074

A

PRZEMYSŁOWY INSTYTUT AUTOMATYKI I POMIARÓW
MERA-PIAP
Al. Jerozolimskie 202 02-222 Warszawa Telefon 23-70-81

Ośrodek Automatyzacji Procesów Produkcji

Pracownia Robotyzacji Procesów Produkcji

Główny wykonawca mgr inż. Zbigniew Pilat

Wykonawcy mgr inż. Jacek Dunaj
Wacław Grzywała
mgr inż. Wojciech Harnik

Konsultant

Nr zlecenia RP-77.3
et.3.1

"Wykonanie programu sterującego dla robota RP-120".

Zadanie 3.1 pt. "Wykonanie i uruchomienie oprogramowania realizującego sterowanie typu MP".

Zleceniodawca CPBR 7.1

Pracę rozpoczęto dnia marzec 90r. zakończono dnia 30.06.90r.

Kierownik Pracowni	Z-ca Dyrektora d/s Automatyki i Pomiarów	Kierownik Ośrodka
mgr inż. Z. Pilat	doc. dr inż. T. Gałązka	dr inż. M. Wrzesień

Praca zawiera:

stron 12
rysunków
fotografii
tabel
tablic
załączników 1

Nr rejestr. 6474

Rozdzielnik - ilość egz:

Egz. 1 OAE
Egz. 2 OAR
Egz. 3 BOINTE
Egz. 4 OAP
Egz. 5 OAP
Egz. 6

Analiza deskryptorowa

ROBOTY PRZEMYSŁOWE, UKŁAD STEROWANIA, OPROGRAMOWANIE

Analiza dokumentacyjna

Wykonanie oprogramowania realizującego podstawowe funkcje robota m.in. ruch we współrzędnych wewnętrznych, obsługa wyjść dwustanowych, skoki programowe, programowania i wykonywanie instrukcji pozycjonowania typu MP - dla celów badań funkcjonalnych prototypu robota IRp-120.

Tytuły poprzednich sprawozdań

Nr rej. PIAP 5925 - Zad.1.1. "Opracowanie założeń".
Nr rej. PIAP 6158 - Zad.2.1. "Wykonanie projektu systemu".
Nr rej. PIAP 6336 - Zad.2.2. "Opracowanie i uruchomienie oprogramowania dla badań funkcjonalnych modelu".

338.45.62(69).002.1/.2 Roboty przemysłowe
UKD 681.3.06 programowanie

PIAP 41/88 10000

2

SPIS TRESCI:

1. WSTĘP.

Opracowane w tym etapie pracy oprogramowanie jest przeznaczone dla prototypu robota IRp-120. Manipulator, pod względem kinematyki (co jest istotne z punktu widzenia programu sterującego), nie różni się od modelu. Zasadniczej zmianie uległ natomiast układ sterowania. Zastosowano w nim, w odróżnieniu od modelu, inną jednostkę centralną - MM16 (z mikroprocesorem INTEL 8086 i koprocesorem arytmetycznym INTEL 8087), inny pakiet pamięci - ML16 (możliwość pomieszczenia całego programu), nowy blok kontroli - MW32 oraz nowy typ panelu (tzw. panel zmodernizowany [4]). Od strony użytkownika oprogramowanie zawiera wszystkie elementy zrealizowane w programie sterującym przeznaczonym dla badań funkcjonalnych modelu robota [7]. W pewnym zakresie zostało nawet rozszerzone (np. wprowadzenie instrukcji pozycjonowania typu MP z programowanym czasem ruchu).

Ze względu na zachowanie jednolitości systemu programowania robotów rodziny IRp przyjęto nazewnictwo instrukcji i funkcji takie samo jak w innych typach (np IRp-6/60) - jest to zmiana w stosunku do założeń projektowych. Problem nazw i symboli, czyli ogólnie mówiąc języka programowania dla robotów IRp, w tym także modelu o udźwigu 120 kg, powinien być rozwiązany kompleksowo po opracowaniu perspektywicznego panelu programowania (cel 66 CPBR 7.1).

2. KONCEPCJA REALIZACJI OPROGRAMOWANIA.

Konfiguracja sprzętowa układu sterowania zdeterminowała sposób realizacji oprogramowania. Najbardziej sensownym i logicznym rozwiązaniem było adaptowanie programu sterującego dla robotów IRp-6/60 [1] z identycznym (w części cyfrowej) układem sterowania. Przy takim podejściu wykonawcy musieli pokonać dwa zasadnicze problemy. Po pierwsze dotychczas istniejące roboty IRp były pięcioosiowe (model IRp-6 na torze jezdnym jest konstrukcją na tyle specyficzną, że niewiele z jego oprogramowania dało się wprost wykorzystać). Po drugie w dotychczasowych modelach (posiadały one sterowanie typu CP), na poziomie programu sterującego operowano pojęciem ruchu, pozycji przede wszystkim w kartezjańskim układzie współrzędnych. Robot ze sterowaniem MP z założenia niejako operuje tylko we współrzędnych wewnętrznych.

Pracę podzielono na dwie części:

1. Oprogramowanie jednostki centralnej MM16.

2. Oprogramowanie panelu programowania.

Drugą z nich wykonano w ramach umowy PIAP nr 48/90 (realizowana ze środków tematu 77.3). Powstałe tam oprogramowanie panelu zostało opracowane można powiedzieć z nadmiarem, tzn. uwzględniając wszystkie funkcje przewidziane dla robota IRp-120 (a więc także te związane ze sterowaniem CP oraz dodatkowo wprowadzone specjalnie dla tego typu robota). W przypadku gdy dla pełnej realizacji konkretnej funkcji potrzebna była bieżąca wymiana informacji z jednostką centralną, przygotowano mechanizmy, które w sposób szybki i naturalny pozwolą dokończyć implementację oprogramowania panelu po uzupełnieniu oprogramowania MM16. Opis prac związanych z panelem programowania zawarty jest w załączniku 1 i stanowi integralną część niniejszego opracowania. Wykonanie oprogramowania jednostki centralnej można podzielić na następujące zagadnienia:

1. Adaptacja oprogramowania IRp-6/60 do nowej konfiguracji robota.

2. Ominięcie fragmentów oprogramowania związanych z kartezjańskim układem współrzędnych (obliczanie modelu kinematyki robota) - procedury te zachowano celowo, aby je wykorzystać przy wprowadzaniu algorytmów sterowania CP dla robota IRP-120.

