פרק 9 : תקשורת טורית - ה UART במיקרו בקר C8051F380

9.1 מבוא

שתי שיטות השידור להעברת נתונים בין 2 מערכות הן תקשורת מקבילית ותקשורת טורית. בתקשורת מקבילית משדרים במקביל על 8 חוטים BYTE ביית - על כל חוט ביט. בתקשורת טורית יש רק 2 חוטים ומשדרים את ה BYTE ביית - ביט אחרי ביט. מכאן ברור שיתרון התקשורת המקבילית על התקשורת הטורית הוא במהירות העברת הנתונים ואילו החיסרון בתקשורת המקבילית הוא מחיר (יותר חוטים).

9.1.1 אופני העברת נתונים

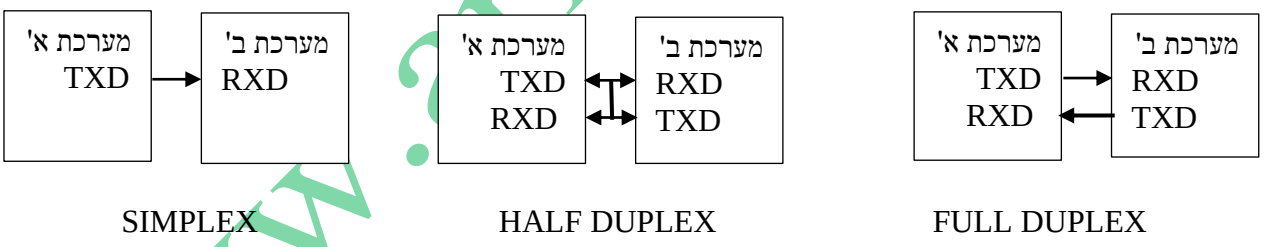
ישנם 3 אופני העברת נתונים : א. SIMPLEX ב. HALF DUPLEX ג. FULL DUPLEX . אופן העברת נתונים מתאר את הכיוון של זרימת האות בין שתי מערכות.

באופן **SIMPLEX** ההעברה של התקשורת היא חד כיוונית. רק מכשיר אחד יכול להעביר את הנתונים לשני. המכשיר השני יכול רק לקלוט . לדוגמה : לוח קשים של מחשב שרק מעביר נתונים למחשב. צג מחשב שרק מקבל נתונים ומציג אותם.

באופן **HALF DUPLEX** - חצי דו כיווני – העברת הנתונים היא דו כיוונית אבל לא באותו זמן. ניתן להעביר נתונים ממערכת א' למערכת ב' ובסיום מערכת ב' מעבירה את הנתונים למערכת א'. לא ניתן לשדר ולקלוט בו זמנית. דוגמאות : מכשירי - *Walkie Talkies* *או מכשירי קשר צבאיים שבהם הצד המשדר והקולט מתחלפים מאחד לשני – "מספר 2 האם שומע ? עבור..."* ואז מספר 2 עונה " מספר 2 שומע.. עבור .." והצד המשדר והצד הקולט מתחלפים בינהם .

באופן **FULL DUPLEX** – דו כיווני מלא – כל מערכת יכולה לשדר ולקלוט בו זמנית. לדוגמה: טלפון.

באיור הבא מתוארים 3 אופני העבודה.



איור 9.1 3 אופני החיבור של תקשורת טורית.

9.1.2 מהו UART ?

בתוך המיקרו בקר קיימת מערכת לתקשורת טורית הנקראת **UART** - **Universal Asynchronous Receiver Transmitter** - **מערכת משדר מקלט א-סינכרונית אוניברסאלית**. מערכת זו יודעת לקבל ביית מהמיקרו ולשדר אותו בצורה טורית בהדק שנקרא TxD של המיקרו. בסיום השידור של ה ביית , ה UART מודיע למיקרו שהוא סיים לשדר את הביית והוא מוכן לקבל ביית חדש לשידור. ה UART משדר את הביית לפי פרוטוקול תקשורת טורית : קודם את ביט ההתחלה – Start Bit , אח"כ את סיביות הנתון ולבסוף את ביט הזוגיות - PARITY (אם הוחלט שעובדים עם זוגיות) ואת ביט הסיום Stop Bit . כפי שהזכרנו - בסיום שידור התו ה UART מודיעה למיקרו (גם בעזרת פסיקה וגם על ידי דגל TI) על סיום שידור התו. כמובן

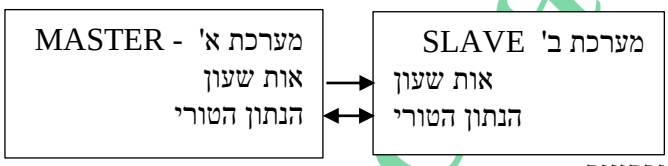
שהתוספת של ביט ההתחלה וביט הסיום מאטים עוד יותר את קצב התקשורת ולכן הומצאו שיטות חדשות יותר (יוסבר בפרקים הבאים של הספר).

בקליטה טורית קורה תהליך הפוך. המיקרו מקבל בהדק שנקרא RxD את הביטים הטוריים אחד אחרי השני. ה UART מזהה את ביט ההתחלה, את הביטים של הנתון , את הביט של הזוגיות (אם הוחלט לעבוד עם זוגיות) ואת ביט הסיום ומודיע למיקרו על ידי פסיקה וגם על ידי דגל RI, על סיום קליטה טורית. המיקרו יקרא את הביית שנקלט וה UART יהיה מוכן לקליטת ביית חדש. באופן עבודה כזה שה UART מבצע את פעולת השידור והקליטה , המיקרו פנוי לטפל בדברים נוספים ולא צריך לנהל את שידור או קליטת הנתונים הטורית.

9.1.3 תקשורת אסינכרונית וסינכרונית

תקשורת טורית – UART - היא תקשורת **אסינכרונית** כי אין לנו תדר שעון מרכזי המסנכרן את מעבר הנתונים.

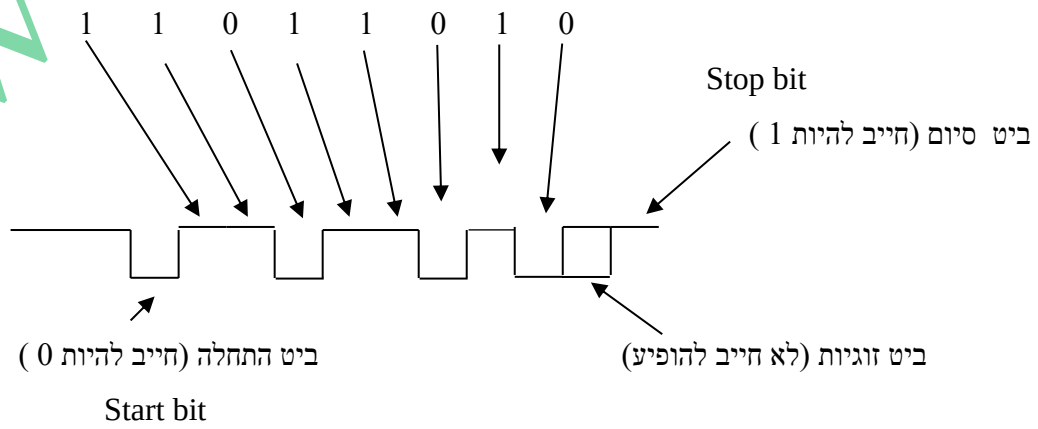
קיימות מערכות תקשורת טורית נוספות כמו I2C או SPI שהן מערכות תקשורת טורית **סינכרוניות** . במערכות אלו קיים אדון ועבד/ים MASTER - SLAVE ויש תדר שעון מרכזי של מערכת האדון MASTER שנותן פולסי שעון לסנכרון לכל מערכות העבד – SLAVE - המתחברות אליו. האיור הבא מתאר תקשורת טורית **סינכרונית**:



איור 9.2 : תקשורת טורית סינכרונית

9.1.4 שידור נתון טורי

נזכיר כיצד נשלח נתון בקו תקשורת טורית. נניח שרוצים לשדר את התו 5BH כלומר 01011011 . התו משודר מהביט הנמוך אל הגבוה (מה LSB אל ה MSB). במצב שאין שידור הקו נמצא בגבוה . התחלת שידור מתבצעת עם הורדת הקו ל 0 לזמן של ביט. ביט זה נקרא ביט התחלה – start bit . לאחר ביט ההתחלה משודרים 8 הביטים של הנתון מהביט הנמוך D0 ועד הביט הגבוה D7 . אם הוחלט לעבוד עם זוגיות אז משודר גם הביט התשיעי (במידה ולא אז אין ביט תשיעי) ולבסוף הקו עולה ל 1 ומציין סיום שידור. ביט זה נקרא ביט סיום/עצירה - stop bit . ביית חדש שישודר מתחיל שוב עם ביט התחלה ואחר כך הביטים של הנתון וביט סיום . האיור הבא מראה שידור של ביית .



