgeben

Hier das Unterprogramm:

```
ANZEIGE     btfss DCFSTATUS,DCFSYNC ;Ist die mitlaufende Uhr mit der DCF-Uhr
                                    ;  synchronisiert?
            goto  ANZEIGEENDE       ;nein: Unterprogramm vorzeitig beenden

            clrf  TEMP2             ;ja: Hilfsregister (TEMP2) loeschen

            movf  UHRSTUNDE,w       ;Die Stunde
            andlw b'00011111'

            call  TABSTUNDEN        ;  mit Hilfe der Tabelle TABSTUNDEN
                                    ;  umwandeln
                                    ;  (von 24 nach 12 und um 1 erhoehen)
            movwf TEMP2             ;  und in das hoeherwertige Nibble von
            swapf TEMP2,f           ;  TEMP2 kopieren

            movf  UHRMINUTE,w       ;Die Minuten
            andlw b'00111111'
            call  TABDIV5           ;  mit Hilfe der Tabelle TABDIV5 durch 5
            movwf TEMP1             ;  dividieren, um 1 erhoehen und in das
            incf  TEMP1,w
            xorwf TEMP2,w           ;niederwertigere Nibble von TEMP2 kopieren
            movwf PORTB             ;TEMP2 am Port B ausgeben

ANZEIGEENDE return
```

Anmerkung:
Die temporären Register TEMP1 und TEMP2 werden hier nur als Hilfsregister benötigt. Sie können daher auch in anderen Unterprogrammen verwendet werden.

4.5.7. BCDBIN2

Aufgabe:
Die im w-Register stehende 2-stellige-BCD-Zahl nach Binär umwandeln. Das Ergebnis befindet sich wieder im w-Register.

Vorgehensweise:

- Die umzuwandelnde BCD-Zahl im temporären Register TEMP1 zwischenspeichern
- w-Register (Arbeitsregister) löschen.
- Die BCD-Zahl bitweise überprüfen.

```
z.B.: BCD-Zahl:        0010 0110   (= 26)
                        |||| ||||
                        |||| ||| -        0  ( = 0 x 1)
                        |||| || ---  +    2  ( = 1 x 2)
                        |||| | -----  +    4  ( = 1 x 4)
                        ||||  ----   +    0  ( = 0 x 8)
                        ||| ------    +    0  ( = 0 x 10)
                        || -------    +   20  ( = 1 x 20)
                        |--------     +    0  ( = 0 x 40)
                         ---------    +    0  ( = 0 x 80)
                       ------------------------------
                                    =   26
```

- Ergebnis in w

Hier das Unterprogramm:

```
BCDBIN2     movwf TEMP1       ;umzuwandelnde BCD-Zahl zwischenspeichern
            clrw              ;Arbeitsregister (w-Register) loeschen
            btfsc TEMP1,0     ;Ist Bit0 der BCD-Zahl gesetzt,
            addlw .1          ;  zum w-Register den Wert 1 addieren
            btfsc TEMP1,1     ;Ist Bit1 der BCD-Zahl gesetzt,
            addlw .2          ;  zum w-Register den Wert 2 addieren
            btfsc TEMP1,2     ;Ist Bit2 der BCD-Zahl gesetzt,
            addlw .4          ;  zum w-Register den Wert 4 addieren
            btfsc TEMP1,3     ;Ist Bit3 der BCD-Zahl gesetzt,
            addlw .8          ;  zum w-Register den Wert 8 addieren
            btfsc TEMP1,4     ;Ist Bit4 der BCD-Zahl gesetzt,
            addlw .10         ;  zum w-Register den Wert 10 addieren
            btfsc TEMP1,5     ;Ist Bit5 der BCD-Zahl gesetzt,
            addlw .20         ;  zum w-Register den Wert 20 addieren
            btfsc TEMP1,6     ;Ist Bit6 der BCD-Zahl gesetzt,
            addlw .40         ;  zum w-Register den Wert 40 addieren
            btfsc TEMP1,7     ;Ist Bit7 der BCD-Zahl gesetzt,
            addlw .80         ;  zum w-Register den Wert 80 addieren, im w-
                              ;  Register befindet sich nun das Ergebnis
            return
```

Anmerkung:
Das temporäre Register TEMP1 wird hier nur zum Zwischenspeichern benötigt, und kann daher auch in anderen Unterprogrammen verwendet werden.

5. Nachbauhinweise

Schritt 1: (Lochraster)-Platinen bestücken

Die Elektronik dieses Projektes wurde auf mehrere kleine Lochrasterplatinen aufgeteilt. Diese entsprechen in etwa der Einteilung bei der Schaltungsbeschreibung. Die Leuchtdioden für die Anzeige (also das Ziffernblatt) ist auf 12 gleiche (Lochraster)-Platinen aufgeteilt.
Selbstverständlich kann eine eigene Platine entworfen werden, auf der die gesamte Elektronik inklusive „Ziffernblatt“ untergebracht ist.

Das Bestücken einer Platine ist erst dann sinnvoll, wenn alle für diese Platine benötigten Bauteile vorhanden sind. Es sollten generell nur erstklassige und neuwertige Bauteile verwendet werden. Auf Bauteile aus ausgeschlachteten Geräten sollte grundsätzlich verzichtet werden, da ihre Funktionalität nicht gewährleistet ist, und eine unnötige Fehlersuche dadurch vermieden werden kann.

Weiters sollte ausreichend Platz und vor allem ausreichend Zeit für die Bestückung und Verdrahtung der Platinen vorhanden sein.

Controllerplatine:
Die Bauelemente entsprechend Bild 5.1, der folgenden Reihenfolge, der Stückliste (Anhang C) und dem Schaltplan (Anhang A) bestücken wobei zunächst nur die Bauteile angelötet werden. Diese jedoch noch nicht verdrahten. Das Verdrahten erfolgt erst wenn alle Bauteile angelötet sind. Die nach dem anlöten überstehenden Anschlüsse mit einem kleinen Seitenschneider entfernen.

Reihenfolge zur Bearbeitung und Bestückung der Controllerplatine:
- Platine auf eine Größe von 43 x 26 mm zuschneiden und die geschnittenen Seiten abschleifen
- 3,5-mm-Bohrungen für die Montage der Platine auf einer Bodenplatte (4 Bohrungen)
- Drahtbrücke nicht vergessen!
- Widerstände R13 (10k) und R14 (270 Ohm): **Tipp**: Vor dem Einlöten des Widerstandes diesen überprüfen, auch wenn die Bauteile in einem Regal sortiert sind. (z.B. mit einem Multimeter). Die Praxis hat gezeigt, dass sich hin und wieder doch falsche Bauteilwerte in das Regal eingeschlichen haben. Dies gilt nicht nur für Widerstände, sondern auch für Dioden, Kondensatoren, Transistoren usw.
- IC-Fassung für IC1 (18polig): **Tipp 1:** Obwohl es bei den Fassungen elektrisch gesehen egal ist wie die Fassungen eingelötet sind, sollte man doch die Fassung so einlöten, dass die Kerbe in die richtige Richtung zeigt. Dies erleichtert das spätere Einsetzten der ICs bzw. erleichtert die Arbeit bei einem IC-Tausch. **Tipp 2:** Beim Einlöten der Fassungen sollte man wie folgt vorgehen: Fassung an der einzusetzenden Stelle lagerichtig einsetzen und zunächst nur einen beliebigen Eckpin anlöten. Fassung kontrollieren und eventuell Nachlöten. Sitzt die Fassung ordentlich, den gegenüberliegenden Pin anlöten. Fassung wieder kontrollieren und eventuell nachlöten. Erst wenn Sie mit der Lage der Fassung zufrieden sind, die restlichen Pins anlöten.
- Keramikkondensatoren C3 und C4 (je 22pF)
- Keramikkondensator C1 (100nF)
- Elko C2 (10µF/25V): **Achtung:** Polarität beachten!
- Quarz X1 (4,096 MHz)

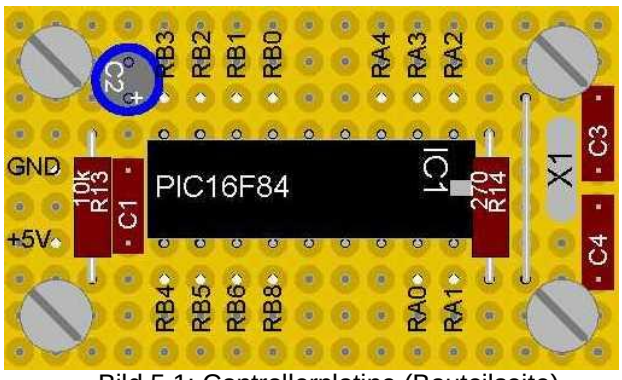


Bild 5.1: Controllerplatine (Bauteilseite)

Nachdem alle Bauteile eingelötet wurden, die Verbindungen auf der Lötseite entsprechend Bild 5.2 mit einem dünnen Draht bzw. mit isolierten Drähten herstellen. Dabei sollte sorgfältigst gearbeitet werden, und man sollte sich dafür ausreichend Zeit nehmen. Befinden sich Verbindungen direkt nebeneinander, so können diese auch nur mit Lötzinn verbunden werden. **Tipp:** Bild 5.2 ausdrucken, und während dem Verlöten die soeben durchgeführte Verbindung im Ausdruck kennzeichnen. Auf diese Weise können Fehler durch fehlende Verbindungen vermieden werden.
Zum Schluss alle Verbindungen und Lötstellen noch einmal sorgfältig überprüfen. (Auch hierfür sollte man sich ausreichend Zeit nehmen!)

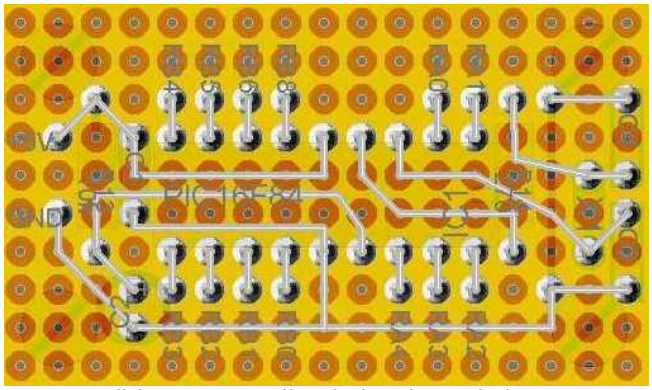


Bild 5.2: Controllerplatine (Lötseite)

Platine für Anpass-Schaltung:
Die Bauelemente entsprechend Bild 5.3, der folgenden Reihenfolge, der Stückliste (Anhang C) und dem Schaltplan (Anhang A) bestücken wobei zunächst nur die Bauteile angelötet werden. Diese jedoch noch nicht verdrahten. Das Verdrahten erfolgt erst wenn alle Bauteile angelötet sind. Die nach dem anlöten überstehenden Anschlüsse mit einem kleinen Seitenschneider entfernen.

Reihenfolge zur Bearbeitung und Bestückung der Anpassplatine:
- Platine auf eine Größe von 28 x 23 mm zuschneiden und die geschnittenen Seiten abschleifen
- 3,5-mm-Bohrungen für die Montage der Platine auf einer Bodenplatte (4 Bohrungen)
- Widerstände R15, R17 (je 10k) und R16 (470 Ohm): **Tipp**: Vor dem Einlöten des Widerstandes diesen überprüfen, auch wenn die Bauteile in einem Regal sortiert sind. (z.B. mit einem Multimeter). Die Praxis hat gezeigt, dass sich hin und wieder doch falsche Bauteilwerte in das Regal eingeschlichen haben. Dies gilt nicht nur für Widerstände, sondern auch für Dioden, Kondensatoren, Transistoren usw..
- Keramikkondensator C5 (100nF)
- Leuchtdiode D157 (3mm gelb): **Achtung:** Polarität beachten! Der längere Anschluss ist die Kathode (plus), der kürzere demnach die Anode (minus).
- Transistor T1 (BC547B): **Achtung:** Polarität beachten!
- Stiftleiste für Jumper JP1

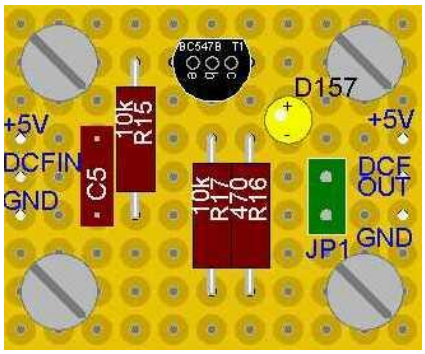


Bild 5.3: Anpassplatine (Bauteilseite)

Nachdem alle Bauteile eingelötet wurden, die Verbindungen auf der Lötseite entsprechend Bild 5.4 mit einem dünnen Draht herstellen. Dabei sollte sorgfältigst gearbeitet werden, und man sollte sich dafür ausreichend Zeit nehmen. Befinden sich Verbindungen direkt nebeneinander, so können diese auch nur mit Lötzinn verbunden werden. **Tipp:** Bild 5.4 ausdrucken, und während dem Verlöten die soeben durchgeführte Verbindung im Ausdruck kennzeichnen. Auf diese Weise können Fehler durch fehlende Verbindungen vermieden werden.
Zum Schluss alle Verbindungen und Lötstellen noch einmal sorgfältig überprüfen. (Auch hierfür sollte man sich ausreichend Zeit nehmen!)

