מודול משדר מקלט - nRF24L01-2.4GHz

1.א חומרה

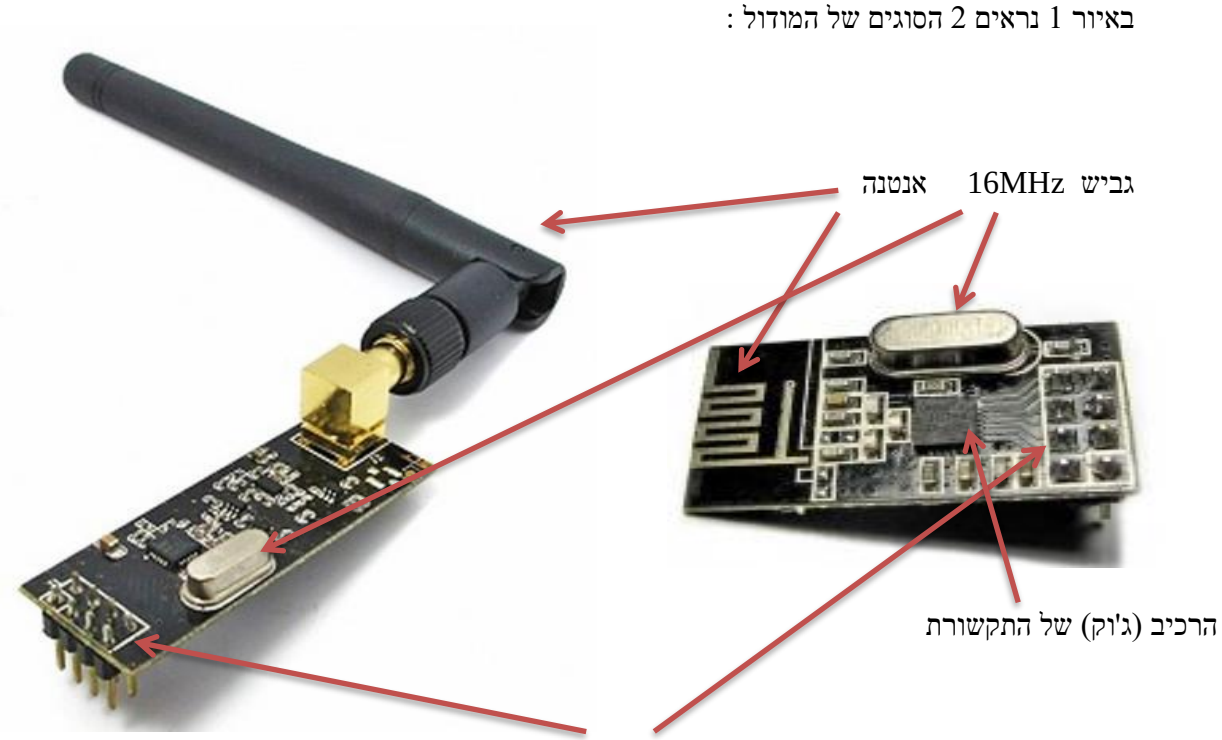
המודול של חברת **Nordic Semiconductor** משלב משדר ומקלט בתדר 2.4GHz , RF סינטסייזר ולוגיקת פס בסיס הכוללת פרוטוקול מאיץ חומרה שפותח על ידי חברה זו ונקרא ESB - Enhanced ShockBurst שהוא פרוטוקול דו כיווני בין 2 משדרים/מקלטים לחילופי מנות (מנה היא packet -והכוונה לכמות מסוימת של בתים) של נתונים. במודול יש רכיב nRF24L01 שהוא מכיל משדר RF, מקלט RF ופס בסיס BASEBAND לניהול התקשורת. עבודה בתחום תדר זה איננו דורש רישיון ממשרד התקשורת ומחיר מודול כזה הוא מספר בודד של $ ...

אתר החברה והסברים ניתן למצוא בכתובת:

https://devzone. file:///C:/Users/user%20name/Downloads/nRF24L01_Product_Specification_v2_0.pdf

המודול תומך בממשק SPI – Serial Peripheral Interface – ממשק טורי היקפי מהיר . בצריכת חשמל נמוכה הוא מגיע למרחק קצר של 50 עד 200 רגל . יש לו גרסה לטווח גדול יותר (1000 מטר ??) על ידי הוספת אנטנה חיצונית במקום האנטנה שהיא קווי זיגזג על המעגל המודפס ועוד מגבר הספק לשידור וקדם מגבר לקליטה.

באיור 1 נראים 2 הסוגים של המודול :

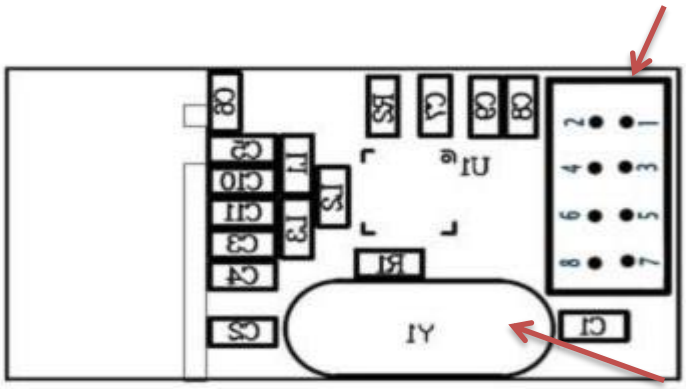


קונקטור – מחבר - התחברות של 8 הדקים אל המודול

איור 1 - מודול NRF24l01. מימין ללא תוספת של אנטנה חיצונית ומשמאל עם אנטנה חיצונית ומגבר.

באיור 2 נראה מבט מהחלק התחתון של הכרטיס עם הקונקטור של – מצד ימין בכרטיס

מחבר בן 8 הדקים להתחברות



גביש

איור 2 : מבט מהחלק התחתון של המודול.

