İstanbul Üniversitesi

Mühendislik Fakültesi

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

NI DAQ ARABİRİMİ TASARIMI

Yöneten: Yard. Doç. Dr. Mustafa DAĞTEKİN

Hazırlayan: Hüseyin KOZAN (1306040082)

DİPLOMA PROJESİ

Haziran 2009

İÇİNDEKİLER

ŞEKİL LİSTESİ

ÖZET

Proje ile National Instruments firmasının ürettiği ve NIDAQmx sürücüsü ile desteklediği veri toplama kartlarından analog sinyal kaydetme ve kaydedilmiş veriyi yükleme, dijital çıkışlardan saat darbesi gönderme, analog çıkıştan istenen gerilim değerini çıkarma işlemlerini gerçekleyecek C++ grafik arabirimi tasarlanacaktır. Ayrıca kaydedilen analog sinyalin üstünde ortalama alma ve FFT dönüşümü gibi sinyal işlemleri de gerçekleştirilebilecektir.

ABSTRACT

With this project a C++ graphical user interface will designed for data acquisition cards which are produced by National Instruments and supported by their NIDAQmx software driver to collect, save and open saved analog waveforms, produce clock pulse from digital outputs, and produce desired output voltage from analog output. And also some signal processing businesses will take place on saved waveforms like averaging and FFT transform.

1. GİRİŞ

Günümüzde üretilen dijital veri alma kartları üreticilerin sunduğu kullanımı ve programlanması kolay yazılım ortamları ile desteklenmektedir. Bu yazılım ortamları genellikle üreticinin tüm ürünlerini desteklemekte ve bu ürünler ile temel sinyal işleme fonksiyonları ve kendi görsel program tasarım ortamlarının yanı sıra .NET Framework ve ANSI C programlama kütüphaneleri ile donanım sürücüsüne programsal olarak erişmeye

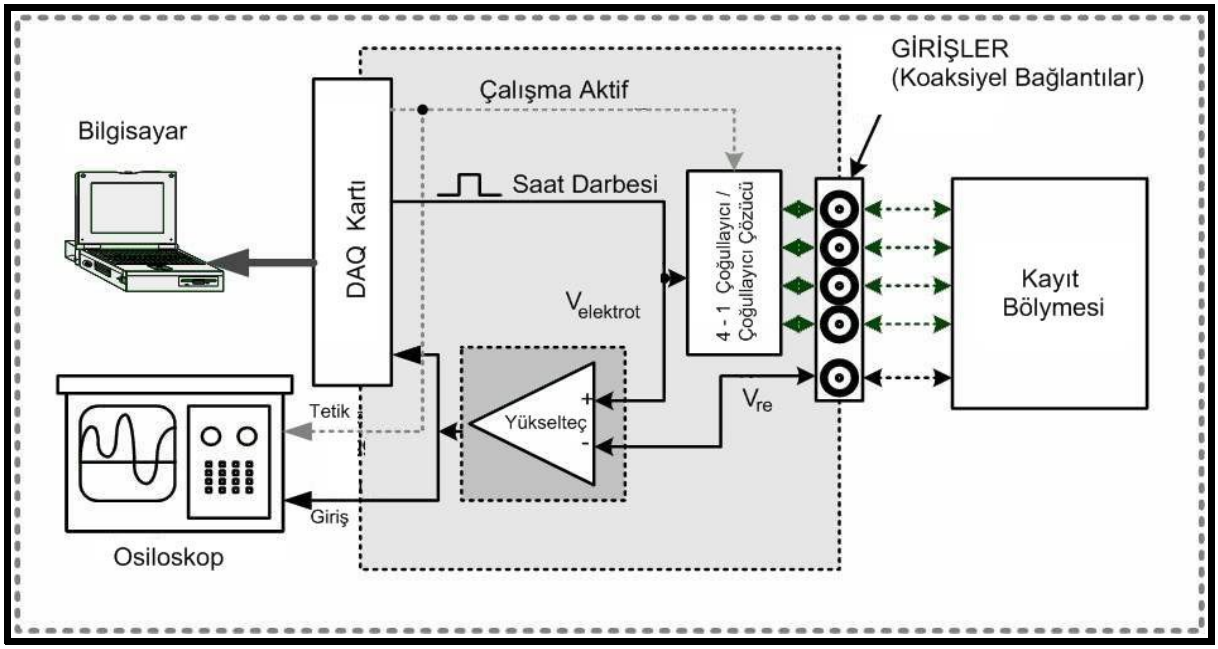
olanak sağlamaktadırlar.

Üreticilerin programlama ortamları her ne kadar avantajlı gibi görünse de programlama ortamlarının ve diğer ticari paketlere verilen desteğin birer ürün olarak sunulması ve maliyetinin yüksek olması kullanıcıları başka arayışlar içine sokmaktadır. Artan maliyetten belki daha da önemlisi kullanılmak istenen iş için program arayüzünün o işe adanmış olarak sunulamaması olabilir.

Bu proje ile özellikle maliyet unsurundan etkilenmeden kullanıcıya yapacağı deneyde kullanımı kolay bir arayüz ile işlemlerini kolaylaştırmak amacını gütmekteyiz.

Projede programlama maliyetini arttıran lisanslı geliştirme ortamları yerine açık kaynak kodlu olan Qt programlama kütüphanesi ve veri alma kartının (DAQCard – Data Acquisition Card) üreticisinin ücretsiz sunduğu ANSI C sürücü kütüphanesi kullanılmıştır.

Yapılacak olan proje ile desteklenecek olan proje, bir nöron kayıt cihazından gelen verileri analiz ederek kuvvetlendirici tasarımı geliştiren bir projedir. Bu projede canlı nöron hücrelerinden gelen verilerin minimum hata ile elde edilmesi için tasarlanıp üretilen devre şeması Şekil 1'deki gibidir.



Şekil 1 : Nöron kayıt cihazı

2. KULLANILAN ARAÇLAR

2.1. Örnekleme Teorisi

Örnekleme, sürekli bir sinyalin belirli aralıklarla ölçülerek ayrık bir sinyale

dönüştürülmesidir. Havada yayılan ses dalgası ya da gözlerimizle algıladığımız ve görmemizi sağlayan ışık huzmeleri birer sürekli sinyallerdir. Sesin bir dijital kayıt cihazında analog-dijital dönüştürücü yardımı ile dijitale çevrilmesi, ışığın fotoğraf makinesi ile bir "anının" kayıt edilmesi birer örneklemedir.

Ölçüm yapılmak istenen frekansın doğru bir şekilde elde edilebilmesi için istenen frekansın en az iki katı hızda ölçüm yapılması gereklidir. Bu olgu **Nyquist-Shannon Ölçme Teoremi [Ny]** olarak da bilinen şu teorem ile açıklanır;

> "*Eğer bir x(t) fonksiyonu B'den daha fazla bir frekansa sahip değilse fonksiyonun değeri saniyede 1/(2B) aralıklarla tam olarak elde edilebilir.*"

Teoremden de anlaşılacağı üzere örnekleyici cihazın **örnekleme frekansı** (ya da örnekleme oranı) örneklenmek istenen sinyalin frekansının iki katı frekansta olmak zorundadır. Örnekleme frekansının yarısı olan bu frekansa **nyquist frekansı** (f_N) denir. Nyquist frekansının üzerindeki sinyallerin örneklenmesi durumunda örneklenen frekans belirsizdir. Bu yüzden bir f frekansındaki bileşenler (N, sayma sayıları olmak üzere) $N f_N + f$ ve $N f_N - f$ frekansındaki bileşenler tarafından algılanamazlar. Bu belirsizliğe **örtüşme** *(aliasing)* denir.

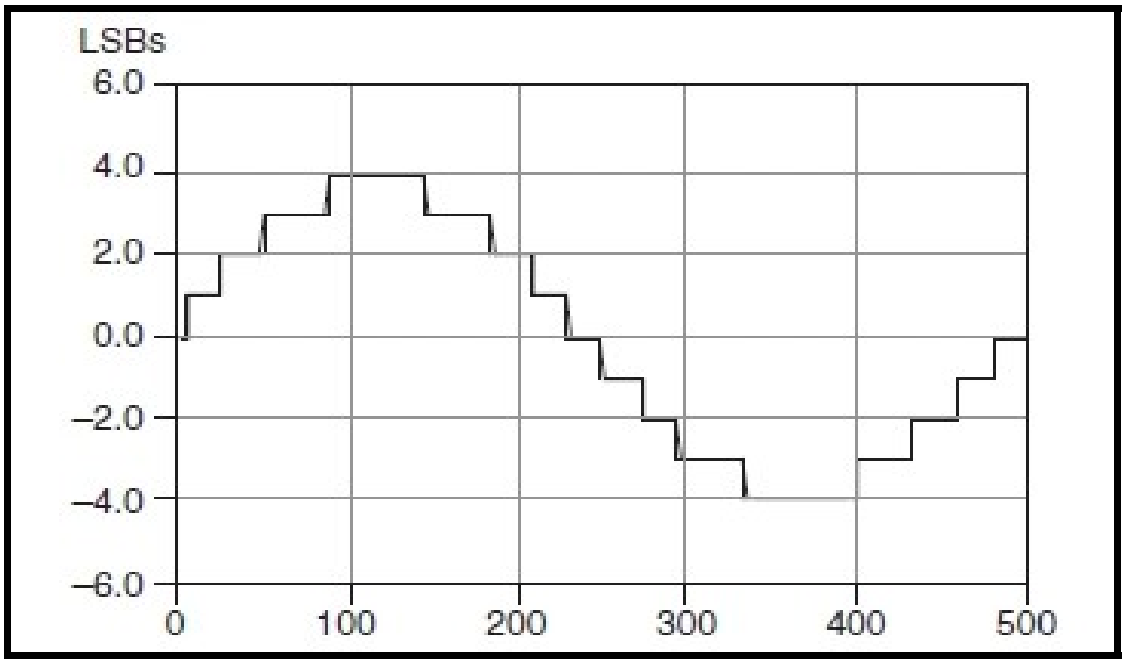
Bu problemi olabildiğince azaltabilmek için analog sinyale ölçüm yapmadan önce örtüşme önleyici (*anti-aliasing*) filtre (ya da çoğunlukla alçak geçirgen filtre – low-pas filter) uygulanır.

Örneklemenin bu temel teoremi dışında ideal olmayan bir analog-dijital çevirici (*ADC – Analog Digital Converter*) aygıtının da sınırlamaları vardır. Bunlardan biri örneklemenin anlık bir zamanda değil de bir zaman aralığının ortalaması olmasıdır. Bu fiziksel bir kısıtlamadır. Kapasitör tabanlı örneklemede kapasitörün geriliminin anlık olarak değiştirilememesinden kaynaklanır.

Bir diğer sınırlama da sensörlerin ve analog devrelerin kendisinin ürettiği gürültülerdir. Bir diğeri de ADC çıktısının yeterli hızda donanım tamponundan alınamayarak ADC'nin belleğini aşan verilerin alınamamasıdır. Bir diğeri de ölçüm sonrasında üretilen değerlerin sınırlı sayıda bit ile ifadesinden kaynaklanan hassasiyet derecelendirmesinin yetersiz kalmasıdır.

Analog-dijital çevrim işleminin tersi olan dijital-analog çevrim işleminin de bazı fiziksel sınırlamaları mevcuttur. Bunlardan analog devre gürültüsü, yeterli hızda dönüşüm gerçekleştirilememesi gibi bazıları ADC'nin sınırlamalarına benzerler. Bir başka sınırlama ise çıkış sinyalinin anlık değişimler yerine dikdörtgensel değişimler ile sinyalin oluşturulması

sonucunda kazancın düşebilmesidir. Bir DAC'nin örnek sinyal çıktısı Şekil 3'teki gibidir.



Şekil 2 : DAC Sinyal Çıktısı

2.2. DAQ Kartı

DAQ, **Data Acquisition** kelimelerinin kısaltılmış halidir ve Türkçe'ye **veri toplama** olarak çevrilebilir. Veri toplama kartları doğada bulunan verileri toplayıp bilgisayarda işlenebilecek hale çevirirler. Bu işlem üç aşamada özetlenebilir.

Birinci safhada fiziksel ortamdaki özelliklerin gerilim, akım gibi elektriksel halde ifade edilebilecek şekle getirmelidirler. Bu işlem sensörler aracılığı ile yapılır. Sensörlere örnek olarak; mikrofon, fotosel, gerilim ölçer, termistör gibi yapılar verilebilir.

İkinci safhada sensörlerden gelen ve analog olan sinyallerin yükseltilip veya alçaltılıp örneklenerek bilgisayara gönderilmesidir. Yükseltme ve alçaltma işlemi sensörden sonra gelen bir elektronik devre ile gerçekleştirilebileceği gibi DAQ kartı yardımı ile de izin verilen ölçüde gerçekleştirilebilir. Örnekleme işlemini ise DAQ kartının içindeki ADC (*Analog Digital Converter – Analog Dijital Çevirici*) yardımıyla gerçekleştirilir.

Üçüncü safhada toplanan dijital verinin içinden istenen özelliklerinin çıkarılmasıdır. Bu işlem dijital veri üstünde çalışan yazılımların yardımı ile olabileceği gibi ikinci safhada olan DAQ kartının programlanarak istenen özellikte (belirli zaman aralıklarıyla, belirli frekans altında/üstünde...gibi) verilerin elde edilmesi ile de gerçekleştirilebilir.

DAQ kartı ile ilgili buraya kadar bahsedilenler daha çok analog verinin alımı ile ilgili olan kısımlardır. Bu tür kartlar sadece analog veri alımı ile sınırlı kalmazlar. Dijital veri

alımı/gönderimi ve analog veri gönderimi işlemlerini de yapabilirler. Ayrıca barındırdıkları zamanlayıcı (*timer*), sayıcı (*counter*) gibi yapıları da veri almada ve vermede kolaylıklar sağlamaktadırlar.

Bu tür kartlarda toplanan verinin aktarımı için gerekli süre bekletilmesi ve/veya kartın programlanması için yeterli miktarda bellek bulunur. Bilgisayar işlemcisi ya da veri yolu ile haberleşirken kesme yönetimli (*interrupt*), programlanmış giriş/çıkış (programmed I/O) veya doğrudan bellek erişimi (*DMA*) yöntemleri kullanılarak sürekli gelen verinin sürekli (*real-time*) çalışamayan bilgisayar mimarisi ile en verimli şekilde çalışması sağlanır.

Kartlar üzerindeki mikrokontrolörleri programlamada ve/veya bilgisayarda çalışan işletim sistemi ile cihazdan veri alımı esnasında yapılan programlamada, üreticiler programlama arabirimleri (API) sunarlar. Sağlanan bu arabirimler sayesinde programlama işlemi kolaylaştırılır.

2.3. Neden DAQ Kartı ?

DAQ kartları veri toplama için tek yol değildirler. Daha çok otomasyonda kullanılan PLC (Programmable Logic Contoller – Programlanabilir Lojik Kontrolcü) ve mikrokontrolörler de (PIC, AVR ... vs) veri toplama işlemi için kullanılabilir.

Sanayide çokça kullanılan PLC, manyetik alan, büyük sıcaklık farkları ve tozlu ortamlar gibi zor koşullara dayanıklı ürünlerdir. İçerisinde ADC, ağ modülleri barındıran PLC modülleri bilgisayara bağlanarak veri toplama işlemi yapabilmesine rağmen DAQ kartlarından düşük olan örnekleme hızları ve esnek olmayan programlama olanakları sebebiyle tercih edilmemiştir.

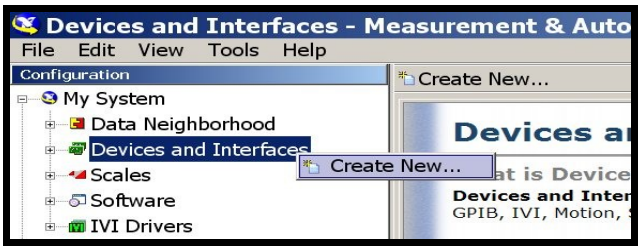
Alt düzey programlama ve örnekleme için özel olarak devre tasarımı gerektiren mikrokontrolörlerin programlama ve özel devre gereksinimi sebebiyle tercih edilmemişlerdir.

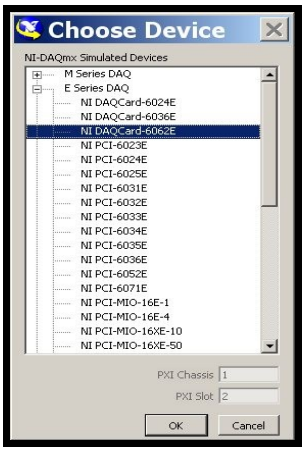
Test ve ölçme işlemlerini kolaylaştıran DAQ kartları veri toplama için en uygun sistemlerdir.

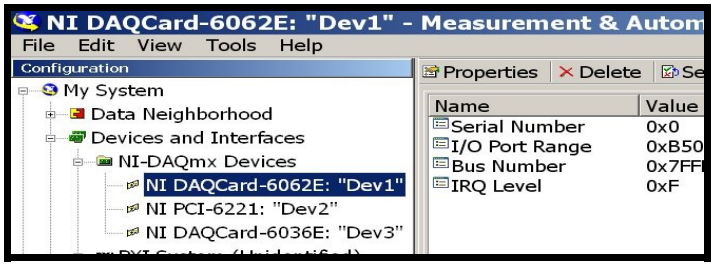
2.4. NIDAQmx Sürücü Yazılımı

Bahsi geçen deneyde kullanılması planlanan DAQ kartı olan NI DAQCard 6062E için gerekli olan sürücü yazılımı NIDAQmx'tir [NIDAQmx]. Sürücü ile birlikte .NET Framework desteği, Labview desteği, LabWindows desteği, Microsoft Visual Studio C++ için Measurement Studio desteği ve ANSI C desteği yanında Measurement & Automation Explorer (MAX) yazılımı da gelmektedir.

MAX (*Measurement & Automation Explorer*) ile var olan National Instruments firmasına ait fiziksel veri alma donanımlarının sürücülerini yönetmek dışında, ölçümde kullanılacak yapılandırmayı içeren görevler oluşturma, sanal sürücüler oluşturup cihazlara ait testler ve kalibrasyonlar yapılabilmektedir. MAX yardımı ile bir simüle edilmiş sürücünün eklenmesi aşağıdaki şekillerde gösterilmiştir.







Şekil 3 : MAX ile sanal sürücü eklenmesi işlemi

Simüle edilmiş sanal sürücüler bazı kısıtlamalar ve önceden belirlenmiş ayarlar ile gelmektedir. Bunlardan bazıları şu şekildedir:

- Simüle edilmiş aygıt, fiziksel bir aygıt ile aynı görevde (*task*) kullanılamaz.
- Analog giriş sinyali %3 gürültü oranına sahip bir sinüs sinyali ile simüle edilmiştir.
- Bir görevde (*task*) birden fazla analog giriş kanalı kullanılmış ise her kanaldaki sinüs sinyali zaman düzleminde 5° kaydırılarak simüle edilir.
- Dijital giriş portuna birer sayıcı bağlanmıştır.
- Sayıcı işlemi görevlerde simüle edilmez.
- Sayıcı verisi her zaman 0 olarak okunur.
- Aygıtın çalışması için zamanlanmış bir döngü kullanılamaz.
- Donanıma bağlı olay tetikleyicileri çalışmaz (*örnekleme saat olayı – sample clock event*)
- Gözcü saatinin *(wachdog timer)* süresi dolmaz.

Sürücü yazılımı ile birlikte gelen ANSI C desteğini sağlayan kütüphane ve başlık dosyasının yanında bazı temel işlemlerin örnek kodları ve NIDAQmx C referans Yardım dosyası da gelmektedir.

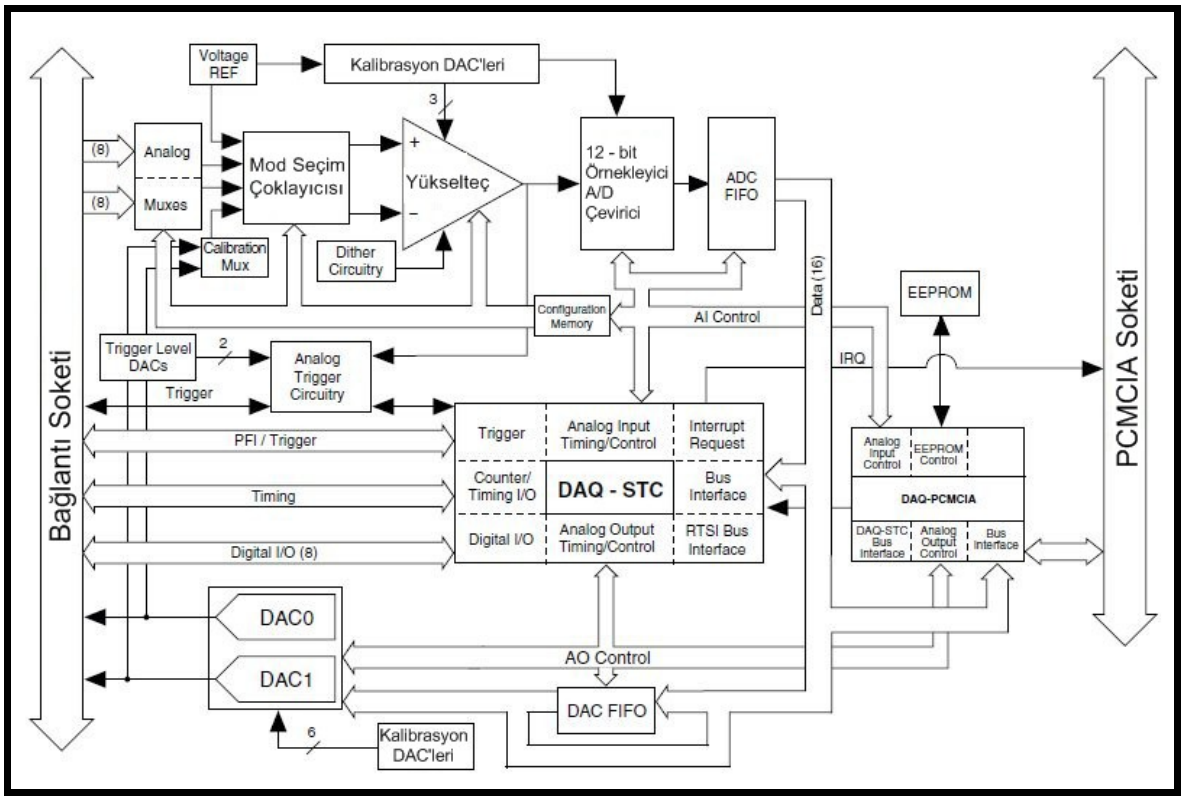
2.5. DAQCard 6062E

Bu projeni destek verdiği deneyde kullanılması planlanan DAQ kartı National Instruments firmasının DAQCard 6062E modelidir [6062E]. Kart, PCMCIA formda olup aşağıdaki özellikleri barındırır:

- 1 adet 12 bit ADC.
- 2 adet 12 bit DAC.
- 8 hat Dijital I/O.
- 2 adet 24 bit Sayıcı / Zamanlayıcı.

Aygıtın üstünde analog giriş bir çoklayıcı (*multiplexer*) tarafından çoklanarak ADC'ye (*Analog Digital Converter*) verilmiştir. Çoklanan bu analog girişlere kanal adı verilmiştir. ADC'nin çevrim kapasitesi birden fazla kanala bölünerek aynı anda birden fazla analog

çevrim yapılması sağlanmıştır. Aygıtın şeması Şekil 5'da verilmiştir.



Şekil 4 : DAQCard 6062E Şeması

Kullanılabilen kanal sayısı seçilen referans modeline göre değişiklik göstermektedir. Eğer bütün kanallarda ölçüm tek uçtan yapılacak ise yani referans olarak kartın toprağı kullanılacak ise bu durumda en fazla 16 kanal kullanılabilir.

Eğer bütün ölçümlerde her kanalın referansı dışardan verilecek ise bu durumda her kanal çifti için iki giriş ucu ayrılır ve en fazla kullanılabilen kanal sayısı 8 olur. Analog giriş uçlarının referansları ayrı ayrı programlanabilmektedir. Bu şekilde her kanal için referanslı veya referanssız model tek tek seçilerek 8 ila 16 arasında kanal elde edilebilmektedir.

DAQCard 6062E, analog giriş için saniyede en fazla 500.000 örnek alabilmektedir. Kartın üzerindeki FIFO tampon belleği 8129 örnek saklayabilmektedir. Kart veri transferi yaparken kesme veya programlanmış I/O yöntemlerinden birini kullanabilmektedir. Kartın diğer teknik özellikleri üreticinin sağladığı kullanıcı kitabında bulunabilir.

2.6. NIDAQmx Sürücü Yazılımı Kullanımı

NI kartlarına programlama yoluyla erişebilmek için sürücü ile birlikte gelen, ANSI C desteğini sağlayan "NIDAQmx.h" başlık dosyası kullanılarak "NIDAQmx.lib" kütüphane dosyasındaki fonksiyonlara erişim sağlanabilir. Başlık dosyasındaki fonksiyonlar aygıta

erişmede bir API (*Applicaiton Programming Interface*) oluştururlar. Bu API'yi kullanabilmek için başlık dosyası kaynak koda eklenir ve kütüphane dosyası da derleyiciye parametre olarak verilerek API fonksiyonlarının çalıştırılabilir koda erişimi sağlanır.

Projede kullanılan GCC (*GNU C Compiler*) derleyicisini Microsoft Windows üstünde çalıştırabilmek için MinGW (*Minimalist GNU for Windows*) ortamı kullanılmıştır. Donanım sürücü kütüphanesini GCC derleyicisine kütüphane olarak verebilmek için NIDAQmx.lib biçiminden NIDAQmx.a biçimine dönüştürmek gereklidir. MinGW ortamı ve gerekli araçlar kurulduktan sonra bu dönüşüm aşağıdaki kodlar bir betik dosyasına yazılarak veya tek tek uygulanarak gerçekleştirilebilir:

```
#!/bin/sh
PATH="$PATH"
dumpbin /exports "C:\NIDAQmx.lib" > nidaq.exports
echo LIBRARY nicaiu.dll > nidaq.defs
echo EXPORTS >> nidaq.defs
sed "1,/ordinal *name/d;/^$/d;/Summary$/,+100d;s/^ \+\([^ ]\+\)\( .*\|\)$/\\1/;s/^_\(.*\)
$/\1/" nidaq.exports |sort >> nidaq.defs
dlltool  -k -d nidaq.defs -l libnidaq.a c:/WINDOWS/system32/nicaiu.dll
```

Şekil 5 : Kütüphane dosyası dönüşüm işlemi

2.7. Qt

Projenin arayüz tasarımında ve programlanmasında kullanılan Qt, Windwos, Linux ve Mac OS X gibi birden çok platformu destekleyen bir uygulama geliştirme araç takımıdır [Qt]. En çok bilinen örnek programlardan KDE masaüstü ortamı, Opera ağ tarayıcısı ve Skype haberleşme yazılımı da bu araç takımı kullanılarak yazılmışlardır.

Qt'nin lisans sürümleri hem açık kaynaklı program geliştirmeyi ve dağıtmayı hem de üretici firmadan gerekli lisans alınarak ticari uygulamalarda kullanmayı mümkün kılar.

Qt'nin projede kullanılmasının birkaç sebebi olarak şunları sayabiliriz;

- C++ programlama dili kullanılarak (ki Python, Ruby, PHP, Perl, Pascal, C# ve Java gibi dilleri de desteklemektedir) kolaylıkla grafik arabirim (*GUI – Graphical User Interface*) oluşturmaya olanak sağlaması,
- grafik ara birim tasarımını hızlandıran Designer aracı,

- yazılan programın değişik dillere çevrimini kolaylaştıran Linguist aracı,
- grafik arabirim programlamadaki olay yönetimini kolaylaştıran sinyal-slot yapısı,
- eklenti (plugin) mimarisi sayesinde dağıtılan programa sonradan eklenti yapabilme imkanı tanıması,
- yazılan program kodunda çok az değişiklikler ile yapılarak Windows, Linux ve Mac OS X üzerinde çalıştırılabilir hale getirilmesi olarak sıralanabilir.

