Multichannel signal feedback with file input applied to cRIO testing

Axelsson, J; Hägg, Jakob

2010

CODEN:LUTEDX/(TEIE-7236)/1-8/(2010)

Industrial Electrical Engineering and Automation

Multichannel signal playback with file input applied to cRIO testing

Jon Axelsson och Jakob Hägg

Dept. of Industrial Electrical Engineering and Automation
Lund University

EIE061

Projekt i industriell elektroteknik och automation

Delområde: elkraftsystem

Multichannel signal playback with file input

applied to cRIO testing

Jon Axelsson
et06ja3@student.lth.se

Jakob Hägg
et06jh9@studentlth.se

Handledare:
Magnus Akke

Innehållsförteckning

Inledning

Projektet har gått ut på att göra ett system kapabelt att simultant spela upp 8 ljudfiler på en PC via ett standardljudkort. Projektet har omfattat både mjukvara som låter användaren konfigurera beteende via en kontrollpanel samt hårdvara i form av en patch-panel som förenklar inkoppling i laboratoriemiljö.

Projektet har dessutom utvidgats med att implementera två logiska ingångar till progammet, detta via samma ljudkort. Detta möjliggör extern triggning av händelser i programmet.

Målet med systemet är att kunna testa ett realtidssystem avsett för att detektera fel i elkraftstationer, det spelar helt enkelt upp fel som blivit inspelade tidigare. Tanken är att man ska kunna kontrollera så att realtidssystemet fungerar korrekt.

Prestanda

Ljudkortet är ett Sound blaster Audigy. Detta är tänkt att kunna användas i ett 7.1 system och har alltså 8 utgångar. Det är dessutom försett med 2 ingångar avsedda för inspelning. Alla in- och utgångar klarar en samplingshastighet på 96 kHz och en upplösning på 24 bitar, LabView begränsar dock upplösningen till 16 bitar. Detta uppfyller dock fortfarande kraven som är 44.1 kHz och 16 bitar.

Hårdvaran består av en anslutningslåda med banankontakthylsor för respektive utgång. Mellan anslutning och ljudkort sitter ett motstånd på 100 Ω för att skydda ljudkortet vid kortslutning. Ljudkortet är kapabelt att lämna ± 3 volt, men utsignalen bör begränsas till ca $\pm 2{,}5$ volt för att garantera att distortionen hålls på låg nivå. ± 2 volt används som systemets maximala utsignal. På lådan finns dessutom 2 triggingångar, dessa ger tillsammans med tillhörande mjukvara möjlighet till extern triggning av händelser i programmet. Då ljudkortets ingångar är högpassfiltrerade för att undertrycka likspänning så används en oscillator med en grundfrekvens på ca 1 kHz. Denna kan genom att logisk etta läggs på respektive triggingång skickas till respektive ljudkortsingång. Detta kan sedan detekteras av mjukvaran, för närvarande används inte ingångarna i felspelarprogrammet. Ingången är användbar upp till omkring 50 Hz.

Ingång för extern triggning

För att skapa en signal som ljudkortet kan detektera används en oscillator. Med hjälpa av två externa logiska styrsignaler kan utsignalen från denna skickas till ljudkortets respektive ingångar. Då oscillatorn och styrningen är byggd med standardgrindar ska styrsignalerna ha en nivå på 0 eller 5 volt. Detta innebär även att oscillatorn lämnar fyrkantsvåg med en amplitud på 5 volt sett från jord. För att anpassa detta till ljudkortets nivåer så delas detta ner och filtreras till ca $\pm 1{,}25$ V.