2.א מאפיינים

- פעולה בתדר 2.4GHz ב ISM band - Industrial , Scientific and Medical – בתחום פס שימושי לתעשייה, למדע ורפואה .
- קצב נתונים של עד 2Mbps (ביטים בשנייה).
- פעולה בתצרוכת הספק נמוכה ביותר.
- 11.3mA בהספק שידור של 0dBm .
- 12.3mA בקליטה ב 2Mbps .
- 900nA במצב עבודה של "הורדת הספק" - power down .
- מייצב מתח בתוך רכיב.
- מתח ספק בין 1.9 עד 3.6 וולט.
- ניהול מנה אוטומטי.
- עובד ב Enhanced ShockBurst
- אפשרות קליטה עם 6 צינורות (pipes).
- גביש 16MHZ + 60ppm.
- ניתן להתחבר אליו עם רכיבים העובדים עם 5 וולט.
- 3 חוצצי FIFO נפרדים גם לשידור וגם לקליטה בני 32 בייטים.

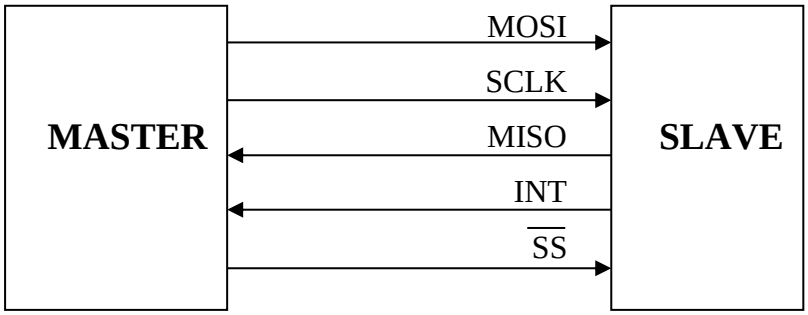
3.א יישומים

- רכיבים היקפיים אלחוטיים למחשב.
- עכבר, לוח מקשים ושלט רחוק.

- אוזניות
- בקרי משחקים
- חיישנים ושעוני ספורט
- RFID
- חיישני רשת רגישים.
- מערכות עקיבה
- צעצועים.

א.4 תקשורת SPI - Serial Peripheral Interface ממשק טורי היקפי

זוהי צורת תקשורת טורית דו כיוונית בין מחשב (MASTER) ורכיב היקפי (SLAVE) הכוללת 5 קווים. הקווים נראים באיור 3:



איור 3 : תקשורת SPI

MOSI - Master Out Slave In - יציאת אדון כניסת עבד. זוהי יציאת הנתון הטורי מהמסטר אל העבד.

SCLK - Serial CLocK - השעון הטורי המתזמן את כניסת הנתון הטורי אל העבד.

MISO - Master In Slave Out - כניסת אדון יציאת עבד. זוהי כניסת הנתון הטורי הנשלח מהעבד אל המסטר.

INT - INTerrupt - קו זה מודיע מודיע למסטר שהסתיים שידור או קליטה של מנה.

$\overline{SS}$ - Slave Select - בחירת עבד. בעזרת רגל זו מודיע המסטר לעבד שהוא פעיל. הקו פעיל בנמוך.

תהליך התקשורת בממשק SPI

העברת הנתון בממשק SPI מתבססת על הכנסת הנתון הטורי מהמסטר לעבד (ברגל ה MOSI) בעלייה של פולס השעון ובירידה עובר ביט נתון מהעבד אל האדון (ברגל MISO). כאן זה מבוצע בצורה הבאה:

1. העברת הנתונים הטורית מתחילה בירידה ברגל $\overline{SS}$.
2. $\overline{SS}$ מוחזקת בנמוך למשך כל התקשורת הטורית ומוחזקת בגבוה בין הפקודות.

3. ביט הנתון מהדק MOSI נכנס בעלייה של השעון ויוצא מהדק MISO בירידה של השעון כאשר ביט ה LSB הוא הראשון.
4. כל הפעולות מתחילות להתבצע אחרי העלייה של $\overline{SS}$.

למודול יש מחבר של 8 הדקים הנראה בטבלה 1 :

התפקיד	הדק	הדק	תפקיד
VCC	2	1	GND
CSN	4	3	CE
MOSI	6	5	SCK
IRQ	8	7	MISO

טבלה 1 : הדקי המחבר במודול

הדק 1 - GND אדמה

הדק 2 - ה Vcc מקבל 3 וולט.

הדק 3 - CE – Chip Enable – שולט על אופן עבודה Standby ושידור וקליטה (לא קיים ב SPI רגיל).

הדק 4 CSN הוא הדק $\overline{SS}$.

הדק 5 - SCK – פולסי השעון הטורי מהמסטר לעבד.

