Programowany zegar z DCF77, część 1

AVT-5022

Proponowany układ jest „przeróbką“ układu opisanego w Elektronice Praktycznej kilka lat temu. Wydaje się jednak, że wskutek postępu w elektronice rok rozwoju tej dziedziny techniki to całe stulecie w innych dziedzinach.

Projekt zegara synchronizowanego atomowym wzorcem czasu DCF77 zaprezentowaliśmy w numerze EP7/94 i, jak na tamte czasy, był to układ bardzo nowoczesny. Trudno jednak nazwać go takim w roku 2001 i dlatego postanowiłem przedstawić Czytelnikom własne opracowanie takiego zegara, w którym wykorzystano nowoczesne, nie istniejące w połowie lat 90. elementy. Właściwie jedynym elementem zasługującym na uwagę jest naprawdę doskonały procesor, ośmielam się twierdzić, że najlepszy w swojej klasie: AT90S8535.

Zanim jednak przedstawimy szczegółowy opis układu, zapoznajmy się z podstawowymi parametrami nowego zegara DCF77.

Temat synchronizacji zegarów za pomocą sygnału DCF był poruszany na łamach Elektroniki Praktycznej dość dawno i dlatego chciałbym w największym skrócie przypomnieć, czym właściwie jest transmisja DCF i czego możemy oczekiwać po synchronizowanym nią zegarze.

Zapewnienie synchronizacji czasu na terenie całego państwa czy nawet kontynentu było od dawna bardzo istotnym problemem. Nawet najdokładniejsze chronometry stosowane w nawigacji morskiej nie zapewniały dostatecznej precyzji, a pomiary astronomiczne były czynnością bardzo kłopotliwą. Do rozwiązanie pozostawały więc dwa problemy: stworzenie wzorca czasu niezależnego od zjawisk astronomicznych i odpowiednio niezawodnego środka transmisji, zapewniającego błyskawiczne przekazywanie informacji na teren przynajmniej jednego państwa. Takim wzorcem czasu, praktycznie idealnie dokładnym, okazały się przemiany zachodzące w izotopach promieniotwórczych, a odpowiednim medium stały się fale radiowe. W wielu krajach (m. in. USA, Wielka Brytania)

Rys. 1. Przybliżony zasięg nadajnika DCF.

stworzono nadajniki czasu wzorcowego.

W Niemczech poradzono sobie z tym problemem już dość dawno, nadając sygnał z bardzo dokładnego wzorca czasu. Jest nim atomowy (cezowy) zegar czasu znajdujący się na Uniwersytecie w Braunschweig, którego błąd szacowany jest na mniejszy niż 1 sekunda na 5 milionów lat. W Mainflingen [50^{O} 0,1' N, 09^{O} 00' E) koło Frankfurtu nad Menem znajduje się nadajnik radiowy nadający na częstotliwości 77,5kHz (z mocą 50kW) dokładną informację czasowa, będącą urzędowym wzorcem czasu w Niemczech.

Informacja o czasie jest kodowana w 59-bitowych słowach przesyłanych co sekundę i zawiera dane o czasie (godziny, minuty) i dacie (rok, miesiąc, dzień, dzień tygodnia). Dodatkowo zawarte są też zapowiedzi zmiany czasu i dodatkowej sekundy oraz informacja o tym, czy obowiązuje czas letni, czy zimowy. Aby zapobiec przekłamaniom, dodano także bity parzystości pomagające wykryć błędy w transmisji.

Według oficjalnych danych, moc nadajnika zapewnia poprawny odbiór w zasięgu 2500km (**rys. 1**), a więc na terenie prawie całej kontynentalnej Europy (z wyłączeniem Islandii i części Finlandii). Praktyka jest jednak nieco inna, ponieważ ostatnio zapoznałem się z informacjami o poprawnym odbiorze transmisji DCF77 nawet na odległość do 5000 km. Oczywiście, odbiór nie był stały, ale przy dobrych warunkach propagacyjnych możliwe było odebranie kilkudziesięciu poprawnych transmisji dziennie.

Do odbioru informacji DCF przeznaczone są specjalne odbiorniki DCF Receiver dostępne w sklepach AVT. Odbiornik taki jest niezbędny do wykorzystania wszystkich możliwości zegara.

Zrealizowanie układu zegara w technice mikroprocesorowej, zresztą jedynej możliwej do zastosowania w tak skomplikowanym systemie, radykalnie uprościło jego budowę. Nie polecałbym jej może zupełnie „zielonym“ elektronikom, ale już średnio zaawansowani elektronicy, mający za sobą wykonanie kilku prostych układów, mogą zdecydować się na montaż zegara DCF77.

Opis działania

Schemat elektryczny zegara DCF77 pokazano na **rys. 2**. „Sercem“ układu jest zaprogramowany procesor typu AT90S8535, otoczony niewielką liczbą elementów dyskretnych. Schemat zegara możemy podzielić na dwie części, odpowiadające fizycznemu rozmieszczeniu elementów na dwóch płytkach obwodów drukowanych. W górnej części schematu jest blok sześciu wyświetlaczy siedmiosegmentowych DP1..DP6 i klawiatura zbudowana z sześciu przycisków S1..S6. Ta część układu połączona jest z głównym blokiem zegara za pomocą złącza CON1 + CON2.