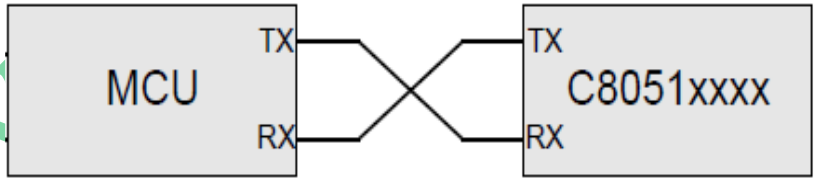
איור 9.3 : שידור הנתון 5Bh בתקשורת טורית .

9.1.5 מאפייני השידור

כמות הביטים הנשלחת בשנייה נקראת **קצב התקשורת או קצב הביטים** - Bit Rate. היחידות הן **סיביות לשנייה או סל"ש.** **באנגלית Bits Per Second – bps** . קצבי שידור מקובלים בתקשורת טורית : 4800, 9600, 19200, 28800, 57600, 115200 ו 230400 ביטים בשנייה.

בתחילת ימי התקשורת הטורית היה מושג נפוץ הנקרא באוד - BAUD - לתיאור של קצב העברת המידע. הוא ייצג את כמות הסמלים בשנייה. סמל הוא יחידת מידע שיכולה לייצג יותר מביט אחד. הבאוד הוחלף במושג קצב ביטים הנמדד בביטים לשנייה.

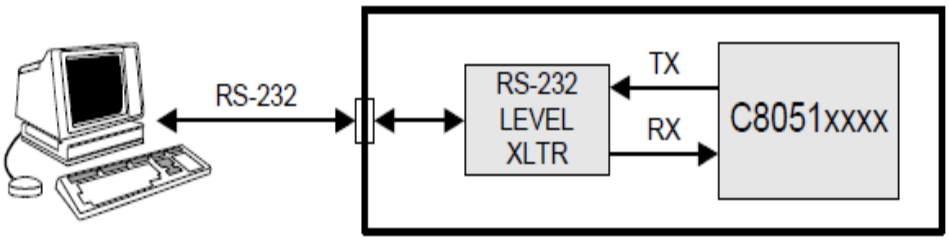
כמות הביטים של הנתון במיקרו שלנו יכולים להיות 8 או 9 ביטים בכל מסגרת (כלומר בין ביט ההתחלה לביט הסיום). ביט הזוגיות הוא ביט שמציין את כמות ה 1 שיש בנתון. בתחילת ימי התקשורת הטורית היו משתמשים בביט הזוגיות כדי שהצד הקולט יידע האם הוא קלט את המידע נכון. לדוגמא : נניח שמשדרים את התו 5bH . בתו יש 5 פעמים 1 . היות וכמות ה 1 איננה זוגית היו שמים בביט הזוגיות 0 . הצד הקולט סופר את כמות ה 1 שקלט. היות והוא קלט 5 פעמים 1 הוא יודע שביט הזוגיות שיקלוט חייב להיות 0 . אם הוא קלט ביט זוגיות 0 מסקנה – הקליטה נכונה. אם אחד הביטים היה מתהפך בדרך אז הצד הקולט היה קולט מספר זוגי של 1 ואז היה רואה שביט הזוגיות הוא 0 והיה מבחין שקלט נתון שגוי. במקרה כזה הוא יכול לבקש מהצד המשדר לשלוח את הנתון פעם נוספת. הוספת ביט הזוגיות גורם לביט טורי נוסף ל 8 הביטים של המידע, דבר שמאט את קצב התקשורת הטורית, הנמוך ממילא. היום יש שיטות חדשות לאיתור שגיאות ואפילו לתיקון שגיאות. ניתן לקבוע את זמן ביט הסיום לאורך זמן של 1, 1.5 או 2 סיביות. סט הפרמטרים הנפוץ ביותר הוא **8N1** שמשמעותו '8' ביטים של מידע , 'N' - ללא סיבית זוגיות, '1' - אורך ביט הסיום היא זמן ביט יחידה. על-מנת לקיים תקשורת תקינה על שני הצדדים לכוון לאותו סט של פרמטרים, למשל 9200 - 8N1 1 (19200 קצב הביטים בשנייה). באיור הבא רואים חיבור של המיקרו C8051Fxx אל מיקרו בקר אחר. החיבור הוא FULL DUPLEX. רואים שהדק השידור של מיקרו אחד מתחברת להדק הקליטה של המיקרו השני ולהפך.



איור 9.4 חיבור בין 2 מיקרו בקרים בתקשורת טורית.

תקן לשידור טורי נקרא גם RS-232 או EIA RS-232 (RS קיצור של Recommended Standard – סטנדרט/תקן מומלץ) והוא משמש להעברת נתונים בין מכשיר DTE – Data Terminal Equipment – ציוד קצה נתונים - כמו מחשב ובין מכשיר DCE – Data Circuit Equipment – ציוד מעגל נתונים – הקולט את המידע. התקן עוסק ברמות מתח של שליחת הנתונים. רמה לוגית של 0 מיוצגת על ידי רמת מתח בין 7 וולט ל 15 וולט ורמה לוגית של '1' מיוצגת על ידי מתח בין 7- עד 15- וולט. הרמה הלוגית 0 גבוהה מזו של ה 1 וזה נקרא לוגיקה שלילית. כמו כן התקן קובע כיצד ייראה המחבר (קונקטור) של התקשורת ועוד.

באיור הבא רואים חיבור בין מחשב ובין מיקרו בקר ממשפחת SILABS עם מתאם רמות מתח מ TTL ל RS232 - המלבן נקרא באיור RS - 232 LEVEL XLTR.



איור 9.5 חיבור בין מיקרו בקר ובין מחשב עם ממיר רמות מתח מ TTL ל RS232 .

בצד הקולט רואים שהקו בגבוה "ומבינים" שהצד השני מחובר . (אם ללא שידור היה 0 לא היינו יודעים האם הצד השני מחובר). ברגע שהצד הקולט מרגיש 0 הוא ממתין זמן של חצי ביט ושוב דוגם את הקו לראות שיש עדיין 0 (להיות בטוח שה 0 הקודם לא היה רעש אקראי) . לאחר מכן הוא דוגם את הקו כל זמן של ביט ואוסף את 8 הביטים של הנתון (ואם יש גם ביט זוגיות אז 9 ביטים) , דוגם את הקו כדי לוודא שיש 1 שזהו ביט הסיום ומודיע למיקרו שנקלט ביית . המיקרו צריך לקרא את הביית שנקלט כדי שהמלט יוכל לקלוט את הביית החדש שישודר.

9.2 התקשורת הטורית במיקרו C8051F380

במיקרו בקר הבסיסי ממשפחת ה 51 היה רק UART אחד בלבד. ב C8051F380 יש 2 מערכות UART הנקראות UART0 ו UART1 . הן עובדות במקביל וכל אחת עצמאית ללא קשר לשנייה. UART0 דומה בפעולתו ל UART שהיה ב 51 המקורי. UART1 עובד בצורה גדולה יותר, יש לו 6 רגיסטרים ב SFR עם אפשרות לקלוט 3 נתונים ללא קריאה של המיקרו בקר. בספר זה נעסוק רק ב UART0 .

9.2.1 UART0

היחידה UART0 היא מערכת תקשורת טורית , אסינכרונית , FULL DUPLEX , המאפשרת את אופני העבודה 1 ו 3 של המיקרו 8051 המקורי. אופן 1 הוא שידור טורי של 8 ביטים ואופן 3 שידור של 9 ביטים בקצב שידור / קליטה משתנים (קצב הנשלט על ידי המשתמש) . במיקרו C8051F380 יש תמיכה עשירה של מקורות תדר שעון ליצירת קצבי שידור סטנדרטיים (נראה בטבלה בעמודים הבאים) . מערכת חוצצי נתונים מאפשרת להתחיל לקלוט ביית נוסף עוד לפני שהמיקרו קרא את הנתון הקודם.

ל UART0 יש 3 רגיסטרים ב SFR .