3. Zmiany związane z ruchem robota we współrzędnych wewnętrznych.

4. Wprowadzenie programowania instrukcji pozycjonowania typu MP z podawanym czasem lub prędkością ruchu i pozycją zapamiętywaną we współrzędnych wewnętrznych (instrukcja QUASILIN).

5. Wykonanie procedury realizującej instrukcję QUASILIN.

6. Zmiany w części programu dotyczącej komunikacji z panelam programowania.

3. OPROGRAMOWANIE JEDNOSTKI CENTRALNEJ MM16.

Całe oprogramowanie jest umieszczone w następujących dwóch katalogach:

- katalog \WH - procedury związane z komunikacją z panelem programowania i z operacjami na obszarze programów użytkowych,
- katalog \AU - fragmenty oprogramowania odpowiedzialne za inicjację systemu, obsługę panelu operacyjnego, panelu programowania i realizację ruchu.

Na obu katalogach oprogramowanie jest pogrupowane w podkatalogi. Układ drzewa katalogów i zawartość poszczególnych kont jest analogiczny jak w przypadku robotów IRp-6/60. Taką samą strukturę oprogramowania źródłowego zachowano na dyskietkach.

3.1. Wprowadzenie konfiguracji robota IRp-120

Określenie konfiguracji robota zostało zawarte w trzech zbiorach w katalogu AU\MASTER\INTMOV\:

1. Zbiór nagłówkowy intmov.h - stałej oznaczającej liczbę osi robota NUMBER_OF_AXES ustawiono na 6 oraz wprowadzono stałą A6 równą 5, która jest wykorzystywana przy odwoływaniu się do tablic opisujących osie robota (A1=0 dla osi pierwszej, A2=1 dla osi drugiej itd).
2. Zbiór nagłówkowy arm.h - ustalenie maksymalnej prędkości osi robota tak jak dla robota IRp-60 na wartość 10875 inkrementów/sek (stała MAX_INCREMENT = 87 - maksymalna wartość inkrementu na 8 ms) oraz:
   - dolnego ograniczenia zakresu ruchu (w inkrementach)

     LOWER_1 = 0
     LOWER_2 = 0
     LOWER_3 = 0
     LOWER_4 = 0
     LOWER_5 = 0
     LOWER_6 = 0

   - górnego ograniczenia zakresu ruchu (w inkrementach - wartości takie jak zmierzone dla modelu robota IRp-120)

     UPPER_1 = 49773
     UPPER_2 = 26833
     UPPER_3 = 26262
     UPPER_4 = 48354
     UPPER_5 = 29973
     UPPER_6 = 45494

- hardware'owej pozycji synchronizacji, wyznaczonej przez przelaczniki synchronizacji w poszczególnych osiach (w inkrementach - wartości takie jak zmierzone dla modelu robota IRp-120)

```
FI_SYNCHR = 24832
TETA_SYNCHR = 14646
ALFA_SYNCHR = 8543
V2_SYNCHR = 24177
T_SYNCHR = 13517
V1_SYNCHR = 22762
```

3. Procedura intmcons.c86 - wprowadzenie sześciowymiarowych tablic określających odpowiednio:

```
lower[6] = {      /* ograniczenie dolne dla */
    LOWER_1,      /* wspolrz. wewnetrzynch  */
    LOWER_2,
    LOWER_3,
    LOWER_4,
    LOWER_5,
    LOWER_6
}

upper[6] = {      /* ograniczenie gorne dla */
    UPPER_1,      /* wspolrzednych wewn.    */
    UPPER_2,
    UPPER_3,
    UPPER_4,
    UPPER_5,
    UPPER_6
}

synchrpos[6] = { /* pozycja synchronizacji */
    FI_SYNCHR,   /* hardwareówej          */
    TETA_SYNCHR,
    ALFA_SYNCHR,
    V2_SYNCHR,
    T_SYNCHR,
    V1_SYNCHR
}

default_geom_pos[6] = { /* defaultowe współrzędne       */
                        /* wewnętrzne położenia         */
                        /* synchronizacji geometrycznej */
                        /* przewidziane do wykorzystania */
                        /* przy wprowadzeniu sterowania CP */
    DEFAULT_GEOM_POS_1,
    DEFAULT_GEOM_POS_2,
    DEFAULT_GEOM_POS_3,
    DEFAULT_GEOM_POS_4,
    DEFAULT_GEOM_POS_5,
    DEFAULT_GEOM_POS_6
}
```

Po wykonaniu tych czynności przekompilowano wszystkie procedudry w katalogu, ponieważ korzystają one ze zbiorów nagłówkowych arm.h i intmov.h. Postacie skompilowane umieszczono w bibliotece katalogu - intmov.lib.

3.2. Ominięcie obliczeń kinematyki robota.

W programie wyjściowym usunięto oblicznie i uaktualnianie pozycji kartezjańskiej robota - wywołanie procedury \AU\MASTER\EXTMOV\actlzext.c86 w:

- AU\MASTER\INTRPRTR\INSTR\posq.c86 - rozpoczęcie pozycjonowania quasi-liniowego,
- AU\MASTER\EXTMOV\quasilin.c86 - wykonanie pozycjonowania quasiliniowego,
- AU\MASTER\INTMOV\eosynchr.c86 - zakończenie synchronizacji robota,
- AU\MASTER\OPPANEL\manuwork.c86 - przejście systemu w tryb pracy ręcznej.

