ZESZYTY NAUKOWE POLITECHNIKI SLĄSKIEJ 1990

Seria: AUTOMATYKA z.101 Nr kol.1083

Piotr Birn, Henryk Krawczyk

Instytut Informatyki Politechniki Gdańskiej

UNIWERSALNY INTERFEJS DLA SYSTEMÓW MIKROKOMPUTEROWYCH CZASU RZECZYWISTEGO

Streszczenie. Interfejs określono jako zbiór funkcji wejścia-wyjścia, gdzie każde fizyczne urządzenie zewnętrzne jest odwzorowane w określone urządzenie logiczne, które może być wykorzystywane niezależnie i współbieżnie. Jest to warstwa pośrednicząca między oprogramowaniem użytkowym a sprzętem, która może być wykorzystywana w różnych dedykowanych systemach czasu rzeczywistego.

1. Wstęp

W systemach automatyki przemysłowej [2,4] istnieje zbiór urządzeń zewnętrznych, takich jak czujniki pomiarowe, mechanizmy wykonawcze itp., które są obsługiwane przez oprogramowanie użytkowe realizujące żądane funkcje. Zwykle oprogramowanie użytkowe jest tworzone z przeznaczeniem dla określonej konfiguracji sprzętowej. Każda, nawet drobna zmiana wymagań funkcjonalnych czy zamiana urządzeń zewnętrznych na podobne, wymaga wprowadzenia istotnych zmian w oprogramowaniu użytkowym lub wytworzenia całkowicie nowego oprogramowania. Powoduje to znaczne koszty nawet przy nieznacznej modyfikacji systemu.

W pracy przedstawiona jest koncepcja interfejsu przekształcającego zbiór urządzeń fizycznych w abstrakcyjny zbiór urządzeń logicznych. Oprogramowanie użytkowe komunikuje się wyłącznie z urządzeniami logicznymi, które stanowią odwzorowanie maksymalnej rzeczywistej konfiguracji. Urządzenia fizyczne pracujące w różnych standardach sprzętowych [3] są dołączone do systemu mikrokomputerowego za pomocą specjalizowanych układów we/wy (np. 8255, 8251). Układy te mogą być w przypadku wielu różnych urządzeń takie same. Przyporządkowanie urządzeń logicznych urządzeniom fizycznym odbywa się w trakcie wykonania programu, w fazie jego inicjalizacji. Proces ten obejmuje sprawdzenie obecności i

poprawności działania układów we/wy. W dalszej kolejności dokonywane jest konfigurowanie interfejsu polegające na określeniu sposobu dołączenia poszczególnych urządzeń fizycznych do określonych układów we/wy. Następnie dokonywane jest sprawdzenie poprawności działania urządzeń zewnętrznych i ostateczne przyporządkowanie ich urządzeniom logicznym.

W trakcie pracy systemu proces diagnostyczny wykonywany w tle może stwierdzić uszkodzenie jednego z urządzeń zewnętrznych. W takim przypadku możliwe jest zastąpienie uszkodzonego urządzenia przez rezerwowe, przy czym fakt ten nie będzie widoczny dla oprogramowania użytkowego. Zamiana taka odbywać się będzie wyłącznie na drodze programowej przez zmianę urządzenia zewnętrznego przyporządkowanemu danemu urządzeniu logicznemu.

Podsumowując, koncepcja interfejsu przedstawionego w pracy posiada następujące własności:

- możliwość samodzielnej identyfikacji dołączonych urządzeń zewnętrznych,
- prostota adaptacji w przypadku zmiany urządzeń zewnętrznych i modyfikacji wymagań funkcjonalnych,
- zwiększenie niezawodności systemu poprzez możliwość automatycznej rekonfiguracji interfejsu.

