Politechnika Wrocławska

WYDZIAŁ ELEKTRYCZNY

INSTYTUT MASZYN, NAPĘDÓW I POMIARÓW ELEKTRYCZNYCH

Laboratorium Napędu robotów

WPR

Roboty przemysłowe – podstawowe pojęcia, środowisko COSIMIR, język programowania MELFA BASIC IV

Spis treści

1 Wprowadzenie

W dzisiejszych czasach koszty, jakość oraz elastyczność reagowania na zmienne potrzeby odbiorcy decydują o rynkowej konkurencyjności producenta. Zrobotyzowane systemy produkcyjne pozwalają na efektywne wykorzystanie kwalifikacji i pracy ludzi oraz posiadanego parku maszynowego, ochronę środowiska, poprawę jakości i ilości produkcji. Przeznaczone są one zarówno dla małych, jak i średnich producentów. Roboty są w stanie przenosić ciężkie elementy, wykonywać precyzyjne prace spawalnicze lub montażowe, wspomagając lub wyręczając człowieka w wielu trudnych, męczących i monotonnych zadaniach, pracując dokładniej bez przerw, w hałasie, zapyleniu czy też w niezdrowych oparach. Roboty pracują w najróżniejszych dziedzinach przemysłu, między innymi: w metalowym, samochodowym, ciężkim, stoczniowym, budowy maszyn i taboru kolejowego, a także spożywczym i chemicznym. Roboty wykorzystywane są w odlewnictwie, meblarstwie oraz przy produkcji między innymi artykułów gospodarstwa domowego i ceramiki sanitarnej. Na tak dużą różnorodność zastosowań pozwala duża elastyczność robota oraz jego oprogramowanie.

Zastosowanie robotów w produkcji powoduje między innymi:

- poprawę poziomu jakości produkcji,
- zwiększenie wydajności - obniżkę kosztów produkcji,
- humanizację miejsca pracy,
- eliminację błędów operatora - mniej braków,
- dokładną kontrolę procesu produkcyjnego,
- stabilizację produkcji i zmniejszenie przestojów,
- wzrost elastyczności produkcji,
- ochronę środowiska.

2 Podstawowe określenia i podział robotów

Podstawowe definicje związane z robotami przemysłowymi (wg normy ISO 8373).

Manipulacja - tok czynności w przemysłowym procesie produkcyjnym, polegający na: uchwyceniu określonego obiektu manipulacji, transportowaniu, pozycjonowaniu lub orientowaniu tego obiektu względem przyjętej bazy, oraz przygotowujący ten obiekt do wykonywania na nim lub za jego pomocą operacji technologicznych.

Manipulator (przemysłowy) - urządzenie przeznaczone do wspomagania lub całkowitego zastąpienia człowieka przy wykonywaniu czynności manipulacyjnych w przemysłowym procesie produkcyjnym, sterowane ręcznie lub automatycznie za pomocą własnego układu sterującego stałoprogramowanego lub zewnętrznego układu sterującego.

Robot (przemysłowy) - urządzenie automatyczne przeznaczone do wykonywania czynności manipulacyjnych w przemysłowym procesie produkcyjnym, mające układ ruchu składający się co najmniej z trzech zespołów ruchu i własny układ sterujący programowalny.

Manipulator (przemysłowy) ręczny - manipulator przemysłowy z napędem ręcznym, składającym się najczęściej z układu ruchu i układu sterującego sterowanego ręcznie, przy czym układ ruchu odtwarza wymuszone przez operatora ruchy układu sterowania z odpowiednim przełożeniem.

Serwomanipulator - manipulator przemysłowy sterowany ręcznie bezpośrednio przez operatora, w którym do wprowadzenia w ruch członów układu kinematycznego wykorzystuje się serwomechanizmy odtwarzające lub wspomagające ruchy operatora. Telemanipulator - manipulator przemysłowy sterowany ręcznie za pośrednictwem sygnałów przekazywanych przewodami lub bezprzewodowo.

Manipulator (przemysłowy) automatyczny - manipulator przemysłowy wyposażony w układ sterujący automatyczny stałoprogramowy lub nie mający własnego układu sterowania, lecz przystosowany do współpracy z zewnętrznym układem sterującym automatycznym.

Robot (manipulator) przemysłowy stacjonarny - robot (manipulator) przemysłowy, który nie ma możliwości przemieszczania się względem podłoża.

Robot (manipulator) przemysłowy mobilny - robot (manipulator) przemysłowy, który może przemieszczać się względem podłoża lub może przemieszczać ramię względem podstawy. Układ ruchu robota (manipulatora) przemysłowego mobilnego składa się z ramienia lub kiści oraz co najmniej jednego zespołu ruchu globalnego służącego do przemieszczania ramienia względem podstawy lub pojazdu mechanicznego służącego do przemieszczenia robota względem podłoża.

Robot (manipulator) przemysłowy monolityczny - robot (manipulator) przemysłowy, w którym co najmniej zespoły ruchu tworzące ramię stanowią konstrukcyjną całość o niezmiennej strukturze kinematycznej.

Robot (manipulator) przemysłowy modułowy - robot (manipulator) przemysłowy, którego układ ruchu jest złożony z modułów ruchu.

Robot (manipulator) przemysłowy konsolowy - robot (manipulator) przemysłowy stacjonarny, którego podstawa albo korpus jest przystosowany do mocowania do maszyny.

Robot (manipulator) przemysłowy portalowy - robot (manipulator) przemysłowy, którego ramię przymocowane jest do ściany, sufitu albo suwnicy, dzięki czemu nie zajmuje miejsca na podłodze lub maszynie.

Robot (manipulator) przemysłowy suwnicowy - robot (manipulator) przemysłowy mobilny, którego zespół ruchu globalnego ma postać suwnicy.

Robot (manipulator) przemysłowy bramowy - robot (manipulator) przemysłowy mobilny, którego zespoły ruchu globalnego tworzą suwnicę bramową.

Robot (manipulator) przemysłowy kartezjański - robot (manipulator) przemysłowy, którego ramię zbudowane z zespołów ruchu liniowego realizuje ruchy odpowiadające zmianom współrzędnych układu prostokątnego.

Robot (manipulator) przemysłowy cylindryczny - robot (manipulator) przemysłowy, którego ramię zbudowane z dwóch zespołów ruchu liniowego i jednego zespołu ruchu obrotowego realizuje ruchy odpowiadające zmianom współrzędnych układu cylindrycznego.

Robot (manipulator) przemysłowy sferyczny - robot (manipulator) przemysłowy, którego ramię zbudowane z dwóch zespołów ruchu obrotowego i jednego zespołu ruchu liniowego realizuje ruchy odpowiadające zmianom współrzędnych układu sferycznego.

Robot (manipulator) przemysłowy przegubowy - robot (manipulator) przemysłowy antropomorficzny, którego ramię jest zbudowane tylko z zespołów ruchu obrotowego.

Robot (manipulator) przemysłowy typu SCARA - robot (manipulator) przemysłowy przeznaczony do manipulacji na płaszczyźnie, którego ramię jest zbudowane z kilku zespołów ruchu obrotowego o równoległych osiach obrotu.

Różnica między manipulatorem a robotem jest następująca: manipulator wykonuje zamknięty cykl ruchów powtarzalnych, na ogół ma on sztywny program (z reguły zmiana programu pracy manipulatora wymaga fizycznych zmian w jego konstrukcji), robot natomiast może realizować dużą liczbę różnorodnych czynności manipulacyjnych za pomocą sygnałów generowanych w programowalnym układzie sterowania. Wykonuje on najczęściej powtarzalny, ale mogący ulec zmianie odpowiednio do zmiany programu, stanu środowiska lub podanej informacji, cykl ruchów manipulacyjnych lub/i lokomocyjnych. Robot ponadto wykorzystując swoje układy wejść/wyjść może pełnić rolę nadrzędną w stosunku do urządzeń technologicznych, z którymi współpracuje.

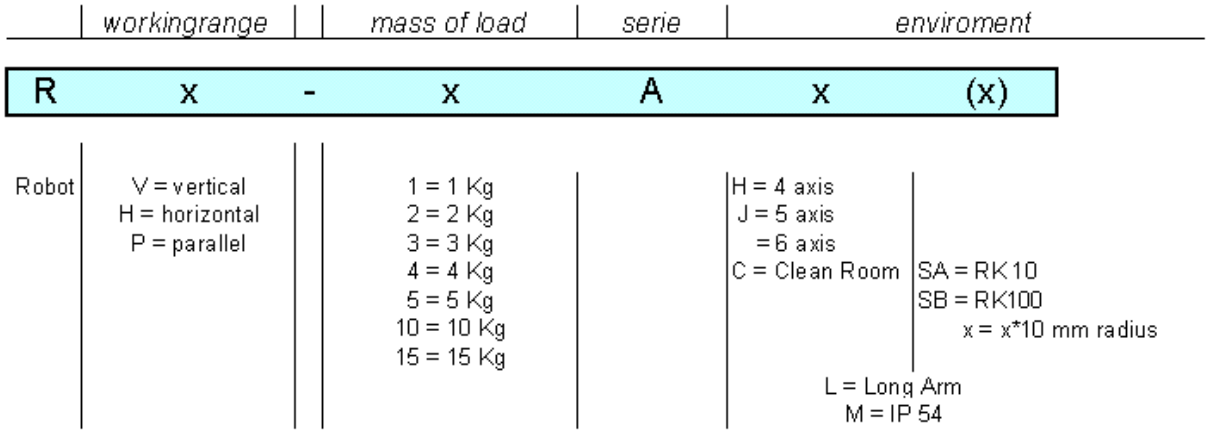
Przez robotyzację będziemy rozumieli działania mające na celu automatyzację pracy produkcyjnej za pomocą manipulatorów i robotów.

Mechanizacja polega na zastępowaniu w procesie produkcyjnym pracy fizycznej człowieka przez pracę maszyn.