Procedur realizujących pozycjonowanie z interpolacją nie modyfikowano. Obecna ich postać ułatwi wprowadzanie sterowania CP dla robota IRp-120. Dodatkowo usunięto charakterystyczną dla konstrukcji IRp-6/60 kontrolę wzajemnego położenia czwartej i piątej osi w procedurze AU\MASTER\INTMOV\inrange.c86 (sprawdzenie, czy nie ma groźby przekroczenia obszaru roboczego manipulatora).

3.3. Ruch robota we współrzędnych wewnętrznych.

Pierwszą czynnością było ustalenie adresów sterowników położenia poszczególnych osi. Są one zawarte w tablicy axis_control() w zbiorze hardcons w katalogu AU\MASTER\HARDWARE\. Aktualne przyporządkowanie dla robota IRp-120 jest następujące (po lewej stronie adresy w zapisie heksadecymalnym):

0EB02H - sterownik osi pierwszej - FI
0EB0AH - sterownik osi drugiej - TETA
0EB06H - sterownik osi trzeciej - ALFA
0EB16H - sterownik osi czwartej - V2
0EB0EH - sterownik osi piątej - T
0EB12H - sterownik osi szóstej - V1

Następnie oprogramowano ruch sześcioma osiami robota przy pomocy joystick'a. Realizuje go procedura joyint.c86 w katalogu AU\MASTER\JOYSTICK\. Wartościami wejściowymi procedury jest tablica joystick_deflection[] informująca o wychyleniu joystick'a (przekazywana jako parametr przy wywołaniu) oraz zmienna globalna select wskazująca, które osie mają być przemieszczane. Zmienna select może przyjmować dwie wartości:

- ARM - ruch trzech pierwszych osi (FI, TETA, ALFA)
- HAND - ruch trzech ostatnich osi (V2, T, V1)

Po ustaleniu, które osie są sterowane, obliczana jest dla nich pozycja zadana jako suma pozycji aktualnej (tablica position[]) i wartości wychylenia joystick'a. Pozycja zadana dla pozostałych osi równa jest pozycji aktualnej. Następnie wywoływana jest procedura sendabs.c86 (w AU\MASTER\INTMOV\) realizująca ruch we współrzędnych wewnętrznych do pozycji zadanej. Parametrem wywołania tej procedury jest (oprócz pozycji zadanej) także okres realizacji przyrostów położenia przez sterowniki osi. Został on ustalony obecnie na 8ms (stała PERIOD_8_MS). Ten sposób wywołania pozwoli na proste wprowadzenie mechanizmu skalowania joysticka poprzez zmianę okresu sterowników osi (możliwe jest wprowadzenie 16 lub 32ms, co odpowiada spowolnieniu ruchu odpowiednio do 50 lub 25% - por. załącznik 1).

3.4. Programowanie pozycjonowania quasiliniowego.

Na początku rozszerzono strukturę opisującą postać instrukcji pozycjonowania w obszarze programów użytkowych o wariant odpowiadający pozycjonowaniu quasiliniowemu z pamiętaniem pozycji we współrzędnych wewnętrznych. Struktura ta o nazwie POS_INSTR (w zbiorze AU\MASTER\INTRPRTR\instruct.h) wygląda obecnie następująco:

```
typedef struct {
    unsigned char code;      /* kod instrukcji          */
    unsigned number;         /* numer instrukcji        */
    struct {                 /* parametry przewidziane */
                             /* do wykorzystania przy  */
                             /* wprowadzaniu funkcji   */
                             /* adaptacyjnych          */
        unsigned supervision : 1;
        unsigned speed_ctrl : 1;
        unsigned contouring : 1;
        unsigned search : 2;
        unsigned time : 1;
        unsigned relative : 1;
        unsigned fine : 1;
    } a;

    unsigned velocity;       /* prędkość lub czas ruchu */
    union {
        INTPOS ip;           /* dla pozycjonowania quasiliniowego */
        INSPOS fp;           /* dla pozycjonowania linowego */
        ORIENT o;            /* dla zmiany orientacji */
        CIRCLE c;            /* dla pozycjonowania kołowego */
    } pos;  /* Pozycja docelowa */
} POS_INSTR;
```

gdzie:

```
typedef struct {
    long wewpoz [6];
} INTPOS;
```

zawiera pozycję we współrzędnych wewnętrznych. Następnie w procedurze wpisywania nowej instrukcji (WH\MMFILGS\storeins.p86) wprowadzono kontrolę typu programowanej instrukcji pozycjonowania. Jeśli jest to pozycjonowanie quasiliniowe to do ciała instrukcji wpisywana jest aktualna pozycja wewnętrzna robota. Wstawienie pozycji do ciała instrukcji realizuje specjalnie w tym celu napisana procedura getintrn.c86 (w AU\MASTER\). Aby podprogram ten, napisany w języku C, mógł być wywoływany z procedury napisanej w PL/M-86 (inna kolejność umieszczania parametrów na stosie) trzeba było dopisać procedurę interfejsową - getintr1.a86 (w AU\MASTER\).

3.5. Wykonanie pozycjonowania quasiliniowego.

Do realizacji pozycjonowania quasiliniowego adaptowano istniejące procedury AU\MASTER\INTRPRTR\INSTR\posq.c86 (rozpoczęcie instrukcji) i AU\MASTER\EXTMOV\quasilin.c86 (wykonanie ruchu). Niezbędne były istotne modyfikacje. W posq.c86 usunięto sprawdzenie, czy zdefiniowane jest narzędzie (program w obecnej wersji pojęciem narzędzia w ogóle nie posługuje się) oraz

wspomniane w 3.2. uaktualnianie pozycji kartezjańskiej. Procedura na początku odczytuje z ciała instrukcji zadaną pozycję i umieszcza ją w tablicy zad_poz[]. Tablica ta jest parametrem wywołania procedury quasilin.c86. Oprócz niej przekazywana jest także stała informująca, czy w instrukcji podany był czas ruchu, czy prędkość oraz adres, gdzie czas lub prędkość ruchu (przeliczone na jednostki używane w quasilin.c86) jest zapisana. W procedurze quasilin sprawdzane są na początku ograniczenia (prędkość, obszr roboczy manipulatora), a następnie obliczane są parametry makrointerpolacji na podstawie przekazanych w wywołaniu parametrów. Dalszy przebieg procedury jest taki sam jak w programie dla IRp-6/60 [1].