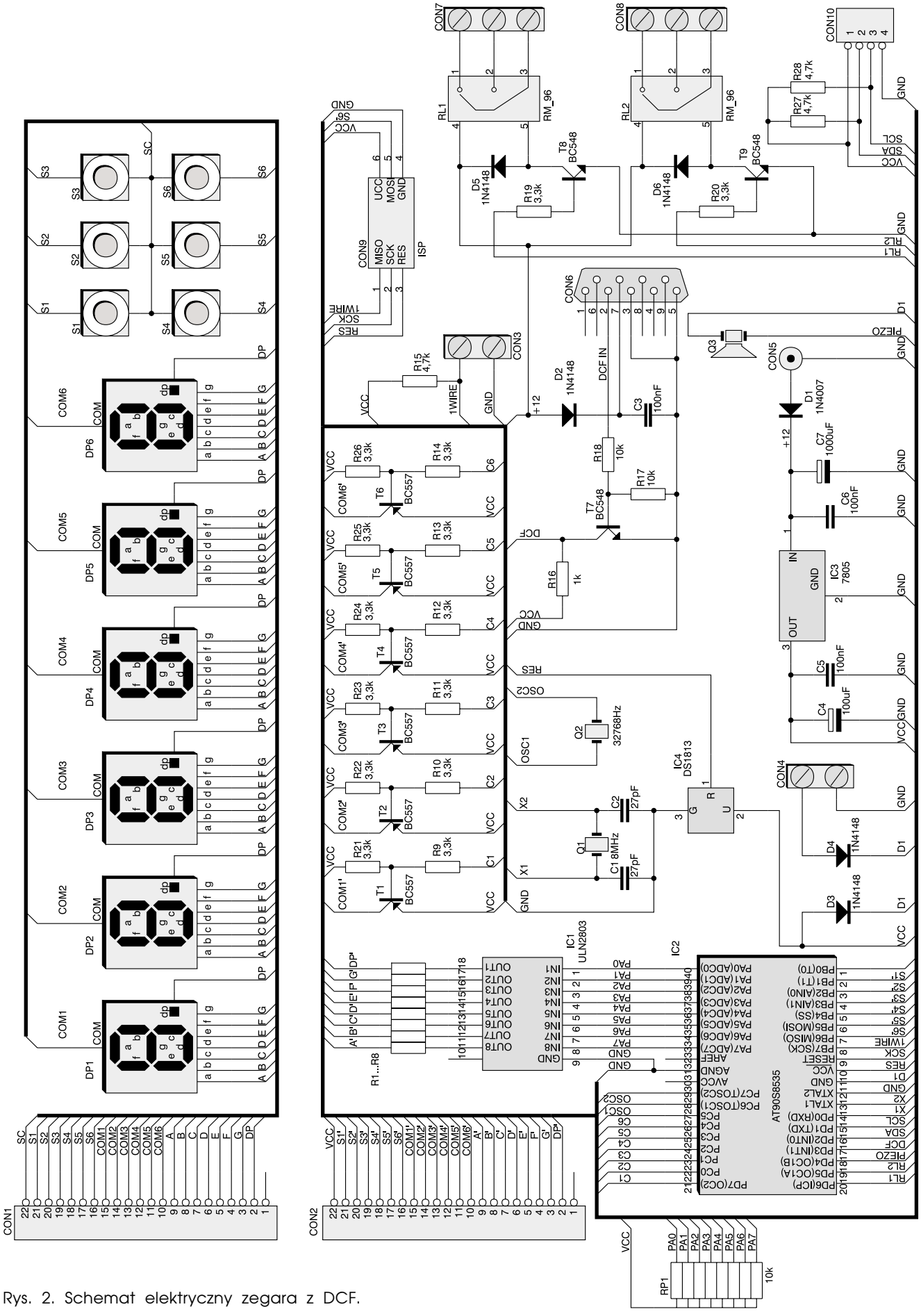
Zastosowanie wyświetlania multipleksowanego spowodowało dodanie do układu odpowiednich elementów sterujących segmentami i wspólnymi anodami wyświetlaczy. Segmenty oraz punkty dziesiętne wyświetlaczy sterowane są za pomocą driverów mocy zawartych w układzie IC1 - ULN2803, natomiast do wspólnych anod wyświetlaczy prąd jest doprowadzany za pomocą tranzystorów T1..T6.

Sygnał DCF doprowadzany jest z odbiornika do złącza DB9 - CON6, a następnie podlega inwersji i dostosowaniu do poziomu TTL w układzie z tranzystorem T7. Następnie kierowany jest na wejście przerwania INT0 procesora.

W układzie zastosowano dwa rezonatory kwarcowe Q1 i Q2 współpracujące z procesorem. Rezonator Q1, o częstotliwości podstawowej 8MHz, jest „zwykłym“ kwarcem współpracującym z oscylatorem systemowym procesora. Natomiast „zegarkowy“ kwarc Q2 stabilizuje częstotliwość dodatkowego oscylatora RTC wbudowanego w strukturę procesora.

Podstawowe dane techniczne zegara DCF77:

- ✓ Wyświetlanie czasu w formacie: godzina, minuta, sekunda.
- ✓ Wyświetlanie daty w formacie: dzień, miesiąc, rok. Opcjonalnie data może być wyświetlana z pominięciem roku, na którego miejscu prezentowana może być informacja o aktualnym dniu tygodnia.
- ✓ Budzik 1 pracujący w trybie codziennym, czyli włączający sygnał budzenia niezależnie od dnia tygodnia.
- ✓ Budzik 2 automatycznie ignorujący sobotę i niedzielę jako dni wolne od pracy i nauki w szkole. Niezależnie od generacji sygnału akustycznego obydwa budziki sterują przekaźnikiem, do którego można podłączyć urządzenia o znacznym poborze mocy.
- ✓ Timer pracujący w zakresie od 1 sekundy do 59 sekund, 59 minut i 99 godzin. Timer steruje drugim przekaźnikiem o znacznej obciążalności styków.
- ✓ Stoper o zakresie liczenia identycznym jak timer.
- ✓ Informacje prezentowane są na typowych wyświetlaczach siedmiosegmentowych LED o standardowych wymiarach. Umożliwia to odczytywanie czasu z odległości kilku, a dla osób obdarzonych sokolim wzrokiem najwyżej z odległości kilkunastu metrów. Ostatnio zapoznałem się z listem od Czytelnika, który napotkał na ogromne problemy podczas prób dołączenia do zbudowanego układu wyświetlaczy o większych wymiarach, niż przewidziane w projekcie. Problem ten nie będzie nawet w najmniejszym stopniu dotyczył naszego zegara DCF77, ponieważ bez jakichkolwiek przeróbek możemy go wyposażyć w dodatkowe wyświetlacze o wysokości cyfr równej 57mm, co umożliwi odczyt z odległości nawet kilkudziesięciu metrów!
- ✓ Dane wprowadzane są do zegara za pomocą sześcioprzyciskowej klawiatury. W każdej chwili możemy dokonać ręcznej zmiany czasu, co nie było możliwe w poprzednim “wcieleniu” zegara DCF.
- ✓ Pomimo, że nasz zegar przeznaczony jest do stałej współpracy z odbiornikiem sygnału DCF77, może on także pracować jako zwykły zegar synchronizowany kwarcem “zegarkowym” 32768Hz. Jednak dokładność wskazań znacznie wtedy spada i jest taka, jaka może być dokładność seryjnie produkowanych i nie selekcjonowanych popularnych rezonatorów kwarcowych.
- ✓ Zegar może być wyposażony w awaryjne źródło zasilania, które przy zaniku napięcia w sieci energetycznej dostarcza prądu do zasilania samego tylko procesora.



Rys. 2. Schemat elektryczny zegara z DCF.