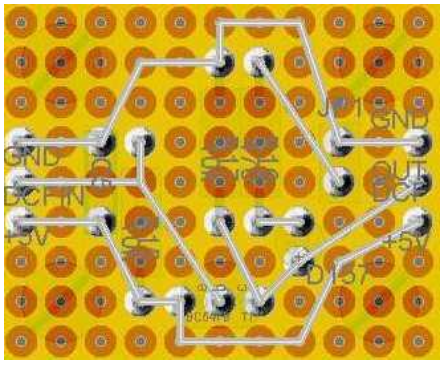


Bild 5.4: Anpassplatine (Lötseite)

Platine zur Ansteuerung der LED-Matrix:
Die Bauelemente entsprechend Bild 5.5, der folgenden Reihenfolge, der Stückliste (Anhang C) und dem Schaltplan (Anhang A) bestücken wobei zunächst nur die Bauteile angelötet werden. Diese jedoch noch nicht verdrahten. Das Verdrahten erfolgt erst wenn alle Bauteile angelötet sind. Die nach dem anlöten überstehenden Anschlüsse mit einem kleinen Seitenschneider entfernen.

Reihenfolge zur Bearbeitung und Bestückung der Ansteuerplatine für die LED-Matrix:
- Platine auf eine Größe von 84 x 23 mm zuschneiden und die geschnittenen Seiten abschleifen
- 3,5-mm-Bohrungen für die Montage der Platine auf einer Bodenplatte (4 Bohrungen)
- Drahtbrücken (2 Stk): **Achtung:** diese befinden sich zum Teil unter IC3
- Widerstände R18 (820 Ohm), R19 (680 Ohm) und R20 (1k5): **Tipp**: Vor dem Einlöten des Widerstandes diesen überprüfen, auch wenn die Bauteile in einem Regal sortiert sind. (z.B. mit einem Multimeter). Die Praxis hat gezeigt, dass sich hin und wieder doch falsche Bauteilwerte in das Regal eingeschlichen haben. Dies gilt nicht nur für Widerstände, sondern auch für Dioden, Kondensatoren, Transistoren usw.

- IC-Fassungen für IC2 und IC3 (je 24polig): **Tipp 1:** Obwohl es bei den Fassungen elektrisch gesehen egal ist wie die Fassungen eingelötet sind, sollte man doch die Fassung so einlöten, dass die Kerbe in die richtige Richtung zeigt. Dies erleichtert das spätere Einsetzten der ICs bzw. erleichtert die Arbeit bei einem IC-Tausch. **Tipp 2:** Beim Einlöten der Fassungen sollte man wie folgt vorgehen: Fassung an der einzusetzenden Stelle lagerichtig einsetzen und zunächst nur einen beliebigen Eckpin anlöten. Fassung kontrollieren und eventuell Nachlöten. Sitzt die Fassung ordentlich, den gegenüberliegenden Pin anlöten. Fassung wieder kontrollieren und eventuell nachlöten. Erst wenn Sie mit der Lage der Fassung zufrieden sind, die restlichen Pins anlöten.
- Keramikkondensatoren C6 und C7 (je 100nF)

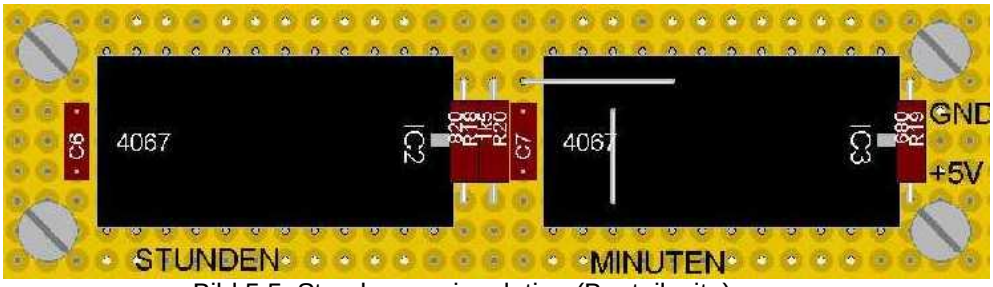


Bild 5.5: Stundenanzeigeplatine (Bauteilseite)

Nachdem alle Bauteile eingelötet wurden, die Verbindungen auf der Lötseite entsprechend Bild 5.6 mit einem dünnen Draht herstellen. Dabei sollte sorgfältigst gearbeitet werden, und man sollte sich dafür ausreichend Zeit nehmen. Befinden sich Verbindungen direkt nebeneinander, so können diese auch nur mit Lötzinn verbunden werden. **Tipp:** Bild 5.6 ausdrucken, und während dem Verlöten die soeben durchgeführte Verbindung im Ausdruck kennzeichnen. Auf diese Weise können Fehler durch fehlende Verbindungen vermieden werden.
Zum Schluss alle Verbindungen und Lötstellen noch einmal sorgfältig überprüfen. (Auch hierfür sollte man sich ausreichend Zeit nehmen!)

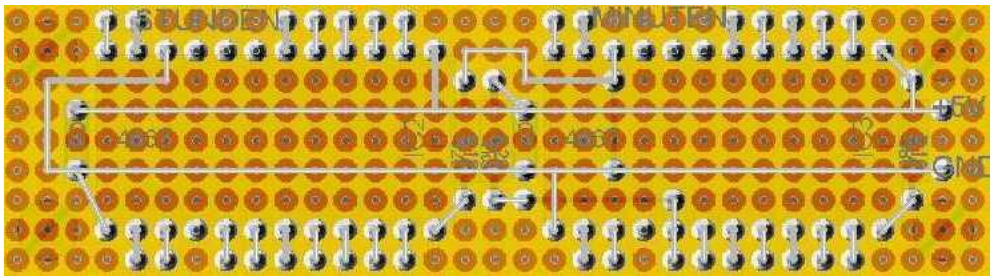


Bild 5.6: Stundenanzeigeplatine (Lötseite)

LED-Platine (Ziffernblatt):
Achtung, diese Platine wird 12mal benötigt!

Die Bauelemente entsprechend Bild 5.7, der folgenden Reihenfolge, der Stückliste (Anhang C) und dem Schaltplan (Anhang A) bestücken wobei zunächst nur die Bauteile angelötet werden. Diese jedoch noch nicht verdrahten. Das Verdrahten erfolgt erst wenn alle Bauteile angelötet sind. Die nach dem anlöten überstehenden Anschlüsse mit einem kleinen Seitenschneider entfernen.

Reihenfolge zur Bearbeitung und Bestückung der LED-Platine:

- Platine auf eine Größe von 63 x 21 mm zuschneiden und die geschnittenen Seiten abschleifen
- 3,5-mm-Bohrungen für die Montage der Platine auf einer Bodenplatte (4 Bohrungen)
- Widerstand R1 (10k): **Tipp**: Vor dem Einlöten des Widerstandes diesen überprüfen, auch wenn die Bauteile in einem Regal sortiert sind. (z.B. mit einem Multimeter). Die Praxis hat gezeigt, dass sich hin und wieder doch falsche Bauteilwerte in das Regal eingeschlichen haben. Dies gilt nicht nur für Widerstände, sondern auch für Dioden, Kondensatoren, Transistoren usw.
- Diode D145 (1N4148): **Achtung:** Polarität beachten!
- Leuchtdioden D2 bis D12 (3mm, rot) und D1 (5mm grün): **Achtung:** Polarität beachten! Der längere Anschluss ist die Kathode (plus), der kürzere demnach die Anode (minus).

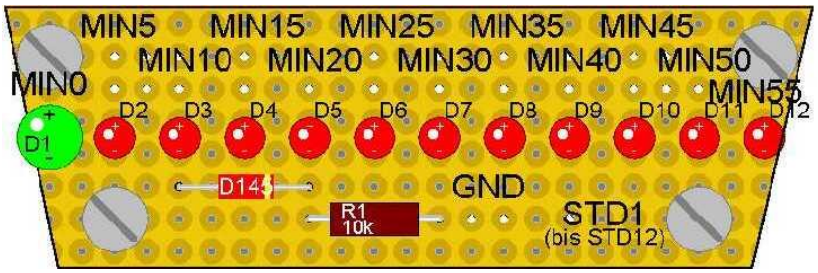


Bild 5.7: LED-Platine (Bauteilseite)

Nachdem alle Bauteile eingelötet wurden, die Verbindungen auf der Lötseite entsprechend Bild 5.8 mit einem dünnen Draht herstellen. Dabei sollte sorgfältigst gearbeitet werden, und man sollte sich dafür ausreichend Zeit nehmen. Befinden sich Verbindungen direkt nebeneinander, so können diese auch nur mit Lötzinn verbunden werden. **Tipp:** Bild 5.8 ausdrucken, und während dem Verlöten die soeben durchgeführte Verbindung im Ausdruck kennzeichnen. Auf diese Weise können Fehler durch fehlende Verbindungen vermieden werden.
Zum Schluss alle Verbindungen und Lötstellen noch einmal sorgfältig überprüfen. (Auch hierfür sollte man sich ausreichend Zeit nehmen!)

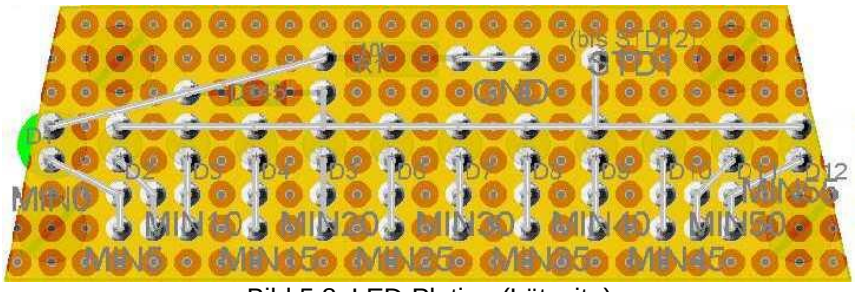


Bild 5.8: LED-Platine (Lötseite)

Stromversorgungsplatine:
Die Bauelemente entsprechend Bild 5.9, der folgenden Reihenfolge, der Stückliste (Anhang C) und dem Schaltplan (Anhang A) bestücken wobei zunächst nur die Bauteile angelötet werden. Diese jedoch noch nicht verdrahten. Das Verdrahten erfolgt erst wenn alle Bauteile angelötet sind. Die nach dem anlöten überstehenden Anschlüsse mit einem kleinen Seitenschneider entfernen.

Reihenfolge zur Bearbeitung und Bestückung der Stromversorgungsplatine:
- Platine auf eine Größe von 43 x 18 mm zuschneiden und die geschnittenen Seiten abschleifen

- 3,5-mm-Bohrungen für die Montage der Platine auf einer Bodenplatte (4 Bohrungen)
- Widerstand R21 (470 Ohm): **Tipp**: Vor dem Einlöten des Widerstandes diesen überprüfen, auch wenn die Bauteile in einem Regal sortiert sind. (z.B. mit einem Multimeter). Die Praxis hat gezeigt, dass sich hin und wieder doch falsche Bauteilwerte in das Regal eingeschlichen haben. Dies gilt nicht nur für Widerstände, sondern auch für Dioden, Kondensatoren, Transistoren usw.
- Diode D159 (1N4001): **Achtung:** Polarität beachten!
- Keramikkondensator C9 (100nF)
- Leuchtdiode D158 (3mm, gelb): **Achtung:** Polarität beachten! Der längere Anschluss ist die Kathode (plus), der kürzere demnach die Anode (minus).
- Spannungsregler IC4 (7805): **Achtung:** Polarität beachten!
- Elko C8 (10µF/25V): **Achtung:** Polarität beachten!
- Stiftleiste für Jumper JP2

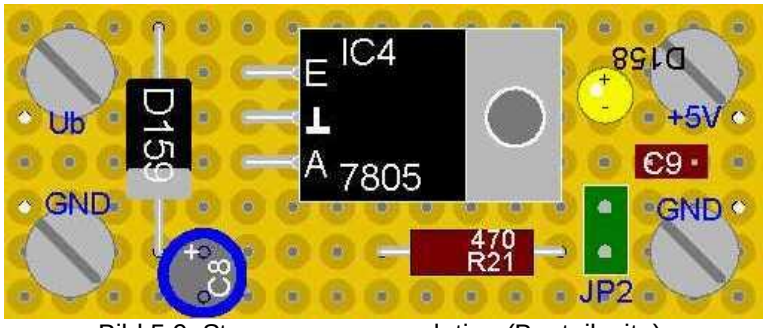


Bild 5.9: Stromversorgungsplatine (Bauteilseite)

Nachdem alle Bauteile eingelötet wurden, die Verbindungen auf der Lötseite entsprechend Bild 5.10 mit einem dünnen Draht herstellen. Dabei sollte sorgfältigst gearbeitet werden, und man sollte sich dafür ausreichend Zeit nehmen. Befinden sich Verbindungen direkt nebeneinander, so können diese auch nur mit Lötzinn verbunden werden. **Tipp:** Bild 5.10 ausdrucken, und während dem Verlöten die soeben durchgeführte Verbindung im Ausdruck kennzeichnen. Auf diese Weise können Fehler durch fehlende Verbindungen vermieden werden.
Zum Schluss alle Verbindungen und Lötstellen noch einmal sorgfältig überprüfen. (Auch hierfür sollte man sich ausreichend Zeit nehmen!)