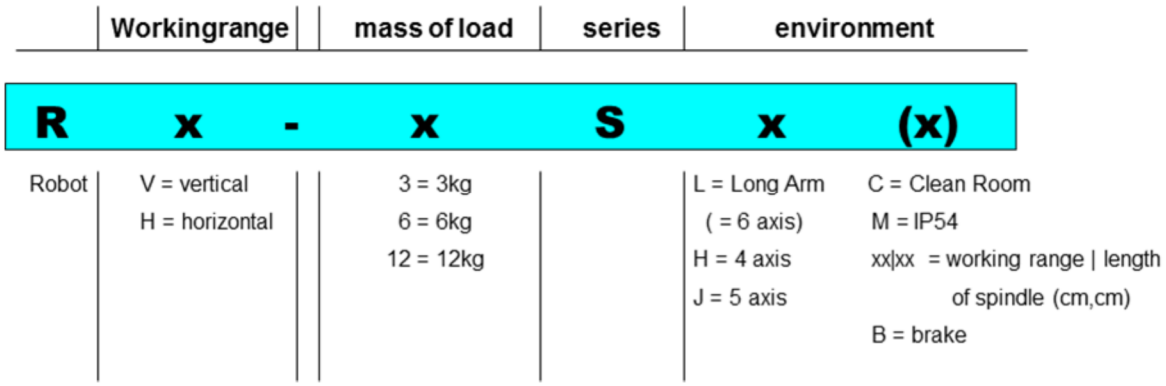
W odniesieniu np. do obrabiarek najpierw nastąpiła mechanizacja ruchów roboczych, a następnie w miarę upływu czasu, mechanizacja ruchów pomocniczych, podawania przedmiotów obrabianych, transportu przedmiotów, narzędzi, przyrządów i uchwytów.

Automatyzacja polega na zastępowaniu człowieka w sterowaniu ręcznym urządzeniami pracującymi bez bezpośredniego udziału człowieka. Urządzenia te przejmują funkcje człowieka związane głównie z jego wysiłkiem umysłowym. Sterowanie wykonywane przez urządzenia nazywa się sterowaniem automatycznym. Urzeczywistnienie tego rodzaju sterowania bez uprzedniego lub równoczesnego wprowadzenia mechanizacji jest niemożliwe. Podstawowym czynnikiem umożliwiającym realizacje automatyzacji stał się postęp w dziedzinie programowalnych urządzeń sterujących. Nastąpił rozwój technik sterowania numerycznego i komputerowego. Obrabiarki sterowane numerycznie automatycznie wykonują cykl obróbki. Szczególny postęp w dziedzinie automatyzacji wiąże się z wprowadzeniem robotów przemysłowych. Pojawiły się roboty do automatycznego spawania, zgrzewania, malowania, montażu mechanicznego i elektrycznego. Nastąpił również gwałtowny rozwój programowalnych urządzeń pomocniczych, wśród których warto wymienić sterowane komputerowo maszyny pomiarowe, myjnie suszarnie, stanowiska do konserwacji, magazyny narzędzi, przyrządów i uchwytów.

Na rys. 2.1 przedstawiono oznaczenia robotów z serii A natomiast na rys. 2.2 oznaczenia robotów z serii S.



Rys. 2.1. Oznaczenia robotów z serii A



Rys. 2.2. Oznaczenia robotów z serii S

3 Środowisko programowe COSIMIR

3.1 Ogólne informacje o środowisku COSIMIR.

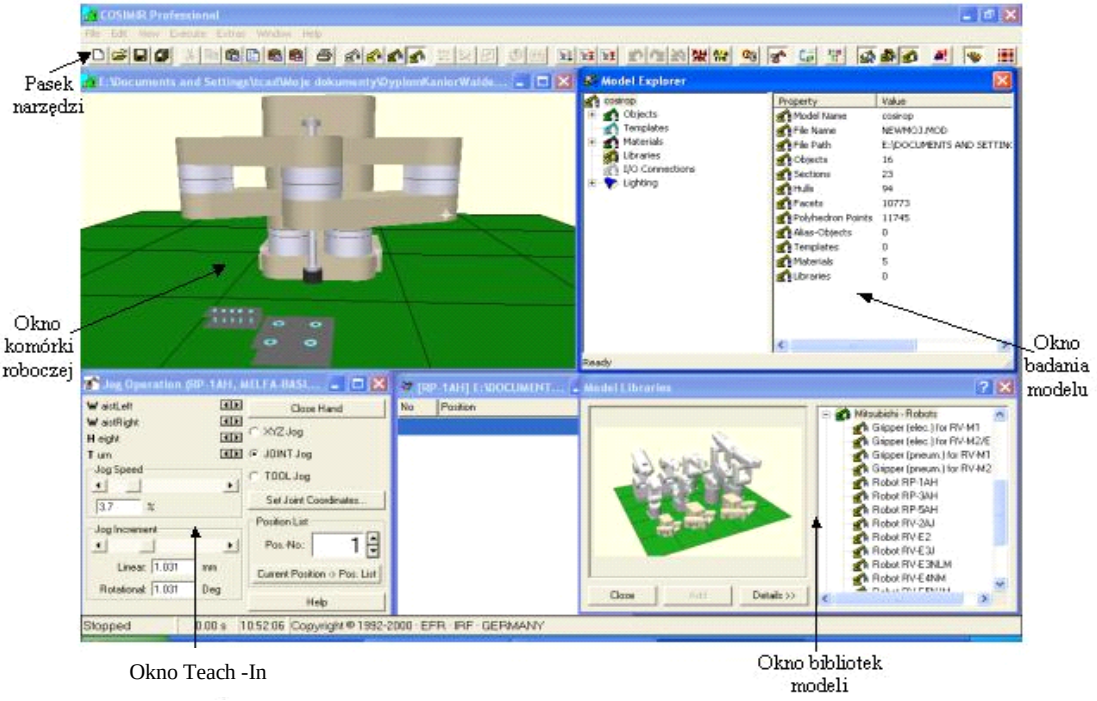
Program COSIMIR przeznaczony jest do tworzenia trójwymiarowych symulacji ułatwiając wdrażanie nowych oraz modernizacji starych procesów przemysłowych bez potrzeby ich wyłączania. Dzięki zastosowaniu takiego oprogramowania jakim jest COSIMIR możemy bezproblemowo i w sposób rzeczywisty zaprojektować nie tylko pojedyncze operacje procesu, ale również bardzo skomplikowane (na przykła:. całe hale produkcyjne – pełny proces montażu jakiegoś wyrobu począwszy od montażu płytek drukowanych, poprzez ich montaż w obudowach a skończywszy na pakowaniu gotowych produktów). Program ten umożliwia również programowanie zastosowanych urządzeń w celu przetestowania poprawności wykonywanych operacji i wykryciu błędów, które w rzeczywistym układzie mogły by spowodować uszkodzenie urządzenia.

COSIMIR posiada bardzo bogatą bibliotekę urządzeń (roboty, przenośniki taśmowe, czujniki, stoły obrotowe itp.) do wykorzystania w symulacjach, jednakże program ten ma bardzo przydatną funkcję jaką jest importowanie. Dzięki tej funkcji możliwe jest wczytywanie do danego projektu elementów z innych projektów, a także z programów typu CAD (np.: programy firmy AUTODESK AUTOCAD lub INVERTOR).

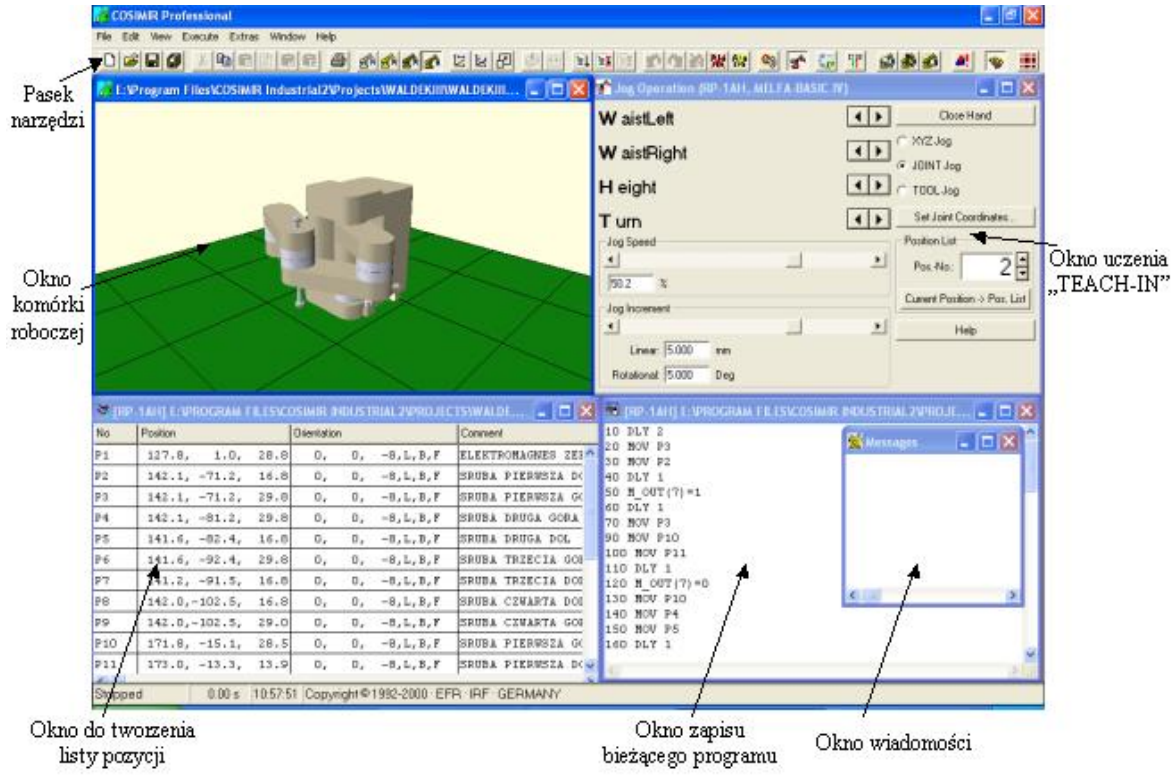
Właściwości COSIMIRA dają możliwość zaprojektowania, zaprogramowania i przeprowadzenie prób pozwalających sprawdzić działanie funkcjonalne procesu produkcji (optymalizacja cyklu pracy i uniknięcie kolizji) oraz opłacalność wdrażania nowych zautomatyzowanych procesów produkcyjnych, bez potrzeby montażu rzeczywistych układów. Stworzone w COSIMIR'ze programy dla symulacji możemy wykorzystać w powstałych rzeczywistych projektach, nie ma potrzeby pisania nowych programów dla rzeczywistych

układów. Dzięki temu można łatwo przystosować np. linię produkcyjną do nowego produktu w trakcie produkcji aktualnego, bez potrzeby zatrzymywania produkcji.

