

Mikroprocesory i mikrokontrolery Laboratorium nr 5	Temat: Obsługa portu szeregowego
Grupa: 21b	Michał Lechowicz

Cel ćwiczenia:

Celem ćwiczenia było zapoznanie się z działaniem i obsługą **portu szeregowego** w procesorze **Intel 8051**. Port szeregowy otwiera nam możliwości komunikacji ze światem zewnętrznym jako jeden z interfejsów komunikacyjnych. Podobnie jak przy innych możliwościach procesora **Intel 8051**, obsługa portu szeregowego sprowadza się w dużym uproszczeniu do używania kilku rejestrów, a także przerwań.

Zadanie na ocenę 5.0:

1. Opis zadania:

W zadaniu należało napisać program, który wykorzystując system przerwań wpisywałby do szesnastobajtowego bufora cyklicznego, umiejscowionego w pamięci RAM mikrokontrolera, dane przychodzące przez **UART**.

2. Universal Asynchronous Receiver / Transmitter (UART)

Za pomocą portu szeregowego (wykorzystując standard UART) możemy wysyłać dane z i do kontrolera **ZL2MCS51**. Odbywa się to dwoma parami przewodów **RX** (P3.0) i **TX** (P3.1). Odpowiednio **RX** służy do odbioru, a **TX** do nadawania. Dla ustawionego trybu pracy 1 komunikacja odbywa się w trybie asynchronicznym. Standard UART zapewnia nam komunikację dwukierunkową po dwóch osobnych przewodach.

3. Kod programu:

```
beginTable EQU 20h ; komorka początku tablicy
sizeTable EQU 16 ; dlugosc tablicy

CSEG AT 0
    ACALL INIT ; idz do poczatku programu
CSEG AT 23h ; pod adres 23h umieszczamy instrukcje
    LJMP UART_INTERRUPT ; skoku do procedury obsługi przerwan dla UART

CSEG AT 0x100
    INIT:
        MOV SCON, #50h ; uart w trybie 1 (8 bit), REN=1
        MOV TMOD, #20h ; licznik 1 w trybie 2
        MOV TH1, #0FDh ; 9600 Bds at 11.0592MHz
        SETB TR1 ; uruchomienie licznika
        CLR TI ; wyzerowanie flagi wyslania
        SETB ES ; zezwolenie na przerwanie z portu szeregowego
        SETB EA ; globalne odblokowanie przerwan
        ACALL RESET_BUFFER ; wywolaj funkcje resetu bufora
        AJMP MAIN_LOOP ; skok do glownej petli

    RESET_BUFFER:
        MOV R0, #beginTable ; inicjalizacja R0 wartoscia 20h, wskazujaca komorkę pamięci
        MOV R1, #sizeTable ; inicjalizacja ilosci znakow do konca bufora
        RET

    MAIN_LOOP:
        SJMP MAIN_LOOP ; oczekiwanie na znak
```

UART_INTERRUPT:

PUSH PSW ; zapamiętanie ważnych rejestrów na stosie PSW i ACC
PUSH ACC

JBC TI, END_INTERRUPT ; sprawdzenie powodu przerwania, jeśli ustawiona flaga nadawania
; wyjdź z przerwania, jeśli nie idź dalej

CLR RI ; zerowanie bitu odbioru - RI

MOV @R0, SBUF ; przenies dane z SBUF do adresu wskazywanego przez R0

INC R0 ; podnieś o jeden wartość R0, która wskazuje komórkę pamięci

DJNZ R1, END_INTERRUPT ; dekrementuj R1 i skacz jeśli nie jest równe 0 do końca przerwania,
; jeśli jest równe idź dalej

ACALL RESET_BUFFER ; wywołaj funkcję resetu bufora

END_INTERRUPT: ; wyslij, ze stosu, do ACC i PSW wartości sprzed przerwania

POP ACC

POP PSW

RETI

END

Kod programu

4. Opis programu:

1. Ustawienie obsługi przerwania pod adresem 23h;

The screenshot displays the MikroC IDE interface with the following components:

- Registers:** A list of registers (r0-r7, PC, SP, etc.) with their current values. The SP register is highlighted.
- Code Editor:** Contains assembly code for file `zad.5.0.a51`. The code includes:
 - Initialization of a table at address 20h.
 - Configuration of the UART in mode 1 (8-bit).
 - Setting of the timer to 20h.
 - Enabling of the interrupt at address 23h.
 - Definition of the `UART_INTERRUPT` service routine.
- Timer/Counter 1:** Configuration window showing Mode 0 (13-bit Timer/Counter) and various control bits.
- Serial Channel:** Configuration window showing Mode 8-Bit Shift Register and baud rate settings.
- Command Window:** Shows the command `Running with Code Size Limit: 2K` and the loaded file path.
- Memory Window:** Displays memory contents starting from address 0x20.

