

iDea

KOMUTLAR

İÇİNDEKİLER

1	Genel Bakış	3
2	rsBASIC Komutları	4
2.1	Etiketler (Labels).....	4
2.2	Yorumlar (Comments)	5
2.3	Semboller (Symbols).....	6
2.4	backward (geri)	7
2.5	dec (azalt).....	8
2.6	do .. until	9
2.7	do .. while.....	10
2.8	for .. next.....	11
2.9	forward (ileri).....	12
2.10	gosub (alt program).....	13
2.11	halt (dur).....	14
2.12	high (yüksek)	15
2.13	if..then (eğer..ise)	16
2.14	inc (artır)	17
2.15	let (ifade tanımla)	18
2.16	low (düşük)	19
2.17	pause (bekle)	20
2.18	pulsout (atım gönder).....	21
2.19	readadc (analog oku).....	22
3	Akış Şeması ve Komutlar	23
3.1	Akış Şeması ile Görsel Programlama.....	23
3.2	Blok Ekleme.....	27
3.3	Blokları Birbirlerine Bağlamak.....	28
3.4	Semboller (Symbols).....	31

1 Genel Bakış

Bu dokümanda *iDea* yazılımı içerisinde yer alan rsBASIC ve Akış Şeması tabanlı yazılım geliştirme arayüzlerine ait komutlar ve kullanım şekilleri anlatılmaktadır.

rsBASIC derleyicisi Basic Stamp PBasic dili ile uyumlu olacak şekilde geliştirilmiş, bu şekilde benzer denetimci kullanıcılarının *iDea* uygulamalarına doğrudan veya ufak değişikliklerle geçebilmeleri hedeflenmiştir.

Dokümanda komutlar Türkçe ve İngilizce karşılıkları parantez içinde verilerek anlatılmıştır. Örneğin: inc (artır) (inc, increment'in kısaltılmasıdır).

iDea yazılımı www.robotsan.com.tr adresinden ücretsiz olarak indirilebilmektedir.

Bu doküman *iDea* yazılımı revizyon 1.0 kullanılarak hazırlanmıştır. Lütfen, web adresinden güncellemeleri kontrol ediniz.

2 rsBASIC Komutları

2.1 Etiketler (Labels)

Etiketler program içerisinde belirli bölümleri işaret etmek ve *goto*, *gosub* gibi komutların bu işaretli noktalara gönderme yapmalarında kullanılırlar.

Etiketler küçük ve büyük harf barındırabilir, ancak Türkçe karakter içeremezler.

Örnek 1

dongu:	‘ Döngü başlangıç etiketi
high 1	‘ 1 numaralı çıkışı aç
pause 1000	‘ 1 sn bekle
low 1	‘ 1 numaralı çıkışı kapat
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

Örnek 2

AnaDongu:	‘ Ana döngü başlangıç etiketi
gosub MotorAc	‘ <i>MotorAc</i> etiketine git
pause 1000	‘ 1 sn bekle
gosub MotorKapat	‘ <i>MotorKapat</i> etiketine git
pause 2000	‘ 2 sn bekle
goto AnaDongu	‘ Başlangıca (AnaDongu etiketine) geri dön

MotorAc:	‘ Motor aç başlangıç etiketi
high 1	‘ 1 numaralı çıkışı aç
return	‘ Bu etiketi/döngüyü sonlandırıp 1 sn bekle komutuna döner

MotorKapat:	‘ Motor kapat başlangıç etiketi
low 1	‘ 1 numaralı çıkışı kapat
return	‘ Bu etiketi/döngüyü sonlandırıp 2 sn bekle komutuna döner

2.2 Yorumlar (Comments)

Yorumlar program içerisinde kullanıcı tarafından bilgilendirici açıklama yazabilmek amacıyla kullanılır ve derleyici tarafından dikkate alınmazlar.

rsBASIC derleyicisinde yorumlar kesme işareti (') ile başlar ve bulunduğu satırın sonuna kadar işlevliğini korur.

Yorumlar içerisinde Türkçe karakter kullanılabilir.

Örnek

dongu:	' Döngü başlangıç etiketi
high 1	' 1 numaralı çıkışı aç
pause 1000	' 1 sn bekle
low 1	' 1 numaralı çıkışı kapat
goto dongu	' Başlangıca (dongu etiketine) geri dön

2.3 Semboller (Symbols)

Semboller *iDea* yazılımında sabit değer, *iBox* sayısal giriş atamalarına yönlendirmede (bacak ataması/*pin0..7*) ve değişken atamalarında kullanılırlar.

rsBASIC derleyicisinde sembol tanımlaması için *symbol* yazılır, isim verildikten sonrasında eşit işareti (=) ile atama yapılır.

Semboller program başlangıcında tanımlanır, tanımlama sırasında rakamlar ve alt çizgi kullanılabilir ancak Türkçe karakter içeremezler. Sembol kullanımı program hafıza kullanımını artırmaz.

iDea tarafından aşağıda listelenen tanımlar desteklenir:

bayt: b0..b15, **kelime:** w0..w15, **noktalı sayı:** f0..f15, **bacak:** pin0...pin7

Örnek 1

symbol MotorA = 2	‘ MotorA değişkeni 2 değeri ile sabit sayı olarak tanımlandı
symbol j = b0	‘ j değişkeni bayt (byte) olarak tanımlandı
symbol Sensor1 = w0	‘ Sensor1 değişkeni kelime (word) olarak tanımlandı
symbol Sensor2 = w1	‘ Sensor2 değişkeni kelime (word) olarak tanımlandı
symbol Sicaklik = f0	‘ Sicaklik değişkeni noktalı sayı (floating) olarak tanımlandı
symbol SolSensor = pin3	‘ SolSensor değişkeni <i>iBox P12</i> girişi olarak tanımlandı

Örnek 2

symbol Anahtar = pin2	‘ Anahtar değişkeni <i>iBox P11</i> girişi olarak tanımlandı
dongu:	‘ Döngü başlangıç etiketi
if Anahtar = 0 then	‘ if..then..else döngüsü (eğer Anahtar “0” ise)
high 1	‘ 1 numaralı çıkışı aç
else	‘ diğer durumda (eğer Anahtar “1” ise)
low 1	‘ 1 numaralı çıkışı kapat
endif	‘ end..if tanımlaması sona erdi
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

2.4 backward (geri)

a ve b olarak tanımlanmış *iBox* motor çıkışlarını geri çevirmek için kullanılır.

