KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANA BİLİM DALI

LabVIEW TABANLI DAMAR İÇİ UYGULANAN BESİN VE İLAÇ HAZIRLAMA SİSTEMİ TASARIMI VE GERÇEK ZAMANLI KONTROLÜ

YÜKSEK LİSANS TEZİ

Elektrik-Elektronik Mühendisi Mehmet EKİCİ

TEMMUZ 2009
TRABZON

KARADENİZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRİK-ELEKTRONİK MÜHENDİSLİĞİ ANA BİLİM DALI

LabVIEW TABANLI DAMAR İÇİ UYGULANAN BESİN VE İLAÇ HAZIRLAMA SİSTEMİ TASARIMI VE GERÇEK ZAMANLI KONTROLÜ

Elektrik-Elektronik Mühendisi Mehmet EKİCİ

Karadeniz Teknik Üniversitesi Fen Bilimleri Enstitüsünce
''Elektrik Yüksek Mühendisi''
Unvanı Verilmesi İçin Kabul Edilen Tezdir.

Tezin Enstitüye Verildiği Tarih: 12.06.2009
Tezin Savunma Tarihi : 13.07.2009

Tez Danışmanı: Prof. Dr. A. Sefa AKPINAR
Jüri Üyesi : Yrd. Doç. Dr. H. İbrahim OKUMUŞ
Jüri Üyesi : Yrd. Doç. Dr. Hüseyin PEHLİVAN

Enstitü Müdürü: Prof. Dr. Salih TERZİOĞLU

Trabzon 2009

ÖNSÖZ

Bu tez, Karadeniz Teknik Üniversitesi Fen Bilimleri Ensitüsü Elektrik-Elektronik Mühendisliği Anabilim Dalı, Elektrik Mühendisliği Anabilim Dalı Yüksek Lisans Programı'nda hazırlanmıştır. Çalışmamda LabVIEW grafiksel programlama ve bu program tabanlı damar içi uygulanan besin ve ilaç hazırlama sistemi tasarımı ile bu sistemin gerçek zamanlı kontrolü konusu işlenmiştir.

Tüm eğitim-öğretim hayatım boyunca maddi ve manevi desteklerini esirgemeyen aileme ve çok kıymetli eşime, tez çalışmam süresince bilimsel desteği ve değerli düşünceleriyle bana her aşamada yardımcı olan danışmanım Prof. Dr. A.Sefa AKPINAR'a teşekkür ederim. Bu tezin, bundan sonraki çalışmalara katkı sağlamasını temenni ederim.

Mehmet EKİCİ
Trabzon 2009

İÇİNDEKİLER

Sayfa No

ÖZET

Bu tezde amacım, LabVIEW tabanlı damar içi uygulanan besin ve ilaç hazırlama sistemi tasarımı yapmak ve gerçek zamanlı kontrolünü sağlamaktır. Tıp alanında son zamanlarda teknoloji kullanımı kat ve kat artmaktadır. Mikroişlemci teknolojisinin gelişimi ile birlikte bilim adamları bu teknolojiyi tıp alanında kullanmaya başlamışlardır. Bunların en büyük örnekleri ülkemizde medikal alanda kullanılmaktadır.

TPN ve onkolojik ilaçların hazırlanması damar içi uygulanan besin ve ilaçlara en iyi örnektir. Ülkemizde bu işlem pek çok yerde manuel olarak yapılmakta ve bundan kaynaklanan ciddi sorunlar yaşanmaktadır. Bunlardan bazıları hastanelerin yenidoğan ve onkoloji ünitelerinde ölümlere kadar yol açmaktadır.

Hastanelerimizde yenidoğan ve onkolojik ilaçların hazırlanmasında kısmende olsa otomasyona geçilmiş olmakla birlikte, bu sistem ve cihazların üretimi dünya çapında iki firmaya bağımlı olduğundan, ülkemizin bu sistemleri tamamen ithal etme durumunda kaldığını, yaptığım çalışmalar esnasında gözlemlemiş bulunmaktayım. Onkolojik ilaçlar oldukça pahalıdır ve manuel karışımlar hazırlanırken israflarla karşılaşılmaktadır. Aynı zamanda klinisyen ya da görevliler için sağlık açısından tehlike arz etmektedir. Dolayısı ile karışımlar el değmeden, sistem tarafından hazırlanmalıdır.

Ayrıca TPN sisteminde otomasyonun uygulanması, yenidoğan yoğun bakımlarında enfeksiyona bağlı ölümleri ve hastane enfeksiyonlarını da önemli derecede azaltmaktadır. Yenidoğan ve diğer yoğun bakım ünitelerinindeki hastaların beslenmelerinin sadece hemşirelerin dikkatine bağımlı kalmadan, sistem tarafından da kontrol edilebilmesi son derece iyi bir gelişme olarak görülmektedir.

Anahtar Kelimeler: LabVIEW, teknoloji, TPN, onkoloji, otomasyon

SUMMARY

LabVIEW BASED IN VASCULAR APPLIED FOOD AND DRUG PREPARATION CONTROL SYSTEM DESIGN AND REAL-TIME CONTROL

My purpose in this thesis, LabVIEW based intravenous applied food and medicines preparation system design and to provide real time control. Recently, in the field of medical technology are increasing rapidly well. With the development of microprocessor technology, in the field of medicine, scientists have begun to use them in our country. Most important examples of these are used in the medical field.

TPN and oncological drugs preparation are the best examples by intravenous applied to food and medicines. In many places in our country this process is done manually and is experiencing serious problems arising from this, these problems are reason to some death at oncology and newborn units in hospitals.

In some hospitals, newborn and oncological drugs preparation are made by automation. These systems and devices are pruducted only two companies all arround the world, in which all these systems need to import that I observed during my studies I have. Oncological drugs are so expensive if they are prepared manually, it causes some waste. At the same time, for stuff and clinicians, there is a health hazards. As a result, the system must be prepared by untouched.

Moreover, the implementation of automation in the TPN system, infection related deaths and hospital infection at newborn units significant decreases also. The patients' feeding of newborn and other intensive care ünits not only dependent on the attention of nurses, also controled by the system is extremely well.

Key Words: LabVIEW, tecnology, TPN, oncology, automation

ŞEKİLLER DİZİNİ

Sayfa No

SEMBOLLER DİZİNİ

1D	: 1 Dimension
2D	: 2 Dimension
CB-68LP	: 68 pin analog kart
DAQ	: Data Acquisation
DBL	: Data type
DC	: Doğru akım
DCV	: Doğru gerilim
Em	: Elektro motor kuvveti
FLC	: Fuzzy Logic Controler
I/O	: Input / Output
i(t)	: Zamana bağımlı Akım
Jm	: Eylemsizlik momenti
Km	: Motor sabiti
L	: Endüktans
LabVIEW	: Laboratory Virtual Instrument Engineering Workbench
ml	: mili litre
ms	: mili saniye
NI	: National Instrument
NI-DAQmx	: National Instrument Multifunction Data Acquisation
PCI	: Peripheral Component Interconnect
PI	: Proportional, Integral
PID	: Proportional, Integral and Derivative
R	: Direnç
RTS	: Neutral-Supported Type
SQRT	: Square Root
TPN	: Total Paranteral Nutrition
V	: Volt
VI	: Virtual Instrument
w(t)	: Açısal hız
γ	: (Gama) Sürtünme sabiti

1.GENEL BİLGİLER

1.1. LabVIEW Program Yapısı

1.1.1. Genel Bakış

Yapılar, düğümün bir önemli tipi olup, VI'da akış uygulamasının yönetimin, sadece kontrol yapısı gibi bir standart programlama dilinde yapar. Bu bölüm, LabVIEW'deki dört yapıyı tanıtmaktadır. Bu yapılar, *For* döngüsü, *While* döngüsü, Durum yapısı ve Dizi yapısıdır. Formül düğümlerini kullanarak, formül uzunluklarının nasıl yerine getirildiğini, özel diyalog kutularının içeriğinin nasıl ortaya çıkarıldığını ve programın nasıl kontrol edileceği gösterilmektedir [1].

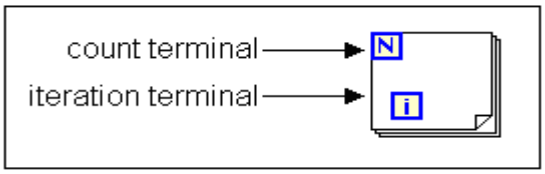
While döngüsü ve *For* döngüsünün kullanımını bilmek ve bu ikisi arasındaki farkları anlamak, grafik programlamada kaymalı kaydedicinin gereksinimini tanımak, yapı durumunun iki tipini anlamak, *Numeric* ve *Boolean*, dizi yapılarını kullanarak uygulamanın sırasının nasıl düzenlenebileceğini öğrenmek, uzun matematik formüllerinin yerine getirilmesi için formül düğümlerini kullanmak, LabVIEW'de istediğin herhangi bir şeyi söyleyen bir diyalog kutusu yaparak ortaya çıkarmak, LabVIEW'in zaman fonksiyonlarının nasıl kullanıldığını anlamak mümkün olacaktır [1,2].

1.1.2. İki Döngü

VI'daki tekrarlayan işlemleri kontrol etmek için *For* döngüsü ile *While* döngüsü kullanılır. *For* döngüsü özel olarak birkaç defa, *While* döngüsü özel şartın sağlanacağı ana kadar yerine getirilir. Bu döngüleri *Functions* menüsü altında *Structs&Constants*'da bulunmaktadır [1].

1.1.2.1. *For* Döngüsü

Aşağıda, Şekil 1.1'de görüldüğü gibi *For* döngüsü, kendi sınırları içinde kodu uygular, sayım terminalinde içerdiği değeri sayıcıya eşitleyen sayıcı zamanlayıcısının bir toplamıdır [1].



Şekil 1.1. For Döngüsü

Tekrar terminali, tamamlanmış döngünün geçerli değerini gösterir. N-1'e kadar birinci tekrarda 0, ikinci tekrarda 1 vs. Sayım terminaline 0 koyduğumuz takdirde döngü işlemez.

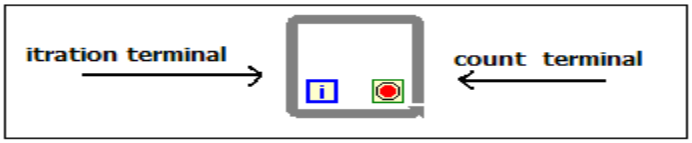
For döngüsü aşağıdaki formülle bulunur.

For i=1 to N-1 (1)

Alt diyagram uygulaması

1.1.2.2. *While Döngüsü*

Aşağıda Şekil 1.2'de görüldüğü gibi While döngüsü sınırı içersinde *boolean* değerinin, şartlı terminalin *FALSE* (Yanlış) sağlayacağı alt diyagramı içerir. LabVIEW her bir tekrardan sonra şartlı terminal değerini gözden geçirir. Eğer değer *TRUE* (Doğru) ise sıradaki tekrar meydana gelir. Şartlı terminalin değeri *FALSE* (Yanlış) olursa ve eğer böyle bırakırsa döngü bir kez tekrarlanır [1].



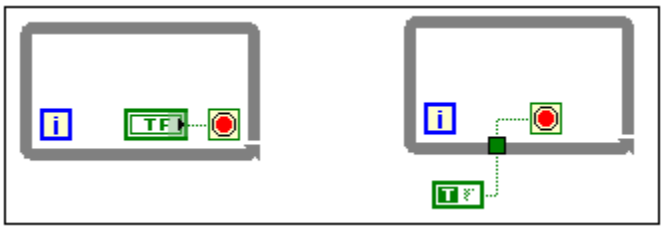
Şekil 1.2. While Döngüsü

While döngüsünün tekrar terminali *For* döngüsünün aynısını yapar.

1.1.3. Devre İçindeki Terminaller ve Onların Davranışları

Döngü içindeki terminaller ve onların davranışları LabVIEW'in veri akışı kuralları altında çalıştığı sürece, döngü çalışmadan ve parça verileri girilmeden önce parçalar yerleştirilmelidir. Döngü çıkışları ancak ve ancak döngü tekrarları tamamlandıktan sonra elde edilir.

Ve böylece bir terminali gözden geçirmek, yenilemek istenirse, döngü içine konulmalıdır. Örneğin, sol *While* döngüsü her devirde *boolean* değerini gözden geçirir. Ne zaman devre *FALSE* (Yanlış) değer verirse kendiliğinden sonuçlanır [1, 2].



Şekil 1.3. Terminaller

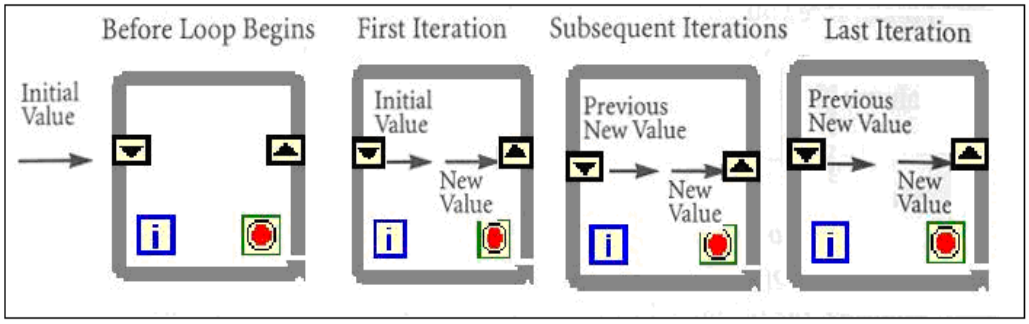
Eğer *boolean* kontrol terminalleri *While* döngüsünün dışına yerleştirilirse, Şekilde görüldüğü gibi sonsuz çevrim oluşturur veya *boolean* terminalinin başlangıç verisine bağlı kalarak bir tek çevrim oluşur. Veri akışında, LabVIEW boolean değerini döngü içinde ya da döngü tamamlandıktan sonra değil, döngüye girmeden önce okur.

Dikkat edilmesi gereken bir husus ise *For* döngüsü ya da *While* döngüsünde başlangıç değerinin sıfır olduğu unutulmamalıdır. Kayıtlı döngü sayısını öğrenmek için sayıma bir eklenmelidir.

1.2. Kaymalı Kaydediciler

Kaymalı kaydediciler (*For* ve *While* döngüsü için elverişli) bir tekrardan sonraki tekrara yerel değişken değerlerini taşır. LabVIEW'in grafiksel yapısı için tek ve gereklidir. Pop–up menüsünden *Add Shift Register* seçilerek döngünün sol veya sağ sınırında oluşturulur.

Kaymalı kaydedici, döngünün sınırlarında dik ve zıt bir Şekilde yerleştirilmiş terminal çiftini içerir. Sağ terminal üzerinde tamamlanmış döngünün verileri bulunur. Öyle ki, 'Shifted' vericisi Şekil 1.4'de görüldüğü gibi, son döngüdür ve gelecek döngünün başlangıcında sol terminalde ortaya çıkar. Bir kaymalı kaydedici *numeric, boolean, string, array* vb. veri tiplerini kapsayabilir. Kaymalı kaydedici parçanın bir terminale girmesi ile otomatik olarak onun veri tipine ayak uydurur.

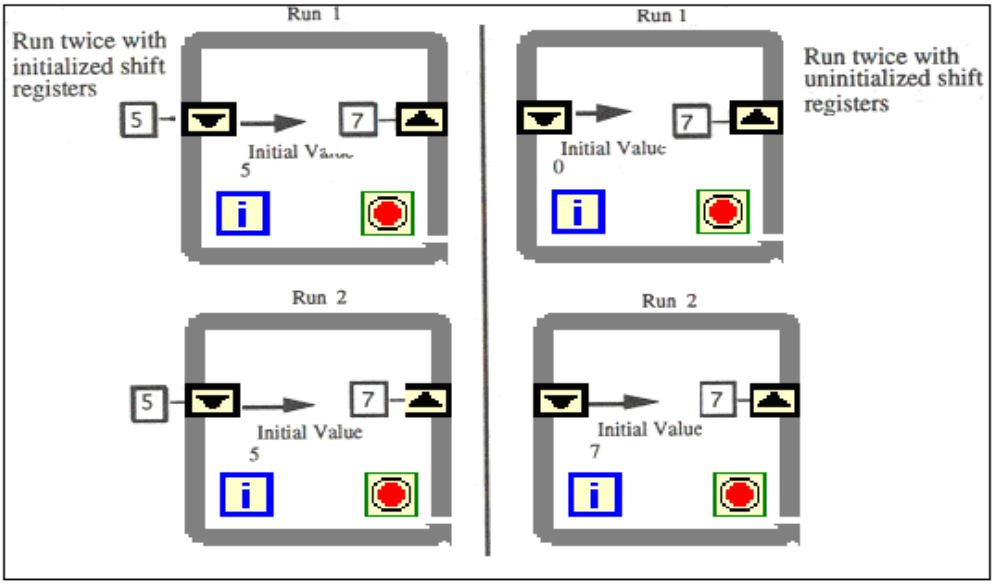


Şekil 1.4. Kaymalı kaydediciler

1.2.1. Kaymalı Kaydedicileri Sıfırlama

Aksi ve özel bir durum olmadıkça kaymalı kaydedici yeniden sıfırlanmalıdır. Özel değer ile kaymalı kaydediciyi yeniden sıfırlama için, Şekildeki gibi kaymalı kaydedicinin sol terminaline girilmelidir. Eğer sıfırlanmazsa, program çalıştığında başlangıç değeri kaymalı kaydedici değer tipi ile aynı olmadığından hata verir. Örneğin, eğer kaymalı kaydedici veri tipi *boolean* ise, başlangıç değeri *FALSE* (Yanlış) olur. Benzer Şekilde değer sayısal ise, başlangıç değeri 0'dır. VI ikinci kez çalıştığında, yeniden sıfırlanmış kaymalı kaydedici en az birinde çevrim değerleri kadar değerler içerir. Yeniden sıfırlamayı iyi anlamak için aşağıda Şekil 1.5 e bakılabilir. Sol kolon ikinci kere sıfırlanmış kaymalı kaydedici olan döngüyü içerir. Sağ kolon ikinci kere yeniden sıfırlanmamış kaymalı kaydedici olan döngüyü içerir.

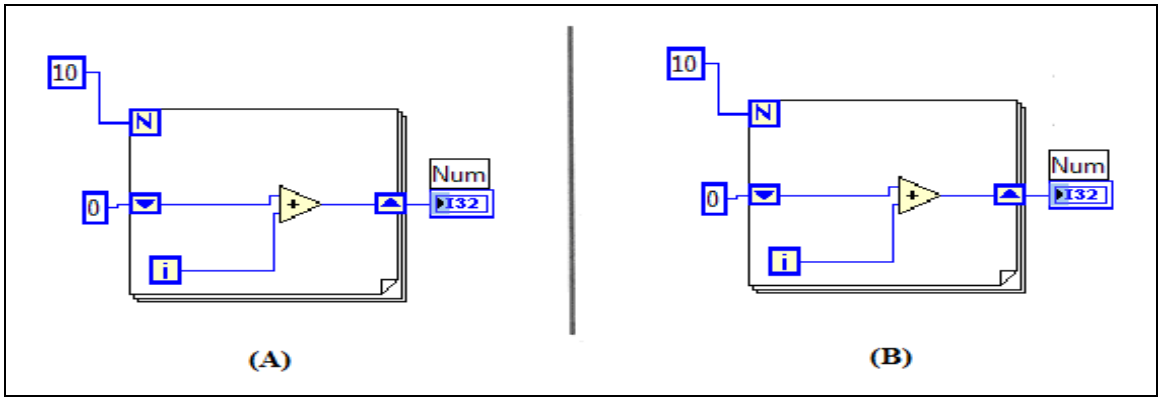
Dikkat edilmesi gereken husus ise LabVIEW, VI'yı kapatıp hafızaya çıkarana kadar kaymalı kaydedicide saklı olan değerleri ıskartaya çıkarmaz. Veya yeniden sıfırlanmamış kaymalı kaydedici içeren VI'yı çalıştırırsanız, alt bölümün başlangıç değerleri bir önceki devreden kalan değerler olacaktır. Problemleri noktalamak oldukça zorlaşır [2].



Şekil 1.5. Kaymalı kaydedicileri sıfırlama

1.2.2.Kaymalı Kaydedicilere Neden İhtiyaç Var

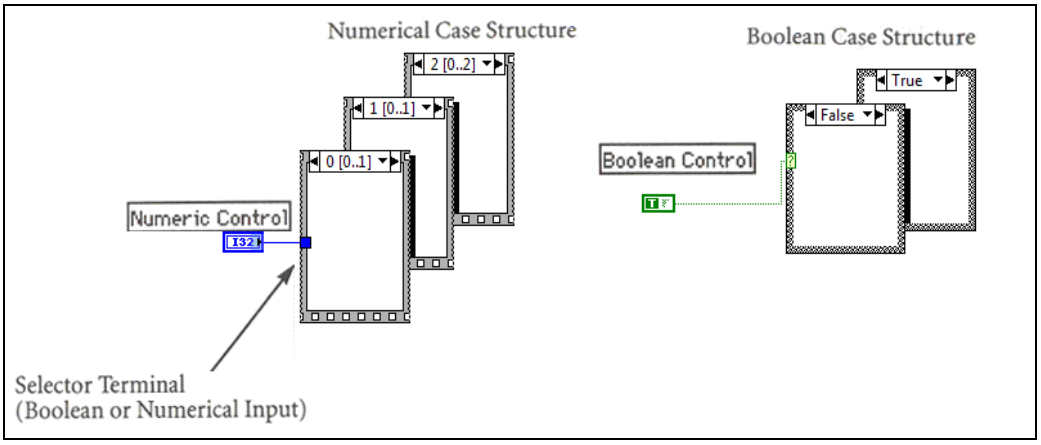
Aşağıda Şekil1.6'daki örneği inceleyelim. A döngüsünde, döngü sayacının toplamı çıkarılmaktadır. Her döngü, kaymalı kayıtla saklanmış yeni toplamdır. Döngü sonunda, sayısal göstergeden toplam 45 kere geçilmiştir. B döngüsünde kaymalı kayıt yoktur, dolayısıyla döngüler arası saklanmış değer yoktur. Bunun için, her çıkışa sıfır eklenmelidir ve böylece döngü sonunda 9 kere geçilmiş olur.



