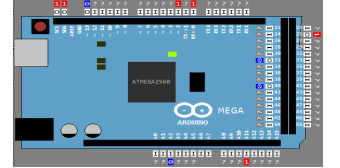


# UnoArduSimV2.8.1 Tam Yardım



## İçindekiler

### Genel bakış

### Kodlama Paneli, Tercihler ve Düzenle/İnceleme

#### Kodlama Paneli

#### Tercihler

#### Düzenle/İnceleme

### Değişken Paneli ve Düzenle/İzle Değişken pencere

### Laboratuvar Tezgah Paneli

#### 'Uno' veya 'Mega'

#### 'I/O' Cihazlar

##### 'Serial' Monitör ('SERIAL')

##### Alternatif Seri ('ALTSER')

##### Katı Hal Diski ('SD\_DRV')

##### TFT Ekran ('TFT')

##### Yapılandırılabilir SPI Bağımlı ('SPISLV')

##### İki Telli I2C Bağımlı ('I2CSLV')

##### Metin LCD I2C ('LCDI2C')

##### Metin LCD SPI ('LCDSPI')

##### Metin LCD D4 ('LCD\_D4')

##### Çoklayıcı LED I2C ('MUXI2C')

##### Çoklayıcı LED SPI ('MUXSPI')

##### Genişleme Bağlantı Noktası SPI ('EXPSPi')

##### Genişleme Bağlantı Noktası I2C ('EXPI2C')

##### '1-Wire' Bağımlı ('OWIISLV')

##### Kayan Yazmaç Bağımlı ('SRSLV')

##### Tek Çekimli ('1SHOT')

##### Programlanabilir 'I/O' Cihaz ('PROGIO')

##### Dijital Darbeci ('PULSER')

##### Analog Fonksiyon Jeneratör ('FUNCGEN')

##### Kademe Motoru ('STEPR')

##### Darbeli Kademe Motoru ('PSTEPR')

##### DC Motoru ('MOTOR')

##### Servo Motoru ('SERVO')

##### Piezzo Hoparlör ('PIEZO')

##### Slayt Direnci ('R=1K')

##### Butona Basınız ('PUSH')

##### Renkli LED ('LED')

##### 4-LED Satırı ('LED4')

##### 7 Bölümlü LED Sayı ('7SEG')

##### Analog Kaydırıcı

##### Pin Tel Bağlayıcı ('JUMP')

### Menüler

#### Dosya:

Doldurmak INO veya PDE Prog (Ctrl-L)

Düzenle/İnceleme (Ctrl-E)

Kaydet

Farklı Kaydet

Sonraki ('#include')

Önceki

Kapat

#### Bul:

Yukarı taşı Çağrı yığını

Aşağı inmek Çağrı yığını

Ara Metnini Ayarla (CTRL-F)

Bul Sonraki Metin

[Bul Önceki Metin](#)

### [Çalıştır:](#)

[İçine Adım \(F4\)](#)  
[Etrafında Adım \(F5\)](#)  
[Dışarıda Adım \(F6\)](#)  
[Komuta Çalıştır \(F7\)](#)  
[Koşula Çalıştır \(F8\)](#)  
[Çalıştır \(F9\)](#)  
[Dur \(F10\)](#)  
[Reset](#)  
[Oynat](#)  
[Yavaş](#)

### [Seçenekler:](#)

[Etrafında Adım Yapıları/ Operatörler](#)  
[Kayıt-Tahsisi](#)  
[Başlatılmamış Hata](#)  
[Katma 'loop\(\)' gecikme](#)  
[Yuvalanmış Kesmelere İzin Ver](#)

### [Yapılandır:](#)

['I/O' Cihazlar](#)  
[Tercihler](#)

### [VarYenile:](#)

[Otomatik İzin Ver \(-\) Opposite of exand the displayed array\), daralt](#)  
[En az](#)  
[Vurgulu değişiklikler](#)

### [Pencereler:](#)

['Serial' Monitör](#)  
[Her şeyi eski haline getir](#)  
[Pin Dijital Dalga Formları](#)  
[Pin Analog Dalga Formu](#)

### [Yardım:](#)

[Hızlı Yardım Dosya](#)  
[Tam Yardım Dosya](#)  
[Hata Düzeltmeleri](#)  
[Değişim / İyileştirmeler](#)  
[hakkında](#)

['Uno' veya 'Mega' Kartı ve 'I/O' Cihazlar](#)

[Zamanlama](#)

['I/O' Cihaz Zamanlama](#)

[Sesler](#)

## [Sınırlamalar ve Desteklenmeyen Öğeler](#)

[Dahil Dosyalar](#)

[Dinamik Bellek Ayırma ve RAM](#)

['Flash' Bellek Tahsisleri](#)

['String' Değişkenler](#)

[Arduino Kütüphaneleri](#)

[İşaretçiler](#)

['class' ve 'struct' Includes 'struct' and 'class'\), nesneleri](#)

[Kapsam](#)

[elemeleri 'unsigned', 'const', 'volatile', 'static'](#)

[Derleyici Direktifleri](#)

[Arduino dili elemanları](#)

[C / C ++ - dil Elemanları](#)

[İşlev Şablonları](#)

[Gerçek Zamanlı Emülasyon](#)

## [Sürüm notları](#)

[Hata Düzeltmeleri](#)

[V2.8.1- Haziran 2020](#)

[V2.8.0- Haziran 2020](#)

[V2.7.0- Mart 2020](#)

[V2.6.0- Ocak 2020](#)

[V2.5.0- Ekim 2019](#)

[V2.4 - Mayıs 2019](#)

[V2.3 - Aralık 2018](#)

[V2.2 - Haziran. 2018](#)

[V2.1.1 - Mar. 2018](#)

[V2.1 - Mar. 2018](#)

[V2.0.2 Şubat 2018](#)

[V2.0.1 - Ocak 2018](#)

[V2.0- Aralık 2017](#)

[V1.7.2- Şubat. 2017](#)

[V1.7.1- Şubat. 2017](#)

[V1.7.0- Aralık 2016](#)

[V1.6.3 - Eylül 2016](#)

[V1.6.2 - Eylül 2016](#)

[V1.6.1 - Ağustos 2016](#)

[V1.6 - Haziran 2016](#)

[V1.5.1 - Haziran 2016](#)

[V1.5 - Mayıs 2016](#)

[V1.4.3 - Nisan 2016](#)

[V1.4.2 - Mart 2016](#)

[V1.4.1 - Ocak 2016](#)

[V1.4 - Aralık 2015](#)

[V1.3 - Ekim 2015](#)

[V1.2 - Haziran 2015](#)

[V1.1 - Mart 2015](#)

[V1.0.2 - Ağustos 2014](#)

[V1.0.1 - Haziran. 2014](#)

[V1.0 - ilk sürüm Mayıs 2014](#)

#### [Değişiklikler / İyileştirmeler](#)

[V2.8.0- Haziran 2020](#)

[V2.7.0- Mart 2020](#)

[V2.6.0 Ocak 2020](#)

[V2.5.0 Ekim 2019](#)

[V2.4 Mayıs 2019](#)

[V2.3 Aralık 2018](#)

[V2.2, Haziran 2018](#)

[V2.1 Mart 2018](#)

[V2.0.1 Ocak 2018](#)

[V2.0 Eylül 2017](#)

[V1.7.2- Şubat 2017](#)

[V1.7.1 - Şubat 2017](#)

[V1.7.0- Aralık 2016](#)

[V1.6.3 - Eylül 2016](#)

[V1.6.2 - Eylül 2016](#)

[V1.6.1 - Ağustos 2016](#)

[V1.6 - Haziran 2016](#)

[V1.5.1 - Haziran 2016](#)

[V1.5 - Mayıs 2016](#)

[V1.4.2 - Mart 2016](#)

[V1.4 - Aralık 2015](#)

[V1.3 - Ekim 2015](#)

[V1.2 Haziran 2015](#)

[V1.1 - Mart 2015](#)

[V1.0.1 - Haziran. 2014](#)

[V1.0 - ilk sürüm Mayıs 2014](#)

## Genel bakış

UnoArduSim ücretsiz bir yazılımdır **gerçek zaman** (görmek Zamanlama için **kısıtlamalar** ) Ben öğrenci ve Arduino meraklısı için geliştirdiğim simülatör aracı. Arduino programlar ile deney yapmanıza ve kolayca hata ayıklamanıza izin vermek için tasarlanmıştır **herhangi bir gerçek donanıma ihtiyaç duymadan** . Hedeflenir **Arduino 'Uno' veya 'Mega'** kartı ve bir dizi sanal 'I/O' cihazlar seçmenize ve bu cihazlar'yi sanal 'Uno' veya 'Mega'e **Laboratuvar Tezgah Paneli** . - kablolama hataları, kopuk / kopuk bağlantılar veya cihazlar'nizin bozulmasına ve test edilmesine neden olan hatalı cihazlar hakkında endişelenmenize gerek yok.

UnoArduSim karşılaştığı tüm ayrıştır veya icra hataları için basit hata mesajları verir ve hata ayıklamayı sağlar **Reset , Çalıştır , Komuta Çalıştır, Koşula Çalıştır , Dur** ve esnek **Adım** operasyonları **Kodlama Paneli** , tüm küresel ve şu anda aktif olan yerel yerel değişkenler, diziler ve includes 'struct' and 'class'), nesneleri'ün eşzamanlı **Değişken Paneli** . Çalıştır zamanlı dizin sınır kontrolü yapılır ve ATmega RAM taşması tespit edilir (ve suçlu program hattı vurgulanır!). Herhangi bir elektrikli ile çatışmak ekli 'I/O' cihazlar işaretlenir ve oluşukça rapor edilir.

Bir INO veya PDE program dosya açıldığında, program'ye yüklenir. **Kodlama Paneli** . program daha sonra bir Ayrıştır, daha sonra kullanıma hazır olan tokenize bir çalıştırılabilir dosyaya dönüştürmek için **benzetilmiş icra** (Arduino.exe'den farklı olarak, bağımsız bir ikili çalıştırılabilir **değil** oluşturulan) Herhangi bir ayrıştır hatası tespit edildi ve işaretlendi ve ayrıştır'ye başarısız olan satırı vurgulayarak ve hatayı bildirerek işaretlendi **Durum çubuğu** UnoArduSim uygulamasının en altında bulunan pencere. bir **Düzenle/İnceleme** pencere, kullanıcınız program'nin sözdizimini vurgulayan bir sürümünü görmenize ve düzenlemenize olanak sağlamak için açılabilir. Simüle edilmiş icra sırasındaki (Durumla eşleşen baud hızı gibi) hatalar Durum Çubuğunda ve bir açılır mesaj kutusu aracılığıyla bildirilir.

UnoArduSim V2.8, cihazın eksiksiz bir şekilde uygulanmasıdır. **Arduino Programlama Dili V1.8.8 belgesinde belirtildiği gibi [arduino.cc](http://arduino.cc)** . Dil Başvurusu web sayfası ve sürüm İndirme sayfasında belirtilen eklerle birlikte Sürüm Notları. UnoArduSim, GNU derleyici'in altındaki Arduino.exe'nin yaptığı tam C / C ++ uygulamasını desteklemese de, yalnızca en gelişmiş programcılarının kullanmak istedikleri bazı C / C ++ elemanının eksik olduğunu bulması muhtemeldir (ve elbette her zaman basit bu eksik özellikler için çalışma çevresini kodlama). Genel olarak, sadece Arduino hobileri ve öğrencileri için en faydalı C / C ++ özellikleri olduğunu düşündüğüm şeyi destekledim - örneğin, 'enum' ve '#define' desteklenir, ancak işlev-işaretçiler değildir. Kullanıcı tanımlı includes 'struct' and 'class'), nesneleri olsa bile ( 'class' ve 'struct' ) ve (çoğu) operatör aşırı yüklemesi destekleniyor, **çoklu miras değil** .

UnoArduSim bir üst seviye dil simülatörü olduğundan, **sadece C / C ++ ifadeleri desteklenir** , **derleme dili ifadeleri değil** . Benzer şekilde, düşük seviyeli bir makine simülasyonu olmadığı için, **ATmega328 kayıtları program'niz için erişilebilir değil** ya okuma ya da yazma için, sicil tahsisi, geçiş ve geri dönüş taklit edilmesine rağmen (bunu menüde seçtiniz) **Seçenekler** ).

V2.6 itibarıyla UnoArduSim Arduino sınırlı alt kümesi için dahili otomatik destek kütüphaneleri sağlanan vardır, bu varlık: 'Stepper.h' , 'Servo.h' , 'SoftwareSerial.h' , 'SPI.h' , 'Wire.h' , 'OneWire.h' , 'SD.h' , 'TFT.h' ve 'EEPROM.h' (versiyon 2). V2.6 3 için bir mekanizma sunar <sup>rd</sup> dosyalar aracılığıyla parti kütüphane desteği sağlanan 'include\_3rdParty' Klasör o dizini yüklemek UnoArduSim içinde bulunabilir. Herhangi '#include' kullanıcı tarafından oluşturulan kütüphanelerin listesi, UnoArduSim **değil** kütüphaneyi bulmak için her zamanki Arduino kurulum dizini yapısını arayın; senin yerine **gerek** karşılık gelen üstbilgiyi (".h") ve kaynağını (".cpp") dosya üzerinde çalıştığınız program dosya ile aynı dizine kopyalamak için (elbette herhangi bir içeriğin '#include' dosya tamamen anlaşılabilir olmalıdır UnoArduSim ayrıştırıcı).

...

QtCreator'da UnoArduSimV2.0'ı çoklu dil desteğiyle geliştirdim ve şu anda yalnızca Pencereleler için kullanılabilir <sup>TM</sup> . Linux veya MacOS'a taşınmak, gelecek için bir projedir! UnoArduSim, yıllar boyunca Queen's Üniversitesi'nde ders aldığım dersler için geliştirdiğim simülatörlerden büyüdü ve oldukça kapsamlı bir şekilde test edildi, ancak hala orada saklanan birkaç hatalar olması gerekiyor. Bir hata'ı rapor etmek istiyorsanız, lütfen bunu bir e-postada (kısaca) açıklayın. [unoArduSim@gmail.com](mailto:unoArduSim@gmail.com) ve ***hata-indükleyici program Arduino kaynak kodunu tam olarak eklediğinizden emin olun.*** böylece hata'ı kopyalayabilir ve düzeltebilirim. Bireysel hata raporlarına cevap vermeyeceğim ve sonraki sürümlerde düzeltmeler için garantili zaman çizelgem yok (neredeyse her zaman geçici çözümler olduğunu unutmayın!).

Alkış,

Stan Simmons, PhD, P.Eng.  
Doçent (emekli)  
Elektrik ve Bilgisayar Mühendisliği Bölümü  
Kraliçe'nin Üniversitesi  
Kingston, Ontario, Kanada








## Kodlama Paneli, Tercihler ve Düzenle/İnceleme



(Kenara: Aşağıda gösterilen pencereler örneği bir kullanıcı tarafından seçilen Pencereler-OS altındadır koyu mavi pencere arka plan rengine sahip renk teması).

### Kodlama Paneli

The **Kodlama Paneli** kullanıcı program'yi gösterir ve onun icra'i izleyerek  
Yüklenen program'nin başarılı bir Ayırıştır'ten sonra, ilk satırdaki 'main()' vurgulanır ve program icra için hazırır.  
Bunu not et 'main()' örtülü olarak Arduino (ve UnoArduSim) tarafından eklenir ve **değil** Kullanıcıya program dosya  
bir parçası olarak dahil. İcra menünün kontrolü altında **Çalıştır** ve onunla ilişkili **Aracı-Bar** düğmeleri ve işlev tuşlu  
kısayollar.

Biri (veya daha fazla) talimatlarında icra adım sonra  
(kullanabilirsiniz **Aracı-Bar** düğmeleri , ,  veya ),  
Önümüzdeki ardından yeşil vurgulanır çalıştı olacak program  
hat - yeşil-vurgulanan çizgi her zaman bir sonraki çizgidir  
**çalıştı olmaya hazır**.

Program icra anda kilitlendi, ve sen tıklayın **Kodlama Paneli**  
pencere, (resimde gösterildiği gibi) sadece karanlık zeytin  
vurgulanan olur tıklandığında çizgi - (v2.7 gibi) yeşil renkle  
vurgulanır sonraki-to-be-çalıştı hattı her zaman kalır. . Bu  
yapar *değil* Bununla birlikte, mevcut program hattını program  
icra ile ilgili olarak değiştirin. Ancak icra'e neden olabilirsiniz  
*ilerlemek için* sadece vurguladığınız çizgiyi tıklayın ve  
ardından **Komuta Çalıştır**  **Aracı-Bar** buton. Bu özellik, bir  
program'deki belirli çizgilere hızlı ve kolay bir şekilde  
erişmenizi sağlar; böylece daha sonra ilgilenilen bir program  
kısının üzerinden satır satır ilerleyebilirsiniz.

Eğer yüklü program'niz varsa '#include' dosyalar'u  
kullanarak aralarında geçiş yapabilirsiniz. **Dosya | Önceki** ve  
**Dosya | Sonraki** (ile **Aracı-Bar** düğmeleri  ve ).






```
/* This is a default program--
   Use File->Load Prog to load a different program
*/

int count;

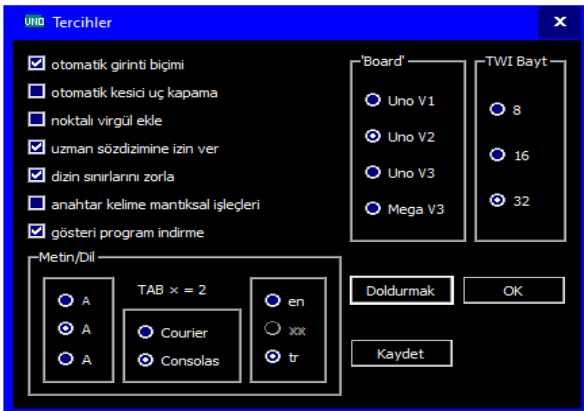
void setup()
{
  count=0;
}

void loop()
{
  count=count+1;
  delay(100);
}

//the "int main()" below is IMPLICIT in Arduino
//but is shown here EXPLICITLY by UnoArduSim
int main()
{
  setup();
  while(true)
  {
    loop();
    serialEventRun();
    delay(1); //added by the Arduino team
  }
}
```

eylemleri **Bul** Menü için izin **YA** metin bulmak **Kodlama Paneli** veya **Değişken Paneli** (**Aracı-Bar** düğmeleri  ve ),  
Veya klavye kısayolları **yukarı ok** ve **aşağı ok** ) **İlk kullandıktan sonra Bul | Set Ara metin** veya **Aracı-Bar** ,  
**VEYA ALTERNATİF OLARAK** için **gezinmek çağrı yığını** içinde **Kodlama Paneli** (**Aracı-Bar** düğmeleri  ve ),  
Veya klavye kısayolları **yukarı ok** ve **aşağı ok** ). Anahtarlar **PgDn** ve **PgUp** sonraki / önceki işlev Seçimi atlamak .

### Tercihler



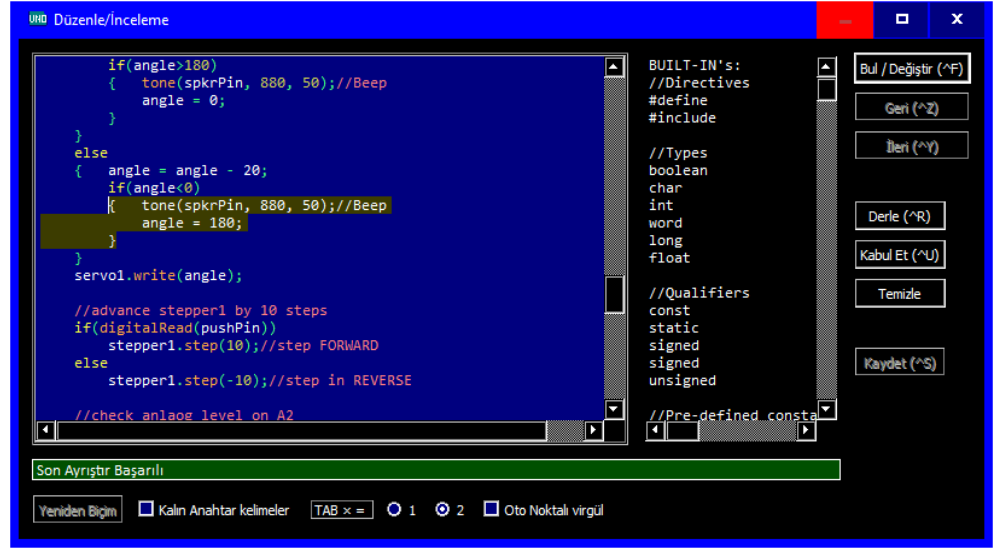
**Yapılandır | Tercihler** kullanıcılara izin verir program'yi ve  
görüntüleme tercihlerini (bir kullanıcının normalde o bir sonraki  
oturumda). Bu nedenle bunlar bir cihazdan kaydedilebilir ve  
yüklenilebilir. 'myArduPrefs.txt' Yüklenen 'Uno' veya 'Mega'  
program ile aynı dizinde bulunan dosya ( 'myArduPrefs.txt'  
varsa otomatik olarak yüklenir).

Bu iletişim kutusu iki adet mono aralıklı yazı tipi ve üç tür boyut  
ve diğer çeşitli tercihler arasında bir seçime izin verir. V2.0'dan  
itibaren, dil seçimi şimdi dahil edilmiştir. - bu daima İngilizce'yi  
içerir (**tr**), artı bir veya iki diğer kullanıcı yerel dilini (bu yerlerde),  
ve iki harfli ISO-639 dil kodunu temel alan bir geçersiz kılma *ilk*  
*satırda* arasında 'myArduPrefs.txt' dosya (eğer varsa).  
Seçenekler yalnızca görünürse bir ".qm" çevirisi dosya, çeviri  
klasöründe bulunur (içinde UnoArduSim.exe giriş dizini).

## Düzenle/İnceleme

İçindeki herhangi bir satıra çift tıklayarak **Kodlama Paneli** (veya menüyü kullanarak **Dosya** ), bir **Düzenle/İnceleme** pencere'te program dosya'nızda değişiklik yapılmasına izin vermek için **şu anda seçili satır** içinde **Kodlama Paneli** vurgulanmış.

Bu pencere, dinamik sözdizimi vurgulama özelliğine sahiptir (farklı vurgulu renkleri, C ++ anahtar kelimeler, yorumlar vb. için kullanılır). İsteğe bağlı kalın sözdizimi vurgulama ve otomatik girinti düzeyi biçimlendirme var (kullanarak kullanmayı seçtiniz. **Yapılandır | Tercihler** ). Ayrıca dahili işlev çağrılarını (veya dahili) uygun şekilde seçebilirsiniz. '#define' sabitler) program'nize verilen liste kutusundan eklenecek - mevcut caret pozisyonunda program'nize eklemek için istediğiniz liste kutusu öğesine çift tıklayın (işlev-çağrı değişken **türleri** sadece bilgi amaçlıdır ve program'nize eklendiğinde sahte yer tutucuları bırakmak için çıkarılmıştır).



pencere vardır **Bul** (Kullanım **CTRL-F**) ve **Bul / Değiştir** yetenek (kullanım **Ctrl-H** ). The **Düzenle/İnceleme** pencere vardır **Geri** ( **CTRL-Z** ), ve **İleri** ( **Ctrl-Y** ) düğmeleri (otomatik olarak görünür).

**Kullanım ALT-sağ-ok** dahili talebi otomatik tamamlama seçeneklerine **Küresel değişkenler**, ve için **üye değişkenler ve işlevler**.

Atmak için **tüm değişiklikler** program'yi düzenleme için ilk açtığınızdan beri yaptınız. **Temizle** buton. Kabul etmek Mevcut durumda, tıklayın **Kabul Et** düğmesini ve program otomatik olarak başka bir Ayrıştır alır (ve hata bulunmazsa 'Uno' veya 'Mega'e indirilir) ve yeni durum ana UnoArduSim pencere'de görünür **Durum çubuğu** .

bir **Derle** ( **Ctrl-R** ) düğmesi (artı ilişkili **Ayrıştır Durumu** Yukarıdaki resimde görüldüğü gibi mesaj kutusu), pencere'ü kapatmaya gerek kalmadan düzenlemelerin test edilmesine olanak sağlamak için eklenmiştir. bir **Kaydet** ( **Ctrl-S** ) düğmesi de kısayol olarak eklenmiş **Kabul Et** artı daha sonra ayrı **Kaydet** Ana pencere'ten).

Her ikisinde de **Temizle** veya **Kabul Et** hiçbir düzenleme yapılmadan **Kodlama Paneli** mevcut çizgi **son Düzenle/İnceleme şapka pozisyonu** , ve bu özelliği kullanmak için kullanabilirsiniz. **Kodlama Paneli** belirli bir hatta (muhtemelen bir **Komuta Çalıştır** ), Ayrıca kullanabilirsiniz **ctrl-PgDn** ve **ctrl-PgUp** program'nizdeki bir sonraki (veya önceki) boş satır aralığına atlamak için - bu, önemli konumlara hızlıca yukarı veya aşağı gitmek için yararlıdır (işlevler arasındaki boş satırlar gibi). Ayrıca kullanabilirsiniz **ctrl-Ev** ve **ctrl-End** program başlangıcına atlamak ve sırasıyla

**'Tab' seviyeli otomatik girinti biçimlendirme** Bu seçenek altında kurulmuştur pencere, açtığında yapılı **Yapılandır | Tercihler**. Tıklayarak o istediği zaman biçimlendirme yeniden yapabilirsiniz **Yeniden Biçim** Daha önce seçtiyseniz düğmesi (sadece etkindir **Otomatik girinti Tercihi**). Ayrıca klavye kullanarak önceden seçilmiş ardışık hatlarının bir grup silme sekme kendi ekleyebilir veya **sağ ok** veya **sol ok** anahtarlar - fakat **Otomatik girinti Tercihi kapalı olmalıdır** Kendi özel sekme seviyelerini kaybetmemek için.

Ne zaman **Otomatik Noktalı Noktalar** işaretli **Girmek** Bir çizgiyi sonlandırmak için, otomatik olarak satır sonlandırıcı noktalı virgül ekler.

Ve bağlamınızı ve süslü ayraç'yi daha iyi izlemenize yardımcı olmak için ' { ' veya ' } ' süslü ayraç **bu süslü ayraç ile eşi arasındaki eşi arasındaki bütün metni vurguluyor** .



## Değişken Paneli ve Düzenle/İzle Değişken pencere

The **Değişken Paneli** hemen altında bulunur **Kodlama Paneli**. Yüklenen program'deki tüm küresel ve aktif (kapsam'teki) yerel değişken / dizin / includes 'struct' and 'class'), nesne'deki mevcut değerleri gösterir. program icra'ın işlevler **içeriği, yalnızca geçerli işlev / kapsam'ın erişebileceği yerel değişkenler'yi ve ayrıca kullanıcı tarafından beyan edilen tüm globları yansıtacak şekilde**. Olarak bildirilen herhangi bir değişkenler 'const' ya da 'PROGMEM' ('Flash'e tahsis edildi hafıza) değiştirilemeyen değerlere sahip ve bu nedenle yer kazanmak için **görüntülenmedi**. 'Servo' ve 'SoftwareSerial' includes 'struct' and 'class'), nesne örnekleri hiçbir faydalı değer içermez, bu nedenle görüntülenmez.

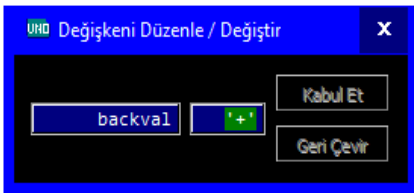
Yapabilirsin **bulmak** belirtildi **Metin** onun **metin** arama komutlarıyla (ile **Aracı-Bar** düğmeleri **text** ve **text** Veya klavye kısayolları **yukarı ok** ve **aşağı ok**), İlk olarak kullanıldıktan sonra **Bul | Set Ara** **metin** veya **text**.

```
LED_pin= 5
angle= 135
i= 3
k= 6
notefreq= 1046
dur= 0.12500
beats= 160
wholenote= 1500
quarternote= 375
msecs= 375
RingTones[0](-)
RingTones[0].frequency= 1046
RingTones[0].duration= 0.12500
```

**Diziler** ve includes 'struct' and 'class'), nesneleri ikisinde de gösterilir **un-genişletilmiş** veya **genişletilmiş** ya sonunda bir artı ' (+) ' veya eksi ' (-) ' sırasıyla dizin'ün simgesi **x** olarak gösterir 'x[]' . genişletmek'e dizin'ün tüm öğelerini gösterecek 'x[] (+) ' içinde **Değişken Paneli** . opposite of expand the displayed array), daralt'ü genişletilmiş'in bir görünümüne geri döndürmek için 'x[] (-) ' . includes 'struct' and 'class'), nesne için un-genişletilmiş varsayılanı 'p1' olarak gösterir 'p1 (+) ' genişletmek'e bütün üyelerini göstermek 'class' veya 'struct' örneği, tek tıkla 'p1 (+) ' içinde **Değişken Paneli** . opposite of expand the displayed array), daralt'e genişletilmiş'li olmayan bir görünüme dönmek için, tek tıkla 'p1 (-) ' . Eğer sen **Karanlık zeytinde vurgulu bunun için herhangi bir hat üzerinde tek tıklama** (Bu değişken veya toplu basit olabilir ' (+) ' veya ' (-) ' Bir dizin veya includes 'struct' and 'class'), nesne veya bir tek dizin elemanı veya includes 'struct' and 'class'), nesne elemanı) hattı, Sonra a yapıyor **Koşula Çalıştır** icra sürdürmeye ve bir sonraki en donmasına neden olur **yazmaya karşı erişim** her yerde seçili ağrega içinde ya da bu tek değişken konumu seçilir.

