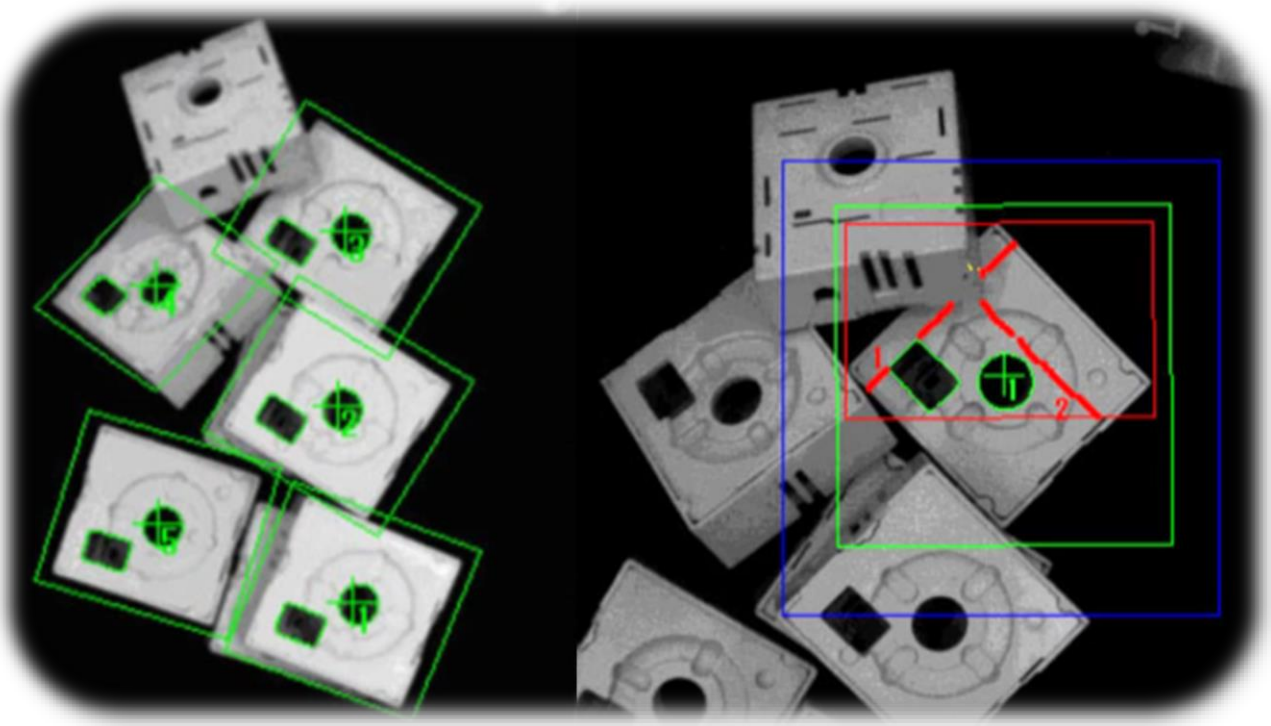
Podloge za vježbe

FANUC KAREL programski jezik

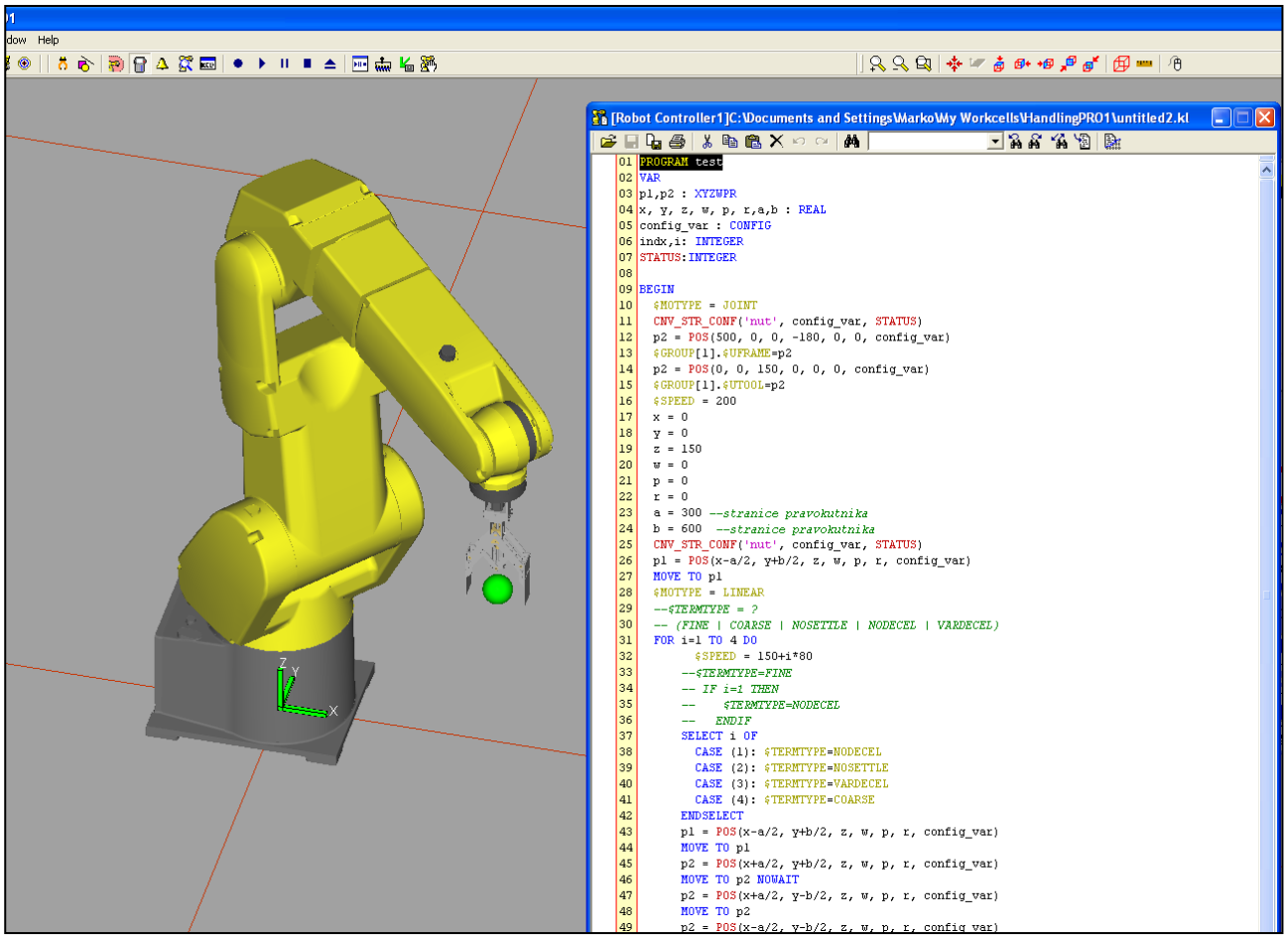


Ver. 3.11.2021.

Sadržaj

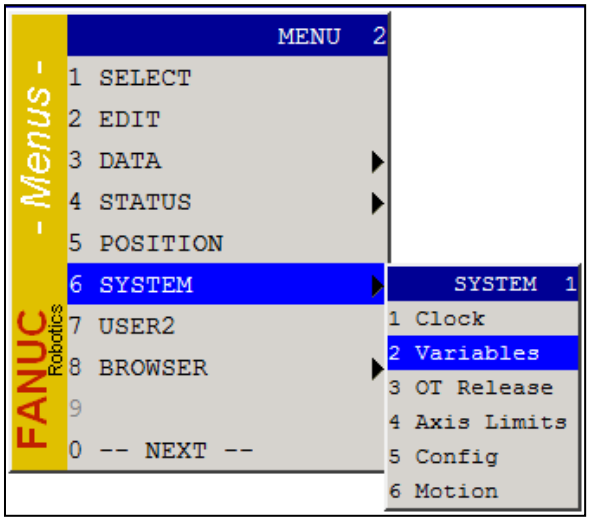
1. Programski jezik KAREL

Programski jezik koji se koristi za programiranje FANUC robota je KAREL. Pisanje programa u KAREL-u moguće je unutar programskom paketu *Roboguide* (Slika 1.1.). Program se može simulirati u virtualnom okruženju na računalu i nakon toga prevesti u strojni jezik te poslati izravno u upravljačku jedinicu robota. Struktura KAREL programa vrlo je slična programskom jeziku PASCAL.



Slika 1.1. Pisanje KAREL programa u Roboguide-u

Prilikom rada sa KAREL programima na robotu potrebno je varijabli *$KAREL_ENB* koja se nalazi u popisu sistemskih varijabli dodijeliti vrijednost 1. Na taj način omogućiti će se pokretanje KAREL programa na upravljačkoj jedinici.



Slika 1.2. Izbornik sistemskih varijabli;

1.1. Sintaksa programskog jezika KAREL

1.1.1. Organizacija programa

Prva ključna riječ koja se koristi u svakom KAREL programu je **PROGRAM**.
Sintaksa: PROGRAM *ime_programa*.
Ime programa može biti duljine najviše 8 znakova.
Nakon naredbe PROGRAM slijedi definiranje **varijabli** (VAR) te konstanti (CONST). Sintaksa za definiranje varijabli je sljedeća:

```
VAR
    a: INTEGER
    b: STRING
    …….
```

Konstante mogu biti jedino cjelobrojni brojevi (INTEGER). Sintaksa je ista kao i kod definiranja varijabli:

```
CONST
    Konstanta1: 25
```

Ovom sintaksom *Konstanta1* poprima vrijednost 25.

Moguće je nakon ključne riječi PROGRAM te prije deklaracije konstanti i varijabli koristiti određene **ključne riječi** koje određuju način izvršavanja programa i sl. Primjer najčešće korištenih ključnih riječi:

- %NOPAUSESHFT – Program se neće pauzirati ako se tipka SHIFT odpusti prilikom izvršavanja.
- %NOLOCKGROUP - Niti jedna grupa kretanja (*Motion group*) neće biti zaključana prilikom poziva programa. Koristi se prilikom izvođenja programa koji nemaju naredbe kretanja omogućava pokretanje programa u T1 ili T2 režimu rada bez kontinuirano pritisnute sigurnosne tipke (*Deadman switch*). Prilikom poziva programa u paralelnom režimu rada (RUN) potrebno je koristiti ovu ključnu riječ

- %COMMENT = 'Komentar' – Ispisuje komentar maksimalne duljine do 16 znakova uz program u SELECT izborniku upravljačke konzole.

