T170B417 Įterptinės sistemos

Inžinerinis projektas: užduotis ir instrukcijos

Turinys

1 Užduotis

Suprojektuoti ir realizuoti įterptinę sistemą, sudarytą iš:

- STM32xxx serijos mikrovaldiklio
- Nurodytų jutiklių
- Pasirinkto indikatoriaus

- Duomenų perdavimo sąsajos į personalinį kompiuterį (PK)

Suprojektuota įterptinė sistema matuojamus duomenis turi perduoti į PK atvaizdavimui ir išsaugojimui.

2 Užduoties variantai

Lentelė 1. Užduoties variantų lentelė

Var. Nr.	Matuojamas fizikinis dydis arba jutiklis	Kanalų skaičius	Vaizduojami ir perduodami į kompiuterį parametrai	Signalo pralaidumo juosta	Dydžio diapazonas, signalo amplitudė	Parodymų atnaujinimo indikatoriuje periodas, s	Perduodamų į kompiuterį parametrų diskretizavimo periodas, s
1.	Garsas (Elektretinis/MEMS mikrofonas)	1	Vidutinė kvadratinė vertė	(0.1-5.0) kHz	parinkti	2	0,5
2.	Vibracijos (Akselerometras)	1	Vidutinė kvadratinė vertė	(0-100) Hz	parinkti	2	0,2
3.	Temperatūra	2	Imties vertė, kanalų skirtumai	(0-1) Hz	-20 .. +40 C	5	2
4.	Barometrinis slėgis	1	Vidurkis	(0-10) Hz	(90-110)kPa	5	1
5.	Apšviestumas	2	Vidurkis, kanalų skirtumai	(0-100) Hz	parinkti	2	0,2
6.	Atstumas (ultragarsinis arba infraraudonųjų sp. jutiklis)	2	Vidurkis, kanalų skirtumai	(0-2) Hz	(1 – 30) cm	2	0,2
7.	Polinkio kampas (Giroskopas)	1	Vidurkis	(0-10) Hz	(0-180) laipsn.	1	0,2
8.	Oro drėgmė	2	Vidurkis, kanalų skirtumai	(0-0,1) Hz	(10-70) %	10	1
9.	Nuolatinė įtampa	3	Vidurkis	(0-0,1) Hz	(0.1 – 25) V	2	1
10.	Nuolatinė srovė	3	Vidurkis	(0-0,1) Hz	(0.1 – 100) mA	2	1
11.	Dažnis	1	Vertė	(0.01-10) kHz	(0.1-5) V	1	
12.	Kintama įtampa	1	Vidutinė kvadratinė vertė, amplitudė	(0.05-3) kHz	1 mV- 10 V	1	0,2
13.	Kintama srovė	1	Vidutinė kvadratinė vertė, amplitudė	(0.05-3) kHz	(0.1- 100) mA	1	0,2
14.	Varža	2	Vidurkis	(0-10) Hz	1 Ω – 100 kΩ	1	1
15.	Talpumas	1	Vidurkis	(0-10) Hz	1 nF – 10 µF	1	1
16.	Rato apsukų greičio matavimas (herkonas)	1	Vidurkis	-	(0-30km/val.)	1	0,2
17.	Garsas (Elektretinis/MEMS mikrofonas)	1	Vidutinė kvadratinė vertė	(0.1-5.0) kHz	parinkti	1	0,5
18.	Vibracijos (Akselerometras)	1	Vidutinė kvadratinė vertė	(0-400) Hz	parinkti	3	0,1
19.	Temperatūra	2	Imties vertė, kanalų diapazonai	(0-0,5) Hz	-0 .. +75 C	2	4

			(maksimumas, minimumas)				
20.	Barometrinis slėgis	1	Diapazonas (maksimumas, minimumas)	(0-10) Hz	(90-110) kPa	3	2
21.	Apšviestumas	2	Vidurkis, diapazonas (maksimumas, minimumas)	(0-10) Hz	parinkti	2	0,2
22.	Atstumas (ultragarsinis arba infraraudonųjų sp. jutiklis)	2	Vidurkis, kanalų skirtumai	(0-2) Hz	(1 – 30) cm	2	0,2
23.	Polinkio kampas (Giroskopas)	1	Vidurkis	(0-50) Hz	(0-180) laipsn.	2	0,2
24.	Oro drėgmė	2	Vidurkis, kanalų skirtumai	(0-0,01) Hz	(30-70) %	10	1
25.	Nuolatinė įtampa	2	Vidurkis	(0-0,1) Hz	(0.1 – 25) V	2	1
26.	Nuolatinė srovė	3	Vidurkis	(0-0,1) Hz	(0.1 – 100) mA	2	1
27.	Dažnis	1	Vertė	(0.1-100) kHz	(0.1-0.5) V	1	
28.	Kintama įtampa	1	Vidutinė kvadratinė vertė, amplitudė	(0.05-1) kHz	1 mV- 10 V	0,5	0,2
29.	Kintama srovė	1	Vidutinė kvadratinė vertė, amplitudė	(0.05-5) kHz	(0.1- 500) mA	0,5	0,2
30.	Varža	2	Vidurkis	(0-1) Hz	0.1 Ω – 10 kΩ	0,5	1
31.	Talpumas	2	Vidurkis, kanalų skirtumas	(0-1) Hz	10 nF – 100 µF	0,5	1
32.	Rato apsukų greičio matavimas (herkonas)	1	Vertė	-	(0-60 aps./min.)	0,5	0,2

Atliekant užduotį 2 studentams reikia realizuoti abiejų studentų užduoties variantus.

3 Vertinimo skalė (rubrika)

Realizuotos funkcijos	**Skiriamas maksimalus balas**
Sudaryta principinė schema ir pagrindinė programa	1
Jutiklių duomenys nuskaitomi	1
Požymio skaičiavimai realizuoti ir testuoti	1
Matavimo tikrinimas kraštiniuose diapazono taškuose	1
Duomenų perdavimas į kompiuterį užduotu periodu	2
Atvaizdavimas indikatoriuje užduotu periodu	2
Tarpinės ataskaitos pateikimas laiku	1
Galutinė ataskaita pateikiama laiku	1
	10 (maksimalus balas)

4 Indikatoriai

4.1 Mokomoji medžiaga

LCD modulis arba OLED modulis

Mokomieji video:

1. OLED display with STM32 || I2C || CubeMx || Keil u5
2. SSD1306 OLED and STM32 || 128x64 || SW4STM || CubeMX
3. https://controllerstech.com/stm32/
4. https://controllerstech.com/oled-display-using-i2c-stm32/

4.2 LCD indikatoriaus su HD44780 valdikliu prijungimas prie STM32 mikroprocesorių

HD44780 (išorinio įrenginio valdiklis/protokolas) arba su juo suderinami protokolai Sintronix ST7066U, Samsung KS0065B

STM32F, STM32F4 architecture , microprocessor family

I2C arba „parallel 4 bit“ (fizinės sąsaja)

HAL - periferinių įrenginių tvarkyklės (driver libraries of microprocessor)

Paieškos frazė (google search): „**HD44780 STM32F4 I2C HAL** compatible library“:

1. STM32 LCD 16×2 Tutorial & Library | Alphanumeric LCD 16×2 Interfacing https://deepbluembedded.com/stm32-lcd-16x2-tutorial-library-alphanumeric-lcd-16x2-interfacing/ (ECUAL.rar)
2. https://github.com/4ilo/HD44780-Stm32HAL (HD44780-Stm32HAL-master.zip)
3. https://adastra-soft.com/hd44780-library-for-stm32/
4. Interface LCD 16×2 via I2C with STM32 https://controllerstech.com/i2c-lcd-in-stm32/ (I2C, STM32CUBEIDE)

Kuo gali skirtis internete rastos atviros prieigos bibliotekos:

1. Patikimumo lygiu (veikti arba neveikti arba veikti nestabiliai)
2. Projekto paruošimo kažkuriai projektavimo aplinkai (Keil, STM32CUBEIDE, ...)
3. Dokumentacijos lygiu
4. Sprendimo efektyvumu (Greitaveika (panaudojanti HAL / LL bibliotekas arba tiesiogine kreiptimi į registrus), naudojant vėlinimus/pertrauktis arba tikrinant būsenos bitus)

4.3 OLED indikatoriaus su SSD1306 valdikliu prijungimas prie STM32 mikroprocesorių

1. OLED display using I2C with STM32 https://controllerstech.com/oled-display-using-i2c-stm32/
2. STM32 open source code-0.96 inch OLED display SPI https://www.programmersought.com/article/18983663074/

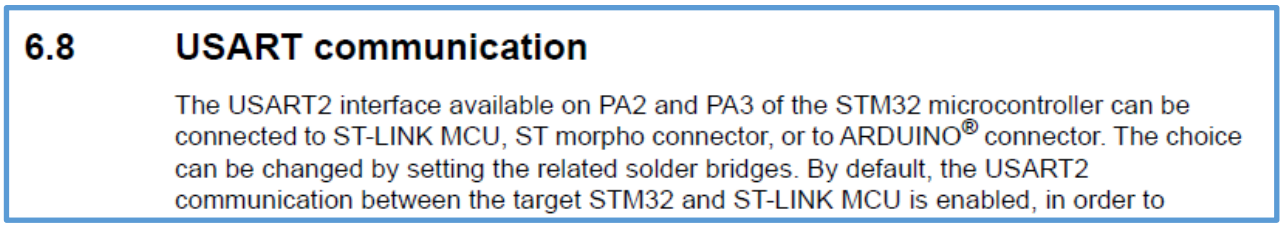
5 Duomenų perdavimas į kompiuterį

5.1 USART sąsajos panaudojimas

Darbe panaudokite mikrovaldiklio UART sąsają duomenų perdavimui į kompiuterį. Priklausomai nuo naudojamo maketo mikrovaldiklio sąsają galima prijungti vienu iš būdų:

1. UART sąsajos signalai prijungti prie USB hub mikroschemos, kuri panaudota integruoto adapterio (on-board ST-Link) realizavimui. Šiuo būdu realizuojamas virtualus COM portas (Virtual COM Port).

1.1. NUCLEO-L073RZ maketuose šis variantas yra paruoštas pagal nutylėjimą (UM1724 User manual STM32 Nucleo-64 boards (MB1136) 6.8 USART communication).



6.8 USART communication

The USART2 interface available on PA2 and PA3 of the STM32 microcontroller can be connected to ST-LINK MCU, ST morpho connector, or to ARDUINO® connector. The choice can be changed by setting the related solder bridges. By default, the USART2 communication between the target STM32 and ST-LINK MCU is enabled, in order to

1.2. Toks variantas iš karto nėra paruoštas **STM32L053-DISCO** makete, bet yra galimas, užtrumpinus dvi kontaktines aikšteles, kaip aprašyta maketo vartotojo vadove (4.11 paragrafas *UM1775 User manual. Discovery kit for STM32L0 series with STM32L053C8 MCU*).

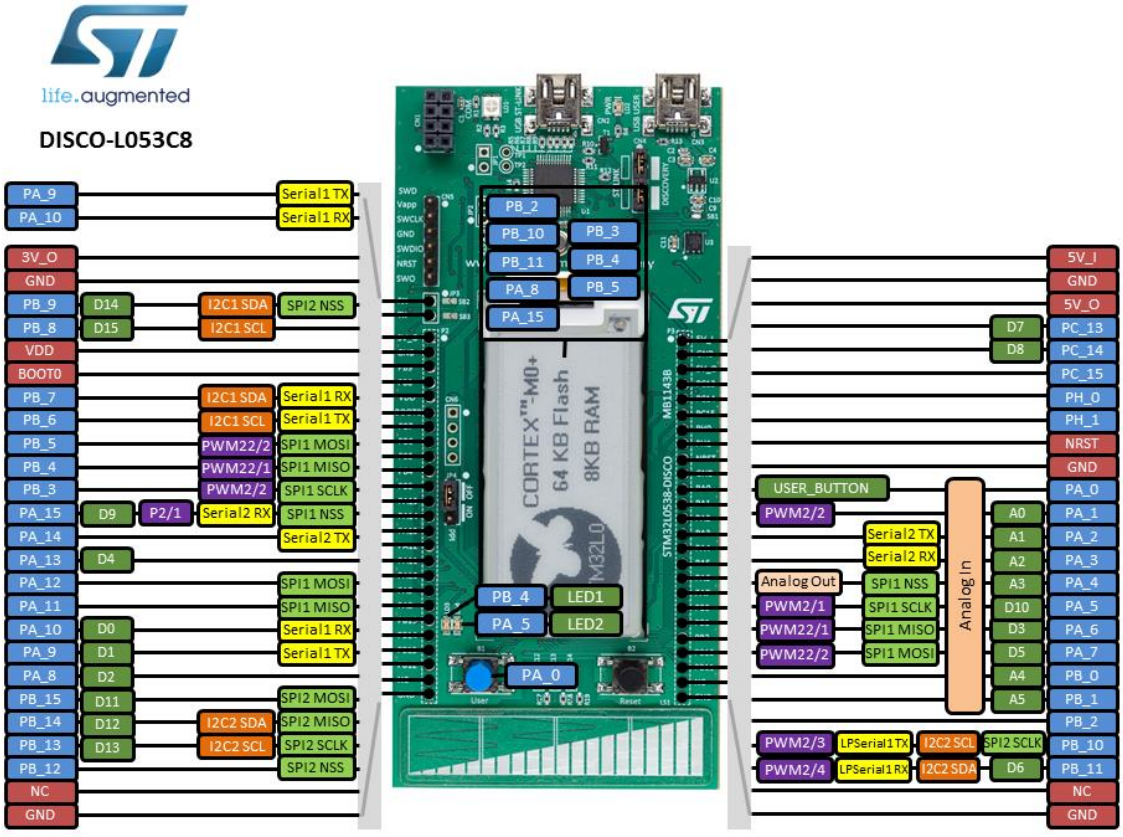
2. UART sąsajos signalus (USARTx_RX, USARTx_TX ir GND) per išorinį UART-USB keitiklį prijungti prie USB sąsajos. Toks variantas yra galimas **STM32L053-DISCO** arba **STM32F4DISCOVERY** maketuose.

UART-USB konverterių (keitiklių) pavyzdžiai (šie keitikliai dažniausiai realizuojami FTDI mikroschemų https://ftdichip.com/ pagrindu):