Şekil 1. 6. Örnek kaymalı kaydediciler

1.2.3.Durum Yapıları

Durum yapısı, 'If Then Else' tipindeki şartlı metinleri çalıştıran bir LabVIEW metodudur. Functions menüsünde *Structs&Constants* paletinde bulunmaktadır. Aşağıda görüldüğü gibi durum yapıları en az iki diyagram ya da seçici terminale girilmiş olan rakamsal bir ifade veya boolean değerine bağlı durumlardan oluşur. Eğer seçici terminale boolean değer girilmiş ise 0'dan 215-1'e kadar durumlarla karşılaşır. Başlangıçta sadece 0 ve 1 durumları kullanılabilir, fakat kolaylıkla daha fazla eklenebilmektedir. İlk olarak panele yerleştirdiğinizde durum yapıları *boolean* şeklinde çıkar, seçici terminale rakamsal ifade girildiğinde sayısal değerler olarak değişir [2].



Şekil 1.7. Durum yapıları

Durum yapıları, çok çeşitli alt diyagramlara sahiptir. Fakat bir anda kartların istif edilmiş Şekil 1.7'de olduğu gibi bir durumu görmek mümkündür. (left) ve (right) düğmelerine basılınca, bir önceki ya da bir sonraki diyagram geçebilirsiniz.

Eğer seçiciye çok küçük sayı girilir ise, LabVIEW bu sayıya en yakın tamsayı değerine çevirir. LabVIEW negatif sayıları 0'a zorunlu çevirir ve en büyük sayılı durumlardan daha büyükse onu da en büyük durumlarla eşitler.

Seçici sol kenarın herhangi bir yerine sabitlenirse, ancak ona bir şeyler girmeniz gerekmektedir. Seçici otomatik veri girmeye ayarlanır. Eğer girişlerin değerini sayısaldan booleane çevirirseniz, 0 ile 1 durumu *FALSE* (Yanlış) ve *TRUE* (Doğru) olur. Eğer başka

durumlarda varsa (2'den n'ye) LabVIEW onları rasgele veri sayarak tanımaz. Onun için, programı çalıştırmadan önce fazla durumlar silinmelidir.

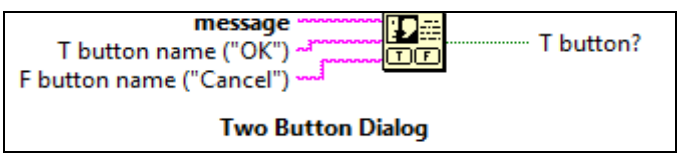
1.2.4. Girdi ve Çıktıların Elde Edilişi

Girdi terminalinden girilen tüm veriler (Tüneller ve seçici terminalinden), tüm durumlarda kullanılabilir. Girdi verilerin veya çıktılarını uygulama durumları da zorunlu değildir, ancak herhangi bir durumda çıktılar uygulanacaksa bu önceden yapılmalıdır. Her durum sonunda çıktı tüneline veriler girilmezse tünel beyazlaşır ve çalışma bozuldu düğmesi sayfaya çıkar.

Eğer durum yapılarının kenarına gelinir ise, elinizdeki durumda *Add Case After* ve Add Case Before menüsü çıkar. Bunlar eldeki durumdan sonra ve önceki durumları oluşturmak içindir. Aynı zamanda cari durumun *Duplicate Case'i* seçerek kopyası elde edilebilinir. Remove Case ile de cari durumunda silinebilinir [2].

1.3. Diyalog Kutuları

Aşağıda Şekil 1.8'de olduğu gibi, *One Button Dialog* ve *Two Button Dialog* fonksiyonları diyalog kutularını getirir. Functions menüsünden *Time&Dialog* paletinden bulunmaktadır. *One Button Dialog*, siz *Ok* düğmesine basana kadar durur, oysa *Two Button Dialog Ok* ya da *Cancel* düğmesine basana kadar durur. 'Button Name' satırında onların isim değişikliğini girebilirsiniz. Diyalog kutuları kipe aittir, yani açık durumda LabVIEW penceresini tıklamazsınız [3].



Şekil 1.8. Diyalog butonları

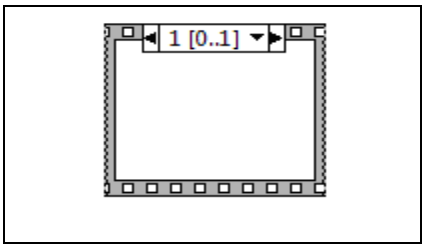
VI Logic

```
if (Number >= 0) then                                    (2)
Square Root Value = SQRT (Number)
else
Square Root Value = -99999.00
Display Message “Error…Negative Number”
end if
```

1.4. Dizi Yapıları

Kontrol akımı adı altında, dizi elementlerinden oluşan sırasıyla çalıştıran programı elde ediniz. Basic, C ve daha başka programlama dilleri aslında kontrol akışına sahiptir, çünkü programda ifadeler sırası ile girilir ve sırası ile çalışır. Veri akışı taslağını kontrol akışıyla elde etmek için, LabVIEW dizi yapısını kullanır. Dizi yapısı, 0. çerçeve, ardından 1. çerçeve, sonra 2. çerçeve, son çerçeve çalışana kadar. Ne zaman ki çerçeve tamamlanır, veri yapıyı terk eder.

Dizi yapısı aşağıdaki Şekil 1.9’da gösterildiği gibi, film çerçevesi gibi görünmekte ve *Functions* menüsünde *Structs&Constants* paletindedir. Durum yapılarında olduğu gibi, sadece bir çerçeve görülebilir, diğer çerçeveleri görmek için üst oklara tıklanmalıdır. Yapı kenarlarından çıkartarak *Add Frame After* ya da *Add Frame Before’*dan yeni çerçeve oluşturulabilir [2].



Şekil 1.9 . Dizi yapısı

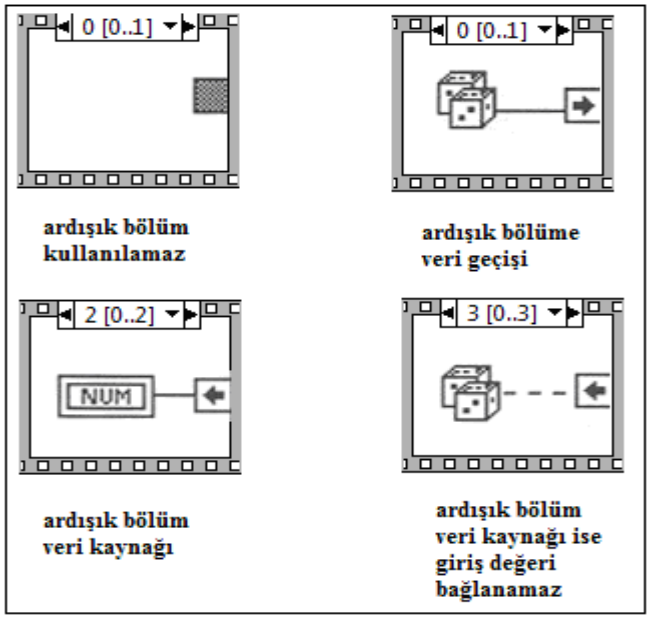
Dizi yapısı verilere bağlı olmayan nokta sınırlarını kontrol etmek için kullanılır. Aynı zamanda her bir çerçeve ile, blok diyagramın kalan kısmı, nokta sıra işleyişi verilerine bağlı kalarak kullanılabilir.

Dizi yapılarının çıktı tüneli, durum yapılarında her durum çıktıları için bir veri kaynağı olduğu gibi değil, sadece bir veri kaynağına sahiptir. Çıktı herhangi bir çerçeveden gelebilir, fakat tek şeridi geçtikten sonra değil, 0 giren yapıyı tamamladıktan sonra çıktığını aklınızda tutun. Girdi tünelindeki veri, her çerçeve için kullanılabilir.

1.4.1. Dizi Yerleşimi

Bir diziden bir alt diziye veri aktarımı için, dizi yerleşimi terminali kullanılmalıdır. Pop – up menüsünde *Add Sequence Local*'i seçerek elde edebilirsiniz. Ancak, eğer yerel dizi veya alt diyagram penceresine yakın ise bu seçeneği kullanamazsınız. Sınır içerisinde herhangi bir yere taşıyabilirsiniz. Terminali taşımak için de local pop – up menüsünden Remove komutunu kullanabilirsiniz.

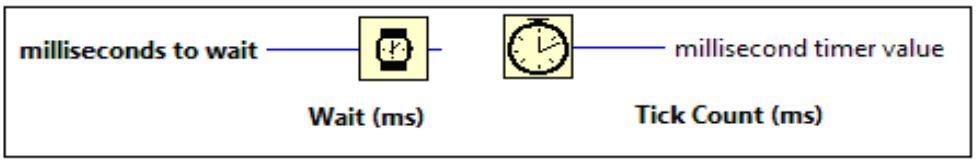
Diyagrama ilk geldiği anda dizi yerel terminali sadece sarı kutudur. Yerel diziye veri kaynağını girdiğiniz zaman, veri kaynağını içeren çerçevesinde dışa işaretli terminal ortaya çıkar. Alt dizi çerçevesinde içe işaretli terminal ise, bu terminalin bu çerçeve için veri kaynağını gösterir. Kaynak çerçevelerden önceki çerçeve, yerel diziyi kullanamaz (Sonunda değer veremez) ve bulanık dörtgen olarak ortaya çıkar. Aşağıdaki Şekil 1.10 yerel dizi terminalinin çeşitli formdaki halidir [2].



Şekil 1.10. Dizi yerleşimi

1.5. Zamanlama

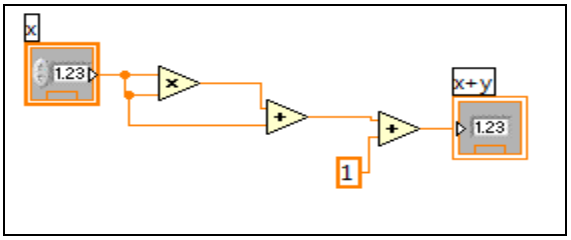
Bazen, VI'nın zamanlaması için kontrol ve monitör kullanabilirsiniz. *Wait* (ms) ve *Tick Count* (ms), Functions menüsünün *Time&Dialog* paletindedir ve işlemleri yerine getirirler. Wait (ms), işleme devam etmeden önce milisaniyelere kadar beklemeyi sağlar, eğer siz döngünüzün saniyede bir vb. çalışmasını istiyorsanız. *Tick Count* (ms), iç saati milisaniyeler olarak değerlendirir, gelecek alıştırmada olduğu gibi genelde geçmiş zamanı hesaplamada kullanılır. İç saat büyük bir kararlılığa sahip değildir, saatin bir saniyesi Windows bilgisayarında 55 ms, Macs'da 17 ms civarındadır, ki işletim sistemi ile sınırlıdır ve LabVIEW bu ortamda çalışmaz. Bu fonksiyonlar aşağıdaki Şekilde gösterilmiştir [3].



Şekil 1.11. Zamanlama

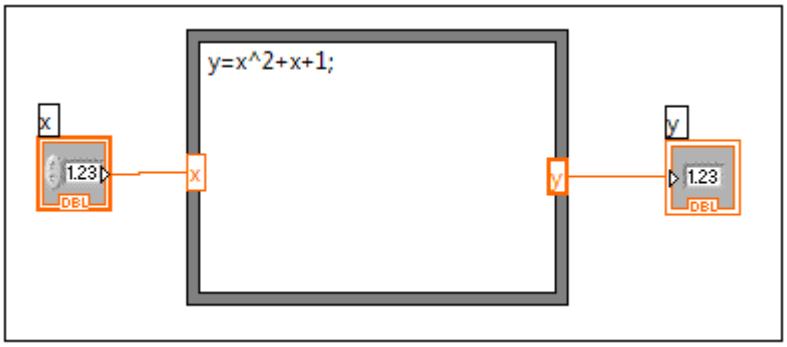
1.6. Formül Düğümleri

Formül düğümleri, blok diyagrama direk olarak cebirsel formülleri girebilmek için kullanılan büyüklükleri değiştirebilen kutulardır. Bu özellik karmaşık formüllerin çözümünde çok kullanışlıdır. Örneğin, Y = x2+x+1 eşitliğini ele alalım. Eğer, LabVIEW aritmetik fonksiyonları kullanılarak çözümlenirse, onun blok diyagramı aşağıdaki gibi olur [3].



Şekil 1.12. Formül Düğümleri

Aynı eşitliği, formül düğümünü kullanarak Şekil 1.13. de ki gibi kullanabiliriz.



Şekil 1.13. Formül Düğümü

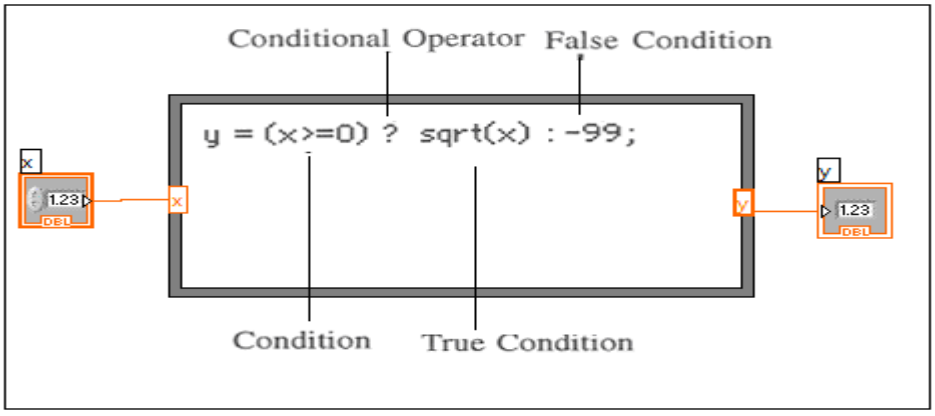
Formül düğümü ile direk olarak tamamlanmış formül ya da formüller, tamamlanmış blok diyagramın alt kısmına girilir. Bazen Şekilde kutu içine de formüller girilir. Formül düğümünün girdi çıktı terminalini oluşturmak için düğüm kenarında Add Input ya da Add Output seçilir. Ardından kutulara değişken tanımlamaları girilir. Adların limiti iki karakterdir ve duyarlı durumdadır. Her formül (;) ifadesi sonlandırılmalıdır.

Functions menüsünden *Structs&Constants*'ı seçerek formül düğümünü blok diyagrama yerleştirilir.

Aşağıdaki örnekte formül düğümü içinde şartlı dallanmayı deneyebiliriz. Şekil 1.13 de olduğu gibi, öyle ki eğer x pozitif ise onun karekökünü bulsun ve y'ye aktarsın. Eğer x negatif ise, kod y'ye -99'u aktarsın.

```
İf (x >= 0) then                                    (3)
y = sqrt (x)
else
y = -99
end if
```

Aşağıda Şekil 1.14'de olduğu gibi, formül düğümünü kullanarak parça kodu uygulayabiliriz.



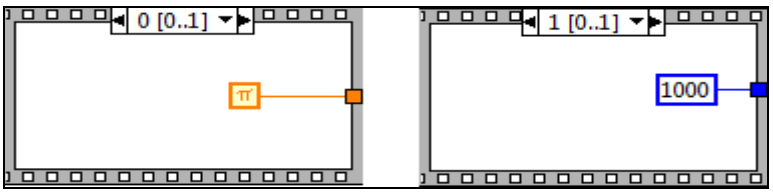
Şekil 1.14. Formül Düğümü

1.6.1. Yapı Kuramları ile İlgili Problemler

Problemlerle ilgili teşhis ve bağlantı problemlerini ortaya çıkarmada yardımcı olur.

1.6.2. Yerel Dizilere Birden Fazla Değerin Girilmesi

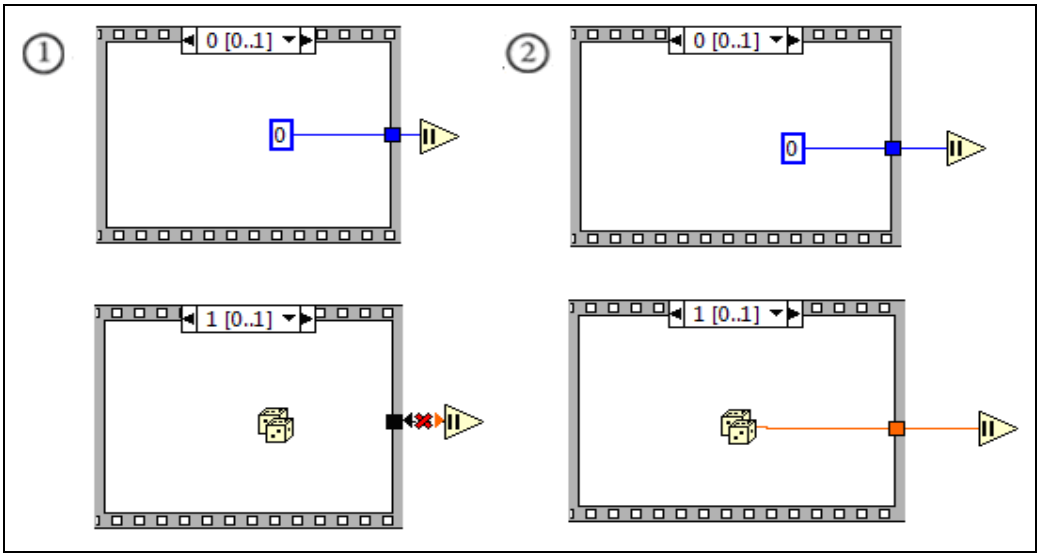
Dizi yapısının alt dizi çerçevesinde bu değeri kullanmanıza karşılık, sadece bir çerçevesinin değişkenine değer girebilirsiniz. Aşağıdaki Şekil 1.15'de 0. çerçevenin yerel dizisinde π değerinin girilmesi gösterilmiştir. Eğer 1. çerçevenin aynı değişkenine aynı değeri girmeye kalkarsak kötü sonuç elde ederiz. Bu hata çok kaynaklı hataların bir tanesidir [2].



Şekil 1.15. Yerel Diziler

1.6.3. Durum Yapılarının Tüm Yapılarında Tünelin Çalışma Bozulması

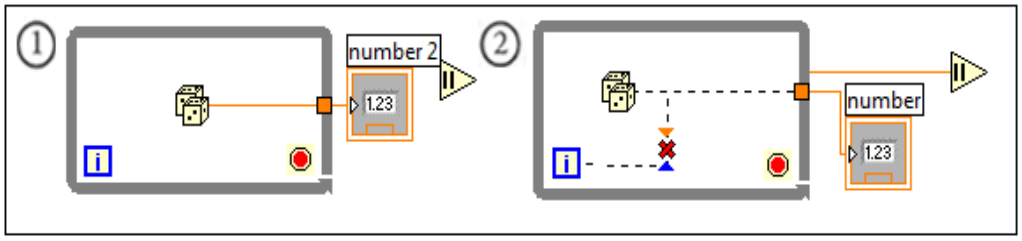
Aşağıda Şekil 1.16'da birinci bölümünde görüldüğü gibi, tüm durum çıktılarını nesneye bağlantı yapmamış isek, bu yapınızın sonuçları yanlış tünele girmiş olur. Bu kaynak yokluğunun bir tanesidir. Çünkü eğer çalışırsa bir durum değeri için bir veri sağlanmaz. Durum yapısı her çalışmanın sadece bir çıktı değerini verir ve sadece bir kere çalıştırır. Ancak bu olay çoklu kaynak ortamından değildir.



Şekil 1.16. Durum Yapıları

1.6.4. Tünellerin Üst Üste Gelmesi

Oluşturduğumuz gibi LabVIEW, tünelleri ortaya çıkarır ve bu tüneller üst üste gelebilir. Şekil 1.17 de görüldüğü gibi üst üste gelme durumu çalışmasını etkilemez, fakat eklemeleri zorlaştırabilir. Tünellerin üst üste gelmesinden kaçınmak iyi olur. Eğer böyle bir şey meydana gelirse, birini açacak Şekilde diğerini taşıyabiliriz [4].

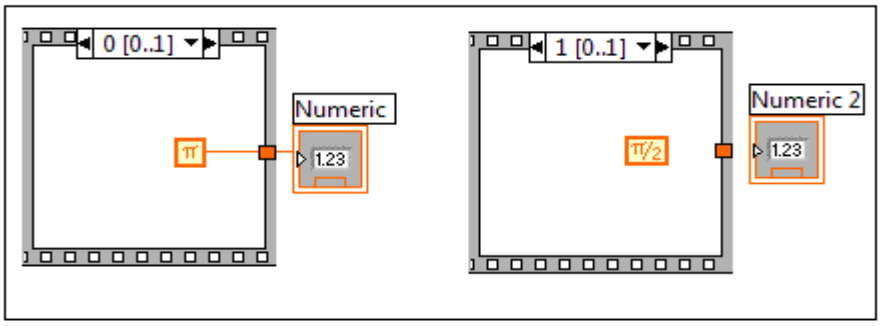


Şekil 1.17. Tünellerin Üst Üste Gelmesi

Hangisinin üst olduğunu söylemek zordur. Eğer alta kaydıysa girmek için hat yapabiliriz.Yapı içindeki nesneden yapı dışındaki nesnelere aktarma gerekiyorsa, mevcut yapı sınırlarında yeniden aynı işlemi yapmalıyız. Onun yerine, ikinci bir tüneli işleme sokabiliriz.