3.6. Komunikacja z panelem.

W związku z wprowadzeniem szóstej osi konieczna była modyfikacja przekazywania do panelu pozycji wewnętrznej robota. Przesyłka przekazująca tę informację ma teraz długość 29 bajtów i następującą postać

1 bajt ze znakiem dwukropka (:)	1
2 bajty z zakodowanego kodu przesyłki	2
2*2 bajty zakodowanej pozycji osi 1	4
2*2 bajty zakodowanej pozycji osi 2	4
2*2 bajty zakodowanej pozycji osi 3	4
2*2 bajty zakodowanej pozycji osi 4	4
2*2 bajty zakodowanej pozycji osi 5	4
2*2 bajty zakodowanej pozycji osi 6	4
2 bajty zakodowanej sumy kontrolnej	2
razem:	29 bajtów

Konieczna była również zmiana procedury przygotowującej tę przesyłkę (AU\MASTER\PROGPANL\getinter.c86). Obecnie umieszcza ona w ciele przesyłki pozycje sześciu osi.

4. ZAKOŃCZENIE.

Omawiane oprogramowanie realizowano etapowo w kolejności przedstawionej w p.3. Począwszy od wprowadzenia ruchu we współrzędnych wewnętrznych wszystkie dalsze działania były weryfikowane na prototypie robota w pamięci RAM. Końcowa postać została zapisana do pamięci EPROM i wraz z panelem (załącznik 1) wstępnie przetestowana z wynikiem pozytywnym. Ostatecznej weryfikacji oprogramowania dokonają użytkownicy podczas badań prototypu robota IRp-120. Wtedy też można będzie wykonać korektę ewentualnych, zauważonych niedociągnięć.

5. LITERATURA.

1. Dokumentacja programu sterującego robotów IRp-6/60.

2. Dokumentacja układu sterowania robotów IRp-6/60.

3. Dokumentacja układu sterowania robota IRp-120 (wyniki tematu 77.1).

4. Dokumentacja panelu programowania robotów IRp-6/60.

5. Wykonanie programu sterującego dla robota IRp-120. Zad.1.1. Opracowanie założeń. spraw. PIAP nr rej. 5925.

6. Wykonanie programu sterującego dla robota IRp-120. Zad.2.1. Wykonanie projektu systemu. spraw. PIAP nr rej. 6158.

7. Wykonanie programu sterującego dla robota IRp-120. Zad.2.2. Opracowanie i uruchomienie oprogramowania dla badań funkcjonalnych modelu. spraw. PIAP nr rej. 6336.

Załącznik 1 do sprawozdania merytorycznego

Przemysłowy Instytut Automatyki i Pomiarów MERA-PIAP

Al. Jerozolimskie 202 02-222 Warszawa

"Wykonanie oprogramowania panelu programowania dla robota IRp-120".

Umowa Nr 48/90 z dnia 1990.05.18

Kierownik pracy: mgr inż. Zbigniew Pilat

Wykonawcy: mgr inż. Wojciech Hernik
mgr inż. Jacek Dunaj
Wacław Grzymała
Jolanta Masalska

