פרוייקט הלווין האוניברסיטאי – תחנת הקרקע

גרסת 2015 בהנחיית ד"ר גרא וייס ומר אבירן סדון

במסמך זה ישנו מדריך הפעלת המערכת והדגמת פעולה.
כמו גם תיעוד בדיקות והדרכה מלאה איך לשנות ולהוסיף]ונקציונאליות באם תדרש בעתיד.

עידן מור, שמעון לחיאני, יניב רדומסקי והוד עמרן.
27/08/2015

תוכן עניינים

מסמך תחזוקה – תחנת קרקע

Negev Sat Ground Station :

בהמשך לפירוט מסמך התחזוקה אשר נכתב בגרסא הפרוייקט של שנת 2014 (ראה מסמכים מצורפים בתקיית הפרוייקט תחת תקיית "מסמכים 2014") בחרנו לשנות כמה אספקטים חשובים עליהם נפרט כאן:

ב communication package , שינינו את פרוטוקול התקשורת מ xml לתקשורת בינארית (רק עבור המידע שנשלח מהלוויין אל תחנת הקרקע ולא בכיוון ההפוך !!, ניתן לראות את הפרוטוקול שפתחנו תחת תקיית "מסמכים 2015"), ועל כן הוספנו למחלקה Message שדה הנקרה tosend ומטרתו להכיל את הבתים שרוצים לשלוח , באם רוצים להשתמש בפרוטוקול בינארי.

אם רוצים להשתמש בפרוטוקול xml בכל זאת בתהליך קבלת המידע מהלוויין, אזי השדה textנשאר ומאותחל כמו שהוא בריצת התוכנית וביצירת אובייקט message אך בעת פרסור המידע יש לפרסר את שדה text ולא את tosend כפי שאנו בצענו.

למעשה, הבחירה האם לשלוח/לקבל בפרוטוקול בינארי או בפרוטוקול xml היא בשימוש המטודה getBytes של Message אשר כרגע שולחת את המידע כ xml באם שולחים אותו אל הלוויין (ולא אם מופעל הסימולטור כשאז בכוונתו לבדוק את התוכנה שלנו), אם רוצים שגם המידע שנשלח אל הלוויין יהיה בבינארי אזי יש לשנות את הקוד של getBytes בהתאם, התמיכה לכך כבר מומשה במטודה של sendMissionב communicationManager, אך לא נבדקה לעומק. (כפי שציינו, נוצר המידע התואם בצורה בינארית אך ב serialWriter כשפונים ל getBytes של Message שולחים למעשה את שזה Text של Message ולא שדה ToSend כפי שנשלח באם אנו עובדים במוד סימולטור כשאז בכוונתנו לבדוק את נכונות קבלת המידע, הפרסור).

בכדי לשלוח/לקבל בצורת xml צריך בסה"כ לבקש מה serial reader /serial writer להתייחס לשדה המתאים אותו רוצים לשלוח/לקבל .(פונקציית getbytes והקוד בהערה בהתאמה ב serialReader).

```
public class Message {
        private String messageText;   יכיל את טקסט התואם לתצורת הישנה
        private Vector<Byte> ToSend;  יכיל את רצף הבתים בהתאם לתצורה החדשה
```

כמו כן, בקומות קריטיים בהם לא היה ניתן לכתוב 2 פונקציות מקבילות ולהשתמש באחת מהן כרצון המפתח, כתבנו בהערה ממוסגרת כי הקוד מתאים לתקשורת מסוג xml וכן גם הקוד עצמו תחת הערה, ואילו הקוד של תקשורת בינארית בקטעים אלו הוספנו תיעוד בהערה כי הקוד תואם לתקשורת בינארית.

לדוגמה :

CommunicationManager.java | SerialReader.java | SerialWriter.java | MessageParser.java | SatelliteSimulator.java

```
78
79              ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
80
81              // Use this code for XML communication only !!!!
82
83              //System.out.println("DEBUG: Message Accepted By Parser");
84              //System.out.println(m.toString());
85              /*
86              Document msg;
87              try {
88                  msg = m.toDocument();
89              }
90              catch (Exception e) {
91                  System.out.println(e.getMessage());
92                  continue;
93              }
94               */
95              ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
96
97
98
99
100
101             ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
102
103             //// Use this code for Binary communication only !!!!
104             try {
105                 parseBinaryData(m.getBytesReceive());
106                 //parseMessage(msg);
107             } catch (InvalidMessageException e) {
108                 /*
109                 if (msg.getElementsByTagName(tagUpPacket).getLength() != 0) {
110                     continue; // Ignore upstream packets sent from airborne control system simulator
111                 }
```

לסיכום, השינויים אותם הכנסנו לצרכי הרחבת יכולת לתקשורת בינארית נכנסים למחלקות : message, messageparser, serialreader, serialwriter.