Kullanırken **Adım** veya **Çalıştır** , gösterilen değişken değerlerinde yapılan güncellemeler, menü altında yapılan kullanıcı ayarlarına göre yapılır. **VarYenile** - Bu, minimum periyodik güncellemelerden anında yapılan güncellemelere kadar çok çeşitli davranışlara izin verir. Azaltılmış veya az güncelleme, CPU yükünü azaltmak için kullanışlıdır ve icra'ın aksi halde aşırı olan şeyin altında gerçek zamanlı olarak kalmamasını sağlamak için gerekli olabilir **Değişken Paneli** pencere güncelleme yükleri. Ne zaman **Oynat** yürürlükte veya **Vurgulu Değişiklikleri** menü seçeneği seçiliyken, değişken'in **Çalıştır** görüntülenen değerinin güncellenmesine neden olacak **hemen** , ve vurgulanır hale gelir - bu, **Değişken Paneli** değişken ve icra artık gerçek zamanlı olmayacak tutan çizgiye ilerleyin (gerekirse).

**icra donduğunda** sonra **Adım** , **Komuta Çalıştır** , **Koşula Çalıştır** veya **Çalıştır** -sonra- **Dur** , **Değişken Paneli** karşılık gelen değişken'i **değiştirilen adres konumları** (varsa) tarafından **en son talimat** bu sırada icra (değişken beyannamesi başlatmaları dahil). Eğer bu talimat **tamamen** bir dolu **includes 'struct' and 'class'), nesne veya dizin , üst (+) veya (-) satır** Çünkü bu toplam vurgulanır. Bunun yerine, komut bir yer şu anda görünür durumda, sonra vurgulanıyor. Ancak, değiştirilmiş konum (lar) şu anda içeride saklanıyorsa bir toplu bir genişletilmiş dizin veya includes 'struct' and 'class'), nesne **ana satır** bir alır **italik yazı tipi vurgulama** içindeki bir şeyin yazıldığı görsel bir ipucu olarak - genişletmek'e tıklamak **son** değiştirilecek öge veya üye vurgulanacak.



The **Düzenle/İzle** pencere size verir **icra sırasında herhangi bir değişken değerini takip edebilme** veya **değerini (durdu) program icra ortasında değiştir** (böylece devam etmenin bu yeni değerle devam etmesinin ne olacağını test edebilirsiniz). **Dur** Önce icra, sonra **sol tıklama** değerini izlemek veya değiştirmek istediğiniz değişken'te. program icra sırasındaki değeri izlemek için, **iletişim kutusunu açık bırak** ve sonra biri **Çalıştır** veya **Adım** komutlar - değeri güncellendi **Düzenle/İzle** güncellemelerini düzenleyen kurallara göre **Değişken**

**Paneli** . **değişken değerini değiştirmek için** , düzenleme kutusu değerini doldurun ve **Kabul Et** . icra'e devam edin (herhangi birini kullanarak **Adım** veya **Çalıştır** komutlar) bu yeni değeri o noktadan itibaren kullanmak için (veya **Geri Çevir** önceki değere).

**program Doldurmak veya Reset'de** hepsine dikkat et **başlatılmamış değer değişkenler**, 0 değerine sıfırlanır ve tüm **başlatılmamış gösterici-değişkenler**, 0x0000 değerine sıfırlanır.



## Laboratuvar Tezgaah Paneli

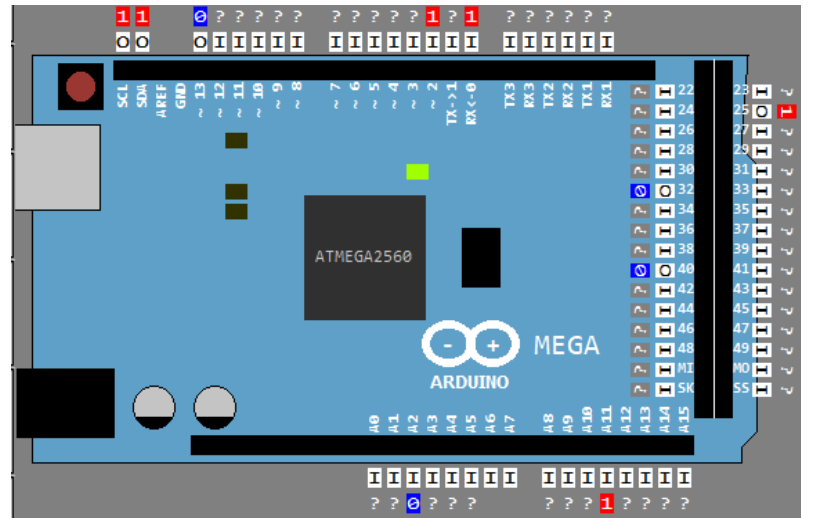
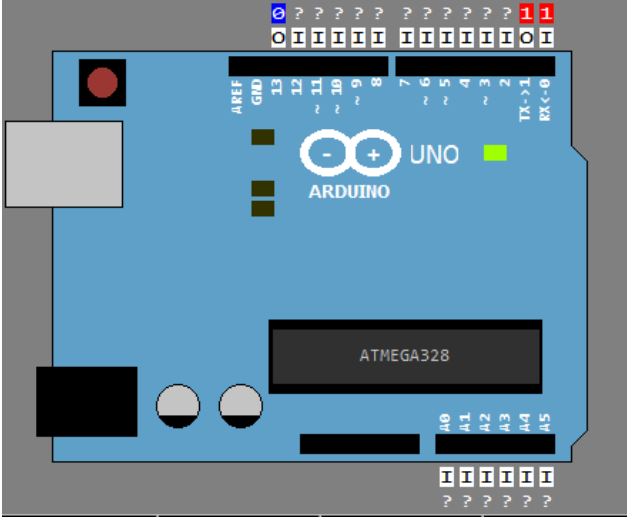
Laboratuvar Tezgaah Paneli, seçebileceğiniz / özelleştirebileceğiniz ve istediğiniz 'Uno' veya 'Mega' pins'e bağlayabileceğiniz bir 'I/O' cihazlar seti ile çevrili 5 voltluk bir 'Uno' veya 'Mega' kartı'ı gösterir.

### 'Uno' veya 'Mega'

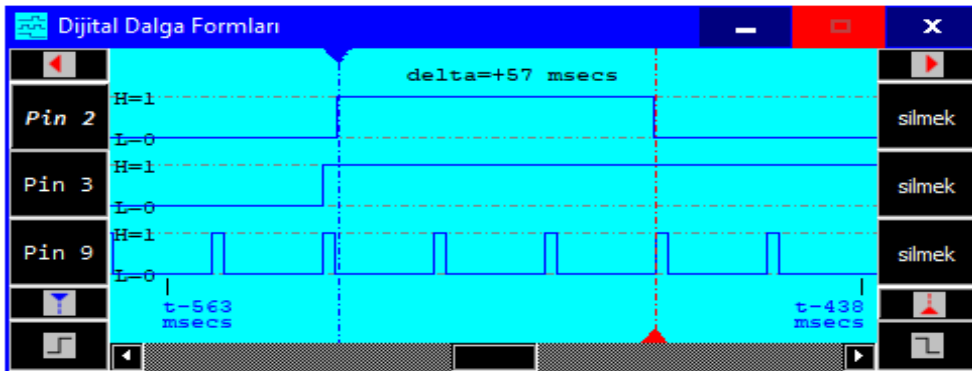
Bu, 'Uno' kartı'nın ve onboard LED'lerinin bir tasviridir. UnoArduSim'e yeni bir program yüklediğinizde, başarılı bir şekilde ayrıştırırsa, 'Uno'ye gerçek bir 'Uno' kartı'nın davranış şeklini taklit eden bir "benzetilmiş indirme" den geçerse - seri RX ve TX LED'in yanıp söndüğünü göreceksiniz (pins 1'deki etkinlikle birlikte) ve 0 olan *bir ana bilgisayarla seri iletişim için kablolu*). Bunu hemen ardından kartı sıfırlamasını belirten bir pin 13 LED flaşı ve yüklenen program icra'nın başlangıcını (ve UnoArduSim otomatik durduğunda) izler. Seçimi kaldırarak bu ekrandan ve ilişkili yükleme gecikmesinden kaçınabilirsiniz. **İndirme'ü göster** itibaren **Yapılandır | Tercihler**.

pencere, 20 'Uno' pins'in tamamında dijital mantık seviyelerini görmeyi sağlar ( '1' için kırmızı 'HIGH' , '0' için mavi 'LOW' , ve ' ? ' belirsiz bir belirsiz voltaj için gri üzerine) ve programlanmış yönlerine ( 'I' için 'INPUT' veya 'O' için 'OUTPUT' ). PWM kullanılarak darbeli olan pins için 'analogWrite()' , veya tarafından 'tone()' , veya tarafından 'Servo.write()' , renk mor ve görüntülenen sembol olarak değişir olur '^' .

Bunu not et **Dijital pins 0 ve 1, 1-kOhm dirençler üzerinden USB çipine kablolanmış bir ana bilgisayarla seri iletişim**.













**Sol tıklandığında** Herhangi bir 'Uno' veya 'Mega' pin açılacak **Pin Dijital Dalga Formları** Geçmişi gösteren pencere **bir saniye değerinde** arasında **dijital düzeyinde aktivite** Bunun üzerine pin. Bunları Pin Dijital Dalga Formları ekranına eklemek için diğer pins'e tıklayabilirsiniz (herhangi bir zamanda en fazla 4 dalga formuna).



Sayfa görünümü için sola veya sağa tıklayın veya Tuşları kullanın Ana Sayfa, PgUp, PgDn, End

Görüntülenen dalga biçimlerinden biri **aktif pin** e signal shape), dalgabiçimi "Pin" tuşuyla basıldığında, (yukarıdaki Pin Dijital Dalga Formları ekran görüntüsünde olduğu gibi) basılı olarak gösteriliyor. programlar sayı düğmesini tıklatarak bir e signal shape), dalgabiçimi seçebilir ve ardından uygun yükselen / düşen kenar polarite seçim düğmesini tıklatarak

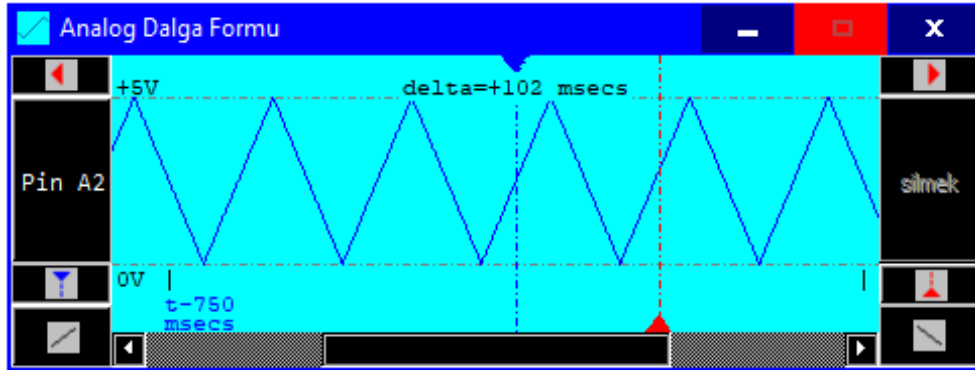
ilgilenilen kenar polaritesini seçebilirsiniz,  veya  veya kısayol tuşlarını kullanarak **yukarı ok** ve **aşağı ok** . Sonra yapabilirsin **atlama** aktif imleç (mavi veya kırmızı imleç, delta zamanları gösterilmiş şekilde çizgilerle) seçilen kutup dijital kenarına ileri veya geri **bu aktif pin'ün** e signal shape), dalgabiçimi imleç düğmelerini kullanarak (  ,  veya  ,  ) (hangi imlecin olduğuna bağlı olarak ile önceden etkinleştirilmiş  veya  ) veya sadece klavye tuşlarını kullanın  $\leftarrow$  ve  $\rightarrow$  .

Bir imleci etkinleştirmek için, renkli etkinleştirme düğmesine tıklayın (  veya  Yukarıda verilen) - **bu aynı zamanda görünümü geçerli konumuna atlar. o imleç** . Alternatif olarak, kısayolu kullanarak imleçler arasında (sırasıyla ortalananmış görünümleriyle) etkinleştirme işlemini hızlıca değiştirebilirsiniz. **'Tab'** tuşuna basın.







Yapabilirsin **atlama** şu anda etkin olan imleci **herhangi bir yere sol tıklayarak** Ekrandaki e signal shape), dalgabiçimi görünüm bölgesinde. Alternatif olarak, kırmızı üzerine veya mavi imleç çizgisini, üstüne sağ tıklayarak (etkinleştirmek için), ardından **sürükleyip yeni bir yer** ve bırakın. İstedığınız bir imleç şu anda ekran dışında bir yerde olduğunda, **herhangi bir yere sağ tıklayın** görünümünde bu yeni ekran konumuna atlamak için. Her iki imleç zaten ekrandaysa, sağ tıklatıldığında etkin imleç arasında geçiş yapılabilir.

**ZOOM IN ve ZOOM OUT'a (zoom her zaman AKTİF imleci üzerine gelir), fare tekerleğini veya klavye kısayollarını kullanarak CTRL yukarı ok ve CTRL aşağı ok.**

Yerine bir **sağ tık herhangi bir 'Uno' veya 'Mega' pin üzerinde** bir açar **Pin Analog Dalga Formu** Görüntüleyen pencere **1 saniyede bir değer geçmiş** arasında **analog düzeyinde aktivite** Bunun üzerine pin. Pin Dijital Dalga Formları pencere'nin aksine, Bir seferde sadece bir pin analog aktivitesini gösterir.



Sayfa görünümü için sola veya sağa tıklayın veya Tuşları kullanın Ana Sayfa, PgUp, PgDn, End

Yapabilirsin **atlama** the mavi veya kırmızı imleç ileri veya geri ok tuşlarını kullanarak bir sonraki yükselen veya düşen "eğim noktasına" doğru çizgiler (  ,  veya  ,  , imleci etkin imlece bağlı olarak tekrar kullanın veya  $\leftarrow$  ve  $\rightarrow$  tuşları) yükselen / düşen eğim seçim butonları ile uyumlu  ,  ("eğim noktası", analog voltajının ATmega pin yüksek dijital-mantık seviyesi eşliğinden geçtiğinde meydana gelir). Alternatif olarak, Pin Dijital Dalga Formları pencere'teki davranışlarına benzer şekilde bu imleç satırlarını atlamak için tıklayabilir veya sürükleyebilirsiniz.

basma **'Ctrl-S'** içeride **pencere, e signal shape), dalgabiçimi'u kaydetmenize olanak verir (X, Y) verileri** istediğiniz metin dosya'ye **X** sol taraftan mikrosaniye cinsindendir ve **Y** civata içinde.

## 'I/O' Cihazlar

Bir dizi farklı cihazlar, 'Uno' veya 'Mega', **Laboratuvar Tezgaah Paneli** . "Küçük" 'I/O' cihazlar (bunlara toplam 16'ya izin verilir) Paneli'nin sol ve sağ kenarları boyunca bulunur. "Büyük" 'I/O' cihazlar (toplamda 8'e kadar izin verilir), "aktif" elementlere sahiptir ve **Laboratuvar Tezgaah Paneli** . İstenilen her bir 'I/O' cihaz tipi sayısı menü kullanılarak ayarlanabilir. **Yapılandır | 'I/O' Cihazlar** .

Her 'I/O' cihaz, bir veya daha fazla pin ataşmanını a. **İki sayı** pin numarası (00, 01, 02,... 10,11,12, 13 ve A0-A5 veya ondan sonra 14-19) karşılık gelen bir düzenleme kutusunda. 2'den 9'a kadar olan pin sayıları için tek bir sayı girebilirsiniz - öncü 0 otomatik olarak sağlanacaktır, ancak pins 0 ve 1 için önce öncü 0'a girmelisiniz. Girişler *normalde* 'I/O' cihaz'ın sol tarafında *normalde* sağda ( *alan izin* ). Tüm 'I/O' cihazlar pin seviyelerine ve pin seviyesindeki değişikliklere doğrudan cevap verecek, böylece pins'e bağlı pins kütüphanesine veya programlanmış'e yönelik kütüphane '**digitalWrite()**' ("bit-çarptım" işlemi için).

Birden fazla cihazlar'yi aynı ATmega pin'e bağlayamazsanız, **elektrik as in**

**electrical pin-driving conflict**), **çakışma**. Böyle bir as in electrical pin-driving conflict), çakışma ya bir 'Uno' veya 'Mega' pin tarafından oluşturulabilir. ' **ОУТРУТ** ' cihaz'e bağlı güçlü iletkenliğe (düşük empedanslı) karşı sürüş (örneğin, bir 'FUNCEN' çıkışına veya bir 'MOTOR'a karşı sürüş) **Enç** çıkışı) veya birbiriyle rekabet eden iki bağlı cihazlar ile (örneğin, aynı pin'e bağlı bir 'PULSER' ve bir 'PUSH' - düğmesi). Bu tür as in electrical pin-driving conflict), çakışma'lar gerçek bir donanım uygulamasında felaket olur ve buna izin verilmez ve kullanıcıya açılır bir mesaj kutusu aracılığıyla işaretlenir).

İletişim kutusu kullanıcının istenen 'I/O' cihazlar'nin tiplerini ve numaralarını seçmesine izin vermek için kullanılabilir. Bu iletişim kutusundan **Kaydet** 'I/O' cihazlar, dosya metnine ve / veya **Doldurmak** 'I/O' cihazlar daha önce kaydedilmiş (veya düzenlenmiş) bir metinden dosya ( **tüm pin bağlantılarını ve tıklanabilir ayarları ve girilen herhangi bir düzenleme kutusu değerlerini içeren** ).

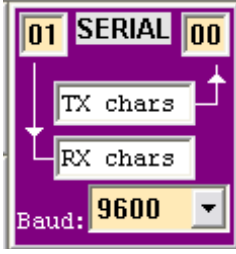
**Sürüm 1.6'dan itibaren, ilgili IO cihazlar'deki periyot, gecikme ve darbe genişliği düzenleme kutularındaki değerlere 'S' (veya 's') ekinin verilebileceğini unutmayın.** Bu olması gerektiğini gösterir **pullu** küresel bir pozisyona göre 'I/O \_\_\_\_S' Ana pencere'te görünen kaydırıcı kontrolü **Aracı-Bar**. Tamamen sağa kaydırıcı ile ölçek faktörü 1,0'dır (birlik), ve sürgü tamamen sola doğru iken ölçek faktörü 0.0'dır (her özel 'I/O' cihaz tarafından uygulanan minimum değerlere tabi). Birden fazla düzenleme kutusu değerini ölçekleyebilirsiniz **eşzamanlı** bu kaydırıcıyı kullanarak. Bu özellik yürütülürken sürgüyü sürüklemenizi sağlar ekli 'I/O' cihazlar'ye göre değişen darbe genişliklerini, periyotlarını ve gecikmelerini kolayca taklit etmek.

Bu bölümün geri kalanı, her bir cihaz türü için açıklamalar sunar.

Bunlardan bazıları cihazlar **girilen değerlerin ölçeklendirilmesini destekleme** ana pencere üzerindeki sürgüyü kullanarak **Aracı-Bar**. cihaz değeri, eki olarak 'S' harfine sahipse, değeri, cihaz minimum değer sınırlamasına tabi olarak kaydırıcı-baş pozisyonuyla belirlenen bir ölçek faktörü (0.0 ile 1.0 arasında) ile çarpılacaktır. (1.0 tamamen sağa doğru, 0.0 tamamen sola doğru) Aşağıda tarif edilen cihazlar hortumunun altındaki 'I/O \_\_\_\_S'.



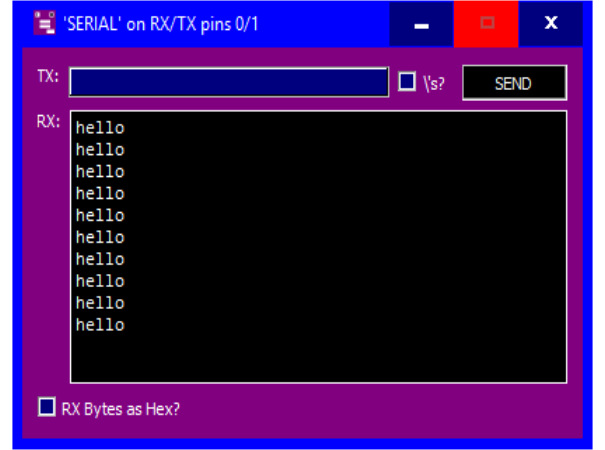
## 'Serial' Monitör ('SERIAL')



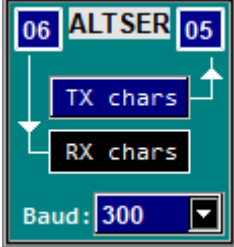
Bu 'I/O' cihaz, 'Uno' veya 'Mega' pins 0 ve 1'de ATmega donanım aracılı seri giriş ve çıkışa ('Uno' veya 'Mega' USB yongası aracılığıyla) izin verir. baud hızı alt kısımdaki açılır liste kullanılarak ayarlanır - seçilen baud hızı **eşleşmek zorunda** program'nizin geçtiği değer '`Serial.begin()`' Doğru iletim / alım için işlev. *Seri haberleşme 8 veri bitinde, 1 stop bitinde ve parite bitinde sabitlenir.* Yapmana izin var **kesmek** (boş) **ama yerine** TX pin 00, fakat RX pin 01 değildir.

program'nize klavye girişi göndermek için, üstteki bir veya daha fazla karakter yazın (TX karakterleri) pencere ı vur **'Enter'** klavye tuşu . (iletimlerin başladığını belirtmek için karakterler italik hale gelir) - ya da zaten ilerliyorsa, eklenen yazılan karakterler italik olacaktır. Daha sonra kullanabilirsiniz '`Serial.available()`' ve '`Serial.read()`' işlevler, pin 0 tamponuna alındıkları sıradaki karakterleri okumak için (önce en soldaki karakter ilk gönderilecektir). Biçimlendirilmiş metinsel ve sayısal çıktılar veya biçimlendirilmemiş bayt değerleri, Arduino çağırılarak alt konsol çıktısına (RX karakterleri) pencere gönderilebilir '`print()`', '`println()`' veya '`write()`' işlevler.

Bunlara ek olarak, **TX ve RX karakterlerini ayarlamak / görüntülemek için daha büyük bir pencere bu tuşa çift tıklayarak (veya sağ tıklayarak) açılabilir 'SERIAL' cihaz** . Bu yeni pencere'ün daha büyük bir TX karakter düzenleme kutusu ve TX karakterlerini 'Uno' veya 'Mega'ya göndermek için tıklanabilecek ayrı bir 'Send' düğmesi var (pin 0'da). Gibi ters eğik çizgi kaçan karakter dizileri yeniden yorumlamak için bir onay kutusu seçeneği de vardır '`\n`' veya '`\t`' ham olmayan ekran için.



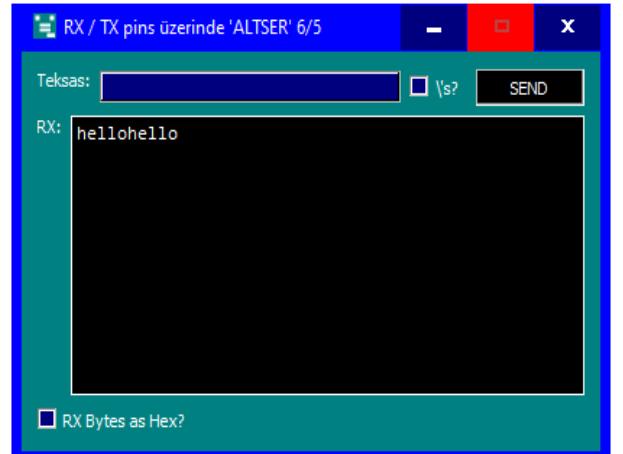
## Alternatif Seri ('ALTSER')



Bu 'I/O' cihaz, doldurmayı seçtiğiniz herhangi bir 'Uno' veya 'Mega' pins çifti üzerinde kütüphane yazılımı aracılı veya alternatif olarak, kullanıcının "bit çarpması", seri giriş ve çıkışına izin verir ( **dışında** Donanıma adanmış pins 0 ve 1 '`Serial`' iletişim). program'niz bir '`#include <SoftwareSerial.h>`' Kitaplığın işlevselliğini kullanmak istiyorsanız, üstte yakın bir çizgi seçin. Donanım tabanlı 'SERIAL' cihaz'te olduğu gibi, 'ALTSER' için baud hızı altındaki listeyi kullanarak ayarlanır - seçilen baud hızı program'nizin geçtiği değerle eşleşmelidir '`begin()`' Doğru iletim / alım için işlev. *Seri haberleşme 8 veri bitinde, 1 stop bitinde ve parite bitinde sabitlenir.*

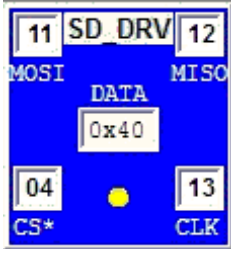
Ayrıca, donanım tabanlı olduğu gibi '`SERIAL`' , **TX ve RX için daha büyük bir pencere ayarı / görüntüleme ALTSER cihaz'e çift tıklayarak (veya sağ tıklayarak) açılabilir** .

Donanım uygulamasından farklı olarak '`Serial`' , sağlanan yok Dahili ATmega kesme işlemleri tarafından desteklenen TX tamponu (yalnızca bir RX tamponu), o '`write()`' (veya '`print`') çağrıları engelliyor (yani, program'niz tamamlanıncaya kadar devam etmeyecek).



## Katı Hal Diski ('SD\_DRV')

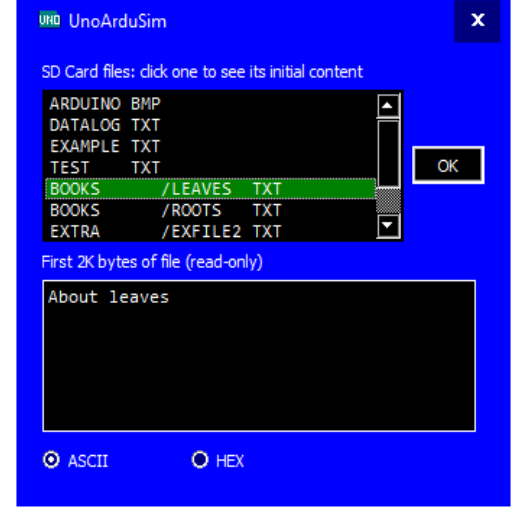
Bu 'I/O' cihaz, kütüphane yazılım aracılı (ancak **değil** "bit-çarptım") 'I/O'te dosya giriş ve çıkış işlemleri **SPI** pins (hangisini seçebilirsiniz **CS \*** pin kullanacaksınız). program'niz kolayca yapabilirsiniz '#include <SD.h>' üste yakın bir çizgi ve kullanabilirsiniz '<SD.h>' işlevler VEYA doğrudan çağrı '**SdFile**' işlevler kendin.



**'SD\_DRV' cihaz'te dizinleri ve dosyalar (ve içeriği) görüntüleyen daha büyük bir pencere çift tıklayarak (veya sağ tıklatarak) açılabilir .**

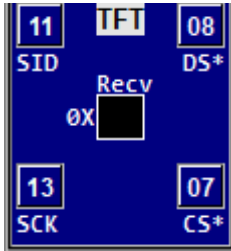
Tüm disk içeriği **-dan yüklendi** bir **SD** yüklenen program dizinindeki alt dizin (varsa) '**SdVolume::init()**', **ve yansıtılır** aynı **SD** dosya'deki alt dizin '**close()**', '**remove()**', ve üzerinde '**makeDir()**' ve '**rmDir()**' .

SPI transferleri sırasında sarı bir LED yanıp söner ve 'DATA' en son 'SD\_DRV'ü gösterir. **tepkî** bayt. Tüm SPI sinyalleri doğrudur ve **E signal shape**), **dalgabiçimi pencere**.



## TFT Ekran ('TFT')

Bu 'I/O' cihaz bir Adafruit öykünür™ kendi doğal dönme = 0 boyutu 1280by-160 piksel TFT (ama 'TFT.h' kütüphanesi kullanıldığı zaman '**TFT.begin()**' 160'a 128 piksel "yatay" görünüm) verir dönme = 1 için başlatma ayarlar. Sen işlevler çağırarak bu cihaz iletmek edebilirsiniz '**TFT.h**' (İlk IL-16'yı gerektirir) kütüphane veya iletmek bunun için size bayt kendi dizisini göndermek için SPI sistemi kullanabilirsiniz.

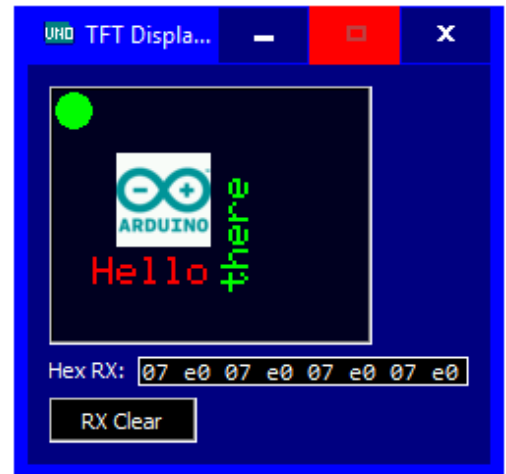


TFT hep bağlanır 'SPI' pins ('SID' için) 'MOSI' ve ('SCK' için) 'SCK' göre - bu değiştirilemez. 'DS\*' pin içindir veri / komut ('LOW' seçer veri modu) seçin ve 'CS\*' pin aktif-düşük yonga seçme olduğu

Eğer bir donanım sıfırlaması yapamaz hiçbir Reset pin sağlanan yoktur Bir pin düşük sürüş Bu cihaz (şekilde '**TFT::begin()**' Geçerli bir 'reset' pin numarasını geçtikten sonra işlev yapmak için çalışır üçüncü parametre olarak

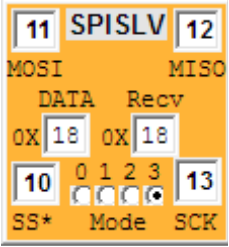
'**TFT(int cs, int ds, int rst)**' Yapıcı). o kendisini her zaman sıfırlar böylece cihaz ancak ana UnoArduSim tıklayın sistem Reset hattına bir gizli bağlantı var mı Reset araç çubuğu simgesi veya 'Uno' veya 'Mega' kartı sıfırlama düğmesi ..