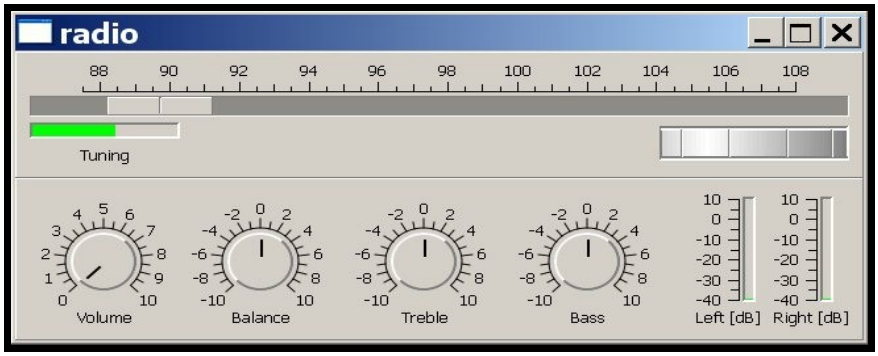
Projede Qt'nin açık kaynak lisanslı sürümü kullanılmıştır. Bu sürüm ayrıca Microsoft Visual Studio entegrasyonu desteği kurularak Visual Studio ile ve MinGW geliştirme ortamı ile birlikte kullanılabilmektedir.

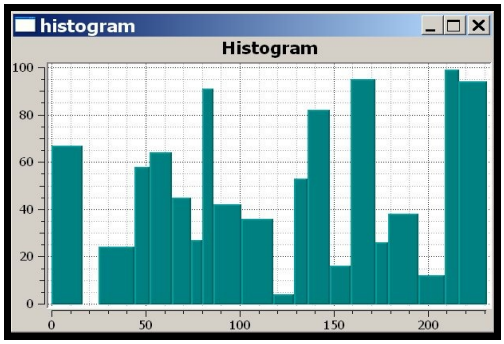
MinGW geliştirme ortamı Windows üzerinde, bir Unix derleyicisi olan GCC'yi ve geliştirme için gerekli diğer araçları sunar [MinGW].

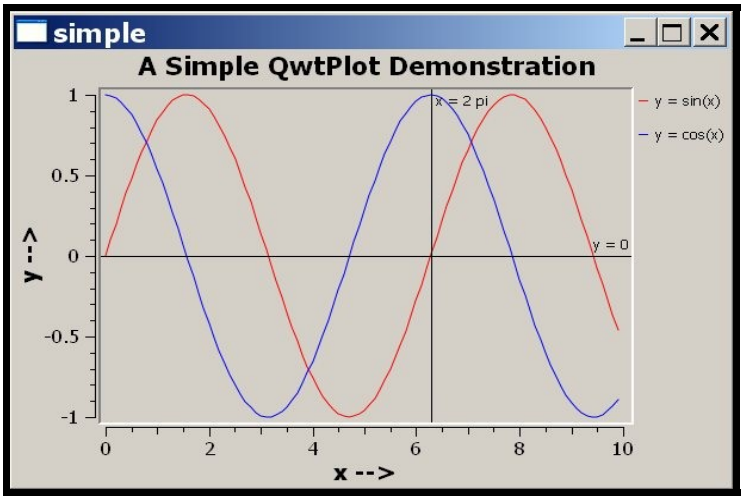
Proje geliştireme sürecinde Windows işletim sistemi kullanılmıştır. Gerek NIDAQmx sürücüsünün hem Windows hem de Linux işletim sistemlerini desteklemesi, gerekse Qt'nin bu iki işletim sistemine verdiği destek sayesinde projenin kolaylıkla Linux işletim sistemine geçirilmesi mümkün olacaktır.

2.8. Qwt

Qwt, Qt kullanılarak geliştirilen, teknik uygulamaların grafik ara birimdeki görsel ihtiyaçlarını karşılamak için geliştirilen bir araç setidir [Qwt]. Şekil 7 de Qwt ile sağlanan bazı görsel nesneler Qwt ile birlikte gelen örnek programlar kullanılarak resmedilmiştir.





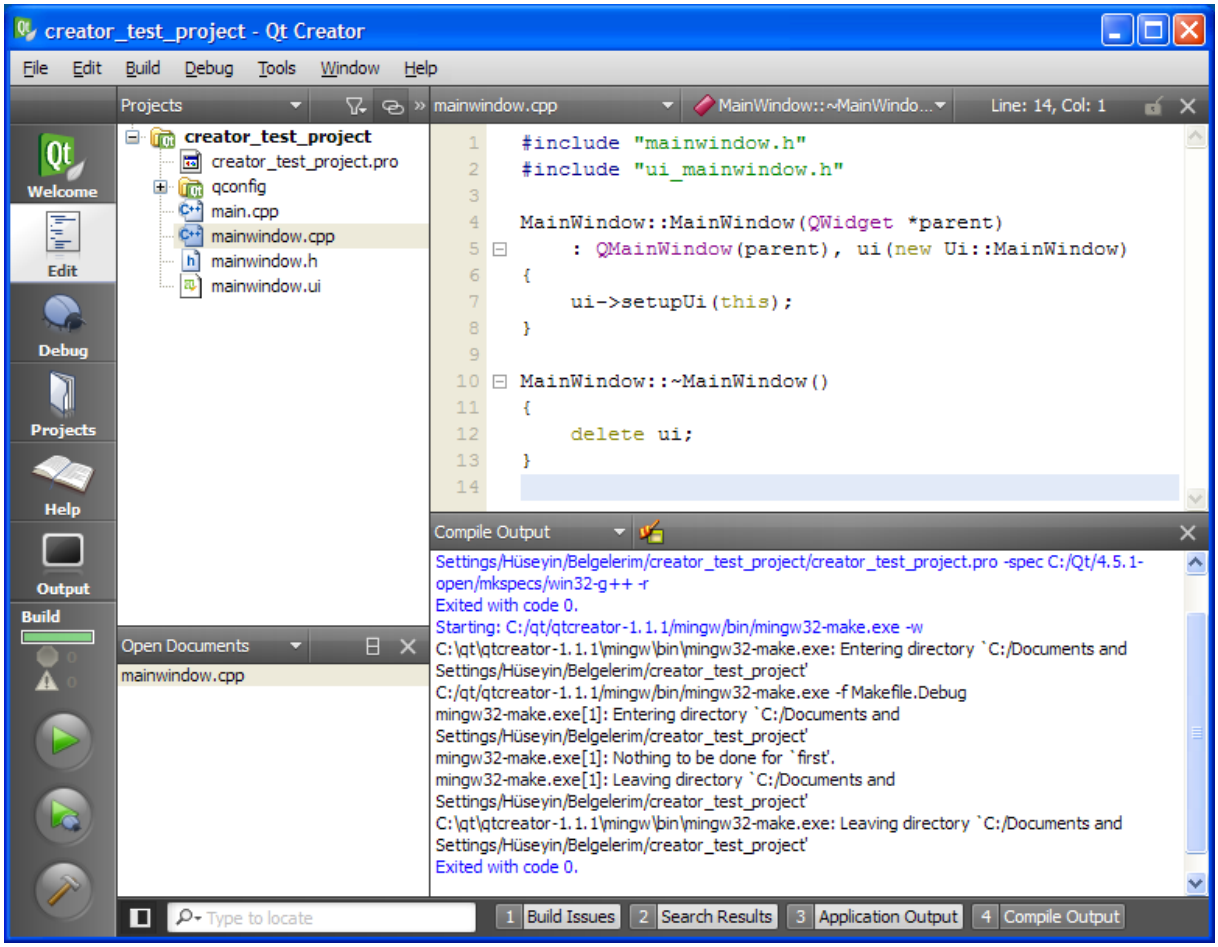


Şekil 6 : Qwt örnekleri

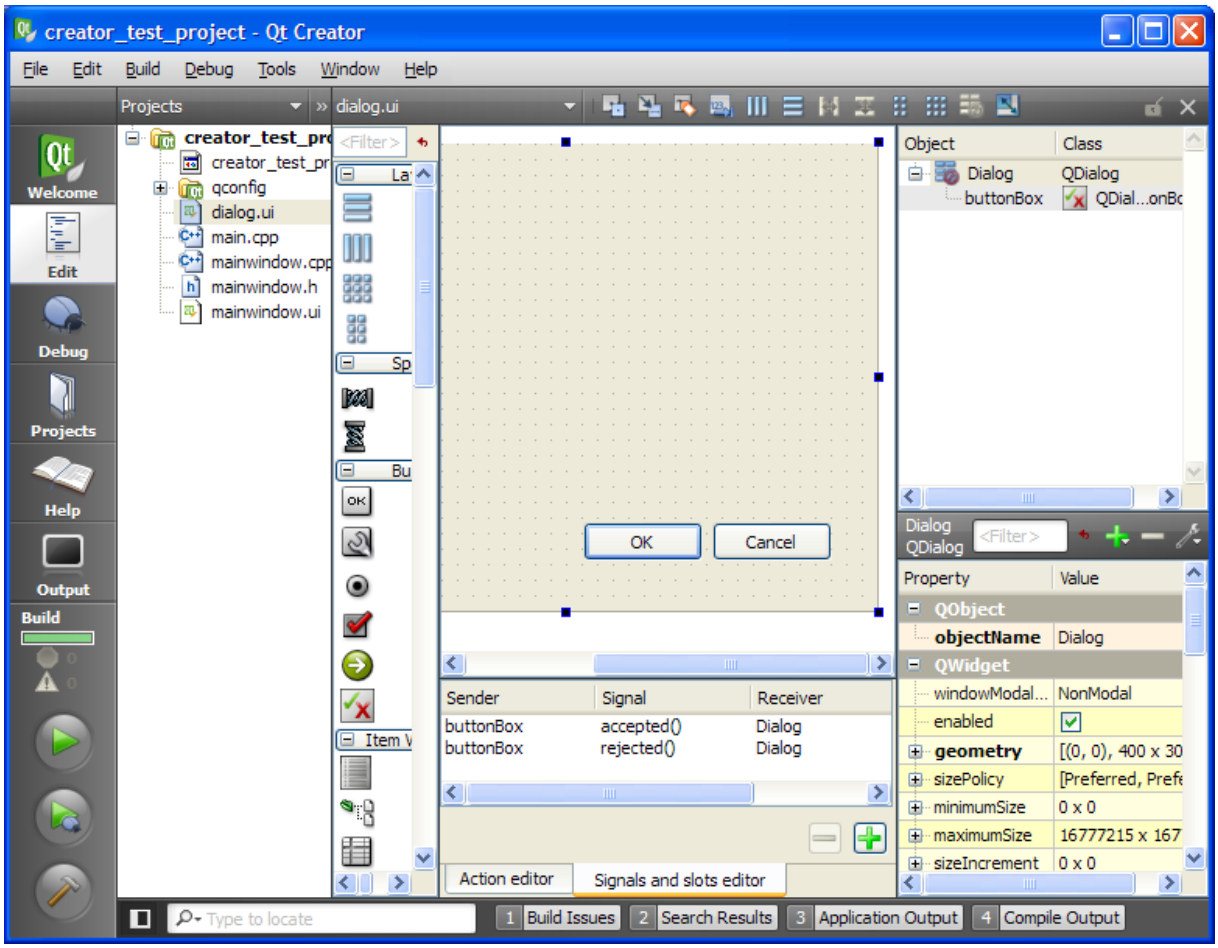
2.9. Geliştireme Ortamı : IDE

Geliştirme aşamasında Qt'nin geliştiricilerinin sağladığı Creator programı kullanılmıştır.

Creator yardımıyla kod yazmak, yazılan kodu derlemek, hata ayıklamak ve grafik arayüz tasarımı yapmak mümkündür. Creator, program kodlarını derlemek için GCC'yi, hata ayıklamak için GDB'yi ve grafik arabirim tasarlamak için Designer aracını kullanır.



Şekil 7 : Creator ekran görüntüsü - Kod



Şekil 8 : Creator ekran görüntüsü - Designer

Qt araç setinde C++ ile desteklenmeyen bazı özellikler vardır. Bunlardan bazıları;

- sinyal-slot mimarisi,
- çok kanallı programlama,
- bellek yönetimi,
- *foreach* anahtar kelimesi,
- *MetaObject* desteğidir.

Bu özellikleri destekleyebilmek için derleyiciden önce çalıştırılması gereken MOC (*Meta Object Compiler*) derleyicisi kodları C++ derleyicisinin derleyebileceği şekle getirir. Bu işlemi gerçekleştirmek için proje dosyasının bulunduğu dizinde komut satırından "qmake" komutu çağrılır. Qmake programı, desteklenmeyen özellikerin C++ derleyicisi ile çalışması için gerekli olan başlık ve kaynak kod dosyalarını üretecek olan dosyaları ve derleme için gereken Makefile dosyasını üretir. Makefile dosyası projeyi derleyecek olan "make" programının kullanacağı toplu derleme komut dosyasıdır.

Proje dizininde “make” komutu verilerek GCC nin Makefile dosyasındaki derleme yordamlarını çalıştırması sağlanır. Derleyicinin çıktısı olan çalıştırılabilir dosya proje dosyasının belirttiği isim ile belirttiği dizine oluşturulur. Proje dosyasının bir örneği Şekil 10'da verilmiştir.

```
TEMPLATE = app
TARGET = Medaq
HEADERS       =   src/mainwindow.h src/mdichild.h src/data_plot.h
SOURCES       =   src/main.cpp src/mainwindow.cpp src/mdichild.cpp src/data_plot.cpp
FORMS         =   src/main.ui src/info.ui src/devselect.ui
CONFIG -= build_all
CONFIG -= debug_and_release
CONFIG += qt thread release
INCLUDEPATH += C:\Qwt-5.1.1\include
LIBS += C:\Qwt-5.1.1\lib\qwt5.dll
LIBS += -lnidaq
DEFINES += QWT_DLL
RESOURCES     = rc/medaq.qrc
TRANSLATIONS = rc/medaq.ts
RC_FILE = rc/medaq.rc
MOC_DIR = tmp
OBJECTS_DIR = tmp/o
UI_DIR = tmp
UI_SOURCES_DIR = tmp
UI_HEADERS_DIR = tmp
RCC_DIR = tmp
DESTDIR = ../MEDAQ
```

Şekil 9 : Örnek Proje Dosyası.

Dosyada verilen parametrelerin bazılarının anlamları şu şekildedir.

- **TEMPLATE = app :** uygulama programı olduğunu belirtir. Şu değerleri alabilir; app, lib, subdirs.
- **TARGET = Medaq** : çalıştırılabilir dosya ismi.
- **HEADERS = ...** : başlık dosyaları.

- **SOURCES = ...** : kaynak kodu dosyaları.
- **FORMS = ...** : Designer dosyaları.
- **CONFIG -= build_all** : debug işleminin uygulanmaması için.
- **CONFIG -= debug_and_release** : debug işleminin uygulanmaması için.
- **CONFIG += qt thread release** : Qt kütüphanesinin kullanılacağını, çok kanallı programlamaya destek vereceğini ve dağıtım için çıktı üretmesi gerektiğini belirtir.
- **INCLUDEPATH = C:\Qwt-5.1.1\include** : Qwt için başlık dosyalarının yolunu belirtir.
- **LIBS += C:\Qwt-5.1.1\lib\qwt5.dll** : Qwt için kütüphane dosyalarının yolunu belirtir.
- **LIBS += -lnidaq** : NIDAQmx statik kütüphanesinin yüklenmesi gerektiğini belirtir.
- **DEFINES += QWT_DLL** : Qwt için C ön derleyicisine verilmek istenen QWT_DLL makrosunu belirtir.
- **RESOURCES = rc/medaq.qrc** : resim dosyalarını yöneten kaynak dosyasını belirtir.
- **TRANSLATIONS = rc/medaq.ts** : dil desteği için üretilecek dosyayı belirtir.
- **RC_FILE = rc/medaq.rc** : programın ikonunu yöneten kaynak dosyasını belirtir.
- **MOC_DIR = tmp** : geçici dosyaların konacağı dizini belirtir.
- **OBJECTS_DIR = tmp/o** : geçici dosyaların konacağı dizini belirtir.
- **UI_DIR = tmp** : geçici dosyaların konacağı dizini belirtir.
- **UI_SOURCES_DIR = tmp** : geçici dosyaların konacağı dizini belirtir.
- **UI_HEADERS_DIR = tmp** : geçici dosyaların konacağı dizini belirtir.
- **RCC_DIR = tmp** : geçici dosyaların konacağı dizini belirtir.
- **DESTDIR = ../MEDAQ** : çalıştırılabilir dosyanın kaydedileceği dizini belirtir.

2.10. Filtreler

Toplanan sinyal üstünde bazı filtreler uygulayarak sinyal analizi için grafik arabirimde kullanıcıya sunulacaktır. Bu amaçla oluşturulan filtreler şu şekildedir:

2.10.1 FFT

Fast Fourier Transform, yani Hızlı Fourier Dönüşümü anlamına gelen işlem zaman domain'inden alınan sinyalin frekans domain'ine dönüştürülmesinde kullanılır.

Hızlı kelimesi dönüşümün özel bir algoritma kullanılmasından gelir. Hızlı olmayan şekli, DFT – Discrete Fourier Transform, yani Ayrık Fourier Dönüşümüdür. Ayrık denmesinin

sebebi, dönüşüm işleminin ayrık değerler kümesi üstünde çalışmasından gelir. Dönüşümün sürekli olan orjinal hali ise şu şekildedir:

$$F(u)=\int_{-\infty}^{\infty} f(x)\,\mathrm{e}^{-i\,2\pi u\,x}\,dx$$

Dönüşüm kompleks domain'de çalışır. Kompleks kısmın sanal kısmı şu şekilde yazılabilir:

$$\mathrm{e}^{i\theta}=\cos(\theta)+i\sin(\theta)$$

Dönüşümün yaptığı işin özü, verilen bir sinyalin sinüz ve kosinüs sinyallerinin toplamı şeklinde ifadesine dönüştmesidir. Dönüşümün ayrık değerler kümesindeki gösterimi, yani DFT ise, şu şekilde gösterilir:

$$F_n=\sum_{k=0}^{N-1} f_k\,\mathrm{e}^{-i\frac{2\pi}{N}kn}$$

Buradaki f_k giriş fonksiyonu, F_n ise dönüşümün sonucudur. Bu işlem doğrudan basit bir algoritma ile gerçekleştirilirse, yapılacak işlemlerin karmaşıklığı $O(N^2)$ olacaktır. Eğer dönüşüm için FFT algoritmaları kullanılırsa karmaşıklık O(N Log N) mertebelerine indirgenebilmektedir. Bu indirgemeyi sağlayan temel etken, dönüşümün yinelenebilir (recursive) bir yapıda yazılarak çalıştırılabilmesidir.

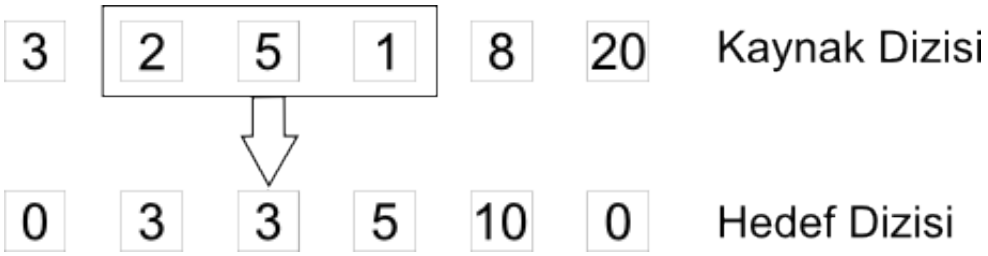
FFT dönüşümü için programda, güvenirliği kanıtlanmış, büyük veriler üstünde hızlı işlem yapabilen ve aynı zamanda açık kaynak olan FFTW kütüphanesi kullanılmıştır [FFTW].

FFTW, dönüşüm hızını arttırabilmek için işlemcinin ve bilgisayarın diğer bileşenlerinin mimari yapısını sınayarak ön bilgi toplar. Topladığı bilgilere göre en uygun algoritmaları seçer.

Kütüphanenin bu ilk aşamada çalışan test süreci uzun zaman alabildiğinden test sonucu oluşan ayarların saklanabilmesi sağlanmıştır. Bizim programımızda da bu özellikten faydalanarak yapılan ilk dönüşümde ayarlar saklanır ve program kapatılıncaya kadar bu ayarlar ile işlem yapılır. Böylece tekrar edilen dönüşümlerde test süresi çıkartılarak işlem hızlandırılmış olur.

2.10.2 Ortalama Filtresi

Bu filtre alçak frekansları geçiren ve yüksek frekanstaki sinyalleri temizleyen bir filtre çeşididir. Filtre şu şekilde çalışır:



Şekil 10 : Ortalama Filtresi

Şekilde görülen kaynak dizinin üzerinde gezen bir pencere içindeki sayıların ortalaması alınarak hedef dizide pencerenin ortasına rastlayan hüzreye yazılır. Hedefte kayıt yerinin bulunabilmesi için pencerenin tek sayılı bir büyüklüğünün olması gerektiği açıkça görülmektedir. Hedef dizinin başında ve sonunda bulunan, pencerenin yarısı kadar sayıdaki hücre sıfır ile doldurulur.

İşlemin doğrudan yapılan bir algoritma için karmaşıklığı O(W N) olacağı görülür. Buradaki W, pencerenin boyunu ve N de dizinin boyunu ifade eder.

Bu işlemin daha verimli bir şekli şu şekilde yapılabilir:

- Pencere büyüklüğü kadar hücre toplanır ve bir değişkende saklanır.
- Ortalama ilgili hücreye yazılır.
- Kaynak dizinin boyutu kadar bir döngü içinde
    - Pencereyi kaydırabilmek için pencereden bir sonraki hücreden, pencerenin şimdiki ilk hücresi çıkartılarak değişkende tutlan toplama eklenir ve ortalama alınır.
    - Ortalama ilgili hücreye yazılır.

Bu şekilde yapılan algoritmanın karmaşıklığı O(N) olacaktır. Buradaki N, kaynak dizinin boyutudur.

3. YÖNTEM

3.1. Bilgi Toplama

Projenin içeriğinin belirmesinden sonra ilk yaptığım iş projede kullanılacak olan araç, teknoloji ve yöntemler hakkında bilgi toplamak oldu. Veri toplama kartının kitaplığı, NIDAQmx sürücü yazılımı ile birlikte gelen NIDAQmx Yardım'ını okudum ve API hakkında bilgi edindim.

Grafik arabirim için kullanacağım programlama ortamını Qt olarak seçtikten sonra Qt ile ilgili Alan Ezust ve Paul Ezust'un yazmış oldukları "**Introduction to Design Patterns in C++ with Qt4**" kitabını okudum. Kendisi de açık bir kaynak olan kitabın içeriğine [QtBook] internet adresinden erişilebilmektedir.

Projeyi aldıktan sonra bilgi toplama sürecinde programı kullanacak olan kullanıcı ile görüşerek programın arabirim ve işleyişi konularında isteklerini ve önerilerini aldım.

3.2. Araçların Kurulumu

Uygulama geliştirme araç takımı olan Qt'yi ve Creator'ı yükledim. Qt ile birlikte MinGW, Unix program geliştirme ortamı ile birlikte gcc, make gibi derleme için gereken araçlar da kurulmuş oldu.

Teknik uygulamalar için geliştirilmiş olan Qwt nesne kütüphanesini derledim. Derlemeden sonra dinamik kütüphane ve başlık dosyaları oluşturuldu.

NI DAQ kartlarının sürücüsünü, MAX'ı, programlama örneklerini ve belgelendirmesini içeren NIDAQmx'in kurlumunu yaptım. Fiziksel aygıtım olmadığından MAX yardımı ile yapacağım testlerde kullanmak üzere simüle edilmiş sanal bir aygıt tanıttım.

NIDAQmx API'sine program içinden erişmek için gerekli olan NIDAQmx.lib statik kütüphane dosyasını gcc derleyicisinin desteklediği format olan libnidaq.a şekline getirmek için yukarıdaki Şekil 6'da verilen betiği kullandım.

Verilen betiğin çalışabilmesi için gerekli olan Microsoft Visual Studio ve MinGW-SYS-CORE yazılım paketlerini kurdum. Dönüşümü yaptıktan sonra elde ettiğim libnidaq.a statik kütüphanesini MinGW geliştirme ortamının kütüphane dizinine aktararak gcc ile erişilebilir hale getirdim. Program kodunun API'ye erişebilmesini sağlayacak olan NIDAQmx.h başlık dosyasını MinGW başlık dosyası dizinine aktardım.

3.3. Örneklerin Derlenmesi

Buraya kadar yaptığım hazırlığın çalışırlığını kontrol etmek için NIDAQmx ile birlikte gelen ANSI C örneklerini derleyerek önceden eklediğim sanal sürücüden veri almayı başardım. Aygıt sürücüsünün çalışırlığını doğruladıktan sonra Qt örneklerinden birkaçını derleyerek grafik arabirimin de çalışırlığını doğruladım.

Her iki ortamın birlikte kullanılarak sanal aygıttan okuduğum sonlu sayıdaki örnekleri bir QwtPlot nesnesi üzerinde grafik halinde çizdirdim.

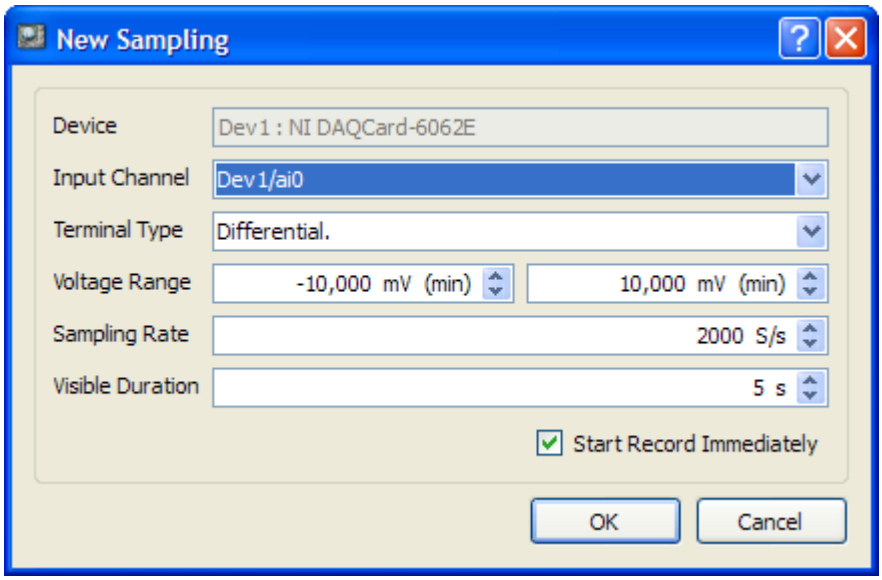
3.4. Programın Oluşturulması

Ana pencereyi Designer aracını kullanarak tasarladım.Designer aracıyla menüleri, araç çubukları ve kullanıcının istediği özellikleri içeren nesneleri düzenledim. Programda kullandığım ikonları GNOME [GNM] masaüstü ortamının ikon setinden aldım.

Ana pencerenin gui dosyasını proje dosyasına ekleyerek uic (user interface compiler)'nin grafik arabirim sınıfını oluşturulmasını sağladım. QMainWindow Sınıfından ve ana pencerenin isminden oluşturulan sınıftan MainWindow sınıfını türettim.

MainWindow sınıfının yapılandırıcısında menülerin olaylarını yöneten QAction sınıfından oluşturulmuş olan action'ların sinyallerini connect ile uygun fonksiyonlara bağladım.