W pracy podano przykład wykorzystania omawianego interfejsu w cyfrowym systemie selektywnego wywołania [1]. Implementacja interfejsu dokonana została w języku C z niewielkimi wstawkami napisanymi w asemblerze. Stąd też możliwe jest jego łatwe przeniesienie na dowolny typ mikrokomputera.

2. Koncepcja interfejsu logicznego

Oprogramowanie komputerowych systemów automatyki składa się na ogół z kilku warstw, przy czym funkcje poziomu niższego stanowią podstawę do konstruowania funkcji poziomów wyższych. Proponowana warstwowa konfiguracja systemu jest pokazana na rys. 1. Warstwa oprogramowania użytkowego wspiera się na warstwach

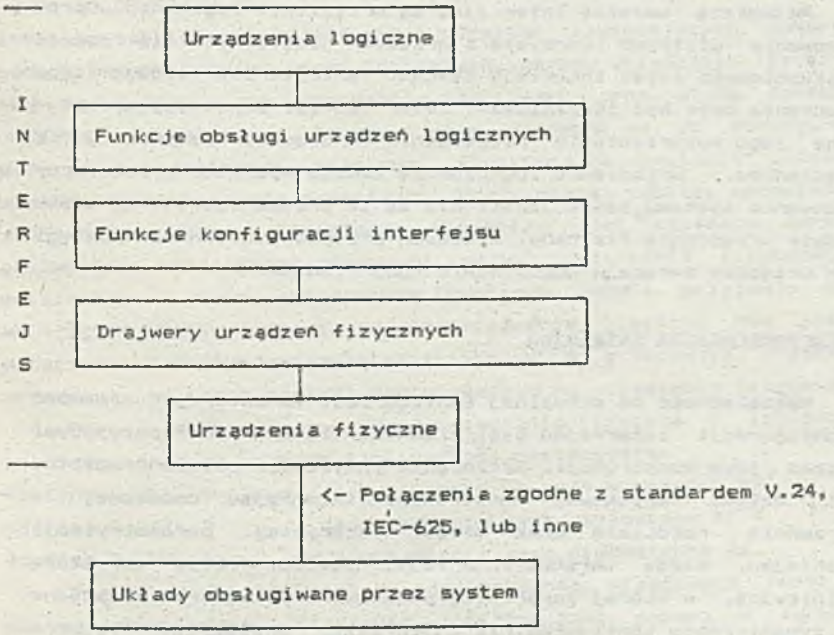
```
---------oprogramowanie użytkowe-----------

--------------interfejs--------------------

---jądro---   ---drajwery urządzeń---------

-------------------sprzęt------------------
```

Rys.1. Warstwy oprogramowania komputerowego systemu automatyki
Fig.1. Software layers in a computized automatic control system

niższych oraz na sprzęcie. Warstwa interfejsu całkowicie uniezależnia oprogramowanie użytkowe od rzeczywistej konfiguracji systemu. Warstwa ta definiuje pewnego rodzaju "urządzenia logiczne" o cechach na tyle abstrakcyjnych, iż mogą być one konkretyzowane dla prawie dowolnej konfiguracji. W odróżnieniu od warstw niższych (jądro i drajwery urządzeń), gdzie sposób komunikacji zależy od specyfiki danego urządzenia, interfejs ustala metodę przekazywania i otrzymywania informacji do/z urządzeń zgodną raczej z charakterem przekazywanej informacji.

Sterowanie w czasie rzeczywistym wymaga zapewnienia współużywalności zasobów systemu. Warunek ten może być spełniony przez wprowadzenie mechanizmu dzierżawienia i zwalniania urządzeń logicznych. Takie funkcje przewidziano dla każdego urządzenia tego typu. Dowolny proces może korzystać z urządzenia tylko po jego uprzednim wydzierżawieniu. Wydzierżawienie wyklucza korzystanie z tego urządzenia przez inne procesy. Funkcja zwolnienia urządzenia przekazuje je do dyspozycji innych procesów.

Proponowana struktura interfejsu pokazana jest na rys. 2.