הדק 6 - MOSI הנתון הטורי מהמסטר לעבד

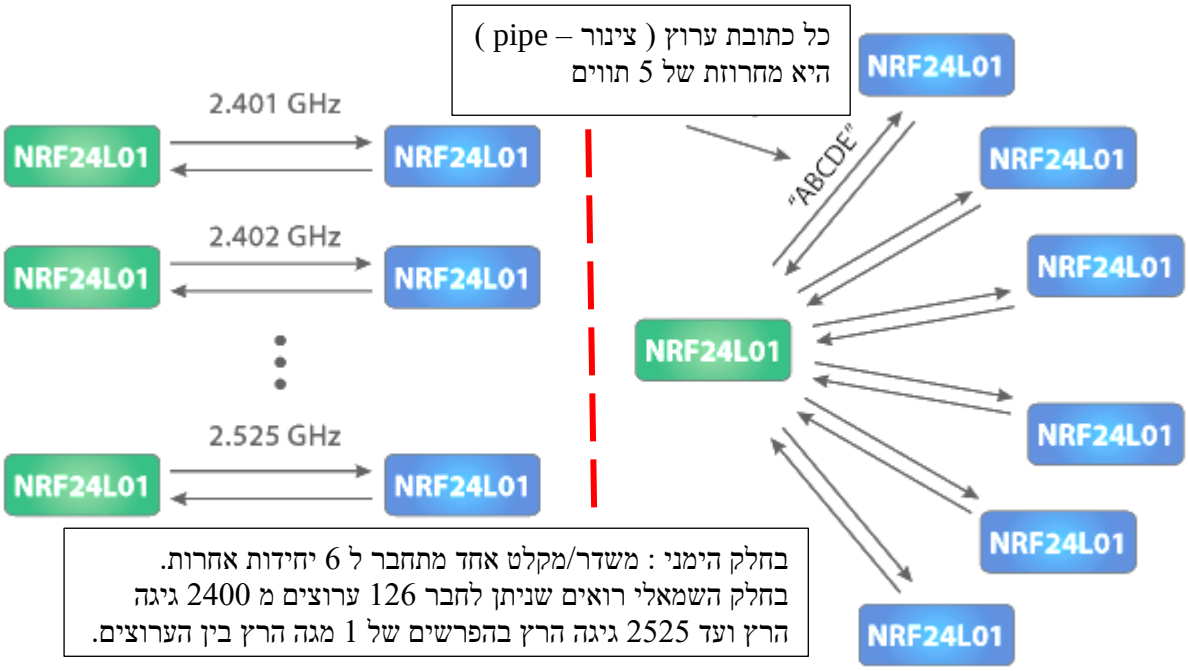
הדק 7 - MISO - הנתון הטורי מהעבד לאדון

הדק IRQ הוא $\overline{INT}$

במידה ואין מספיק הדקים חופשיים במסטר אז ניתן לחבר את הדק CE אל ה Vcc ולהשאיר את הדק IRQ פתוח ולבצע POLLING (שאילתה) .

פקודות SPI יכולות להיות בעלות אורך משתנה . הדק CSN חייב להיות ב 0 במשך כל זמן הפקודה ואז לעלות ל 1 בסיום הבייט האחרון. הבייט הראשון של הפקודה מגדיר את סוג הפקודה והרכיב מוציא את מצב רגיסטר הסטאטוס שלו. הבתים הבאים תלויים בסוג הפקודה והם יכולים להיות ערכים לכתוב לרגיסטרים ברכיב או נתוני payload (**payload** באנגלית הוא מטען ייעודי בעברית והכוונה לנתוני המשלוח – הנתונים הנשלחים) או בתים ריקים אם הפקודה קוראת את יציאת הרכיב.

המודול עובד בתחום תדר השימושי בראוטרים WiFi, בלוטות וחלק מהטלפונים האל חוטיים. תחום התדר הוא מ 2.400GHz ועד 2.525GHz , כלומר 0.125GHz או 125MHz . המרווח בין הערוצים הוא 1MHZכך שנקבל 126 ערוצים הממוספרים מ 0 ועד 125 . בדרך כלל תקשורת ה WiFi משתמשת בערוצים הנמוכים וכדאי לנו להשתמש ב 25 הערוצים הגבוהים יותר לפרויקטים שלנו. שני המשדרים משדרים וקולטים נתונים במנות – packets – של מספר בתים בזמן שידור/קליטה. קיים במודול תיקון שגיאה ושידור חוזר ויש אפשרות שיחידה אחת תתקשר עם עד 6 יחידות דומות באותו הזמן בחיבור כוכב . ספריית רשת ה RF24 מאפשרת הרחבה למספר "שכבות" של משדרים/מקלטים המחוברים ביניהם. באיור 4 ניתן לראות חיבור של משדר/מקלט אחד ל 6 משדרים PTX



איור 4 - חיבור מקלט אל 6 משדרים / מקלטים

השידור/קליטה מתבצע במנות. כל מנה יכולה להיות של בייט אחד עד 32 בייטים. ניתן לקבוע האם עובדים עם ACKnowledge – ACK – אישור או ללא אישור - NOACK . עבודה עם אישור אומרת שהמשדר משדר מנה וממתין לאישור המקלט שהוא קיבל את המנה . אם המקלט לא מודיע שמקבל, המשדר יישלח שוב את המנה עד לקבלת אישור מהמקלט. ניתן לקבוע את כמות השידורים החוזרים ואת הזמן בין שידור חוזר של מנה למנה. בין המשדר למקלט מתחבר PIPE - צינור (באופן דמיוני) - שיש לו כתובת ייחודית של 2 עד 4 בתים שהם כתובת בסיס ועוד בייט תחילי (Prefix). המנה של הנתונים נשלחת בעזרת הצינור. הצינור פועל בשיטת FIFO - First In First Out , כלומר הנתון שנשלח ראשון הוא זה שיצא ראשון.

א.5 מנה - PACKET ב ESB

מנה עם payload בין 0 ל 32 בייטים בשיטת ESB נראה באיור 5:

Preamble 1 byte	Address 3-5 byte	Packet Control Field 9 bit	Payoad 0-32 byte	CRC 1-2 byte

איור 5 : תיאור מנה ב ESB

ה **Preamble** הוא בייט פתיחה המשמש כרצף ביטים המשמש לגילוי 0 ו 1 במקלט. הוא יכול להיות או 01010101 או 10101010 . אם הביט הראשון של הכתובת הוא 1 אז הבייט של ה Preamble נקבע כ 10101010 ואם הביט הראשון של הכתובת 0 אז הוא נקבע כ 01010101 . הדבר בא להבטיח שיש מספיק מעברים ב Preamble לייצוב של המקלט .

ה **Address** - זוהי הכתובת למקלט. הכתובת מבטיחה שהמנה הנכונה אובחנה במקלט. הכתובת יכולה להיות 3 עד 5 בתים וקובעים אותה בפקודה ששולחים אל רגיסטר AW ברכיב.

ה **Packet Control Field** – שדה בקרת המנה - מכיל 9 ביטים. באיור 6 מתוארים 9 הביטים כאשר ביט ה MSB מתואר משמאל .

Payload Length 6 bit	PID 2 bit	NO_ACK 1 bit

איור 6 - שדה בקרת המנה - **Packet Control Field**

Payload Length הוא 6 ביטים המציינים מהי כמות הבתים הנשלחת במנה. המספר יכול להיות בין 0 ל 32 בתים. המספר 0 מציין שהחבילה ריקה ומשמש לבדיקת ACK (אישור). המספר 100000 מציין חבילה של 32 בתים. למספר 100001 מתייחסים כ DON'T CARE . שדה זה שימושי רק במצב שעובדים עם payload דינמי (במצב עבודה כזה כמות הבתים במנה יכול להשתנות ממנה למנה).