א. SCON0 - Serial Control Register 0 - רגיסטר בקרה טורית 0

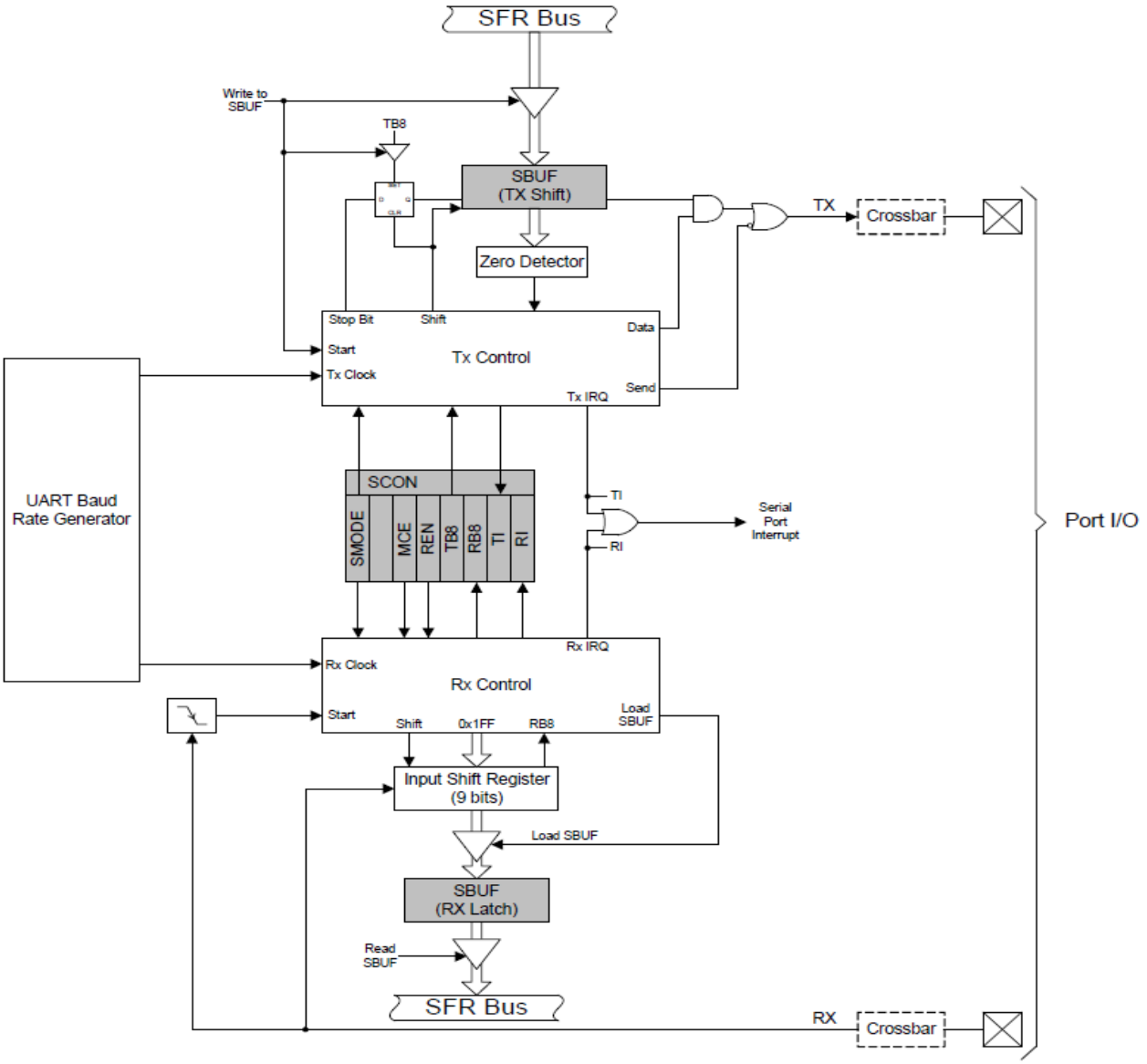
ב. SBUF0 - Serial BUFfer 0 – חוצץ טורי 0 . למעשה יש 2 חוצצים טוריים SBUF0 , אחד לשידור ואחד לקליטה. כאשר המיקרו כותב נתון ל SBUF0 הוא פונה ל SBUF0 של השידור . כאשר המיקרו קורא נתון הוא קורא מה SBUF של הקליטה.

במיקרו 51 המקורי היו 3 רגיסטרים השייכים לתקשורת הטורית. רגיסטר ה SCON השולט על אופן העבודה של התקשורת (כמות סיביות, זוגיות, קצב תקשורת) ושני רגיסטרים של SBUF הנמצאים בכתובת 99H באזור ה SFR. אחד מקבל את הבית לשידור כאשר המיקרו כותב אליו ואז הוא משדר אותו והשני מכיל את הבית שנקלט והמיקרו קורא אותו. ההבדל בגישה אליהם מתבצע בעזרת קווי קריאה /כתיבה פנימיים. לא ניתן לקרוא נתון מה SBUF0 של השידור ולא ניתן לכתוב ל SBUF0 של הקליטה.

כאשר UART0 מסיים לשדר את הנתון הוא מבקש פסיקת תקשורת טורית (SERIAL) וביט TI ב SCON0 עולה ל 1 . דבר דומה קורה בקליטה. כאשר הסתיימה קליטה של נתון יש בקשת פסיקה טורית וביט RI ב SCON0 עולה ל 1 . היות וגם סיום השידור וגם סיום קליטה יוצרים אותה בקשת פסיקה (כתובת 23H - פסיקה מספר 4) , על המתכנת לבדוק את הביטים TI ו RI כדי לדעת מאיזו סיבה יש בקשת פסיקה (האם סיום שידור או סיום קליטה). יש לזכור שכאשר המיקרו פונה לטיפול בפסיקה הביטים RI ו TI אינם מתאפסים על ידי החומרה ויש לאפס אותם בתוכנה.

9.2.2 סכמה מלבנית של UART0

באיור הבא מתוארת הסכמה המלבנית של UART0 .



איור 9.6 הסכמה המלבנית של UART0 .

- החלק העליון באיור, מהמלבן Tx Control ומעלה זהו המשדר של ה UART. בחלק העליון שלו נמצא ה SBUF של השידור שהוא רגיסטר הזזה. במלבן Tx Control יש מעגלי אלקטרוניקה השולטים על רגיסטר ההזזה ומוציאים ממנו את הנתון לפי הפרוטוקול הטורי שהסברנו עם ביט התחלה וסיום. תחילת השידור הטורי מתבצעת עם העברת הנתון ל SBUF.

- החלק התחתון של האיור, מהמלבן Rx Control כלפי מטה - זהו המקלט של ה UART. בחלק התחתון שלו נמצא ה SBUF של הקליטה. ה Rx Control הוא מערכת בקרה הכולל מעגלי אלקטרוניקה השולטים על רגיסטר הזזה של 9 ביט - Input Shift Register - הקולט את הנתון הטורי ביט אחרי ביט כולל הזזה. הקליטה מגיעה כאשר מזהים ירידה מ 1 ל 0 בהדק RX. ירידה זו נכנסת לרגל START של מלבן ה RX Control . כל ביט שנקלט מקבל הזזה ימנית בעזרת הקו Shift . בסיום קליטת הנתון מערכת הבקרה נותנת פולס Load SBUF לחוצץ (מצויר כמשולש) שמעביר את הנתון שנקלט ברגיסטר ההזזה ל SBUF של הקליטה. כמו כן מוציאה מערכת הבקרה בקו Rx IRQ בקשת פסיקה של תקשורת טורית.

- בין מערכת השידור והקליטה נמצא רגיסטר ה SCON - Serial CONtrol - בקרה טורית אשר מבקר על התקשורת הטורית.

- באמצע משמאל נמצא המלבן UART Baud Rate Generator - מחולל קצב הבאוד של ה UART. זהו המלבן שנותן את תדר השעון להפעלת המשדר והמקלט . אות השעון שהוא נותן למערכת השידור והקליטה נקרא Tx Clock ו Rx Clock בהתאמה.

- אפשר לראות שהגישה ל SBUF של השידור היא בעזרת אות כתיבה - Write to SBUF ואילו הגישה ל SBUF של הקליטה היא בעזרת אות קריאה - Read SBUF .

- בחלק הימני ביותר של האיור יש את קו ה TX (למעלה באיור) ואת קו ה RX (למטה באיור) המתחברים בעזרת הקרוסבר להדק המתאים של הרכיב .

- מימין ל SCON יש שער OR המתחבר אל Tx IRQ ואל Rx IRQ . כאשר ה UART סיים לשדר את הביית שנמצא ב SBUF של השידור הוא מוציא '1' אל הדק Tx IRQ וגם לביט TI ב SCON . ביציאת שער ה OR יש '1'. יציאה זו מתחברת למערכת הפסיקות ויש בקשת פסיקה של UART . דבר דומה קיים במערכת הקליטה של ה UART . כאשר ה UART סיים לקלוט נתון הוא מוציא '1' אל שער ה OR וגם לביט RI ב SCON . ביציאת שער ה OR יהיה '1' ונקבל בקשת פסיקה. אם נאפשר פסיקת תקשורת טורית אז כאשר המיקרו פונה לתוכנית הפסיקה, יש לבדוק האם הפסיקה היא של סיום שידור או סיום פסיקה ובהתאמה לטפל בפסיקה. היות והתקשורת הטורית היא FULL DUPLEX יכול להיות מצב שנקבל פסיקה גם מהמשדר וגם מהמקלט בו זמנית. יש לזכור שבתוכנית הפסיקה יש לאפס את ביט ה TI או/ו ה RI בתוכנה כדי להכין אותם למחזור שידור/קליטה חדשים.בצד ימין של האיור רואים את הדקי השידור Tx והקליטה Rx של יחידת ה UART. כדי שהדקים אלו יצאו אל הדקי הרכיב יש לאפשר את הקרוסבר בעזרת ביט XBARE של xbr1 . הפקודה היא ; XBR1=0x40 . כמו כן יש לאפשר ברגיסטר ה xbr0 שבאיור הבא את הביט URT0E ולשים בו 1 כדי שהדק השידור יתחבר אל P0.4 והדק הקליטה ל P0.5 . הדקים אלו קבועים ולא ניתן לשנות אותם.