Tarafından **çift tıklayarak** (veya **sağ tıklayarak** aşağıda gösterildiği gibi), bu cihaz ile, daha büyük bir pencere) (en son alınan 8 bayt ile birlikte, bu tam 160'a 128 piksel LCD ekran göstermek için açılır



## Yapılandırılabilir SPI Bağımlı ('SPISLV')

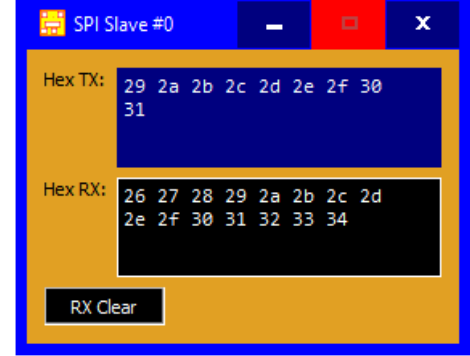
Bu 'I/O' cihaz, aktif modu düşük olan seçilmiş modda bir SPI bağımlısı öykünür **SS \*** ("slave-select") pin'ü **MISO** pin çıkışı (ne zaman **SS \*** yüksektir, **MISO** itilmiş değildir). program'niz bir `#include <SPI.h>` dahili SPI Arduino includes 'struct' and 'class'), nesne ve kütüphanesinin işlevselliğini kullanmak istiyorsanız Alternatif olarak, kendi "bit-çarptınız" ifadesini yaratmayı seçebilirsiniz. **MOSI** ve **SCK** bu cihaz'e iletmek'e



cihaz, kenar geçişlerini algılıyor **CLK** Seçilen moda göre giriş '**MODE0**', '**MODE1**', '**MODE2**' veya '**MODE3**', program'nizin programlanmış SPI moduyla eşleşmesi için seçilmesi gerekir.

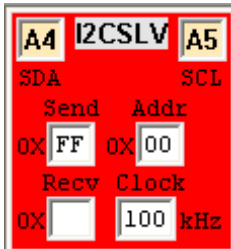
**cihaz'i çift tıklatarak (veya sağ tıklatarak), daha büyük bir pencere arkadaş açabilirsiniz.** bunun yerine izin verir y 32 bayt-maksimum tamponu doldurmak (verilerini otomatik olarak döndüren SPI

cihazlar'yi taklit etmek için) ve alınan son 32 baytı (onaltılık çiftler olarak) görmek için. **Bunu not et** sonraki TX tampon baytı otomatik olarak yalnızca 'DATA'e gönderilir **sonra** Dolu '**SPI.transfer()**' tamamlandı!



## İki Telli I2C Bağımlı ('I2CSLV')

Bu 'I/O' cihaz yalnızca *Köle modu* cihaz. cihaz'e 'Addr' düzenleme kutusunda iki-hex-sayı girişi kullanılarak bir I2C veri yolu adresi atanabilir (yalnızca I2C atanmış adresini içeren otobüs işlemleri). cihaz, açık tahliyesi hakkında veri gönderiyor ve alıyor (yalnızca aşağı açılır) **SDA** pin ve açık tahliyesindeki veriyolu saatine yanıt veriyor (yalnızca aşağıya çekerek) **SCL** pin. 'Uno' veya 'Mega'ün üretilmesinden sorumlu otobüs yöneticisi olacağı halde **SCL** sinyal, bu köle cihaz de çeker **SCL** otobüsün düşük zamanını iç hızına uygun olana (gerekirse 'Clock' düzenleme kutusunda ayarlanabilir) uzatmak için (gerekirse) düşük fazı boyunca düşüktür.

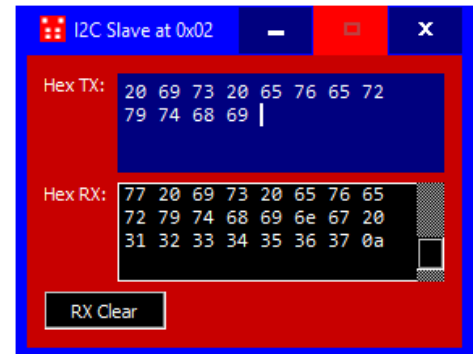


program'niz bir `#include <Wire.h>` hattının işlevselliğini kullanmak istiyorsanız '**TwoWire**'. Bu cihaz ile etkileşim kütüphane. Alternatif olarak, bu çarpımlı cihaz'e ait iletmek'e kendi bit-çarpımlı verilerinizi ve saat sinyallerinizi oluşturmayı seçebilirsiniz.

'Uno' veya 'Mega' master'a geri gönderim için tek bir bayt 'Send' düzenleme kutusuna ayarlanabilir ve tek bir (en son alınan) bayt (salt okunur) 'Recv' düzenleme kutusunda görüntülenebilir. **Bunu not et** 'Send' düzenleme kutusu değeri her zaman **Sonraki** Bu cihaz dahili veri tamponundan aktarım için bayt.

**cihaz'i çift tıklatarak (veya sağ tıklatarak), daha büyük bir pencere arkadaş açabilirsiniz.** bunun yerine, maksimum 32 baytlık bir FIFO

arabelleğini doldurmanıza (TWI cihazlar'yi bu tür işlevlerle taklit etmek için) ve en son alınan verinin (en fazla 32'ye kadar) baytını görüntülemenize izin verir. satır başına 8 baytlık hex-sayı gösterimi). Bu iki düzenleme kutusundaki satır sayısı, seçilen TWI arabellek boyutuna karşılık gelir (bunlar kullanılarak seçilebilir) **Yapılandır | Tercihler** ). Bu Arduino'dan beri bir seçenek olarak eklendi. '**Wire.h**' kütüphane kullanır **beş** Bu tür RAM, RAM belleğinin pahalı olduğu uygulama kodunda tamponlar. Arduino kurulumunu düzenleyerek '**Wire.h**' dosya tanımlanmış sabiti değiştirmek için '**BUFFER\_LENGTH**' (ve aynı zamanda arkadaş düzenleme '**utility/twi.h**' dosya değışecek TWI tampon uzunluğu) hem kullanıcı hem de 16 veya 8 olacak şekilde *could* RAM bellek yükünü önemli ölçüde azaltmak hedeflenen 'Uno' veya 'Mega' **donanım uygulaması** - UnoArduSim bu nedenle gerçek dünyadaki bu olasılığı yansıtıyor **Yapılandır | Tercihler** .





### Metin LCD I2C ('LCDI2C')

Bu 'I/O' cihaz bir 1,2, o4 4-hattı taklit karakter LCD, üç moddan birinde:

a) sırt çantaşı 0 yazın (ile Adafruit tarzı liman genişletici donanım I2C otobüsü adresi 0x20-0x27 sahip)

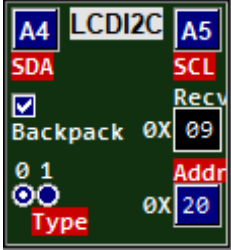
Donanım sahip

b) sırt tipi 1 (DFRobot tarzı noktası genişletici I2C bus adresi 0x20-0x27)

c) sırt çantaşı (Ana mod birleşik I2C ara yüzün I2C Bus adresi 0x3C-0x3F)

Her cihaz modu için kütüphane kod Destekleme UnoArduSIm yükleme dizininin 'include\_3rdParty' klasörünün içindeki sağlanmış: 'Adafruit\_LiquidCrystal.h' , 'DFRobot\_LiquidCrystal.h' , ve

'Native\_LiquidCrystal.h' , sırasıyla.



cihaz onun 'Addr' edit-kutusunda iki altıgen-sayı girişini (sadece cevap verecektir kullanarak herhangi I2C veriyolu adresi atanabilir I2C atandığı adresini içeren otobüs işlemleri). cihaz (ACK = 0 veya NAK = 1, ve yanıt verir) otobüs adres ve veri aldığı açık-drenaj (açılan okunur) SDA pin. Yalnızca yazma LCD komutları ve DDRAM verilerinizi - Sen **olumsuz** Yazılı DDRAM konumlarından verileri geri okumayı ..



**Çift tıklama** veya **sağ tık** Ayrıca ekran boyutu ve karakter kümesini ayarlayabileceğiniz LCD ekran monitör pencere açın.

### Metin LCD SPI ('LCDSPI')

Bu 'I/O' cihaz 1,2, o4 4-hattı taklit karakter LCD, iki moddan birinde:

a) sırt çantaşı (Adafruit tarzı SPI portu genişletici)

b) sırt çantaşı (Ana modu SPI entegre arayüzü - aşağıda gösterildiği gibi)

Her cihaz modu için kütüphane kod Destekleme UnoArduSIm yükleme dizininin 'include\_3rdParty' klasörünün içindeki sağlanmış: 'Adafruit\_LiquidCrystal.h' , ve 'Native\_LiquidCrystal.h' , sırasıyla.



Pin 'SID' 'SCK' saat pin olduğunu 'SS' aktif-düşük cihaz-seçmektir, seri verileri ve 'RS' veri / komut pin olduğunu. Yalnızca yazma LCD komutları ve DDRAM verilerini (tüm SPI işlemleri yazıyor vardır) can - Eğer **olumsuz** Yazılı DDRAM konumlardan veri geri okuyun.



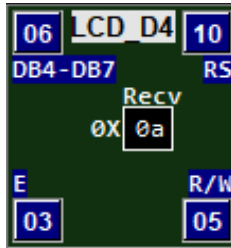
**Çift tıklama** veya **sağ tık** Ayrıca ekran boyutu ve karakter kümesini ayarlayabileceğiniz LCD ekran

monitör pencere açın.

### Metin LCD D4 ('LCD\_D4')

Bu 'I/O' cihaz 1,2, o4 4-hattı taklit 4-bit paralel veri yolu arayüzüne sahip karakter LCD. Veri bayt okumak / yazılır **iki yarım** onun 4 verilere pins 'DB4-DB7' (burada düzenleme kutusu içeren **düşük olan birbirini takip eden 4 pin numaralarının sayılı** ), - veri 'E' düşen kenarlarında zamanlanmaktadır ('R/W' pin tarafından kontrol edilen veri yönü ve LCD veri / 'RS' pin komut modu ile) pin, sağlar.

kütüphane kod Destekleme içeride sağlanmış: 'include\_3rdParty' senin UnoArduSIm kurulum dizininin klasör: 'Adafruit\_LiquidCrystal.h' , , ve 'Native\_LiquidCrystal.h' Her iki eser.



**Çift tıklama** veya **sağ tık** Ayrıca ekran boyutu ve karakter kümesini ayarlayabileceğiniz LCD ekran monitör pencere açın.





## Çoklayıcı LED I2C ('MUXI2C')

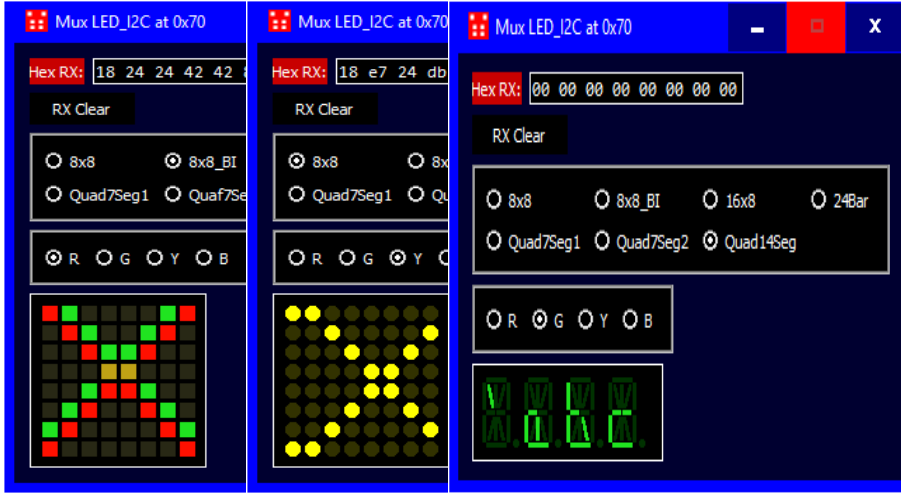


Bu 'I/O' cihaz I2C arayüzlü HT16K33 denetleyicisi taklit (having I2C bus adresi 0x70-0x77) neye çoklanmış LED ekranların birkaç farklı türde bir bağlı olabilir:

- a) 8x8 veya 16x8 LED dizin
- b) 8x8 çift renkli LED dizin
- c) 24-iki renkli-LED çubuk
- d) 4-sayı 7 kademeli görüntüler iki stilleri
- e) bir 4-sayı 14 kademeli bir sayısal gösterge

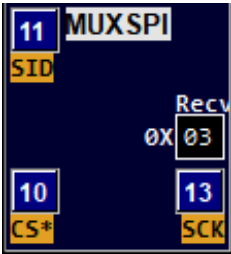
Hepsi tarafından desteklenen 'Adafruit\_LEDBackpack.h' Kod 'include\_3rdParty' içerde sağlanan Klasör:

Çift tıklama (Veya sağ tıklama) Daha büyük pencere açmak için seçim ve görüntülemek için çeşitli renkli LED bir görüntüler.

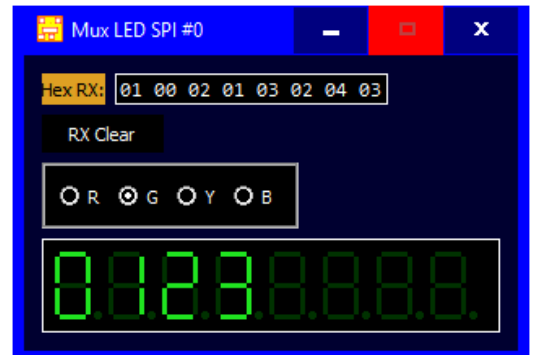


## Çoklayıcı LED SPI ('MUXSPI')

Bir çoklanmış-LED kontrol destekleyen, MAX6219 göre 'MAX7219.h' Kod sekiz 7 segmentli basamak iletmek kadar 'include\_3rdParty' klasörünün içindeki sağladı.



Çift tıklama (Veya sağ tıklama) Daha büyük pencere açmak için görüntülemek için renkli 8-sayı 7-Segment ekran.



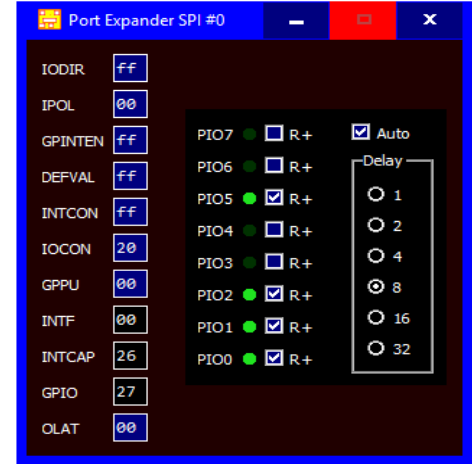
## Genişleme Bağlantı Noktası SPI ('EXPSPi')



Bir 8-bit'lik liman genişletici destekleyen, MCP23008 göre 'MCP23008.h' kod 'include\_3rdParty' içine sağlanan Klasör. Sen MCP23008 kayıt için yazıyorum ve GPIO pin geri okuyabilir seviyeleri. Kesmeler her GPIO pin değişikliği etkinleştirilebilir - Bir tetiklenen kesme 'INT' pin iletmek edecektir.

Çift tıklama (Veya sağ tıklama) açmak **Daha büyük pencere Görmek** 8 GPIO noktası hatları ve ekli pull-up dirençler. Sen tıklatarak el barfiks değiştirebilir veya periyodik bir yukarı-sayımı şekilde onları değiştirecek bir sayaç ekleyebilirsiniz. sayısı artar küçültme gecikme ile tespit edildiği oranı kullanıcı tarafından seçilen faktörü

(bir artış yaklaşık olarak her 30 milisaniye için 1x faktörü karşılık, daha yüksek gecikme faktörleri daha yavaş bir yukarı-sayım oran vermek)

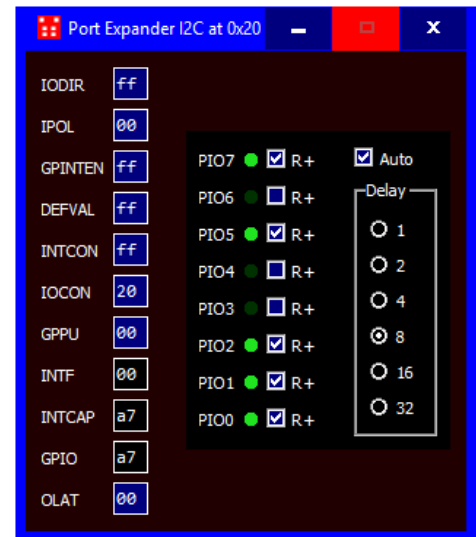


## Genişleme Bağlantı Noktası I2C ('EXPI2C')

Bir 8-bit'lik liman genişletici destekleyen, MCP23008 göre 'MCP23008.h' Kod-16'nın içine sağlanan Klasör. Yetenekleri 'EXPSPi' cihaz maç.

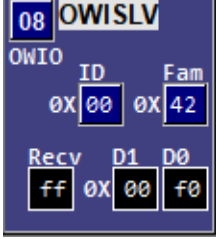


Çift tıklama (Veya sağ tıklama) Daha büyük pencere açmak için 'EXPSPi' cihaz ileri geri olarak.



## '1-Wire' Bağımlı ('OWIISLV')

Bu 'I/O' cihaz, pin OWIO'ya bağlı küçük bir '1-Wire' veri yolu cihazlar kümesinden birini taklit eder. Seçtiğiniz 'Uno' veya 'Mega' pin üzerinde bir '1-Wire' veri yolu oluşturabilirsiniz (bu bağımlılardan biri veya daha fazlası ile '1-Wire' cihazlar). Bu cihaz kabini arayarak kullanılabilir. 'OneWire.h' kütüphane yerleştirdikten sonra işlevler '#include <OneWire.h>' program'nizin üstündeki çizgi. Alternatif olarak, bu cihaz'e OWIO'da bit-çarpılmış sinyalleri de kullanabilirsiniz (elektriksel as in electrical pin-driving conflict), çakışma'a neden olmadan düzgün şekilde yapmak çok zor olsa da - böyle bir as in electrical pin-driving conflict), çakışma'u kullanırken bile mümkündür 'OneWire.h' işlevler, fakat bu çatışmalar UnoArduSim'de rapor edilmiştir).

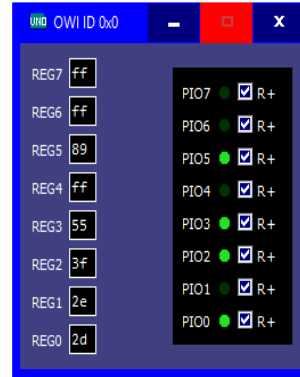


Her gerçek dünya OWISLV cihaz'in benzersiz bir 8-byte i (64-bit!) Olması gerekir Dahili seri numarası - UnoArduSim'de bu, kısa bir 1-byte onaltılı sağlayan kullanıcı tarafından basitleştirilmiştir. 'ID' değer (varsayılan olarak cihaz yük / toplama işleminde varsayılan olarak atanan), artı 'Fam' Bunun için aile kodu cihaz. UnoArduSim, sıcaklık sensörünü kapsayan V2.3 (0x28, 0x29, 0x3A, 0x42) ve küçük bir Aile kodları grubunu tanıır ve paralel IO (PIO) cihazlar (tanınmayan bir Aile kodu, cihaz'i genel 8 baytlı bir karalama defteri haline getirir Genel sensör ile cihaz.

cihaz Ailesinin PIO kaydı yoksa, kayıtlar **D0** ve **D1** temsil etmek ilk iki karalama defteri baytı, aksi takdirde PIO'yu temsil eder Sırasıyla "status" (gerçek pin seviyeleri) kaydı ve PIO pin mandalı veri kaydı.

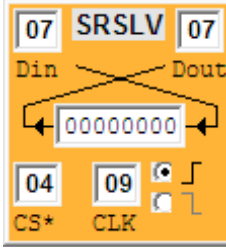
Tarafından **çift tıklayarak** (veya **sağ tıklayarak**) cihaz'te, daha büyük **OWIMonitor** pencere açıldı. Bu daha büyük pencere'ten tüm cihaz kayıtlarını inceleyebilir, düzenlemeleri ve kaydırıcıyı kullanarak scratchpad konumlarını SCR0 ve SCR1 olarak değiştirebilirsiniz (yine, SCR0 ve SCR1 yalnızca **D0** ve **D1** PIO yoksa) veya harici pin PIO pull-up'larını ayarlayın. SCR0 ve SCR1 düzenlendiğinde, UnoArduSim, bu düzenlenmiş değerleri cihaz'ten imzalanmış değer çıktısını temsil eden bir başlangıç (Reset'den başlayarak) değerini temsil eden kullanıcı "tercihi" olarak hatırlar; Sürgü % 100'e sıfırlanır (ölçek ölçeği 1.0) düzenleme zamanında. Slider daha sonra taşındığında, '**signed**' SCR1'deki değer, sürgü konumuna göre ölçeklenir (ölçek faktörü 1,0'dan 0,0'a kadar) - özelliği, program'nizin sensör değerlerini sorunsuz şekilde değiştirmedeki tepkisini kolayca test etmenizi sağlar. .

PIO pins içeren cihaz için, pin seviyeli onay kutularını ayarladığınızda, UnoArduSim bu kontrol edilmiş değerleri hatırlar. mevcut pull-up'lar pins'e harici olarak uygulanır. Bu harici çekme değerleri daha sonra pin mandal verileri ile birlikte kullanılır (kayıt **D1**) nihai gerçek pin seviyelerini belirlemek ve PIO pin'e bağlı yeşil LED'ı hafifletmek veya söndürmek için (pin sadece gider '**HIGH**' harici bir çekme yapılırsa, **ve** karşılık gelen **D1** mandal bit bir '**1**').



## Kayan Yazmaç Bağımlı ('SRSLV')

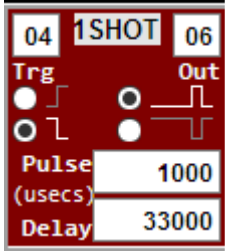
Bu 'I/O' cihaz, aktif bir düşük değere sahip basit bir shift-register cihaz'e öykünür **SS \*** ("slave-select") pin'ü **'Dout'** pin çıkışı (ne zaman **SS \*** yüksektir, **'Dout'** itilmiş değildir). program'nız dahili SPI Arduino includes 'struct' and 'class'), nesne ve kütüphanesinin işlevselliğini kullanabilir. Alternatif olarak, kendi "bit-çarpıtınız" ifadesini yaratmayı seçebilirsiniz. **'Din'** ve **CLK** bu cihaz'e iletmek'e



cihaz, kenar geçişlerini algılıyor **CLK** Yazmacının kaymasını tetikleyen girdi - algılanan kutupsallık **CLK** kenar bir radyo düğmesi kontrolü kullanılarak seçilebilir. Her birinde **CLK** kenar (algılanan kutupluluğun), yazmaç yakalar **gürültü** Kalan bitler aynı anda bir pozisyon MSB pozisyonuna doğru sola kaydırıldığı için kaydırma yazmacının en az anlamlı bit (LSB) pozisyonuna getirin. Her ne zaman **SS \*** düşük olduğunda, kaydırma yazmacının MSB konumundaki mevcut değer, üzerine itilmiş'dur. **'Dout'**.

## Tek Çekimli ('1SHOT')

Bu 'I/O' cihaz, 'Out'ünde seçilen kutupsallığa ve darbe genişliğine sahip bir darbe oluşturabilen bir dijital'e tek seferlik. pin, kendisine alınan tetikleyici bir kenardan belirli bir gecikmeden sonra meydana geliyor **Trg** (tetik) pin girişi. Belirtilen tetikleyici kenar alındıktan sonra, zamanlama başlar ve ardından 'Out'e kadar yeni bir tetikleme darbesi tanınmaz. darbe üretildi (ve tamamen bitti).



Bu cihaz'ın olası bir kullanımı Bir tetikleyici darbeye yanıt olarak bir aralık darbesi üreten ultrason aralığı sensörleri. program'nizin yarattığı bir pin çıkış sinyaline senkronize edilmiş bir pin giriş sinyali (seçtiğiniz gecikmeden sonra) üretmek istediğiniz yerde de kullanılabilir. **'Pulse'** ve **'Delay'** değerleri Ana pencere'ten ölçeklendirilebilir **Aracı-Bar** 'I/O \_\_\_\_S' son ekleyerek ölçek faktörü kaymak kontrolü 'S' (veya 's') ile ikisinden birine (veya her ikisine).

Bu cihaz'ın başka bir kullanımı, kesintileri kullanan bir program'yi test etmektir ve bir durumda ne olacağını görmek istersiniz. **özel program talimatı** kesintiye uğradı. pin 2'ye (veya pin 3)

bağladığınız 'I/O' Cihaz'ı geçici olarak çıkarın ve 'Out' pin'ü 'Uno' veya 'Mega' pin 2'ye (veya sırasıyla pin3'e bağlı olan) bir '1SHOT' cihaz ile değiştirin, ardından 'Trg' girişini tetikleyebilirsiniz. (yükselen kenar hassasiyetinin oraya ayarlandığını varsayarak) yerleştirerek komut çifti { **'digitalWrite (LOW)'** , **'digitalWrite (HIGH)'** } **hemen önce** Kesintinin gerçekleşmesini istediğiniz talimatın içinde. 1SHOT; s'Delay'yi ayarlayın 'Out'de üretilen nabzın, bu tetikleme talimat çiftini takip eden program talimatı içinde gerçekleşmesi için. Bazı talimat maskelerinin kesintiye uğradığını unutmayın (örneğin **'SoftwareSerial.write (byte)'** , and böylece Kesilemez

## Programlanabilir 'I/O' Cihaz ('PROGIO')



Bu 'I/O' cihaz aslında bir 'I/O' cihaz taklit etmek için program (ayrı bir program ile) yapabileceğiniz çıplak bir 'Uno' kartı'dır davranışını tamamen tanımlayabilirsiniz. Bu köle 'Uno'un, ortasında beliren master (main' Uno ' veya 'Mega') ile ortak olarak paylaşacağı dört adede kadar pins (IO1, IO2, IO3 ve IO4) seçebilirsiniz. **Laboratuvar Tezgah Paneli** . Diğer cihazlar'te olduğu gibi, bu 'Uno' slave ile ana kartı arasındaki herhangi bir elektrikli as in electrical pin-driving conflict), çakışma tespit edilecek ve işaretlenecektir. Tüm bağlantıların **direkt olarak** kablolu, **dışında pin 13** (burada Reset'de elektriksel bir as in electrical pin-driving conflict), çakışma'yi önlemek için iki pins arasında bir dizi R-1K direnci varsayılır). V2.8 itibariyle, master ve slave pins arasındaki

bağlantılar **eşlenen** : eğer master da bir 'Uno' ise, bu **varsayılan** haritalama bir kimlik (pin 1'in pin 0 ile eşleşmesi ve tam tersi 'Serial' iletişimine izin vermek için tersi hariç); eğer master bir 'Mega' ise, pins 1 ve 0 tekrar çevrilir, **ve tüm** SPI ve TWI pins, karşılık gelen ana ve bağımlı alt sistemler arasında doğrudan bağlantı için eşlenmiştir. Bu **varsayılan** haritalama olabilir **Geçersiz kılınan** belirterek **IODevs.txt** dosya, PROGIO program dosya adını izleyen 4 ana pin numarasının açık bir listesi - bu değerler -1 ise, varsayılan eşleme ile aynıdır. Bu cihaz için yazılan pin numaraları **köle 'Uno' köylüleridir**. Soldaki resim, SPI sistemi pins'e (**SS \***, **MISO**, **MOSI**, **SCK**) belirtilen 4 köle pins'ü göstermektedir - ustasının 'Uno' veya 'Mega' olup olmadığına bakılmaksızın, bu köleyi program'e davranışını programsal olarak tanımlayabileceğiniz genel SPI slave (veya master).

Tarafından **çift tıklayarak** (veya **sağ tıklayarak** ) bu cihaz'te, bu 'Uno'in kölesinin kendi **Kodlama Paneli** ve ilişkili **Değişken Paneli** , tıpkı de olduğu gibi. Ayrıca kendi **Aracı-Bar** , . Hangi için kullanabilirsiniz **yük** ve **icra kontrolü** bir

köle program - ikon hareketleri Ana pencere ile aynı klavye kısayollarına sahiptir. ( **Doldurmak** olduğu **Ctrl-L** , **Kaydet** olduğu **Ctrl-S** vb.). Yüklendikten sonra, çalıştır'ten **ya** Ana UnoArduSim pencere veya bu Bağımlı'nın içinden pencere Monitörü - Her iki durumda da Ana program ve Bağımlı program, yürütmeleri ilerledikçe gerçek zamanlı geçişle senkronize olarak kilitli kalır. **Sahip olacak Kodlama Paneli'yi seçmek yük, arama ve icra odağı , tık üzerinde üst pencere başlık çubuğu - odaklanmayan Kodlama Paneli'nin araç çubuğu vardır. grileşmiş eylemler .**

Köle cihazlar için bu 'PROGIO' cihaz'e programlanmış olabilecek bazı olası fikirler aşağıda listelenmiştir. Seri, I2C veya SPI cihaz emülasyonu için, sırasıyla sırasıyla Gönderme ve Alma tamponları için diziler ile kodlama yapan uygun program'yi kullanabilirsiniz. karmaşık cihaz davranışını taklit etmek:

a) Bir SPI yöneticisi veya köle cihaz. UnoArduSimV2.4. genişletildi 'SPI.h' isteğe bağlı olarak S {PI işlemine izin veren köle kipine izin veren kütüphane 'mode' parametresi 'SPI.begin(int mode = SPI\_MASTR) ' . Açıkça geçmek 'SPI\_SIV' köle modunu seçmek için (varsayılan ana moda dayanmak yerine). Artık bir kullanıcı kesmesini tanımlayabilirsiniz: işlev (haydi arayalım 'onSPI' ) ya de, çağrı yaparak baytları aktarmak için başka bir eklenti eklendi 'SPI.attachInterrupt(user\_onSPI) ' . **Şimdi c** Alling 'rxbyte=SPI.transfer(tx\_byte) ' içinden 'user\_onSPI' işlev kesme bayrağını temizler ve dönüş **hemen** değişken'inizdeki yeni alınan bayt ile 'rxbyte' . Alternatif olarak, takma işleminden kaçınabilirsiniz. bir SPI kesintisi ve bunun yerine sadece arayın 'rxbyte=SPI.transfer(tx\_byte) ' ana program'nizin içinden - bu çağrı **icra bloğu** SPI baytı aktarılanaya kadar, ve olacak o zaman dönüş yeni alınan bayt içinde 'rxbyte' .

b) Genel **seri 'I/O'** cihaz. Master kartı ile aşağıdakilerden birini kullanarak iletişim kurabilirsiniz: 'Serial' veya bir 'SoftwareSerial' köle program- içinde tanımlanmış 'SoftwareSerial' tanımlamalısın 'txpin' ve 'rxpin' zıt Köle, efendinin aktardığı pin'i alır (ve tersi), ancak bunun için **Yalnızca 'Serial' , 1 ve 0 pins zaten sizin için çevrilmiş durumda.**

c) Genel bir 'I2C' master veya slave cihaz. UnoArduSim'in tamamlanması için Bağımlı operasyonu eklendi; 'Wire.h' kütüphane (işlevler 'begin(address) ' , 'onReceive() ' ve 'onRequest' şimdi köle modu işlemlerini desteklemek amacıyla uygulanmıştır).