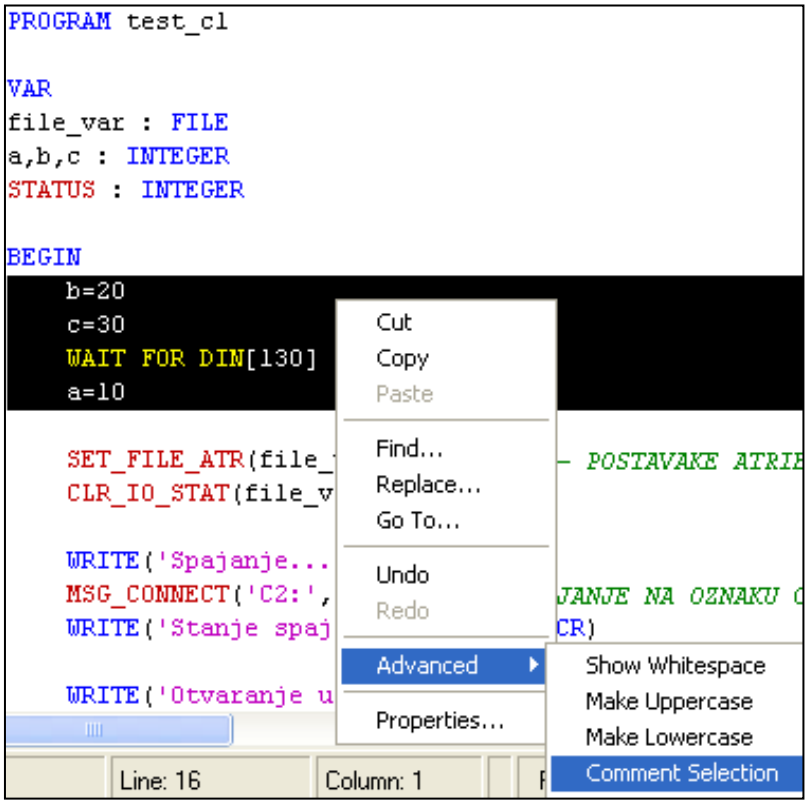
Početak glavnog dijela programa označava se naredbom **BEGIN**. Nakon ove naredbe započinje glavni dio programa u kojem se izvršavaju naredbe, pozivaju potprogrami, rutine, upravlja robotom, vrši komunikacija i sl.

Kraj programa označava se sa ključnom riječi **END**.
Sintaksa: END ime_programa
Nakon poziva naredbe END program završava te više nije moguće pozivati naredbe.

Pisanje **većeg broja naredbi** u jednom programskom redu razdvaja se znakom ; (točka zarez). Broj znakova koje je moguće napisati u jedan red KAREL programa je ograničen te iznosi 126.

Primjer: i=5; j=7; IF i>j THEN; WRITE (i); ENDIF

Komentari se u KAREL programu mogu pisati bilo u glavnom dijelu programa ili između naredbe PROGRAM i BEGIN.
Sintaksa: -- Komentar
Na slici 1.3. prikazan je način kako se veći dio programa može postaviti kao komentar ili ukloniti komentar. Naredbe su *Comment selection* te *Uncomment selection.*



Slika 1.3. Postavljanje komentara u programskom paketu *Roboguide*

1.1.2. Tipovi varijabli

Programski jezik KAREL kao i računalni programski jezici koristi više različitih tipova varijabli. Mnogi od tipova varijabli slični su sa većinom drugih programskih jezika; INTEGER(cjelobrojni brojevi), REAL (decimalni brojevi), BOOLEAN (varijabla koja poprima vrijednost istina ili laž). No za razliku od računalnih programski jezika programski jezik KAREL koristi tipove varijabli koje su specifične za rad sa robotima.
U opisu varijabli, a kasnije naredbi koristiti će se navedeno označavanje. Varijable su napisane *kurzivom,* dok su neobavezni dijelovi u sintaksi stavljeni u <izlomljene zagrade>.

1.1.2.1. INTEGER

Varijabla INTEGER u programskom jeziku KAREL poprima vrijednosti od najmanjeg negativnog cijelog broja do najvećeg pozitivnog cijelog broja:

- Najmanji cijeli broj: -2147483648
- Najveći cijeli broj: 2147483646

Za pregled stanja određene naredbe koristi se predefinirana varijabla STATUS koja se mora deklarirati kao tip INTEGER. Nakon pozvane naredbe se u varijablu STATUS zapisuje cjelobrojni broj:

- STATUS = 0 – naredba je uspješno izvršena
- STATUS <> 0 – naredba je neuspješno izvršena te vrijednost varijable status predstavlja kodni broj greške

1.1.2.2. REAL

Varijabla REAL poprima vrijednosti decimalnih brojeva i to:

- najmanji decimalni broj: $-3{,}4028236 \cdot 10^{38}$
- najveći decimalni broj: $3{,}4028236 \cdot 10^{38}$

Svaki decimalni broj koji se koristi u KAREL programu mora biti zaokružen na 7 decimalnih mjesta inače program javlja grešku.

1.1.2.3. STRING

String (kombinacija slova, brojeva i posebnih znakova) je varijabla kojoj se mora predodrediti koliko znakova će imati.
Sintaksa: *str*: STRING[20]
Varijabla *str* ima preddefiniranih 20 mjesta za pohranu znakova.

1.1.2.4. BOOLEAN

Bulova varijabla (eng. *Boolean*) može poprimiti vrijednost TRUE (istina) ili FALSE (*laž*). Također unutar KAREL programa 0 predstavlja istinu dok svaki drugi broj označava *laž* tj. grešku.

1.1.2.5. XYZWPR

XYZWPR varijabla je koja može poprimiti 6 decimalnih vrijednosti. XYZ predstavljaju 3 translacije dok WPR su tri rotacije u prostoru.
Za rad samo sa jednom od koordinata varijable ovog tipa potrebno je u programu specificirati sljedeće:

pozicijska_var.X, pozicijska_var.Y, pozicijska_var.Z, …, pozicijska_var.R

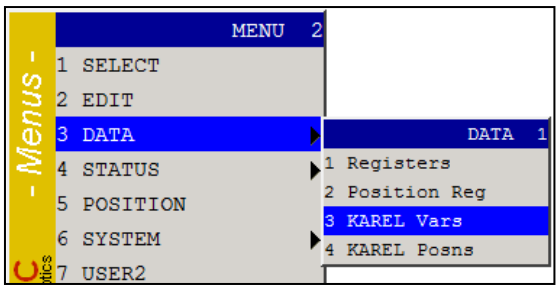
Pojedinoj koordinati moguće je promijeniti vrijednost ili koristiti vrijednost iste za određenu operaciju.

1.1.2.6. CONFIG

Varijabla vezana uz konfiguraciju robota. Najčešća konfiguracija robotske ruke je 'NUT000'.

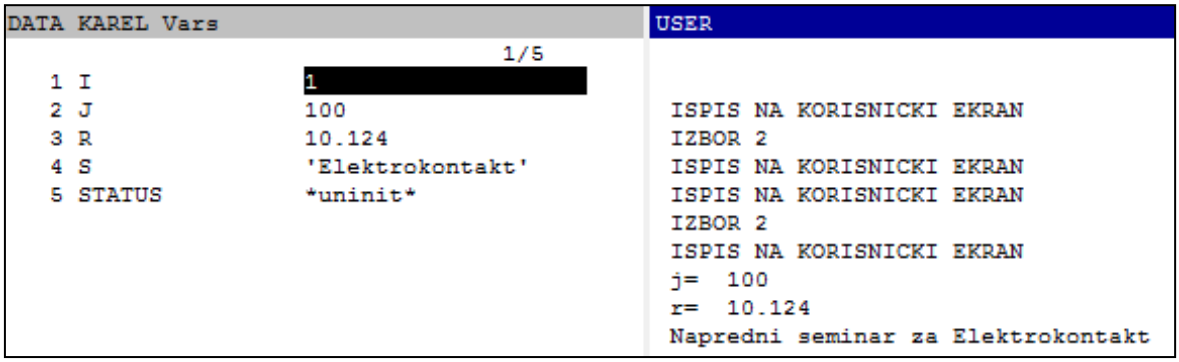
1.1.2.7. Pregled KAREL varijabli pomoću upravljačke jedinice

Na upravljačkoj jedinici robota moguće je sa izbornika *DATA→KAREL Vars* imati pregled trenutnog stanja KAREL varijabli u upravljačkoj jedinici robota:

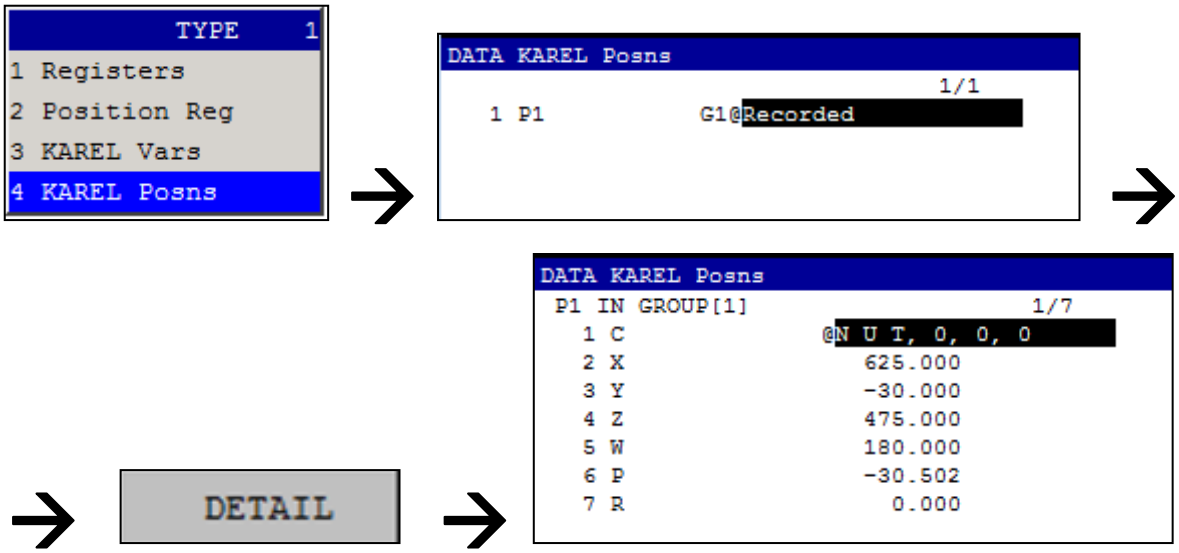


Slika 1.4. Izbornik KAREL varijabli

Pregled stanja varijabli moguće je isključivo sa lijevog ekrana upravljačke konzole. Iz tog razloga prethodno pokretanju KAREL programa potrebno ga je označiti u izborniku *SELECT* također sa lijevog ekrana te nakon toga prikazati stanje *DATA KAREL Vars*. Na desnom ekranu moguće je prikazati korisnički ekran (*USER*) na koji se mogu ispisati poruke iz KAREL programa.