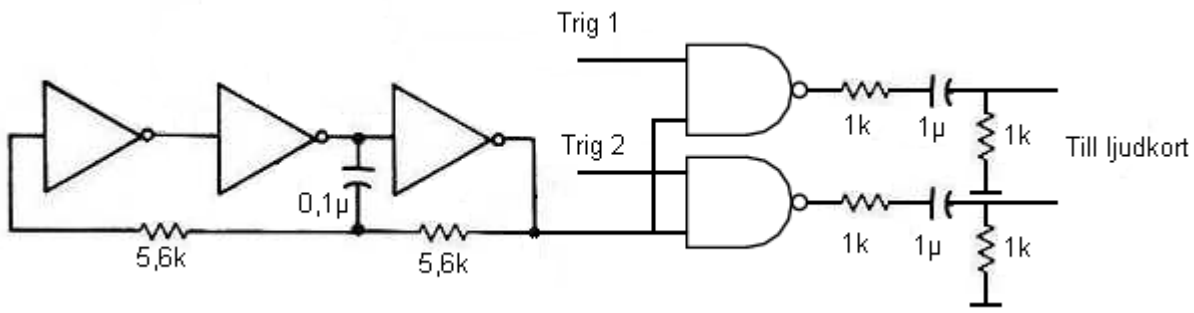

Bild 1: Kopplingsschema.

Komponenterna i oscillatorn har valts för en frekvens på ca 1 kHz.

Mjukvara

Felspelaren är utvecklad i National Instruments program LabView. Att LabView valdes berodde på att det var enkelt att få access till ljudkortet samt att det är ett relativt enkelt språk att skriva i. I DLAB används dessutom LabView för att

programmera cRIO enheterna samt styrprogram till cRIO enheterna. Detta innebär att det finns mycket kunskap inom LabView-programmering vilket är en fördel vid eventuell vidareutveckling/felsökning.

Olika filformat som indata i programmet

Programmet använder sig av National Instruments TDMS format vid uppspelning, som indata går det dock att skicka in ytterligare två typer nämligen zip-filer och binära filer. Båda dessa filformat omvandlas sen till TDMS. De binära filernas uppbyggnad är efter egen upplaga och omvandlas till TDMS och sparas undan i vald mapp. Zip-filerna innehåller komprimerade TDMS-filer som packas upp och sparas i en vald mapp.

Felspelaren – programuppbyggnad

Huvudstrukturen för felspelarens program kan ses i bild 2 nedan. Den övre loopen är avbrottshanteraren och i den undre finns alla avbrottsrutinerna. Ett avbrott kan till exempel vara att användaren trycker på startknappen, i den övre loopen kommer då motsvarande fall för startknappen att aktiveras och lägga en uppgift uppgiftskön. I den nedre loopen finns det ett VI som ligger och kontrollerar kön var tionde millisekund och om det finns uppgifter att utföra kommer de att exekveras i den turordningen som uppgifterna kommit in.