1.6.5. Dizi Yapısının Çoklu Çerçevesinden Çıktı Alma

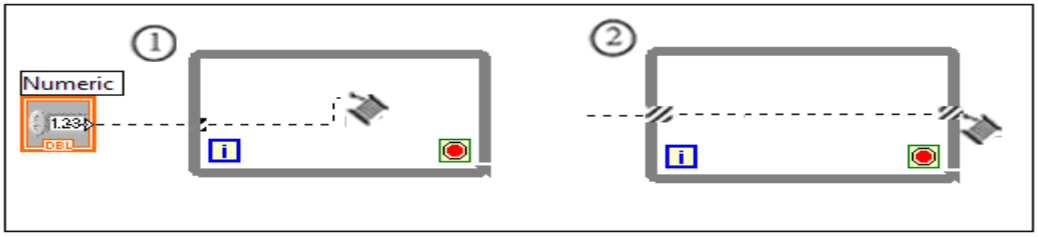
Aşağıda Şekil 1.18 'de çok kaynaklı hataların diğer versiyonları gösterilmiştir. İki yapılı çerçeveler bir tünele değerleri girmeye çalışıyorlar. Tünel beyaza dönerek yapılan işlemin hatalı olduğunu haber eder.



Şekil 1.18. Dizi yapısının çoklu çerçevesinden çıktı alma

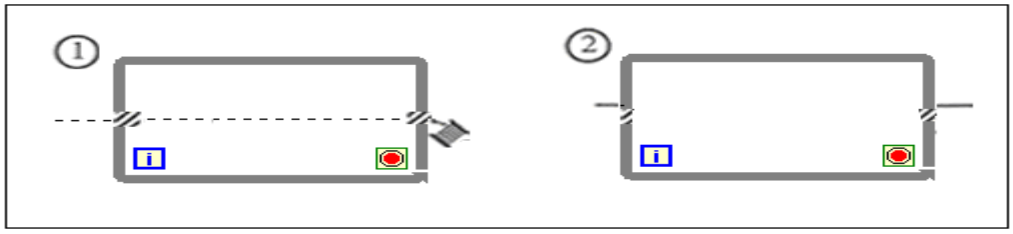
1.6.6. Yapı İçerisinden Ziyade Altından İşleme Sokma

Şekil 1.19'da olduğu gibi, yapı içinden işlem yapmak için yapı kenarına ya da iç nesnelere tıklanması gerekmektedir [4].



Şekil 1.19. Yapı altından işleme sokma

Eğer, iç nesneye ya da kenara tıklamazsanız, Şekilde olduğu gibi işleme yapı altından geçer.



Şekil 1.20. Yapı altından işleme sokma

İşlem aletleri yapının sol kenarını kestiği zaman, fare düğmesine dokunur dokunmaz, LabVIEW'in yeni tünel oluşturacağını göstermek içi ışıklı tünel ortaya çıkar. Eğer fareyi tıklamadan yapı içerisinden fareyi taşımaya devam edersek, yapı sağ kenarına dokununcaya kadar 2. ışıklı tünel ortaya çıkar. Eğer, fareyi tıklamadan işlem aracını sağ kenarı aşacak Şekilde taşımaya devam edersek, tünellerin ikisi de kaybolur ve işlem içe tercihen alttan geçer.

1.6.7. Paketleme

LabVIEW, alt diyagramın tekrar çalışması için While döngüsü ve For Döngüsü gibi iki yapıya sahiptir. Her ikisi de değişken kutulardır. Alt diyagramın tekrarlanması için devre sınırları içerisinde olması gerekir. Şartlı terminalin değeri TRUE (Doğru) olduğu sürece While döngüsü devrededir. For döngüsü özellikle birkaç defa çalışır.

While döngüsü ve For döngüsünde kullanılabilen kaymalı kaydedici, bir döngüden diğerinin başlangıcına transfer edilen değerlerdir. Kaymalı kaydediciyi birkaç önceki döngülerin değerlerinin transferi için ayarlayabiliriz. Her döngü için gösterilmesi isteniyorsa, kaymalı kaydedicinin sol terminaline yeni element eklenmelidir.

Veri akışı kontrolü için, LabVIEW iki yapıya sahiptir; durum yapısı ve dizi yapısı. Sadece bir durum yapısı ya da çerçeve gözükür.

Durum yapısı, onun seçici terminaline girdilere bağlı farklı diyagramların dallanmasında kullanılabilir. Durum yapısının her durumu için alt diyagramı sınır içine yerleştirilir. Durum seçimi ya boolean (2 durum) yada sayısal (215-1 durum) olabilir. LabVIEW seçici terminaline girilen boolean veya sayısal veri tipine göre otomatik olarak ayarlanır [2]. Dizi yapısını, özellikle sınırlanmış fonksiyonlar diyagramın çalıştırmada kullanılır. Diyagramın çalışacak tek kısmı 1. çerçevede (0. çerçeve) yerleştirilir, 2. kısmı 2. çerçeveye vs.

Yerel diziler, dizi yapısı çerçeveleri yapısı arasında değerleri geçirmede kullanılır. Yerel diziden geçen veri, çerçeveden önce gelen çerçevede değil sadece alt dizi çerçevelerinden oluşturulmuş yeni yerel dizi şeridinde kullanılır.

Formül düğümleri ile diyagrama direk formül girilebilir. Bu özellik tamamlanmış fonksiyonel eşitliklerde çok kullanışlıdır. Değişken adları durum için duygusal olduğu ve ifadelerin (;) ile sonlandırıldığı unutulmamalıdır.

VI zamanlayıcısı kontrolü veya gösterimi için Functions menüsünden *Time&Dialog* paletinden yararlanılır. *One Button Dialog* ve *Two Button Dialog* fonksiyonları seçilen mesajı içeren diyalog kutlarını ortaya çıkarır. *Wait* (ms) fonksiyonu VI'yı birkaç milisaniyede durdurur. *Tick Count* (ms) iç saat değerini gösterir.

1.7. Diziler ve Kümeler

1.7.1. Genel Bakış

Bu bölümde, iki yeni veri tipinden (Diziler ve kümeler) bahsedilecektir. Bu birleşik veri tipleri, veri depolama ve idaresinde büyük esnekliğe izin verir. Dizin yönetimi için gömülü fonksiyonların nasıl kullanıldığı bu bölümde ele alınmaktadır. Dizi kontrolleri ve göstergelerinin nasıl kullanıldığını, bir dizi oluşturmak için otomatik indekslemeyi

kullanmak, gömülü dizi idare fonksiyonları, polymorphism fikrini kavramak, dizilerin kümelerden nasıl ayrıldığını anlamak, Bundle ve Unbundle fonksiyonlarını kullanarak bir kısmının ayrımı ve toplanması bu bölümde anlatılmaktadır [2].

1.7.2. Diziler

Dizi aynı veri tipteki veri elementlerinin koleksiyonudur. Bir dizi hafıza uygunsa her boyuta 231 element düşecek Şekilde bir ya da fazla boyutlu olabilir. Dizi, tablo ve grafik dışında LabVIEW'de herhangi bir tipte olabilir. İndeks ile dizi elementlerine ulaşılabilir. İndeks 0'dan N - 1'e kadardır, N dizideki elementlerin sayısı. Bir boyutlu (1D) dizi aşağıda Şekil 1.21'de verilmiştir. 1. elementin indeksi 0, 2.'nin 1 vs.

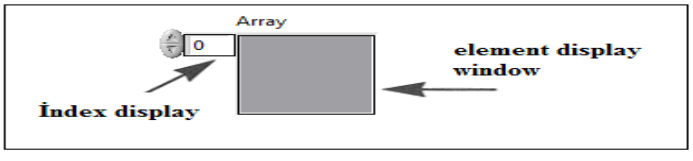
0	1	2	3	4	5	6	7	8	9
1.2	3.2	8.2	8.0	4.8	5.1	6.0	1.0	2.5	1.7

Şekil 1.21. Dizi yapısı

Dalganın her noktasına dizideki bir elementi karşılayacak Şekilde, dizide dalga formu kullanılabilir. Eğer, DAQ ortamında birkaç kanaldan dalgalar varsa, onlar 2 boyutlu (2D) dizide bulunur. Öyle ki 2D her sütunu bir kanal verilerin karşılığıdır.

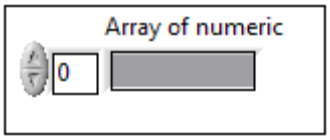
1.7.3. Dizi Kontrol ve Göstergelerin Oluşturulması

Dizi ve küme gibi kompleks tipli verileri yapmak için iki adım gereklidir. Şekildeki gibi dizi kontrolü veya göstergesini yapmak için *Array Shell* ile veri nesnesi karıştırılabilir. Veri sayısal, boolean, string olabilir. *Array Shell*, Controls menüsü Structs&Constants paletinde bulunabilir [4].



Şekil 1.22. Dizi

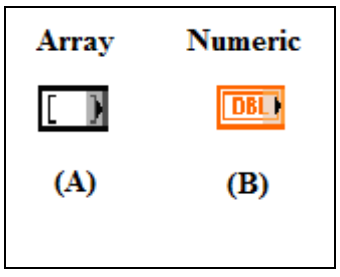
Dizi oluşturmak için, veri nesnesi element pencere gösterimine taşınabilir ya da nesne direk olarak pencerenin pop–up menüsünü kullanarak pencere içerisine bırakılabilir. Şekil 1.22'de ki gibi yeni veri tipine uygun gelecek Şekilde element penceresi boyut alır.



Şekil 1.23. Boyutlandırılmış Dizi

Eğer, element penceresinin boyutları değiştirilmek isteniyorsa, pozisyon aracı kullanarak standart boyutlarında olduğundan emin olunmalıdır. Köşesine gelerek yapılabilir. Eğer, daha fazla elementin görünmesi isteniyorsa, pozisyon araçlarından Şekilde kursör ile çekilebilir. Böylece, birçok element görünebilir. İndeks göstergesindeki element daima sayı gösterimindekinin karşılığıdır.

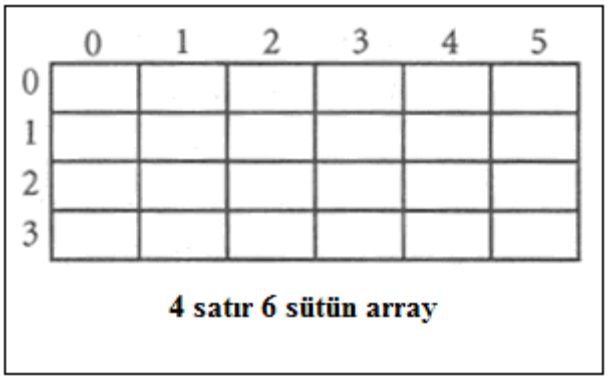
İlk başta Array Shell ön panele bırakıldığı zaman onun blok diyagram terminali siyahtır, veri tipi tanımsız karakterdir. Artı köşegenli parantezi vardır, Şekildeki gibi LabVIEW'in dizi yapılarını belirtme yoludur. Diziye veri işlendiğinde (Element gösterge penceresine kontrol veya gösterge yerleştirilmesiyle), dizi blok terminalinin rengi siyahtan yeni veri yazılış rengini alır (Parantezi etkilemez), Şekil (B). Dizinin tek değer taşımaktan gittikçe kalınlaştığı görülebilir.



Şekil 1.24. Dizinin kalınlaşması

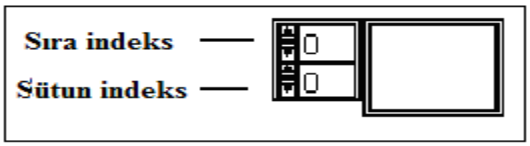
1.7.4. İki Boyutlu Diziler

İki boyutlu dizilerin, sütun ve satır indisi sıfırla başlar, elementlerin girilmesi için iki indise ihtiyaç vardır.



Şekil 1.25. Dizide boyut

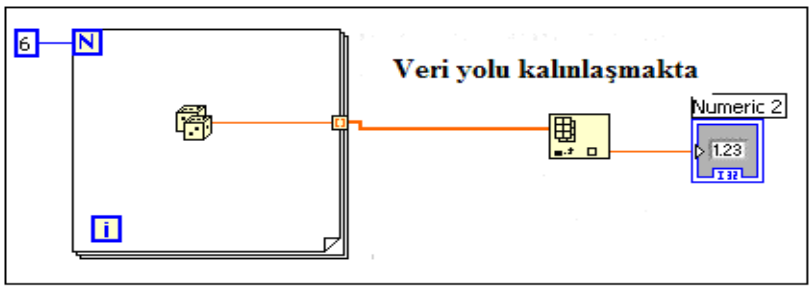
Add Dimension menüsü seçilerek dizi kontrol veya *İndeks Display* göstergesi kullanılarak boyut eklenebilir. Şekil 1.26'da 2D dijital kontrol dizisi verilmiştir.



Şekil 1.26. 2D Dijital Kontrol Dizisi

1.7.5. Otomatik İndekslemeyi Kullanarak Dizi Oluşumu

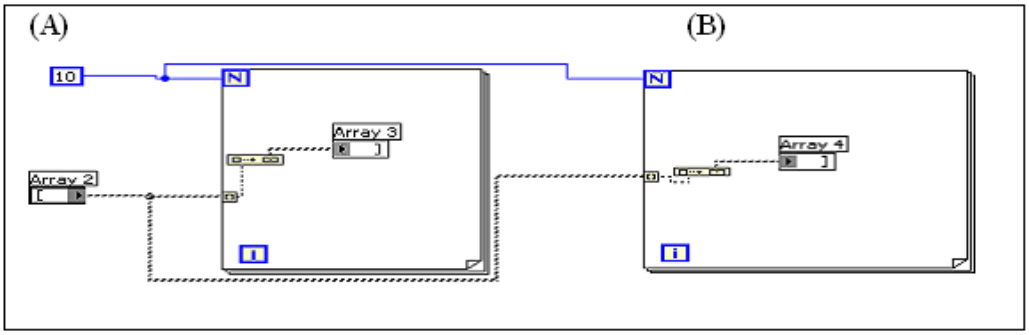
For döngüsü ve *While* döngüsü ile kendi sınırları içerisinde indeks ve hesap yapılabilir. Bu özelliğe otomatik indeksleme denir. Aşağıda Şekil 1.27'de *For* döngüsünün kendi sınırları içerisinde otomatik indeks yapması verilmiştir. Döngüler tamamlandıktan sonra dizi döngü dışı göstergeye geçer. Döngü sınırında girdilerin diziye geçmesiyle gitgide kalınlaştığına dikkat edilmelidir [5].



Şekil 1.27. Otomatik indeksleme

Dizi oluşturmadan döngü dışı tünel değeri gerekli ise, otomatik indekslemeyi tünel üzerindeki (siyah kare) pop – up menüsünden Disable Indexing seçilerek kullanılmaz hale getirilebilir. Şekil 1.27'de görüldüğü gibi otomatik indeksleme kullanılmaz ve sadece döngü dışına Random Number (0-1) fonksiyonun son değeri geri döner. Bağlantının döngüyü terkinden sonra etki halinde kaldığına dikkat edilmelidir.

İndeksleme aynı zamanda devreye dizi girmede kullanılır. Eğer indeksleme Şekil 1.28.(A)'da ki gibi kullanılırsa, döngü her döngü zamanının dizi elementinden indeksi siler. Şekil 1.28.(B)'de ki gibi indeksleme kullanılmaz ise, giriş dizi döngüye bir kere girer.

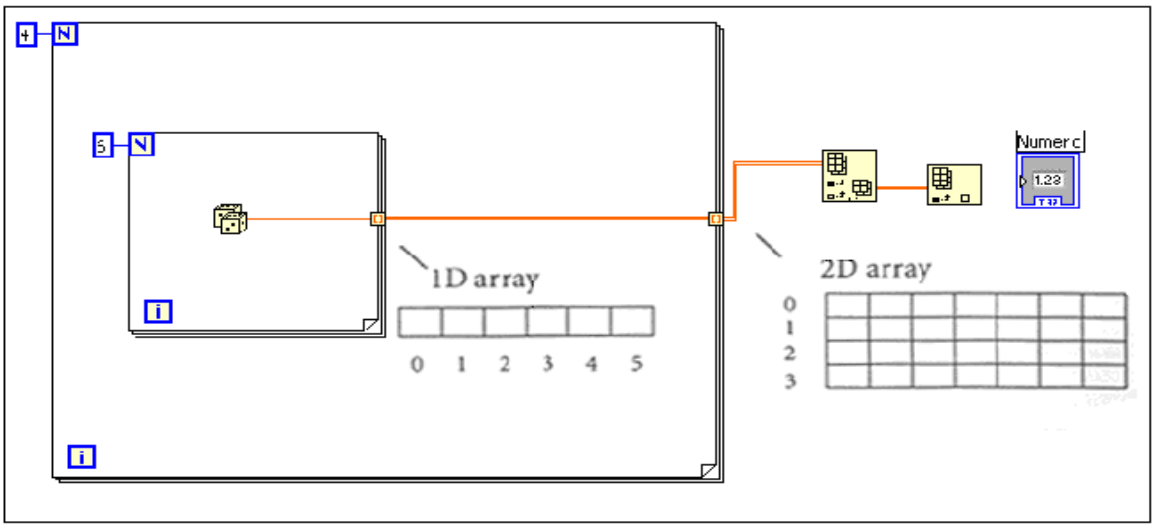


Şekil 1.28. Dizide indeksleme

For döngüsü bazen dizi işlemede kullanıldığı için, döngüye dizi girdi ve çıktılarda LabVIEW otomatik indekslemeyi yanlış Şekilde kullanabilir. Yanlışlıkla, LabVIEW *While* döngüsü için otomatik indekslemeyi kullanmayabilir. Bu durumda dizi üzerinde tünel çıkartarak pop–up menüsünden *Enable Indexing* seçilmelidir. İndeksleme durumuna çok dikkat etmek gerekir ki, hataya düşmekten alı koysun.

1.7.6. İki Boyutlu Dizileri Oluşturma

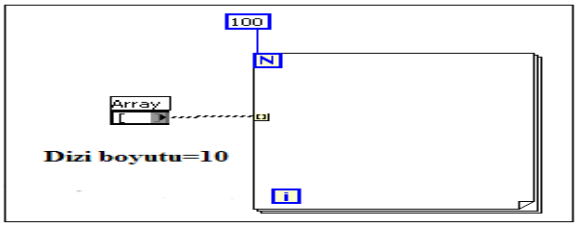
Birbirinin içinde olacak Şekilde iki *For* döngüsünü iki boyutlu dizi oluşturmada kullanılabilir. İçteki *For* döngüsü satır için, dıştaki *For* döngüsü bu istifleri matrix sütunlarını doldurmak için kullanılır. Şekilde otomatik indeksleme rastgele sayılı 2D dizi oluşturmak için 2 tane For döngüsü gösterilmiştir [5].



Şekil 1.29. İki boyutlu dizileri oluşturma

1.7.7. *For* Döngüsü Sayacını Kurmada Otomatik İndeksleme Kullanımı

For döngüsüne dizi girilmesinde otomatik indeksleme kullanılıyorsa, LabVIEW otomatik dizi boyutunda sayaç kurar, bu N'ye kadar inmekten kurtarır. Eğer birden fazla dizi otomatik indeksleme yapılırsa ya da N'ye kadar girdi girerek sayaç kurulursa, sayaç iki seçenekten küçük olan haline gelir. Aşağıda Şekil 1.30'da dizi boyutu N değildir. For döngüsü sayacı kurar ve ikisinden küçüğüdür.

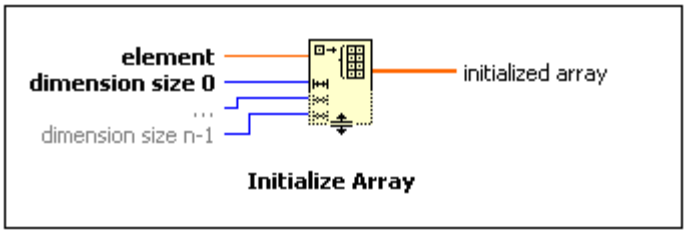


Şekil 1.30. For döngüsünde otomatik indeksleme

1.7.8. Dizilerin İşlenmesi İçin Fonksiyonlar

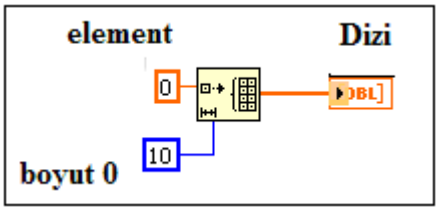
LabVIEW, Functions menüsünde *Array&Cluster* paletinde dizi işlemlerine sahiptir. Daima dizilerin sıfır indeksli (İlk elementin indeksi 0, 2.'nin 1 vs.) olduğu akılda tutulmalıdır. Bazı fonksiyonlar aşağıda incelenmiştir.

Initialize Array, seçilen değerin boyutlu diziyi yeniden oluşturur. Bu özellik kesin boyutu bilinen dizi için hafıza yerleşiminde kullanılır [4].



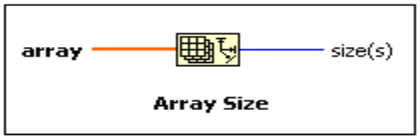
Şekil 1.31. Başlangıç dizisi

Bu Şekilde 1 boyutlu dizi, her elementi sıfırı içeren 10 elementli dizi yeniden başlatılmıştır.



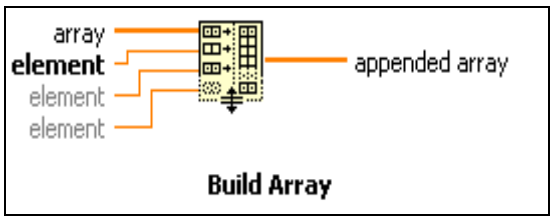
Şekil 1.32 . 1 boyutlu 10 elementli dizi

Array Size, diziye girilen elementlerin sayısını verir. Eğer girilen dizi n boyutlu ise, dizi büyüklüğü n elementli, 1 boyutlu her elementi boyut büyüklüğünü içerir.