Slika 1.5. Pregled ekrana upravljačke konzole – Lijevo: trenutno stanje KAREL varijabli aktivnog KAREL programa; Desno: Korisnički ekran



Slika 1.6. Pregled ekrana upravljačke konzole – Pozicijske KAREL varijable

1.1.3. Sistemske varijable

Sistemske varijable (eng. *system variable*) su preddefinirane u upravljačkoj jedinici robota. Neke od sistemskih varijabli mogu se koristiti u programima upravljačke konzole. U KAREL programima omogućen je rad sa većinom sistemskih varijabli.

OPREZ!!!

Promjena određenih sistemskih varijabli može utjecati na rad robota te može uzrokovati grešku u radu. Mijenjati se smiju samo one varijable za koje je poznat učinak na rad robota.

Sintaksa za pridruživanje vrijednosti sistemskoj varijabli ista je kao i za bilo koju drugu varijablu. Sistemske varijable započinju znakom $ te su napisane velikim slovima bez razmaka.

Sintaksa: $SISTEMSKA_VARIJABLA = vrijednost_sistemske_varijable

1.1.3.1. $MOTYPE

Sistemska varijabla $MOTYPE može poprimiti vrijednosti:

- LINEAR – način gibanja od početne do krajnje točke je linearan po pravcu
- CIRCULAR – kružno gibanje
- JOINT – gibanje interpolacijom zglobova

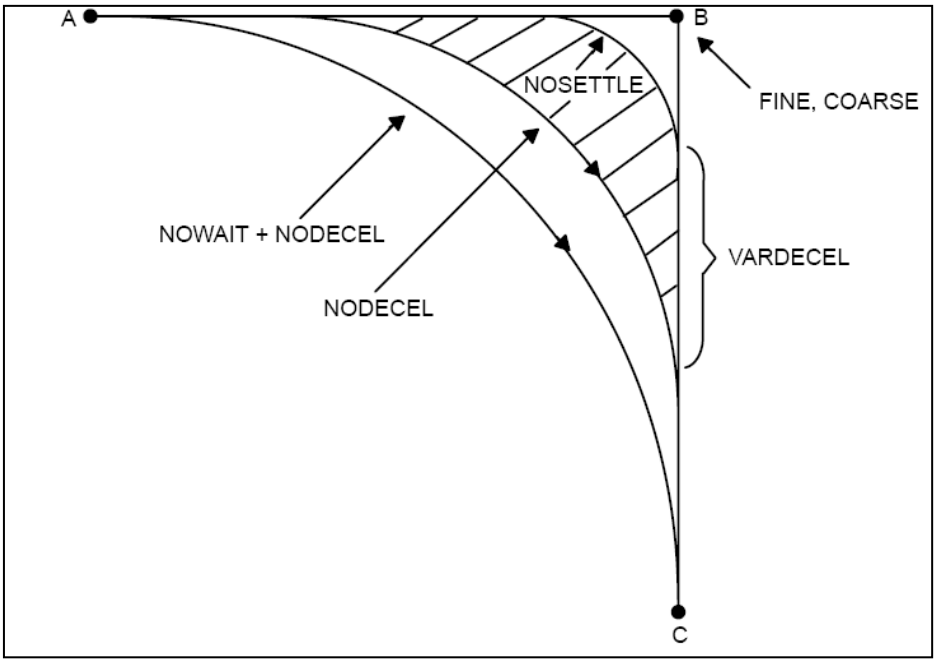
Način gibanja određuje putanju od početne do krajnje točke.

1.1.3.2. $TERMTYPE

Sistemska varijabla koja određuje način izvršavanja tog gibanja tj. vremena i brzinu ubrzanja (eng. *acceleration*, u daljnjem tekstu akceleracija) i usporenja (eng. *deceleration*, u daljnjem tekstu deceleracija)
Vrijednosti koje sistemska varijabla $TERMTYPE može poprimiti su:

- FINE
- VARDECEL
- NODECEL
- CLEAR
- …..

Prikaz načina izvođenja pojedinog gibanja od točke A do C kroz točku B sa različitim vrijednostima varijable $TERMTYPE prikazani su na slici 1.7.



Slika 1.7. Način izvođenja gibanja – sistemska varijabla $TERMTYPE

Sintaksa: $TERMTYPE = vrijednost
Primjer: $TERMTYPE = FINE

1.1.3.3. $SPEED

Sistemska varijabla $SPEED određuje brzinu izvršavanja pojedinog gibanja. Brzina se definira kao x mm/s kod linearnog li kružnog gibanja.

1.1.3.4. $UFRAME

Varijabla $UFRAME koristi se za definiranje pozicija i konfiguracije korisničkog koordinatnog sustava. $UFRAME varijabla ne briše niti jedan korisnički koordinatni sustav već kreira novi koji je isključivo vezan uz KAREL program.
Sintaksa: $UFRAME = *p1*
Gdje točka *p1* sadrži podatke o poziciji, orijentaciji te konfiguraciji koordinatnog sustava.

1.1.3.5. $UTOOL

Varijabla $TOOL koristi se za definiranje pozicija i konfiguracije koordinatnog sustava alata. $TOOL varijabla ne briše niti jedan korisnički koordnatni sustav već kreira novi koji je isključivo vezan uz KAREL program.
Sintaksa: $UTOOL = *p2*
Gdje točka *p2* sadrži podatke o poziciji, orijentaciji te konfiguraciji koordinatnog sustava.

1.1.3.6. $FASTCLOCK

Ova varijabla ne može se mijenjati već se može koristiti za mjerenje proteklog vremena. Mjera razlučivanja ove varijable je minimalno 8 *ms* u koracima od *8 ms*. Pomoću ove varijable moguće je primjerice mjeriti koliko vremenski traje neki blok naredbi, određeno gibanje i slično.

1.1.3.1. $MCR_GRP[1].$PRGOVERRIDE

Varijabla koja kontrolira sveukupnu brzinu robota prilikom izvođenja programa. Početna vrijednost ove varijable je 100,000. Ukupna brzina robota prilikom izvođenja TP i KAREL programa jednaka je umnošku $MCR[1].GRNOVERRIDE × $MCR_GRP[1].$PRGOVERRIDE. Razlučivost varijable $MCR_GRP[1].$PRGOVERRIDE je 0.001 dok je raspon od [0.001, 100.000]

1.1.3.2. $MCR.$GENOVERRIDE

Brzina koja se prikazuje na ekranu teach pendanta. Razlučivost je 1% a raspon od 1% do 100%.

1.1.3.3. $SCR.$ITP_TIME

Određuje korak između dvije naredbe gibanja. Minimalna vrijednost je 4 [*ms*] a maksimalna 124 [*ms*],

1.2. Tijek programa

Svaki KAREL program izvodi se sekvencijalno red po red od početka do kraja programa; od naredbe BEGIN do naredbe END. Takav tijek izvođenja može se mijenjati uz pomoć odgovarajućih upravljačkih naredbi. Jedna takva naredba je **GOTO** – naredba za bezuvjetan skok na određenu oznaku u programu.
Sintaksa GOTO oznaka

…..
oznaka

Naredba **CALL PROG** poziva drugi program koji se tada počinje izvoditi. Nakon završetka pozvanog programa nastavlja se izvoditi programski kod glavnog programa od sljedećeg reda u programu. Pozivati se mogu:

- Programi upravljačke konzole sa ekstenzijom .TP
- Programi upravljačke konzole sa ekstenzijom .MR
- Drugi KAREL programi sa ekstenzijom .PC

Prilikom poziva programa ne definira se koja vrsta programa se poziva. Zato u upravljačkoj jedinici robota ne bi smjeli postojati .TP i .PC programi istog imena.
Sintaksa: CALL PROG(*ime_programa*, *indx*)
Ime_programa: STRING
indx: INTEGER

Naredba koja onemogućava izvršavanje daljnjeg tijeka programa je naredba **WAIT FOR** x. X predstavlja primjerice određeni digitalni ulaz te se program ne izvodi dalje sve dok se signal tog digitalnog ulaza ne upali.

Naredba **DELAY** koristi se za pauziranje programa tj. njegovo odgođeno izvršavanje određen broj vremenskih jedinica.
Sintaksa: DELAY 1000,
Gdje 1000 predstavlja čekanje od 1000 *ms*.
ABORT naredba prekida program. Program se više ne može nastaviti.
PAUSE naredba zaustavlja program do poziva naredbe
CONTINUE nastavlja izvršavanje programa

1.2.1. Operacije

Popis mogućih aritmetičkih, logičkih i relacijskih operacija:

Operacija	Operator					
Aritmetička	+	-	*	/	DIV	MOD
Relacijska	<	<=	=	<>	>=	>
Logička	AND	OR	NOT			

1.2.2. Ulazno izlazni sustav

1.2.2.1. Digitalni signali

Sintaksa za korištenje određenog digitalnog ulaznog ili izlaznog signala unutar KAREL programa:

- *DIN[x]* – digitalni ulaz DI[x]
- *DOUT[x]* – digitalni izlaz DO[x]

1.2.2.2. Robotski signali

Sintaksa za korištenje određenog robotskog ulaznog ili izlaznog signala unutar KAREL programa:

- *RDI[x]* – digitalni ulaz RI[x]
- *RDO[x]* – digitalni izlaz RO[x]

OPREZ!!!
Promjena robotskih ili digitalnih signala izravno iz KAREL programa može utjecati na rad fizičkih komponenata u sustavu.

1.2.2.3. Tipke upravljačke konzole

Ključna riječ TPIN[n] koristi se za očitavanje stanja pojedine tipke upravljačke konzole. Vrijednost *TRUE* predstavlja da je tipka pritisnuta dok *FALSE* označava da tipka nije pritisnuta.