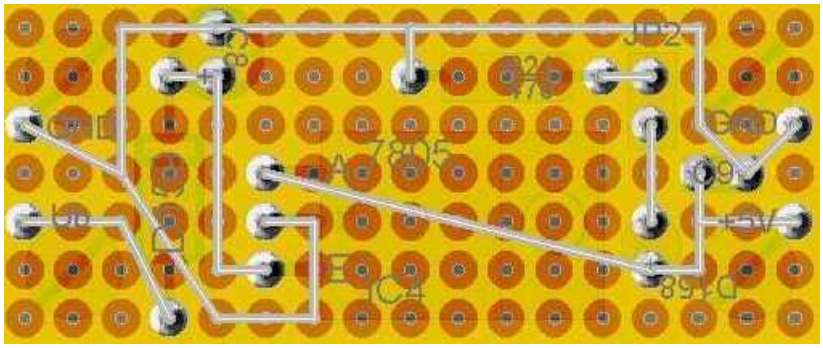


Bild 5.10: Stromversorgungsplatine (Lötseite)

Schritt 2: Montage und Verdrahtung der (Lochraster)-Platinen

Die im Schritt 1 vorbereiteten Platinen auf eine etwa. 27 x 27 cm große, etwa 4mm dicken Sperrholzplatte montieren, und entsprechend dem Schaltplan (Anhang A) mit isolierten Drähten verdrahten. Ich habe bei meinem Prototyp aus der Sperrholzplatte ein regelmäßiges 12-Eck mit einem „Durchmesser“ von 27 cm geschnitten und die 12 LED-Platinen entsprechend dieser Platte montiert (siehe auch Bild 5.11)

Schritt 3: Test

- Die ICs (IC1 bis IC3) noch **nicht** in die Fassungen einsetzen.
- Zuerst mit einem Multimeter prüfen, ob zwischen Betriebsspannung und Masse kein Kurzschluss herrscht. (Multimeter im Mode „Durchgangstester“ an den Verbindungsleitungen zwischen den einzelnen Platinen). Ist kein Kurzschluss feststellbar, als nächstes an den IC-Sockeln gemäß dem Schaltplan ebenfalls diesen Test durchführen. Eventuell festgestellte Kurzschlüsse müssen natürlich aufgespürt und entfernt werden!
- Als nächstes (mit dem Multimeter) prüfen, ob die +5V und die Masseleitung an allen Platinen vorhanden ist. (Multimeter noch immer im Mode „Durchgangstester“)
- Eine 9V-Spannungsquelle (zum Beispiel ein Steckernetzteil, diese kann auch Spannung größer als 9 Volt liefern) anschließen. **Achtung**: Auf die Polarität achten.
- Spannung an den ICs an den jeweiligen Pins messen. Wird keine oder eine grob abweichende Spannung als 5V gemessen, so liegt ein Bestückungs- und/oder Verdrahtungsfehler vor, welcher unbedingt aufgespürt und beseitigt werden muss.
- Spannung von der Wanduhr nehmen
- **Programmierten** Mikrocontroller PIC16F84 (IC1), sowie IC2 und IC3 (je 4067) im **ausgeschalteten** Zustand einsetzen. **Achtung:** Auf die Polarität achten!
- Spannung wieder anlegen, und die Jumper JP1 und JP2 stecken. Die Leuchtdiode D158 muss leuchten. Nach einer Weile muss auch die Leuchtdiode D157 zu „blinken“ beginnen, wobei die Leuchtdiode länger leuchtet, und nur sehr kurz erlischt.
- Nach einigen Minuten sollte dann die richtige Zeit angezeigt werden.
- Die beiden Jumper JP1 und JP2 werden nun nicht mehr benötigt und können entfernt werden

Geschafft! Das folgende Bild zeigt die fertige Wanduhr im Betreib.

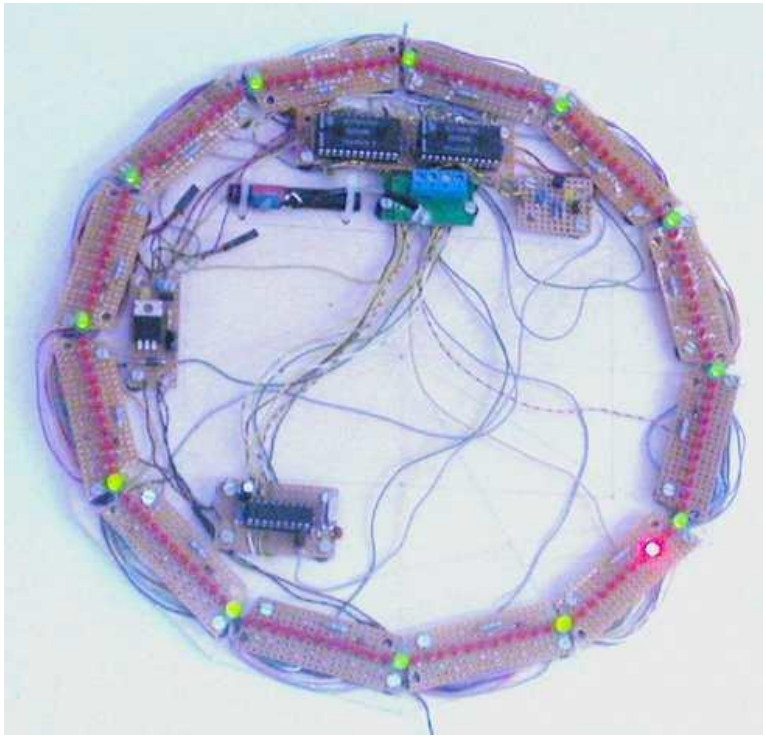
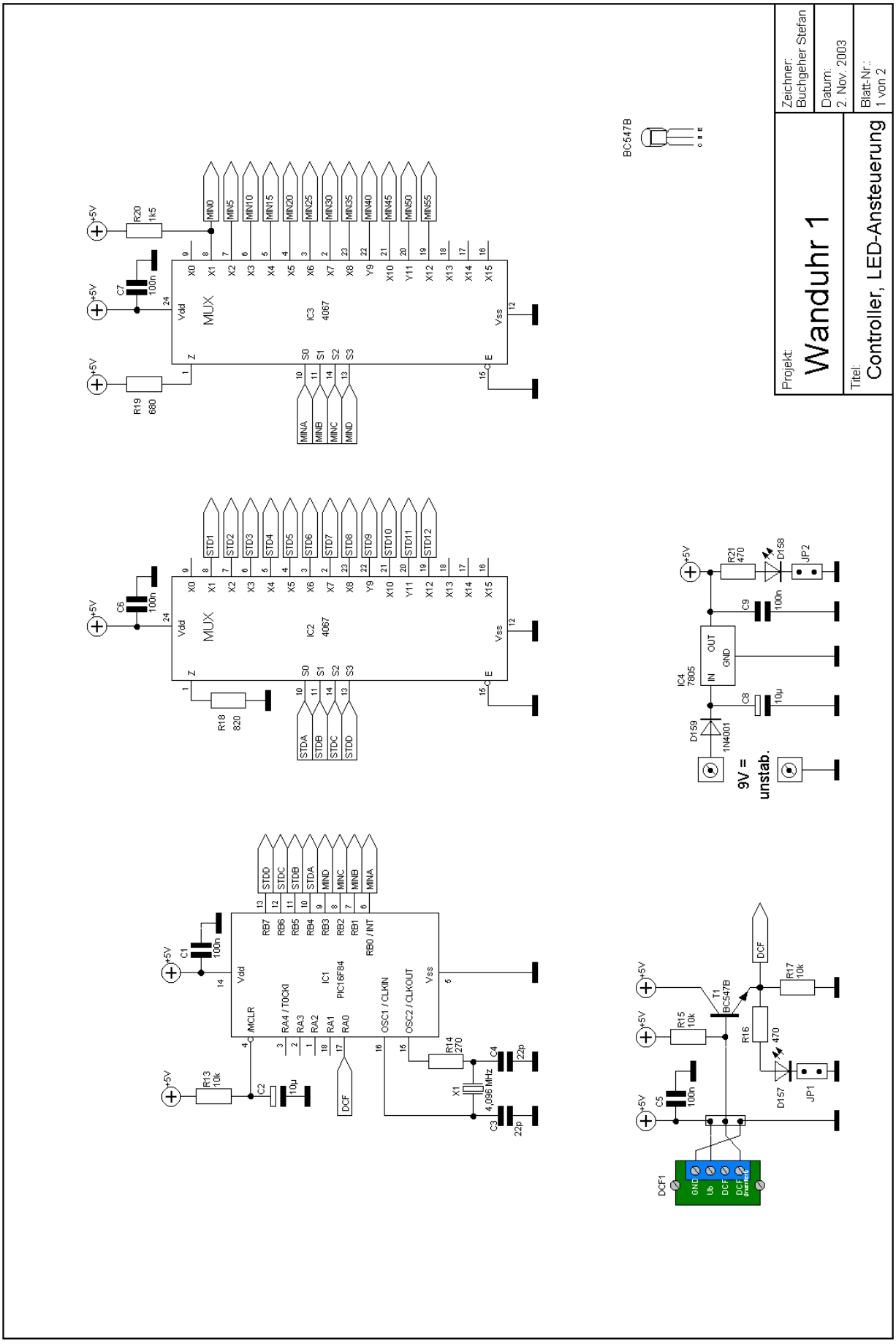
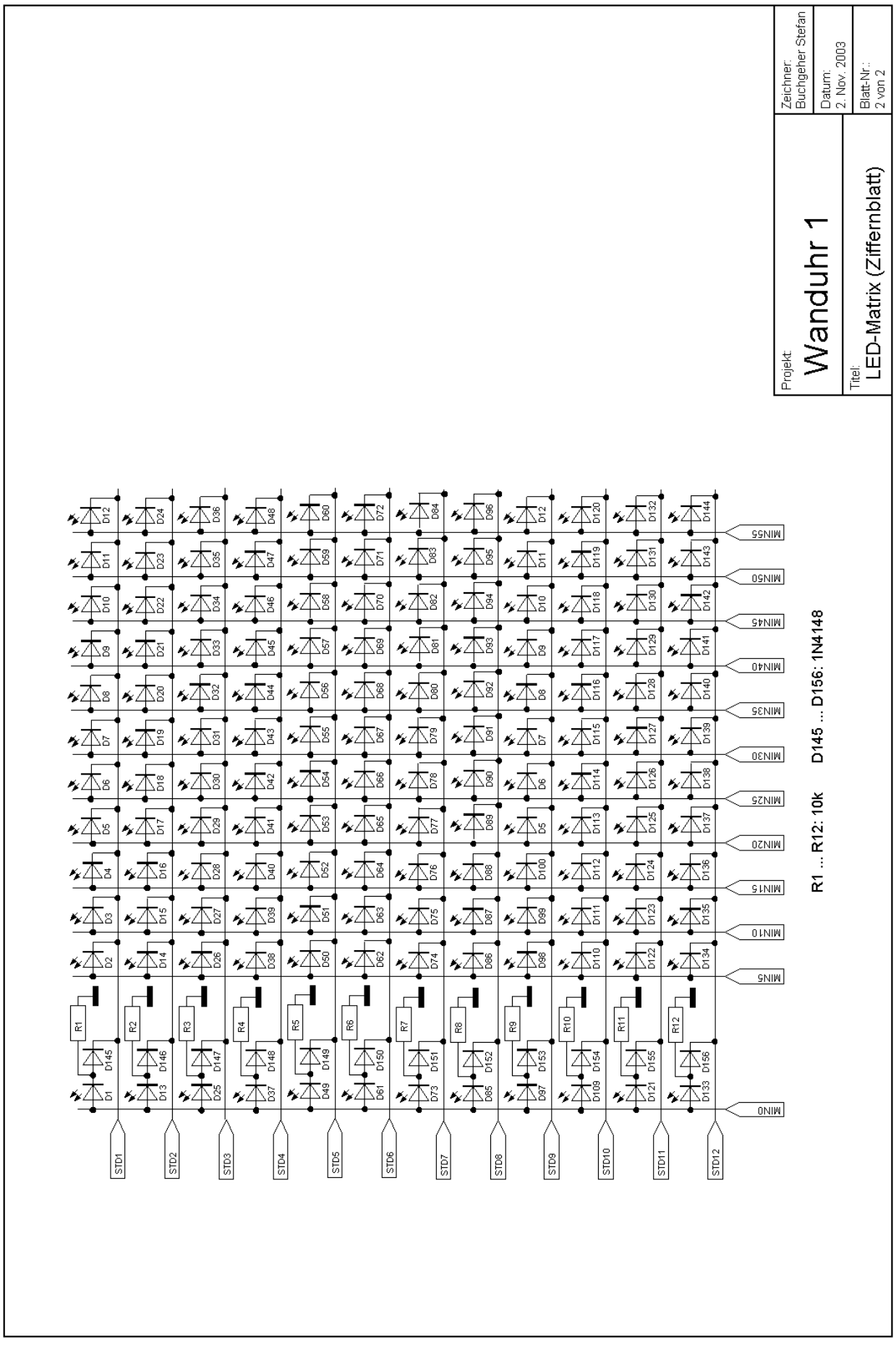