Lipiec 1990

S P I S T R E Ś C I

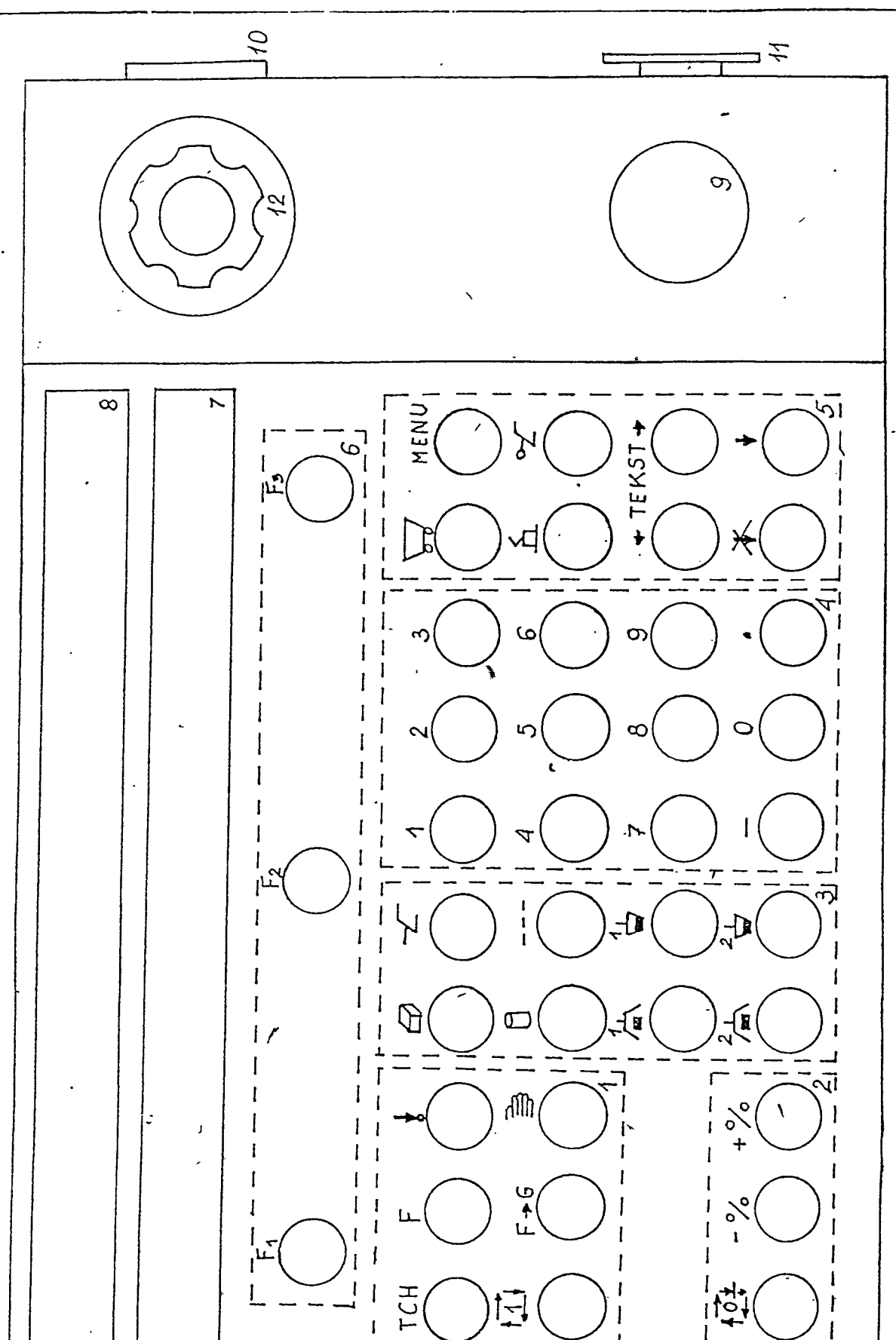
14

1. Wstęp

Program sterujący panelu programowania został napisany w języku asemblera Intel 8080 przy wykorzystaniu oprogramowania firmy Microtec Research Inc. dla komputera IBM PC.
Ogólna struktura programu jak i część procedur została zaadaptowana z panelu dla robotów IRp-6/60 [1].
W niniejszym opracowaniu szczególną uwagę zwrócono na funkcje nowe i modyfikowane. Program sterujący dla jednostki centralnej robota IRp-120 jest wykonywany etapowo. Najpierw powstaje wersja realizująca tylko sterowanie MP a następnie również CP. Przedstawione tu oprogramowanie panelu jest postacią docelową dla programu głównego obejmującego sterowanie CP. W wersji MP część funkcji panelu nie będzie po prostu wykorzystywana. Niektóre instrukcje być może trzeba będzie skorygować lub uzupełnić po zakończeniu prac nad jednostką centralną.

2. Panel programowania dla robota IRp-120.

Konstrukcyjnie panel bazuje na tzw. wersji zmodernizowanej panelu programowania dla robotów IRp-6/60.
W związku z wprowadzeniem dodatkowej grupy programowanych instrukcji /instrukcje technologiczne/ koniecznym okazało się zamontowanie nowego przycisku oznaczonego "TCH". Został on dołączony jako przycisk o numerze Ø w grupie 1 [2]. Wprowadzono także diodę świecącą związaną z tym przyciskiem dołączając ją jako D1 [2] . Zmiany te wiążą się z modyfikacją płyty czołowej panelu programowania dla IRp-120. Przedstawiono ją na rysunku nr 1.



RYS.1. Widok płyty czołowej panelu programowania robota IRp-120.

3. Funkcje panelu programowania.

Dodatkowy przycisk wiąże się z wprowadzeniem nowego stanu panelu programowania nazwanego "programowanie instrukcji technologicznych". W stanie tym zmienna MODE /określająca stan panelu/ przyjmuje wartość Ø. W panelu robota IRp-120 mamy więc w sumie osiem stanów.

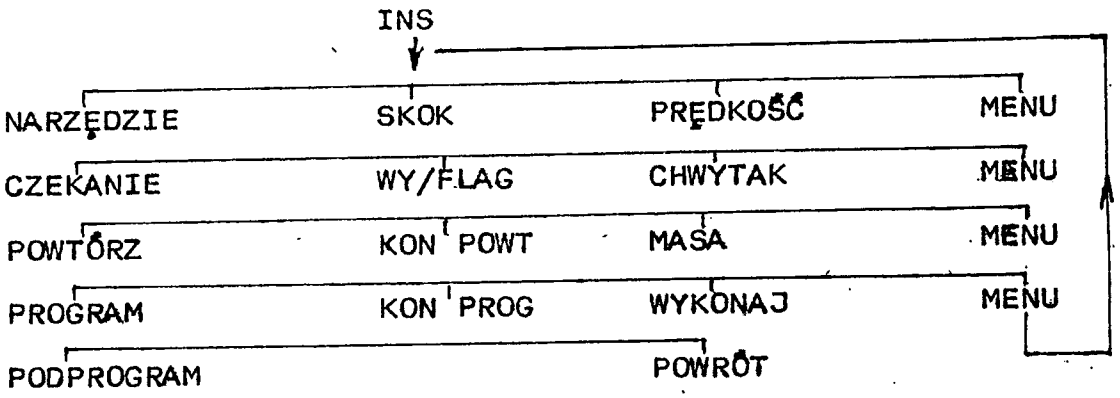
Tabela 1.

Mnemonik stanu	Wartość MODE	Procedura obsługi	Nazwa stanu
MODTCH	Ø	proces.a8Ø	programowanie instrukcji technologicznych
MODPOS	1	poz. a8Ø	programowanie instrukcji pozycjonowania
MODANI	2	ani.a8Ø	programowanie innych instrukcji
MODSTR	3	strt.a8Ø	start programu
MODEDI	4	edi.a8Ø	edycja programu
MODMAN	5	man.a8Ø	ręczne operowanie systemem
MODAUT	6	manto.a8Ø	praca automatyczna
MODERR	7	merr.a8Ø	błąd

W stanach 1,3,4,6,7 panel zachowuje się dokładnie tak samo jak panel dla robotów IRp-6/60. Poniżej zostaną opisane stany programowania innych instrukcji, ręcznego operowania systemem i nowy stan programowania instrukcji technologicznych.