והקוד בתצורתו הנוכחית עובד בתקשורת הבינארית!

FunCube Ground Station :

GUI Package:

TelemetryWindow – החלון המרכזי שמציג את הנתונים שנתקבלו מהלוויין, חלון זה למעשה הינו המנשק משתמש למול הלוויין. בחלון זה ניתן לבחור את קובץ ה wav שהוקלט מהלוויין ולעבד את המידע שהוא מכיל לפי הפורמט שמתואר באתר הלוויין באמצעות הרצת קובץ Bat המכיל פקודה להרצת decoder שפורסם באתר החברה ובוצעו לו שינויים והתאמות לאפליקציה שלנו (נכתב ב c++ עבור linux).

אם רוצים להריץ קוד אחר עבור decodeצריך לקמפל את הקוד החדש לפורמט exe ושלנות את הנתבי בהתאם בתוכן קובץ ה bat.

Domain Package :

Parser – מחלקה סטטית המשמשת כמחלקת עזר לביצוע פרסור המידע שנתקבל לאחר תהליך ה decode. מכיל מטודות סטטיות לצרכי פרסור בלבד. לא נוצר מופע שלה במהלך ריצת המערכת.

GroundStation – אובייקט מסוג singleton, מכיל את פונקציית ה main, ממנו רצה המערכת כולה. מאתחל את החלון של מנשק המשתמש וכן את המשתמשים המשמשים אותו בריצת המערכת.

BitStream- מחלקה שבאמצעותה אנו מייצגים מידע שנתקבל בפריים מסוים וממנו בונים אובייקט telemetry באמצעות רצף סיביות. יורשת ממחלקת הספרייה IO.Stream, מכילה מיני מטודות לצורך ייצוג המידע בדרכים שונות, כתיבה וקריאה לאובייקט זה.

Telemetry – מכיל מידע אודות כל חיישני הלוויין ב real-time וכן payload שמתווסף לכל פריים שנשלח מהלוויין. המידע מאורגן באמצעות dictionary הממפה מאינדקס החיישן ,שזהו המפתח, לבין ערך המידע TelemetryValue שהוא מדד.

TelemetryValue – מחלקה אבסטרקטית המייצגת מידע שנדגם מחיישן לוויין כלשהו שאותו ניתן להציג בכמה דרכים : כמשתנה מספרי, כמשתנה מספרי עם דיוק לאחר הנקודה העשרונית, כטקסט ועוד. מחלקת זו היא למעשה מחלקת אב לכל דגימת חיישן כלשהו מהלוויין, בהמשך נראה כי היא מייצגת המון דגימות.

כל המחלקות הבאות הינן מחלקות יורשות מהמחלקה הנ"ל וממשות את כל המטודות הווירטואליות שלה, כל אחת מייצגת מידע מסוג שונה (לרוב מדובר מחיישן מסויים) , ההבדלים ביניהן אינן משמעותיים ולרוב הן נוצרו לצרכי נוחות ע"מ לבצע הפרדה ברורה בין מקורות המידע של הלוויין:

BoolOnOffTelemetryValue – מציג מידע אודות חיישן או רכיב ברמה הבוליאנית האם הוא עובד או לא.

NullTelemetryValue- מציג מידע לא מוכר שהגיע מהלוויין, בשימוש בעיקר לצרכי בדיקות.

AntsDeployTelemetryValue- מציג האם הלוויין פרס את הזרוע שלו (ישנן 4 זרועות הממוספרות מ 0 עד 3) ועבור כל אחת נקבל מיגע האם נפרסה או לא.

SunSensorTelemetryVallue- מציג את תוצאת המדידה מכל אחת מקולטי השמש לפי טבלת ההמרה שמצוינת בטבלת המשוואות והערכים.

DeviceDataValidTelemetryValue- מציג האם המידע שמספק הלוויין הינו תקין, גם כאן המידע הינו בוליאני , אך מוצג כטקסט בצורה שונה לעומת BoolOnOffTelemetryValue.

EPSResetCauseTelemetryValue- מציג את הגורם לאיפוס חיישן ה EPS :

Power on– דולק.