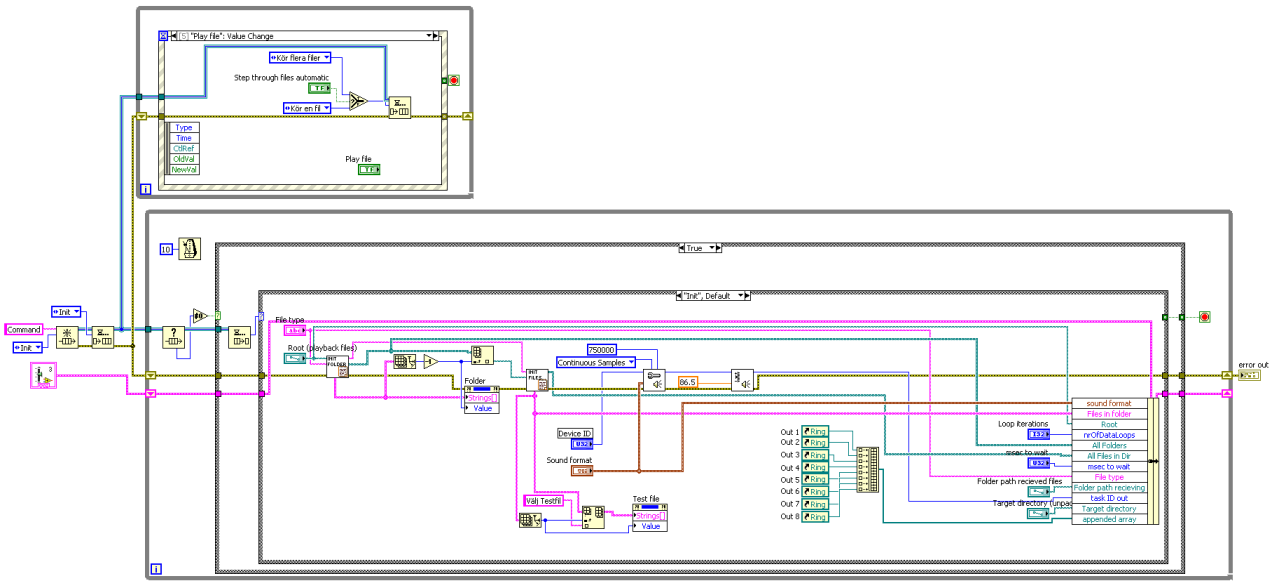

Bild 2: Blockdiagram för felspelaren.

I början när programmet startas läggs några initieringsuppgifter till i början av kön. I initieringen ställs en del av knapparna på frontpanelen in, den ställer in ljudkortet efter valda parametrar och alla variabelvärden som måste finnas tillgängliga för de olika uppgifterna sparas i ett kluster.

De olika uppgifterna som finns implementerade för närvarande är *Init, Välj mapp, Kör en fil, Kör flera filer, Välj testfil, Channels, Sound format, Init sound, Init variables, Data div* och *Avsluta*. Uppgifterna kommer att utföras beroende på vilka ändringar användaren utför på frontpanelen, dessa ändring kommer då att skapa avbrott. Denna typ av programuppbyggnad gör det väldigt enkelt att lägga till och ta bort uppgifter. Om man kommer på att man behöver fler typer uppgifter som ska kunna utföras så är det endast att lägga till ett namn på uppgiften i den namnvektorn syns allra först i programmet (där kön skapas). Uppgiften kan sedan läggas till som ett *case* i case-satsen i den nedre loopen och där programmeras sedan själva uppgiften. Case-satsen i den nedre loopen består alltså av alla uppgifter som kommer att behövas i programmet.

Alla avbrott som sker är enbart användarrelaterade vilket alltså innebär att det inte finns några timers eller andra saker som generar avbrott.

Nedan följen en kortfattad beskrivning av varje uppgift.

Init

I denna rutin ställs *Folder*-listan och *Test file*-listan in efter den sökväg som man angett i *root* samt efter vilken filtyp som valts. Den ställer också in ljudkortet med några initiala inställningar och skickar referenser för *channels* till klustret som innehåller alla variabler som rutinerna delar på.

Välj mapp

Denna rutin kommer att köras då användaren ställer in vilken mapp i *Folder* listan som ska användas för att hämta filer. Med andra ord så uppdateras *Test file*-listan efter det att användaren ställt in det nya värdet.

Kör en fil

Det första som utförs i rutinen är att den skapar den waveform-datan som används som indata till det VI som skickar data till ljudkortet. Datan skapas utifrån vald fil och *channels*. Rutinen uppdaterar också grafen som finns på frontpanelen samt dess legend palett.

Kör fler filer

I rutinen så skapas först början på loggen, därefter stängs en del av knapparna på frontpanelen av, detta är mer en kosmetisk detalj. Det spelar ingen roll om någon knapp eller lista skulle ändras då detta först skulle uppdateras efter alla filer är uppspelade. Efter detta så körs varje fil för sig och loggen uppdateras allt eftersom filerna körs.

Välj testfil

I denna rutin uppdateras en del variabelvärden som behövs vid uppspelning av vald fil.

Channels

Denna rutin uppdaterar variabelvärden som används för att hålla reda på vilken data som ska spelas upp i de olika kanalerna.