List. 1.

```
Sub Display_time
'Przed wejściem w pętlę programową, w której prowadzona będzie kontrola czasu bieżącego,
'a także jego ewentualna ręczna korekta, program określa rodzaj wyświetlania
'oraz zawiadamia o rozpoczęciu pracy za pomocą krótkiego sygnału akustycznego:

    Display_type = 1
    Short_beep
    Do
    'W programie zadeklarowane zostały trzy zmienne pomocnicze, których wartość bezpośrednio
    'decyduje o tym, co aktualnie będzie ukazywało się na wyświetlaczach. Ponieważ w tym momencie
    'mamy zamiar wyświetlać informacje o czasie, zmienne te przyjmują następujące wartości:

        Disp1 = _sec
        Disp2 = _min
        Disp3 = _hour

        'Z podprogramu wyświetlania i korekty daty możemy przejść do wyświetlania aktualnej daty,
        'a dalej do innych funkcji. Ich zmiana dokonywana jest za pomocą naciśnięcia przycisku S1:

        Reset Portb.0
        If Pinb.0 = 1 Then Display_date

        'Do funkcji ręcznej korekty aktualnego czasu możemy przejść po jednoczesnym naciśnięciu
        'przycisków S3 i S6:

        Reset Portb.2: Reset Portb.5

        If Pinb.2 = 1 And Pinb.5 = 1 And Time_set_flag = 0 Then
        'Wejście w tryb ręcznego ustawiania nowego czasu sygnalizowane jest trzema długimi
        '(ok. 1 s) sygnałami akustycznymi. Następuje też zmiana wartości zmiennej pomocniczej
        'TIME_SET_FLAG określającej tryb pracy:
            For R = 1 To 3
                Beep
            Next R
            Time_set_flag = 1
        End If

        'Naciśniecie przycisku dołączonego do pinu 2 portu B powoduje zwiększanie wartości minut.
        'Każda taka operacja sygnalizowana jest krótkim sygnałem akustycznym. Oczywiście, taka
        'akcja jest możliwa tylko po wejściu programu w tryb ustawiania czasu.

        Reset Portb.2
        If Pinb.2 = 1 And Time_set_flag = 1 Then
            Incr _min
            If _min = 60 Then _min = 0
            Short_beep
        End If

        'Naciśniecie przycisku dołączonego do pinu 5 portu B powoduje zwiększanie wartości
        'godzin. Każda taka operacja sygnalizowana jest krótkim sygnałem akustycznym.

        Reset Portb.5
        If Pinb.5 = 1 And Time_set_flag = 1 Then
            Incr _hour
            If _hour = 24 Then _hour = 0
            Short_beep
        End If

        'Naciśniecie przycisku dołączonego do pinu 1 portu B spowoduje wyjście układu z trybu
        'ustawiania czasu, co zostanie zasygnalizowane długim sygnałem akustycznym.

        Reset Portb.1
        If Pinb.1 = 1 And Time_set_flag = 1 Then
            _sec = 0
            Time_set_flag = 0
            Beep
        End If

        'A teraz jedna z najważniejszych funkcji programu: wysyłanie danych na magistralę I^C,
        'co umożliwia dołączenie do naszego zegara dodatkowych wyświetlaczy o dużych wymiarach.
        'Dane konwertowane są najpierw na kod BCD, a następnie wysyłane na I^C pod kolejne adresy
        'układów PCF8574:
        X = Makebcd(_hour)
        I²Csend 112, X
        X = Makebcd(_min)
        I²Csend 114, X
        X = Makebcd(_sec)
        I²Csend 116, X
    Loop
End Sub
```

Podstawowym elementem wykonawczym zegara jest przetwornik piezoelektryczny z generatorem (Q3), sterowany bezpośrednio z wyjścia 4 portu D procesora. Dwa przekaźniki RL1 i RL2, których cewki zasilane są za pośrednictwem tranzystorów T8 i T9, można zastosować do sterowania urządzeniami o znacznym poborze mocy, zasilanych z sieci energetycznej. Ważnym elementem układu jest złącze magistrali I²C - CON10. Umożliwia ono dołączenie do zegara dodatkowych wyświetlaczy, np. modułów AVT-859 o wysokości 57mm.

Zegar powinien być zasilany napięciem stałym o wartości ok. 12VDC, doprowadzonym do złącza CON5. Stabilizator scalony zrealizowany na układzie IC3 dostarcza napięcia +5VDC, niezbędnego do zasilania części cyfrowej układu zegara.

Należy zwrócić uwagę na nietypowe zasilanie procesora, który jest dołączony do szyny zasilającej VCC o napięciu +5VDC za pośrednictwem diody D3. W związku z tym napięcie zasilania procesora jest zmniejszone o ok. 0,6V i wynosi tylko ok. 4,4VDC. Co spowodowało zastosowanie tak nietypowego rozwiązania? To cała historia. Podczas uruchamiania kilku układów z procesorem AT90S8535, wykorzystujących wbudowany w jego strukturę oscylator i generator przerwań RTC, napotkałem na nieoczekiwane i dziwaczne trudności. W niektórych układach oscylator nie działał w ogóle, a w innych pracował w niekontrolowany sposób, włączając się i wyłączając w nieoczekiwanych momentach. Ani sprawdzanie części hardware'owej układu, ani kodu programu nie dawało rezultatu, podobnie jak wertowanie karty katalogowej procesora. Na rozwiązanie problemu natknąłem się dopiero podczas lektury erraty do karty

Tab. 1. Sposób kodowania informacji DCF

Numer impulsu (numer sekundy)	Znaczenie impulsu
0	Początek transmisji. Zawsze = 0.
1-14	Przerwa, bez znaczenia - wszystkie zera.
15	0- antena normalna; 1- antena pomocnicza.
16	0-normalnie; 1- zapowiedz zmiany czasu (przez godzinę przed zmianą).
17-18	(w kolejności bity 18,17) 10-czas zimowy; 01-czas letni.
19	0-normalnie; 1-zapowiedz dodatkowej sekundy.
20	Start informacji czasowej. Zawsze = 1.
21-24	(w kolejności bity 24,23,22,21) jednostki minut w BCD.
25-27	(w kolejności bity 27,26,25) dziesiątki minut w BCD.
28	bit parzystości dla bitów 21-27.
29-32	(w kolejności bity 32,31,30,29) jednostki godzin w BCD.
33-34	(w kolejności bity 34,33) dziesiątki godzin w BCD.
35	bit parzystości dla bitów 29-34.
36-39	(w kolejności bity 39,38,37,36) jednostki dni miesiąca w BCD.
40-41	(w kolejności bity 41,40) dziesiątki dni miesiąca w BCD.
42-44	(w kolejności bity 44,43,42) dni tygodnia w BCD - 1=Pn; 7=Nd.
45-48	(w kolejności bity 48,47,46,45) jednostki miesiąca w BCD.
49	dziesiątki miesiąca w BCD.
50-53	(w kolejności bity 53,52,51,50) jednostki lat w BCD.
54-57	(w kolejności bity 57,56,55,54) dziesiątki lat w BCD.
58	bit parzystości dla bitów 36-57.
59	brak impulsu.