External– גורם חיצוני

Brown Out– בעיית טמפרטורה.

Watchdog– תהליך תוכניתי-חומרתי.

Jtag– חיבור חומרתי.

Other

EPSPowerTrackingTelemetryValue – מציג את 'מוד' הפעולה של חיישן זה:

Hw default

MPPT

SW Fixed

Unknown

SoftwareAbfTelemtryValue – מציג האם חיישן זה מאופשר או לא.

MultuplierOfssetTelemetryValue – מחלקה המייצגת תוצאות מדידות של חיישנים שונים אשר להם נוסחת המרה קבוע המכפילה בקבוע כלשהו (שדה ה Multiplier של מחלקה זו) ומתווסף ערך סף מסוים (שדה ה offset של מחלקה זו) דוגמה לחיישן כזה הינו חיישן מדידת טמפרטורה של פאת לוויין.

MultuplierPowerTelemetryValue- בדומה למחלקה הקודמת, מייצג חיישנים אשר את הערך שמפיקים ממדידותיהן צריך להעלאות בחזקה. דוגמת חיישן כזה הינו PaPower.

PaCurrentTelemetryValue- יורש מ MultuplierOfssetTelemetryValue, תוך השמת ערכים שונים להכפלה בקבוע והוספת סף בהתאם לחיישן.

PaPowerTelemetryValue- יורש מ MultuplierPowerTelemetryValue, תוך השמת ערכים להעלאה חזקה עבור בסיס מסויים בהתאם לנוסחת החיישן.

MultiplierTelemetryValue- בדומה ל MultuplierOfssetTelemetryValue רק ללא offset , ז"א שימוש עבור MultuplierOfssetTelemetryValue עם שדה ה offset=0 יפיק את אותה תוצאה.

FrameIdTelemetryValue- מציג את סוג הפריים שנתקבל :

Whole orbit data.

High resolution data.

Fitter messages.

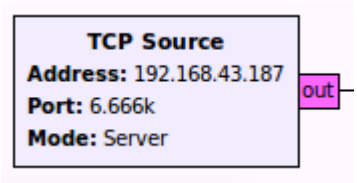
PaTemperatureTelemetryValue- מציג את טמפרטורת חיישן ה pa לפי טבלת ההמרה שמצורפת בקובץ הנוסחאות של הלוויין.

ניתן להוסיף מידע מסוגים שונים בצורה זו ע"י כתיבת מחלקה חדשה המייצגת את המידע ובלבד שתהא חלק בתוך ה constructorשל המחלקה telemetry שתבנה אובייקט סוג מחלקה זו בעת קבלת רצף של בתים.

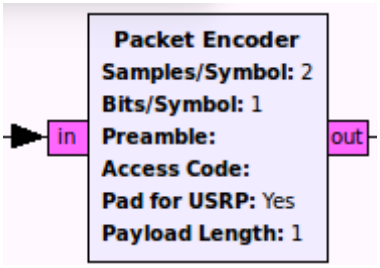
מסמך תחזוקה gnu-radio.

במסמך זה נתייחס לרכיב ה gnu-radio בלבד, בו נציג את רכיבי המערכת כל רכיב נקרא בלוק (block).

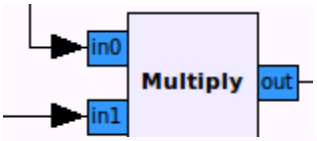
נציג כל בלוק מהמערכת בה עשינו שימוש בנוסף למס' מילות הסבר.



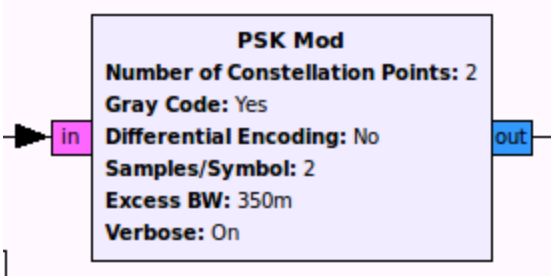
TcpSource –רכיב תקשורת אשר מהווה "צינור קלט" דרכו נכנסות פאקטות בחיבור tcp דרך הפורט וה – ip המצויינים בבלוק. הבלוק יכול להיות במצב client/server. החיבור למערכת "out" הצבע הורוד מציין סטרימר של בתים.