Sound format

Denna rutin uppdaterar variabelvärden som används vid inställning av ljudkortet.

Init sound

Rutinen uppdaterar inställningar för ljudkortet och även inställningar för hur den ska spela upp.

Init variables

Här uppdateras ett antal olika variabelvärden i klustret.

Data div

Här uppdateras Data div variabelns motsvarande variabel i klustret.

Avsluta

Tar bort anslutningen till ljudkortet och avslutar uppgiftsloopen och avslutar därmed exekveringen av hela programmet.

Felspelaren - användargränssnitt

Programmet kan användas på två sätt, antingen så skickas en fil i taget manuellt av användaren eller så kan programmet stega igenom alla filer i en vald mapp samt ta emot filer som skickas tillbaka och behandla dessa på ett lämpligt vis. I det automatiska läget skapas också en logg över filer som spelats upp och filer som tagits emot. Bild 3 nedan visar frontpanelen för spelaren, här finns det ett antal inställningsmöjligheter och de olika delarna av gränssnittet beskrivs nedan.

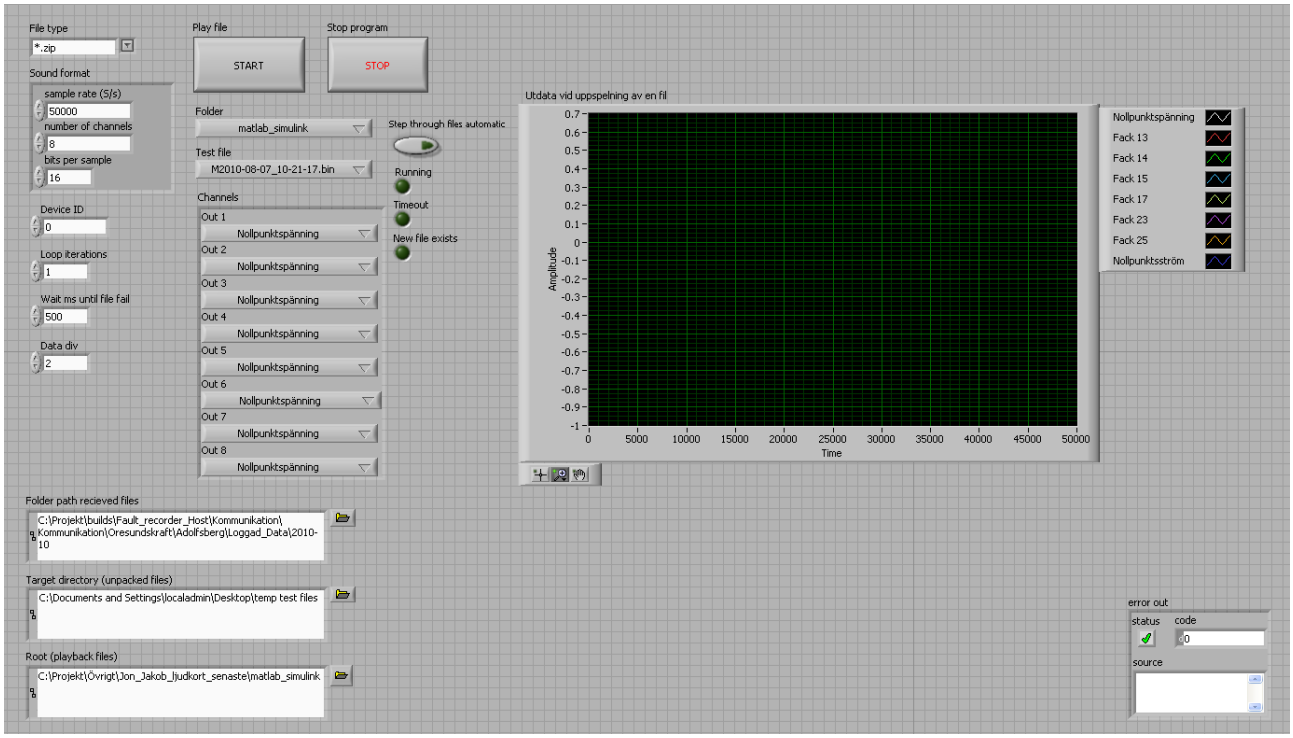

Bild 3: Frontpanel för felspelaren.