Rys. 2. Organizacja proponowanego interfejsu logicznego
Fig.2. Structure of the proposed logical interface

Najniższą warstwę interfejsu stanowi zbiór urządzeń fizycznych.

Przez urządzenia fizyczne rozumiemy układy dołączone bezpośrednio do systemu mikrokomputerowego i pośredniczące w komunikacji z układami obsługiwanymi przez system. Urządzenia fizyczne mogą pracować w jednym z rozpowszechnionych standardów, np. V.24, IEC-625. Drajwery urządzeń fizycznych przeznaczone są do bezpośredniej ich obsługi. Są one ściśle związane z rodzajem urządzenia fizyczego i standardem jego pracy oraz niezależne od jego aktualnego adresu (adres urządzenia fizycznego jest podawany jako parametr). Każde odwołanie do urządzenia fizycznego odbywa się za pośrednictwem odpowiedniego drajwera, dlatego też drajwery powinny być napisane ze szczególnym uwzględnieniem efektywności tej komunikacji.

Kolejną warstwę interfejsu stanowią funkcje konfiguracyjne realizujące odwzorowanie urządzeń fizycznych w urządzenia logiczne przez modyfikację zawartości odpowiednich tablic konfiguracyjnych. Odwzorowanie to może być dokonywane podczas inicjalizacji systemu (np. po włączeniu zasilania) oraz w czasie normalnej pracy systemu przez czuwający w tle proces diagnostyczny lub na każde żądanie operatora systemu.

Najwyższą warstwą interfejsu są urządzenia logiczne. Oprogramowanie użytkowe korzysta z urządzeń logicznych za pomocą udostępnionego przez interfejs zestawu funkcji. Dla każdego typu urządzenia może być zdefiniowany zbiór funkcji zapewniający optymalne jego wykorzystanie niezależnie od sposobu komunikacji z urządzeniem. Urządzenia logiczne są zawsze widziane przez oprogramowanie systemu, nawet jeżeli nie są im przyporządkowane odpowiednie urządzenia fizyczne. W takich przypadkach funkcje obsługi tych urządzeń zwracają odpowiednie kody błędów.

3. Parametryzacja intefejsu

Niezależność od aktualnej konfiguracji sprzętowej i łatwość rekonfiguracji interfejsu były głównymi założeniami przyjętymi podczas jego konstrukcji. Spełnienie powyższych założeń mozliwe jest dzięki warstwowej konstrukcji interfejsu omówionej w poprzednim rozdziale oraz dzięki szerokiej parametryzacji interfejsu. Każda warstwa interfejsu posiada swoją strukturę definicyjną, w której zawarte są podstawowe parametry niezbędne do dynamicznego konfigurowania interfejsu. Urządzenia fizyczne opisane są za pomocą poniższej tablicy:

```
struct
 {int  NAME;      /* Nazwa urządzenia fizycznego */
  long ADR_BEG;   /* Adres bazowy urządzenia fizycznego */
  long ADR_END;   /* Adres końcowy urządzenia fizycznego */
```

```
    int  STATE;     /* Stan urządzenia fizycznego */
  } MCDEV[];
```

Jest to tablica składająca się z struktur zawierających numery będące kodami nazw urządzeń fizycznych, adresy danych urządzeń fizycznych oraz stan danego urządzenia fizycznego. Adresy urządzeń fizycznych wpisywane są podczas generacji aktualnej wersji oprogramowania, natomiast stan urządzeń wpisywany jest do struktury podczas inicjalizacji interfejsu. Struktura ta umożliwia określenie, które z urządzeń są sprawne i prawidłowo podłączone. Każdy wiersz tablicy odpowiada jednemu urządzeniu fizyczhemu. Co więcej, w tablicy jest zapisany adres bazowy i końcowy danego urządzenia fizycznego lub wartość specjalna oznaczająca brak urządzenia M_NODEV.