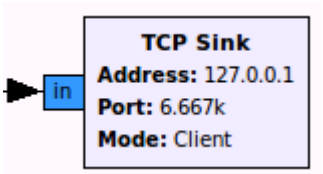
PacketEncoder – רכיב ה "אורז את stream הפאקטות שהתקבלו ממה – tcp בתוספת header ומשם מוזרם למודולצייה כלשהיא כגון bpsk/gmsk וכדומה. ניתן לשנות את אורך ה payload.



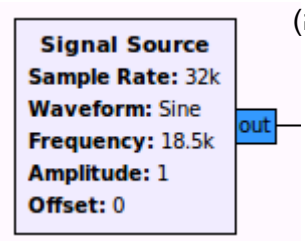
Multiply – הבלוק מתפקד כמרכיב גל מידע על גל נושא דהיינו לקיחת אות המידע ואפננו על גבי אות נושא בתדר גבוהה יותר ובהספק חזק יותר.



PSK Mod – משמש כבלוק מרכזי במערכת פועל כמודולטור של המידע. כניסת ה "in" מקבלת רצף בתים ויציאת ה "out" מוציאה רצף של מספרים קומפלקסים.

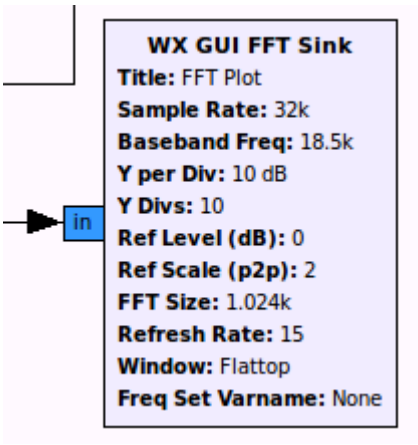


tcpSink - יציאת tcp פשוטה המקבלת כקלט רצפים קומפלקסיים ושולחת אותם לפורט הנתון. ניתן לשנותה ל server כמובן.

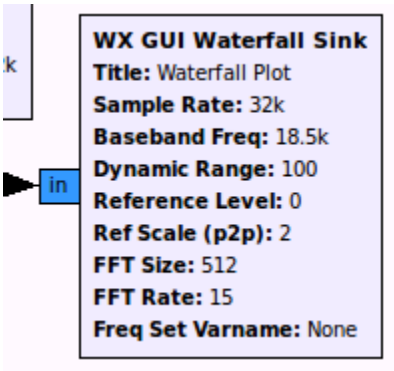


signalSource – בלוק המדמה את גל הנושא (תדר גבוה) עליו מורכב גל המידע מבלוק ה psk. השימוש נעשה ע"מ ליצר יכולת שידור למרחקים גדולים.

ComplexToFloat – בלוק טכני המבצע המרה ממספר קומפלקסי ל2 מספרים המייצגים את המספר המורכב.

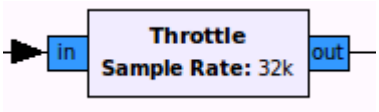


WX GUI FFT SINK – בלוק עזר למשתמש לצורכי הצגת המידע העובר בקו משמש לדיבוג והצגת המידע.



WX GUI WATERFALL SINK – בדומה בלוק לעיל. משמש עזר למשתמש לצורכי הצגת המידע העובר בקו משמש לדיבוג והצגת המידע.

FileSink – בלוק המשמש לכתיבת המידע לקובץ.



Throttle

– בלוק המשמש כצוואר בקבוק ע"מ למנוע עומס על המעבד.

AudioSink

– בלוק אשר מקבל כקלט stream ופולט שמע.

מדריך למשתמש תחנת קרקע

Negev Sat Ground Station

בכדי להקים סביבת פיתוח עבור התוכנה יש הצורך בדברים הבאים:

1. התקנת java jdk 1.8.0 32-bit וסביבת eclipse.
2. הורדת java serial communication package RXTXcomm ניתן להוריד מ: http://www.jcontrol.org/download/rxtx_en.html
3. Sqlite – ניתן להוריד מ : http://www.sqlite.org/2014/sqlite-shell-osx-x86-3080600.zip
4. שאר קבצי ה jar שנמצאים בתקיית הפרויקט תחת תקיית lib .
5. התקנת הכלי vspe לצורך הקמת חיבור tcp ע"י com וירטואלי (אפשר כל כלי אחר, אנו השתמשנו בכלי זה). הורדה : http://www.eterlogic.com/Products.VSPE.html
6. התקנת הכלי gnu radio על מערכת linux או wm.