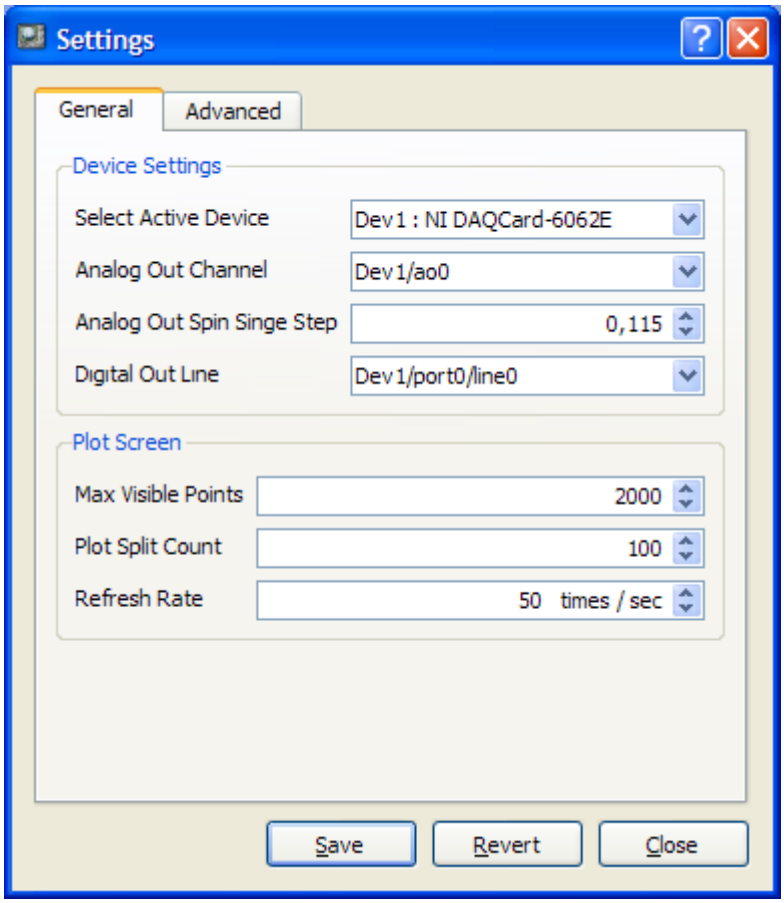
File > New menüsünde bulunan seçenek ile kullanıcının bilgisayarındaki DAQ aygıtının ve sinyal almak istediği kanalı belirleyebileceği, aynı zamanda ölçüm tipi ve gerilim aralığını belirleyebileceği bir diyalog penceresi tasarladım. Bu aygıt seçim diyaloğunun sınıfını yazarak aygıttan aldığım bilgilerle diyalog üzerindeki nesneleri doldurdum. Bu diyalog pencresini Şekil 11'de görebilirsiniz.



Şekil 11 : Yeni ölçüm diyalog penceresi.

Aygıt, kanal, ölçüm tipi ve gerilim aralığı gibi bilgileri kullanıcıdan aldıktan sonra NIDAQmx API'si ile bu bilgilere uygun, sürekli veri alan bir örnekleme görevi (*task*) oluşturdum. Okuduğum verileri QwtPlot nesnesi ile ekrana grafik olarak çizdirdim.

Kullanıcının kullanmak istediği aygıtı, çıkış kanal ve portlarını, ve programın diğer ayarlarını seçebileceği ayarlar diyalog penceresini tasarlayıp seçilenlerin ayar dosyasına kaydedilmesi için program kodunu yazdım.



Şekil 12 : Ayarlar Diyalog Penceresi

Analog ve dijital çıkışlar için ana pencerede oluşturduğum nesneler için NIDAQmx kütüphane fonksiyonlarının yardımıyla çıkış işlemlerini gerçekleştirdim.

Aygıtı yönetebilmek ve gelen veriyi uygun biçimde çizime aktarabilmek için aygıt yöneticisi sınıfını tasarladım. Bu sınıf ile :

- Yeni ölçüm başlatma,
- Başlatılan ölçümden verileri bir zamanlayıcı yardımıyla alıp çiziciye aktarma,
- Kaydetme işlemini başlatma, duraklatma, durdurma
- Ölçüm işlemini bitirip kaynakları temizleme
- Analog ve dijital çıkış işlemlerini gerçekleştirme,
- Ve hataları gösterme işlemlerini gerçekleştirir.

Arayüzdeki sınırlamalar için ve aygıt yöneticisinde kullanılmak üzere aygıt bilgisi toplayıcı sınıfı tasarladım. Bu sınıf ile:

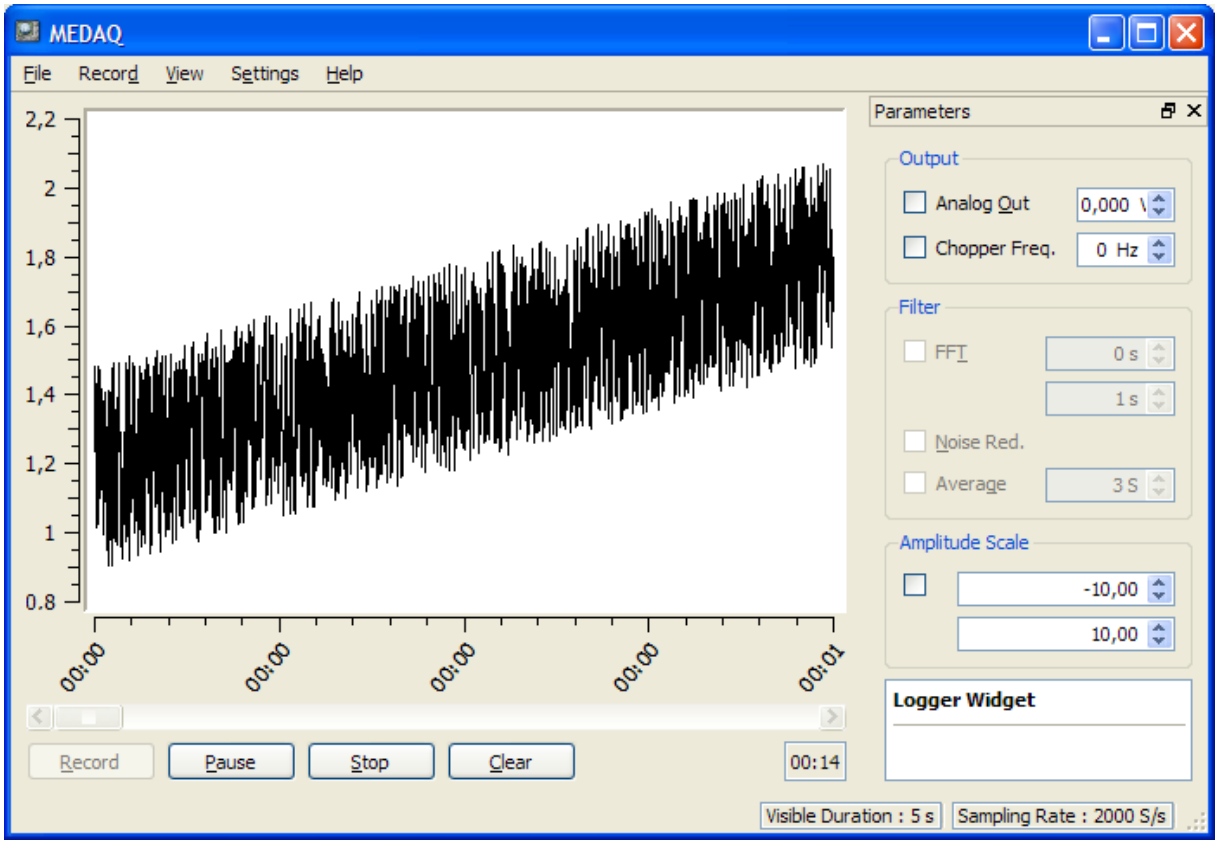
- Aygıt adını,
- Sanallık bilgisini,
- Ölçme hızı aralığını,
- Analog ve dijital kanalları

- Analog giriş için sonlandırma referans modlarını
- Ve giriş/çıkış gerilim aralığını sürücü kütüphanesinden almasını sağladım.

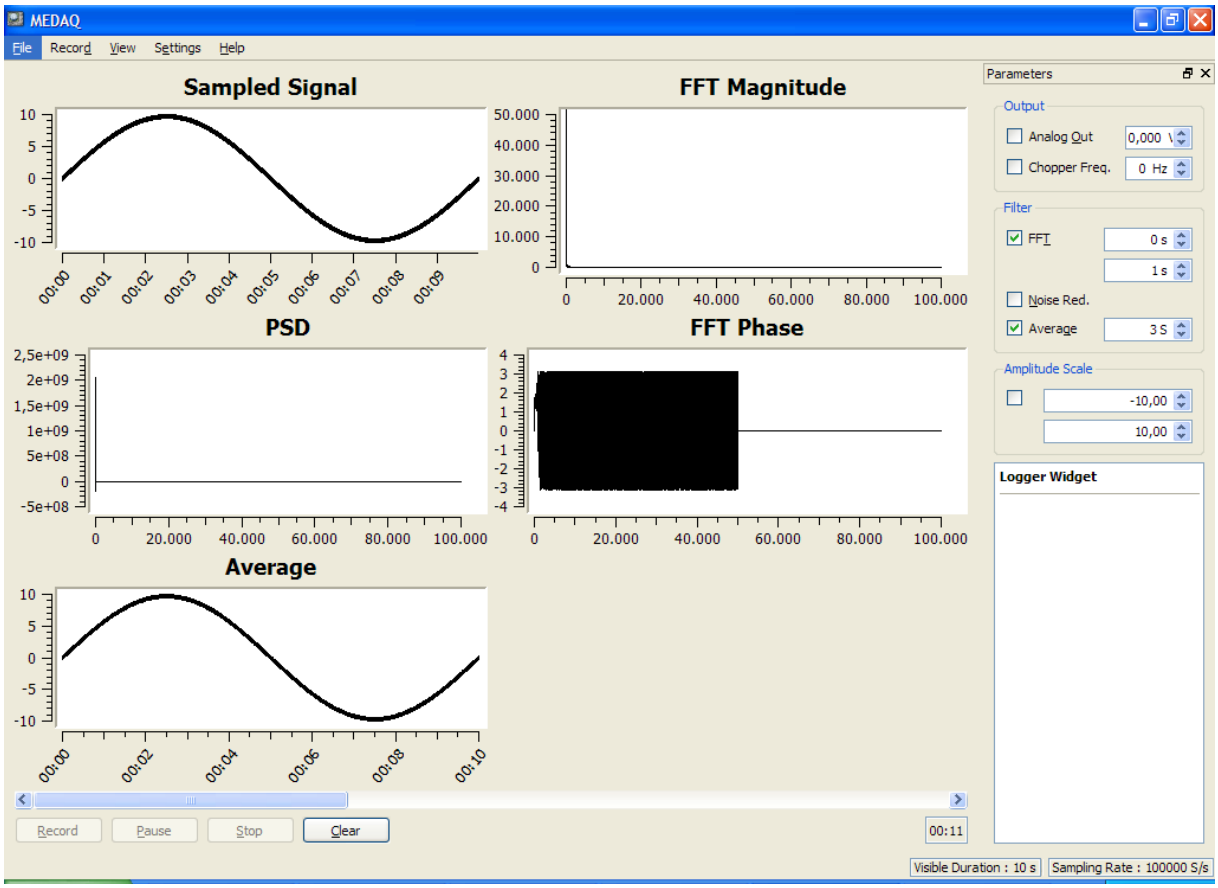
Bellekte topladığım veriyi diske kaydetmek ve sonradan yükleyebilmek için gerekli dosya formatını ve fonksiyonları gerçekleştirdim.

En son olarak da filtreleri daha önce açıkladığım şekilde gerçekleştirerek sonuçlarını gösterecek çizim nesnelerini programa ekledim.

Bu aşamaya kadar yaptığım çalışmaların sonucu olan programın ekran görüntüsü Şekil 12'de görebilirsiniz.



Şekil 13 : Medaq ana penceresi – Veri alma



Şekil 14 : Medaq ana penceresi – Filtreler

SONUÇ

Bu proje ile yüksek bütçeli programlar ile gerçekleştirilebilen, uygulamaya özgü veri toplama ve analiz programlarının açık kaynak araçlar ve kütüphaneler kullanarak da gerçekleştirilebileceğini kanıtlamış oldum.

KAYNAKLAR

[Ny] : http://en.wikipedia.org/wiki/Nyquist%E2%80%93Shannon_sampling_theorem

[NIDAQmx] : http://www.ni.com/dataacquisition/nidaqmx.htm

[6062E] : http://sine.ni.com/nips/cds/view/p/lang/en/nid/14632

[Qt] : http://trolltech.com/products

[MinGW] : http://www.mingw.org/

[Qwt] : http://qwt.sourceforge.net/

[NP] : http://notepad-plus.sourceforge.net/uk/site.htm

[QtBook] : http://cartan.cas.suffolk.edu/oopdocbook/opensource/

[GNM] : http://www.gnome.org/

[FFTW] : http://fftw.org/

EKLER

EK A - KAYNAK KODLARI

main.cpp

```
/*
   Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

   This file is part of Medaq.

   Medaq is free software: you can redistribute it and/or modify
   it under the terms of the GNU Lesser General Public License as
published by
   the Free Software Foundation, either version 3 of the License, or
   (at your option) any later version.

   Medaq is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
   GNU Lesser General Public License for more details.

   You should have received a copy of the GNU Lesser General Public
License
   along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#include <QApplication>
#include <QTranslator>
#include <QSharedMemory>
#include <QDebug>

#include "mainwindow.h"

int main(int argc, char *argv[])
{
       QSharedMemory sharedMemory("MEDAQSHAREDMEMORY");
//UniqueNameForApplication
       if (sharedMemory.create(1) && sharedMemory.error() !=
QSharedMemory::AlreadyExists)
             qDebug() << "This is the only instance";
       else
             exit(0);

   QApplication app(argc, argv);

   // lupdate *.pro ile pro da tanimlanan
   // kod dosyalarindan tr ile verilen
   // string ler cekilir.
   // lrelease *.pro ile ts ler qm ye
   // cevrilir.
   // qm asagidaki kod ile ayni
   // dizinden yuklenir.
   QTranslator translator;
   translator.load("medaq_tr");
   app.installTranslator(&translator);

   Q_INIT_RESOURCE(medaq);
   MainWindow mainWin;
   mainWin.show();
   return app.exec();
```

```
}
```

MainWindow.h

```
/*
   Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

   This file is part of Medaq.

   Medaq is free software: you can redistribute it and/or modify
   it under the terms of the GNU Lesser General Public License as
published by
   the Free Software Foundation, either version 3 of the License, or
   (at your option) any later version.

   Medaq is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
   GNU Lesser General Public License for more details.

   You should have received a copy of the GNU Lesser General Public
License
   along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include "ui_MainWindow.h"

class DeviceController;
class QwtPlotCurve;
class QwtPlot;

QT_BEGIN_NAMESPACE
class QAction;
class SettingsDialog;
class QLabel;
QT_END_NAMESPACE

class MainWindow : public QMainWindow, private Ui::MainWindow
{
    Q_OBJECT

public:
    MainWindow();

protected:
    void closeEvent(QCloseEvent *event);

private slots:

    void settingsDialog();
    void infoDialog();
    void aboutDialog();
    void helpDialog();
      void showStatusbar(bool);

      void handleStateChanges(int state);
      void handleLogMessages(QString messsage);
```

```
        void toggleManualScale();

        // out
        void toggleAnalogOutCheck();
        void toggleDigitalOutCheck();

        // filter
        void toggleFFTFilter();
        void updateFFTFilter();

        void toggleAverageFilter();
        void updateAverageFilter();
        void scrollAverageFilter(int step);

        void newSamplingDialog();
        void saveRecord();
        void openRecord();

        void fftMinSpinChanged(int);
        void fftMaxSpinChanged(int);

        void recordTimeChanged(int);

private:
    void connections();
    void readSettings();
    void writeSettings();

private:
    QAction * dockToggleViewAction;
      DeviceController * dc;
      QLabel * samplingRateLabel;
      QLabel * visibleDurationLabel;

      // plot
      QwtPlot * fftMagnitudePlot;
      QwtPlot * fftPhasePlot;
      QwtPlot * psdPlot;
      QwtPlot * averagePlot;
      QwtPlotCurve * fftMagnitudeCurve;
      QwtPlotCurve * fftPhaseCurve;
      QwtPlotCurve * psdCurve;
      QwtPlotCurve * averageCurve;

      // filter
      QString                    fftWisdom;
      QVector<double>         fftMagnitudeData;
      QVector<double>         fftPhaseData;
      QVector<double>         psdData;
      QVector<double>         fftBottomData;
      QVector<double>         averageData;
      QVector<double>         averageRecordData;
      QVector<double>         averageBottomData;

};
#endif
```