Kullanım Şekli

rsBASIC
backward motor

Açıklama

- “motor” a veya b olabilir.
- a motoru *iBox* P1 ve P2 çıkışlarına bağlanmalıdır.
- b motoru *iBox* P3 ve P4 çıkışlarına bağlanmalıdır.
- “backward a” komutu low P1, high P2 ile eşdeğerdir.
- “backward b” komutu low P3, high P4 ile eşdeğerdir.

Örnek

dongu:	‘ Döngü başlangıç etiketi
backward a	‘ a motorunu geri çevir, P1’i kapat, P2’yi aç
backward b	‘ b motorunu geri çevir, P3’ü kapat, P4’ü aç
pause 1000	‘ 1 sn bekle
forward a	‘ a motorunu ileri çevir, P1’i aç, P2’yi kapat
forward b	‘ b motorunu ileri çevir, P3’ü aç, P4’ü kapat
pause 1000	‘ 1 sn bekle
halt a	‘ a motorunu durdur, P1’i ve P2’yi kapat
halt b	‘ b motorunu durdur, P3’ü ve P4’ü kapat
pause 1000	‘ 1 sn bekle
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

2.5 dec (azalt)

Sembol ile tanımlanmış bir değeri azaltmak için kullanılır.

Kullanım Şekli

rsBASIC
dec sembol

Örnek

symbol i = b0	‘ i değişkeni bayt olarak tanımlandı
i = 20	‘ i değişkenine 20 sabit değeri başlangıçta atandı
dongu:	‘ Döngü başlangıç etiketi
dec i	‘ i değişkenini bir azalt
if i = 10 then	‘ if..then..else döngüsü (eğer i = 10 ise)
high 1	‘ 1 numaralı çıkışı aç
else	‘ diğer durumda (eğer i 10’dan farklı ise)
low 1	‘ 1 numaralı çıkışı kapat
endif	‘ end..if tanımlaması sona erdi
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

2.6 do .. until

Bir kod parçasının tekrar etmesinin istenildiği durumlarda kullanılır. (do until..loop) veya (do .. loop until) döngüsü içerisindeki kod parçası ifade geçerli olana kadar çalıştırılır.

Kullanım Şekli

rsBASIC
do until ifade
loop
do ifade
loop until

Açıklama

- “ifade” mantıksal bir karşılaştırmadır.
- “do..loop until”, “for..next” ve “do until”den farklı olarak ifadenin doğru olup olmadığını döngünün sonunda karşılaştırdığı için en az bir kere döngü içerisindeki kodu çalıştırır.

Örnek

```
symbol i = w0          ‘ i değişkenini kelime olarak tanımladık
i = 5                  ‘ i değişkenine 5 sabit değerini atadık
do until i < 2         ‘ i < 2 olana kadar loop’a kadarki kodu çalıştır
    high 1             ‘ 1 numaralı çıkışı aç
    dec i              ‘ i’nin değerini bir azalt
loop
do                    ‘ i > 2 olana kadar loop’a kadarki kodu çalıştır
    low 1              ‘ 1 numaralı çıkışı kapat
    inc i              ‘ i’nin değerini bir artır
loop until i > 2
```

2.7 do .. while

Bir kod parçasının tekrar etmesinin istenildiği durumlarda kullanılır. (do while..loop) veya (do .. loop while) döngüsü içerisindeki kod parçası ifade geçersiz olana kadar çalıştırılır.

Kullanım Şekli

rsBASIC
do while ifade
loop
do ifade
loop while

Açıklama

- “ifade” mantıksal bir karşılaştırmadır.
- “do..loop while”, “for..next” ve “do while” dan farklı olarak ifadenin doğru olup olmadığını döngünün sonunda karşılaştırdığı için en az bir kere döngü içerisindeki kodu çalıştırır.

Örnek

```
symbol i = w0          ‘ i değişkenini kelime olarak tanımladık
i = 1                  ‘ i değişkenine 1 sabit değerini atadık
do while i < 2         ‘ i < 2 olduğu sürece loop’a kadarki kodu çalıştır
    high 1             ‘ 1 numaralı çıkışı aç
    inc i              ‘ i’nin değerini bir artır
loop
do                    ‘ i > 2 olduğu sürece loop’a kadarki kodu çalıştır
    low 1              ‘ 1 numaralı çıkışı kapat
    dec i              ‘ i’nin değerini bir azalt
loop while i > 2
```

2.8 for .. next

Bir kod parçasının tekrar etmesinin istenildiği durumlarda kullanılır. for..next döngüsü içerisindeki kod parçası başlangıç-son değerleri arasında tekrarlayarak çalıştırılır.

Kullanım Şekli

rsBASIC
for değişken = başlangıç to son
next

Açıklama

- “başlangıç” değişkenin ilk değeri, “son” son değeridir.
- “değişken” istenilen son değere ulaşana kadar for..next arasında yer alan kod çalıştırılır. Her döngü çalıştırılışında “next” bölümüne gelindiğinde değişken bir artırılır.
- “değişken” bayt veya kelime olabilir.

Örnek

symbol i = b0	‘ i değişkenini bayt olarak tanımladık
dongu:	‘ Döngü başlangıç etiketi
for i = 1 to 10	‘ 10 kere 1 numaralı çıkışı açıp/kapatacağız
high 1	‘ 1 numaralı çıkışı aç
pause 500	‘ 500 ms bekle
low 1	‘ 1 numaralı çıkışı kapat
pause 500	‘500 ms bekle
next	‘ i’yi bir artır, gerekli koşul sağlanıyorsa for döngüsüne git
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

2.9 forward (ileri)

a ve b olarak tanımlanmış **iBox** motor çıkışlarını ileri çevirmek için kullanılır.

