



ISO TR 8373 tanımına göre endüstriyel robot, üç veya daha fazla programlanabilir eksenli olan otomatik kontrollü, programlanabilir, çok amaçlı, bir yerde sabit duran veya tekerlekleri olan endüstriyel uygulamalarda kullanılan manipülatördür. Amerikan Robot Enstitüsü ise robot tanımını, "programlanabilir, çok fonksiyonlu, malzemeleri, parçaları, aletleri ve özel aygıtları taşıyabilir ve çeşitli programlanmış fonksiyonları yaparak çeşitli görevleri yerine getirir" şeklinde yapmaktadır.

Japonya'da ise robot, hafıza aygıtı takılmış, rutin işleri yaparak insanın yerini alabilen, belirlenmiş hareketleri otomatik olarak yerine getirebilen, ekipmanı tamamen sağlanmış makine olarak tanımlanmaktadır.

Robot kelimesi ilk olarak 1921 yılında Çekoslovak oyun yazarı Karel Capek tarafından yazılan Rossum's Universal Robots adlı bilimsel oyunda kullanılmıştır. Robot kelimesi Çekçe'de köle gibi çalışan anlamına gelen "robota" ve "robotnik" kelimelerinden türemiştir. Robotun tarihçesine bakılacak olursa 3000 yıl önce robot fikri üzerine roman yazımı ile başlayan gelişme, M.Ö. 1000'de abaküsün icadı, 1623'te dört fonksiyonlu hesap makinesi, 1801 'de otomatik dokuma tezgâhları, 1938'de programlanabilir sprej boyama makinesi, 1939'da ilk elektronik bilgisayar,

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

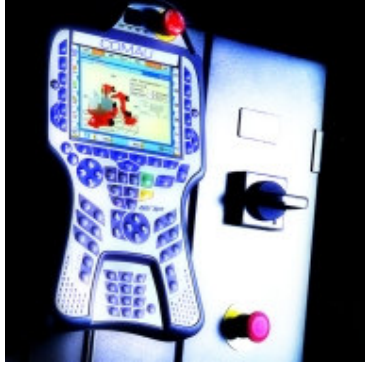
www.powerotomasyon.net



1956'da ilk programlanabilir robot tasarımı, 1973'te ilk minibilgisayar kontrollü endüstriyel robot olarak süregelen ve günümüzdeki yapay zekâ gelişmiş endüstriyel robotlara ulaşmıştır.

TEACH PENDANT TUŞLAR VE KOMUTLAR:

n



U1:Weld Timer. Execute ve Inculute yapar.

U2:Weld On-Off

U3:Yanıyor ise timer ile ilgili alarm yoktur.Basıldığında timer ile ilgili alarmlar resetlenir.

U4: Yedek

A1:punta ile ilgili parametreleri görmemizi sağlar.weld program noyu görürüz.

A2:eğer robot sialer robotuysa kullanılır.

REC:Yeni bir Liner,Joint hareket satırını oluşturur.

T1:Tabanca yarım açık

T2:Tabanca tam açık

Back:Geriye alma

Step Disb:Startı sürekli hale getirir.

MOD:Hareket satırını kaydetme.Final posisyonunu gördükten sonra.

JointPos(JNTPOS):Eksene göre kayıt. Nokta sonunda JNT yazar.

POS:Tabancaya göre kayıt yapar. Nokta sonunda PNT yazar.

FLY: 1 2 3 Nokta tanımlanır. Dairesel hareketlerde kullanılır. 1.noktadan sonra ikinciye uğrar fakat durmadan 3.noktaya geçer.Bu üç noktayıda aynı satır içinde tanımlarız.Daha çok sealer uygulamalarında kullanırız.

JNT:Eksenlerin en rahat şekliyle gider.

LİNER: X-Y-Z olarak hareketle, düz olarak gider.

CIRC:Dairesel hareket. 3 nokta verilir.

1.nokta: başlama noktası

2.nokta: orta nokta

3.nokta: bitiş noktası

Kayıt ederken 2.noktadan 3.noktaya git şeklinde kayıt eder.1.noktayı zaten Rec yaptığımızda tanımlamış oluruz.

EXUC:Robot stroke end alarmı verirse yani eksen zorlanması varsa bu tuş ile bu alarm 30 sn. Devreden çıkartılır.

ARM:Tool değiştirmede kullanılır.

TYP:koordinat sistemi seçimi. Tool, User,Joint

EZ:Yeni program çağırma. EZ-→Program no→Enter

Star+Step Diseb:Hareketi sürekli hale getirir.

ARM:Tool değiştirmede kullanılır.

USER:Herhangi bir düzlemde X-Y-Z koordinatlarını kendim ayarlarım.

Comau Robotlarında iki çeşit programlama dili vardır.

EZ ve PDL programlama dilleridir.

EZ kullanıcı ile TP arasında arayüz programıdır.

PDL yeni program açma,yeni rutin oluşturma,aynı rutin ve programları EXPORT ederek başka programlarda da kullanama gibi vs.. işlemleri yapabiliriz.

Pcint:Teacpendant üzerindeki görüntüyü pc ,e görmemiz için kullandığımız bir programdır.

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



Comau Robot Program Yapıları:

Program 0:

Robotun ana programıdır. Diğer programlar ; program1,program2,program33,program34 vs.. bunun içindedir.

Program 0 içinde sürekli bir cycle vardır.Robot otomatik konumdayken ve home pozisyonundayken sürekli tarama yapar.

Goto 1_1 satırını okur ve 1_1 ,den tekrar başlar.Bu sırada eğer PLC,sen uygun şartlar gelirse gelen şarta göre program seçer.

Çift tabancalı robotlarda MOVE TO pnt0001 j satırı gun 1 için home pozisyonu ,, MOVE TO pnt0002 j satırı gun 2 için home pozisyonudur.Bu satırlar global satırlardır ve her programın içinde kullanılır.

Program 1:

Short model için kullanılır.bu programın içinde modellere göre model rutinleri ,flap açma-kapama,çene açma-kapama,tipdress,tipchange,toolchange,balans,weld vs..programlar vardır.

Buradaki rutinler IF sorgulaması ile döngüye dahil edilir.

IF BIT_TEST(\$WORD[28],16 ON)

Camlı araç

ENDIF

eğer buradaki word 26 bit 16 ON geliyorsa robot bu IF içindeki ilgili rutini çalıştırır.Gelmiyorsa endif satırını okuyup devam eder.Aynı IF döngüsünde farklı rutinler çalıştırmak istersek ;

IF BIT_TEST , ELSE ,ENDIF

Komutlarını kullanırız.

Program 2:

Long model için kullanılır.bu programın içinde modellere göre model rutinleri ,flap açma-kapama,çene açma-kapama,tipdress,tipchange,toolchange,balans,weld vs..programlar vardır.

Buradaki rutinler IF sorgulaması ile döngüye dahil edilir.

Program33:

Tipdress programısır.Tek tabancalı programlarda tk bir rutin vardır.Çifttabancalı programlarda ravv_cg ve ravv_xg olmak üzere tabanca tipine göre isimlendirilmiş iki adet rutin varır.

BIT_SET(\$WORD[102],1) komutuyla kalemtraş başlatılır

BIT_CLEAR(\$WORD[102],1) komutuyla kalemtraş dururulur.

Pre_alarm_test:timer da girilen stepper değerlerine göre kalamtraşa gitmesini sağlar.

Program34:

Program 34 tipchange programıdır.manuel yada otomatik olarak çalıştırılır.timerda ayarlanan punta sayısı kadar puntayı atar ve punta adeti dolduğunda elektrod değişimine gider.

Checkgun:

Robotun adaptörlerini kontrol ettiğimiz manuel olarak civi noktasına görürdüğümüz bir programdır.

Robot bu notaya giderken normal punta noktasına gidiyormuş gibi balans,press1 vs. Rutinlerini çalıştırıp gider.

Hareket Satırı Yazma:

Program içinde REC tuşu ile yeni bir hareket satırı oluştururuz.

Kaynak Satırı Yazma:

Ins→Appl→Weld→1(1.program için 1)→Enter→False

Not:kaynak satırı için öncelikle hareket satırı oluşturmamız gerekir.

Spot Noktası Yazma:

Open Program→Spot.Lib.Cod→Shift+Help→

Open Program→Program. * →File name= Spot.Lib.Cod→Shift+Help→WP---istediğimiz ismi yazarız--- →Save→

Program No(düzeltilme yaptığımız program noyu yaz)→Wp. Yaz.

Program İçinde Çene Açma-Kapama Satırı Yazma:

Ins→Set→Hand→Open yada Close

Open Hand1:Çene yarım açık

Open Hand2:Çene tam açık.

ADNAN OKYAR (2001-2009)

Programdan Satır Silme:

Kürsörü silmek istediğimiz satırın üzerine getirip SHIFT +DEL tuşlarına aynı anda basarız.

Program İçinde Bilgi Satırı Yazma:

Ins→Line→Char

Program İçinde Nokta Bypasslama:

Hangi satırı bypass yapıcaksak o satırın üzerine gelerek Chng→Comm yaparız.

Delay Komutu:

Ins→Flow→Delay→Enter→istenilen süreyi yaz.

Begin:Başlangıç komutu.

END MOVE:Kaynak satırı ile beraber oluşur.Kaynak bitti komutu.

Satırlar Arasına Boşlu Verme:

Ins→Line→Enter

Return Komutu Yazma:

Geri dönüş komutudur.

Ins→Cool→Rtrn

Move L olan satırı Move J yapma:

Chng→Motn→Jnt→Yes→Enter

Robotu Kalibrasyon Pozisyonuna Götürme:

(F3)Exec→Shift+Help→Move Arm(1) To \$ Cal_Sys →Enter→Enter→Enter→Start

Kalibrasyonda Kaçıklık Var İse Yeniden Bu Değerlerin Yüklenmesi:

Conf→Arm→Rmem→Load (1) →Yes→Cont_Save (SM yükleme)

Kalibrasyon Değerlerinin Okunması:

Conf→Arm→Vcal→ (1) eğer panoya bir robot bağlıysa 1 değerini gireriz.

Hangi Eksenin Kalibrasyonunun Olmadığını Görme:

Display→Arm→ Status 1→Enter

Eğer Turn=1 2 3 4 5 6 ise robot kalibrasyondadır.

Eğer Turn=1 2 3 _ 5 6 ise 4.eksenin kalibrasyonu yoktur.

Kalibrasyon Ayarlama:

Conf→Arm→Turn1→Enter

Arızaları Görme:

Filr→Util→Sear→Shift+Help→Error Log→Enter

Buradan robotun o anki ve geçmişteki son 20 arızasını görebiliriz.

Tabanca Tam Açık-Yarım Açık İntutlarını Görme:

Disp→Inp→DIN→1→Enter

2.Input 1 ise çene tam açık

3.Input 1 ise çene yarım açık

2-3 İntutları her ikisinde 0 ise çene tam kapalıdır.

Hız Satırı Yazma:

Ins→Var→ARM_SPD

Robotta hareketleri istediğimiz yerde hızlandırıp yavaşlatabiliriz.

P1-----P2-----P3-----P4

ADNAN OKYAR (2001-2009)

%15 %80 %100

Move Joint P1
\$ ARM_SPD : =15
Move Joint P2
\$ ARM_SPD : =80

Move Joint P3
\$ ARM_SPD : =100
\$ARM_OVR:=100
\$ARM_SPD_OVR:=50
\$SPD_OPT:=SPD_LIN
\$LIN_SPD:=1
MOVE LINEAR TO pnt0001p
MOVE LINEAR TO pnt0002p,
 WITH \$LIN_SPD:=0.5
END MOVE
MOVE LINEAR TO pnt0003p,
 WITH \$ARM_SPD_OVR:=100
 WITH \$LIN_SPD:=0.5
END MOVE
MOVE LINEAR TO pnt0004p

Hızlanma Zamanı→Sabit Hız→Yavaşlama Zamanı

\$GEN_OVR:=50 (2.5 sn. Verilen hızın yarısı kadar hızla çalışır)
\$ARM_OVR:=50(5sn. Verilen hıza daha çabuk ulaşır)
\$ARM_ACC:=5(Hızın yükselme zamanı)
\$ARM_DEC:=5(Yavaşlama zamanı)
\$ARM_SPD:=50(sabit hızı)

MOVE LINEAR TO pnt0001p
\$ARM_ACC_OVR:=50
MOVE LINEAR TO pnt0002p,
 WITH \$ARM_DEC_OVR:=50
END MOVE
MOVE LINEAR TO pnt0004p

Noktaya Yaklaşma Açısı:

Nosette: yavaş yaklaşma

Coarse :kaba yaklaşma (1.5 mm. Hassasiyet ile yaklaşır)

Fine : hassas yaklaşma (0.5 mm. hassasiyet ile yaklaşır)

Program içerisinde yazılış şekli: \$TERM_TYPE:=FINE

Koordinat Sistemi:

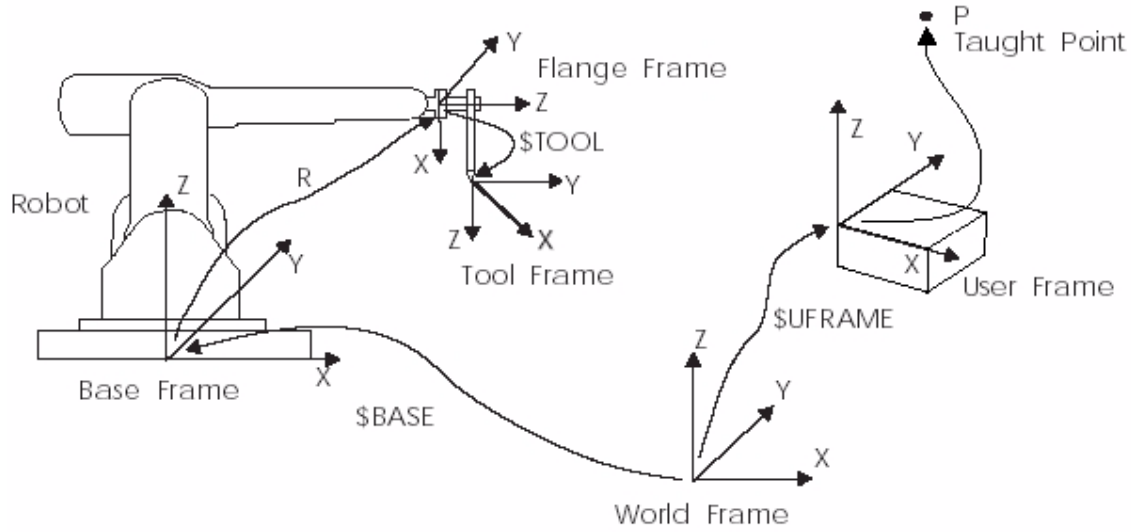
ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net





Tüm Programların COD ve VAR dosyalarını Kopyalama:

Filr→copy→pr. *

Dairesel Hareket (cırc motion):

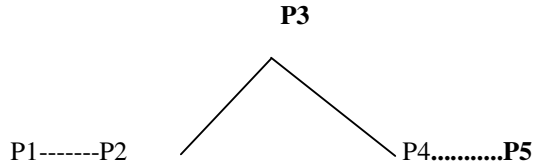
P3 .

P1,----P2.

.P4-----P5

CIRC komutta p2,yi kaydettikten sonra robotu p3 noktasına getirip CIRC seçip REC yaparız.Robotu p4,e getirip modpos yaparız.Yes→Enter yaparız.

FLY Hareket :



Robot p2,den p3,e linear gider.P3,ün yakınından geçip p4,e gider.p3,de durmaz.

\$FLY_PER:=50 (noktanın yanından geçme mesafesi)

Sicura Programını Skipleme:

Char→sağ tuş→shift+sel→char→yes→Shift+Sel

Kaynak Satırı Ekleme:

Ins→Appl→Spot→Weld→Progno→False→Enter

Yeni bir kaynak satırı ekleyebilmek için hareket satırının bitiminde END MOVE olması gerekir, bunu koymak için str sonuna

Virgül konulur.

Kaynak Satırı İptal Etme:

WITH CONTITION FALSE

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



Chg→com

Weld Program No Değiştirme:

Chng(F3)→Line(F5)→Hangi program ile kaynak yapmak istiyorsak onu gir.

Input FORCE İşlemi:

Set(F7)→Inp(F3)→Forc(F1)→DIN(F2)→Port Index: Hangi input Force yapılacaksa onu yaz.

FORCE İşlemini İptal Etme:

Set (F7)→Turnf(F5)→Yes

EZ_PDL2 Yükleme:

Filr→Unitl→Inst→Shift+Help→EZ_PDL2→APPL→Enter→Enter→

Conf→Cntl→Rstr→Cold→Yes

Robot Çalışırken PDL ,de programı Görme:

Robotu manuele alırsız.

Prog→Edit→Prog no gir→otomotiğe al →exit→oklarla bakarız.

PDL,deki Programı EZ,de çalıştırma:

Chng→Enum→Yes(Y)→Enter→Prog→Edit→Shift+Help→Check.Cod→Enter→Enter→Shift+yukarı tuş→Program check→

Bir boşluk bırak→EZ yaz→Save→Exit→Mem→Load→Check.Cod→ Yes(Y)→Enter

PDL,de Programı Çalıştırma:

Dspl→Prog→hangi programın aktif olduğunu gör→buradan prog(0) deactivate et→hangi programı kullanacaksın :

Prog→Edit→Shift+Help→Program seç→Run→Run→Start

Program İçinde Satır Silme:

Shift+Del yaparak satırı silebiliriz.

Program İçinde JUMP komutu kullanma:

Ins→Flow→Goto→0...999 arası bir adres.

Ins→Flow→Labl→0...999 arası bir adres.

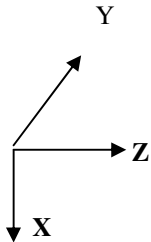
Not:Goto adresi ile Labl adresi aynı olmalı.

Robot İçin TCP Tanımlama.

SETp→var→Pos→Jnt Pos

Tabanca İçin TCP Tanımlama:

Tbl(F5)→Tool→Edit→Tabanco no gir



Tabanca ucunu X,e getirip F1(X) yap

ADNAN OKYAR (2001-2009)

Tabanca ucunu X,e getirip F2(Y) yap
Tabanca ucunu X,e getirip F3(Z) yap

Tbl→Tool→View→Tabanca üzerindeki değerleri sırayla gir.

Programa TOOL Tanımlama.

Ins→Caol→ +BI→Tool tanıttıktan sonra startlayarak programı çalıştır.

Robotta Profibus Adresini görme.

A1→View→Comm→PFDP(profibusDP)

Slave Type:PRIMARY

System Words: 1

User Words: 3

Privileged Words:0

Slave Address:116 (PROFİBUS ADRESİ)

Id Number: 1185

Not:Slave Profibus adresi hariç diğer değerler tüm robotlarda standart.

Profibus ON olmuyorsa:

Filr→Unility→Inst→RI_inst→Cold Start

Profibus Run Olduğunu Görme:

Dspl(F2)→Fbus(F3)→Slv(F2)→

PFDP=Run Olmalı, eğer stop ise profibus PLC ile haberleşmiyordur.

Profibus Adresini Tanımlama:

Prog→View→Shift+Help→RI_Inst.cod.(* . pdl.cod disketini yükle)

Prog→Go→RI_Inst.cod→Mem→Erase→

Prog→EZ. *

Prog→Prog. *

Prog→Tool. *

Go→RI_Inst.cod

PFP/MOD: (ekranda sorular gelir)

NO

YES

1

0

3

-----Profibus adresi gir.

1185

YES

NO

NO

YES

Save.Mod

Save→Conf→Cont→Restart→Wormstart→

Prog→Act→View→buradan aktif olup olmadığını görürüz.

IBS_SL

Modify

Save_mem

View

IBS_SL

IOS

PFDP_SL

DP_SL

RIO

Save_C3G

Quit

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



Profibus Devre Dışı Bırakma:

Exec→Smt: BIT_SET (\$ PROFDP_INIT [1] , 2)

Profibus Devreye Alma:

Exec→Smt: BIT_CLEAR (\$ PROFDP_INIT [1] , 2)

Tıp Dress Zamanının Ayarlanması:

Edit→Par(F1)→Dress(F1)→2000ms.. vs..