MainWindow.cpp

```
/*
   Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)
```

```
      This file is part of Medaq.

      Medaq is free software: you can redistribute it and/or modify
      it under the terms of the GNU Lesser General Public License as
published by
      the Free Software Foundation, either version 3 of the License, or
      (at your option) any later version.

      Medaq is distributed in the hope that it will be useful,
      but WITHOUT ANY WARRANTY; without even the implied warranty of
      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
      GNU Lesser General Public License for more details.

      You should have received a copy of the GNU Lesser General Public
License
      along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#include <QtGui>
#include <QStatusBar>
#include "MainWindow.h"
#include "InfoDialog.h"
#include "SettingsDialog.h"
#include "DeviceController.h"
#include "DeviceSensor.h"
#include "NewSamplingDialog.h"
#include <qwt_plot.h>
#include <qwt_plot_curve.h>
#include "fftw3.h"
#include <QVector>
#include "math.h"
#include "TimeScale.h"
#include <qwt_scale_widget.h>
#include <qwt_plot_picker.h>


MainWindow::MainWindow(){

    setupUi(this);

    connections();
    readSettings();

}

void MainWindow::connections(){

      menuView->insertAction(actionStatusBar, parametersDock-
>toggleViewAction());
      menuHelp->addAction(QWhatsThis::createAction());

      samplingRateLabel = new QLabel;
      samplingRateLabel->setFrameStyle(QFrame::StyledPanel |
QFrame::Sunken);
      statusBar()->insertPermanentWidget(0, samplingRateLabel);

      visibleDurationLabel = new QLabel;
      visibleDurationLabel->setFrameStyle(QFrame::StyledPanel |
QFrame::Sunken);
      statusBar()->insertPermanentWidget(0, visibleDurationLabel);
```

```
        dc = new DeviceController(plotter);

        // device controller
        connect(dc, SIGNAL(stateChanged(int)), this,
SLOT(handleStateChanges(int)));
        connect(dc, SIGNAL(logMessage(QString)), this,
SLOT(handleLogMessages(QString)));

        // button
        connect(recordButton, SIGNAL(clicked()), dc, SLOT(startRecording()));
        connect(pauseButton, SIGNAL(clicked()), dc, SLOT(pauseRecording()));
        connect(stopButton, SIGNAL(clicked()), dc, SLOT(stopRecording()));
        connect(clearButton, SIGNAL(clicked()), dc, SLOT(clearSampling()));
        connect(clearButton, SIGNAL(clicked()), samplingRateLabel,
SLOT(clear()));
        connect(clearButton, SIGNAL(clicked()), visibleDurationLabel,
SLOT(clear()));

        // action
        connect(actionExit, SIGNAL(triggered()), this, SLOT(close()));
        connect(actionStatusBar, SIGNAL(toggled(bool)), this,
SLOT(showStatusbar(bool)));
      connect(actionInfo, SIGNAL(triggered()), this, SLOT(infoDialog()));
      connect(actionSettings, SIGNAL(triggered()), this,
SLOT(settingsDialog()));
      connect(actionAbout, SIGNAL(triggered()), this, SLOT(aboutDialog()));
      connect(actionHelp, SIGNAL(triggered()), this, SLOT(helpDialog()));
        connect(actionStart_Record, SIGNAL(triggered()), dc,
SLOT(startRecording()));
        connect(actionPause, SIGNAL(triggered()), dc,
SLOT(pauseRecording()));
        connect(actionStop, SIGNAL(triggered()), dc, SLOT(stopRecording()));
        connect(actionClear, SIGNAL(triggered()), dc, SLOT(clearSampling()));
        connect(actionClear, SIGNAL(clicked()), samplingRateLabel,
SLOT(clear()));
        connect(actionClear, SIGNAL(clicked()), visibleDurationLabel,
SLOT(clear()));
        connect(actionNew, SIGNAL(triggered()), this,
SLOT(newSamplingDialog()));
        connect(actionSave, SIGNAL(triggered()), this, SLOT(saveRecord()));
        connect(actionOpen, SIGNAL(triggered()), this, SLOT(openRecord()));

        // scroll
        connect(timeScroll, SIGNAL(valueChanged(int)), plotter,
SLOT(scrollTime(int)));
        connect(timeScroll, SIGNAL(valueChanged(int)), this,
SLOT(scrollAverageFilter(int)));

        // scale
        connect(manualScaleCheck, SIGNAL(stateChanged(int)), this,
SLOT(toggleManualScale()));
        connect(minScaleSpin, SIGNAL(valueChanged(double)), this,
SLOT(toggleManualScale()));
        connect(maxScaleSpin, SIGNAL(valueChanged(double)), this,
SLOT(toggleManualScale()));

        // filter
        connect(fftCheck, SIGNAL(stateChanged(int)), this,
SLOT(toggleFFTFilter()));
        connect(fftMinSpin, SIGNAL(valueChanged(int)), this,
```

```
SLOT(fftMinSpinChanged(int)) );
      connect(fftMaxSpin, SIGNAL(valueChanged(int)), this,
SLOT(fftMaxSpinChanged(int)) );
      connect(fftMinSpin, SIGNAL(valueChanged(int)), this,
SLOT(updateFFTFilter()) );
      connect(fftMaxSpin, SIGNAL(valueChanged(int)), this,
SLOT(updateFFTFilter()) );
      connect(averageCheck, SIGNAL(stateChanged(int)), this,
SLOT(toggleAverageFilter()));
      connect(averageSpin, SIGNAL(valueChanged(int)), this,
SLOT(updateAverageFilter()) );

      // out
      connect(analogOutCheck, SIGNAL(stateChanged(int)), this,
SLOT(toggleAnalogOutCheck()));
      connect(analogOutSpin, SIGNAL(valueChanged(double)), dc,
SLOT(updateAnalogOut(double)));
      connect(digitalOutCheck, SIGNAL(stateChanged(int)), this,
SLOT(toggleDigitalOutCheck()));
      connect(digitalOutSpin, SIGNAL(valueChanged(int)), dc,
SLOT(updateDigitalOut(int)));

      // plotter
      connect(plotter, SIGNAL(recordTimeChanged(int)), this,
SLOT(recordTimeChanged(int)));



      QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
      settings.beginGroup("Settings");
      QString currentDevice = settings.value("currentDevice",
QString("")).toString();
      double analogOutStep = settings.value("analogOutStep",
0.1).toDouble();
      settings.endGroup();

      analogOutSpin->setSingleStep(analogOutStep);

      if( ! currentDevice.isEmpty())
      {
            analogOutSpin-
>setMinimum(DeviceSensor::getAOMinVoltage(currentDevice));
            analogOutSpin-
>setMaximum(DeviceSensor::getAOMaxVoltage(currentDevice));
      }
      else
      {
            QMessageBox::warning(this, tr("First Start"),
                  tr("<p><strong>There is no active device !</strong></p>"
                  "<p>It seems this is your first start or you did not
select any device at settings."
                  "If you do not select and save a device you cannot
acquire data from your device !</p>"
                  "<p>I will open settings, please select a device and "
                  "<strong><font size=\"+1\">save the
settings</font></strong>. "
                  "<br>If you change other settings you may <strong><font
size=\"+1\">restart</font></strong> the application.<br>"
                  "<font size=\"-1\"><strong>Hint :</strong> Wait tooltip
on controls or use \"what is this\" mode from title bar.</font></p>"),
```

```
                QMessageBox::Ok,QMessageBox::Ok);
            QTimer::singleShot(500, actionSettings, SLOT(trigger()));
      }

      if(dc->getState() != DeviceController::NoDevice)
            handleStateChanges(DeviceController::Waiting);
      else
            handleStateChanges(DeviceController::NoDevice);
}

void MainWindow::closeEvent(QCloseEvent *event){

      writeSettings();
      event->accept();

}

void MainWindow::settingsDialog(){

      SettingsDialog * sd = new SettingsDialog(this);
      if (sd->exec() == QDialog::Accepted)
      {
            QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
            settings.beginGroup("Settings");
            double analogOutStep = settings.value("analogOutStep",
0.1).toDouble();
            settings.endGroup();

            analogOutSpin->setSingleStep(analogOutStep);
      }
}

void MainWindow::infoDialog(){

    InfoDialog * id = new InfoDialog(this);
    id->show();

}

void MainWindow::aboutDialog(){

   QMessageBox::about(this, tr("About MEDAQ"),
            tr("<p>The <b>MEDAQ</b>, <b>M</b>edical <b>E</b>xperimentation
<b>D</b>ata <b>A</b>c<b>Q</b>uisition was created "
                        "by Huseyin Kozan for an experimentation which
collects data from an Neural Recording Board with a "
                        "National Instruments Data Acquisition Card at
Istanbul University.</p>"
                        "<p>Huseyin Kozan - <a
href='http://huseyinkozan.com.tr'>http://huseyinkozan.com.tr</a></p>"));
}

void MainWindow::helpDialog(){

    QMessageBox::warning(this, tr("Attention!"),tr("Not implemented,
yet !"));
}

void MainWindow::readSettings(){
```

```
    QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);

      settings.beginGroup("MainWindow");
    QPoint pos = settings.value("pos", QPoint(0, 0)).toPoint();
    QSize size = settings.value("size", QSize(640, 480)).toSize();
      QByteArray mainWindowState = settings.value("State").toByteArray();
      double minScale = settings.value("minScale", 0.0).toDouble();
      double maxScale = settings.value("maxScale", 0.0).toDouble();
      bool manualScale = settings.value("manualScale", false).toBool();
      double analogOut = settings.value("analogOut", 0.0).toDouble();
      int digitalOut = settings.value("digitalOut", 0.0).toInt();
      settings.endGroup();

    move(pos);
    resize(size);
      restoreState(mainWindowState);
      minScaleSpin->setValue(minScale);
      maxScaleSpin->setValue(maxScale);
      manualScaleCheck->setChecked(manualScale);
      analogOutSpin->setValue(analogOut);
      digitalOutSpin->setValue(digitalOut);
}

void MainWindow::writeSettings()
{
    QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);

      settings.beginGroup("MainWindow");
    settings.setValue("pos", pos());
    settings.setValue("size", size());
      settings.setValue("State", saveState());
      settings.setValue("minScale", minScaleSpin->value());
      settings.setValue("maxScale", maxScaleSpin->value());
      settings.setValue("manualScale", manualScaleCheck->isChecked());
      settings.setValue("analogOut", analogOutSpin->value());
      settings.setValue("digitalOut", digitalOutSpin->value());
      settings.endGroup();

}

void MainWindow::handleStateChanges(int state)
{
      switch(state)
      {
            case DeviceController::NoDevice:

                  actionNew->setEnabled(false);
                  actionOpen->setEnabled(true);
                  actionSave->setEnabled(false);

                  actionStart_Record->setEnabled(false);
                  actionPause->setEnabled(false);
                  actionStop->setEnabled(false);
                  actionClear->setEnabled(false);

                  recordButton->setEnabled(false);
                  pauseButton->setEnabled(false);
                  stopButton->setEnabled(false);
                  clearButton->setEnabled(false);
```

```
        timeScroll->setEnabled(false);

        fftCheck->setEnabled(false);
        noiseCheck->setEnabled(false);
        averageCheck->setEnabled(false);
        averageSpin->setEnabled(false);
        fftMinSpin->setEnabled(false);
        fftMaxSpin->setEnabled(false);

        break;
    case DeviceController::Waiting:

        actionNew->setEnabled(true);
        actionOpen->setEnabled(true);
        actionSave->setEnabled(false);

        actionStart_Record->setEnabled(false);
        actionPause->setEnabled(false);
        actionStop->setEnabled(false);
        actionClear->setEnabled(false);

        recordButton->setEnabled(false);
        pauseButton->setEnabled(false);
        stopButton->setEnabled(false);
        clearButton->setEnabled(false);
        timeScroll->setEnabled(false);

        fftCheck->setEnabled(false);
        noiseCheck->setEnabled(false);
        averageCheck->setEnabled(false);
        averageSpin->setEnabled(false);
        fftMinSpin->setEnabled(false);
        fftMaxSpin->setEnabled(false);

        plotter->clearPlotter();

        fftMagnitudeData.clear();
        fftPhaseData.clear();
        psdData.clear();
        fftBottomData.clear();
        averageData.clear();
        averageRecordData.clear();
        averageBottomData.clear();

        break;
    case DeviceController::Plotting:

        actionNew->setEnabled(true);
        actionOpen->setEnabled(true);
        actionSave->setEnabled(false);

        actionStart_Record->setEnabled(true);
        actionPause->setEnabled(false);
        actionStop->setEnabled(false);
        actionClear->setEnabled(true);

        recordButton->setEnabled(true);
        pauseButton->setEnabled(false);
        stopButton->setEnabled(false);
        clearButton->setEnabled(true);
        timeScroll->setEnabled(false);
```

```
        fftCheck->setEnabled(false);
        noiseCheck->setEnabled(false);
        averageCheck->setEnabled(false);
        averageSpin->setEnabled(false);
        fftMinSpin->setEnabled(false);
        fftMaxSpin->setEnabled(false);

        break;
    case DeviceController::Paused:

        actionNew->setEnabled(true);
        actionOpen->setEnabled(true);
        actionSave->setEnabled(false);

        actionStart_Record->setEnabled(true);
        actionPause->setEnabled(false);
        actionStop->setEnabled(true);
        actionClear->setEnabled(true);

        recordButton->setEnabled(true);
        pauseButton->setEnabled(false);
        stopButton->setEnabled(true);
        clearButton->setEnabled(true);
        timeScroll->setEnabled(false);

        fftCheck->setEnabled(false);
        noiseCheck->setEnabled(false);
        averageCheck->setEnabled(false);
        averageSpin->setEnabled(false);
        fftMinSpin->setEnabled(false);
        fftMaxSpin->setEnabled(false);

        break;
    case DeviceController::Recording:

        actionNew->setEnabled(true);
        actionOpen->setEnabled(true);
        actionSave->setEnabled(false);

        actionStart_Record->setEnabled(false);
        actionPause->setEnabled(true);
        actionStop->setEnabled(true);
        actionClear->setEnabled(true);

        recordButton->setEnabled(false);
        pauseButton->setEnabled(true);
        stopButton->setEnabled(true);
        clearButton->setEnabled(true);
        timeScroll->setEnabled(false);

        fftCheck->setChecked(false);
        noiseCheck->setChecked(false);
        averageCheck->setChecked(false);

        fftCheck->setEnabled(false);
        noiseCheck->setEnabled(false);
        averageCheck->setEnabled(false);
        averageSpin->setEnabled(false);
        fftMinSpin->setEnabled(false);
        fftMaxSpin->setEnabled(false);
```

```
                break;
            case DeviceController::Scrolling:

                actionNew->setEnabled(true);
                actionOpen->setEnabled(true);
                actionSave->setEnabled(true);

                actionStart_Record->setEnabled(false);
                actionPause->setEnabled(false);
                actionStop->setEnabled(false);
                actionClear->setEnabled(true);

                recordButton->setEnabled(false);
                pauseButton->setEnabled(false);
                stopButton->setEnabled(false);
                clearButton->setEnabled(true);
                timeScroll->setEnabled(true);

                fftCheck->setEnabled(true);
                noiseCheck->setEnabled(true);
                averageCheck->setChecked(false);
                averageCheck->setEnabled(true);
                averageSpin->setEnabled(true);
                fftMinSpin->setEnabled(true);
                fftMaxSpin->setEnabled(true);

                dc->stopSampling();

                plotter->prepareForScroll();
                timeScroll->setMaximum(plotter->getMaxScrollStep());

                plotter->scrollTime(0);

                fftMinSpin->setMinimum(0);
                fftMinSpin->setMaximum(0);
                fftMaxSpin->setMinimum(1);
                fftMaxSpin->setMaximum( qFloor( plotter-
>recordData.size() / plotter->getSamplingRate() ) );

                break;
        }
}

void MainWindow::handleLogMessages(QString message)
{
        loggerEdit->append(message);
}

void MainWindow::newSamplingDialog()
{
        NewSamplingDialog d(this);
        if( d.exec() == QDialog::Accepted )
        {
                dc->clearSampling();
                samplingRateLabel->clear();
                visibleDurationLabel->clear();
                QString channelName = d.channelName;
                int terminalConfiguration;
                if(d.terminalConfiguration == 0)
                        terminalConfiguration = DeviceSensor::AIReferenced;
```

```
            else if(d.terminalConfiguration == 1)
                  terminalConfiguration = DeviceSensor::AINonReferenced;
            else if(d.terminalConfiguration == 2)
                  terminalConfiguration = DeviceSensor::AIDifferential;
            else if(d.terminalConfiguration == 3)
                  terminalConfiguration =
DeviceSensor::AIPseudoDifferential;
            double minVoltage = d.minVoltage;
            double maxVoltage = d.maxVoltage;
            unsigned int visibleDuration = d.visibleDuration;
            unsigned int samplingRate = d.samplingRate;
            bool startRecordImmediately = d.startRecordImmediately;

            samplingRateLabel->setText(tr("Sampling Rate : %1
S/s").arg(samplingRate));
            visibleDurationLabel->setText(tr("Visible Duration : %1
s").arg(visibleDuration));

            dc->newSampling( channelName, terminalConfiguration,
                                          minVoltage, maxVoltage,
                                          visibleDuration, samplingRate,
startRecordImmediately);
      }
}

void MainWindow::toggleManualScale()
{
      if(manualScaleCheck->isChecked())
      {
            plotter->setScale(minScaleSpin->value(), maxScaleSpin-
>value());
      }
      else
      {
            plotter->setScale(0,0,true);
      }
}

void MainWindow::toggleAnalogOutCheck()
{
      QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
      settings.beginGroup("Settings");
      QString currentAOChannel = settings.value("currentAOChannel",
QString("")).toString();
      settings.endGroup();

      if( ! currentAOChannel.isEmpty())
      {
            if(analogOutCheck->isChecked())
            {
                  dc->startAnalogOut(analogOutSpin->value());
            }
            else
            {
                  dc->stopAnalogOut();
            }
      }
}
```

```
void MainWindow::toggleDigitalOutCheck()
{
	QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
	settings.beginGroup("Settings");
	QString currentDOLine = settings.value("currentDOLine",
QString("")).toString();
	settings.endGroup();

	if( ! currentDOLine.isEmpty())
	{
		if(digitalOutCheck->isChecked())
		{
			dc->startDigitalOut(digitalOutSpin->value());
		}
		else
		{
			dc->stopDigitalOut();
		}
	}
}

void MainWindow::saveRecord()
{
	QString saveName = QFileDialog::getSaveFileName(this,
							tr("Save File"), "",
tr("Medaq File (*.medaq)"));
	if( ! saveName.isNull()){
		QFile file(saveName);
		file.open(QIODevice::WriteOnly);
		QDataStream out(&file);

		quint32 visibleDuration = plotter->getVisibleDuration();
		quint32 samplingRate = plotter->getSamplingRate();

		QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
		settings.beginGroup("Settings");
		QString currentDevice = settings.value("currentDevice",
QString("")).toString();
		settings.endGroup();

		qreal minScale = DeviceSensor::getAIMinVoltage(currentDevice);
		qreal maxScale = DeviceSensor::getAIMaxVoltage(currentDevice);

		// Write a header with a "magic number" and a version
		out << (quint32) 0x11111111; // magic number, this file is
medaq file
		out << (quint32) 100; //1.00, internal version
		out << visibleDuration;
		out << samplingRate;
		out << minScale;
		out << maxScale;

		// Write the data
		for(int i=0; i<plotter->recordData.size(); ++i){
			out << plotter->recordData.at(i);
		}
	}
}
```

```
void MainWindow::openRecord()
{
	QString openName = QFileDialog::getOpenFileName(this,
                                                tr("Save File"), "", tr("Medaq
File (*.medaq)"));
	if( ! openName.isNull())
	{
		dc->clearSampling();
		samplingRateLabel->clear();
		visibleDurationLabel->clear();

		QFile file(openName);
		file.open(QIODevice::ReadOnly);
		QDataStream in(&file);

		quint32 visibleDuration;
		quint32 samplingRate;
		qreal minScale;
		qreal maxScale;

		// Read and check the header
		quint32 magic;
		in >> magic;
		if (magic != 0x11111111){
			loggerEdit->append( tr("The file that you trying to open
is not a Medaq file !") );
			return;
		}

		// Read the version
		quint32 version;
		in >> version;
		if (version < 100){
			loggerEdit->append( tr("The file that you trying to open
is an old file format !") );
			return;
		}
		if (version > 100){
			loggerEdit->append( tr("The file that you trying to open
is a newer file format !") );
			return;
		}

		in >> visibleDuration;
		in >> samplingRate;
		in >> minScale;
		in >> maxScale;

		samplingRateLabel->setText(tr("Sampling Rate : %1
S/s").arg(samplingRate));
		visibleDurationLabel->setText(tr("Visible Duration : %1
s").arg(visibleDuration));
		minScaleSpin->setMinimum(minScale);
		minScaleSpin->setMaximum(maxScale);
		maxScaleSpin->setMinimum(minScale);
		maxScaleSpin->setMaximum(maxScale);

		dc->clearSampling();

		unsigned int dataSize =  (     file.size() -
```

```
(4*sizeof(quint32)) - (2*sizeof(qreal))    )   /    sizeof(double);
            for(int i=0; i<dataSize; ++i)
            {
                  double x = 0;
                  in >> x;
                  plotter->recordData.append(x);
            }
            file.close();

            recordTimeChanged( dataSize / samplingRate );
            plotter->prepareForPlot(visibleDuration, samplingRate);
            plotter->prepareForScroll();
            timeScroll->setMaximum(plotter->getMaxScrollStep());
            dc->stopRecording();    // state = Scrolling

      }
}

void MainWindow::showStatusbar(bool value){
      if(value)
      {
            statusBar()->show();
      }
      else
      {
            statusBar()->hide();
      }
}

void MainWindow::toggleFFTFilter()
{
      if(fftCheck->isChecked())
      {
            fftMagnitudePlot = new QwtPlot(tr("FFT Magnitude"), this);
            fftPhasePlot = new QwtPlot(tr("FFT Phase"), this);
            psdPlot = new QwtPlot(tr("PSD"), this);
            fftMagnitudePlot->setCanvasBackground(Qt::white);
            fftPhasePlot->setCanvasBackground(Qt::white);
            psdPlot->setCanvasBackground(Qt::white);
            gridLayout->addWidget(fftMagnitudePlot, 0, 1);
            gridLayout->addWidget(fftPhasePlot, 1, 1);
            gridLayout->addWidget(psdPlot, 1, 0);
            fftMagnitudeCurve = new QwtPlotCurve(tr("FFT Magnitude
Curve"));
            fftPhaseCurve = new QwtPlotCurve(tr("FFT Phase Curve"));
            psdCurve = new QwtPlotCurve(tr("PSD Curve"));

            QwtPlotPicker * fftMagnitude_picker = new
QwtPlotPicker(QwtPlot::xBottom, QwtPlot::yLeft,
                  QwtPicker::PointSelection | QwtPicker::DragSelection,
                  QwtPlotPicker::CrossRubberBand, QwtPicker::AlwaysOn,
                  fftMagnitudePlot->canvas());
            fftMagnitude_picker->setRubberBandPen(QColor(Qt::green));
            fftMagnitude_picker->setRubberBand(QwtPicker::CrossRubberBand);
            fftMagnitude_picker->setTrackerPen(QColor(Qt::blue));

            QwtPlotPicker * fftPhase_picker = new
QwtPlotPicker(QwtPlot::xBottom, QwtPlot::yLeft,
                  QwtPicker::PointSelection | QwtPicker::DragSelection,
                  QwtPlotPicker::CrossRubberBand, QwtPicker::AlwaysOn,
                  fftPhasePlot->canvas());
```

```
            fftPhase_picker->setRubberBandPen(QColor(Qt::green));
            fftPhase_picker->setRubberBand(QwtPicker::CrossRubberBand);
            fftPhase_picker->setTrackerPen(QColor(Qt::blue));

            QwtPlotPicker * psd_picker = new
QwtPlotPicker(QwtPlot::xBottom, QwtPlot::yLeft,
                  QwtPicker::PointSelection | QwtPicker::DragSelection,
                  QwtPlotPicker::CrossRubberBand, QwtPicker::AlwaysOn,
                  psdPlot->canvas());
            psd_picker->setRubberBandPen(QColor(Qt::green));
            psd_picker->setRubberBand(QwtPicker::CrossRubberBand);
            psd_picker->setTrackerPen(QColor(Qt::blue));

            // call the update function
            updateFFTFilter();

            fftMagnitudeCurve->attach(fftMagnitudePlot);
            fftPhaseCurve->attach(fftPhasePlot);
            psdCurve->attach(psdPlot);

            fftMagnitudePlot->replot();
            fftPhasePlot->replot();
            psdPlot->replot();

            plotter->setTitle(tr("Sampled Signal"));
      }
      else
      {
            if(fftMagnitudePlot)
            {
                  delete fftMagnitudeCurve;
                  delete fftMagnitudePlot;
                  fftMagnitudeCurve = NULL;
                  fftMagnitudePlot = NULL;
            }
            if(fftPhasePlot)
            {
                  delete fftPhaseCurve;
                  delete fftPhasePlot;
                  fftPhaseCurve = NULL;
                  fftPhasePlot = NULL;
            }
            if(psdPlot)
            {
                  delete psdCurve;
                  delete psdPlot;
                  psdCurve = NULL;
                  psdPlot = NULL;
            }

            plotter->setTitle(tr(""));
      }
}

void MainWindow::updateFFTFilter()
{
      if(fftCheck->isChecked())
      {
            // calculate fft
            // get data from plotter record about the duration
            int start   = fftMinSpin->value() * plotter->getSamplingRate();
```

```
            int end           = fftMaxSpin->value() * plotter-
>getSamplingRate();

            if( start >= plotter->recordData.size() )
                  return;
            if( end > plotter->recordData.size() )
                  return;

            // do fft

            int size = end - start;

            double *in  = (double*) fftw_malloc( sizeof(double) * size );
            fftw_complex *out = (fftw_complex*) fftw_malloc(
sizeof(fftw_complex) * size );
            fftw_plan p = fftw_plan_dft_r2c_1d( size, in, out,
FFTW_ESTIMATE );

            if(fftWisdom.isEmpty())
            {
                  char * s = fftw_export_wisdom_to_string();
                  fftWisdom = QString::fromLatin1(s);
            }
            else
            {

      fftw_import_wisdom_from_string(fftWisdom.toLatin1().data());
            }

            if( p!=NULL && in!=NULL && out!=NULL )
            {
                  statusBar()->showMessage(tr("Deriving the FFT
data ..."));

                  memcpy(&(in[0]), &(plotter->recordData.data()[start]),
size * sizeof(double));             // take data from plot screen

                  fftw_execute(p);

                  fftMagnitudeData.fill(0.0, size);
                  fftPhaseData.fill(0.0, size);
                  psdData.fill(0.0, size);
                  fftBottomData.resize(size);

                  for(int i=0; i<size; ++i)
                  {
                        register int xr = out[i][0];
                        register int xi = out[i][1];
                        fftMagnitudeData[i] = qSqrt( xr*xr + xi*xi );
                        fftPhaseData[i] = atan2( (double) xi,(double) xr );
                        psdData[i] = xr*xr + xi*xi ;
                        fftBottomData[i] = i;
                  }

                  statusBar()->showMessage(tr(""));

            }
            else
            {
                  statusBar()->showMessage(tr("Could not derive fft"));
                  qDebug() << "cannot create plan";
```

```
            }

            fftw_destroy_plan(p);
            fftw_free(in); fftw_free(out);

            // set datas
            if(fftMagnitudeCurve)
                  fftMagnitudeCurve->setRawData( fftBottomData.data(),
fftMagnitudeData.data(), size );
            if(fftPhaseCurve)
                  fftPhaseCurve->setRawData( fftBottomData.data(),
fftPhaseData.data(), size );
            if(psdCurve)
                  psdCurve->setRawData( fftBottomData.data(),
psdData.data(), size );

            fftMagnitudePlot->replot();
            fftPhasePlot->replot();
            psdPlot->replot();
      }
}

void MainWindow::fftMinSpinChanged(int val)
{
      fftMaxSpin->setMinimum(val+1);
}

void MainWindow::fftMaxSpinChanged(int val)
{
      fftMinSpin->setMaximum(val-1);
}

void MainWindow::recordTimeChanged(int sec)
{
      int m = sec / 60;
      int s = sec % 60;
      QString s_m;
      QString s_s;
      QString z("0");

      if(m<10)
            s_m = z + QString::number(m);
      else
            s_m = QString::number(m);

      if(s<10)
            s_s = z + QString::number(s);
      else
            s_s = QString::number(s);

      recordTimeLabel->setText(s_m + QString(":") + s_s);
}

void MainWindow::toggleAverageFilter()
{
      if(averageCheck->isChecked())
      {
            averagePlot = new QwtPlot(tr("Average"), this);
            averagePlot->setCanvasBackground(Qt::white);
            gridLayout->addWidget(averagePlot, 2, 0);
            averageCurve = new QwtPlotCurve(tr("Average Curve"));
```

```
            QwtPlotPicker * average_picker = new
QwtPlotPicker(QwtPlot::xBottom, QwtPlot::yLeft,
                  QwtPicker::PointSelection | QwtPicker::DragSelection,
                  QwtPlotPicker::CrossRubberBand, QwtPicker::AlwaysOn,
                  averagePlot->canvas());
            average_picker->setRubberBandPen(QColor(Qt::green));
            average_picker->setRubberBand(QwtPicker::CrossRubberBand);
            average_picker->setTrackerPen(QColor(Qt::blue));

            int size = plotter->getSamplingRate() * plotter-
>getVisibleDuration();
            averageData.fill(0.0, size);
            averageBottomData.resize(size);
            for( int i = 0 ; i < size ; ++i )
            {
                  averageBottomData[i] = i;
            }

            // call the update function
            updateAverageFilter();

            if(averageCurve)
                  averageCurve->setRawData( averageBottomData.data(),
averageData.data(), averageBottomData.size() );

            averageCurve->attach(averagePlot);

            // time scale
            TimeScaleDraw * tsd = new TimeScaleDraw(QTime(0,0), plotter-
>getSamplingRate());
            averagePlot->setAxisScaleDraw(QwtPlot::xBottom, tsd );
            averagePlot->setAxisScale(QwtPlot::xBottom, 0, size);
            averagePlot->setAxisLabelRotation(QwtPlot::xBottom, -50.0);
            averagePlot->setAxisLabelAlignment(QwtPlot::xBottom,
Qt::AlignLeft | Qt::AlignBottom);
            // To avoid this "jumping canvas" effect, we add a permanent
margin.
            QwtScaleWidget *scaleWidget = averagePlot-
>axisWidget(QwtPlot::xBottom);
            const int fmh = QFontMetrics(scaleWidget->font()).height();
            scaleWidget->setMinBorderDist(0, fmh / 2);

            averagePlot->replot();

            plotter->setTitle(tr("Sampled Signal"));
      }
      else
      {
            if(averagePlot)
            {
                  delete averageCurve;
                  delete averagePlot;
                  averageCurve = NULL;
                  averagePlot = NULL;
            }

            plotter->setTitle(tr(""));
      }
}
```

```
void MainWindow::updateAverageFilter()
{
	if(averageCheck->isChecked())
	{
		int winSize = averageSpin->value();

		if( winSize < 3 || winSize % 2 == 0 || winSize > plotter-
>recordData.size() )
			return;

		int recordSize = plotter->recordData.size();
		int half = (winSize - 1) / 2;
		averageRecordData.fill ( 0.0, recordSize );
		double * r = plotter->recordData.data();
		double * a = averageRecordData.data();

		// data without boundary spaces
		for(int i = half; i < recordSize - half; ++i)
		{
			double total = 0;

			for(int j = -half; j <= half; ++j)
			{
				total += r[i+j];
			}

			a[i] = total / winSize;
		}

		scrollAverageFilter(timeScroll->value());

		averagePlot->replot();
	}
}


void MainWindow::scrollAverageFilter(int step)
{
	if( ! averageCheck->isChecked())
		return;
	if ( step > plotter->getMaxScrollStep() )
		return;

	int size = plotter->getSamplingRate() * plotter-
>getVisibleDuration();
	if( size > plotter->recordData.size())
	{
		// copy recorddata and return
		memcpy(&(averageData.data()[0]), &(averageRecordData.data()
[0]), averageRecordData.size() * sizeof(double));
		averagePlot->replot();
		return;
	}

	QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
	settings.beginGroup("Settings");
		int plotSplitCount                              =
settings.value("plotSplitCount", 50).toInt();
	settings.endGroup();
```

```
      int sourceIndex = step*(size/plotSplitCount);
      memcpy(&(averageData.data()[0]), &(averageRecordData.data()
[sourceIndex]), averageData.size() * sizeof(double));


      unsigned int plotSize = averageBottomData.size();
      double * d = averageBottomData.data();
      int samplingRate = plotter->getSamplingRate();
      QwtValueList vlist_y[3];
      for ( unsigned int i = 0 ; i < plotSize ; ++i )
      {
            register int x = sourceIndex + i;
            d[i] = x;
            if( 0 == x % samplingRate )
                  vlist_y[0] << d[i];
      }
      averagePlot->setAxisScale(QwtPlot::xBottom, averageBottomData[0],
averageBottomData[plotSize - 1]);
      QwtScaleDiv scdiv_y(averageBottomData.at(0),
averageBottomData.at(averageBottomData.size()-1), vlist_y);
      scdiv_y.setTicks(QwtScaleDiv::MajorTick, vlist_y[0]);
      averagePlot->setAxisScaleDiv(QwtPlot::xBottom, scdiv_y);

      averagePlot->replot();
}
```

NewSamplingDialog.h

```
/*
      Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

      This file is part of Medaq.

      Medaq is free software: you can redistribute it and/or modify
      it under the terms of the GNU Lesser General Public License as
published by
      the Free Software Foundation, either version 3 of the License, or
      (at your option) any later version.

      Medaq is distributed in the hope that it will be useful,
      but WITHOUT ANY WARRANTY; without even the implied warranty of
      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
      GNU Lesser General Public License for more details.

      You should have received a copy of the GNU Lesser General Public
License
      along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef DEVSELECT_H
#define DEVSELECT_H

#include <QDialog>
#include <QString>
#include "ui_NewSamplingDialog.h"

class NewSamplingDialog : public QDialog, private Ui::NewSamplingDialog
{
    Q_OBJECT
```

```
public:
    NewSamplingDialog(QWidget * parent = 0);
    ~NewSamplingDialog()
      {
      }

      QString channelName;
      int terminalConfiguration;
      double minVoltage;
      double maxVoltage;
      unsigned int visibleDuration;
      unsigned int samplingRate;
      bool startRecordImmediately;

private slots:
      void saveValues();

private:
      QStringList channelList;
      QStringList terminalList;

};
#endif
```

NewSamplingDialog.cpp

```
/*
   Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

   This file is part of Medaq.

   Medaq is free software: you can redistribute it and/or modify
   it under the terms of the GNU Lesser General Public License as
published by
   the Free Software Foundation, either version 3 of the License, or
   (at your option) any later version.

   Medaq is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
   GNU Lesser General Public License for more details.

   You should have received a copy of the GNU Lesser General Public
License
   along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#include <QtGui>
#include <QSettings>
#include "NewSamplingDialog.h"
#include "DeviceSensor.h"

NewSamplingDialog::NewSamplingDialog(QWidget * parent) :
      QDialog(parent)
      //channelName(""),
      //terminalConfiguration(0),
      //minVoltage(0),
      //maxVoltage(0),
      //visibleDuration(0),
      //samplingRate(0),
      //startRecordImmediately(false)
```

```
{
    /*setAttribute(Qt::WA_DeleteOnClose);*/
    setupUi(this);

      QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
      settings.beginGroup("NewSamplingDialog");
      channelName = settings.value("channelName", QString()).toString();
    terminalConfiguration = settings.value("terminalConfiguration",
0).toInt();
      minVoltage = settings.value("minVoltage", 0.0).toDouble();
      maxVoltage = settings.value("maxVoltage", 0.0).toDouble();
      visibleDuration = settings.value("visibleDuration", 0).toUInt();
      samplingRate = settings.value("samplingRate", 0).toUInt();
      startRecordImmediately = settings.value("startRecordImmediately",
0).toBool();
      settings.endGroup();

      settings.beginGroup("Settings");
      QString currentDevice = settings.value("currentDevice",
QString("")).toString();
      settings.endGroup();

      deviceEdit->setText(currentDevice + " : " +
DeviceSensor::getLongDeviceName(currentDevice));

      minSpin->setMinimum(DeviceSensor::getAIMinVoltage(currentDevice));
      minSpin->setMaximum(DeviceSensor::getAIMaxVoltage(currentDevice));
      maxSpin->setMinimum(DeviceSensor::getAIMinVoltage(currentDevice));
      maxSpin->setMaximum(DeviceSensor::getAIMaxVoltage(currentDevice));

      rateSpin-
>setMinimum(DeviceSensor::getAIMinSampleRate(currentDevice));
      rateSpin-
>setMaximum(DeviceSensor::getAIMaxSampleRate(currentDevice));

      // fill channel list and select saved
      channelList = DeviceSensor::getAIChannelNames(currentDevice);
      foreach(QString chan, channelList){
            channelCombo->addItem(chan);
    }
      int count = -1;
      int index = -1;
      foreach(QString chan, channelList){
            count++;
            if(channelName == chan)
                  index = count;
    }
      if(index != -1){
            channelCombo->setCurrentIndex(index);
      }

      // fill terminalconf list and select saved
      terminalList =
DeviceSensor::getAITerminalConfigurations(currentDevice);
      foreach(QString term, terminalList){
            terminalCombo->addItem(term);
    }
      terminalCombo->setCurrentIndex(terminalConfiguration);
```

```
        minSpin->setValue(minVoltage);
        maxSpin->setValue(maxVoltage);
        rateSpin->setValue(samplingRate);
        visibleSpin->setValue(visibleDuration);
        startRecordCheck->setChecked(startRecordImmediately);

        connect(buttonBox, SIGNAL(accepted()), this, SLOT(saveValues()));

}

void NewSamplingDialog::saveValues()
{
        channelName = channelList.at(channelCombo->currentIndex());
        terminalConfiguration = terminalCombo->currentIndex();
        minVoltage = minSpin->value();
        maxVoltage = maxSpin->value();
        visibleDuration = visibleSpin->value();
        samplingRate = rateSpin->value();
        startRecordImmediately = startRecordCheck->isChecked();

        QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
        settings.beginGroup("NewSamplingDialog");
        settings.setValue("channelName", channelName);
        settings.setValue("terminalConfiguration", terminalConfiguration);
        settings.setValue("minVoltage", minVoltage);
        settings.setValue("maxVoltage", maxVoltage);
        settings.setValue("visibleDuration", visibleDuration);
        settings.setValue("samplingRate", samplingRate);
        settings.setValue("startRecordImmediately", startRecordImmediately);
        settings.endGroup();
}
```