List. 2.

```
Multiplexing:
'Po zgłoszeniu przerwania timera0 wykonane zostaną następujące czynności:
Porta = 0            'wstępne ustawienie portu A w stan niski
Portc = 255          'wstępne ustawienie portu C w stan wysoki
Incr Digit_number    'zwiększenie numeru kolejnego wyświetlacza (kolejnej pozycji dziesiętnej)
If Digit_number = 7 Then Digit_number = 1     'zamknięcie cyklu zliczania wyświetlaczy do modulo 6

Select Case Digit_number     'w zależności od numeru wyświetlacza

    Case 1:
    'dziesiątki sekund
        Temp = Disp1 / 10        'obliczenie wartości cyfry, która ma zostać ukazana na pierwszym
                                 'wyświetlaczu
        Porta = Lookup(temp, 7segment)               'przekodowanie otrzymanej wartości na kod
                                                     'wyświetlacza siedmiosegmentowego
        Reset Portc.1            'włączenie zasilania anod segmentów pierwszego wyświetlacza

    Case 2:
    'sekundy
        Temp = Disp1 / 10        'obliczenie wartości cyfry, która ma zostać ukazana na pierwszym
                                 'wyświetlaczu
        Temp = Temp * 10
        Temp = Disp1 - Temp
        Porta = Lookup(temp, 7segment)               'przekodowanie otrzymanej wartości na kod
                                                     'wyświetlacza siedmiosegmentowego
        Reset Portc.0            'włączenie zasilania anod segmentów pierwszego wyświetlacza

    Case 3:
    'dziesiątki minut
    'konstrukcja programu analogiczna do dziesiątek sekund

    Case 4:
    'minuty
    'konstrukcja programu analogiczna do sekund

    Case 5:
    'dziesiątki godzin
        Temp = Disp3 / 10
        Porta = Lookup(temp, 7segment)
        If Dcf_receiving_flag = 1 Then Porta = Porta + 1
            'jedno z wielu dodatkowych uwarunkowań umieszczonych w podprogramie obsługi
            'wyświetlania multipleksowanego, omówione dla przykładu. Dodanie 1 do wartości
            'wysyłanej do portu A powoduje migotanie kropki dziesiętnej, synchronicznie
            'z odbieranymi impulsami kodu DCF.
        Reset Portc.5

    Case 6:
    'godziny
    'konstrukcja programu analogiczna do sekund

End Select

Return
'Przekodowywanie wartości na kod wyświetlacza siedmiosegmentowego odbywa się na podstawie
'danych zawartych w poniższej tabeli:
7segment:
Data 252, 96, 218, 242, 102, 182, 190, 224, 254, 246
```

katalogowej, gdzie firma ATMEL umieściła rozbrajającą uwagę:

„(..) When using an external 32 kHz crystal as asynchronous clock source for Timer2, the timer may count incorrectly at voltages above 4.0V. Keep the supply voltage below 4.0V when clocking Timer2 from an external crystal. (..)“, z której wynika, że przy kwarcu 32kHz napięcie zasilania powinno być mniejsze niż 4V.

Nie wnikam w to, dlaczego generator kwarcowy 32768Hz przy zasilaniu procesora napięciem większym od 4V działa nieprawidłowo. Ważne jest tylko to, że obniżenie napięcia o 0,6V spowodowało natychmiastowe usunięcie problemów ze sprzętowym RTC. Jeżeli już procesor jest zasilany poprzez diodę separującą go od reszty układu, to nic prostszego jak zapewnić mu zasilanie awaryjne, które można dołączyć do złącza CON4.

Aby poznać działanie zegara, należy prześledzić sterujący nim program.

Zacznijmy od najprostszej funkcji, jaką jest zwykłe wskazywanie aktualnego czasu, na razie bez omawiania sposobu synchronizowania go z sygnałem DCF77. Aby zbudować metodami programistycznymi zegar czasu rzeczywistego, zwykle musimy się trochę pomęczyć. Musimy obliczyć wartość, którą będzie przeładowywany timer odpowiedzialny za odmierzanie czasu, przygotować procedurę obsługi przerwania, w której zliczane będą sekundy i minuty określające upływający czas. Oprogramowanie zegara zostało napisane w języku MCS BASIC, w którym taki sposób tworzenia programowego RTC byłby zbyt nieudolny. A zatem wystarczy tylko napisać:

```
Config Clock = Soft,
Gosub = Sectic
```

aby poinstruować kompilator o konieczności utworzenia w programie zegara RTC, którego działanie oparte jest na przerwaniach otrzymywanych z timera2. Od tego momentu mamy do dyspozycji, podobnie jak w starym QBASIC, dwie zmienne główne: `TIME$` i `DATE$` oraz zmienne pomocnicze: `_sec`, `_min`, `_hour`, `_day`, `_month` i `_year`. Wszystkie czynności związane z odliczaniem czasu, wyznaczaniem liczby dni miesięcy czy też pilnowaniem kolejnych lat przestępnych są wykonywane odtąd automatycznie, a podprogram RTC zajmuje w pamięci procesora ok. 300B. Programowy RTC z BASCOM-a ma tylko jedną wadę, o której lojalnie informuje nas Autor programu: zawiera w sobie pluskwę, która da znać o sobie w roku 2101, gdyż zostanie on zidentyfikowany jako rok 2001. No cóż, chyba niewiele nas to obchodzi.

WYKAZ ELEMENTÓW

Rezystory
RP1: R-PACK SIL10kΩ
R1..R8: 47Ω
R9..R14, R19..R26: 3,3kΩ
R15, R27, R28: 4,7kΩ
R16: 1kΩ
R17, R18: 10kΩ

Kondensatory
C1, C2: 27pF
C3, C5, C6: 100nF
C4: 100µF/10V
C7: 1000µF/16V

Półprzewodniki
D1: 1N4007
D2..D6: 1N4148
DP1..DP6: wyświetlacz siedmiosegmentowy LED wsp. anoda
IC1: ULN2803
IC2: AT90S8535
IC3: 7805
IC4: DS1813
T1..T6: BC557
T7..T9: BC548

Różne
Q1: rezonator kwarcowy 8MHz
Q2: rezonator kwarcowy 32768Hz
Q3: przetwornik piezoelektryczny z generatorem
CON1, CON2: goldpin kątowy 22pin
CON3, CON4: ARK2 (3,5mm)
CON5: gniazdko zasilania
CON6: DB9/M lutowane w płytkę
CON7, CON8: ARK3
CON10: 4 x goldpin
RL1, RL2: RM96/12V
S1..S6: microswitch

To, że wykorzystujemy timer2 do obsługi programowego RTC nie oznacza, że nie możemy wykorzystywać tak „miłej" gratki, jak przerwania generowane dokładnie co jedną sekundę, także do innych celów. W poleceniu konfigurującym zegar czasu rzeczywistego możemy dodać (i dodaliśmy) dyrektywę, wskazującą do jakiego podprogramu ma następować skok po upływie każdej kolejnej sekundy. W tym przypadku jest to podprogram *SECTIC*.

Na początek zajmijmy się najprostszym fragmentem programu, który odpowiedzialny jest za wyświetlanie i dokonywanie ręcznych korekt bieżącego czasu. Wspomnijmy tylko jeszcze, że pozostałe dwa timery zawarte w strukturze procesora: TIMER0 i TIMER1 zostały także uruchomione i skonfigurowane w następujący sposób:

```
Config Timer1 = Timer
Prescale = 64
Config Timer0 = Timer
Prescale = 64
On Timer1 Dcf_start
On Timer0 Multiplexing
```

Przerwania generowane przez timer0 wykorzystywane będą do sterowania multipleksowanym wyświetlaniem danych, a timer1 obsługiwać będzie procedury pomiaru czasu trwania impulsów kodu DCF. Przykładową procedurę pokazano na **list. 1**.