Elektrod Değişiminden Sonraki Zaman:

Edit→Par(F1)→Hand(F1)→2000ms.. vs..

Tıp Dressing Resetleme:

Exec→Smt: BIT_SET (\$ WORD [101] , 6)

Exec→Smt: BIT_CLEAR (\$ WORD [101] , 6)

Tıp Change Resetleme:

Exec→Smt: BIT_SET (\$ WORD [101] , 7)

Exec→Smt: BIT_CLEAR (\$ WORD [101] , 7)

Exec→Smt: Shift+Help→CAL→(Resume Arm1) Enter→LOCK yazıyorsa →UNLOCK ARM→Enter→Enter

Eğer Exec,den UNLOCK olmuyorsa :

Ins(F2)→Line(F7)→Shift+char ile UNLOCK yaz.

Hava Basıncı Ayarı:

U1→Smt: INC_APPL(ON)→Enter→Context.Prog:Sb_Apll.

Tabanca Çeneler Açılmadığında , Hava Basıncı Yetmediğinde.

Bu urumda tabanca silindirinde problem olabilir.Hava kayıpları vardır,silindirde çizilmeler vardır vs..

Exec→PRESS(1)→Enter→Enter→Smt→Press(1)→Prog.Context→Shift+Help→Sb-Appl→Enter→U1,U2,U3 , ON yap.

T1 ve T2 ile tekrar açmayı dene

Trafo Thermal Switch Kontrol Etme:

A2→Wiew→I/O listesinden Slot4 içinde

Robot üzerinde tabanca varken sürekli 1,dir.

Overttemperature Alarmını Bypasslama:

Trafo aşırı ısınma alarmıdır.Trafo soğutma suyunun yetersizliğinden yada trafo su kanallarının tıkanmasından kaynaklı olabilir.

X90-→X → X30→95

X90-→J → X30→104

X30—95 Ve 104 Nolu uçları köprüle.

Spot Librari Spot İlave Etme:

Mem→Load→Program Name(OPT): SPOT_LİB→Enter

Robotu Kaynaksız Çalıştırma:

Exec→INCL_APPL(OFF)→Enter→Contex_Program=Shift+Help→Sb_appl ,yi rutinlerden seç

Robotu Kaynaklı Çalıştırma:

Exec→INCL_APPL(ON)→Enter→Contex_Program=Shift+Help→Sb_appl ,yi rutinlerden seç

Tool Change Outputları:

DOT:25=OFF Take Gun

DOT:26=ON

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net




```
--\\ robot flaps
--\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\
open_flap1 = 2001
close_flap1 = 2002
open_flap2 = 2003
close_flap2 = 2004
--//*
flap1_open = 2801
flap1_close = 2802
flap2_open = 2803
flap2_close = 2804
```

Exported(Çağırma):

Herhangi bir programı bir başka rutin yada programın içinde kullanmak istersek bunu exported yapmalıyız.

Bütün programları kendi başına açarız ve daha sonra nerede kullanmak istersek o programa yazılıp exported ederiz.

Eğer exported edilmezlerse kullanılamazlar.Örneğin ez_ulib programının içinde magazin kapaklarının açılıp kapanması için

Ve bunların sinyal kontrolü için rutinler vardır.hangi programda çalışmasını istiyorsak o programın PDL ,de ilgili satırın altına

Open _flaps yazmak yeterlidir.Fakat exported edileceği zaman bunların içindeki rutinler tek tek exported edilmelidir. Kapağın

Açılması için ROUTINE open_flap1 EXPORTED FROM ez_ulib denilir.Açık olduğunun kontrolü ve kapatmak için rutinler vardır.Bu rutinler tek tek yazılıp ez_ulib,den exported edilir.

Örneğin:

VAR

```
time_swb : INTEGER EXPORTED FROM by_power
time_lwb : INTEGER EXPORTED FROM by_power
model : INTEGER EXPORTED FROM prog_0
stepper1 : BOOLEAN EXPORTED FROM prog_0
stepper2 : BOOLEAN EXPORTED FROM prog_0
pnt0001j : JOINTPOS
home_pos : JOINTPOS EXPORTED FROM lrobot
ROUTINE press(ai_wprog : INTEGER) EXPORTED FROM sb_appl
ROUTINE tf(ai_tool_idx, ai_frame_idx : INTEGER) : INTEGER EXPORTED FROM tt_tool
ROUTINE rtf(ai_rmt_tool_idx, ai_rmt_frame_idx : INTEGER) : INTEGER EXPORTED FROM tt_tool
ROUTINE z(ai_zone : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE vl(ar_lin_spd : REAL) : REAL EXPORTED FROM hmove
ROUTINE ver EXPORTED FROM hmove
ROUTINE v(ai_spd : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE orn(ai_orn_type : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE mp(ai_ref, ai_spd, ai_zone : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE gp(ai_ref : INTEGER; ar_lin_spd : REAL; ai_zone, ai_orn_type : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE ap(ai_ref, ai_orn_type : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE check_home EXPORTED FROM prog_0
ROUTINE reset_ios EXPORTED FROM prog_0
ROUTINE close_flaps EXPORTED FROM prog_0
ROUTINE open_flaps EXPORTED FROM prog_0
ROUTINE check_stepper EXPORTED FROM prog_0
ROUTINE gun_stepper2 EXPORTED FROM prog_0
ROUTINE gun_stepper1 EXPORTED FROM prog_0
ROUTINE check_stepper2 EXPORTED FROM prog_0
ROUTINE check_stepper1 EXPORTED FROM prog_0
ROUTINE weld_init EXPORTED FROM sb_appl
ROUTINE dcp_fault(ai_numero_allarme : INTEGER; as_mod_name : STRING) EXPORTED FROM lrobot
ROUTINE code_prg : INTEGER EXPORTED FROM lrobot
```

ADNAN OKYAR (2001-2009)

ROUTINE home_check EXPORTED FROM lrobot
ROUTINE sendsig(n : INTEGER; stato : BOOLEAN) EXPORTED FROM lrobot
ROUTINE waitsig(n : INTEGER; stato : BOOLEAN) EXPORTED FROM lrobot
ROUTINE a_tool(ai_t_idx : INTEGER) EXPORTED FROM tt_tool
ROUTINE l30lhr2_swb EXPORTED FROM prog_1
ROUTINE l30lhr2_lwb EXPORTED FROM prog_2
ROUTINE l30lhr1_swb EXPORTED FROM prog_1
ROUTINE l30lhr1_lwb EXPORTED FROM prog_2
ROUTINE l30rhr2_swb EXPORTED FROM prog_1
ROUTINE l30rhr2_lwb EXPORTED FROM prog_2
ROUTINE l30rhr1_swb EXPORTED FROM prog_1
ROUTINE l30rhr1_lwb EXPORTED FROM prog_2
ROUTINE x20lhr2_swb EXPORTED FROM prog_1
ROUTINE x20lhr2_lwb EXPORTED FROM prog_2
ROUTINE x20lhr1_swb EXPORTED FROM prog_1
ROUTINE x20lhr1_lwb EXPORTED FROM prog_2
ROUTINE x20rhr2_swb EXPORTED FROM prog_1
ROUTINE x20rhr2_lwb EXPORTED FROM prog_2
ROUTINE x20rhr1_swb EXPORTED FROM prog_1
ROUTINE x20rhr1_lwb EXPORTED FROM prog_2
ROUTINE x20lhr3_swb EXPORTED FROM prog_1
ROUTINE x20lhr3_lwb EXPORTED FROM prog_2
ROUTINE x20lhr4_swb EXPORTED FROM prog_1
ROUTINE x20lhr4_lwb EXPORTED FROM prog_2
ROUTINE x20rhr3_swb EXPORTED FROM prog_1
ROUTINE x20rhr3_lwb EXPORTED FROM prog_2
ROUTINE x20rhr4_swb EXPORTED FROM prog_1
ROUTINE x20rhr4_lwb EXPORTED FROM prog_2
ROUTINE x30lhr2_swb EXPORTED FROM prog_1
ROUTINE x30lhr2_lwb EXPORTED FROM prog_2
ROUTINE x30lhr1_swb EXPORTED FROM prog_1
ROUTINE x30lhr1_lwb EXPORTED FROM prog_2
ROUTINE x30rhr2_swb EXPORTED FROM prog_1
ROUTINE x30rhr2_lwb EXPORTED FROM prog_2
ROUTINE x30rhr1_swb EXPORTED FROM prog_1
ROUTINE x30rhr1_lwb EXPORTED FROM prog_2
ROUTINE x30lhr3_swb EXPORTED FROM prog_1
ROUTINE x30lhr3_lwb EXPORTED FROM prog_2
ROUTINE x30lhr4_swb EXPORTED FROM prog_1
ROUTINE x30lhr4_lwb EXPORTED FROM prog_2
ROUTINE x30rhr3_swb EXPORTED FROM prog_1
ROUTINE x30rhr3_lwb EXPORTED FROM prog_2
ROUTINE x30rhr4_swb EXPORTED FROM prog_1
ROUTINE x30rhr4_lwb EXPORTED FROM prog_2
ROUTINE service EXPORTED FROM prog_34
ROUTINE tipdress EXPORTED FROM prog_33
ROUTINE incl_appl(ab_sts : BOOLEAN) EXPORTED FROM sb_appl

ROUTINE main EXPORTED FROM prog_0

COMAU ROBOTTA KALİBRASYONUN AYARLANMASI

Robotun kalibrasyonu doğruysa, Teach Pendant'da ekranın sağ üst köşesinde **Ar:CAL** yazar ; eğer herhangi bir şekilde eksenlerin bir veya birkaçında kaçıklık olduysa **Ar:Turn** yazar ki bu da Turn değerlerinin ayarlanmasını gerektirir.

ADNAN OKYAR (2001-2009)

Power Automation & Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



NOT : Eğer **Ar: ---** yazıyorsa bu kalibrasyon yapmak gerekiyor anlamına gelmez, kalibrasyon bilgilerini Flash Memory'den çağırmak yeterlidir. Bunun için şu komutları kullanınız :
Configure → Arm → Reten_Mem → Load (CARL)

Hangi eksen(ler)de problem olduğunu görmek için durum penceresinin altındaki komutlardan sırayla Display → Arm → Status (DAS) seçilir.

Örnek : Cal : 123456
Turn : 12_456

Örnekten de görüldüğü gibi açılan pencerede problem olan eksenin numarası gözüküyor. Sadece bu eksenin turn değerini ayarlamak yeterlidir.

Eğer Cal numaralarından herhangi biri gözükmiyorsa, bu kalibrasyon sabitlerinde problem olduğunu gösterir. Bu durumda robot panosunun içindeki kağıttan daha önce kaydedilmiş kalibrasyon sabitlerinin TP'dekiyle aynı değerler olup olmadığı kontrol edilir. TP'deki değerler Configure → Arm → Vcal (CAV) komutlarıyla görülür. Farklılık varsa ilk olarak kalibrasyon işlemi yapılmalıdır(İşlemi yaparken Dr:ON olmalıdır.)

1) İşlem yapılacak eksen uygun konuma getirdikten sonra Configure → Arm → Calibrate (CAC) komutları seçilir, Arm numarası girilir (1) , eksen numarası girilir, şifre girilir ve Yes komutu seçilir. Eğer koordinat, kağıtta yazanla birebir tutmuyorsa ki bu ilk denemede çok zor, bu işlem robotun hızı düşürülerek daha hassas bir şekilde aynı rakamı tutturana kadar yapılmalıdır.

2) Kalibrasyon sabitleri ayarlandıktan sonra Turn-Set değerlerini de ayarlamak gerekir. Bunun için hareket şekli JNT olmalıdır(TYP tuşuyla seçilir.) ve robot hızının da düşük olması gerektiği unutulmamalıdır(İşlemi yaparken Dr:ON olmalıdır.). Configure → Arm → Turn_Set (CAT) komutları kullanarak eksenler sırayla kalibrasyon çizgilerine getirilerek ayarlanır; milimetrik ayar yapmak gerekli değildir. Eğer çok kaçıklık olursa 59409-59421-59422 nolu arızalar ekranda gözükür. 59421 uyarısı geliyorsa + yönde ; 59422 uyarısı geliyorsa – yönde ayar yapmak gerekir.

3) Robot kalibrasyon pozisyonunun doğruluğunu gözle kontrol etmek için menüden EXEC : SHIFT +HELP tuşlarına birlikte basarak çıkan komutlar menüsünden **MOVE TO \$CAL.SYS** seçilir ve 2 defa ENTER tuşuna basılır. Dr:ON butonuna basıldıktan sonra düşük hızda START tuşuna basılarak robot kalibrasyon pozisyonuna gönderilir ve kalibrasyon çizgilerinin her eksen için ayrı ayrı tutup tutmadığı kontrol edilir.

*** Eğer "TURN" değeri kaybolmamışsa ve ayarlamak istiyorsak programları deaktif etmemiz gerekir.

BACKUP İKİ ŞEKİLDE ALINIR

1-DİSKETE BACKUP ALMA

2-PG , YE BACKUP ALMA

DİSKETE BACKUP ALMA PROSÜDÜRÜ :

1-DİSKET BOŞMU KONTROL ET

2-DİSKETİ ROBOT FLOPY,SİNE YERLEŞTİR

2-TEACPENDANT ÜZERİNDEN Fılr (F4)---Util (F7)---Bkup (F2) MENÜLERİNİ SEÇEREK KOMLE BACKUP ALIRIZ

DİSKETTEN BACKUP YÜKLEME :

1- Fılr (F4)---Copy (F1)---Sour file (OPT) (*) : FD : ---- Dest filename (*) : RD: MENÜLERİNİ KULLANARAK KOMPLE BACKUP YÜKLERİZ.

PG , YE BACKUP ALMA :

1-BACKUP KABLOSUNU PG ÜZERİNDE COM_1 KONNEKTÖRÜNE TAK

2-KABLONUN DİĞER UCUNU ROBOT ÜZERİNDEKİ COM P KONNEKTÖRÜNE TAK

3-PG İÇİNDE C DOSYASINDA ÖRNEĞİN A ADINDA BİR KLOSÖR AÇ

4-DAHA SONRA PCINT PROGRAMINI AÇ

5-DOWLOAD SYSTEM SOFWARE SEÇENEĞİNİ SEÇ

6-C : / PCINT YAZISI ÇIKAR BURADA ESC TUSUNA BAS

7-COM 1 - COM 2 ÇIKAR COM 1 SEÇ VE ROBOTU MANUELE AL

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



8- Filr(F4)---Copy (F1)---)---Sour file (OPT) (*) : *.* YAZIP ENTER TUŞUNA BAS DAHA SONRA ALT + D TUŞLARINA BAS. BURADA KARŞIMIZA C : PCINT YAZISI ÇIKAR, PCINT YAZISINI SİLİP DAHA ÖNCEDEN C DOSYASINDA AÇYIĞIMIZ (örneğin A klasörü) KLOSÖRÜN ADINI YAZARIZ. SONRA Dest filename (*) : YAZISI CIKAR BURAYA COMP : YAZIP ENTER,LARIZ . BACKUP ALMAYA BAŞLARIZ. ALT + X YAPARAK PROGRAMDAN CIKARIZ.EN SON EXIT YAPARAK PROGRAMI KAPATIRIZ.

PG , DEN BACKUP YÜKLEME :

1-BACKUP KABLOSUNU PG ÜZERİNDE COM 1 KONNEKTÖRÜNE TAK.
2-KABLONUN DİĞER UCUNU ROBOT ÜZERİNDEKİ COM P KONNEKTÖRÜNE TAK
3- PG İÇİNDE C DOSYASINDA ÖRNEĞİN A ADINDA BİR KLOSÖR AÇ
4-DAHA SONRA YÜKLEYECEĞİN PROGRAMI YENİ AÇTIĞIN DOSYANIN İÇİNE YÜKLE.
5- PCINT PROGRAMINI AÇ C : / PCINT YAZISI ÇIKAR BURADA ESC TUSUNA BAS COM 1 - COM 2 ÇIKAR COM 1 SEÇ VE ROBOTU MANUELE AL
6- Filr(F4)---Copy (F1)---)---Sour file (OPT) (*) : COMP :YAZIP ENTER TUŞUNA BAS DAHA SONRA ALT + D TUŞLARINA BAS. BURADA KARŞIMIZA C : PCINT YAZISI ÇIKAR, PCINT YAZISINI SİLİP DAHA ÖNCEDEN C DOSYASINDA AÇYIĞIMIZ (örneğin A klasörü) KLOSÖRÜN ADINI YAZARIZ. SONRA Dest filename (*) : YAZISI CIKAR BURAYA RD :YAZIP ENTER,LARIZ . BACKUP YÜKLEMeye BAŞLARIZ. ALT + X YAPARAK PROGRAMDAN CIKARIZ.EN SON EXIT YAPARAK PROGRAMI KAPATIRIZ.

NOT: EĞERKİ KOMPLE BACKUP YÜKLEMESİ YAPTIYSAN COLD START YAP. SADECE BİR PROGRAM YÜKLEDİYSEN O PROGRAMI MEMORY LOAD YAPMAK YETERLİDİR

C3G ROBOTLARDA SOFTWARE YÜKLEME

- Laptopımızda herhangi bir sürücüye (c sürücüsü olsun) yeni bir folder(dosya) yaratıyoruz.
- Gerekli 3 sistem disketini buraya kopyalıyoruz(1 tanesi o robota ait, 2 tanesi tüm robotlarda genel, **versiyon numarasına** dikkat etmemiz gerekiyor.).Versiyon numarasını robot paneli kapaklarından alabiliriz.Yada system disketlerinden o robota ait olan 1/3 içinden **C3GLOAD.BCF** dosyası notepad ile açarak bakabiliriz.(Her notebookta C Sürücüsünde tüm robotların System Disketleri klosöründe bulabiliriz.) C3G load data for S/W version 5.52. şeklinde yazar.
- INSERT DISKETTE NUMBER 2/3, INSERT DISKETTE NUMBER 3/3 satırlarından sonraki WAIT komutlarının önüne “ ! ” koyarak bekleme kaldırılacak ve save edilecek.
- Bu folderın yanına tool.pdl ve tool.doc folderlarını yaratıp içlerine disketleri kopyalıyoruz. (C3G.PDL2.TOOL *.PDL *.DOC) (çok gerekli değil)
- **WinPcInt** programını açıyoruz.
- Pano ile laptop arasında parallel port kablosu takılmalı.(daha hızlı yükleme işlemi için ,bu şekilde 40 dk işlem 10dk,ya düşüyor.)
-
- PC'de Help-Key mapping altında gerekli kısayollar gösteriliyor.
- EXEC komutuyla **\$CAL_USER:=ARM_JNTP** yapıp configure-save yapıyoruz. Robotun home'unu kullanıcı kalibrasyon pozisyonu olarak tanıtmak yararıma olacaktır. Bu sayede sistem disketleri yüklendikten sonra çizgilere almadan home'da kalibrasyon işlemi yapabiliriz.
- Daha sonra  properties açtık. Burada **PORT=COM 1**
File transfer folder : C:\COMAU ROBOT BACKUPS\C3G\UNDERBODY R1
System download folder : C:\COMAU C3G SYSTEM DISKS\VERSION 5.52\H4 RZDC1470
Bu 2 folder backup ve software bilgilerinin nereden alınacağını gösteriyor. Folderları önceden oluşturmak gerekiyor.
- Filer→Copy→Source→ yapıp bir şey yazmadan entera basıyoruz, destination olarak “COMP:” yazıyoruz. Şimdi otomatik olarak backup C sürücüsüne yarattığımız foldere kaydedilir.
- Direkt olarak UNDERBODY R1 kısayolunu kullanarak yükleme yaparız. Yaklaşık 4 dakika sürüyor. Bu işlemi yaparken  kısayoluna basıp on-line olmalıyız.
- Yükleme işlemi bittikten sonra parallel port kablosunu çıkarabiliriz.
- Şimdi: Filer→Copy→Source: COMP:C3G.SYS → DEST:C3G.OLD yapıyoruz. Daha sonra Conf→Load→C3G.OLD ve Conf→Save yapıyoruz.