Packet Identification – PID - זיהוי המנה . 2 ביטים אלו משמשים לגילוי האם המנה הנקלטת היא חדשה או חוזרת (לדוגמה - אם לא היה אישור - ACK – מהמקלט והמשדר משדר שוב את המנה). כך מונעים הצגת אותה מנה של נתונים מספר פעמים במקלט. הבדיקה של ה PID מתבצעת ביחד עם

בדיקת ה CRC וכך המקלט יודע האם המנה שקלט תקינה. המשדר מקדם את ה PID בכל פעם שהוא משדר מנה פעם נוספת.

הביט **NO_ACK** - קיצור של NO ACKNOLEDGE - הביט שימושי במצב עבודה אישור אוטומטי - Auto ACKnoledge - אם יש בביט זה 1 זה אומר למקלט שהמשדר לא צריך ACK בסיום המנה הנוכחית.

Payload - זהו המשלוח של הנתונים שהמשדר שולח למקלט. כמות הבתים הנשלחת היא בין 0 ל 32 והיא משודרת לאוויר כמו שהיא רשומה בבתים.

ה CRC - Cyclic redundancy Check – בדיקת יתירות מחזורית - שהוא קוד לגילוי שגיאות. הוא יכול להיות בייט אחד או שני בתים. הוא מחושב על הכתובת , על שדה בקרת המנה - Packet Control Field וה payload עצמו. הפולינום המשמש ל CRC עם בייט אחד הוא :

$$X^8 + X^2 + X + 1$$

הערך ההתחלתי הוא : 0xFF . הפולינום לבדיקת CRC עם 2 בתים הוא :

$$X^{16} + X^{12} + X^5 + 1$$

הערך ההתחלתי כאן הוא 0xFFFF .

בשיטת ESB חבילה איננה מתקבלת אם בדיקת CRC לא הצליחה.

א.6 - סכימה מלבנית של רכיב nRF24L01

באיור 7 מתוארת סכימה מלבנית של הרכיב .

באיור רואים 4 חלקים המרכיבים את הרכיב.

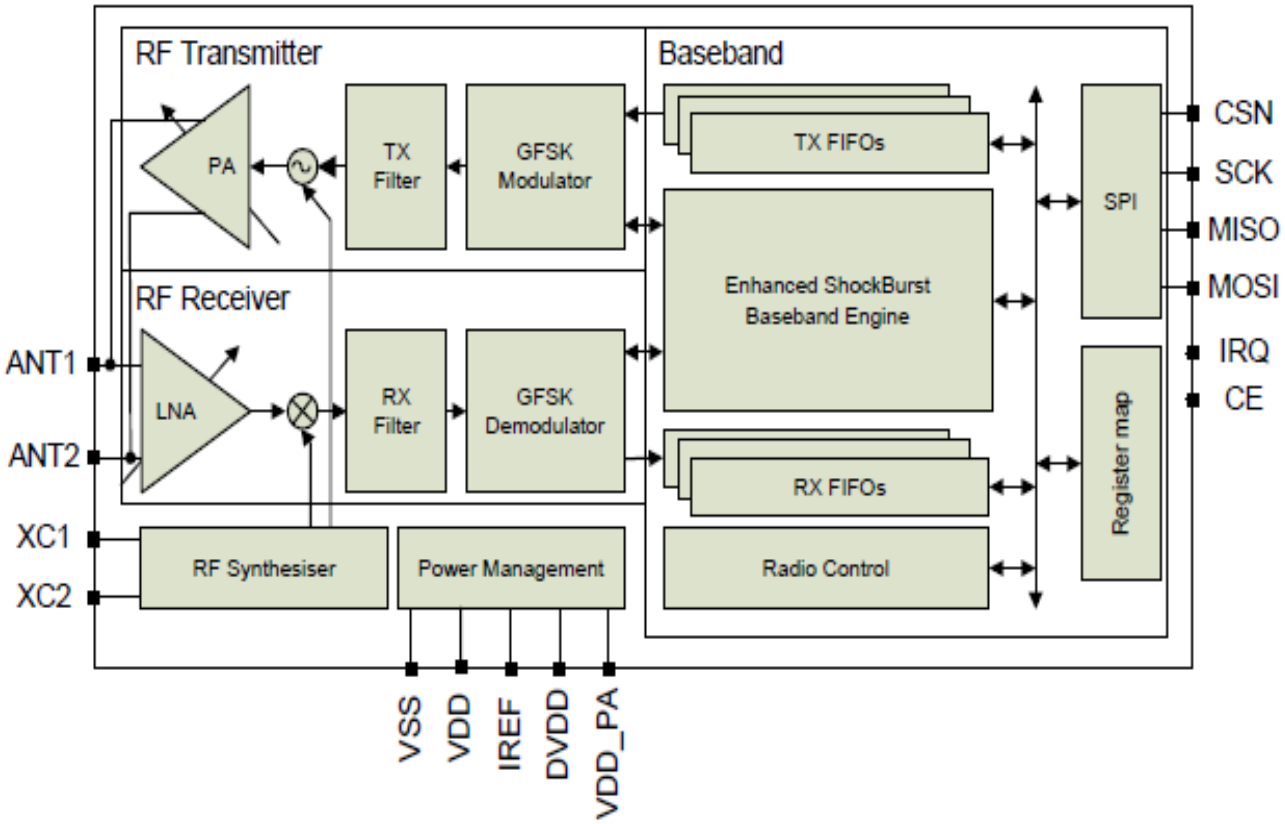
א.1. משדר RF - RF Transmitter

נראה בחלק העליון משמאל וכולל את המלבנים :

- GFSK Modulator - אפנן GFSK - Gaussian frequency-shift keying – שהוא אפנן cnp,uj vzz, ,sr dtuxh/
- TX Fikter - מסנן שידור
- PA שהוא Power Amplifier – מגבר הספק להוצאת האות המשודר לאנטנה. ניתן לתכנת אותו ל 4 רמות הספק שידור : 0dBm , -6dBm , -12dBm , -12dbM.