Prekidači upravljačke konzole	
TPIN[250]	EMERGENCY STOP
TPIN[249]	ON/OFF prekidač
TPIN[247]	Desna sigurnosna (DEADMAN) tipka
TPIN[248]	Lijeva sigurnosna (DEADMAN) tipka
Strelice	
TPIN[212]	Strelica gore
TPIN[213]	Strelica dolje
TPIN[208]	Strelica desno
TPIN[209]	Strelica lijevo
TPIN[0]	Lijeva ili desna tipka SHIFT
TPIN[204]	Strelica gore + SHIFT
TPIN[205]	Strelica dolje + SHIFT
TPIN[206]	Strelica desno + SHIFT
TPIN[207]	Strelica lijevo + SHIFT
Numeričke tipke (sa i bez tipke SHIFT)	
TPIN[13]	ENTER
TPIN[8]	BACK SPACE
TPIN[48]	0
TPIN[49]	1
TPIN[50]	2
TPIN[51]	3
TPIN[52]	4
TPIN[53]	5
TPIN[54]	6
TPIN[55]	7
TPIN[56]	8
TPIN[57]	9
Funkcijske tipke	
TPIN[128]	PREV
TPIN[129]	F1
TPIN[131]	F2
TPIN[132]	F3
TPIN[133]	F4
TPIN[134]	F5
TPIN[135]	NEXT
TPIN[136]	PREV + SHIFT
TPIN[137]	F1 + SHIFT
TPIN[138]	F2 + SHIFT
TPIN[139]	F3 + SHIFT
TPIN[140]	F4 + SHIFT
TPIN[141]	F5 + SHIFT
TPIN[142]	NEXT + SHIFT
Tipke izbornika	
TPIN[143]	SELECT
TPIN[144]	MENU
TPIN[145]	EDIT
TPIN[146]	DATA
TPIN[147]	FCTN
TPIN[148]	ITEM
TPIN[149]	+%
TPIN[150]	-%
TPIN[151]	HOLD
TPIN[152]	STEP
TPIN[153]	RESET
TPIN[240]	DISP
TPIN[203]	HELP
TPIN[154]	ITEM + SHIFT
TPIN[155]	+% + SHIFT
TPIN[156]	-% + SHIFT
TPIN[157]	HOLD + SHIFT
TPIN[158]	STEP + SHIFT
TPIN[159]	RESET + SHIFT
TPIN[227]	DISP + SHIFT
TPIN[239]	HELP + SHIFT
Funkcijske tipke	
TPIN[173]	Korisnička tipka 1
TPIN[174]	Korisnička tipka 2
TPIN[175]	Korisnička tipka 3
TPIN[176]	Korisnička tipka 4
TPIN[177]	Korisnička tipka 5
TPIN[178]	Korisnička tipka 6
TPIN[210]	Korisnička tipka 7
TPIN[179]	Korisnička tipka 1 + SHIFT
TPIN[180]	Korisnička tipka 2 + SHIFT
TPIN[181]	Korisnička tipka 3 + SHIFT
TPIN[182]	Korisnička tipka 4 + SHIFT
TPIN[183]	Korisnička tipka 5 + SHIFT
TPIN[184]	Korisnička tipka 6 + SHIFT
TPIN[211]	Korisnička tipka 7 + SHIFT

Tipke za pokretanje robota		
TPIN[185]		FWD
TPIN[186]		BWD
TPIN[187]		COORD
TPIN[194]	TPIN[188]	+X
TPIN[195]	TPIN[189]	+Y
TPIN[196]	TPIN[190]	+Z
TPIN[197]	TPIN[191]	+X rotacija
TPIN[198]	TPIN[192]	+Y rotacija
TPIN[199]	TPIN[193]	+Z rotacija
TPIN[226]		FWD + SHIFT
TPIN[207]		BWD + SHIFT
TPIN[202]		COORD + SHIFT
TPIN[220]	TPIN[214]	+X + SHIFT
TPIN[221]	TPIN[215]	+Y + SHIFT
TPIN[222]	TPIN[216]	+Z + SHIFT
TPIN[223]	TPIN[217]	+X rotacija + SHIFT
TPIN[224]	TPIN[218]	+Y rotacija + SHIFT
TPIN[225]	TPIN[219]	+Z rotacija + SHIFT

1.2.3. Uvjetne naredbe

1.2.3.1. IF…THEN…ELSE naredba

Osnovna IF…THEN…ELSE naredba ima dvije forme:
Sintaksa:
IF uvjet THEN
    Blok naredbi
ENDIF
Blok naredbi izvodi se samo ako je *uvjet* zadovoljen

Drugi oblik naredbe je sljedeći:
IF uvjet THEN
    Prvi blok naredbi
ELSE
    Drugi blok naredbi
ENDIF
Prvi blok naredbi izvršava se samo ako je *uvjet* zadovoljen, ako uvjet nije zadovoljen izvodi se *Drugi blok naredbi*.
Svi uvjeti se izražavaju logičkim funkcijama.

1.2.3.1. SELECT naredba

SELECT naredba ima sljedeću sintaksu:

```
SELECT ulazna_var OF
  CASE (1):
      --blok naredbi koji se izvršava ako je ulazna_var = 1
  CASE (2):
      --blok naredbi koji se izvršava ako je ulazna_var = 2
  ELSE:
      --blok naredbi koji se izvršava ako je ulazna_var <> 1 i ulazna_var <> 2
ENDSELECT
```

Varijabla *ulazna_var* ne može biti deklarirana kao tip REAL.

1.2.4. Petlje

1.2.4.1. FOR PETLJA

Sintaksa FOR naredbe:
FOR *brojac* = *pocetak* to *kraj* do
    Blok naredbi
ENDFOR

Brojac cjelobrojni broj koji definira broj ciklusa FOR petlje
Pocetak definira početnu vrijednost varijable *brojac*
Kraj definira završnu vrijednost varijable *brojac*
Izvođenje bloka naredbi unutar FOR petlje završava kada varijabla *brojac* postigne završnu vrijednost. Inkrement varijable *brojac* je 1.

1.2.4.2. WHILE PETLJA

While petlja izvršava blok naredbi sve dok je uvjet zadovoljen. Provjera uvjeta nalazi se na početku petlje.

```
WHILE uvjet
DO
        Blok naredbi
ENDWHILE
```

1.2.4.1. REPEAT UNTIL

Repeat until petlja izvršava blok naredbi sve dok je uvjet zadovoljen. Provjera uvjeta nalazi se na kraju petlje.

```
REPEAT
        Blok naredbi
UNTIL uvjet
```

1.2.5. Naredbe za rad sa numeričkim i registrima položaja

1.2.5.1. SET_POS_REG ()

Naredba koja se koristi za spremanje pozicije unutar pozicijskog registra upravljačke jedinice robota. Stanje izvršene naredbe sprema se u varijablu *status.*
Sintaksa: SET_POS_REG (*poz_registar_broj, pozicija, status,* <*broj_grupe*>)
Ulazni/izlazni parametri

- [ulazni] *poz_registar_broj*: INTEGER
- [ulazni] *pozicija*: XYZWPR
- [izlazni] *status*: INTEGER
- <izlazni> *broj_grupe*: INTEGER

Primjer: SET_POS_REG (5, *trenutna_pozicija, status*)
U primjeru se u pozicijski registar PR[5] upisuje vrijednost varijable *trenutna_pozicija* (u koju je zapisana trenutna pozicija – pogledaj naredbu CUR_POS()

1.2.5.2. GET_POS_REG ()

Naredba kojom se u zadanu varijablu (*pozicija* : XYZWPR) zapisuju podaci pozicijskog registra upravljačke jedinice robota. Stanje izvršene naredbe sprema se u varijablu *status.*
Sintaksa: GET_POS_REG (*poz_registar_broj, status, <broj_grupe>*)
Ulazni/izlazni parametri

- [ulazni] *poz_registar_broj*: INTEGER
- [izlazni] *status*: INTEGER
- <ulazni> *broj_grupe*: INTEGER

Primjer: *pozicija* = GET_POS_REG (10, *status*)
U primjeru se u varijablu *pozicija* zapisuju podaci iz pozicijskog registra PR[10].

1.2.5.3. SET_REAL_REG ()

Naredba kojom se u određeni registar zapisuje decimalan broj. Stanje izvršene naredbe sprema se u varijablu *status.*
Sintaksa: SET_REAL_REG (*registar_broj, decimalan_broj, status*)
Ulazni/izlazni parametri

- [ulazni] *registar_broj*: INTEGER
- [ulazni] *decimalan_broj*: REAL
- [izlazni] *status*: INTEGER

Primjer: SET_REAL_REG (1, *broj, status*)
U primjeru se u registar R[1] zapisuje vrijednost decimalnog broja *broj*.

1.2.5.4. SET_INT_REG ()

Naredba kojom se u određeni registar zapisuje cjelobrojni broj. Stanje izvršene naredbe sprema se u varijablu *status.*
Sintaksa: SET_INT_REG (*registar_broj, decimalan_broj, status*)
Ulazni/izlazni parametri

- [ulazni] *registar_broj*: INTEGER
- [ulazni] *decimalan_broj*: INTEGER
- [izlazni] *status*: INTEGER

Primjer: SET_INT_REG (25, *broj, status*)
U primjeru se u registar R[25] zapisuje vrijednost cjelobrojnog broja *broj*.

1.2.5.5. GET_REG ()

Naredba kojom se u definiranu varijablu zapisuje vrijednost određenog registra. Stanje izvršene naredbe sprema se u varijablu *status.*
Sintaksa: GET_REG (*registar_broj, zastava, cjelobrojan_broj, decimalan_broj, status*)

Ulazni/izlazni parametri

- [ulazni] *registar_broj*: INTEGER
- [izlazni] *zastava*: BOOLEAN
- [izlazni] *cjelobrojan_broj*: INTEGER
- [izlazni] *decimalan_broj*: INTEGER
- [izlazni] *status*: INTEGER

Za zadani registar (*registar_broj*) provjerava se da li je njegova vrijednost cjelobrojan ili decimalan broj. Ako je decimalan onda *zastava* ima vrijednost TRUE i vrijednost iz registra se zapisuje u varijablu *decimalan_broj*. Ako je vrijednost varijable *zastava* FALSE tada se vrijednost registra zapisuje u varijablu *cjelobrojan_broj*.

Primjer: GET_REG (19, *zastava ,cj_broj, dec_broj, status*) ; vrijednost R[19] = 30

U primjeru se iz registra R[19] zapisuje vrijednost u varijablu *cj_broj*.