תהליך ההתקנה :

- לאחר התקנת java jdk 1.8.0 32-bit וסביבת eclipse יש למקם את rxtxSerial.dll בתקיית ה bin של jre (שנוצרה בהתקנת ה jdk), בד"כ יהיה במיקום : c:\program files(x86)\java\j.d.k 1.8.0\jre\bin
- למקם את RXTXcom.jar בתקיית ה ext של jdk , בד"כ יהיה במיקום c:\program files(x86)\java\j.d.k 1.8.0\jre\lib\ext
- התקנת ה vspe.
- העתק את תקיית הפרויקט NegevSat-2015 ל workspace של eclipse , ולאחר פתיחת ה eclipse צור פרויקט חדש בשם NegevSat-2015, התוכנה תתריע לך שהפרויקט קיים (כך צריך להיות) ותטען את הפרויקט וכל תלויותיו.
- בדוק שתחת הגדרת project->properties->javabuildpath->libraries->jre system library נבחר jdk 1.8 32 bit.
- אם יש בעיות קומפילציה, הן נובעות מאי-זיהוי של קבצי ה jar, כנראה שלא הועתקו למקומות שצויינו קודם לכן.
- התקנת gnu radio במערכת linux באמצעות הפקודה: apt-get install gnuradio

תהליך הרצה :

- לפני הרצה בפעם הראשונה יש להריץ פקודה אשר בונה db ע"י הרצת פקודות ב cmd :

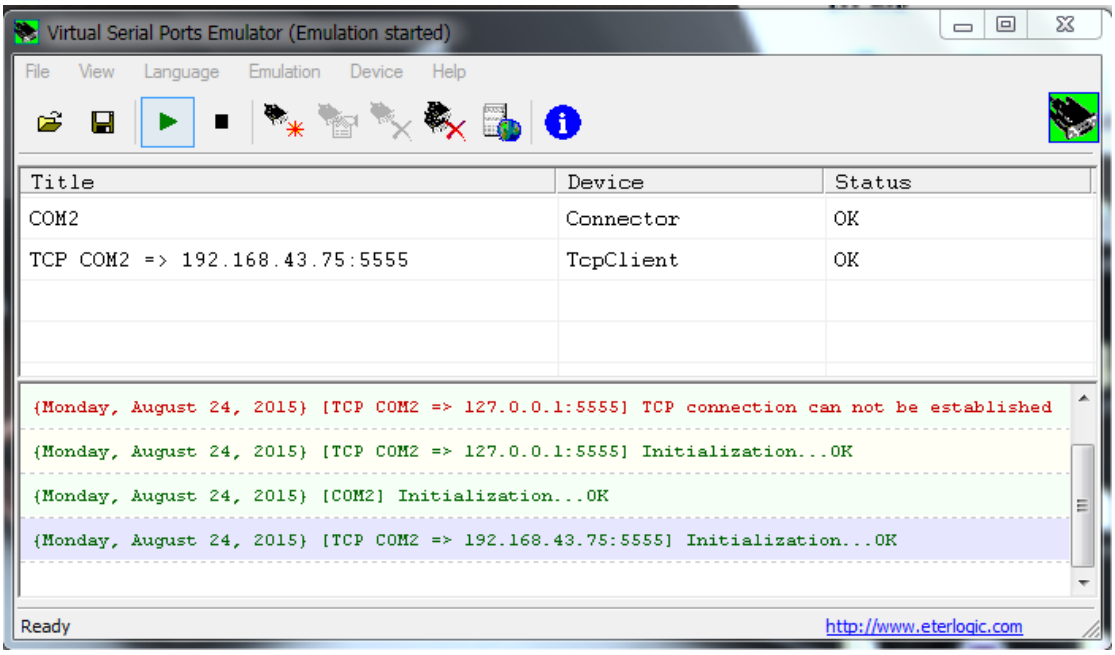
```
cd C:\sqlite
sqlite3.exe negevSatDB.db
.quit
```

- בכדי להריץ את התוכנית לבדה יש להריץ את
- GUI\negevsatgui\NegevSatGui.java

אם רוצים להריץ את התוכנית בהתקשרות ישירה עם התוכנה המוטסת וללא ה gnu radio:

בתוכנת vspe נגדיר :

-connector ל com2.
- tcp client שמקונפג להתחבר ל ip של התוכנה המוטסת ואשר אצלה מוגדר tcp server בהתאמה.



הרצת ה Negevsat כמו בהסבר למעלה.