Bit	7	6	5	4	3	2	1	0
Name	CP1AE	CP1E	CP0AE	CP0E	SYSCKE	SMB0E	SPI0E	URT0E

איור 9.7 : רגיסטר XBR0 : 0 – הדקי התקשורת של ה UART לא מתחברים. 1 – מתחברים להדקי P0.4 (Tx) ו P0.5 (Rx)

9.2.3 רגיסטר בקרת התקשורת הטורית – SCON0

באיור הבא מתואר רגיסטר ה **SCON0** - Serial CONtrol 0 - בקרה טורית 0. בעזרתו שולטים על התקשורת הטורית. איתו קובעים האם עובדים עם 8 או 9 ביט, האם עובדים עם UART בודד או כמה מעבדים עם UART .

נסביר כל ביט בעזרת האיור הבא :

Bit	7	6	5	4	3	2	1	0
Name	S0MODE	-	MCE0	REN0	TB80	RB80	TI0	RI0
Type	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	1	0	0	0	0	0	0

SFR Address = 0x98; SFR Page = All Pages; Bit-Addressable

Bit	Name	Function
7	S0MODE	**Serial Port 0 Operation Mode.** Selects the UART0 Operation Mode. 0: 8-bit UART with Variable Baud Rate. 1: 9-bit UART with Variable Baud Rate. בחירת אופן הפעולה עם פורט טורי 0 (כמות סיביות מידע): 0 – UART של 8 ביט עם קצב תקשורת משתנה. 1 – UART של 9 ביט עם קצב תקשורת משתנה. קצב תקשורת משתנה אומר שהמשתמש בעזרת תוכנה קובע את הקצב.
6	Unused	Read = 1b, Write = don't care. ללא שימוש.
5	MCE0	**Multiprocessor Communication Enable.** The function of this bit is dependent on the Serial Port 0 Operation Mode: **Mode 0: Checks for valid stop bit.** 0: Logic level of stop bit is ignored. 1: RI0 will only be activated if stop bit is logic level 1. **Mode 1: Multiprocessor Communications Enable.** 0: Logic level of ninth bit is ignored. 1: RI0 is set and an interrupt is generated only when the ninth bit is logic 1. אפשור תקשורת מרובת מעבדים. 0 – המיקרו מתחבר אל UART אחד. 1- המיקרו מתחבר אל מספר UART. מצב הקיים רק בעבודה עם 9 ביט
4	REN0	**Receive Enable.** 0: UART0 reception disabled. 1: UART0 reception enabled. אפשור פסיקה. 0 – חסימת קליטה. 1 - אפשור קליטה
3	TB80	**Ninth Transmission Bit.** The logic level of this bit will be sent as the ninth transmission bit in 9-bit UART Mode (Mode 1). Unused in 8-bit mode (Mode 0). הביט ה 9 שישודר בעבודה עם UART של 9 ביטים.
2	RB80	**Ninth Receive Bit.** RB80 is assigned the value of the STOP bit in Mode 0; it is assigned the value of the 9th data bit in Mode 1. הביט ה 9 שנקלט בעבודה עם UART של 9 ביטים.
1	TI0	**Transmit Interrupt Flag.** הסבר בהמשך Set by hardware when a byte of data has been transmitted by UART0 (after the 8th bit in 8-bit UART Mode, or at the beginning of the STOP bit in 9-bit UART Mode). When the UART0 interrupt is enabled, setting this bit causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software.
0	RI0	**Receive Interrupt Flag.** הסבר בהמשך Set to 1 by hardware when a byte of data has been received by UART0 (set at the STOP bit sampling time). When the UART0 interrupt is enabled, setting this bit to 1 causes the CPU to vector to the UART0 interrupt service routine. This bit must be cleared manually by software.

איור 9.8 : רגיסטר SCON0

RI0 Receive Interrupt flag - דגל פסיקת קליטה

הביט עולה ל 1 על ידי החומרה של ה UART כאשר הסתיימה קליטה טורית של ביית. המתכנת יבדוק את הביט . אם יש בו 1 יודעים שנקלט בית חדש. יש למשוך את הבית מה SBUF ולאפס את הביט בתוכנה (CLR RI) כדי להתכונן לקליטת ביית חדש.

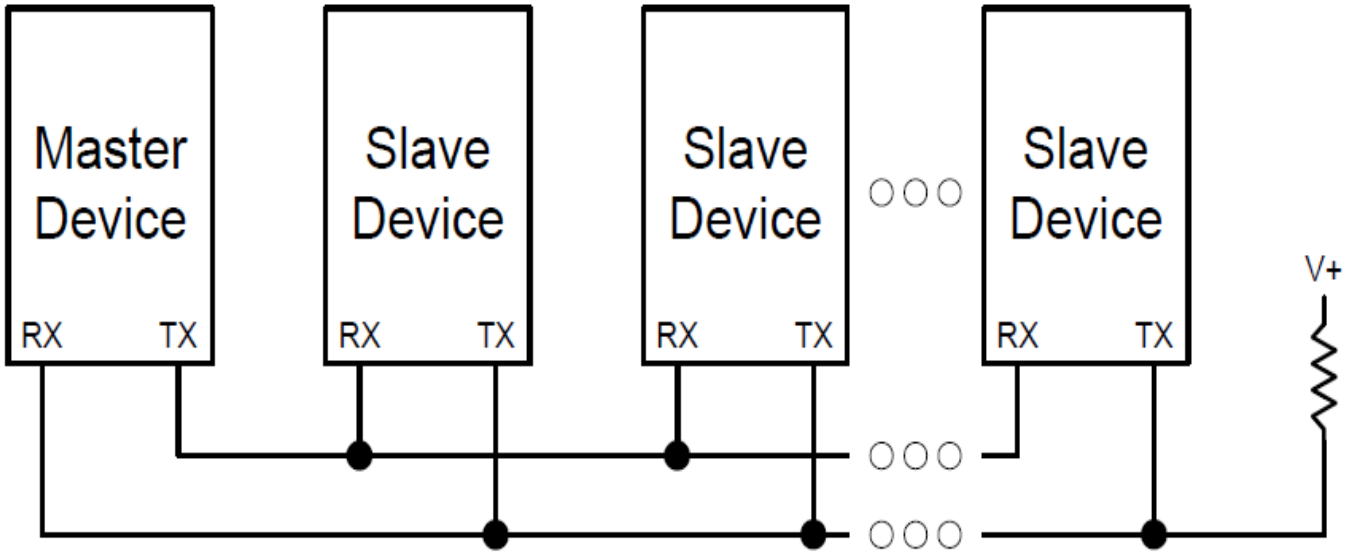
TI0 Transmit Interrupt flag – דגל פסיקת שידור

הביט עולה ל 1 על ידי החומרה של ה UART כאשר הסתיים שידור התו שנמצא ב SBUF . המתכנת בודק את הביט ואם יש בו 1 הוא יודע שהסתיים שידור הבייט ואז הוא שם בביט אפס בעזרת הפקודה CLR TI ואז יכול לשלוח ביית חדש לשידור.

הערה : ניתן להשתמש בבדיקת 2 הביטים כאשר עובדים בשיטת השאילתה – POLLING , אבל גם בשיטת הפסיקה. כאשר מתקבלת פסיקת תקשורת טורית, היא יכולה להתקבל או בסיום שידור ביית או בסיום קליטת ביית. בעזרת 2 הביטים TI0 ו RI0המתכנת יודע האם הפסיקה היא של השידור או של הקליטה.

9.2.4 עבודה בסביבה מרובת מעבדים

האיור הבא מתאר חיבור בסביבה מרובת מעבדים. רואים שמספר יחידות של עבד – SLAVE - מתחברות אל אדון -MASTER אחד.