Kullanım Şekli

rsBASIC
forward motor

Açıklama

- “motor” a veya b olabilir.
- a motoru **iBox** P1 ve P2 çıkışlarına bağlanmalıdır.
- b motoru **iBox** P3 ve P4 çıkışlarına bağlanmalıdır.
- “forward a” komutu high P1, low P2 ile eşdeğerdir.
- “forward b” komutu high P3, low P4 ile eşdeğerdir.

Örnek

dongu:	‘ Döngü başlangıç etiketi
backward a	‘ a motorunu geri çevir, P1’i kapat, P2’yi aç
backward b	‘ b motorunu geri çevir, P3’ü kapat, P4’ü aç
pause 1000	‘ 1 sn bekle
forward a	‘ a motorunu ileri çevir, P1’i aç, P2’yi kapat
forward b	‘ b motorunu ileri çevir, P3’ü aç, P4’ü kapat
pause 1000	‘ 1 sn bekle
halt a	‘ a motorunu durdur, P1’i ve P2’yi kapat
halt b	‘ b motorunu durdur, P3’ü ve P4’ü kapat
pause 1000	‘ 1 sn bekle
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

2.11 halt (dur)

a ve b olarak tanımlanmış **iBox** motor çıkışlarını durdurmak için kullanılır.

Kullanım Şekli

rsBASIC
halt motor

Açıklama

- “motor” a veya b olabilir.
- a motoru **iBox** P1 ve P2 çıkışlarına bağlanmalıdır.
- b motoru **iBox** P3 ve P4 çıkışlarına bağlanmalıdır.
- “halt a” komutu low P1, low P2 ile eşdeğerdir.
- “halt b” komutu low P3, low P4 ile eşdeğerdir.

Örnek

dongu:	‘ Döngü başlangıç etiketi
backward a	‘ a motorunu geri çevir, P1’i kapat, P2’yi aç
backward b	‘ b motorunu geri çevir, P3’ü kapat, P4’ü aç
pause 1000	‘ 1 sn bekle
forward a	‘ a motorunu ileri çevir, P1’i aç, P2’yi kapat
forward b	‘ b motorunu ileri çevir, P3’ü aç, P4’ü kapat
pause 1000	‘ 1 sn bekle
halt a	‘ a motorunu durdur, P1’i ve P2’yi kapat
halt b	‘ b motorunu durdur, P3’ü ve P4’ü kapat
pause 1000	‘ 1 sn bekle
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

2.12 high (yüksek)

Bir sayısal çıkışı yüksek duruma getirmek için kullanılır.

Kullanım Şekli

rsBASIC
high çıkış

Açıklama

- “çıkış” 1..4 veya 9..19 arasında bir değer olabilir. Bu değerler *iBox*'ta P1..P4 ve P9..P16 arasındaki bağlantılara karşılık gelmekte olup sayısal giriş/çıkış için ayrılmıştır.

Örnek

```
dongu:          ' Döngü başlangıç etiketi
                ' P19'u aç
high 19         ' P19'u aç
pause 1000      ' 1 sn bekle
low 19          ' P19'u kapat
pause 1000      ' 1 sn bekle
goto dongu      ' Başlangıca (dongu etiketine) geri dön
```

2.13 if..then (eğer..ise)

Bir kod parçasının belirli koşullarda çalışmasının istendiği durumlarda kullanılır. if (eğer) tanımından sonraki koşul doğru ise devamındaki bölüm değil ise sonrasındaki kod parçası çalıştırılır.

Kullanım Şekli

rsBASIC
if...then \ elseif...then \ else \ endif
if...then goto (etiket)
if...and/or..then goto (etiket)
if...then gosub
if...and/or...then gosub

Açıklama

- **if .. then** ifadesi arasına sağlanması gereken koşul yazılır. Örneğin, a>b, a=pin1, a=120 and b<100.

Örnek

```
symbol s1 = pin2          ' s1 değişkeni iBox P11 girişi olarak tanımlandı
symbol s2 = pin3          ' s2 değişkeni iBox P12 girişi olarak tanımlandı

if s1=0 and s2=0 then     ' eğer s1 ve s2 "0" ise
    high 1                ' 1 numaralı çıkışı aç
elseif s1=1 and s2=0 then ' eğer s1 ve s2 "1" ise
    low 1                  ' 1 numaralı çıkışı kapat
else                       ' yukarıdaki iki koşulda sağlanmamışsa
    high 1                 ' 1 numaralı çıkışı aç
    pause 1000             ' 1 sn bekle
    low 1                  ' 1 numaralı çıkışı kapat
endif                     ' end..if tanımlaması sona erdi
```

2.14 inc (artır)

Sembol ile tanımlanmış bir değeri artırmak için kullanılır.

Kullanım Şekli

rsBASIC
inc sembol

Örnek

symbol i = b0	‘ i değişkeni bayt olarak tanımlandı
i = 2	‘ i değişkenine 2 sabit değeri başlangıçta atandı
dongu:	‘ Döngü başlangıç etiketi
inc i	‘ i değişkenini bir artır
if i = 10 then	‘ if..then..else döngüsü (eğer i = 10 ise)
high 1	‘ 1 numaralı çıkışı aç
else	‘ diğer durumda (eğer i 10’dan farklı ise)
low 1	‘ 1 numaralı çıkışı kapat
endif	‘ end..if tanımlaması sona erdi
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

2.15 let (ifade tanımla)

İfade tanımlamak ve değişkenler (semboller) üzerinde işlem yapmak için kullanılır.