InfoDialog.h

```
    /*
        Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

        This file is part of Medaq.

        Medaq is free software: you can redistribute it and/or modify
        it under the terms of the GNU Lesser General Public License as
published by
        the Free Software Foundation, either version 3 of the License, or
        (at your option) any later version.

        Medaq is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
        GNU Lesser General Public License for more details.

        You should have received a copy of the GNU Lesser General Public
License
        along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef INFODIALOG_H
#define INFODIALOG_H

#include <QDialog>
#include <QWidget>
```

```
#include "ui_InfoDialog.h"

class InfoDialog : public QDialog, private Ui::InfoDialog
{
    Q_OBJECT

public:
    InfoDialog(QWidget * parent = 0);

private slots:
      void deviceComboChanged(int);

private:
      QStringList deviceList;
};
#endif
```

InfoDialog.cpp

```
   /*
      Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

      This file is part of Medaq.

      Medaq is free software: you can redistribute it and/or modify
      it under the terms of the GNU Lesser General Public License as
published by
      the Free Software Foundation, either version 3 of the License, or
      (at your option) any later version.

      Medaq is distributed in the hope that it will be useful,
      but WITHOUT ANY WARRANTY; without even the implied warranty of
      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
      GNU Lesser General Public License for more details.

      You should have received a copy of the GNU Lesser General Public
License
      along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#include <QtGui>
#include <QString>
#include <QStringList>
#include "InfoDialog.h"
#include "DeviceSensor.h"

InfoDialog::InfoDialog(QWidget * parent) : QDialog(parent)
{
    setAttribute(Qt::WA_DeleteOnClose);
    setupUi(this);

      deviceCombo->clear();
      deviceList = DeviceSensor::getDeviceNames();

      foreach(QString dev, deviceList){
            deviceCombo->addItem(DeviceSensor::getLongDeviceName(dev) + " :
" + dev);
    }

      connect(deviceCombo,SIGNAL(currentIndexChanged(int)), this,
SLOT(deviceComboChanged(int)));
```

```
	deviceComboChanged(0);
}

void InfoDialog::deviceComboChanged(int index){
	QString text;
	QStringList strList;

	QString deviceName =
DeviceSensor::getLongDeviceName(deviceList.at(index));

	text += "<h2>" + deviceName + " : " + deviceList.at(index) + "</h2>";
	if(DeviceSensor::isSimulated(deviceList.at(index))){
		text += tr("<p><strong>Simulated Device</strong><br>Note that;
some values may not consistent !</p>") + "<br>";
	}
	text += tr("<strong>Sample Rates :</strong>") + "<br>";
	text += tr("Analog Input Min : ");
	text +=
QString::number(DeviceSensor::getAIMinSampleRate(deviceList.at(index))) +
"<br>";
	text += tr("Analog Input Max : ");
	text +=
QString::number(DeviceSensor::getAIMaxSampleRate(deviceList.at(index))) +
"<br>";
	text += tr("Analog Output Min : ");
	text +=
QString::number(DeviceSensor::getAOMinSampleRate(deviceList.at(index))) +
"<br>";
	text += tr("Analog Output Max : ");
	text +=
QString::number(DeviceSensor::getAOMaxSampleRate(deviceList.at(index))) +
"<br>";
	text += tr("Digital Output Max : ");
	text +=
QString::number(DeviceSensor::getDOMaxRate(deviceList.at(index))) + "<br>";

	text += tr("<strong>Voltage Ranges :</strong>") + "<br>";
	text += tr("Analog Input Min : ");
	text +=
QString::number(DeviceSensor::getAIMinVoltage(deviceList.at(index))) +
"<br>";
	text += tr("Analog Input Max : ");
	text +=
QString::number(DeviceSensor::getAIMaxVoltage(deviceList.at(index))) +
"<br>";
	text += tr("Analog Output Min : ");
	text +=
QString::number(DeviceSensor::getAOMinVoltage(deviceList.at(index))) +
"<br>";
	text += tr("Analog Output Max : ");
	text +=
QString::number(DeviceSensor::getAOMaxVoltage(deviceList.at(index))) +
"<br>";

	//text += tr("<strong>Terminal Configurations :</strong>") + "<br>";
	//strList =
DeviceSensor::getAITerminalConfigurations(deviceList.at(index));
	//foreach(QString str, strList){
	//    text += str + "<br>";
	//}
```

```
	strList.clear();
	text += tr("<strong>Analog Input Channels :</strong>") + "<br>";
	strList = DeviceSensor::getAIChannelNames(deviceList.at(index));
	foreach(QString str, strList){
		text += str + "<br>";
	}
	strList.clear();
	text += tr("<strong>Analog Output Channels :</strong>") + "<br>";
	strList = DeviceSensor::getAOChannelNames(deviceList.at(index));
	foreach(QString str, strList){
		text += str + "<br>";
	}
	strList.clear();
	text += tr("<strong>Digital Output Lines :</strong>") + "<br>";
	strList = DeviceSensor::getDOLines(deviceList.at(index));
	foreach(QString str, strList){
		text += str + "<br>";
	}

	infoEdit->setText(text);
}
```

SettingsDialog.h

```
/*
	Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

	This file is part of Medaq.

	Medaq is free software: you can redistribute it and/or modify
	it under the terms of the GNU Lesser General Public License as
published by
	the Free Software Foundation, either version 3 of the License, or
	(at your option) any later version.

	Medaq is distributed in the hope that it will be useful,
	but WITHOUT ANY WARRANTY; without even the implied warranty of
	MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
	GNU Lesser General Public License for more details.

	You should have received a copy of the GNU Lesser General Public
License
	along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef SETTINGSDIALOG_H
#define SETTINGSDIALOG_H

#include <QDialog>
#include <QWidget>
#include "ui_SettingsDialog.h"

class QString;
class QStringList;

class SettingsDialog : public QDialog, private Ui::SettingsDialog
{
    Q_OBJECT

public:
    SettingsDialog(QWidget * parent = 0);
```

```
public slots:
      void deviceComboChanged(int);
      void analogOutComboChanged(int);
      void digitalOutComboChanged(int);
      void saveButtonClicked();
      void revertButtonClicked();

public:
      QString selectedDevice;
      QString selectedAOChannel;
      QString selectedDOLine;

private:
      QStringList deviceList;
      QStringList channelList;
      QStringList lineList;
};

#endif
```

SettingsDialog.cpp

```
   /*
      Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

      This file is part of Medaq.

      Medaq is free software: you can redistribute it and/or modify
      it under the terms of the GNU Lesser General Public License as
published by
      the Free Software Foundation, either version 3 of the License, or
      (at your option) any later version.

      Medaq is distributed in the hope that it will be useful,
      but WITHOUT ANY WARRANTY; without even the implied warranty of
      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
      GNU Lesser General Public License for more details.

      You should have received a copy of the GNU Lesser General Public
License
      along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#include <QtGui>
#include <QString>
#include <QStringList>
#include "SettingsDialog.h"
#include "DeviceSensor.h"

SettingsDialog::SettingsDialog(QWidget * parent) : QDialog(parent)
{
    setAttribute(Qt::WA_DeleteOnClose);
    setupUi(this);

      deviceCombo->clear();
      deviceList = DeviceSensor::getDeviceNames();

      foreach(QString dev, deviceList){
            deviceCombo->addItem(dev + " : " +
DeviceSensor::getLongDeviceName(dev));
```

```
    }

      connect(deviceCombo,SIGNAL(currentIndexChanged(int)), this,
SLOT(deviceComboChanged(int)));
      connect(analogOutCombo,SIGNAL(currentIndexChanged(int)), this,
SLOT(analogOutComboChanged(int)));
      connect(digitalOutCombo,SIGNAL(currentIndexChanged(int)), this,
SLOT(digitalOutComboChanged(int)));
      connect(saveButton,SIGNAL(clicked()), this,
SLOT(saveButtonClicked()));
      connect(revertButton,SIGNAL(clicked()), this,
SLOT(revertButtonClicked()));
      connect(closeButton,SIGNAL(clicked()), this, SLOT(close()));

      deviceComboChanged(0);
      revertButtonClicked();

}

void SettingsDialog::deviceComboChanged(int index){
      if(index < deviceList.size())
      {
            selectedDevice = deviceList.at(index);

            if( ! selectedDevice.isEmpty())
            {
                  analogOutCombo->clear();
                  channelList =
DeviceSensor::getAOChannelNames(selectedDevice);
                  foreach(QString chan, channelList){
                        analogOutCombo->addItem(chan);
                  }

                  digitalOutCombo->clear();
                  lineList = DeviceSensor::getDOLines(selectedDevice);
                  foreach(QString line, lineList){
                        digitalOutCombo->addItem(line);
                  }
            }
      }
}


void SettingsDialog::analogOutComboChanged(int index){
      if(index < channelList.size() && index >= 0 )
            selectedAOChannel = channelList.at(index);
}


void SettingsDialog::digitalOutComboChanged(int index){
      if(index < lineList.size() && index >= 0 )
            selectedDOLine = lineList.at(index);
}


void SettingsDialog::saveButtonClicked(){

      QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
      settings.beginGroup("Settings");
      settings.setValue("currentDevice", deviceList.at(deviceCombo-
```

```
>currentIndex()));
	settings.setValue("currentAOChannel", channelList.at(analogOutCombo-
>currentIndex()));
	settings.setValue("currentDOLine", lineList.at(digitalOutCombo-
>currentIndex()));
	settings.setValue("maxVisiblePoint", maxVisiblePointSpin->value());
	settings.setValue("plotSplitCount", plotSplitCountSpin->value());
	settings.setValue("fifoSize", fifoSizeSpin->value());
	settings.setValue("refreshRate", refreshRateSpin->value());
	settings.setValue("analogOutStep", analogOutStepSpin->value());
	settings.endGroup();

}

void SettingsDialog::revertButtonClicked(){

	QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
	settings.beginGroup("Settings");
	QString currentDevice = settings.value("currentDevice",
QString("")).toString();
	QString currentAOChannel = settings.value("currentAOChannel",
QString("")).toString();
	QString currentDOLine = settings.value("currentDOLine",
QString("")).toString();
	unsigned int maxVisiblePoint = settings.value("maxVisiblePoint",
1000).toUInt();
	unsigned int plotSplitCount = settings.value("plotSplitCount",
50).toUInt();
	unsigned int fifoSize = settings.value("fifoSize", 1000000).toUInt();
	unsigned int refreshRate = settings.value("refreshRate",
20).toUInt();
	double analogOutStep = settings.value("analogOutStep",
0.1).toDouble();
	settings.endGroup();

	// for device combo
	int count = -1;
	int index = -1;
	foreach(QString dev, deviceList){
		count++;
		if(currentDevice == dev)
			index = count;
    }
	if(index != -1){
		deviceCombo->setCurrentIndex(index);
	}

	// for channel combo
	count = -1;
	index = -1;
	foreach(QString chan, channelList){
		count++;
		if(currentAOChannel == chan)
			index = count;
    }
	if(index != -1){
		analogOutCombo->setCurrentIndex(index);
	}

	// for line combo
```

```
      count = -1;
      index = -1;
      foreach(QString line, lineList){
            count++;
            if(currentDOLine == line)
                  index = count;
    }
      if(index != -1){
            digitalOutCombo->setCurrentIndex(index);
      }

      maxVisiblePointSpin->setValue(maxVisiblePoint);
      plotSplitCountSpin->setValue(plotSplitCount);
      fifoSizeSpin->setValue(fifoSize);
      refreshRateSpin->setValue(refreshRate);
      analogOutStepSpin->setValue(analogOutStep);

}
```

DeviceSensor.h

```
/*
   Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

   This file is part of Medaq.

   Medaq is free software: you can redistribute it and/or modify
   it under the terms of the GNU Lesser General Public License as
published by
   the Free Software Foundation, either version 3 of the License, or
   (at your option) any later version.

   Medaq is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
   GNU Lesser General Public License for more details.

   You should have received a copy of the GNU Lesser General Public
License
   along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef DEVICESENSOR_H
#define DEVICESENSOR_H

#include <QObject>

#include <NIDAQmx.h>

class QString;
class QStringList;

class DeviceSensor : public QObject
{
    Q_OBJECT

public:
    DeviceSensor();
      virtual ~DeviceSensor();

      static QStringList      getDeviceNames(void);
```

```
        static QString          getLongDeviceName(QString deviceName);

        static QStringList      getAIChannelNames(QString deviceName);
        static QStringList      getAOChannelNames(QString deviceName);
        static QStringList      getDOLines(QString deviceName);

        static QStringList      getAITerminalConfigurations(QString
deviceName);

        static bool             isSimulated(QString deviceName);

        static double           getAIMinSampleRate(QString deviceName);
        static double           getAIMaxSampleRate(QString deviceName);
        static double           getAOMinSampleRate(QString deviceName);
        static double           getAOMaxSampleRate(QString deviceName);
        static double           getDOMaxRate(QString deviceName);

        static double           getAIMinVoltage(QString deviceName);
        static double           getAIMaxVoltage(QString deviceName);
        static double           getAOMinVoltage(QString deviceName);
        static double           getAOMaxVoltage(QString deviceName);

        enum                TerminalConfigurations {
                                    AIDefault = DAQmx_Val_Cfg_Default,
                                    AIReferenced = DAQmx_Val_RSE,
                                    AINonReferenced = DAQmx_Val_NRSE,
                                    AIDifferential = DAQmx_Val_Diff,
                                    AIPseudoDifferential =
DAQmx_Val_PseudoDiff,
                                    AOReferenced = DAQmx_Val_RSE,
                                    AODifferential = DAQmx_Val_Diff,
                                    AOPseudoDifferential =
DAQmx_Val_PseudoDiff
                                };
};
#endif
```