Şekil 1.33. Diziye girilen elementlerin sayısı

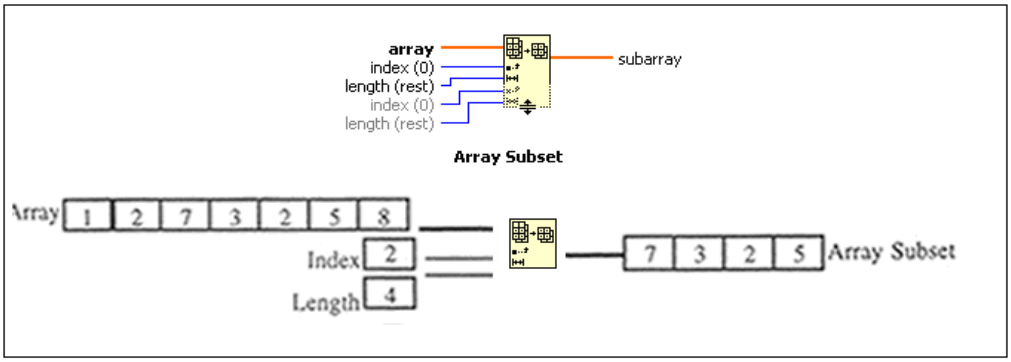
Build Array, iki diziyi birleştirir ya da ekstradan elementleri bir diziye ekler. Fonksiyonu diyagram penceresine eklendiğinde halini alır. Girdi sayısını artırmak için fonksiyon büyüklüğü değiştirilebilir. *Build Array*'ın iki tip girdisi vardır, dizi ve element, bu yüzden tek değerli elementler ile diziler birleştirilebilir.



Şekil 1.34. İki diziyi birleştirme

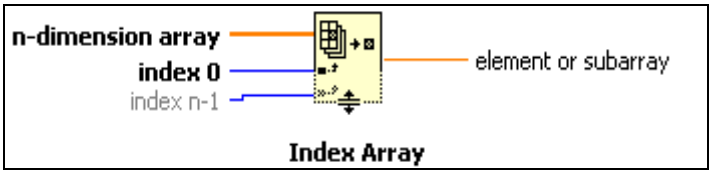
Özellikle *Build Array* fonksiyonunun girdilerine dikkat edilmelidir. Dizi girdileri element girilmediği sürece köşeli paranteze sahiptir. Onlar kesinlikle değiştirilemez ve eğer dikkat edilmezse kötü bağlantı karışıklığına neden olabilir.

Array Subset.,indekste başlayıp element uzunluğunu içeren kısmı verir.



Şekil 1.35. Alt dizi

Index Array, dizi elementi indeksler. Aşağıda Şekil 1.36'da olduğu gibi, *Index Array* fonksiyonu dizinin 3. elementi ile işlemdedir. 3. elementin indeksinin iki olduğu unutulmamalıdır. Çünkü indeks sıfırdan başlar ve ilk elementin indeksi sıfırdır.



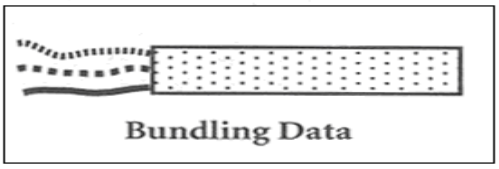
Şekil 1.36. Dizi elementi indeksleme

Burada *Index Array* fonksiyonu sıradan skaler element çıkarıyor. Artı bu fonksiyonu bir ve daha fazla 2D diziyi orjinalinin alt dizisini bölmede kullanılır.

İndeksleme kullanılmadığı zaman, indeks terminalini şeklinin doludan boş kutuya değiştiği unutulmamalıdır. Aynı menüde *Enable Indexing* komutuyla tekrar çalıştırılabilir.

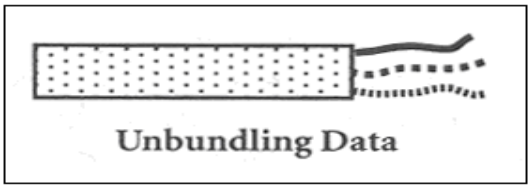
1.8. Kümeler

Küme, dizi gibi verileri guruplandıran veri yapısıdır. Küme farklı tipteki verileri guruplayabilir, ancak onun asıl işi, pascal ya da program altında C'ye kaydetmektir. Küme, daha birçok telefon kablosunu andıran bağlantıların paketi olarak düşünülebilir. Kablodaki her tel kümenin farklı elementine karşılık gelir. Blok diyagramda küme tek bir telden kurulduğu için, kümeler tel karışıklığını ve alt VI'ların gerektirdiği bağlantı terminallerini azaltır. Veriler onaylandığında küme veri tipinin sık sık ortaya çıktığı görülebilir [2].



Şekil 1.37. Veriyi kümeleme

Küme ve dizi elementleri de sıralanmış olmasına rağmen, bir kere indeksleme yerine küme elementlerinin tümü bir kere de serbest bırakılabilir. Kümeyi serbest bırakmak, telefon kablosunu dağıtarak çeşitli renkte telleri ortaya çıkartmak gibi düşünülebilir. Dizilerden farklı kümelerin büyüklüğü sabittir ya da sabit tel sayısına sahiptir.

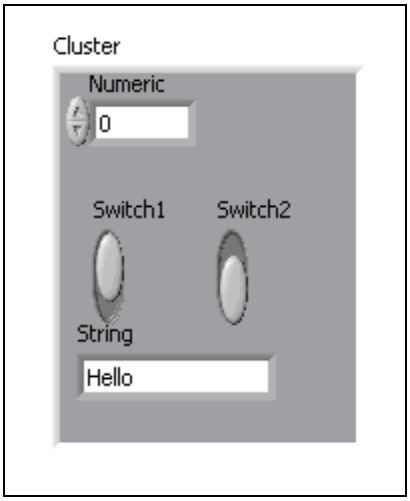


Şekil 1.38. Veriyi kümeden çıkarma

Küme terminali ancak aynı veri tipinde ise bağlanabilir. Yani iki kümede eşit sayıda element olmalı ve elementler birbirini karşılamalıdır. Kümelerde veri tipleri bir arada olduğu sürece, dizilerde olduğu gibi kümelerde de polimorfizm uygulanabilir.

1.8.1. Küme Kontrolü ve Göstergeleri Kurma

Küme gözünü (*Controls menüsü Array&Graph paleti*) panel penceresine yerleştirerek küme kontrolleri ve göstergeleri kurulabilir. Ardından, kümeye girdi yerleştirildikten sonra panel penceresi de LabVIEW'e yerleştirilebilir. Dizilerde olduğu gibi, nesneler direkt küme içine menü çıkartarak bırakılabilir ya da nesne kümeye taşınabilir. Küme içinde nesneler, kontroller ya da göstergeler olmalıdır. Aynı anda bir kümeye hem kontrol hem de gösterge birleştirilemez. Küme veri yönünün (kontrol ya da gösterge), ilk olarak yerleşimini tanır. Aşağıda Şekil 1.39'da kontrollü küme verilmiştir.



Şekil 1.39. Kontrollü küme

1.8.2. Küme Sırası

Küme elementleri kutuda yerlerine bağlı olmaksızın mantıksal sıraya sahiptirler. Kümeye girilen tek nesne elementi 0, element 1 vs. Eğer bir element silinirse, sıra otomatik ayarlanır. Sıra, küme kenarından ve pop – up menüsünden *Cluster Order*'ı seçerek değiştirilebilir. Yeni düğmeler palet yerini değiştirebilir ve küme Şekilde olduğu gibi değişmiş Şekilde ortaya çıkar. Kümeye özel elementleri girmek için, küme sırası takip edilmelidir. Element adla değil sırayla girilmelidir.

Element üzerindeki beyaz kutu küme sırasındaki yerini gösterir. Küme kursörüyle elemente tıklamak küme sırasında element yerlerini aletler paketindeki gösterilen sayıya çevirir. Bu alana yeni sayı girerek bu sıra değiştirilebilir. X düğmesine tıklayarak sıra eski haline döndürülebilir. Sırayı istenilen şekle getirildiğinde, giriş düğmesine tıklanarak saklanabilir ve küme sıra giriş modundan çıkılabilir.

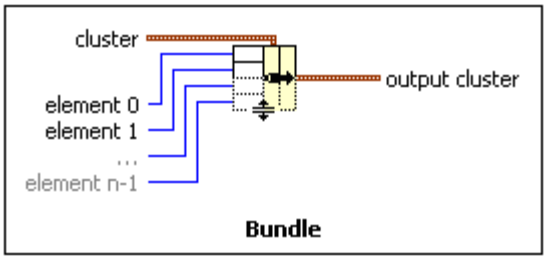
1.8.3. Veri Girişi ve Alt VI'ları Kurmada Küme Kullanımı

VI bağlaç levhası maksimum 20 terminali alabilir. VI, alt VI olarak çağrıldığında 20 terminale bilgi geçilemez. Bağlantı sıkıcı olabilir ve kolaylıkla yanlış yapılabilir. Kümeye birkaç kontrol toplamak ve kümeyi alt VI'ya geçirmek için birçok değer geçişi için bir terminal kullanılabilir. Büyük terminal için büyük tel kullanılabilir, sonuç yan teldedir.

1.8.4. Verileri Bir Araya Toplamak

Bundle fonksiyonu (*Array&Cluster menüsü*) özel girdileri yeni kümeye toplar ya da çıkış kümesinde yer değiştirmeye izin verir. Diyagram penceresine konulduğunda fonksiyon  olarak ortaya çıkar. Pozisyon aleti ile fonksiyon köşesinden çıkarak girdi sayısı yükseltilebilir. Girdi terminaline girildiğinde, girilen elementin veri tipini gösteren sembol orijinal boş terminalde ortaya çıkar. Sonuç olarak kümenin sırası paketlere girilenlerin sırasıdır, yukarıdan aşağı doğru.

Eğer sadece yeni küme oluşturulmak isteniyorsa, Bundle fonksiyonunun girdi kümesinin merkezine girdi girmeye gerek yoktur.



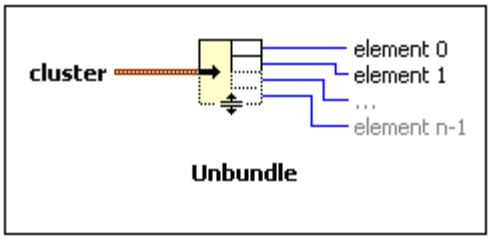
Şekil 1.40. Bundle

1.8.5. Küme Elementinin Yer Değişimi

Kümede elementin yeri değiştirilmek isteniyorsa, ilk olarak küme elementlerinin girdi terminalinin sayısını alacak Şekilde *Bundle*'ı büyütmek gerekir. *Bundle* fonksiyonunun orta terminali ile kümeyi bağlamak ve karşı girdileri elementlerin değerleri ile yer değiştirerek girmek gerekir.

1.8.6. Küme Parçalama

Unbundle fonksiyonu (*Array&Cluster menüsü*) kümeyi özel parçalara böler. Çıktı parçalarının sırası, küme sırası gibi yukarıdan aşağı doğru düzenlenir. Küme elementlerinin sırası sadece aralarında farklılık sağlanabilecek olaydır. Diyagram penceresine yerleştirildiğinde  fonksiyonu çıkar. Pozisyon aleti ile fonksiyon köşesinden çekerek çıktı sayısı artırılabilir. *Unbundle* fonksiyonu girdi kümesinin element sayısına eşit sayı alabilecek kadar büyütülmelidir, yoksa kötü bağlantı üretir. Girdi kümesini doğru olarak parçalamaya girildiğinde, çıktı terminallerinin önceki yüzü küme veri tiplerinin sembolleri ile değişecektir [5].



Şekil 1.41.Unbundle

Küme verileri girildiğinde, küme sırası tamamlayıcıdır. Örneğin, eğer iki boolean varsa anahtar 1 yerine anahtar 2 girmek hataya yatkındır, çünkü onların parçalama fonksiyonundaki referansları, ad değil sıradır. VI hatasız çalışır fakat sonuçlar doğru olmayabilir.

1.8.7.Paketleme

Aynı tip element koleksiyonuna dizi denir. LabVIEW'de diziler her tipte olabilir, yalnız *chart, graph* ya da başka dizi dışında. İki adımda dizi oluşturulmalıdır. İlki, dizi kutusu (*Controls menüsü Array&Graph paleti*) pencereye yerleştirilmelidir ve ardından ayarlanmış kontrol veya gösterge kutu içine alınmalıdır.

LabVIEW, diziyi beceriyle kullanmak için birçok fonksiyon, *Array&Cluster* menüsünde *Build Array, Index Array*, sunar. Bu bölümde, 1D dizi ile çalışmak için dizi fonksiyonları kullanıldı. Fakat bu fonksiyonlar akıllı ve çok boyutlu dizilerde benzer Şekilde çalışır.

Her ikisi de *For* döngüsü ve *While* döngüsü dizileri kendi sınırları kendi sınırlarında otomatik indekslemeyi kullanarak hesaplayabilir, bu dizi kurma ve çalıştırmada kullanışlı bir özelliktir. Yanlışlıkla, LabVIEW For döngüsü için indekslemeyi kullandığını ve While döngüsü için de indekslemeyi kullanmadığı unutulmamalıdır.

Çeşitli veri tipindeki girdi verilerini ayarlama fonksiyonunu özelliğine polimorfizm denir. Bir çok fonksiyonun polimorfik olduğu gibi, aritmetik fonksiyonların polimorfik özellikleri de görülmektedir [2].

Kümelerde veri grubudur, ancak diziden farklı olarak türlü tipte veriyi kabul eder. Onlar iki adımda ön panelde kurulmalıdır. Küme içindeki nesnelerin ya kontroller ya da göstergeler olacağı akılda tutulmadır. Bir kümede kontrol ya da gösterge birleştirilemez.

VI ile çalışan terminal ve bağlantı sayılarını azaltmada kümeler kullanışlıdır. Örneğin, eğer VI birçok kontroller ve göstergeler ön paneli ile terminallerin iş birliği gerekiyorsa, bir küme altında bir terminal kullanarak gruplamak daha kolaydır.

Unbundle fonksiyonu (*Array&Cluster menüsü*) kümeyi kendi özel parçalarına böler. Bu parçalar blok diyagramda sıradan terminal gibi kullanılabilir. *Bundle* fonksiyonu (*Array&Cluster menüsü*) tüm özel parçaları bir tek kümeye toplar ya da küme içi elementleri yer değiştirir.

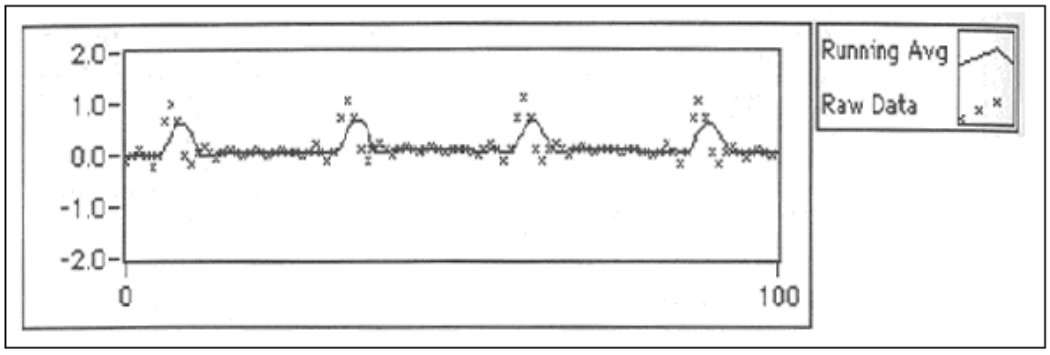
1.9. Tablo ve Grafikler

1.9.1. Genel Bakış

LabVIEW'in tabloları ve grafikleri, grafik formunda verilerin şeklini gösterir. Etkileşimli olarak tablolar eskiye yeni veri ekleyerek veri çizerler. Bundan dolayı bağlam içindeki akım değeri ile önceki veri görülebilmektedir. Grafikler bir daha geleneksel Şekilde, önce meydana getirilen değerlerin dizilerini çizer. Bu bölümde, tablolar ve grafiklerin uygun veri tiplerini ve onun kullanımı için çeşitli yolları gösterilmektedir. Bu bölümde grafik tablolarının kullanımı, bir tablonun üçmodunu tanımlayabilme, *strip, scobe* ve *sweep, boolean* anahtarlarının mekaniksel etkisini, tablo ve kümelerin fonksiyonlarının farklılıkları, tekli ve çoklu için tablo ve grafiklerin oluşturulmasında kabul edilen veri tiplei gibi konular açıklanmaya çalışılmıştır [1, 2].

1.9.2. Dalga Formu Tablosu

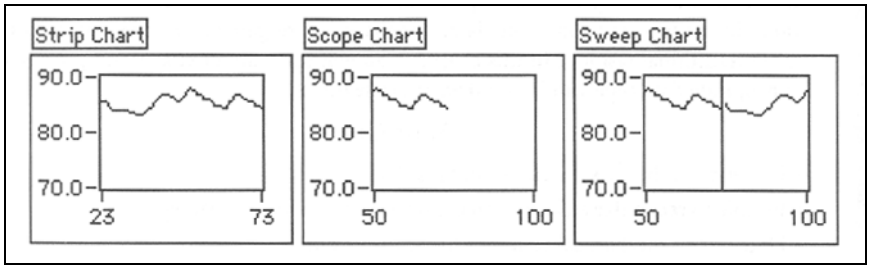
LabVIEW'in üç farklı interaktif veri göstergeli yenileme moduyla bir çeşit dalga formu vardır. Dalga formlu tablo, *Controls* menüsünde *Array&Graph* paletindedir. Bu bir ya da birden fazla aktarımları Şekil 1.42'de gösterildiği gibi sayısal göstergedir.



Şekil 1.42. Dalga Formu Tablosu

1.9.3. Tablo Yenileme Modları

Dalga formu tablosunun üç yenileme modu vardır; Şekil 1.43'de ki gibi *strip* tablo, *scope* tablo, *sweep* tablo aşağıdaki Şekilde verilmiştir. Yenileme modu Data Operations ▸ Update Mode menüsünden dalga formlu tablo seçilerek çıkartılabilir [1].

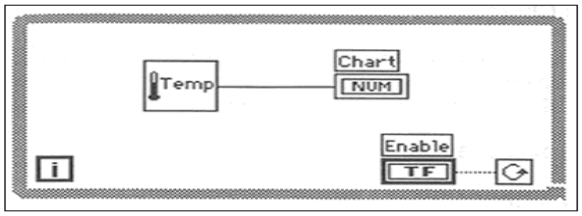


Şekil 1.43. *Strip, Scope, Sweep* tablosu

Çizgi tablosu kağıt çizgisel tabloya benzer kıvrım gösterime sahiptir. scope ve sweep tablosu osilaskoba benzer izinden takip eden gösterime sahiptir. *Scope* tablosu, grafik alanın sağ kenarına grafik ulaştığında, grafik silinir ve grafik tekrar sol kenardan başlar. *Sweep* tablo da alan tabloya benzerdir, ancak veri sağ kenara ulaştığında gösterim silinmez. Onun yerine, dik hareketli çizgi yeni veri başlangıcını belirler ve yeni veri eklendikçe gösterim boyunca devam eder. İz düşümü takibinde iz düşümünde üçü de yavaş olduğundan, scope ve sweep tablosu çizgi tablodan anlamlı ve hızlı çalışır.

1.9.4. Tekli Grafik Tablo

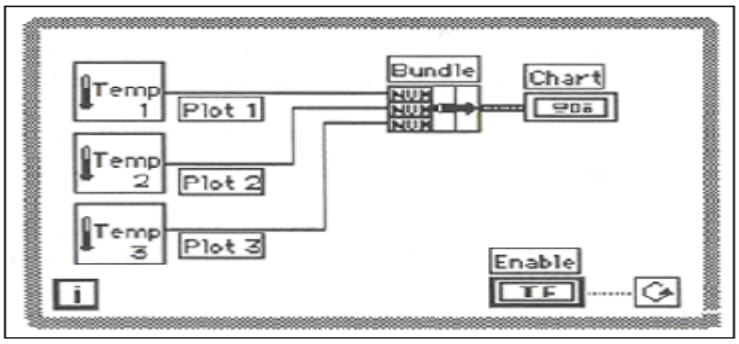
Aşağıda Şekil 1.44'de olduğu gibi, direk olarak dalga formlu tabloya skaler girdi girebilirsiniz. Şekil 1.44'de ki örnekte, her döngü tekrarlandığında yeni derece değeri tabloya, grafiğe aktarılır.



Şekil 1.44. *Tekli* grafik tablosu

1.9.5. Çoklu Grafik Tablo Çalışması

Dalga formlu tablolar aynı zamanda birden fazla grafiği birleştirebilirler. *Bundle* fonksiyonu (*Array&Cluster menüsü*) kullanarak verileri birlikte paketlemek gerekir. Aşağıdaki Şekilde, Bundle fonksiyonu üç farklı VI verilerini paketler ya da gruplar, öyle ki bu çıktılar dereceden elde edilmiştir. Onun için dalga formu tabloda gösterilebilir. Dalga formlu tablo ikonundaki değişikliğe dikkat edilmelidir. Daha çok grafik eklemek için pozisyon alet ile paketle girdi terminalinin boyutunu değiştirerek kolaylıkla bu sayı yükseltilebilir [2].



Şekil 1.45. Çoklu grafik tablo

1.10. Dijital Göstergeyi Saklama ya da Gösterme

Diğer sayısal göstergeler gibi, tablolar da dijital göstergeyi saklama ve gösterme seçeneğine sahiptir. (*pop – up menüsünde Show seçeneği*). Dijital gösterge tablonun son yeni değerini gösterir.

1.10.1. *Scrollbar*

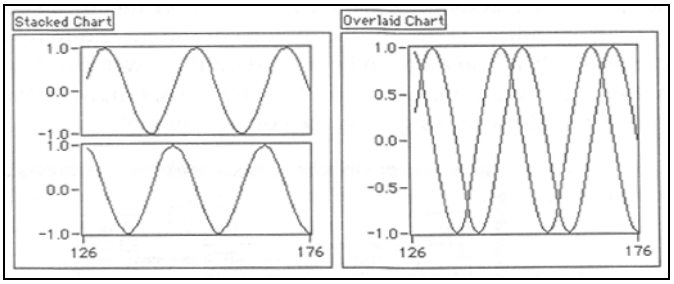
Tablolar saklanabilir ya da gösterilebilir *scrollbarlara* sahiptir. *Scrollbarları* kullanarak grafikte geçen eski veri gösterilebilir.