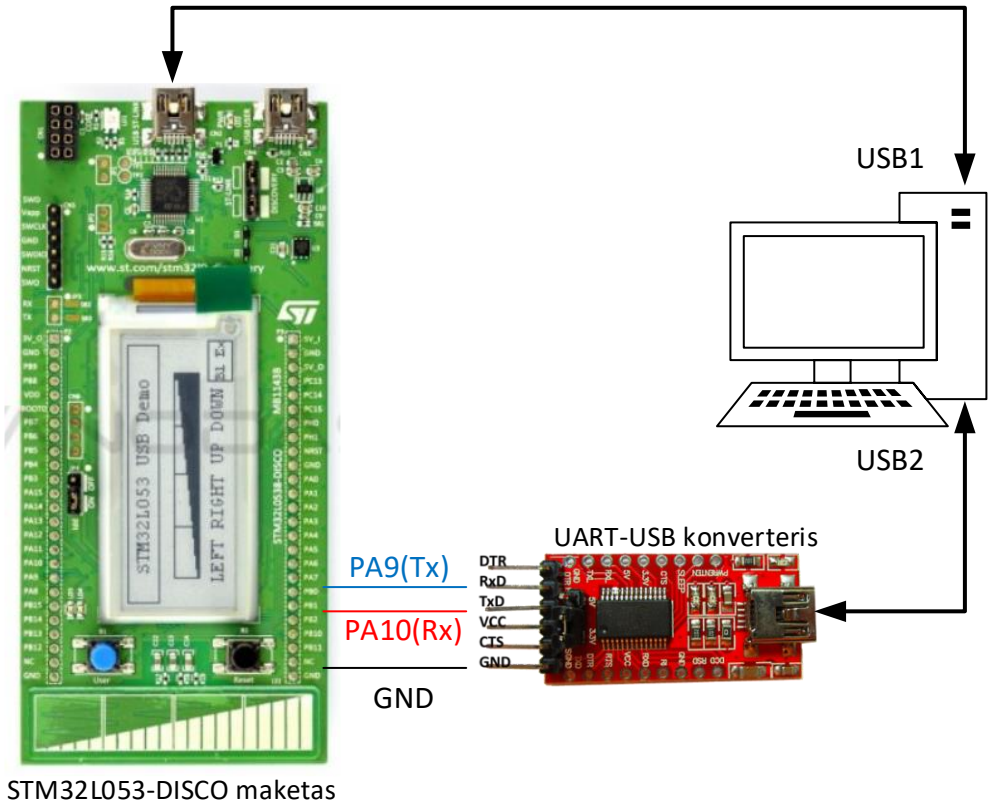
1. https://www.anodas.lt/ft232-ftdi-usb-ttl-5v-3-3v-konverteris
2. https://rcl.lt/products/akad012-konverteris-usb-rs232-ttl-pl2303hx-3-3v-ir-5v

1 pav. UART –USB arba RS232-USB konverterio pavyzdys

5.2 UART ryšys panaudojant STM32L053-DISCO



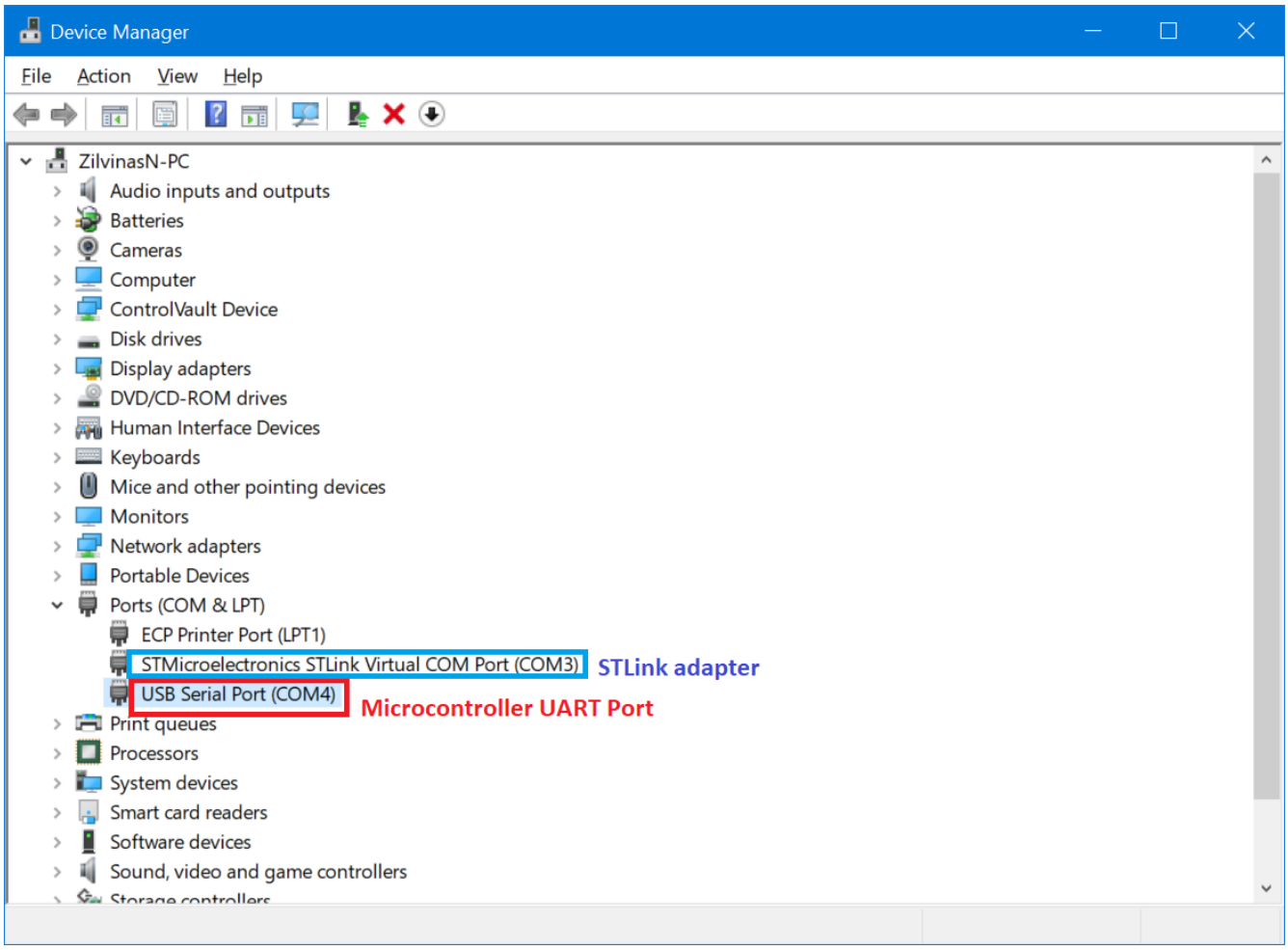
2 pav. STM32L053-DISCO maketo išoriniai prievadai



3 pav. STM32L053-DISCO maketo prijungimas prie kompiuterio USB per RS232-USB konverterį

5.3 COM prievado aptikimas Windows operacijų sistemoje

Prijungus maketą ir UART-USB konverterį **Windows Device Manager** programoje jie abu turi būti matomi **Port (COM & LPT)** skyrelyje. Čia mums svarbu įsidėmėti, kad vartotojo prievadui priskirtas numeris COM4, nes jį reikės nurodyti terminalinėje programoje.

