LabVIEW 2

Kékesi Kristóf
NEPTUN kód: ZI6I4M
Mérőpár: Bor Gergő

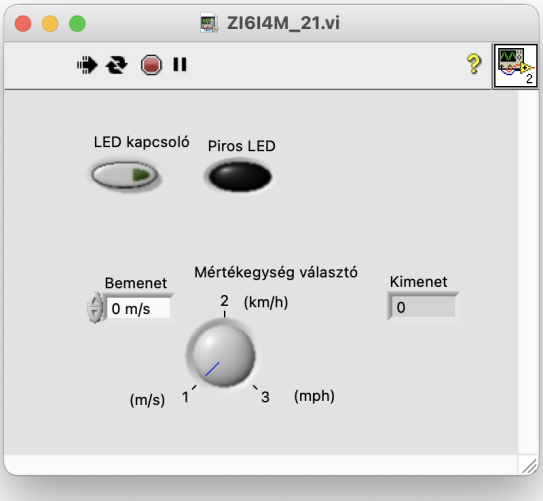
Mérés ideje: 2024.03.17. 15:15-18:00 és 2024.03.20. 15:15-18:00
Mérés helye: Pázmány Péter Katolikus Egyetem, Információs Technológiai és Bionikai Kar
1083 Budapest, Práter utca 50/A 321-es labor
kekesi.kristof.mihaly@hallgato.ppke.hu

***Kivonat*—**
A jegyzőkönyv részletesen leírja a március 13-án és 20-án megoldandó LabVIEW feladatokat, valamint az ezek megoldásához szükséges információkat. A dokumentum célja, hogy átfogó útmutatást nyújtson a feladatok megoldásának folyamatáról és a reprodukálhatósághoz szükséges lépésekről.
Az első feladat a m/s-ről km/h-ra való váltást célozza meg, amelynek megoldásához programot kell létrehozni. Ezután egy változtatható mértékegységre váltó program elkészítése következik többállású kapcsoló segítségével. A harmadik feladat három dobókocka dobásának szimulációját foglalja magában, majd ezt követi egy 10000 dobást szimuláló, majd az eredményeket vizualizáló program. Ezt követően a feladatok között szerepel különböző jelek előállítása, és ezek effektív értékének kiszámolása. Következő feladatként a fázishasítás szimulációját változtatható hasítási ponttal (bekapcsolási pillanattal) kell elkészíteni. Ebben a feladatban olyan programot kell készíteni, amely képes szimulálni a fázishasítás jelenségét különböző bekapcsolási pillanatoknál. Emellett a programnak lehetőséget kell biztosítania a kapott függvény effektív értékének kiszámolására..
A jegyzőkönyv részletesen ismerteti az egyes feladatok megoldásához szükséges lépéseket, beleértve a szükséges eszközök és eljárások használatát is. Ezáltal segíti az azt olvasókat a feladatok hatékony és pontos megoldásában, valamint elősegíti a feladatok reprodukálhatóságát és értelmezhetőségét.

***Keywords*-National Instruments LabVIEW; Virtual Instrument; Front panel; Block panel; subVI; Fázishasítás; Effektív érték;**