DeviceSensor.cpp

```
    /*
        Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

        This file is part of Medaq.

        Medaq is free software: you can redistribute it and/or modify
        it under the terms of the GNU Lesser General Public License as
published by
        the Free Software Foundation, either version 3 of the License, or
        (at your option) any later version.

        Medaq is distributed in the hope that it will be useful,
        but WITHOUT ANY WARRANTY; without even the implied warranty of
        MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
        GNU Lesser General Public License for more details.

        You should have received a copy of the GNU Lesser General Public
License
        along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#include "DeviceSensor.h"
```

```
#include <QString>
#include <QStringList>
#include <QByteArray>
#include <QMap>
#include <QMessageBox>

#include <NIDAQmx.h>

DeviceSensor::DeviceSensor(){

}

DeviceSensor::~DeviceSensor(){

}

QStringList DeviceSensor::getDeviceNames(){
      QByteArray ba(1024,0);
    DAQmxGetSysDevNames(ba.data(), 1024);
    QString devices(ba);
      QStringList deviceList = devices.split(",");
      for(int i=0; i<deviceList.size(); ++i){
            deviceList[i] = deviceList[i].trimmed();
      }
      return deviceList;
}

QStringList DeviceSensor::getAIChannelNames(QString deviceName){
      QByteArray ba(1024,0);
      DAQmxGetDevAIPhysicalChans(deviceName.toAscii().constData(),
ba.data(), 1024);
      QString channels(ba);
      QStringList channelList = channels.split(",");
      for(int i=0; i<channelList.size(); ++i){
            channelList[i] = channelList[i].trimmed();
      }
      return channelList;
}

QStringList DeviceSensor::getAOChannelNames(QString deviceName){
      QByteArray ba(1024,0);
      DAQmxGetDevAOPhysicalChans(deviceName.toAscii().constData(),
ba.data(), 1024);
      QString channels(ba);
      QStringList channelList = channels.split(",");
      for(int i=0; i<channelList.size(); ++i){
            channelList[i] = channelList[i].trimmed();
      }
      return channelList;
}

QStringList DeviceSensor::getDOLines(QString deviceName){
      QByteArray ba(1024,0);
      DAQmxGetDevDOLines(deviceName.toAscii().constData(), ba.data(),
1024);
      QString lines(ba);
      QStringList lineList = lines.split(",");
      for(int i=0; i<lineList.size(); ++i){
            lineList[i] = lineList[i].trimmed();
      }
      return lineList;
```

```
}

QStringList DeviceSensor::getAITerminalConfigurations(QString deviceName){
	QStringList conf;
	conf << "Referenced Single-Ended."
		<< "Non-Referenced Single-Ended."
		<< "Differential."
		<< "Pseudodifferential.";
	return conf;
}

bool DeviceSensor::isSimulated(QString deviceName){
	bool32 b;
	DAQmxGetDevIsSimulated(deviceName.toAscii().constData(), &b);
	return b;
}

double DeviceSensor::getDOMaxRate(QString deviceName){
	float64 rate = 0;
	DAQmxGetDevDOMaxRate(deviceName.toAscii().constData(), &rate);
	return rate;
}

double DeviceSensor::getAIMinSampleRate(QString deviceName){
	float64 rate = 0;
	DAQmxGetDevAIMinRate(deviceName.toAscii().constData(), &rate);
	return rate;
}

double DeviceSensor::getAIMaxSampleRate(QString deviceName){
	float64 rate = 0;
	DAQmxGetDevAIMaxSingleChanRate(deviceName.toAscii().constData(),
&rate);
	return rate;
}

double DeviceSensor::getAOMinSampleRate(QString deviceName){
	float64 rate = 0;
	DAQmxGetDevAOMinRate(deviceName.toAscii().constData(), &rate);
	return rate;
}

double DeviceSensor::getAOMaxSampleRate(QString deviceName){
	float64 rate = 0;
	DAQmxGetDevAOMaxRate(deviceName.toAscii().constData(), &rate);
	return rate;
}

double DeviceSensor::getAIMinVoltage(QString deviceName){
	float64 marray[100] = {0};
	DAQmxGetDevAIVoltageRngs(deviceName.toAscii().constData(), marray,
100);
	float64 min = 0;
	for (int i=0; i<100; i++)
	{
		if (min > marray[i]) min = marray[i];
	}
	return min;
}

double DeviceSensor::getAIMaxVoltage(QString deviceName){
```

```
	float64 marray[100] = {0};
	DAQmxGetDevAIVoltageRngs(deviceName.toAscii().constData(), marray,
100);
	float64 max = 0;
	for (int i=0; i<100; i++)
	{
		if (max < marray[i]) max = marray[i];
	}
	return max;
}

double DeviceSensor::getAOMinVoltage(QString deviceName){
	float64 marray[100] = {0};
	DAQmxGetDevAOVoltageRngs(deviceName.toAscii().constData(), marray,
100);
	float64 min = 0;
	for (int i=0; i<100; i++)
	{
		if (min > marray[i]) min = marray[i];
	}
	return min;
}

double DeviceSensor::getAOMaxVoltage(QString deviceName){
	float64 marray[100] = {0};
	DAQmxGetDevAOVoltageRngs(deviceName.toAscii().constData(), marray,
100);
	float64 max = 0;
	for (int i=0; i<100; i++)
	{
		if (max < marray[i]) max = marray[i];
	}
	return max;
}

QString DeviceSensor::getLongDeviceName(QString deviceName){

	QMap<QString, int> names;
    names.insert("NI PCI-DIO-96",   0x0160      );
    names.insert("NI PCI-MIO-16XE-50",    0x0162      );
    names.insert("NI DAQCard-6036E",      0x0245      );
    names.insert("NI DAQCard-6062E",      0x02C4      );
    names.insert("NI SCXI-1126",    0x0301      );
    names.insert("NI SCXI-1121",    0x0302      );
    names.insert("NI SCXI-1129",    0x0303      );
    names.insert("NI SCXI-1120",    0x0304      );
    names.insert("NI SCXI-1100",    0x0306      );
    names.insert("NI SCXI-1140",    0x0308      );
    names.insert("NI SCXI-1122",    0x030A      );
    names.insert("NI SCXI-1160",    0x030C      );
    names.insert("NI SCXI-1125",    0x030D      );
    names.insert("NI SCXI-1161",    0x030E      );
    names.insert("NI SCXI-1162",    0x0310      );
    names.insert("NI SCXI-1163",    0x0312      );
    names.insert("NI SCXI-1124",    0x0314      );
    names.insert("NI SCXI-1162HV",  0x0318      );
    names.insert("NI SCXI-1163R",   0x031C      );
    names.insert("NI SCXI-1102",    0x031E      );
    names.insert("NI SCXI-1102B",   0x031F      );
    names.insert("NI SCXI-1141",    0x0320      );
    names.insert("NI SCXI-1112",    0x0321      );
```

```
names.insert("NI SCXI-1191",    0x0322      );
names.insert("NI SCXI-1190",    0x0323      );
names.insert("NI SCXI-1192",    0x0324      );
names.insert("NI SCXI-1600",    0x0329      );
names.insert("NI SCXI-1104",    0x032D      );
names.insert("NI SCXI-1104C",   0x032F      );
names.insert("NI SCXI-1531",    0x0330      );
names.insert("NI SCXI-1540",    0x0331      );
names.insert("NI SCXI-1530",    0x0332      );
names.insert("NI SCXI-1102C",   0x033E      );
names.insert("NI SCXI-1143",    0x0340      );
names.insert("NI SCXI-1120D",   0x0344      );
names.insert("NI SCXI-1127",    0x0345      );
names.insert("NI SCXI-1128",    0x0346      );
names.insert("NI SCXI-1142",    0x0360      );
names.insert("NI SCXI-1500",    0x038F      );
names.insert("NI SCXI-1520",    0x0407      );
names.insert("NI SCXI-1581",    0x048B      );
names.insert("NI SCC-A10",      0x04AC      );
names.insert("NI SCC-CI20",     0x04AD      );
names.insert("NI SCC-FT01",     0x04AE      );
names.insert("NI SCC-TC01",     0x04B2      );
names.insert("NI SCC-TC02",     0x04B3      );
names.insert("NI SCC-FV01",     0x04B4      );
names.insert("NI SCC-DI01",     0x04B6      );
names.insert("NI SCC-DO01",     0x04B7      );
names.insert("NI SCC-LP01",     0x04B8      );
names.insert("NI SCC-LP02",     0x04B9      );
names.insert("NI SCC-LP03",     0x04BA      );
names.insert("NI SCC-LP04",     0x04BC      );
names.insert("NI SCC-RTD01",    0x04BD      );
names.insert("NI SCC-SG01",     0x04BE      );
names.insert("NI SCC-CO20",     0x04BF      );
names.insert("NI SCC-AI01",     0x04DA      );
names.insert("NI SCC-AI02",     0x04DB      );
names.insert("NI SCC-AI03",     0x04DC      );
names.insert("NI SCC-AI04",     0x04DD      );
names.insert("NI SCC-AI05",     0x04DE      );
names.insert("NI SCC-AI06",     0x04DF      );
names.insert("NI SCC-AI07",     0x04E0      );
names.insert("NI SCC-AI13",     0x04E1      );
names.insert("NI SCC-AI14",     0x04E2      );
names.insert("NI SCC-SG02",     0x04E3      );
names.insert("NI SCC-SG03",     0x04E4      );
names.insert("NI SCC-SG04",     0x04E5      );
names.insert("NI SCC-SG11",     0x04E6      );
names.insert("NI SCC-ACC01",    0x04EA      );
names.insert("NI SCC-RLY01",    0x04F0      );
names.insert("NI SCC-AO10",     0x04F6      );
names.insert("NI DAQCard-DIO-24",     0x075C      );
names.insert("NI DAQCard-6024E",      0x075E      );
names.insert("NI DAQCard-6715",0x075F      );
names.insert("NI PCI-DIO-32HS",0x1150      );
names.insert("NI PCI-MIO-16XE-10",    0x1170      );
names.insert("NI PCI-MIO-16E-1",      0x1180      );
names.insert("NI PCI-MIO-16E-4",      0x1190      );
names.insert("NI PXI-6070E",    0x11B0      );
names.insert("NI PXI-6040E",    0x11C0      );
names.insert("NI PXI-6030E",    0x11D0      );
names.insert("NI PCI-6032E",    0x1270      );
names.insert("NI PCI-6704",     0x1290      );
```

```
names.insert("NI PCI-6534",     0x12B0      );
names.insert("NI PCI-6602",     0x1310      );
names.insert("NI PXI-6533",     0x1320      );
names.insert("NI PCI-6031E",    0x1330      );
names.insert("NI PCI-6033E",    0x1340      );
names.insert("NI PCI-6071E",    0x1350      );
names.insert("NI PXI-6602",     0x1360      );
names.insert("NI PXI-6508",     0x13C0      );
names.insert("NI PXI-6534",     0x1490      );
names.insert("NI PCI-6110",     0x14E0      );
names.insert("NI PCI-6111",     0x14F0      );
names.insert("NI PXI-6031E",    0x1580      );
names.insert("NI PXI-6071E",    0x15B0      );
names.insert("NI PXI-6509",     0x1710      );
names.insert("NI PCI-6503",     0x17D0      );
names.insert("NI PCI-6713",     0x1870      );
names.insert("NI PCI-6711",     0x1880      );
names.insert("NI PCI-6052E",    0x18B0      );
names.insert("NI PXI-6052E",    0x18C0      );
names.insert("NI PXI-6704",     0x1920      );
names.insert("NI 6040E",  0x1930      );
names.insert("NI PCI-6133",     0x1AD0      );
names.insert("NI PXI-6133",     0x1AE0      );
names.insert("NI PCI-6624",     0x1E30      );
names.insert("NI PXI-6624",     0x1E40      );
names.insert("NI PCI-6733",     0x2410      );
names.insert("NI PXI-6733",     0x2420      );
names.insert("NI PCI-6731",     0x2430      );
names.insert("NI PXI-4200",     0x24B0      );
names.insert("NI PXI-4472",     0x24F0      );
names.insert("NI PCI-4472",     0x2510      );
names.insert("NI PCI-4474",     0x2520      );
names.insert("NI PCI-6123",     0x27A0      );
names.insert("NI PXI-6123",     0x27B0      );
names.insert("NI PCI-6036E",    0x2890      );
names.insert("NI PXI-4461",     0x28A0      );
names.insert("NI PCI-6013",     0x28B0      );
names.insert("NI PCI-6014",     0x28C0      );
names.insert("NI PCI-6023E",    0x2A60      );
names.insert("NI PCI-6024E",    0x2A70      );
names.insert("NI PCI-6025E",    0x2A80      );
names.insert("NI PXI-6025E",    0x2AB0      );
names.insert("NI PXI-6527",     0x2B10      );
names.insert("NI PCI-6527",     0x2B20      );
names.insert("NI PXI-6713",     0x2B80      );
names.insert("NI PXI-6711",     0x2B90      );
names.insert("NI PCI-6601",     0x2C60      );
names.insert("NI PCI-6035E",    0x2C80      );
names.insert("NI PCI-6703",     0x2C90      );
names.insert("NI PCI-6034E",    0x2CA0      );
names.insert("NI PXI-6608",     0x2CC0      );
names.insert("NI PXI-6115",     0x2EC0      );
names.insert("NI PCI-6115",     0x2ED0      );
names.insert("NI PXI-6120",     0x2EE0      );
names.insert("NI PCI-6120",     0x2EF0      );
names.insert("NI PXI-2593",     0x7023      );
names.insert("NI SCXI-1193",    0x7024      );
names.insert("NI SCXI-1166",    0x703D      );
names.insert("NI SCXI-1167",    0x703E      );
names.insert("NI PXI-2566",     0x703F      );
names.insert("NI PXI-2567",     0x7040      );
```

```
names.insert("NI SCXI-1130",    0x704B      );
names.insert("NI PXI-2530",     0x704C      );
names.insert("NI PXI-4220",     0x704F      );
names.insert("NI PXI-4204",     0x7050      );
names.insert("NI PXI-2529",     0x7067      );
names.insert("NI PCI-6723",     0x7073      );
names.insert("NI PXI-4462",     0x707E      );
names.insert("NI PCI-6509",     0x7085      );
names.insert("NI PXI-6528",     0x7086      );
names.insert("NI PCI-6515",     0x7087      );
names.insert("NI PCI-6514",     0x7088      );
names.insert("NI PXI-2568",     0x708C      );
names.insert("NI PXI-2569",     0x708D      );
names.insert("NI SCXI-1169",    0x7090      );
names.insert("NI SCC-SG24",     0x7099      );
names.insert("NI USB-9421",     0x709F      );
names.insert("NI USB-9472",     0x70A1      );
names.insert("NI USB-9481",     0x70A2      );
names.insert("NI WLS-9211",     0x70A3738E  );
names.insert("NI ENET-9211",    0x70A3738F  );
names.insert("NI USB-9201",     0x70A4      );
names.insert("NI USB-9221",     0x70A5      );
names.insert("NI WLS-9215",     0x70A6738E  );
names.insert("NI ENET-9215",    0x70A6738F  );
names.insert("NI USB-9263",     0x70A7      );
names.insert("NI USB-9233",     0x70A8      );
names.insert("NI PCI-6528",     0x70A9      );
names.insert("NI PCI-6229",     0x70AA      );
names.insert("NI PCI-6259",     0x70AB      );
names.insert("NI PCI-6289",     0x70AC      );
names.insert("NI PXI-6251",     0x70AD      );
names.insert("NI PXI-6220",     0x70AE      );
names.insert("NI PCI-6221",     0x70AF      );
names.insert("NI PCI-6220",     0x70B0      );
names.insert("NI PXI-6229",     0x70B1      );
names.insert("NI PXI-6259",     0x70B2      );
names.insert("NI PXI-6289",     0x70B3      );
names.insert("NI PCI-6250",     0x70B4      );
names.insert("NI PXI-6221",     0x70B5      );
names.insert("NI PCI-6280",     0x70B6      );
names.insert("NI PCI-6254",     0x70B7      );
names.insert("NI PCI-6251",     0x70B8      );
names.insert("NI PXI-6250",     0x70B9      );
names.insert("NI PXI-6254",     0x70BA      );
names.insert("NI PXI-6280",     0x70BB      );
names.insert("NI PCI-6284",     0x70BC      );
names.insert("NI PCI-6281",     0x70BD      );
names.insert("NI PXI-6284",     0x70BE      );
names.insert("NI PXI-6281",     0x70BF      );
names.insert("NI PCI-6143",     0x70C0      );
names.insert("NI PCI-6511",     0x70C3      );
names.insert("NI PCI-6513",     0x70C8      );
names.insert("NI PXI-6515",     0x70C9      );
names.insert("NI PCI-6512",     0x70CC      );
names.insert("NI PXI-6514",     0x70CD      );
names.insert("NI PXI-2570",     0x70D0      );
names.insert("NI PXI-6513",     0x70D1      );
names.insert("NI PXI-6512",     0x70D2      );
names.insert("NI PXI-6511",     0x70D3      );
names.insert("NI PCI-6722",     0x70D4      );
names.insert("NI PXI-2532",     0x70E1      );
```

```
names.insert("NI PCI-6224",     0x70F2      );
names.insert("NI PXI-6224",     0x70F3      );
names.insert("NI DAQPad-6015",  0x70FA      );
names.insert("NI DAQPad-6016",  0x70FB      );
names.insert("NI PXI-6723",     0x70FF      );
names.insert("NI PXI-6722",     0x7100      );
names.insert("NI SCXI-1521",    0x7103      );
names.insert("NI PXI-6143",     0x710D      );
names.insert("NI PCI-6510",     0x7124      );
names.insert("NI PCI-6516",     0x7125      );
names.insert("NI PCI-6517",     0x7126      );
names.insert("NI PCI-6518",     0x7127      );
names.insert("NI PCI-6519",     0x7128      );
names.insert("NI USB-9421 (DSUB)",    0x712E      );
names.insert("NI USB-9472 (DSUB)",    0x7132      );
names.insert("NI WLS-9215 (BNC)",     0x7135738E  );
names.insert("NI ENET-9215 (BNC)",    0x7135738F  );
names.insert("NI PXI-2575",     0x7137      );
names.insert("NI PXI-2585",     0x713C      );
names.insert("NI PXI-2586",     0x713D      );
names.insert("NI SCXI-1175",    0x713F      );
names.insert("NI PXI-4224",     0x7142      );
names.insert("NI SCXI-1521B",   0x7143      );
names.insert("NI PCI-6132",     0x7146      );
names.insert("NI PXI-6132",     0x7147      );
names.insert("NI PCI-6122",     0x7148      );
names.insert("NI PXI-6122",     0x7149      );
names.insert("NI PXI-2564",     0x7150      );
names.insert("NI 9221",   0x715F      );
names.insert("NI 9421",   0x7160      );
names.insert("NI 9421 (DSUB)", 0x7161      );
names.insert("NI 9472",   0x7162      );
names.insert("NI 9472 (DSUB)", 0x7163      );
names.insert("NI 9481",   0x7164      );
names.insert("NI 9401",   0x7165      );
names.insert("NI PCI-6230",     0x716B      );
names.insert("NI PCI-6225",     0x716C      );
names.insert("NI PXI-6225",     0x716D      );
names.insert("NI PCI-4461",     0x716F      );
names.insert("NI PCI-4462",     0x7170      );
names.insert("NI PCI-6010",     0x7171      );
names.insert("NI DAQPad-6015 (Mass Termination)",0x7172      );
names.insert("NI DAQPad-6015 (BNC)", 0x7173      );
names.insert("NI PXI-6230",     0x7177      );
names.insert("NI USB-6008",     0x717A      );
names.insert("NI USB-6009",     0x717B      );
names.insert("NI PCIe-6251",    0x717D      );
names.insert("NI PCIe-6259",    0x717F      );
names.insert("NI USB-6501",     0x718A      );
names.insert("NI PCI-6521",     0x718B      );
names.insert("NI PXI-6521",     0x718C      );
names.insert("NI PCI-6154",     0x7191      );
names.insert("NI USB-9201 (DSUB)",    0x71A1      );
names.insert("NI USB-9221 (DSUB)",    0x71A2      );
names.insert("NI PXI-2594",     0x71A5      );
names.insert("NI SCXI-1194",    0x71A6      );
names.insert("NI PXI-2595",     0x71A7      );
names.insert("NI SCXI-1195",    0x71A8      );
names.insert("NI PXI-2596",     0x71A9      );
names.insert("NI PXI-2597",     0x71AA      );
names.insert("NI PXI-2598",     0x71AB      );
```

```
names.insert("NI PXI-2599",     0x71AC      );
names.insert("NI 9211",   0x71B0      );
names.insert("NI 9215",   0x71B1      );
names.insert("NI 9215 (BNC)",   0x71B2      );
names.insert("NI 9205 (DSUB)",  0x71B3      );
names.insert("NI 9263",   0x71B4      );
names.insert("NI PXI-2584",     0x71BB      );
names.insert("NI PCI-6221 (37-pin)",  0x71BC      );
names.insert("NI USB-9239",     0x71C2      );
names.insert("NI USB-9237",     0x71C3      );
names.insert("NI WLS-9237",     0x71C3738E  );
names.insert("NI ENET-9237",    0x71C3738F  );
names.insert("NI PCI-6520",     0x71C5      );
names.insert("NI PXI-2576",     0x71C6      );
names.insert("NI USB-9211A",    0x71D9      );
names.insert("NI USB-9215A",    0x71DA      );
names.insert("NI USB-9215A (BNC)",    0x71DB      );
names.insert("NI USB-6525",     0x71DF      );
names.insert("NI PCI-6255",     0x71E0      );
names.insert("NI PXI-6255",     0x71E1      );
names.insert("NI 9233",   0x71E7      );
names.insert("NI SCXI-1502",    0x71E8      );
names.insert("NI PCI-6233",     0x7209      );
names.insert("NI PXI-6233",     0x720A      );
names.insert("NI PCI-6238",     0x720B      );
names.insert("NI PXI-6238",     0x720C      );
names.insert("NI USB-6251",     0x7252      );
names.insert("NI USB-6259",     0x7253      );
names.insert("NI 9234",   0x7263      );
names.insert("NI 9206",   0x7264      );
names.insert("NI 9205",   0x7265      );
names.insert("NI SCXI-1503",    0x726A      );
names.insert("NI SCC-CTR01",    0x726E      );
names.insert("NI USB-6210",     0x726F      );
names.insert("NI USB-6211",     0x7270      );
names.insert("NI USB-6215",     0x7271      );
names.insert("NI USB-6218",     0x7272      );
names.insert("NI PXI-4461",     0x7273      );
names.insert("NI PXI-4462",     0x7274      );
names.insert("NI PCI-6232",     0x7279      );
names.insert("NI PXI-6232",     0x727A      );
names.insert("NI PCI-6239",     0x727B      );
names.insert("NI PXI-6239",     0x727C      );
names.insert("NI PCI-6236",     0x7281      );
names.insert("NI PXI-6236",     0x7282      );
names.insert("NI PXI-2554",     0x7283      );
names.insert("NI 9237",   0x7285      );
names.insert("NI USB-6251 (Mass Termination)",    0x72A0      );
names.insert("NI USB-6259 (Mass Termination)",    0x72A1      );
names.insert("NI USB-9234",     0x72B5      );
names.insert("NI WLS-9234",     0x72B5738E  );
names.insert("NI ENET-9234",    0x72B5738F  );
names.insert("NI 9411",   0x72B9      );
names.insert("NI 9422",   0x72BA      );
names.insert("NI 9423",   0x72BB      );
names.insert("NI 9435",   0x72BC      );
names.insert("NI 9474",   0x72BD      );
names.insert("NI 9485",   0x72BE      );
names.insert("NI 9403",   0x72BF      );
names.insert("NI 9425",   0x72C0      );
names.insert("NI 9476",   0x72C1      );
```

```
names.insert("NI 9477",   0x72C2      );
names.insert("NI 9264",   0x72C3      );
names.insert("NI 9265",   0x72C4      );
names.insert("NI 9201",   0x72C5      );
names.insert("NI 9201 (DSUB)", 0x72C6      );
names.insert("NI 9221 (DSUB)", 0x72C7      );
names.insert("NI 9203",   0x72C8      );
names.insert("NI 9217",   0x72C9      );
names.insert("NI 9219",   0x72CA      );
names.insert("NI 9239",   0x72CB      );
names.insert("NI SensorDAQ",    0x72CC      );
names.insert("NI PXI-2545",     0x72D0      );
names.insert("NI PXI-2546",     0x72D1      );
names.insert("NI PXI-2547",     0x72D2      );
names.insert("NI PXI-2548",     0x72D3      );
names.insert("NI PXI-2549",     0x72D4      );
names.insert("NI PXI-2555",     0x72D5      );
names.insert("NI PXI-2556",     0x72D6      );
names.insert("NI PXI-2557",     0x72D7      );
names.insert("NI PXI-2558",     0x72D8      );
names.insert("NI PXI-2559",     0x72D9      );
names.insert("NI USB-6221",     0x72DC      );
names.insert("NI USB-6229",     0x72DE      );
names.insert("NI PXI-4498",     0x72EF      );
names.insert("NI PXI-4496",     0x72F0      );
names.insert("NI USB-6005 VSA", 0x72F3      );
names.insert("NI 9229",   0x72FA      );
names.insert("NI USB-9229",     0x72FD      );
names.insert("NI USB-6509",     0x72FF      );
names.insert("NI USB-9219",     0x730C      );
names.insert("NI WLS-9219",     0x730C738E  );
names.insert("NI ENET-9219",    0x730C738F  );
names.insert("NI USB-4431",     0x7318      );
names.insert("NI PXI-2535",     0x731C      );
names.insert("NI PXI-2536",     0x731D      );
names.insert("NI PXIe-6124",    0x7322      );
names.insert("NI PXI-6529",     0x7327      );
names.insert("NI USB-6255",     0x732D      );
names.insert("NI USB-6255 (Mass Termination)",    0x732E      );
names.insert("NI USB-6225",     0x732F      );
names.insert("NI USB-6225 (Mass Termination)",    0x7330      );
names.insert("NI PXI-2533",     0x7335      );
names.insert("NI PXI-2534",     0x7336      );
names.insert("NI 9402",   0x7337      );
names.insert("NI USB-6212",     0x7339      );
names.insert("NI USB-6216",     0x733B      );
names.insert("NI USB-6281",     0x733F      );
names.insert("NI USB-6281 (Mass Termination)",    0x7340      );
names.insert("NI PXI-4461",     0x7342      );
names.insert("NI USB-6289",     0x7343      );
names.insert("NI USB-6289 (Mass Termination)",    0x7344      );
names.insert("NI USB-6221 (BNC)",     0x7345      );
names.insert("NI USB-6229 (BNC)",     0x7346      );
names.insert("NI USB-6251 (BNC)",     0x7347      );
names.insert("NI USB-6259 (BNC)",     0x7348      );
names.insert("NI PXI-4495",     0x7359      );
names.insert("NI USB-9239 (BNC)",     0x7367      );
names.insert("NI USB-9229 (BNC)",     0x7368      );
names.insert("NI USB-9263 (BNC)",     0x7369      );
names.insert("NI PXI-4461",     0x7370      );
names.insert("NI 9229 (BNC)",   0x737A      );
```

```
    names.insert("NI 9239 (BNC)",   0x737B      );
    names.insert("NI 9263 (BNC)",   0x737C      );
    names.insert("NI 9235",   0x7381      );
    names.insert("NI 9236",   0x7382      );
    names.insert("NI 9225",   0x7388      );
    names.insert("NI USB-6212 (Mass Termination)",    0x7389      );
    names.insert("NI USB-6216 (Mass Termination)",    0x738A      );
    names.insert("NI PXIe-4498",    0x73A1      );
    names.insert("NI PXIe-4496",    0x73A2      );
    names.insert("NI USB-9213",     0x73A3      );
    names.insert("NI ELVIS II",     0x73A6      );
    names.insert("NI PXIe-2527",    0x73C5      );
    names.insert("NI PXIe-2529",    0x73C6      );
    names.insert("NI PXIe-2530",    0x73C8      );
    names.insert("NI PXIe-2532",    0x73C9      );
    names.insert("NI PXIe-2569",    0x73CA      );
    names.insert("NI PXIe-2575",    0x73CB      );
    names.insert("NI PXIe-2593",    0x73CC      );
    names.insert("NI USB-4432",     0x73D1      );
    names.insert("NI 9213",   0x73E2      );
    names.insert("NI 9426",   0x73E3      );
    names.insert("NI 9475",   0x73E4      );
    names.insert("NI 9478",   0x73E5      );
    names.insert("NI 9237 (DSUB)", 0x73E6      );
    names.insert("NI PXI-2501",     0x9020      );
    names.insert("NI PXI-2503",     0x9030      );
    names.insert("NI PXI-2527",     0x9040      );
    names.insert("NI PXI-2565",     0x9050      );
    names.insert("NI PXI-2590",     0x9060      );
    names.insert("NI PXI-2591",     0x9070      );

      uInt32 productNumber;
      DAQmxGetDevProductNum(deviceName.toAscii().constData(),
&productNumber);
      return names.key(productNumber);
}
```

DeviceController.h

```
/*
   Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

   This file is part of Medaq.

   Medaq is free software: you can redistribute it and/or modify
   it under the terms of the GNU Lesser General Public License as
published by
   the Free Software Foundation, either version 3 of the License, or
   (at your option) any later version.

   Medaq is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
   GNU Lesser General Public License for more details.

   You should have received a copy of the GNU Lesser General Public
License
   along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef DEVICECONTROLLER_H
```

```
#define DEVICECONTROLLER_H

#include <QObject>
#include <NIDAQmx.h>

#include <QVector>
QT_BEGIN_NAMESPACE
class QTimer;
QT_END_NAMESPACE


class Plotter;

class DeviceController : public QObject
{
    Q_OBJECT

public:
    DeviceController(Plotter * plotter);
      virtual ~DeviceController();

      QString getLastError(){ return lastError; }
      int         getState(){ return state; }
      bool newSampling( QString channelName, int terminalConfiguration,
                                    double minVoltage, double maxVoltage,
                                    unsigned int visibleDuration, unsigned
int samplingRate,
                                    bool startRecordImmediately = false );
      static int32 CVICALLBACK callback_Wrapper( TaskHandle taskHandle,

      int32 everyNsamplesEventType,

      uInt32 nSamples, void *callbackData);
      int32 CVICALLBACK callback( TaskHandle taskHandle,
                                                int32
everyNsamplesEventType,
                                                uInt32 nSamples, void
*callbackData);
      bool startAnalogOut(double value);
      bool stopAnalogOut();
      bool updateAnalogOut(double value);
      bool startDigitalOut(int frequency);
      bool stopDigitalOut();
      bool updateDigitalOut(int frequency);

signals:
      void stateChanged(int state);
      void logMessage(QString message);

public slots:
      void timerFunction();
      void startRecording(); // to recording
      void pauseRecording(); // to plotting
      void stopRecording(); // to scrolling
      void clearSampling(); // to waiting
      void stopSampling();

public:
      enum State {NoDevice, Waiting, Plotting, Recording, Scrolling,
Paused} state;
```

```
private:
	void saveLastError(QString errorPrefix);

private:
	QVector<double>	fifo;
	unsigned int	fifo_r;
	unsigned int	fifo_w;
	unsigned int	fifo_size;
	volatile bool	overflow;
	QAtomicInt	fifo_count;

	QString		lastError;
	TaskHandle	taskHandle;
	TaskHandle	taskHandleAnalogOut;
	TaskHandle	taskHandleDigitalOut;
	unsigned int	refreshRate;
	QTimer		* timer;
	Plotter		* plotter;

};

#endif
```

DeviceController.cpp

```
/*
	Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

	This file is part of Medaq.

	Medaq is free software: you can redistribute it and/or modify
	it under the terms of the GNU Lesser General Public License as
published by
	the Free Software Foundation, either version 3 of the License, or
	(at your option) any later version.

	Medaq is distributed in the hope that it will be useful,
	but WITHOUT ANY WARRANTY; without even the implied warranty of
	MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
	GNU Lesser General Public License for more details.

	You should have received a copy of the GNU Lesser General Public
License
	along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#include "DeviceController.h"

#include <NIDAQmx.h>

#include <QVector>
#include <QSettings>
#include <QDir>
#include <QDebug>
#include "Plotter.h"
#include <QTimer>
#include "DeviceSensor.h"


DeviceController::DeviceController(Plotter * plotter) :
	taskHandle( NULL ),
```

```
	taskHandleAnalogOut( NULL ),
	taskHandleDigitalOut( NULL ),
	fifo_r(0),
	fifo_w(0),
	overflow(false),
	fifo_count(0),
	plotter(plotter),
	state(Waiting)
{
	QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
	settings.beginGroup("Settings");
	QString currentDevice = settings.value("currentDevice",
QString("")).toString();
	fifo_size = settings.value("fifoSize", 1000000).toUInt();
	refreshRate = settings.value("refreshRate", 20).toUInt();
	settings.endGroup();

	fifo.resize(fifo_size);

	if(currentDevice.isEmpty())
		state = NoDevice;
	emit stateChanged(state);

	timer = new QTimer(this);
	connect(timer, SIGNAL(timeout()), this, SLOT(timerFunction()));

}



DeviceController::~DeviceController()
{
	clearSampling();
	stopAnalogOut();
	stopDigitalOut();
}



bool DeviceController::newSampling( QString channelName, int
terminalConfiguration,
				double minVoltage, double maxVoltage,
				unsigned int visibleDuration, unsigned
int samplingRate,
				bool startRecordImmediately )
{
	if( state == NoDevice )
	{
		lastError = tr("No device");
		emit logMessage(lastError);
		return false;
	}

	if( state == Plotting || state == Recording )
	{
		lastError = tr("A sampling is already running. First stop
sampling than retry again.");
		emit logMessage(lastError);
		return false;
	}
```

```
	if( 0 != DAQmxCreateTask( "", &taskHandle) )
                                                { saveLastError(tr("While
creating the analog in task:\n")); return false; }

	if( 0 != DAQmxCreateAIVoltageChan( taskHandle,
channelName.toAscii().constData(), "",

	terminalConfiguration, minVoltage, maxVoltage, DAQmx_Val_Volts, NULL
) )
                                                { saveLastError(tr("While
creating the analog in voltage channel:\n")); return false; }

	if( 0 != DAQmxCfgSampClkTiming( taskHandle, NULL /* OnboardClock */,
samplingRate /* per second per channel */,
                                                        DAQmx_Val_Rising,
DAQmx_Val_ContSamps, fifo_size /* buffer size */ ) )
                                                { saveLastError(tr("While
configuring the clock timing:\n")); return false; }

	if( 0 != DAQmxRegisterEveryNSamplesEvent( taskHandle,
DAQmx_Val_Acquired_Into_Buffer /* for input */,

	samplingRate/refreshRate /* nSamples */, 0, &callback_Wrapper,

	reinterpret_cast<void*>(this) ) )
                                                { saveLastError(tr("While
registering the callback function:\n")); return false; }

	if( 0 != DAQmxStartTask( taskHandle ) )
                                                { saveLastError(tr("While
starting the analog in task:\n")); return false; }


	plotter->prepareForPlot(visibleDuration, samplingRate);

	if( startRecordImmediately )
	{
		startRecording();
	}
	else
	{
		state = Plotting;
		emit stateChanged(state);
	}

	timer->start(1000/refreshRate);

	return true;
}


/* static wrapper function be able to reach to the class members */
int32 CVICALLBACK DeviceController::callback_Wrapper( TaskHandle
taskHandle,

		int32 everyNsamplesEventType,

		uInt32 nSamples, void *callbackData)
```

```
{
      DeviceController * dc = reinterpret_cast<DeviceController *
>(callbackData);
    dc->callback( taskHandle,  everyNsamplesEventType,  nSamples, NULL);
    return 0;
}


int32 CVICALLBACK DeviceController::callback( TaskHandle taskHandle,
                                                                  int32
everyNsamplesEventType,
                                                                  uInt32
nSamples, void *callbackData)
{
      Q_UNUSED(callbackData)
      Q_UNUSED(everyNsamplesEventType)
      /*
        The array to read samples into, organized according to fillMode.
            DAQmx_Val_GroupByChannel = Group by channel (non-interleaved)
                nSamples from channel 1, than nSamples from channel 2, ...
            DAQmx_Val_GroupByScanNumber   Group by scan number
(interleaved)
                1 sample from channel 1, than 1 sample from channel 2, ...

        ! nSamples for each channel => channel count x nSamples = buffer
size

            got only one channel
    */
      int32 read = 0;
      double * buff = new double[nSamples];

      if( 0 != DAQmxReadAnalogF64( taskHandle, DAQmx_Val_Auto /* read all
available sample */,

      DAQmx_Val_WaitInfinitely, DAQmx_Val_GroupByScanNumber,
                                                      buff /* read array */,
nSamples /* arraySizeInSamps */,
                                                      &read, NULL ) )
    {
        saveLastError(tr("While reading the sampling values:\n"));
            return false;
    }

      if( ( fifo_count + nSamples ) > fifo_size )
      {
            overflow = true;
            qDebug() << "overflow in callback";
      }
      else
      {
            if( fifo_w + nSamples < fifo_size )
            {
                  memcpy(&(fifo.data()[fifo_w]), buff,
nSamples*sizeof(double));
                  fifo_w += nSamples;
            }
            else
            {
                  unsigned int first = fifo_size - fifo_w;
                  unsigned int second = nSamples - first;
```

```
                memcpy(&(fifo.data()[fifo_w]), &(buff[0]),
first*sizeof(double));
                memcpy(&(fifo.data()[0]), &(buff[first]),
second*sizeof(double));
                fifo_w = second;
            }
            fifo_count.fetchAndAddOrdered(nSamples);
      }

      delete [] buff;

      return 0;
}


void DeviceController::saveLastError(QString errorPrefix)
{
      char errBuff[2048]={'\0'};
      lastError.clear();
      DAQmxGetExtendedErrorInfo(errBuff,2048);
      lastError = QString("<p><font color='#FF0000'>") +  errorPrefix +
QString("</p></font>") + QString::fromAscii(errBuff);
      emit logMessage(lastError);
}


void DeviceController::timerFunction()
{
      if( overflow )
      {
            fifo_r = fifo_w;
            overflow = false;
            fifo_count.fetchAndStoreOrdered(0);
            qDebug() << "overflow occured";
            return;
      }

      if( state == Waiting || state == Scrolling )
      {
            qDebug() << "Entered timer fnc while state is waiting or
scrolling.";
            if(fifo_count > 0)
            {
                  fifo_r = fifo_w;
                  fifo_count.fetchAndStoreOrdered(0);
            }
            return;
      }

      // get data from fifo
      unsigned int count = fifo_count;

      if(count == 0)
            return;

      double * buff = new double[count];

      if( fifo_r + count < fifo_size )
      {
            memcpy(&(buff[0]), &(fifo.data()[fifo_r]),
count*sizeof(double));
```

```
            fifo_r += count;
      }
      else
      {
            unsigned int first = fifo_size - fifo_r;
            unsigned int second = count - first;
            memcpy(&(buff[0]), &(fifo.data()[fifo_r]),
first*sizeof(double));
            memcpy(&(buff[first]), &(fifo.data()[0]),
second*sizeof(double));
            fifo_r = second;
      }
      fifo_count.fetchAndAddOrdered(-count);

      // pump data to either plot or record

      if( state == Plotting )
      {
            plotter->plot(buff, count);
            plotter->replot();
      }
      else if( state == Recording )
      {
            plotter->recordAndPlot(buff, count);
            plotter->replot();
      }

      delete [] buff;
}


void DeviceController::startRecording()
{
      if( state == NoDevice )
      {
            lastError = tr("No device");
            emit logMessage(lastError);
            return;
      }

      state = Recording;
      emit stateChanged(state);
}


void DeviceController::pauseRecording()
{
      state = Paused;
      emit stateChanged(state);
}


void DeviceController::stopRecording()
{
      if(timer->isActive())
            timer->stop();

      state = Scrolling;
      emit stateChanged(state);
}
```

```
void DeviceController::stopSampling()
{
	if(timer->isActive())
		timer->stop();

	if( 0 != DAQmxClearTask(taskHandle))
			{ saveLastError(tr("While clearing the analog out
task:\n")); }
}

void DeviceController::clearSampling()
{
	if(timer->isActive())
		timer->stop();

	if( 0 != DAQmxClearTask(taskHandle))
			{ saveLastError(tr("While clearing the analog out
task:\n")); }

	// clear buffer
	fifo_w = fifo_r = 0;
	fifo_count = 0;
	overflow = false;

	state = Waiting;
	emit stateChanged(state);
}

bool DeviceController::startAnalogOut(double value){

	QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
	settings.beginGroup("Settings");
	QString currentDevice = settings.value("currentDevice",
QString("")).toString();
	QString currentAOChannel = settings.value("currentAOChannel",
QString("")).toString();
	settings.endGroup();


	if( 0 != DAQmxCreateTask( "", &taskHandleAnalogOut) )
					{ saveLastError(tr("While
creating the analog out task:\n")); return false; }

	if( 0 != DAQmxCreateAOVoltageChan( taskHandleAnalogOut,

	currentAOChannel.toAscii().constData(),
					"",

	DeviceSensor::getAOMinVoltage(currentDevice),

	DeviceSensor::getAOMaxVoltage(currentDevice),
					DAQmx_Val_Volts,
					NULL ) )
				{ saveLastError(tr("While
creating the analog out voltage channel:\n")); return false; }
	if( 0 != DAQmxStartTask( taskHandleAnalogOut ) )
				{ saveLastError(tr("While
starting the analog out task:\n")); return false; }

	updateAnalogOut(value);
```

```
	return true;
}


bool DeviceController::stopAnalogOut(){
	if(taskHandleAnalogOut)
	{
		if( 0 != DAQmxClearTask(taskHandleAnalogOut))
			{ saveLastError(tr("While clearing the analog out
task:\n")); return false; }
		return true;
	}
	return false;
}

bool DeviceController::updateAnalogOut(double value){
	if(taskHandleAnalogOut)
	{
		float64 data[1] = {0.0};
		data[0] = value;
		if( 0 !=
DAQmxWriteAnalogF64(taskHandleAnalogOut,1,1,2.0,DAQmx_Val_GroupByChannel,da
ta,NULL,NULL))
							{ saveLastError(tr("While
writing the analog out value:\n")); return false; }
		return true;
	}
	return false;
}

bool DeviceController::startDigitalOut(int frequency){

	QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
	settings.beginGroup("Settings");
	QString currentDevice = settings.value("currentDevice",
QString("")).toString();
	QString currentDigitalLine = settings.value("currentDigitalLine",
QString("")).toString();
	settings.endGroup();


	if( 0 != DAQmxCreateTask( "", &taskHandleDigitalOut) )
							{ saveLastError(tr("While
creating the digital out task:\n")); return false; }

	if( 0 != DAQmxCreateDOChan( taskHandleDigitalOut,
currentDigitalLine.toAscii().constData(), "", DAQmx_Val_ChanForAllLines ) )
							{ saveLastError(tr("While
creating the digital out line:\n")); return false; }
	if( 0 != DAQmxCfgSampClkTiming( taskHandleDigitalOut, NULL,
frequency, DAQmx_Val_Rising, DAQmx_Val_ContSamps, 2 ) )
							{ saveLastError(tr("While
configuring the digital out clock timing:\n")); return false; }
	if( 0 != DAQmxStartTask( taskHandleDigitalOut ) )
							{ saveLastError(tr("While
starting the digital out task:\n")); return false; }

	updateDigitalOut(frequency);
```

```
	return true;
}

bool DeviceController::stopDigitalOut(){
	if(taskHandleDigitalOut)
	{
		if( 0 != DAQmxClearTask(taskHandleDigitalOut))
			{ saveLastError(tr("While clearing the digital out
task:\n")); return false; }
		return true;
	}
	return false;
}

bool DeviceController::updateDigitalOut(int frequency){
	if(taskHandleDigitalOut)
	{
		uInt8 data[2] = {0x00, 0x01};
		if( 0 != DAQmxCfgSampClkTiming( taskHandleDigitalOut, NULL,
frequency, DAQmx_Val_Rising, DAQmx_Val_ContSamps, 2 ) )
							{ saveLastError(tr("While
updating the configuration of digital out clock timing:\n")); return false;
}
		if( 0 !=
DAQmxWriteDigitalLines(taskHandleDigitalOut,2,1,2.0,DAQmx_Val_GroupByChanne
l,data,NULL,NULL))
							{ saveLastError(tr("While
writing the digital out value:\n")); return false; }
		return true;
	}
	return false;
}
```