Kullanım Şekli

rsBASIC
let değişken = sembol
let a = b + 5

Açıklama

– Değişkenler üzerinde aşağıda belirtilen matematiksel işlemler yapılabilir:

- + : toplama
- - : çıkartma
- * : çarpma
- / : bölme
- & : ve
- | : veya
- ^ : xor
- %/ : mod bölümü (kalanı verir)
- |/ : nor
- ^/ : xnor
- inc/dec : artır/azalt

Örnek

symbol i = b0 ‘ i değişkeni bayt olarak tanımlandı
symbol j = b1 ‘ j değişkeni bayt olarak tanımlandı
symbol k = f0 ‘ k değişkeni noktalı sayı olarak tanımlandı

let i = pin1 ‘ i değişkeni pin 1 (iBox P10) değerine eşitlendi

let j = i * 2 ‘ j değişkeni için değer atandı

let k = j / 3 ‘ k değişkenine değer atandı

2.17 pause (bekle)

Belirli bir süre bekleme yapmak için kullanılır.

Kullanım Şekli

rsBASIC
pause süre

Açıklama

- “süre”, milisaniye olarak bekleme yapılacak değerdir. 0 – 65535 değerleri arasında olabilir.

Örnek

dongu:	‘ Döngü başlangıç etiketi
backward a	‘ a motorunu geri çevir, P1’i kapat, P2’yi aç
pause 1000	‘ 1 sn bekle
forward a	‘ a motorunu ileri çevir, P1’i aç, P2’yi kapat
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

2.18 pulsout (atım gönder)

Sayısal bir çıkıştan belirli uzunluktaki (süredeki) bir atım göndermek için kullanılır.

Kullanım Şekli

rsBASIC
pulsout çıkış, süre

Açıklama

- “çıkış” 9..16 arasında bir değer olabilir. Bu değerler *iBox*'ta P9..P16 arasındaki bağlantılara karşılık gelmekte olup sayısal giriş/çıkış için ayrılmıştır.
- “süre”, belirtilen süre için çıkış yapılacak bağlantıdaki atımın uzunluğunu belirleyen değerdir. 0 – 65535 değerleri arasında olabilir.
- “pulsout” komutu genelde RC Servomotorların denetimi için kullanılır.

Örnek

symbol i = b0	‘ i değişkenini bayt olarak tanımladık
dongu:	‘ Döngü başlangıç etiketi
for i = 1 to 100	‘ Belirli sayıda (100) atım üretmek istiyoruz
pulsout 10, 500	‘ P10 çıkışında 0,5ms’lik bir atım gönder
pause 20	‘ 20 ms bekle
next	‘ i’yi bir artır, gerekli koşul sağlanıyorsa for döngüsüne git
goto dongu	‘ Başlangıca (dongu etiketine) geri dön

2.19 readadc (analog oku)

Analog giriş kanallarından 10 bit çözünürlükte analog sinyal okuma yapmak için kullanılır.

Kullanım Şekli

rsBASIC
readadc giriş, değişken

Açıklama

- “giriş”, “0” ile “3” arasındaki bir değer olabilir. Bu değerler *iBox*'ta P5-P8 arasındaki bağlantılara karşılık gelmekte olup analog giriş okumak için ayrılmıştır.
- “değişken” analog sayısal dönüştürme işlemi sonucundaki değer saklandığı kelime sembolüdür. 0-1023 (10bit/0-5V aralığında) değerleri arasında olabilir.

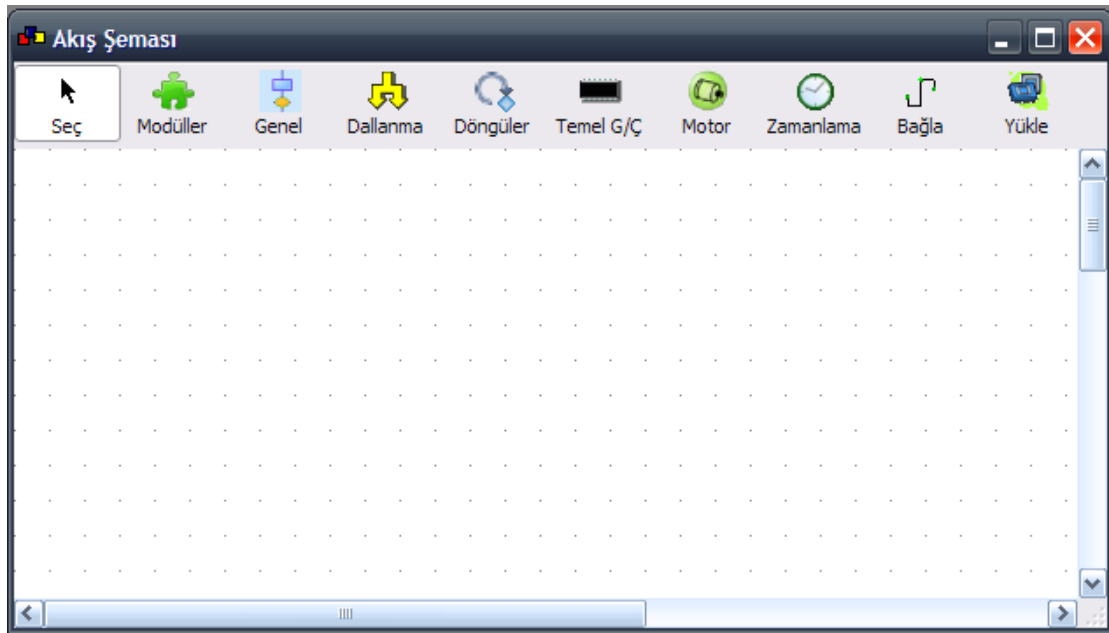
Örnek

```
symbol mesafe = w0          ‘ mesafe değerini saklayacağımız değişken
dongu:                      ‘ Döngü başlangıç etiketi
    readadc 0, mesafe       ‘ P5 analog girişe ait mesafe değerini okuduk
    if mesafe > 512 then    ‘ mesafe 512’den büyükse (2,5V’tan büyükse)
        forward a          ‘ a motorunu ileri çevir, P1’i aç, P2’yi kapat
    else                    ‘ diğer durumda (mesafe 512’den küçükse)
        backward a         ‘ a motorunu geri çevir, P1’i kapat, P2’yi aç
    endif
    goto dongu              ‘ Başlangıca (dongu etiketine) geri dön
```