Bild 5.11: Wanduhr in Betrieb

Anhang A: Schaltpläne




Projekt:
Wanduhr 1
Zeichner:
Buchgeher Stefan
Datum:
2. Nov. 2003
Titel:
LED-Matrix (Ziffernblatt)
Blatt-Nr.:
2 von 2
R1 ... R12: 10k
D145 ... D156: 1N4148
STD1
STD2
STD3
STD4
STD5
STD6
STD7
STD8
STD9
STD10
STD11
STD12
MIN0
MIN5
MIN10
MIN15
MIN20
MIN25
MIN30
MIN35
MIN40
MIN45
MIN50
MIN55

Anhang B: Listing des PIC-Mikrocontroller (wanduhr1.asm)

```
;*************************************************************************************************
;** Analog/Digitale-Wanduhr                                                                     **
;**                                                                                             **
;** Wanduhr mit PIC16F84 und 144 LEDs in Form einer 12x12-Matrix, wobei immer nur eine LED      **
;** leuchtet. Diese LED zeigt also sowohl die Stunde und die Minute an. Dabei ergibt sich       **
;** eine etwas ungewöhnlich Darstellungsform. Hinzu kommt, dass diese Uhr "nur" eine Genauig-   **
;** keit von 5 Minuten besitzt. Mehr zur Hardware bzw. zur allgemeinen Beschreibung im          **
;** Elektor Ausgabe 12/94                                                                       **
;**                                                                                             **
;** Hardware: Prozessor: PIC16F84, Takt (Quarz 4,096 MHz)                                       **
;**           DCF-Modul                                                                         **
;**           Ansteuerelektronik fuer die LEDs                                                  **
;**                                                                                             **
;** Aufgabe(n) der Software:                                                                    **
;** Die Dekodierung des DCF-Signals erfolgt in mehreren Schritten:                              **
;**  + Abtastung des DCF-Signals alle 4 ms mit Hilfe eines Timer-Interrupt (TMR0). Die ISR      **
;**    hat die Aufgabe Low- und High-Impulse im Datenstrom zu erkennen. Diese Informationen     **
;**    werden in einem Register namens DCFSTATUS abgelegt. Dieses Register stellt das "Herz"    **
;**    der gesamten DCF-Dekodierung dar. Es enthaelt weiters zwei Hilfsbits, ein Bit welches    **
;**    gesetzt wird, wenn eine neue Sekunde begonnen hat, ebenso ein Bit wenn eine neue         **
;**    Minute festgestellt wurde.                                                               **
;**    Wurde ein Empfangsfehler oder ein Telegrammfehler festgestellt, so wird dies eben-       **
;**    falls im DCF-Statusregister (DCFSTATUS) angezeigt.                                       **
;**                                                                                             **
;**  + Das Hauptprogramm muss sich jetzt nur mehr darum kuemmern, ob eine neue Sekunde bzw.     **
;**    eine neue Minute begonnen hat, indem es staendig (zyklisch) diese beiden Botschafts-     **
;**    bits abfragt. Erkennt das Hauptprogramm, dass eine neue Sekunde begonnen hat (das        **
;**    Botschaftsbit DCFNEUESEK ist gesetzt), so fuehrt es das Unterprogramm DCFUPSEKUNDE       **
;**    aus. Die Aufgabe dieses Unterprogramms ist das hinzufuegen des neuen DCF-Impulses in     **
;**    das DCF-Telegramm. Ob ein Low- oder ein High-Impuls hinzugefuegt werden muss, diese      **
;**    Information befindet sich ja im Register DCFSTATUS (DCFLOW bzw. DCFHIGH). Da das DCF-    **
;**    Telegramm keine Information ueber die Anzahl der Sekunden bereitstellt, muss dies        **
;**    manuell im Programm erfolgen. Bevor das Unterprogramm DCFUPSEKUNDE verlassen wird,       **
;**    muss das Botschaftsflag DCFNEUESEK geloescht werden, damit es erst wieder abge-          **
;**    arbeitet wird, wenn von der ISR eine neue Sekunde detektiert wird. Ist das               **
;**    Botschaftsflag DCFNEUEMIN gesetzt so wird das Unterprogramm DCFUPMINUTE ausgefuehrt.     **
;**    Dieses Unterprogramm liest aus den Registern DCFTELEGRAMMx die Bits fuer die Minuten-    **
;**    Stunden-, und Datumsinfo aus und kopiert es in die entsprechenden Register im BCD-       **
;**    Format (DCFMINBCD, DCFSTDBCD, DCFTAGBCD...). Anschliessend die soeben gewonnen Zeit-     **
;**    und Datumsinformationen mit den Zeit- und Datumsinformationen der vorhergehenden         **
;**    Minute vergleichen. Die Zeit- und Datumswerte der soeben empfangenen Minute sind nur     **
;**    dann gueltig, wenn sie sich um maximal 1 von den Zeit- und Datumsinformationen der       **
;**    vorhergehenden Minute unterscheiden.                                                     **
;**                                                                                             **
;**    Parallel zur Dekodierung des DCF-Eingangs erzeugt das Unterprogramm INNEREUHR eine       **
;**    reine Softwareuhr. Fuer diesen Zweck ist eine genaue 1-Sekunden-Zeitbasis notwendig.     **
;**    Diese Zeitbasis wird von der schon erwaehnten Timer-0-ISR (Interrupt Service Routine)    **
;**    zusaetzlich zur 4ms-Zeitbasis erzeugt. Auch fuer diese Zeitbasis wird in der ISR ein     **
;**    entsprechendes Flag gesetzt, und das Hauptprogramm ruft das Unterprogramm INNEREUHR      **
;**    auf, wenn dieses Flag gesetzt ist. Diese zusaetzliche Softwareuhr mag auf dem ersten     **
;**    Blick unnoetig erscheinen. Es hat sich aber gezeigt, dass ein DCF-Empfang oft auch       **
;**    ueber eine laengere Zeit nicht moeglich ist. In diesem Fall wuerde die Uhr "stehen".     **
;**    Diese Zeit wird mit einer zusaetzlichen Softwareuhr so ueberbrueckt, dass der            **
;**    Betrachter der Uhr davon nichts bemerkt.                                                 **
;**                                                                                             **
;** Entwickler: Buchgeher Stefan                                                                **
;** Entwicklungsbeginn der Software: 30. Mai 2002                                               **
;** Funktionsfaehig seit: 21. August 2002                                                       **
;** Letzte Bearbeitung: 3. Maerz 2004                                                           **
;*************************************************************************************************

              List    p=PIC16F84

;******************************** Register (in Registerseite 0) ******************************
TMR0          equ     1             ;Timer0-Register
PC            equ     2             ;Programmzaehler
STAT          equ     3             ;Statusregister
PORTA         equ     5             ;PortA-Register
PORTB         equ     6             ;PortB-Register
INTCON        equ     0B            ;Interrupt-Register


;******************************** Register (in Registerseite 1) ******************************
OPTREG        equ     1             ;OPTION-Register
```

```
TRISA           equ     5               ;Richtungsregister PortA
TRISB           equ     6               ;Richtungsregister PortB


;******************************** Eigene Register (in Registerseite 0) ************************
ISR_STAT_TEMP   equ     0C              ;Zwischenspeicher des Statusregister der ISR
ISR_w_TEMP      equ     0D              ;Zwischenspeicher des Arbeitsregister der ISR
FLAGSISRHP      equ     0E              ;beinhaltet Botschaftsflags ISR -> HP
ZAEHLERISR1SEK equ      0F              ;Zaehlregister fuer 1-Sekunden-Zeitbasis
DCFSTATUS       equ     10              ;DCF-Statusregister
DCFPULS         equ     11              ;Impulszeit des DCF-Impulses
DCFPAUSE        equ     12              ;Zeit zwischen 2 DCF-Impulsen
DCFTELEGRAMM1   equ     13              ;in diese Register wird das DCF-Telegramm
DCFTELEGRAMM2   equ     14              ;  im Sekunden-Takt eingelesen. Beginnt eine
DCFTELEGRAMM3   equ     15              ;  neue Minute, so werden diese Register
DCFTELEGRAMM4   equ     16              ;  ausgewertet, und in die entsprechenden
DCFTELEGRAMM5   equ     17              ;  DCF-Zeit-Datums-Register kopiert (z.B. in das
DCFTELEGRAMM6   equ     18              ;  Register DCFMINBCD)
DCFTELEGRAMM7   equ     19              ;Reserve
DCFTELEGRAMM8   equ     1A              ;Reserve
DCFMINBCD       equ     1B              ;Beinhaltet die aktuelle dekodierte Minute (BCD-Format)
DCFSTDBCD       equ     1C              ;Beinhaltet die aktuelle dekodierte Stunde (BCD-Format)
DCFTAGBCD       equ     1D              ;Beinhaltet den aktuell dekodierten Tag (BCD-Format)
DCFMONBCD       equ     1E              ;Beinhaltet den aktuell dekodierten Monat (BCD-Format)
DCFJAHRBCD      equ     1F              ;Beinhaltet das aktuell dekodierte Jahr (BCD-Format)
DCFWOCHENTAG    equ     20              ;Beinhaltet den aktuell dekodierten Wochentag
                                        ;  (1=Montag, 2=Dienstag, usw, 7=Sonntag)
DCFZUSATZINFOS equ      21              ;Beinhaltet Zusatzinformationen (Sommer/Winterzeit,
                                        ;  Reserveantenne usw.)
DCFMINALT       equ     22              ;In den Registern DCFxxxALT befinden sich die Uhrzeit-
DCFSTDALT       equ     23              ;  und Datumsinfos der vorangegangenen Minute. Diese
DCFTAGALT       equ     24              ;  werden zur Ueberpruefung des neuen DCF-Telegramms
DCFMONALT       equ     25              ;  benoetigt
DCFJAHRALT      equ     26
DCFWTAGALT      equ     27
UHRSEKUNDE      equ     28              ;Sekundenzaehler der inneren quarzgesteuerten Uhr
UHRMINUTE       equ     29              ;Minutenzaehler der inneren quarzgesteuerten Uhr, wird
                                        ;  mit jedem gueltigen DCF-dekod. Wert synchronisiert
UHRSTUNDE       equ     2A              ;Stundenzaehler der inneren quarzgesteuerten Uhr, wird
                                        ;  mit jedem gueltigen DCF-dekod. Wert synchronisiert
DATUMWTAG       equ     2B              ;Wochentagszaehler der inneren quarzgest. Uhr, wird
                                        ;  mit jedem gueltigen DCF-dekod. Wert synchronisiert
DATUMTAG        equ     2C              ;Tageszaehler der inneren quarzgesteuerten Uhr, wird
                                        ;  mit jedem gueltigen DCF-dekod. Wert synchronisiert
DATUMMONAT      equ     2D              ;Monatszaehler der inneren quarzgesteuerten Uhr, wird
                                        ;  mit jedem gueltigen DCF-dekod. Wert synchronisiert
DATUMJAHR       equ     2E              ;Jahreszaehler der inneren quarzgesteuerten Uhr, wird
                                        ;  mit jedem gueltigen DCF-dekod. Wert synchronisiert
TEMP1           equ     4A              ;allgemeines Hilfsregister 1
TEMP2           equ     4B              ;allgemeines Hilfsregister 2


;******************************** Bits in Registern der Registerbank 0 ************************
;Register STAT
C               equ     0               ;Carrybit im Statuswort-Register
Z               equ     2               ;Zerobit im Statuswort-Register
RP0             equ     5               ;Seitenauswahlbit im Statuswort-Register

;Register INTCON
T0IF            equ     2               ;TMR0-Interruptflag im INTCON-Register
GIE             equ     7               ;Interruptfreigabeflag


;******************************** Bits in den eigenen Registern der Registerbank 0 ************
;Register FLAGSISRHP
FLAG4MSEK       equ     0
FLAG1SEK        equ     1

;Register DCFSTATUS
DCFPORTNEU      equ     0               ;Akt. Zustand DCF-Porteingang
DCFPORTALT      equ     1               ;Vorhergehender Zustand am DCF-Porteingang
DCFNEUESEK      equ     2               ;gesetzt, wenn neue Sekunde begonnen
DCFNEUEMIN      equ     3               ;gesetzt, wenn neue Minute begonnen
DCFLOW          equ     4               ;gesetzt, wenn Low-Signal erkannt
DCFHIGH         equ     5               ;gesetzt, wenn High-Signal erkannt
DCFFEHLER       equ     6               ;gesetzt, wenn ein Fehler erkannt
DCFSYNC         equ     7               ;gesetzt, wenn Uhr mit DCF synchronisiert

;Register DCFZUSATZINFOS
```

```
DCFRESANT       equ     0               ;Reserveantenne
DCFSOMWIN       equ     1               ;Vorankuendigung der Sommer/Winterzeit-Umstellung
DCFSOMMER       equ     2               ;Sommerzeit
DCFWINTER       equ     3               ;Winterzeit
DCFSCHALTSEK    equ     4               ;Vorankuendigung einer zusaetzliche Schaltsekunde


;******************************** Portbelegung ************************************************
;Port A
DCFINPORT       equ     PORTA
DCFINTRIS       equ     TRISA
DCFIN           equ     0


;******************************** Konstanten **************************************************
KONSTISR1SEK    equ     .250
DCFLOWMIN       equ     .20
DCFLOWMAX       equ     .30
DCFHIGHMIN      equ     .40
DCFHIGHMAX      equ     .60


;******************************** Ziele der Registeroperationen *******************************
w               equ     0               ;w ist Zielregister
f               equ     1               ;f ist Zielregister


;******************************** Makros ******************************************************
;Umschalten zu Registerbank 0
bank0           MACRO
                bcf     STAT,RP0
                ENDM

;Umschalten zu Registerbank 1
bank1           MACRO
                bsf     STAT,RP0
                ENDM


;******************************** Konfigurations-Bits *****************************************
_lp_osc         equ     h'3FFC'
_xt_osc         equ     h'3FFD'
_hs_osc         equ     h'3FFE'
_rc_osc         equ     h'3FFF'

_wdt_off        equ     h'3FFB'
_wdt_on         equ     h'3FFF'

_pwrt_off       equ     h'3FFF'
_pwrt_on        equ     h'3FF7'

_cp_off         equ     h'3FFF'
_cp_on          equ     h'000F'

        __config        _hs_osc & _wdt_off & _pwrt_off & _cp_off

                ORG     0x000
                goto    Beginn
                ORG     0x004
                goto    ISR


;******************************** Tabellen ****************************************************

;**********************************************************************************************
;** Tabelle TABDIV5:                                                                         **
;** Division des w-Register durch 5                                                          **
;**********************************************************************************************
TABDIV5         addwf   PC,f
                dt      .0,.0,.0,.0,.0
                dt      .1,.1,.1,.1,.1
                dt      .2,.2,.2,.2,.2
                dt      .3,.3,.3,.3,.3
                dt      .4,.4,.4,.4,.4
                dt      .5,.5,.5,.5,.5
                dt      .6,.6,.6,.6,.6
                dt      .7,.7,.7,.7,.7
                dt      .8,.8,.8,.8,.8
                dt      .9,.9,.9,.9,.9
```

```
                dt      .10,.10,.10,.10,.10
                dt      .11,.11,.11,.11,.11
                dt      .0,.0,.0,.0


;*************************************************************************************************