1.2.6. Naredbe orijentirane gibanju

1.2.6.1. CNV_STR_CONF ()

Naredba koja se koristi za pretvaranje string izraza konfiguracije robota (*ulazni_string*) u varijablu tipa CONFIG (*izlazna_konfiguracija*). Stanje izvršene naredbe sprema se u varijablu *status.*

Sintaksa: CNV_STR_CONF (*ulazni_string, izlazna_konfiguracija, status*)

Ulazni/izlazni parametri

- [ulazni] *ulazni_string*: STRING
- [izlazni] *izlazna_konfiguracija*: INTEGER
- [izlazni] *status*: INTEGER

Primjer: CNV_STR_CONF ('NUT000', *izlazna_konfiguracija, status*)

1.2.6.2. CUR_POS ()

Naredba koja se koristi za dobivanje podataka o trenutnoj poziciji vrha središta alata robota.

Sintaksa *trenutna_pozicija* = CUR_POS (0,0)

Gdje je *trenutna_pozicija* varijabla tipa XYZWPR.

1.2.6.3. MOVE TO ()

Naredba MOVE TO koristi se za pokretanje vrha središta alata robota u zadanu točku. Prije poziva ove naredbe potrebno je postaviti sljedeće parametre:

- Izbor korisničkog koordinatnog sustava ($UFRAME)
- Izbor koordinatnog sustava alata ($UTOOL)
- Brzina ($SPEED)

- Način gibanja ($MOTYPE)
- Izvršavanje gibanja ($TERMTYPE)

Primjer:

```
$GROUP[1].$UFRAME = kord_home
$GROUP[1].$UTOOL = ut_skare
$GROUP[1].$MOTYPE = LINEAR
$GROUP[1].$TERMTYPE = FINE
$GROUP[1].$SPEED = 200
MOVE TO p1 <NOWAIT>
Ostale_naredbe
```

Točka središta alata pomaknuti će se u točku *p1* i to linearno (LINEAR), brzinom (SPEED) 200 mm/s te će se zaustaviti (FINE) u točci *p1*. Tek kada se vrh središta alata zaustavi u točci *p1* počinju se izvoditi *Ostale_naredbe.*

Dodatno može se koristiti sintaksa: MOVE TO *p1* NOWAIT. Korištenjem ključne riječi NOWAIT ne dolazi do čekanja nakon naredbe MOVE TO *p1* NOWAIT već se *Ostale_naredbe* izvode paralelno sa naredbom gibanja. *Ostale_naredbe* izvode se do nove naredbe gibanja. Tek kada se točka središta alata dovede u točku *p1* može se započeti nova naredba gibanja.

Ovakvo paralelno procesuiranje omogućeno je jer se naredbe kretanja (izvode se u okolini gibanja, eng. *motion environment*) izvode odvojeno od naredbi za koje je zadužen prevoditelj programa (eng. *interpreter*).

1.2.7. Ostale naredbe

1.2.7.1. WRITE

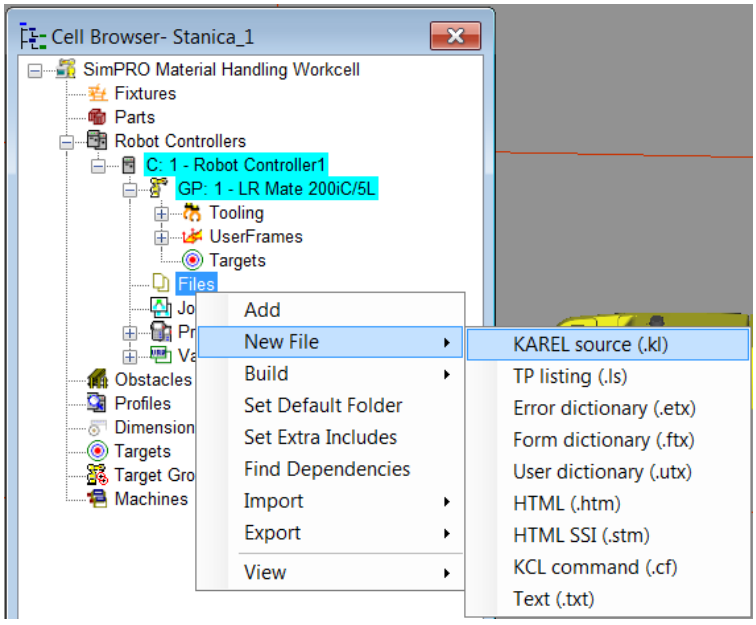
Pomoću WRITE naredbe moguće je poslati podatke određenim komunikacijskim kanalom. Najjednostavnija primjena služi za ispis poruke na korisnički ekran (*USER SCREEN).* Navigacija na korisnički ekran je moguća pritiskom na tipku MENU→USER SCREEN na upravljačkoj konzoli. Ključna riječ CR označava prelazak u novi red.

- WRITE('Poruka',CR) → ispis poruke *'Poruka*' na korisnički ekran
- WRITE (r::5:2,CR) → ispis realnog broja r na sveukupno 5 mjesta sa 2 mjesta iza decimalne točke
- WRITE ('Vrijednost varijable i:', i, CR)

1.3. Kreiranje jednostavnog programa

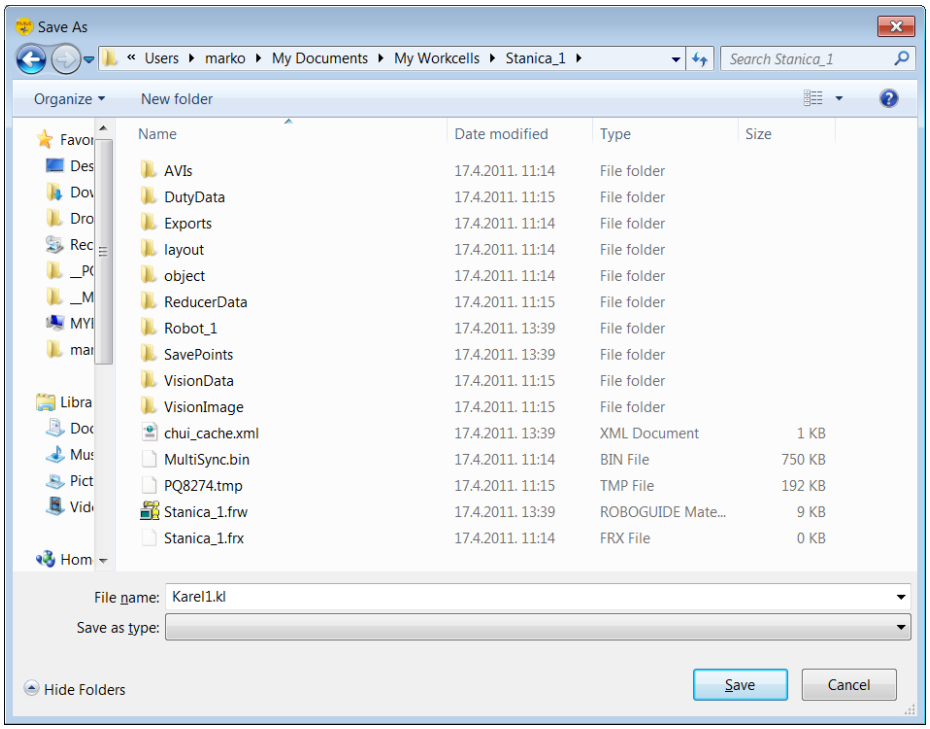
1.3.1. Pisanje programa

Za kreiranje Karel programa potrebno je iz preglednika robotske stanice pritisnuti (d) na *Files* te izabrati KAREL source.



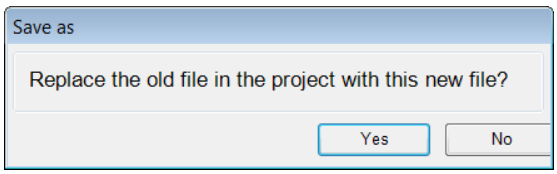
Slika 1.8. Dodavanje nove KAREL datoteke

Otvaranjem editora potrebno je sačuvati KAREL datoteku  . Datoteku je potrebno imenovati sa ekstenzijom kl.



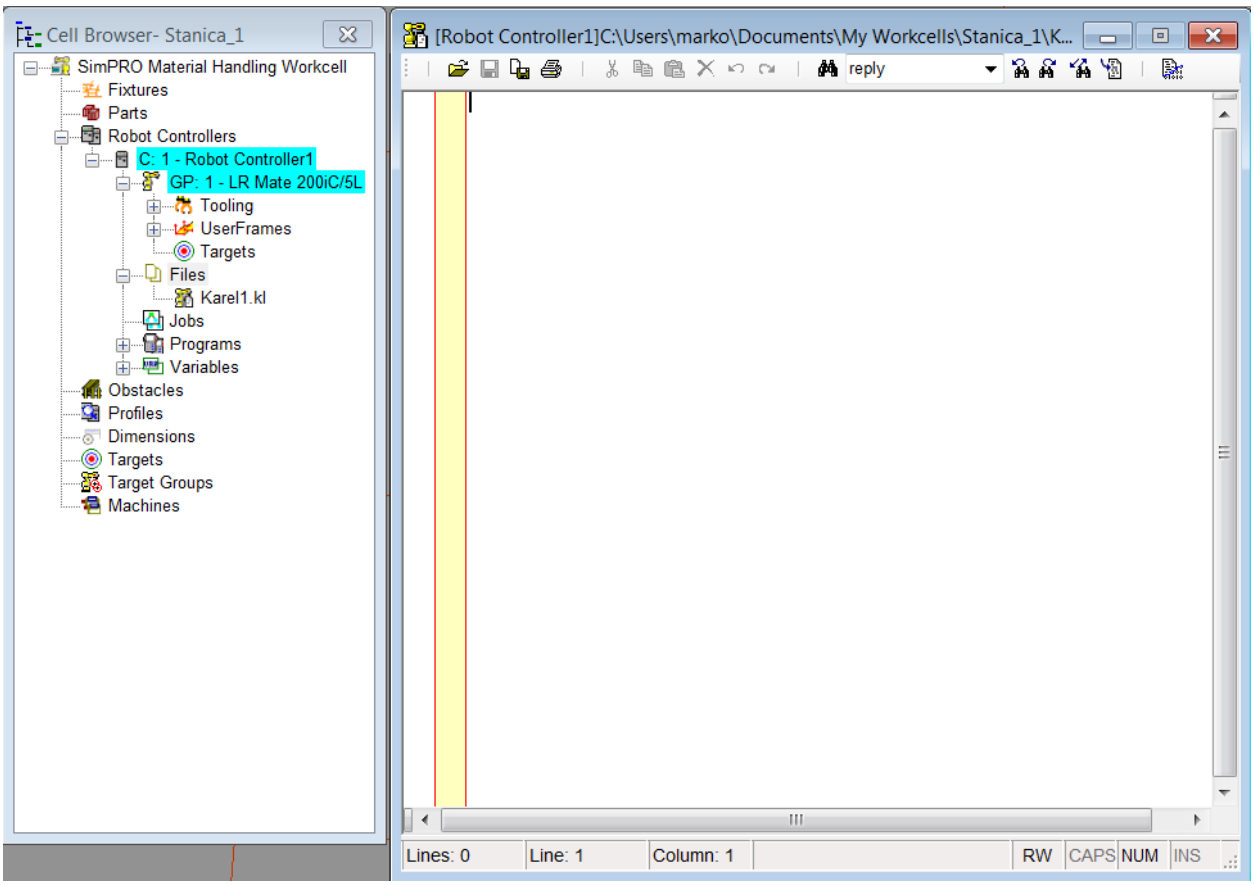
Slika 1.9. Spremanje novog KAREL programa

Na upit potrebno je odgovoriti sa *Yes*.