Méréssel kapcsolatos fogalmak

- LabVIEW: A National Instruments (NI) által készített Laboratory Virtual Instrument engineering Workbench azaz LabVIEW. Első verziója 1986-ban jelent meg, használata azóta világszerte elterjedt, a világ egyik vezető szoftvere a méréstechnika és automatizálás területén. Számos kiegészítővel működik, ezeket szintén a National Instruments (NI) készíti. Ezekre példa az Elvis I, és az Elvis II panelek.
  A LabVIEW környezetben egy úgynevezett virtuális műszer (Virtual Instrument, VI) megvalósítására van lehetőség grafikus programozási környezetben, de általános célú programok fejlesztésére is felhasználható. A grafikus programozási környezet annyit jelent, hogy a programozás során nem szöveges kód készül, hanem különböző függvényeket és vagy utasításokat reprezentáló elemek, komponensek összekapcsolásával épül fel a program. LabVIEW környezetben folyamat vezérelt, adatfolyam elvű programozásra van lehetőség, a program végrehajtási sorrendjét az utasítások kapcsolódási rendszere határozza meg. Ezeket a kapcsolatokat a Block Diagram nevű panelen lehet beállítani, az úgynevezett vezetékek segítségével. Ezek a vezetékek típus biztosak, szállíthatnak számokat, szöveget, listákat, és még más általános gyakori típusokat.
  A LabVIEW rendelkezik saját fájlformátummal, ez a .vi. A program két fontos része a Block Diagram és a Front panel. Ezeket a paneleket a Connector panel köti össze. [1] [2]
- Front panel: Amikor megnyitunk egy új, vagy létező VI-t, akkor megjelenik egy grafikus felület, másnéven a Front panel. A jobb egérgomb meg nyomásával jelenik meg a Controls panel. Ebből a menüsorból lehet kiválasztani kontrolokat, például egy kapcsolót vagy egy LEDet. A Front panel csak kijelzésre és a felhasználó általi adatbevitelre szolgál, programozási rész és kapcsolók beállítása nem ezen a felületen történik, arra a Block Diagram grafikus felülete szolgál. A program futtatását is a Front Panel ablakán kell elvégezni, a program kontroljait és indikátorait is itt lehet kezelni. [3]
  A Front Panel grafikus megjelenése az 1. és a 4. ábrákon látható.
- Block diagram: Itt történik a VI (Virtual Instrument) programozása. Ezen az ablakon van lehetőségünk beállítani, hogy a Front panelen elhelyezett komponensek, hogyan viselkedjenek például egy gomb megnyomása esetén, milyen karakterisztikával rendelkezzenek. A kom-



1. ábra. A 2. feladat Front Panelje.

ponenseket vezetékekkel összekötve lehet a program logikai működését előállítani. A jobb egérgomb megnyomásával megjeleníthető a 'Functions' eszköztár. Itt lehet különböző beépített függvényeket, operátorokat használni (ezeket subVI-knek hívjuk.). Ilyenek például a operátorok, logikai műveletek, mátrixműveletek, struktúrák, mint a case struktúra, while és for loop és hasonlóak. [4]
A Block Diagram grafikus megjelenése a 3. ábrán látható.

- Connector panel: Ha szeretnénk használni egy VI - t subVI-ként akkor a Connector Panelen a következő módosításokat kell elvégezni. Minden VI-nek van egy ilyen menüje a jobb felső sarokban. Ez arra szolgál, hogy összekössük egy Virtuális Instrument (VI) kontroljait, indikátorait egy másik VI-vel mint a subVI komponens lábai. A Connektor panel, helyes beállításával képesek vagyunk például számokat kinyerni a subVI-ból és eltárolni azokat. [5]
- Indikátor: Az indikátorok arra szolgálnak, hogy megjelenítsük az adatokat, grafikonokat és egyéb információkat, amik szükségesek a programunk működésére vagy irányítására. Ezt a menüpont a Front panel jobb egér kattintásával lehet elérni a 'Controls Palette' nevű menüsorban. Amennyiven elhelyezünk egy indikátort a Front Panelen, akkor ez a Block diagramon is meg fog jelleni. (Ez visszafelé is érvényes, amit a Block Diagrammon helyezünk el indikátor, vagy kontroll, az a Front Panelen is meg fog jelenni.) Amennyiben duplán kattintunk egy adott indikátorra, vagy kontroll komponensre a Front panelelen akkor, automatikuson megnyílik a Block diagram, és fordított esetben is. [6]
- Kontrol: Olyan grafikus elemek, komponensek, amik a Front Panelen és a Block Diagrammon jelennek meg, amikkel a felhasználó interaktálni tud, és ezzel együtt az elkészített program paramétereit módosítja. Egy VI (Virtual Instrument) subVI-ként való használata esetén ezeket bemenetként lehet használni egyenesen a Block Diagramról. [6]
- VI: A LabVIEW programban egy függvényként leírható programozható 'Virtual Instrument' rövidítése. Amikor létrehozunk egy LabVIEW projektet, VI-ken keresztül tudjuk megoldani a feladatainkat, problémáinkat. Ezeknek a fájlkiterjesztése a .vi fájlformátum.
- SubVI: A LabVIEW lehetővé teszi, hogy már elkészített VI-ket használjunk más VI-kben. Ezt az úgynevezett subVI-k segítségével tudjuk elérni. Ehhez a kívánt VI jobb felső sarkában található ikonra kattintva be kell állítani melyik kontrollokat és indikátorokat szeretnénk megjelíneti egy másik VI-ben, amikor az aktuális VI-t subVI-ként használjuk. Erre az ikonra kattintva van lehetőségünk a subVI ikonjának módosítására, a subVI lábkiosztásának tömörítésére és mezőinek szétválasztására. [7]
- Hisztogramm: A hisztogramm egy diagram, amely oszlopok sorozatából áll, amelyek magassága vagy hossza az adott kategóriák vagy értéktartományok gyakoriságát vagy eloszlását mutatja. Általában az x-tengelyen az érték tartománya vagy kategóriák vannak megjelenítve, míg az y-tengelyen az értékek vagy kategóriák gyakorisága vagy eloszlása jelenik meg. A hisztogramm gyakran használt az adatok vizuális elemzésére és ábrázolására, különösen az adatok eloszlásának megértésére.
A hisztogramm grafikus megjelenése a 2. ábrán látható.
- Effektív érték: 'A villamos áram effektív értéke (vagy négyzetes középértéke) az áram hőhatására ad útmutatást. Az effektív érték annak az egyenáramnak az értékével egyenlő, amely azonos idő alatt ugyanakkora munkát végez (hőt termel), mint a vizsgált váltakozóáram.' - [8] [9]
Az effektív értékkel ki tudjuk fejezni, hogy a vizsgált függvénynek mekkora a kiterjedése. Hasznos információ, hogy az effektív érték nagysága nem lehet nagyobb mint a vizsgált függvény amplitudója.
Az effektív érték kiszámításának módját a 1. képleten lehet látni.