;** Tabelle TABSTUNDEN:                                                                          **
;** Ausgabetabelle der Stunde                                                                    **
;*************************************************************************************************
TABSTUNDEN      addwf   PC,f
                dt      .12
                dt      .1,.2,.3,.4,.5,.6,.7,.8,.9,.10,.11,.12
                dt      .1,.2,.3,.4,.5,.6,.7,.8,.9,.10,.11
                dt      .0,.0,.0,.0,.0,.0,.0,.0,.0


;****************************** ISR - Timer0 ***************************************************

;*************************************************************************************************
;** Interrupt Service Routine:                                                                   **
;**                                                                                             **
;** Aufruf:                                                                                      **
;**     alle 4 ms                                                                               **
;**                                                                                             **
;** Aufgaben:                                                                                    **
;**     + w-Register (=Arbeitsregister) und Status-Register zwischenspeichern (PUSH)            **
;**     + Zeitbasis fuer 4ms und 1 Sekunde erzeugen                                             **
;**     + Das Timer-Interrupt-Flag T0IF wieder loeschen                                         **
;**     + w-Register (=Arbeitsregister) und Statusregister wiederherstellen (POP).              **
;*************************************************************************************************
ISR
PUSH            movwf   ISR_w_TEMP              ;w-Register retten
                swapf   STAT,w                  ;Statusregister
                movwf   ISR_STAT_TEMP           ;  retten

                ;Beginn der eigentlichen ISR-Routine
                bsf     FLAGSISRHP,FLAG4MSEK    ;Botschaftsflag setzen

                decfsz  ZAEHLERISR1SEK,f        ;Zaehlregister fuer 1-Sekunden-Zeitbasis um 1
                                                ;  vermindern

                goto    ISRWEITER1
                bsf     FLAGSISRHP,FLAG1SEK     ;Botschaftsflag setzen
                movlw   KONSTISR1SEK            ;Zaehlregister fuer den Sekundentakt mit
                movwf   ZAEHLERISR1SEK          ;  der Konstanten KONSTISR1SEK laden

ISRWEITER1
                ;Ende der eigentlichen ISR-Routine
ISRFERTIG       bcf     INTCON,T0IF             ;T0-Interruptflag loeschen

POP             swapf   ISR_STAT_TEMP,w         ;Status-Register
                movwf   STAT                    ;  und
                swapf   ISR_w_TEMP,f            ;  w-Register
                swapf   ISR_w_TEMP,w            ;  wieder herstellen

                retfie


;****************************** Unterprogramme *************************************************

;*************************************************************************************************
;** Initialisierung des Prozessor:                                                               **
;**     + TMR0-ISR soll alle 4 ms aufgerufen werden, dazu Vorteiler mit 16 laden (Bei einem     **
;**       4,096-MHz-Quarz)                                                                      **
;**     + Ports: Port A: Ausgaenge (unbenutzt)                                                  **
;**              Port B: Ausgaenge                                                              **
;**              (Der DCF-Eingang wird separat mit dem Befehl bsf  DCFINTRIS,DCFIN als           **
;**              Eingang definiert)                                                             **
;**     + Zaehlregister fuer den Sekundentakt vorbelegen                                        **
;**     + DCF-Statusregister (DCFSTATUS) initialisieren (Fehlerflag setzen, alle anderen        **
;**       Flags loeschen)                                                                       **
;**     + Pegel von DCFIN einlesen und im Flag DCFPORTNEU des Register DCFSTATUS sichern         **
;**     + Anzeige loeschen                                                                      **
;**     + diverse Register vorbelegen (initialisieren)                                          **
;*************************************************************************************************
INIT            clrf    TMR0                    ;Timer0 auf 0 voreinstellen

                bank1                           ;Registerseite 1
                movlw   b'00000011'             ;interner Takt, Vorteiler = 16 an TMR0
```

```
                movwf   OPTREG
                clrf    TRISA                   ;Port A: Ausgaenge (unbenutzt)

                clrf    TRISB                   ;Port B: Ausgaenge

                bsf     DCFINTRIS,DCFIN         ;DCFIN als Eingang definieren
                bank0                           ;Registerseite 0

                movlw   KONSTISR1SEK            ;Zaehlregister fuer den Sekundentakt mit
                movwf   ZAEHLERISR1SEK          ;  der Konstanten KONSTISR1SEK laden

                clrf    DCFSTATUS               ;DCF-Statusregister (DCFSTATUS) initialisieren
                bsf     DCFSTATUS,DCFFEHLER     ;  Fehlerflag setzen, alle anderen Flags
                                                ;  loeschen

                btfsc   DCFINPORT,DCFIN         ;Pegel von DCFIN einlesen und im Flag
                bsf     DCFSTATUS,DCFPORTNEU    ;  DCFPORTNEU des Register DCFSTATUS sichern

                clrf    PORTB                   ;Anzeige (Ziffernblatt loeschen)

                clrf    UHRSEKUNDE              ;Sekundenzaehler initialisieren (mit 0)
                clrf    UHRMINUTE               ;Minutenzaehler initialisieren (mit 0)
                clrf    UHRSTUNDE               ;Stundenzaehler initialisieren (mit 0)
                clrf    DATUMWTAG               ;Wochentagzaehler initialisieren (mit 0)
                clrf    DATUMTAG                ;Tageszaehler initialisieren (mit 0)
                clrf    DATUMMONAT              ;Monatszaehler initialisieren (mit 0)
                clrf    DATUMJAHR               ;Jahreszaehler initialisieren (mit 0)

                movlw   .99
                movwf   DCFMINALT               ;Vergleichsregister initialisieren (mit 99)
                movwf   DCFSTDALT               ;Vergleichsregister initialisieren (mit 99)
                movwf   DCFTAGALT               ;Vergleichsregister initialisieren (mit 99)
                movwf   DCFMONALT               ;Vergleichsregister initialisieren (mit 99)
                movwf   DCFJAHRALT              ;Vergleichsregister initialisieren (mit 99)
                movwf   DCFWTAGALT              ;Vergleichsregister initialisieren (mit 99)
                return


;******************************** DCF Routinen ***********************************************

;*****************************************************************************************
;** DCFROUTINE:                                                                         **
;**                                                                                     **
;** Das Unterprogramm DCFROUTINE ist für die DCF-Dekodierung die wichtigste Komponente. **
;**                                                                                     **
;** Diese Routine wird ca. alle 4 ms aufgerufen                                         **
;**                                                                                     **
;** Aufgabe und Vorgehensweise:                                                         **
;**    Die Aufgabe des Unterprogramms DCFROUTINE besteht darin aus den Abtastungen heraus-  **
;**    zufinden, ob ein Low, ein High oder ein Minutenwechsel gesendet wurde. Dazu wird bei  **
;**    jedem Aufruf des Unterprogramms DCFROUTINE entweder das Zaehlregister DCFPULS oder    **
;**    das Zaehlregister DCFPAUSE um 1 erhoeht (inkrementiert), wenn sich der Pegel vom DCF-  **
;**    Eingang zum vorhergehenden Aufruf dieses Unterprogramms nicht veraendert hat. Ist     **
;**    der DCF-Eingang Low (also logisch 0), so wird das Zaehlregister DCFPULS um 1 erhoeht,  **
;**    ist der DCF-Eingang High (also logisch 1), so wird das Zaehlregister DCFPAUSE um 1    **
;**    erhoeht. Beide Zaehlregister koennen maximal den Wert 255 aufnehmen, was einer Zeit   **
;**    von 1020 ms (oder 1,02 Sekunden) entspricht (=255 * 4ms).                             **
;**    Unterscheidet sich der aktuelle Pegel des DCF-Eingangs mit dem Pegel vom vorher-      **
;**    gehenden Aufruf, so spricht man von einer Flanke, wobei hier zwischen einer fallenden **
;**    und einer steigenden Flanke unterschieden werden muss. Bei einer fallenden Flanke     **
;**    (der aktuelle Pegel des DCF-Eingangs ist logisch 0), erfolgt zunaechst eine Aus-      **
;**    wertung des Zaehlregisters DCFPULS. Beinhaltet dieses Zaehlregister einen Wert        **
;**    zwischen den Konstanten DCFLOWMIN und DCFLOWMAX, so wurde ein LOW des DCF-Telegramms  **
;**    ermittelt. Im DCF-Statusregister (DCFSTATUS) werden daher die Flags DCFLOW und        **
;**    DCFNEUESEK gesetzt. Das Flag DCFNEUESEK dient als Zeichen einer neu empfangenen und   **
;**    gueltigen Sekunde. Anschliessend wird das Zaehlregister DCFPULS geloescht. Beinhaltet **
;**    das Zaehlregister DCFPULS einen Wert zwischen den Konstanten DCFHIGHMIN und           **
;**    DCFHIGHMAX, so wurde ein HIGH des DCF-Telegramms ermittelt. Im DCF-Statusregister     **
;**    (DCFSTATUS) werden daher die Flags DCFHIGH und DCFNEUESEK gesetzt. Das Flag           **
;**    DCFNEUESEK dient auch hier als Zeichen einer neu empfangenen und gueltigen Sekunde,   **
;**    und auch das Zaehlregister DCFPULS wird anschliessend geloescht. Beinhaltet das       **
;**    Zaehlregister DCFPULS einen Wert der weder zwischen den Konstanten DCFLOWMIN und      **
;**    DCFLOWMAX noch zwischen den Konstanten DCFHIGHMIN und DCFHIGHMAX liegt, so handelt es **
;**    sich um einen Uebertragungsfehler. In diesem Fall wird das Fehlerflag (DCFFEHLER) im  **
;**    DCF-Statusregister (DCFSTATUS) gesetzt und auch das Zaehlregister DCFPULS wird an-    **
;**    schließend geloescht. Das Flag DCFNEUESEK wird hier aber nicht gesetzt, da es sich in **
;**    diesem Fall um keine gueltige Sekunde handelt. Das Fehlerflag wird erst bei der       **
;**    Erkennung einer neuen Minute wieder geloescht. Achtung: Solange das Fehlerflag ge-    **
;**    setzt ist erfolgt keine Auswertung des Zaehlregisters DCFPULS. Dieses Zaehlregister   **
```

```
;**     wird aber dennoch bei jeder fallenden Flanke zurueckgesetzt.                          **
;**     Bei einer steigenden Flanke (der aktuelle Pegel des DCF-Eingangs ist logisch 1) wird  **
;**     das Zaehlregister DCFPAUSE geloescht.                                                 **
;**     In der 59. Sekunde erfolgt keine Absenkung des Traegers. Da es also in der 59.        **
;**     Sekunde keine fallende Flanke gibt, gibt es auch keine steigende Flanke. Das Zaehl-   **
;**     register DCFPAUSE laeuft ueber, da es nur einen Zaehlbereich von 0 bis 255 besitzt.   **
;**     Dieser Ueberlauf wird hier ausgenuetzt. Zur Bestimmung einer neuen Minute bzw. zur    **
;**     Bestimmung des Telegrammbeginns. Ist das Fehlerflag gesetzt, so wird es nun ge-       **
;**     loescht. War es nicht gesetzt, so wird das Flag DCFNEUEMIN im DCF-Statusregister      **
;**     (DCFSTATUS) gesetzt.                                                                  **
;**                                                                                           **
;** Anmerkung:                                                                                **
;**     Die temporaeren Register TEMP1 und TEMP2 werden hier nur als Hilfsregister benoetigt. **
;**     Sie koennen daher auch in anderen Unterprogrammen verwendet werden.                   **
;*******************************************************************************************
;
DCFROUTINE      bcf     DCFSTATUS,DCFPORTALT  ;DCFSTATUS,DCFPORTNEU -> DCFSTATUS,DCFPORTALT
                btfsc   DCFSTATUS,DCFPORTNEU
                bsf     DCFSTATUS,DCFPORTALT

                movf    DCFINPORT,w
                movwf   TEMP1
                bcf     DCFSTATUS,DCFPORTNEU  ;DCFIN -> DCFSTATUS,DCFPORTNEU, TEMP1,DCFPORTALT
                btfsc   TEMP1,DCFIN
                bsf     DCFSTATUS,DCFPORTNEU
                clrf    TEMP2
                btfsc   TEMP1,DCFIN
                bsf     TEMP2,DCFPORTALT

                movf    DCFSTATUS,w
                xorwf   TEMP2,f

                btfss   TEMP2,DCFPORTALT      ;Hat sich am Pin DCFIN der Pegel geaendert?
                goto    DCFINCPULSPAUSE       ;nein: Puls- bzw. Pause-Zaehler um 1 erhoehen
                btfss   DCFSTATUS,DCFPORTNEU  ;ja: -> steigende oder fallende Flanke?
                goto    DCFHLFLANKE           ;fallende Flanke
DCFLHFLANKE     clrf    DCFPAUSE              ;steigende Flanke: Zaehlregister DCFPAUSE
                goto    DCFROUTINEFERTIG      ;  loeschen und UP verlassen

                ;Fallende Flanke
DCFHLFLANKE     btfss   DCFSTATUS,DCFFEHLER   ;Fehlerflag gesetzt:
                goto    DCFPLAUSIBEL          ;nein: Ermittlung, ob die ermittelte Pulszeit
                                              ;  einem HIGH oder einem LOW entspricht
                                              ;  (Plausibilitaetspruefung)
                goto    DCFPULSLOESCHEN       ;ja: Puls-Zaehler loeschen
DCFPLAUSIBEL
                ;Pruefen, ob ein High empfangen wurde
DCFCHECKHIGH    movlw   DCFHIGHMIN            ;unteren High-Grenzwert laden
                subwf   DCFPULS,w
                btfss   STAT,C                ;Grenzwert < DCFPULS ?
                goto    DCFCHECKLOW           ;nein: kein HIGH-Pegel, pruefen, ob LOW
                movlw   DCFHIGHMAX            ;ja: DCFPULS mit oberen HIGH-Grenzwert
                                              ;  vergleichen
                subwf   DCFPULS,w
                btfsc   STAT,C                ;DCFPULS < Grenzwert ?
                goto    DCFCHECKLOW           ;nein: es handelt sich um kein HIGH-Signal
                                              ;  pruefen, ob ein LOW empfangen wurde
                bcf     DCFSTATUS,DCFLOW      ;ja: Es wurde ein High-Signal empfangen,
                                              ;  das Flag DCFLOW loeschen,
                bsf     DCFSTATUS,DCFHIGH     ;  das Flag DCFHIGH setzen,
                bsf     DCFSTATUS,DCFNEUESEK  ;  das Flag DCFNEUESEK setzen
                goto    DCFPULSLOESCHEN       ;  und Puls-Zaehler loeschen

                ;Pruefen, ob ein Low empfangen wurde
DCFCHECKLOW     movlw   DCFLOWMIN             ;unteren Low-Grenzwert laden
                subwf   DCFPULS,w