Stan urządzeń logicznych zdefiniowany jest w następującej tablicy:

```
    struct
     {int NAME;     /* Nazwa urządzenia logicznego */
      int ADR_DEV; /* Adres bazowy urządzenia fizycznego */
      int STATE;    /* Stan urządzenia logicznego */
     } LDEV[];
```

Jest to tablica składająca się z struktur zawierających numery będące kodami nazw urządzeń logicznych, adresu urządzenia fizycznego odpowiadającego urządzeniu logicznemu oraz stanu danego urządzenia logicznego. Adresy i stan wpisywane są do struktury podczas inicjalizacji interfejsu. Struktura ta umożliwia budowę samokonfigurowania się interfejsu. Każdy wiersz tablicy odpowiada jednemu urządzeniu logicznemu. W tablicy jest zapisany adres bazowy (w przestrzeni adresowej we/wy) urządzenia fizycznego związanego z danym urządzeniem logicznym. Jeżeli przyjmuje on wartość D_NODEV, oznacza to, że urządzenie logiczne nie jest połączone z żadnym urządzeniem fizycznym. W przeciwnym wypadku urządzenie logiczne posiada odpowiadające mu urządzenie fizyczne.

Stany, jakie mogą przyjmować urządzenia fizyczne i logiczne zdefiniowane w powyższych tablicach, są następujące:

```
#define D_OK   0          /* urządzenie pracuje poprawnie */
#define D_FAIL 0          /* urządzenie jest uszkodzone */
#define D_UNA  0          /* urządzenie jest niedostępne */
```

Struktura umożliwiająca udostępnianie właściwych funkcji obsługujących dane urządzenie logiczne jest podana w formie natępującej tablicy:

```
    struct
     {int (*F_rxctrl_getc)();       /* Odczyt znaku */
      int (*F_rxctrl_putc)();       /* Zapis znaku */
```

```
.
.
.
.
} FUNCT;
```

Zawiera ona adresy procedur obsługujących poszczególne funkcje urządzeń logicznych. Podczas inicjalizacji wypełniana jest adresem funkcji iof_null() (nic nie rób), a następnie adresami właściwych procedur. Umożliwia to dynamiczne konfigurowanie systemu. Podane w deklaracji nazwy procedur obsługujących poszczególne funkcje urządzeń logicznych związane są makrodefinicjami z funkcjami widzianymi przez użytkownika, np:

```
#define rxctrl_getc() (*F_rxctrl_getc)()
#define rxctrl_putc() (*F_rxctrl_putc)()
```

Poniżej podano specyfikację przykładowych funkcji odczytu i zapisu znaku z/do urządzenia odbiorczego [1]:

```
/*****************************************************************/
/* Funkcja: rxctrl_getc(uchar *ch,int wait_time)                 */
/* Odczyt znaku sterowania z urządzenia odbiorczego              */
/* We: *ch : wskaźnik odebranego znaku                           */
/*      wait_time : max czas  oczekiwania na  odczyt znaku       */
/* Wy: E_OK (>=0) : czas jaki pozostał do momentu upływu czasu */
/*                  wait_time, *ch -> odebrany znak              */
/*     E_TIME     : błędna wartość podanego czasu                */
/*     E_TIMEOUT  : przekroczenie czasu                          */
/*     E_NODEV    : urządzenie niedostępne                       */
/*     E_ERROR    : błąd przy odbiorze znaku                     */
/*     E_PHDEV    : brak urządzenia fizycznego                   */
/*****************************************************************/
/* Funkcja: rxctrl_putc(uchar ch,int wait_time)                  */
/* Zapis znaku sterowania do urządzenia odbiorczego              */
/* We: ch : znak przeznaczony do wysłania                        */
/*     wait_time :  maksymalny czas  oczekiwania                 */
/* Wy: E_OK (>=0) : czas, jaki pozostał do momentu upływu czasu */
/*                  wait_time                                    */
/*     E_TIME     : błędna wartość podanego czasu                */
/*     E_TIMEOUT  : przekroczenie czasu                          */
/*     E_NODEV    : urządzenie niedostępne                       */
/*     E_PHDEV    : brak urządzenia fizycznego                   */
/*****************************************************************/
```