1.10.2. Tabloları Temizleme

Edit modundan tablo silmek için, tablo pop – up menüsünden alt menü olan *Data* Operations'dan *Clear Chart* seçilir. Eğer çalıştır modunda ise, pop – up menüsünün Clear Chart seçeneğindedir. Ya da palette *Clear* düğmesine tıklanabilir.

1.10.3. Üst Üste Olan ve Takip Eden Grafikler

Çoklu grafik tablosu varsa, üst üste gelen grafikler adı altında eşit ölçekli grafikler seçilebilir, bunun adı da istiflenmiş grafik olur. Tablonun pop – up menüsünden *Stack Plots* veya *Overlay Plots* seçilebilir. Aşağıda Şekil3.5'de grafik arasında ki farklılıklar verilmiştir [2].



Şekil 1.46. Üst üste takip eden grafik

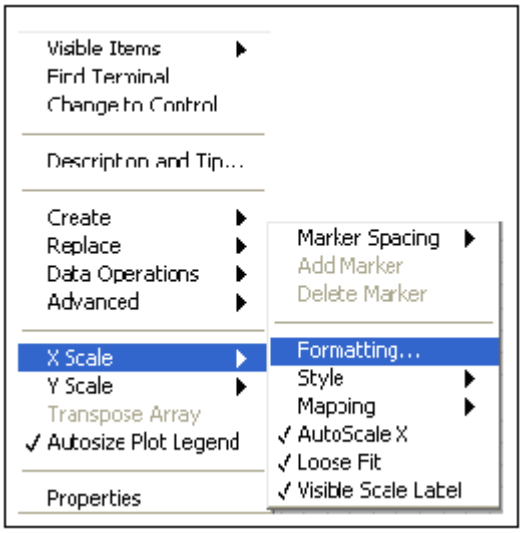
1.10.4. Tablo Tarihi Uzunluğu

Yanlışlıkla, bir tablo 1024'e kadar veri noktalarını kaydedebilir. Bu tablonun büyüklüğü değiştirilmek istenirse, pop – up menüsünden *Chart History Length* seçeneği seçilmeli ve pop–up menüsündeki yeni değeri 100000 noktaya çıkartılmalıdır.

1.10.5. Tablo ve Grafik Parçaları

Grafikleri sürekli kullanmak için grafik ve tabloların çok güçlü özellikleri vardır. Tablo ve grafikler, otomatik olarak onlardaki Şekillerin ve noktaların gösterimi için yatay ve dikey ölçeklerini ayarlar. Otomatik ölçekleme özelliği Data Operations menüsünden Auto Scale X ve *Auto Scale* Y seçenekleri ya da X Scale veya Y Scale'i pop–up

menüsünden kullanılır ya da kullanılmaz. Aynı zamanda bu özellik paketten de kontrol edilebilir. LabVIEW *Auto Scalling On* varsayımına hata verir. Fakat kullandığınız bilgisayar ve video sisteme bağlı olarak otomatik ölçeklendirme, tablo ve grafiğin yavaşlamasına neden olabilir. Eğer, otomatik ölçeklendirme kullanılmak istenmiyorsa, direkt isim ve belirtme aleti ile yatay ve dikey ölçekler değiştirilebilir. LabVIEW kontrol veya göstergede olduğu gibi. LabVIEW seçilen oranla nokta yoğunluğunu otomatik kurar. X ve Y ölçekleri her biri alt menü seçeneklerine sahiptir [4].



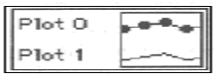
Şekil 1.47. X scale ve Y scale in kontrolü

Otomatik ölçeklendirmeyi açma ve kapama için *Auto Scale* kullanılır. Normalde, uygun seçenek işlettildiği zaman ölçekler veri oranına tam uygun hale gelir. Lose Fit seçeneği, LabVIEW'in daha güzel sayılara yuvarladığında kullanılabilir. Uygunluğu kaldır ile sayılar ölçeğin çoklu artışları için yuvarlanır. Örneğin, eğer noktalama 5'er artarsa, minimum ve maksimum değerler de 5 ile çarpım olarak ayarlanır.

1.10.6. Legend (Yazı)

Tablo ve grafik alıcı, gösterim stili oluşturulmadığı sürece her yeni grafik ya da tablo için bozuk stil kullanır. Eğer tablo ve grafiğin birleştirilmesi isteniyorsa, özel gösterimler için belli özellikler kullanılabilir. (Örneğin, 3. grafiğin mavi olması),

özellikleri *legend* kullanarak ayarlanabilir. Tablo ve grafik pop – up menüsünden *Show* alt menüsünü seçerek görülür yada görülmez yapılabilir. Örnek yazı aşağıda Şekil 1.48'de verilmiştir [2].



Şekil 1.48. Örnek yazı

Legend seçildiği zaman sadece bir gösterim çıkar. Boyut değiştir aleti ise legend köşesinden aşağıya çekerek daha çok gösterimler alınabilir. Grafik özellikleri girildikten sonra, yazı görünür iken grafik kayıt kaydedicileri saklar. Eğer tablo veya grafikte, tanımlanandan fazla *legend* yüklenirse LabVIEW yanlış stilde çizer.

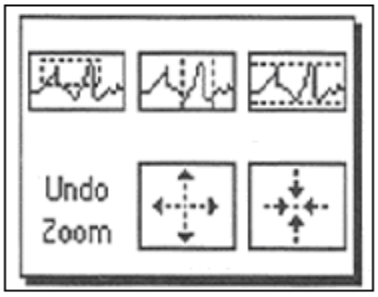
Tablo, grafik vücudu taşındığı zaman, legend onunla hareket eder. Sadece *legend*'in kendini taşıyarak yeri değiştirilebilir. *Legend*'i solundan büyütmek başlıklara fazla yer vermektir veya sağından büyütmek grafik örneklerine fazla yer vermektir. Her grafik 0'dan başlayarak sayı ile belirlenmiştir. Bu başlığı LabVIEW diğer başlıkları gibi değiştirilebilir. Her grafik kendi çizgi, renk ve nokta stilini değiştirmek için kendi pop – up menüsüne sahiptir.

1.10.7. Palet Kullanımı

Palet ile VI çalışma süresi boyunca birçok kullanışlı fonksiyonlar yerine getirilebilir. Tablo veya grafikleri silinebilir, X ve Y ekseni ölçekleri ve herhangi bir zamanda ölçeğin gösterim formatı değiştirilebilir. Tablo ve grafik pop–up menüsünden Show menüsü paleti aşağıda Şekil 1.49'da gösterilmiştir.

Şekil 1.49. Paket kullanımı

Eğer düğmesine basılırsa, LabVIEW grafiğin X verilerini ölçeklendirir. Eğer düğmesine basılırsa, LabVIEW grafiğin Y verilerini ölçeklendirir. Eğer, grafiğin her iki ekseninin ölçeklendirilmesi gerekirse, kilit düğmesine tıklayarak kilid açılabilir ve düğmeleri, X ve Y ölçeklerinin kesinlik ve format ile bağlı olarak çalışma zaman kontrolünü verir. Geriye kalan üç düğme ise, grafiğin seçenekler modu kontrolünü sağlar. Normalde, standart işlem modundadır. Anlamı, grafiğin içine tıklanarak kursörü çevrede gezdirilebilinir. Eğer düğmesine basılırsa, el kursörü ile grafiğin görünür verileri tıklanarak ve taşınarak grafiğin akışı başlatılmak için bu mod açılmış olur. Zoom düğmesine basılarak, gelen pop–up menüsünden yakınlaştırma metotlarından bir kaçını Şekil 1.50'de ki gibi sunar [2].



Şekil 1.50. Zoom yapısı

Aşağıda kiler bu yakınlaştırma seçeneklerinin açılımıdır.

Zoom by Rectangle. Yani, yakınlaştırmak istenilen alan etrafında kursör ile dikdörtgen çizilebilir. Düğme bırakıldığında, eksenler yakınlaştırma alanında yeniden ölçeklendirilir.

Zoom by Rectangle. X verisini yakınlaştırma (Y aynı kalır).

Zoom by Rectangle. Y verisini yakınlaştırma (X aynı kalır).

Undo
Zoom *Undo Last Zoom*. Önceki ölçeklere geri alır.

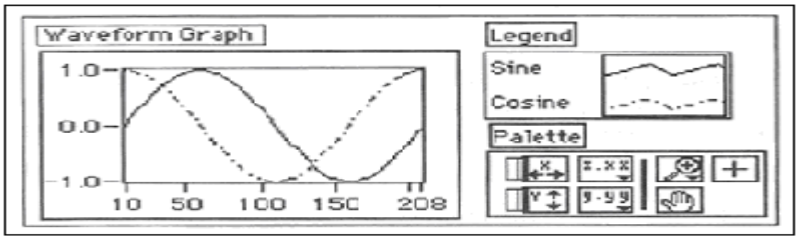
 *Zoom In About a Point.* Özel bir noktaya fare getirilirse, düğme bırakılana kadar grafik yakınlaştırılır.

 *Zoom Out About a Point.* Özel bir noktaya fare getirilirse, düğme bırakılana kadar grafik uzaklaştırılır.

Son iki mod için  ile  *shifte* tıklayarak yakınlaştırmayı ters yönde yapar.

1.10.8. Grafikler

Tablolara karşın, interaktif olarak, grafikler birden hazırlanmış dizileri gösterir. LabVIEW daha çok esneklik için iki çeşit grafik sunar; Dalga formlu ve XY grafikleri. Her ikisi de VI ön panelinde aynı görünür, fakat tamamen farklı özelliklerdedirler. Şekil 1.51'de birçok seçenekli grafik örneği gösterilmiştir.

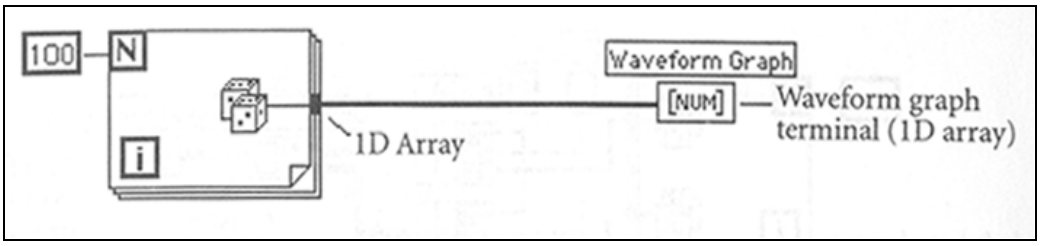


Şekil 1.51. Grafikler

Controls menüsü Array&Graph paletinden her iki tip grafik göstergeleri bulunabilir. Dalga formlu grafikler belli birimsel noktalı tek değişkenli fonksiyonları aktarırlar, zaman değerlendirme dalga formlu gibi. Bu tip tek tek dağıtılmış noktalarda dizi verilerinin aktarımı için idealdir. XY grafikleri genel anlamlıdır, kartezyen grafik zaman tabanı değerlendirme ya da dairesel Şekil gibi çok değişkenli fonksiyonlar aktarımı için idealdir. İki çeşit farklı girişe sahiptir, bu nedenle karıştırmamak için dikkatli olmalıdır [1,2].

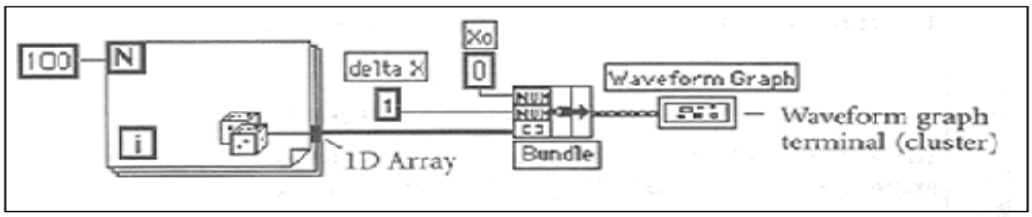
1.11. Dalga Formu Grafiklerinin Tekli Gösterimi

Aşağıda Şekil 1.52'de gösterildiği gibi *For* döngüsü, sınırları içinde kodu uygular, alt diyagramı çağırır, sayım terminalinde içerilen değeri, söylenene eşitleyen sayıcı zamanlayıcısının bir toplamıdır.



Şekil 1.52. Dalga formu grafiklerinin tekli gösterimi

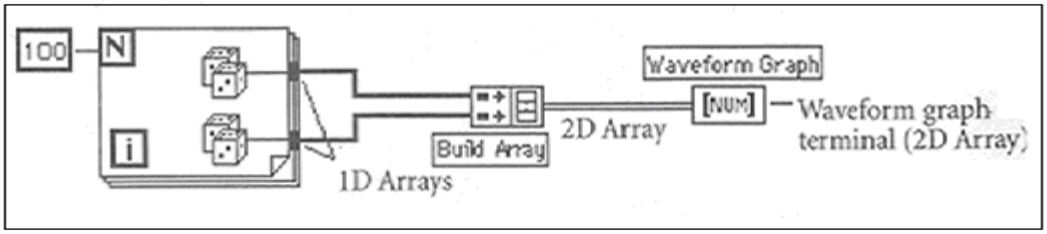
Grafik zaman tabanı esnekliği değiştirmek isteniyorsa (Örneğin, X0= 0 örneklemeye başlandı veya örnekleme uzaysal olup delta X= 1 olursa); başlangıç x değerini içeren küme, delta x değeri, dizi verileri dalga formlu grafiğe girilebilir. Grafik terminalinin küme göstergesi gibi ortaya çıkmasına dikkat edilir [2].



Şekil 1.53. Dalga formu grafik terminalleri

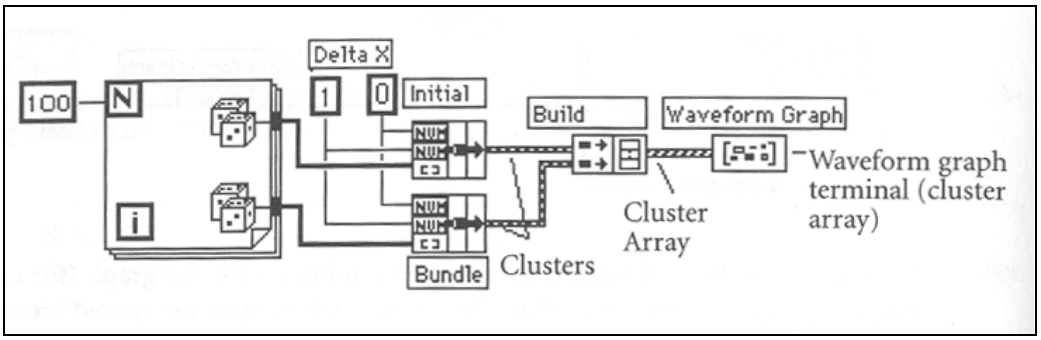
1.11.1. Dalga Formlu Grafiklerin Çoklu Gösterimi

Tek grafikli örneklerde olduğu gibi, birden çok dalga formlu grafikleri veri tipleri dizisi oluşturarak gösterilebilir. Aşağıda ki iki Şekilde çoklu grafik giriş metodu, gösterilmiştir. Önceki örnekler gibi, grafik terminali girilen veri tipini almıştır.



Şekil 1.54. Veri tipleri

Yukarıda ki Şekil 1.54'de x'in başlangıç değeri 0 ve delta x değeri 1 olarak her iki grafik için verilmiştir. İki boyutlu dizinin 100 sütunlu 2 satırlı, 2x100 olduğu unutulmamalıdır. Grafik her zaman iki boyutlu dizinin satırlarını gösterir. Eğer veriler sütunla organize edilmiş ise, grafiğe aktarılan dizinin değiştirilmiş hali olduğundan emin olunmalıdır. Aşağıda ki Şekil 1.55'de, her dizi için başlangıç x ve delta x değerleri özel verilmiştir. X parametrelerinin iki veri tabanlarında eşit olmasına gerek yoktur [4].



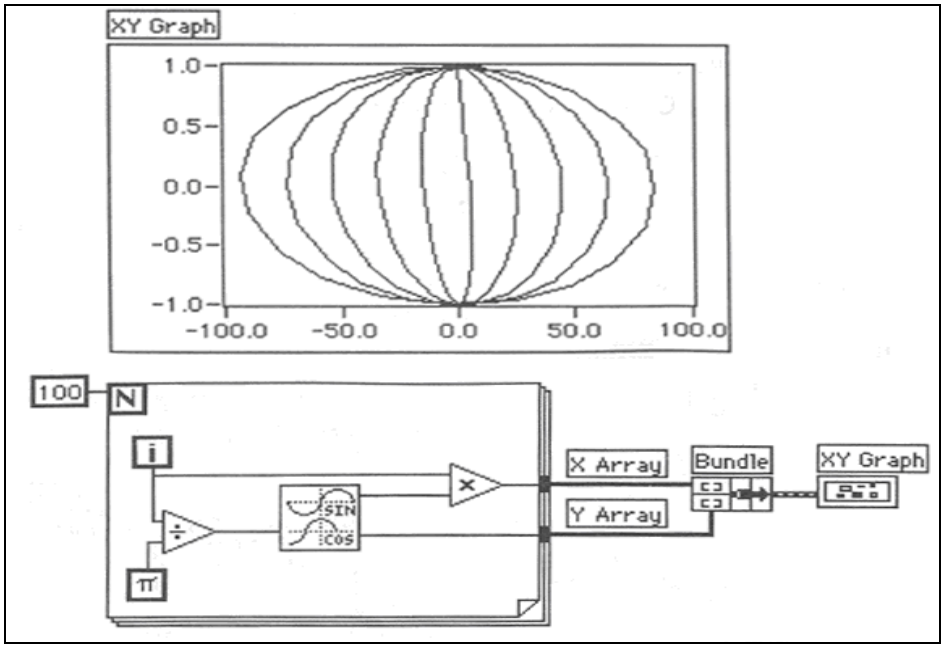
Şekil 1.55. Başlangıç değerleri

 *Build Array* fonksiyonu (*Array&Cluster menüsü*) 1D sıralanış girdisinden 2D sıralanışı veya küme girdilerinden küme dizisi meydana getirir.

1.11.2. Tek ve Çok Yönlü Çizim – XY Grafikleri

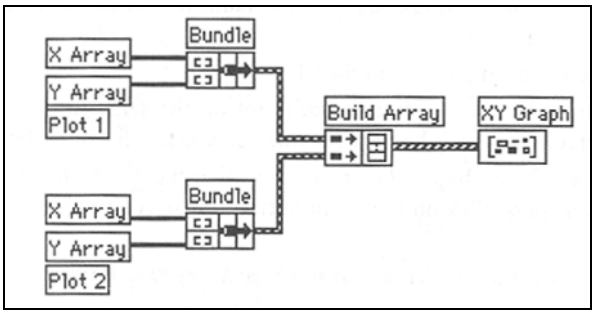
Şu ana kadar kullanılan dalga formu grafikleri, örnek olarak denenmiş dalga formlarını çizmek için bile idealdir. Ancak, örnek olarak düzensiz aralıklarla deneniyorsa veya karmaşık matematik fonksiyonu çizilecekse, koordinatları (XY) kullanarak değerleri belirlemek gerekecektir. XY grafikleri bu değişik veri tiplerini çizer. Dalga formu

grafiklerinden farklı bir veri girdisi gerektirirler. Tek çizim XY grafiği ve onun tekabül eden blok şeması aşağıda Şekil 1.56'da gösterilmiştir.



Şekil 1.56. Tek ve çok yönlü çizim

Bundle fonksiyonu (*Array&Cluster menüsü*) x ve y düzeneğini XY grafiğine bağlı kümeye bağlar. XY grafiği sarılmış bir x düzeneğini (Üst girdi) ve bir y düzeneğinin (Alt girdi) bekler. XY grafiği terminali şimdi küme göstergesi olarak görünür. Çok yönlü çizim XY grafiği için, örnek olarak aşağıda gösterildiği gibi x ve y değerleri kümesinin düzeneği kurulabilir.



Şekil 1.57. XY grafiği için küme değerleri

1.12. Stringler ve I/O Dosyası

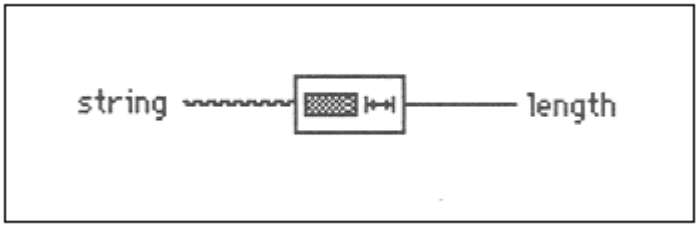
1.12.1. Genel Bakış

Bu bölüm, string ile yapılabilen güçlü nesnelerin bazılarını tanıtmaktadır. LabVIEW string veri fonksiyonlarının etkilenmesine izin veren, onun dizi fonksiyonlarına benzer birçok gömülü *string* fonksiyonlarına sahiptir. Bir disk dosyasından verinin nasıl bulunup getirileceğini ve verinin nasıl kaydedileceğini göstermektedir. LabVIEW 'in string fonksiyonlarının nasıl kulanıldığını, sayının string veya stringin sayıya nasıl çevrildiğini, bir disk dosyasının veri kaydetmesi ve daha sonra kaydedilen verinin LabVIEW'in içinden okunması için I/O dosya VI larının kullanımı bu bölümde açıklanmaktadır [2].