                btfss   STAT,C                ;Grenzwert < DCFPULS ?
                goto    DCFSETFEHLER          ;nein: kein LOW-Pegel, Empfangsfehler
                movlw   DCFLOWMAX             ;ja: DCFPULS mit oberen LOW-Grenzwert
                                              ;  vergleichen
                subwf   DCFPULS,w
                btfsc   STAT,C                ;DCFPULS < Grenzwert ?
                goto    DCFSETFEHLER          ;nein: es handelt sich um kein LOW-Signal
                                              ;  -> Empfangsfehler
                bcf     DCFSTATUS,DCFHIGH     ;ja: Es wurde ein LOW-Signal empfangen,
                                              ;  das Flag DCFLOW setzen,
                bsf     DCFSTATUS,DCFLOW      ;  das Flag DCFHIGH loeschen,
                bsf     DCFSTATUS,DCFNEUESEK  ;  das Flag DCFNEUESEK setzen
                goto    DCFPULSLOESCHEN       ;  und den Puls-Zaehler loeschen
```

```
               ;Empfangsfehler
DCFSETFEHLER   bsf     DCFSTATUS,DCFFEHLER   ;Bei einem Empfangsfehler
               bcf     DCFSTATUS,DCFHIGH     ;  Fehlerflag (DCFFEHLER) setzen,
               bcf     DCFSTATUS,DCFLOW      ;  die Flags DCFLOW und DCFHIGH loeschen,
               goto    DCFPULSLOESCHEN       ;  und den Puls-Zaehler loeschen

               ;Puls- oder Pausen-Zaehler erhoehen
DCFINCPULSPAUSE btfsc  DCFSTATUS,DCFPORTNEU
               goto    DCFINCPAUSE

DCFINCPULS     incf    DCFPULS,f             ;Puls-Zaehler um 1 erhoehen
               goto    DCFROUTINEFERTIG

DCFINCPAUSE    incfsz  DCFPAUSE,f            ;Pause-Zaehler erhoehen und auf Ueberlauf
                                             ;  pruefen
               goto    DCFROUTINEFERTIG      ;kein Ueberlauf: Unterprogramm verlassen

               btfsc   DCFSTATUS,DCFFEHLER   ;Ueberlauf: NEUE MINUTE: wenn Fehlerflag
               goto    DCFROUTINEW1

               bsf     DCFSTATUS,DCFNEUEMIN  ;nicht gesetzt, Flag DCFNEUEMIN setzen
               goto    DCFROUTINEFERTIG      ;  und Unterprogramm verlassen

DCFROUTINEW1   bcf     DCFSTATUS,DCFFEHLER   ;Fehlerflag gesetzt: Fehlerflag loeschen
               goto    DCFROUTINEFERTIG      ;  und Unterprogramm verlassen (Flag DCFNEUEMIN
                                             ;  wird in diesem Fall nicht gesetzt)

               ;Puls-Zaehler loeschen
DCFPULSLOESCHEN        clrf    DCFPULS

DCFROUTINEFERTIG
               return


;*************************************************************************************************
;** Neue Sekunde:                                                                               **
;** Aufgaben:                                                                                   **
;**     + je nachdem ob das Bit DCFLOW oder das Bit DCFHIGH gesetzt ist dieses Bit dem         **
;**       Telegramm (Register DCFTELEGRAMM1 bis DCFTELEGRAMM8) hinzufuegen                     **
;**     + Anforderungsflag (Flag DCFNEUESEK) zuruecksetzen                                     **
;**                                                                                           **
;** Vorgehensweise:                                                                             **
;**     Ist das Flag DCFLOW (im Register DCFSTATUS) gesetzt das Carryflag (im SFR STAT)        **
;**     loeschen, ist aber das Flag DCFHIGH (ebenfalls im Register DCFSTATUS) gesetzt das      **
;**     Carryflag setzten. Dieses Carryflag nun dem Telegramm hinzufuegen. Dieser Vorgang     **
;**     erfolgt mit einem so genannten Schiebebefehl. Dabei werden alle Bits des Registers    **
;**     DCFTELEGRAMM1 auf die naechst hoehere Position verschoben. (Bit 6 wandert ins Bit 7,  **
;**     Bit 5 wandert ins Bit 6 usw. Bit 0 wandert ins Bit 1). Der Inhalt vom Carryflag       **
;**     wandert ins Bit 0. Der Inhalt von Bit 7 wird ins Carryflag geschoben. Bei den         **
;**     Register DCFTELEGRAMM2 bis DCFTELEGRAMM6 erfolgt derselbe Vorgang. (Der Inhalt von    **
;**     Bit 7 des Registers DCFTELEGRAMM1 wird ins Carry geschoben, und das Carry aber weiter **
;**     in das Bit 0 des Registers DCFTELEGRAMM2)                                             **
;**                                                                                           **
;** Anmerkung:                                                                                  **
;**     Die Flags DCFLOW und DCFHIGH koennen nie gleichzeitig gesetzt sein. Es ist nur        **
;**     moeglich, dass entweder nur DCFLOW oder nur DCFHIGH gesetzt ist, aber nie beide       **
;**     gleichzeitig.                                                                         **
;*************************************************************************************************
DCFUPSEKUNDE   btfsc   DCFSTATUS,DCFLOW      ;Ist das Flag DCFLOW im Register DCFSTATUS
                                             ;  gesetzt?
               goto    DCFSCHIEBELOW
               btfsc   DCFSTATUS,DCFHIGH     ;nein: Wenn das FLAG DCFHIGH im Register
DCFSCHIEBEHIGH bsf     STAT,C                ;  DCFSTATUS gesetzt ist, Dem Telegramm ein
               goto    DCFSCHIEBE            ;  Low mit Hilfe des Carry hinzufuegen
DCFSCHIEBELOW  bcf     STAT,C                ;ja: Dem Telegramm ein High mit Hilfe des Carry
                                             ;  hinzufuegen
DCFSCHIEBE     rlf     DCFTELEGRAMM1,f
               rlf     DCFTELEGRAMM2,f
               rlf     DCFTELEGRAMM3,f
               rlf     DCFTELEGRAMM4,f
               rlf     DCFTELEGRAMM5,f
               rlf     DCFTELEGRAMM6,f
;              rlf     DCFTELEGRAMM7,f
;              rlf     DCFTELEGRAMM8,f

               bcf     DCFSTATUS,DCFNEUESEK  ;Anforderungsflag loeschen
               return
```

```
;*************************************************************************************************
;** Neue Minute:                                                                                 **
;** Aufgaben:                                                                                    **
;**     + Fehlende 59. Sekunde "nachholen" (Unterprogramm DCFUPSEKUNDE aufrufen)                 **
;**     + Datum, Uhrzeit und die Zusatzinformationen aus den Registern DCFTELEGRAMM1 bis         **
;**       DCFTELEGRAMM8 zusammensetzen                                                           **
;**     + Das soeben ermittelte Telegramm mit dem Telegramm der vorhergehenden Minute ver-       **
;**       gleichen. Die soeben ermittelte Minute muss dabei um 1 groesser als die vorher-        **
;**       gehende Minute sein, und die soeben ermittelten Werte fuer die Stunde, Tag, Monat,     **
;**       Jahr und Wochentag muss mit den Werten des vorhergehenden Telegramms ueberein-         **
;**       stimmen. ACHTUNG: Bei dieser einfachen Ueberpruefung wird der Uebergang von z. B.      **
;**       13:59 auf 14:00 als falsch gewertet, da sich hier die Stunde aendert, und auch die     **
;**       Minute um mehr als 1. Wuerden hier alle moeglichen Uebergaenge beruecksichtigt,        **
;**       dann wuerde dieses Unterprogramm noch umfangreicher als es ohnehin schon ist. Durch    **
;**       diese Vereinfachung dauert die Synchronisierung im schlimmsten Fall nur eine Minute    **
;**       laenger.                                                                               **
;**     + Das alte Datum bzw. die alte Uhrzeit (von der vorhergehenden Minute) durch das neue    **
;**       Datum bzw. Uhrzeit ersetzen. Dies ist fuer die Ueberpruefung des naechsten Tele-       **
;**       gramms notwendig.                                                                      **
;**     + Wenn das neu empfangene Datum bzw. die neu empfangene Uhrzeit gueltig ist, die BCD-    **
;**       kodierten Datums- bzw. Uhrzeitkomponenten in eine binaere Form umwandeln und damit     **
;**       die mitlaufende Uhr synchronisieren. Das Flag DCFSYNC im Register DCFSTATUS setzen.    **
;**       Dieses gesetzte Flag kennzeichnet, dass die mitlaufende Uhr mit der DCF-Uhr            **
;**       synchronisiert ist                                                                     **
;**     + Anforderungsflag (Flag DCFNEUEMIN im Register DCFSTATUS) zuruecksetzen                 **
;**                                                                                            **
;** Anmerkung:                                                                                   **
;**     Das temporaere Register TEMP1 wird hier nur als Hilfsregister benoetigt, und kann      **
;**     daher auch in anderen Unterprogrammen verwendet werden.                                **
;*************************************************************************************************
DCFUPMINUTE     call    DCFUPSEKUNDE            ;Fehlende Sekunde nachholen

                clrf    DCFJAHRBCD              ;Jahr zusammensetzen (BCD-Format)
                btfsc   DCFTELEGRAMM1,1
                bsf     DCFJAHRBCD,7
                btfsc   DCFTELEGRAMM1,2
                bsf     DCFJAHRBCD,6
                btfsc   DCFTELEGRAMM1,3
                bsf     DCFJAHRBCD,5
                btfsc   DCFTELEGRAMM1,4
                bsf     DCFJAHRBCD,4
                btfsc   DCFTELEGRAMM1,5
                bsf     DCFJAHRBCD,3
                btfsc   DCFTELEGRAMM1,6
                bsf     DCFJAHRBCD,2
                btfsc   DCFTELEGRAMM1,7
                bsf     DCFJAHRBCD,1
                btfsc   DCFTELEGRAMM2,0
                bsf     DCFJAHRBCD,0

                clrf    DCFMONBCD               ;Monat zusammensetzen (BCD-Format)
                btfsc   DCFTELEGRAMM2,1
                bsf     DCFMONBCD,4
                btfsc   DCFTELEGRAMM2,2
                bsf     DCFMONBCD,3
                btfsc   DCFTELEGRAMM2,3
                bsf     DCFMONBCD,2
                btfsc   DCFTELEGRAMM2,4
                bsf     DCFMONBCD,1
                btfsc   DCFTELEGRAMM2,5
                bsf     DCFMONBCD,0

                clrf    DCFWOCHENTAG            ;Wochentag zusammensetzen (BCD-Format)
                btfsc   DCFTELEGRAMM2,6
                bsf     DCFWOCHENTAG,2
                btfsc   DCFTELEGRAMM2,7
                bsf     DCFWOCHENTAG,1
                btfsc   DCFTELEGRAMM3,0
                bsf     DCFWOCHENTAG,0

                clrf    DCFTAGBCD               ;Tag zusammensetzen (BCD-Format)
                btfsc   DCFTELEGRAMM3,1
                bsf     DCFTAGBCD,5
                btfsc   DCFTELEGRAMM3,2
                bsf     DCFTAGBCD,4
                btfsc   DCFTELEGRAMM3,3
                bsf     DCFTAGBCD,3
                btfsc   DCFTELEGRAMM3,4
                bsf     DCFTAGBCD,2
```

```
                btfsc   DCFTELEGRAMM3,5
                bsf     DCFTAGBCD,1
                btfsc   DCFTELEGRAMM3,6
                bsf     DCFTAGBCD,0

                clrf    DCFSTDBCD               ;Stunde zusammensetzen (BCD-Format)
                btfsc   DCFTELEGRAMM4,0
                bsf     DCFSTDBCD,5
                btfsc   DCFTELEGRAMM4,1
                bsf     DCFSTDBCD,4
                btfsc   DCFTELEGRAMM4,2
                bsf     DCFSTDBCD,3
                btfsc   DCFTELEGRAMM4,3
                bsf     DCFSTDBCD,2
                btfsc   DCFTELEGRAMM4,4
                bsf     DCFSTDBCD,1
                btfsc   DCFTELEGRAMM4,5
                bsf     DCFSTDBCD,0

                clrf    DCFMINBCD               ;Minute zusammensetzen (BCD-Format)
                btfsc   DCFTELEGRAMM4,7
                bsf     DCFMINBCD,6
                btfsc   DCFTELEGRAMM5,0
                bsf     DCFMINBCD,5
                btfsc   DCFTELEGRAMM5,1
                bsf     DCFMINBCD,4
                btfsc   DCFTELEGRAMM5,2
                bsf     DCFMINBCD,3
                btfsc   DCFTELEGRAMM5,3
                bsf     DCFMINBCD,2
                btfsc   DCFTELEGRAMM5,4
                bsf     DCFMINBCD,1
                btfsc   DCFTELEGRAMM5,5
                bsf     DCFMINBCD,0

;               clrf    DCFZUSATZINFOS          ;Zusaetzliche Informationen
;               btfsc   DCFTELEGRAMM5,7         ;  zusaetzliche Schaltsekunde
;               bsf     DCFZUSATZINFOS,DCFSCHALTSEK
;               btfsc   DCFTELEGRAMM6,0         ;  Sommerzeit
;               bsf     DCFZUSATZINFOS,DCFSOMMER
;               btfsc   DCFTELEGRAMM6,1         ;  Winterzeit
;               bsf     DCFZUSATZINFOS,DCFWINTER
;               btfsc   DCFTELEGRAMM6,2         ;  Vorankuendigung: Wechsel
;               bsf     DCFZUSATZINFOS,DCFSOMWIN ;   Sommerzeit <-> Winterzeit
;               btfsc   DCFTELEGRAMM6,3         ;  Reserveantenne
;               bsf     DCFZUSATZINFOS,DCFRESANT

                ;Neues Telegramm mit dem alten Vergleichen
                bcf     DCFSTATUS,DCFFEHLER     ;zuerst Fehlerflag loeschen

                movf    DCFMINALT,w             ;Minuten pruefen
                subwf   DCFMINBCD,w
                movwf   TEMP1                   ;TEMP1 = DCFMINBCD - DCFMINALT
                decfsz  TEMP1,w                 ;TEMP1 - 1 = 0 ?
                bsf     DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf    DCFSTDALT,w             ;Stunden pruefen
                subwf   DCFSTDBCD,w
                btfss   STAT,Z                  ;DCFSTDBCD - DCFSTDALT = 0?
                bsf     DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf    DCFTAGALT,w             ;Tag pruefen
                subwf   DCFTAGBCD,w
                btfss   STAT,Z                  ;DCFTAGBCD - DCFTAGBCD = 0?
                bsf     DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf    DCFMONALT,w             ;Monat pruefen
                subwf   DCFMONBCD,w
                btfss   STAT,Z                  ;DCFMONBCD - DCFMONALT = 0?
                bsf     DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf    DCFJAHRALT,w            ;Jahr pruefen
                subwf   DCFJAHRBCD,w
                btfss   STAT,Z                  ;DCFJAHRBCD - DCFJAHRALT = 0?
                bsf     DCFSTATUS,DCFFEHLER     ;nein: Fehlerflag setzen

                movf    DCFWTAGALT,w            ;Wochentag pruefen
                subwf   DCFWOCHENTAG,w
                btfss   STAT,Z                  ;DCFWOCHENTAG - DCFWTAGALT = 0?
```

```
                bsf     DCFSTATUS,DCFFEHLER   ;nein: Fehlerflag setzen

                ;Altes Telegramm durch das neue Telegramm ersetzen
                movf    DCFMINBCD,w
                movwf   DCFMINALT

                movf    DCFSTDBCD,w
                movwf   DCFSTDALT

                movf    DCFTAGBCD,w
                movwf   DCFTAGALT

                movf    DCFMONBCD,w
                movwf   DCFMONALT

                movf    DCFJAHRBCD,w
                movwf   DCFJAHRALT

                movf    DCFWOCHENTAG,w
                movwf   DCFWTAGALT

                btfsc   DCFSTATUS,DCFFEHLER   ;DCF-Telegramm korrekt?
                goto    DCFUPMINUTEENDE       ;nein: UP beenden
                movf    DCFSTDBCD,w           ;ja: DCFSTDBCD
                call    BCDBIN2               ;von BCD nach binaer umwandeln