Rys. 1 Ustawienie procedury przerwania

2. Inicjalizacja rejestru SCON (dla UART), wektora przerwań (linie 16 i 17) oraz licznika T1 (linie 12 do 14);

The screenshot displays the Keil uVision IDE interface for the assembly file 'zad.5.0.a51'. The main window shows the assembly code with the following key sections:

- INIT:** Initializes SCON to 0x50 (8-bit UART, REN=1), TMOD to 0x20 (Timer 1, Mode 2), TH1 to 0xFD (9600 Bds at 11.0592MHz), and TR1 to 1 (Timer 1 on). It also sets TI (Timer Interrupt) and ES (External Interrupt Enable).
- RESET_BUFFER:** A sub-routine to reset the UART buffer.
- MAIN_LOOP:** A loop that jumps to the interrupt service routine.
- UART_INTERRUPT:** An interrupt service routine that pushes PSW and ACC, checks the TI flag, moves data from SBUF to R0, increments R0, decrements R1, and calls RESET_BUFFER.

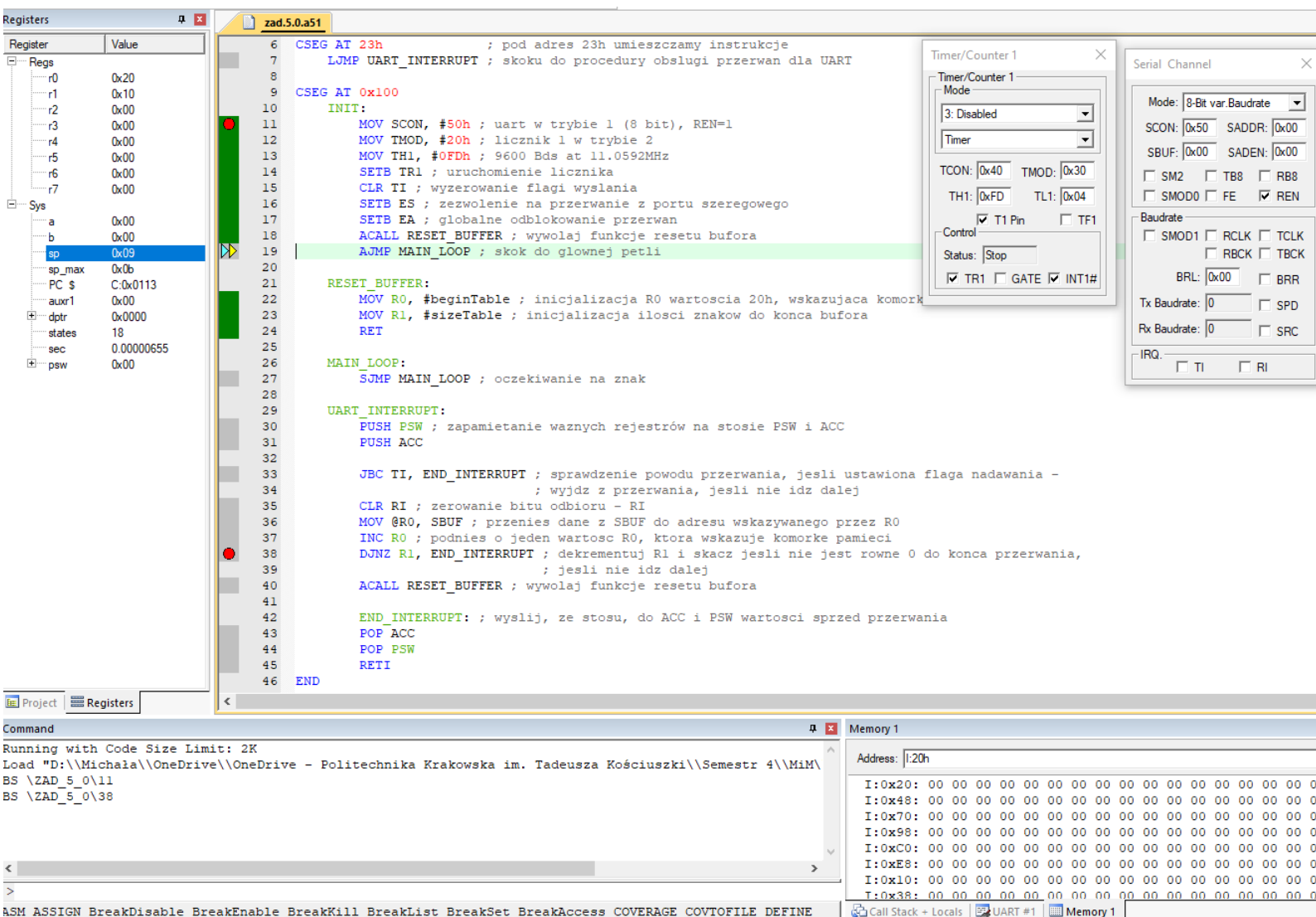
On the right, the **Timer/Counter 1** window is configured with Mode 2 (8-bit auto-reload), TCON=0x40, TMOD=0x20, TH1=0xFD, and TL1=0x04. The **Serial Channel** window is configured with Mode 8-Bit var.Baudrate, SCON=0x50, SADDR=0x00, SBUF=0x00, SADDEN=0x00, and Baudrate 28645.