Plotter.h

```
/*
	Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

	This file is part of Medaq.

	Medaq is free software: you can redistribute it and/or modify
	it under the terms of the GNU Lesser General Public License as
published by
	the Free Software Foundation, either version 3 of the License, or
	(at your option) any later version.

	Medaq is distributed in the hope that it will be useful,
	but WITHOUT ANY WARRANTY; without even the implied warranty of
	MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
	GNU Lesser General Public License for more details.

	You should have received a copy of the GNU Lesser General Public
License
	along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef PLOTTER_H
#define PLOTTER_H

#include <qwt_plot.h>
```

```
class QwtPlotCurve;
class QwtPlotItem;

class Plotter : public QwtPlot
{
    Q_OBJECT

public:
    Plotter(QWidget * parent = 0);
      ~Plotter()
      {
      }

      void prepareForPlot(unsigned int visibleDuration, unsigned int
sampingRate);
      void prepareForScroll();
      void plot(double * buff, unsigned int size);
      void recordAndPlot(double * buff, unsigned int size);
      int getMaxScrollStep();
      void setScale(double min, double max, bool autoScale = false);
      unsigned int getVisibleDuration() { return visibleDuration; }
      unsigned int getSamplingRate() {return samplingRate; }

signals:
      void recordTimeChanged(int);

public slots:
      void scrollTime(int step);
      void clearPlotter();

public:
      QVector<double> recordData;

private:
      QVector<double> plotData, plotBottomData;

      QwtPlotCurve * samplingCurve;

      unsigned int visibleDuration;
      unsigned int samplingRate;
      unsigned int plotSplitCount;
      bool        ready;
      int               recordTime;

};
#endif
```

Plotter.cpp

```
/*
    Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

    This file is part of Medaq.

    Medaq is free software: you can redistribute it and/or modify
    it under the terms of the GNU Lesser General Public License as
published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    Medaq is distributed in the hope that it will be useful,
```

```
      but WITHOUT ANY WARRANTY; without even the implied warranty of
      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
      GNU Lesser General Public License for more details.

      You should have received a copy of the GNU Lesser General Public
License
      along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#include <qwt_plot.h>
#include <qwt_plot_curve.h>
#include <qwt_scale_widget.h>
#include <qwt_plot_picker.h>
#include "Plotter.h"
#include "TimeScale.h"
#include <qmath.h>
#include <QSettings>
#include <QDir>
#include <QDebug>
#include <QTime>

Plotter::Plotter(QWidget * parent) :
      QwtPlot(parent),
      ready(false),
      recordTime(0)
{
      setCanvasBackground(Qt::white);


}

void Plotter::prepareForPlot(unsigned int visibleDuration, unsigned int
samplingRate)
{

      this->visibleDuration = visibleDuration;
      this->samplingRate = samplingRate;

      QSettings settings(QDir::currentPath() + "/medaq.ini",
QSettings::IniFormat);
      settings.beginGroup("Settings");
            unsigned int maxVisiblePoint  =
settings.value("maxVisiblePoint", 1000).toInt();
            plotSplitCount                          =
settings.value("plotSplitCount", 50).toInt();
      settings.endGroup();

      unsigned int size;
      if ( visibleDuration * samplingRate > maxVisiblePoint )
      {
            size = maxVisiblePoint;
      }
      else
      {
            size = visibleDuration * samplingRate;
      }

      plotData.fill(0.0, size);
      plotBottomData.resize(size);

      for(int i=0; i<plotBottomData.size(); ++i){
```

```
            plotBottomData[i] = i;
      }

      samplingCurve = new QwtPlotCurve(tr("Sampling"));
      samplingCurve->setPen(QPen(Qt::black));
      samplingCurve->attach(this);
      samplingCurve->setRawData(plotBottomData.data(), plotData.data(),
size);

      // time scale
      TimeScaleDraw * tsd = new TimeScaleDraw(QTime(0,0), samplingRate);
      setAxisScaleDraw(QwtPlot::xBottom, tsd );
    setAxisScale(QwtPlot::xBottom, 0, size);
    setAxisLabelRotation(QwtPlot::xBottom, -50.0);
    setAxisLabelAlignment(QwtPlot::xBottom, Qt::AlignLeft |
Qt::AlignBottom);
      // To avoid this "jumping canvas" effect, we add a permanent margin.
      QwtScaleWidget *scaleWidget = axisWidget(QwtPlot::xBottom);
    const int fmh = QFontMetrics(scaleWidget->font()).height();
    scaleWidget->setMinBorderDist(0, fmh / 2);

      ready = true;
}


void Plotter::prepareForScroll()
{
      if(!ready)
            return;

      unsigned int size = visibleDuration * samplingRate;
      plotData.fill(0.0, size);
      plotBottomData.resize(size);
      for(int i=0; i<plotBottomData.size(); ++i){
            plotBottomData[i] = i;
      }

      samplingCurve->setRawData(plotBottomData.data(), plotData.data(),
size);

      setAxisScale(QwtPlot::xBottom, 0, size);

      QwtPlotPicker * d_picker = new QwtPlotPicker(QwtPlot::xBottom,
QwtPlot::yLeft,
            QwtPicker::PointSelection | QwtPicker::DragSelection,
            QwtPlotPicker::CrossRubberBand, QwtPicker::AlwaysOn,
            canvas());
      d_picker->setRubberBandPen(QColor(Qt::green));
      d_picker->setRubberBand(QwtPicker::CrossRubberBand);
      d_picker->setTrackerPen(QColor(Qt::blue));



}

void Plotter::plot(double * buff, unsigned int size)
{
      if(!ready)
            return;

      unsigned int plotSize = plotData.size();
```

```
	if( size < plotSize ) // size is smaller than plot size
	{
		if((visibleDuration*samplingRate) > plotSize)		// take by
jump
		{
			unsigned int stepLength =
qFloor((visibleDuration*samplingRate) / plotSize);
			unsigned int newSize = qFloor(size / stepLength);

			memmove(&(plotData.data()[0]), &(plotData.data()
[newSize]), (plotSize-newSize) * sizeof(double)); // shift
			memset(&(plotData.data()[(plotSize-newSize)]),0, newSize
* sizeof(double)); // clear shifted place

			unsigned int plotIndex = plotSize-1;
			for(int j=size-1; j>=0 && plotIndex >= (plotSize-
newSize); j-=stepLength)
			{
				plotData[plotIndex] = buff[j];
				plotIndex--;
			}
		}
		else		// take original
		{
			memmove(&(plotData.data()[0]), &(plotData.data()[size]),
(plotSize-size) * sizeof(double)); // shift
			memcpy(&(plotData.data()[(plotSize-size)]), &(buff[0]),
size * sizeof(double));
		}
	}
	else		// size is bigger than plot size or equal
	{
		if((visibleDuration*samplingRate) > plotSize)		// take
by jump
		{
			unsigned int stepLength =
qFloor((visibleDuration*samplingRate) / plotSize);
			unsigned int newSize = qFloor(size / stepLength);

			if(newSize < plotSize)
			{
				memmove(&(plotData.data()[0]), &(plotData.data()
[newSize]), (plotSize-newSize) * sizeof(double)); // shift
				memset(&(plotData.data()[(plotSize-newSize)]),0,
newSize * sizeof(double)); // clear shifted place
			}
			else
			{
				memset(&(plotData.data()[0]),0, plotSize *
sizeof(double)); // clear all
			}

			int plotIndex = plotSize-1;
			for(int j=size-1; j>=0 && plotIndex >= 0; j-=stepLength)
			{
				plotData[plotIndex] = buff[j];
				plotIndex--;
			}
		}
		else		// take original
```

```
            {
                  memcpy(&(plotData.data()[0]), &(buff[size-plotSize]),
plotSize * sizeof(double));
            }
      }
}


void Plotter::recordAndPlot(double * buff, unsigned int size){
      if(!ready)
            return;

      int last = recordData.size();
      recordData.resize(recordData.size()+size); // initialize new emelents
0 by qt
      memcpy(&(recordData.data()[last]), &(buff[0]), size *
sizeof(double));

      int sec = qFloor( recordData.size() / samplingRate );
      if( sec >= recordTime )
      {
            recordTime = sec;
            emit recordTimeChanged(recordTime);
      }

      plot(buff, size);
}

void Plotter::scrollTime(int step){
      if( ! ready )
            return;

      int sourceIndex = 0;
      if( recordData.size() <= plotData.size() ) // stopped before signel
filled the plot screen
      {
            plotData.fill(0.0, plotData.size());
            memcpy(&(plotData.data()[0]), &(recordData.data()[0]),
recordData.size() * sizeof(double));
      }
      else
      {
            if( step > getMaxScrollStep() )           // if step is out of
scope return
                  return;

            sourceIndex = step*(plotData.size()/plotSplitCount);
            memcpy(&(plotData.data()[0]), &(recordData.data()
[sourceIndex]), plotData.size() * sizeof(double));
      }

      unsigned int plotSize = plotBottomData.size();
      double * d = plotBottomData.data();
      QwtValueList vlist_y[3];
      for ( unsigned int i = 0 ; i < plotSize ; ++i )
      {
            register int x = sourceIndex + i;
            d[i] = x;
            if( 0 == x % samplingRate )
                  vlist_y[0] << d[i];
      }
```

```
	setAxisScale(QwtPlot::xBottom, plotBottomData[0],
plotBottomData[plotSize - 1]);
	QwtScaleDiv scdiv_y(plotBottomData.at(0),
plotBottomData.at(plotBottomData.size()-1), vlist_y);
	scdiv_y.setTicks(QwtScaleDiv::MajorTick, vlist_y[0]);
	setAxisScaleDiv(QwtPlot::xBottom, scdiv_y);

	replot();
}


int Plotter::getMaxScrollStep(){
	if(!ready)
		return 0;

	if(recordData.size()<=plotData.size())
		return 0;

	int piece = qFloor(plotData.size()/plotSplitCount);
	if(piece<1)
		piece = 1;
	return qFloor( (recordData.size()-plotData.size()) / piece);
}


void Plotter::clearPlotter(){

	ready = false;

	plotData.fill(0.0, plotData.size());
	plotData.clear();
	recordData.clear();

	clear();

	replot();
}

void Plotter::setScale(double min, double max, bool autoScale)
{
	if(autoScale)
	{
		setAxisAutoScale(QwtPlot::yLeft);
	}
	else
	{
		setAxisScale(QwtPlot::yLeft, min, max);
	}
	replot();
}
```

TimeScale.h

```
/*
	Copyright 2008 Huseyin Kozan (posta@huseyinkozan.com.tr)

	This file is part of Medaq.

	Medaq is free software: you can redistribute it and/or modify
	it under the terms of the GNU Lesser General Public License as
published by
```

```
      the Free Software Foundation, either version 3 of the License, or
      (at your option) any later version.

      Medaq is distributed in the hope that it will be useful,
      but WITHOUT ANY WARRANTY; without even the implied warranty of
      MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
      GNU Lesser General Public License for more details.

      You should have received a copy of the GNU Lesser General Public
License
      along with Medaq.  If not, see <http://www.gnu.org/licenses/>.
*/

#ifndef TIMESCALE_H
#define TIMESCALE_H

#include <qwt_scale_draw.h>
#include <QTime>

class TimeScaleDraw: public QwtScaleDraw
{
public:
    TimeScaleDraw(const QTime &base, int samplingRate):
        baseTime(base),
            rate(samplingRate)
    {
    }
    virtual QwtText label(double v) const
    {
        QTime upTime = baseTime.addSecs((int)v/rate);
            return upTime.toString("mm:ss");
    }
private:
    QTime baseTime;
      int rate;
};
#endif
```

EK B– QMAKE PROJE DOSYASI

```
TEMPLATE = app
TARGET = Medaq

HEADERS = src/MainWindow.h \
 src/InfoDialog.h \
 src/SettingsDialog.h \
 src/NewSamplingDialog.h \
 src/DeviceSensor.h \
 src/Plotter.h \
  src/TimeScale.h \
 src/DeviceController.h

SOURCES = src/main.cpp \
 src/MainWindow.cpp \
 src/InfoDialog.cpp \
 src/SettingsDialog.cpp \
 src/NewSamplingDialog.cpp \
```

```
 src/DeviceSensor.cpp \
 src/Plotter.cpp \
 src/DeviceController.cpp

FORMS = src/MainWindow.ui \
 src/InfoDialog.ui \
 src/NewSamplingDialog.ui \
 src/SettingsDialog.ui

CONFIG += build_all debug_and_release qt thread

CONFIG(debug, debug|release) {
   LIBS += qwtd5.lib
}
else {
   LIBS += qwt5.lib
}
LIBS += NIDAQmx.lib
LIBS += libfftw.lib

DEFINES += QWT_DLL

RESOURCES = resource/medaq.qrc
TRANSLATIONS = resource/medaq.ts
RC_FILE = resource/medaq.rc

MOC_DIR = tmp
OBJECTS_DIR = tmp/o
UI_DIR = tmp
UI_SOURCES_DIR = tmp
UI_HEADERS_DIR = tmp
RCC_DIR = tmp
```

EK C– LİSANS

GNU GPL (Genel Kamu Lisansı) Sürüm 3

29 Haziran 2007

Önsöz

GNU GPL yazılım ve diğer tür işler için copyleft (özgür) bir lisanstır.

Pek çok yazılımın ve diğer kullanışlı şeyin lisansı onu paylaşma ve değiştirme özgürlüğünüzü elinizden almak için hazırlanmıştır. Buna karşılık, GNU GPL (Genel Kamu Lisansı), bir programı değiştirme ve paylaşma hakkınızın mahfuz tutulmasını ve tüm kullanıcıları için

özgür yazılım olarak kalmasını güvence altına almayı amaçlar. Free Software Foundation pek çok yazılımı için GNU GPL kullanmaktadır; ve GPL bu lisansı taşıyan diğer işlere de uygulanır. Siz de kendi programlarınıza bu lisansı uygulayabilirsiniz.

Özgür yazılım dediğimizde bahsettiğimiz ücret değil özgürlüktür. Bizim Genel Kamu Lisanslarımız, sizin özgür yazılımların kopyalarını dağıtma (ve isterseniz onları ücretlendirme) özgürlüğünüzü, yazılım kaynak kodlarının size dağıtım esnasında veya isterseniz verilmesini, yazılımı değiştirme ya da istediğiniz bölümlerini yeni programlarınızda kullanabilmenizi ve bunları yapabileceğinizi bilmeniz için tasarlanmıştır.

Haklarınızı korumak için, başkalarının bu haklarınızı inkâr etmesini veya sizden bu haklardan vazgeçmenizi istemesini engellememiz gerekir. Bu nedenle, yazılımın kopyalarını dağıttığınız veya değiştirdiğiniz taktirde bazı sorumluluklarınız vardır: diğerlerinin özgürlüklerine karşı olan sorumluluklar.

Örneğin, öyle bir programın kopyalarını ister ücretsiz ister ücretli dağıttığınız taktirde, sizden o programı alanlara da programı alırken elde ettiğiniz özgürlükleri geçirmeniz gerekir. Onların da kaynak kodu aldığını veya isterlerse alabileceklerini garantilemeniz gerekir. Ve onların haklarından haberdar olmaları için onlara bu mukaveleyi göstermelisiniz.

GNU GPL kullanan geliştiriciler haklarınızı iki aşamalı olarak korur: (1) yazılım üzerindeki hakları beyan etmek, ve (2) size yasal olarak kopyalama, dağıtma ve/veya değiştirme hakkı veren bu Lisans'ı sağlamak.

Geliştiricilerin ve yayımcıların korunması için, GPL bu özgür yazılım için herhangi bir garanti bulunmadığını net bir şekilde açıklar. Hem kullanıcıların ve hem de yayımcıların iyiliği için, GPL değiştirilen sürümlerde bulunan problemlerin yanlış bir şekilde önceki sürümlerin yayımcılarına atfedilmemesi için değiştirilen sürümlerin değiştirildiklerine dair işaretlenmelerini gerektirir.

Bazı aygıtlar, aygıt üreticilerinin aygıt içindeki yazılımın değiştirilmiş hallerini aygıta yüklemesi veya çalıştırması mümkün olduğu halde, kullanıcıların bunu yapmasına izin vermeyecek şekilde tasarlanmıştır. Bu temelde kullanıcıların yazılımı değiştirme özgürlüklerini koruma amacıyla örtüşmemektedir. Bu suistimal bireylerin kullanımı için olan ürünlerde sistematik olarak uygulanmakta ve görülmektedir, ki en kabul edilemez olduğu yer de budur. Bu nedenle, bu ürünlerin bu uygulamasına mani olmak için GPL'nin bu sürümünü hazırladık. Benzeri problemler başka alanlarda da kendini gösterdiği taktirde, buradaki tedarigimizi GPL'nin gelecekteki sürümlerinde, kullanıcıların özgürlüklerini koruma ihtiyacına binaen, o alanlara da genişletmeye hazırız.

Son olarak, her program yazılım patentlerinin tehdidi altındadır. Devletler patentlerin genel amaçlı bilgisayarlarda yazılım kullanımını ve geliştirilmesini kısıtlayan patentlere izin vermemelidir, ama izin verenlerde özgür bir yazılıma uygulanan patentlerin onu uygulamada sahipli hale getirmesi tehlikesinden kaçınmak istiyoruz. GPL, bunu engellemek için, programın özgürlüğünü kısıtlayan patentlerin kullanılamamasını kesinleştirmektedir.

Kopyalama, dağıtma ve değiştirmeyle ilgili detaylı şart ve koşullar aşağıdadır.

ŞART ve KOŞULLAR
Tanımlar.

“Bu Lisans” GNU GPL'nin üçüncü sürümünü ifade eder.

"Telif Hakları" (“Copyright”) aynı zamanda diğer tür iş ve ürünler için geçerli olan telif hakkı benzeri kanunları kapsar.

“Program” bu Lisans ile lisanslanabilen ve telif hakları kapsamına alınabilen her türlü işi ifade eder. "Siz", "Lisans Sahibi" veya "alıcı" olarak ifade edilebilen lisans sahibi, gerçek bir kişi veya kurum olabilir.

Bir işi "değiştirmek" bir işin bir kısmını veya tamamını telif hakkı izni gerektiren bir biçimde tam bir kopyasını oluşturma haricinde uyarlamak veya kopyalamak anlamına gelir. Sonuçta ortaya çıkan işe önceki işin "değiştirilmiş sürümü" veya "değiştirilmiş hali" ya da önceki işi "temel alan" iş denir.

"kapsanan iş" değiştirilmemiş Programı veya Programı temel alan bir işi anlatır.

Bir işi "yaymak" (propagate), onu çalıştırma veya yerel kopyasını değiştirme dışında izinsiz yapıldığında ilgili telif hakkı kanunu ya da kanunları kapsamında sizi dolaylı veya dolaysız olarak telif hakları ihlâlinden mesûl tutacak şekilde o işle birşey yapmak anlamına gelir. Yayma, kopyalamayı, değiştirerek veya değiştirmeden dağıtmayı, halka açmayı ve bazı ülkelerde diğer eylemleri içerir.

Bir işi "taşımak" (convey) başkalarının kopya elde etmesini ya da yapmasını mümkün kılan her türlü yayma işlemidir. Bir kopya transfer edilmeden bir bilgisayar ağı yoluyla bir kullanıcıyla etkileşim taşıma değildir.

Etkileşimli bir kullanıcı arayüzü (1) uygun bir telif hakları bildirimi gösteren ve (2) kullanıcıya işle ilgili (sunulan garantiler dışında) herhangi bir garanti olmadığını, lisans sahiplerinin bu Lisans kapsamında işi devredebileceklerini, ve bu Lisansın bir kopyasına nasıl bakabileceklerini bildiren uygun ve görünür "İlgili Yasal Bildirimler" gösterir. Eğer arayüz kullanıcıya bir menü gibi bir dizi komut veya seçenek sunuyorsa, listede görünür bir öğe bu kriteri yerine getirir.
Kaynak Kod.

Bir işin "kaynak kod"u o işte değişiklik yapmak için tercih edilen biçimini anlatır. "Nesne kod"u bir işin kaynak kod halinde olmayan tüm biçimlerini anlatır.

Bir "Standart Arayüz" bilinen bir standart oluşturma kurumu tarafından tanımlanmış resmî bir arayüzü veya belirli bir programlama lisanı için oluşturulan arayüzlerde o dilde çalışan geliştiriciler arasında yaygın olan arayüzü tanımlar.

Bir çalıştırılabilirin "Sistem Kütüphaneleri" işin tamamı hariç olmak üzere (a) bir Ana Bileşen paketlenirken dâhil edilen ama o Ana Bileşenin bir parçası olmayan, ve (b) işin o Ana Bileşenle kullanılabilmesini mümkün kılan veya halka kaynak kod biçiminde bir uyarlamanın açık olduğu durumda bir Standart Arayüz uyarlamak için kullanılan her şeyi kapsar. Bu bağlamda bir "Ana Bileşen", çalıştırılabilir işin koştuğu -eğer varsa- belirli bir işletim sisteminin elzem bir bileşenini (çekirdek, pencere sistemi vd.), işi üretmek için kullanılan çekirdeği, veya onu koşmak için kullanılan nesne kodu yorumlayıcısını anlatır.

Bir işe nesne kodu biçiminde "Karşılık Gelen Kaynak" nesne kodunu oluşturmak, yüklemek, (çalıştırılabilir bir iş için) koşturmak ve (bu işlemleri kontrol etmek için gereken betikler dahil) işi değiştirmek için gereken tüm kaynak kodları ifade eder. Ancak, İşin Sistem

Kütüphanelerini ve işin parçası olmayan ama bahsedilen işlemleri gerçekleştirmek için kullanılan ve kolay erişilebilir olan genel amaçlı araçları kapsamaz. Karşılık Gelen Kaynak örneğin işin kaynak dosyalarıyla ilintili arayüz tanımlama dosyalarını ve işin altprogramları ve diğer parçaları arasında veri iletişimi veya kontrol akışı gibi işin özellikle gerektirmesi tasarlanmış olan altprogramların (paylaşılan kütüphanelerin ve dinamik bağlanan altprogramların) kaynak kodlarını içerir.

Karşılık Gelen Kaynağın, Karşılık Gelen Kaynağın diğer parçalarından kullanıcıların otomatik olarak oluşturabildiği hiçbir şeyi içermesi gerekmez.

Bir işin kaynak kod halindeki Karşılık Gelen Kaynağı işin kendisidir.
Temel İzinler.

Bu Lisans kapsamında verilen tüm haklar Programın telif hakkı şartı için verilmiştir ve belirtilen koşullar karşılandığı taktirde geri çevrilemez ve geri alınamaz. Bu Lisans değiştirilmemiş Programı koşma konusunda size sağlanan sınırsız izni aşikâr bir şekilde sınırsız tasdik eder. Kapsanan bir işin koşulması sonucunda oluşan çıktı sadece çıktının içeriği kapsanan bir işle örtüştüğü taktirde bu Lisans tarafından kapsanır. Bu Lisans sizin adil kullanım veya eşdeğer diğer uygulamalarınızı telif hakkı kanunları tarafından sağlandığı şekliyle teslim ve tasdik eder.

Kapsanan işlerden taşımadıklarınızı lisansınızın uygun kıldığı şekilde koşulsuz olarak yapabilir, koşabilir veya yayabilirsiniz. Kapsanan işleri sadece size özel değişiklikler yapmaları için veya size özel değişiklikler yaptıkları o işleri koşmak için size imkân sağlamaları amacıyla telif haklarını kontrol etmediğiniz tüm malzemenin yayılması konusunda bu Lisansın şartlarına uyduğunuz sürece başkalarına yayabilirsiniz. Kapsanan işleri yapan veya koşanlar sadece size özel olarak, sizin adınıza, sizin yönlendirmeniz ve kontrolünüz dahilinde, telif hakları size ait malzemenin sizinle olan ilişkilerinin dışında başka bir kopyasını yapmalarına izin vermeyen şartlara uygun olarak bunu yapmalıdır.

Başka herhangi bir durumda yaymaya sadece aşağıda sayılan durumlarda izin verilir. Altlisanslamaya izin yoktur; bölüm 10 bunu gereksiz kılar.
Kullanıcıların Önlemlerin Etkisiz Kılınmasına Karşı Hukuka Dayalı Yasal Haklarının Korunması.

Kapsanan hiçbir iş 20 Aralık 1996 tarihinde kabul edilmiş olan WIPO telif hakları akdinin 11'inci maddesindeki yükümlülükleri yerine getiren tatbiki hiçbir kanun veya benzer önlemlerin boşluklardan yararlanılarak atlatılmasını yasaklayan veya sınırlayan benzeri hiçbir kanun kapsamında etkili bir teknolojik tedbirin bir parçası olarak kabul edilmemelidir.

Kapsanan bir işi taşıdığınızda, atlatmanın bu Lisansın kapsadığı hakların icrası yoluyla etkilenmesi kapsamı dahilinde teknolojik önlemlerin alınmasını menetme yasal hakkınızdan feragat etmiş ve işin kullanıcılarının, sizin veya üçüncü kişilerin teknolojik önlemlerin atlatılmasını menetme yasal haklarının uygulanmasında işin çalıştırılmasını veya değiştirilmesini sınırlama niyetinizi reddetmiş olursunuz.
Aynî Kopyaları Yaymak.