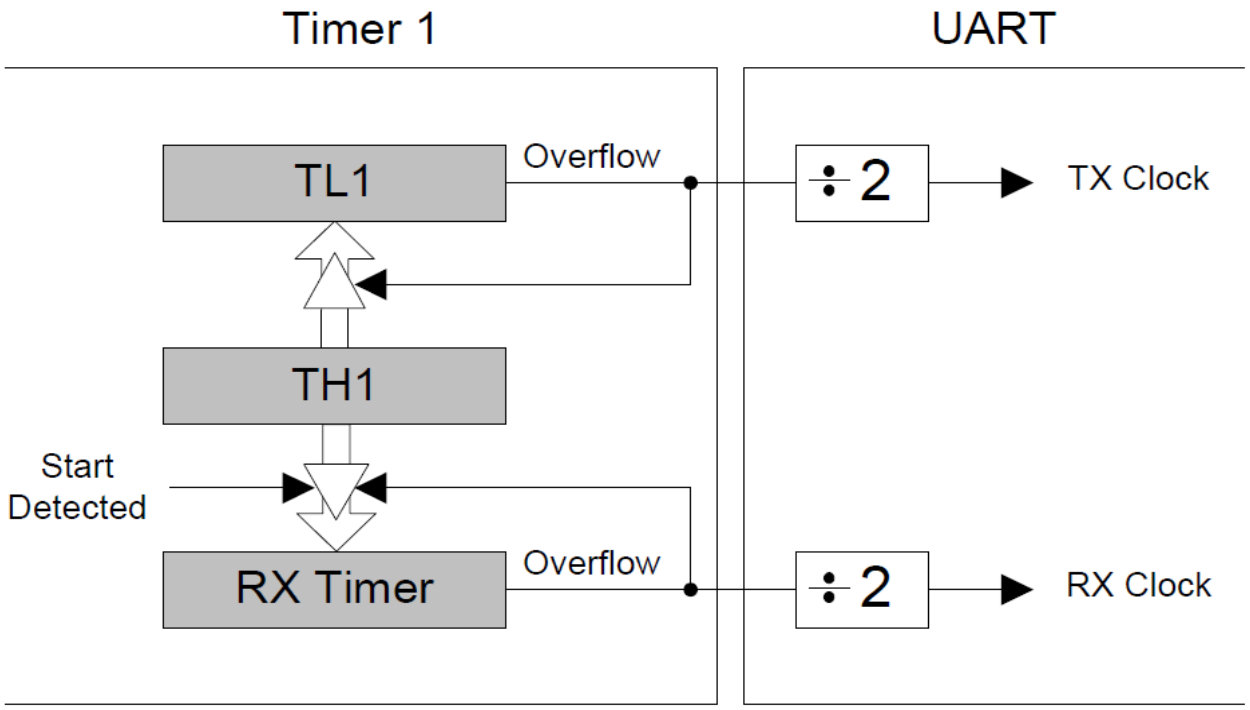


איור 9.9 עבודה בסביבה מרובת מעבדים

לכל רכיב עבד יש כתובת משלו. בעזרת הביט ה 9 (הנקרא TB8 ברגיסטר ה SCON0) המסטר מודיע האם הוא שולח נתון או כתובת. כאשר בביט ה 9 יש '0' זה אומר ש 8 הביטים המשודרים הם נתון. כאשר המסטר שם בביט ה 9 '1' זה אומר ש 8 הביטים הבאים הם של כתובת. לכל עבד יש כתובת משלו . בתחילת התקשורת המסטר שולח כתובת (ביט 9 הוא 1). רק העבד שזוהי הכתובת שלו ייצור קשר עם המסטר.

9.2.5 UART Baud Rate Generator - מחולל קצב הבאוד של ה UART

פולסי השעון הקובעים את קצב התקשורת של ה UART מתקבלים מטיימר 1 שעובד באופן של טעינה חוזרת אוטומטית של 8 ביט. באופן זה טיימר TL1 הוא הסופר ובסיום הספירה הוא נטען לערך שיש ב TH1 . באיור הבא רואים את יצירת קצב הבאוד.



איור 9.10 יצירת קצב הבאוד .

תדר השעון של TX נוצר על ידי TL1 . תדר השעון של RX נוצר על ידי רגיסטר שנקרא באיור RX Timer . זהו רגיסטר שיש בו את התדר שיש ב TL1 . לרגיסטר זה לא ניתן לגשת . גם תדר ה TX וגם ה RX מחולקים ב 2 ויוצרים את קצב התקשורת TX Clock ו RX Clock . הטיימר RX Timer מאופשר ומשתמש באותו ערך של טעינה חוזרת שיש ב TH1 אבל הוא טוען את הערך שיש ב TH1 רק כאשר יש ירידה בהדק ה RX Start Detected – התגלה אפס - כדי להיות מסונכרן עם הירידה הראשונה של קו הקליטה מ 1 ל 0 האומרת שמתחילה קליטה ולא להיות קשור למצב של השידור .

כפי שהזכרנו קודם, טיימר 1 הוא זה שמספק את פולסי ההפעלה ל UART . את טיימר 1 מפעילים באופן 2 שהוא טעינה חוזרת אוטומטית שבה TL1 בלבד הוא המונה ובסיום הספירה הוא נטען לערך שב TH1 . המשתמש יכול לקבוע מאיפה יגיעו פולסי הספירה לטיימר 1 (פנימי או חיצוני) ואת תדר הפולסים. ישנן 6 אפשרויות:

- תדר המתנד הפנימי של שעון המערכת הנקרא SYSCLK.
- תדר SYSCLK אחרי חלוקה ב 4 .
- תדר SYSCLK אחרי חלוקה ב 12 .
- תדר SYSCLK אחרי חלוקה ב 48 .
- תדר שעון ממתנד חיצוני.
- תדר שעון ממתנד חיצוני אחרי חלוקה ב 8 .

יש לטעון את TH1 בערך שייצור גלישה (מעבר של TL1 מ FFH לערך שב TH1) של 2 פעמים קצב התקשורת הרצוי או במילים אחרות קצב התקשורת/באוד צריך להיות חצי מקצב הגלישה . אם נסמן את תדר הפולסים לספירה של TL1 ב $T1_{CLK}$, את קצב הגלישה נסמן כ T1_Overflow_Rate ואת הוא קצב התקשורת/באוד ב UARTBaudRate נקבל את הנוסחאות הבאות:

A) $$UARTBaudRate = \frac{1}{2} \times T1\_Overflow\_Rate$$

קצב הגלישה של TL1 תלוי בתדר הספירה שלו חלקי כמות הפולסים שהוא סופר בין גלישה לגלישה (256-TH1) כאשר TH1 הוא הערך שטוענים ל TH1 נקבל את קצב הגלישות בעזרת משוואה B :

B) $$T1\_Overflow\_Rate = \frac{T1_{CLK}}{256 - TH1}$$

שינוי נושא בנוסחה שבמשוואה B ייתן לנו את הערך שיש לטעון ל TH1 .

$$\mathbf{256 - TH1 = T1_{CLK} \ / \ T1\_Overflow\_Rate}$$

או :

C) $$\mathbf{TH1 = 256 - T1_{CLK} \ / \ T1\_Overflow\_Rate}$$

אם נציב את **T1_Overflow_Rate** שבמשוואה A בנוסחה C נקבל :

D) $$\mathbf{TH1 = 256 - T1_{CLK} \ / \ (2*UARTBaudRate)}$$

דוגמה:

ידוע שעובדים עם המתנד הפנימי ו SYSCLK=48MHz . נבחר יחס חלוקה של 4 . רוצים קצב תקשורת של 57600 ביטים בשנייה . מהו ערך הטעינה שנטען את TH1 ?

פתרון :

אם בחרנו יחס חלוקה של 4 אז התדר המגיע לספירה ונקרא $T1_{CLK}$ שווה: 48MHz / 4= 12MHz

נציב בנוסחה D את הנתונים ונקבל (המספר לא יוצא שלם ולכן נעגל למספר הקרוב ביותר):

$$TH1 = 256 – 12*10^6 / 2*57600 = 256 – 104.1666 = = \sim 152 = 98H$$

הערה : כדאי לשים לב שלא קיבלנו מספר שלם ועשינו עיגול של התוצאה ויש סטייה קלה מהתדר הרצוי . במקרה כאן במקום קצב של 57600 נשתמש בנוסחה D ונקבל קצב של : $12*10^6/ \ 2* (256 -152) = 57,692.3$ bps .