Poniżej na rys. 3.1 i rys. 3.2 umieszczone są obrazy okien dialogowych środowiska COSIMIR.



Rys. 3.1. Widok okien dialogowych podczas tworzenia projektu.



Rys. 3.2. Widok okien dialogowych podczas tworzenia kodu programu procesu.

Ze względu, iż zakres niniejszej pracy dyplomowej nie obejmuje dokładnego opisu środowiska COSIMIR, dlatego też w celu głębszego zapoznania się ze sposobem tworzenia projektów i programowania we wspomnianym programie do sterowania procesami przemysłowymi należy korzystać z HELP'u COSIMIR'a.

3.2 Ogólny opis sposobów programowania robotów na przykładzie robota RP-1AH

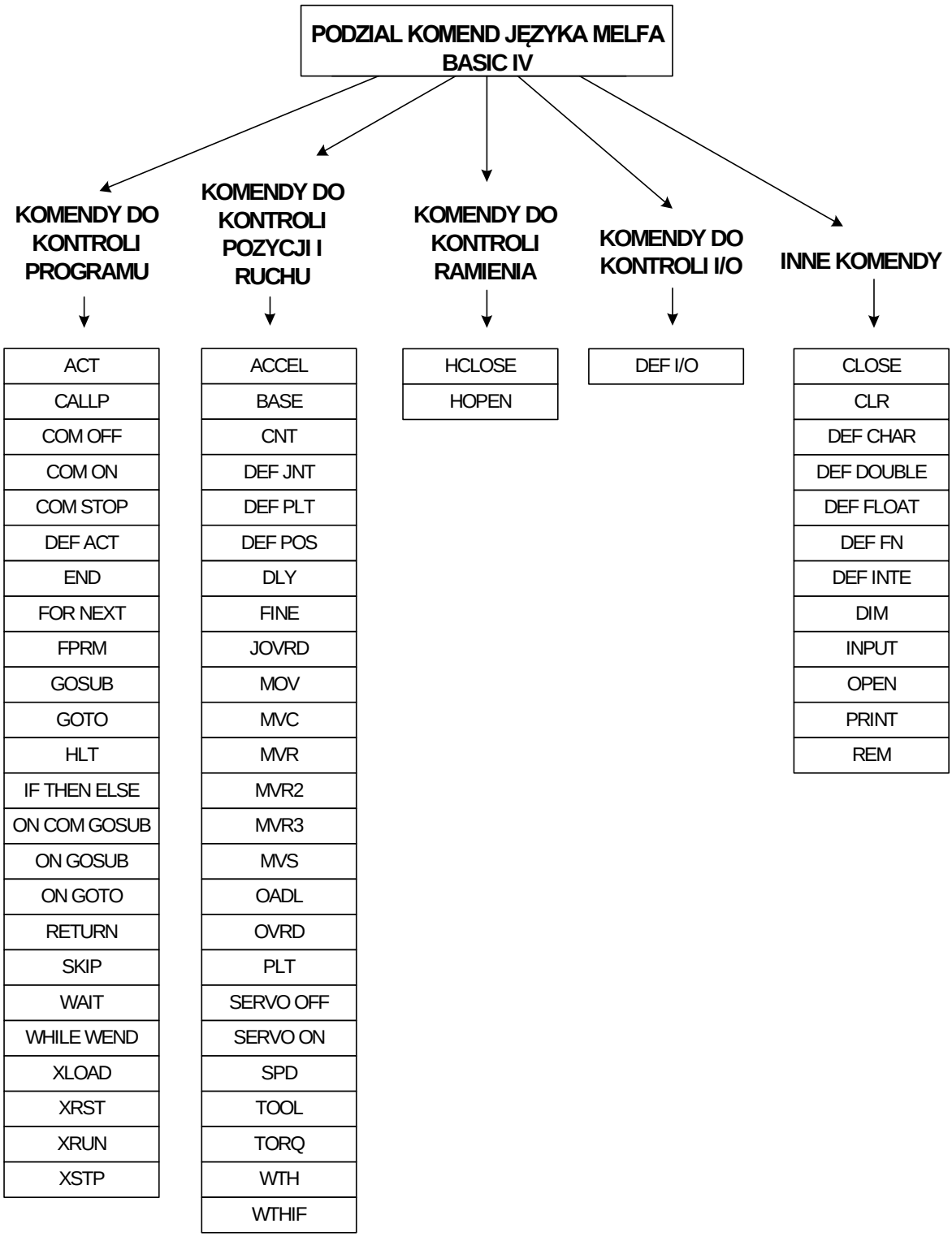
Melfa Basic IV jest językiem programowania zaprojektowanym specjalnie dla robotów Mitsubishi. Przy użyciu tego języka można bezpośrednio programować ruch robota, jak również realizować wiele specjalnych funkcji takich jak na przykład: obliczenia. Struktura Melfa Basic IV jest bardzo zbliżona do znanego od wielu lat, prostego języka programowania jakim jest BASIC. Liczba funkcji obu języków jest podobna.

Programowanie robotów firmy Mitsubishi można realizować na kilka sposobów. Najczęściej stosowane to programowanie z poziomu operatora przy pomocy TEACHING PADA (programowanie przez pokazywanie) oraz programowanie przy użyciu PC-ta w języku MELFA BASIC IV lub MOVEMASTER COMMAND korzystając z oprogramowania COSIMIR/COSIROP.

Ze względu na to, iż robot RP-1AH jest robotem który porusza się z punktu do punktu, programowanie polega na ustawieniu pozycji a następnie użyciu ich w pisaniu wierszy poleceń dla robota. Sposób pisania programów przy zastosowaniu Teaching Pad'a i komputera PC jest taki sam. Składnia wiersza poleceń opisana jest w kolejnym podrozdziale. Programowanie TEACHING PADEM zwane jest inaczej programowanie przez pokazywanie. Pisanie programów przy użyciu TEACHING PADA jest bardziej żmudne i trwa o wiele dłużej od metody pisania w programie COSIMIR. Ze względu na to, że nie zawsze można korzystać z komputera, użytkownik często jest zmuszony do korzystania TEACHING PADA.

W celu napisania poleceń do wykonania przez robota używa się komend języka MELFA –BASIC IV. Komendy tego środowiska programowego można podzielić na komendy kontroli programu, pozycji i ruchu, kontroli ramienia oraz kontroli I/O, a także komendy służące innym funkcją na przykład: definiowanie zmiennych itp.

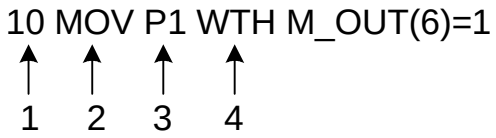
Na rys. 3.3 widoczny jest podział komend języka MELFA – BASIC IV, a ich opis i zastosowanie przedstawiono w rozdziale 4 i 5.



Rys. 3.3. Podział komend języka Melfa-Basic IV.

3.3 Składnia wiersza poleceń w języku MELFA-BASIC IV.

Każdy język programowania ma specyficzny dla siebie sposób zapisywania poleceń w programach. Poniżej przedstawiono składnię jednego wiersza programu obrazującego jego składnię.



1 – numer linii programu (numerowanie odbywa się w porządku rosnącym od 1do 32767.
Linia może zawierać do 127 znaków i zawierać może TYLKO jedną instrukcję za wyjątkiem powyższego przykładu.
2 – komenda,
3 – parametr (-y) danej instrukcji,
4 – dodatkowe instrukcje wykonywane równolegle z komendą ruchu (MOV, MVS, MVR, MVR2, MVR3, MVC).

Język MELFA – BASIC IV tak jak każdy język programowania posiada zasady pozwalające na właściwy zapis przez osobę programującą i odpowiednie przetworzenie na kod zrozumiały dla robota. Są one następujące:

- na początku wiersza musi być jego numer wyjustowany do lewej strony;
- gradacja numeracji musi być stała i o wartości zdefiniowanej w oprogramowaniu, domyślnie każdy następny wiersz jest zwiększany o 10;
- program musi się kończyć komendą END (bardzo ważne);
- a kursor musi przejść do następnego wiersza.

4 Programowanie.

Rodzaje programowania:

- Przez pokazywanie z użyciem ręcznego panelu sterującego.
- Programowanie w języku MELFA BASIC IV lub Movemaster Command z wykorzystaniem oprogramowania COSIROP/COSIMIR.

Idea programowania przez pokazywanie jest następująca:

- Uruchomienie ręcznego panelu sterującego.
- Wybranie z głównego menu pozycji TEACH, podania nazwy programu i zatwierdzenie przez naciśnięcie przycisku [INP/EXE].
- Ustawienie robota w odpowiedniej pozycji.
- Zapamiętanie pozycji.

- Po zapamiętaniu odpowiedniej ilości pozycji najwygodniej jest wczytać je do komputera w celu naniesienia ewentualnych poprawek.

Aby uruchomić zmodyfikowany program należy wgrać go ponownie do sterownika.