Programın kaynak kodunun aynî kopyalarını aldığınız şekliyle bariz ve uygun şekilde her kopyada uygun bir telif hakkı bilgisi yayınlamak şartıyla herhangi bir biçimde yayabilirsiniz; bu Lisansı ve kısım 7'ye uymak koşuluyla eklediğiniz şartları beyan eden bildirilerin tamamını ve herhangi bir garanti bulunmadığına dair olan bildirilerin tamamını eksiksiz olarak bulundurun; ve tüm alıcılara Programla birlikte bu Lisansın bir kopyasını verin.

Yaydığınız kopyalar için bir ücret talep edebilirsiniz ve/veya bir ücret karşılığında destek veya garanti teklif edebilirsiniz.
Değiştirilmiş Kaynak Sürümlerini Yaymak.

Kısım 4 kapsamında Programı temel alan bir işi veya Programdan işi üretecek değişiklikleri kaynak kod biçiminde şu koşulları da karşılamak şartıyla yayabilirsiniz:

1. İş onda değişiklik yaptığınıza dair gözle görülür bildiriler içermeli ve ilgili bir tarih vermelidir.
2. İş bu Lisans ve kısım 7 altına eklenen şartlar gereğince yayınlandığına dair gözle görülür bildiriler içermelidir. Bu gereklilik kısım 4'teki "tüm bildirileri değiştirmeden tutma" gerekliliğinde değişiklik yapar.
3. İşin bir kopyasını elde eden herkese tüm işi bu Lisans kapsamında bir bütün olarak lisanslamalısınız.Bu Lisans bu sayede kısım 7'de verilecek tüm ek koşullar ile birlikte işin tamamını ve nasıl paketlenmiş olurlarsa olsunlar tüm parçalarını kapsayacaktır. Bu Lisans işin başka herhangi bir şekilde lisanslanmasına izin vermez, ama ayrıca bunun için bir izin aldıysanız onu geçersiz kılmaz.
4. Eğer işin etkileşimli kullanıcı arayüzleri varsa, her biri Uygun Yasal Bildirileri içermelidir; ancak eğer Programda etkileşimli arayüzlerden Uygun Yasal Bildiriler göstermeyen varsa, sizin işinizin bunları gösterecek şekilde değiştirmesi gerekmez.

Kapsanan bir işin doğaları gereği kapsanan işin uzantısı olmayan ve daha büyük bir iş oluşturmak için o işle birleştirilmeyen başka ayrı ve bağımsız işlerle bir depolama veya dağıtım ortamında derlenmiş hali, derlenmiş hâl ve sonuçta ortaya çıkan telif hakları işlerin her birinin izin verdiğinin ötesinde derlenmiş halin kullanıcının erişim veya yasal haklarını sınırlamak için kullanılmadığı taktirde, "yekûn" olarak anılır. Kapsanan bir işin bir yekûne dahil olması bu Lisansın yekûnün diğer kısımlarını kapsamasına neden olmaz.
Kaynak Hâlde Olmayan Biçimleri Yaymak.

Makine tarafından okunabilir Karşılık Gelen Kaynağı da yaymak şartıyla kısım 4 ve 5'te verilen şartlar dahilinde kapsanan bir işi nesne kodu biçiminde bu Lisansın şartları kapsamında şu yollardan biriyle yayabilirsiniz.

1. Nesne kodunu, özellikle yazılım değiş-tokuşu için kullanılan dayanıklı bir fiziksel ortama maktu Karşılık Gelen Kaynakla birlikte olmak şartıyla, (fiziksel bir dağıtım ortamı dahil) fiziksel bir üründe yayabilirsiniz.
2. Nesne kodunu, en az üç yıl ve sizin o ürün modeli için yedek parça veya müşteri desteği sağlamayı teklif ettiğiniz süre boyunca geçerli yazılı bir teklifle birlikte, nesne koduna sahip olan herhangi kişilere (1) bu Lisans tarafından kapsanan üründeki tüm yazılıma Karşılık Gelen Kaynağın bir kopyasını özellikle yazılım değiş-tokuşu için kullanılan dayanıklı bir fiziksel ortamda bu kaynağın yaymanıza karşılık makul bir tutarı aşmayan bir ücret karşılığında veya (2) Karşılık Gelen Kaynağa bir ağ sunucusu üzerinden ücretsiz olarak erişim imkânıyla, (fiziksel bir dağıtım ortamı dahil) fiziksel bir üründe yayabilirsiniz.
3. Karşılık Gelen Kaynağı sağlamaya yönelik yazılı bir teklifin bir kopyasıyla birlikte nesne kodunun müstakil kopyalarını yayabilirsiniz. Bu alternatife sadece bazen ve ticari olmayacak şekilde, ve sadece eğer siz nesne kodunu böyle bir teklifle elde ettiyseniz, kısım 6 madde (b)'ye uygun olarak izin verilir.
4. Tasrih edilmiş bir yerden ücretli veya ücretsiz erişim sunma yoluyla nesne kodunu yayabilirsiniz, ve ek bir ücret talebinde bulunmadan Karşılık Gelen Kaynağa aynı yer yoluyla aynı şekilde eşdeğer erişim sunabilirsiniz. Alıcıların nesne koduyla birlikte Karşılık Gelen

Kaynağın bir kopyasını almalarını şart koşmanız gerekmez. Eğer nesne kodunun kopyalanacağı yer bir ağ sunucusu ise, nesne kodunun yanında Karşılık Gelen Kaynağın nereden edinilebileceğini belirten açık talimatlar bulundurmak koşuluyla Karşılık Gelen Kaynak eşdeğer kopyalama kolaylıklarını sunan ve sizin tarafınızdan veya bir üçüncü kişi tarafından işletilen farklı bir sunucu olabilir. Karşılık Gelen Kaynağın hangi sunucularda barındırıldığından bağımsız olarak, bu gereklilikleri karşılamak gerektiği sürece mevcut ve erişilebilir olduğundan emin olmakla zorunlu olursunuz.

5. Diğer eşdüzey kullanıcıları' (peer) işin nesne kodunun ve Karşılık Gelen Kaynağının kısım 6(b) kapsamında ücret talebi olmaksızın umuma açık olarak nerede bulunduğu konusunda bilgilendirmek şartıyla, nesne kodunu eşdüzeyler / kullanıcılar arası (peer-to-peer) aktarım yoluyla yayabilirsiniz.

Nesne kodunun ayrılabilir bir parçası Nesne kodunun, kaynak kodu bir Sistem Kütüphanesi olarak Karşılık Gelen Kaynağa dahil edilmeyen ayrılabilir bir parçasının, nesne kodu işini yayarken dahil edilmesi gerekmez.

Bir "Kullanıcı ürünü" ya (1) normalde, kişisel ve/veya ailevi amaçla ve/veya evin idamesi amacıyla kullanılan gerçek herhangi bir mülk anlamında bir "tüketici ürünü"dür veya (2) ikametgâhta kullanılmak üzere tasarlanan veya satılan herhangi bir şeydir. Bir ürünün tüketici ürünü olup olmadığını belirlemede tereddüt yaşanan durumlar kapsam dikkate alınarak çözümlenmelidir. Belirli bir kullanıcı tarafından alınan belirli bir ürün için, "normal kullanım" o belirli kullanıcının durumu ve o belirli kullanıcının o ürünü kullanıp kullanmadığına, kullanması beklenip beklenmediğine, veya ne şekilde kullanması beklendiğine bakılmaksızın o sınıftaki bir ürünün tipik ve yaygın kullanımına işaret eder. Bir ürün ürünün ticari, endüstriyel veya tüketici-dışı kullanımları -bu kullanımlar ürünün tek bariz kullanım şekli olmadıkları sürüce- dikkate alınmaksızın bir tüketici ürünüdür.

Bir Kullanıcı Ürünü için "Kurulum Bilgisi" Kullanıcı Ürününün Karşılık Gelen Kaynağının değiştirilmiş bir hâlinden elde edilen Kullanıcı Ürününde kapsanan bir işin kurulumu ve çalıştırılması için gereken yöntemler, prosedürler, yetkilendirme anahtarları veya diğer bilgilerdir. Bilginin değiştirilmiş nesne kodunun çalışabilir kalmasının hiçbir durumunda sırf değişiklik yapıldı diye engellenmediğini ve müdahaleye maruz kalmadığını garantilemesi yeterli olmalıdır.

Bu kısımdaki bir nesne kodu işini bir Kullanıcı Ürünü ile, ürünün içinde veya özellikle o üründe kullanılmak üzere yayarsanız ve eğer yayma sahip olma ve Kullanıcı Ürününü kullanma hakları belirli bir süre için veya süresiz olacak şekilde alıcıya nakledilerek bir aktarımın (aktarımın tavsifi nasıl olursa olsun) parçası olursa, bu kısımdaki Karşılık Gelen Kaynak Kurulum Bilgisine eşlik etmelidir. Ama bu gereklilik ne siz ne de herhangi bir üçüncü kişi değiştirilmiş nesne kodunu Kullanıcı Ürününe kurma veya yükleme kudretini elinde bulundurduğu taktirde (örneğin iş ROM'a yüklendiğinde veya kurulduğunda) geçerli değildir.

Kurulum Bilgisi sunma gerekliliği alıcı tarafından değiştirilmiş veya kurulumuş bir iş ve işin içinde değiştirildiği veya işin kurulduğu ya da yüklendiği Kullanıcı Ürünü için destek servisi, garanti veya güncelleme gerekliliği getirmez. Değiştirme işleminin kendisi maddeten ve olumsuz şekilde ağın işleyişini etkilerse veya ağ içerisinde iletişim için var olan kural veya protokolleri ihlal ederse bir ağa erişim reddedilebilir.

Bu kısma uygun olarak Karşılık Gelen Kaynak yayıldığında ve Kurulum Bilgisi sunulduğunda, alenen belgelendirilmiş ve umuma kaynak kod biçiminde açık olarak uyarlanmış bir biçemde olmalıdır, ve açmak, okumak veya kopyalamak için herhangi bir

parola veya anahtar gerektirmemelidir.
Ek Koşullar.

"Ek koşullar" bu Lisansın şartlarına bir veya daha fazla koşulun istisnasını oluşturarak ilâve edilen şartlardır. Programın tamamına tatbik edilen ek koşullar ilgili kanun kapsamında geçerli olmak koşuluyla bu Lisansa dahil edilmiş kabul edilmelidir. Eğer ek koşullar Programın sadece bir kısmına atfedilmişse, o kısım o koşullar kapsamında ayrıca kullanılabilir, ama Programın tamamı, ek koşullar dikkate alınmaksızın, bu Lisans kapsamındadır.

Kapsanan bir işinin bir kopyasını yaydığınızda, kendi tercihinize bağlı olarak o kopyadan ve istediğiniz herhangi bir kısmından ek koşulları kaldırabilirsiniz. (Belirli durumlarda işi değiştirdiğinizde kaldırılmalarını gerektirecek şekilde ek koşullar yazılmış olabilir.) Kapsanan bir işe sizin tarafınızdan eklenmiş bir malzemeye uygun telif hakları koşullarına sahip olmak veya verebilmek şartıyla ek koşullar koyabilirsiniz.

Bu Lisansın diğer koşullarıyla uyumsuz olmayacak şekilde Kapsanan bir işe eklediğiniz bir malzeme için o malzemenin telif haklarını elinde bulunduranlar tarafından yetkili kılınırsanız bu Lisansın şartlarına şu koşullar kapsamında eklemelerde bulunabilirsiniz:

1. Bu Lisansın 15 ve 16'ncı kısımlarındaki şartlardan farklı olarak taahhüdünüzü sınırlandırabilir veya garanti vermeyebilirsiniz; veya
2. Belirteceğiniz mahsus yasal bildirilerin veya o malzemeye yazarların yaptığı katkıların malzemede veya onu içeren işler tarafından gösterilen Mahsus Yasal Bildirilerde korunmasını şart koşabilirsiniz; veya
3. O malzemenin kökeni hakkında yanlış veya yalan bilgi verilmesini yasaklayabilir veya o malzemenin değiştirilmiş hallerinin ve sürümlerinin asıl hal ve sürümlerinden farklı olduklarını makul şekillerde göstermelerini şart koşabilirsiniz; veya
4. Malzemenin yazarlarının veya lisans sahiplerinin isimlerinin ilan edilme amacıyla kullanılmasını sınırlandırabilirsiniz; veya
5. Malzemeyi yayanlardan o malzemenin (veya değiştirilmiş hal ya da sürümlerinin) lisans sahiplerinin ve yazarlarının mukaveleden doğan ve mukaveleyle alıcıya yüklenen mesuliyet varsayımı çerçevesinde lisanslayanlara ve yazarlarına karşı her türlü zararın ödenmesini ve karşılanmasını talep edebilirsiniz.

Kısıtlayıcı tüm diğer ek koşullar kısım 10 kastı dahilinde "ilave kısıtlama" olarak kabul edilir. Eğer aldığınız Program veya herhangi bir parçası bu Lisansın hükümlerine tabi olduğunu ve ilave kısıtlamalar bulunduğunu belirten bir bildirim taşıyorsa, o koşulu kaldırabilirsiniz. Eğer bir lisans belgesi ilave kısıtlama taşıyor ama bu Lisans kapsamında yeniden lisanslamaya veya yaymaya ruhsat veriyorsa, ilave kısıtlamanın o tür bir yeniden lisanslama veya yayma durumunda devam etmemesi koşuluyla kapsanan bir işe o lisans belgesi hükümlerince malzeme ekleyebilirsiniz.

Kapsanan bir işe bu kısma zıt düşmemek şartıyla koşul eklerseniz, ilgili kaynak dosyalarda o dosyalara hasreden ek koşullarla ilgili bir beyanat veya tatbik edilmesi gereken koşulların nerede bulunabileceğini belirten bir ibare yerleştirmelisiniz.

Kısıtlayıcı veya değil, ek koşullar ayrı yazılan bir lisans biçiminde veya istisnalar olarak belirtilebilir; her iki durumda da yukarıdaki gereklilikler geçerlidir.
Sonlandırma.

Bu Lisansla sağlanan yollar dışında kapsanan bir işi kesinlikle yayamaz ve değiştiremezsiniz.

Yaymak veya değiştirmek için aksi her türlü teşebbüs geçersizdir ve bu Lisans kapsamındaki tüm haklarınızı (11'inci kısım 3'üncü paragrafta sağlanan patent lisansları dâhil) sonlandırır.

Ancak, bu Lisansı ihlal davranışlarınızı keserseniz, belirli telif hakları sahibinden olan lisansınız (a) telif hakları sahibi sarih ve nihaî olarak sonlandırana kadar geçici olarak ve (b) eğer telif hakları sahibinin ihlali takiben 60 gün içinde sizi makûl bir yolla ihlâlinizinden haberdar etmemesi halinde daimi olarak iade edilir.

Ayrıca, belirli telif hakları sahibinden olan lisansınız, telif hakları sahibi sizi makûl bir yolla ihlalinizden haberdar ederse, bu sizin bu Lisansı ihlalinize dair aldığınız ilk ihbarname ise ve ihbarnameyi aldıktan sonra 30 gün içinde ihlali durdurursanız, daimi olarak iade edilir.

Bu kısım kapsamında haklarınızın sonlandırılması bu Lisans kapsamında sizden kopyalar veya haklar alan üçüncü kişilerin lisanslarını sonlandırmaz. Eğer haklarınız sonlandırıldıysa ve daimî olarak iade edilmediyse, kısım 10 kapsamında aynı malzeme için yeni lisans almaya ehliyetsiz olursunuz.
Kopya Sahibi Olmak İçin Kabul Gerekmez.

Programın bir kopyasını almak veya çalıştırmak için bu Lisansı kabul etmeniz gerekmez. Kaynaktan kaynağa aktarım yöntemleri sonucu kapsanan bir işin bir kopyasının edinilmesi için aktarılmasına yardımcı olmak da benzer şekilde kabul etmeyi gerektirmez. Ancak, bu Lisans dışında hiçbir şey size kapsanan herhangi bir işin yayılması veya değiştirilmesi izni vermez. Bu Lisansı kabul etmediğiniz taktirde bu eylemler telif hakları ihlaline girer. Bundan dolayı, kapsanan bir işi yayarak veya değiştirerek, öyle yapabilmek için bu Lisansı kabul ettiğinizi dolaylı olarak belirtmiş olursunuz.
Alıcıdan Alanların Downstream Recipients Otomatik Lisanslanması.

Kapsanan bir işi her aktarmanızda, alıcı bu Lisans kapsamında otomatik olarak lisansın ilk sahibinden o işi koşmak, değiştirmek ve yaymak için bir lisans alır. Üçüncü kişileri bu Lisansa uymaya zorlama sorumluğunuz yoktur.

Bir "varlık aktarım işlemi" bir organizasyonun veya kurumun kendisinin veya tüm mallarının kontrolünün başkasına verilmesi, organizasyon veya kurumun alt bölümlere ayrılması, veya organizasyonların birleşmesi işlemidir. Kapsanan bir işin yayılması bir varlık aktarım işlemi sonucunda gerçekleşmişse, işlemin işin bir kopyasını alan tüm halefler önceki paragraf kapsamında ilgili selefin işle ilgili sahip olduğu veya verebildiği tüm lisansları ve ilgili seleften işin Karşılık Gelen Kaynağına (eğer sahipse veya makûl şekilde alabilecekse) sahip olma hakkını da elde eder.

Bu Lisans kapsamında temin ve tasdik edilen hakların kullanımı üzerine başka herhangi bir kısıtlama getiremezsiniz. Örneğin, bir lisans ücreti, işletme payı, veya bu Lisans kapsamında temin edilen hakların kullanımı için başka herhangi bir ücret yükleyemezsiniz ve Programı veya herhangi bir parçasını yaparak, kullanarak, satışa çıkararak veya ithal ederek import patent haklarının ihlâl edildiğini iddia eden çapraz cross-claim ya da karşı dava counterclaim dahil dava açamazsınız.
Patentler.

Bir "iştirakçi" Programın veya Programı temel alan bir işin bu Lisans kapsamında kullanımına ruhsat veren telif hakları sahibidir. Bu veçhile lisanslanan işe iştirakçinin "iştirakçi sürümü" denir.

Bir iştirakçinin "asli patent mülkiyeti" iştirakçinin sahip olduğu ya da iştirakçi tarafından şu

anda kontrol edilen veya ileride kontrol edilecek olan ve bu Lisans tarafından izin verilen iştirakçi sürümünün yapılması, kullanılması veya satılması yoluyla bir anlamda ihlâl edilecek olan ama iştirakçi sürümünün sadece değiştirilmesi sonucu ihlal edilmesini tüm patent mülkiyetidir. Bu tanımın amacı gereği, "kontrol" bu Lisansın gerekleri ile uyumlu bir biçimde patent alt lisansları verme hakkını içerir.

Her iştirakçi size iştirakçinin asli patent mülkiyeti dahilinde iştirakçi sürümünü yapmak, kullanmak, satmak, satılığa çıkarmak, ithal etmek, koşmak, içeriğini değiştirmek ve yaymak için size has olmayan, cihanşümul ve işletme payı içermeyen patent lisansı sağlar.

Takip eden üç paragrafta, bir "patent lisansı" bir patenti mecburi olarak tatbik ettirmemek için ismi konmamış her türlü hususi mukavele ve taahhüdü (bir patenti tatbik etmek için hususi izni veya patent ihlalinden dolayı dava etmeme akdini) ifade eder. Bir üçüncü kişiye böyle bir patent lisansını vermek o üçüncü kişiye karşı patentin tatbiki zorunluluğunu kaldıracak bir mukavele yapma veya taahhütte bulunma anlamına gelir.

Bilinçli olarak bir patent lisansına dayanarak kapsanan bir işi taşırsanız ve işin Karşılık Gelen Kaynağı umuma açık bir ağ sunucusu veya diğer kolayca erişilebilir yollardan ücretsiz ve bu Lisansın şartları kapsamında umuma açık olmazsa, ya (1) Karşılık Gelen Kodu bahsi geçen şekilde açık olmasını sağlamalısınız, ya (2) kendinizi o iş için patent lisansının sağladığı yarardan mahrum kılmayı ayarlamanız, ya da (3) bu Lisansın gerektirdikleriyle uyumlu bir biçimde patent lisansını alıcıdan alanlar için genişletmeniz gerekir. "Bilinçli olarak dayanmak" patent lisansına rağmen kapsanan işi bir ülkede yaymanızın veya alıcınızın kapsanan işi bir ülkede kullanmasının o ülkede varolan ve geçerli olduğuna inandığınız bir veya daha fazla patenti ihlal edeceği fiili bilgisine sahip olmanız demektir.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

Bir patent lisansı, kapsamında bu Lisansla sağlanan hakların bir veya daha fazlasını taşımıyorsa, kullanılmasını yasaklıyorsa veya kullanılmamasını şart koşuyorsa "ayrımcıdır". Eğer yazılım dağıtma işinde olan üçüncü bir kişiyle işi taşıma eyleminizin kapsamı dahilinde üçüncü kişiye ödeme yaptığınız ve kapsanan işi sizden alacak kişilere (a) tarafınızdan taşınmış kapsanan iş kopyalarıyla (veya onlardan yapılan kopyalarla) ilgili veya (b) kapsanan işi içeren belirli ürün veya derlemeler için ve onlarla ilgili ayrımcı bir patent lisansı veren bir mukavele yapmışsanız, siz o anlaşmaya girmemişseniz veya o patent lisansı 28 Mart 2007'den önce verilmemişse, kapsanan bir işi taşıyamazsınız.

Bu Lisanstaki hiçbir şey ima edilen herhangi bir lisans veya ihlal savunmasını uygulanabilir patent kanunları kapsamı dışında hariç tutuyor veya sınırlıyormuş gibi algılanmamalıdır.
Başkalarının Özgürlüğünü Teslim Etmeme.

Eğer size bu Lisansın şartlarıyla uyuşmayan bazı zorunluluklar yüklenirse (bir mahkeme kararı, mukavele veya başka suretle), bu zorunluluklar sizi bu Lisansın şartlarından muaf kılmaz. Eğer kapsanan bir işi bu Lisans kapsamında ve süresince sürekli olarak karşılayabilecek şekilde taşıyamıyorsanız, bunun sonucu olarak, onu taşıyamazsınız. Örneğin, Programı taşıdıklarınızdan telif hakkı ücreti almanızı zorunlu kılan şartları kabul ettiyseniz, hem o şartları hem de bu Lisansı yerine getirmenin tek yolu Programı taşımaktan tamamen

çekilmektir.
GNU Affero GPL (Genel Kamu Lisansı) ile Kullanım.

Bu Lisansın koşullarıyla ilintili olarak, kapsanan herhangi bir işi GNU Affero GPL'nin (Genel Kamu Lisansı) 3'üncü sürümüyle lisanslanmış bir işle birleştirip veya bağlayıp tek bir birleştirilmiş iş ortaya çıkarabilir ve sonuçta ortaya çıkan işi taşıyabilir veya devredebilirsiniz. Bu Lisansın koşulları kapsanan iş olan kısma tatbik edilmeye devam edecektir, ama GNU Affero GPL'nin (Genel Kamu Lisansı) bir ağ üzerinden etkileşimle ilgili olan 13'üncü kısmı birleşime tatbik edilecektir.
Bu Lisansın Gözden Geçirilmiş Sürümleri.

Free Software Foundation zaman zaman GNU GPL'nin (Genel Kamu Lisansı) gözden geçirilmiş veya yeni sürümlerini yayımlayabilir. Bu yeni sürümler temel düşünce olarak şu andaki sürüme benzer olacaktır, ancak detayları yeni problemleri veya kaygıları gidermek amacıyla farklı olabilir.

Her sürüme onu diğerlerinden ayıran bir sürüm numarası verilir. Eğer Program GNU GPL'nin (Genel Kamu Lisansı) belirli bir sürümü "veya daha sonraki sürümlerinden biri" kapsamında olduğunu belirtiyorsa, o belirli sürümün veya Free Software Foundation tarafından daha sonra yayımlanan herhangi bir sürümün şart ve koşulları kapsamında hareket edebilirsiniz. Eğer Program GNU GPL (Genel Kamu Lisansı) için herhangi bir sürüm numarası belirtmezse, Free Software Foundation tarafından yayımlanan herhangi bir sürümü seçebilirsiniz.

Eğer program bir vekilin GNU GPL'nin (Genel Kamu Lisansı) hangi ilerki sürümlerinin kullanılabileceğine bir vekilin karar verebileceğini belirtiyorsa, o vekilin bir sürümün kabulüne dair umumî beyanatı sizi Program için o sürümü seçmeye kalıcı olarak yetkili kılar.

Daha sonraki lisans sürümleri size ek veya farklı izinler verebilir. Ancak, daha sonraki bir sürüme göre hareket etme kararınız sonucunda hiç bir müellif veya telif hakları sahibi üzerine ek mecburiyetler yüklenmez.
Garanti Feragatnamesi.

PROGRAM İÇİN, İLGİLİ KANUNLARIN İZİN VERDİĞİ ÖLÇÜDE, HERHANGİ BİR GARANTİ YOKTUR. AKSİ YAZILI OLARAK BELİRTİLMEDİĞİ MÜDDETÇE TELİF HAKKI SAHİPLERİ VE/VEYA BAŞKA ŞAHISLAR YAZILIMI "OLDUĞU GİBİ", SATILABİLİRLİĞİ VEYA HERHANGİ BİR AMACA UYGUNLUĞU DA DAHİL OLMAK ÜZERE AŞİKAR VEYA ZIMNEN HİÇBİR GARANTİ VERMEKSİZİN DAĞITMAKTADIRLAR. YAZILIMIN KALİTESİ VEYA PERFORMANSI İLE İLGİLİ TÜM RİSKLER SİZE AİTTİR. YAZILIMIN KUSURLU OLDUĞU ORTAYA ÇIKTIĞINDA DAHİ, HERHANGİ BİR KUSURDAN DOĞABİLECEK OLAN GEREKLİ BÜTÜN SERVİS, TAMİR VEYA DÜZELTME MASRAFLARI SİZE AİTTİR.
Mesuliyetin Sınırlanması.

İLGİLİ KANUNUN İCBAR ETTİĞİ VEYA YAZILI ANLAŞMA DAHİLİNDEKİ DURUMLAR HARİCİNDE HERHANGİ BİR ŞEKİLDE TELİF HAKKI SAHİBİ VEYA YUKARIDA İZİN VERİLDİĞİ ŞEKİLDE YAZILIMI DEĞİŞTİREN VEYA AKTARAN HERHANGİ BİR KİŞİ, YAZILIMIN KULLANIMI VEYA KULLANILAMAMASI (VEYA VERİ KAYBI OLUŞMASI, VERİNİN YANLIŞ, KULLANIŞSIZ, KULLANILAMAZ VEYA BOZUK HALE GELMESİ, SİZİN VEYA ÜÇÜNCÜ ŞAHISLARIN ZARARA UĞRAMASI VEYA YAZILIMIN BAŞKA YAZILIMLARLA BERABER ÇALIŞMAYI BAŞARAMAMASI) YÜZÜNDEN OLUŞAN GENEL, ÖZEL, DOĞRUDAN YA DA

DOLAYLI HERHANGİ BİR ZARARDAN, BÖYLE BİR TAZMİNAT TALEBİ TELİF HAKKI SAHİBİ VEYA İLGİLİ KİŞİYE BİLDİRİLMİŞ OLSA DAHİ, SORUMLU DEĞİLDİR.
Kısım 15 ve 16'nın Yorumlanması.

Eğer yukarıda belirtilen garanti feragatnamesi ve taahhüt sınırları şartları dikkate alınarak yerel ortamda yasal etki dahilinde uygulanamıyorsa, konuyu ele alan mahkemeler, Programın bir kopyası ile birlikte bir ücret karşılığı olarak bir garanti veya mesuliyet kabulü gelmediği taktirde, Programla ilintili tüm bireysel mesuliyetlerden tam bir feragata en yakın yerel kanunları uygulamalıdır.