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



- Normal olarak bütün eksenlerin turn olduğunu görüyoruz. Kullanıcı kalibrasyonunu önceden robotun Home pozisyonu olarak tanıttıysak problem yok. Conf→Arm→Turn yapıp Shift +7 'ye basarak tp'deki gibi “ / ” işaretine basarak opsiyonları çıkarıyoruz ve “user” kalibrasyon pozisyonunu seçiyoruz. Arm numarasını 1, eksen numarasını “ * “ olarak giriyoruz.
- Conf→Save işlemi sırasında “63543-10: SCC: RPT/FLASH-MEM bad checksum on hour meter” yazısı ekranda (TP) gözükür ama önemli değildir. TP'den alarm tuşuna basıp silmek gibi laptoptan CTRL+Home yaparak alarmı cancel ederiz.
- Şimdi diğer kalan dosyaları yükleyeceğiz. Filer→Copy Source: COMP Dest:boş bırakıyoruz. Bunu yaptıktan sonra ekrana “not all files transferred” yazısı gelir, önemli değil; C3G.SYS ve ERROR.LOG dosyalarının üzerine yazılmadığı için olur.
- Daha sonra cold-start yapıyoruz ve EZ ile Spot-welding otomatik olarak yükleniyor.
- TP'de A_SW1B available on CRT yazıyorsa , laptopla bağlandıktan sonra çıkarken SWAP yapmayı unutmuşuz demektir.
- C3G'de software yüklerken bir şekilde sistem tekrar bootmon'a geçer ve işlem yarıda kalırsa (mesela disketlerde problem varsa) RESET/ INIT yapılıp Y(yes) yapılır.

C3G PANOSU	C4G PANOSU
RBC2 + SCC2	SMP+MCP
SAU	DSA
Motor 300V AC Resolver	Motor 600 V AC Encoder

MCP : MOTION CONTROL PROCESSOR

SMP : SYSTEM MASTER PROCESSOR

MCP üzerinde 32 MB'lık, SMP üzerinde de 64 MB'lık flash memory modülleri var. Bunlar okunabilir, üzerinde işlem yapılabilir fakat biz kullanmayacağız. Formatlanmamalı yoksa bir daha kullanamayız. İçinde COMAU'ya ait özel yazılımlar var.

Panoda MCP kartı ayrı olarak gözüküyor, SMP'nin içinde hem MCP hem de SMP kullanılarak tek kart yapılmış. Bu kart yandığı zaman, üzerindeki flash memory modülünü alıp yeni karta takıyoruz ve hiç bir yükleme işlemi yapmadan sadece kartı takıp enerjiyi açıyoruz.

*** C3G panolarındaki **anex** modülleri (SCC üzerinde ve 3 tane) analog sinyalleri, dijital sinyale çevirmeye yarıyor. Her 2 eksen için 1 adet anex kartı var. Robot “Following error out of range” , “high current required” gibi arızalar veriyorsa, titreyerek hareket ediyorsa veya uzun süre enerji kapalı kaldığında herhangi bir eksenin kalibrasyonu kayboluyorsa bu kart arızalı olabilir.

C4G panolarında da aynı mantıkta anex cardları ve eksen kartları (current loop module) var.

*** C3G'de motor frenlerini dışarıdan kontrol etmek için (herhangi bir şekilde robot kilitlendiğinde) Brake Release Module var (COMAU kodu : CR17041181) .

C3G'de X30 kablosu 108 pinli idi, C4G'de bu sayı 42 pine indirilmiş; IOM modülüne ait kablolar yok, sadece güvenlik devresi elemanlarıyla ilgili kablolar var.

C4G'de acil stopu devre dışı bırakmak için 27 ve 28 nolu pinler köprüleniyor.

WinPCint programında “File Transfer Folder” backup'ı nereye alacağımızı gösterir.

NH1- 0360

UD (USER DIRECTORY) : Yapılan robot programları bunun içindedir. Dosyaların üzerine geldiğimizde sağ click,leyerek translate yaptığımızda otomatik olarak PDL'e çevirir. Bu işlem sadece dışarıda bir bilgisayarda çalışma yapacağımız zaman gerekli normalde çift tıklayınca gözüküyor.

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



DATA : Tool , Base ve Frame'e ait datalar vardır. "To-set"ten gelen bilgiler var. Bunlar sistem disketleri dışında olan 2 disketin içinde olan bir program.

EZ : EZ dosyaları,

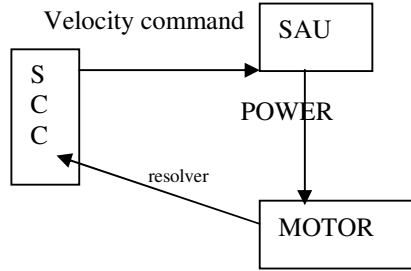
INST : Install ile ilgili programlar var.

SYS : Eski robottaki C3G.SYS yerine NH1-0360.c4g gibi bir dosya var. E-mail, log, util ve vdt kısımlarından oluşuyor. Eski robotta sadece error.log vardı, şimdi action.log da var.

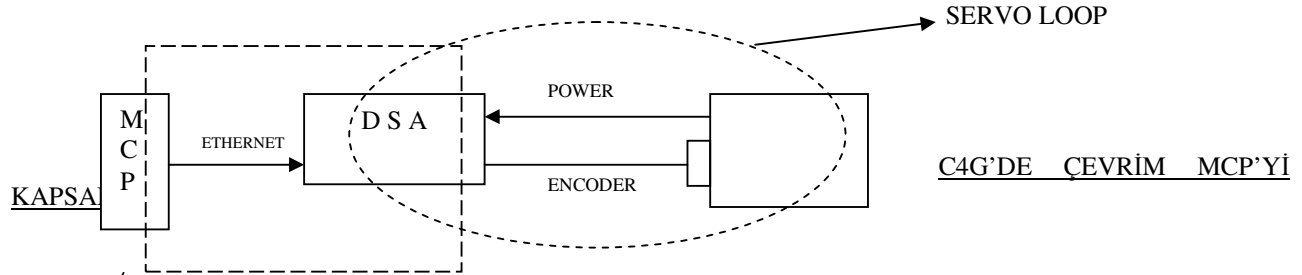
SMP : user-programs, system software, application software(EZ, glue, stud,vs...), kalibrasyon parametreleri var. (C3G'deki RBC2 ve SCC2 kartlarını kapsıyor) Kartı değiştirdiğimiz zaman bazen "System in minimum configuration" arızası gelebilir.

MCP : Robot servo kodları var.

*** C3G'de motorun resolver bilgileri SCC'ye oradan da "velocity comand" ile SAU'ya gidiyor, oradan da motora güç veriliyordu.



C3G'DE ÇEVİRİM SCC'Yİ KAPSIYORDU.



C4G'DE ÇEVİRİM MCP'Yİ

Robotun pozisyon bilgileri,
Bunlara ait alarm bilgileri Burada kontrol edilir.

MCP, ETHERNET BAĞLANTISIYLA DSA'DAN GEREKLİ BİLGİLERİ ALIR.

APS : Auxiliary Power Supply , 3 faz giriş , 5 tane 24V beslemesi var.

X1 : Kabin havalandırması

X2 : DSA

X3 : RPU

X4 : Frenler

X5 : I/O FIA (Field Interface)

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



X6 : Power failure Signal (controller)
X7 : 3 faz girişı

ÖNEMLİ NOT :

C3G Controller Unit : 110 V dolaşıyor.
C4G Controller Unit : 24 V dolaşıyor.

DSA : DIGITAL SERVO AMPLIFIER (C3G'DE SAU2 İLE AYNI)

2 üniteden oluşuyor, alt bölümde güç modülleri ve motor bağlantı kabloları var , buna monte edilen üst bölümde güç kaynağı, CPU ve DSP kartları var. İkisi arasında bilgisayar tipi yassı kablolarla bağlantı var. Buradaki CPU C3G'deki SCC kartı ile aynıdır. Güç modülleri 3 tip ;

1. ve 3. eksen için 25/90A , 2. eksen için 30/128A , 4. 5. ve 6. eksenler için 10/43A'lık modüller var.

a/b: a, her zaman sabit (0-a amper arasında) b, max. akım değerini gösterir ve en fazla 2 sn. dayanabilir.
Eğer DSA'da öndeki ünite problem olduğunu düşünüyorsak (CPU ve eksen kartlarının olduğu modül) aynı hardware'e sahip başka bir robotla değiştirebiliriz. Herhangi bir problem olmaz (eksen sayısı farklı robotlar arasında değişim yapılmaz.)

DSP : Data Signal Processor , her biri 2 eksen içermek üzere toplam 3 tane eksen kartı var. 5 tane karta kadar çıkabiliyor.

***Servo tabanca varsa 4.DSP kartı kullanılıyor.

X3 : Sürekli 24V var. (Fren beslemeleri için)

X4 : Fren çekip-bırakma (24V - 0V)

X5 : Axes overstroke limit switch (C3G'lerdeki gibi küçük bypass limit-switchleri yok)

X6 : HSI/analog digital inputs (high speed input) , yedek dijital I/O'lar 2output, 4 input (2 high speed inputs, C3G'de 1 tane vardı.

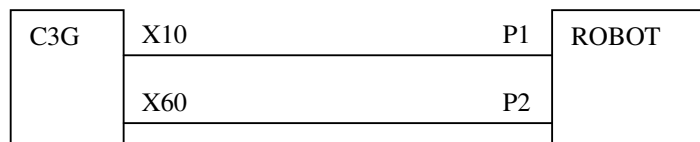
***Robotun içinde interbus var, 20 ms'de bir yenilenme oluyor INTERBUS. Yapılan bir değişiklik yaklaşık 0,1s'de bir robota ulaşıyor. High speed inputunun özelliği bu çevrim içine girmemesi ve direkt olarak SCC'ye gitmesi.

ÖRNEK:

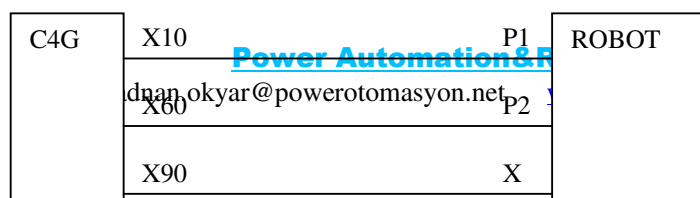
Üstüste dizili birçok parça olsun. En yüksek noktası PNT0001P, en alt noktası PNT0002P olsun.

```
MOVE JOINT TO PNT0001P
MOVE LINEAR TO PNT0002P
WITH CONDITION [1]
END MOVE
VACUUM
MOVE AWAY 100
CONDITION[1]
WHEN $DIN[1] DO
LOCK
UNLOCK
CANCEL CURRENT
```

Robot 1'den 2'ye gelirken gripperın üzerindeki high-speed inputunu yollayan sensör görene kadar hızlı gidiyor, görür görmez duruyor, vakumlayıp götürüyor. DIN yerine HDIN kullanılmasının sebebi reaksiyon süresinin çok düşük olmasıdır.



ADNAN OKYAR (2001-2009)



Power Automation & Robotics

adnan.okyar@powerotomasyon.net

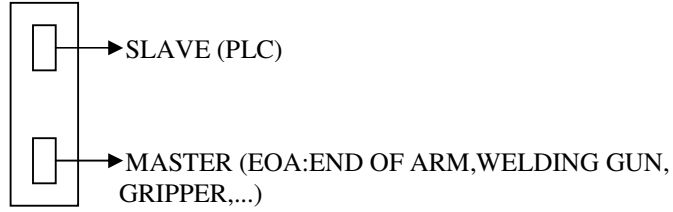
ms

omasyon.net

X10 ile encoder bilgileri , X60 motor kabloları ve X90'la da her türlü bus bilgisi bu bağlantıyla taşınıyor (gripper, weld gun,...)

!!! C3G ve C4G'de X10 ve X60 kabloları aynı değil , X10 konnektörünün tipi C3G ve C4G aynı , yanlışlıkla değiştirilebilir, dikkat edilmeli.

RPU (Robot Processing Unit) üzerinde SPARE 1/FIELDBUS(OPTIONAL) kartı yuvası var. Buraya ; INTERBUS kartı takılıyor.



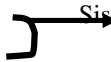
*** Laptopla bağlanırken IP adresine ihtiyacımız var (172.22.121.2)

*** TP ekranındaki harita işareti dili seçmek için kullanılıyor (Türkçe de mevcut)

Aynı ekrandaki "Login" tuşuna(F2) 2 sn'den fazla basılı tutarsak logoff oluyoruz. Logoff iken hiçbir şekilde robot hareket ettirilemiyor.

Tuş takımı içindeki "nokta" küçük harf, büyük harf veya sayı moduna geçmemizi sağlıyor.

Username : ADMIN
Password : ADMIN



Sistemde default olarak var.

Sisteme yeni User eklemek ;

*** Setup tuşuyla yapılıyor. Log-in'e giriyoruz. Daha sonra User seçilip ve gerekli bilgiler girilir (Username,şifre).

*** Configuration-save yapmayı unutursak Coldstart yapıldığında eklenen user gider.

*** Userları ancak ADMIN silebilir, bunun için sileceğimiz user LOGOUT olmalıdır.

*** Programmer (modify,rutin yazma,...), maintenance (calibration,turn-set,...) ve service (daha kapsamlı bilgiler) seçenekleri müdahale edebileceğimiz konuları seçer.

SETUP-ARM-CALIB'den kalibrasyonu görüyoruz. Turn-set'ten ayarlıyoruz.

C3G'de kalibrasyon sabitlerini değiştirmek için ;

EXEC \$CAL_DATA[1]:=1345

CAL-ARM-VIEW yaptığımızda 1**** 2**** gibiye yukarıdaki statement çalışmaz.

♣ TP-INT menüsünden EXEC tuşunu seçip ;

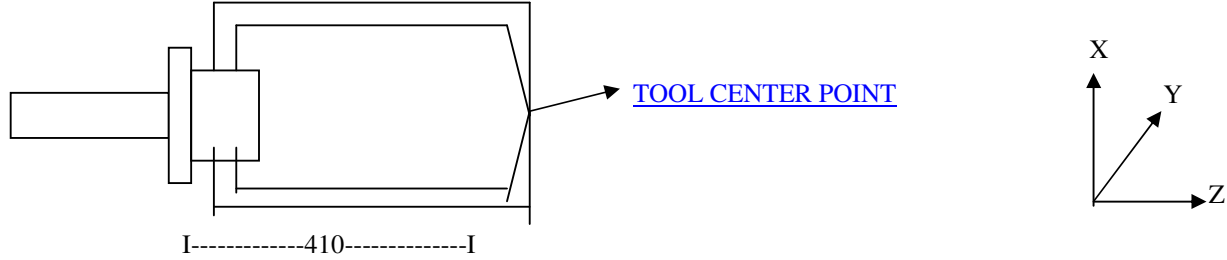
ADNAN OKYAR (2001-2009)

\$CAL_USER:= ARM_JNTP

yazarsak kullanıcı kalibrasyon pozisyonunu (robot o anda hangi konumdaysa) tanıtmış oluruz. Tüm robotlarda robotların home pozisyonlarını tanımlamak avantajlı olacaktır.

JOINT: Tüm eksenler çalışıyor, aynı oranda hareket ediyorlar. Örnek olarak 1. eksen 180^0 , 2. eksen 90^0 yol katedecekse beraber başlarlar ve yolun yarısına geldiklerinde 1. eksen 90^0 , 2. eksen 45^0 yol katetmiş olur.

♣ Robot yeni iken tool değeri sıfırdır (\$TOOL:=0,0,0,0,0,0) Bu şekildeyken robot 6.eksenin flanşına göre hareket eder. (Bkz.sonraki resim x işareti)

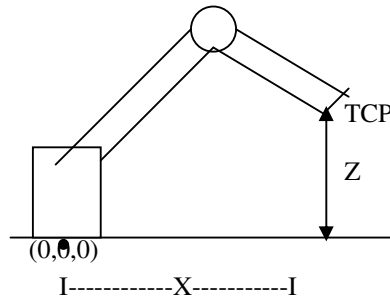


\$tool:=20,-30,410,0,0,0 olarak toolu tanıttığımızda robot tabancanın ucuna göre hareket eder.

LINEAR : Robot linear olarak hareket ederken aslında TCP noktası düz bir çizgi üzerinde hareket eder.

- a) JOINTPOS
- b) POSITION
- c) EXT POSITION (8.eksen(servo tabanca) varsa)

- a) move joint to PNT0001j : hafızaya eksenlerin(jointlerin) derecesini kaydeder.
Ax1=34,6 Ax2=27,8 Ax3= 143,796
- b) move joint to PNT0002p : hafızaya eksenlerin TCP'ye olan uzaklıklarını kaydeder.
X=1123 Y=34,76 Z=119,68



*** **Sealer programı yaparken jpos yerine pos kullanılmalı.** İlk başta tüm bead değerlerindeki distancelar sıfırlanacak ve tüm sealer noktaları parçaya geçecek şekilde modifiye edilecek. Daha sonra uygun distance değerleri girilecek fakat bu yalnızca pos kullanılırsa işe yarar. Ayrıca TCP doğru dürist set edilmiş olmalı.

- ♣ ConfigTool_2.51 : C3G'deki "tools 1-2 " disketlerinin benzeri.
HELP I/O : I/O'lara tanımlarına da text olarak bakmamıza yarar.
IO_INST : # of words , inputs-outputs , fieldbus kartın içindeki bilgileri (node sayısı)
TO_SET : TCP noktası ölçülür (tabanca, nozzle)
- ♦ EZ_2.15 : C4G'de kullanılmıyor. Bunun yerine IDE var. IDE'de otomatikte programları görebiliriz.

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net www.powerotomasyon.net



***C3G’de eski software’i yeni versiyonla güncellemek istersek backup içindeki C3G.SYS’i C3G.OLD yapıyoruz. Tools disketi içindeki utility’i kullanarak (programı) C3G.OLD’u yeni versiyona uygun olacak şekilde C3G’ye çeviririz, yükleriz ve normal çalıştırırız. Karşılaşılan tek problem sistemin orijinal hafızasının boyut paylaşımının değiştirilmesiyle oluşabilir.

RAM %50	SHARED MEMORY %50
------------	----------------------

RAM %70	SHARED MEMORY %30
------------	----------------------

Bu şekilde yeni versiyon yüklendiğinde problem çıkacağından düzeltmemiz gerekiyor. Bunun için ; BOOTMON’da MOUNT RD: yazarız, daha sonra CONF yazarız (bu komut ekranda bazı figürler verir) Bize değiştireyim mi YES/NO diye soruyor. “NO” yazarsak herhangi birşey olmaz, “YES” dersek bize “Memory”i nasıl böleceğimizi sorar, bunu orijinal haline getiririz.

♥ **RELV250E** : Software hakkında bilgi

♠ **TP4İ_1.61** : TP yazılımı , hiçbir zaman kullanmayacağız.

Ω **WINC4G_V251** : Laptopla C4G arasındaki haberleşmeyi sağlar.

[WIN C4G software ile C4G panosuna bağlanırken :](#)

IP adresini kontrol ediyoruz (TP’de var 172.22.121.2), apply tuşuna basıyoruz. Daha sonra  tuşuyla (connect) bağlanıyoruz.

Networkte problem varken sayfa 178/478’teki bilgileri tekrar yeni bağlantı yapmak için kullanıyoruz (C4G_SW_MANUELS , C4G CONTROL UNIT USE)

WIN C4G programında TERMINAL bölümünü ana ekran olarak kullanıyoruz. PC, ROBOT directory’leri var.

ROBOT’un içinde UD’de user_made programları görebiliriz.

⌘ **BACKUP ALMA** : Files – Utility – Backup of saveset – All . Bu işlemi yapmadan önce USB memory stick takılmış olmalıdır. Backup süresi ortalama 1 dakika sürmektedir.

⌘ **BACKUP YÜKLEME** : Restore of saveset – All . Conf-load ve conf-save yapılması unutulmamalı!!

Utility-Device Change-User directory : (Ram-disk)
XD : Flash Disk

Backup yüklendiğinde system dosyasıC4G (özel dosyalar, yani sadece read-only dosyalar) yüklenmez. Bunları ayrı olarak yüklüyoruz. Flash diskimizin içindeki XD\SYS klasörü içinden .C4G uzantılı dosyayı UD\SYS klasörü içine yükleriz. Configure\Save yapıp cold restart yapıyoruz.

Örnek: EXEC \$CAL_[DATA]:=0 çıkıp eski değerine getirmek için yukarıdaki şekilde kullanırız.