4.1 Programowanie w języku MELFA

- Nazwa programu może zawierać do 12 znaków. Zalecane jest stosowanie w nazwie do znaków, ze względu na ograniczenia ręcznego panelu sterowania (program mający nazwę dłuższą niż znaki nie może być uruchomiony za pomocą ręcznego panelu sterującego).
- W przypadku, gdy program jest wywoływany za pomocą funkcji CALLP, jego nazwa może składać się z więcej niż 4 znaków.
- Nazwa programu może się składać z dużych liter oraz cyfr.
- Jeżeli użyty jest zewnętrzny sygnał wyjściowy do wyboru programu, nazwa programu musi składać się z samych cyfr.

4.1.1 Etykiety:

Etykietę definiuje się, aby oznaczyć rozgałęzienia programu. Np.: w przypadku spełnienia jakiegoś warunku program przechodzi do miejsca oznaczonego przez etykietę. Nazwa etykiety musi być poprzedzona znakiem *****

*10 GOTO * LBL*

.....

*100 * LBL*

Znaki, które nie mogą być użyte w nazwie etykiety: słowa kluczowe języka nazwy zaczynające się od symbolu lub znaku specjalnego etykieta nie może się nazywać jak słowo zarezerwowane jako nazwa zmiennej lub funkcji.

4.1.2 Znaki używane w programowaniu

O – znak może być używany bez żadnych ograniczeń

X – znak nie może być używany

ξ – znak może być używany pod pewnymi warunkami

Class	Available characters	Program name	Variable name	Label name
Alphabetic characters	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z	○	○	○
	a b c d e f g h i j k l m n o p q r s t u v w x y z	×	△ Note1)	△ Note 1)
Numerals	0 1 2 3 4 5 6 7 8 9	○	△Note2)	○
Symbols	" ' & ( ) * + − . , / : ; = < > ? @ ` [¥] ^ [ ] ~ |	×	×	×
	! # $ %	×	Available for type specification	×
	_ (Underscore)	×	△Note3)	△Note4)
Spaces	Space character	×	×	×

Ad 1). W nazwach zmiennych i etykietach automatycznie konwertowane są na duże znaki.

Ad 2). Tylko litery alfabetu mogą być używane jako pierwsza litera w nazwie zmiennej. Cyfry mogą używane jako kolejne znaki.

Ad 3). Znak ‘_’ może być używany jako minimum drugi znak. Zmienna, której nazwa ma jako drugi znak ‘_’ jest zmienną zewnętrzną.

Ad 4). Jeżeli ‘_’ jest używany w nazwie etykiety to musi być stosowana następująca konwencja: *L_xxxx.

4.1.3 Zmienne – klasy zmiennych.

- Zmienne pozycji (lokacji): nazwa takiej zmiennej zaczyna się od litery ‘P’
- Zmienne złączowe: określają przemieszczenia poszczególnych złączy robota. Nazwa takiej zmiennej zaczyna się od litery ‘J’.
- Zmienne arytmetyczne: zmienne przechowujące liczby całkowite i rzeczywiste. Nazwa zmiennej zaczyna się od litery ‘M’.
- Zmienne znakowe: przechowują ciągi znaków. Na końcu nazwy zmiennej znajduje się znak ‘$’ (C1$ =„napis”).

4.1.4 Znaki specjalne:

Rozróżnianie dużych i małych liter: Małe i duże litery są rozróżniane tylko w komentarzach oraz w wartościach zmiennych znakowych. W innych przypadkach małe litery konwertowane są na duże.

- Znak podkreślenia (_): Kiedy znak ten występuje w nazwie zmiennej jako drugi znak, to wtedy dana zmienna jest zmienną zewnętrzną.
- Apostrof (‘): Znak otwiera komentarz (komentarz jednoliniowy). Znak ten użyty na początku linii zastępuje polecenie REM:

100 MOV P1 'GET

150 'GET PARTS

- Znak (*): Znak ten umieszczany jest przed nazwą etykiety.
- Przecinek (,): Używany jako ogranicznik w miejscach, gdzie znajduje się kilka parametrów lub przyrostków: P1=(200,120,...).
- Kropka (.): Kropka stosowana jest w celu uzyskania pewnych składowych danych znajdujących się pewnych strukturach. Można np. uzyskać dziesiętny punkt, poszczególne składowe zmiennych pozycji i złączowych.

$$M1 = P1.X$$

- Spacja: Używana jako część łańcucha znaków lub jako część komentarza. Znak ten jest wymagany między liczbą określającą linię a słowem komendy.

4.1.5 Stałe numeryczne i alfanumeryczne:

- Stałe numeryczne - przykład:

dziesiętne: 234, 7471, -435, +546, -5454

szenastkowe: &H03FA, &H1AE5, &HA5

binarne: &B0101, &B110110101, &B10101111

- Stałe alfanumeryczne - przykład:

"My new program"

"Interrupt signal"

„Wejscie cyfrowe"

4.2 Komendy ruchu:

4.2.1 Ruch w interpolacji złączowej (MOV).

Ruch robota do określonej lokalizacji odbywa się w interpolacji złączowej. (trajektoria końcowa jest nieistotna – ruch jest interpolowany w jednostkach zmiennych złączowych)

MOV - ruch robota do zadanej lokalizacji z interpolacją złączową. Mogą być dołączane polecenia dodatkowe: WTH lub WTHIF.

Przykład:

MOV P1 ‘Ruch do P1.

MOV P1+P2 ‘Ruch po pozycji uzyskanej jako suma elementów współrzędnych P1 i P2.

MOV P1*P2 ‘Ruch po pozycji uzyskanej jako iloczyn elementów współrzędnych P1 i P2

MOV P1,-50 ‘Ruch z P1 do położenia cofniętego o 50mm w kierunku chwytaka

MOV P1 WTH M_OUT(6)=1 'Start w kierunku P1 z jednoczesnym załączeniem sygnału ‘wyjściowego na kanale cyfrowym 6

MOV P1 WTHIF M_IN(5)=1,SKIP ‘Jeżeli podczas ruchu do P1 został podany sygnał na ‘wejście kanału cyfrowego 5 ruch do P1 jest zatrzymany a program jest kontynuowany do ‘następnego STOP.

4.2.2 Ruch w interpolacji liniowej (MVS).

Robot porusza się z jednego punktu do drugiego po linii prostej (trajektoria obliczana przez sterownik). Ta najkrótsza droga nie jest najszybszą, ponieważ w tym przypadku sterownik musi sterować większą ilością osi niż przy interpolacji osiowej.

MVS – ruch robota do zadanej lokalizacji w interpolacji liniowej. Mogą być dołączane polecenia dodatkowe: WTH lub WTHIF.

Przykład:

MVS P1. 'Moves to P1

MVS P1+P2 ‘Moves to the position obtained by adding the P1 and P2 coordinate ‘elements.

MVS P1*P2 ‘Moves to the position relatively converted from P1 to P2.

MVS P1,-50 ‘Moves from P1 to a position retracted 50mm in the hand direction.

MVS ,-50 ‘Moves from the current position to a position retracted 50mm in the hand ‘direction.

MVS P1 WTH M_OUT(17)=1 ‘Starts movement toward P1, and simultaneously turns ‘output signal bit 17 ON.

MVS P1 WTHIF M_IN(20)=1,SKIP ‘If the input signal bit 20 turns ON during ‘movement to P1, the movement to P1 is stopped, and the program proceeds to the next ‘stop.

4.2.3 Ruch w interpolacji kołowej.

Ruch robota odbywa się wzdłuż zadanego łuku poprzez trzy współrzędne z wykorzystaniem interpolacji kołowej.

MVR P1, P2, P3 ' Moves with circular interpolation between P1 . P2 . P3.

MVR P1, P2, P3 WTH M_OUT (17) = 1 ‘Circular interpolation between P1 . P2 . P3 ‘starts, and the output signal bit 17 turns ON.

MVR P1, P2, P3 WTHIF M_IN (20) = 1, SKIP ‘If the input signal bit 20 turns ON during ‘circular interpolation between P1 . P2 . P3, circular interpolation to P1 is stopped, and ‘the program proceeds to the next step.

MVR P1, P2, P3 TYPE 0,1 ‘Moves with circular interpolation between P1 . P2 . P3.

MVR2 P1, P3, P11. ‘Circular interpolation is carried out from P1 to P3 in the direction ‘that P11 is not passed. P11 is the reference point.

MVR3 P1, P3, P10 ‘Moves with circular interpolation from P1 to P3 in the direction with ‘the smallest fan angle. P10 is the center point.

MVC P1, P2, P3 . ‘Moves with circular movement from P1 . P2 . P3 .P1

4.2.4 Ruch ciągły.

Ruch robota odbywa się w postaci gładkich przejść między zadanymi lokacjami, bez zatrzymania w poszczególnych położeniach. Start i koniec ruchu ciągłego jest realizowany przy pomocy poleceń ruchu. Prędkość może być zmieniana podczas ruchu.

CNT ‘oznacza start i koniec ruchu ciągłego.

CNT 1, 100, 200 ‘określa start ruchu ciągłego wraz z punktem startowym odległym o 100 mm i punktem końcowym odległym o 200 mm od zadanego położenia

CNT 0 ‘koniec ruchu ciągłego.

4.3 Sterowanie przyspieszeniem/prędkością.

Wartość przyspieszenia/opóźnienia ustalana jest jako procent maksymalnej wartości przyspieszenia/opóźnienia.

Prędkości – jako wartości procentowe oraz bezwzględne.

OALD – załącza optymalną wartość przyspieszenia/opóźnienia z uwzględnieniem wartości przenoszonego obciążenia.

SPD – ustala prędkość ruchu w interpolacji liniowej i kołowej jako wartość prędkości bezwzględnej w (mm/s).

JOVRD – określa prędkość ruchu w interpolacji złączowej jako % wartość prędkości maksymalnej.