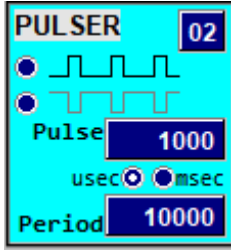
d) Genel bir dijital **Verici** . kullanma 'delayMicroseconds() ' ve 'digitalWrite() ' içeride aramalar 'loop() ' 'PROGIO' program'nizde 'HIGH' ve 'LOW' Bir darbe katarının aralıkları. Ayrı ekleyerek 'delay() ' içinde ara 'setup() ' işlev, bu puls treninin başlangıcını geciktirebilirsiniz. Darbe genişliklerini bile değiştirebilirsiniz Bir sayaç kullanarak zaman ilerledikçe değişken. Ayrıca ayrı kullanabilirsiniz **'IOx'** Taklit edilmiş bir '1Shot'nin zamanlama'sini başlatmak için tetikleyici olarak pin (veya çift atış, uç atış vb.) cihaz ile üretilen darbe genişliğini, zaman ilerledikçe istediğiniz şekilde değiştirmek için kontrol edebilirsiniz.

e) Bir rastgele işaretleyici. Bu üzerinde bir değişiklik dijital **Verici** Ayrıca aramaları kullanır. 'random() ' ve 'delayMicroseconds() ' hangi zamana rastgele saatler üretmek 'digitalWrite() ' Master ile paylaşılan seçilen herhangi bir pin'te bir sinyal. Dördü kullanarak **'IOx'** pins, dört eşzamanlı (ve benzersiz) sinyale izin verir.

Ve Hatırla, **herhangi bir köle kullanabilirsiniz 'Uno' program istediğiniz talimatlar veya alt sistemler .**

## Dijital Darbeci ('PULSER')

Bu 'I/O' cihaz seçilen herhangi bir 'Uno' veya 'Mega' pin uygulanabilir periyodik bir sinyal üretir basit dijital darbe e signal shape), dalgabiçimi jeneratör benzetilmiştir.



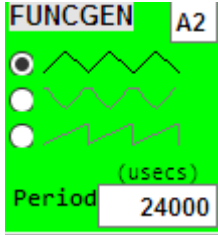
(Mikrosaniye) dönemi ve darbe genişliği 50 mikrosaniye düzenleme kutular-izin verilen minimum süreyi kullanılarak ayarlanır ve minimum darbe 10 mikrosaniyedir genişlik olabilir. Sen mikrosaniye ('usec') ve milisaniye ('msec') 'de zamanlama değerler arasında seçim yapabilir ve Yapılandır gelen 'Save' zaman bu seçim diğer değerlerle birlikte kaydedilir | I / O Cihazlar iletişim.

polarite de seçilebilir: pozitif öncü darbeleri (5V 0) veya negatif öncü darbeleri (0 V 5V) ya.

'Pulse' ve 'Period' değerleri ana gelen ölçeklendirilebilir **Aracı-Bar** ya bir (ya da her ikisi de), bir son ek 'S' (veya 's') halinde eklenerek 'I/O\_\_\_\_S' ölçek faktörü sürgü kontrol. Bu, daha sonra 'Pulse' veya 'Period' değerini değiştirmek için izin verir **dinamik** icra sırasında.

## Analog Fonksiyon Jeneratör ('FUNCGEN')

Bu 'I/O' cihaz, seçilen herhangi bir 'Uno' veya 'Mega' pin'e uygulanabilen periyodik bir sinyal üreten basit bir analog e signal shape), dalgabiçimi jeneratörünü taklit eder.



Dönem (mikrosaniye cinsinden) düzenleme kutusu kullanılarak ayarlanabilir; izin verilen minimum süre 100 mikrosaniyedir. Yarattığı e signal shape), dalgabiçimi sinüzoidal, üçgen veya testere dişi olarak seçilebilir (kare dalga oluşturmak için bunun yerine bir 'PULSER' kullanın). Küçük periyotlarda, üretilen e signal shape), dalgabiçimi'u modellemek için döngü başına daha az numune kullanılır (periyotta sadece 4 örnek = 100 mikrosaniye).

'Period' değeri Ana pencere'ten ölçeklendirilebilir **Aracı-Bar** 'I/O\_\_\_\_S' (veya 's') harfini ek olarak ekleyerek 'I/O\_\_\_\_S' ölçek faktörü kaydırıcı kontrolü.

## Kademe Motoru ('STEPR')

Bu 'I/O' cihaz, dijital dijital kontrolörüne sahip 6V bipolar veya tek kutuplu Step motor taklit eder. **ya iki** (on **P1** , **P2** ) **veya dört** (on **P1** , **P2** , **P3** , **P4** kontrol sinyalleri. Devir başına adım sayısı da ayarlanabilir. Kullanabilirsiniz 'Stepper.h' işlevler 'setSpeed()' ve 'step()' iletmek'e 'STEPR'de. Alternatif olarak, 'STEPR' ayrıca cevap ver kendi kendine 'digitalWrite()' " " çarptım "iletmek sinyalleri.



arasında motor kontrolör çizip

Motor hem mekanik hem de elektriksel olarak doğru bir şekilde modellenmiştir. Motor-kontrolör voltaj düşüşleri ve değişken relüktans ve endüktans, tutma torkuna göre gerçekçi bir atalet momentiyile birlikte modellenmiştir. Motor rotor sargısı, R = 6 ohm değerinde modellenen bir dirence ve 1.0 milisaniyelik elektriksel bir zaman sabiti oluşturan L = 6 milli-Henries endüktansına sahiptir. Gerçekçi modelleme nedeniyle pin darbelerinin çok dar kontrol olduğunu fark edeceksiniz. *alamadım* motor kademeli olarak - hem sınırlı akım yükselme süresi hem de rotor ataletinin etkisi nedeniyle. Bu, tabii ki uygun bir 'Uno' veya 'Mega'ten gerçek bir step motor sürerken neyin gözlemlendiğini kabul eder ( **ve gerekli** ) motor telleri ve 'Uno' veya 'Mega'!

Arduino'da talihsiz bir hata 'Stepper.h' Kütüphane kodu, sıfırlandığında Step motorun Adım pozisyon 1'de (dört adımda) olmayacağı anlamına gelir. Bunun üstesinden gelmek için kullanıcı kullanmalıdır. 'digitalWrite()' onunda 'setup()' pin seviyelerinin kontrolü başlatmak için 'step(1)' 2-pin (0,1) veya 4-pin (1,0,1,0) kontrolüne uygun seviyeler ve motorun 10 milisaniyelik 12 öglen referans başlangıç istenen motor pozisyonuna hareket etmesine izin verin.

V2.6 AS, bu cihaz şimdi, seçebilirler.Ayrıca (zaman kapalı bir veya birden fazla adımlarla eşzamanlı için yeşil veya kırmızı) bir 'sync' LED içerir devir başına adım sayısına ek olarak, iki ilave (gizli) değerleri, isteğe bağlı olarak Örneğin mekanik yüke belirtmek için IODEvs.txt dosya belirtilen, 50, 20, 30 belirler, devir başına 20 adım, bu motor rotorunun

kendisinin 50 kez eylemsiz bir yük momenti ve yüzde 30 bir yük momenti değerleri olarak tam motorlu tutma tork.

Bunu not et **vites küçültme doğrudan desteklenmiyor** Alan yetersizliğinden dolayı, program'nizde bir modulo-N sayıcı değişken uygulayarak taklit edebilirsiniz ve yalnızca 'step()' bu sayaç 0'a çarptığında (vites faktörü N ile azaltılır).

### **Darbeli Kademe Motoru ('PSTEPR')**

Bu 'I/O' cihaz bir 6V emüle **Mikro adım** iki kutuplu entegre bir kontrolör kontrol edilmiş göre olan Kademe Motoru **Bir darbeli 'Step'** pin, bir aktif-düşük 'EN\*' pin ve 'DIR' (yönü) (etkin) pin . devir başına tam adım sayısı da tam aşama (1,2,4,8, ya da 16) her bir mikro-aşama sayısının ile birlikte direkt olarak ayarlanabilir. Bu ayarlara ek olarak, iki ekstra (gizli) değerleri may isteğe Örneğin mekanik yüke belirtmek için IODEvs.txt dosya belirtilen, 50, 4, 20, 30 belirler, devir başına 20 adım, tam adım başına 4 mikro adımları, bu motorun 50 kez eylemsiz bir yük momenti değerleri olarak rotor kendisi ve motor tam tutma momentinin yüzde 30 bir yük momenti.

Kontrol iletmek için kod yazmanız gerekir uygun pins.

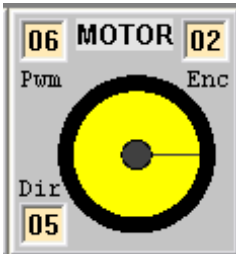


Motor doğru mekanik ve elektriksel olarak, hem modellenmiştir. Motorlu kontrolör voltaj düşüşleri ve değişken manyetik dirençli ve endüktans tork tutma göre atalet gerçekçi bir an birlikte modellenmiştir. sarma motoru rotoru, R = 6 ohm model direnci ve 1.0 milisaniyelik bir elektrik zaman sabiti oluşturur = 6 mili Henries L endüktansı sahiptir.

Bu cihaz (Bir veya daha fazla adımlarla kapalı senkronize için yeşil veya kırmızı) sarı 'STEP' faaliyeti LED ve 'sync' LED içerir.

### **DC Motoru ('MOTOR')**

Bu 'I/O' cihaz, darbe genişlik modülasyonlu bir sinyal tarafından entegre bir kontrolör kontrolörü itilmiş ile 6 voltluk bir besleme 100: 1 dişli DC motor taklit eder **pwm** giriş) ve bir yön kontrol sinyali (üzerinde **dir** giriş). Motorda ayrıca iletir **Enç** pin çıktı. Kullanabilirsiniz 'analogWrite()' iletmek'e **pwm** 490 Hz ile pin (pins 3,9,10,11'de) veya 980 Hz (pins 5,6'da) PWM e signal shape), dalgabiçimi'da 0,0 ile 1,0 arasında görev döngüsü ( 'analogWrite()' 0 - 255 arası değerler). Alternatif olarak, 'MOTOR' ayrıca cevap ver kendi kendine 'digitalWrite()' " çarptım "iletmek sinyalleri.



Motor hem mekanik hem de elektriksel olarak doğru bir şekilde modellenmiştir. Motor-kontrolör transistör voltaj düşüşleri ve gerçekçi yüksüz dişli torku muhasebesi, saniyede yaklaşık 2 devir tam hız verir ve 5 kg cm'nin (yaklaşık sabit bir PWM çalışma çevriminde meydana gelen) yaklaşık 5 kg-cm'nin üzerinde durma torku sağlar. 2,5 kg-cm toplam motor artı yük atalet momenti. Motor rotor sargısı, R = 2 ohm olarak modellenen bir dirence ve 150 mikrosaniyelik bir elektriksel zaman sabiti oluşturan L = 300 mikro-Henries endüktansına sahiptir. Gerçekçi modelleme nedeniyle PWM darbelerinin çok dar olduğunu fark edeceksiniz. *alamadım* motorun dönmesi - hem sonlu akım yükselme süresi hem de her dar darbeden sonraki önemli kapalı kalma süresi nedeniyle. Bunlar, statik sürtünme altında dişli kutusu yayına benzer kirşlerin üstesinden gelmek için yetersiz rotor momentumuna neden olmak üzere birleştirilir. Sonuç kullanırken 'analogWrite()' , yaklaşık 0.125'in altındaki bir görev çevrimi motorun tomurcuklanmasına neden olmaz - bu, gerçek bir redüktörlü bir 'Uno' veya 'Mega'dan tabii ki uygun bir sürüş yaparken gözlenenleri kabul eder ( **ve gerekli** ) motor motorla 'Uno' veya 'Mega' arasında kontrolör modülü!.

Taklit edilmiş motor kodlayıcı, tekerlek devri başına 8 tam yüksek-düşük periyodu olan% 50 görev döngüsü e signal shape), dalgabiçimi üreten şafta monte edilmiş bir optik kesinti sensörüdür (bu nedenle program'niz tekerlek dönüşündeki değişiklikleri 22,5 derecelik bir çözünürlükte algılayabilir).

Taklit edilmiş motor kodlayıcı, tekerlek devri başına 8 tam yüksek-düşük periyodu olan% 50 görev döngüsü e signal shape), dalgabiçimi üreten şafta monte edilmiş bir optik kesinti sensörüdür (bu nedenle program'niz tekerlek dönüşündeki değişiklikleri 22,5 derecelik bir çözünürlükte algılayabilir).



## Servo Motoru ('SERVO')

Bu 'I/O' cihaz, pozisyon kontrollü PWM-ıtilmiş 6 volt besleme DC servo motor taklit eder. Servo işletimi için mekanik ve elektriksel modelleme parametreleri, standart bir HS-422 servosunununkilerle yakından eşleşecektir. Servo, 180 milisaniyede yaklaşık 60 derecelik bir maksimum dönme hızına sahiptir. Eğer sol alt onay kutusu işaretliyse servo bir **Sürekli-rotasyon** Aynı maksimum hızda servo, ancak şimdi PWM darbe genişliği **hız** açıdan ziyade



program'niz bir `#include <Servo.h>` ilan etmeden önce `'Servo'` örnek (ler) `'Servo.h'` kütüphane işlevini kullanmayı seçerseniz, Örneğin `'Servo.write()'`, `'Servo.writeMicroseconds()'` Alternatif olarak, 'SERVO' ayrıca `'digitalWrite()'` "Çarpma çarpması" sinyalleri. İç uygulama nedeniyle UnoArduSim, 6 'SERVO' cihazlar ile sınırlıdır.

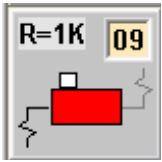
## Piezzo Hoparlör ('PIEZO')



Bu cihaz, seçilen herhangi bir 'Uno' veya 'Mega' pin üzerindeki sinyalleri "dinlemenizi" sağlar ve program işleminizi hata ayıklamak için LED'lere yardımcı olabilir. Ayrıca, zil seslerini uygun bir şekilde çalarak da biraz eğlenebilirsiniz. `'tone()'` ve `'delay()'` aramalar (e signal shape), dalgabıçımı dikdörtgeninin filtrelenmemesine rağmen "saf" notaları duymazsınız).

Bir 'PIEZO'yi pin'e cihaz iletir'ye bağlayarak bağlı bir 'PULSER' veya 'FUNCEN' cihaz'ı de dinleyebilirsiniz.

## Slayt Direnci ('R=1K')



Bu cihaz, kullanıcının bir 'Uno' veya 'Mega' pin'e 1 k-Ohm çekme direncini + 5V'a veya 1 k-Ohm çekme direncini toprağa bağlamasına izin verir. Bu, cihaz gerçek donanımına eklenen elektrik yüklerini taklit etmenize olanak sağlar. Sürgülü düğmeyi sol tıklatarak **vücut** istediğiniz yukarı çekme veya aşağı çekme seçimini değiştirebilirsiniz. Bu cihazlar'den bir veya birkaç tanesini kullanmak, program'nizin okuması ve yanıt vermesi için tek bir (veya çoklu) -bit "kod" belirlemenize izin verir.

## Butona Basınız ('PUSH')



Bu 'I/O' cihaz normalde açık öykünür **anlık VEYA mandallama** tek kutuplu, tek atış (SPST) 10 k-ohm çekme (veya aşağı çekme) rezistanslı basma düğmesi. cihaz için yükselen bir geçiş seçimi seçilirse, buton kontakları cihaz pin ile + 5V arasına 10 k-Ohm toprağa çekilecek şekilde bağlanacaktır. cihaz için bir geçiş kenarı geçişi seçildiğinde, buton kontakları cihaz pin ile toprak arasında, 10 k-Ohm'luk bir + 5V çekme ile kablolanacaktır.

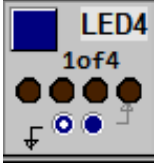
Düğmeye sol tıklayarak veya herhangi bir tuşa basarak, düğmeyi kapatın. İçinde **anlık** modundayken, fare düğmesini veya tuşunu basılı tuttuğunuz süre boyunca kapalı kalır ve **mandal** modu ('latch'ye tıklayarak etkin) düğmesine), düğmeye tekrar basana kadar kapalı kalır (ve farklı bir renkte). Temas sıçraması (1 milisaniye boyunca) her seferinde kullan **Uzay-bar** düğmesine basın.

## Renkli LED ('LED')



Seçtiğiniz 'Uno' veya 'Mega' pin (dahili gizli seri 1 k-Ohm akım sınırlama direnci ile) arasına bir LED'i toprakla ya da + 5V'a bağlayabilirsiniz - bu, bağlanan pin 'HIGH' veya yerine ne zaman 'LOW'. LED rengi, düzenleme kutusunu kullanarak kırmızı ('R'), sarı ('Y'), yeşil ('G') veya mavi ('B') olarak seçilebilir.

## 4-LED Satırı ('LED4')

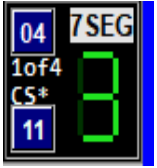


Bu 4 renkli LED sırasını seçilen 'Uno' veya 'Mega' pins (her biri dahili gizli seri 1 k-Ohm akım sınırlayıcı dirençli) arasına şasiye veya + 5V'a bağlayabilirsiniz - bu size LED'lerin ışığını alma seçeneğini sunar bağlı pin 'HIGH' veya yerine ne zaman 'LOW'.

The '1of4' pin düzenleme kutusu, ortalama alınacak tek bir pin sayısını kabul eder. **dört ardışık ilk 4** LED'e bağlanacak 'Uno' veya 'Mega' pins.

LED rengi ('R', 'Y', 'G' veya 'B') bir **gizli seçenek** Bu olabilir **sadece düzenleme IODevices.txt dosya** (hangi kullanarak oluşturabilirsiniz **Kaydet** -den **Yapılandır** | **I/O** Cihazlar iletişim kutusu).

## 7 Bölümlü LED Sayı ('7SEG')



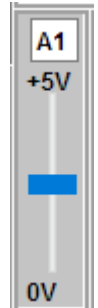
Bu 7 Bölümlü Sayı LED ekranını bir seçilen küme **onaltılı kodunu veren dört ardışık 'Uno' veya 'Mega' pins** İstenen görüntülen sayı için ('0' ila 'F') ve CS \* pin'ü (AÇIK için etkin DÜŞÜK) kullanarak bu sayı'yu açın veya kapatın.

Bu cihaz, kullanan bir dahili kod çözücüyü içerir. **aktif YÜKSEK** arka arkaya dört seviyedeki seviyeleri '1of4' İstenilen onaltılı sayı'un belirlenmesi için pins'in gösterilmesi. En düşük pin numarasındaki Te seviyesi (ekranda '1of4' düzenleme kutusu) 4-bit onaltılı kodunun en az anlamlı bitidir.

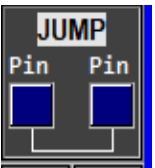
LED segmentlerinin rengi ('R', 'Y', 'G' veya 'B') **gizli seçenek** Bu olabilir **sadece düzenleme IODevices.txt dosya** kullanarak oluşturabilirsiniz **Kaydet** -den **Yapılandır** | **I/O** Cihazlar iletişim kutusu.

## Analog Kaydırıcı

Sürgü kontrollü bir 0-5V potansiyometre, tarafından okunacak bir statik (veya yavaş yavaş değişen) analog voltaj seviyesi üretmek için seçilen herhangi bir 'Uno' veya 'Mega' pin'e bağlanabilir. 'analogRead()' 0 ila 1023 arasında bir değer olarak. analog kaydırıcısını sürüklemek için fareyi kullanın veya atlamak için tıklayın.



## Pin Tel Bağlayıcı ('JUMP')



İki bağlayabilirsiniz Bu cihaz kullanılarak birlikte 'Uno' veya 'Mega' pins (ikinci pin numarasını doldurduğunuzda herhangi bir elektrikli as in electrical pin-driving conflict), çakışma tespit edilirse, seçilen bağlantıya izin verilmez ve pin'ün bağlantısı kesilir.

Bu jumper cihaz'ın sınırlı bir faydası vardır ve en çok kesintilerle birleştirildiğinde, program testi, deneyi ve öğrenme amaçları **UnoArduSim V2.4'ten itibaren bir 'PROGIO' cihaz kullanarak bulabilirsiniz bunun yerine interrupt-tilmiş yöntemlerinden daha fazla esneklik sunuyor.**

Bu cihaz için üç olası kullanım aşağıdaki gibidir:

- 1) Yapabilirsin **dijital girişi oluşturmak program'nizi test etmek için** Verilen setlerden herhangi biri kullanılarak üretililecek olandan daha karmaşık zamanlama'ye sahiptir. standart 'I/O' cihazlar, aşağıdaki gibidir:

Bir kesme işlev tanımla (hadi diyelim) `'myIntr()'` ve yap `'attachInterrupt(0, myIntr, RISING)'` senin içinde `'setup()'`. Bağlayın **Verici** cihaz - pin2 - şimdi `'myIntr()'` çalıştır olacak her zaman bir **Verici** yükselen kenar oluşur. Sizin `'myIntr()'` işlev, programlanmış'iniz olan bir algoritma olabilir. (değişkenler küresel sayacını ve hatta `'random()'`) kendi tasarınıza ait e signal shape), dalga biçimi'yu mevcut `'OUTPUT'` pin (bunun pin 9 olduğunu söyleyelim). şimdi **JUMP** pin 9 istediğiniz 'Uno' veya 'Mega'e 'INPUT'pin'ü dijital e signal shape), dalga biçimi'u bu pin girişine uygulamak için uygulamak (program'nizi test etmek için; bu e signal shape), dalga biçimi'a verilen cevabı). Bir dizi bakliyat veya seri karakter oluşturabilir veya basitçe kenar geçişlerini, isteğe bağlı karmaşıklık ve değişken aralıklarla oluşturabilirsiniz. Lütfen bunu not al ana program'niz ararsa `'micros()'` (veya ona dayanan herhangi bir işlev'ü çağırır), `'return'` değer **artırılacak** içinde harcanan zamana göre `'myIntr()'` işlev kesme her yandığında. Çağrılarını kullanarak doğru zamanlanmış kenarlardan hızlı bir patlama oluşturabilirsiniz. `'delayMicroseconds()'`, den içinde `'myIntr()'` (belki bir bütün oluşturmak için **bayt** bir yüksek baud hızı transferi), ya da sadece bir geçiş oluşturun kesme başına (belki üretmek **azıcık** düşük baud hızı aktarımının) **Verici** cihaz **'Period'** zamanlama ihtiyaçlarınıza uygun olarak seçildi (hatırlayın **Verici** minimum sınırlar **'Period'** 50 mikrosaniye).

## 2) Yapabilirsin **alt sistem geri döngüleri ile deneme:**

Örneğin, 'SERIAL' 'I/O' cihaz TX'in bağlantısını kesin. **'00'** pin (boş olarak düzenleyin) ve ardından **JUMP** 'Uno' veya 'Mega' pin **'01'** pin'e geri dön **'00'** ATmega'nın donanım döngüsünü taklit etmek `'Serial'` alt sistemi. Şimdi testinizde program, içinde `'setup()'` bir yap **tek** `'Serial.print()'` bir kelime veya karakter, ve senin içinde `'loop()'` geri alınan herhangi bir karakteri geri yankı `'Serial.available()'` yaparak `'Serial.read()'` ardından bir `'Serial.write()'` ve sonra ne olduğunu izleyin. Bunun benzer olduğunu gözlemleyebilirsiniz. `'SoftwareSerial'` Geri döngü **başaramayacak** (gerçek hayatta olduğu gibi - yazılım aynı anda iki şey yapamaz).

Ayrıca deneyebilirsiniz **SPI** kullanarak geri döngü **JUMP** pin 11'i (MOSI) tekrar pin 12'ye (MISO) bağlamak için.





## 3) Yapabilirsin **Sayıyı say ve / veya belirli seviye geçişlerinin aralığını ölç herhangi bir 'Uno' veya 'Mega'de pin X çıktı** bir kompleksin sonucu olarak ortaya çıkan Arduino talimatı veya kütüphane işlev (örnek olarak: `'analogWrite()'` veya `'OneWire::reset()'`, veya `'Servo::write()'`) , aşağıdaki gibi:

**JUMP** pin **X** kesmek için pin **2** ve senin içinde `'myIntr()'` kullanın `'digitalRead()'` ve bir `'micros()'` telefon etmek, ve kaydedilen seviyelerle ve zamanlarla karşılaştırmak (önceki kesintilerden). Gerekirse bir sonraki kesintinin kenar hassasiyetini değiştirebilirsiniz, kullanma `'detachInterrupt()'` ve `'attachInterrupt()'` itibaren **iceride** sizin `'myIntr()'` . pin'ü takip edemeyeceğinizi unutmayın birbirine çok yakın gerçekleşen geçişler (toplam icra sizin `'myIntr()'` işlev), örneğin I2C veya SPI transferleri veya yüksek baud hızı ile olanlar gibi `'Serial'` transferler (işlev kesintiniz, donanım tarafından üretilen bu transferlerin son teknoloji zamanlama'sini rahatsız etmese de) Ayrıca, yazılım aracılı transferlerin (benzeri `'OneWire::write()'` ve `'SoftwareSerial::write()'`) zamanlama bozulmalarını önlemek için kasten kesinti yapılmasını önler (kütüphane kodlarıyla tüm kesintileri geçici olarak devre dışı bırakır), bu nedenle bu yöntemi kullananları ölçemezsiniz.






Yine de aynı kenar boşlukları ölçümlerini yapabilmenize rağmen **görsel** içinde **Dijital Dalga Formları** pencere, çok sayıda geçiş üzerindeki minimum veya maksimum boşluklarla ya da geçişleri saymakla ilgileniyorsanız, bunu kullanarak `'myIntr()'` -artı- **JUMP** teknik daha uygundur. Ve yapabilirsin örneğin, ana program'nin ürettiği geçişlerin aralığındaki değişikliklerin ölçülmesi (yazılımınızın, farklı icra zamanlarında değişen icra yolları alan etkisinden dolayı), bir tür yapmak program "profil".

## Menüler









### Dosya:

<b><u>Doldurmak INO veya PDE Prog (Ctrl-L)</u></b> 	Kullanıcının, seçilen uzantıya sahip bir program dosya seçmesine izin verir. program'ye hemen Ayrıştır verilir
<b><u>Düzenle/İnceleme (Ctrl-E)</u></b>	Yüklenen program'yi görüntülemek / düzenlemek için açar.
<b><u>Kaydet</u></b> 	Düzenlenen program Kaydet orijinal program dosya'ye geri döner.
<b><u>Farklı Kaydet</u></b>	Kaydet, program içeriğini farklı bir dosya adı altında düzenledi.
<b><u>Sonraki ('#include')</u></b> 	İlerler <b>Kodlama Paneli</b> sonrakini görüntülemek ' <b>#include</b> ' dosya
<b><u>Önceki</u></b> 	Döndürür <b>Kodlama Paneli</b> önceki dosya'ye göster
<b><u>Kapat</u></b>	Değiştirilmiş dosya (ler) i kaydetmesini hatırlattıktan sonra UnoArduSim'den çıkar.

### Bul:

<b><u>Yukarı taşı Çağrı yığını</u></b> 	Çağrı yığınınındaki önceki arayan işlevine atla - <b>Değişken Paneli'deki</b> bu işleve göre ayarlanır
<b><u>Aşağı inmek Çağrı yığını</u></b> 	Çağrı yığınınında bir sonraki çağrılan işleve atla - <b>Değişken Paneli'deki</b> bu işleve göre ayarlanır
<b><u>Ara Metnini Ayarla (CTRL-F)</u></b> 	Etkinleştir <b>Aracı-Bar</b> Bir sonraki aranacak metni tanımlamak için Bul düzenleme kutusu (ve şu anda vurgulanan satırdaki ilk sözcüğü <b>Kodlama Paneli</b> veya <b>Değişken Paneli</b> onlardan biri varsa odak).
<b><u>Bul Sonraki Metin</u></b> 	Bir sonraki Metin oluşumuna atlayın. <b>Kodlama Paneli</b> (etkin odağı varsa) veya ekranda bir sonraki Metin <b>Değişken Paneli</b> (eğer bunun yerine aktif odağa sahipse).
<b><u>Bul Önceki Metin</u></b> 	İçindeki önceki Metin oluşumuna atla <b>Kodlama Paneli</b> (etkin odağı varsa) veya önceki <b>Değişken Paneli</b> (eğer bunun yerine aktif odağa sahipse).

## Çalıştır:

<b>İçine Adım (F4)</b>		icra'i bir talimatla ileri ya da <i>denilen bir işlev içine</i> .
<b>Etrafında Adım (F5)</b>		icra'i bir talimatla ileri ya da <i>tek bir tam olarak işlev araması</i> .
<b>Dışarıda Adım (F6)</b>		Gelişmeler icra by <i>Sadece şu anki işlev'ü bırakacak kadar</i> .
<b>Komuta Çalıştır (F7)</b>		program'yi çalıştırır, <i>istenen program hattında durma</i> - Komuta Çalıştır'yi kullanmadan önce, önce vurgulu'a istenen program hattını tıklamanız gerekir.
<b>Koşula Çalıştır (F8)</b>		program'yi, vurgulu'da mevcut olan değişken'e bir yazma gerçekleşene kadar çalıştırır. <b>Değişken Paneli</b> (İlk vurgulu'u kurmak için birine tıklayın).
<b>Çalıştır (F9)</b>		program'yi çalıştırır.
<b>Dur (F10)</b>		Durdular program icra (ve <i>zaman donuyor</i> ).
<b>Reset</b>		program'yi sıfırlar (tüm değişkenler değeri 0 değerine sıfırlanır ve tüm gösterici-değişkenler 0x0000 olarak sıfırlanır).
<b>Oynat</b>		Ardışık program satırlarını otomatik olarak adımlar <i>yapay gecikme ile</i> ve mevcut kod satırının vurgulanması. Gerçek zamanlı işlem ve sesler kayboluyor.
<b>Yavaş</b>		Zamanı 10 kat düşürür.