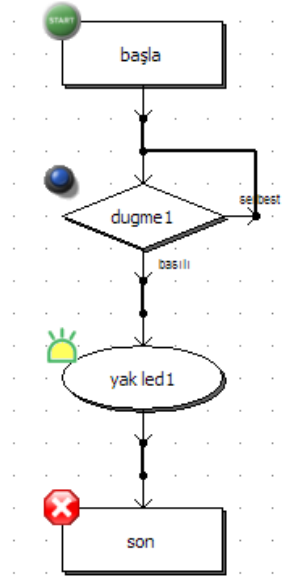
3 Akış Şeması ve Komutlar

3.1 Akış Şeması ile Görsel Programlama

Daha önce akış şemaları (flowchart) ile karşılaşmış, hatta kullanmış olabilirsiniz. Basitçe anlatmamız gerekirse, akış şeması farklı türlerde olabilen belirli sayıda bloktan oluşur. Bu bloklar, bloklar arasındaki akışı gösteren, diğer bir deyişle bir bloktan, sonraki adımda hangi bloğa geçileceğini belirten, yönlü çizgiler ile birbirlerine bağlıdır. RS IDE’de bloklar arasındaki akış kontrol uygulaması için kullanılmaktadır. Örnek bir akış şeması Şekil 2’te verilmiştir.



Şekil 1 Akış şeması penceresi.



Şekil 2 Örnek bir akış şeması.

Akış şeması penceresi iki bölüme ayrılmıştır. Pencerenin üst tarafında araç paleti yer alır. Pencerenin geri kalan kısmı ise blokların yerleştirileceği çalışma alanını oluşturur. Akış şeması penceresi aktif halde iken programın menü kısmına “Düzenle” ve “Görüntüle” başlıkları eklenecektir (

Şekil 3). “Düzenle” menüsündeki “Çalışma Alanı...” seçeneği ile çalışma alanının boyutu ve arkaplan rengi değiştirilebilmektedir (

Şekil 4). “Semboller” seçeneği seçildiğinde akış şemasında kullanılacak sabitler ve değişkenlerin tanımlanmasına olanak sağlayan diyalog kutusu görüntülenir (

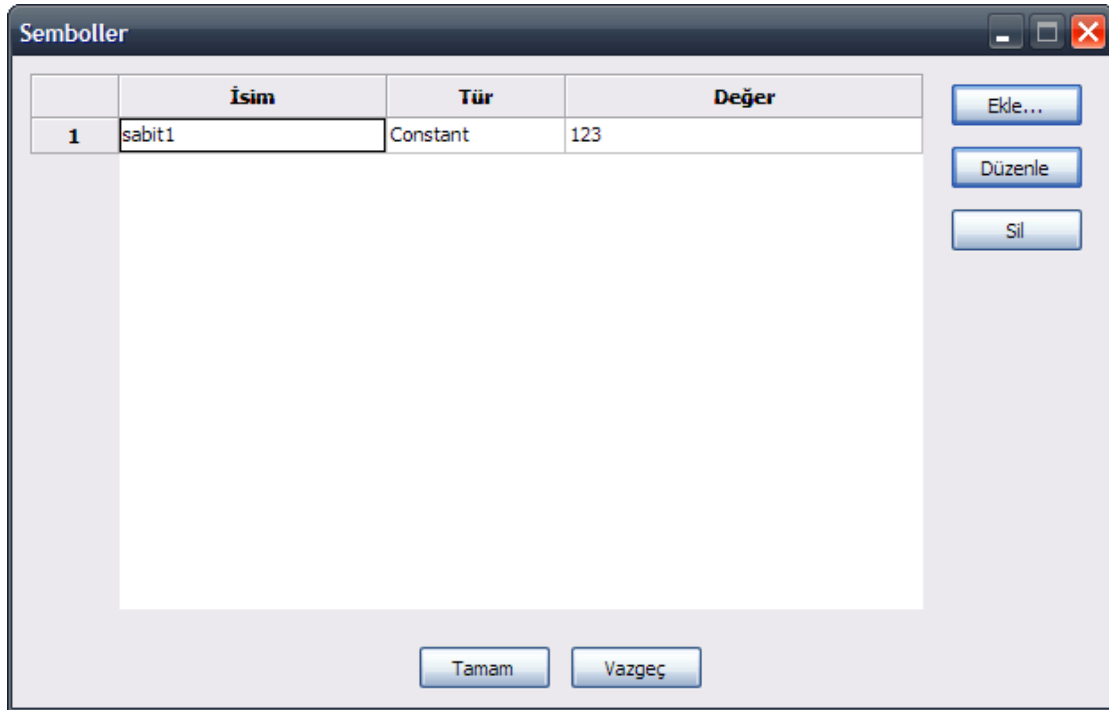
Şekil 5).



Şekil 3 Düzenle başlığı.



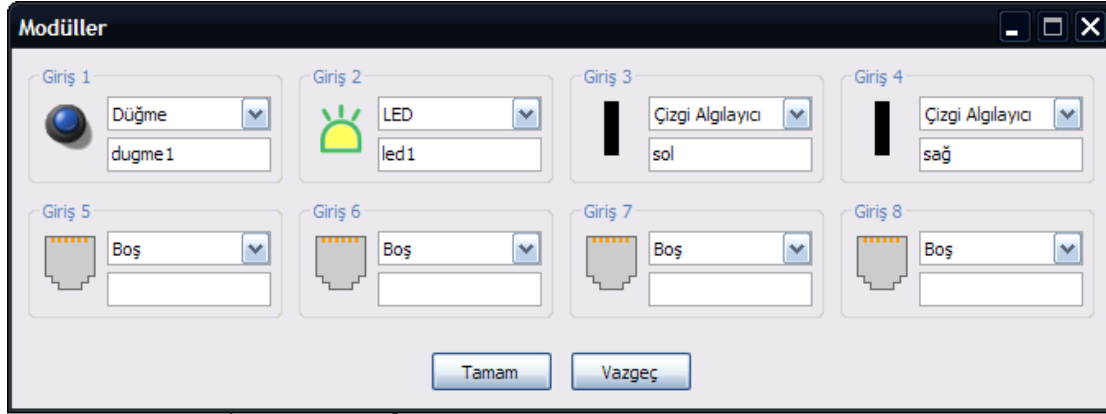
Şekil 4 Çalışma alanı özellikleri.