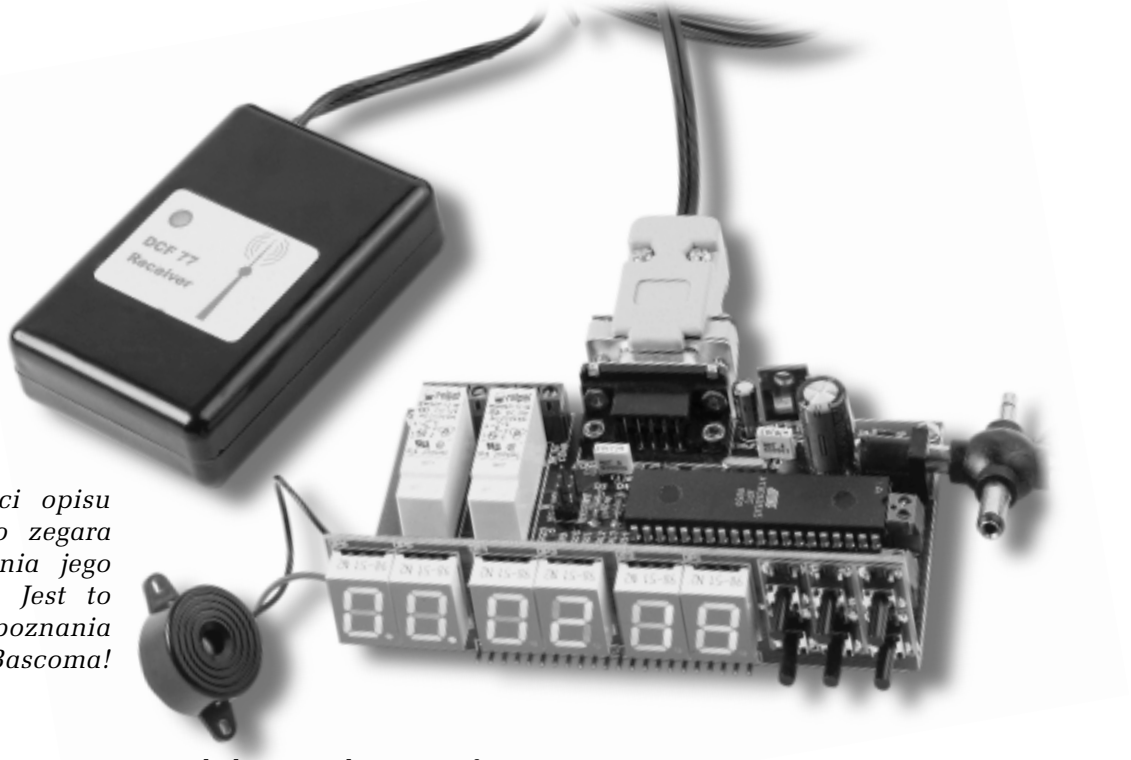
Podprogramy realizujące wyświetlanie i korektę daty, wyświetlania i ustawianie budzików i timerów są skonstruowane bardzo podobnie do programu wyświetlania czasu. Nie będziemy zatem ich analizować i przejdziemy do zapoznania się z podprogramem obsługi wyświetlaczy siedmiosegmentowych (**list. 2**). Skok do tego podprogramu następuje przy każdym wystąpieniu przerwania timer0, wówczas zmiana wyświetlanych cyfr odbywa się z częstotliwością ok. 488Hz (pamiętamy o włączeniu preskalera timera o stopniu podziału 64). Podprogram obsługi wyświetlania multipleksowanego prezentuję w postaci bardzo uproszczonej, ale wystarczającej do zapoznania się z jego działaniem. Omawianie wszystkich dodatkowych uwarunkowań umieszczonych w tym podprogramie, a służących zróżnicowaniu wyświetlania w zależności od aktualnej funkcji zegara tylko gmatwałoby opis programu, nie wnosząc wiele w zrozumienie jego działania.

Zbigniew Raabe, AVT
zbigniew.raabe@ep.com.pl

Wzory płytek drukowanych w formacie PDF są dostępne w Internecie pod adresem: ***http://www.ep.com.pl/?pdf/czerwiec01.htm*** *oraz na płycie CD-EP06/2001B w katalogu* ***PCB***.

Programowany zegar z DCF77, część 2

AVT-5022



W drugiej części opisu programowanego zegara z DCF77 autor odsłania jego programowe tajniki. Jest to doskonała okazja do poznania możliwości Bascoma!

Chyba najciekawszym fragmentem programu sterującego zegarem jest podprogram analizujący odbierany sygnał DCF77 i korygujący aktualny czas oraz datę. Przedstawiono go na **list. 3**.

Uruchomiony został *TIMER1*, skonfigurowany do pracy z preskalerem o stosunku podziału 64. A zatem częstotliwość podawana na wejście rejestru tego timera wynosi dokładnie 125000Hz. Należy też zauważyć, że w przypadku wystąpienia przepełnienia timera1 nastąpi skok do podprogramu *DCF_START* (dyrektywa *ON TIMER DCF_START*).

Podczas konfigurowania programu udzielone zostało także zezwolenie na obsługę przerwania zewnętrznego *INT0*. To, czy przerwanie będzie inicjowane opadającym, czy wstępującym zboczem sygnału możemy określić za pomocą polecenia konfiguracyjnego *CONFIG INT0 = FALLING* lub *CONFIG INT0 = RISING*. Podczas analizy podprogramu dekodowania transmisji musimy pamiętać, że sygnał DCF został odwrócony w fazie (zanegowany) przez układ z tranzystorem T7.

Omówiliśmy sposób dekodowania sygnału DCF, który okazał się niezbyt skomplikowany. Pozostała jednak otwarta jedna sprawa: skąd program wie, że jest to sygnał DCF i należy rozpocząć jego dekodowanie? Na szczęście jednoznaczne określenie startu transmisji sygnału DCF nie jest także sprawą trudną. Zauważmy, że transmisja kończy się na 58 impulsie, a impuls 59 w ogóle nie występuje. Czas trwania przerwy pomiędzy początkami wszystkich impulsów jest stały i wynosi dokładnie jedną sekundę. Jest tylko jeden wyjątek: przerwa pomiędzy ostatnim i pierwszym impulsem wynosi nie jedną, ale dwie sekundy! To właśnie zjawisko wykorzystamy do jednoznacznego określenia początku transmisji sygnału (kodu) DCF.