OVRD – ustala procentową wartość prędkości w całym programie względem procentowej wartości prędkości maksymalnej (dotyczy wszystkich typów interpolacji).

ACCEL – określa procentową wartość przyspieszenia i opóźnienia względem ich wartości maksymalnej lub dezaktywacja.

OALD – załączenie funkcji umożliwiającej realizację optymalnej wartości przyspieszenia/opóźnienia.

SPD 30 – ustala prędkość dla instrukcji ruchu realizowanych w interpolacji liniowych i kołowych na 30mm/s.

JOVRD 70 – ustala prędkość dla instrukcji realizowanych w interpolacji złączowej na 70% prędkości max.

OVRD 50 – dla instrukcji ruchu w wszystkich trybach interpolacji ustala prędkość na 50% wartości maksymalnej.

ACCEL 60,80 – ustawia przyspieszenie na 60% i opóźnienie na 80%

ACCEL – przyspieszenia i opóźnienia ustalone na wartość max. - 100%.

Przykład:

10 ACCEL 100,50 ‘100 oznacza 100% = czas przyspieszania 0.2s;

‘50 oznacza 200% = czas hamowania 0.4s

20 MOV P1 ‘Ruch do punktu P1 z interpolacją osiową

30 MOV P2 ‘Ruch do punktu P2 z interpolacją osiową

$$t = \frac{100\%}{A[\%]} \cdot 0{,}2s$$

gdzie: A – parametr komendy ACCEL

4.3.1 Oszacowanie prędkości.

Ruch w interpolacji liniowej i kołowej :

wartość na (sterowniku) Controller (T/B) ×OVRD wartość ×SPD wartość

Ruch w interpolacji złączowej:

wartość na (sterowniku) Controller (T/B) ×OVRD wartość ×JOVRD wartość.

4.4 Sterowanie chwytakiem:

HOPEN - otwiera wyznaczony chwytak.

HCLOSE - zamyka wyznaczony chwytak.

TOOL -przesunięcie układ współrzędnych chwytaka .

DLY -zatrzymanie ruchu chwytaka.

Przykład:

HOPEN 1 Otwiera chwytak 1.

HOPEN 2 Otwiera chwytak 1

HCLOSE 1 Zamyka chwytak 1.

TOOL (0,0,95,0,0,0) Ukł. Współ. Przesunięty w osi z 95mm (w kierunku wydłużenia chwytaka (ustawia długość chwytaka na 95mm).

4.4.1 Zegar DLY

Funkcja ta powoduje zatrzymanie programu na pewien określony z góry okres czasu lub wysłanie sygnału przez pewien czas.

Składnia:

DLY *wartość* – gdzie *wartość* to czas z dokładnością 0.01s

Przykład:

100 DLY 1 ‘zatrzymuje program na 1 sekundę.

110 M_OUT(4)=1

DLY 0.5 ‘wysyła sygnał na wyjściowy kanał cyfrowy 4 przez 0.5 sekundy

120 HOPEN 1

130 DLY 0.5 ‘czeka pół sekundy na otwarcie chwytaka

4.5 Sygnały wejściowe/wyjściowe.

Sygnały I/O są to sygnały binarne robota z serii Mitsubishi wyposażone są standardowo w obsługę 16 kanałów wyjściowych i 16 wejściowych.

DLY – opóźnienie działania programu (zegar) ,

CLR – resetuje zmienne,

WAIT –zatrzymanie programu,

M_OUTx –podaje daną wartość na wyjściowy kanał cyfr,

M_INx – sczytuje wartość z wejściowy kanału cyfr,

4.5.1 Sygnały wejść (M_Inx).

M_IN(?) - Sczytuje wartość z kanału ?,

M_INB(?) -Sczytuje wartość z kanału ? I kolejne do ?+7,

M_INW (?) - Sczytuje wartość z kanału ? I kolejne do ?+15,

M_DIN.

Przykłady:

100 M2 = M_IN (2) ‘Przypisuje 1 bitowej zmiennej M2 wartość kanału 2

110 M1 = M_INB (20) ‘Przypisuje 8 bitowej zmiennej M1 wartości kanałów od 20 do 27

120 M1= M_INW (5) ‘Przypisuje 16 bitowej zmiennej M1 wartości kanałów od 5 do 20

4.5.2 Sygnały wyjścia (M_OUTx).

M_OUT(?) - Zapisuje wartość na kanał ?,

M_OUTB(?) - Zapisuje wartość na kanał ? I kolejne do ?+7,

M_OUTW (?) - Zapisuje wartość na kanał ? I kolejne do ?+15,

M_DOUT.

Przykłady:

100 M_OUT (2)=0 ‘Zapisuje wartość 0 na kanał 2

110 M_OUTB (20)=0 ‘Zapisuje wartość 0 na kanał od 20 do 27

120 M_OUTW (5)=1 ‘Zapisuje wartość 1 na kanał od 5 do 20

4.5.3 WAIT.

WAIT – komenda powodująca zatrzymanie programu.

Składnia:

WAIT ? Gdzie ? - Parametr na który czekamy.

Przykład:

20 WAIT M_IN(3)=0 ‘Czeka aż na kanale 3 będzie wartość 0

40 WAIT M_01=100 ‘Czeka aż na M_01 =100

4.5.4 CLR.

CLR – resetuje dane kanały lub zmienne.

Składnia:

CLR ? Gdzie ? wybór funkcji

Przykład:

100 CLR 1 ‘czyści sygnały wyjściowe

110 CLR 2 ‘czyści wewnętrzne wartości zmiennych i tablice

120 CLR 3 ‘czyści zewnętrzne wartości zmiennych i tablice

130 CLR 0 ‘czyści wszystkie zmienne i sygnały wyjściowe

4.6 Sterowanie kolejnością wykonywania instrukcji.

Poniższe instrukcje pozwalają na sterowanie kolejnością wykonywania instrukcji przez robota.

GOSUB – przywołanie wyznaczonej linii lub etykiety programu.

ON GOSUB – Wywołuje program z wyznaczoną ilością zmiennych. Wartość warunku wzrasta wraz ze wzrostem liczb całkowitych.

RETURN – realizuje powrót do następnej linii programu po instrukcji GOSUB, która przywołała daną linie programu lub etykietę.

CALLP – wywołuje dany program. Do wywoływanego programu można przekazywać zmienne.

FPRM – pobiera wartość zmiennych przesłanych do programu przez instrukcje CALLP.

Przykład:

FPRM M10, P10 ‘Pobiera wartości zmiennych M10 oraz P10 które zostały przekazane do ‘programu za pomocą instrukcji CALLP

CALLP „20”, M1, P1 ‘Transportuje zmienne M1 oraz P1 do programu „20” i wywołuje ‘ten program

CALLP „10” ‘Wywołanie programu o nazwie 10

RETURN ‘Powrót do linii następującej po instrukcji GOSUB z której był wywołany podprogram

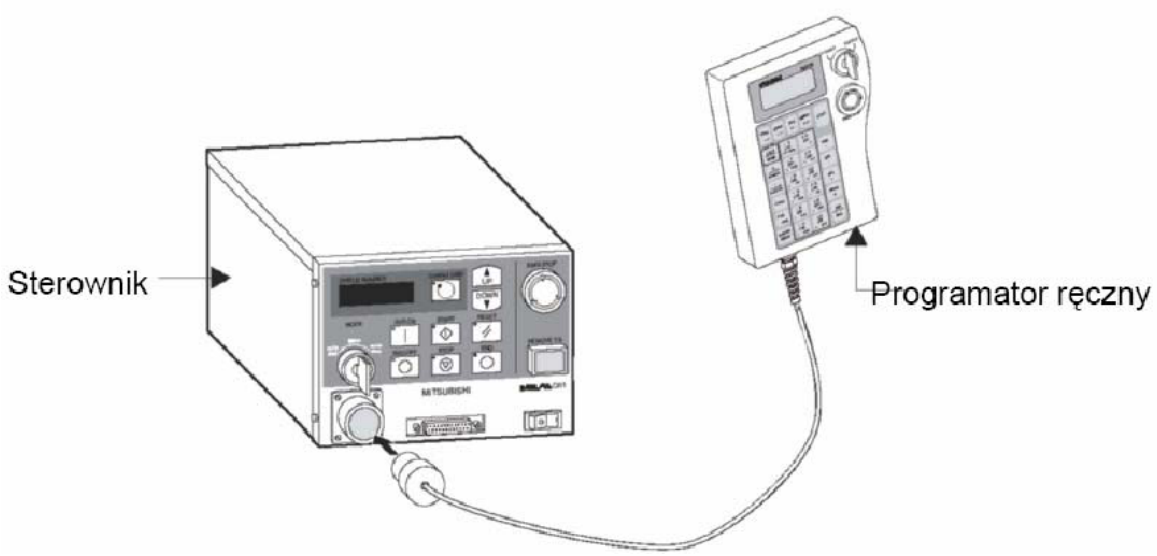
ON M1 GOSUB 100, 200, 300 ‘Jeżeli M1 == 1przywołuje linię programu o numerze ‘100, jeżeli M1 == 2 – linię 200...