Şekil 5 Sembol düzenleme ekranı.

“Düzenle” menüsündeki “Modüller” seçeneği seçildiğinde, robot platform üzerinde bulunan kontrol kutusuna bağlanacak ek modüllerin (çizgi algılayıcı modülü gibi) tanımlanmasına olanak tanıyan diyalog kutusu görüntülenir (

Şekil 6).



Şekil 6 Modül düzenleme ekranı.

“Görüntüle” menüsündeki “Izgara” seçeneği ile çalışma alanındaki ızgara (eşit aralıklarla yerleştirilmiş olan noktalar) gösterilir veya gizlenir. “İleri Seviye Bloklar” seçeneği ise genel kullanım haricinde daha ileri seviye uygulamalarda ihtiyaç duyulabilecek blokların kullanılabilmesine olanak sağlamak veya kullanıcıdan gizlenmektedir.



Şekil 7 Araç paleti.

Akış şemasını tanımlamak için araç paletindeki araçlar kullanılmaktadır (

Şekil 7). “Seç” aracı blok ve onları birbirlerine bağlayan çizgileri seçmek, “Bağla” aracı blokları birbirlerine bağlayan çizgileri çizmek, “Yükle” aracı ise akış şemasını RSPic’e yüklemek için kullanılır. Diğer araçlar (“Modüller” haricinde) akış şemasına farklı türlerden bloklar yerleştirmek için kullanılmaktadır. Modüller aracına basıldığında, modüllere yönelik blokları içeren yeni bir araç paleti görüntülenir (Şekil 8). Bu paletten ana palete geri dönmek için “Geri Dön” aracı seçilmelidir.



Şekil 8 Modüller paleti.

3.2 Blok Ekleme

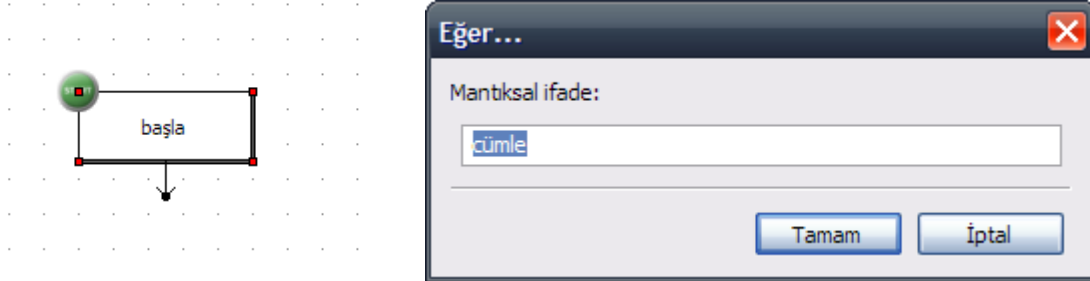
Akış şemasına yeni bir blok eklemek için, öncelikle araç paletinden ilgili blok kategorisine ait araç seçilmelidir (örneğin, “Genel”). “Dallanma” aracı haricindeki tüm araçlarda, çalışma alanının sol tarafında blokların kendilerini içeren dikey bir palet görüntülenecektir (Şekil 9).



Şekil 9 Blok paleti (dikey).

Blok paletinden herhangi bir blok seçildikten sonra çalışma alanında herhangi bir noktaya farenin sol tuşu ile tıkladığında, o noktaya seçilen türde yeni bir blok yerleştirilecek ve eğer bloğun parametreleri varsa parametreler ile ilgili diyalog kutusu görüntülenecektir. Örnek olarak verilmiş olan “Başla” bloğu ve “Dallanma” bloğu eklendiğinde görüntülenen ilgili parametre diyalog kutusu Şekil 10’da gösterilmiştir. Eklenmiş olan blok otomatik olarak seçili hale gelecek ve araç paletinden de “Seç” aracı aktif hale geçecektir. Seçili bloklar dört köşesinde yer alan kırmızı küçük kutular ile işaretlenmiştir. “Seç” aracı aktif iken çalışma alanı üzerinde boş bir alana farenin sol tuşu ile tıkladığında bloklar üzerindeki seçim kaldırılır; eğer herhangi bir bloğun üzerine tıklanırsa o blok seçili hale gelir. Bir blok seçili iken üzerinde farenin sol tuşuna basıp tuş basılı iken fareyi sürükleyerek bloğun konumunu değiştirebilirsiniz. Eğer bloğun parametreleri varsa, farenin sol tuşu ile bloğun üzerinde çift tıklama yapıldığında ilgili diyalog kutusu görüntülenir. Bloğun üzerinde sağ tuşa basıldığında çıkan menüden “Özellikler” seçeneği seçilerek bloğun şekli ve rengi değiştirilebilmektedir (Şekil 14). Yerleştirilmiş olan bir bloğu silmek için blok seçili iken klavyeden “Delete/Sil”

tuşuna basılması veya “Düzenle” menüsünden “Sil” seçeneğinin seçilmesi ve çıkan soruya olumlu yanıt verilmesi yeterlidir.



Şekil 10 Parametre diyalog kutusu.