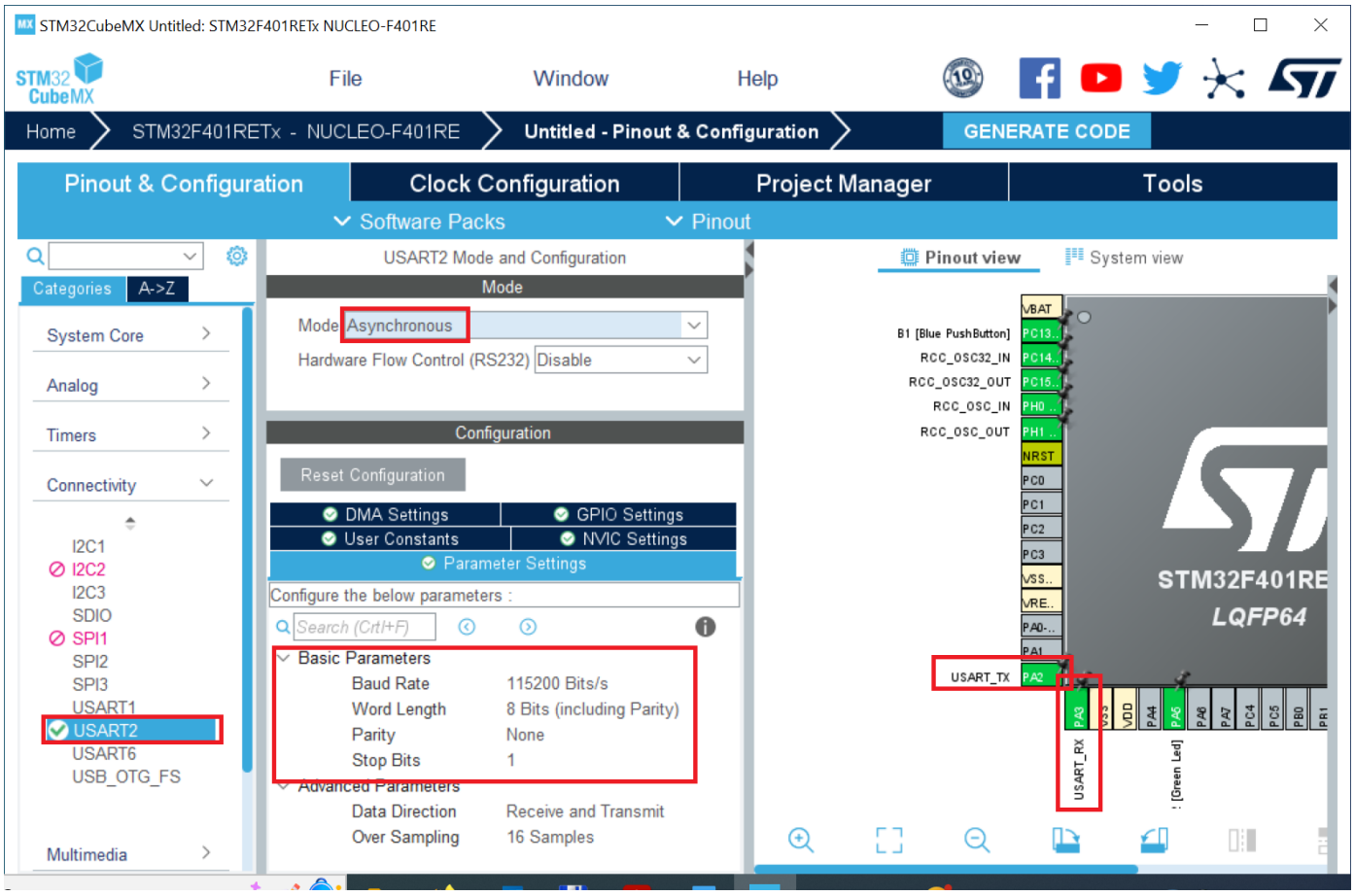


5.4 UART ryšys su kompiuteriu Nucleo tipo maketuose

5.4.1 Nucleo-F401RE (Nucleo-64 plokštės tipas)

Pagal nutylėjimą šio tipo plokštėse STM32 mikrovaldiklio USART2 sąsaja (prievadai PA2 ir PA3) yra prijungta prie ST-LINK adapterio virtualaus COM porto realizavimui.

Daugiau informacijos skaitykite **STM32 Nucleo-64 boards** (**MB1136**) User manual (6.8 paragrafas 25 psl.)
https://www.st.com/resource/en/user_manual/um1724-stm32-nucleo64-boards-mb1136-stmicroelectronics.pdf

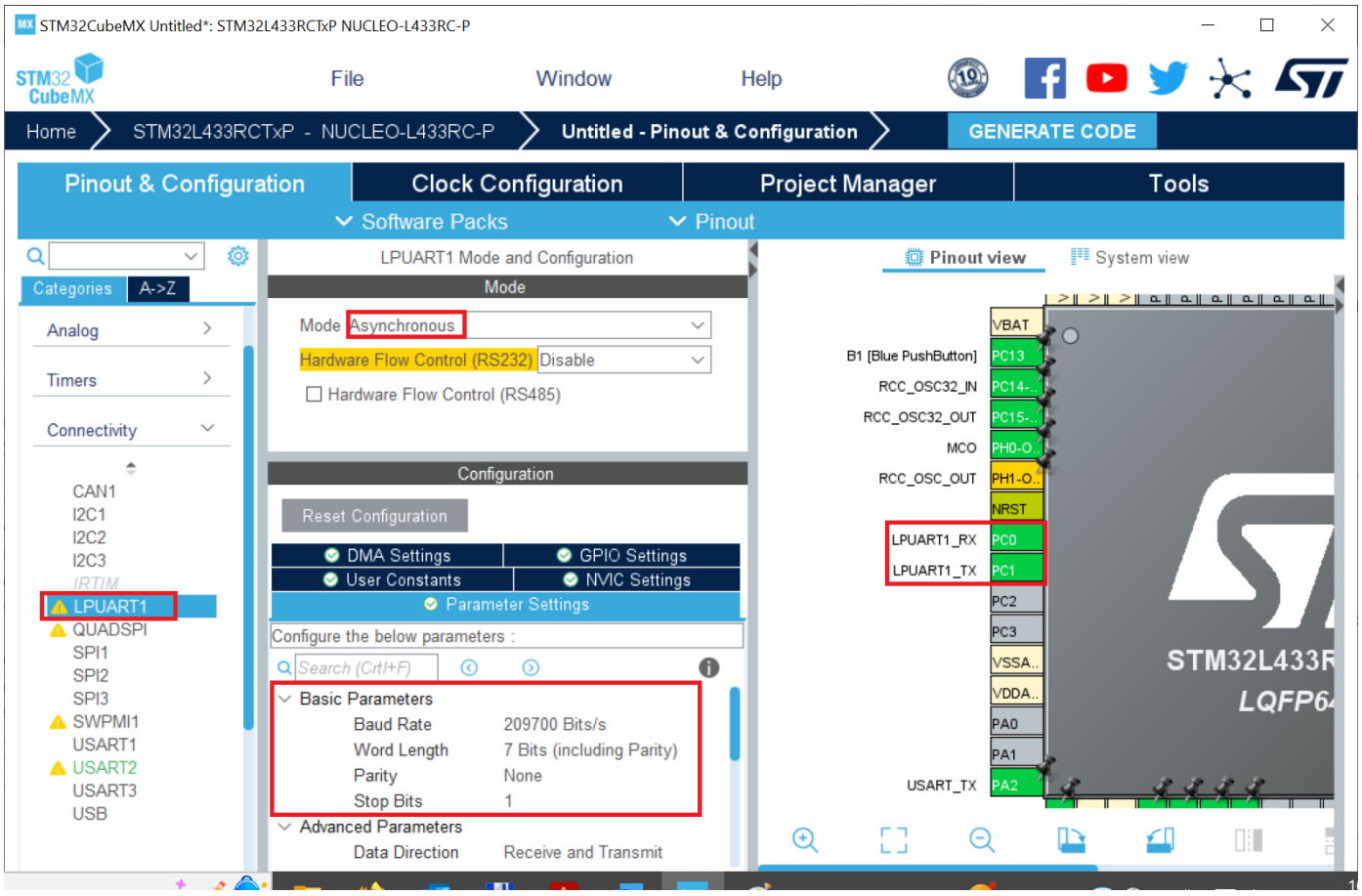


5.4.2 Nucleo-F433CP (Nucleo-64-P MB1319 tipas)

Šio tipo maketuose pagal nutylėjimą:

- STM32 mikrovaldiklio **LPUART1** yra prijungtas prie ST-LINK adapterio virtualaus COM (Virtual COM) realizavimui. Norint naudoti šį prievadą **UART-USB adapteris nereikalingas**, nes jungiame per tą pačią USB jungtį, kaip ir ST-LINK adapterį.
- STM32 mikrovaldiklio **USART1** yra prijungtas prie ARDUINO ir ST morpho jungčių. Norint per šią sąsają jungtis prie kompiuterio USB prievado yra **reikalingas UART-USB adapteris**.