1.12.2. String Fonksiyonlarını Kullanma

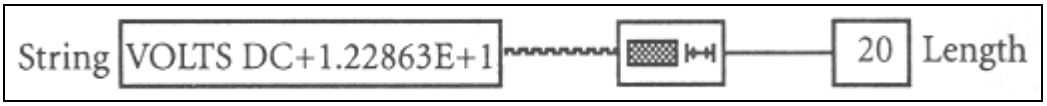
En basit metin mesajları için stringler kullanılır. Örneğin, enstrüman kontrollerinde, karakter *stringler*i gibi sayısal veri geçişlerinde kullanılır. Daha sonra, veri işlemek için bu stringleri sayıya çevirir. İstenilen stringleri de I/O dosya ve VI'larında diske sayısal veri olarak kaydedilir. LabVIEW ilk olarak onları bir dosyaya kaydetmeden önce sayısal verileri string veriye çevirir.

Aşağıda ki fonksiyonlar, *Functions* menüsünün *String* paletinde mevcuttur, ihtiyaç duyulan bazı string aktivitelerini kolaylaştırır.



Şekil 1.58. String uzunluğu

String Length, verilen bir stringte karakterin sayılarına dönüştürülür.

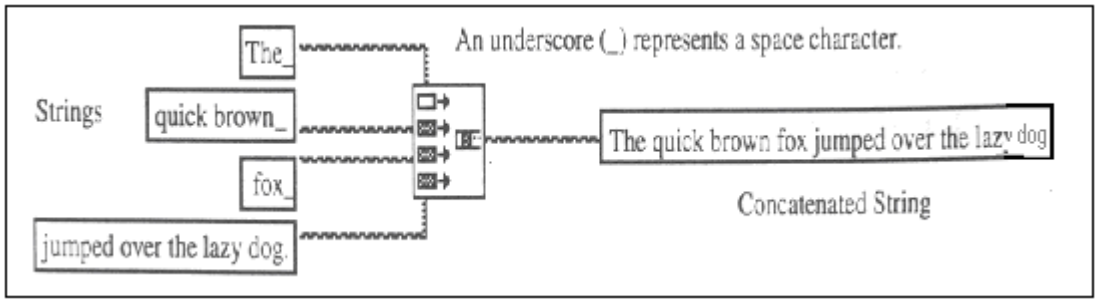


Şekil 1.59. String uzunluğunun sayıya dönüşümü

Concatenate Strings, bir tek çıkış stringi içinde bütün giriş stringlerini birleştirir.

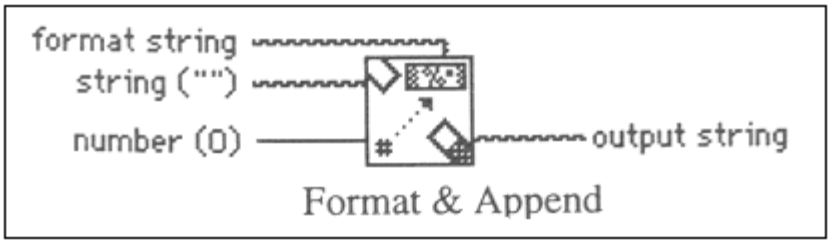
Şekil 1.60. Stringlerin sıralı bağlanması

Fonksiyon diyagram penceresine yerleştirildiği zaman  gibi görünür. Giriş sayılarını artırmak için pozisyon aracı ile fonksiyon tekrar genişletilebilir.



Şekil 1.61. String bağlama örneği

Birçok örnekte stringler sayıya veya sayılar stringlere çevirilmelidir. *Format&Append* ve *Format&Strip* fonksiyonları bu yeteneklere sahiptir. *Format&Strip* daha ele alınacaktır.



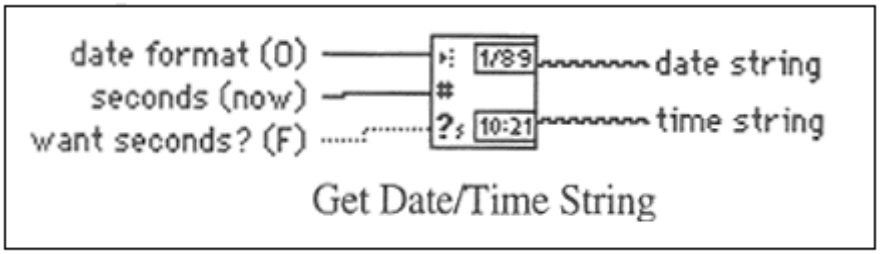
Şekil 1.62. String biçimlendirme

Format&Append, bir string gibi giriş sayısını biçimlendirir, Format String de kurguya göre biçimlendirir. Bu kurgular *Functions Reference* bölümünde listelenmiştir. Eğer orada bir tane ise, o çevrilmiş stringi, stringe bağlanan girişe ekler. Sonraki tablo *Format&Append*'in özelliği hakkında bazı örnekler verir. Bu örneklerde, altı çizilmiş karakter sınırlanmış boş karakteri temsil eder. %f kesri formatla birlikte sabit olmayan noktaların sayısı olarak girdi numarasını, %d ondalık sayı olarak ve %e bilimsel rakamlar sistemi ile birlikte sabit olmayan noktaların sayısı olarak biçimlendirilir [2].

string	format string	number	output string
(empty)	score=%2d%%	87	score=87%
score= _	%2d%%	87	score=87%
(empty)	level=%7.2e V	0.03642	level=3.64E–2 V
(empty)	%5.3f	5.67 N	5.670 N

Şekil 1.63. String biçimlendirme tablosu

Get Date / Time String, şimdiki tarihi içeren Date String'i ve şimdiki zamanı içeren *Time String*'i verir. Bu fonksiyon verinin tarihini etiketlemek için yararlıdır. Varsayılandan başka bir değeri kullanmak gerekmedikçe, girdilerin hiç birini Get Date / Time String bağlamaya gerek olmadığına dikkat edilmelidir.



Şekil 1.64. Gate Date/Time string işlevi

1.12.3. I/O Dosyası

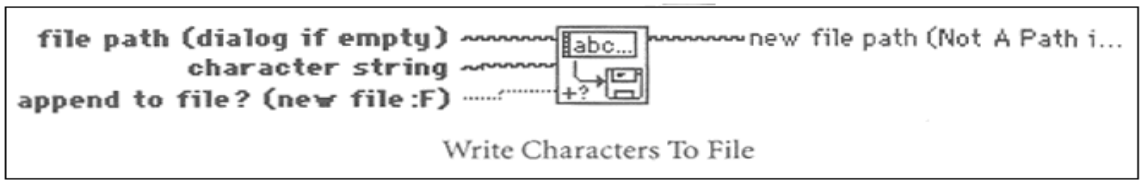
Dosya girdi ve çıktı işlemleri bilgiyi bir disk dosyasından depolar ve düzeltir. LabVIEW, I/O dosyasını her yönden koruyacak basit işlevler sağlar. Bu işlevler Functions menüsündeki *File&Error* paletinde bulunur.

1.12.4. I/O Çalışma Şekli

Dosya fonksiyonları dosya giriş bilgisi gerektirir. Eğer bir dosya yolu girilmezse, fonksiyon bir dosya adı seçilmesini veya girilmesini isteyen bir diyalog kutusu ortaya sunacaktır. Çağrıldığında dosya fonksiyonları bir dosya açar veya üretir, veriyi okur veya yazar ve sonra da dosyayı kapatır. Üretilmiş dosyalar yalnızca sıradan text dosyalarıdır. Dosyaya veri yazıldığ an veriyi görebilmek için herhangi bir kelime işlem programı kullanarak dosya açılabilir [2].

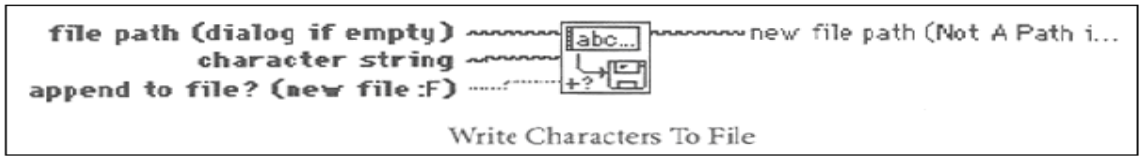
Veriyi dosyadan saklamak için, en yaygın uygulamalardan biri de asıl dosyayı düzenlemektir. Böylece veri Spreadsheets programında açılabilir. Çoğu Spread programında şeritler sütunları, karakterler de dizinleri ayırır. *Write To Spreadsheet File* ve *Read From Spreadsheet File*, spreadsheet içeriğindeki dosyalarla ilgilidir.

Write Characters From File, yeni dosyaya karakter string belirler ya da önceden var olan bir dosyaya string ilave eder.



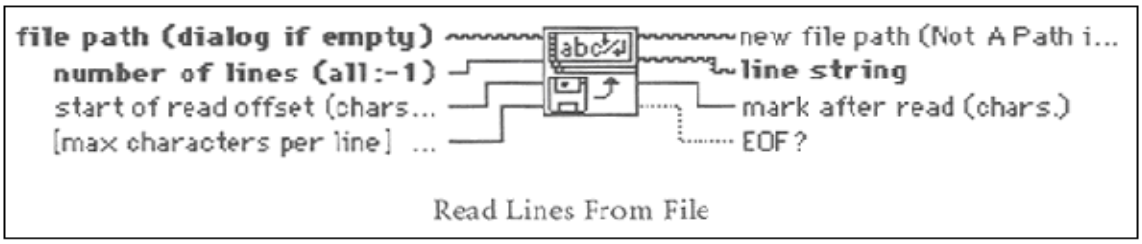
Şekil 1.65. Dosyaya karakter yazma

Read Characters From File, belirli bir karakter dizimiyle başlayan dosyadan belirli sayıdaki karakterleri okur.



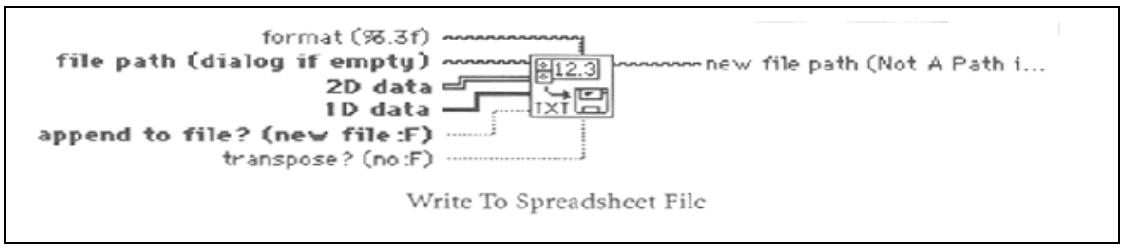
Şekil 1.66. Dosyadan karakter okuma

Read Lines From File, belirli bir karakter dizimiyle başlayan dosyadan belirli sayıdaki hatları okur.



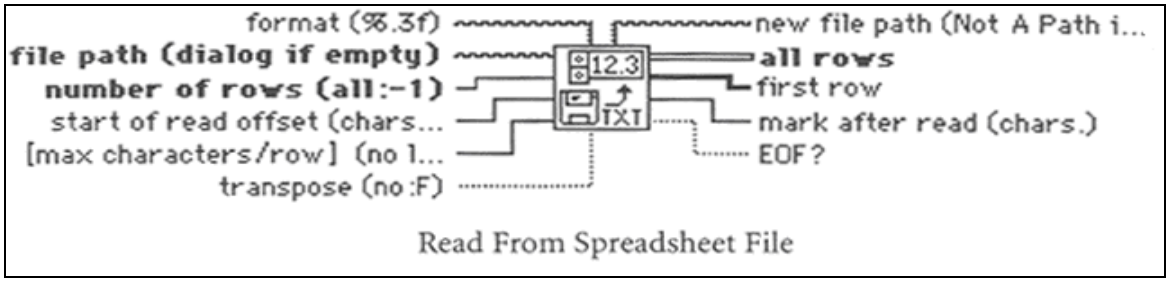
Şekil 1.67. Dosyadan belirli sayıda hat okuma

Write To Spreadsheet File, belirli tekil numaralardaki 1D ve 2D dizileri asıl diziyle değiştirir ve yeni bir string yazar ya da önceden varolan dosyanın stringini değiştirir. Tercihen veri aktarılabilir. 1D ve 2D veri için girişleri bağlamayacak (ya da bir tanesi reddedecek). VI tarafından oluşturulan asıl dosyalar birçok spreadsheet uygulamaları tarafından okunabilir.



Şekil 1.68. Tabloya yazma

Read To Spreadsheet File, belirlenmiş karakter ofsetinden başlayarak sayısal metin dosyasından sayıları ve hatları okur ve veriyi bir 2D tek denklem numaralarına çevirir. Düzenek isteğe bağlı olarak çevirilebilir. Metin formatında tutulan spreadsheet program dosyalarını bu VI okuyacaktır.



Şekil 1.69. Tablodan okuma

Bütün dosya fonksiyonları fonksiyon referans bölümünde daha detaylı olarak tanımlanmıştır.

1.13. Yapılan Literatür Çalışması

Jen-Heo Teng ve diğerleri, labview tabanlı güç analizörü tasarladılar. Harmonik analizör ve ani güç ölçümünü gerçekleştirerek LabVIEW tabanlı uygulamalarını gerçeklediler [6].

Kelvin R. Aaron ve diğerleri, National Instrument (NI)'a ait Data Acquisition (DAQ) kartını kullanarak labview tabanlı DC motorun kapalı çevrim kontrolünü yaptılar. DC motoru saat yönünde ve saat yönünün tersi yönünde kontrol etmeyi başarmışlardır.

Bunun için labview ortamında vi oluşturmuşlar, ön panelden motor için uygun pozisyon girilmekte oluşturulan özel vi motoru kontrol etmektedir [7].

Dr Jovitha Jerome ve diğerleri, DC motor da hız regülasyonu için akıllı br kontrolör tasarlayarak LabVIEW ortamında gerçeklediler. Sistemlerinde yapay sinir ağları, fuzzy lojik kontrolör ve Neuro Fuzzy kontrolör kullandılar. Doğrusal olmayan dinamikleri alarak yapay sinir ağları ve fuzzy lojik kontrolör uyguladılar. Sonuçları karşılaştırdıklarında Neuro-Fuzzy kontrolörün daha avantajlı olduğunu gözlemlediler [8].

Saliman A. Isa ve diğerleri, radio-teleskop anteninin astronomcular tarafından internet üzerinden kontrol edilebileceğini yaptıkları çalışmada göstermişlerdir. Antenin pozisyonunu doğru bir Şekilde kontrol etmek için iki adet DC motor kullanmışlardır. National Instrument (NI) donanımı olarak Signal Accessory kartını ve DAQ kartını sürücü devreyi kontrol etmek için kullandılar. Labview programını kullanarak sistemlerini tasarladılar. İstenilen giriş parametrelerine göre sistemi kontrol etmeyi başardılar. Bunun için Internet Developer Toolkit fonksiyonunu kullanarak internet üzerinden uzaktan erişim ve kontrolü sağladılar [9].

P. Thpsatorn ve diğerleri, serbest uyartımlı DC motoru fuzzy lojik kontrol (FLC) kullanarak, LabVIEW tabanlı hız kontrolünü yaptılar. Sistem için gerekli olan uygun gerilimi sağlamak için (FLC) tasarladılar. Fuzzy kurallarına göre motorun hız hatası (e) ve değişimin hız hatası(ce) olarak aldılar. Sonuç olarak, Fuzzy Lojik Kontrolü PI ve PID kontrole göre kıyaslamasını yaptılar ve (FLC) nin daha avantajlı olduğunu gösterdiler [10].

He Wenhai, pompalar üzerine yaptığı çalışmada pompalardaki sıvı akış hızını ele aldı ve ölçme sistemi tasarladı. Bu ölçme işleminde labview programını kullandı. Sistem için ultrasonik sensörler kullandı. Elde ettiği işaretleri ayırmak için korelasyon fonfsiyonlarının matematiksel modelini çıkarmıştır. Bu model için DAQ kartı kullanmıştır [11].

Qui Tang ve diğerleri, güç sistemlerinde harmonik ölçümü üzerine çalışma yaptılar. Bu çalışmada interplotting windowed FFT yöntemini önerdiler ve analizini yaptılar. Analitik sonuçlara göre labview ortamında gerilim ve akım değerleri için harmonik ölçüm sistemi tasarladılar. Yapılan ölçümleri uzaktan sisteme taşımak için AppletVIEW Toolkit fonksiyonunu kullandılar. Ölçüm sonuçları sistemin geçerliliği ve uygulanabilirliliğini gösterdi [12].

2. YAPILAN ÇALIŞMALAR

2.1. Giriş

Bu tezde yapılan çalışma, özgün bir çalışmadır. LabVIEW tabanlı damar içi uygulanan besin ve ilaç hazırlama sistemi tasarımı ve gerçek zamanlı kontrolü gerçekleştirilmiştir. Tıp alanında son zamanlarda teknoloji kullanımı kat ve kat artmaktadır. Mikroişlemci teknolojisinin gelişimi ile birlikte bilim adamları bu teknolojiyi tıp alanında kullanmaya başlamışlardır, bunların en büyük örnekleri ülkemizde medikal alanda kullanılmaktadır.

TPN ve onkolojik ilaçların hazırlanması damar içi uygulanan besin ve ilaçlara en iyi örnektir. Ülkemizde bu işlem pek çok yerde manuel olarak yapılmakta ve bundan kaynaklanan ciddi sorunlar yaşanmaktadır. Bunlardan bazıları hastanelerin yenidoğan ve onkoloji ünitelerinde ölümlere kadar yol açmaktadır.

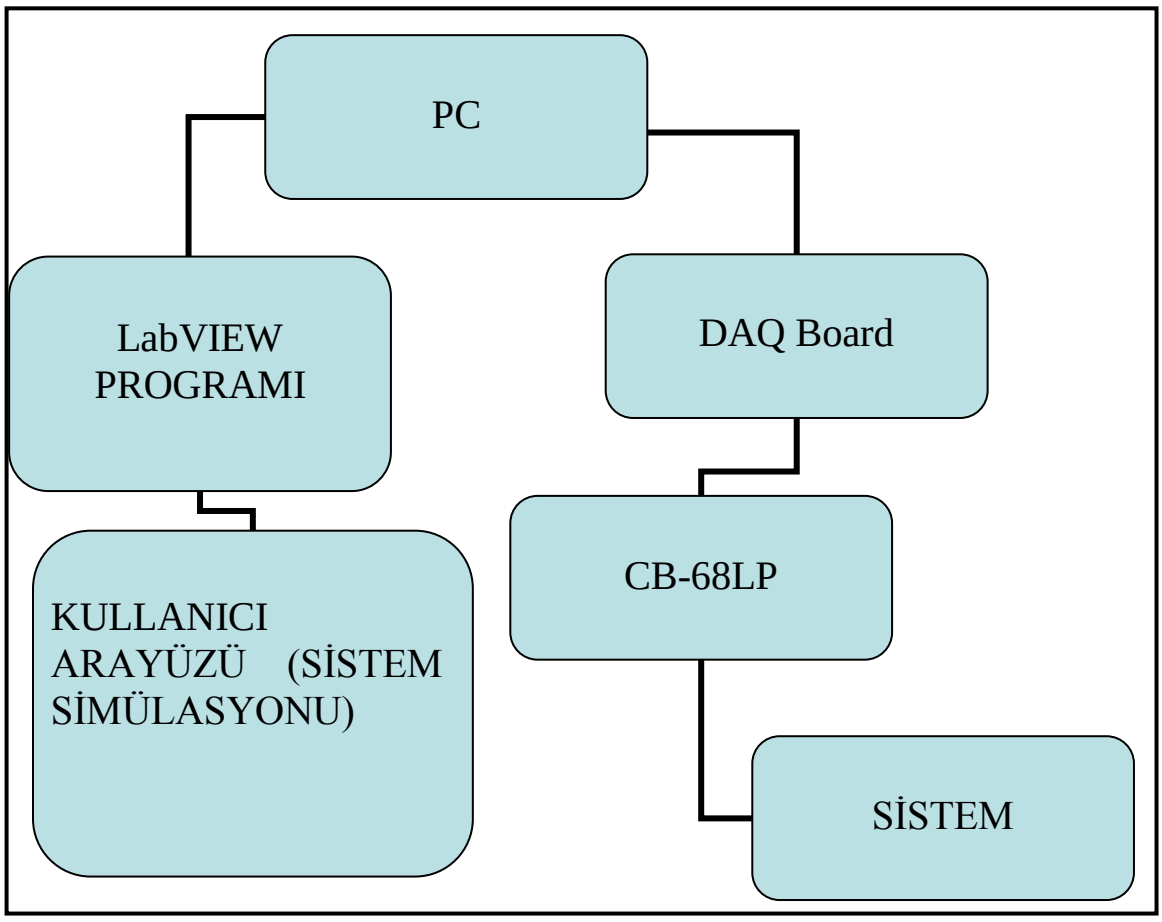
Hastanelerimizde yenidoğan ve onkolojik ilaçların hazırlanmasında kısmende olsa otomasyona geçilmiş olamakla birlikte, bu sistem ve cihazların üretimi dünya çapında iki firmaya bağımlı olduğundan, ülkemizin bu sistemleri tamamen ithal etme durumunda kaldığını, yaptığım çalışmalar esnasında gözlemlemiş bulunmaktayım. Onkolojik ilaçlar pahalıdır ve manuel karışımlar hazırlanırken israflarla karşılaşılmaktadır. Aynı zamanda klinisyen ya da görevliler için sağlık açısından tehlike arz etmektedir. Dolayısı ile karışımlar el değmeden sistem tarafından hazırlanmalıdır.

Ayrıca TPN sisteminde otomasyonun uygulanması, yenidoğan yoğunbakımlarında enfeksiyona bağlı ölümleri ve hastane enfeksiyonlarını da önemli derecede azaltmaktadır. Yenidoğan ve diğer yoğun bakım ünitelerinindeki hastaların beslenmelerinin sadece hemşirelerin dikkatine bağımlı kalmadan, sistem tarafından da kontrol edilebilmesi son derece iyi bir gelişme olarak görülmektedir.