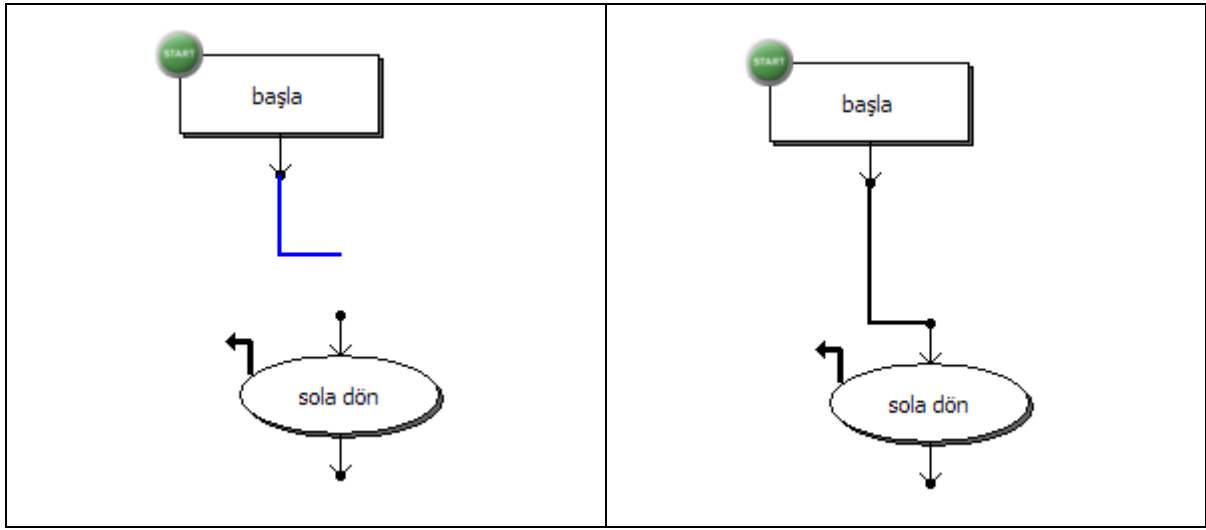
Şekil 11 Blok özellikleri

Her akış şemasında mutlaka bir adet “Başla” bloğu bulunmalıdır. Bu bloğu yerleştirmek için araç paletinden “Genel” kategorisini seçmelisiniz.

3.3 Blokları Birbirlerine Bağlamak

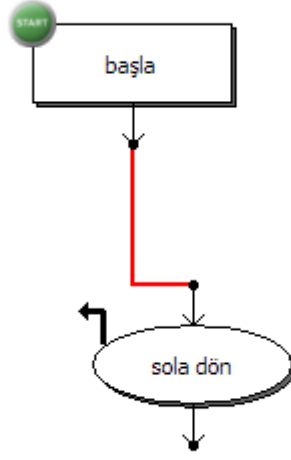
Çalışma alanına yerleştirilmiş olan blokları birbirlerine bağlamak için “Bağla” aracı kullanılmaktadır. Bloklar birbirlerine sadece bağlantı bacaklarından bağlanabilirler. Bu bağlantı bacakları bloğun çevresinde ufak siyah daireler ile gösterilmiştir (Şekil 12a). Blokların girdi ve çıktı olmak üzere iki farklı türde bağlantı bacakları olabilir. Eğer bağlantı bacağı bloğa blok yönünde bir ok ile bağlanıyorsa girdi bacağı, blok yönünden çıkan bir ok ile bağlanıyorsa çıktı bacağıdır. Sadece girdi ve çıktı bacaklarını birbirlerine bağlayabilirsiniz; iki girdi bacağı veya iki çıktı bacağı kendi aralarında birbirlerine bağlamak mümkün değildir ve denendiğinde kullanıcı bir hata mesajı ile uyarılacaktır. “Bağla” aracı aktif iken, bir

bağlantı bacağının dairesine farenin sol tuşu ile tıklandığında, çizgi çizme moduna geçilir. Bu modda iken fare hareket ettirildikçe seçilmiş olan bağlantı bacağında farenin o an bulunduğu noktaya mavi renkli geçici bir çizgi çizilir. Bağlanmak istenen diğer bloğun ilgili bacağının dairesine tıklandığında iki blok birbirine bağlanmış olur ve aralarında siyah renkli kalıcı bir çizgi çizilir (Şekil 12b). Araç paletinden başka bir araç seçilene kadar “Bağla” aracı aktif olmaya devam eder, dolayısıyla yukarıdaki işlemleri tekrar ederek yeni bağlantılar oluşturabilirsiniz.



Şekil 12 Çizgi modu.

Mevcut bağlantıları silmek için önce araç paletinden “Seç” aracını aktif hale getirmek gerekmektedir. “Seç” aracı aktifken, bir bağlantı bacağının dairesine farenin sol tuşu ile tıklandığında o bacağa bağlı olan çizgi seçili hale gelir. Seçili çizgi kırmızı renk ile gösterilmektedir (Şekil 13). Çizgi seçili iken klavyeden “Delete/Sil” tuşuna basarak veya “Düzenle” menüsünden “Sil” seçeneğini seçerek çizgiyi silebilirsiniz. Bağlantı çizgileri bloklar hareket ettirildiğinde otomatik olarak hareket eder ve bir blok silindiğinde o bloğa bağlı olan çizgiler de blok ile birlikte silinir.



Şekil 13 Seçili bağlantı çizgisi.

3.4 Semboller (Symbols)

Semboller *iDea* yazılımında sabit değer, *iBox* sayısal giriş atamalarına yönlendirme (bacak ataması/*pin0..7*) ve değişken atamalarında kullanılırlar.

Semboller program başlangıcında tanımlanır, tanımlama sırasında rakamlar ve alt çizgi kullanılabilir, ancak Türkçe karakter içeremezler.

Sembol kullanımı program hafıza kullanımını artırmaz.

iDea Akış Şemasını kullanarak sembol tanımlamak için araç paletinde “Düzenle” menüsü altında “Semboller” seçimi yapılır ve gelen diyalog kutusunda istenilen atamalar yapılır. Atamalar *iDea* rsBASIC yazılımı ile uyumludur.