אם רוצים להריץ את התוכנית בתצורה מלאה כוללgnu :

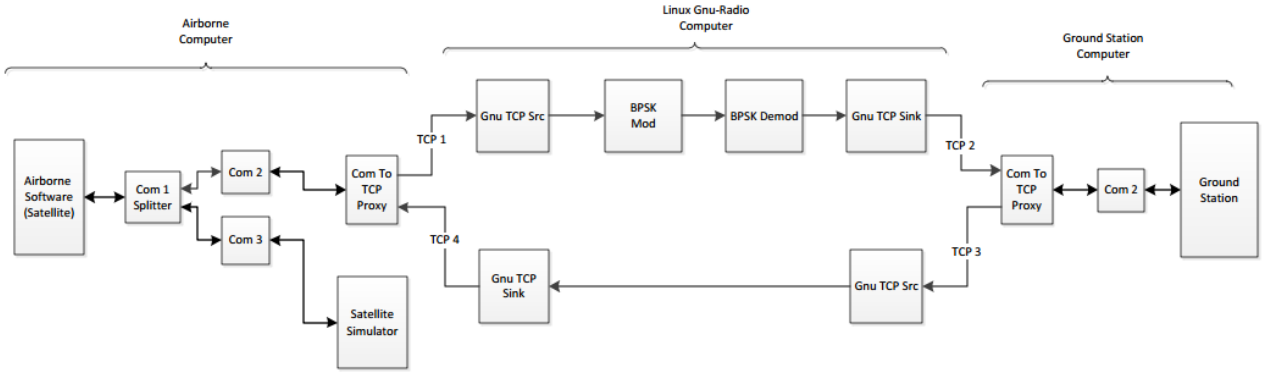
בתקיה הבאה ישנם סרטונים לעזרה :

https://drive.google.com/a/post.bgu.ac.il/folderview?id=0BygKDN4TfnyYfng3RDZHUEwtcWhWLVJkS3I1bEk0UmhiVTM0TzI1TmItOHZPTnBFVUtPT00&usp=sharing

- קנפוג ה gnu כפי שהראינו בסרטון
- קנפוג ה vspe וה proxy2 כפי שהראינו בסרטון

- הרצת ה gnu
- הרצת ה proxy2 שנמצא בתקיית הפרויקט תחת תקיית Sim+Proxy(בסנכרון עם proxy1 כפי שרואים בסרטון).
- הרצת ה negevsat כפי שהראינו למעלה.

תאור חיבורים בין 3 התחנות המתוארות:



FunCube Ground Station

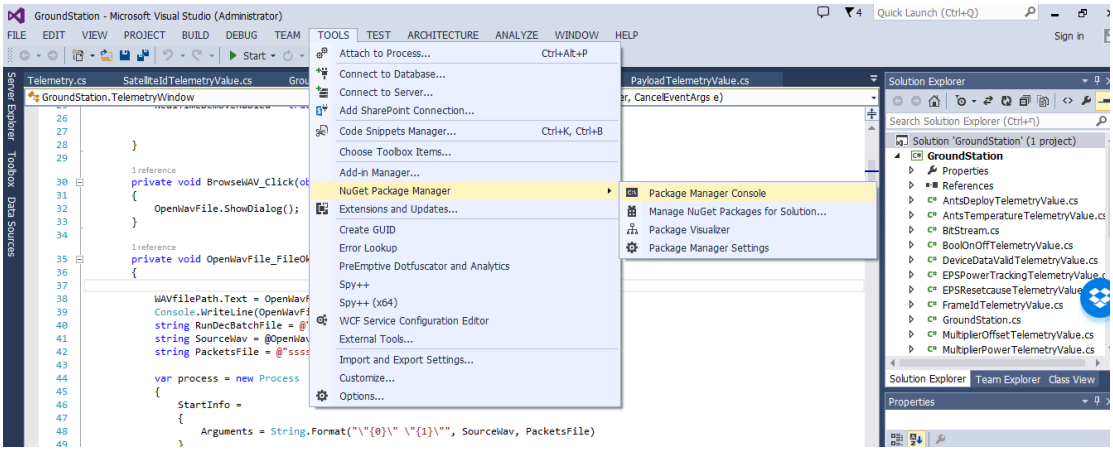
-מומלץ להתקין את sdr# לניגון קבצי wav לצרכי עזר בלבד.

- יש להתקין את visual studio 2013.