Daugiau informacijos skaitykite **STM32 Nucleo-64-P boards** (**MB1319**) User manual (6.9 paragrafas 28 psl.)
https://www.st.com/resource/en/user_manual/um2206-stm32-nucleo64p-boards-mb1319-stmicroelectronics.pdf

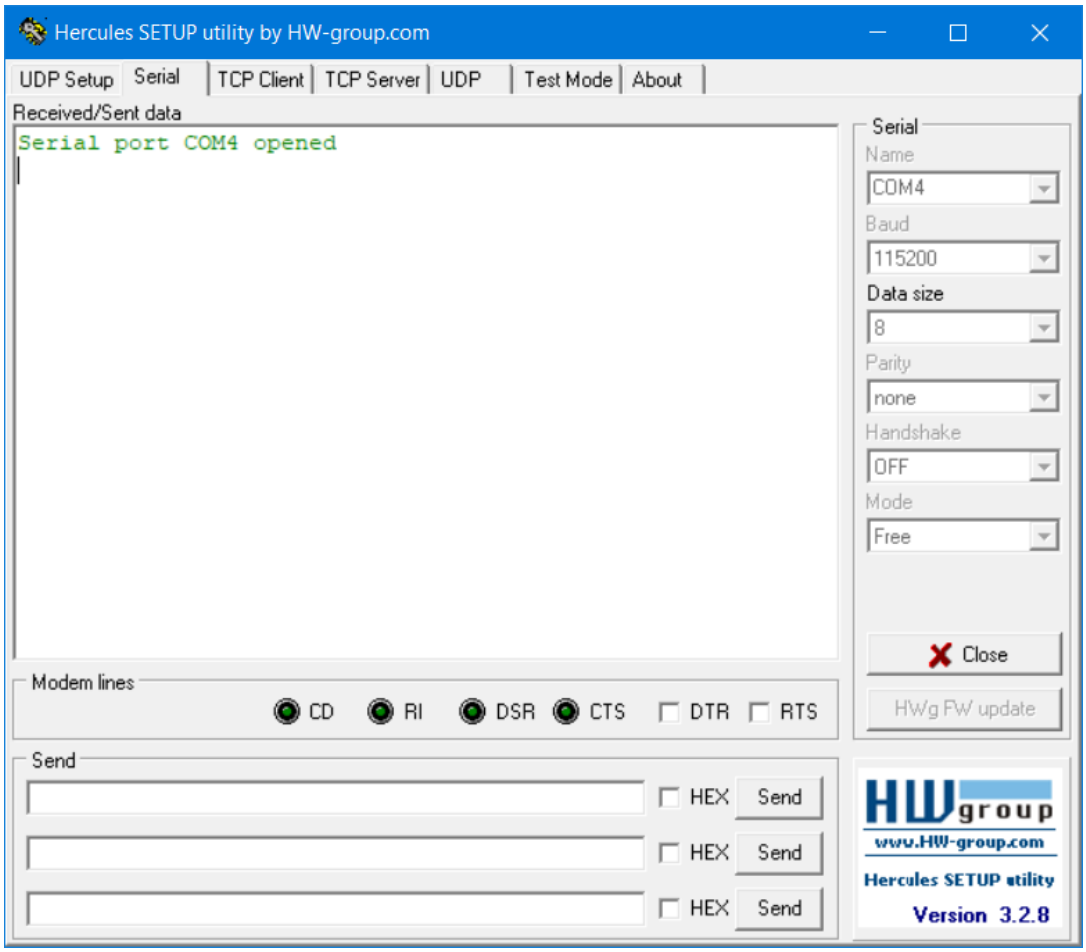


5.5 Terminalinės programos naudojimas

Kompiuteryje naudojama terminalinė programa duomenų priėmimui ir išsaugojimui. Galimų panaudoti atvirojo kodo programų pavyzdžiai yra:

1. Tera Term https://download.cnet.com/Tera-Term/3000-2094_4-75766675.html
2. Hercules Setup Utility https://www.hw-group.com/software/hercules-setup-utility

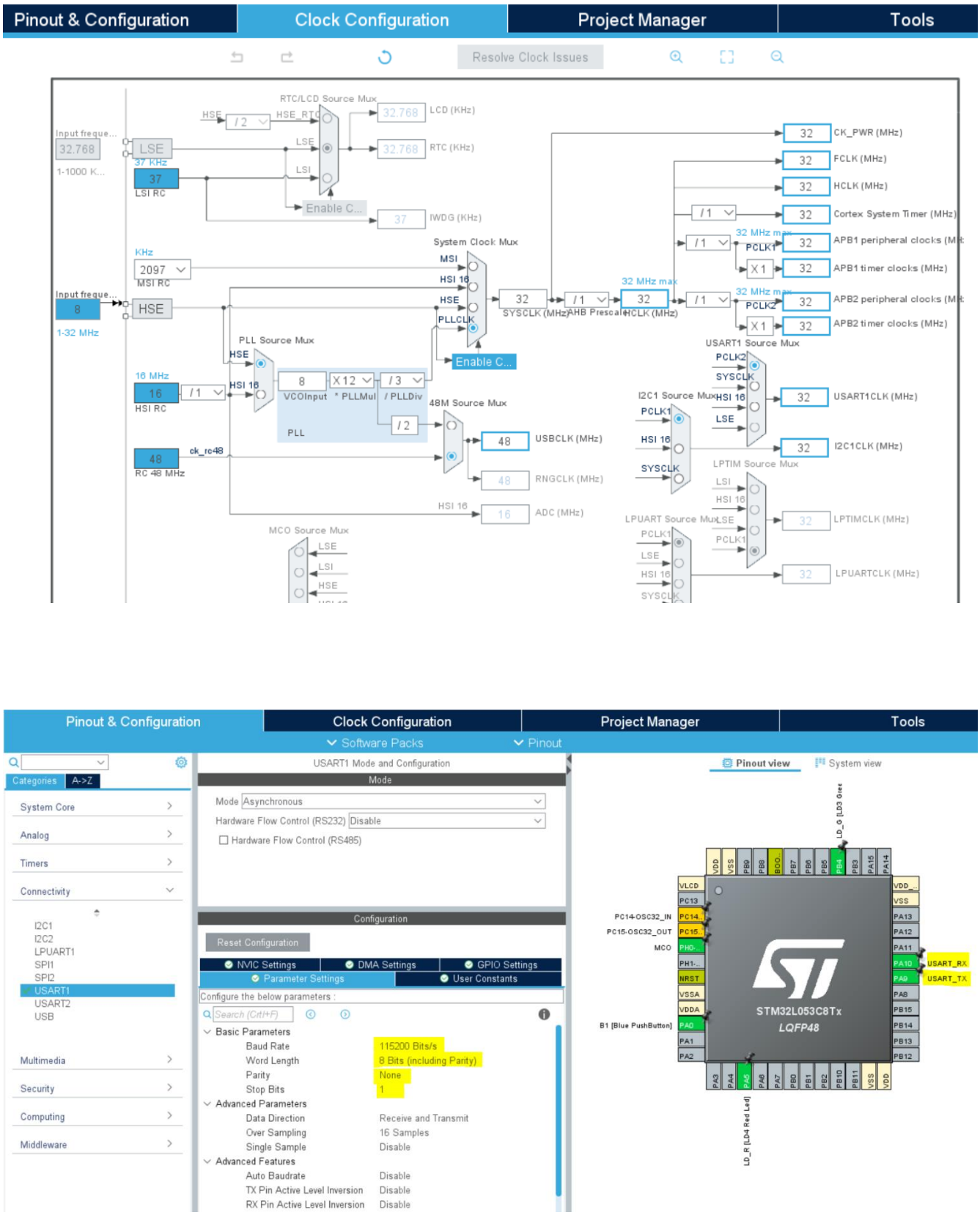
Žemiau parodytas **Hercules** terminalinės programos langas, pasiruošęs priimti duomenis per COM4 prievadą.