The **Registers** window on the left shows the state of various registers, including R0-R7, SP, PC, and PSW.

The **Command** window at the bottom shows the command line and the **Memory** window shows the memory dump.

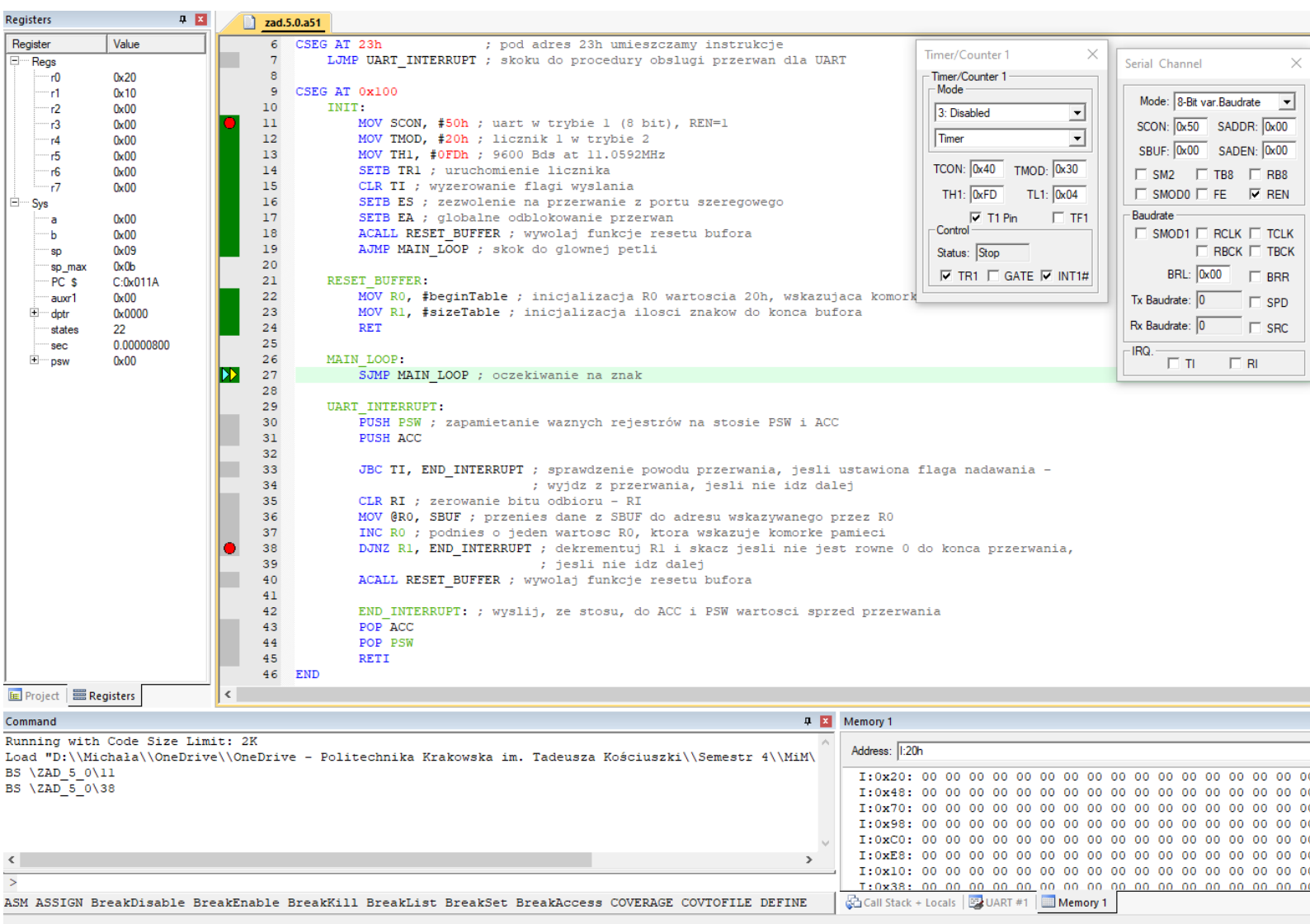
Rys. 2. Inicjalizacja programu

3. Inicjalizacja rejestrów R0 (wskazującego komórkę pamięci, która jest początkiem bufora cyklicznego) i R1 (wskazującego ilość znaków do przepełnienia się bufora);