GOSUB *LBL ‘Przywołanie miejsca w programie znaczonego przez etykietę LBL

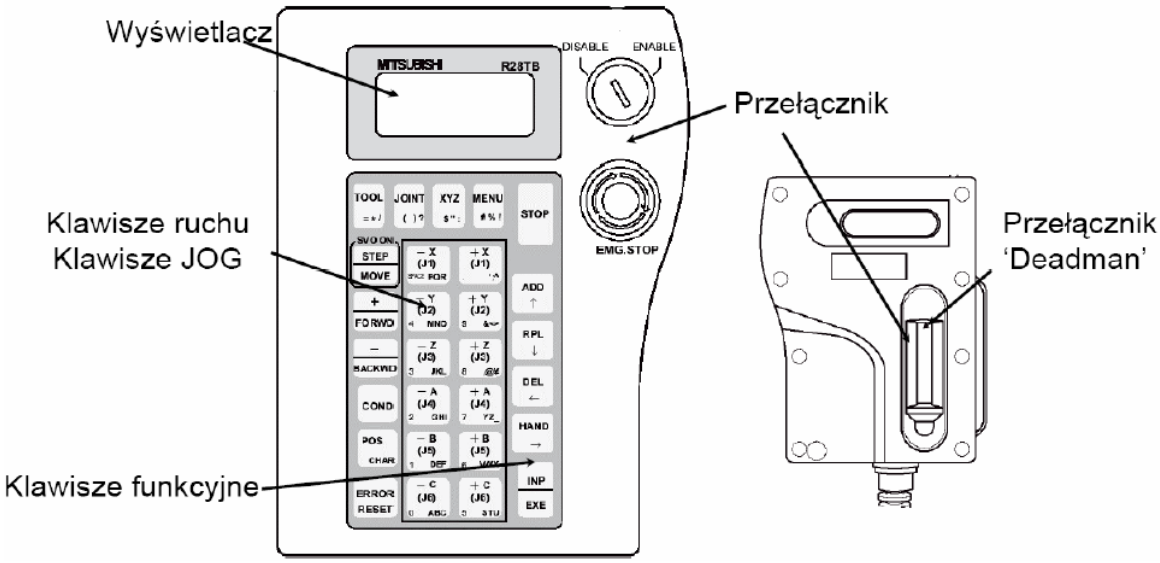
GOSUB 100 ‘Przywołanie linii 100.

4.7 Programowanie przy użyciu TECHING PADA (ręcznego panelu sterującego).

W panelu czołowym sterownika CR-1 znajduje się gniazdo służące do podłączenia ręcznego panelu sterującego. Za pomocą panelu sterującego możliwa jest realizacja szeregu funkcji.



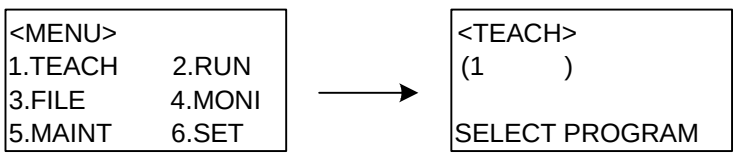
Rys. 4.1. Sposób przyłączenia ręcznego panelu sterującego do kontrolera.



Rys. 4.2. Ręczny panel sterujący.

Idea programowania przez pokazywanie jest następująca:

- Uruchomienie ręcznego panelu sterującego.
- Wybranie z głównego menu pozycji TEACH, podania nazwy programu i zatwierdzenie przez naciśnięcie przycisku [INP/EXE].

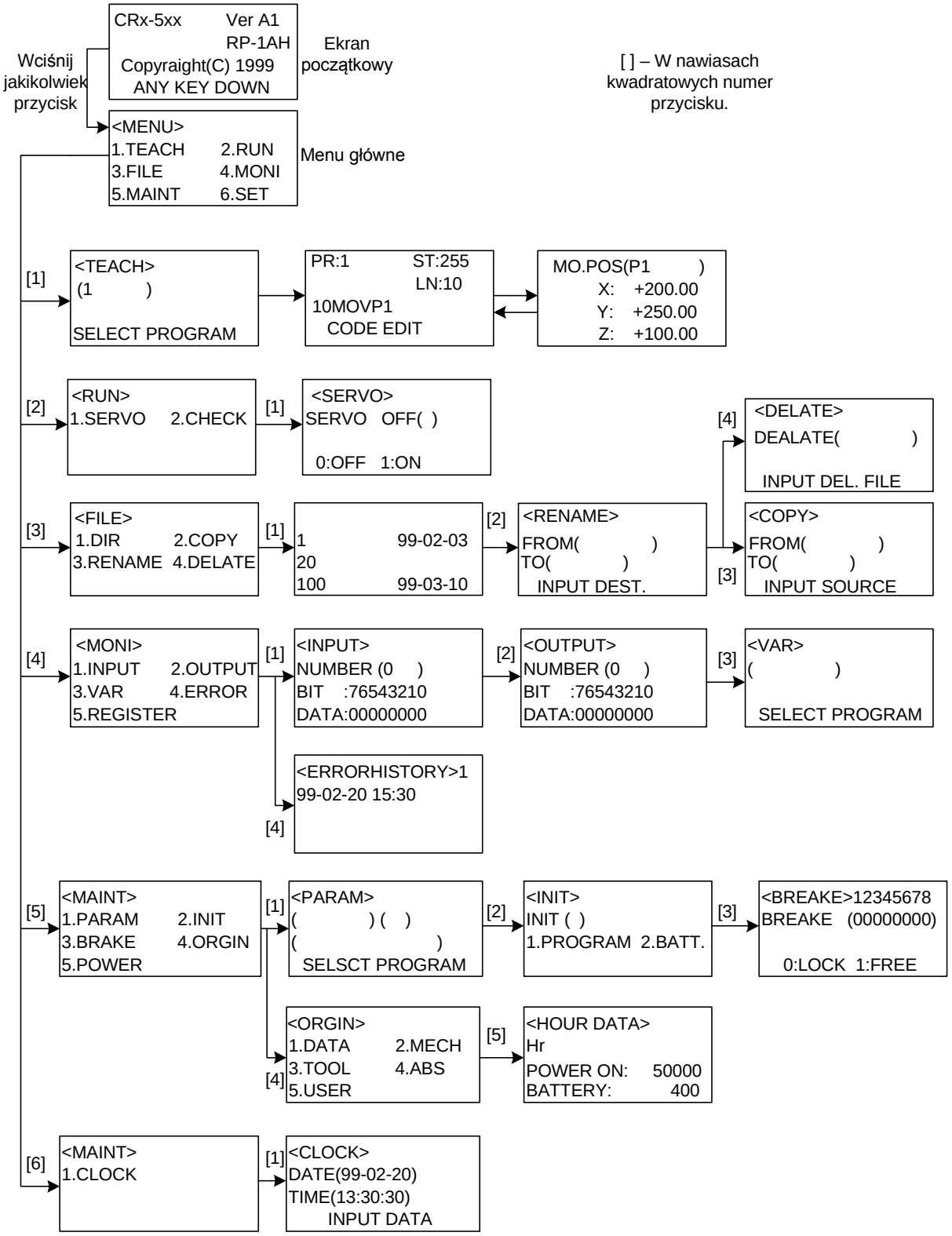


- Ustawienie robota w odpowiedniej pozycji.
- Zapamiętanie pozycji.

- Po zapamiętaniu odpowiedniej ilości pozycji najwygodniej jest wczytać je do komputera w celu naniesienia ewentualnych poprawek.

Aby uruchomić zmodyfikowany program należy wgrać go ponownie do sterownika.

Na następnej stronie na rys. 3 znajduje się główne MENU ręcznego panelu sterującego.



Rys. 4.3. Główne MENU ręcznego panelu sterującego.

5 Komendy języka MELFA – BASIC IV

Nazwa komendy	Do czego służy	Składnia	Uwagi
1	2	3	4
ACCEL (Accelerate)	Ustalenie przyspieszenia/ opóźnienie przejazdu dla ruchu z interpolacją liniową lub kołową	ACCEL [Zakres przyspieszenia], [Zakres opóźnienia]	[Zakres przyspieszenia] / [Zakres opóźnienia] – są liczbami całkowitymi i określają w procentach osiągniecie od zera maksymalnej prędkości.
ACT (Act)	Powoduje ustawienie statusu przerwania, włącza lub wyłącza poszczególne przerwanie	ACT [Numer ważności] = [włączone/wyłączone]	[Numer ważności] – wartości od 1 do 8 – im większa wartość, tym mniejszy priorytet [włączone/ wyłączone] – wartości 1 lub 0.
BASE (Base)	Definiuje bazę konwersji współrzędnych, definiuje bazy układów współrzędnych	BASE [Baza konwersji współrzędnych]	[Baza konwersji współrzędnych] - Definicja bazy układu współrzędnych.
CALLP (Call P)	Umożliwia uruchomienie danego programu.	CALLP „[Nazwa programu]”, [Argument], [Argument],	[Nazwa programu] - zmienna typu łańcuchowego – nazwa programu; [Argument] - określa zmienne, które będą przeniesione do włączanego programu.
CLOSE (Close)	Umożliwia zamknięcie aktywnego portu komunikacji.	CLOSE [# Numer portu]	[# Numer portu] - numer, do którego został przypisany port komunikacji.
CLR (Clear)	Czyści wszystkie zmienne, tj. wyjścia cyfrowe, zmienne lokalne, zmienne globalne.	CLR [Typ]	[Typ]: 1 – wszystkie wyjścia cyfrowe czyści; 2 – wszystkie lokalne zmienne numeryczne; 3 – Wszystkie globalne zmienne numeryczne.
CNT (Continuous)	Umożliwia wykonanie ruchu z odpowiednią interpolacją	CNT [Włączony/wyłączony], [Wartość numeryczna 1], [Wartość numeryczna 2]	[Włączony/wyłączony] – „1” – funkcja CNT jest aktywna, „0” – nieaktywna, [Wartość numeryczna 1] – odległość jaką osiągnie ramię od zadanego punktu początkowego ruchu, [Wartość numeryczna 2] – odległość jaką osiągnie ramię od zadanego punktu końcowego ruchu.