*** PDL’de program satırlarından robotu hareket ettirebilmek için tüm programlar deactivate edilir, IDE kapatılır, TP-INT açılır. Prog-Edit yapıp açtığımız programda her satır için run ve start yaparız.

*** Eğer RBC sağlam, SCC yüklü değil ve backupımız yoksa

MOUNT RD: (nereden yükleyeceğimizi, bilgi alacağımızı seçiyoruz.)

SEND *.* (otomatik olarak FD’ye yüklüyor. Bu sayede backupı alıyoruz)

Başka bir yolu da laptopla yapmak; ilk başta C:\TEMP1\KERMITE.EXE programını çalıştırıyoruz.

Laptopta	SET SPEED 38400
	SERVER
TP’de	SPEED 38400
	SEND *.*

Bu şekilde de backup alabiliyoruz (bağlantı Floppy’e giren kablo ile yapılacak).

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



SOFTWARE YÜKLEME

*** Eski sistemde 3 disket vardı. 2 ve 3 hepsinde aynı 1. disket robotun tipine göre değişiyordu. 2 ve 3. disketler bir tek versiyon farklıysa farklı oluyordu.

C4G'de ise sadece 1 adet CD var, her tip robot bilgileri bunun içinde var (Ver.2.51 ve TP'de gözüktüyor.)

Kısayol ile  properties'e girip "File Transfer Folder"dan robotun software bilgilerini nereden alacağını yazarız.

D:\SwC4G_2.51 (CD'den)

Şimdi ya TP'den ya da laptoptaki emulatörden sistem software'ini tekrar yükleyeceğiz.

Conf-Cntrl-Restart-Reload-Upgrade : Sadece 5.51'den 5.53'e gibi küçük versiyon değişimlerinde kullanılıyor ve pek birşey sormuyor.

TP'den yüklerken ;

Setup-ReloadSw-C4G dosyası seçilir. COMP: ve FULL: seçenekleri işaretlenir.

Bunları yaptıktan sonra OK deriz ve yükleme işlemi başlar, belli bir süre sonra otomatik cold-restart olur. Daha sonra ;

Device : Comp : (eğer laptoptan yüklüyorsak) XD: (flasdiskten yüklüyorsak) bir tanesi seçilir.

Family Robot : NH2, NH4,... seçeneklerinden NH4 seçilir.

Type Robot : Robotun üzerindeki etiketten alırsınız.

NH4 200-2,7 (Kilo(taşıma kapasitesi)-metre(ulaştığı max.mesafe))

Assi ausiliari : (harici eksen sayısı)

Nessun asse ausiliario: Sıfır anlamında

Slitta : Muhtemelen tucker için

Pinza elettrica : Muhtemelen servo tabanca için

Numero seriale robot : Etiketden robot seri numarası yazılır.

Lingua C4G : Menü dilini seçiyoruz.

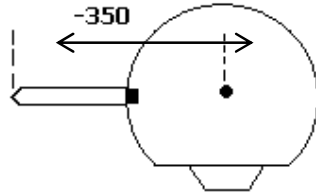
Salva yapıp kaydediyoruz ve fine'a basıyoruz.

Setup-Execute-Sysinfo-Arm : İçinde sistem identifier'dan robotun nosunu görürüz.

TCP NOKTASI AYARLAMA

Sayfa 48/84'te robot flange uzunlukları var.

2 tane tool'u takabileceğimiz delik var, bu 2 deliği de kullanarak TCP noktasını öğretmek mümkün.



"Tool calculation" software'ini install ederken bize XTOOL, YTOOL ve ZTOOL'u soruyor.

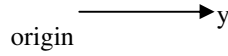
SETUP menüsünden Toolframe dosyasına girilir ve TO_SET işlemine başlanır.

1-Tool : Robota monte edilmiş ekipmanın (tabanca,vs...) boyutlarını ölçmek için kullanıyoruz (Local tool). Bunun dışında yere monte edilmiş tabancanın da boyutlarını ölçmek için kullanılır (Remote tool). Geniş açıklaması aşağıdadır.

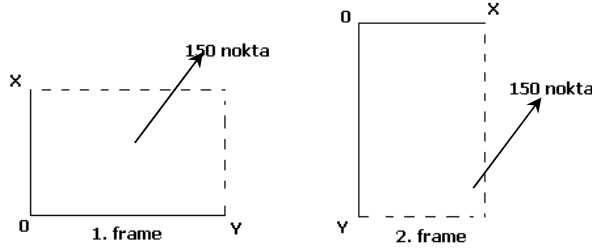
2-Uframe: Bir tane referans noktası tanıtip x ve y ayrıca origin noktasına götürüp bu değerleri öğretiyoruz. Origin aldığımız nokta framemimiz olur.



AN OKYAR (2001-2009)



Tek bir frame tanıtır bu frame'e 150 nokta kaydederek diyelim, başka bir frame tanıtır noktaları modifiye etmeye gerek kalmadan kaydırabiliriz.



3-Positioner: Ark kaynağındaki gibi kullanılan döner fişstürün ta kendisidir. Bunların örneğini Comau cd'sinin ilk sayfasındaki "positioner COMAU"dan görebiliriz. Robot harici bir eksen olarak positioneri kontrol eder. Bu yüzden encoder ve servo sürücüsü vardır.

4-Conveyor: Robot kontrol etmez ama yine üzerinde encoder var. Robot buna göre hareket eder.

TOOL işlemini yaparken :

F1'e basıyoruz, daha sonra tabanca 6.eksende ise LOCAL TOOL F1'e , başka bir yerde ise F2'ye basacağız. Daha sonra Tool sayısını giriyoruz [1...18] ; tabanca sayısına göre 1 veya 2 (ayrı ayrı kullanmak işimizi kolaylaştırır). Biz 1'i seçiyoruz.

Tool'u ölçmenin 2 yolu var.

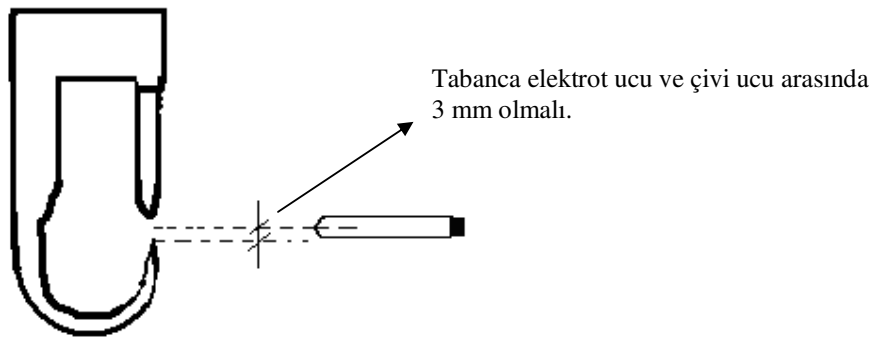
TOTAL (F1)

Bu method flanşa taktığımız çubuğu kullanarak ve bilinen bir referans noktası kullanarak yapılır. İlk başta bizim çubuğu referans noktasına daha sonra tabancanın ucunu bu referans noktasına getiririz.

Change values'tan (F2) değerleri değiştiriyoruz. Xtool:0 Ytool:-350 Ztool:-75 (sayfa 48/84)

*** Değerleri girmeden önce robotu hareket ettirsek 6.eksendeki flange'ın orta noktası etrafında dönüyor (BASE'te iken). Değerleri girdikten sonra monte ettiğimiz 200mm'lik çubuğun ucunun etrafında dönüyor.

Robotun 200 mm'lik çubuğunu belirlediğimiz referans noktasına götürüp MOVE PERFORMED(F1), daha sonra tool'un ucunu referans noktasına götürüp MOVE PERFORMED(F1) yapıyoruz.



Orient z-tool required along + z base olmalı; + olmasının sebebi tabancadan kaynaklıdır. F4 ile değiştirmek mümkün.

Gazaltı kaynak robotunda;

Punta robotunda;

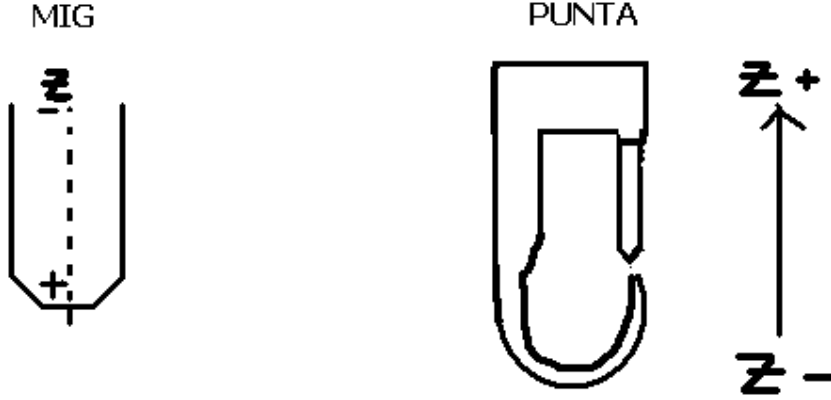
ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net





4 POINTS METHOD(F3)

Bu methodda çubuğa ihtiyacımız yok. Tool olarak 6.eksendeki flanşa yerleştirdiğimiz çubuğu belirlediğimiz bir referans noktasında robotun en az 4 farklı pozisyonu için tanıtıyoruz.

PARTIAL METHOD (İstasyonun içinde sabit bir çivi noktamız olmalı)

Bunun için ilk etapta robot kurulurken en az 1 defa TOTAL metodu kullanılarak tool tanıtılmış olmalı (sabit çivi noktasında). Daha sonra yapılacak olan tüm işlemlerde partial metodu kullanılır ve sadece tool'un ucu kullanılıyor (200 mm'lik çubuk kullanılmıyor).

Identification of payload (opsiyonel) : bizde bu yok.

WINSPLIT : Word sayfalarını (dosyalarını) *.txt dosyasına çevirir. Burada değişik yapıyoruz.

WINBACK : Bununla uzantıları tekrar *.win yapıp bilgisayarda START menüsünden RUN komutunda CMD yazıp DOS'a çıkıyoruz. Backup aldığımız dosyayı seçiyoruz.

C:\STATION 40 R4 Bunun içinde uzantısı *.win olan dosyaları seçeriz.

DOS'ta winsplit yazarsak bizden .win uzantılı dosyayı istiyor, bunu seçip .txt'e çeviririz ve daha sonra gene doste edit dosya ismi.txt yazarsak değişiklik yapacağımız dosya çıkıyor. Değişiklik yaptıktan sonra save edip çıkıyoruz. Daha sonra winback programı çalıştırılır. Dosya ismini yazarız, değişiklik yaptığımız dosyayı (.txt) eski uzantılı haline (.win) çeviririz ve yarattığımız *.txt dosyalarını sileriz, burada dikkat etmek gerekir, CALIB1.txt dosyasını silmemeliyiz.

UFRAME :

SETUP-UFRAME(F2)

LOCAL UFRAME(F1) : Tool 6.eksende

UFRAME REMOTE TOOL(F2) : Tool 6.eksende değil, başka yerde.

Warning : Üzerindeki \$Tool'u kontrol et, Frame'i oluştururken TP ekranında yazan XYZAER değerlerini kullanacağım.

F1 : continues (devam etmek için)

F2 : change values : X,Y,Z,A,E,R'de değişiklik yapmak için.

- Şimdi parça üzerinde origin noktası belirleyip, tool'un ucunu bu noktaya getireceğiz ve move performed (F1) tuşuna basılacak.
- Sonra robotu x pozisyonuna getir, move performed(F1) tuşuna bas.
- Sonra robotu y pozisyonuna getir, move performed(F1) tuşuna bas.
- Save et (F2). Frame numarası veriliyor.

***Framesiz çalışması istenen bölgelerde aframe(0) komutu kullanılmalı.

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



HELP I/O NASIL HAZIRLANIR ?

- C3G'de TOOLS 1 disketinden helpIO2.cod ve helpIO.cod dosyaları robot panelinin ramdiskine kopyalanır.
- Bazı komutlar eklemek için helpIO.cod dosyasının içine girilir(EDIT), ya da bilgisayarda edit edilip sonradan yüklenebilir.
- Daha sonra startup dosyasına aşağıdaki komutlar eklenir.

SYS_CALL ('ml', 'helpio2')
SYS_CALL ('pg', 'helpio') → C3G'DE

SYS_CALL ('ml', 'ud:/helpio2')
SYS_CALL ('ml', 'ud:/helpio') → C4G'DE

- Prog-Save yapılır ve daha sonra CCRC yapılır.
- HelpIO ve helpIO2 dosyalarını öncelikle CD'den alıp Ramdiske atmak lazım, sonra edit.
- Display-Vars-Word-Shift+7(/ işaretini çıkarmak için)-ctrl+end'e basınca yazdığımız açıklamalar görülür.
- H4 RZDC1470 : Robotun konfigürasyonunu belirtiyor (H4-standart).

***** Önemli Not: C3G'de motor-on yaptığımızda robottan bir inleme sesi geliyorsa (çok tiz bir ses) bu motorun tipiyle invertörün frekansının tutmamasından kaynaklanır ve bunu SAU2 ana kartı üzerindeki G7'nin uçlarını köprüleyerek gideririz.*****

Robot panosunda işlemciyi meşgul etmeden sürekli aktif halde çalışan programlara örnek

İşlemciyi meşgul etmeden(loop şeklinde çalışmayacak) bir program yaratıp sürekli aktif kalmasını sağlayarak robot parçayı düşürdüğünde alarm verdireceğiz.

```
PROGRAM mon_sig NOHOLD
CONST
    ic_panel_inputu = 1915
    dis_panel_inputu = 1916
VAR dummy : boolean (FALSE) NOSAVE
ROUTINE sendsig (n:integer; stato:BOOLEAN) EXPORTED FROM lrobot
ROUTINE SEND_14 (flag:BOOLEAN)
BEGIN
    sendsig (ic_panel_inputu, flag)
    If flag then
        ENABLE CONDITION [251]
    ELSE
        ENABLE CONDITION [250]
    ENDIF
END SEND_14
ROUTINE SEND_15 (flag:BOOLEAN)
BEGIN
    sendsig (dis_panel_inputu, flag)
    If flag then
        ENABLE CONDITION [253]
    ELSE
        ENABLE CONDITION [252]
    ENDIF
END SEND_15
BEGIN
    CONDITION[250] :
        WHEN $DIN[14] = ON DO
            Send_14 (TRUE)
        END CONDITION
    CONDITION[251] :
        ADNAN OKYAR (2001-2009)
```

```

        WHEN $DIN[14] = OFF DO
            Send_14 (FALSE)
        END CONDITION
    CONDITION[252] :
        WHEN $DIN[15] = ON DO
            Send_15 (TRUE)
        END CONDITION
    CONDITION[253] :
        WHEN $DIN[15] = OFF DO
            Send_15 (FALSE)
        END CONDITION
    IF $DIN[14]=ON THEN
        ENABLE CONDITION [250]
    ELSE
        ENABLE CONDITION [251]
    ENDIF
    IF $DIN[15]=ON THEN
        ENABLE CONDITION [252]
    ELSE
        ENABLE CONDITION [253]
    ENDIF
    CYCLE
    WAIT FOR DUMMY
    DUMMY:=FALSE
    END mon_sig

```

*** Dummy: herhangi bir isim de kullanabiliriz yeter ki üstte tanıtılan var ve en aşağıda cycle kısmında yazan isim eşit olsun.

*** Flag: hiçbir şey yazılmadan değeri her zaman TRUE'dur. Yani "IF FLAG THEN \$DOUT[11]=TRUE" VE "IF FLAG =TRUE THEN \$DOUT[11]=TRUE" satırları aynıdır.

*** CONDITION'lar bir çevrim halinde çalışmaz, tetiklenmek üzere arka planda sürekli beklerler.

*** "Wait for dummy" sayesinde program deaktif olmuyor ve devamlı burada bekliyor. Arkadaki condition'lar ise sinyallerin durmuna göre değişir.

```

PROGRAM mon_sig2 NOHOLD
CONST
    İç_panel_inputu = 1915
    Dis_panel_inputu = 1916
VAR input_14_old, input_15_old : BOOLEAN (FALSE) NOSAVE
ROUTINE sendsig (n: INTEGER; stato: BOOLEAN) EXPORTED FROM lrobot
BEGIN
    CYCLE
        input_14_old := $DIN[14]
        input_15_old := $DIN[15]
        sendsig (iç_panel_inputu, input_14_old)
        sendsig (dis_panel_inputu, input_15_old)
    WAIT FOR ($DIN[14] < > input_14_old) OR ($DIN[15] < > input_15_old)
    END mon_sig2

```

FREN KAÇIRAN BİR MOTOR DEĞİŞTİRİLDİKTEN SONRA ESKİ NOKTALARI BİREBİR ELDE ETMEK İÇİN

C3G'de motor çalışıyor fakat fren kaçırıyorsa (yani motor tam olarak bozuk değilse) ve motoru değiştireceksek sonradan noktalarda kaçıklık yaşamamak için kompresörü fren kaçıran motorda kullanarak motoru değiştirdikten sonra tam olarak aynı noktaları elde etmek mümkün.

ADNAN OKYAR (2001-2009)

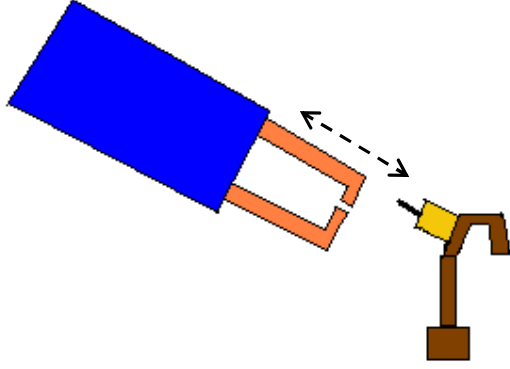
Power Automation & Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



İlk başta yeni bir program açıyoruz ve 2 nokta tanıtlıyoruz (prog-open-REF olsun). Tanıtacağımız 2 nokta fikstürde robotun ulaşabileceği rahat bir nokta olacak ve kompresörü buraya monte edeceğiz.



Kompresör(sarı renkli) kenedin üzerine takılı. İlk nokta referans noktası olacak ve tool'a alıp -z,+z eksenini kullanarak rahat ve dik olarak robot tabancasının kompresörün hareketli çivisine bastığını görürüz. İlk noktayı belirlediğimizde kompresörü sıfırlamalıyız. Daha sonra hareket ettirir ve çiviye ittirdiğimizde kolaylık olsun diye yuvarlak bir rakam seçeriz. 2.nokta ölçü aletini dik olarak itecek şekilde olmalı. Bu 2.noktaya geldikten sonra örneğin 10 mm görüyorsak, sıfırlıyoruz,bu sefer ilk noktaya aldığımızda -10 mm değer okuyacağız. Kompresör yeni motor takılana kadar aynı şekilde kalmalı.

Daha sonra motoru değiştiririz ve CAL'dan kalibrasyonu çıkarız. Eğer noktalar kaçıkça, robotu tekrar yeni açtığımız programdaki 2.noktaya götürürüz ve ölçü aletinin değerine bakarız. Örneğin ekranda 8,85mm gördüysek orijinali 10 mmdir. Bu durumda ;

EXEC: WRITE LUN_TP (ARM_JNTP,NL) yaptığımızda eksenlerin o anki durumunu görürüz.

Örneğin 183000, 78000, 126000,97000, 55000, 41000 çıkısın. Daha sonra eğer 3.eksen motoru değiştiyse 3. eksenin değerini bir yere not alırız. Sonra TP'den ölçü aletinde 10mm'yi görecektir şekilde (TOOL seçiliyken) robotu ileri veya geri hareket ettiririz.(DİKKAT!! Kaçıklık 2 kat ise yön hatası yapmışızdır.) 10 mm.'yi görünce tekrar

EXEC:WRITE LUN_TP(ARM_JNTP,NL) yapıp eksenlerin yeni değerine bakarız.