רמות ההספק הן :

- RF24_PA_MIN מתאים ל -18dBm 0.0158489 מילי וואט
- RF24_PA_LOW מתאים ל -12dBm 0.063 מילי וואט .
- RF24_PA_HIGH מתאים ל -6dBm 0.425 מילי וואט .
- RF24_PA_MAX מתאים ל 0dBm 1 מילי וואט.



איור 7 : סכמה מלבנית של רכיב nRF24L01 .

א.2. מקלט ה RF - RF Receiver .

נמצא מתחת למשדר ה RF וכולל את המלבנים הבאים :

- LNA - Low Noise Amplifier - מגבר רעש נמוך . תפקידו לקלוט את אות ה RF מהאנטנה ולהגביר אותו . זהו מגבר שניתן לקבוע את ההגברה שלו בתכנה.
- RX Filter - מסנן הקליטה
- GFSK Demodulator – גלאי GFSK

א.3. Power Management + RF Sythesizer

נמצאים מצד שמאל למטה בשרטוט . מתחת למקלט. המלבנים המרכיבים אותו הם :

- **Power Managment** - מנהל את ההספק הנצרך של הרכיב. ניתן לתכנת את הרכיב למצבי עבודה כמו : א. רגיל ב. standby ג. Power Down .
- **RF Sythesizer** - מכיל בתוכו מתנד שעון ייחוס , מסננת, מכפלי תדר, מחלקי תדר ומשווי פאזה. מצד שמאל שלו יש לו את היציאות XC1 XC2 המתחברות אל גביש חיצוני וקובעות את תדר הייחוס שלו. התדר שיקבל או יוציא הסינטסייזר הוא מכפלה או חלוקה של תדר שעון ייחוס זה. במודול nRF24L01 מתחבר בין הדקים אלו גביש של 16MHz .

א.4. Baseband - פס הבסיס

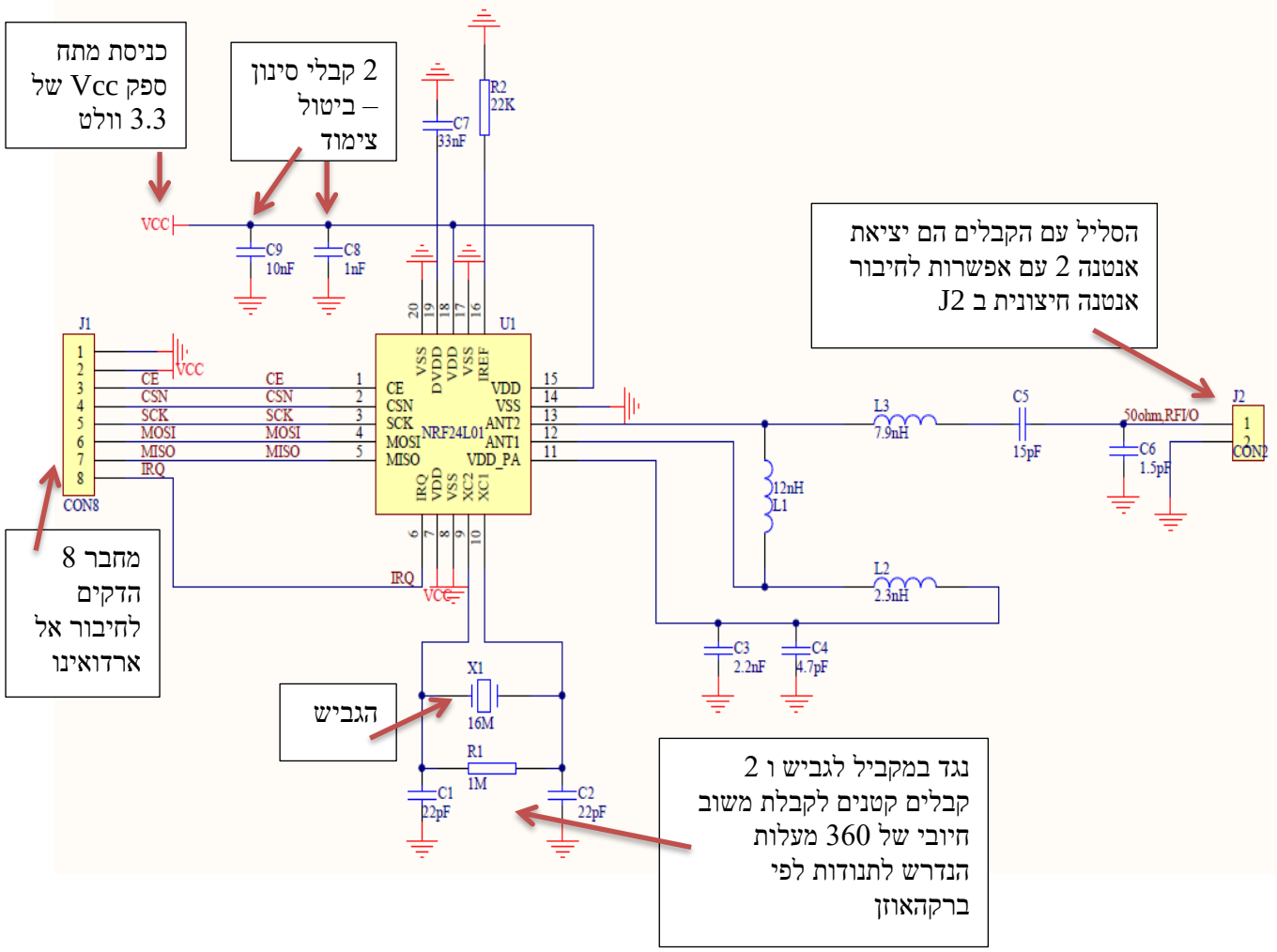
כל החלק הימני שבסכמה . מכיל את המלבנים הבאים :