COM OFF (Communication OFF)	Wyłącza komunikację przez linię komunikacyjną	COM [Linia komunikacji] OFF	[Linia komunikacji] – numer linii.
COM ON (Communication ON)	Włącza komunikację przez linię komunikacyjną.	COM [Linia komunikacji] ON	[Linia komunikacji] – numer linii.
COM STOP (Communication Stop)	Zatrzymuje komunikację przez linię komunikacyjną.	COM [Linia komunikacji] STOP	[Linia komunikacji] – numer linii.
DEF ACT (Define act)	Ustawienie procedury przerwania.	DEF ACT [Priorytet przerwania], [Wyrażenie] , [Proces]	Priorytet przerwania] – liczba rzeczywista z przedziału 1 do 8; [Wyrażenie] – warunek logiczny, który umożliwi zadziałanie przerwania; [Proces] – wykorzystuje się tutaj komendy GOTO lub GOSUB – skok do odpowiedniej części programu.
DEF PLT (Define pallet)	Przemieszczenie końcówki narzędzia robota (elementu wykonawczego robota) po prostej od aktualnej pozycji o współrzędnych X_o,Y_o,Z_o do pozycji $X=X_o+[dX]$, $Y=Y_o+[dY]$, $Z=Z_o+[dZ]$, w układzie odniesienia związanym z podstawą.	DEF PLT [Nr palety],[Punkt Startowy],[Punkt Końcowy A], [Punkt Końcowy B]	[dX],[dY],[dZ] – stała całkowita lub rzeczywista, zmienna, wyrażenie matematyczne, odległość w [mm]; '=' dla '[=dX]' – oznaczenie dla pozycji podanej w sposób absolutny. Prędkość przejazdu: zadawana poprzez parametr prędkości dla ruchu PTP lub komendy SPEED z opcją /P (SPEED /P).
DEF INTE/FLOAT/DOUBLE (Defien Integer/Float/Double)	Programowe definiowanie typu zmiennej: typu całkowitego, typu rzeczywistego 16-bitowego, typu rzeczywistego 32-bitowego.	DEF INTE [Nazwa] DEF FLOAT[Nazwa] DEF DOUBLE [Nazwa]	[Nazwa] - ciąg znakowy pełniący później funkcję nazwy deklarowanej zmiennej. Maksymalnie 8 znaków alfanumerycznych. [INT] – typ zmiennej, która jest liczbą całkowitą z zakresu od -32768 do 32767. [FLOAT] – typ zmiennej, która jest liczbą rzeczywistą 16-bitową z zakresu ±1.70141E+38 [DOUBLE] – typ zmiennej, która jest liczbą rzeczywistą 16-bitową z zakresu ±1.701411834604692E+308
DEF IO (Define IO)	Programowego definiowania typu zmiennej: typu wejściowego lub typu wyjściowego.	DEF IO [Nazwa]=[Typ zmiennej],[Numer we/wy]	[Nazwa] - ciąg znakowy pełniący później funkcję nazwy deklarowanej zmiennej, maksymalnie 8 znaków alfanumerycznych; [Typ zmiennej] - bit, bajt, słowo 16 bitowe, liczba całkowita; [Numer we/wy] - numer bitu na porcie wejść/wyjść cyfrowych.

DEF JNT (Define Joint)	Programowe definiowanie zmiennej typu pozycja dla trybu PTP.	DEF JNT[Nazwa]	[Nazwa] – ciąg znakowy pełniący później funkcję nazwy deklarowanej zmiennej, maksymalnie 8 znaków alfanumerycznych.
DEF POS (Define Position)	Programowe definiowanie zmiennej typu pozycja w układzie odniesienia związanym z podstawą lub narzędziem.	DEF POS [Nazwa]	[Nazwa] – ciąg znakowy pełniący później funkcję nazwy deklarowanej zmiennej, maksymalnie 8 znaków alfanumerycznych.
DEF CHAR (Define Character)	Programowe definiowanie zmiennej typu znakowego – łańcuch znaków.	DEF CHAR [Nazwa]	[Nazwa] – ciąg znakowy pełniący później funkcję nazwy deklarowanej zmiennej, maksymalnie 8 znaków alfanumerycznych.
DEF FN (Define Function)	Programowe definiowanie funkcji matematycznej.	DEF FN [Nazwa](Argument1, Argument2,)=[Działanie funkcji]	[Nazwa] – ciąg znakowy pełniący później funkcję nazwy deklarowanej zmiennej, maksymalnie 8 znaków alfanumerycznych. [Argument1, Argument2] – argumenty, wykorzystywane w działaniu funkcji. [Działanie funkcji] – działanie matematyczne realizowane przez funkcję [Nazwa].
DIM (Dim)	Definiowanie tablicy zmiennych. Tablica może być jedno, dwu lub trzy wymiarowa.	DIM [Nazwa](<Liczba zmiennych w pierwszym wymiarze>,<Liczba zmiennych w drugim wymiarze>,<Liczba zmiennych w trzecim wymiarze>)	[Nazwa] – ciąg znakowy pełniący później funkcję nazwy deklarowanej zmiennej, maksymalnie 8 znaków alfanumerycznych. <Liczba zmiennych w N wymiarze> - określa wielkość tablicy w poszczególnych wymiarach.
DLY (Delay)	Opóźnienie w wykonywaniu programu o zadanym czasie.	DLY [Czas]	[Czas] – stała całkowita lub rzeczywista, zmienna, wyrażenie matematyczne – liczba dodatnia podawana w sekundach począwszy od 0,05 [s].
END (End)	Zakończenie wykonywania danego programu.	END	Zakończenia programu
FINE (Fine)	Definiuje wartość o jaką może przesunąć się w danej osi ramię robota osiągając zadaną pozycję.	FINE [Odstępstwo], [Numer osi]	

FOR – NEXT (For-next)	Tworzenie pętli. W pętli FOR licznik zwiększany jest od wartości początkowej do wartości końcowej o określony krok.	FOR [Licznik] = [Wartość początkowa] TO [Wartość końcowa] STEP [Krok] . . . ‘program wykonywany w pętli NEXT [Licznik]	
FPRM (FPRM)	Odczyt zmiennych z programu, do którego zostały wysłane z programu głównego przy użyciu komendy CALLP.	FPRM [Zmienna 1], [Zmienna 2], ...	
GOSUB (Go Subroutine)	Skok bezwarunkowy do podprogramu.	GOSUB [Docelowa linia]	[Docelowa linia] – numer linii lub nazwa podprogramu.
GOTO (Go To)	Skok bezwarunkowy do podprogramu.	GOTO [Docelowa linia]	[Docelowa linia] – numer linii lub nazwa podprogramu.
HLT (Halt)	Przerwanie wykonywanie programu i ruchu robota.	HLT	
HOPEN / HCLOSE (Hand Open/Hand Close)	Steruje załączaniem i wyłączaniem narzędzi wykonawczych robota	HOPEN [Nr] HCLOSE [Nr]	[Nr] - numer narzędzia. Stała całkowita, wyrażenie matematyczne – liczba dodatnia od 1 do 8 – maksymalnie 8 narzędzi.
IF-THEN-ELSE (If Then Else)	Funkcja warunkowa, jeżeli spełniony jest warunek to wykonaj proces 1, jeżeli nie, to wykonaj proces 2.	IF [Warunek] THEN [Proces1] ELSE [Proces2]	
INPUT (Input)	Umożliwia odczytanie informacji przychodzącej z portu komunikacyjnego i przypisanie jej zmiennej.	INPUT [#Numer portu], [Nazwa zmiennej]	[# Numer portu] – numer, do którego został przypisany port komunikacji. [Nazwa zmiennej] – zmienna, do której zapisywane są informacje przychodzące na port komunikacyjny.
JOVRD (J Override)	Powoduje opóźnienie w wykonywaniu programu o zadany czas.	DEF JNT[Czas]	[Czas] – stała całkowita lub rzeczywista, zmienna, wyrażenie matematyczne – liczba dodatnia podawana w sekundach począwszy od 0,05 [s].
MOV (Move)	Powodująca przemieszczenie narzędzia wykonawczego robota do zadanego położenia w trybie ruchu PTP.	MOV [Położenie] <warunek>	[Położenie] - położenie w układzie odniesienia związanego z podstawą lub dla bezpośredniego sterowania osiami. Stała całkowita lub rzeczywista, zmienna, wyrażenie matematyczne.

MVC (Move C)	Umożliwiają przemieszczenie narzędzia roboczego robota po kole. Figura rysowana jest począwszy od pozycji początkowej, poprzez położenie przejściowe 1, następnie przez położenie przejściowe 2 i kończy się w położeniu początkowym. Jeśli aktualna pozycja jest inna niż położenie początkowe, to robot przemieści końcówkę do położenia początkowego ruchem z interpolacją liniową.	MVC [Położenie początkowe], [Położenie przejściowe 1], [Położenie przejściowe 2] <warunek>	[Położenie] - położenie w układzie odniesienia związanego z podstawą lub dla bezpośredniego sterowania osiami. Stała całkowita lub rzeczywista, zmienna, wyrażenie matematyczne. Prędkość przejazdu: zadawana poprzez rozkaz SDP.
MVS (Move S)	Umożliwia przemieszczenie narzędzia roboczego robota po linii prostej od aktualnego położenia do podanego położenia.	MVS [Położenie] <warunek>	[Położenie] – położenie w układzie odniesienia związanego z podstawą lub dla bezpośredniego sterowania osiami. Stała całkowita lub rzeczywista, zmienna, wyrażenie matematyczne. Prędkość przejazdu: zadawana poprzez rozkaz SDP.
MVR2 (Move R2)	Umożliwia przemieszczenie narzędzia roboczego robota po łuku. Figura rysowana jest począwszy od pozycji początkowej i kończy się w położeniu końcowym. Ruch wykonany jest po trajektorii łuku opisanego trzema kompletami współrzędnych, natomiast robot nie osiąga trzeciego punktu w składni procedury.	MVR2 [Położenie początkowe],[Położenie końcowe],[Położenie na obwodzie okręgu przez które robot nie wykonuje ruchu] <warunek>	[Położenie] – położenie w układzie odniesienia związanego z podstawą lub dla bezpośredniego sterowania osiami. Stała całkowita lub rzeczywista, zmienna, wyrażenie matematyczne. Prędkość przejazdu: zadawana poprzez rozkaz SDP.
MVR3 (Move R3)	Umożliwia przemieszczenie narzędzia roboczego robota po łuku. Figura rysowana jest począwszy od pozycji początkowej i kończy się w położeniu końcowym. Realizowany łuk jest na okręgu, o podanych współrzędnych środka.	MVR3 [Położenie początkowe],[Położenie końcowe],[środek okręgu] <warunek>	[Położenie] – położenie w układzie odniesienia związanego z podstawą lub dla bezpośredniego sterowania osiami. Stała całkowita lub rzeczywista, zmienna, wyrażenie matematyczne. Prędkość przejazdu: zadawana poprzez rozkaz SDP.