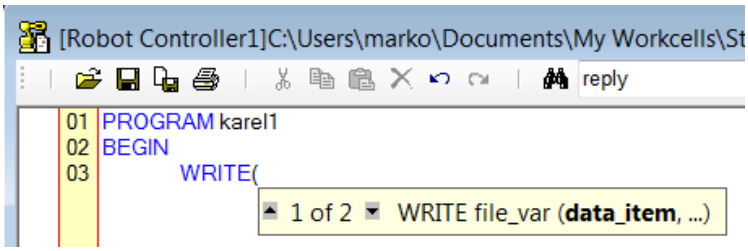
Slika 1.10. Upit

Nakon uspješnog kreiranja nove KAREL datoteke preglednik bi trebao izgledati kao na sljedećoj slici. Uz njega treba biti otvoren editor za KAREL datoteku.



Slika 1.11. Sučelje za pisanje KAREL programa

Jednostavan program koji na ekran upravljačke konzole ispisuje *Prvi program* prikazan je na sljedećoj slici. Prilikom upisivanja ključne riječi uključuje se kontekstualni izbornik koji daje pregled potrebnih parametara zadane naredbe te njihove varijante.

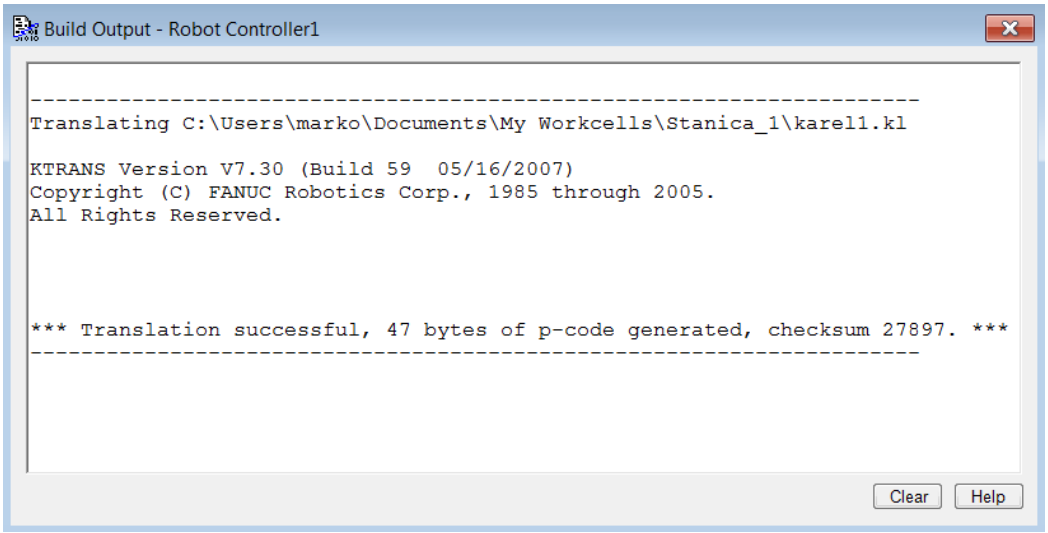


Slika 1.12. Virtualna upravljačka konzola – osnovna

Sintaksa programa:

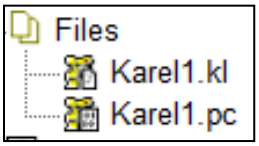
```
PROGRAM karel1
BEGIN
        WRITE('Prvi program',CR)
END karel1
```

Pritiskom na tipku  (*Build*) pojavljuje se sljedeća obavijest ukoliko nije bilo grešaka u sintaksi.

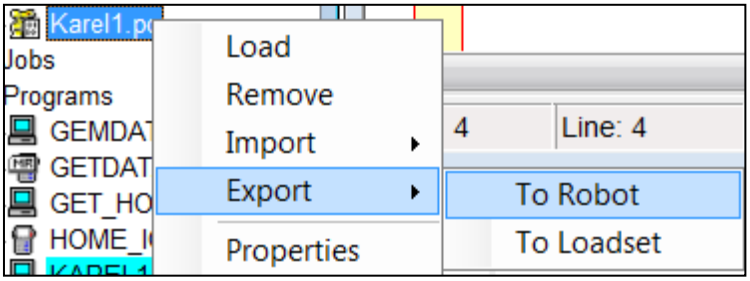


Slika 1.13. Kompajliranje programa – bez grešaka u sintaksi

Sada se u pregledniku robotske stanice nalazi nova datoteka sa ekstenzijom .pc. Ovo je datoteka koju je potrebno prenijeti (eksportirati) u upravljačku jedinicu realnog robota.

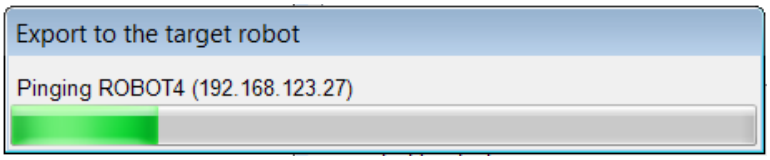


Slika 1.14. pc datoteka



Slika 1.15. Kopiranje prevedenog programa na upravljaču jedinicu realnog robota - izbor

Pritiskom (d) na *Export* te zatim na *To Robot* omogućeno je kopiranje pc datoteke na realnu upravljačku jedinicu. Pritiskom na tipku te zatim izborom željenog robota u robotskom susjedstvu (*Robot Neighborhood*) te potvrdom na tipku *Export* označena pc datoteka kopira se na upravljačku jedinicu robota.



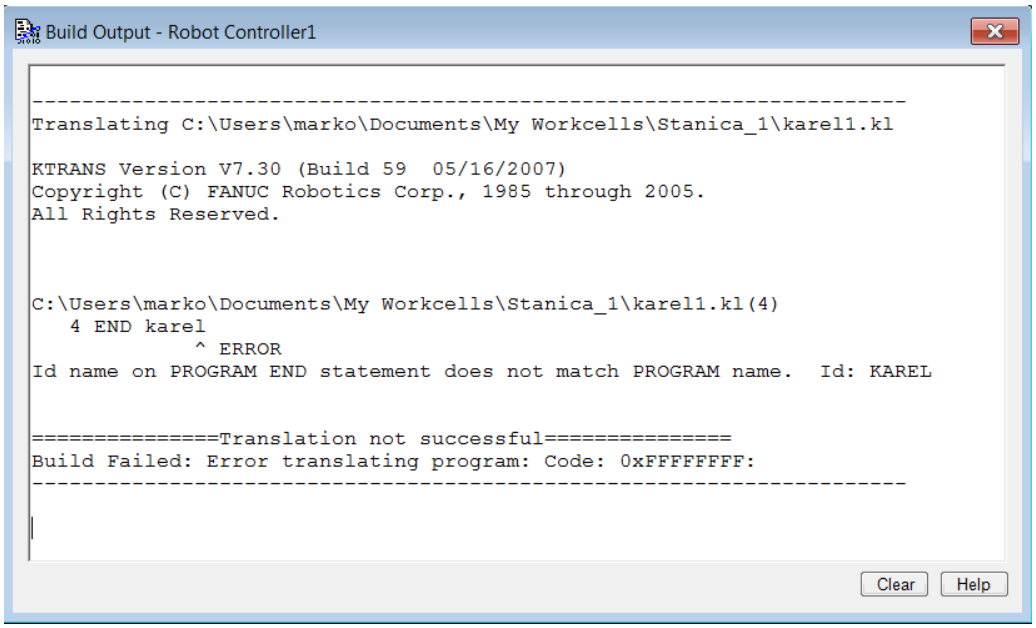
Slika 1.16. Kopiranje prevedenog programa na upravljaču jedinicu realnog robota - slanje

Prilikom prevođenja programa zbog pogreške u sintaksi korisnik dobiva obavijest vezanu uz grešku. U sljedećem programu nalazi se greška u sintaksi te je program preveden (*Build)*:

```
PROGRAM karel1
BEGIN
```

```
        WRITE('Prvi program',CR)
END karel
```

Greška koja se javlja je sljedeća:



Slika 1.17. Prikaz greške prilikom prevođenja programa sa greškom u sintaksi

Greška se nalazi u 4. redu tj. ključno ime programa iza riječi END te BEGIN se razlikuje te je ovu grešku potrebno ispraviti („karel1“ < > “karel“).

1.3.1.1. Pokretanje programa

Pokretanje programa iz virtualne upravljačke jedinice robota moguće je na dva načina:

- Izravno pokretanje KAREL programa
- Poziv KAREL programa iz TPP

Neovisno o vrsti pokretanja (*run*) programa sljedeći uvjeti moraju biti zadovoljeni za uspješno izvođenje (isti princip vrijedi i za realne upravljačke jedinice):

1. Upravljačka konzola upaljena - ON 
2. Prekid postojećeg programa – izvršava se tipkom Fctn te izborom (1) Abort all
3. Poništenje grešaka – kombinacija tipki SHIFT + Reset
4. Program mora biti označen - (l) te tipka ENTER nakon pritiska na tipku *Select* upravljače konzole
5. Na realnom robotu - gljiva na upravljačkoj konzoli i upravljačkoj jedinici izvučena
6. Na realnom robotu – postavka rada u T1 (max 250mm/s)!