Oprogramowanie użytkowe dla komunikacji z otoczeniem zewnętrznym wykorzystuje jedynie tego typu funkcje. Dzięki temu

zmiana konfiguracji czy typu urządzenia nie jest widoczna na tym poziomie.

4. Przykład zastosowań interfejsu logicznego

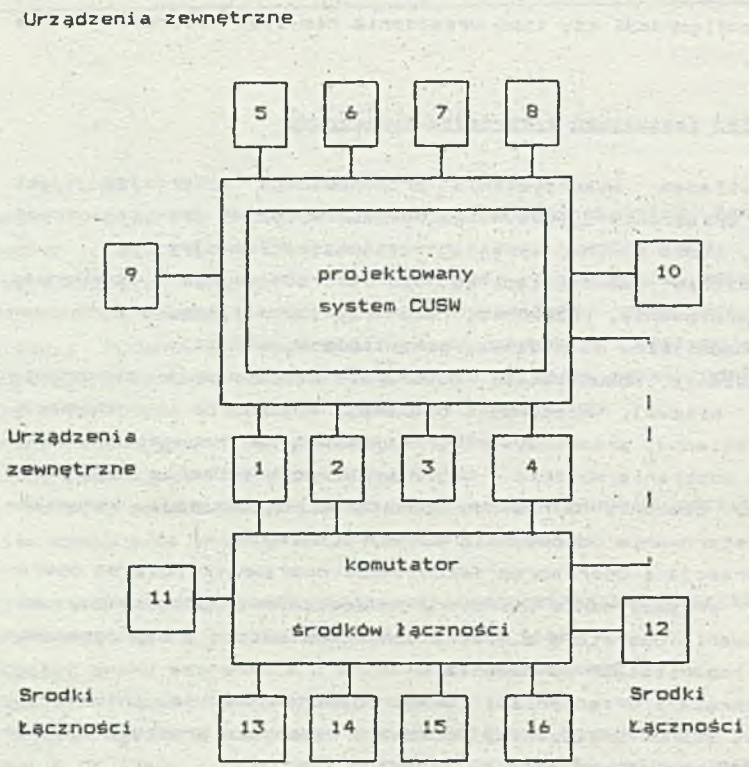
Przykładem wykorzystania proponowanego interfejsu jest cyfrowe urządzenie selektywnego wywołania (CUSW) dla zastosowań morskich, które powinno spełniać następujące funkcje:

1. Zarządzanie komunikatami do nadania (redagowanie i korygowanie, drukowanie, kodowanie, nadawanie komunikatów, programowanie komunikatów do nadania, ponawianie wysyłania),
2. Zarządzanie komunikatami odebranymi (zapewnienie synchronizacji bitowej, bajtowej i blokowej, analiza i dekodowanie, wyświetlanie, przechowywanie, drukowanie, alarmowanie w przypadku odebrania wołania o zagrożeniu, potwierdzanie jego przyjęcia, czekanie na reakcję operatora, retransmisja komunikatów, sterowanie odpowiedzią automatyczną),
3. Konwersacja z operatorem (akceptacja poprawnych poleceń operatora, przywoływanie operatora po zestawieniu połączenia, informowanie operatora o odebranych komunikatach i zaprogramowanych komunikatach do nadania).
4. Współpraca z urządzeniami zewnętrznymi (konwersja informacji, odczyt stanów urządzenia, polecenie wykonania prostych operacji, jak np. start, stop),
5. Rekonfiguracja systemu (inicjalizacja - restart, testowanie urządzenia CSW, włączanie i wyłączanie urządzeń logicznych).
6. Zmiana trybu pracy (ręczna, automatyczna.).