$$U_{\text{effektív}} = \sqrt{\frac{1}{T}\int_0^T u^2 dt} \tag{1}$$

- Fázishasítás: A fázishasítás egy olyan fogalom, amelyet gyakran találunk különböző tudományterületeken, például fizikában, matematikában, elektrotechnikában és még a pszichológiában is. Az alapelve, hogy egy adott folyamat vagy jelenség ciklikus, ismétlődő mintázatokat mutat, és ezeket a mintázatokat különböző fázisokra lehet bontani, ahol minden fázisban bizonyos jellemzők vagy tulajdonságok figyelhetők meg.
Fizikai értelemben a fázishasítás általában hullámok vagy rezgések viselkedését írja le. Például egy fázisban levő rezgésfolyamatot általában szinuszhullámokkal vagy koszinuszhullámokkal írják le, és ezeknek a hullámoknak a fázisa az időbeli eltolódásukat jelenti. Az elektromos rendszerekben, például az áramkörök tervezésében vagy a digitális jelek feldolgozásában, a fázishasítás gyakran kulcsfontosságú a kívánt működés elérésében vagy a jellemzők optimalizálásában.
Matematikai értelemben a fázishasítás gyakran kapcsolódik a trigonometriához és a komplex számokhoz. Például a Fourier-transzformáció segítségével egy periodikus jelent lebontanak a különböző frekvenciájú komponenseire, amelyek fázisát és amplitúdóját meghatározzák.
Az elektronikában és a jelek feldolgozásában a fázishasítás fontos szerepet játszik a jelfeldolgozási technikákban. Például az analóg jelek digitálisra alakítása során az ADC-k (Analog-to-Digital Converter) jelentősége a fázishasításon alapul.
Összességében a fázishasítás egy olyan fogalom, amely széles körben alkalmazható és megértése és alkalmazása fontos számos tudományterületen és iparágban. A fázishasítás lehetővé teszi számunkra, hogy jobban megértsük és modellezzük a periodikus vagy ismétlődő folyamatokat, és segítségével optimalizálhatjuk vagy javíthatjuk ezeket a folyamatokat azáltal, hogy megérthetjük és befolyásolhatjuk az egyes fázisokban bekövetkező változásokat.
- Periódikus jelalak: A periódikus jelalak egy olyan jel vagy hullámforma, amely egy meghatározott időközönként ismétlődő mintázattal rendelkezik. Ez azt jelenti, hogy a jel bizonyos időközönként ugyanazt a mintázatot vagy hullámformát ismétli. Példák erre lehetnek a sinuszhullámok, négyzethullámok, háromszöghullámok és a fűrészhullámok.
A periódus hossza az az időtartam, amely alatt a jel egy teljes ciklust teljesít. A periódikus jelalak jellemzően matematikai formulákkal írható le, és a frekvencia,