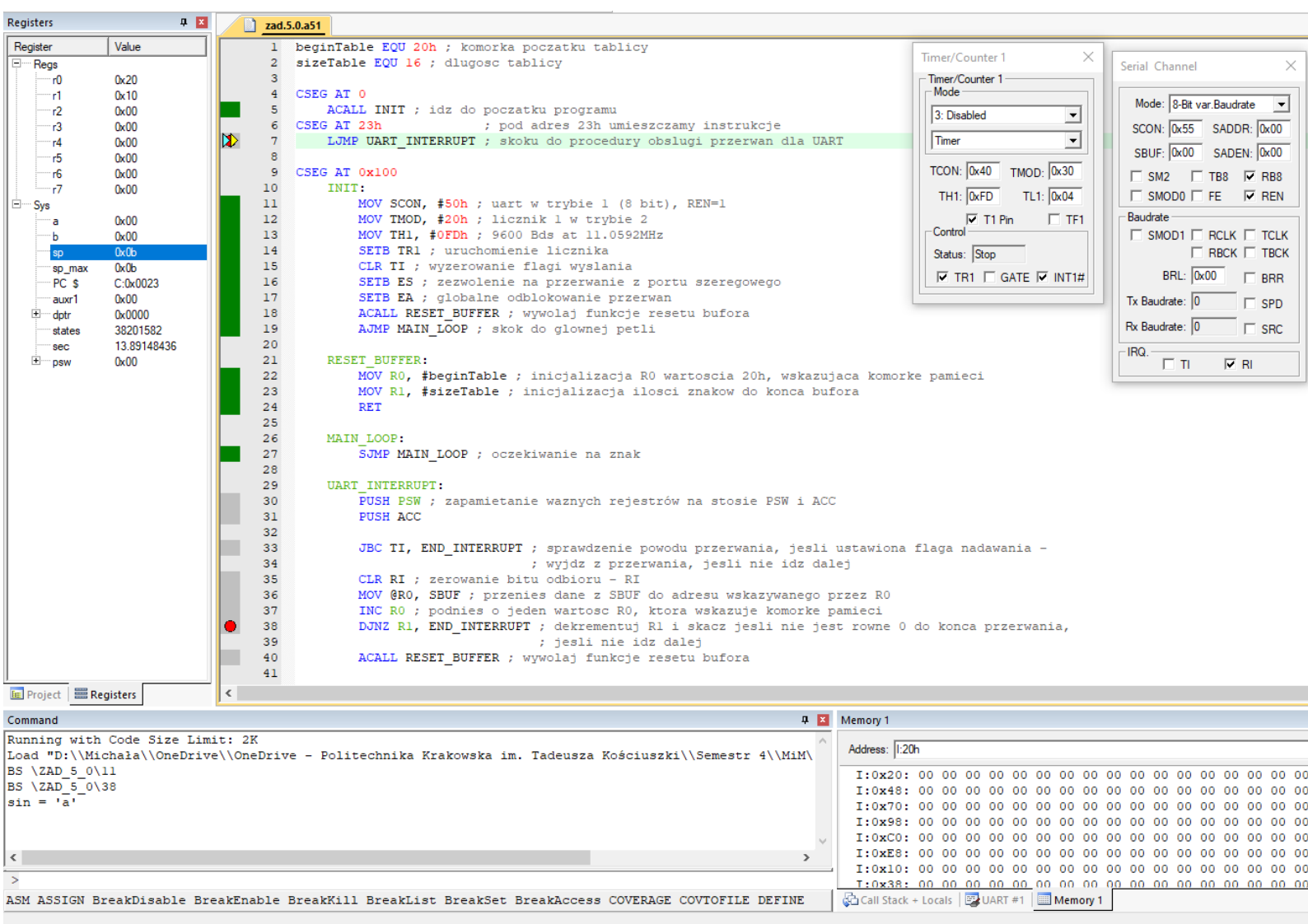
Rys. 3. Ustawienie rejestrów

- Następnie program wchodzi do głównej pętli, gdzie oczekuje na przerwanie ze strony portu szeregowego (UART);



Rys. 4. Oczekiwanie na znak od użytkownika

5. Gdy użytkownik wyśle znak przez UART, następuje aktywacja flagi RI oraz wejście do obsługi przerwania;



Rys. 5. Wejście do obsługi przerwania

6. W obsłudze przerwania, rejestr PSW oraz Akumulator są przenoszone na stos;

The screenshot displays the Keil IDE interface with the following components:

- Registers Window:** Shows the state of registers. The stack pointer (SP) is at 0x0D. The Program Counter (PC) is at 0x0120. The PSW (Program Status Word) is at 0x00.
- Assembly Code:**

```

6 CSEG AT 23h ; pod adres 23h umieszczamy instrukcje
7 LJMP UART_INTERRUPT ; skoku do procedury obsługi przerw dla UART
8
9 CSEG AT 0x100
10 INIT:
11 MOV SCON, #50h ; uart w trybie 1 (8 bit), REN=1
12 MOV TMOD, #20h ; licznik 1 w trybie 2
13 MOV TH1, #0FDh ; 9600 Bds at 11.0592MHz
14 SETB TR1 ; uruchomienie licznika
15 CLR TI ; wyzerowanie flagi wysłania
16 SETB ES ; zezwolenie na przerwanie z portu szeregowego
17 SETB EA ; globalne odblokowanie przerw
18 ACALL RESET_BUFFER ; wywołaj funkcję resetu bufora
19 AJMP MAIN_LOOP ; skok do głównej petli
20
21 RESET_BUFFER:
22 MOV R0, #beginTable ; inicjalizacja R0 wartoscia 20h, wskazująca komórkę
23 MOV R1, #sizeTable ; inicjalizacja ilości znaków do końca bufora