Üzerinde çalıştığım sistem ise; LabVIEW ortamında yukarıda bahsettiğim sistemi gerçeklemeye yöneliktir. LabVIEW programı, grafiksel programlamaya yönelik olmasından dolayı bu tür sistemleri tasarlamaya son derece uygun bir programdır. Yaptığım çalışmanın simülasyonu, tasarladığım sürücü devre ile gerçek zamanlı haberleşerek sistemin çalışması, hem simülasyon ortamında hem de analog ortamda gözlenebilmektedir.

2.2. Sistemin Akış Diyagramı

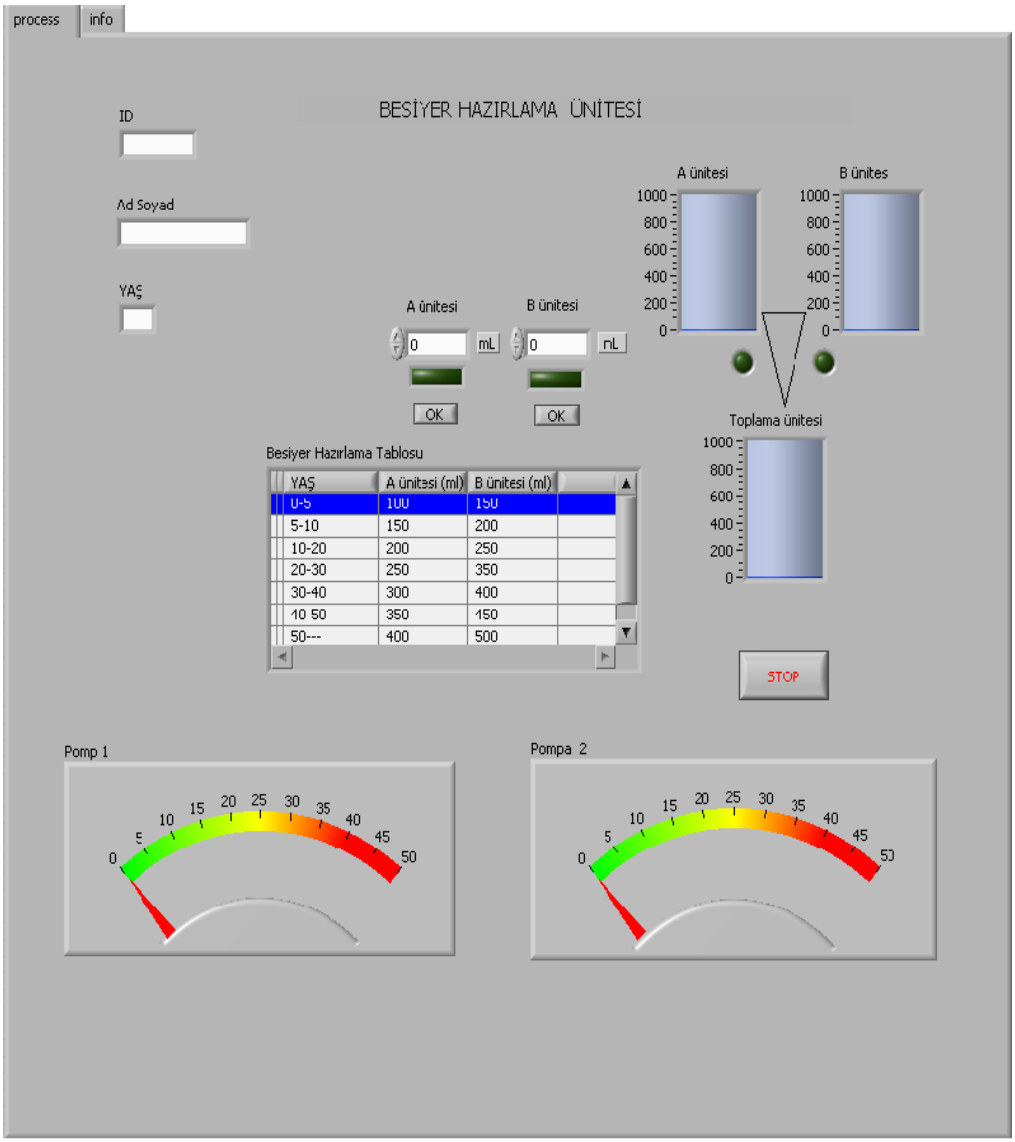
Aşağıdaki Şekil 2.1'de sisteme ait akış diyagramı gösterilmektedir. LabVIEW tabanlı gerçekleştirilen sistemin işleyişi bundan sonraki bölümlerde akış diyagramına bağlı olarak ele alınmıştır. Oluşturulan simülasyon, DAQ kartı üzerinden sistem ile haberleşmekte ve sistem gerçek zamanlı çalışmaktadır.



Şekil 2.1. Sisteme ait akış diyagramı

2.3. Sistem İçin Hazırlanan LabVIEW Programı

2.3.1. *Front* Diyagramı



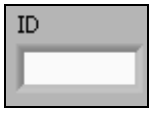
Şekil 2.2. *Front* diyagramı

Tasarladığım ön arayüz diyagramı iki kısımdan oluşmaktadır. *Process* ve *info* arayüzleri. Process kısmında hasta ID, Ad Soyad ve Yaş girişleri, A ünitesi, B ünitesi ve Toplama Ünitesi, Yaşa göre A ünitesi ve B ünitesi tablosu ve pompa 1 ve pompa 2 DC gerilim seviye göstergeleri bulunmaktadır. Sistemin kontrolü ön arayüz diyagramdaki

process arayüzünden yapılmaktadır. Hastalar için oluşturduğum database kısmı ise info arayüzündeki tabloya kaydedilmektedir. A ünitesi ve B ünitesinden gerekli olan sıvı miktarları arayüzden girildiğinde görsel olarak A ünitesi ve B ünitesinde kalan sıvı miktarları gözlenebilmektedir. Kanal sayımız bu sistem için iki adettir. Dolayısıyla sistemde iki adet özel DC pompamız mevcuttur. Sistem process arayüzü ile gerçek zamanlı çalışmaktadır.

2.3.1.1. Ön Arayüz Diyagramının Yapısı

Hasta ID girişi bu noktadan yapılmaktadır. Labview programında *numerik* kontrol olarak kullanılmaktadır. Data tipi DBL formatındadır.

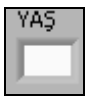


Şekil 2.3. ID girişi

Hasta Ad Soyad girişi bu noktadan yapılmaktadır. Labview programında String kontrol olarak kullanılmaktadır.

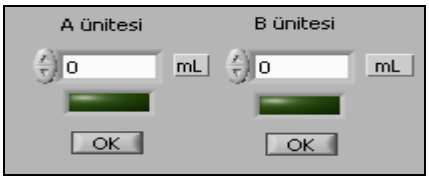
Şekil 2.4. Ad Soyad girişi

Hastanın yaş bilgisi bu noktadan yapılmaktadır. Labview programında *numerik* kontrol olarak kullanılmaktadır. Data tipi DBL formatındadır.



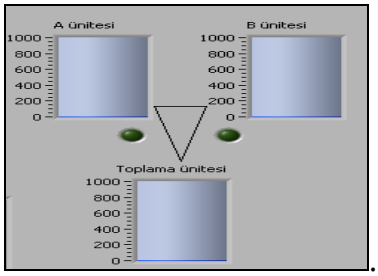
Şekil 2.5. Yaş girişi

Hasta için gerekli olan parametreler ml cinsinden bu girişlerden yapılmaktadır.



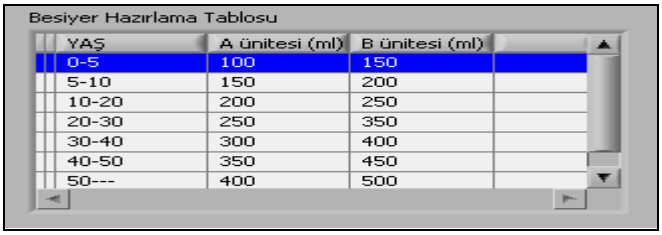
Şekil 2.6. Parametrelerin ml cinsinden girişleri

A ünitesi, B ünitesi ve toplama ünitesinin gerçek zamanlı takip arayüzü sekil 2.7'de ki gibidir.



Şekil 2.7. A, B ve toplama üniteleri

Yaş bilgisine göre hangi üniteden kaç ml alınacağı bilgisini veren örnek olarak oluşturduğum besiyer hazırlama tablosu.



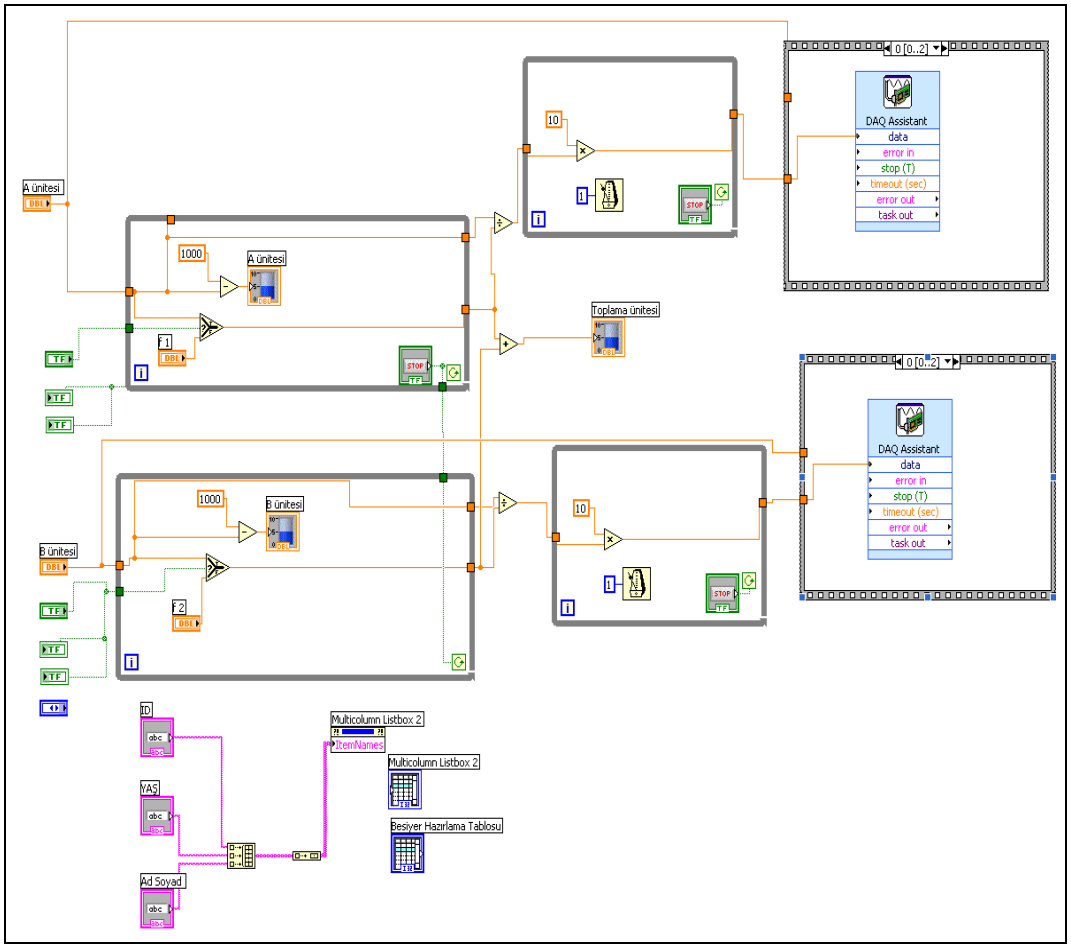
YAŞ	A ünitesi (ml)	B ünitesi (ml)
0-5	100	150
5-10	150	200
10-20	200	250
20-30	250	350
30-40	300	400
40-50	350	450
50---	400	500

Şekil 2.8. Besiyer hazırlama tablosu

Pompalar üzerinde ki gerilimin takibi Şekil 2.9'daki gösterge üzerinden takip edilmektedir.

Şekil 2.9. Gerilim takip göstergesi

2.3.2 Blok Diyagramı



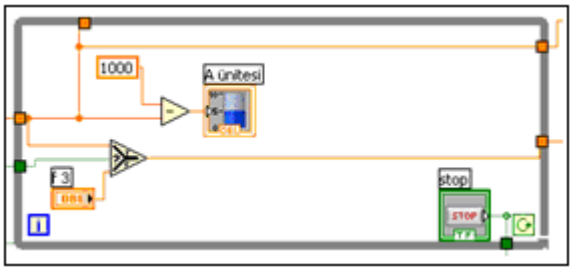
Şekil 2.10. Blok diyagramı

Blok diyagramı, ön arayüz diyagramıyla eş zamanlı çalışmaktadır, sistem tasarımı blok diyagramında hazırlanmakta ve sistemin çalışmasına ve dataların akışının gözlenmesine imkân sağlamaktadır, bu da sistemin her ayrıntısını gözle takip etme imkânı sunmaktadır.

Blok diyagramı kendi içerisinde while loop ve sequence structure yapısı içermektedir, bu yapılar özelliklerine göre sistemdeki veri akışını sağlamaktadırlar. Sistemde görülen tanklar içindeki sıvı miktarları tam dolu iken 1000 ml olarak alınmıştır. Sistemimizde iki adet, A ve B ünitesi bulunmaktadır. Bu iki ünite içinde aynı birbirine parelel döngüler kullanılmıştır. Sistem eş zamanlı aynı işi yapan iki adet üniteyi kontrol

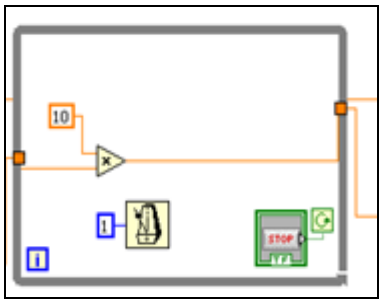
etmektedir. Anlatım kolaylığı olması açısından Aünitesini referans alıp bu ünite ile ilgili veri akışını anlatmanın yeterli olacağını düşünmekteyim.

Aşağıdaki while döngüsü, A ünitesinin sıvı seviyesindeki değişimi göstermektedir. Ön panelden A ünitesi için giriş yapıldığında veri bu döngüye girerek A ünitesindeki kalan sıvı seviyesini Şekil 2.10'da ki gibi belirlemektedir. Ön panelden girilen data bu döngüde sistemin çalıştığını gösteren veri ile karşılaştırılarak, girilen miktarın 1000 ml den düşülmesiyle tank seviyesi bilgisi ön panele gönderilmektedir.



Şekil 2.11. A ünitesi kalan sıvı miktarı

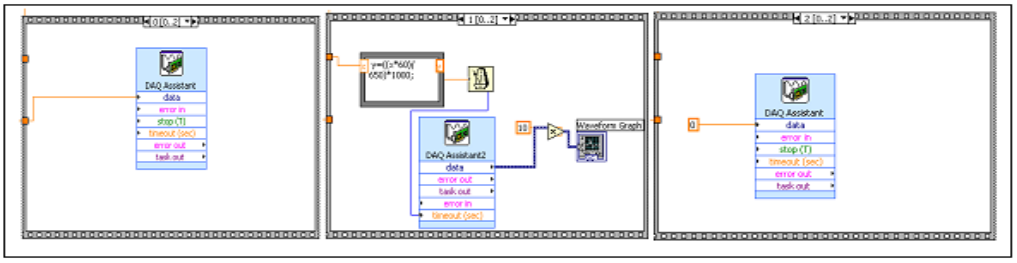
Bu while döngüsünden sonra veri ikinci while döngüsüne geçiş yapmaktadır. Burda ise veri DAQ Assistant için gerekli olan 10 değerine ayarlanmaktadır. Bir önceki döngüden gelen 1 bilgisi bu döngüde 10 seviyesine çıkartılarak sequence structure döngüsüne iletilmektedir. Şekil 2.12'de gösterilmektedir.



Şekil 2.12. 1 verisini 10 ile çarpma

Sequence structure döngüsü, DAQ Assistant NI-DAQmx ten aldığı analog çıkışlar sayesiyle gelen verinin ve bu döngüdeki zamanlamaya göre bize 10V seviyesinde bize çıkış vermektedir. Bu analog çıkış sayesiyle ünitelerden sıvı transferi yapmak için

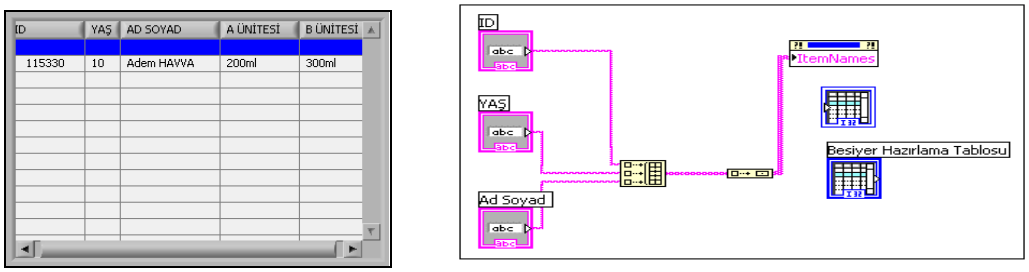
kullandığımız 24V DC pompalarımızı sürücü devre üzerinden kolayca sürebilmekteyiz, bu sayde ön panelden girerek ünitelerden istediğimiz miktarda sıvıyı toplama ünitesine tranfer edebilmekteyiz. Aşağıda Sequence structure döngüsünün işleyişi üç aşamada gösterilmiştir. Şekil 2.13'de Sequence structure döngüsü kedi içinde 0, 1, 2 basamaklarından oluşmakta ve bu basamaklar Şekil 2.13'de gösterilmiştir.



Şekil 2.13. Sequence structure döngüsü

Birinci adımda DAQ Assistant veri girişi 10 seviyesine çekilmekte ve anolog çıkış 10V çıkış üretmekte ve sıvı transfer pompası çalışmaya başlamaktadır. Kullanılan sıvı tranfer pompası dakikada maksimum 650ml sıvı tranferi yapmaktadır. İkinci adımda motorun çalışması için tasarladığım süre denklemine göre ön panelden akmasını istediğim sıvının ml olarak akmasını sağlayacak süre belirlenmiş olmaktadır. Üçüncü adım ise DAQ Assistant veri girişini 0 seviyesine çekmekte ve anolog çıkışa 0V vermektedir. Dolayısıyla sıvı transfer pompası durmaktadır.

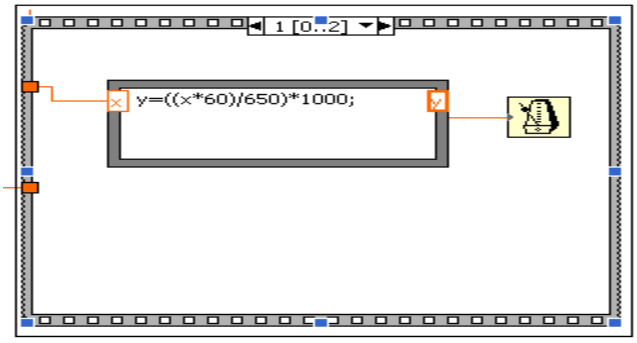
Ön panelin info arayüzünde bulıunan listede hasta ID, YAŞ, AD, SOYAD, A ÜNİTESİ, B ÜNİTESİ sütünları yer almaktadır. Bu listenin blok diyagramında karşılığı Şekilde gösterilmiştir. Şekil 2.14'de, ön panelden girilen değerler sırasıyla listeye kaydedilmektedir. Dolayısıyla yapılan çalışmalarla ilgili database kısmı info arayüzüne kaydedilmektedir.



Şekil 2.14. Info ara yüzüne veri kaydetme

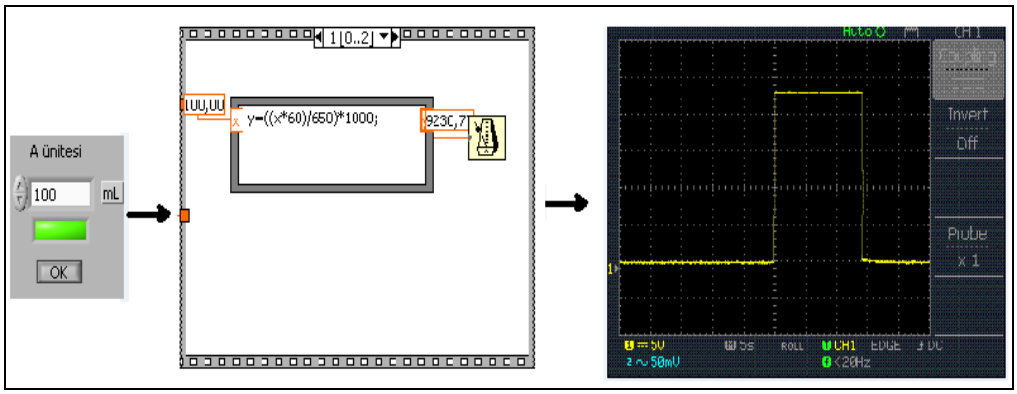
2.4. Yapılan Ölçümler

Sistem için kullandığım pompalar dakikada 650 ml sıvı transferi yapmaktadır. Bundan dolayı sistem için tasarladığım mantıksal denetleyici Şekil 2.15'de ki gibi aşağıda gösterilmiştir.



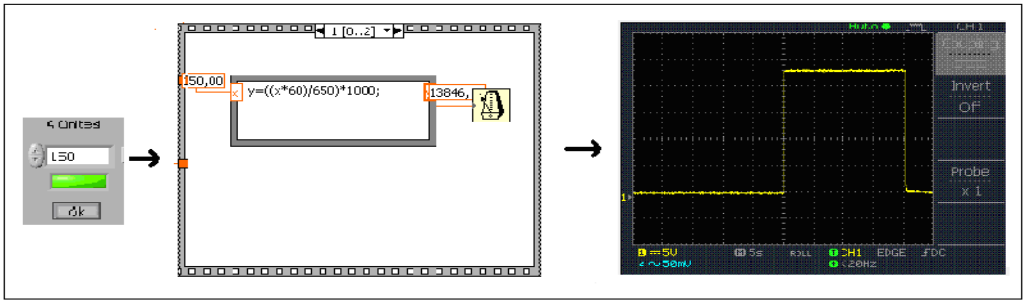
Şekil 2.15. Mantıksal denetleyici

Bu Şekilden de görüleceği gibi yukarıdaki fonksiyona göre pompaları sistemi 24V besleyerek çalışmasını sağlamaktadır. Aşağıda sistemin çalışmasıyla ilgili ölçümler ve pompaların çalışması için gerekli olan 24V DC gerilim ve çalışma süreleri grafiksel olarak verilmiştir.