amely a periódusok számát jelzi másodpercenként, a jel tulajdonságainak fontos mérőszáma.
A periódikus jelalakok sokféle formában megjelenhetnek, például hanghullámokban, elektromos jelekben vagy akár fizikai rezgésekben. Például egy hanghullám esetében a periódus az idő, amely alatt a hanghullám egy teljes ciklust tesz meg, vagyis egy teljes rezgési vagy oszcillációs folyamatot mutat be.
Az elektronikában és a jelek feldolgozásában a periódikus jelalakok fontosak, mivel lehetővé teszik az időbeli mintázatok elemzését és manipulálását. Például a hangrögzítés és lejátszás során a periódikus hanghullámokat digitális jelekbe alakítják át és fordítva, hogy könnyen tárolhatók és továbbítandók legyenek.
A periódikus jelalakok alkalmazása sokféle területen megtalálható, például az akusztikában, a kommunikációs rendszerekben, az elektromos hálózatok tervezésében és még sok más területen. Ezeknek az alakzatoknak a megértése és kezelése lehetővé teszi számunkra, hogy hatékonyan kommunikáljunk és kezeljük az időben ismétlődő jeleket és hullámokat a mindennapi életben és a műszaki alkalmazásokban egyaránt. [10] [11] [12]

I. LabVIEW feladat

Az első feladatban két különálló programot kell elkészítenünk egy LabVIEW VI-ben. Az első program célja egy egyszerű kontroll-indikátor kapcsolat kialakítása, amelynek során egy tolókapcsoló bekapcsolásakor egy piros ovális LED is be kell hogy kapcsoljon. Ezt a kapcsolatot a Block Diagram ablakban kell létrehoznunk, összekötjük a kontrolt az indikátor LED-del. A LED ovális alakját a Front Panelen kell beállítanunk, ehhez rákattintunk a komponens keretére, majd az egérrel elhúzzuk, hogy megkapjuk a kívánt formát.
A feladat második része egy mértékegységváltó program megvalósítása, amely átváltja a m/s-ben megadott sebességeket km/h-ra. Ezt a funkciót a 'numeric' menüpont alatt található 'multiply' komponenssel valósítottuk meg, ami két adott számot megszoroz egymással. Ezt a megfelelő képlettel (2. képlet) használva kapjuk meg az átváltott sebességet.

$$km/h = m/s \cdot 3,6 \tag{2}$$

A Front Panelen ezek után a cimkék átnevezésével érthetőbbé tesszük a programot a program felhasználója számára.

II. LabVIEW feladat

A második feladat az első feladatra épít, azonban némi változással. Ebben a feladatban egy háromállású kapcsolóval lehet beállítani, hogy a mértékegység m/s-ről milyen mértékegységre változzon a megadott mennyiség. A választható mértékegységek a km/h, m/s és a mph. Ehhez használjuk az 2. és 3. képleteket.

$$\text{mph} = \text{m/s} \cdot 2,24. \tag{3}$$

A 3. egyenlet segítségével az mph (miles per hour) mértékegységet váltjuk át m/s-be. Ahhoz, hogy az indikátor több állásba tudjon állni, a Front Panelen elhelyezünk egy olyan kontroll komponenset, amely több lehetőséget kínál, erre a 'Dial'-t választottuk. Ennek beállításai a következők:

- A komponens típusát 'Byte'-ra állítjuk, hogy csak egész számokat fogadjon el.
- A minimális és maximális értékeket 1-re és 3-ra állítjuk, hogy a dial csak a megfelelő intervallumon, 3 értéket vehessen fel.
- A 'Coerce to nearest' opcióval biztosítjuk, hogy csak a legközelebbi egész számok legyenek érvényesek.