3.1. Programowanie innych instrukcji /stan 2/.

Zmiany w tym stanie wiążą się z rozszerzeniem repertuaru instrukcji dla robota IRp-120 w stosunku do modeli IRp-6/60. Menu dla tego stanu ma w tej chwili pięć linii i wygląda następująco:



Poszczególne instrukcje realizują:

NARZĘDZIE - deklaracja narzędzia aktywnego

SKOK - skok warunkowy lub bezwarunkowy do instrukcji o numerze podanym jako parametr instrukcji SKOK

PRĘDKOŚĆ - deklaracja prędkości podstawowej i maksymalnej dla ruchu liniowego i kołowego

CZEKANIE - oczekiwanie na spełnienie warunku lub upływ czasu

WY/FLAG - ustawienie wyjścia lub flagi

CHWYTAK - wykonanie operacji zamknięcia /otwarcia jednego z dwóch chwytaków /numer chwytaka jest parametrem instrukcji/

POWTÓRZ - początek pętli programowej

KON POWT - koniec pętli programowej

te instrukcje były dostępne w robotach IRp-6/60 i działanie ich nie ulegało zmianie [1] .

Instrukcjami nowymi są:

MASA - deklaracja obciążenia, z jakim pracuje robot; wartość obciążenia określa parametr instrukcji i może on przyjmować jedną z trzech wartości: 1ØØ % ; 5Ø % ; Ø % . Domyślnie, na początku pracy robota przyjmowana jest wartość Ø % - robot nieobciążony

PROGRAM - deklaracja początku programu użytkowego - instrukcja bez parametrów

KON PROG - deklaracja końca programu użytkowego - instrukcja bez parametrów

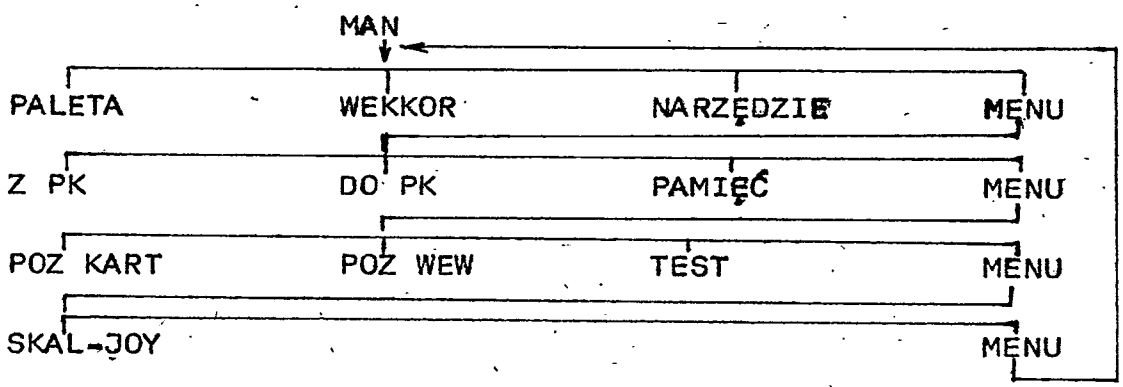
WYKONAJ - wywołanie podprogramu - parametrem instrukcji jest numer podprogramu, który ma być wykonany

PODPROGRAM - deklaracja początku podprogramu - parametrem instrukcji jest numer podprogramu

POWRÓT - deklaracja powrotu z podprogramu do miejsca wywołania /za instrukcją WYKONAJ/ - instrukcja bez parametrów

3.2. Ręczne operowanie systemem /stan 5/.

Menu tego stanu dla robota IRp-120 wygląda następująco:



Poszczególne funkcje realizują:

PALETA - definiowanie palety - wprowadzanie i usuwanie definicji palety a także odczyt parametrów palety z możliwością doprowadzenia robota do poszczególnych programowanych pozycji;

parametrami palety są:

- numer palety
- wymiar palety [M x N]

gdzie: M - liczba wierszy /rzędów/
N - liczba kolumn palety

- pozycja początku obsługi palety
- pozycja nad elementem [1, 1]
- pozycja nad elementem [1, N]
- pozycja nad elementem [M, 1]

Na podstawie trzech ostatnich parametrów program sterujący może obliczyć pozycję nad dowolnym elementem palety. Funkcja PALETA jest nowa, wprowadzona specjalnie dla robota IRp-120.

WEKKOR - definiowanie wektora korekcji - wykorzystywane będzie w kolejnych wersjach programu sterującego robota IRp-120 przy wprowadzeniu funkcji adaptacyjnych [1].

NARZĘDZIE - definiowanie narzędzia - wprowadzanie i zmiana parametrów opisujących narzędzie, usuwanie definicji narzędzia, wskazywanie systemowi narzędzia aktywnego /wykorzystywanego przy poruszaniu manipulatorem za pomocą joystick a we współrzędnych kartezjańskich lub cylindrycznych [1]

Z PK - odczyt programu użytkowego z pamięci kasetowej

DO PK - zapis programu użytkowego do pamięci kasetowej [1]

PAMIĘĆ - podanie operatorowi informacji o ilości wolnej pamięci w obszarze programów użytkowych [1]

POZ KART - odczyt pozycji robota w układzie kartezjańskim związanym z jego podstawą, podawane są trzy współrzędne opisujące położenie punktu roboczego narzędzia TCP: x, y, z oraz trzy kąty Enlera opisujące orientację osi narzędzia: mutacja, precesja, obrót. Funkcja POZ KART jest nowa, wprowadzona specjalnie dla robota IRp-120

POZ WEW - odczyt pozycji robota w układzie współrzędnych wewnętrznych - położenia poszczególnych osi FI, TETA, ALFA, V2, T, V1 wyrażone w inkrementach - elementarnych jednostkach położenia osi 1

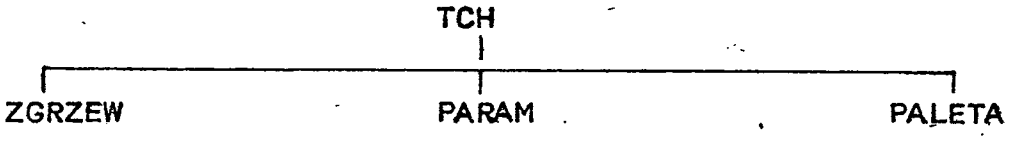
TEST - przejście do testowania panelu programowania

SKAL-JOY - skalowanie joystick'a - możliwość wybrania jednej z trzech wartości maksymalnej prędkości ruchu przy ręcznym operowaniu robotem we współrzędnych wewnętrznych i prędkości nominalnej albo 50 % lub 25% tej prędkości. Prędkość nominalna / 100 % / jest przyjmowana domyślnie przez system na początku pracy /po włączeniu robota/ i wynosi ona 2500 inkrementów /sek co stanowi ok. 23% maksymalnej, wynikającej z konstrukcji, prędkości robota. Funkcja SKAL-JOY jest nowa, wprowadzona specjalnie dla robota IRp-120

3.3. Programowanie instrukcji technologicznych /stan 0 /.

Jest to nowy stan panelu programowania wprowadzony dla robota IRp-120.