5.6 UART komunikacijų programavimas mikrovaldiklyje

5.6.1 USART sąsajos konfigūravimas

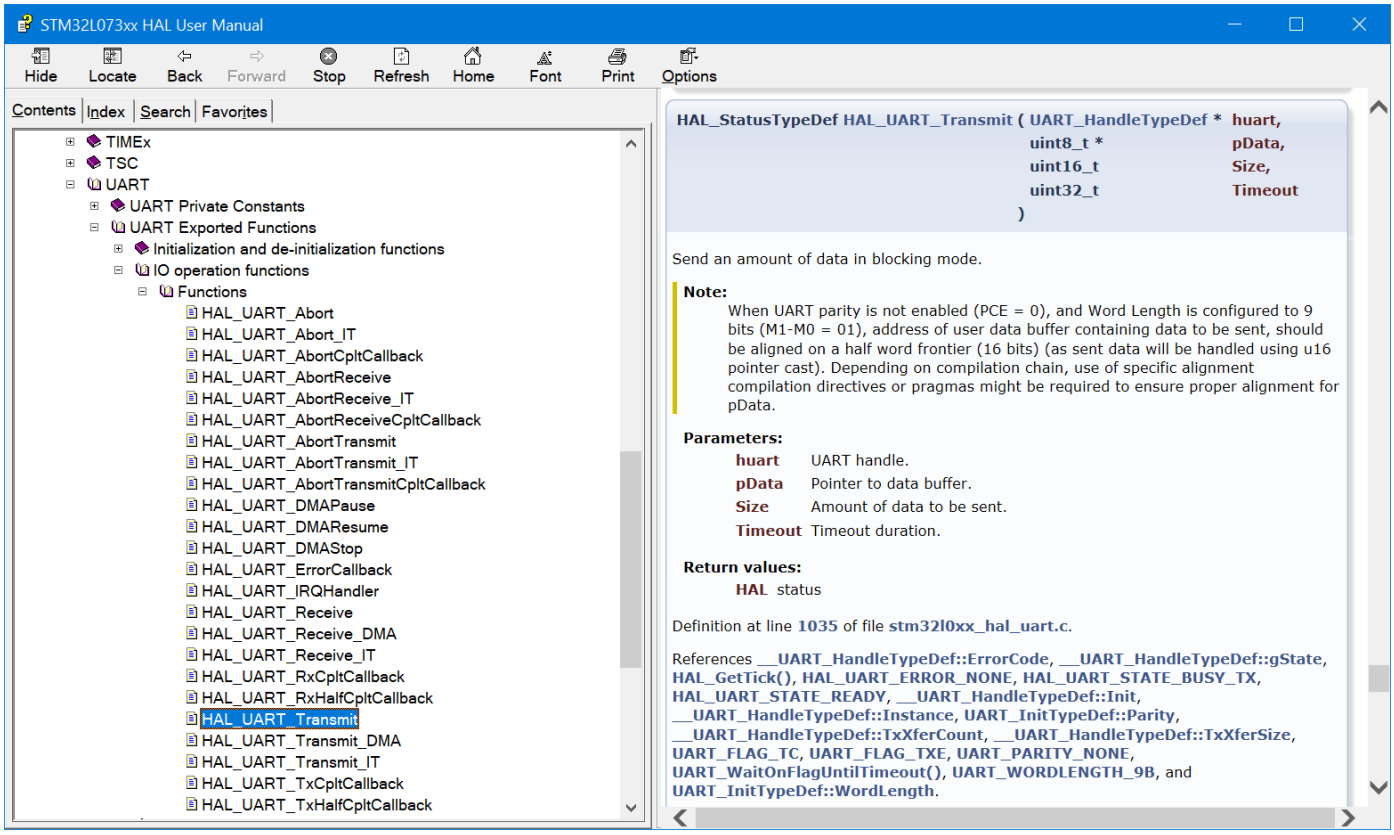
Naudokimės STM32CubeMX aplinka.



4 pav. STM32L053-DISCO UART1 sąsajos konfigūravimas

5.6.2 USART programavimas HAL funkcijų pagalba

5.6.2.1 HAL funkcijų UART skaitymui /rašymui santrauka



Išsiuntimo funkcijos:

- HAL_UART_Transmit
- HAL_UART_Transmit_IT
- HAL_UART_Transmit_DMA
- HAL_UART_TxCpltCallback

Priėmimo funkcijos:

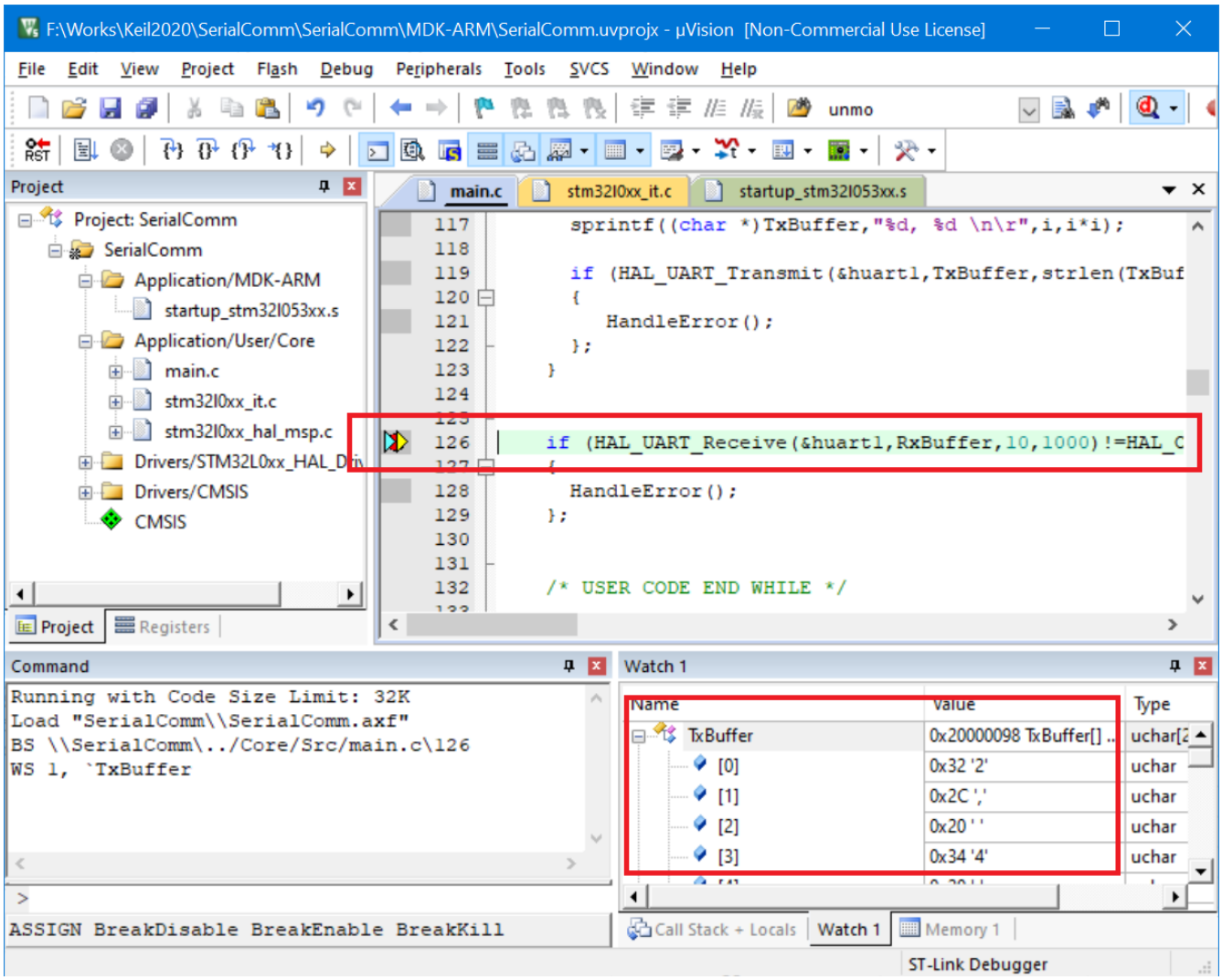
- HAL_UART_Receive
- HAL_UART_Receive_IT
- HAL_UART_Receive_DMA
- HAL_UART_RxCpltCallback