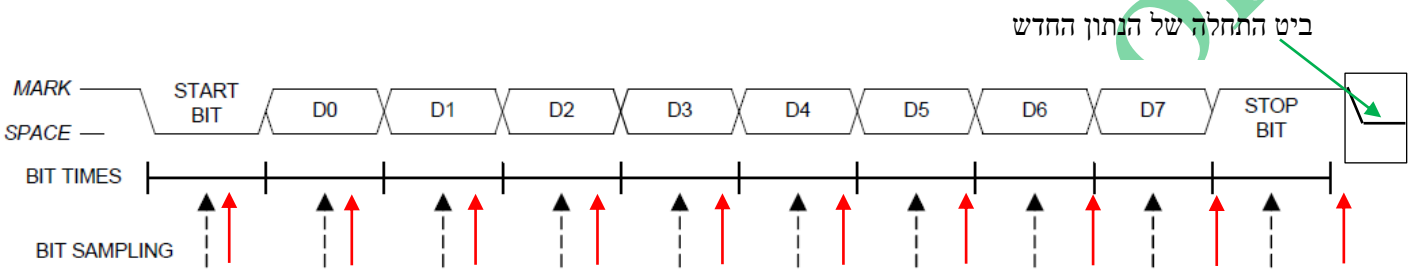
סטיית התדר היחסית כאן היא של : (57692-57600) / 57600 = 0.001597 שהיא 0.1597% .

9.2.6 שגיאות תקשורת בגלל הבדל תדר בין המשדר והמקלט

האם כתוצאה מסטיית התדר הקטנה שבסעיף הקודם בין המשדר למקלט תהיה בעיית תקשורת שגויה ? התשובה היא לא . במידה ותהיה סטיית תדר גדולה מגבולות שנחשב בהמשך - אז כן . עבור סטייה כזו קטנה של כ 0.16% לא תהיה טעות בתקשורת. הסיבה לכך היא שהסטייה איננה מצטברת אלא מתואמת לכל תחילת ביט התחלה חדש. בכל פעם שמגיע ביט סיום יגיע אחריו ביט ההתחלה של הנתון החדש . ביט התחלה מתחיל כאשר יש ירידה בקו מ 1 ל 0 ומרגע זה מתחילה קליטה חדשה.

נסביר זאת בעזרת הדוגמה הבאה ונחשב מהי הסטייה בין קצב השידור וקצב הקליטה המותר כדי שלא נקבל שגיאת תקשורת.

באיור הבא מתואר שידור נתון של 8 ביט. סה"כ משודרים עם ביט התחלה וסיום – 10 ביטים .



איור 9.11 שידור 8 ביטים בתקשורת טורית.

הקליטה מתחילה כאשר יש ירידה של הקו מ 1 ל 0 זהו ה start bit – ביט התחלה. המקלט ממתין זמן של חצי ביט , בודק שוב שיש 0 בקו (בודק שה 0 הקודם לא היה רעש) ומתחיל לדגום את הקו כל זמן של ביט בין דגימה לדגימה. הדגימות הן במרכז הביט כי שם אין תופעות מעבר של רעשים כתוצאה מזמני עלייה או ירידה. נניח שתדר המקלט נמוך מתדר המשדר (זמן בין ביט לביט גדול יותר) . בעזרת החיצים האדומים הראנו שבכל ביט יש תזוזה קלה של הדגימה מהמרכז. כל עוד הבדל התדר לא גורם לשגיאה

בקריאה של ביט הסיום לא תהיה בעיית קליטה כי בביט ההתחלה החדש תתחיל דגימה חדשה . באיור רואים שאם החץ האדום הימני היה זז עוד קצת ימינה אז הדגימה הייתה של 0 ולא של 1 וזו שגיאת קליטה.

כמובן שדבר דומה קיים אם תדר הקליטה קטן מתדר השידור. במקרה כזה החיצים האדומים יופיעו משמאל לחיצים השחורים וגם כאן יש לדאוג שאחרי 10 דגימות לא נדגום את ביט D7 כאילו הוא ביט הסיום.

מכאן ברור שהפרש הזמן – נסמן ב dt בין הביט של השידור שנסמן ב Tx ובין זמן הביט במקלט שנסמן ב Tr כפול 10 ביטים חייב להיות קטן מזמן של חצי ביט שידור.

$(Tx-Tr) * 10 < 0.5*Tx$

$Tx-Tr < 0.05*Tx$

כלומר הפרש זמן (או תדר) קטן מ 5% בין תדר השידור והקליטה יאפשר עדיין קליטה ללא שגיאות.

במידה ויש תקשורת עם 11 ביטים (9 ביטים של נתון + ביט התחלה + ביט סיום) אז ההפרש יהיה :

$(Tx-Tr)*11 < 0.5 *Tx$

$Tx-Tr < 0.045 * Tx$

כלומר הפרש זמן (או תדר) קטן מ 4.5% בין תדר השידור והקליטה יאפשר עדיין קליטה ללא שגיאות.

דוגמה:

תדר השידור הוא 57600 ביטים בשנייה עבור תקשורת של 8 ביטים. מה הם גבולות תדר הקליטה כדי שלא תהיינה שגיאות בגלל התדרים השונים.

פתרון :

ראינו שבשידור של 8 ביטים מותרת שגיאה של 5% . מכאן : 57600 * 5% = ±2,880 כלומר תדר המקלט יכול להיות קטן מ : 57600 + 2880 = **60480** וגדול מ : 57600-2880 = **54720** ועדיין לא נקבל שגיאות קליטה.

בטבלה שבהמשך חוסכים מאתנו את החישובים כדי לדעת איזה ערך טעינה יש להעביר ל TH1 . בנוסף לכך מתארים מה צריך להעביר לרגיסטר CKCON ClocK CONtrol - בקרת השעון כדי שיגיעו פולסים בקצב הרצוי . נזכיר כיצד נראה הרגיסטר:

Bit	7	6	5	4	3	2	1	0
Name	T3MH	T3ML	T2MH	T2ML	T1M	T0M	SCA[1:0]	

T1M=1 – ללא חלוקת תדר. T1M=0 אז חלוקה לפי SCA[1:0] : 00- 12 , 01 - 4 , 10 –48 , 11 – חלוקת תדר היצוני ב 8 .

האיור הבא מראה את הטבלה :

	Target Baud Rate (bps)	Actual Baud Rate (bps)	Baud Rate Error	Oscillator Divide Factor	Timer Clock Source	SCA1-SCA0 (pre-scale select*	T1M	Timer 1 Reload Value (hex)
SYSCLK = 12 MHz	230400	230769	0.16%	52	SYSCLK	XX	1	0xE6
	115200	115385	0.16%	104	SYSCLK	XX	1	0xCC
	57600	57692	0.16%	208	SYSCLK	XX	1	0x98
	28800	28846	0.16%	416	SYSCLK	XX	1	0x30
	14400	14423	0.16%	832	SYSCLK / 4	01	0	0x98
	9600	9615	0.16%	1248	SYSCLK / 4	01	0	0x64
	2400	2404	0.16%	4992	SYSCLK / 12	00	0	0x30
	1200	1202	0.16%	9984	SYSCLK / 48	10	0	0x98
SYSCLK = 24 MHz	230400	230769	0.16%	104	SYSCLK	XX	1	0xCC
	115200	115385	0.16%	208	SYSCLK	XX	1	0x98
	57600	57692	0.16%	416	SYSCLK	XX	1	0x30
	28800	28846	0.16%	832	SYSCLK / 4	01	0	0x98
	14400	14423	0.16%	1664	SYSCLK / 4	01	0	0x30
	9600	9615	0.16%	2496	SYSCLK / 12	00	0	0x98
	2400	2404	0.16%	9984	SYSCLK / 48	10	0	0x98
	1200	1202	0.16%	19968	SYSCLK / 48	10	0	0x30
SYSCLK = 48 MHz	230400	230769	0.16%	208	SYSCLK	XX	1	0x98
	115200	115385	0.16%	416	SYSCLK	XX	1	0x30
	57600	57692	0.16%	832	SYSCLK / 4	01	0	0x98
	28800	28846	0.16%	1664	SYSCLK / 4	01	0	0x30
	14400	14388	0.08%	3336	SYSCLK / 12	00	0	0x75
	9600	9615	0.16%	4992	SYSCLK / 12	00	0	0x30
	2400	2404	0.16%	19968	SYSCLK / 48	10	0	0x30

Note: SCA1-SCA0 and T1M define the Timer Clock Source. X = Don't care

איור 9.12 : קביעת ערך הטעינה ל TH1 עבור קצבי שידור סטנדרטיים ועבודה עם המתנד הפנימי SYSCLK .

נסביר את העמודות בטבלה משמאל לימין.

העמודה השמאלית ביותר אומרת מהו תדר המתנד הפנימי SYSCLK .

Target Baud Rate - מתארת את קצב תקשורת המטרה - הרצוי .

Actual Baud Rate – הקצב המעשי שמקבלים (בדוגמה בסעיף קודם ראינו שיש הבדל בין תדר המטרה הרצוי והתדר המעשי שהמערכת מוציאה).

Baud Rate Error – שגיאת קצב הבאוד. – הסטייה באחוזים בין התדר הרצוי – המטרה והתדר המעשי .

Oscillator Devide Factor - מקדם חלוקת תדר המתנד - בכמה לחלק את תדר ה SYSCLK .