Podczas występowania impulsów DCF *TIMER1* był wykorzystywany do pomiaru ich czasu trwania. Podczas pomiaru czasu prawidłowego impulsu timer1 nie ulegał nigdy przepełnieniu, a maksymalna wartość jego rejes-

List. 3.

```
'Podprogram analizujący odbierany kod DCF77
Dcf:
    Pause_counter = 0    'wyzerowanie zmiennej określającej czas upływający pomiędzy impulsami,
                         'patrz: listing podprogramu DCF_START
    Dcf_flag = Not Dcf_flag      'zmienna określająca poziom impulsu DCF zostaje zanegowana

    If Dcf_flag = 0 Then 'jeżeli na wejściu INT0 jest stan niski, co oznacza początek impulsu, to:
        Config Int0 = Rising     'przerwanie INT0 ma reagować na wstępujące zbocze sygnału
        Dcf_receiving_flag = 1   'ustaw zmienną sygnalizującą fakt odbierania impulsu
        Counter1 = 0             'wyzeruj rejestry timera0
        Start Timer1             'w celu zmierzenia czasu trwania impulsu uruchom timer0
    Else     'jeżeli na wejściu INT0 jest stan wysoki, co oznacza koniec odbierania impulsu, to:
        Stop Timer1      'wstrzymaj pracę timera0
        Config Int0 = Falling    'przerwanie INT0 ma reagować na opadające zbocze sygnału
        Dcf_receiving_flag = 0   'wyzeruj zmienną sygnalizującą fakt odbierania impulsu

'W tym momencie program ma "trochę czasu" na dokonanie analizy czasu trwania odebranego
'uprzednio impulsu DCF i wyciagnięcie z niej odpowiednich wniosków. Dla impulsu o czasie trwania
'100 ms, czyli oznaczającego logiczne 0, stan timera0 powinien wynosić 12500. A zatem,
'uwzględniając konieczny margines błędu wynikający z cech transmisji AM:

        If Timer1 < 15000 And Timer1 > 10000 Then Dcf_bit = 0
'jeżeli stan rejestru timera0 zawiera się pomiędzy 10000 a 15000, to odebrany bit ma wartość 0

        If Timer1 > 20000 And Timer1 < 30000 Then Dcf_bit = 1
'jeżeli stan rejestru tmera0 zawiera się pomiędzy 20000 a 30000, to odebrany bit ma wartość 1

'Czas trwania impulsu DCF drastycznie wykraczający poza zadane wartości może świadczyć tylko o
błędzie w transmisji i musi powodować jej anulowanie. A zatem:
        If Timer1 < 10000 Then Start_dcf_flag = 0
'anuluj transmisję jeżeli zawartość rejestru timera0 jest mniejsza od 10000

        If Timer1 < 20000 And Timer1 > 15000 Then Start_dcf_flag = 0
'anuluj transmisję jeżeli zawartość rejestr timera0 jest mniejsza od 20000 i większa od 15000

        If Start_dcf_flag = 1 Then
'Jeżeli czas trwania odebranego impulsu mieścił się w zadanym przedziale i została określona jego
'wartość logiczna, to w zależności od numeru impulsu program musi wykonać następujące czynności.
        Select Case Dcf_counter
            Case 1: _sec =0          'jeżeli odebrany został impuls o numerze 0, to wyzeruj
                                     'zmienną sekund
            Case 21: Dcf_temp.0 = Dcf_bit'bit 0 zmiennej pomocniczej DCF_TEMP
                                          'przyjmuje wartość odebranego impulsu
            Case 22: Dcf_temp.1 = Dcf_bit'bit 1..
            Case 23: Dcf_temp.2 = Dcf_bit'bit 2..
            Case 24: Dcf_temp.3 = Dcf_bit'bit 3..
                    Dcf_min = Dcf_temp           'zakończyła się transmisja informacji
                                         'o jednostkach aktualnej minuty i w związku z tym
                                         'zmienna pomocnicza DCF_MIN przyjmuje obliczoną wartość
                                         'zmiennej DCF_TEMP,
                    Dcf_temp = 0         'która następnie zostaje wyzerowana
            Case 25: Dcf_temp.0 = Dcf_bit'bit 0 zmiennej pomocniczej DCF_TEMP
                                          'przyjmuje wartość odebranego impulsu
            Case 26: Dcf_temp.1 = Dcf_bit'bit 1..
            Case 27: Dcf_temp.2 = Dcf_bit'bit 2..
                    Dcf_temp = Dcf_temp * 10     'zakończyła się transmisja informacji
                                         'o dziesiątkach aktualnej minuty i w związku z tym
                                         'obliczona wartość zostaje pomnożona przez 10,
                                         'a następnie:
                    Dcf_min = Dcf_temp + Dcf_min     'obliczona zostaje ostateczna wartość minut
                    Dcf_temp = 0                     'zmienna pomocnicza DCF_TEMP zostaje wyzerowana

'Omawianie dekodowania godzin, dnia miesiąca, miesiąca, roku i dnia tygodnia nie ma chyba
'większego sensu. Odbywa się ono na takiej samej zasadzie, co dekodowanie aktualnej minuty i jego
'analiza nie wnosiłaby niczego nowego w zrozumienie zasady działania programu. Pozwólmy zatem
'programowi mozolnie dekodować następne informacje i przenieśmy się na sam koniec transmisji,
'sygnalizowany odebraniem 58 impulsu

            Case 58:
                    Start_dcf_flag = 0    'koniec transmisji, zerujemy jej flagę
                    _min = Dcf_min        'przyporządkowanie zmiennej minut odebranej wartości
                    _hour = Dcf_hour      'przyporządkowanie zmiennej godzin odebranej wartości
                    _month = Dcf_month    'przyporządkowanie zmiennej miesiąca odebranej wartości
                    _day = Dcf_day        'przyporządkowanie zmiennej dnia miesiąca odebranej wartości
                    _year = Dcf_year      'przyporządkowanie zmiennej roku odebranej wartości
                    _week_day = Dcf_week_day         'przyporządkowanie zmiennej dnia tygodnia
                                                     'odebranej wartości
        End Select           'koniec wyboru kolejnego impulsu

        Incr Dcf_counter     'zwiększ wartość licznika impulsów o 1
        Counter1 = 0         'wyzeruj rejestry timera1
        Start Timer1         'uruchom timer1
        End If

    End If
Return
```

tru wynosiła ok. 30000. Po dokonaniu pomiaru Timer1 zostaje ponownie uruchomiony w innym celu: zmierzenia czasu trwania przerwy pomiędzy impulsami. Ta przerwa podczas trwania transmisji wynosi 800 lub 900ms, co wiąże się z jednokrotnym przepełnieniem Timera1, natomiast po jej zakończeniu prawie 2 sekundy, co spowoduje dwukrotne wystąpienie przerwania i dwukrotny skok do podprogramu *DCF_START*.

```
Dcf_start:
Incr Pause_counter
'zwiększ o 1 wartość zmiennej
'określającej czas przerwy
'pomiędzy impulsami
If Pause_counter = 2 Then
  Pause_counter = 0
  Dcf_counter = 0
  'licznik impulsów kodu DCF
  ' zostaje wyzerowany
  Start_dcf_flag = 1
  'wskaźnik rozpoczęcia i trwania
  'transmisji DCF zostaje
  'ustawiony na 1
  'TU WŁAŚNIE NASTĘPUJE
  'POCZĄTEK TRANSMISJI DCF77

End If
Return
```

Mam nadzieję, że analiza przedstawionych listingów fragmentów programu sterującego naszym zegarem pozwoli Czytelnikom na pełne zrozumienie sposobu dekodowania sygnału DCF i zasady pracy programowego zegara czasu rzeczywistego (RTC). Nie wspomnieliśmy jeszcze o dodatkowych funkcjach zegara: o sposobie realizacji timera i budzików, a także o stoperze.