- יש להתקין את ' NuGet Packet Manager, תוסף עבור vs2013

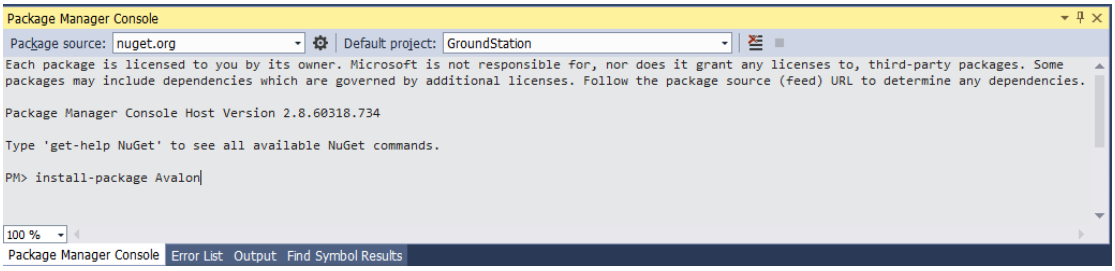
לאחר מכן יש ללחוץ על GroundStation.sln :

- כל שגיאת קומפילציה נובעת מחוסר ב dll שאינו מגיע בחבילת ההתקנה של vs ועל כן יש להתקינו באמצעות ה ' NuGet Packet Manager באופן הבא:



בחלון התחתון שייפתח יש לכתוב את הפקודה install-package name כאשר name הוא שם ה dll שחסר ובגללו יש בעיית קומפילציה (בתיאור הבעייה מוזכר שמו).

לדוגמה:



לאחר שהתוכנית עוברת קומפילציה יש להריץ אותה, ויפתח החלון הראשי שבו ניתן לטעון קובץ wav לפרסור והצגה.

מסמך בדיקות negevsat תחנת קרקע

את הפרויקט בדקנו בדרכים שונות ומגוונות.

עבור על צוות בצענו בדיקות יחידה ובדיקות שילוב, כמו כן בצענו מבחני קבלה עבור האינטגרציה שבין הצוותים ו flow של מידע מקצה לקצה תוך שימוש בכלי בדיקה וניתור בכדי לקבל אינדיקציה אודות נכונות המידע וזיהוי בעיות.

עבור תחנת הקרקע כתבנו בדיקות יחידה עבור המטודות שכתבנו , תוך שימוש בהשמת ערכים שנראו לנו בעייתים מצד אחד או נכונים מצד אחר בכדי לבדוק את תקינות היחידה.

כמו כן, בצענו גם הרצת בדיקות יחידה גנריים (הגרלת ערכים בטווחי ערכים שונים) אשר עזרו לנו לגלות בעיות שלא חשבנו שיווצרו.

במחלקה commTest ניתן לראות את פירוט הבדיקות שכתבנו עבור פרסור נכון של מידע שהתקבל (כל המטודות הקשורות בפרסינג של מידע שהגיע) בתצורה של בדיקות יחידה:

שם הבדיקה	יחידות נבדקת (ללא תלות)	תוצאה
testParseEnergy	parseEnergy,DB	תוצאה תקינה. אוסף הבתים שנוצר בבדיקה פורסר כראוי ואף נכנס לבסיס הנתונים

testParseTemp	parseTemp,DB	תוצאה תקינה. אוסף הבתים שנוצר בבדיקה פורסר כראוי ואף נכנס לבסיס הנתונים
testParseStatic	parseStatic , DB	תוצאה תקינה. אוסף הבתים שנוצר בבדיקה פורסר כראוי ואף נכנס לבסיס הנתונים
testTempGenerate	parseTemp,DB, *intToByteArray*,*LongToByteArray*	בחלק מהמקרים נתגלו בעיות, בעיקר בגלל ערכים שה DB לא יכול להכיל וגם עבור נסיון יצירת אובייקטים עם ערכים שגויים ל constructors. דבר הגרם לבצע הגנות באמצעות try/catch בכדי להבטיח שהמערכת לא תקרוס באם יתקבל מידע מסוג זה.
testStaticGenerate	parseStatic,DB , *intToByteArray*,*LongToByteArray*	בחלק מהמקרים נתגלו בעיות, בעיקר בגלל ערכים שה DB לא יכול להכיל וגם עבור נסיון יצירת אובייקטים עם ערכים שגויים ל constructors. דבר הגרם לבצע הגנות באמצעות try/catch בכדי להבטיח שהמערכת לא תקרוס באם יתקבל מידע מסוג זה.
testEnergyGenerate	parseEnergy,DB, *intToByteArray*,*LongToByteArray*	בחלק מהמקרים נתגלו בעיות, בעיקר בגלל ערכים שה DB לא יכול להכיל וגם עבור נסיון יצירת אובייקטים עם ערכים שגויים ל constructors. דבר הגרם לבצע הגנות באמצעות try/catch בכדי להבטיח שהמערכת לא תקרוס באם יתקבל מידע מסוג זה.