24 RET
25
26 MAIN_LOOP:
27 SJMP MAIN_LOOP ; oczekiwanie na znak
28
29 UART_INTERRUPT:
30 PUSH PSW ; zapamiętanie ważnych rejestrów na stosie PSW i ACC
31 PUSH ACC
32
33 JBC TI, END_INTERRUPT ; sprawdzenie powodu przerwania, jeśli ustawiona flaga nadawania -
34 ; wyjdź z przerwania, jeśli nie idź dalej
35 CLR RI ; zerowanie bitu odbioru - RI
36 MOV @R0, SBUF ; przenies dane z SBUF do adresu wskazywanego przez R0
37 INC R0 ; podnieś o jeden wartość R0, która wskazuje komórki pamięci
38 DJNZ R1, END_INTERRUPT ; dekrementuj R1 i skacz jeśli nie jest równe 0 do końca przerwania,
39 ; jeśli nie idź dalej
40 ACALL RESET_BUFFER ; wywołaj funkcję resetu bufora
41
42 END_INTERRUPT: ; wyslij, ze stosu, do ACC i PSW wartości sprzed przerwania
43 POP ACC
44 POP PSW
45 RETI
46 END

```
- Timer/Counter 1 Window:** Shows the configuration for Timer/Counter 1. Mode is set to 3: Disabled. TCON is 0x40, TMOD is 0x30. TH1 is 0xFD, TL1 is 0x04. Control bits TR1, GATE, and INT1# are checked.
- Serial Channel Window:** Shows the configuration for the serial channel. Mode is 8-Bit var. Baudrate. SCON is 0x55, SADDR is 0x00. SBUF is 0x00, SADEN is 0x00. SM2, TB8, RB8, SMOD0, FE, and REN are checked.
- Memory Window:** Shows the memory contents starting at address 0x20h. The memory is filled with zeros.
- Command Window:** Shows the command line with the path to the project files and the command to run the program.