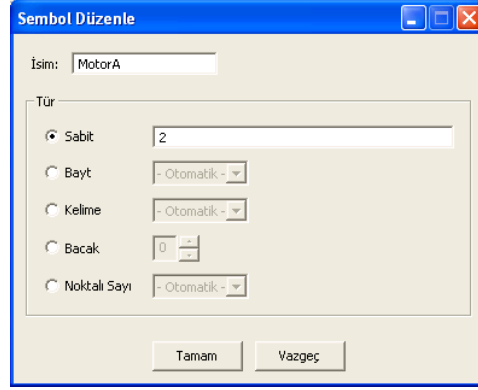
Örnek 1

iDea rsBASIC örneği

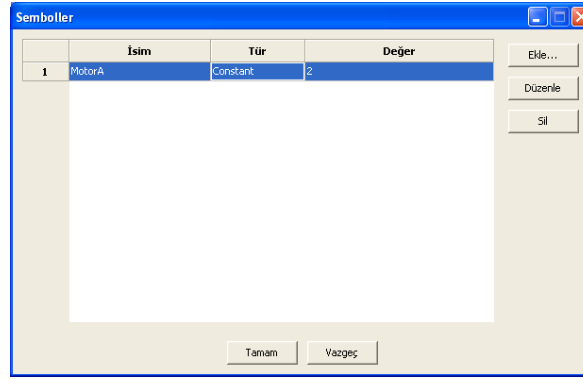
symbol MotorA = 2 ‘ MotorA değişkeni 2 değeri ile sabit sayı olarak tanımlandı
high MotorA ‘ MotorA çıkışı (2 numaralı *iBox* çıkışı) aç

Yukarıda verilen örneğin Akış Şemasında uygulamak için aşağıdaki adımlar yürütülür.

1. Yeni bir Akış Şeması yaratılır.
2. Araç paletinde *Düzenle* Menüsü altında *Semboller* seçilir.
3. Gelen diyalog kutusunda sağ tarafta yer alan *Ekle* düğmesine basılarak *Sembol Düzenle* diyalogu açılır ve Şekil 1’deki gibi düzenlenir. *Tamam*’a basılarak çıkıldıktan sonra Şekil 2’deki gibi ekran görülür ve bu ekranda *Tamam* seçilerek sembol tanımlama işlemi tamamlanır.

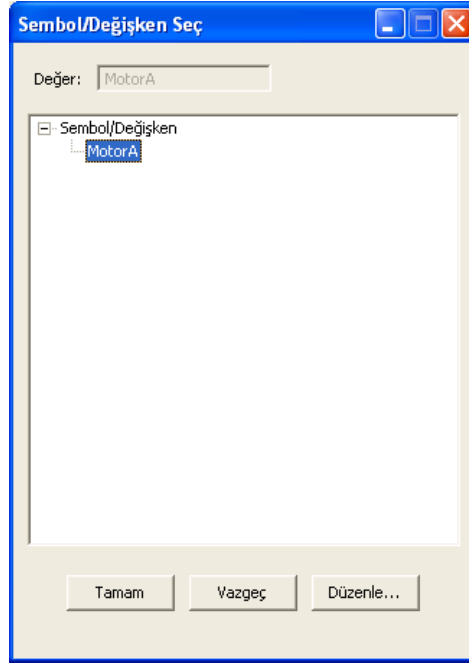


Şekil 1 Sembol Düzenle diyalogu.



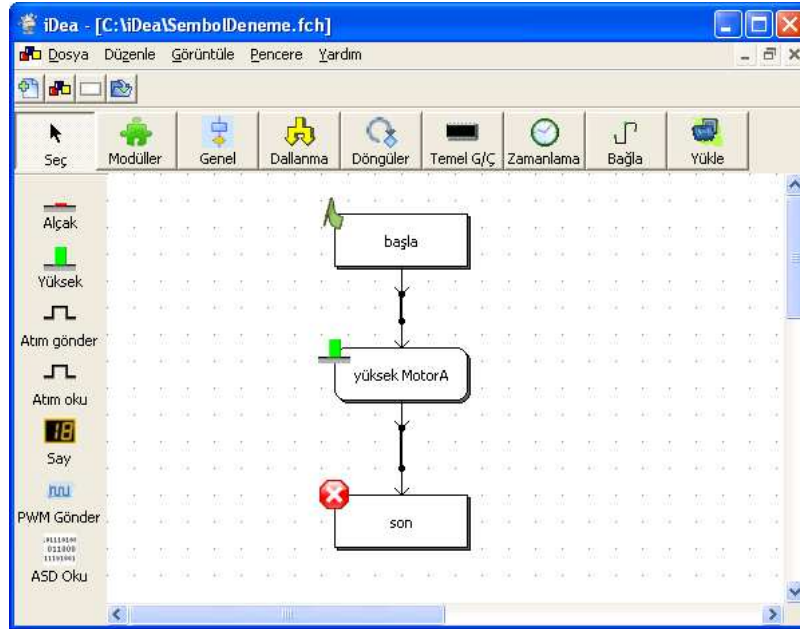
Şekil 2 Semboller diyalogu.

4. *MotorA* değerini yüksek duruma getirebilmek için araç paletinden *Temel G/Ç*, dikey Blok paletinden *Yüksek x* bloğu seçilir. Yüksek x bloğunda sembol olarak *MotorA* değeri Bu işlemden sonra Şekil 3'te gösterilen Sembol/Değişken seç ekranında olduğu gibi *MotorA* değişkeni seçilir.



Şekil 3 Sembol/Değişken Seç diyalogu.

5. BASIC örneğini tamamlayabilmek için Araç paletinden Genel seçimi yapılarak *Başla* ve *Son* blokları ile Şekil 4'teki akış şeması hazırlanır.



Şekil 4 Sembol deneme Akış Şeması örneği.