Umożliwia on programowanie instrukcji związanych z konkretnym procesem technologicznym. Menu dla tego stanu przedstawiono poniżej.



Poszczególne instrukcje pozwalają zaprogramować:

ZGRZEW - ruch robota do zaprogramowanego panelu z obsługą zgrzewarki w pozycji docelowej; ruch jest typu pozycjonowania Quasiliniowego /wszystkie osi jednocześnie rozpoczynają ruch i zatrzymują się wszystkie w tym samym momencie/, po dojściu w pobliże punktu zaprogramowanego / w tzw. strefę zerową/ układ sterowania robota wysyła do zgrzewarki rozkaz zaciśnięcia szczęk i wykonania zgrzewania /możliwość wyboru jednego z pięciu programów zgrzewania - wykorzystywa ne są do tego wyjścia dwustanowe robota/; wprowadzona jest kontrola: rozwarcie szczęk po zgrzewaniu /wykrycie ewentualnego przygrzania/, kolizji z otoczeniem i dodatkowego sygnału informacyjnego /możliwość dołączenia np. sprzęgła przeciążeniowego tak jak w modelu IRp-60Z/. Instrukcja ZGRZEW nie ma parametrów podawanych jawnie, korzysta z wartości zaprogramowanych w instrukcji PARAM.

PARAM - określenie parametrów z jakimi ma być wykonywana instrukcja ZGRZEW; parametrami tymi są: numer programu zgrzewania / od 1 do 5/, numer strefy zerowej, /przewidziano możliwość wyboru jednej z dwóch stref zerowych oznaczonych numerami 1, 2/ oraz prędkość ruchu podczas wykonywania instrukcji ZGRZEW /podawana w procentach maksymalnej prędkości konstrukcyjnej robota, jest więc liczbą z zakresu od Ø do 1ØØ % /

PALETA - instrukcja obsługi palety, jej wykonanie polega na dojściu robota do pozycji początku obsługi palety, a następnie nad element obsługiwany, tzn. robot wykonuje zadany podprogram i wraca do pozycji początku obsługi; jeżeli nie był to ostatni element przewidziany do obsługi, cykl jest powtarzany dla elementu kolejnego. Po obsłużeniu wszystkich elementów następuje przejście do kolejnej instrukcji programu. Parametrami instrukcji PALETA są: numer palety / od 1 do 9/, prędkość dojścia do pozycji początku obsługi /w procentach, od Ø do 1ØØ % /, prędkość dojścia nad element obsługiwany / od Ø do 1ØØ % /, numer podprogramu obsługi elementu palety, współrzędne pierwszego i ostatniego elementu obsługiwanego.

4. Realizacja oprogramowania.

4.1. Nowe przesyłki z panelu programowania do jednostki centralnej.

Rozbudowa funkcjonalna oprogramowania panelu wymagała rozbudowy komunikacji między panelem a jednostką centralną. W celu obsługi realizacji nowych funkcji panelu wprowadzono dodatkowe przesyłki do jednostki centralnej są to:

- przesyłka numer 57 o mnemoniku MSKALA - żądanie przesyłania informacji o aktualnej skali joysticka;
- przesyłka numer 58 o mnemoniku MPALDE - rozkaz usunięcie definicji palety o wskazanym numerze z pamięci systemu;
- przesyłka numer 59 o mnemoniku MPALST - rozkaz zapamiętania definicji palety o podanym numerze / w przesyłce jest zawarty wymiar palety/;
- przesyłka numer 60 o mnemoniku MSKAST - rozkaz zapamięta-

nia nowej skali joysticka.

Kody wszystkich przesyłek z panelu do jednostki centralnej znajdują się w zbiorze ptmcod.h zaś ich długość w zbiorze ptmlen.c86 /Załącznik nr 2/.

4.2. Nowe przesyłki z jednostki centralnej do panelu programowania.

W związku z wprowadzeniem mechnizmu paletyzacji oraz skalowania joysticka listę przesyłek z MM16 do panelu rozszerzono o:

- przesyłkę o numerze 24 o mnemoniku PSKALA - przesyłanie informacji o skali joysticka;
- przesyłkę numer 25 o mnemoniku PPALET - zawiera ona numer i parametry palety.

Należy zaznaczyć, że dzięki uniwersalności przesyłek do/z panelu programowania, związanych z programowaniem instrukcji, mogły one zostać wykorzystywane przy obsłudze programowania instrukcji dodatkowo wprowadzonych dla robota IRp-120. Pełną listę kodów przesyłek zawiera zbiór ptpcod.h zań ich długość zbiór ptplen.a80.

4.3. Modyfikacja zbioru wyświetlanych tekstów /text.a80/.

Ogólnie zmiany w zbiorze text.a80 można podzielić na dwie grupy:

1/ związane z wprowadzeniem nowego stanu panelu, nowych funkcji i instrukcji od strony systemu menu:
   - dodanie menu dla stanu "programowania instrukcji technologicznych" /MENTCH/;
   - ustalenie długości menu w poszczególnych stanach /MNLSET - liczba linii menu/;
   - ustalenie tekstów poszczególnych linii menu.