## Seçenekler:

<b><u>Etrafında Adım Yapıları/ Operatörler</u></b>	Herhangi bir adım sırasında, inşaatçılar, yıkıcılar ve işlevler operatör aşırı yüklenmesinden sağa doğru ilerleyin (yani bu işlevler içinde durmayacaktır).
<b><u>Kayıt-Tahsisi</u></b>	işlev yerlilerini istif yerine boş ATmega kayıtlarına atayın (bir miktar azalmış RAM kullanımı oluşturur).
<b><u>Başlatılmamış Hata</u></b>	program'nizin bir değişken kullanmaya çalıştığı herhangi bir yerde Ayrıştır hatası olarak işaretleyin (veya bir dizin içinde en az bir değer).
<b><u>Katma 'loop()' gecikme</u></b>	Her seferinde 1000 mikrosaniye gecikme süresi ekler 'loop()' denir (başka program araması yapılmaması durumunda 'delay()' her yerde) - gerçek zamanın çok gerisinde kalmaktan kaçının.
<b><u>Yuvalanmış Kesmelere İzin Ver</u></b>	İle yeniden etkinleştirmeye izin ver 'interrupts()' bir kullanıcı tarafından kesme servis rutini içinden.

## Yapılandır:

<b><u>'I/O' Cihazlar</u></b>	Kullanıcının, istenen 'I/O' cihazlar'nin tiplerini ve numaralarını seçmesine izin vermek için bir iletişim kutusu açar. Bu iletişim kutusundan Kaydet 'I/O' cihazlar'yi bir metin dosya'ye ve / veya Doldurmak 'I/O' cihazlar'ye daha önce kaydedilmiş (veya düzenlenmiş) bir metin dosya'den (tüm pin bağlantılarını ve tıklanabilir ayarları ve girilen değerleri içeren) yazabilirsiniz.
<b><u>Tercihler</u></b>	Kullanıcının, kaynak program satırlarının otomatik girintisini içeren, Uzman sözdizimine izin veren, yazıyüzü yazı tipi seçimi, daha büyük yazı tipi boyutu seçen, dizin sınırlarını uygulayan, mantıksal operatör anahtar kelimelerine izin veren, program indirme'ü gösteren tercihleri belirlemesi için bir iletişim kutusu açar. , 'Uno' veya 'Mega' kartı versiyonunun seçimi ve TWI tampon uzunluğu (I2C cihazlar için).

## VarYenile:

<b><u>Otomatik İzin Ver (-)</u></b> <b><u>Opposite of exand the</u></b> <b><u>displayed array), daralt</u></b>	UnoArduSim'in opposite of exand the displayed array), daralt'e gerçek zamanlıın altına düşmesi durumunda genişletilmiş diziler / includes 'struct' and 'class'), nesneleri göstermesine izin ver.
<b><u>En az</u></b>	Sadece <b>Değişken Paneli</b> saniyede 4 kez görüntüler.
<b><u>Vurgulu değişiklikler</u></b>	Vurgulu çalışırken değişken değerlerini değiştirdi (yavaşlamaya neden olabilir).

## Pencereler:

<b><u>'Serial' Monitör</u></b>	Seri G / Ç cihaz'ı pins 0 ve 1'e (eğer değilse) bağlayın ve daha büyük bir yukarı çekin 'Serial' TX / RX metin pencere'ü izleyin.
<b><u>Her şeyi eski haline getir</u></b>	Küçültülmüş tüm çocuk pencereler'i geri yükleyin.
<b><u>Pin Dijital Dalga Formları</u></b>	Küçültülmüş Pin Dijital Dalga Formları pencere'ü geri yükleyin.
<b><u>Pin Analog Dalga Formu</u></b>	Küçültülmüş Pin Analog Dalga Formu pencere'ü geri yükleyin.

## Yardım:

<b><u>Hızlı Yardım Dosya</u></b>	UnoArduSim_QuickHelp PDF dosya'yi açar.
<b><u>Tam Yardım Dosya</u></b>	UnoArduSim_FullHelp PDF dosya'yi açar.
<b><u>Hata Düzeltmeleri</u></b>	Önceki sürümden bu yana önemli hata düzeltmelerini görüntüleyin.
<b><u>Değişim / İyileştirmeler</u></b>	Önceki sürümden bu yana yapılan önemli değişiklikleri ve iyileştirmeleri görüntüleyin.
<b><u>hakkında</u></b>	Sürüm, telif hakkı görüntüler.

## 'Uno' veya 'Mega' Kartı ve 'I/O' Cihazlar

'Uno' veya 'Mega' ve ekli 'I/O' cihazlar elektriksel olarak doğru bir şekilde modellenmiştir ve programlar'unuzun asıl donanımla nasıl davranacağına dair iyi bir fikir edinebilirsiniz ve tüm pin çatışmalar işaretlenecektir.

### Zamanlama

UnoArduSim çalışır bir PC veya tablette yapabileceği kadar hızlı bir şekilde ( *vakaların çoğunda* ) program modelinin gerçek zamanlı olarak gerçekleştirilmesi, **fakat sadece program'niz içeriyorsa** en azından biraz küçük `'delay()'` aramalar veya doğal olarak gerçek zamanlı olarak senkronize edilmesini sağlayacak diğer aramalar (aşağıya bakınız).

Bunu başarmak için UnoArduSim, gerçek zamanlı olarak doğru bir şekilde takip etmesini sağlayan Pencere geriler geri arama zamanlayıcısı işlev'ü kullanır. Bir program talimatının icra'i bir zaman dilimi diliminde simüle edilir ve daha uzun icra gerektiren talimatlar (aramalar gibi) `'delay()'` ) birden fazla zamanlayıcı dilimi kullanmanız gerekebilir. Geri arama zamanlayıcı işlev'ün her yinelenmesi, sistem donanım saatini kullanarak sistem zamanını düzeltir; böylece program icra, gerçek zamanlı olarak kilit adımda kalması için sürekli ayarlanır. **Sadece icra oranı şart gerçek zamanın gerisinde kalmak** kullanıcının sıkı döngüler oluşturduğu zaman **gecikme olmadan** veya 'I/O' cihazlar, aşırı sayıda pin seviyesi değişim olayı ve bununla bağlantılı işlem aşırı yükü üretecek olan çok yüksek 'I/O' cihaz frekansları (ve / veya baud hızı) ile çalışmak üzere yapılandırılmıştır. UnoArduSim telafi etmek için bazı zamanlayıcı aralıkları atlayarak bu aşırı yük ile başa çıkar ve bu program'ye ilerlemesini yavaşlatır. **gerçek zamanın altında** .

Ek olarak, büyük diziler ile programlar görüntüleniyor veya tekrar sıkı döngüler var **gecikme olmadan** yüksek bir işlev çağrı frekansına neden olabilir ve **Değişken Paneli** güncelleme yükünün gerçek zamanın gerisinde kalmasına neden olan görüntüleme güncellemesi - UnoArduSim, değişken'in yenileme sıklığını otomatik olarak azaltır, ancak daha fazla redüksiyon gerektiğinde, **En az** , -den **VarYenile** Menü saniyede yalnızca dört yenileme belirtmek için.

Her program talimatı veya operasyonu için alt milisaniye icra süresinin doğru şekilde modellenmesi **yapılmadı** - simülasyon amaçlı yalnızca çoğu kaba tahminler kabul edilmiştir. Ancak, zamanlama'nin `'delay()'` , ve `'delayMicroseconds()'` işlevler ve işlevler `'millis()'` ve `'micros()'` hepsi mükemmel doğru ve **işlevler gecikmesinden en az birini kullandığınız sürece** program'nizin herhangi bir yerindeki bir ilmek **veya** Gerçek zamanlı operasyonla doğal olarak kendisini bağlayan bir işlev kullanıyorsanız `'print()'` seçilen baud hızı'ye bağlı olan), sonra program'nizin simüle edilmiş performansı gerçek zamana çok yakın olacaktır (yine, aşırı derecede yüksek frekanslı pin seviyesi değişim olaylarını veya kullanıcı tarafından izin verilen aşırı Değişkenler güncellemelerini engelleyen).

Bireysel program talimatlarının etkisini görmek için **çalışan tavuk** , işleri yavaşlatabilmek arzu edilebilir. Menü altında kullanıcı tarafından 10'luk bir zaman yavaşlatma faktörü belirlenebilir **Çalıştır** .

### 'I/O' Cihaz Zamanlama

Bu sanal cihazlar pins girişinde meydana gelen değişikliklerin gerçek zamanlı sinyali alır ve daha sonra 'Uno' veya 'Mega' tarafından algılanabilen pins çıkışında karşılık gelen çıktılar üretir - bu nedenle bunlar program icra ile senkronize edilir. Dahili 'I/O' cihaz zamanlama kullanıcı tarafından ayarlanır (örneğin, baud hızı seçimi veya Saat frekansı aracılığıyla) ve simülatör olayları, gerçek zamanlı dahili çalışmayı izlemek için ayarlanır.

### Sesler

Her 'PIEZO' cihaz, bu tür değişikliklerin kaynağına bakılmaksızın, ekli pin'te meydana gelen elektriksel seviye değişikliklerine karşılık gelen ses üretir. Sesleri program icra ile senkronize tutmak için UnoArduSim başlar ve icra başlatılır / durdurulurken ilgili bir ses arabelleğinin oynatılmasını durdurur.

V2.0'dan itibaren, ses artık Qt ses API'sini kullanacak şekilde değiştirildi - maalesef QAudioOutput `'class'` Ses örneklerinin bitmesini önlemek için ses arabelleğinin çevrilmesini desteklemiyor (daha uzun işletim sistemi pencereleme işlemsel gecikmeler sırasında olabileceği gibi). Bu nedenle, işletim sistemi gecikmeleri sırasında can sıkıcı ses tıklamalarının ve ses bozulmalarının büyük çoğunluğunu önlemek için, ses aşağıdaki kurallara göre kapatılmıştır:

UnoArduSim "aktif" pencere olmadığı sürece ses kesilir (yeni bir pencere çocuğu henüz yaratılmış ve aktif hale getirilmişse hariç), **ve hatta** UnoArduSim "aktif" ana pencere'tür ancak fare işaretçisi **dışında** arasında ana



pencere müşteri alanı.

Bunun, sesin geçici olacağı anlamına geldiğini unutmayın. fare işaretçisini gezdirirken kral gibi sessiz **bir çocuk üzerinde pencere**, ve sessizleşecek **Eğer bu çocuk pencere'ü etkinleştirmek için tıklanır** (Ana UnoArduSim pencere tekrar etkinleştirmek için tekrar tıklanana kadar) .

**UnoArduSim ana pencere'ün müşteri alanındaki herhangi bir yeri tıklatarak ses her zaman kapatılabilir.**

Tamponlama nedeniyle, sound, bağlı 'PIEZO'nin pin'ünde ilgili olay süresinden 250 milisaniyeye kadar gerçek zamanlı bir gecikmeye sahiptir.

## Sınırlamalar ve Desteklenmeyen Öğeler

### Dahil Dosyalar

Bir '<>' - braketli '#include' arasında '<Servo.h>', '<Wire.h>', '<OneWire.h>', '<SoftwareSerial.h>', '<SPI.h>', '<EEPROM.h>' ve '<SD.h>' olduğu desteklenir ancak bunlar yalnızca öykünür - gerçek dosyalar aranmaz; bunun yerine işlevleri doğrudan "UnoArduSim'de" yerleşiktir ve sabit Arduino sürümü için geçerlidir.

Herhangi bir alıntı '#include' (örneğin "supp.ino", "Myutil.cpp" veya "Mylib.h") desteklenir, ancak tüm bu dosyalar gerekir **içinde ikamet etmek aynı dizin ebeveyn olarak program dosya** o onların içeriyor '#include' (başka dizinlerde arama yapılamaz). The '#include' özelliği, gösterilen program kodunun miktarını en aza indirmek için yararlı olabilir. **Kodlama Paneli** herhangi bir zamanda. Başlık dosyalar ile '#include' (yani olanlar ".h" uzatma) ek olarak simülatörün aynı isimli dosya'ye sahip ".cpp" uzatma (ayrıca, program üst dizininde de mevcutsa).

### Dinamik Bellek Ayırma ve RAM

Operatörler 'new' ve 'delete' yerli Arduino gibi desteklenir 'String' includes 'struct' and 'class'), nesneleri, **ancak doğrudan aramalar** 'malloc()', 'realloc()' ve 'free()' bu güveniyor.

değişken bildirimleri için aşırı RAM kullanımı Ayırıştır zamanında işaretlenir ve program icra sırasında RAM bellek taşması işaretlenir. bir menüdeki öğe **Seçenekler** AVR derleyici tarafından yapıldığı gibi normal ATmega kayıt tahsisini taklit etmenize veya yalnızca yığın kullanan alternatif bir derleme düzenini modellemenize izin verir (yalnızca hata'ın kayıt tahsis modellememde ortaya çıkması durumunda güvenlik seçeneği olarak). Yığın içeriğine bakmak için bir işaretçi kullanıyorsanız, gerçek bir donanım uygulamasında ne olacağını doğru bir şekilde yansıtmalıdır.

### 'Flash' Bellek Tahsisleri

'Flash' belleği 'byte', 'int' ve 'float' değişkenler / diziler ve bunlara karşılık gelen işlevler okuma erişimi desteklenir. herhangi 'F()' işlev çağırısı ('Flash' -macro) arasında herhangi bir hazır bilgi dizesi olduğu desteklenir, ancak yalnızca desteklenen 'Flash' - bellek dizisi doğrudan erişim işlevler 'strcpy\_P()' ve 'memcpy\_P()', diğer işlevler'i kullanmak için önce 'Flash' dizesini normal bir RAM'e kopyalamanız gerekir. 'String' değişken , ve sonra o RAM ile çalış 'String' . Kullandığınızda 'PROGMEM' değişken değiştirici anahtar kelime görünmeli **önünde** değişken adı ve bu değişken **ayrıca bildirilmeli** gibi 'const' .

### 'String' Değişkenler

Yerli 'String' kütüphane neredeyse birkaç istisna dışında tamamen desteklenir.

The 'String' Desteklenen operatörler +, + =, <, <=, >, > =, ==, !=, ve [] . Bunu not et: 'concat()' bir tane al **tek** olan argüman 'String', veya 'char' veya 'int' aslına eklenecek 'String' includes 'struct' and 'class'), nesne, **değil** Arduino Reference web sayfalarında yanlışlıkla belirtildiği gibi iki argüman).

## Arduino Kütüphaneleri

Bir tek 'Servo.h' , 'SoftwareSerial.h' , 'SPI.h' , 'Wire.h' , 'OneWire.h' , 'Stepper.h' , 'SD.h' , 'TFT.h' ve 'EEPROM.h' için **Arduino V1.8.8** bırakın şu anda UnoArduSim desteklenir. V2.6 3 için bir mekanizma sunar<sup>rd</sup> dosyalar aracılığıyla parti kütüphane desteği sağlanan 'include\_3rdParty' Klasör o dizini yüklemek UnoArduSim içinde bulunabilir. çalışılıyor '#include' ".Cpp" ve ".h" Diğer as-yet desteklenmeyen bir dosyalar kütüphaneler olacak değil iş Onlar düşük seviyeli montaj talimatları ve desteklenmeyen direktifleri ve tanınmayan dosyalar içerecektir olarak!

## İşaretçiler

Basit tipteki işaretçiler, diziler veya includes 'struct' and 'class'), nesneleri desteklenir. Bir işaretçi, aynı tipte bir dizin'e eşit olabilir (örn. 'iptr = intarray' ) ama sonra olacaktı *takip eden diziler sınır kontrolü yok* gibi bir ifadede 'iptr[index]' .

İşlevler göstericileri iade edebilir veya 'const' işaretçiler, ancak sonraki 'const' döndürülen işaretçide yoksayılır.

Var **destek yok** işlev aramaları için **kullanıcı tarafından bildirilen işlev-işaretçiler** .

## 'class' ve 'struct' Includes 'struct' and 'class'), nesneleri

Her ne kadar çoklu-morfizm ve kalıtım (herhangi bir derinlikte) desteklense de, 'class' veya 'struct' sadece en fazla olacak şekilde tanımlanabilir **bir** baz 'class' (yani **çoktan** kalıtım desteklenmez). Yapıcı bildirim satırlarındaki Base-'class' yapıcı başlatma çağrıları (iki nokta üst üste yazarak) desteklenir, ancak **değil** aynı kolon notasyonunu kullanarak üye-başlatmalar. Bu, içeren includes 'struct' and 'class'), nesneleri anlamına gelir 'const' 'static' olmayan değişkenler veya referans tipi değişkenler, desteklenmez (bunlar sadece belirtilen yapım zamanı üye başlatma işlemleriyle mümkündür)

Kopya atama operatörü aşırı yüklemeleri, taşıma yapıcıları ve taşıma atamalarıyla birlikte desteklenir, ancak kullanıcı tanımlı includes 'struct' and 'class'), nesne dönüşümü ("tipleme") işlevler desteklenmez.

## Kapsam

İçin destek yok 'using' anahtar kelime veya 'namespace' , yada ... için 'file' kapsam. Tüm yerel olmayan beyannameler, uygulamanın küresel olduğu varsayılarak uygulanır.

herhangi 'typedef' , 'struct' veya 'class' tanım (yani gelecekteki bildirimler için kullanılabilir), yapılmalıdır **global** kapsam ( **yerel** işlev'teki bu tür öğelerin tanımları desteklenmez).

## elemeleri 'unsigned', 'const' , 'volatile', 'static'

The 'unsigned' örnek tüm normal yasal bağlamlarda çalışır. The 'const' Anahtar kelime, kullanıldığında, **önce** değişken ismi veya işlev ismi veya 'typedef' bildirilen ad - adın arkasına yerleştirmek Ayırıştır hatasına neden olur. İçin işlev bildirimleri, yalnızca işaretçi döndüren işlevler'e sahip olabilir 'const' beyanlarında görünmektedir.

*Herşey* UnoArduSim değişkenler 'volatile' uygulama tarafından 'volatile' anahtar kelime, tüm değişken bildirimlerinde dikkate alınmaz. İşlevler'nın ilan edilmesine izin verilmiyor 'volatile' işlev çağrı argümanları da yoktur.

The 'static' anahtar kelime değişkenler normal ve includes 'struct' and 'class'), nesne üyeleri ve işlevler üyesi için, ancak includes 'struct' and 'class'), nesne örnekleri için açıkça reddedilir ( 'class' / 'struct' ), üye olmayan işlevler ve tüm işlev argümanları için.

## Derleyici Direktifleri

'#include' ve düzenli '#define' ikisi de destekleniyor, ancak **makro değil** '#define' . The '#pragma' direktif ve şartlı dahil etme yönergeleri ( '#ifdef' , '#ifndef' , '#if' , '#endif' , '#else' ve '#elif' )

ayrıca **desteklenmiyor**. The '#line', '#error' ve önceden tanımlanmış makrolar '\_LINE\_', '\_FILE\_', '\_DATE\_', ve '\_TIME\_' ) ayrıca **desteklenmiyor**.

## Arduino dili elemanları

Tüm yerli Arduino dil unsurları şüpheli hariç 'goto' talimatı (bunun için makul olan tek kullanım, program'nizin başka türlü başa çıkamayacağı bir hata koşulu durumunda bir sıçrama (bir kurtarma ve güvenli kapatma sonsuz döngüye) olacaktır

## C / C ++ - dil Elemanları

Yapı tanımlarındaki üyeler için bit tasarrufu sağlayan "bit alanı niteleyicileri" **desteklenmiyor**.

'union' olduğu **desteklenmiyor**.

Garip "virgül operatörü" **desteklenmiyor** (normalde yalnızca tek bir ifade beklendiğinde virgülle ayrılmış birkaç ifade gerçekleştiremezsiniz, örneğin 'while()' ve 'for( ; ; )' yapıları).

## İşlev Şablonları

"Generic" türündeki argümanları kabul etmesine izin vermek için "template" anahtar sözcüğünü kullanan kullanıcı tanımlı işlevler **desteklenmiyor**.

## Gerçek Zamanlı Emülasyon

Yukarıda belirtildiği gibi, birçok farklı bireysel Arduino program talimatının icra kez **değil** Doğru bir şekilde modellenmiştir, böylece gerçek zamanlı bir hızda çalışmak için program'nizin bir tür hükmetmeye ihtiyacı olacaktır. 'delay()' talimat (en az bir kere 'loop()' ) veya doğal olarak gerçek zamanlı pin seviyesindeki değişikliklerle senkronize edilen bir talimat (örneğin, 'pulseIn()', 'shiftIn()', 'Serial.read()', 'Serial.print()', 'Serial.flush()' vb.).

Görmek **Zamanlama** ve **sesler** Sınırlamalar hakkında daha fazla ayrıntı için yukarıda.

## Sürüm notları

### Hata Düzeltmeleri

#### V2.8.1- Haziran 2020

- 1) Otomatik biçimlendirme tercihiyle kaldırılan fazladan boş satırlar Tercihi, Düzenle/İnceleme'te başlangıçta vurgulanan satırın, üstünden kaldırılan boş satır sayısı ile aşağı doğru kaydırılmasına neden oldu.
- 2) 'analogRead()' sadece tam dijital pin numarasını kabul ediyordu.
- 3) Yalnızca bir işlev argümanının 'unsigned' özelliğinde farklı olan işlev aşırı yüklemeleri içeren bir 'class', yanlış işlev aşırı yüklemesine sahip olabilir. Bu, 'unsigned' taban-10 baskı aşırı yükü işlev'in çağrılacağı LCD 'print(char/int/long)'ü etkiledi ve bu, LCD 'printFloat()'in '.' ondalık noktası yerine '46'yı yazdırmasına neden oldu.
- 4) Önceden eşlenmiş bir dahili'in otomatik tamamlama ('Enter' üzerine), satırdaki yazma hatasını düzeltmek için geri izleme sonrasında çalışmadı.
- 5) Boş bir 'RS' ye sahip bir 'TFT' cihazı yürütmeye çökmeye neden olabilir.

#### V2.8.0- Haziran 2020

- 1) Bir Ayırıştırma uyarısı (ancak Ayırıştırma hatası değil) oluştuğunda, V2.7 hatalı arabellekteki (bunun yerine en son '#include' arabelleğinde) sorunlu satırı vurgulamaya çalışabilir ve bu da ( uyarı pop-up'ı görünür) satır numarası bu '#include' arabelleğinin sonunun ötesindeyse.
- 2) 'I2CSLV', 'SPISLV', 'TFT', 'LCDI2C' ve 'LCDSPI' cihazlarının daha büyük monitör pencerelerinde alınan baytların tümü hala doğru şekilde bildirilmemiştir (bu işlevlerini etkilememiştir).
- 3) Değişkenler türü 'unsigned char' türüne yükseltildi 'int' aritmetik ifadelerde, ancak 'unsigned' arızaya yol açan durum 'unsigned' sonuçta ortaya çıkan ifade.
- 4) Bir 'LED4' veya '7SEG' cihaz'ın pin'inin ilk geçerli bir pin ayarından değiştirilmesi (kesilmesi) üst 3 pins'ü ayırmayı başaramadı ve açıklanamayan çökmelere neden olabilir - buna ek olarak, tüm 'LED'lerin işlenmesini uyumlaştırmak için V2.7'de yapılan değişiklikler 'LED4' cihaz'ı tamamen kıldı.
- 5) Sürüm 2.6'da 'LOW' kesintilerini desteklemek için yapılan değişikliklerde yapılan bir hata, pin 3'e bağlı 'FALLING' kesintilerinin pin 2'deki kesme düzeyi değişikliği algılamasını bozmasına neden oldu.
- 6) 'SoftwareSerial' aldığı her karakter sırasında kullanıcı kesmelerini yanlış bir şekilde devre dışı bırakıyordu.
- 7) Ayırıştırma hatası içeren bir 'PROGIO' programı yüklendiğinde, hata işaretlendi ancak bu program zaten 'PROGIO' Monitor penceresindeki varsayılan bir 'PROGIO' programı ile değiştirilmişti.
- 8) V2.4 sürümünden beri, Pin 1 ve Pin 0'daki pin değişiklikleri indirme sırasında görülmüdü.
- 9) Açık bir SD dosya üzerinde okuma veya yazma işlemlerinin döngü yapması icra dizilemesinin başarısız olmasına neden olarak kapsam seviye derinliği nedeniyle dahili bir UnoArduSim hatasına neden oldu.
- 10) 'SD\_DRV' cihaz'teki 'MISO' pin'i değiştirmeye çalışmak MOSI pin'i bozdu.
- 11) Önceki tüm sürümler, 'analogWrite()' pins 9 veya 10'da tüm aktif 'Servo' cihazlar için 'Servo' zamanlama bozulur.

#### V2.7.0- Mart 2020

- 1) (Pencereler-varsayılan) ışık OS teması kabul edildiğinde, **Kodlama Paneli** Renk vurgulama v2.6 tanıtılan (bunun yerine, sistem bir geçersiz kılma işleminden elde edilen sadece gri vurgulu) gösterilmemektedir.
- 2) İlk biçimlendirme Sürüm 2.6 yanlışlıkla kıldı otomatik girinti tırnağı 'switch()' yapı.
- 3) Sürüm 2.6 tanıtılan yeni yonga seçme navigasyon özelliği gösterdi **yanlış değerler** yerel için değişkenler değilken şu anda yürütülen işlev içinde ve iç içe üyesi işlev çağrılar ile başarısız oldu.

- 4) 'TFT :: text ()' çalışıyordu, ancak 'TFT :: print ()' işlevleri çalışmadı (sonsuz dek basitçe engellendiler). Ayrıca, 'Serial.begin ()' daha önce yapılmışsa 'TFT :: loadImage ()' başarısız oldu (bu normal bir durumdur ve şimdi gereklidir).
- 5) Sürüm 2.6, mevcut ve geçmiş 'RX' 'I2CSLV', 'SPISLV', 'TFT', 'LCDI2C' and 'LCDSPI' cihazlar (ve Monitör pencereler) için bayt için yanlış değeri ile bir hata sundu.
- 6) Oynat birçok çalıştı kod satırları üzerinde atlamak vurgulayarak | V2.4 yapılan bir değişiklik Çalıştır neden oldu.
- 7) V2.4, talimatında bir 'I/O' cihaz üzerinde 'SS\*' veya 'CS\*' de-iddia beri hemen aşağıdaki 'SPI.transfer ()' cihaz aktarılan veri byte almayı başarısız olmasına o neden olur. Buna ek olarak, bayt resepsiyon 'SPI\_MODE1' ve 'SPI\_MODE3' usta tarafından gönderilen bir sonraki byte başlangıcına kadar işaretlenmediğinden (ve cihaz 'CS\*' daha önce de-seçilmişse bayt tamamen kayboldu).
- 8) yeni Gelen 'SPI\_SLV' V2.4 beri izin modu 'bval = SPI.transfer ()' sadece doğru değeri geri 'bval' bayt aktarımı zaten tam ve zaman bekleyen olsaydı 'transfer ()' aradı.
- 9) ile cevap vermek artık bayt olduğunda 'SPISLV' cihazlar üzerinde 'DATA' düzenleme kutusu şimdi varsayılan 0xFF alır.
- 10) Senkronizasyon LED durumu 'PSTEPR' cihazlar tam adım 1'den fazla mikro aşamasına sahiptir yanlış oldu.
- 11) Elektrik çatışmalar 'SPI' saat, 'PWM' sinyali veya a üzerindeki geçişler tepki 'I/O' cihazlar neden 'tone' Sinyal, rapor edilmedi ve açıklanamayan yol açabilir (bozuk) veri alımı.
- 12) kesme arasındaki aralık, dahili işlevler arasında zamanlama kullanımının ya da sistem zamanlayıcılar, ya da talimat döngüler değiştirilmiş V2.4 bir (arızalı) değişim gecikme (her örneklerdir üretmek üzere, (en az 250 mikro saniye) çok küçük olduğunda 'delay ()' ve 'delayMicroseconds ()'). V2.5 daha sonra yapılan bir değişiklik, yanlış hizalama neden 'shiftOut ()' Veri ve saat sinyali, bir kesme bitler arasında ne zaman.
- 13) Giriş anahtarı aracılığı ile dahili işlev otomatik tamamlama metin kabul yerleştirilmiş işlev çağrının metinden parametre türlerini şerit başarısız oldu.
- 14) Yeni (kullanıcı kesme-itilmiş) program yükleniyor önceden çalışan program hala indirme sırasında bir kazaya sebep olabilir bekleyen bir kesme varken (nedeniyle arızasına yeni kesme rutin icra teşebbüs).
- 15) Includes 'struct' and 'class', nesne üyeli otomatik tamamlamaları includes 'struct' and 'class', nesneleri iç için ('ALT'-sağ ok erişilebilir) '#include' dosyalar en kısa sürede şimdi erişilebilir onların '#include' dosya başarıyla ayrıştırıldı olduğunu.
- 16) Bir işlev parametre adı içinde bir yanlışlıkla alana sahip bir işlev beyanı bir belirsiz Ayrıştır hata mesajı neden oldu.
- 17) icra ana program, Dosya bir modül farklı yılında durdurulmuş zaman | Önceki eylem etkin hale başaramadı.
- 18) Tek alıntılanan süslü ayraç ('{ ' ve ' } ') Hala kapsam parantez gibi) yanlış (sayıldı Ayrıştır ve ayrıca otomatik sekme girinti biçimlendirme karıştı ..
- 19) 'OneWire::readBytes(byte\* buf, int count)' vardı hemen görüntülenen güncellemek için başarısız olmuştur 'buf' Değişken Paneli içinde içeriği.
- 20) Sekizli-mandal 'OWISLV' cihazlar bir mandal-register yazma işlemi ile kaldığı yönünde çıktı üretmesi pin seviyeleri gösterdi.