Miután a Front Panelt megfelelően beállítottuk, a Block Diagramon fejezzük be a program helyes működését. A 'Dial' komponens kimenetét egy 'Case Structure'-be kötjük, ahol a különböző értékek alapján különböző kódok futnak le. Ez a 'Dial' által kiválasztott mértékegység átváltást jelöli. A 'Case Structure'-ön kívül lévő mennyiséget szorozzuk meg a 'Dial' által kiválasztott mértékegységgel, ami a 'Case Structure'-ön belül található.
Végül, a mértékegység váltási részét egy subVI-be helyezzük át a 'Virtual Instrument'-ben. Fontos lépés a kontrollok és indikátorok megosztása a subVI lábain keresztül. Ezt a jobb felső sarokban lehet beállítani, a subVI ikonjával együtt. Ez lehetővé teszi a subVI belsejében lévő kontrollok és indikátorok elérését az azt hívó VI-n belül.

III. LabVIEW feladat

A harmadik feladatban egy olyan VI-t hozunk létre, amely egy gomb lenyomására három hat oldalú dobókocka dobását szimulálja, majd ezek eredményeit megjeleníti három szám indikátoron. A program felépítése a következő lépésekből áll: Először is, a szükséges indikátorokat és kontrollokat elhelyezzük a Front Panelen, ezek egy gomb, három szám indikátor és egy LED. A címkéket átírjuk, hogy érthető legyen a felhasználó számára a jelentésük, és a gomb karakterisztikáját úgy állítjuk be, hogy csak egyszer adja ki a logikai igaz értéket lenyomás után.
A program felépítése a Block Diagramon való szerkesztésekkel folytatódik. A gomb logikai értékét egy 'Switch Structure'-höz kötjük, ahol igaz érték esetén egy véletlenszerű számot választunk 1 és 6 között, ami egy dobókocka dobását reprezentálja. Ezután ezt az értéket megjelenítjük az adott szám indikátoron. Ezt a folyamatot háromszor ismételjük meg a három egyidejű dobás miatt.
A dobások eredményeit összeadjuk a 'Case Structure'-ön kívül, majd ellenőrizzük, hogy az összeg egyenlő-e 18-cal. Ha igen, akkor bekapcsoljuk a Front Panelen elhelyezett LED-et. Fontos figyelni arra, hogy a 'Case Structure' akkor is adja meg a három értéket, amikor a gomb értéke hamis. Erre a célra a 0 értéket választottuk, hiszen dobókockával nem lehet ezt a számot dobni.
Végül, hogy megértsük az egyes eredmények valószínűségeit, fontos tudnunk, hogy hányféleképpen érhetjük el az adott összeget a három dobókocka dobásakor, ahol minden kocka 1 és 6 közötti számot dob. Ezeknek a valószínűsége:

$$3 = 1+1+1 \quad \text{(1 féle képpen)}$$

$$4 = 1+1+2 \quad \text{(3 féle képpen)}$$

$$5 = 1+1+3 = 1+2+2 \quad \text{(6 féle képpen)}$$

$$6 = 1+1+4 = 2+2+2 = 1+2+3 \quad \text{(10 féle képpen)}$$

$$7 = 1{+}1{+}5 = 1{+}2{+}4 = 1{+}3{+}3 = 2{+}2{+}3 \quad \text{(15 féle képpen)}$$