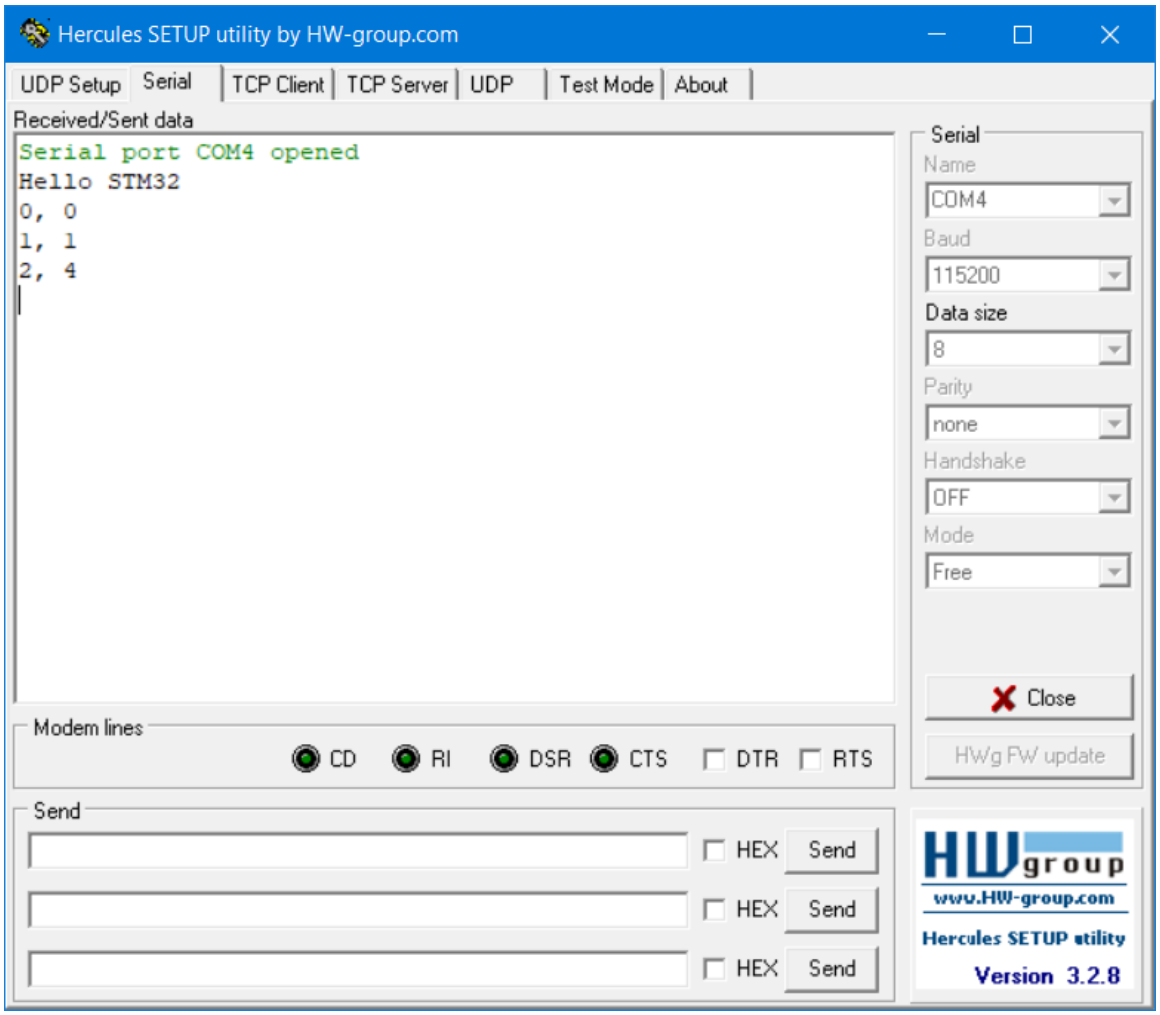
5.6.2.2 Programos fragmentas su blokuojančiomis funkcijomis

```
/* USER CODE BEGIN Includes */
#include <stdio.h>
```

```
#include <string.h>
/* USER CODE END Includes */

/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
uint8_t TxBuffer[20];
uint8_t RxBuffer[20];
int i;
void HandleError()
{
 uint32_t uart_err;
 uart_err=HAL_UART_GetError(&huart1);
}

/* USER CODE END 0 */

int main(void)
{
sprintf((char *)TxBuffer,"Hello STM32 \n\r");
if (HAL_UART_Transmit(&huart1,TxBuffer,strlen(TxBuffer),1000)!=HAL_OK)
{
  HandleError();
};

  while (1)
  {
       for (i=0;i<3;i++)
       {
          sprintf((char *)TxBuffer,"%d, %d \n\r",i,i*i);
          if (HAL_UART_Transmit(&huart1,TxBuffer,strlen(TxBuffer),1000)!=HAL_OK)
          {
              HandleError();
          };
       }

       if (HAL_UART_Receive(&huart1,RxBuffer,10,1000)!=HAL_OK)
       {
             HandleError();
       };
       /* USER CODE END WHILE */
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
    }
```

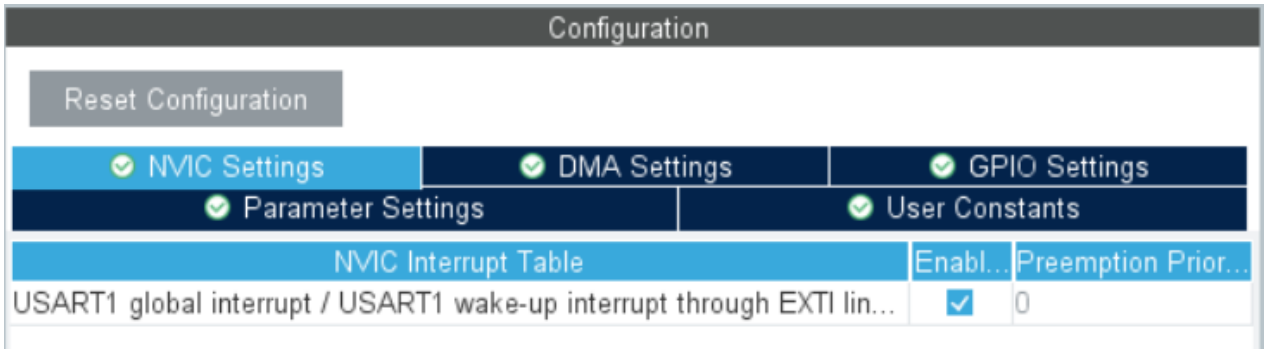
Programą įvykdžius iki eilutės, kurioje parodytas stabdymo taškas, terminalinėje programoje matome perduotus iš mikrovaldikli duomenis:


F:\Works\Keil2020\SerialComm\SerialComm\MDK-ARM\SerialComm.uvprojx - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Project
Project: SerialComm
SerialComm
Application/MDK-ARM
startup_stm32l053xx.s
Application/User/Core
main.c
stm32l0xx_it.c
stm32l0xx_hal_msp.c
Drivers/CMSIS
CMSIS
main.c
stm32l0xx_it.c
startup_stm32l053xx.s
117 sprintf((char *)TxBuffer,"%d, %d \n\r",i,i*i);
119 if (HAL_UART_Transmit(&huart1,TxBuffer,strlen(TxBuf
120 {
121 HandleError();
122 };
123 }
126 if (HAL_UART_Receive(&huart1,RxBuffer,10,1000)!=HAL_C
128 HandleError();
129 };
132 /* USER CODE END WHILE */
Command
Running with Code Size Limit: 32K
Load "SerialComm\\SerialComm.axf"
BS \\SerialComm\../Core/Src/main.c\126
WS 1, `TxBuffer
ASSIGN BreakDisable BreakEnable BreakKill
Watch 1
Name Value Type
TxBuffer 0x20000098 TxBuffer[] .. uchar[
[0] 0x32 '2' uchar
[1] 0x2C ',' uchar
[2] 0x20 ' ' uchar
[3] 0x34 '4' uchar
Call Stack + Locals
Watch 1
Memory 1
ST-Link Debugger