                movwf   UHRSTUNDE             ;und die Stunde der mitlaufenden Uhr
                                              ;ueberschreiben

                movf    DCFMINBCD,w           ;DCFMINBCD
                call    BCDBIN2               ;von BCD nach binaer umwandeln
                movwf   UHRMINUTE             ;und die Minute der mitlaufenden Uhr
                                              ;ueberschreiben

                clrf    UHRSEKUNDE            ;Sekunde loeschen

                movf    DCFTAGBCD,w           ;DCFTAGBCD
                call    BCDBIN2               ;von BCD nach binaer umwandeln
                movwf   DATUMTAG              ;und den Tag des mitlaufenden Datums
                                              ;ueberschreiben

                movf    DCFMONBCD,w           ;DCFMONBCD
                call    BCDBIN2               ;von BCD nach binaer umwandeln
                movwf   DATUMMONAT            ;und den Monat des mitlaufenden Datums
                                              ;ueberschreiben

                movf    DCFJAHRBCD,w          ;DCFJAHRBCD
                call    BCDBIN2               ;von BCD nach binaer umwandeln
                movwf   DATUMJAHR             ;und das Jahr des mitlaufenden Datums
                                              ;ueberschreiben

                movf    DCFWOCHENTAG,w        ;DCFWOCHENTAG
                movwf   DATUMWTAG

                bsf     DCFSTATUS,DCFSYNC

DCFUPMINUTEENDE bcf     DCFSTATUS,DCFFEHLER
                bcf     DCFSTATUS,DCFNEUEMIN  ;Anforderungsflag loeschen

                return


;*****************************************************************************************
;** INNERE UHR                                                                          **
;**                                                                                     **
;** Aufgabe:                                                                            **
;**    Fuer den Fall dass kein gueltiges DCF-Telegramm empfangen werden kann sorgt dieses   **
;**    Unterprogramm dafuer, dass die Uhrzeit trotzdem jede Sekunde aktualisiert wird. Ist  **
;**    fuer eine laengere Zeit kein korrekter DCF-Empfang moeglich, so wuerden die Minuten, **
;**    Stunden, Tage usw. stehen bleiben, da ja im Unterprogramm DCFUPMINUTE nur gueltige   **
;**    DCF-Telegramme uebernommen werden.                                               **
;**    Dieses Unterprogramm wird also parallel zu den Unterprogrammen für die DCF-Dekod-    **
;**    ierung aufgerufen.                                                               **
;**                                                                                     **
;** Vorgehensweise:                                                                     **
;**    Die Sekunden (Register UHRSEKUNDE) um 1 erhoehen. Beinhalte dieses Register nun den  **
;**    Wert 60, so beginnt eine neue Minute. Daher die Sekunden loeschen und die Minuten    **
;**    (Register UHRMINUTE) um 1 erhoehen. Beinhaltet dieses Register nun den Wert 60, so   **
;**    beginnt eine neue Stunde. Daher die Minuten loeschen und die Stunden (Register       **
;**    UHRSTUNDE) um 1 erhoehen. Beinhaltet dieses Register nun den Wert 24, so beginnt ein **
```

```
;**     neuer Tag. Daher die Stunden loeschen. Ist das Mitzaehlen des Datums notwendig, so    **
;**     muss nun ein Register fuer den Tag (z.B. DATUMTAG) um 1 erhoeht werden und dieses     **
;**     Ueberprueft werden, wobei diese Pruefung nun nicht mehr so einfach ist, da ja jeder   **
;**     Monat unterschiedlich viele Tage besitzt. Erschwerend kommt auch noch hinzu, dass     **
;**     auch die Schaltjahre miteinbezogen werden muessen!                                    **
;**                                                                                           **
;** Achtung:                                                                                  **
;**     Dieses Unterprogramm muss daher jede Sekunde aufgerufen werden                        **
;*******************************************************************************************
INNEREUHR       incf    UHRSEKUNDE,f            ;Sekundenzaehler um 1 erhöhen
                movf    UHRSEKUNDE,w
                sublw   .60
                btfss   STAT,Z                  ;Pruefen, ob Sekunde=60
                goto    INNERERUHRFERTIG
                clrf    UHRSEKUNDE              ;Wenn Sekunde=60, Sekunde loeschen
                incf    UHRMINUTE,f             ;  und Minute um 1 erhöhen
                movf    UHRMINUTE,w
                sublw   .60
                btfss   STAT,Z                  ;Pruefen, ob Minute=60
                goto    INNERERUHRFERTIG
                clrf    UHRMINUTE               ;Wenn Minute=60, Minute loeschen
                incf    UHRSTUNDE,f             ;  und Stunde erhoehen
                movf    UHRSTUNDE,w
                sublw   .24
                btfsc   STAT,Z                  ;Pruefen, ob Stunde=24
                clrf    UHRSTUNDE               ;Wenn Stunde=24, Stunden loeschen
INNERERUHRFERTIG
                return


;****************************** Wanduhrspezifisches Unterprogramm **************************

;*******************************************************************************************
;** Zeit anzeigen:                                                                        **
;**                                                                                       **
;** Aufgabe:                                                                              **
;**     Wenn die mitlaufende Uhr mit der DCF-Uhr synchronisiert ist (Flag DCFSYNC im Register **
;**     DCFSTATUS gesetzt) die Uhrzeit wie folgt ausgeben. Andenfalls dieses Unterprogramm    **
;**     vorzeitig beenden:                                                                **
;**     + Die Stunden von "24-nach-12" umwandeln und um 1 erhoehen. In das hoeherwertige      **
;**       Nibble von TEMP2 kopieren                                                       **
;**     + die Minuten durch 5 dividieren, um 1 erhoehen und in das niederwertige Nibble von   **
;**       TEMP2 kopieren                                                                  **
;**     + TEMP2 am Port B ausgeben                                                        **
;**                                                                                       **
;** Anmerkung:                                                                            **
;**     Die temporaeren Register TEMP1 und TEMP2 werden hier nur als Hilfsregister benoetigt. **
;**     Sie koennen daher auch in anderen Unterprogrammen verwendet werden.               **
;*******************************************************************************************
ANZEIGE         btfss   DCFSTATUS,DCFSYNC       ;Ist die mitlaufende Uhr mit der DCF-Uhr
                                                ;  synchronisiert?
                goto    ANZEIGEENDE             ;nein: Unterprogramm vorzeitig beenden

                clrf    TEMP2                   ;ja: Hilfsregister (TEMP2) loeschen

                movf    UHRSTUNDE,w             ;Die Stunde
                andlw   b'00011111'

                call    TABSTUNDEN              ;  mit Hilfe der Tabelle TABSTUNDEN umwandeln
                                                ;  (von 24 nach 12 und um 1 erhoehen)
                movwf   TEMP2                   ;  und in das hoeherwertige Nibble von TEMP2
                swapf   TEMP2,f                 ;  kopieren

                movf    UHRMINUTE,w             ;Die Minuten
                andlw   b'00111111'
                call    TABDIV5                 ;  mit Hilfe der Tabelle TABDIV5 durch 5
                movwf   TEMP1                   ;  dividieren, um 1 erhoehen und in das
                incf    TEMP1,w
                xorwf   TEMP2,w                 ;  niederwertigere Nibble von TEMP2 kopieren
                movwf   PORTB                   ;TEMP2 am Port B ausgeben

ANZEIGEENDE     return


;****************************** Weitere, allgemeine Routinen *******************************

;*******************************************************************************************
;** Umwandlung von BCD nach Binaer:                                                       **
;** Aufgabe:                                                                              **
```

```
;**     + Die im w-Register stehende 2-stellige-BCD-Zahl nach binaer umwandeln                **
;**                                                                                           **
;** Vorgehensweise:                                                                           **
;**     + Die umzuwandelnde BCD-Zahl im temporaeren Register TEMP1 zwischenspeichern          **
;**     + w-Register (Arbeitsregister) loeschen.                                              **
;**     + Die BCD-Zahl bitweise ueberpruefen.                                                 **
;**         z.B.: BCD-Zahl: 0010 0110  (= 26)                                                 **
;**                         |||| ||||                                                        **
;**                         |||| ||| -   0  ( = 0 x 1)                                       **
;**                         |||| || -- + 2  ( = 1 x 2)                                       **
;**                         |||| | --- + 4  ( = 1 x 4)                                       **
;**                         ||||  ---- + 0  ( = 0 x 8)                                       **
;**                         ||| ------ + 0  ( = 0 x 10)                                      **
;**                         || ------- +20  ( = 1 x 20)                                      **
;**                         | -------- + 0  ( = 0 x 40)                                      **
;**                          --------- + 0  ( = 0 x 80)                                      **
;**                       ------------------------------                                      **
;**                                  =26                                                      **
;**     + Ergebnis in w                                                                       **
;**                                                                                           **
;** Anmerkung: Das temporaere Register TEMP1 wird hier nur zum Zwischenspeichern benoetigt    **
;**     und kann daher auch in anderen Unterprogrammen verwendet werden.                      **
;*******************************************************************************************
BCDBIN2         movwf   TEMP1           ;umzuwandelnde BCD-Zahl zwischenspeichern
                clrw                    ;Arbeitsregister (w-Register) loeschen
                btfsc   TEMP1,0         ;Ist Bit0 der BCD-Zahl gesetzt,
                addlw   .1              ;  zum w-Register den Wert 1 addieren
                btfsc   TEMP1,1         ;Ist Bit1 der BCD-Zahl gesetzt,
                addlw   .2              ;  zum w-Register den Wert 2 addieren
                btfsc   TEMP1,2         ;Ist Bit2 der BCD-Zahl gesetzt,
                addlw   .4              ;  zum w-Register den Wert 4 addieren
                btfsc   TEMP1,3         ;Ist Bit3 der BCD-Zahl gesetzt,
                addlw   .8              ;  zum w-Register den Wert 8 addieren
                btfsc   TEMP1,4         ;Ist Bit4 der BCD-Zahl gesetzt,
                addlw   .10             ;  zum w-Register den Wert 10 addieren
                btfsc   TEMP1,5         ;Ist Bit5 der BCD-Zahl gesetzt,
                addlw   .20             ;  zum w-Register den Wert 20 addieren
                btfsc   TEMP1,6         ;Ist Bit6 der BCD-Zahl gesetzt,
                addlw   .40             ;  zum w-Register den Wert 40 addieren
                btfsc   TEMP1,7         ;Ist Bit7 der BCD-Zahl gesetzt,
                addlw   .80             ;  zum w-Register den Wert 80 addieren, im w-Register
                                        ;  befindet sich nun das Ergebnis
                return


;****************************** Hauptprogramm ************************************************

;*******************************************************************************************
;** Aufgaben des Hauptprogramms                                                             **
;**     + Aufrufen von INIT (Initialisierungen)                                             **
;**     + Interrupt freigeben                                                               **
;**     + Taetigkeiten und/oder Unterprogramme, die alle 4ms durchgefuehrt werden muessen   **
;**             + Unterprogramm DCFROUTINE aufrufen                                         **
;**     + Taetigkeiten und/oder Unterprogramme, die jede Sek. durchgefuehrt werden muessen  **
;**             + Unterprogramm INNEREUHR aufrufen                                          **
;**             + Unterprogramm ANZEIGE aufrufen                                            **
;**     + Bei einem Minutenwechsel (Flag DCFNEUEMIN ist gesetzt) das Unterprogramm          **
;**       DCFUPMINUTE abarbeiten                                                            **
;**     + Bei einem Sekundenwechsel (Flag DCFNEUESEK ist gesetzt) das Unterprogramm         **
;**       DCFUPSEKUNDE abarbeiten                                                           **
;*******************************************************************************************
Beginn          call    INIT                    ;Initialisierungen
                movlw   b'10100000'             ;TMR0 freigeben durch
                movwf   INTCON                  ;Setzen von GIE und T0IE im Register INTCON

HPSCHLEIFE      btfsc   FLAGSISRHP,FLAG4MSEK
                goto    HPWEITER1

                btfsc   FLAGSISRHP,FLAG1SEK
                goto    HPWEITER2

                btfsc   DCFSTATUS,DCFNEUEMIN    ;Wenn Anforderungsflag DCFNEUEMIN gesetzt
                call    DCFUPMINUTE             ;Unterprogramm DCFUPMINUTE ausfuehren
                btfsc   DCFSTATUS,DCFNEUESEK    ;Wenn Anforderungsflag DCFNEUESEK gesetzt
                call    DCFUPSEKUNDE            ;Unterprogramm DCFUPSEKUNDE ausfuehren

                goto    HPSCHLEIFE

HPWEITER1       call    DCFROUTINE              ;Taetigkeiten, die alle 4 ms durchgefuehrt
```

```
                                        ;  werden muessen

                bcf     FLAGSISRHP,FLAG4MSEK  ;Anforderungsflag (fuer 4ms-Aktivitaeten)
                                        ;  zuruecksetzen
                goto    HPSCHLEIFE

HPWEITER2       call    INNEREUHR             ;Taetigkeiten, die jede Sekunde durchgefuehrt
                                        ;  werden muessen
                call    ANZEIGE

                bcf     FLAGSISRHP,FLAG1SEK   ;Anforderungsflag (fuer 1-Sek.-Aktivitaeten)
                                        ;  zuruecksetzen
                goto    HPSCHLEIFE

                end
```