עבור בדיקות אינטגרציה , ובדיקות של flow שלם החל מקבלת מידע ב inputstream ועד פרסורו והכנסתו לבסיס הנתונים השתמשו ב simulator שפותח שלמעשה דימה את התוכנה המוטסת ששולחת מידע אל תחנת הקרקע.

בבדיקות אלו בדקנו זרימה שלמה בין הקומפוננטות שדרכם עובר המידע , כאשר בכל קומפוננטה ישנן מטודות אשר מבצעות מניפולציה כלשהי על המידע ומעבירה לקומפוננטה הבאה ב flow .

הוספנו ל simulator בדיקות של שליחת מידע בצורה בינארית בהתאם לשינוי הפרוטוקול שבצענו.

פירוט הבדיקות שמומשו באמצעות ה simulator :

שם הבדיקה	קומפוננטות ב flow שנבדק	תוצאה
sendTemperatureToGroundByBtye	,OutputQ,SerialWriter,GNU Mode, GNU Demode, SerialReader,InputQ,MessageParser, DB	תוצאה תקינה. המידע שנוצר ונשלח מהסימולטור ייצר הודעה תקינה אשר נכנסה לתור ההודעות היוצאות, משם נשלחה באמצעות ה serialwriter התקבל ב serialreader, נכנס לתור ההודעות המתקבלות, פורסר כראוי ב messageparser , ונכנס לבסיס הנתונים
sendEnergyToGroundByBtye	,OutputQ,SerialWriter,GNU Mode, GNU Demode, SerialReader,InputQ,MessageParser, DB	תוצאה תקינה. המידע שנוצר ונשלח מהסימולטור ייצר הודעה תקינה אשר נכנסה לתור ההודעות היוצאות, משם נשלחה באמצעות ה serialwriter התקבל ב serialreader, נכנס לתור ההודעות המתקבלות, פורסר כראוי ב messageparser , ונכנס לבסיס הנתונים
sendEnergyToGroundGenerate	,OutputQ,SerialWriter,GNU Mode, GNU Demode, SerialReader,InputQ,MessageParser, DB	בחלק מהמקרים נתגלו בעיות, בעיקר בגלל ערכים שה DB לא יכול להכיל וגם עבור נסיון יצירת אובייקטים עם ערכים שגויים ל constructors. דבר הגרם לבצע הגנות באמצעות try/catchבכדי להבטיח שהמערכת לא תקרוס באם יתקבל מידע מסוג זה. אך כל התקשורת והעברת המידע בין הקומפוננטות בוצע כראוי ללא איבוד מידע או פגימה בתוכנו.
sendEnergyToGroundGenerate	,OutputQ,SerialWriter,GNU Mode, GNU Demode, SerialReader,InputQ,MessageParser, DB	בחלק מהמקרים נתגלו בעיות, בעיקר בגלל ערכים שה DB לא יכול להכיל וגם עבור נסיון יצירת אובייקטים עם ערכים שגויים ל

		constructors. דבר הגרם לבצע הגנות באמצעות try/catchבכדי להבטיח שהמערכת לא תקרוס באם יתקבל מידע מסוג זה. אך כל התקשורת והעברת המידע בין הקומפוננטות בוצע כראוי ללא איבוד מידע או פגימה בתוכנו.

עבור בדיקות שילובים עם התוכנה המוטסת הבדיקות שבוצעו היו ברובן ידניות מפני שמורכבות המערכת והתלות ברכיבי תקשורת רבים (תוכנת vspe, קוד proxy שנכתב במיוחד עבור החיבור המיויחד הזה, לעיתים נדירות יש איבוד מידע שלא הצלחנו בוודאות להבין מהיכן נובע באמצעות כלי ניתור ובדיקה כמו wireshark.) וכן הזמן המועט בו התוכנה המוטסת יכולה לפעול ברצף לא מאפשרים ביצוע בדיקות אוטומטיות מספקות ולכן בצענו המון בדיקות ידניות שבדקו את שילוב כל המערכת וראינו אכן שהמערכת מבצעת כנדרש את המוטל עליה.