5.6.2.3 *Programos fragmentas su neblokuojančiomis funkcijomis*

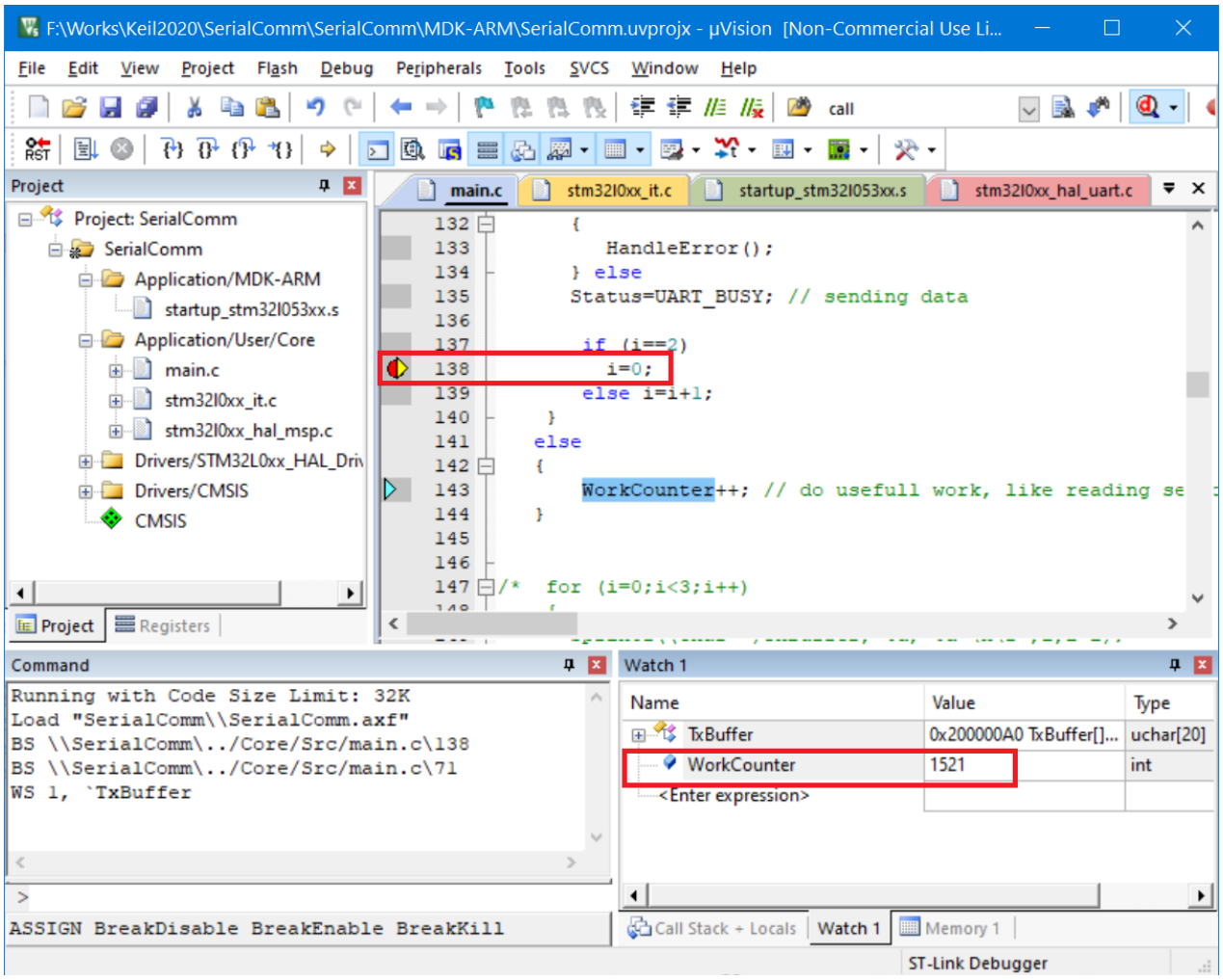
Reikia aktyvuoti pertraukties aptarnavimą ir realizuoti HAL_UART_TxCpltCallback funkciją.




```
/* USER CODE BEGIN Includes */
```

```c
#include <stdio.h>
#include <string.h>
/* USER CODE END Includes */

/* USER CODE BEGIN 0 */
uint8_t TxBuffer[20];
uint8_t RxBuffer[20];
int i;
typedef enum {UART_READY,UART_BUSY,UART_FINISHED} UART_Status;
UART_Status Status=UART_READY;
long WorkCounter=0;

/*
* Implement Transmit complete callback function
*/
void HAL_UART_TxCpltCallback(UART_HandleTypeDef *huart)
{
 if (Status==UART_BUSY)
   Status=UART_READY;  // transmition finished, can send more if needed
}

void HandleError()
{
      uint32_t uart_err;
      uart_err=HAL_UART_GetError(&huart1);
}

int main(void)
{
  i=0;
  while (1)
  {
      if (Status==UART_READY)
      {
      sprintf((char *)TxBuffer,"%d, %d \n\r",i,i*i);
      if (HAL_UART_Transmit_IT(&huart1,TxBuffer,strlen(TxBuffer))!=HAL_OK)
       {
         HandleError();
        } else
      Status=UART_BUSY; // sending data

            if (i==2)
            {
              i=0;
              Status=UART_FINISHED; // stop sending
            }
               else i=i+1;
      }
     else
    {
       WorkCounter++;      // do usefull work, like reading sensors
    }
     /* USER CODE END WHILE */
   /* USER CODE BEGIN 3 */
  }
```

```
  /* USER CODE END 3 */
}
```



WorkCounter pasiekta reikšmė rodo, kiek procesorius galėjo skirti resursų, kol UART periferinis įrenginys siuntė duomenis į kompiuterį.

5.6.2.4 Programos fragmentas naudojant DMA kanalą

5.7 Pavyzdžiai

Send and Receive data to PC without UART (STM32 USB COM) https://controllerstech.com/send-and-receive-data-to-pc-without-uart-stm32-usb-com/

Tutorial about data sending over USB: https://www.youtube.com/watch?v=92A98iEFmaA

6 Bendrieji reikalavimai techniniams sprendimams

Projektuojamas įrenginys turi būti paruoštas konfigūravimui:

1. Po pagaminimo gamykloje,
2. Eksploatacijos metu pas vartotoją.

Konfigūravimo (parametrizavimo) duomenys, pvz., PIN kodas, laikas ir data, įrenginio komunikacijų protokolo adresas, matavimo kanalų kalibravimo koeficientai, keitiklių charakteristikų ištiesinimo koeficientai ir pan., turi būti įvedami ir keičiami vykdant programą (*run-time*), o ne ją kompiliuojant (*design-time*). Konfigūravimo duomenys turi būti išsaugomi energetiškai nepriklausomoje atmintyje. Konfigūravimo duomenų įvedimui turi būti numatyta sąsaja, pvz., klaviatūra, mygtukai, komunikacijų sąsaja UART (per USB ar *Bluetooth*) su protokolu, palaikančiu įvedimą iš kompiuterio ar išmanaus įrenginio.

7 Ataskaitoje pateikti

1. Įrenginio principinę schemą
2. Komponentų parametrų skaičiavimą ir pagrindimą
3. Programos algoritmo aprašą
4. Komunikacijų su kompiuteriu protokolo aprašą
5. Įrenginio testavimą ir rezultatus
6. Matavimo paklaidos įvertinimą (jeigu matavimo uždavinys)
7. Išvadas (kur toks įrenginys gali būti taikomas)
8. Priedas. Programos tekstas