MVR (Move R)	Umożliwia przemieszczenie narzędzia roboczego robota po łuku. Figura rysowana jest począwszy od pozycji początkowej, poprzez położenie przejściowe i kończy się w położeniu końcowym. Jeśli aktualna pozycja jest inna niż położenie początkowe, to robot przemieści końcówkę do położenia początkowego ruchem z interpolacją liniową.	MVR [Położenie początkowe],[Położenie przejściowe],[Położenie końcowe] <warunek>	[Położenie] – położenie w układzie odniesienia związanego z podstawą lub dla bezpośredniego sterowania osiami. Stała całkowita lub rzeczywista, zmienna, wyrażenie matematyczne. Prędkość przejazdu: zadawana poprzez rozkaz SDP.
OADL (Optima Acceleration)	Automatycznie ustala optymalne przyspieszenie/opóźnienie ramienia robota.	OADL [Włącznik]	[Włącznik] – włącza się komendą ON, wyłącza OFF
ON COM GOSUB (On Communication Go Subroutine)	Umożliwia skok do podprogramu w momencie gdy pojawi się przerwanie na zdefiniowanym porcie komunikacyjnym oznaczonym numerem linii komunikacyjnej.	ON COM ([Numer linii komunikacyjnej]) GOSUB [Docelowa linia]	[Numer linii komunikacyjnej] – numer portu komunikacyjnego. [Docelowa linia] - miejsce w którym zaczyna się podprogram.
ON GOSUB (On Go Subroutine)	Skok do podprogramu w momencie gdy zmienna przyjmie określoną wartość.	ON [Nazwa zmiennej] GOSUB [Docelowa linia1], [Docelowa linia2],...	[Nazwa zmiennej] - zmienna, od której wartości wykonywane są różne podprogramy. [Docelowa linia] - miejsce w którym zaczyna się podprogram.
ON GOTO (On Go To)	Umożliwia skok do określonej linii w programie w momencie gdy zmienna przyjmie określoną wartość.	ON [Nazwa zmiennej] GOTO [Docelowa linia1], [Docelowa linia2],...	[Docelowa linia] – miejsce w którym zaczyna się dalsza część programu. [Nazwa zmiennej] - zmienna, od której wartości wykonywane są różne podprogramy.
OPEN	Umożliwia odczytanie informacji przychodzącej z portu komunikacyjnego i przypisanie jej zmiennej.	OPEN „[Nazwa portu komunikacyjnego] ” AS [#Numer portu]	[# Numer portu] – numer, do którego został przypisany port komunikacji. [Nazwa portu komunikacyjnego] – port komunikacyjny, który chcemy uaktywnić, wyróżniamy następujące porty komunikacyjne: COM1: - RS 232, COM2: - Ethernet (definiuje się go w parametrach sterownika) oraz COM3: - Ethernet (definiuje się go w parametrach sterownika).
OVRD (Override)	Umożliwia przekroczenie zakresu przy dochodzeniu ramienia do określonego punktu.	OVRD [Zakres]	[Zakres] – podawany w procentach 0 – 100 określa o ile można przekroczyć pozycję.

PLT (Pallet)	Umożliwia obliczenie ilości pozycji w palecie	PLT [Nr palety],[Operacja]	[Nr palety] – numer palety, liczba całkowita z przedziału od 1 do 8, liczba ta odpowiada numerowi palety zdefiniowanej w komendzie DEF PLT. [Operacja] – ustawienie ilości osi (stopni) dla danej palety.
PRINT (Print)	Umożliwia wysłanie informacji na port komunikacyjny.	PRINT [#Numer portu], [Nazwa zmiennej]	[# Numer portu] – numer, do którego został przypisany port komunikacji. [Nazwa zmiennej] – zmienna, do której zapisywane są informacje przychodzące na port komunikacyjny.
REM (Remarks)	Umożliwia dodanie własnych komentarzy.	REM [Komentarz]	[Komentarz] – dowolny łańcuch znakowy
RETURN (Release Mechanizm)	Umożliwia powrót z podprogramu wywołanego procedurą GOSUB do następnej linii po słowie GOSUB.	RETURN	
SERVO ON / OFF (Servo ON/OFF)	Steruje załączaniem i wyłączaniem serwonapędów napędzających wszystkie osie robota.	SERVO ON SERVO OFF	
SKIP (Skip)	Umożliwia skok do następnej linii	SKIP	10 MOV P1 WTHIF M_IN(17)=0,SKIP ‘Gdy na wejściu 17 pojawi się stan niski – skok do następnej linii
SPD (Speed)	Ustala prędkość przejazdu dla ruchu z interpolacją liniową lub kołową (rozkazy MVS, MVC, MVR, MVR2 itd.).	SPD [Prędkość]	[Prędkość] - liczba całkowita lub rzeczywista, prędkość podana jest w [mm/s]. W systemie parametrem M_NSPD zdefiniowano znamionową wartość prędkości przejazdu. Uwaga!!! W momencie uruchomienia robota prędkość przejazdu dla ruchu liniowego jest przyjęta na poziomie wartości znamionowej (parametr M_NSPD). Zmiany prędkości dokonuje się komendą SPD i jest ona ustawiana tylko dla komend występujących po tym rozkazie. Po zakończeniu programu (komenda END) prędkość automatycznie ustawiana jest na wartość znamionową.
TOOL (Tool)	Projektowe narzędzi konwersji danych	TOOL [Narzędzie konwersji danych]	[Narzędzie konwersji danych] - współrzędne, o jakie będą przekształcane podane wcześniej pozycje.
TORQ (Torque)	Ustala górną granicę momentu dla każdej z osi.	TORQ [Numer osi], [Wartość numeryczna]	[Numer osi] – liczba rzeczywista z zakresu 1-8. [Wartość numeryczna] – wartość momentu dla poszczególnej osi w [%]

WAIT (Wait)	Wstrzymuje działanie programu do momentu, aż pojawi się określony stan na zmiennej numerycznej typu M	WAIT [Zmienna numeryczna]=[Wartość liczbowa]	[Zmienna numeryczna] – zmienna numeryczna typu M (np.: M_IN(1), M_RUN, M_01,…..) [Wartość liczbowa] – wartość jaką powinna przyjąć M – zmienna
WHILE – WEND (While End)	Pętla warunkowa – dopóki spełniony jest warunek logiczny – wykonuj określone zadanie.	WHILE [Warunek logiczny] .... WEND	Przykład: 110 WHILE (M1>=-5) AND(M1<=5) ‘Jeżeli warunek spełniony skok do linii 20, jeżeli nie – skok do linii 50 120 M1=-(M1+1) 130 PRINT #1, M1 140 WEND 150 END
WTH (With)	Jeżeli chcemy wykonywać kilka czynności jednocześnie.	WITH [Proces]	Przykład: 10 MOV P1 WTH M_OUT(17)=1 DLY 10 ‘Robot rozpoczyna wykonywać ruch i równocześnie pojawia się stan wysoki na wyjściu 17 i trawa przez 10 s.
WTHIF (With If)	Jeżeli chcemy wykonywać kilka czynności jednocześnie, przy spełnionym określonym warunku należy użyć komendy WTHIF.	WITH [Proces]	Przykład: 10 MOV P1 WTHIF M_IN(17)=1, HLT ‘Robot wykonuje ruch, lecz w momencie pojawienia się na wejściu 17 stanu wysokiego zostaje wstrzymany cały proces.
LOAD (X Load)	Wczytuje program zapisany w odpowiednim slocie.	XLOAD [Numer slotu], “[Nazwa programu]”	[Numer slotu] – slot, w którym znajduje się uruchamiany program. [Nazwa programu] - nazwa programu, pod jaką zapisany jest w sterowniku.
XRUN (X Run)	Uruchomia programu zapisanego w odpowiednim slocie.	XRUN [Numer slotu], “[Nazwa programu]”, [Warunek]	[Numer slotu] – slot, w którym znajduje się uruchamiany program. [Nazwa programu] - nazwa programu, pod jaką zapisany jest w sterowniku. [Warunek]: 0 – tryb cyklicznej pracy, 1 – tryb pojedynczego wykonania programu.
XSTP (X Stop)	Zatrzymuje program zapisany w odpowiednim slocie.	XSTP [Numer slotu]	[Numer slotu] – slot, w którym znajduje się uruchamiany program.

XRST (X Reset)	Komenda powoduje restart programu w określonym slocie i powrót do początkowej linii.	XRST [Numer slotu]	[Numer slotu] – slot, w którym znajduje się uruchamiany program.

6 Literatura:

Opracowanie zostało przygotowane w oparciu o poniższą literaturę:

1. Marcin Kowal, „Zastosowanie robotów ramieniowych firmy Mitsubishi w procesie montażu prostych urządzeń elektrycznych”, Praca magisterska, Wrocław 2012
2. Kamil Florków, „Zastosowanie robotów przemysłowych firmy MITSUBISHI do automatyzacji wybranych procesów przemysłowych, Praca magisterska, Wrocław 2010
3. Waldemar Kanior, „Zastosowanie robota typu SCARA do automatyzacji wybranych procesów technologicznych”, Praca magisterska, Wrocław 2006
4. W. Henno “Sterowanie robotami przemysłowymi”, 2002;
5. Marwick Manufacturing Group “Introduction to industrial robots”;
6. Barbara Krasnoff “ROBOTS: REEL TO REAL”, 1982;
7. http://www.telemanipulators.com/