- במרכז יש את Enhanced ShockBurst Baseband Engine - מנוע ה ESB . זהו המלבן ששולט על כל תהליך ה ESB הכולל את תקשורת ה SPI , ניהול חוצצי ה FIFO של השידור ושל הקליטה.
- מעל המלבן הזה יש את 3 חוצצי ה FIFO של השידור ומתחת למלבן יש את 3 חוצצי ה FIFO של הקליטה. בחוצצים אלו נמצא את המנה המשודרת או הנקלטת בהתאמה. כאשר יש שידור משודרים הנתונים שהכנסנו לחוצצי ה FIFO של השידור כאשר מנוע ה ESB דואג לשלוח את המנה לפי פרוטוקול ESB . בחוצצי ה FIFO של הקליטה נמצאות המנות שנקלטו.
- מימין למעלה נמצא את מלבן ה SPI שאליו מתחברים קווי ה SPI .
- Register Map – מפת הרגיסטרים היא מערכת המפענחת לאיזה רגיסטר אנחנו פונים ברכיב.
- Radio Control עוזר למנוע ה ESB לעבוד באופני העבודה השונים של הרכיב.

א.5 הסכמה החשמלית

באיור 8 מתאר את הסכמה החשמלית של הרכיב. השרטוט נעזר באתר :

https://github.com/pikipity/Home-Control/blob/master/datasheet/nrf24l01/NRF24L01%20RF%20Board/schematic/NRF24L01_SCH.pdf



איור 8 : סכמה חשמלית של רכיב nRF24L01

באיור 8 מתוארת הסכמה החשמלית של המודול עם הסברים על תפקיד הרכיבים שמסביב לרכיב NRf24L01 .

א.6 - חיבור מודול nRF24L01 לארדואינו

ישנן מספר ספריות אותן כתבו מתכננים שונים ובהן פונקציות עבור מודול זה וחיבורו אל כרטיסי הארדואינו. בספריות השונות משתמשים בהדקי המודול והם מתחברים אל כרטיסי הארדואינו המחוברים ל ATMEGA 328 או לאדוינו מגה בהדקים שונים. בטבלה 2 נראה את ההדקים של המודול והחיבור שלהם לכרטיסי הארדואינו השונים.

שם האות בתקשורת SPI	מספר ההדק במחבר שבמודול	השם של ההדק במודול	מספר ההדק בארדואינו בספריה TMRh20 RF24 Library	מספר ההדק בארדואינו בספריה RF24 Library	מספר ההדק בארדואינו בספריה Mirf Library	מספר ההדק בארדואינו מגה בספריה RF24 Library	מספר ההדק בארדואינו בהספרייה RH_NRF24 RadioHead Library	מספר ההדק בארדואינו מגה בספרייה RH_NRF24 RadioHead Library
GND	**1**	**GND**	**GND**	**GND ***	**GND**	**GND ***	**GND ***	**GND ***
VCC	**2**	**VCC**	**3.3 V**	**3.3V ***	**3.3V**	**3.3V ***	**3.3V ***	**3.3V ***
CE	**3**	**CE**	**7**	**9**	**8**	**9**	**8**	**8**
CSN	**4**	**CSN**	**8**	**10**	**7**	**53**	**10**	**53**
SCK	**5**	**SCK**	**13**	**13**	**13**	**52**	**13**	**52**
MOSI	**6**	**MO**	**11**	**11**	**11**	**51**	**11**	**51**
MISO	**7**	**MI**	**12**	**12**	**12**	**50**	**12**	**50**
IRQ	**8**	**IRQ**	**-**	**2**		**per library**	**N/C**	**N/C**

טבלה 2 – חיבורים בין הדקי המודול והדקי כרטיסי הארדואינו עבור ספריות שונות.

את חיבורי ההדקים של CSN ו CE ניתן להגדיר בתוכנה . לעומת זאת הקווים האחרים חייבים להיות :
הדק 13 של הארדואינו - SCK , הדק 11 של הארדואינו – MOSI, הדק 12 בארדואינו MISO .

כדאי לשים לב שחיבור ה Vcc הוא ל 3.3 וולט ולא ל 5 וולט אם כי ניתן לחבר אותו ללא חשש להדקי היציאות של הארדואינו שעובד עם 5 וולט.

הערה : רוב הבעיות לתקשורת מקוטעת (עם שגיאות) היא בגלל זרם לא מספיק או רעש על קו הספק של ה 3.3 וולט. מומלץ לשים קבל של 0.1Uf בין ה 3.3 וולט לאדמה (בכניסות המודול- אולי אפילו בהלחמה) . ניתן לחבר גם קבלים של 1 או 10 מיקרו פאראד.

ב. תוכנה וספריות לעבודה עם ארדואינו

ב.1 כללי

ישנן מספר ספריות העוסקות במודול. תחילה נתקין את הספרייה TMRH20 התומכת גם בארדואינו וגם ב RaspberryPi . הסברים של כותב/ת הספרייה ניתן למצוא בבלוג שלו :

http://tmrh20.blogspot.co.il/

אם יש בספריה של הארדואינו גרסה ישנה של הספרייה יש למחוק אותה. עבור לספריה ה דרך : github

https://github.com/TMRh20/RF24

בצד ימין במרכז יש בצבע ירוק clone or download . לחץ והורד את קובץ ה zip . אם תסתכל בספריה של הארדואינו תראה שכתוב שם : RF24-master.ZIP . יש לשנות את השם ל RF24.ZIP . הקלקה כפולה על הספרייה תראה לך ספריה הנקראת RF24_master . יש לשנות את שמה ל RF24 . אחרי הטענת הספרייה נתחיל עם **"GettingStarted"** הנמצא בספרית ה examples . יש להשתמש ב 2 כרטיסי ארדואינו כאשר לכל אחד מחובר מודול nRF24L01 . הרחק אותם מספר מטרים האחד מהשני . ניתן להריץ את 2 הכרטיסים על אותו מחשב , באותו IDE של הארדואינו כאשר לכל אחד יש COM משלו (לפי Tools -> Port) או ב 2 מחשבים שונים.