Timer Source Clock – מקור השעון של הטיימר – מאיפה מגיעים פולסי הספירה לטיימר .

***SCA1 – SCA0 pre scale Select** - מה צריך לשים בביטים SCA1- SCA0 שברגיסטר בקרת השעון CKCON – ClocK CONtrol שהוסבר בפרק על הטיימרים (7.2.3 - פרק 7 סעיף 2.3) . 2 הביטים קובעים את יחס חלוקת השעון .

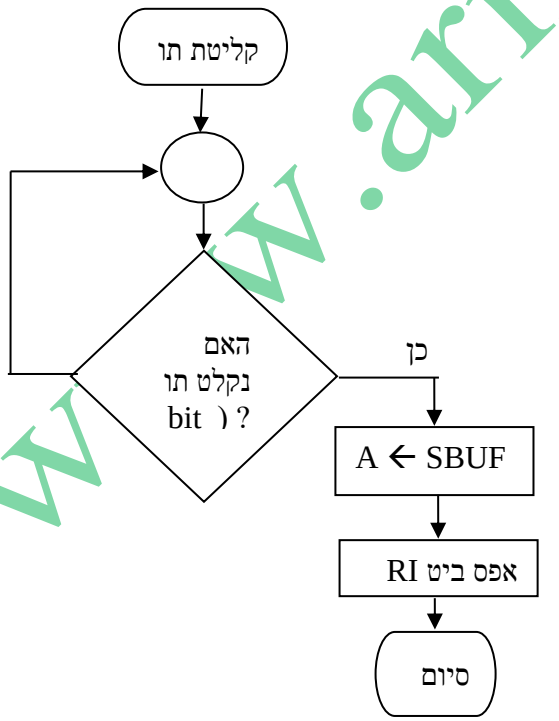
T1M - מה צריך לשים בביט T1M שברגיסטר CKCON - הביט קובע את מקור השעון. .אם המתכנת שם בביט 0 אז הפולסים מגיעים לאחר חלוקה של שעון המערכת לפי הביטים 0 ו 1 (SCA1- SCA0). אם שמים בביט 1 אז הפולסים הם שעון המערכת ללא חלוקה.

Timer 1 Reload Value (hex) - ערך טעינה לטיימר 1 (ליתר דיוק ל TH1) בהקסה דצימלי.

9.3 תוכנה :

9.3.1 קליטה של תו

כדי לבדוק האם נקלט תו נעזר באיור תרשים המלבנים הבא המתאר פרוצדורה / פונקציה לקליטת תו טורי:



איור 9.13 : תרשים זרימה של קליטת ביית טורי

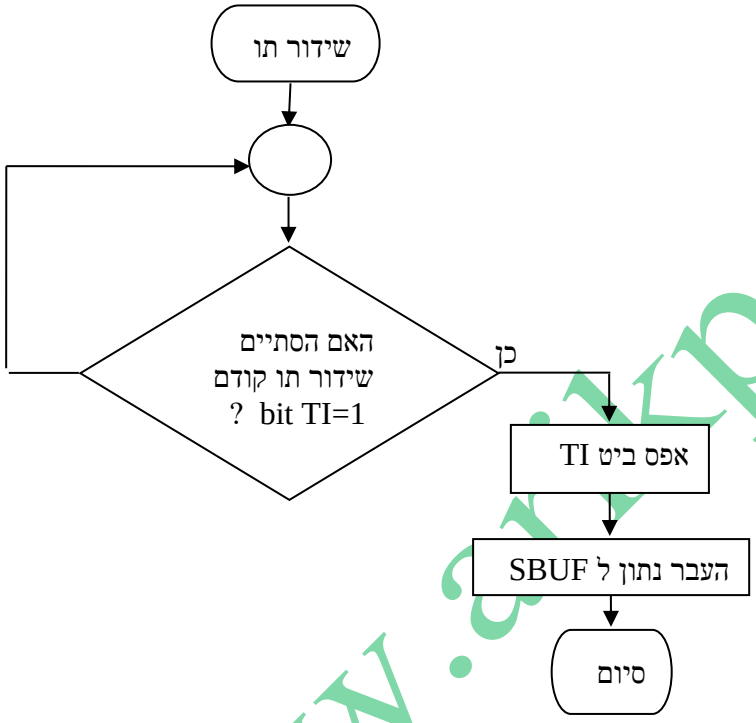
בודקים את הביט RI המציין שנקלט נתון טורי. אם בביט יש 0 זה אומר שלא נקלט תו. אם בביט יש 1 זה אומר שנקלט תו חדש. במקרה כזה "מושכים" את התו מה SBUF של הקליטה ומאפסים את ביט RI כדי שיעלה ל 1 שיגיע תו חדש נוסף. נרשום תכנית באסמבלי מצד שמאל ובשפת C51 מימין (נניח שהוכללו הספריות המתאימות והוגדר משתנה char myReceivedData) :

```
receiveTav:
        jnb ri,$
        Mov a, sbuf
        Clr ri
        Ret
```

```
void  recieveTav( ) {
  while (RI==0);
 myReceivedData=SBUF;
 RI=0
}
```

9.3.2 שידור של תו

נבדוק האם הסתיים שידור התו הטורי :



איור 9.14 תרשים זרימה של שידור ביית טורי .

הסבר : בודקים האם הסתיים שידור התו הקודם בעזרת ביט TI. אם לא (TI = 0) חוזרים שוב לבדוק את הביט. ברגע שה UART סיים לשדר את התו הוא מעלה את הביט ל 1 ואז מנקים את הביט ושולחים ל SBUF ביית חדש לשידור . נרשום תכנית באסמבלי מצד שמאל ובשפת C51 מימין (נניח שהוכללו הספריות המתאימות והוגדר משתנה char myTransmittedData) :

פרוצדורת השידור תיראה :

```
sendTav:
        jnb ti,$
        Clr ti
        Mov sbuf,a
        Ret
```

```
void sendTav ( )        {
  while(TI=0);
   TI=0;
    SBUF=myTranmittedData;
}
```

9.4. תרגילי דוגמה

8.4.1 תרגיל 1 :

רשום פקודות לעבודה בתקשורת טורית בין 2 מערכות מבוקרות מיקרו, בקצב תקשורת של 9600bps , ללא ביט תשיעי. תדר SYSCLK הוא 24MHz . עבוד עם שאילתה (POLLING). הדק Tx0 מתחבר ל P0.4 והדק Rx0 ל P0.5 .

פתרון : נעבוד לפי השורה השישית בטבלה שבפרקים הקודמים עבור תדר שעון SYSCLK =24MHz

9600	9615	0.16%	2496	SYSCLK / 12	00	0	0x98

Bit	7	6	5	4	3	2	1	0
Name	S0MODE	-	MCE0	REN0	TB80	RB80	TI0	RI0

Bit	7	6	5	4	3	2	1	0
Name	T3MH	T3ML	T2MH	T2ML	T1M	T0M	SCA[1:0]	

Bit	7	6	5	4	3	2	1	0
Name	GATE1	C/T1	T1M[1:0]		GATE0	C/T0	T0M[1:0]	

Bit	7	6	5	4	3	2	1	0
Name	CP1AE	CP1E	CP0AE	CP0E	SYSCKE	SMB0E	SPI0E	URT0E

UART I/O Output Enable.
0: UART I/O unavailable at Port pin.
1: UART TX0, RX0 routed to Port pins P0.4 and P0.5.

```
Mov scon0,#00010010B;  8 ביט קצב תקשורת משתנה, אפשור קליטה
; שמים 1 ב TI0 עבור תו השידור הראשון. אומרים שתו קודם כבר שודר
Mov ckcon,# 00000000B ;
                T1M=0 חלוקת התדר לפי SCA[1:0] : 00 - 12 , 01 - 4 , 10 – 48 , 11 – חלוקת תדר חיצוני ב 8 .
Mov tmod,#20h;          טיימר 1 באופן 2
Mov th1,#98h ;  9600 לקצב שידור
Mov tl1,#98h
Setb tr1                ;  הרצת הטיימר
Mov xrb0,#00000001B  ; P0.4  P0.5  חיבור הדקי Tx0 Rx0 להדקי
mov xbr1,#40h; enable crossbar אפשור הקרוס בר
```

אותה התוכנית בשפת C51 (בהנחה שהוכללו הספריות המתאימות) :

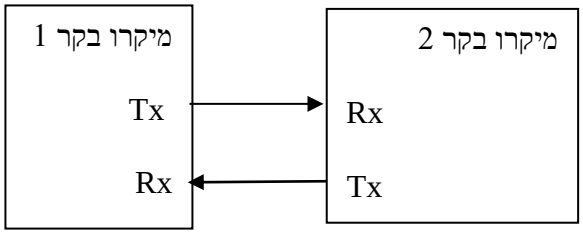
```
SCON0 = 0x12;
CKCON = 0;
TMOD = 0x20;
TH1 = TL1 = 0x98;
TR1=1;
XBR0=1;
XBR1=0x40;
```

9.4.2 תרגיל 2 :

חבר בין 2 מערכות מיקרו בקרים בעזרת תקשורת טורית. התקשורת היא FULL DUPLEX תקשורת של 8 ביט בקצב 115200 סל"ש . SYSCLK = 48MHz בשתי המערכות. א. צייר את החומרה הרלוונטית לתרגיל. ב. יש לרשום פרוצדורה / פונקציה עבור אחד מהמיקרו בקרים שתאתחל את תקשורת ה UART . ג. יש לרשום פרוצדורה / פונקציה שתשדר בלוק/מערך נתונים מכתובת 50H ועד 5FH ד. יש לרשום פרוצדורה / פונקציה שתקלוט את בלוק /מערך התווים ותשמור עלי החל מכתובת 60H ועד 6FH .