File type

Anger vilken filtyp som programmet kommer att leta efter. Enbart filer av denna typ kommer att visas i *Test file*-listan.

Sound format

Kluster som används för att ställa in ljudkortet, här anges uppspelningstakten, antal kanaler samt upplösningen.

Device ID

Ljudkortets enhetsnummer, i detta fall är Device ID satt til noll för soundblaster-kortet. Det har hänt att Device ID har ändrats mellan ljudkorten i datorn därför ges möjligheten att ändra Device ID tills dess att rätt ljudkort hittats. I normalfallet ska värdet vara satt till noll.

Wait ms until file fail

Här anger användaren hur lång tid som programmet ska vänta på att ta emot mätdata från cRIO-enheten som testas. Skulle tiden gå ut sker det ett timeout och detta anges i loggen över den automatiska uppspelningen och nästa fil spelas upp.

Data div

Största möjliga nivå för insignaler till det VI som används för att spela upp datan är ± 1 och då mätdatan från cRIO enheterna kan ha nivåer på ± 10 används *Data div* för att skala ner amplituden på datan till nivåer under ± 1 .

Loop iterations

Om användaren vill att en eller flera filer ska köras kontinuerligt flera gånger går det att ställa in detta med *Loop iterations*. Detta innebär att data från samma fil kan spelas upp flera gånger efter vartannat. Normalfallet är satt till ett men vid kontroll av utsignaler från ljudkortet kan det vara praktiskt att köra en fil ett flertal gånger om man till exempel vill mäta direkt med oscilloskåp på utgångarna.

Folder path for recieved files

Denna sökväg anger mappen där mätdatan som hämtas från cRIO hamnar.

Target directory (unpacked files)

Sökväg som anger mappen där zip- och bin-filer packas upp och sparas som TDMS-filer.

Root (playback files)

Anger sökväg till mapp där filerna finns som ska spelas upp, här sparas också loggen som skapas vid automatisk uppspelning.

Play file

Knapp för att starta uppspelning av antingen en eller flera filer.

Stop program

Knapp för att stoppa program, om uppspelning av filer pågår kommer programmet att stanna först efter att alla filer är uppspelade.

Channels

Här väljs vilken data som ska spelas upp i de olika kanalerna, kanalnamnen är samma som på patch-panelen.

Step through files automatic

Om denna knapp är intryckt (vilket innebär att den lyser grönt) kommer programmet att vid tryck på startknappen stega igenom alla filer som finns i den valda mappen.

Running

Dioden är aktiverad under automatisk uppspelning och stängs av då alla filerna är färdiguppspelade.

New file exists

Dioden visar om mätdata tagits emot inom timeout-tiden.

Timeout

Dioden visar om mätdata inte tagits emot inom timeout-tiden.

Utdata vid uppspelning av fil

Grafen kan används för att visualisera uppspelade filer på skärmen, denna funktion fungerar enbart vid uppspelning av filer manuellt. Amplituden är skalad för att motsvara utspänningen på ljudkortet och om amplituden ligger över 2 V måste amplituden på indatan skalas ner. Detta då signalen annars kan komma att klippa. Amplituden kan ställas in med hjälp av *Data div.* Till höger om grafen syns legend paletten vilken talar om vilken färg de olika signalerna för respektive kanal har i grafen.

Manuell uppspelning en fil i taget

Då programmet startas måste användaren bestämma sig för för vilka inställningar som ljudkortet ska använda sig av, normalfallet är redan inställt dvs en samplingsfrekvens på 50 kHz, 8 kanaler, 16 bitars upplösning samt vilket ljudkort som ska användas (vilket ställs in med *Device ID*.)