## **V2.6.0- Ocak 2020**

- 1) eklendiğinde Bir hata yığılmasına yol açtı V2.3 tanıtıldı **Kapat** buton kullanılmıştır **Bul / Değiştir** yerine (iletişim onun Kapat Başlık çubuğu düğmesi).
- 2) Bir kullanıcı program yaptıysak '#include' Diğer kullanıcı dosyalar, **Kaydet** düğme içeride **Düzenle/İnceleme** modifiye dosya kaydetme aslında başarısız olurdu varolan Ayrıştır veya İcra hata farklı dosya içeride bayraklı olsaydı.
- 3) bir **Temizle** a sonra **Kaydet** Ayrıca kafa karıştırıcı olabilir - Bu nedenlerle **Kaydet** ve **Temizle** düğme işlevleri değiştirildi (bkz **Değişiklikler ve geliştirmeler**).
- 4) a içindeki Dengesiz parantez 'class' tanım V2.5 beri askıda neden olabilir.

- 5) Doğrudan mantıksal test '**long**' değerler döndü '**false**' eğer 16 düşük bit hiçbirini kuruldu.
- 6) bir işaretçi değişken a parantez içinde döngü değişken olarak ilan edildiği UnoArduSim bir hata işaretleme edilmiş '**for()**' Beyan.
- 7) UnoArduSim kıyasla işaretçiler mantıksal testler izin vermemek olmuştur '**NULL**' veya '**0**'.
- 8) UnoArduSim (sabitler izin verildiği tek bir tam sayı) bir tam sayıdır değişken içeren işaretçi Aritmetik izin vermeme edilmiştir.
- 9) Kesmeler kullanılarak ayarlanır '**attachInterrupt(pin, name\_func, LOW)**' Sadece a üzerinde hissedilen olmuştur **geçiş** için '**LOW**'.
- 10) Birden fazla 'I2CSLV' cihazı kullanırken, adreslenmemiş bir bağımlı, daha sonraki veri yolu verilerini veri yolu (veya genel çağrı 0x00) adresiyle eşleştiriyor olarak yorumlayabilir ve bu nedenle yanlışlıkla ACK sinyali vererek, veri yolu ACK düzeyini bozabilir ve bir 'requestFrom()'.
- 11) Sayısal bir değer geçen '**0**' (veya '**NULL**') Bir işlev çağrısında bir işaretçi işlev bağımsız değişken olarak hemen bırakılır.
- 12) bir iç içe geçme sonra sekme girinti düzeyi '**switch()**' yapılar çok sık olduğu zamanları 'auto-indent formatting' seçimi **Yapılandır | Tercihler** kullanıldı.
- 13) İki uyumlu işaretçileri Çıkarma artık bir tür sonuçlanır '**int**'.
- 14) UnoArduSim o olarak ilan edilmemiş olsa bile bir üyesi includes 'struct' and 'class'), nesne için kullanıcı tanımlı bir varsayılan kurucu beklediklerinden '**const**'.
- 15) Bir icra arası olarak, bir 'STEPR' çekilmiş konum, 'SERVO' veya 'MOTOR' Motor gerçek son bilgisayarlı pozisyon arkasında hareket 30 milisaniye için yukarı tarafından gecikme olabilir.
- 16) '**Stepper::setSpeed(0)**' Çünkü bir böl-by-sıfır çökmesine neden oldu.
- 17) Tek satır '**if()**', '**for()**', ve '**else**' artık kimsenin bu kadar çok oto-girinti sekmelerin neden oluşturur.

## **V2.5.0- Ekim 2019**

- 1) bir hata V2.4 içine (A çökmesine neden) kıldı 'SD' kartı başlatma.
- 2) Yeni de 'SPI' alt sistemini kullanarak '**SPI\_SLV**' modunda düzgün çalışmamasına yol '**SPI\_MODE1**' ve '**SPI\_MODE3**'.
- 3) ('ALT-right=arrow' tarafından istendiği gibi) Otomatik tamamlama pop-up işlevler includes 'struct' and 'class'), nesne parametrelere sahip sabitlenmiştir; Pop-up listesi şimdi de (baz-) sınıfı üyelerini miras içerir ve otomatik tamamlamalar şimdi de görünmesini '**Serial**'.
- 4) V2.4 dönüş değeri için yana '**SPI.transfer()**' ve '**SPI.transfer16()**' Bir kullanıcı kesme rutini bu transfer sırasında ateş eğer yanlış oldu.
- 5) V2.4 olarak, hızlı periyodik dalga yüksek zoom bakıldığında belirgin gerçek süresinden daha uzun bir etki süresine sahip olarak gösterilmiştir.
- 6) inşaatçı '**File::File(SdFile &sdf, char \*fname)**' böylece, çalışma değildi '**File::openNextFile()**' (Bu yapıcı dayandığı) da çalışma değildi.
- 7) UnoArduSim yanlış includes 'struct' and 'class'), nesne-değişkenler bir Ayırıştır hata ilan edilmiş ve işlevler includes 'struct' and 'class'), nesne-geri, ilan '**static**'.
- 8) Bir 'Servo' ile Atama ifadeleri, '**Stepper**' veya '**OneWire**' Lhs includes 'struct' and 'class'), nesne değişken, ve bir 'Rhs' işlev veya yapıcı includes 'struct' and 'class'), nesne-dönen bir iç yanlış hesap sebep nihai bir giden ilişkili includes 'struct' and 'class'), nesneleri sayısı arasında çökmesine.
- 9) Çeşidi includes 'struct' and 'class'), nesneleri üzerinde Boole testler '**File**' hep dönüyorlardı '**true**' bile dosya açık değildi.

## **V2.4 - Mayıs 2019**

- 1) Koşula Çalıştır izleme noktaları, bitişik bir değişken'e (adreste 1 byte daha düşük) bir yazıyla yanlışlıkla tetiklenebilir.

- 2) Baud hızı seçimleri algılanabilir fare bir 'SERIAL' veya 'SFTSER'un içindeyken cihaz baud açılır liste kutusu (Baud hızı gerçekten tıklanmadığında bile).
- 3) V2.2'den beri, seri alım hataları 38400 baud hızında gerçekleşmiştir.
- 4) V2.3'te yapılan değişiklik '**SoftwareSerial**' Bazen, devre dışı bırakma kesintisini hatalı bir şekilde bildirmek için (ve böylece ilk RX alımında başarısız).
- 5) V2.3'te yapılan bir değişiklik, SPISLV cihazlar'nin 'DATA' düzenleme kutusuna doğrudan girilen altıgen değerlerini yanlış yorumlamasına neden oldu.
- 6) V2.3'te yapılan bir değişiklik SRSLV cihazlar'ye neden oldu bazen hatalı ve sessizce, 'Dout' pin'lerinde elektriksel bir as in electrical pin-driving conflict), çakışma tespit eder ve bu nedenle orada bir pin atamasına izin vermez. Bir pin ile 'Dout' takma girişimleri cihaz çıkarıldıktan sonra nihai bir çökmeye neden olabilir.
- 7) pin 16'dan sonra bir 'LED4' cihaz takmaya çalışmak, çarpmalara neden olur.
- 8) Hızlı tekrarlanan çağrılarla tetiklenen bir hata '**analogWrite(255)**' için '**MOTOR**' program kullanıcısındaki kontrol sonuçlandı '**MOTOR**' yanlış olması için hızlar (çok yavaş).
- 9) V2.3'ten beri, hatalı bir as in electrical pin-driving conflict), çakışma tespiti nedeniyle SRSLV cihazlar'ye bir 'Dout' pin atanamadı (ve böylece bir pin atamasına izin verme).
- 10) SPI slave'leri şimdi vericilerini ve alıcı mantığını sıfırlar. '**SS\***' pin gider '**HIGH**'.
- 11) çağrı '**Wire.h**' işlevler '**endTransmission()**' veya '**requestFrom()**' kesintiler şu anda \ devre dışı bırakıldığında, şimdi bir icra hatası oluşturuyor ('**Wire.h**' çalışmak için etkin olan kesintiler gerekir).
- 12) 'Ctrl-Home' ve 'Ctrl-End' şimdi Düzenle/İnceleme'te beklendiği gibi çalışmaktadır.
- 13) The '**OneWire**' otobüs komutu **0x33** ('**ROM\_READ**') çalışmıyordu ve otobüsü astı.

## **V2.3 - Aralık 2018**

- 1) V2.2'de tanıtılan bir hata, içindeki bir dizin ögesinin değerini düzenlemeyi imkansız hale getirdi **Düzenle/İzle**.
- 2) Sürüm 2.0'dan bu yana '**RAM free**' Araç Çubuğu kontrolü yalnızca karanlık bir Pencerele-OS teması kullanılıyorsa görünürdü.
- 3) üzerinde **Dosya | Doldurmak**ve I / O cihaz dosya hakkında **Doldurmak**, dosya filtreleri (gibi '\*.ino' ve '\*.txt') Çalışmıyordu - bunun yerine her türden dosyalar gösterildi.
- 4) dosya'nin "değiştirilmiş" durumu, yapıldıktan sonra kaybedildi **Kabul Et** veya **Derle** sonradan **Dosya | Düzenle/İnceleme başka düzenleme yapılmadıysa (Kaydet** devre dışı bırakıldı ve otomatik istemi olmadı **Kaydet** program'de **Kapat**).
- 5) Operatörler '**/=**' ve '**%=**' sadece için doğru sonuçlar verdi '**unsigned**' sol taraftaki değişkenler
- 6) Üçlü şartlı '**(bval) ? S1:S2**' ne zaman yanlış sonuç verdi '**S1**' veya '**S2**' değişken yerine yerel bir tabirdi.
- 7) İşlev '**noTone()**' olmak için düzeltildi '**noTone(uint8\_t pin)**'.
- 8) Uzun süredir devam eden bir hata **Reset** bu Reset, parametrelerinden biri eksik olarak adlandırılan bir işlev'ün ortasında yapıldığında (ve böylece varsayılan bir başlatıcı değeri alıyorsa).
- 9) Üye ifadeleri (örn. '**myobj.var**' veya '**myobj.func()**') miras değildi '**unsigned**' sağ tarafın mülkü ('**var**' veya '**func()**') ve diğerleriyle doğrudan karşılaştırılmaz veya birleştirilemez. '**unsigned**' türleri - bir ara '**unsigned**' İlk önce değişken istendi.
- 10) UnoArduSim, bir işlev; tanımında herhangi bir parametrenin bulunması durumunda ısrar ediyordu varsayılan başlatıcı ile birlikte, işlev'ün daha önce bildirilmiş bir prototip olduğunu belirtir.
- 11) Aramalar '**print(byte bvar, base)**' yanlışlıkla terfi '**bvar**' bir '**unsigned long**', ve bu yüzden çok fazla rakam yazdırıldı.
- 12) '**String(unsigned var)**' ve '**concat(unsigned var)**' ve operatörler '**+=(unsigned)**' ve '**+(unsigned)**' yanlış oluşturulmuş '**signed**' yerine dizeleri.
- 13) Bir yüklenen bir 'R=1K' cihaz **IODevices.txt** dosya pozisyonlu '**U**' yanlışlıkla kaydırıcısını (her zaman) içinde çizdi **zıt konum** gerçek elektrik konumundan.



- 14) Temerrüde güvenmeye çalışmak '`inverted=false`' bildirilirken argüman '`SoftwareSerial()`' includes '`struct`' and '`class`'), nesne bir çökmeye neden oldu ve geçti '`inverted=true`' yalnızca program kullanıcısı bir sonraki işlem yaptıysa çalıştı '`digitalWrite(txpin, LOW)`' gerekli boşta kalmayı sağlamak için '`LOW`' seviyesi **Teksas** pin.
- 15) 'I2CSLV' cihazlar, pin düzenleme kutularındaki değişikliklere cevap vermedi (A4 ve A5 varsayılanları geçerli kaldı).
- 16) 'I2CSLV' ve 'SPISLV' cihazlar, fare sınırlarını terk ettiğinde kısmi düzenlemeleri algılamamış ve düzeltmemiştir.
- 17) SPISLV veya SRSLV'yi izleyen cihazlar için Pin değerleri **IODevs.txt** Onaltılık olarak dosya.
- 18) Birden fazla SPISLV cihaz MISO'dan pin'e bağlanmaya çalışmak her zaman bir Elektriksel As in electrical pin-driving conflict), çakışma hatası üretti.
- 19) pin'ü '`OUTPUT`' geri dön '`INPUT`' modu pin veri mandalı seviyesini sıfırlayamadı '`LOW`'.
- 20) kullanma '`sendStop=false`' aramalarda '`Wire.endTransmission()`' veya '`Wire.requestFrom()`' başarısızlığa neden oldu.
- 21) UnoArduSim bir yanlış izin verdi '`SoftwareSerial`' alımı ile aynı anda gerçekleşmesi '`SoftwareSerial`' şanzıman.
- 22) Değişkenler ile ilan edildi '`enum`' türüne, bildirim satırlarından sonra yeni bir değer atanamadı ve UnoArduSim, (legal) ile işaretlendiğinde '`enum`' üyelerini tanıımıyordu '`enumname::`' örnek.

## **V2.2 - Haziran. 2018**

- 1) Tanımının gerektirdiğinden daha az sayıda argüman içeren bir işlev'ü aramak (bu, işlev'ün "ileri tanımlanmış" olduğu, yani daha önce prototip bildirim satırına sahip olmadığı zaman) bellek ihlaline ve çökmesine neden oldu.
- 2) V2.1'den beri **Dalga Şekilleri** sırasında güncellenmemişti **Çalıştır** (yalnızca **Dur**, veya sonra **Adım**) - ek olarak, **Değişken Paneli** uzun süre boyunca değerler güncellenmiyordu **Adım** operasyonlar.
- 3) Bazı küçük **E signal shape**), **dalgabiçimi** V2.0'dan bu yana var olan kaydırma ve yakınlaştırma sorunları giderildi.
- 4) Erken versiyonlarda bile, Reset, t = 0'dan önceki son döngüsünü yapacak olan bir PULSER veya FUNCGEN ile t = 0'da sadece bir ol **kısmi** döngüsü, sonuçlandı **E signal shape**), **dalgabiçimi** sonra t = 0, (sırasıyla) sağa veya sola, bu (veya geri kalan) kesirli çevrim miktarı ile gerçek konumundan kaymaktadır.
- 5) Sözdizimi renk vurgulamasıyla ilgili bazı sorunlar giderildi. **Düzenle/İnceleme** .
- 5) V2.0'dan bu yana, includes '`struct`' and '`class`'), nesneleri dizin'teki icra'e bir includes '`struct`' and '`class`'), nesne tıklatıldığında düzgün çalışmadı.
- 6) '`delay(long)`' olması için düzeltildi '`delay(unsigned long)`' ve '`delayMicroseconds(long)`' olması için düzeltildi '`delayMicroseconds(unsigned int)`' .
- 7) V2.0'dan itibaren, işlevler kullanılarak '`attachInterrupt()`' işlevler'in bu amaç için geçerli olup olmadığı kontrol edilmiyordu (ör. '`void`' dönüş ve hiçbir arama parametresi yok).
- 8) Etkisi '`noInterrupts()`' işlevler tarihinde '`micros()`', '`mills()`', '`delay()`' , '`pulseInLong()`'; üzerindeki etkisi '`Stepper::step()`' ve üzerine '`read()`' ve '`peek()`' zaman aşımaları; tüm RX seri alımlarında ve '`Serial`' şanzıman, şimdi doğru bir şekilde yeniden üretilir.
- 9) Kullanıcı araya girme yordamları içinde harcanan süre şimdi tarafından döndürülen değerde muhasebeleştirilir '`pulseIn()`' tarafından üretilen gecikme '`delayMicroseconds()`' , ve görüntülenen kenarların konumunda **Pin Dijital Dalga Formları** .
- 10) Aramalar **includes '`struct`' and '`class`'**), **nesne üyesi** Daha büyük karmaşık ifadelerin bir parçası olan ya da kendi başlarına birden fazla karmaşık argüman içeren işlev çağrılarının içinde bulunan işlevler, e, g, '`myobj.memberfunc1() + count/2`' veya '`myfunc(myobj.func1(), count/3)`' , hatalı yığın alanı tahsisleri nedeniyle icra zamanında hesaplanan / geçirilen hatalı değerlere sahip olacaktır.
- 11) değişkenler göstergesinin Diziler'si düzgün çalıştı, ancak ekranda gösterilen değerleri hatalı gösterdi. **Değişken Paneli**.
- 12) Basit tipte dinamik diziler ile yaratıldığında '`new`' , sadece ilk eleman 0 değerine (varsayılan) bir başlatma yapıyordu - şimdi tüm elemanlar yapıyor.

- 13) `'noTone()'` , veya sonlu bir tonun sonu, artık pin'ü sıfırlayamaz `'OUTPUT'` ve gider `'LOW'` ).
- 14) Sürekli dönen 'SERVO' cihazlar, 1500 mikrosaniye darbe genişliğinde mükemmel bir şekilde hareket etmektedir.
- 15) SdFile çağırmak :: `Is()` (SD kart dizini listesi) düzgün çalıştı, ancak yanlış şekilde Waveforms pencere'te bazı kopyalanmış SPI transferlerini gösterdi.

### **V2.1.1 - Mar. 2018**

- 1) Dil dışındaki dillerle İngilizce olmayan yerel ayarlardaki tutarsızlıklar giderildi **'myArduPrefs.txt'** görüntülenene radyo dili düğmeleriyle **Tercihler** iletişim kutusunda ve içindeki çevrilmiş çizgilerle eşleştirme **'myArduPrefs.txt'**.
- 2) Tahsisler `'new'` Şimdi sabit olmayan bir tamsayı dizin boyutunu kabul edin.
- 3) Tıklayarak **Değişken Paneli** genişletmek'e çok boyutlu bir dizin'e gereksiz bir boş `'[]'` dirsek çifti .
- 4) Takip eden gereksiz karakterlere sahip Dizin-element referansları (örn. `'y[2]12'` ) Ayrıştır zamanında hata olarak yakalanmadı (fazladan karakterler görmezden gelindi).

### **V2.1 - Mar. 2018**

- 1) Yeni V2.0.x sürümlerindeki bir hata, Pencereler yığınının her güncellemeyle büyümesine neden oldu. **Değişken Paneli** - sonra Milyonlarca güncelleme (birçok dakika icra değerinde) bir çökmeye neden olabilir.
- 2) 'static'e aramalariki nokta üst üste kullanarak üye işlevler `': : '` gösterimde Ayrıştır'e yazım başarısız oldu `'if()'` , `'while()'` , `'for()'` , ve `'switch()'` Parantezler ve işlev çağırısı argümanları veya dizin indeksleri olarak kullanılan ifadelerin içindeyken.

### **V2.0.2 Şubat 2018**

- 1) V2.0'da tanıtılan bir hata **Dosya | Doldurmak** bir çökse `'#include'` eksik veya boş dosya olarak adlandırılır
- 2) İçinde bir **IOdevs.txt** dosya, o `'I/O'` eski 'Oneshot' yerine 'One-Shot' ismi bekleniyordu; ikisi de şimdi kabul edildi.

### **V2.0.1 - Ocak 2018**

- 3) İngilizce olmayan yerlerde `'en'` yanlış olarak seçilen **Tercihler**, İngilizceyi garip hale getirme (seçimin ardından yeniden seçilmesinin gerekli olması).
- 4) Kullanıcının bir Cihaz pin düzenleme kutusu değerini tamamlanmamış bir durumda (`'A_'` gibi) terk etmesi ve bir `'SRS:V'` in `'DATA'` bitlerini terk etmesi mümkün olmuştur.
- 5) Maksimum Analog Sürgü sayısı 4 ile sınırlandırılmıştır (şimdi 6 olarak düzeltilmiştir).
- 6) UnoArduSim artık ısrar ediyor `'='` dizin toplu başlatmada görünür.
- 7) UnoArduSim, "inverted\_logic" argümanının sağlanmasında ısrar etmişti. `'SoftwareSerial()'` .
- 8) Bit kaydırma işlemleri artık kaydırılan değişken'in boyutundan daha uzun vardiyalara izin vermektedir.

### **V2.0- Aralık 2017**

- 1) Olarak bildirilen tüm işlevler `'unsigned'` Yine de sanki değerleri sanki değerleri `'signed'` . Bu etkisi olmadıysa `'return'` değer bir atandı `'unsigned'` değişken, ancak uygunsuz bir sebep olurdu eğer `MSB == 1` olsaydı, olumsuz yorum ve daha sonra bir `'signed'` değişken, veya eşitsizlikte test edilmiştir.
- 2) Analog Sürgüleri sadece maksimuma ulaşıyordu `'analogRead()'` 1022 değeri, doğru 1023 değil.
- 3) Bir hata, istemciden V1.7.0'da, SPI sisteminin SCK pin'ün taşınmasını hızlandırmak için kullanılan mantığı kullanarak geri getirdi. `'SPI_MODE1'` ve `'SPI_MODE3'` ilk bayt aktarıldıktan sonra başarısız olmak (sahte bir ekstra SCK geçişi her baytı izledi). Ayrıca, aktarılan baytlar için bir 'SPISLV' düzenleme 'DATA' kutusundaki güncellemeler ertelendi,
- 4) Renkli LED cihaz, 'B'i (Mavi için) renk seçeneği olarak kabul etmedi (kabul edilse bile).
- 5) 'SPISLV' ve 'I2CSLV' cihazlar ayarları kullanıcıya kaydedilmedi **'I/O' Cihazlar** dosya.

- 6) Kopyalama '**Servo**' örnekler hatalı olduğu için başarısız oldu '**Servo::Servo(Servo &tcopy)**' kopya kurucu uygulaması.
- 7) Menzil dışında '**Servo.writeMicroseconds()**' değerler doğru şekilde bir hata olarak tespit edildi, ancak hata mesajı metnine eşlik eden belirtilen sınır değerler yanlış.
- 8) 115200 yasal baud hızı kabul edilmedi bir cihazdan yüklendiğinde '**I/O Cihazlar**' metin dosya.
- 9) Ekli Analog Kaydırıcı cihaz'ın neden olduğu elektriksel pin çatışmalar her zaman tespit edilmedi.
- 10) Nadir durumlarda, hatalı bir dize işaretçisini (0-terminator dizesi eksikken) bir '**String**' işlev, UnoArduSim'in çökmesine neden olabilir.
- 11) **Kodlama Paneli** vurgulu'daki mevcut Ayrıştır hata satırını **yanlış** program modülü (ne zaman '**#include**' kullanıldı).
- 12) Bir cihaz'e sahip olan bir 'I/O' Cihazlar dosya'nin yüklenmesi, (yanlış) iletmek'in 'Uno' pin'e karşı (program) hata mesajı pop-up'ında asılı kalmasına neden olacaktır.
- 13) UnoArduSim yanlışlıkla izin vermişti kullanıcı altıgen olmayan karakterleri SPISLV ve I2CSLV için genişletilmiş genişletilmiş tamponuna yapıyor.
- 14) Bildirim satırı başlatma sağ taraftaki değer '**return**' includes 'struct' and 'class'), nesne üyesinden işlev'e olan değer (olduğu gibi '**int angle = myservo1.read();**').
- 15) '**static**' üye değişkenler açık olmak '**ClassName::**' bir satırın başlangıcında görüldüğü takdirde örneklerin tanınmaması (örneğin, bir temel atama 'class' değişken),
- 16) çağrı '**delete**' tarafından oluşturulmuş bir işaretçi üzerinde '**new**' sadece işlev parantez notasyonu kullanıldığı gibi kabul edildi. '**delete(pptr)**'.
- 17) UnoArduSim uygulaması '**noTone()**' pin argümanının sağlanması konusunda yanlış ısrar ediyordu.
- 18) Kullanılan bir program'de global 'RAM' baytını arttıran değişiklikler '**String**' değişkenler (üzerinden **Düzenle/İnceleme** veya **Dosya | Doldurmak**), 'Uno'un küresel silinmesinde yığın silinmesi nedeniyle yolsuzluğa yol açabilir. '**String**' Eski program'ye ait includes 'struct' and 'class'), nesneleri, yeni program'ye ait yığın (yanlış) kullanırken. Bazı durumlarda bu program kazasına neden olabilir. İkinci bir Doldurmak veya Ayrıştır sorunu çözmesine rağmen, bu hata sonunda düzeltildi.
- 19) İçin dönüş değerleri '**Wire.endTransmission()**' ve '**Wire.requestFrom()**' ikisi de 0'da sıkışmıştı - bunlar şimdi düzeltildi.

### **V1.7.2- Şubat. 2017**

- 1) pin 2'deki kesinti de (istemeyerek) pin 3'teki (ve tersi) sinyal aktivitesi ile tetiklendi.

### **V1.7.1- Şubat. 2017**

- 1) İşlev '**delayMicroseconds()**' bir gecikme üretiyordu **milli**-saniye (1000 kez çok büyük).
- 2) Açık tiplere '**unsigned**' Daha uzun bir tamsayı tipine değişken yanlış verdi ('**signed**') sonuç.
- 3) 0'dan büyük onaltı harfler **7FFF** şimdi '**long**' tanımı gereği, ve böylece şimdi üretecek '**long**' Elde ettikleri aritmetik ifadeler.
- 4) İstmeden V1.7.0 tarafından sunulan bir hata, sayısal değişmezlerin alternatif C ++ stil tiplere'ünü engelledi (örneğin, '**(long) 1000\*3000**' kabul edilmedi).
- 5) '**Serial**' program kullanıcısı tarafından hiç gerekmediğinde, artık 'Uno' RAM'deki birçok baytı kullanmaz.
- 6) Kullanıcı tarafından beyan edilen global değişkenler, hiç kullanılmamışlarsa, 'Uno' RAM'de artık yer kaplamaz
- 7) Bekar değişkenler olarak ilan edildi '**const**', '**enum**' üyeler, ve değişmezleri dizmek için işaretçiler, 'Uno' RAM'de artık yer kaplamaz (Arduino derlemesi ile aynı fikirde olmak için),
- 8) İçin RAM bayt gerekli '**#include**' yerleşik kütüphaneler artık Arduino koşullu derleme sonuçlarıyla yakından eşleşiyor.
- 9) kullanma '**new**' imleç üzerine gerçek bildirim satırı başarısız oldu (yalnızca bir sonraki '**new**' işaretçiye atama

çalıştı).

10) Bir SD disk dizininin “beklemede” gösterisinin program'nin askıda kalmasına neden olabileceği bir hata düzeltildi.

### **V1.7.0– Aralık 2016**

0) Kullanıcı kesintilerinin ele alınmasıyla ilgili bir takım sorunlar şimdi düzeltildi:

a) Bir Arduino işlev sırasında meydana gelen 0 ve 1 kenarlarını keser, beklerken bloklar '**pulseIn()**', '**shiftIn()**', '**SPI.transfer()**', '**flush()**', ve '**write()**' kesinti sırasında icra akışında hataya neden oldu

b) Kesilen işlev'ün yerel değişkenler'sinin birden fazla kopyası görüldü **Değişken Paneli** (kesme-dönüş başına bir kopya) ve bu V1.6.3'te düzeltildi, ancak diğer kesme sorunları kaldı).

c) İşlev '**delayMicroseconds()**' bir kullanıcı kesme yordamının içinden çağrıldığında herhangi bir gecikme yaratmıyordu.

d) İşlevler benzeri engelleme çağrıları '**pulseIn()**' itibaren **içeride** bir kesme rutin çalışmıyordu.

1) V1.6.3'de sunulan hata'de değer güncellemesi **Değişken Paneli** değerler gerçekte değiştiğinde yürütülürken (bu yalnızca iki veya daha fazla sürdü) **Dur** veya menü **VarYenile** kullanıcı eylemleri). Ayrıca, bir Komuta Çalıştır yapıldıktan sonra **Azaltmaya İzin Ver** tetiklenmişti **Değişken Paneli** bazen yeniden çizmedi (bu yüzden eski değerler ve yerel-değişkenler, bir sonraki Adım'e kadar orada görünmüş olabilir).

2) The **Kodlama Paneli** davranışını vurgulama **Etrafında Adım** komut içinde yanıltıcı görünebilir '**if()** -**else**' zincirler - bu şimdi düzeltildi (gerçek adım işlevi doğru olsa da).

3) İşlev '**pulseIn()**' zaman aşımını mikrosaniye yerine milisaniye cinsinden ayarlamamış; ayrıca, etkin olmayan ve aktif seviyelere geçiş ilk kez görülüyorsa, zaman aşımını da yanlış bir şekilde başlatıyordu.

4) HEX değişmezlerini kullanma 0x8000 ve 0xFFFF ödevlerde veya aritmetikte '**long**' değişkenler tamsayısı nedeniyle yanlış sonuçlar verdi işaretlenmemiş işaret uzantısı.

5) A'ya geçmek veya '**float**' herhangi birinden '**unsigned**' değeri MSB = 1 olan tamsayı türü hatalı bir sonuç nedeniyle hatalı sonuçlar verdi '**signed**' yorumlama.

6) Herşey '**bit\_()**' işlevler şimdi de işlemlerini kabul ediyor '**long**' -değişkenler boyutunda ve UnoArduSim geçersiz bit konumları için test eder (değişken boyutunun dışına çıkar).

7) Bir 'PULSER' Cihaz'daki 'Pulse' (genişlik) düzenleme kutusuna geçersiz bir giriş, 'Period' değerinin bozulmasına neden oldu (bir sonraki kullanıcı 'Period' düzenleme girişi tarafından belirlenene kadar).

8) Kullanarak bir 'PULSER' veya 'FUNCEN' cihaz'i silmek menü Yapılandır, periyodik sinyali, sürmekte olduğu pin'den çıkarmıyordu (artık Reset gerekli değil).

9) Bir 1-D başlatma yeteneği '**char**' Alıntı yapılan bir dize ile izin eksikti (ör. '**char strg[] = "hello";**')

10) genişletilmiş 'SERIAL' veya 'SFTSER' Monitöründeki onaltılı ekran pencereler, 127'den büyük bayt değerleri için yanlış en önemli karakteri gösterdi.