Za uspješno prevođenje programa (*Build*) potrebno je također paziti na određeni slijed. Preduvjeti za uspješno prevođenje programa su:

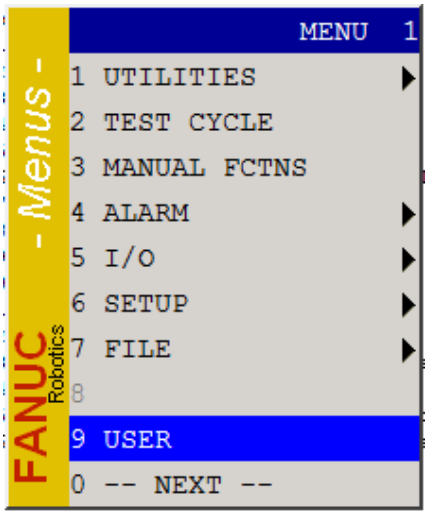
1. Program nije aktivan
2. Program nije pauziran
3. Program nije pauziran zbog nastanka greške

4. Program nije u *STEP* režimu rada

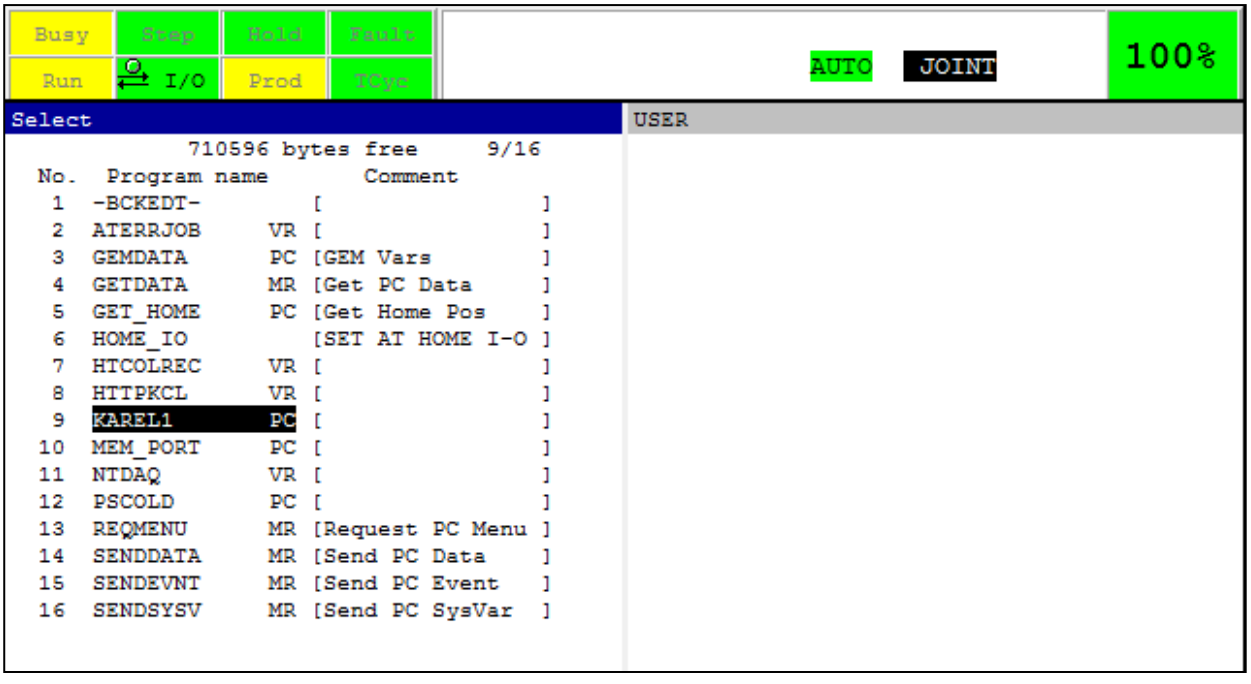
Jedini način uspješnog prevođenja programa je neaktivnost tj. stanje *ABORT*.

1.3.1.1.1.Izravno pokretanje KAREL programa

Kombinacijom tipki SHIFT i DISP moguće je ekran upravljačke konzole podijeliti na 1,2 ili 3 pod-ekrana koji služe za povećanu preglednost. Za pokretanje programa iz prethodnog poglavlja potrebno je na desni ekran postaviti korisnički ekran – *User*. Označavanjem desnog ekrana (l) te pritiskom tipke MENU sa upravljače konzole dobiva se pregled glavnog izbornika. Pritiskom na izbor pod rednim brojem (9) *USER* prikazuje se korisnički ekran. Na lijevom ekranu upravljačke konzole potrebno je pritisnuti tipku SELECT koja daje prikaz trenutnih programa spremljenih u upravljačkoj jedinici robota. Pokretanje programa sa upravljačke konzole moguće je jedino iz lijevog prozora sa ekrana *SELECT*.



Slika 1.18. Glavni izbornik upravljačke jedinice

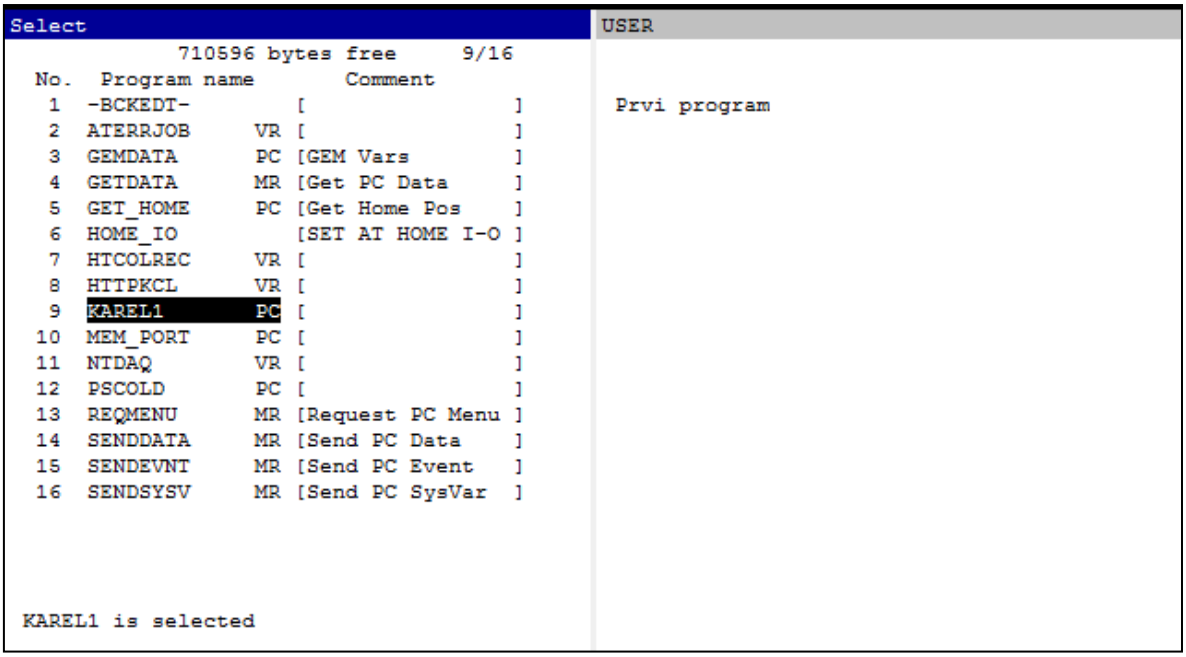


Slika 1.19. Stanje upravljačke konzole prije pokretanja programa

Prije pokretanja TPP ili KAREL programa sa upravljačke jedince potrebno je izvršiti sljedeće radnje:

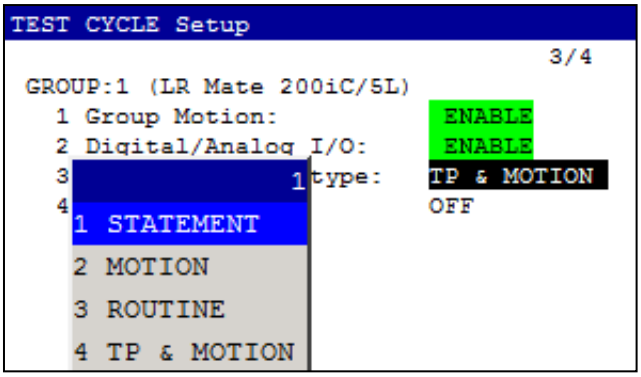
1. Upravljačka konzola upaljena - ON 
2. Prekid postojećeg programa – izvršava se tipkom Fctn te izborom (1) Abort all
3. Poništenje grešaka – kombinacija tipki SHIFT + Reset
4. Program mora biti označen - (l) te tipka ENTER nakon pritiska na tipku *Select* upravljače konzole
5. Na realnom robotu - gljiva na upravljačkoj konzoli i upravljačkoj jedinici izvučena
6. Na realnom robotu – postavka rada u T1 (max 250mm/s)!

Program se pokreće kombinacijom tipki SHIFT i FWD sa upravljačke konzole. Nakon izvršenja programa na korisničkom ekranu se nalazi ispisano: *Prvi program.*



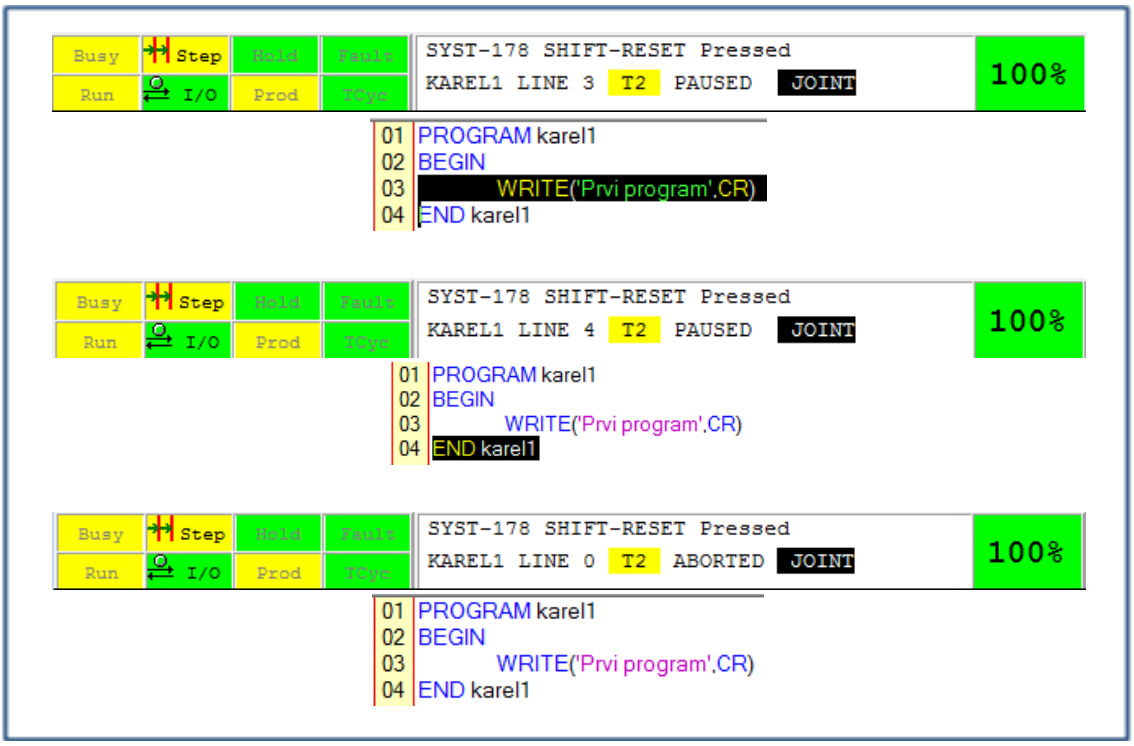
Slika 1.20. Pokretanje programa

Za detaljan uvid u rad programa (KAREL i TPP) program je moguće pokrenuti u koračnom (*STEP* tipka na upravljačkoj jedinici) režimu rada. Na ovaj način program se izvršava liniju po liniju koda. Postavke za koračni režim rada mogu se podesiti sa glavnog izbornika (*MENU*) te pritiskom na *TEST CYCLE*. Potrebno je izabrati *STATEMENT* iz izbora *Step statement type*