התהליך הוא הבא :

1. חבר מודול nRF24L01 לאחד מכרטיסי הארדואינו (היעזר בטבלה מספר 1).
2. התחל את תוכנת הארדואינו IDE . דאג לחיבור ה COM הנכון וטען את הקובץ GettingStarted.ino לכרטיס הארדואינו.
3. פתח את מסך המוניטור הטורי (צד ימין למעלה בצבע ירוק) וקבע את קצב התקשורת ל 115200 . כאשר תסגור את מסך המוניטור ותפעיל אותו שוב תקבל את ההודעות :

```
RF24/examples/GettingStarted
*** PRESS 'T' to begin transmitting to the other node
```

4. חזור על סעיף 1 לגבי המודול השני וכרטיס הארדואינו השני.
5. פתח תוכנה חדשה של הארדואנו IDE . דאג לחיבור ה COM הנכון , בצע קומפילציה וטען את הקובץ GettingStarted.ino לכרטיס הארדואינו השני.
6. פתח מסך מוניטור נוסף ואם צריך כוון אותו לקצב 115200 .
7. סגור את מסך המוניטור ובקוד התכנית השנייה שנה שורה אחת שתיראה כך :

```
1. /****************** User Config ***************************/
2. /***      Set this radio as radio number 0 or 1         ***/
3. bool radioNumber = 1;
4.
```

8. בצע קומפילציה וטעינה ללוח הארדואינו השני.
9. עכשיו פתח את שני מסכי המוניטור ורשום "T" בקצה חלון הארדואינו הראשון.
10. במסך המוניטור תקבל :

```
5. RF24/examples/GettingStarted
6. *** PRESS 'T' to begin transmitting to the other node
7. *** CHANGING TO TRANSMIT ROLE -- PRESS 'R' TO SWITCH BACK
8. Now sending
9. Sent 523513060, Got response 523511232, Round-trip delay 1828
   microseconds
10.        Now sending
11.        Sent 524519544, Got response 524517728, Round-trip delay 1816
   microseconds
```

ובמסך השני תקבל :

```
RF24/examples/GettingStarted
*** PRESS 'T' to begin transmitting to the other node
Sent response 523511232
Sent response 524517728
Sent response 525521620
```

ניתן להפוך את הכיוונים על ידי כתיבת "R" בצד המשדר ו "T" בצד השני ונקבל אותן הדפסות כמו מקודם אבל במסכי המוניטור ההפוכים.

ב.2 פונקציות (פקודות, מתודות) למודול

ספריה המטפלת במודול ניתן למצוא באתר :

https://tmrh20.github.io/RF24/classRF24.html

אנחנו נרשום את הפונקציות הנפוצות יותר.

ב.2.א קביעת הדקי הארדואינו לשידור או קליטה

RF24 (uint8_t _cepin, uint8_t _cspin)

קובעים את הדקי הארדואינו שיהיו CE ו CS . לדוגמה :

RF24 myRadio (7,8);

myRadio הוא מזהה , אובייקט, שנשתמש בו בכל התכנית. (ניתן לתת לו שם אחר כמובן) קבענו את הדק 7 של הארדואינו כהדק CE ואת הדק 8 כהדק CS .

ההדקים האחרים של התקשורת **לא מוגדרים וחייבים** להיות :

הדק 13 של הארדואינו - SCK , הדק 11 של הארדואינו – MOSI, הדק 12 בארדואינו MISO .

הערה : בתכנות הנקרא object oriented נשתמש בכל הפקודות שבהמשך באותו שם מזהה myRadio שאיתו יצרנו את הרדיו. תמיד יופיע המזהה ולאחריו נקודה ולאחריה הפקודה - מתודה

(method) - ולאחריה בסוגריים יופיעו פרמטרים. האובייקט במקרה הזה הוא Radio וישנן הרבה מתודות שיכולות לפעול על האובייקט. (המתודה היא כמו פונקציה או רוטינה בשפות נמוכות יותר).
דוגמה :

myRadio.begin( ) ;

המתודה היא begin . השורה שרשמנו מאתחלת את תקשורת הרדיו. נרשום שורה זו ב ( )setup .

מודול ה nRF24L01 משתמש ב pipes – צינורות – המחברים בין משדר למקלט. לצינור יש כתובת שצריך לקבוע אותה. לצינור של המשדר ולצינור המקלט חייבת להיות אותה כתובת. אח"כ ניתן להשתמש במספר צינורות בו זמנית.

ב.2.ב. דוגמה לפתיחת צינור תקשורת לקריאה

myRadio.openReadingPipe (uint_8t number, const uint8_t *address)

פתיחת צינור לקריאה.
ה number הוא מספר בין 0 ל 5 . ניתן לפתוח עד 6 צינורות . אנחנו נשים במספר הצינור בדרך כלל 1 .
Address - כתובת הצינור. זהו מערך של בתים דו ממדי של 5 שורות .
דוגמה: נגדיר מערך דו ממדי ונכניס נתון רק לשורה הראשונה :

byte addresses[ ] [6] = {"1Node"}; הגדרת הכתובת של הצינור הראשון

כאשר נרצה לפתוח את התקשורת בצינור הראשון נרשום :

myRadio.openReadingPipe(1,addresses[0]);

ניתן לרשום מספר צינורות לתקשורת.

ב.2.ב.1 - פתיחת המקלט , האזנה לנתונים נקלטים ושמירתם במערך.

אחרי שפתחנו בסעיף הקודם את צינור התקשורת לקריאה ניתן לפתוח את המקלט ולבדוק קליטת נתונים. **myRadio.startListening ();**

ואז ניתן לבדוק האם נקלטו נתונים מהמשדר:

if(myRadio.available())

ואם התשובה היא TRUE אז רושמים :

myRadio.read(&myData, sizeof(myData)) ;

myData הוא מערך בגודל רצוי לנו. sizeof(myData) אומר לקלוט כמות נתונים בגודל המערך myData . לדוגמה אם הגדרנו byte myData[8] ; אז נקלוט 8 נתונים.
ניתן להוסיף בסוף גם 0 אם רוצים אישור ACK (acknowledge) או 1 ללא אישור NOACK .