$$8 = 1+1+6 = 1+2+5$$

$$8 = 1+3+4 = 2+2+4 = 2+3+3 \quad \text{(21 féle képpen)}$$

$$9 = 1+2+6 = 1+3+5 = 1+4+4 = 2+2+5$$

$$9 = 2+3+4 = 3+3+3 \quad \text{(25 féle képpen)}$$

$$10 = 1+3+6 = 1+4+5 = 2+2+6 = 2+3+5$$

$$10 = 2+4+4 = 3+3+4 \quad \text{(27 féle képpen)}$$

$$11 = 1+4+6 = 1+5+5 = 2+4+5 = 3+3+5$$

$$11 = 4+4+3 = 2+3+6 = \quad \text{(27 féle képpen)}$$

$$12 = 6+5+1 = 4+3+5 = 4+4+4 = 5+2+5$$

$$12 = 6+4+2 = 6+3+3 \quad \text{(25 féle képpen)}$$

$$13 = 6+6+1 = 5+4+4 = 3+4+6 = 6+5+2$$

$$13 = 5+5+3 \quad \text{(21 féle képpen)}$$

$$14 = 6+6+2 = 5+5+4$$

$$14 = 4+4+6 = 6+5+3 \quad \text{(15 féle képpen)}$$

$$15 = 6+6+3 = 6+5+4 = 5+5+5 \quad \text{(10 féle képpen)}$$

$$16 = 6+6+4 = 5+5+6 \quad \text{(6 féle képpen)}$$

$$17 = 5+6+6 \quad \text{(3 féle képpen)}$$

$$18 = 6+6+6 \quad \text{(1 féle képpen)}$$

Ezeknek a tudatában behelyettesíthetünk az általános valószínűségszámítás képletbe. Ez a következőféleképpen néz ki (4. képlet):

$$p = \frac{k}{\ddot{o}}, \tag{4}$$

ahol k a kedvező lehetőségek számát, míg ö az összes lehetőségek számát jelöli. Az összes lehetőség száma a következőképp számolható ki (5. képlet):

$$\ddot{o} = 6^3 = 216. \tag{5}$$

Az előző sorokban kiszámolt kedvező lehetőségek alapján kiszámolható az adott számok valószínűsége három hatoldalú dobókocka dobás után kapott számok összegeként:

$$p(3) = \frac{1}{216} \approx 0,004\%,$$

$$p(4) = \frac{3}{216} = \frac{1}{72} \approx 0,01389\%,$$

$$p(5) = \frac{6}{216} = \frac{1}{36} \approx 0,0278\%,$$

$$p(6) = \frac{10}{216} = \frac{5}{108} \approx 0,0463\%,$$

$$p(7) = \frac{15}{216} = \frac{5}{72} \approx 0,069\%,$$

$$p(8) = \frac{21}{216} = \frac{7}{72} \approx 0,097\%,$$

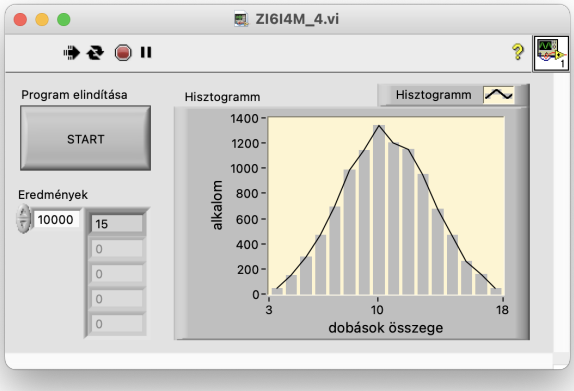
$$p(9) = \frac{25}{216} \approx 0,115\%,$$

$$p(10) = \frac{27}{216} = \frac{1}{8} \approx 0,125\%,$$

$$p(11) = \frac{27}{216} = \frac{1}{8} \approx 0,125\%,$$

$$p(12) = \frac{25}{216} \approx 0,115\%,$$

$$p(13) = \frac{21}{216} = \frac{7}{72} \approx 0,097\%,$$



2. ábra. A 4. feladat Front Panele.

$$p(14) = \frac{15}{216} = \frac{5}{72} \approx 0,069\%,$$

$$p(15) = \frac{10}{216} = \frac{5}{108} \approx 0,0463\%,$$

$$p(16) = \frac{6}{216} = \frac{1}{36} \approx 0,0278\%,$$

$$p(17) = \frac{3}{216} = \frac{1}{72} \approx 0,01389\%,$$

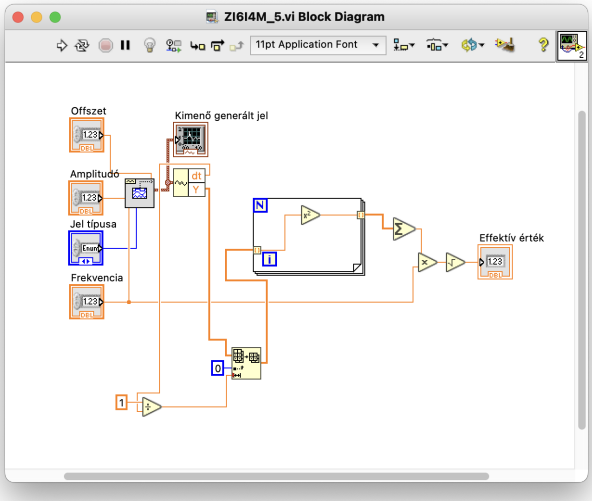
$$p(18) = \frac{1}{216} \approx 0,004\%.$$