11) E signal shape), dalgabiçimi pencereler, kullanıcı tarafından yapılan kullanıcı programındaki değişiklikleri yansıtmıyordu. '**analogWrite()**' ne zaman yeni bir değer ya% 0 ya da% 100 görev döngüsü idi.

12) Uygulanması '**Serial.end()**' şimdi düzeltildi.

13) bir '**myArduPrefs.txt**' Tek satırda 3 kelimedenden fazla olan dosya (veya boşluktaki '**I/O**' Cihazlar dosya ismi) hatalı bir dahili işaretçi nedeniyle çökmeye neden olabilir.

14) Son satırında '**I/O**' Cihazlar dosya kabul edilmediyse kabul edilmedi **satır beslemeli bitmek**.

15) Dörtten fazla Analog Sürgüsü eklemek, LED 'I/O' cihaz işaretçilerinin üzerine yazılan sessiz bir hata'e neden oldu

16) V1.6.0 ile başlayan, analog e signal shape), dalgabiçimi **ilk yarı** her biri için **üçgen** e signal shape), dalgabiçimi hepsi **sıfır** (e signal shape), dalgabiçimi-tablo hesaplamasındaki hata nedeniyle).

17) Tekrarlanan yapmak **Komuta Çalıştır** kesim noktası hattında artık avans başına birden fazla tıklama

gerektirmediginde.

18) iřlev dizin parametresine adres ifadelerinin iletilmesi Ayırıştırıcı tarafından kabul edilmedi.

19) İřaretçi veya dizin referanslarını ieren ifadeleri döndüren özyinelemeli iřlevler, bu bileřen ifadelerindeki sıfırlanmamış "hazır" bayraklarından dolayı yanlış sonuçlar verdi.

20) çağrı 'class' üye-iřlevler ile **herhangi bir includes 'struct' and 'class'), nesne iřaretçisi deėiřken veya iřaretçi ifadesi** alıřmıyordu.

21) includes 'struct' and 'class'), nesneleri by-value deėerini döndüren iřlevler kullanıcısı, ilk iřlev çağrısında deėerlerini yalnızca başarıyla döndürdüler **Eėer** isimsiz bir includes 'struct' and 'class'), nesne inşa ettiler (gibi 'String("dog")' - sonraki aramalarda, sıkıřmış bir "hazır" bayrak nedeniyle geri dönüş atlandı.

22) Komutu önlemek için bir güvence olmamıştır. **Pencereler | 'Serial' Monitör** from yeni ekleyerek 'SERIAL' cihaz, aslında buna yer kalmadıėında.

23) Sabit bir pin cihaz ('SPISLV' gibi) eklenmesi pin as in electrical pin-driving conflict), akıřma açılır mesajına neden olmuřsa, **Laboratuvar Tezgah Paneli** yeniden izim yinelenen bir "hayalet" cihaz gösterebilir en saėdaki 'I/O' cihaz'i (sonraki yeniden izmeye kadar) kaplamak.

24) Periyodik olmayan pin sinyalleri için güvenilirmez 'PIEZO' sesleriyle ilgili bazı sorunlar giderildi.

25) 'PROGMEM' deėiřkenler řimdi açıka açıka ilan edilmiş olmalıdır 'const' Arduino ile aynı fikirdeyim.

26) "Hibir yığın alanı yok" yanlış bir icra hatası olarak iřaretlendi. 'SD.open()' adlı dosya'yi bulamıyor 'openNextFile()' Rehberdeki son dosya'ye ulařtı.

27) Ayırıştırıcı hata uygunsuz bir yerde süslü ayra'i kabul etmemiřti '}'.

28) Bir hata ile **Deėiřken Paneli** üye includes 'struct' and 'class'), nesne-yapıcı iadesi üzerine yapılan kaldırmalar düzeltildi (hata, yalnızca üyeler olarak bařka includes 'struct' and 'class'), nesneleri ieren includes 'struct' and 'class'), nesneleri için bařvuruda bulundu).

### **V1.6.3 - Eylül 2016**

1) Kesilen iřlev'ün yerel deėiřkenler'si **Deėiřken Paneli** iřlev giriři kesildiėinde, birden fazla kopyaya yol açabilir Kesin iřlev iadesinde ortaya ıkıyor (ve muhtemel bir icra hatası veya ökmesi).

2) E signal shape), dalgabiimi pencereler'teki programatik deėiřiklikleri yansıtmıyordu. 'analogWrite()' % 0 veya% 100 olan yeni bir görev döngüsüne.

3) genişletilmiş 'SERIAL' veya 'SFTSER' Monitöründeki onaltılı ekran pencere, 127'den büyük bayt deėerleri için yanlış MSB karakteri gösterdi.

### **V1.6.2 - Eylül 2016**

1) Yanlış sayıda veya tür argümanlarla yapılan iřlev aėrıları uygun bir Ayırıştırıcı hata mesajı vermedi (yalnızca jenerik "geerli bir tanımlayıcı deėil" mesajı belirdi).

2) The **Aracı-Bar** sıfırlama düėmesi řimdi 'Uno' kartı sıfırlama düėmesine aynı řekilde alıřır.

3) Ayırıştırıcı-hata metni artık bir üç nokta göstermeden 16 karakterden sonra kesilmez.

### **V1.6.1 - Aėustos 2016**

1) V1.6'da, 'Uno' kartı'nın 'myArduPrefs.txt' Varsayılan sürüm 2 deėerinden farklı olan dosya, bařlangıta bir özel duruma neden oldu (bařlatılmamış bir pin 13 olayı nedeniyle).

2) deėiřken'in deėerini, üzerine çift tıklayarak **Deėiřken Paneli** hatalı "bellek ayırma yok" hata pop-up'larına neden olabilir (kullanıcı tanımlı herhangi bir programlar için 'class').

3) 'SoftwareSerial' eriřimine izin verilmedi 'write(char\* ptr)' ve 'write(byte\* ptr, int size)' iřlevler hatalı iřlev ařırı yük algılaması nedeniyle.

4) Yalıtılmış bir ".h" kütüphanesi için karřılık gelen ".cpp" dosya'nin otomatik olarak dahil edilmesiyle ilgili sorun düzeltildi. '#include'.

## **V1.6 - Haziran 2016**

- 1) V1.5'te 'Enter' anahtarındaki otomatik girinti **Düzenle/İnceleme** (yeni bir satıra girerken) kaybedilmişti.
- 2) pin ile çatışmak ekli güçlü iletkenli harici 'I/O' cihazlar tespiti eklenmiştir 'Serial' pin 1, SPI pins SS, MOSI ve SCK, I2C pins SCL ve SDA'da (hepsi ilgili olduğunda 'begin()' denir), ve herhangi bir ilan üzerine 'SoftwareSerial' TX pin.

## **V1.5.1 - Haziran 2016**

- 1) V1.5'te Temaya uyarlanabilir yeni sözdizimi Vurgulu renkleri her seferinde uygun şekilde sıfırlanmadı **Düzenle/İnceleme** açıldı ve böylece (beyaz arkaplan temalı) her saniye sadece doğru oldu.
- 2) Kesmek 'RISING' ve 'FALLING' hassasiyetler olmuştu karşısında gerçek tetikleyici kenar polaritesine.

## **V1.5 - Mayıs 2016**

- 1) V1.4.1'de tanıtılan bir hata, çıplak dizge değişmezlerinin beklenen işlevler üyesi 'String' includes 'struct' and 'class'), nesne, olduğu gibi 'mystring1.startsWith("Hey")' .
- 2) Orijinalde bir hata SD UnoArduSim uygulamasına yalnızca izin verilir SD çağrılarını kullanarak eriş 'read()' ve 'write()' (üzerinden erişim 'Stream' işlevler önlenmiştir).
- 3) 'R=1K' Sürgü hareket ettirildiğinde sürgü anahtarları doğru şekilde çizilmiyordu.
- 4) **Temizle** Confirm-Kaydet dosya diyalog kutusunda uygulama çıkışını engellemeliydi.
- 5) Eksik bir kapanış teklifi veya kapanış '>' -parantez kullanıcı dosya'de '#include' takılmaya neden olur.
- 6) Bir hata'nın sözdizimi vurgulaması düzeltildi 'String' ve kullanıcı 'class' veya 'struct' ve dahil etmek için genişletilmiş vurgulama yapıcı işlev çağırır.
- 7) Küçük sorunlar düzeltildi **Düzenle/İnceleme** metin değişiklikleri / vurgulama ve **Geri** buton.

## **V1.4.3 - Nisan 2016**

- 1) kullanma **Yapılandır | I/O Cihazlar** yeni cihazlar'yi eklemek ve daha sonra yeni eklenen cihazlar'den birini çıkarmak için sıfırlama sırasında bir çökmeye neden olabilir veya başka bir cihaz'ın çalışmasını durdurabilir.
- 2) Bir değiştirme 'String' değişken çift tıklayarak **Değişken Paneli** başarısız (yeni 'String' yanlış okundu).
- 3) 'FUNCGEN' ve 'PULSER' cihazlar'deki Pin değişiklikleri, sıfırlama ilk yapılamaya kadar tanınmadı.

## **V1.4.2 - Mart 2016**

- 1) V1.4.1 vardı **tanıtılan** talihsiz bir Ayırıştır hata herhangi birini içeren atamaları engelledi 'class' includes 'struct' and 'class'), nesneleri (dahil 'String' ).
- 2) V1.4.1'de yapılan eksik bir hata düzeltmesi 'unsigned' tür değerleri 'char' tamsayı değerleri yerine ASCII karakterleri olarak yazdırmak için.
- 3) Karmaşık üye ifadesi işlev çağrı argümanları her zaman geçerli işlev parametresi eşleşmeleri olarak tanınmadı.
- 4) **Herşey** tamsayı değişmezleri ve ifadeleri çok cömertçe 'long' ve bu nedenle icra'ın **gerçek** Arduino'da meydana gelebilecek ekleme / çarpma işlemlerinde ortaya çıkabilecek taşma (negatif) 'int' büyüklükteki değerler
- 5) Bir karışımını içeren ifadeler 'signed' ve 'unsigned' tamsayı türleri her zaman doğru şekilde işlenmedi 'signed' değer yanlış olarak görülecektir 'unsigned' ).
- 6) pin-as in electrical pin-driving conflict), çakışma durumlarda, "value =" hata mesajları, kullanıcının önceden temizlenmiş olduğu bir as in electrical pin-driving conflict), çakışma'dan gelen Reset'den sonra bile eski pin değerleri gösterebilir.

### **V1.4.1 - Ocak 2016**

- 1) Aramalar '`print(char)`' Şimdi ASCII karakterleri gibi düzgün yazdırın (sayısal değerler yerine).
- 2) Kesinti yanıtı şimdi varsayılan olarak etkin olduğunda '`attachInterrupt()`' denir, bu yüzden artık sizin herhangi bir ihtiyaç yoktur '`setup()`' etkinleştirici işlev'ü aramak için '`interrupts()`' .
- 3) Çoklu '`#include`' kullanıcı-dosyalar örnekleri İçinde bir dosya şimdi doğru bir şekilde işleniyor.

### **V1.4 - Aralık 2015**

- 1) bir **sürüncemede kalan** hata yanlış bir şekilde işaretlendi. **sıfıra bölme** birlikten kesirli bir değere bölünürken durum.
- 2) Sabit '`SoftwareSerial`' (istemeden eklenenler tarafından kırıldı '`class`' -V1.3'te üye doğrulama kontrolü bülteni).
- 3) Hat sonu işlev eksik noktalı virgüllü aramalar yakalanmadı ve Ayırıştırıcı'nın bir sonraki satırı atlamasına neden oldu.
- 4) Kötü biçimlendirilmiş '**I/O** Cihazlar' metin dosya yanlış bir hata mesajı verdi.
- 5) Ayırıştır Çok satırlı ifadelerde ve ifadelerde hatalı (bitişik) satırın vurgulanması sorunu giderildi
- 6) İşaretçiler kullanılarak yapılan mantıksal testler '`not`' ( ! ) operatör ters çevrildi.

### **V1.3 - Ekim 2015**

- 1) Not defteri değişkenler'nin yanlış dahili kullanımı, zaman zaman neden oldu " **maksimum karalama defteri yuvalama derinliği aşıldı** "Ayırıştır hataları.
- 2) Parantez *tek tırnak içinde*, süslü ayraç, noktalı virgül, parentheses alıntı yapılan dizelerin içinde ve kaçan karakterler yanlış işlendi.
- 3) Boş boyutlu ve başlangıç listesi olmayan bir Dizin, RESET askıda kalmasına neden oldu ve sadece tek bir eleman içeren diziler'e izin verilmedi (ve hatalı bir şekilde başlatılmış işaretçi olarak yanlış yorumlanmasına neden oldu).
- 4) Ayırıştır hataları bazen vurgulu'ya yanlış (bitişik) çizgi koyar.
- 5) İşaretçiyi '`const`' olmayan bir kişiye geçirme işaretçisi kabul eden bir işlev '`const`' (tersine çevrilmek yerine) yasaklanmıştı.
- 6) Başlatma ifadeleri yanlış bir şekilde miras alıyordu '`PROGMEM`' değişken'ten gelen elemeler başlatıldı
- 7) '`PROGMEM`' değişkenler'nin byte boyutunun yanlış olarak sayıldığını **iki defa** Ayırıştır işlemi sırasında 'Flash' bellek paylaşımına karşı.
- 8) 'Send' düzenleme kutusuna yazarak bir 'I2CSLV' bazen nedeniyle çökmeye neden olur '`sscanf`' hata.
- 9) Yeni bir 'I/O' olan yeni bir program yükleniyor **Cihazlar** Dizindeki dosya ile alakasız pin ile çatışmak **eski** pin yol tarifi.
- 10) Çıkış karakter dizisi kullanımı, (karakterlerin) içindeki karakter dizilerinin iletilmesinden çok, alınanlara hatalı şekilde uygulandı. '**Serial** Monitör' tamponlar pencere.
- 11) '`while()`' ve '`for()`' tamamen boş gövdeli halkalar, örneğin '`while(true);`' veya '`for(int k=1;k<=100;k++)`' ; ' Ayırıştırıcı'yi (bir uyarı mesajı ile) geçti ancak icra zamanında başarısız oldu.

### **V1.2 - Haziran 2015**

- 1) Ya çağrı yapan en basit kullanıcı işlevler '`digitalRead()`' veya '`analogRead()`' veya '`bit()`' (önce), tahsis edilen işlev kazı kazan alanı yetersiz olduğundan (varsa işlev yığınının başında sadece iki kazı kazan baytı tahsis edildiyse) yerel değişken'i (varsa) bozmuş olabilir. işlev'ün içindeki herhangi bir sayısal ifade, 4 baytlık bir karalama defteri tahsisine neden olmak için yeterlidir ve bu nedenle bu sorunu önler. Bu talihsiz hata, orijinal V1.0 sürümünden bu yana hizmet vermektedir.
- 2) İşlevler '`void`' açık bir şekilde '`return`' ve birden fazla '`void`' işlevler olmayan birden fazla '`return`' Açıklamada, icra'de düşüş yaşanacağını **süslü ayraç kapanıyor** (ulaşıldıysa).
- 3) herhangi '`return`' içindeki ifadeler '`if()`' süslü ayraç eksik olan içerikler, arayanlara geri dönüş hedefinin hatalı olmasına neden olmuştur.

- 4) **'PULSER'** ve **'FUNCGEN'** darbe genişlikleri veya 0 değeri periyotları, çarpımlara neden olabilir (0 artık izin verilmez).
- 5) süslü ayraç'nin olmadığı yerde, **'else'** bir sonra devam **'if ()'** izledilerse işe yaramadı **'break'** , **'continue'** veya **'return'** .
- 6) Ne zaman **çoklu 'enum'** **Kullanıcı tarafından beyanlar yapıldı** , ilkinde tanımlanan ama sabit olan ilk **'enum'** hatalı üretilen " **'enum'** uyumsuzluk" Ayırıştır hataları (bu hata, V1.1'de tanıtıldı).
- 7) işlev prototip'ün son parametresi için boş bir tanımlayıcı Ayırıştır hatasına neden oldu.
- 8) **Komuta Çalıştır** Karmaşık hatlarda ayarlanan kesme noktaları her zaman doğru bir şekilde ele alınmamıştır (ve bu yüzden kaçırılmış olabilir).
- 9) **'HardwareSerial'** ve **'SoftwareSerial'** Reset'de temizlenmemiş özel bir uygulama TX beklemede olan tampon kullandı (bu nedenle son zamandan kalan karakterler görünebilir).
- 10) Ayırıştırıcı, yasadışı bit çevirme kontrolü yapamadı **'float'** ve işaretçi aritmetik geçersiz operatörlerle denendi.

## **V1.1 - Mart 2015**

- 1) Dizin endeksleri **'byte'** veya **'char'** değişkenler büyüklüğü hatalı dizin ofsetlerine neden oldu (bitişik bir değişken 0 olmayan bir yüksek bayt içeriyorsa).
- 2) İşaretçilerin mantıksal testi, işaretçi değerinin kendisinden ziyade sivri olmayan değeri sınamayanlar için test etti.
- 3) herhangi **'return'** içine gömülü ifadeler **'for ()'** veya **'while ()'** döngüler yanlış işlendi.
- 4) includes **'struct'** and **'class'**), nesneleri diziler için toplu başlatma listeleri veya diğer includes **'struct'** and **'class'**), nesneleri / dizileri içeren includes **'struct'** and **'class'**), nesneleri veya tamamen boş başlatma listeleri uygun şekilde kullanılmamıştır.
- 5) Erişim **'enum'** kullanarak üye değerleri **'enumname::'** örnek desteklenmedi.
- 6) Beyan çizgisi başlatma **'char [ ]'** Alıntı yapılmış bir dize ile dizin çalışmıyordu.
- 7) Önceden başlatma yapılmadan işlev'e geçirilen bir dizin, "kullanılmış ancak başlatılmadı" hatasıyla yanlış işaretlendi.
- 8) dizin adlarını içeren işaretçi ifadeleri yanlış kullanılmış.
- 9) Olarak bildirilen işlev parametreleri **'const'** kabul edilmedi.
- 10) Pin Analog Dalga Formu pencere PWM sinyallerini göstermedi ( **'servo.write ()'** ve **'analogWrite ()'** ).
- 11) includes **'struct'** and **'class'**), nesne işaretçisiyle erişilen üye işlevler hatalı üye erişimi verdi.
- 12) Dalga formları, bir **Komuta Çalıştır** kesim noktası'na ulaşıldı.
- 13) Bir işlev parametresi başka bir işlev çağrısına doğrudan bir argüman olarak kullanıldığında, bir işlev parametresi kullanıldığında kayıt tahsisi modellemesi başarısız olabilir

## **V1.0.2 - Ağustos 2014**

'Uno' kartı'nın çevresine A0-A5 pins'in sabit siparişi.

## **V1.0.1 - Haziran. 2014**

Orijinal program'deki bayt sayısının üç katından daha uzun olan düzenleme pastalarını kesen bir hata düzeltildi.

## **V1.0 - ilk sürüm Mayıs 2014**



## Değişiklikler / İyileştirmeler

### V2.8.0- Haziran 2020

- 1) Yeni 'Mega2560' kartı eklendi **Tercihler** seçenek (Kartı numarası == 10). Bu özellikler çok daha fazla 'I/O' pins, 4 daha fazla harici kesinti (pins 18, 19, 20, 21), üç daha 'HardwareSerial' portlar (pins 22-27'de) ve RAM belleği 2 KByte'den 8 Kbyte'a çıkarıldı.
- 2) 'SFTSER' cihazı 'ALTSER' olarak yeniden adlandırıldı (çünkü artık 'Seri1', 'Seri2' ve 'Seri3' ile de kullanılabilir).
- 3) 'PROGIO' cihaz artık 4 ana pin numarasının isteğe bağlı bir takip listesinin 4 'Uno' köle kartı pins ile 4 master ('Uno' veya 'Mega') kartı pins ile açık eşleme.
- 4) cihaz pin düzenleme kutusunun içine tıklamak artık girişi kolaylaştırmak için tam sayıyı seçer ve Yapılandır | 'I/O' Cihazlar'daki cihazlar-sayı düzenleme kutusunun içine tıklamak da aynı şeyi yapar.
- 5) Ayırıştırma ve Yürütme uyarı açılır pencereleri için ilgili kaynak satırının turuncu vurgulaması eklendi.
- 6) İçin destek eklendi 'digitalPinToInterrupt()' çağırır.
- 7) Sınıflar artık kullanıcı tanımlı kurucuları yoksa her zaman varsayılan bir kurucu alır (temel sınıfları veya includes 'struct' and 'class'), nesne üyeleri olmasa bile).

### V2.7.0- Mart 2020

- 1) (Hata varsa kırmızı çalışmaya hazır ise yeşil) geçerli kod-line ek olarak, UnoArduSim şimdi, her modül için, (koyu zeytin arka plan ile vurgulanmıştır) son kullanıcı tıklandığında veya yığın-navigasyon kod satırı, verme korur o sete daha kolay ve bulmak geçici kesim noktası hatları (modül başına bir anda izin verilir, ancak şu anda görüntülenen modülünde tek bir 'Run-To' de yürürlükte).
- 2) 'SPI' ve 'I2C' dahil olmak üzere yeni 'I/O' cihazlar (ve destekleyici 3. taraf kütüphane kodu) eklendi **Liman Genişleticiler** Ve 'SPI' ve 'I2C' **Çoklayıcı LED** Kontrol ve gösterge elemanları (LED diziler, 4-alfanümerik ve 4-sayı ya da 8-sayı 7 kademeli görüntüler).
- 3) 'Wire' operasyonlar artık (bu destekleri bir 'I2C' Liman Expander'dan harici kesmeler) rutinleri kesme iç kullanıcıdan izin verilmez.
- 4) Dijital dalga artık (arasında bir ara seviyesini gösteren 'HIGH' ve 'LOW') pin itilmiş olmak değilken.
- 5) Tek üzerinde ilerlerken hangi karışıklığı önlemek için 'SPI.transfer()' talimatlar, UnoArduSim şimdi ekli 'I/O' cihazlar şimdi işlev dönmeden önce onların (mantık gecikmeli) nihai 'SCK' saat kenarını almak emin kılar.
- 6) Ne zaman otomatik sekme biçimlendirme **Tercih** etkin olduğu bir kapanış süslü ayraç yazarak '}' içinde **Düzenle/İnceleme** şimdi eşleştirme açma-süslü ayraç sekme girinti pozisyonuna bir sıçrama neden olur '{' ortak.
- 7) bir **Yeniden Biçim** düğmesine eklendi **Düzenle/İnceleme** (Neden hızlı otomatik sekme girinti yeniden biçimlendirme) - Otomatik sekme girinti Tercihi etkinden bu düğme yalnızca etkindir.
- 8) Bir daha net hata mesajı artık (gibi zaman Önek anahtar kelime oluşuyor 'const', 'unsigned', veya 'PROGMEM') bir bildiriminde bir tanımlayıcı (o) tanımlayıcısı önce gerekmektedir izler.
- 9) Initialized küresel değişkenler, daha sonra hiç kullanılmamış bile, şimdi hep bir bellek adresi atanır ve böylece görünür görünecektir.

### V2.6.0 Ocak 2020

- 1) Eklenen karakter LCD ekran cihazlar 'SPI', 'I2C' ve 4-bi-paralel arayüze sahip. Destekleyici kütüphane kaynak kodu, yeni yükleme 'include\_3rdParty' klasöre eklendi (ve normal kullanılarak erişilebilir '#include' yönerge) - Kullanıcıların yerine alternatif iletmek LCD cihaz kendi işlevler yazmayı tercih edebilir.
- 2) **Kodlama Paneli** vurgulama bir hata kodu-line için hazır bir kod-line için ayrı vurgulu renkleri ile, geliştirilmiş, ve diğer herhangi bir kod-hattı için edilmiştir.

3) **Bul** Menü ve 'func' eylemler yerine artık önceki / olarak artık atlama işlev hattını itibaren sonraki (ve bir sonraki aşağı yukarı önceki) ve yükselmesine aracı-bar (veya alçalma) çağrı yığını, arayan ilgili kod satır seçtikten (ya da adlandırılır) işlev sırasıyla burada **Değişken Paneli** İçerik, işaretlenmiş olan kod hattını içeren işlev için değişkenler göstermek için ayarlanır.

4) konfüzyon, a önlemek için **Kaydet** Tamamlandı içeride **Düzenle/İnceleme** hemen bir ayırd edici güc neden olurDerleVe eğer Kaydet Bir sonraki kullanarak başarılı oldu Temizle veya Kapat şimdi sadece metin döner Bu son kaydedilen metne.

5) bir atımlı giriş Kademe Motoru ('PSTEPR') ile 'STEP' (darbe), 'EN\*' (etkin) ve 'DIR' (yön) girişi ve bir mikro-adım-adım başına ortam (1,2,4,8, ya da 16) eklendi .

6) (Bir fazla adım cevher tarafından kapalı iken senkronize için YEŞİL veya KIRMIZI.) 'STEPR' ve 'PSTEPR' cihazlar Hem artık 'sync' LED var

7) 'PULSER' cihazlar şimdi 'Period' ve 'Pulse' için mikrosaniye ve milisaniye arasında bir seçim var.

8) Dahili-işlev otomatik tamamlamalar artık parametre adının önünde parametre türünü korur.

9) Bir önceki geri geçiş yaparken **Kodlama Paneli**, Onun daha önce vurgulanan hattı hemen yeniden vurgulanır.

10) dan Temizle veya Kapat, geçici kırılma noktasını ayarlamak kullanmaya bir yardım olarak **Düzenle/İnceleme** vurgulu yaprakları **Kodlama Paneli** satırdaki son imlecin içinde tarafından ziyaret **Düzenle/İnceleme**.

11) Bir kullanıcı tarafından tanımlanan (ya da 3. taraf) '**class**' Şimdi kullanmasına izin verilir '**Print**' veya '**Stream**' baz sınıf olarak. Bu destek, yeni bir 'include\_Sys' klasörü (UnoArduSim yükleme klasöründe) eklenmiştir her tabanı için kaynak kodu sağladığı '**class**'. Bu durumda, bu tür base çağrıları '**class**' işlevler kullanıcıya aynı şekilde muamele edilecektir kodu (bu) basan edilebilir yerine örneğin (adım atan edilemez bir dahili işlev olarak '**Serial.print()**').

12) Üye-işlev otomatik tamamlamalar şimdi parametre adını içerir **onun yerine** onun türü.

13) UnoArduSim Ayrıştırır hemen ihtiyari (ve uygun) ile başlar edilecek bir değişken bildiriminde bir includes 'struct' and 'class'), nesne adı sağlar '**struct**' ya 'class' kelime, ardından '**struct**' veya '**class**' adlandırın.

## **V2.5.0 Ekim 2019**

1) desteği eklendi '**TFT.h**' hariç kitaplığı ( '**drawBitmap()**' ) Ve ilişkili bir 'TFT' ilave 'I/O' Cihaz (128 160 piksel). sırayla büyük sırasında aşırı gerçek zamanlı gecikme önlemek için Not '**fillXXX()**' Transfer, 'SPI' transferlerin sa kısmı **dolgu ortasında** 'SPI' otobüsün devamsızlık olacaktır.

2) 'SD', a yoluyla büyük dosya aktarımları sırasında bayt sekansının ortasında 'SPI' transfer kısmı Benzer 'SPI' otobüs mevcut olacaktır.

3) azalmış '**Stream**' -kullanım havai böylece bayt '**RAM free**' değeri daha yakından Arduino derleme eşleşir.

4) UnoArduSim şimdi kullanıcıyı uyarır zaman '**class**' sahip birden fazla üye bir deklarasyon hattında ilan etti.

5) 'File | Save As' kullanmaşimdi bu kaydedilmiş-içine bu dizinin geçerli dizini.

6) eksik iki '**remove()**' üyesi işlevler eklendi '**String**' sınıf.

7) Bir kurucu-işlev prototip içinde UnoArduSim hemen olanak vermez yapıcı baz çağrıları tam işlev vücut tanımı hemen (çok çaba isteyen derleyici ile anlaşmak gibi) aşağıdaki edilmiştir.

8) EdGE geçiş süresi dijital dalga şekilleri en yüksek zum ayarında hızlı 'SPI' sinyallerinin canlandırmayı sağlayacak şekilde düşürülmüştür.

9) un oArduSim şimdi bazı kurucular ilan edilmesini sağlar '**private**' veya '**protected**' (İç sınıf

kullanım için).