Zarówno kontrola zgodności bieżącego czasu z ustawionym czasem budzików, jak i odliczanie czasu przez timer i stoper odbywa się w podprogramie *SEC_TIC*, do którego wykonywany jest skok po upływie każdej sekundy (tak, jak to zostało ustalone w konfiguracji programowego RTC). Dodatkowo, w podprogramie *SEC_TIC* jest instrukcja warunkowa pozwalająca na detekcję upływu kolejnych minut:

```
Sectic
If Temp2 <> _min Then
  'Czynności do wykonania po
  'upływie kolejnej minuty, np.
  'porównanie czasu budzika
End If
  'Pozostałe czynności do
  'wykonania
Temp2 = _sec
Return
```

Podprogram „obsługi" timera i jednego z budzików zamieszczono na **list. 4**.

Program sterujący pracą zegara został z konieczności opisany bardzo fragmentarycznie, ponieważ pełny jego listing zajmuje około 5 stron formatu A4. Chciałem jedynie przedstawić Czytelnikom jego najważniejsze fragmenty, a w szczególności sposób dekodowania sygnału DCF77.

Montaż i uruchomienie

Na **rys. 3** przedstawiono rozmieszczenie elementów na dwóch płytkach obwodów drukowanych, wykonanych na laminacie dwustronnym z metalizacją. Montaż zegara rozpoczynamy od wlutowania w płytkę rezystorów i podstawek pod układy scalone, a kończymy

List. 4.

```
Sectic:
If Temp2 <> _min Then            'jeżeli upłynęła kolejna minuta, to:
    Readeeprom Alarm_hours, 3'odczytaj z pamięci EEPROM dane o godzinie budzenia
    Readeeprom Alarm_minutes, 4  'odczytaj z pamięci EEPROM dane o minucie budzenia
    If Alarm_hours = _hour And Alarm_minutes = _min And Alarm_on_flag = 1 Then
                                 'jeżeli odczytane dane są zgodne z aktualnym czasem oraz
                                 'jeżeli budzik był aktywny, to:
        Alarm_counter = 30       'licznik trwania alarmu zostaje ustawiony na 30
        Alarm_counter_flag = 1   'flaga włączenie alarmu zostaje ustawiona na 1
        Set Portd.6              'zostaje włączony przekaźnik
    End If
' dalsze czynności do wykonania po upływie minuty
'.............................................
End If

If Timer_on_flag = 1 Then        'jeżeli timer jest włączony, to
    Decr Timer_seconds           'zmniejsz wartość sekund timera
    If Timer_seconds = 255 Then  'jeżeli wartość sekund wynosi 255, to:
        Decr Timer_minutes       'zmniejsz wartość minut
        Timer_seconds = 59
        If Timer_minutes = 255 Then  'jeżeli wartość minut wynosi 255, to:
            Decr Timer_hours
            Timer_minutes = 59       'wartość minut staje się równa 59
            If Timer_hours = 255 Then    'jeżeli wartość godzin wynosi 255, to:
                Timer_on_flag = 0        'koniec pracy timera
                Readeeprom Timer_minutes,6          'ponownie odczytaj z pamięci EEPROM zapisane
                                                    'tam wartość minut timera
                Readeeprom Timer_seconds, 7         'ponownie odczytaj z pamięci EEPROM zapisane
                                                    'tam wartość sekund timera
                Readeeprom Timer_hours, 5 'ponownie odczytaj z pamięci EEPROM zapisane
                                                    'tam wartość godzin timera
                Reset Pord.6         'wyłącz przekaźnik timera
                Beep                 'wygeneruj sygnał akustyczny
            End If
        End If
    End If
End If

'w podprogramie SECTIC odbywa się także odliczanie czasu trwania alarmu budzika:

'jeżeli alarm był uaktywniony, to zmniejsz wartość jego licznika
If Alarm_counter_flag = 1 Then Decr Alarm_counter

'jeżeli licznik czasu trwania alarmu osiągnął 0, to zresetuj sygnalizację włączenia alarmu
If Alarm_counter = 0 Then Alarm_counter_flag = 0

'jeżeli alarm jest aktywny, to naprzemiennie włączaj sygnalizację akustyczną
If Alarm_counter_flag = 1 And Flash_flag = 1 Then Portd.4 = Not Portd.4

Return
```

na zamontowaniu kondensatorów elektrolitycznych i pozostałych elementów o dużych gabarytach.

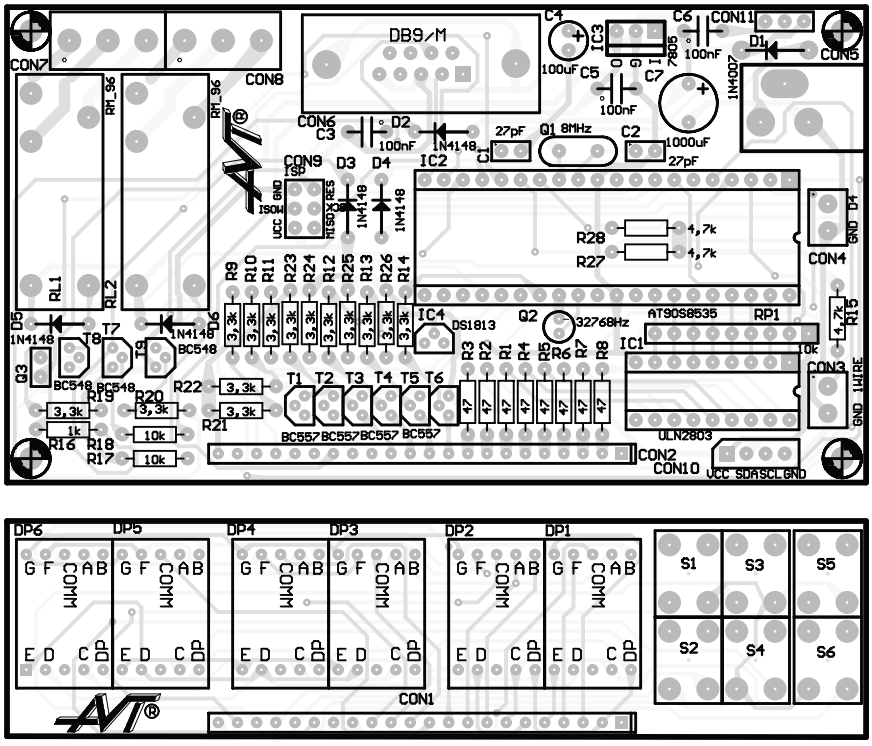
Zmontowane płytki musimy połączyć ze sobą za pomocą kątowych goldpinów. Taki sposób montażu zapewni nie tylko solidne połączenie mechaniczne, ale także ustawienie płytek idealnie pod kątem prostym względem siebie.