183000, 78000, 125888, 97000, 55000, 41000. Bu durumda kaydettiğimiz 3.eksenin ilk değerinden 2.değeri çıkarırız.(126.000-125.888=0.112 : aradaki kaçıklık) Daha sonra robotu kalibrasyon noktasına götürürüz. EXEC: MOVE ARM[1] BY (,0.112) yapıp start'a bastığımızda 3.eksen yazdığımız değer kadar hareket eder. Daha sonra CONFIG-ARM-CAL'da o anki değeri çıkarız ve tekrar ölçü aletinin olduğu yerdeki noktaya götürüp kontrol ederiz, eğer çok az bir kaçıklık varsa bu önemli değildir.

ROBOTTA MANUEL HIZI AYARLAMA

Önce read-only olan \$man_scale değişkenini read&write'a çeviriyoruz.

EXEC:  yapıyoruz ve bu sayede read-only dosyalara girebiliyoruz. Normalde orijinal robot hızımız 0.150 P-mastlarda farklı. Daha sonra EXEC: \$man_scale:=0.3 yaparsak ve EXEC WRITE LUN_TP (\$man_scale,NL) yaparsak hızı değiştirdiğimizi görürüz.

PROGRAM prog_0 EZ, PROG_ARM = 1, STACK = 2048

```
-----
--                                     --
--                                     --
--      First created 05.11.01      --
--                                     --
--      Last update : 16.08.02      --
--                                     --
--      by adnan okyar              --
--                                     --
--      power bilişim                --
-----
```

CONST

```
--\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
--\ \ profdp inputs and
--\ \ outputs addresses
```

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



```

--////////////////
--\\ robot outputs
--////////////////
home_position = 1801
tipchnng_running = 1803
tipchange_pos = 1804
tipdressing = 1805
end_of_program = 1806
program_in_execution = 1807
end_of_weld_1 = 1809
--\\ orange lamp blinks
prealarm_elec = 1812
--\\ orange lamp on
alarm_elec = 1813
workarea_free_to_op = 1912
--////////////////
--\\ robot inputs
--////////////////
start_program = 2601
req_tipchnng = 2609
tipchange_done = 2610
req_tipdress = 2613
workarea_free_from_op = 2712
--////////////////
--\\ robot flaps
--////////////////
open_flap1 = 2001
close_flap1 = 2002
open_flap2 = 2003
close_flap2 = 2004
--//*
flap1_open = 2801
flap1_close = 2802
flap2_open = 2803
flap2_close = 2804
VAR
time_swb : INTEGER EXPORTED FROM by_power
time_lwb : INTEGER EXPORTED FROM by_power
model : INTEGER EXPORTED FROM prog_0
stepper1 : BOOLEAN EXPORTED FROM prog_0
stepper2 : BOOLEAN EXPORTED FROM prog_0
pnt0001j : JOINTPOS
home_pos : JOINTPOS EXPORTED FROM lrobot
ROUTINE press(ai_wprog : INTEGER) EXPORTED FROM sb_appl
ROUTINE tf(ai_tool_idx, ai_frame_idx : INTEGER) : INTEGER EXPORTED FROM tt_tool
ROUTINE rtf(ai_rmt_tool_idx, ai_rmt_frame_idx : INTEGER) : INTEGER EXPORTED FROM tt_tool
ROUTINE z(ai_zone : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE vl(ar_lin_spd : REAL) : REAL EXPORTED FROM hmove
ROUTINE ver EXPORTED FROM hmove
ROUTINE v(ai_spd : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE orn(ai_orn_type : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE mp(ai_ref, ai_spd, ai_zone : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE gp(ai_ref : INTEGER; ar_lin_spd : REAL; ai_zone, ai_orn_type : INTEGER) : INTEGER EXPORTED
FROM hmove
ROUTINE ap(ai_ref, ai_orn_type : INTEGER) : INTEGER EXPORTED FROM hmove
ROUTINE check_home EXPORTED FROM prog_0
ROUTINE reset_ios EXPORTED FROM prog_0
ROUTINE close_flaps EXPORTED FROM prog_0

```

ADNAN OKYAR (2001-2009)

ROUTINE open_flaps EXPORTED FROM prog_0
 ROUTINE check_stepper EXPORTED FROM prog_0
 ROUTINE gun_stepper2 EXPORTED FROM prog_0
 ROUTINE gun_stepper1 EXPORTED FROM prog_0
 ROUTINE check_stepper2 EXPORTED FROM prog_0
 ROUTINE check_stepper1 EXPORTED FROM prog_0
 ROUTINE weld_init EXPORTED FROM sb_appl
 ROUTINE dcp_fault(ai_numero_allarme : INTEGER; as_mod_name : STRING) EXPORTED FROM lrobot
 ROUTINE code_prg : INTEGER EXPORTED FROM lrobot
 ROUTINE home_check EXPORTED FROM lrobot
 ROUTINE sendsig(n : INTEGER; stato : BOOLEAN) EXPORTED FROM lrobot
 ROUTINE waitsig(n : INTEGER; stato : BOOLEAN) EXPORTED FROM lrobot
 ROUTINE a_tool(ai_t_idx : INTEGER) EXPORTED FROM tt_tool
 ROUTINE l30lhr2_swb EXPORTED FROM prog_1
 ROUTINE l30lhr2_lwb EXPORTED FROM prog_2
 ROUTINE l30lhr1_swb EXPORTED FROM prog_1
 ROUTINE l30lhr1_lwb EXPORTED FROM prog_2
 ROUTINE l30rhr2_swb EXPORTED FROM prog_1
 ROUTINE l30rhr2_lwb EXPORTED FROM prog_2
 ROUTINE l30rhr1_swb EXPORTED FROM prog_1
 ROUTINE l30rhr1_lwb EXPORTED FROM prog_2
 ROUTINE x20lhr2_swb EXPORTED FROM prog_1
 ROUTINE x20lhr2_lwb EXPORTED FROM prog_2
 ROUTINE x20lhr1_swb EXPORTED FROM prog_1
 ROUTINE x20lhr1_lwb EXPORTED FROM prog_2
 ROUTINE x20rhr2_swb EXPORTED FROM prog_1
 ROUTINE x20rhr2_lwb EXPORTED FROM prog_2
 ROUTINE x20rhr1_swb EXPORTED FROM prog_1
 ROUTINE x20rhr1_lwb EXPORTED FROM prog_2
 ROUTINE x20lhr3_swb EXPORTED FROM prog_1
 ROUTINE x20lhr3_lwb EXPORTED FROM prog_2
 ROUTINE x20lhr4_swb EXPORTED FROM prog_1
 ROUTINE x20lhr4_lwb EXPORTED FROM prog_2
 ROUTINE x20rhr3_swb EXPORTED FROM prog_1
 ROUTINE x20rhr3_lwb EXPORTED FROM prog_2
 ROUTINE x20rhr4_swb EXPORTED FROM prog_1
 ROUTINE x20rhr4_lwb EXPORTED FROM prog_2
 ROUTINE x30lhr2_swb EXPORTED FROM prog_1
 ROUTINE x30lhr2_lwb EXPORTED FROM prog_2
 ROUTINE x30lhr1_swb EXPORTED FROM prog_1
 ROUTINE x30lhr1_lwb EXPORTED FROM prog_2
 ROUTINE x30rhr2_swb EXPORTED FROM prog_1
 ROUTINE x30rhr2_lwb EXPORTED FROM prog_2
 ROUTINE x30rhr1_swb EXPORTED FROM prog_1
 ROUTINE x30rhr1_lwb EXPORTED FROM prog_2
 ROUTINE x30lhr3_swb EXPORTED FROM prog_1
 ROUTINE x30lhr3_lwb EXPORTED FROM prog_2
 ROUTINE x30lhr4_swb EXPORTED FROM prog_1
 ROUTINE x30lhr4_lwb EXPORTED FROM prog_2
 ROUTINE x30rhr3_swb EXPORTED FROM prog_1
 ROUTINE x30rhr3_lwb EXPORTED FROM prog_2
 ROUTINE x30rhr4_swb EXPORTED FROM prog_1
 ROUTINE x30rhr4_lwb EXPORTED FROM prog_2
 ROUTINE service EXPORTED FROM prog_34
 ROUTINE tipdress EXPORTED FROM prog_33
 ROUTINE incl_appl(ab_sts : BOOLEAN) EXPORTED FROM sb_appl

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



ROUTINE main EXPORTED FROM prog_0

ROUTINE open_flaps

BEGIN

sendsig(close_flap1, OFF)

sendsig(close_flap2, OFF)

sendsig(open_flap1, ON)

sendsig(open_flap2, ON)

DELAY 2000

sendsig(open_flap1, OFF)

sendsig(open_flap2, OFF)

END open_flaps

ROUTINE close_flaps

BEGIN

sendsig(open_flap1, OFF)

sendsig(open_flap2, OFF)

DELAY 1000

sendsig(close_flap1, ON)

sendsig(close_flap2, ON)

END close_flaps

ROUTINE reset_ios

BEGIN

\$WORD[19] := 0

BIT_CLEAR(\$WORD[18], 2)

BIT_CLEAR(\$WORD[18], 3)

BIT_CLEAR(\$WORD[18], 4)

BIT_CLEAR(\$WORD[18], 5)

BIT_CLEAR(\$WORD[18], 6)

BIT_CLEAR(\$WORD[18], 7)

BIT_CLEAR(\$WORD[18], 8)

BIT_CLEAR(\$WORD[18], 9)

BIT_CLEAR(\$WORD[18], 10)

BIT_CLEAR(\$WORD[18], 11)

BIT_CLEAR(\$WORD[18], 12)

BIT_CLEAR(\$WORD[18], 13)

BIT_CLEAR(\$WORD[18], 14)

BIT_CLEAR(\$WORD[18], 15)

BIT_CLEAR(\$WORD[18], 16)

END reset_ios

ROUTINE check_home

BEGIN

home_check

sendsig(home_position, ON)

close_flaps

END check_home

ROUTINE check_stepper

BEGIN

--\\

--\\ c3g ---> swim

-- \$word[101] bit 6 tipdress done

-- \$word[101] bit 7 tipchange done

--\\ c3g <--- swim

ADNAN OKYAR (2001-2009)

Power Automation & Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



```

-- $word[109] bit 5 tipdress request
-- $word[109] bit 6 pre-alarm elec
-- $word[109] bit 7 tipchange request
--/////////////////////////////////
IF BIT_TEST($WORD[109], 7, ON) OR BIT_TEST($WORD[26], 9, ON) THEN
  GOTO tip2
ENDIF
IF BIT_TEST($WORD[109], 5, ON) OR BIT_TEST($WORD[26], 13, ON) THEN
  BIT_CLEAR($WORD[18], 1)
  BIT_SET($WORD[18], 5)
  tipdress
  BIT_SET($WORD[101], 6)
  DELAY 500
  BIT_CLEAR($WORD[101], 6)
  reset_ios
  check_home
ELSE
  IF BIT_TEST($WORD[109], 6, ON) THEN
    BIT_SET($WORD[18], 12)
    DELAY 250
    BIT_CLEAR($WORD[18], 12)
  ELSE
    IF BIT_TEST($WORD[109], 7, ON) OR BIT_TEST($WORD[26], 9, ON) THEN
tip2::
      BIT_CLEAR($WORD[18], 1)
      BIT_SET($WORD[18], 3)
      BIT_SET($WORD[18], 13)
      DELAY 500
      BIT_CLEAR($WORD[18], 13)
      service
      BIT_SET($WORD[101], 7)
      DELAY 500
      BIT_CLEAR($WORD[101], 7)
      BIT_CLEAR($WORD[18], 3)
      BIT_SET($WORD[18], 5)
      -- tipdress
      BIT_SET($WORD[101], 6)
      DELAY 500
      BIT_CLEAR($WORD[101], 6)
      reset_ios
      check_home
    ELSE
      ENDIF
    ENDIF
  ENDIF
ENDIF
END check_stepper

ROUTINE gun_stepper2
BEGIN
  BIT_CLEAR($WORD[18], 1)
  IF BIT_TEST($WORD[102], 4, ON) THEN
    GOTO serv2
  ENDIF
  IF (stepper2 = TRUE) OR BIT_TEST($WORD[26], 13, ON) THEN
    -- BIT_SET($WORD[18], 5)
    BIT_SET($WORD[101], 9) -- bit 1
    $TERM_TYPE := FINE
    $ARM_SPD_OVR := 50

```

ADNAN OKYAR (2001-2009)

```

$ARM_ACC_OVR := 50
$ARM_DEC_OVR := 50
tipdress
$ARM_SPD_OVR := 100
$ARM_ACC_OVR := 100
$ARM_DEC_OVR := 100
ELSE
serv2::
IF BIT_TEST($WORD[102], 4, ON) OR BIT_TEST($WORD[26], 9, ON) THEN
  BIT_SET($WORD[18], 3)
  BIT_SET($WORD[101], 9) -- bit 1
  $TERM_TYPE := FINE
  $ARM_SPD_OVR := 50
  $ARM_ACC_OVR := 50
  $ARM_DEC_OVR := 50
  service
  $ARM_SPD_OVR := 100
  $ARM_ACC_OVR := 100
  $ARM_DEC_OVR := 100
  BIT_SET($WORD[101], 7)
  DELAY 500
  BIT_CLEAR($WORD[101], 7)
  BIT_CLEAR($WORD[18], 3)
  -- BIT_SET($WORD[18], 5)
  $ARM_SPD_OVR := 50
  $ARM_ACC_OVR := 50
  $ARM_DEC_OVR := 50
  -- tipdress
  $ARM_SPD_OVR := 100
  $ARM_ACC_OVR := 100
  $ARM_DEC_OVR := 100
  BIT_CLEAR($WORD[102], 4)
  -- reset_ios
  -- check_home
ELSE
ENDIF
ENDIF
END gun_stepper2

```

```

ROUTINE gun_stepper1
BEGIN
  BIT_CLEAR($WORD[18], 1)
  IF BIT_TEST($WORD[102], 3, ON) THEN
    GOTO serv1
  ENDIF
  IF (stepper1 = TRUE) OR BIT_TEST($WORD[26], 13, ON) THEN
    -- BIT_SET($WORD[18], 5)
    BIT_SET($WORD[101], 9) -- bit 1
    $TERM_TYPE := FINE
    $ARM_SPD_OVR := 50
    $ARM_ACC_OVR := 50
    $ARM_DEC_OVR := 50
    tipdress
    $ARM_SPD_OVR := 100
    $ARM_ACC_OVR := 100
    $ARM_DEC_OVR := 100
  ELSE
serv1::

```

ADNAN OKYAR (2001-2009)

```

IF BIT_TEST($WORD[102], 3, ON) OR BIT_TEST($WORD[26], 9, ON) THEN
  BIT_SET($WORD[18], 3)
  BIT_SET($WORD[101], 9) -- bit 1
  $TERM_TYPE := FINE
  $ARM_SPD_OVR := 50
  $ARM_ACC_OVR := 50
  $ARM_DEC_OVR := 50
  service
  BIT_SET($WORD[101], 7)
  DELAY 500
  BIT_CLEAR($WORD[101], 7)
  BIT_CLEAR($WORD[18], 3)
  -- BIT_SET($WORD[18], 5)
  -- tipdress
  BIT_CLEAR($WORD[102], 3)
  -- reset_ios
  -- check_home
  $ARM_SPD_OVR := 100
  $ARM_ACC_OVR := 100
  $ARM_DEC_OVR := 100
ELSE
  ENDIF
ENDIF
END gun_stepper1

```

```

ROUTINE check_stepper2
BEGIN
  IF BIT_TEST($WORD[109], 5, ON) THEN
    stepper2 := TRUE
  ELSE
    IF BIT_TEST($WORD[109], 7, ON) THEN
      BIT_SET($WORD[102], 4)
      BIT_SET($WORD[18], 13)
      DELAY 500
      BIT_CLEAR($WORD[18], 13)

    ELSE
      IF BIT_TEST($WORD[109], 6, ON) THEN
        BIT_SET($WORD[18], 12)
        DELAY 500
        BIT_CLEAR($WORD[18], 12)

      ELSE
        ENDIF
      ENDIF
    ENDIF
  ENDIF
END check_stepper2

```

```

ROUTINE check_stepper1
BEGIN
  IF BIT_TEST($WORD[109], 5, ON) THEN
    stepper1 := TRUE
  ELSE
    IF BIT_TEST($WORD[109], 7, ON) THEN
      BIT_SET($WORD[102], 3)
      BIT_SET($WORD[18], 13)
      DELAY 500
      BIT_CLEAR($WORD[18], 13)

```

ADNAN OKYAR (2001-2009)

```

ELSE
  IF BIT_TEST($WORD[109], 6, ON) THEN
    BIT_SET($WORD[18], 12)
    DELAY 500
    BIT_CLEAR($WORD[18], 12)
  ELSE
    ENDIF
  ENDIF
ENDIF
END check_stepper1

```

ROUTINE main

```

BEGIN
  $TERM_TYPE := FINE
  reset_ios
  stepper1 := FALSE
  stepper2 := FALSE
  weld_init -- Initialize weld
  press(1)
  DELAY 500
  home_pos := pnt0001j
  SYS_CALL('ms', 'Irobot')
  home_check
  sendsig(home_position, ON)
jump1::
  $BIT[100] := OFF
  $BIT[101] := OFF
  $BIT[133] := OFF
  $BIT[134] := OFF
  reset_ios
  stepper1 := FALSE
  stepper2 := FALSE
  $TERM_TYPE := FINE
  incl_appl(ON)

  press(1)
  home_check
  sendsig(home_position, ON)
  WIN_CLEAR(WIN_CLR_ALL)
  WRITE ('ROBOT IS IN HOME POS', NL)
  WRITE (' ', NL)
  WRITE ('waitsig START PROGRAM', NL)
  WHILE BIT_TEST($WORD[26], 1, OFF) DO
    IF ($FDIN[34] = ON) AND (SCRN_GET(PDV_TP) = SCRN_USER) THEN -- AKO JE NEKO PRITISNUO
      DUGME NA USER_SCREEN-u
      WAIT FOR ($BIT[100] = TRUE) OR ($BIT[101] = TRUE) -- BIT 100 SET THE GUN, BIT 101 EXIT
      IF $BIT[101] = TRUE THEN -- AKO JE ODUSTAO OD PROMENE ELEKTRODE
        GOTO jump1
      ENDIF
      GOTO l_100
    ENDIF
    IF ($FDIN[33] = ON) AND (SCRN_GET(PDV_TP) = SCRN_USER) THEN
      WAIT FOR ($BIT[100] = TRUE) OR ($BIT[101] = TRUE)
      IF $BIT[101] = TRUE THEN
        GOTO jump1
      ENDIF
      GOTO l_101
    ENDIF
  ENDIF

```

ADNAN OKYAR (2001-2009)

```

ENDWHILE
waitsig(start_program, ON)
$TIMER[11] := 0
WIN_CLEAR(WIN_CLR_ALL)
WRITE ('waitsig PROGRAM CODE', NL)
SELECT code_prg OF
CASE (0):
    GOTO jump1
CASE (1):
    WIN_CLEAR(WIN_CLR_ALL)
    WRITE ('CODE 1 RECEIVED', NL)
    WRITE ('PROG_1 IN EXECUTION', NL) --
    WRITE ('swb_model_prog', NL) --
    -- swb
    model := 1
    l30lhr1_swb
    --
CASE (2):
    WIN_CLEAR(WIN_CLR_ALL)
    WRITE ('CODE 2 RECEIVED', NL)
    WRITE ('PROG_2 IN EXECUTION', NL) --
    WRITE ('lwb_model_prog', NL) --
    -- lwb
    model := 2
    l30lhr1_lwb
    --
CASE (32):
l_101:: WIN_CLEAR(WIN_CLR_ALL)
    WRITE ('CODE 32 RECEIVED', NL)
    WRITE ('PROG_34 IN EXECUTION', NL)
    WRITE ('Maintenance program', NL)
    BIT_CLEAR($WORD[18], 1)
    BIT_SET($WORD[18], 3)
    BIT_SET($WORD[18], 13)
    DELAY 3000
    BIT_CLEAR($WORD[18], 13)
    BIT_SET($WORD[101], 9) -- bit 1
    service
    BIT_SET($WORD[101], 7)
    DELAY 500
    BIT_CLEAR($WORD[101], 7)
    BIT_CLEAR($WORD[18], 3)
    BIT_SET($WORD[18], 5)
    BIT_SET($WORD[101], 9) -- bit 1
    tipdress
    $BIT[133] := TRUE
    BIT_CLEAR($WORD[18], 5)
    reset_ios
CASE (33):
l_100:: WIN_CLEAR(WIN_CLR_ALL)
    WRITE ('CODE 33 RECEIVED', NL)
    WRITE ('PROG_33 IN EXECUTION', NL) --
    WRITE ('Tipdress program', NL)
    DELAY 3000
    BIT_CLEAR($WORD[18], 1)
    -- BIT_SET($WORD[18], 5)
    BIT_SET($WORD[101], 9) -- bit 1
    tipdress