IV. LabVIEW feladat

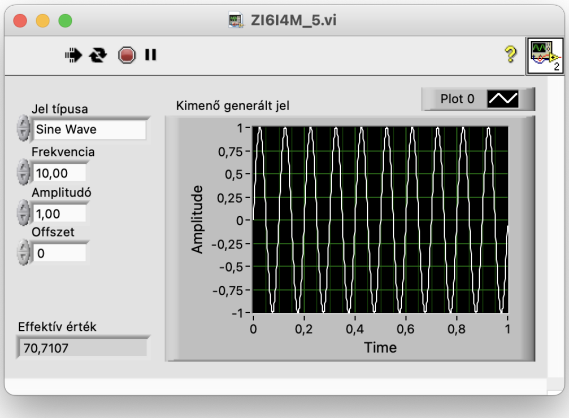
A negyedik feladat egy olyan programot foglal magában, amely a harmadik fejezetben előállított VI-t, vagyis virtuális instrumentumot használja fel, mint subVI. Ehhez módosítanunk kell a harmadik feladatban elkészített programot úgy, hogy az eredetileg a felhasználótól kapott adatok, amiket kontrollként használt a program, most az azt meghívó Virtuális Instrumentumból fog érkezni. Ugyanígy az indikátorokat is kimenetként kell beállítani, hogy azok az alkalmazó Virtuális Instrumentumban megjelenhessenek. Ezt a Front Panel jobb felső sarkában lévő ikonra kattintva tehetjük meg. Az adott kontrollokat és indikátorokat itt helyezhetjük ki a subVI adott lábaira.

Miután sikeresen átalakítottuk az eredeti Virtuális Instrumentet subVI-ként használhatóvá, azt egy nagyobb folyamatba kell illesztenünk. Ezt úgy tehetjük meg, hogy a VI-t egy 'for loop' komponensbe helyezzük, amelyet 10000 iterációra állítunk be konstans szám használatával. Ezáltal a subVI-t többször fogjuk meghívni a kívánt műveletek végrehajtásához, és az eredményeket az egyes iterációk során lehetőségünk lesz megjeleníteni.

Az eredmények megjelenítésére hisztogrammot választottunk (a hisztogramm a 2. ábrán, a 4. feladat Front Panelén látható). Ennek beállításakor figyelembe kell vennünk a várható minimum és maximum értékeket, amelyeket az adatok tartományához igazítunk. Emellett meg kell határoznunk, hogy hány számot várunk a hisztogrammot megjelenítő eszköztől, és ezeket milyen oszlopokban szeretnénk megjeleníteni, hogy az eredmények áttekinthetőek legyenek és könnyen értelmezhetők legyenek az alkalmazó számára. Mivel három hatoldalú dobókozkával dobva a mimimum összeg 3, és a maximum dobható szám 18, ezért a hisztogramm komponenset ezekkel a számokkal kötjük össze. A várható értékek száma 16.



3. ábra. Az 5. feladat Block Diagramja.



4. ábra. Az 5. feladat Front Panelje.

V. LabVIEW feladat

Ebben a feladatban egy olyan programot kell létrehoznunk, amely képes különböző jelek előállítására, például szinusz, négyszög, fűrész és háromszög hullámjelek. A programnak több kontrolt kell tartalmaznia, amelyeken keresztül az előállított jel amplitúdóját, offszetét és frekvenciáját lehet állítani. Emellett a programnak képesnek kell lennie kiszámítani a megjelenített jel effektív értékét is. Ehhez a programot úgy kell kialakítani, hogy először egy listába gyűjtjük a függvény értékeit, majd ezeket felhasználva behejesítjük az effektív érték képletébe (1. képlet), majd elosztuk a kapott értéket a periódus előfordulásának számával a listán belül.
A program Blokk Diagramja és Front Panelje a 3. és a 4. ábrákon láthatóak, amelyek bemutatják a VI struktúráját és felületét, ahogyan az a LabVIEW környezetben megjelenik.