2/ związane ze zmianą repertuaru wyświetlanych tekstów:
   - teksty dla nowych instrukcji /ZGRZEW, PARAM, PALETA, PROGRAM itd. por.p. 3.1, 3.3/;

- teksty dla nowych funkcji /SKAL-JOY, PALETA, POZKART/;
- usunięci tekstów nieużywanych / np. w związku ze zmianą nazw osi/.

4.4. Obsługa stanu ręczne operowanie systemem /procedura man.a80/.

Sekwencje instrukcji realizujących obsługę poszczególnych funkcji zaczynają się od następujących adresów /MANADR/:

PALETA	-	MAN 36Ø
WEKKOR	-	MAN /funkcja nieobsługiwana w obecnej wersji programu sterującego IRp-129/
NARZĘDZIE	-	MAN 85
ZPK	-	MAN 1Ø
DO PK	-	MAN 23
PAMIĘĆ	-	MAN 16Ø
POZ KART	-	MAN 18Ø
POZ WEW	-	MAN 22Ø
TEST	-	MAN 17Ø
SKAL-JOY	-	MAN 4ØØ

Definiowanie palety i odczyt pozycji karteżjańskich są przeznaczone dla wersji programu ze sterowaniem CP i wraz z jej powstawaniem będą dokończone. Wprowadzenie możliwości odczytu pozyćji sześciu osi robota /POZ WEW - w IRp-6/60 odczytywano pięć osi/ wiązało się przede wszystkim ze zmianą długości przesyłki podającej współrzędne wewnętrzne. Została ona wydłużona o cztery bajty /tj. dwa bajty informacji [1]/ i zajmuje obecnie 29 bajtów /zbiór ptplen.a80/ w procedurze man.A80 po odczytaniu pozycji piątej osi, w górnej linii wyświetlacza pojawia się napis V1 = /TXT155/, a następnie dopisywana jest liczba zakodowana w czterech kolejnych bajtach przesyłki /dwóch bajtach informacyjnych/ - wywołanie podprogramu MAN 2ØØ.
Przy obsłudze skalowania joysticka program ustawia na początku stan obu linii wyświetlaczy. Następnie wysyła do MM16 rozkaz podania aktualnej skali.
Odczytana wartość jest dopisywana do górnej linii. Następnie

program czeka na wciśnięcie któregoś z przycisków funkcyjnych. Oznacza to podanie nowej wartości skali. Informacja o tym jest przesyłana do jednostki MM16.
Pozostałe funkcje są obsługiwane tak samo jak w robotach IRp-6/60.

4.5. Programowanie innych instrukcji /procedura osi a80/.

Podprogramy realizujące oprogramowanie poszczególnych instrukcji mają następujące adresy /ANIADR/:

NARZĘDZIE - ANI 3Ø
SKOK - ANI 25Ø
PRĘDKOŚĆ - ANI 45
CZEKANIE - ANI 235
WY/FLAG - ANI 13Ø
CHWYTAK - ANI 7Ø
POWTÓRZ - ANI 29Ø
KON POWT - ANI 3ØØ
MASA - ANI 33Ø
PROGRAM - ANI 34Ø
KON PROG - ANI 35Ø
WYKONAJ - ANI 36Ø
PODPROGRAM - ANI 37Ø
POWRÓT - ANI 38Ø

W obsłudze instrukcji dostępnych w IRp-6/60 nie wprowadzono żadnych zmian. Instrukcje PROGRAM, KONPROG i POWRÓT są bezparamentowe więc ich obsługa jest podobna. Do przesyłki wstawiany jest tylko kod instrukcji i inicjowana jest transmisja /dołączenie pozostałych elementów tj. dwukropka na początku i sumy kontrolnej na końcu organizuje procedura komunikacyjna/.
Instrukcje MASA, WYKONAJ i PODPROGRAM mają po jednym parametrze Przy programowaniu ich do komórki następnej za kodem wpisywana jest wartość odpowiedniego parametru /np. numer podprogramu/ i dopiero teraz jest inicjowana transmisja. Kody nowych instrukcji uzupełniono w zbiorze /define.h/.

4.6. Programowanie innych instrukcji.

Wprowadzenie nowego stanu i programowanie związanych z nim instrukcji wiązało się z istotnymi zmianami w wielu parametrach oprogramowania. Przede wszystkim należało wprowadzić rozpoznanie przycisku Ø w pierwszej grupie /main.a80/. Reakcją na jego wciśnięcie jest przejście panelu do stanu Ø /MODTCH w define.h/. Zewnętrznym objawem dla operatora jest pojawienie się odpowiedniego menu /menmax.a80 i getime.a80/ i zaświecenie didy przy przycisku TCH /disp.a80/, /main.a80/. Wewnętrznie program przechodzi do procedury obsługi stanu MODTCH - PROCES /proces.a80/. Umożliwia ona zaprogramowanie jednej z trzech instrukcji:

```
ZGRZEW  -  PROC 1Ø
PARAM   -  PROC 2Ø
PALETA  -  PROC 3Ø
```

Wyjście z procedury - na ogólnych zasadach [1] , tj. po naciśnięciu innego przycisku funkcyjnego. Kody nowych instrukcji umieszczono w zbiorze define.h.

5. Zakończenie

Przedstawione oprogramowanie zostało przygotowane a następnie sprawdzone /na pamięci RAM i EPROM/ we współpracy z robotem IRp-120 z oprogramowaniem realizującym sterowanie typu MP. Dla części funkcji przewidzianych dla wersji CP, przygotowano miejsce, mechanizmy umożliwiające łatwe ich dokończenie w pozostałym zakresie.
Istotnym plusem jest to, że panel z takim oprogramowaniem będzie mógł pracować z oboma typami robota /z MP i CP/.
Po prostu w ewrsji MP jego funkcje będą wykorzystane nie w pałnym zakresie.

6. Literatura

[1] Dokumentacja programu sterującego IRp-6/60.

[2] Dokumentacja panelu programowania robotów IRp-6/60.

6. Literatura

[1] Dokumentacja programu sterującego IRp-6/60.

[2] Dokumentacja panelu programowania robotów IRp-6/60.