```

ADNAN OKYAR (2001-2009)

```

$BIT[134] := TRUE
BIT_CLEAR($WORD[18], 5)
reset_ios
ELSE:
  dcp_fault(2, 'prog_0')
  DELAY 500
  GOTO jump1
ENDSELECT

GOTO jump1
END main

BEGIN
--////////////////////////////////////
--\\ main program          \\
--////////////////////////////////////
main
END prog_0

```

```

PROGRAM prog_34 EZ, PROG_ARM = 1, STACK = 2048
VAR pnt0096p, pnt0097p, pnt0098p, pnt0099p : POSITION
pnt0091p, pnt0092p, pnt0093p, pnt0094p, pnt0095p : POSITION
pnt0086p, pnt0087p, pnt0088p, pnt0089p, pnt0090p : POSITION
pnt0081p, pnt0082p, pnt0083p, pnt0084p, pnt0085p : POSITION
pnt0076p, pnt0077p, pnt0078p, pnt0079p, pnt0080p : POSITION
pnt0071p, pnt0072p, pnt0073p, pnt0074p, pnt0075p : POSITION
pnt0066p, pnt0067p, pnt0068p, pnt0069p, pnt0070p : POSITION
pnt0061p, pnt0062p, pnt0063p, pnt0064p, pnt0065p : POSITION
pnt0056p, pnt0057p, pnt0058p, pnt0059p, pnt0060p : POSITION
pnt0051p, pnt0052p, pnt0053p, pnt0054p, pnt0055p : POSITION
pnt0046p, pnt0047p, pnt0048p, pnt0049p, pnt0050p : POSITION
pnt0041p, pnt0042p, pnt0043p, pnt0044p, pnt0045p : POSITION
pnt0036p, pnt0037p, pnt0038p, pnt0039p, pnt0040p : POSITION
pnt0031p, pnt0032p, pnt0033p, pnt0034p, pnt0035p : POSITION
pnt0026p, pnt0027p, pnt0028p, pnt0029p, pnt0030p : POSITION
pnt0021p, pnt0022p, pnt0023p, pnt0024p, pnt0025p : POSITION
pnt0016p, pnt0017p, pnt0018p, pnt0019p, pnt0020p : POSITION
pnt0011p, pnt0012p, pnt0013p, pnt0014p, pnt0015p : POSITION
pnt0006p, pnt0007p, pnt0008p, pnt0009p, pnt0010p : POSITION
pnt0001p, pnt0002p, pnt0003p, pnt0004p, pnt0005p : POSITION
CONST
  home_position = 1801
  workarea_free_to_op = 1912
  tipchange_running = 1803
  tipchange_pos = 1804
  req_tipchng = 2609
  tipchange_done = 2610
  robot_in_toolchange = 2005
VAR
  home_pos, rip0000j : JOINTPOS FOR ARM[1] EXPORTED FROM lrobot
  tip_reg : INTEGER
  choose_num_gun : INTEGER
  stepper1 : BOOLEAN EXPORTED FROM prog_0
  stepper2 : BOOLEAN EXPORTED FROM prog_0
  ADNAN OKYAR (2001-2009)

```

```

ROUTINE elektromak_2 EXPORTED FROM prog_34
ROUTINE elektromak EXPORTED FROM prog_34
ROUTINE press(ai_wprog : INTEGER) EXPORTED FROM sb_appl

ROUTINE tool_change(ai_tool : INTEGER) EXPORTED FROM tool_ch
ROUTINE tf(ai_t_idx, ai_f_num : INTEGER) : INTEGER EXPORTED FROM tt_tool
ROUTINE weld(ai_wprog : INTEGER; ab_before : BOOLEAN) : INTEGER EXPORTED FROM sb_appl
ROUTINE weld_init EXPORTED FROM sb_appl
ROUTINE gun_open_check EXPORTED FROM dgun
ROUTINE gun_open(ab_check : BOOLEAN) EXPORTED FROM dgun
ROUTINE waitsig(n : INTEGER; stato : BOOLEAN) EXPORTED FROM lrobot
ROUTINE sendsig(n : INTEGER; stato : BOOLEAN) EXPORTED FROM lrobot
ROUTINE dcp_fault(ai_numero_allarme : INTEGER; as_mod_name : STRING) EXPORTED FROM lrobot
ROUTINE a_tool(ai_t_idx : INTEGER) EXPORTED FROM tt_tool
ROUTINE check_home EXPORTED FROM prog_0
ROUTINE reset_ios EXPORTED FROM prog_0
ROUTINE gun_semi_open_check EXPORTED FROM dgun
ROUTINE gun_semi_open(ab_check : BOOLEAN) EXPORTED FROM dgun

```

```

ROUTINE service EXPORTED FROM prog_34
ROUTINE elektromak_2
BEGIN
  MOVE JOINT TO pnt0085p
  MOVE JOINT TO pnt0088p
  MOVE JOINT TO pnt0089p
  MOVE JOINT TO pnt0090p
  MOVE JOINT TO pnt0091p
  MOVE JOINT TO pnt0092p
  MOVE JOINT TO pnt0093p
  MOVE JOINT TO pnt0095p
  MOVE JOINT TO pnt0096p
END elektromak_2

```

```

ROUTINE elektromak
BEGIN
  $BIT[101] := ON
  $DOUT[32] := ON
  $DOUT[27] := ON
  $DOUT[20] := ON
  $DOUT[29] := ON
  MOVE JOINT TO pnt0036p
  MOVE JOINT TO pnt0037p
  MOVE JOINT TO pnt0038p
  MOVE JOINT TO pnt0039p
  MOVE JOINT TO pnt0040p
  MOVE JOINT TO pnt0041p
  MOVE JOINT TO pnt0042p
  MOVE JOINT TO pnt0043p
  $DOUT[28] := ON

```

ADNAN OKYAR (2001-2009)

```

MOVE JOINT TO pnt0044p
MOVE JOINT TO pnt0045p
MOVE JOINT TO pnt0046p
$DOUT[28] := OFF
MOVE JOINT TO pnt0047p
MOVE JOINT TO pnt0048p
MOVE JOINT TO pnt0049p
MOVE JOINT TO pnt0050p
$DOUT[28] := ON
MOVE JOINT TO pnt0051p
MOVE JOINT TO pnt0052p
MOVE JOINT TO pnt0053p
MOVE JOINT TO pnt0054p
MOVE JOINT TO pnt0055p
$DOUT[28] := OFF
MOVE JOINT TO pnt0056p
--EKEKTROT_KONTROL--
MOVE JOINT TO pnt0057p
--EKEKTROT_KONTROL--
MOVE JOINT TO pnt0058p
MOVE JOINT TO pnt0059p
--EKEKTROT_KONTROL--
MOVE JOINT TO pnt0060p
MOVE JOINT TO pnt0061p
MOVE JOINT TO pnt0062p
MOVE JOINT TO pnt0063p
MOVE JOINT TO pnt0064p
MOVE JOINT TO pnt0065p
MOVE JOINT TO pnt0066p
MOVE JOINT TO pnt0067p
MOVE JOINT TO pnt0068p
MOVE JOINT TO pnt0069p
MOVE JOINT TO pnt0070p
MOVE JOINT TO pnt0071p
MOVE JOINT TO pnt0072p
--EKEKTROT_KONTROL--
MOVE JOINT TO pnt0074p
--EKEKTROT_KONTROL--
MOVE JOINT TO pnt0075p
MOVE JOINT TO pnt0076p
--EKEKTROT_KONTROL--
MOVE JOINT TO pnt0078p
MOVE JOINT TO pnt0079p
MOVE JOINT TO pnt0080p
MOVE JOINT TO pnt0081p
MOVE JOINT TO pnt0082p
MOVE JOINT TO pnt0083p
MOVE JOINT TO pnt0084p
END elektromak

```

ROUTINE service

BEGIN

```

sendsig(robot_in_toolchange, OFF)
sendsig(home_position, OFF)
sendsig(workarea_free_to_op, ON)
sendsig(tipchange_running, ON)
BIT_SET($WORD[19], 11)

```

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



```

BIT_SET($WORD[19], 12)
$PAR := tf(0, 0)
MOVE TO home_pos

IF BIT_TEST($WORD[102], 3, ON) OR (stepper1 = TRUE) THEN
  --TABANCA 863 u AL--
  tool_change(1)
  sendsig(robot_in_toolchange, OFF)
  MOVE TO home_pos
  stepper1 := TRUE
  weld_init
  gun_open(ON)
  gun_open_check
  -- elektromak

  press(16)
  MOVE JOINT TO pnt0006p
  MOVE JOINT TO pnt0030p
  $PROG_SPD_OVR := 30
  sendsig(tipchange_pos, ON)
  -- waitsig(2807, ON)
  -- waitsig(2808, ON)
  MOVE JOINT TO pnt0007p
  $PROG_SPD_OVR := 100
  waitsig(tipchange_done, ON)
  -- MOVE JOINT TO pnt0003p
  -- waitsig(2809, ON)
  sendsig(tipchange_pos, OFF)
  PULSE $DOUT[17] FOR 500
  DELAY 500
  PULSE $DOUT[17] FOR 500
  MOVE JOINT TO pnt0006p
  MOVE TO home_pos
  a_tool(1)
  press(16)
  gun_open(ON)
  gun_open_check
  MOVEFLY JOINT TO pnt0011p ADVANCE
  MOVEFLY JOINT TO pnt0012p ADVANCE
  -- MOVE JOINT TO pnt0013p
  MOVE JOINT TO pnt0014p
  -- BIT_SET($WORD[102], 1)
  $DOUT[27] := ON
  BIT_SET($WORD[18], 5)
  MOVE JOINT TO pnt0031p
  MOVE JOINT TO pnt0015p
  MOVE LINEAR TO pnt0027p
  gun_semi_open(ON)
  gun_semi_open_check
  PULSE $DOUT[17] FOR 1000
  DELAY 500
  PULSE $DOUT[17] FOR 1000
  -- DELAY 500
  -- PULSE $DOUT[17] FOR 1000
  gun_open(ON)
  gun_open_check
  MOVE JOINT TO pnt0015p
  MOVE JOINT TO pnt0014p

```

ADNAN OKYAR (2001-2009)

```

-- MOVEFLY JOINT TO pnt0013p ADVANCE
BIT_SET($WORD[101], 6)
BIT_CLEAR($WORD[18], 5)
MOVEFLY JOINT TO pnt0012p ADVANCE
MOVEFLY JOINT TO pnt0011p ADVANCE
-- press(1)
PULSE $DOUT[17] FOR 1500
DELAY 500
PULSE $DOUT[17] FOR 1500
-- BIT_CLEAR($WORD[102], 1)
$DOUT[27] := OFF
BIT_CLEAR($WORD[101], 6)
stepper1 := FALSE
MOVE TO home_pos
ELSE
IF BIT_TEST($WORD[102], 4, ON) OR (stepper2 = TRUE) THEN
--TABANCA 864 u AL--
tool_change(2)
sendsig(robot_in_toolchange, OFF)
MOVE TO home_pos
MOVE JOINT TO pnt0010p
stepper2 := TRUE
press(18)
weld_init
gun_open(ON)
gun_open_check
-- elektromak_2

-- MOVE JOINT TO pnt0004p
-- MOVE JOINT TO pnt0005p
$PROG_SPD_OVR := 20
sendsig(tipchange_pos, ON)
-- waitsig(2807, ON)
-- waitsig(2808, ON)
MOVE JOINT TO pnt0008p
waitsig(tipchange_done, ON)
MOVE JOINT TO pnt0035p

$PROG_SPD_OVR := 100
-- MOVE JOINT TO pnt0004p
-- waitsig(2809, ON)
MOVE JOINT TO pnt0032p
sendsig(tipchange_pos, OFF)
MOVE TO home_pos
----KALEMTRAS----
a_tool(2)
press(1)
gun_semi_open(ON)
gun_semi_open_check
MOVEFLY JOINT TO pnt0016p ADVANCE
MOVEFLY JOINT TO pnt0017p ADVANCE
MOVEFLY JOINT TO pnt0018p ADVANCE
MOVE LINEAR TO pnt0099p
MOVE JOINT TO pnt0020p
MOVE LINEAR TO pnt0097p
MOVE LINEAR TO pnt0026p
press(1)
MOVE JOINT TO pnt0033p

```

ADNAN OKYAR (2001-2009)

```

-- BIT_SET($WORD[102], 1)
$DOUT[27] := ON
BIT_SET($WORD[18], 5)
press(18)
PULSE $DOUT[17] FOR 1000
DELAY 500
PULSE $DOUT[17] FOR 1000
DELAY 500
PULSE $DOUT[17] FOR 1000
DELAY 500
gun_open(ON)
DELAY 500
gun_open_check
MOVE LINEAR TO pnt0026p
MOVE LINEAR TO pnt0097p
gun_semi_open(ON)
gun_semi_open_check
MOVE LINEAR TO pnt0098p
BIT_SET($WORD[101], 6)
MOVE JOINT TO pnt0018p
PULSE $DOUT[17] FOR 500
DELAY 500
PULSE $DOUT[17] FOR 500
BIT_CLEAR($WORD[18], 5)
MOVEFLY JOINT TO pnt0017p ADVANCE
BIT_CLEAR($WORD[101], 6)
-- BIT_CLEAR($WORD[102], 1)
$DOUT[27] := OFF
MOVEFLY JOINT TO pnt0016p ADVANCE
stepper2 := FALSE
MOVE TO home_pos
ENDIF
ENDIF

END service

BEGIN
service
END prog_34

```

```

PROGRAM prog_1 EZ, PROG_ARM = 1, STACK = 2048
VAR pnt0295p, pnt0296p : POSITION
jnt0007j, jnt0008j : JOINTPOS FOR ARM[1]
pnt0290p, pnt0291p, pnt0292p, pnt0293p, pnt0294p : POSITION
pnt0285p, pnt0286p, pnt0287p, pnt0288p, pnt0289p : POSITION
pnt0279p, pnt0281p, pnt0282p, pnt0283p, pnt0284p : POSITION
pnt0274p, pnt0275p, pnt0276p, pnt0277p, pnt0278p : POSITION
pnt0269p, pnt0270p, pnt0271p, pnt0272p, pnt0273p : POSITION
pnt0264p, pnt0265p, pnt0266p, pnt0267p, pnt0268p : POSITION
pnt0259p, pnt0260p, pnt0261p, pnt0262p, pnt0263p : POSITION
pnt0254p, pnt0255p, pnt0256p, pnt0257p, pnt0258p : POSITION
pnt0249p, pnt0250p, pnt0251p, pnt0252p, pnt0253p : POSITION
pnt0244p, pnt0245p, pnt0246p, pnt0247p, pnt0248p : POSITION
pnt0239p, pnt0240p, pnt0241p, pnt0242p, pnt0243p : POSITION
pnt0234p, pnt0235p, pnt0236p, pnt0237p, pnt0238p : POSITION
pnt0229p, pnt0230p, pnt0231p, pnt0232p, pnt0233p : POSITION
pnt0214p, pnt0215p, pnt0216p, pnt0217p, pnt0218p : POSITION
pnt0209p, pnt0210p, pnt0211p, pnt0212p, pnt0213p : POSITION

```

ADNAN OKYAR (2001-2009)

pnt0204p, pnt0205p, pnt0206p, pnt0207p, pnt0208p : POSITION
 pnt0199p, pnt0200p, pnt0201p, pnt0202p, pnt0203p : POSITION
 pnt0192p, pnt0193p, pnt0194p, pnt0195p, pnt0198p : POSITION
 pnt0187p, pnt0188p, pnt0189p, pnt0190p, pnt0191p : POSITION
 pnt0182p, pnt0183p, pnt0184p, pnt0185p, pnt0186p : POSITION
 pnt0177p, pnt0178p, pnt0179p, pnt0180p, pnt0181p : POSITION
 pnt0171p, pnt0172p, pnt0173p, pnt0174p, pnt0175p : POSITION
 jnt0002j, jnt0003j, jnt0004j, jnt0005j, jnt0006j : JOINTPOS FOR ARM[1]
 pnt0118p, pnt0119p, pnt0120p, pnt0122p, pnt0123p : POSITION
 wp6329_1_2p, wp6331_0_1p, wp6331_0_2p, wp6332_0_1p, wp6332_0_2p : POSITION EXPORTED FROM spot_lib
 wp6221_0_4p, wp6221_0_5p, wp6221_0_6p, wp6328_1_2p, wp6329_1_1p : POSITION EXPORTED FROM spot_lib
 wp6220_0_3p, wp6220_0_4p, wp6220_0_5p, wp6220_0_6p, wp6221_0_3p : POSITION EXPORTED FROM spot_lib
 wp6216_1_6p, wp6217_1_3p, wp6217_1_4p, wp6217_1_5p, wp6217_1_6p : POSITION EXPORTED FROM spot_lib
 pnt0113p, pnt0115p, pnt0116p, pnt0117p, pnt0124p : POSITION
 pnt0109p, pnt0111p, pnt0125p, pnt0126p, pnt0127p : POSITION
 pnt0099p, pnt0100p, pnt0101p, pnt0128p, pnt0129p : POSITION
 pnt0093p, pnt0094p, pnt0095p, pnt0130p, pnt0131p : POSITION
 str_all : ARRAY[50] OF STRING[80] EXPORTED FROM Irobot
 lrvb_fault : BOOLEAN EXPORTED FROM Irobot NOSAVE
 home_pos, rip0000j : JOINTPOS FOR ARM[1] EXPORTED FROM Irobot
 pnt0087p, pnt0132p, pnt0133p, pnt0134p, pnt0135p : POSITION
 pnt0082p, pnt0085p, pnt0136p, pnt0137p, pnt0138p : POSITION
 pnt0076p, pnt0077p, pnt0079p, pnt0080p, pnt0139p : POSITION
 pnt0071p, pnt0074p, pnt0075p, pnt0140p, pnt0141p : POSITION
 pnt0069p, pnt0070p, pnt0142p, pnt0143p, pnt0144p : POSITION
 wp6314_3_9_t_vk_abp, wp6316_3_1_t_vk_abp, wp6316_3_2_t_vk_abp, wp6316_3_3_t_vk_abp,
 wp6316_3_4_t_vk_abp : POSITION EXPORTED FROM ez_ulib
 wp6314_3_4_t_vk_abp, wp6314_3_5_t_vk_abp, wp6314_3_6_t_vk_abp, wp6314_3_7_t_vk_abp,
 wp6314_3_8_t_vk_abp : POSITION EXPORTED FROM ez_ulib
 wp6308_1_1_t_vk_abp, wp6310_0_5_t_vk_abp, wp6310_0_7_t_vk_abp, wp6314_3_10_t_vk_abp,
 wp6314_3_3_t_vk_abp : POSITION EXPORTED FROM ez_ulib
 wp6314_3_1_t_vk_abp, wp6314_3_2_t_vk_abp : POSITION EXPORTED FROM ez_ulib
 pnt0056p, pnt0057p, pnt0145p, pnt0146p, pnt0147p : POSITION
 pnt0052p, pnt0053p, pnt0054p, pnt0055p, pnt0148p : POSITION
 pnt0047p, pnt0048p, pnt0149p, pnt0150p, pnt0151p : POSITION
 pnt0045p, pnt0152p, pnt0153p, pnt0154p, pnt0155p : POSITION
 pnt0036p, pnt0038p, pnt0156p, pnt0157p, pnt0158p : POSITION
 pnt0033p, pnt0159p, pnt0160p, pnt0161p, pnt0162p : POSITION
 pnt0025p, pnt0163p, pnt0164p, pnt0165p, pnt0166p : POSITION
 pnt0012p, pnt0013p, pnt0167p, pnt0168p, pnt0169p : POSITION
 pnt0007p, pnt0008p, pnt0009p, pnt0010p, pnt0170p : POSITION
 gun_1_selected : BOOLEAN EXPORTED FROM screen
 gun_2_selected : BOOLEAN EXPORTED FROM screen
 ----- SPOT WELDING POINTS -----