Na rys.3 przedstawiono schemat blokowy takiego urządzenia w maksymalnej konfiguracji. Urządzenie takie charakteryzuje się

Oznaczenia do rys. 3:

1. Nadajnik komunikacyjny HF/MF
2. Odbiornik łączności HF/FM
3. Odbiornik nasłuchowy HF/MF
4. Radiotelefon zawierający:
   - nadajnik komunikacyjny VHF/UHF
   - odbiornik nasłuchowy VHF/UHF (70 kanał)
   - odbiornik łączności VHF/UHF
5. Konsola operatorska 6. Konsola mostkowa
7. Urządzenie nawigacyjne 8. Drukarka
9. Urządzenie sygnału akustycznego 10. Układy zasilania
11. Modem 12. Telex 13. Dalekopis
14. Magnetofon 15. Klucz Morsa 16. Faksymile



Rys. 3. Schemat blokowy systemu nawiązywania łączności na statku
Fig.3. Block diagram of the communication establishing system on a ship

znaczną różnorodnością w zakresie wyposażenia sprzętowego, szczególnie urządzeń zewnętrznych. Konkretny zestaw typów urządzeń dołączonych do systemu CUSW tworzy jego konfigurację. Dowolną konfigurację opisać można jako następujący ciąg symboli:

(x1, x2, x3, x4, x5),

gdzie:

x1 = A - stacja zautomatyzowana, N - stacja niezautomatyzowana,

x2 = S - stacja statkowa, B - stacja brzegowa,

x3 = V - stacja o zakresie VHF, M - stacja o zakresie MF/HF,

x4 = Z - zespolone urządzenia radiokomunikacyjne,

R - rozdzielone urządzenia radiokomunikacyjne,

x5 = W - sprzęt tylko dla wywołania w niebezpieczeństwie,

Ł - sprzęt dla łączności rutynowej.

Teoretycznie wyróżnić można co najmniej 32 różnych konfiguracji urządzenia CUSW. Praktycznie rozpatruje się następujące konfiguracje: 1) ASMZX 2) ASMRX 3) NSMRX 4) ASVZX 5) ABMZX 6) ABVZX 7) XBMXW 8) XBVXŁ. Niewystępowanie danej cechy w zadanej konfiguracji oznaczono przez X. Np. Konfiguracja ABMZX oznacza urządzenie CUSW pracujące na zautomatyzowanej stacji brzegowej w zakresie MF/HF ze zespolonymi odbiornikami łączności.

Miejsce wykorzystania CUSW (na stacji brzegowej bądź statkowej), przyjęta konfiguracja systemu czy różne warianty pracy urządzenia powodują, że powyższe funkcje użytkowe mogą być realizowane w różny sposób. Co oznacza, że zakres wykorzystanych operacji i danych może być inny. Istotne staje się wyłonienie elementów wspólnych, które mogą stanowić bazę przy opracowywaniu oprogramowania dla konkretnej konfiguracji. W tym celu przeanalizowano opisane powyżej 8 różnych konfiguracji. Wyniki tej analizy

Tabela 1. Różnorodność wykonywanych funkcji w zależności od konfiguracji

konfiguracja / funkcje	1	2	3	4	5	6	7	8
1	A	A	A	A	A	A	A	A
2	B	B	B	B	C	C	C	C
3	D	D	D	D	E	E	F	F
4	G	H	I	K	L	M	N	P
5	R	R	R	R	R	R	R	R
6	S	S	S	T	S	S	S	S