התוכנית תיראה כך :

```
if( myRadio.available())  // האם יש נתונים בחוצץ הקליטה (ניתן לרשום בסוגריים מספר צינור
 // ממנו רוצים למשוך נתונים) ?
{
while (myRadio.available())
{
myRadio.read( &myData, sizeof(myData) ) ;  קרא את הנתון והכנס אותו
// אם נרצה הדפסה של הנתון שנקלט נרשום את 2 השורות הבאות
Serial.print("Got data on pipe : ");
Serial.println(myData);
}
myRadio.stopListening();  // עצור את הקליטה (עצור האזנה )
```

ב.2.ג - שידור נתון

חוזרים על סעיף ב.2.ב אבל במקום קריאה מבצעים כתיבה.

byte addresses[ ] [6] = {"1Node"}; הגדרת הכתובת של הצינור הראשון

ואז רושמים שורה להשתמש בצינור שכתובתו במיקום 0 במערך addrsses אבל לכתיבה לצינור:

myRadio.openWritingPipe(1, addresses[0]);

ואז כאשר נרצה לכתוב נתון הנמצא ב myData נרשום :

myRadio.write(&myData, sizeof(myData));

ג. פונקציות (מתודות) נוספות

נציין מספר מתודות נוספות שימושיות.

ג.1 קביעת קצב התקשורת

radio.setDataRate(RF24_250KBPS);

קצב התקשורת נקבע ל 250kbs (קילו ביטים בשנייה). ניתן לרשום גם RF24_1MBPS לקצב של 1 מגה ביטים בשנייה או RF24_2MBPS ל 2 מגה ביטים בשנייה . כדאי לקחת ערך תקשורת נמוך ואז טווח השידור הוא הגדול ביותר .

ג.2 קביעת רמת הספק שידור

כיוון מגבר ההספק (PA – Power Amplifier) לאחת מ 4 רמות :

radio.setPALevel(RF24_PA_MAX);

רמות ההספק הן :	RF24_PA_MIN	מתאים ל -18dBm	0.0158489 מילי וואט
	RF24_PA_LOW	מתאים ל -12dBm	0.063 מילי וואט .
	RF24_PA_HIGH	מתאים ל -6dBm	0.425 מילי וואט .
	RF24_PA_MAX	מתאים ל 0dBm	1 מילי וואט.

לתזכורת : את חישוב ההספק עושים בעזרת הנוסחאות :

$$xdbm = 10 \log_{10} Pout/1mw$$

או אם נעביר אגפים נקבל את ההספק ביחידות של וואט :

$$P = 10^{(x-30)/10}$$

במודול שיש לו הספק גדול יותר והמכיל מגבר הספק וקדם מגבר לקליטה יש אפשרות ל 20dBm (הגבר פי 100) ואז ניתן לשדר בהספק של 100 מילי וואט.

ג.3 קביעת הערוץ הרצוי

ניתן לבחור 126 ערוצים שונים עם תדרים בין 2.4 ל 2.524 ג'יגה הרץ עם רזולוציה של 1MHz בין ערוץ אחד לשני. כדאי לבחור ערוץ מעבר ל 100 . (נשים מספר בין 0 ל 125). לדוגמה :

radio.setChannel(118);

נעבוד כאן עם ערוץ 108 שהוא בתדר 2400 +110 = 2510MHz , כלומר 2.51GHz .

הערה : חלק מהמדינות אוסר על שימוש בתדר מעבר ל 2.483GHz ואז יש לבחור ערוץ קטן מ 83 . כדאי לבצע בדיקה של מספר ערוצים ולראות איפה יש ערוץ "נקי" ללא תקשורת של מערכת נוספת.

תוכנה להפעלת סריקה כזו ניתן למצוא באתר

http://arduino-info.wikispaces.com/Nrf24L01-Poor+Man%27s+2.4+GHz+Scanner

ג.4 קביעת כמות ניסיונות שידור חוזרים וההשהיה בין השידורים

ניתן לקבוע את כמות הניסיונות לשדר פעם נוספת - (retries) - את המנה אשר המקלט לא קלט . כמו כן ניתן לקבוע את זמן ההשהיה בין שידור מנה אחת לאחרת (במכפלות של 250 מיקרו שניות).

setRetries(uint8_t delay,uint8_t count)

לדוגמה הפקודה : **radio.setRetries(10,15);** אומרת שיתבצעו עד 15 ניסיונות לשידור חוזר כאשר בין כל שידור לשידור תתבצע השהיה של 10*250USec , כלומר 2500 מיקרו שניות שהם 2.5 מילי שניות.

ג.5 קביעת אורך CRC

בדיקת CRC (בדיקת יתירות מחזורית - Cyclic Redundancy Check) היא קוד לגילוי שגיאות בהעברת נתונים. לפני העברת המידע מחושב ה CRC ומתווסף למידע המועבר (ראה פרק א.5) . הצד הקולט בודק את ה CRC ואם הוא נכון הוא מאשר שהמידע הועבר ללא שגיאות.

radio.setCRCLength(פרמטרים לאורך ה CRC);

הפרמטרים האפשריים הם : RF24_CRC_8 עבור 8 ביט או RF24_CRC_16 עבור 16 ביט.

לא ניתן לבטל בדיקת CRC אם קבענו auto ack (אישור אוטומטי) .

ג.6 קבלת אורך CRC

radio.getCRCLength( );

הערך שיוחזר הוא אחד מהשלושה הבאים : RF24_CRC_DISABLED אם לא אפשרנו בדיקת CRC או RF24_CRC_8 עבור 8 ביט או RF24_CRC_16 עבור 16 ביט.

ג.7 אפשור או מניעה של אישור אוטומט (auto ack) של מנה

setAutoAck (0 או 1)

ברירת המחדל היא אפשור. אם נשים 0 בביט בסוגריים שבפקודה לא נאפשר אישור אוטומטי.

ג.8 אפשור Payload דינמי

radio.enableDynamicPayloads( );

כאשר לא מעוניינים לשדר כל הזמן מנות גדולות אלא לשדר מנות קטנות יותר נשתמש בפקודה הזו. זה מאפשר payload דינמי וזו דרך נוחה להחזיר נתונים חזרה לשולח ללא צורך לשנות את אופני העבודה ב 2 היחידות.