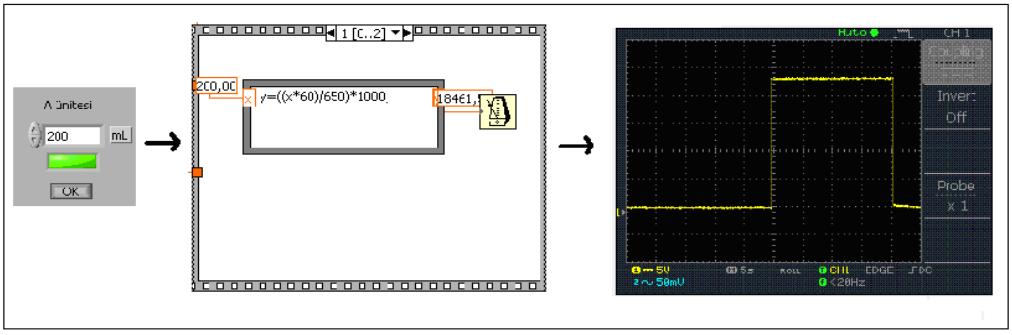
Şekil 2.16. 100 ml sıvıyı transfer etmek için pompanın çalışma süresi

Yukarıda ki Şekilden görüldüğü gibi 100 ml sıvı için pompanın beslenme süresi 9,23 sn olarak görülmektedir. Osiloskopta pompanın uçlarındaki gerilimin dalga şekli gözlenmektedir.



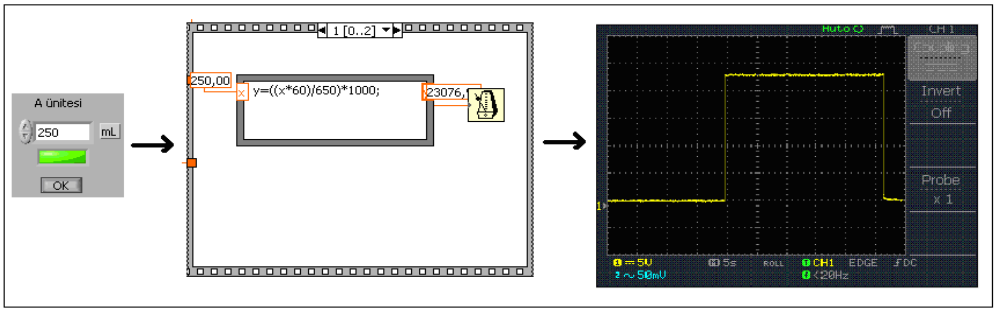
Şekil 2.17. 150 ml sıvıyı transfer etmek için pompanın çalışma süresi

Yukarıda ki Şekilden görüldüğü gibi 150 ml sıvı için pompanın beslenme süresi 13,8 sn olarak görülmektedir. Osiloskopta pompanın uçlarındaki gerilimin dalga şekli gözlenmektedir.



Şekil 2.18. 200 ml sıvıyı transfer etmek için pompanın çalışma süresi

Yukarıda ki Şekilden görüldüğü gibi 200 ml sıvı için pompanın besleme süresi 18,4 sn olarak görülmektedir. Osiloskopta pompanın uçlarındaki gerilimin dalga şekli gözlenmektedir.

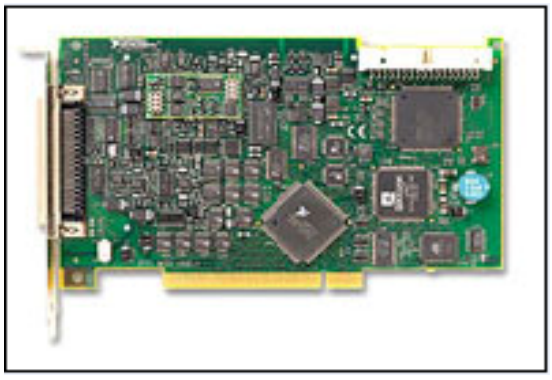


Şekil 2.19. 250 ml sıvıyı transfer etmek için pompanın çalışma süresi

Yukarıda ki şekilden görüldüğü gibi 250ml sıvı için pompanın besleme süresi 23 sn olarak görülmektedir. Osiloskopta pompanın uçlarındaki gerilimin dalga şekli gözlenmektedir.

2.5. Data Acquisition (DAQ) Kartı

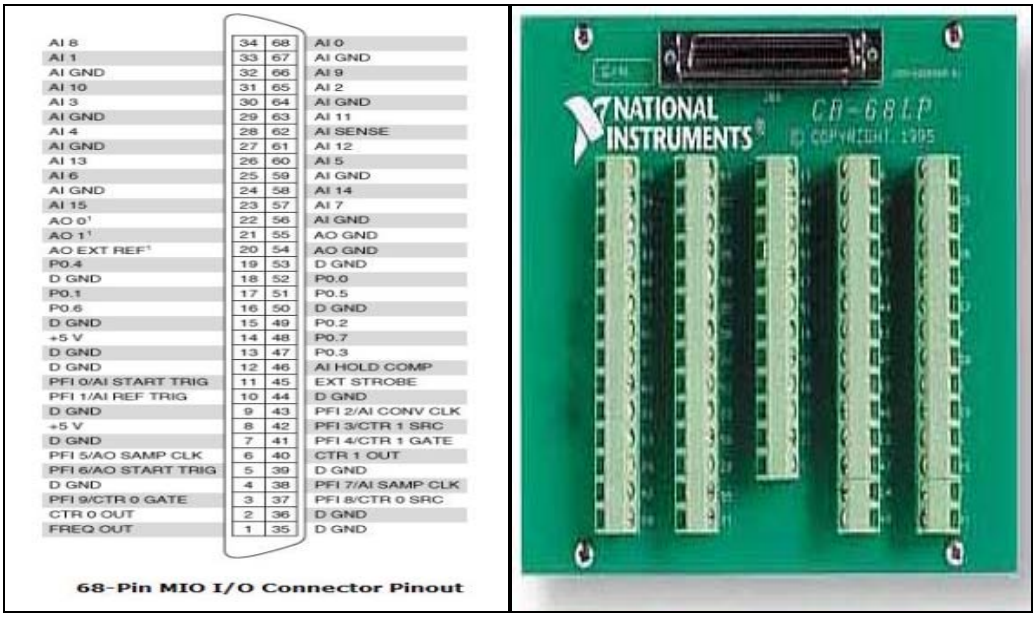
Yapmış olduğum çalışmada, National Instruments firmasına ait PCI-MIO-16E-1 16Ch, 1.25MS/s, 12-Bit, Multifunction I/O kartını kullandım. Bu kart çok amaçlı veri toplama kartı olarak kullanılmaktadır. LabVIEW ortamında yapmış olduğum simülasyan programı sıvı tarnsfer pompalarını sürmek için tasarladığım sürücü devre ile bu kart üzerinden haberleşmaktedir. Yapmış olduğum sistemde bu karta ait anaolog çıkışlar kullanılmaktadır. Bu kart, form faktörü olarak PCI, eş zamanlı kanal sayısı 16, bit sayısı 12, ölçme parametresi DCV, Maksimum frekansı 625 kHz, minimum DC gerilim sınırı 0.05 V, maksimum DC gerilim sınırı 10 V olamaktadır. Şekil 2.20'de PCI-MIO-16E-1 görülmektedir.



Şekil 2.20. DAQ kartı

2.6. CB-68LP Kartı

PCI-MIO-16E–1 kartı RTS kablo üzerinden CB-68LP kartına bağlanmaktadır. DAQ Assistant üzerinden tanımladığımız anolog çıkış kanalı PCI-MIO-16E–1 kartı üzerinden CB-68LP kartına aktarılmakta buradan sırasıyla 21,22,55 inci pinlerden sürücü devreye anolog çıkış alınmaktadır. Şekil 2.21, 68 adet pin giriş çıkışları belirtilmektedir.



Şekil 2.21. CB-68LP kartı ve giriş çıkışlar

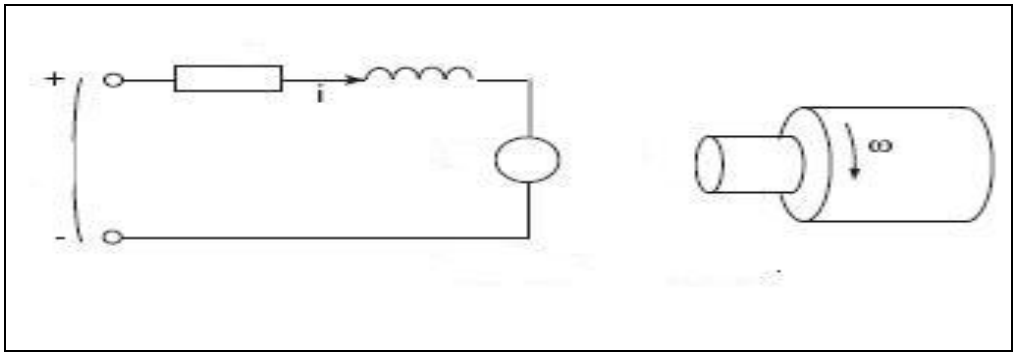
2.7. Sürücü Devre ve Pompa Sistem

Sistemi kontrolünü yapmak için sürücü devresi tasarlandı. Sürücü devre 24 VDC gerilim ile beslenmektedir. Sürücü devre için kullanılan elamanlar 2 adet 4 N25 entegre,2 adet tristör ve iki adet 1kΩ direnç kullanılmıştır. Sistem iki adet pompa içermektedir. Bu pompalar sıvı transferi için özel seçilmiştir. Aşağıda ki Şekil 2.22'de özellikleri verilmektedir. Kullanılan pompalar Hargraves(ABD) firmasına ait olup 24 volt DC gerilim ile çalışmaktadır. Dakikada 650 ml sıvı transferi yapabilmektedir ve resimden de anlaşılabileceği üzere boyutları çok küçüktür. Bu özelliğinden dolayı yer konusundan da bir problem yaşanmamaktadır.

Şekil 2.22. Kullanılan pompa ve teknik özellikler

2.8. DC Motorun Dinamik Davranışı

Motordan geçen akım i(t) olarak verilmiştir. R direnç ve L de motora ait endüktastır. Akım DC motordan geçince bir elektro motor kuvveti Em oluşturur. Bu kuvvet açısal hıza sebeb olur w(t). Bu açısal hız elektro motor kuvvetiyle Em=Km.w(t) şeklinde orantılıdır. Km motor sabitidir. Jm kalkış momentidir. Sürtünmeden kaynaklanan sabit ise γ dır.Denklem (4), (5) DC motorun dinamiğini ifade etmektedir. Şekil 2.23'de dinamik eş değer devre görülmektedir [14].



Şekil 2.23. Dinamik eş değer

$$L\frac{di}{dt} = u(t) - i(t).R - Km.w(t) \qquad (4)$$

$$Jm.\frac{dw}{dt} = Km.i(t) - \gamma.w(t) \qquad (5)$$

Sistemde kullanılan pompalarda rotorun oluşturduğu w(t)'ye bağlı olarak diyafram üzerinde titreşimlere sebeb olup sıvıyı transfer etmektedir.

3. SONUÇLAR VE BULGULAR

Yaptığım ölçümler sonucunda, sistemin gerçek zamanlı çalışmasına ilişkin elde ettiğim sonuçlar aşağıdaki Şekil 6.24'deki gibidir.

	A	B	C
1	Sıvı transferi (ml)	Pompanın nominal çalışma süresi (sn)	Yapılan Ölçümler (sn)
2	100	9,23	10,05
3	150	13,8	14,37
4	200	18,4	19,65
5	250	23	24,17
6	300	27,6	28,63
7	350	32,3	33,45
8	400	36,9	37,65
9	450	41,5	42,62
10	500	46,1	47,29
11	550	50,7	51,82
12	600	55,3	56,53
13	650	60	61,14
14	700	64,5	65,35
15	750	69,2	70,45
16	800	73,8	74,93
17	850	78,4	79,61
18	900	83	84,65
19	950	87,6	88,83
20	1000	92,2	93,97

Şekil 2.24. Sıvı transferi için gerekli olan zamanlamalar

Yukarıdaki Şekil 2.24'de sıvı transferi için gerekli olan zamanlamalar yaptığım ölçümler sonucunda karşımıza çıkmaktadır. Örneğin, 100 ml sıvı transferi için pompanın çalışma süresi 9,23 sn olması gerekirken bu süre 10,05 sn bulmaktadır. Aynı şekilde 200 ml sıvıyı transfer etmek için pompanın çalışma süresi 13,8 sn olması gerekirken bu süre 14,37 sn bulmaktadır. Anlaşılan şu ki yaklaşık olarak 0,5 sn lik bir gecikme yaşanmaktadır. Bunun sebebi ise simülasyon ve tasarlanan sürücü devreden kaynaklanmaktadır. Sistemin optimum değerlerde çalışması ise aşağıda 100 ml den 1000 ml ye kadar sıvı transferi için adım adım ölçülen değerler açıklanmıştır.

- 100 ml sıvı için pompanın beslenme süresi 9,23 sn olarak görülmektedir.
- 150 ml sıvı için pompanın beslenme süresi 13,8 sn olarak görülmektedir.
- 200 ml sıvı için pompanın beslenme süresi 18,4 sn olarak görülmektedir.
- 250 ml sıvı için pompanın beslenme süresi 23 sn olarak görülmektedir.

- 300 ml sıvı için pompanın beslenme süresi 27,6 sn olarak görülmektedir.
- 350 ml sıvı için pompanın beslenme süresi 32,3 sn olarak görülmektedir.
- 400 ml sıvı için pompanın beslenme süresi 36,9 sn olarak görülmektedir.
- 450 ml sıvı için pompanın beslenme süresi 41,5 sn olarak görülmektedir.
- 500 ml sıvı için pompanın beslenme süresi 46,1 sn olarak görülmektedir.
- 550 ml sıvı için pompanın beslenme süresi 50,7 sn olarak görülmektedir.
- 600 ml sıvı için pompanın beslenme süresi 55,3 sn olarak görülmektedir.
- 650 ml sıvı için pompanın beslenme süresi 60 sn olarak görülmektedir.
- 700 ml sıvı için pompanın beslenme süresi 64,5 sn olarak görülmektedir.
- 750 ml sıvı için pompanın beslenme süresi 69,2 sn olarak görülmektedir.
- 800 ml sıvı için pompanın beslenme süresi 73,8 sn olarak görülmektedir.
- 850 ml sıvı için pompanın beslenme süresi 78,4 sn olarak görülmektedir.
- 900 ml sıvı için pompanın beslenme süresi 83 sn olarak görülmektedir.
- 950 ml sıvı için pompanın beslenme süresi 87,6 sn olarak görülmektedir.
- 1000 ml sıvı için pompanın beslenme süresi 92,2 sn olarak görülmektedir.

4. ÖNERİLER

LabVIEW tabanlı yapmış olduğum sistem iki anolog çıkışa sahip ,bir başka değişle iki kanaldan oluşmaktadır.Bu kanallar iki adet pompayı kontrol etmektedir.Programda kontrolör olarak mantıksal denetleyici tasarlanmıştır.Pompaların sıvı transfer süreleri bu denetleyici tarafından sağlanmaktadır.Fakat çok kanallı yapılarda,sistem için daha kararlı bir kontrolör tasarlanmalıdır.Kanal sayısı artıkça mantıksal denetleyici yerine bu tür yapılar için PID ve FLC kontrolör tasarlanarak çok kanallı yapılarda daha kararlı sistem oluşturulabilir.LabVIEW programı sahip olduğu PID ve FLC Tools fonksiyonu sayesiyle bu konuda çok kalaylık sağlıyabilmektedir.İleriki çalışmalarda aynı sistemin kontrolünü PID ve FLC Tools fonksiyonlarını kullanarak daha kararlı ve sistematik bir çalışma yapmayı düşünüyorum.

5. KAYNAKLAR

1. Bitter, R. ve Nawrocki, M., LabVIEW Advanced Programing Techniques Book, Second Edition, Springer Berlin Heidelberg, New York, 2006.

2. Sumathi, S. ve Surkka, P., LabVIEW based Advanced Instrumentation Systems, Taylor & Francis Group LLC., New York, 2007.

3. www.ni.com, National Instruments Corp. LabVIEW Basics I CBT Exercises, 11 Mayıs 2009.

4. www.ni.com, National Instruments Corp., LabVIEW Simulation Interface Toolkit User Guide, 15 Mayıs 2009.

5. Blume, P. A., The labVIEW Style Book, Second Edition, Springer, NewYork, 2007.

6. Teng, J. H., Chan, S. Y., Lee, J. C. ve Lee R., A LabVIEW Based Virtual for Power Analyzers ,IEEE Trans. on Instrumentation and Measurment, 2 (2000) 179-183.

7. Foster, N. L., Haze, D. P. ve Basher, H., Closed-Loop Position Control System Using LabVIEW, IEEE Trans. on Instrumentation and Measurment, 3 (2002) 283-286.

8. Jerome, J., Aravid, A. P. ve Arunkumar, V., LabVIEW based Intelligent Controllers for Speed Regulation of Electric Motor, IEEE Trans. on Instrumentation and Measurment, 4 (2005) 935-940.

9. Saliman, A. ve Hasanul, A, Control of a Radio-Telescope via the Internet, IEEE Trans. on Instrumentation and Measurment, 3 (2006) 40-45.

10. Thepsatorn, P., Numsomran, A. ve Tipsuwanpon, V., DC Motor Speed Control using Fuzzy Logic based on LabVIEW, SICE-ICASE International Joint Conference Oct. 2006, Bexco Korea, Busan, 3617-3620.

11. Wenhai H., Design of the Measurement System of the Pump Based on LabVIEW, IEEE Trans. on Instrumentation and Measurment, 5 (2007) 475-478.

12. Tang, Q., Wang, Y. ve Guo, S.,Design of Power System Harmonic Measurement System Based on LabVIEW, IEEE Trans. on Instrumentation and Measurment, 4 (2008) 489-493.

13. Wu, B. ve Cai, C.,. Hydraulic Monitoring System Based on LabVIEW, IEEE Trans. on Instrumentation and Measurment , 7 (2008) 254-258.

14. Sharma, V., Development of Control Lab Interface for Data Acquisition using Lab VIEW ,Masters' Degree Project , Stockholm University , Sweden., 2007.

15. www.ni.com, Getting Started with LabVIEW, National Instruments, 30 Mayıs 2009.

16. Caven, J. ve Jacman, J., An con-Based Approach to System Control Development, IEEE Trans. on Instrumentation and Measurment, 4 (1990) 260-264.

17. Wu, R.C., Kang, J. ve Sung, C., Realization of Load Management by Power Line Carrier and LabVIEW, IEEE Trans. on Instrumentation and Measurment, 3 (2008) 537-541.

18. Huang, J., Hwang, R. ve Shih, W., Feasibility of Load Management on Kaohsiung Polytechnic Institute Using Networ Approaches, IEEE Trans. on Instrumentation and Measurment, 5 (1995) 631-635.

19. Swain, N., Anderson, J., Sia A., Swain, M., Fullon, M., Garret, J. ve Tucker, O., Remote Data Acquisition, Control and Analysis usingLabVIEW Front Panel and Real Time Engine, IEEE Trans. on Instrumentation and Measurment, 4 (2003) 1-6.

20. Zhenmei, L., Shen, J., Wang, Y. ve Yan, Y., Remote Monitoring System of Power Network Based on Virtual Instrument and Wavelet Transform, IEEE Trans. on Instrumentation and Measurment, 5 (2007) 203-206.

21. Liu, J. ve Zhao, X., Instrument Design Based on LabVIEW**,** Publishing House of Electronics Industry, 7 (2003) 115-127.

22. Jianhua, K. , Baoqiang, W. , Jie, G. ve Yanjie, W.,. Research on Lightning Location Method Based on LabVIEW, IEEE Trans. on Instrumentation and Measurment, 5 (2007) 86-90.

23. Yuanjiang, L., Yingzhong, T. ve Ming, L., The Design of Dynamic Integrated Testing System Using Digital AC Servo Motor Based on LabVIEW, Shanghai Leading Academic Discipline, 8 (2005) 1700-1705.

24. Chengwei,L., Limei, Z. ve Xiaoming, H., The Study on Virtual Medical Instrument based on LabVIEW, IEEE Trans. on Instrumentation and Measurment, 5 (2005) 4072-4075.

25. O'Brien, E., LabVIEW usage as part of the biomedical engineering senior design experience, IEEE Trans. on Instrumentation and Measurment, 6 (2002) 2595-2599.

ÖZGEÇMİŞ

30.05.1982 tarihinde Amasya'da doğdu. İlkokulu Amasya Beyazıt İlkokulunda, ortaokul ve liseyi Amasya Anadolu Lisesi'nde tamamladı. 2001 yılında Karadeniz Teknik Üniversitesi, Mühendislik Mimarlık Fakültesi, Elektrik-Elektronik Mühendisliği Bölümünü kazandı. 2006 yılında aynı bölümden mezun oldu. Aynı yıl güz döneminde Elektrik Elektronik Mühendisliği Anabilim Dalı'nda yüksek lisans eğitimine başladı. 2006-2007 yılları arasında özel bir firmada mühendis olarak çalıştı. Aralık 2007'de Elektrik-Elektronik Mühendisliği Bölümün'de Araştırma Görevlisi olarak göreve başladı. Yabancı dil olarak iyi düzeyde İngilizce bilmektedir.