פתרון :

א. חיבור בין 2 המיקרו בקרים.



איור 9.15 : חיבור בין 2 המיקרו בקרים בתקשורת טורית.

ב. אתחול התקשורת הטורית . תחילה באסמבלי ואחר כך בשפת C51 .

ניעזר בשורה השנייה שבטבלה עבור SYCLK=48MHz הנראית כך:

115200	115385	0.16%	416	SYSCLK	XX	1	0x30
Target Baud Rate (bps)	**Actual Baud Rate (bps)**	**Baud Rate Error**	**Oscillator Divide Factor**	**Timer Clock Source**	**SCA1-SCA0 (pre-scale select***	**T1M**	**Timer 1 Reload Value (hex)**

באסמבלי :

```
initUart:
        mov xrb0,#00000001B  ;   P0.4  P0.5  להדקי  Tx0 Rx0  חיבור הדקי
        mov xbr1,#40h;     enable crossbar אפשור הקרוס בר
        mov scon0,#0001000B;  אפשור קליטה ,8 ביט קצב תקשורת משתנה
        mov ckcon,# 00001000B ;  T1M = 1 ,  SCA1-SCA0  אין חלוקת תדר ולא משנה מה יש בביטים
        mov tmod,#20h;          2 טיימר 1 באופן
        mov th1,#30h  ; 115200  לקצב שידור
        mov tl1,#30h
        setb tr1 ;  1 הרצת טיימר
        ret
```

בשפת C51 (בהנחה שהוגדרו הספריות המתאימות).

```
void initUart( )
{
  XBR0 = 1;
  XBR1 = 0x40;
  SCON0 = 0X10;
  CKCON = 0X8;
```

```
  TMOD = 0x20;
  TH1 = TL1 = 0x30;
  TR1 = 1;
}
```

ג. הפרוצדורה באסמבלי

```
sendBlock:
                mov r0,#50h ;   כתובת תחילת הבלוק
again:          mov a,@r0       ; העברת הנתון מהזיכרון לאקומולטור
                lcall transmit  ; קרא לפרוצדורת השידור
                Inc r0          ;  הגדלת המצביע שיראה על הכתובת הבאה
                Cjne r0,#60h,again     ;  ? האם הגענו לסוף בלוק הנתונים
                ret             ;  לולאה אין סופית
Transmit:       jnb ti,$        ; ? האם הסתיים שידור תו קודם
                clr ti          ;  ניקוי הביט שמראה שהסתיים שידור קודם
                mov sbuf,a      ; העברת הנתון לשידור
                ret             ;  חזור
```

פונקציית השידור בשפת C51 .

```
char data arrayT [0x10] _at_ 0x50 ;     // הגדרת מערך בתים המתחיל בכתובת 50H ב RAM הפנימי
unsigned char  x;  //  משתנה גלובאלי כדי שיוכר בפונקציות שמתחתיו
void transmit( ); // הצהרה על פונקציה שמופיעה מתחת השורה הקוראת לפונקציה כדי שתוכר על ידי הקומפיילר.
void sendArray( )       //   פונקציית שידור המערך
{
  for (x=0;x<0x10;x++)          // לולאה המתבצעת 10H פעמים
          transmit( );      //  זימון/קריאה לפונקציית שידור תו אחד.
}
void transmit( )     //  פונקציית שידור של תו אחד.
{
  while (TI==0);   //    ממתינים לסיום השידור הקודם. בסיום שידור הנתון הביט עולה ל 1
  TI=0;  // איפוס ביט TI כדי להכין אותו לשידור לשידור נתון חדש
  SBUF=arrayT[x];     // שולחים ל SBUF נתון מהמערך והוא מיד משודר על ידי ה UART
}
```

ד. פונקציית הקליטה באסמבלי :

```
ReceiveBlock:
            mov r1,#60h ;   כתובת תחילת הבלוק
again:      lcall receive   ; קרא לפרוצדורת השידור
            mov @r1,a       ; העברת הנתון מהאקומולטור לזיכרון
            inc r1          ;  הגדלת המצביע שיראה על הכתובת הבאה
            cjne r1,#70h,again     ;  ? האם הגענו לסוף בלוק הנתונים
            ret             ;  לולאה אין סופית
Receive :   jnb ri,$            ; ?  האם הסתיימה קליטת תו
            mov a,sbuf          ; העברת הנתון הנקלט לאקומולטור
            clr ri              ;  ניקוי הביט שמראה שהסתיים קליטת תו
            ret                 ;  חזור
```

ובשפת C51 :

```
char data arrayR [0x10] _at_ 0x60 ;     // הגדרת מערך בתים המתחיל בכתובת 60H ב RAM הפנימי
unsigned char  x;  //  משתנה גלובאלי כדי שיוכר בפונקציות שמתחתיו
void receive( ); // .הצהרה על פונקציית קליטת תו שמופיעה מתחת השורה הקוראת לפונקציה (כדי שתוכר על ידי הקומפיילר)
void receiveArray( )   //   פונקציית קליטת המערך
{
  for (x=0; x < 0x10 ; x++)     // לולאה המתבצעת 10H פעמים
          receive( );       //  .זימון/קריאה לפונקציית קליטת תו אחד
}
void receive( )    //  .פונקציית קליטה של תו אחד
{
  while (RI==0);   //    ממתינים לסיום השידור הקודם. בסיום שידור הנתון הביט עולה ל 1
  RI=0;  // איפוס ביט TI כדי להכין אותו לשידור לשידור נתון חדש
  arrayR[x] = SBUF;   //  .מעבירים את הנתון שנקלט למערך במיקום המתאים
}
```

UART1 9.5

UART1 לא מופיע בתוכנית הלימודים אבל נכתוב עליו מספר משפטים.

- UART1 בדומה ל UART0 הוא פורט טורי , אסינכרוני העובד ב FULL DUPLEX ומציע מגוון אפשרויות של פורמט נתונים. יש לו מחולל קצב תקשורת משלו עם טיימר של 16 ביט ואפשרויות חלוקת תדר שונות כך שניתן לקבל קצבי תקשורת רבים. יש לו FIFO (חוצץ זיכרון העובד לפי העיקרון של הראשון שנכנס יוצא ראשון – First In First Out) היכול לקבל 3 בתים של נתונים לפני שהנתונים אובדים וקורית גלישה. במילים אחרות : הוא יכול לקלוט 3 בתים מבלי שהמיקרו יבצע קריאה שלהם . בביית הנתון הרביעי שייקלט מבלי שהמיקרו "משך" את הנתונים הקודמים נתחיל לאבד את הנתונים הקודמים שנקלטו.
- ל UART1 יש 6 רגיסטרים הנמצאים ב SFR :
  3 משמשים ליצירת קצב התקשורת .
  2 משמשים לפורמי נתונים , בקרה וסטאטוס.
  1 משמש לקליטה או שידור נתון (למעשה זה 2 רגיסטרים אחד לשידור ואחד לקליטה והקריאה או הכתיבה מזהה
  למי מהם פונים (כמו ב UART0) .
- כתיבה ל SBUF1 ניגשת לרגיסטר אחיזת שידור - TRANSMIT HOLDING .
- קריאה מ SBUF1 ניגשת תמיד אל הביית הראשון ב FIFO של הקליטה.
- גם עם UART1 ניתן לעבוד עם פסיקות והן קורות בכל סיום שידור של ביית או בסיום קליטה של ביית. גם כאן יש 2 ביטים TI1 ו RI1 שנמצאים ב SCON1 ומראים שהסתיים שידור או הסתיימה קליטה. כתובת הפסיקה היא 0X83 בזיכרון התוכנית והיא פסיקה מספר 16 (כולל reset).
- גם כאן הביטים TI ו RI אינם מאופסים בחומרה כאשר נענים לפסיקה ויש לאפס אותם בתוכנה.