Odbiornik DCF77 dołączamy do złącza DB9 umieszczonego na tylnej krawędzi płytki bazowej zegara. Odbiornik nie wymaga oddzielnego źródła zasilania, a o jego działaniu świadczy migotanie diody umieszczonej w jego obudowie oraz punktu dziesiętnego na pierwszym wyświetlaczu zegara.

Sygnał DCF77 nadawany jest na bardzo niskiej częstotliwości - 77kHz. Dlatego fala radiowa rozchodzi się przy ziemi i jest podatna na zakłócenia bardzo zależne od warunków pogodowych i pory dnia. Zdarza się, że w niektórych rejonach o silnych zakłóceniach odbiór tego sygnału jest czasami niemożliwy. Dotyczy to zwłaszcza dużych aglomeracji miejskich. Zwiększenie zakłóceń następuje o wschodzie i zachodzie słońca oraz w obecności urządzeń elektronicznych (monitory, komputery, telewizory, silniki itd.). Dlatego bardzo ważne jest, aby znaleźć dla odbiornika jak najlepsze miejsce. Istotne jest także jego zorientowanie względem nadajnika oraz odległość odbiornika od urządzeń elektronicznych (zalecane jest minimum 2m). Dotyczy to w szczególności komputerów i komutatorowych silników elektrycznych, zarówno AC, jak i DC. Natomiast, wbrew wcześniejszym obawom i złym doświadczeniom sprzed paru lat, nie stwierdziłem poważniejszych zakłóceń pracy odbiornika wywoływanych przez procesor sterujący zegarem. Odbiornik pracował poprawnie nawet po umieszczeniu go w odległości kilku centymetrów od procesora.

Odbiornik powinien leżeć na płaskiej powierzchni (dioda LED do góry), nie może leżeć pod katem, ani w pobliżu metalowych przedmiotów. Należy uzyskać jak najlepszy sygnał poprzez obracanie odbiornika wokół jego osi. Można to poznać po regularnym zapalaniu się diody LED. Powinna ona zapalać się co 1s na czas ok. 0,1 do 0,2 sekundy i gasnąć.

Układ zegara zmontowany ze sprawdzonych elementów nie wymaga uruchamiania i działa natychmiast po włożeniu w podstawkę zaprogramowanego procesora. Jednak to, że działa, nie oznacza wcale, że już umiemy go obsługiwać. Zajmijmy się więc nieco rozbudowanymi procedurami obsługi.



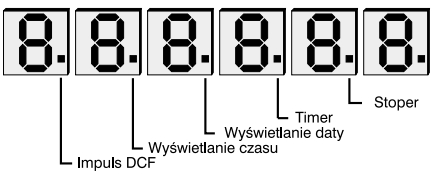
Rys. 3. Rozmieszczenie elementów na płytkach drukowanych.

Tab. 1. Zestawienie funkcji pełnionych przez klawiaturę zegara.

	S1	S2	S3	S4	S5	S6
Czas	Przejście do kolejnej funkcji		+ S6 – przejście do ustawiania czasu			+ S3 - przejście do ustawiania czasu
Ustawianie czasu		Koniec ustawiania	Zmiana minut			Zmiana godzin
Data	Przejście do kolejnej funkcji					
Ustawianie daty		Koniec ustawiania	Zmiana dnia miesiąca			Zmiana miesiąca
Budzik 1	Przejście do kolejnej funkcji					
Ustawianie budzika 1		Koniec ustawiania	Zmiana minut			Zmiana godzin
Budzik 2	Przejście do kolejnej funkcji	Budzik włączony/ /wyłączony				
Ustawianie budzika 2		Koniec ustawiania	Zmiana minut			Zmiana godzin
Timer	Przejście do kolejnej funkcji	Start Timer		Stop Timer	Reset timer	
Ustawianie timera		Koniec ustawiania	Zmiana minut	Zmiana sekund		Zmiana godzin
Stoper	Przejście do kolejnej funkcji	Start stoper		Stop stoper	Reset stoper	

Po pierwszym włączeniu zasilania układ przechodzi automatycznie w tryb wyświetlania aktualnego czasu, z tym że na wyświetlaczach ukazuje się początkowo godzina 00:00, a zegar rozpoczyna zliczanie czasu od tej wartości. Mamy teraz dwie możliwości do wyboru: albo poczekać na odebranie transmisji DCF77 i automatyczne skorygowanie wskazywanego czasu i dat, albo wykonać to ręcznie. Ponieważ jednak oczekiwanie na zdekodowanie transmisji może trwać do 2 minut (nawet przy dobrych warunkach propagacyjnych), dokonajmy ręcznej korekty czasu.

Podczas wyświetlania czasu, podobnie jak przy korzystaniu z innych funkcji zegara, możemy przejść w tryb ustawiania naciskając jednocześnie klawisze S3 i S6.



Rys. 4. Znaczenie kropek dziesiętnych przy cyfrach wyświetlacza.

Zmiana trybu pracy sygnalizowana jest trzykrotnym sygnałem akustycznym, którego zadaniem jest ostrzeżenie operatora, że dalsze naciskanie klawiszy może wprowadzić istotne zmiany w pracy zegara.

Do ustawiania czasu, daty oraz innych wartości wykorzystujemy także klawisze S3 i S6. Naciskanie klawisza S3 powoduje cykliczną zmianę minut, a klawisza S6 - godzin. Ustawianie czasu kończymy naciskając klawisz S2. Zestawienie funkcji inicjowanych za pomocą poszczególnych klawiszy klawiatury zegara zamieszczono w **tab. 1**.

Na **rys. 4** przedstawiono wykorzystanie kropek dziesiętnych na wyświetlaczach. Sygnalizują one różne tryby pracy zegara z wyjątkiem budzików. Wejście w tryb ustawiania lub kontroli budzików sygnalizowane jest bowiem wyświetleniem „A1“ lub „A2“ na dwóch pierwszych wyświetlaczach.

Zbigniew Raabe, AVT
zbigniew.raabe@ep.com.pl

Wzory płytek drukowanych w formacie PDF są dostępne w Internecie pod adresem: ***http://www.ep.com.pl/?pdf/lipiec01.htm*** *oraz na płycie CD-EP07/2001B w katalogu* ***PCB***.