## **V2.4 Mayıs 2019**

- 1) Tüm 'I/O' cihaz dosyaları şimdi dile çevrilmiş biçimde ve **Tercihler** dosyasıyla birlikte, sonraki okumalarda eşleşmeyen hataları önlemek için hepsi artık UTF-8 metin kodlamasına kaydedilmiştir.
- 2) Yeni eklendi '**PROGIO**' Çıplak programlanabilir bir köle olan Cihaz ile ortak olarak 4 pins'e kadar paylaşan 'Uno' kartı **Laboratuvar Tezgah Paneli** master 'Uno', - bir köle 'Uno' sahip olabilir **yok hayır** 'I/O' cihazlar'ye aittir.
- 3) Şimdi 'Ctrl' cihaz'ı, 'Ctrl' tuşuna basarken tıklayarak silebilirsiniz.
- 4) İçinde **Düzenle/İnceleme** , genel, yerleşikler ve üye değişkenler ve işlevler için metin otomatik tamamlama özelliği eklendi (tamamlama isteğinde bulunmak için ALT-sağ oku kullanın veya **Girmek** Yerleşikler listesi şu anda eşleşen bir seçimi vurguluyorsa).
- 5) İçinde **Tercihler** yeni bir seçim, satır sonlandırıcı bir noktalı virgölün n üzerine otomatik olarak eklenmesini sağlar. **Girmek** tuşa basıldığında (geçerli satır bağımsız ve eksiksiz görünen yürütülebilir bir ifade ise).
- 6) basma '**Ctrl-S**' bir **E signal shape**), **dalgabiçimi** pencere tümü bir dosya'ye kaydetmenizi sağlar (**X, Y** e signal shape), dalgabiçimi'un görüntülenen bölümündeki noktaları gösterir (X, en soldaki e signal shape), dalgabiçimi'dan gelen mikrosaniye gelin ve Y voltur).
- 7) Bir 'SFTSER' cihaz şimdi bir gizli (isteğe bağlı) var '**inverted**' değer (**hem TX hem de RX için geçerlidir**) baud hızı değerinden sonra satırının sonuna eklenebilir. **IODevs.txt** dosya.
- 8) Eklendi '**SPISettings**' sınıf, işlevler '**SPI.transfer16()**' , '**SPI.transfer(byte\* buf, int count)**' , '**SPI.beginTransaction()**' , ve '**SPI.endTransaction()**' , Hem de '**SPI.usingInterrupt()**' ve '**SPI.notUsingInterrupt()**' .
- 9) SPI kütüphanesi işlevler eklendi '**SPI.detachInterrupt()**' bir SPI kütüphane uzantısı ile birlikte '**SPI.attachInterrupt(void myISRfunc)**' (gerçek kütüphane işlev yerine '**SPI.attachInterrupt(void)**' (genel olarak tanınmaya ihtiyaç duymamak için '**ISR(int vector)**' düşük seviyeli vektörlü kesme işlev beyannameleri).
- 10) SPI sistemi artık bağımlı modda da kullanılabilir. '**SS**' pin (pin 10) '**INPUT**' pin ve sürüş '**LOW**' sonra '**SPI.begin()**' veya belirterek '**SPI\_SLV**' isteğe bağlı olarak '**mode**' parametresi '**SPI.begin(int mode=SPI\_MSTR)**' (başka bir UnoArduSim uzantısı '**SPI.h**' ). Alınan bayt daha sonra kullanılarak toplanabilir '**rxbyte = SPI.transfer(tx\_byte)**' ya bir SPI-kesme işlev'ün içinde veya daha önce eklenmiş bir kullanıcı-kesme hizmeti işlev'ün içinde '**SPI.attachInterrupt(myISRfunc)**' . Köle modunda, '**transfer()**' SPDR'de bir veri baytı hazır olana kadar bekler (bu nedenle normal olarak tam bir bayt alımı için beklemeyi engeller, ancak bir kesme işleminde dönüş hemen çünkü alınan SPI baytı zaten orada). Her iki durumda da, '**tx\_byte**' SPDR'ye yerleştirildiği için, ekli ana SPI'nin bir sonraki '**transfer()**' .
- 11) 'Wire.h'un UnoArduSim uygulamasına Bağımlı modu desteği eklendi. İşlev '**begin(uint8\_t slave\_address)**' şimdi olduğu gibi '**onReceive(void\*)**' ve '**onRequest(void\*)**' .
- 12) '**Wire.end()**' ve '**Wire.setClock(freq)**' şimdi çağrılabilir; SCL frekansını bir ile '**freq**' 100.000 (varsayılan standart mod SCL frekansı) veya 400.000 (hızlı mod) değeri.
- 13) 'I2CSLV' cihazlar şimdi 0x00 genel arama veri yolu adresi vb. 0x00 artık bu kölelerden biri için benzersiz bir I2C veri yolu adresi olarak seçilemez.
- 14) Modellenmiş icra, temel tamsayı ve atama işlemlerinde gecikmeler dizin ve işaretçi işlemleri azaltılmış ve şimdi her kayan nokta işlemi için 4 mikrosaniye eklenmiştir.

## **V2.3 Aralık 2018**

- 1) İzleme şu anda etkin **Aracı-Bar** 'I/O' S' kaydırıcı sürekli ve pürüzsüz için 'I/O' cihaz'ın ölçeklendirilmesi, kullanıcının 'S' ekini eklediği değerlerini belirtir.
- 2) Yeni bir **'LED4'** 'I/O' cihaz (4 LED'lerin sırası açık **4 ardışık pin numarası**) Eklendi.
- 3) Yeni bir **'7SEG'** 'I/O' cihaz (onaltılı kodu ile 7 segmentli LED sayı) **4 ardışık pin numarası** ve aktif düşük **CS \*** giriş seçimi) eklendi.
- 4) Yeni bir **'JUMP'** İki 'Uno' pins arasında bir tel atlama teli gibi davranan 'I/O' cihaz eklenmiştir. Bu bir sağlar 'OUTPUT' pin'e kablolanacak 'INPUT' pin (bu yeni özelliğin olası kullanımları için yukarıdaki cihaz'e bakınız).
- 5) Yeni bir **'OWISLV'** 'I/O' cihaz ve üçüncü taraf eklendi '**<OneWire.h>**' şimdi kütüphane ile kullanılabilir '**#include**' Böylece programlar kullanıcı küçük bir '1-Wire' bus cihazlar alt kümesine arayüzlemeyi test edebilir.
- 6) The **Çalıştır** Menü **Reset** komut şimdi **Reset** buton.
- 7) Daha fazla netlik için, ne zaman **Yapay 'loop()' gecikme** altında seçili **Seçenekler** menü, açık '**delay(1)**' İçindeki döngünün altına çağrı eklenir. '**main()**' - bu şimdi bağlı kullanıcı kesintileri tarafından kesilebilir gerçek bir gecikme 'Uno' pins 2 ve 3.
- 8) Elektrik pin ile çatışmak açık boşaltma veya CS seçili, 'I/O' cihazlar (örneğin, I2CLV veya SPISLV) şimdi ilan edildi **yalnızca icra zamanında gerçek bir as in electrical pin-driving conflict), çakışma oluştuğunda**, cihaz ilk bağlandığında hemen bir hataya neden olmak yerine.
- 9) İşlev '**pulseInLong()**' Arduino ile aynı fikirde olmak için şimdi 4-8 mikrosaniye doğrudur (önceki doğruluk 250 mikrosaniye idi).
- 10) Küresel bir değişken'in başlatılması sırasında işaretlenen hatalar şu anda değişken'in **Kodlama Paneli**.

## **V2.2, Haziran 2018**

- 1) üzerinde **Kaydet** ikisinden de **Tercihler** iletişim kutusu veya **Yapılandır | 'I/O' Cihazlar**, the '**myArduPrefs.txt**' dosya şu anda yüklü olan program'nin dizinine kaydedilir - her biri **Dosya | Doldurmak** sonra otomatik olarak yükler dosya, aynı program dizininden belirtilen IODev dosya ile birlikte.
- 2) İşlev '**pulseInLong()**' **olmuştu** eksik, ancak şimdi eklendi (dayanıyor '**micros()**' ölçümleri için).
- 3) program kullanıcısı '**#include**' bir '**\*.h**' dosya, UnoArduSim şimdi otomatik olarak karşılık gelen yüklemeye çalışır '**\*.c**' dosya **Eğer** karşılık gelen '**\*.cpp**' dosya bulunamadı.
- 4) süslü ayraç'ın otomatik olarak takılması '**}**' (açık her süslü ayraç'ten sonra '**{**' ) ekledi **Tercihler**.
- 5) Yeni bir **Seçenekler** menü seçimine şimdi izin veriyor '**interrupts()**' Bir kullanıcı kesintisi rutini içerisinden çağrılmak - bu sadece eğitim amaçlıdır, çünkü pratikte kesinti yapılmasından kaçınılmalıdır.
- 6) İşaretçilere Tipleme '**int**' değer şimdi desteklenir (ancak bir uyarı açılır mesajı görünecektir).
- 7) UnoArduSim şimdi etiketli program satırlarını destekliyor (örn. '**LabelName: count++;**' kullanıcı rahatlığı için (ancak '**goto**' hala **izin verilmeyen** )
- 8) İcra uyarıları şimdi aradığında '**tone()**' Ne zaman pins 3 veya 11'de aktif PWM'ye müdahale edebilir? '**analogWrite()**' Aynı pin'te aktif olan bir Servo'ya müdahale eder, Bir seri karakter varışının kaybedilmesi nedeniyle, kesintiler şu anda devre dışıdır, ve kesintiler o kadar hızlı gerçekleşeceği için UnoArduSim bazılarını özleyecektir.

## **V2.1 Mart 2018**

- 1) Görüntülenen **Değişken Paneli** değerler artık sadece her 30 milisaniyede bir yenileniyor (ve Minimal seçeneği bu yenileme hızını daha da azaltabilir), ancak **VarYenile** güncelleme azaltmasına izin vermemek için menü seçeneği kaldırıldı.
- 2) değişken değerinin baytlarının yalnızca bir kısmını hedefleyen işlemler (işaretçiler aracılığıyla yapılanlar gibi) şimdi değişken değerinde yapılan değişikliğin ABD'ye yansımaları **Değişken Paneli** Görüntüle.

### **V2.0.1 Ocak 2018**

- 1) Belgelenmemiş Arduino işlevler '**exp()**' ve '**log()**' şimdi eklenmiş.
- 2) 'SERVO' cihazlar şimdi sürekli dönme yapılabilir (pals genişliği açtığı yerine hızı kontrol eder).
- 3) İçinde **Düzenle/İnceleme**, bir kapanış süslü ayraç '}' şimdi bir açılış süslü ayraç yazdığınızda otomatik olarak eklenir '{' eğer seçtiyseniz **Tercih**.
- 4) eğer tıklarsanız **Düzenle/İnceleme** pencere başlık çubuğu 'x' Çıkamak için, görüntülenen program dosya'yi değiştirdiyseniz ancak kaydetmediyseniz, iptal etme şansınız var.

### **V2.0 Eylül 2017**

- 1) Uygulama, QtCreator'a aktarılmıştır, bu nedenle GUI'nin bazı küçük görsel farkları vardır, ancak bazı geliştirmelerden başka fonksiyonel farkları yoktur:
  - a) Durum satırı Ana pencere'ün altındaki ve **Düzenle/İnceleme** iletişim kutusu geliştirildi ve vurgulu renk kodlaması eklendi.
  - b) Arasında ayrılan dikey boşluk **Kodlama Paneli** ve **Değişken Paneli** şimdi, paylaşılan sınırlarında, sürükleyip bırak özelliği (ancak görünmez) bir ayırıcı çubuk vasıtasıyla ayarlanabilir.
  - c) 'I/O' cihaz düzenleme kutusu değerleri, yalnızca kullanıcı fare işaretçisini cihaz'ın dışına çıkardıktan sonra doğrulanır; bu, kullanıcı yazarken yasal değerleri zorlamak için uygunsuz otomatik değişiklikleri önler.
- 2) UnoArduSim şimdi aracılığıyla birden fazla dili destekliyor **Yapılandır | Tercihler**. Kullanıcı yerel ayarının diline ek olarak İngilizce her zaman seçilebilir (bu dil için özel bir \*.qm çevirisi dosya olduğu sürece UnoArduSim 'translations' klasöründe bulunur).
- 3) Ses şimdi Qt ses API'sini kullanmak için değiştirildi - bu, belirli durumlarda sesin kesilmesi için gerekli can sıkıcı ses kesintilerini ve tıklamaları önlemek normal kullanıcı fare tıklamalarının neden olduğu daha uzun işletim sistemi pencerelemesi sırasında operasyonel gecikmeler - daha fazla ayrıntı için bu konuda.
- 4) Bir kullanıcı rahatlığı olarak, boşluklardaki şimdi cihaz-count düzenleme kutularındaki 0 değerini temsil etmek için kullanılır. **Yapılandır | I/O Cihazlar** (böylece artık cihazlar'yi kaldırmak için boşluk çubuğunu kullanabilirsiniz).
- 5) Ölçeklendirilmemiş (U) niteleyicisi şimdi 'PULSER', 'FUNCGEN' ve '1SHOT' cihazlar'de isteğe bağlıdır (varsayılan olarak kabul edilir).
- 6) UnoArduSim şimdi izin verir (değişmez sayısal değerlere ek olarak) '**const**' tamsayı değerli değişkenler ve 'enum'

### **V1.7.2- Şubat 2017**

- 1) LED cihazlar için renk seçimi mavisi (B) eklenmiştir.

### **V1.7.1 - Şubat 2017**

- 1) Ekler '**1**' ve / veya '**U**' şimdi sayısal değişmez sabitlerin sonuna kabul edilir ( '**long**' ve / veya '**unsigned**' ), ve ('**0b**' veya '**0B**' önceden eklenmiş) ikili sabitleri şimdi de kabul edildi. Ondalık herhangi bir sayısal sabit **ile başlayan** '**0**' şimdi bir olarak kabul edilir **sekizli** değer. (Arduino ile aynı fikirdeyim).
- 2) Kaçışın olmadığı sıkı bir döngüde çalıştırırken (örneğin '**while(x) ; x++ ;**' nerede x her zaman doğrudur),

**Dur** Şimdi ikinci kez program icra'in gerçekten durmasını sağlar (ve hatalı program hattında).

### **V1.7.0– Aralık 2016**

1) Yeni bir **Aracı-Bar** program icra sırasında boş RAM baytını gösteren bir özellik eklenmiştir (genel değişkenler tarafından kullanılan toplam bayt, yığın tahsisleri ve yerel değişkenler yığını).

2) Kullanıcı kesintisi işlevler şimdi kendileri de Arduino işlevler gibi engelleme çağırabilir '**pulseIn()**' (ancak işlev'ün kesilmesi işlev'ün engellenmesi tamamlanıncaya kadar geri dönmeyeceğinden bu yalnızca dikkatli kullanılmalıdır).

3) Kullanıcı kesintileri engellenen Akış okuma işlemleri sırasında artık devre dışı bırakılmadığından davranış artık gerçek Arduino akış okuma işlemiyle eşleşiyor.

4) Artık kesilebilecek Arduino işlevler'i bloke edip engelleyebilirsiniz (gibi '**delay()**' ve '**pulseIn()**') ve Durum Çubuğu mesajlarının, böyle bir işlev'ün içindeki kesim noktası kesmesine ne zaman çarptığınızı (veya icra şu anda böyle bir işlev'ün içindeyken-Dur'u tıklattığınızda) göstermesi için artırıldı.

5) Yeni bir **Koşula Çalıştır** komut (ve **Aracı-Bar** item) eklendi - herhangi bir tuşa tek tıkla **Değişken Paneli** değişken (basit, toplam olabilir dizin veya includes 'struct' and 'class'), nesne veya bir dizin elemanı veya includes 'struct' and 'class'), nesne üyesi) vurgulu'a **Koşula Çalıştır** - icra sonraki donacak **yazmaya karşı erişim** bu toplam değişken'in içinde veya o tek bir yerde.

6) icra'in ardından **Adım, Komuta Çalıştır, Koşula Çalıştır**veya **Çalıştır-sonra-Dur** eylem **Değişken Paneli** şimdi karşılık gelen değişken vurgulamaktadır **değiştirilen adres konumları** (varsa) tarafından **en son talimat bu sırada icra** - eğer bu konum şu anda bir genişletilmiş dizin ya da includes 'struct' and 'class'), nesne'in içine gizlenmişse, genişletmek'e tıklandığında, bu son değiştirilmiş elemanın veya üyenin vurgulanmasına neden olacaktır.

7) Kullanıcı şimdi belirli bir değere özel bir saat tutabilir **Değişken Paneli** değişken / üye / eleman çalıştırılırken - satırdaki satırın üzerine çift tıklayın. **Değişken Paneli** açmak için **Düzenle/İzle Değişken Değeri** pencere, sonra birini yapın **Çalıştır** veya **Adım** komutlar - gösterilen değer icra'de güncellemeleri yöneten aynı kurallara göre güncellenecektir. **Değişken Paneli**. icra'i durdurduktan sonra, yeni bir değer girmenize izin verilir ve **Kabul Et** icra'e devam etmeden önce **Geri Çevir** öncesi **Kabul Et** Daha önce fikrinizi değiştirirseniz değer).

8) F4-F10 gaz pedalı tuşları **Çalıştır** menüsüne uyacak şekilde ayarlanmış **Aracı-Bar** komutlar (soldan sağa).

9) Üzerlerine çift tıklamaya ek olarak, 'SERIAL', 'SFTSER', 'SPISLV', 'I2CSLV' üzerine sağ tıklayarak cihazlar şimdi daha büyük boyutlu bir TX / RX bayt / karakter pencere (ve 'SD\_DRV"de bir dosyalar-izleme pencere).

10) 'SERIAL' veya 'SFTSER"daki TX düzenleme kutusu artık aktif karakter iletimi sırasında devre dışı bırakılmamıştır (bu yüzden şimdi olanları ekleyebilir veya değiştirebilirsiniz), ancak ilgili 'Seri; Monitör alt ögesinde bir satır başı (veya 'Send' düğmesi tıklatmasıyla tıklayabilirsiniz) pencere) bir kez daha aktarım boş duruma dönene kadar göz ardı edilir (aktarım başladığında hazır olan karakterler şimdi italik olarak gösterilir, etkindir). Ek olarak, kullanıcı şimdi seri bir akışta uyarılır. '**begin()**' daha önce başlamışlarsa, ekteki cihaz'i (şu an devam etmekte olan) iletimleri, daha sonra hiçbir çerçeveleme senkronizasyonu olmayacağından, alım hatalarına yol açmaktadır.

11) Varsayılan eklendi '**loop()**' Gecikme 250 mikrosaniyeden bir milisaniyeye yükseltildi, böylece kullanıcı bir miktar eklemeyi ihmal ettiğinde gerçek zamanın çok gerisinde kalmadı '**delay()**' (açık veya doğal) içeride bir yerde '**loop()**' veya çağırıldığı bir işlev'ün içinde.

12) Öbek ayırma desteğine Diziler ve basit türler eklenmiştir '**new**' talimat.

13) program sınır dışı adres erişimi kullanıcısı için daha kapsamlı kontroller (ve ilişkili hata mesajları) eklenmiştir (örneğin, 'Uno' RAM'in dışında veya '**PROGMEM**' ) Erişir.

14) İşaretçi değerleri **Değişken Paneli** şimdi gerçek Arduino pointer değerlerine daha çok benziyor.

15) Kullanıcı '**myArduPrefs.txt**' dosya şimdi her yüklendi **Dosya | Doldurmak**, sadece UnoArduSim lansmanında

değil.

16) Bir Ayırıştır hatası şimdi denenirken işaretlendi ' **attachInterrupt()** ' bir kullanıcı için işlev değil ki ' **void** ' geri dönen, ya da işlev parametreleri olan ya da daha önce bir yerde bildirilmemiş olan ' **attachInterrupt()** ' .

17) ' **static** ' üye-değişkenler şimdi **Değişken Paneli** (genişletilmiş) includes 'struct' and 'class'), nesne'in her bir örneğinde görünmek yerine, küreler olarak.

18) İşlev ' **availableForWrite()** ' uygulamasına eklendi. ' **Serial** ' .

19) Tüm özel ' **PROGMEM** ', ' **typedef** ' sevmek ' **prog\_char** ' ve ' **prog\_int16** ' şimdi kaldırıldı (Arduino'da kullanımdan kaldırıldılar).

20) Yanlış hecelenmiş veya geçersiz bildirim türlerinden kaynaklanan Ayırıştır hataları için iyileştirilmiş hata mesajları.

21) İzin verilen maksimum program büyüklüğü artırıldı.

### **V1.6.3 - Eylül 2016**

1) Eklendi iyileştirilmiş ayırıştır hata mesajı ' **attachInterrupt()** ' bir kesme-işlev olmayan anlamına gelir **daha önce prototiplenmiş** .

2) Çok boyutlu dizin başlatma listeleri için geliştirilmiş bir Ayırıştır hata iletisi eklendi.

### **V1.6.2 - Eylül 2016**

1) Bir eklendi **Bul-Metin** kontrolü düzenle **Aracı-Bar** Metin aramayı kolaylaştırmak için **Kodlama Paneli** ve **Değişken Paneli** ).

2) The **Aracı-Bar** Reset butonu şimdi 'Uno' kartı Reset butonuyla aynı şekilde çalışıyor.

### **V1.6.1 - Ağustos 2016**

Önceden yinelenen yükleme ve ayırıştırma işlemlerini önlemek için bir çek eklendi ' **#include** ' dosyalar, .

### **V1.6 - Haziran 2016**

1) Yeni seçilen bir '1SHOT' (bir kerelik) 'I/O' Cihaz, seçilen polaritenin tetikleyici bir sinyal kenarından seçilen bir gecikmeden sonra bir darbe üreten.

2) Yeni bir özellik eklendi 'I/O' cihaz düzenleme kutusu değerlerini kolayca yapar **pullu** icra boyunca ana küresel bir 'I/O\_\_\_\_S' Ölçek kaydırıcısını sürükleyerek **Aracı-Bar** (sadece ölçeklendirmeyi belirten bir değerden sonra tek bir 's' veya 'S' harfini yazın).

### **V1.5.1 - Haziran 2016**

1) EEPROM kütüphanesi işlevler için destek eklendi ' **update()** ' , ' **put()** ' ve ' **get()** ' , ve örneğin dizin notasyonu ile bayt erişimi için, örneğin ' **EEPROM[k]** ' .

2) **Otomatik'e İzin Ver (-) Opposite of exand the displayed array)**, **daralt** menüye eklendi **VarYenile** icra gerçek zamanın gerisinde kaldığında genişletilmiş diziler / includes 'struct' and 'class'), nesneleri'ün otomatik kontratı yapılıp yapılmayacağı konusunda açık kontrol yapılmasını sağlamak

3) Karakterler bir ' **String** ' değişken şimdi de erişilebilirdizin notasyonu ile,Örneğin ' **mystring[k]** ' .

### **V1.5 - Mayıs 2016**

1) **Düzenle/İnceleme** şimdi ctrl-E kısayoluna sahip ve **Derle** (ctrl-R), ayrıca dahili Ayırıştır hata kutusu, düzenlemelerin kapatılmasına gerek kalmadan test edilmesini sağlar pencere.

2) **Düzenle/İnceleme** şimdi de destekliyor **İleri**ve yeni **Kaydet** (ctrl-S) düğmesi (eşdeğer **Kabul Et** artı bir sonraki ana pencere **Kaydet**) ve şimdi bir seçenek sunar **'Tab'** boyut (kullanılarak kaydedilebilecek yeni bir tercih **Yapılandır | Tercihler**).

3) Artık yazılabilir tüm düzenleme kutuları seçilen Pencere İşletim sistemi tema renklerini izlemekte ve kontrast için tüm salt okunur 'RECV' düzenleme kutuları siyah arka plan üzerinde beyaz metin kullanmaktadır. The **Düzenle/İnceleme** arkaplan ve sentaks-vurgulu renkleri artık seçilen temaya adapte oluyor.

4) UnoArduSim artık bir font seçimine izin veriyor - bu seçim ve büyüklüğü **Yapılandır | Tercihler** (yani, **'myArduPrefs.txt'** dosya).

5) Arduino önceden tanımlanmış ikili değişmez değerleri (gibi **'B01011011'**) şimdi izin verilir.

6) Kaçan altıgen, sekizli ve 4-sayı Unicode alıntı karakter dizileri şimdi sayısal değişmezler olarak kullanılabilir.

7) Bir 'PUSH' cihaz basmalı tuşa ilk fare tıklaması yapıldıktan sonra, kullanıcı daha sonra basmalı düğmeli kontaklara basmak için bir tuşa basabilirsiniz (herhangi bir tuş).

8) **Düzenle/İnceleme** şimdi kısa bir görsel flaş ipucundan sonra geçici ilk salt okunur durumunu serbest bırakır (ve seçilen ilk satırın vurgulanmasını kaldırır).

9) UnoArduSim şimdi çoklu kontrol ediyor **'Stepper'** ve **'Servo'** pin çatışmalar, yani hatalı kullanıcı program, daha önce eklenmiş olan pins'e bağlanmaya çalışıyor **'Stepper'** veya **'Servo'** değişkenler.

10) Bir operatörün sol veya sağ tarafının eksik olmasından kaynaklanan bir Ayrıştırma hatası (LHS veya RHS ifadesi veya değişken eksik) şimdi açık bir hata mesajı veriyor.

11) Kullanılmayan **'String'** sınıf **'flags'** üye değişken, Arduino V1.6.6 ile aynı fikirdeydi. bir **'String'** includes 'struct' and 'class'), nesne şimdi 6 baytı kapsıyor (artı karakterleri yığın ayırma).

## **V1.4.2 - Mart 2016**

1) İleri tanımlanmış işlevler (yani, ilk çağrılarında önce prototip bildirimi olmayanlar) şimdi yalnızca sonraki işlev tanım dönüş tipi ilk kullanımlarından çıkarılan türle uyuşmadığında uyarılar oluşturur (ayrıştırma hataları değil).

2) 1'e eşit bir boyuta sahip olan Diziler artık reddedilmez (standart C ++ kurallarına uymak için).

3) düzenleme kutuları artık beyaz arka plan üzerinde siyah olarak ayarlanmamış - artık kullanımda olan Pencere İşletim Sistemi temasının belirlediği paleti kullanıyorlar.

4) 'SERIAL', 'SFTSER', 'SPISLV' ve 'I2CSLV' cihaz genişletilmiş Monitör pencereler (çift tıklayarak açılarak) şimdi ana 'I/O' Cihaz'ının arka plan rengini alır.

## **V1.4 - Aralık 2015**

1) **'Stepper.h'** Kütüphane işlevselliği ve ilgili 'I/O' cihazlar şimdi eklenmiştir.

2) **Tüm 'I/O' Cihaz ayarları ve değerleri** (seçilen pins'e ek olarak) şimdi seçilen kullanıcı 'I/O' Cihazlar'ın metni dosya'nın bir parçası olarak kaydedilir. daha sonra yeniden yüklemek için.

3) **LED 'I/O'** cihaz rengi artık cihaz'teki bir düzenleme kutusu kullanılarak kırmızı, sarı veya yeşil olarak ayarlanabilir.

4) Değişken bildirim başlatıcılarının şimdi birden fazla satıra yayılmasına izin verildi.

5) Dizin endeksleri artık kendilerine izin elementi olarak izin veriyor.

6) **Yapılandır | Tercihler** şimdi izin verilecek bir onay kutusu içeriyor **'and'**, **'or'**, **'not'** C standardı yerine kullanılacak anahtar kelimeler **'&&'**, **'||'**, ve **'!'** mantıksal operatörler.

7) "Show Program Indirme" taşındı **Yapılandır | Tercihler**



### **V1.3 - Ekim 2015**

- 1) The '**PUSH**' cihaz Şimdi 'latch' etiketli "basma" onay kutularına sahipler, "kilitleme" yapmak için ("anlık" yerine), yani, tekrar basılınca kadar basıldığında kapalı konumda mandallanır (ve renk değiştirir) kişileri serbest bırakmak için
- 2) Düğüm seçimi ile tam yetenek 'SPISLV' cihazlar eklendi ( '**MODE0**' , '**MODE1**' , '**MODE2**' veya '**MODE3**' ). Çift tıklamak, yaklaşmakta olan REPLY (TX) baytlarının tanımlanabileceği ve geçmiş (RX) baytların görüntülenmesi için pencere TX / RX tamponlarını açar. Önceki versiyonun basit shift-register slave cihaz'ı 'SRSLV' cihaz olarak değiştirildi.
- 3) **Kalın** yazıyüzü şimdi için seçilebilir **Kodlama Paneli** ve **Değişken Paneli** (menüden **Seçenekler**), ve **kalın anahtar kelimelerin ve operatörlerin vurgulanması** şimdi açılıp kapatılabilir **Düzenle/İnceleme** .
- 4) UnoArduSim şimdi izin veriyor '**bool**' eş anlamlı olarak '**boolean**' .
- 5) Hata raporlamadaki netlik için değişken beyannamelerinin artık birden fazla satıra yayılmasına izin verilmemektedir (başlatıcı listelerine sahip diziler hariç).
- 6) Sözdizimi renklendirme hızı **Düzenle/İnceleme** iyileştirildi (bu, daha büyük programlar ile göze çarpmak).
- 7) İsteğe bağlı 200 mikrosaniye ek yük (menüde **Seçenekler** ) her çağrıya '**loop**()' - bu, gerçek zamanlı olarak gerinin çok gerisinde kalmaktan kaçınmaya çalışmaktır. program kullanıcısı eklemeyi '**delay**()' her yerde (bkz. Zamanlama tartışması).

### **V1.2 Haziran 2015**

- 1) SD kütüphanesi şimdi tam olarak uygulanmıştır ve (küçük) bir 8Mbayt SD Disk 'I/O' cihaz ('SD\_DRV') eklenmiştir (ve tüm Arduino örnek SD programlar'a karşı test edilmiş fonksiyonellik).
- 2) Arduino gibi, UnoArduSim artık bir işlev'ün çağrılmasını beklediğinde işlev argümanını otomatik olarak adresine çevirir.
- 3) Ayırıştır-hata mesajları eksik noktalı virgül olduğunda ve tanınmayan bildirimlerden sonra artık daha uygun.
- 4) Bayat **Değişken Paneli** hat olayları artık işlev çağrısı / dönüşünde kaldırılıyor.

### **V1.1 - Mart 2015**

- 1) Ana pencere, şimdi yapmak için maksimize edilebilir veya yeniden boyutlandırılabilir. **Kodlama Paneli** ve **Değişken Paneli** daha geniş (daha büyük ekranlar için).
- 2) Yeni bir menü Bul (ile **Araç Çubuğu düğmeleri** içinde daha hızlı gezinmeye izin vermek için eklenmiş **Kodlama Paneli** ve **Değişken Paneli** (PgUp ve PgDown veya yukarı ok, aşağı ok ile metin arama).
- 3) The **Düzenle/İnceleme** pencere şimdi ctrl-PgUp ve ctrl-PgDn gezinti atlayışlarına (sonraki boş satıra) izin veriyor ve artırıldı **Bul / Değiştir** işlevsellik.
- 4) bir menüye yeni öğe eklendi **VarYenile** Kullanıcının ağır altında bir hesaplama tasarrufu yaklaşımı seçmesine izin vermek **Değişken Paneli** yükleri güncelle.
- 5) 'Uno' pins ve ekli LED şimdi, zaman donduğunda bile (yani icra durdurulduğunda bile) 'I/O' cihazlar'ye yapılan değişiklikleri yansıtıyor.
- 6) Diğer işlevler kullanıcısı şimdi bir işlev kullanıcısı kesintisinden (Arduino 1.06 güncellemesine göre) çağrılabilir.
- 7) bir **daha büyük yazı tipi** şimdi menüden seçilebilir **Seçenekler**.

**V1.0.1 - Haziran. 2014**

E signal shape), dalgabiçimi pencereler şimdi analog pins'i 14-19 yerine A0-A5 olarak etiketliyor.

**V1.0 - ilk sürüm Mayıs 2014**