przedstawiono w tabeli 1. Kolejnymi literami alfabetu (nie mylić z opisem konfiguracji) oznaczono zauważone różnice przy implementacji danej z analizowanych funkcji dla każdej z konfiguracji. I tak gospodarka komunikatami nadawanymi jest w zasadzie jednakowa (stąd wszędzie A). Gospodarka komunikatami odebranymi różni się zdecydowanie dla stacji statkowej (B)i stacji brzegowej (C). Podobne różnice występują dla funkcji konwersacji z operatorem; przy czym dla stacji brzegowej wyróżnić możemy dwa alternatywne podejścia: występuje tylko jedna konsola (E) lub w miejsce konsoli 2 istnieje połączenie z komputerem nadrzędnym. Z uwagi na różnorodność urządzeń zewnętrznych (dotyczy to przede wszystkim urządzeń nadawczo-odbiorczych zasady współpracy z urządzeniem CSW) mogą być różne dla każdej konfiguracji (litery od G do P). Z tabeli 1 wynika, że niektóre funkcje użytkowe powinny być opracowywane w różnych wariantach. Niemniej dzięki wykorzystywaniu

omawianego interfejsu logicznego liczba takich wariantów jest wyraźnie zmniejszona.

5. Wnioski

Przedstawiony interfejs został tak zaprojektowany, aby możliwa była jego prosta adaptacja przy ewentualnych zmianach wymagań funkcjonalnych lub wykorzystaniu innych rozwiązań konstrukcyjnych. Struktura interfejsu i zestaw jego funkcji jest również niezależna od fizycznej konfiguracji systemu. Dzięki temu możliwe jest wprowadzenie zmian w przypadku modyfikacji funkcji wykonywanych na obiekcie lub rodzaju urządzeń z nim współpracujących. Jeżeli urządzenie logiczne straci powiązanie z danym urządzeniem fizycznym, to proces diagnostyczny pracujący w tle wykryje taką sytuację i podejmie odpowiednią decyzję rekonfiguracyjną. Poza tym proponowana koncepcja obsługi urządzeń fizycznych zapewnia wyłączne użytkowanie danego urządzenia przez jeden proces, zabezpiecza przed nadmiernie długim czasem oczekiwania na przydzielenie urządzenia (jest to ważne np. w wypadku jego uszkodzenia). Co więcej, straty czasowe związane z pracą interfejsu logicznego nie wydają się krytyczne dla zastosowań komputerowych systemów automatyki.

LITERATURA

[1] Bartkowski T.,Birn P.,Cofta P., Krawczyk H., Smoczyński M., Szpryngier P.: Specyfikacja oprogramowania urządzeń CSW. Opracowanie naukowo - badawcze, Instytut Informatyki PG, Gdańsk 1989.

[2] Niederliński A.: Systemy komputerowe automatyki przemysłowej, t1 - Sprzęt i oprogramowanie. WNT, Warszawa 1984.

[3] Nowakowski W. i inni: Systemy interfejsu w miernictwie. WKiŁ, Warszawa 1987.

[4] Orłowski H.: Komputerowe układy automatyki. WNT, Warszawa 1987.

Recenzent: Prof.dr h.inż.J.Błażewicz

Wpłynęło do Redakcji do 1990-04-30.

UNIVERSAL INTERFACE FOR REAL-TIME MICROCOMPUTER SYSTEMS

S u m m a r y

The interface as a set of some input-output functions is defined, where each physical peripherial device is mapped into logic one and can be used independetly and concurrently. The interface forms a layer which ports user applications to the hardware. It can be applied in different dedicated real-time systems.

УНИВЕРСАЛЬНЫЙ ИНТЕРФЕЙС ДЛЯ МИКРОКОМПЬЮТЕРНЫХ СИСТЕМ РЕАЛНОГО ВРЕМЕНИИ

Р е з ю м е

Интерфейс представлен как множество функций входа-выхода, причем каждое внешнее физическое устройство имеет свое отображение в определенном логическом устройстве, которое может быть использовано независимо и синхронно. Он является посредником между прикладным программным обеспечением и оборудованием и может быть использован в различных проблемно-ориентировочных системах реалного времени.