Slika 1.21. Postavke za koračni režim rada

Kod koračnog režim rada aktivirana je žuta *Step* ikona dok kod normalnog režima rada ova je ikona zelena. Promjenu u koračni tj. kontinuirani režim rada moguće je izvršiti pritiskom na tipku *STEP* sa upravljačke konzole. Željeni program potrebno je označiti te pokrenuti (*SHIFT + FWD*). Svakim pritiskom na tipku *FWD* izvršava se jedna linija koda. Za nastavak izvršavanja programa u kontinuiranom režimu potrebno je pritisnuti tipku *STEP*. Na sljedećoj slici prikazan je koračni režim pokretanja programa karel1.pc te pritom aktivne linije koda.



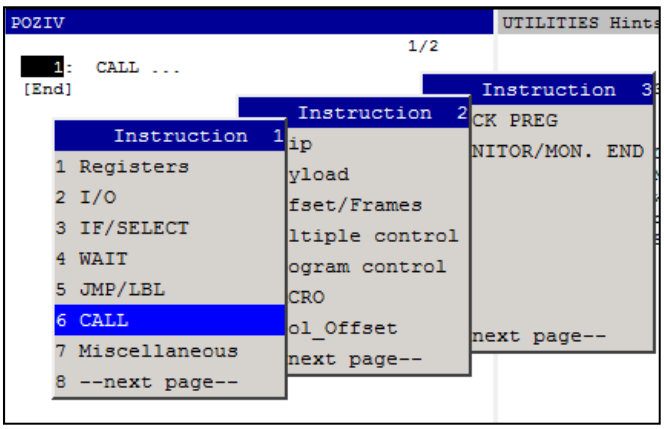
Slika 1.22. Korački režim rada – karel1.pc

1.3.1.1.1. Pokretanje KAREL programa pozivom iz TPP

Za povećanu fleksibilnost rada te korištenje željenih funkcija iz TPP i KAREL programa moguć je poziv KAREL programa izravno iz TPP. Tok izvršavanja programa je sljedeći:

1. TPP se izvršava liniju po liniju koda dok se ne aktivira linija koda sa pozivom KAREL programa
2. TPP pauzira na liniji koda sa pozivom KAREL programa
3. KAREL program se izvršava
4. TPP se nastavlja tek kada KAREL program završi

Ujedno za simulaciju KAREL programa unutar Roboguide-a potrebno je izvršiti poziv KAREL programa iz TPP. Naredba za poziv nalazi se u izborniku [INST] -> CALL -> CALL Program -> KAREL kao što je prikazano na donjoj slici.



Slika 1.23. Poziv KAREL programa iz TPP

Za ovaj primjer napisan je jednostavan KAREL program orijentiran gibanju. Program unutar petlje sa čekanjem od 1000ms te ponavljanjem 5 puta pokreće vrha alata robota 30mm u smjeru x osi te 20mm u negativnom smjeru y osi trenutnog koordinatnog sustava

.

```
01 PROGRAM KAREL2
02 VAR
03   i, STATUS: INTEGER
04   config_var:CONFIG
05   p1 : XYZWPR
06
07 BEGIN
08
09    $GROUP[1].$UFRAME = $MNUFRAME[1,1];
10    $GROUP[1].$UTOOL =  $MNUTOOL[1,1]
11    $GROUP[1].$MOTYPE=LINEAR;
12    $GROUP[1].$SPEED=300
13    $GROUP[1].$TERMTYPE=NODECEL
14    CNV_STR_CONF('nut000', config_var, STATUS)
15
16    P1=CURPOS(0,0)
17
18    FOR i=1 TO 5 DO
19
20       P1.x=P1.x+30;  P1.y=P1.y-20
21          MOVE TO P1
22       DELAY 1000
23
24   ENDFOR
25
26   DELAY 1000
27   WRITE('KRAJ',CR)
28
29 END KAREL2
```

Slika 1.24. KAREL program orijentiran gibanju – karel2.kl

1.4. Zadaci - KAREL

1. Izraditi program koji svakih 2 sekunde ispisuje u novi red korisničkog ekrana tekst *'CIKLUS[i]'* te se ponavlja 10 puta.

2. Preuzeti trenutnu poziciju robota unutar KAREL programa te svaku od koordinata X, Y, Z inkrementirati za 10mm.

3. Kreirati novu *HOME* poziciju (400,0,200,0,0,0 'NUT000') te ju spremiti u pozicijski registar 10 upravljačke konzole *PR[10]*.

4. Zbrojiti vrijednosti iz numeričkih registara R[1] i R[2] te ih spremiti u numerički registar R[3].

5. Napisati program koji provjerava da li se robot nalazi unutar zadanog radnog područja. Radno područje opisano je pravokutnom prizmom čiju bazu definiraju točke T1 (0,-100,0), T2(600,100,0) dok visina prizme iznosi 500mm u pozitivnom smjeru osi z koordinatnog sustava *World.* Stranice prizme paralelne su sa osima koordinatnog sustava svijeta robota (*World*). U ovisnosti da li se robot nalazi unutar ili izvan područja napisati poruku na korisnički ekran te upaliti tj. ugasiti određeni digitalni signal.

6. Izraditi TP program orijentiran gibanju koji se sastoji od gibanja od proizvoljne točke P1 do proizvoljne točke P2. Zadatak je pronaći takvu vrijednost varijable *$GEN_OVERRIDE* pomoću KAREL programa da trajanje gibanja bude u rasponu od [2 *s* – 2.1 *s*].
   Nakon izvršenog gibanja te registracije trajanja pomoću programskog elementa TIMER potrebno je promjenom sistemske varijable $GEN_OVERRIDE unutar KAREL programa dovesti trajanje gibanja u traženi vremenski okvir. Opetovanim izvođenjem gibanja nakon promjene vrijednosti varijable u pozitivnom ili negativnom smjeru približiti će se željenoj vrijednosti. Zapisati dobivenu vrijednost varijable $GEN_OVERRIDE u numerički registar R[1] te vrijeme trajanja ciklusa u R[2].

1.5. Primjeri KAREL programa

```
PROGRAM KAREL_01
BEGIN
        WRITE('Ispis na USER SCREEN',CR)
END KAREL_01
```

```
PROGRAM KAREL_02
VAR
  i, STATUS: INTEGER
  p1 : XYZWPR

BEGIN

   $GROUP[1].$UFRAME = $MNUFRAME[1,1];
   $GROUP[1].$UTOOL =  $MNUTOOL[1,1]
   $GROUP[1].$MOTYPE=LINEAR;
   $GROUP[1].$SPEED=300
   $GROUP[1].$TERMTYPE=NODECEL

   P1=CURPOS(0,0)

   FOR i=1 TO 5 DO

      P1.x=P1.x+20;  P1.y=P1.y-10
      WRITE(p1,CR)
         MOVE TO P1
     DELAY 1000

  ENDFOR

  DELAY 1000
  WRITE('KRAJ',CR)

END KAREL_02
```

```
PROGRAM KAREL_03

VAR
 STATUS:INTEGER
 rand,x,y,z,w,p,r:REAL
 p1 : XYZWPR
 config_var:CONFIG

BEGIN

    CNV_STR_CONF('nut000', config_var, STATUS)
    X=485;Y=0;Z=250;W=0;P=0;R=0;
    p1= POS(x,y,z,w,p,r, config_var)

   $GROUP[1].$UFRAME = $MNUFRAME[1,1];
   $GROUP[1].$UTOOL =  $MNUTOOL[1,1]
   $GROUP[1].$MOTYPE=LINEAR;
   $GROUP[1].$SPEED=400
   $GROUP[1].$TERMTYPE=FINE

   MOVE TO p1

END KAREL_03


PROGRAM KAREL_04

VAR
        i, j, STATUS:INTEGER
        r:REAL
        s:STRING[30]

BEGIN
        -- Tipovi podataka
        j=100
        r=10.123558
        s='robota'

        --PROMJENA VRIJEDNOSTI SISTEMSKE VARIJABLE
          $MCR.$GENOVERRIDE=1

        --ISPIS PODATAKA
        WRITE('j= ' ,j,CR)
        WRITE('r= ', r::6::3,CR)
        WRITE('Programiranje ',s,CR)

        --VREMENSKO ČEKANJE OD 1000ms
        DELAY 1000
```

```
    --RAD SA DIGITALNIM SIGNALIMA
    IF DIN[101]= ON THEN
        DOUT[101]=ON
    ENDIF

    --RAD SA ROBOTSKIM SIGNALIMA
    IF RDI[1]=ON THEN
        RDO[1]=OFF
    ENDIF

      i=2
      ---SELECT NAREDBA
    SELECT i OF
      CASE (1):
        WRITE('IZBOR 1',CR)
      CASE (2):
        WRITE('IZBOR 2',CR)
      ELSE:
        WRITE('krivi izbor',CR)
    ENDSELECT


    --FOR PETLJA
    FOR I=1 TO 5 DO
        DELAY 100
    ENDFOR

    i=7

    --WHILE PETLJA
    WHILE I>1 DO
        I=I-1
        DELAY 100
    ENDWHILE

    --GOTO NAREDBA;SKOK NA OZNAKU
    GOTO OZN
        --OVAJ RED SE PRESKAČE
    OZN::

END KAREL_04
```