Nästa sak som kan vara lämpligt att kolla upp är *File type* som ska sättas till den typ av fil som man tänkt att använda

för att hämta data från.

Sökvägen för mappen där alla filer hämtas ifrån eller en undermapp till denna ska ställas in i *Root. Target directory* ska också ställas in där användaren vill att eventuella filer som packas upp ska sparas.

Det går nu också att ställa in önskad mapp med *Folder*-listan och användaren ombeds också att välja vilken testfil som ska användas. Detta görs med *Test file*-listan. När både mapp och testfil är vald kan alla kanaler ställas in med *Channels*. Sist kan det vara bra att kontrollera att inte *step through files automatic* är aktiverad.

Nu är det viktigaste inställt för att kunna testköra vilket görs genom att startknappen trycks in. Nu kommer utdatan att synas på grafen och om någon av signalerna på grafen skulle ha en amplitud över 2 V så måste storleken reduceras. Detta görs med hjälp av *Data div*. Alternativt kan det vara så att signalerna har en amplitud långt under 2 V, då kan det också vara lämpligt att ändra *Data div* så att så mycket som möjlighet av ljudkortets upplösning används.

Det kan också vara lämpligt att kontrollera att *loop iterations* är inställd på önskat värde (default är 1.)

Automatisk uppspelning av flera filer

Om flera filer ska spelas upp är det samma procedur som vid manuell uppspelning som gäller förutom att automatikknappen måste vara intryckt samt att sökvägen för mottagna filer ska rätt inställd vilket görs med *Folder path recieved files*. Det enda utöver detta är möjligtvis att ställa in *Wait ms until file fail* som anger hur lång tid som datorn ska vänta på att ta emot filer. Det är viktigt att tänka på att inte ställa denna alltför lågt, det kan vara så att det blir väldigt stora filer som ska sparas undan i cRIOn vilket kan ta väldigt lång tid. Därför är det bättre att verkligen sätta tiden i överkant så att cRIOn hinner med.

Data från MATLAB till felspelaren

Om inspelade filer har konverteras för användning i MATLAB eller om egna testfiler har skapats så går det att konvertera datan till TDMS format. Detta kan till exempel göras genom att spara undan data från arbetsminnet i MATLAB. Då kommer datan att lagras i filer med .mat format. Om variabelnamn är valda på ett korrekt sätt går det sedan att omvandla denna data till en annan form av binär fil. Denna binära fil kan sedan användas som indata i felspelaren som i sin tur omvandlar den till TDMS-format. För att omvandla från .mat filerna till den andra formen av binär fil används m-funktionen intobin där invariabeln är namnet på filen i form av en sträng, till exempel intobin('M2010-XX-XX_XX-XX-XX.mat'). I samma mapp som .mat filen ligger kommer sedan en fil med samma namn fast med .bin som ändelse att skapas. Om det finns flera .mat filer som man vill omvandla går det att använda sig av m-scriptet allintobin vilket tar alla .mat filer i den aktuella mappen och omvandlar dessa till bin filer med motsvarande namn.

Slutsats och möjliga förbättringar

Att använda ljudkort som signalgenerator har stor potential så länge kraven på bandbredd på signalen ryms inom ljudkortets bandbredd. Begränsningen i antal signaler bestäms i princip bara av antalet kanaler på ljudkortet, samt hur många ljudkort som används och vi vet idag inget som hindrar att man skulle använda ett ljudkort med betydligt fler kanaler, till exempel ett ljudkort avsett för studiobruk med 24 kanaler. Ljudkortets ingångar hade kunnat användas för inspelning istället för att ge större flexibilitet. Önskas dessutom samtidigt en eller flera trigingångar så kan till exempel parallellporten användas för detta. En annan förbättring som troligtvis skulle öka användbarheten för det givna applikationsområdet är att förse utgångarna med förstärkare som kan skala signalerna till valbar ström och spänning.