Anhang C: Stückliste

Nr.	Bezeichnung	St.	Liefe-rant	Bestell. Nr.	E-Preis	Bemerkungen
R14	Widerstand 270 Ohm	1	Conrad	418188	0,14	
R16, R21	Widerstand 470 Ohm	2	Conrad	418218	0,14	
R19	Widerstand 680 Ohm	1	Conrad	418234	0,14	
R18	Widerstand 820 Ohm	1	Conrad	418242	0,14	
R20	Widerstand 1k5	1	Conrad	418277	0,14	
R1-R13,R15,R17	Widerstand 10k	15	Conrad	418374	0,11	
C3, C4	Keramikkondensator 22pF	2	Conrad	457167	0,14	
C1, C5-C7, C9	Keramikkondensator 100nF	5	Conrad	453358	0,25	
C2, C8	Elko 10µF/25V	2	Conrad	460524	0,14	
D1,D13,D25, D37,D49,D61, D73,D85,D97, D109,D121, D133	LED 5mm diffus, grün	12	Conrad	184730	0,36	
D2-D12,D14-D24,D26-D36, D38-D48,D50-D60,D62-D72, D74-D84, D86-D96,D98-D108, D110-D120, D122-D132, D134-D144	LED 3mm diffus rot	132	Conrad	184560	0,05	
D157,D158	LED 3mm diffus gelb	2	Conrad	184918	0,09	
D145-D156	Diode 1N4148	12	Conrad	162280	0,04	
D159	Diode 1N4001	1	Conrad	162213	0,09	
T1	Transistor BC547B	1	Conrad	155012	0,21	
IC1	PIC16F84	1	Conrad	146773	10,80	Programmiert mit wanduhr1.hex
IC2, IC3	4067	2	Conrad	173320	2,30	
IC4	Spannungsregler 7805	1	Conrad	179205	0,65	
X1	Quarz 4,096 MHz	1	Conrad	168629	1,40	
DCF1	DCF-Empfangsmodul	1	Conrad	641138	11,50	
	IC-Präzisions-Sockel 18polig	1	Conrad	189634	0,57	für IC1
	IC-Präzisions-Sockel 24polig	2	Conrad	189650	0,85	für IC2 und IC3
	Steckernetzteil PA300, unstabilisiert	1	Conrad	518305	6,47	
	Zum Steckernetzteil passendes Gegenstück	1				
	Lochraster-Platine	2	Conrad	527769	3,25	
	Sperrholzplatte 4mm dick (ca. 27 x 27 cm)	1				
	div. Montagematerial					