VI. LabVIEW feladat

A hatodik feladatban egy fázishasítás szimulációt kellett elkészíteni, amivel a bekapcsolási pillanat változásával nézzük meg a kimeneti függvény jelalakját, és annak az effektív értékét.

A fázishasításnak célja az effektív érték csökkentése a legegyszerűbb áramkörök felhasználásával. A fázishasítás során két jelet szorzunk meg egymással, hogy megkapjuk a kívánt hasított jelet. Ez a két jel a szinusz függvény és a square wave, magyar nevén a négyszögjel. A két jel frekvenciájánál figyelni kell arra, hogy a négyszögjel periódusideje kétszer akkora legyen mint a szinusz függvényé, ezzel a szinuszfüggvény pozitív és negatív szeletein is teljes legyen a négszögjel egy periódusa. Ezek után a négszögjellel kell műveleteket elvégezni, mivel a LabVIEW a négyszögjelet $[-1, 1]$ között generálja, a kezdő szakasza 1-től indul, ezek után vált csak 0-ra. Ezzel így az alap függvény nem megfelelő a mi felhasználásunknak. Először felcseréljük a függvény ismétlődő fázisát, ezért azt megszorozzuk -1-el. Ezek után hozzáadunk a függvényhez kettőt, így az új értékkészlete $[0, 2]$ lesz. Utolsó lépésként elosztjuk 2-vel, így megkapjuk a kívánt 0-val kezdődő, majd 1-re ugróm függvényt. A hasítási pontot százalékban adjuk meg, ezt egy a Front Panelen elhelyezett szám kontrollal lehet irányítani. Ez a jelgenerátor alap négszögjeléhez hasonlóan nem úgy működik mint a számunkra elvárt. 100-ból a megadott számot kivonva kapjuk meg azt a százalékot, hogy az eredeti szinusz függvény hány százaléka lesz jelen a kapott hasított függvényben. Ezt a jelgenerátor 'duty cycle' lábára kötve tudjuk szabályozni a hasítási pont (bekapcsolási pillanat) helyét a függvényen.

Hivatkozások

[1] „Bevezetés a LabVIEW világába”. (), cím: https://megtestesules.info/hobbielektronika/2019/labview19_01.pdf.

[2] F. Gergely. „Friedl Gergely- LabVIEW Segédlet”. (), cím: https://maxwell.sze.hu/~friedl/Szab%C3%A1lyoz%C3%A1si_rendszerek/LabVIEW%20seg%C3%A9dlet.pdf.

[3] N. Instruments. „LabVIEW Supplemenal - LabVIEW Front Panel Explained”. (), cím: https://www.ni.com/en/support/documentation/supplemental/08/labview-front-panel-explained.html.

[4] N. Instruments. „LabVIEW Supplemenal - LabVIEW Block Diagram Explained”. (), cím: https://www.ni.com/en/support/documentation/supplemental/08/labview-block-diagram-explained.html.

[5] N. Instruments. „LabVIEW Fundamentals - Building the Connector Pane”. (), cím: https://www.ni.com/docs/en-US/bundle/labview/page/building-the-connector-pane.html.

[6] L. M. Hub. „LabVIEW Basics - 03 | Controls and Indicators”. (), cím: https://www.labviewmakerhub.com/doku.php?id=learn:tutorials:labview:basics:controls_indicators.

[7] Digilent.com. „Creating a subVI in National Instruments LabVIEW”. (), cím: https://digilent.com/blog/creating-a-subvi-in-labview/.

[8] „Effektív Érték.pdf (PPKE ITK Bevezetés a méréstechnikába és jelfeldolgozásba)”.

[9] „Effektív érték kiszámításához képlet - A szinuszos váltakozó feszültség középértékei - Ham.hu”. (), cím: https://wiki.ham.hu/index.php?title=Effekt%C3%ADv_%C3%A9rt%C3%A9k.

[10] H.-G. M. (szerkesztő), *Fáziskésleltetés: Elmélet és gyakorlat*. Miskolci Egyetem Kiadó, 2010.

[11] K. I., *Fáziskésleltető rendszerek*. Műszaki Könyvkiadó, 2003.
[12] K. L., *Fáziskésleltető és fázismódosító hálózatok*. Typotex Kiadó, 2007.