VAR wp6306_1_2p, wp6311_0_5p, wp6311_0_7p, wp6315_3_1p, wp6315_3_2p : POSITION EXPORTED FROM spot_lib
 wp6315_3_3p, wp6315_3_4p, wp6315_3_5p, wp6315_3_6p, wp6315_3_7p : POSITION EXPORTED FROM spot_lib
 wp6315_3_8p, wp6315_3_9p, wp6315_3_10p, wp6315_3_11p, wp6315_3_12p : POSITION EXPORTED FROM spot_lib
 wp6317_3_1p, wp6317_3_3p, wp6317_3_4p, wp6317_3_6p, wp6317_3_7p : POSITION EXPORTED FROM spot_lib
 wp6317_3_9p, wp6319_3_1p, wp6308_1_1p, wp6310_0_5p, wp6310_0_7p : POSITION EXPORTED FROM spot_lib
 wp6314_3_1p, wp6314_3_2p, wp6314_3_3p, wp6314_3_4p, wp6314_3_5p : POSITION EXPORTED FROM spot_lib
 wp6314_3_6p, wp6314_3_7p, wp6314_3_8p, wp6314_3_9p, wp6314_3_10p : POSITION EXPORTED FROM spot_lib
 wp6316_3_1p, wp6316_3_2p, wp6316_3_3p, wp6316_3_4p, wp6216_1_5p : POSITION EXPORTED FROM spot_lib

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



CONST

home_position = 1801
end_of_program = 1806
program_in_execution = 1807
end_of_weld_1 = 1809
end_of_weld_2 = 1810
tipdressing = 1805
changing_electrode = 1803
workarea_free_from_op = 2712
workarea_free_to_op = 1912

arkakenedi_ac = 1913
arkakenet_acildi = 2713

zone2_1_free_from_r1 = 2704
zone2_1_free_to_r1 = 1904
zone2_2_free_from_r1 = 2705
zone2_2_free_to_r1 = 1905
zone3_3_free_from_r1 = 2706
zone3_3_free_to_r1 = 1906
zone4_4_free_from_r1 = 2709
zone4_4_free_to_r1 = 1909
priority_job_from_r1 = 2710
priority_job_to_r1 = 1911
zone5_5_free_from_r1 = 2708
zone5_5_free_to_r1 = 1908
zone6_6_free_from_r1 = 2711
zone6_6_free_to_r1 = 1910

tipdress_done = 6
tipdress_memory = 85
tipchange_memory = 86
tipchange_done = 7
tipchange_pre_alarm = 1812
tipchange_position = 1804
tipchange_req_from_plc = 2609
tipchange_done_from_plc = 2610
tipdress_req_from_plc = 2613

r_81_bypassli = 11
gun_1_bypassed = 14
gun_2_bypassed = 15

gun1_tool_change = 2013
gun2_tool_change = 2014

ROUTINE r_4 EXPORTED FROM prog_1
ROUTINE r_1_a EXPORTED FROM prog_1
ROUTINE devami EXPORTED FROM prog_1
ROUTINE r_1 EXPORTED FROM prog_1
ROUTINE r_2 EXPORTED FROM prog_1
ROUTINE roof_swb_citali EXPORTED FROM prog_1

ROUTINE beck_up_cgun EXPORTED FROM prog_1
ROUTINE beck_up EXPORTED FROM prog_1
ROUTINE a_frame(ai_f_num : INTEGER) EXPORTED FROM tu_frame
ROUTINE before_hand(ai_time : INTEGER) EXPORTED FROM sb_appl
ROUTINE open_cover_gun_1 EXPORTED FROM ez_ulib

ADNAN OKYAR (2001-2009)

```

ROUTINE check_cover_gun_1_open EXPORTED FROM ez_ulib
ROUTINE close_cover_gun_1 EXPORTED FROM ez_ulib
ROUTINE check_cover_gun_1_close EXPORTED FROM ez_ulib
ROUTINE open_cover_gun_2 EXPORTED FROM ez_ulib
ROUTINE check_cover_gun_2_open EXPORTED FROM ez_ulib
ROUTINE close_cover_gun_2 EXPORTED FROM ez_ulib
ROUTINE check_cover_gun_2_close EXPORTED FROM ez_ulib
ROUTINE pre_alarm_test EXPORTED FROM prog_1
ROUTINE rear_roof_swb EXPORTED FROM prog_1
ROUTINE roof_swb EXPORTED FROM prog_1
ROUTINE init_x_gun_horiz_high EXPORTED FROM gun_lib
ROUTINE init_c_gun_vert_high EXPORTED FROM gun_lib
ROUTINE tool_change(ai_tool : INTEGER) EXPORTED FROM tool_ch
ROUTINE call_prog_34 EXPORTED FROM prog_34
ROUTINE call_prog_33 EXPORTED FROM prog_33
ROUTINE call_prog_78 EXPORTED FROM prog_78
ROUTINE call_prog_77 EXPORTED FROM prog_77
ROUTINE horiz_balance(ab_order : BOOLEAN) EXPORTED FROM dgun
ROUTINE gun_semi_open_check EXPORTED FROM dgun
ROUTINE gun_semi_open(ab_check : BOOLEAN) EXPORTED FROM dgun
ROUTINE gun_open_check EXPORTED FROM dgun
ROUTINE gun_open_appr : INTEGER EXPORTED FROM dgun
ROUTINE gun_open(ab_check : BOOLEAN) EXPORTED FROM dgun
ROUTINE gun_close_appr : INTEGER EXPORTED FROM dgun
ROUTINE gun_close(ab_check : BOOLEAN) EXPORTED FROM dgun
ROUTINE weld(ai_wprog : INTEGER; ab_before : BOOLEAN) : INTEGER EXPORTED FROM sb_appl
ROUTINE weld_init EXPORTED FROM sb_appl
ROUTINE a_tool(ai_t_idx : INTEGER) EXPORTED FROM tt_tool
ROUTINE waitsig(n : INTEGER; stato : BOOLEAN) EXPORTED FROM lrobot
ROUTINE ver EXPORTED FROM lrobot
ROUTINE sendsig(n : INTEGER; stato : BOOLEAN) EXPORTED FROM lrobot
ROUTINE home_check EXPORTED FROM lrobot
ROUTINE dcp_fault(ai_numero_allarme : INTEGER; as_mod_name : STRING) EXPORTED FROM lrobot
ROUTINE code_prg : INTEGER EXPORTED FROM lrobot
ROUTINE call_prog_1 EXPORTED FROM prog_1

```

```

ROUTINE r_4
BEGIN
  gun_open(ON)
  gun_open_check
  MOVE JOINT TO pnt0281p
  MOVE JOINT TO pnt0282p
  MOVE JOINT TO pnt0283p,
    WITH CONDITION[weld(1, FALSE)],
  ENDMOVE
  gun_open(ON)
  gun_open_check
  MOVE JOINT TO pnt0284p
  MOVE JOINT TO pnt0285p
  MOVE JOINT TO pnt0286p,
    WITH CONDITION[weld(1, FALSE)],
  ENDMOVE
  gun_open(ON)
  gun_open_check
  MOVE JOINT TO pnt0287p
  MOVE JOINT TO pnt0290p
  MOVE JOINT TO pnt0288p,
    WITH CONDITION[weld(1, FALSE)],

```

ADNAN OKYAR (2001-2009)

```

ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0289p
MOVE JOINT TO pnt0291p
END r_4

```

```

ROUTINE r_1_a
BEGIN
  waitsig(zone5_5_free_from_r1, ON)
  sendsig(zone5_5_free_to_r1, OFF)
  MOVE JOINT TO pnt0294p
  MOVE JOINT TO pnt0295p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  MOVE JOINT TO pnt0245p
  MOVE JOINT TO pnt0246p,
    WITH CONDITION[weld(1, FALSE)], -- Welding operation
  ENDMOVE
  gun_semi_open_check
  MOVE JOINT TO pnt0247p
  MOVE JOINT TO pnt0248p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  gun_semi_open_check
  MOVE JOINT TO pnt0249p
  ---KENET_AC---
  sendsig(arkakenedi_ac, ON)
  waitsig(arkakenet_acildi, ON)
  MOVE JOINT TO pnt0250p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  sendsig(arkakenedi_ac, OFF)
  gun_semi_open_check
  MOVE JOINT TO pnt0251p
  MOVE JOINT TO pnt0252p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  gun_semi_open_check
  MOVE JOINT TO pnt0253p
  MOVE JOINT TO pnt0254p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  MOVE JOINT TO pnt0259p
  MOVE JOINT TO pnt0260p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  DELAY 500
  gun_open(ON)
  gun_open_check
  MOVE JOINT TO pnt0255p
  MOVE JOINT TO pnt0256p
  gun_semi_open(ON)
  gun_semi_open_check
  sendsig(zone5_5_free_to_r1, ON)
END r_1_a

```

ROUTINE devami

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net www.powerotomasyon.net



```
BEGIN
END devami
```

```
ROUTINE r_1
```

```
BEGIN
  waitsig(zone5_5_free_from_r1, ON)
  sendsig(zone5_5_free_to_r1, OFF)
  MOVE JOINT TO pnt0292p
  MOVE JOINT TO pnt0293p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  MOVE JOINT TO pnt0236p
  MOVE JOINT TO pnt0237p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  MOVE JOINT TO pnt0238p
  MOVE JOINT TO pnt0239p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  MOVE JOINT TO pnt0240p
  MOVE JOINT TO pnt0241p,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  MOVE JOINT TO jnt0003j,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  MOVE JOINT TO jnt0004j,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  gun_open(ON)
  gun_open_check
  MOVE JOINT TO jnt0005j,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  MOVE JOINT TO jnt0006j,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  MOVE JOINT TO jnt0007j,
    WITH CONDITION[weld(2, FALSE)],
  ENDMOVE
  gun_open(ON)
  gun_open_check
  MOVE JOINT TO jnt0008j
  MOVE JOINT TO pnt0244p
  MOVE JOINT TO pnt0243p
  gun_semi_open(ON)
  gun_semi_open_check
  sendsig(zone5_5_free_to_r1, ON)
END r_1
```

```
ROUTINE r_2
```

```
BEGIN
END r_2
```

```
ROUTINE roof_swb_citali
```

```
BEGIN
  a_tool(1) -- select tool number
  waitsig(zone2_2_free_from_r1, ON)
```

ADNAN OKYAR (2001-2009)

```

DELAY 200
waitsig(zone2_2_free_from_r1, ON)
sendsig(zone2_2_free_to_r1, OFF)
$TERM_TYPE := FINE
$ARM_ACC_OVR := 85
$ARM_DEC_OVR := 85
MOVEFLY JOINT TO pnt0007p ADVANCE
MOVEFLY JOINT TO pnt0071p ADVANCE
MOVEFLY JOINT TO pnt0008p ADVANCE
MOVEFLY JOINT TO pnt0009p ADVANCE
MOVEFLY JOINT TO pnt0010p ADVANCE
MOVEFLY JOINT TO pnt0093p ADVANCE
MOVE JOINT TO pnt0263p
MOVE JOINT TO pnt0265p
pre_alarm_test
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO pnt0264p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
MOVE JOINT TO pnt0266p
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0267p
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO pnt0268p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0269p
MOVE JOINT TO pnt0270p
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO pnt0271p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check

MOVE JOINT TO pnt0272p
-- r_4
waitsig(zone6_6_free_from_r1, ON)
sendsig(zone6_6_free_to_r1, OFF)
-- MOVE JOINT TO pnt0273p
gun_open(ON)
gun_open_check
-- MOVE JOINT TO pnt0273p
waitsig(zone2_2_free_from_r1, ON)
sendsig(zone2_2_free_to_r1, OFF)
sendsig(zone6_6_free_to_r1, ON)
MOVE JOINT TO pnt0172p
MOVE JOINT TO wp6316_3_1p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
-----
-----

```

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net



```

MOVE JOINT TO pnt0119p
MOVE JOINT TO wp6316_3_3p,
ENDMOVE
DELAY 300
MOVE JOINT TO wp6316_3_3p,
  WITH CONDITION[weld(11, FALSE)],
ENDMOVE
MOVE JOINT TO wp6316_3_4p,
ENDMOVE
DELAY 300
MOVE JOINT TO wp6316_3_4p,
  WITH CONDITION[weld(11, FALSE)],
ENDMOVE
MOVE JOINT TO pnt0213p,
  WITH CONDITION[weld(12, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0025p
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO wp6314_3_2p,
ENDMOVE

DELAY 300
MOVE JOINT TO wp6314_3_2p,
  WITH CONDITION[weld(12, FALSE)],
ENDMOVE
MOVE JOINT TO pnt0123p
MOVEFLY LINEAR TO pnt0101p ADVANCE
waitsig(zone2_1_free_from_r1, ON)
sendsig(zone2_1_free_to_r1, OFF)
MOVE LINEAR TO wp6217_1_3p,
ENDMOVE
MOVE LINEAR TO wp6217_1_3p,
  WITH CONDITION[weld(1, FALSE)],
ENDMOVE

-----
---R81'IN_PUNTALARINI_ATACAK---
-----
IF BIT_TEST($WORD[28], r_81_bypassli) = ON THEN
  r_1
ENDIF
-----
MOVE LINEAR TO wp6217_1_4p,
ENDMOVE
DELAY 300
sendsig(zone2_1_free_to_r1, ON)
MOVE LINEAR TO wp6217_1_4p,
  WITH CONDITION[weld(1, FALSE)], -- Welding operation
ENDMOVE
---
MOVEFLY JOINT TO pnt0124p ADVANCE
MOVE JOINT TO pnt0125p
MOVE JOINT TO pnt0185p
DELAY 300
MOVE LINEAR TO wp6329_1_1p,
  WITH CONDITION[weld(7, FALSE)],

```

ADNAN OKYAR (2001-2009)

```

ENDMOVE
gun_open(ON)
gun_open_check
MOVEFLY JOINT TO pnt0094p ADVANCE
MOVEFLY JOINT TO pnt0079p ADVANCE
MOVEFLY JOINT TO pnt0080p ADVANCE
MOVE JOINT TO pnt0115p
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO wp6217_1_6p,
ENDMOVE
DELAY 300
waitsig(zone2_1_free_from_r1, ON)
sendsig(zone2_1_free_to_r1, OFF)
MOVE JOINT TO wp6217_1_6p,
  WITH CONDITION[weld(1, FALSE)], -- Welding operation
ENDMOVE

```

 ---R81'IN_PUNTALARINI_ATACAK---

```

IF BIT_TEST($WORD[28], r_81_bypassli) = ON THEN
  r_1_a
ENDIF
MOVE LINEAR TO pnt0109p
sendsig(zone2_1_free_to_r1, ON)
MOVE LINEAR TO wp6217_1_5p,
ENDMOVE
DELAY 300
MOVE LINEAR TO wp6217_1_5p,
  WITH CONDITION[weld(14, FALSE)],
ENDMOVE
MOVEFLY LINEAR TO pnt0111p ADVANCE
-- MOVE JOINT TO pnt0217p
-- MOVE JOINT TO pnt0218p
-- MOVE JOINT TO pnt0126p
MOVE LINEAR TO wp6329_1_2p,
ENDMOVE
DELAY 300
MOVE LINEAR TO wp6329_1_2p,
  WITH CONDITION[weld(7, FALSE)],
ENDMOVE
MOVE LINEAR TO pnt0120p
MOVE JOINT TO wp6314_3_3p,
ENDMOVE
DELAY 300
MOVE JOINT TO wp6314_3_3p,
  WITH CONDITION[weld(12, FALSE)],
ENDMOVE
MOVE JOINT TO wp6314_3_4p,
ENDMOVE
DELAY 300
MOVE JOINT TO wp6314_3_4p,
  -- WITH CONDITION[weld(2, FALSE)],
ENDMOVE
MOVE JOINT TO wp6314_3_6p,
ENDMOVE
DELAY 300
MOVE JOINT TO wp6314_3_6p,

```

ADNAN OKYAR (2001-2009)

```

    WITH CONDITION[weld(12, FALSE)],
ENDMOVE
MOVE JOINT TO wp6314_3_8p,
ENDMOVE
DELAY 300
MOVE JOINT TO wp6314_3_8p,
    WITH CONDITION[weld(12, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0033p
MOVEFLY JOINT TO pnt0074p ADVANCE
-----
--BECK UP FOR SWB--
-----
-- beck_up
-----
MOVEFLY JOINT TO pnt0075p ADVANCE
MOVEFLY JOINT TO pnt0076p ADVANCE
MOVE LINEAR TO pnt0296p
sendsig(end_of_weld_1, ON)
-- sendsig(end_of_program, ON)
sendsig(zone2_1_free_to_r1, ON)
sendsig(zone2_2_free_to_r1, ON)
-----
open_cover_gun_1
check_cover_gun_1_open
-----
MOVEFLY JOINT TO pnt0077p ADVANCE
MOVEFLY JOINT TO pnt0116p ADVANCE
MOVE JOINT TO pnt0117p
IF ($DIN[70] = ON) OR (gun_1_selected = ON) AND BIT_TEST($WORD[26], 13, ON) THEN
    $BIT[tipdress_memory] := ON
ENDIF --
IF ($DIN[72] = ON) OR (gun_1_selected = ON) AND BIT_TEST($WORD[26], 9, ON) THEN
    $BIT[tipchange_memory] := ON
ENDIF
END roof_swb_citali

ROUTINE beck_up_cgun
BEGIN
    a_tool(2)
    -- MOVEFLY JOINT TO pnt0047p ADVANCE
    -- MOVEFLY JOINT TO pnt0045p ADVANCE
    -- MOVE JOINT TO pnt0165p
    MOVE JOINT TO pnt0166p,
        -- WITH CONDITION[weld(4, FALSE)],
    ENDMOVE
    MOVE JOINT TO pnt0167p,
        -- WITH CONDITION[weld(4, FALSE)],
    ENDMOVE
    MOVE JOINT TO pnt0168p
    MOVE JOINT TO pnt0170p
    IF BIT_TEST($WORD[28], 16, ON) THEN
        -----ROOFF FLAP-----
        MOVE JOINT TO pnt0189p
        MOVE JOINT TO pnt0190p
        MOVE JOINT TO pnt0191p

```

ADNAN OKYAR (2001-2009)

```

MOVE JOINT TO pnt0192p
MOVE JOINT TO pnt0193p
MOVE JOINT TO pnt0194p,
ENDMOVE
MOVE JOINT TO pnt0195p,
ENDMOVE
MOVE JOINT TO pnt0198p,
ENDMOVE
MOVE JOINT TO pnt0199p,
ENDMOVE
MOVE JOINT TO pnt0200p
MOVE JOINT TO pnt0201p,
ENDMOVE
MOVE JOINT TO pnt0202p
MOVE JOINT TO pnt0203p,
ENDMOVE
MOVE JOINT TO pnt0204p
MOVE JOINT TO pnt0205p
MOVE JOINT TO pnt0206p
MOVE JOINT TO pnt0207p
MOVE JOINT TO pnt0208p
MOVE JOINT TO pnt0209p
-----
ENDIF
END beck_up_cgun

ROUTINE beck_up
BEGIN
a_tool(1)
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0132p
MOVE JOINT TO pnt0133p
MOVE JOINT TO pnt0134p
MOVE JOINT TO pnt0135p
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO pnt0136p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
MOVE JOINT TO pnt0137p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
MOVEFLY JOINT TO pnt0138p ADVANCE
MOVEFLY JOINT TO pnt0139p ADVANCE
MOVE JOINT TO pnt0140p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0141p
MOVE JOINT TO pnt0142p
MOVE JOINT TO pnt0143p
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO pnt0144p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE

```

ADNAN OKYAR (2001-2009)

```

MOVE JOINT TO pnt0145p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
MOVEFLY JOINT TO pnt0146p ADVANCE
MOVEFLY JOINT TO pnt0147p ADVANCE
MOVE JOINT TO pnt0149p
MOVE JOINT TO pnt0148p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
MOVE JOINT TO pnt0150p
MOVE JOINT TO pnt0151p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
MOVE JOINT TO pnt0152p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
MOVE JOINT TO pnt0153p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
MOVE JOINT TO pnt0155p
MOVE JOINT TO pnt0154p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0156p
MOVE JOINT TO pnt0157p
MOVE JOINT TO pnt0158p
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO pnt0159p,
  WITH CONDITION[weld(2, FALSE)], -- Welding operation
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0160p
MOVE JOINT TO pnt0161p
END beck_up
ROUTINE pre_alarm_test
BEGIN
IF $DIN[71] = ON THEN -- Stepper pre-alarm
  sendsig(tipchange_pre_alarm, ON)
  IF BIT_TEST($WORD[26], 11, ON) THEN
    sendsig(2005, ON)
  ELSE

    IF BIT_TEST($WORD[26], 12, ON) THEN
      sendsig(2006, ON)
    ELSE
      ENDIF
    ENDIF
  ELSE
    IF $DIN[71] = OFF THEN
      sendsig(tipchange_pre_alarm, OFF)
      sendsig(2005, OFF)
      sendsig(2006, OFF)

    ENDIF
  ENDIF

```

ADNAN OKYAR (2001-2009)

```

ENDIF
END pre_alarm_test

ROUTINE rear_roof_swb
BEGIN
  a_tool(2) -- select tool number
  $ARM_SPD_OVR := 100
  $ARM_ACC_OVR := 100
  $ARM_DEC_OVR := 100
  init_x_gun_horiz_high
  weld_init -- init. for welding
  gun_semi_open(ON)
  gun_semi_open_check
  waitsig(zone5_5_free_from_r1, ON)
  sendsig(zone5_5_free_to_r1, OFF)
  IF BIT_TEST($WORD[28], 16, ON) THEN
    -----ROOFF FLAP-----
    $ARM_SPD_OVR := 100
    $ARM_ACC_OVR := 85
    $ARM_DEC_OVR := 85
    waitsig(priority_job_from_r1, ON)
    sendsig(priority_job_to_r1, OFF)
    waitsig(zone2_1_free_from_r1, ON)
    sendsig(zone2_1_free_to_r1, OFF)
    MOVE JOINT TO pnt0212p
    MOVEFLY JOINT TO pnt0048p ADVANCE
    MOVEFLY JOINT TO pnt0189p ADVANCE
    MOVEFLY JOINT TO pnt0190p ADVANCE
    MOVEFLY JOINT TO pnt0191p ADVANCE
    MOVEFLY JOINT TO pnt0192p ADVANCE
    MOVE JOINT TO pnt0193p
    MOVE JOINT TO pnt0194p,
      WITH CONDITION[weld(5, FALSE)],
    ENDMOVE
    MOVE JOINT TO pnt0195p,
      WITH CONDITION[weld(6, FALSE)],
    ENDMOVE
    MOVE JOINT TO pnt0198p,
      WITH CONDITION[weld(6, FALSE)],
    ENDMOVE
    MOVE JOINT TO pnt0199p,
      WITH CONDITION[weld(6, FALSE)],
    ENDMOVE
    MOVE JOINT TO pnt0200p
    MOVE JOINT TO pnt0215p
    MOVE JOINT TO pnt0201p,
      WITH CONDITION[weld(6, FALSE)],
    ENDMOVE
    MOVE JOINT TO pnt0202p
    MOVE JOINT TO pnt0203p,
      WITH CONDITION[weld(6, FALSE)],
    ENDMOVE
    MOVEFLY JOINT TO pnt0208p ADVANCE
    MOVEFLY JOINT TO pnt0209p ADVANCE
    MOVEFLY JOINT TO pnt0210p ADVANCE
    MOVEFLY JOINT TO pnt0211p ADVANCE
    -----
    GOTO l_41
  
```

ADNAN OKYAR (2001-2009)

```

ENDIF
MOVE JOINT TO pnt0131p
waitsig(zone2_1_free_from_r1, ON)
-- waitsig(zone2_2_free_from_r1, ON)
-- sendsig(zone2_2_free_to_r1, OFF)
-----
MOVEFLY JOINT TO pnt0048p ADVANCE
----CYCLE TIME 140 SN.-----
GOTO l_60
MOVEFLY JOINT TO pnt0047p ADVANCE
MOVEFLY JOINT TO pnt0045p ADVANCE
MOVE JOINT TO wp6310_0_5p,
  WITH CONDITION[weld(4, FALSE)],
ENDMOVE
pre_alarm_test
MOVEFLY JOINT TO pnt0052p ADVANCE
MOVEFLY JOINT TO pnt0085p ADVANCE
MOVE JOINT TO wp6310_0_7p,
  WITH CONDITION[weld(4, FALSE)],
ENDMOVE
l_60::
----CYCLE TIME 140 SN.-----
gun_open(ON)
gun_open_check
-- sendsig(zone2_1_free_to_r1, OFF)
-- waitsig(zone2_1_free_from_r1, ON)
MOVE JOINT TO wp6308_1_1p,
  WITH CONDITION[weld(4, FALSE)],
ENDMOVE
MOVE JOINT TO pnt0053p
--BECK UP FOR swb_Cgun--
-----
IF BIT_TEST($WORD[28], 9, ON) THEN
  waitsig(zone4_4_free_from_r1, ON)
  sendsig(zone4_4_free_to_r1, OFF)
  beck_up_cgun
  sendsig(zone4_4_free_to_r1, ON)
ELSE
ENDIF
sendsig(end_of_weld_2, ON)
sendsig(workarea_free_to_op, ON)
sendsig(end_of_program, ON)
MOVEFLY JOINT TO pnt0054p ADVANCE
l_41::
MOVE JOINT TO pnt0056p
-- sendsig(zone2_1_free_to_r1, ON)
IF BIT_TEST($WORD[28], 16, ON) THEN
  sendsig(zone2_1_free_to_r1, ON)
  sendsig(priority_job_to_r1, ON)
  sendsig(end_of_weld_2, ON)
  sendsig(end_of_program, ON)
  sendsig(workarea_free_to_op, ON)
  -- GOTO gun1_1
ENDIF
IF ($DIN[70] = ON) OR (gun_2_selected = ON) AND BIT_TEST($WORD[26], 13, ON) THEN
  $BIT[33] := OFF
  call_prog_33
  gun_2_selected := OFF

```

ADNAN OKYAR (2001-2009)

```

ENDIF --
IF ($DIN[72] = ON) OR (gun_2_selected = ON) AND BIT_TEST($WORD[26], 9, ON) THEN
  $BIT[34] := OFF
  call_prog_34
  gun_2_selected := OFF
ENDIF
sendsig(zone2_2_free_to_r1, ON)
gun_open(ON)
gun_open_check
IF BIT_TEST($WORD[28], gun_1_bypassed, ON) THEN
  GOTO l_1
ENDIF
open_cover_gun_2
MOVE JOINT TO pnt0055p
l_1::
  sendsig(zone5_5_free_to_r1, ON)
END rear_roof_swb

```

ROUTINE roof_swb

BEGIN

```

a_tool(1) -- select tool number
waitsig(zone2_2_free_from_r1, ON)
sendsig(zone2_1_free_to_r1, OFF)
$TERM_TYPE := FINE
$ARM_ACC_OVR := 80
$ARM_DEC_OVR := 80
MOVEFLY JOINT TO pnt0007p ADVANCE
MOVEFLY JOINT TO pnt0071p ADVANCE
MOVEFLY JOINT TO pnt0008p ADVANCE
MOVEFLY JOINT TO pnt0009p ADVANCE
MOVEFLY JOINT TO pnt0010p ADVANCE
MOVEFLY JOINT TO pnt0093p ADVANCE
-- waitsig(zone6_6_free_from_r1, ON)
-- sendsig(zone6_6_free_to_r1, OFF)
MOVE JOINT TO pnt0012p
pre_alarm_test
MOVE JOINT TO pnt0275p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0276p
MOVE JOINT TO pnt0277p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check
-- sendsig(zone6_6_free_to_r1, OFF)
MOVE JOINT TO pnt0278p
MOVE JOINT TO pnt0279p,
  WITH CONDITION[weld(22, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0278p
DELAY 500
MOVE JOINT TO wp6316_3_1p,
ENDMOVE

```

ADNAN OKYAR (2001-2009)

```

gun_semi_open(ON)
gun_semi_open_check
DELAY 500
MOVE JOINT TO wp6316_3_1p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
MOVE JOINT TO pnt0119p
MOVE JOINT TO wp6316_3_3p,
ENDMOVE
DELAY 500
MOVE JOINT TO wp6316_3_3p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
MOVE JOINT TO wp6316_3_4p,
ENDMOVE
DELAY 500
MOVE JOINT TO wp6316_3_4p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
MOVE JOINT TO pnt0214p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0025p
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO wp6314_3_2p,
ENDMOVE
DELAY 500
MOVE JOINT TO wp6314_3_2p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
MOVEFLY JOINT TO pnt0123p ADVANCE
MOVE LINEAR TO pnt0101p
MOVE LINEAR TO wp6217_1_3p,
ENDMOVE
-----
---R81'IN_PUNTALARINI_ATACAK---
-----
IF BIT_TEST($WORD[28], r_81_bypassli) = ON THEN
  MOVE JOINT TO pnt0257p
  r_1
ENDIF
DELAY 500
MOVE LINEAR TO wp6217_1_3p,
  WITH CONDITION[weld(1, FALSE)], -- Welding operation
ENDMOVE
MOVE LINEAR TO wp6217_1_4p,
ENDMOVE
DELAY 500
MOVE LINEAR TO wp6217_1_4p,
  WITH CONDITION[weld(1, FALSE)], -- Welding operation
ENDMOVE
$ARM_SPD_OVR := 30
$ARM_ACC_OVR := 30
$ARM_DEC_OVR := 30
MOVEFLY JOINT TO pnt0124p ADVANCE

```

ADNAN OKYAR (2001-2009)

```

MOVE JOINT TO pnt0125p
MOVE LINEAR TO pnt0184p
$ARM_SPD_OVR := 100
$ARM_ACC_OVR := 65
$ARM_DEC_OVR := 65
DELAY 500
MOVE LINEAR TO wp6329_1_1p,
  WITH CONDITION[weld(8, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check
MOVEFLY JOINT TO pnt0094p ADVANCE
MOVEFLY JOINT TO pnt0079p ADVANCE
MOVEFLY JOINT TO pnt0080p ADVANCE
MOVE JOINT TO pnt0115p
gun_semi_open(ON)
gun_semi_open_check
MOVE JOINT TO wp6217_1_6p,
ENDMOVE
-----
---R81'IN_PUNTALARINI_ATACAK---
-----
IF BIT_TEST($WORD[28], r_81_bypassli) = ON THEN
  MOVE JOINT TO pnt0258p
  r_1_a
ENDIF
DELAY 500
MOVE JOINT TO wp6217_1_6p,
  WITH CONDITION[weld(1, FALSE)], -- Welding operation
ENDMOVE
MOVE LINEAR TO pnt0109p
MOVE LINEAR TO wp6217_1_5p,
ENDMOVE
DELAY 500
MOVE LINEAR TO wp6217_1_5p,
  WITH CONDITION[weld(1, FALSE)], -- Welding operation
ENDMOVE
MOVEFLY LINEAR TO pnt0111p ADVANCE
MOVE LINEAR TO wp6329_1_2p,
ENDMOVE
DELAY 500
MOVE LINEAR TO wp6329_1_2p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
MOVE LINEAR TO pnt0120p
MOVE JOINT TO wp6314_3_3p,
ENDMOVE
DELAY 500
MOVE JOINT TO wp6314_3_3p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
DELAY 500
MOVE JOINT TO wp6314_3_6p,
ENDMOVE
DELAY 500
MOVE JOINT TO wp6314_3_6p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE

```

ADNAN OKYAR (2001-2009)

```

MOVE JOINT TO wp6314_3_8p,
ENDMOVE
DELAY 500
MOVE JOINT TO wp6314_3_8p,
  WITH CONDITION[weld(2, FALSE)],
ENDMOVE
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0033p
----
gun_open(ON)
gun_open_check
-- MOVE JOINT TO pnt0038p
----ROOF FLAP-----
IF BIT_TEST($WORD[28], 16, ON) THEN
  sendsig(zone2_1_free_to_r1, OFF)
  $ARM_SPD_OVR := 100
  $ARM_ACC_OVR := 85
  $ARM_DEC_OVR := 85
  MOVE LINEAR TO pnt0186p
  MOVE JOINT TO pnt0230p
  gun_semi_open(ON)
  gun_semi_open_check
  MOVE LINEAR TO pnt0188p,
    WITH CONDITION[weld(2, FALSE)], -- Welding operation
  ENDMOVE
  gun_open(ON)
  gun_open_check
  MOVE LINEAR TO pnt0186p
ENDIF
MOVE JOINT TO pnt0074p
--BECK UP FOR SWB--
MOVE JOINT TO pnt0075p
MOVE JOINT TO pnt0076p
MOVE LINEAR TO pnt0296p
sendsig(end_of_weld_1, ON)
-- sendsig(end_of_program, ON)
sendsig(zone2_1_free_to_r1, ON)
open_cover_gun_1
check_cover_gun_1_open
MOVEFLY JOINT TO pnt0077p ADVANCE
MOVEFLY JOINT TO pnt0116p ADVANCE
MOVE JOINT TO pnt0117p
IF ($DIN[70] = ON) OR (gun_1_selected = ON) AND BIT_TEST($WORD[26], 13, ON) THEN
  $BIT[tipdress_memory] := ON
ENDIF --
IF ($DIN[72] = ON) OR (gun_1_selected = ON) AND BIT_TEST($WORD[26], 9, ON) THEN
  $BIT[tipchange_memory] := ON
ENDIF
END roof_swb
ROUTINE call_prog_1
BEGIN
  a_tool(0) -- select tool number
  a_frame(0) -- select frame number
  -----
  ---> SWB MOD.
  -----
  sendsig(workarea_free_to_op, ON)

```

ADNAN OKYAR (2001-2009)

```

MOVE JOINT TO home_pos
$TIMER[10] := 0
sendsig(end_of_program, OFF)
sendsig(zone5_5_free_to_r1, ON)
sendsig(zone6_6_free_to_r1, ON)
sendsig(zone4_4_free_to_r1, ON)
sendsig(zone3_3_free_to_r1, ON)
sendsig(zone2_1_free_to_r1, ON)
sendsig(zone2_2_free_to_r1, ON)
sendsig(priority_job_to_r1, ON)
sendsig(home_position, OFF)
$ARM_SPD_OVR := 100
$ARM_ACC_OVR := 100
$ARM_DEC_OVR := 100
$TERM_TYPE := FINE
-----
IF BIT_TEST($WORD[28], gun_1_bypassed, ON) THEN
    waitsig(workarea_free_from_op, ON)
    sendsig(workarea_free_to_op, OFF)
    sendsig(end_of_weld_1, ON)
    GOTO gun2_1
ENDIF
-----
tool_change(1)
init_c_gun_vert_high
a_tool(1) -- select tool number
weld_init -- init. for welding
gun_open(ON)
gun_open_check
MOVE JOINT TO pnt0070p
waitsig(workarea_free_from_op, ON)
sendsig(workarea_free_to_op, OFF)
IF BIT_TEST($WORD[28], 12) THEN
    roof_swb_citali
ELSE
    roof_swb
ENDIF
-----
sendsig(zone4_4_free_to_r1, ON)
-----
IF BIT_TEST($WORD[28], gun_2_bypassed, ON) THEN
    sendsig(end_of_weld_2, ON)
    sendsig(end_of_program, ON)
    sendsig(workarea_free_to_op, ON)
    GOTO gun1_1
ENDIF
gun2_1::
-----
tool_change(2)
a_tool(2) -- select tool number
rear_roof_swb
-----
IF BIT_TEST($WORD[28], gun_1_bypassed, ON) THEN
    GOTO gun2_2
ENDIF
gun1_1::
-----
tool_change(1)

```

ADNAN OKYAR (2001-2009)

```

IF $BIT[tipdress_memory] = ON OR (gun_1_selected = ON) AND BIT_TEST($WORD[26], 13, ON) THEN
  $BIT[tipdress_memory] := OFF
  $BIT[33] := ON
  call_prog_33
  gun_1_selected := OFF
  $BIT[33] := OFF
ENDIF --
IF $BIT[tipchange_memory] = ON OR (gun_1_selected = ON) AND BIT_TEST($WORD[26], 9, ON) THEN
  $BIT[tipchange_memory] := OFF
  sendsig(end_of_weld_1, OFF)
  sendsig(end_of_weld_2, OFF)
  $BIT[34] := ON
  call_prog_34
  gun_1_selected := OFF
  $BIT[34] := OFF
ENDIF
-----
a_tool(1) -- select tool number
gun_open(ON)
MOVEFLY JOINT TO pnt0129p ADVANCE
MOVEFLY LINEAR TO pnt0128p ADVANCE
-----
gun2_2::
-----
a_frame(0) -- select frame number
a_tool(0) -- select tool number
MOVE JOINT TO home_pos
sendsig(home_position, ON)
sendsig(end_of_program, OFF)
sendsig(end_of_weld_1, OFF)
sendsig(end_of_weld_2, OFF)
sendsig(workarea_free_to_op, OFF)
END call_prog_1

BEGIN
-----
---> SWB MOD.
-----
call_prog_1
END prog_1

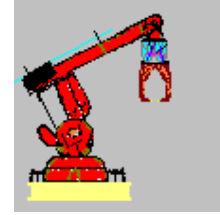
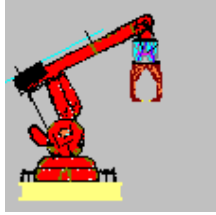
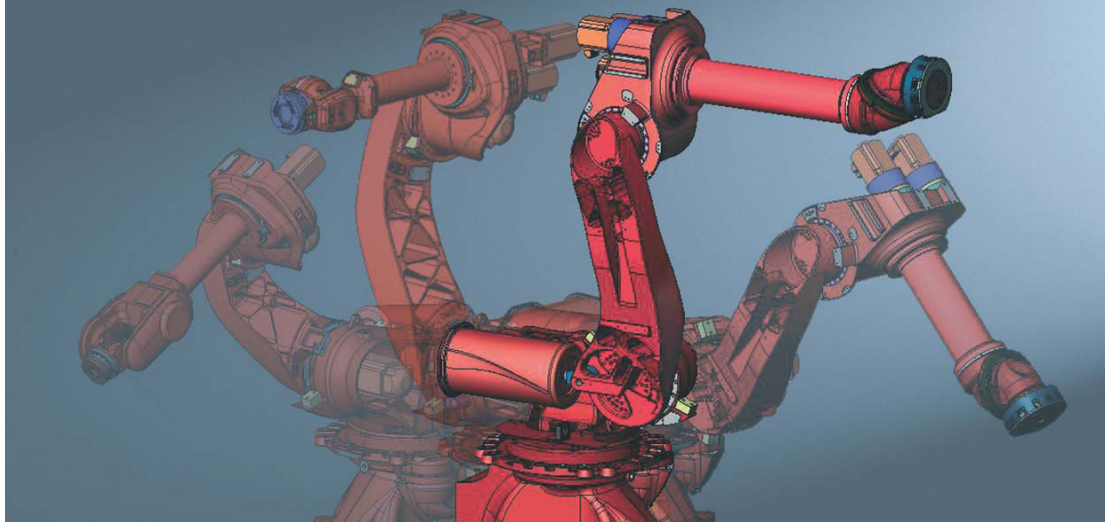
```

ADNAN OKYAR (2001-2009)

Power Automation & Robot Systems

adnan.okyar@powerotomasyon.net www.powerotomasyon.net





2001-2009

Hazırlayan: [ADNAN OKYAR](#)

Adnan OKYAR

POWER BİLİŞİM

BİLGİSAYAR-OTOMASYON-ELEKTRİK-MAKİNA

✉ Turangünes Cad. NO.284 Bekirpasa-iZMiT/KOCAELİ /TÜRKİYE

☎Tel :+90 0262 323 61 04 -323 12 85

☎Fax :+90 0262 323 12 86

GSM :0535 226 82 59

e-mail1:adnanokyar@powerbilisimtr.com

e-mail2: adnanokyar@yahoo.com

msn: adnanokyar@hotmail.com

web: www.powerbilisimtr.com

ADNAN OKYAR (2001-2009)

Power Automation&Robot Systems

adnan.okyar@powerotomasyon.net

www.powerotomasyon.net