Rys. 6. Zapamiętanie rejestrów PSW i ACC na stosie

7. Następnie, jeśli przerwanie wystąpiło z powodu aktywacji flagi TI, program idzie do procedury zakończenia przerwania, a jeśli nie, idzie dalej;

The screenshot displays the Keil uVision IDE interface. The main window shows the assembly code for 'zad.5.0.a51'. The code includes initialization, a main loop, and an interrupt service routine (UART_INTERRUPT). The UART_INTERRUPT routine checks the TI flag and either resets the buffer or continues the loop. The Registers window shows the current state of registers, and the Memory window shows the contents of memory addresses starting from 0x20. The Command window shows the execution progress.

Registers:

Register	Value
r0	0x20
r1	0x10
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
PC	0x0123
SP	0x0d
SP_MAX	0x0d
AUXR1	0x00
DPTR	0x0000
STATES	38201590
SEC	13.89148727
PSW	0x00

Assembly Code:

```
6 CSEG AT 23h ; pod adres 23h umieszczamy instrukcje
7 LJMP UART_INTERRUPT ; skoku do procedury obsługi przerw dla UART
9 CSEG AT 0x100
10 INIT:
11 MOV SCON, #50h ; uart w trybie 1 (8 bit), REN=1
12 MOV TMOD, #20h ; licznik 1 w trybie 2
13 MOV TH1, #0FDh ; 9600 Bds at 11.0592MHz
14 SETB TR1 ; uruchomienie licznika
15 CLR TI ; wyzerowanie flagi wysłania
16 SETB ES ; zezwolenie na przerwanie z portu szeregowego
17 SETB EA ; globalne odblokowanie przerw
18 ACALL RESET_BUFFER ; wywołaj funkcję resetu bufora
19 AJMP MAIN_LOOP ; skok do głównej petli
21
22 RESET_BUFFER:
23 MOV R0, #beginTable ; inicjalizacja R0 wartoscia 20h, wskazująca komórkę
24 MOV R1, #sizeTable ; inicjalizacja ilości znaków do końca bufora
25 RET
26
27 MAIN_LOOP:
28 SJMP MAIN_LOOP ; oczekiwanie na znak
29
30 UART_INTERRUPT:
31 PUSH PSW ; zapamiętanie ważnych rejestrów na stosie PSW i ACC
32 PUSH ACC
33
34 JBC TI, END_INTERRUPT ; sprawdzenie powodu przerwania, jeśli ustawiona flaga nadawania -
35 ; wyjdź z przerwania, jeśli nie idź dalej
36 CLR RI ; zerowanie bitu odbioru - RI
37 MOV @R0, SBUF ; przenies dane z SBUF do adresu wskazywanego przez R0
38 INC R0 ; podnieś o jeden wartość R0, która wskazuje komórkę pamięci
39 DJNZ R1, END_INTERRUPT ; dekrementuj R1 i skacz jeśli nie jest równe 0 do końca przerwania,
40 ; jeśli nie idź dalej
41 ACALL RESET_BUFFER ; wywołaj funkcję resetu bufora
42
43 END_INTERRUPT: ; wyslij, ze stosu, do ACC i PSW wartości sprzed przerwania
44 POP ACC
45 POP PSW
46 RETI
47 END
```

Memory 1:

Address	Value
I:0x20	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0x48	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0x70	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0x98	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0xC0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0xE8	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0x10	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
T:0x38	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Command:

```
Running with Code Size Limit: 2K
Load "D:\Michala\OneDrive\OneDrive - Politechnika Krakowska im. Tadeusza Kosciuszki\Semestr 4\MiM\
BS \ZAD_5_0\11
BS \ZAD_5_0\38
sin = 'a'
```

Rys. 7. Sprawdzenie powodu przerwania

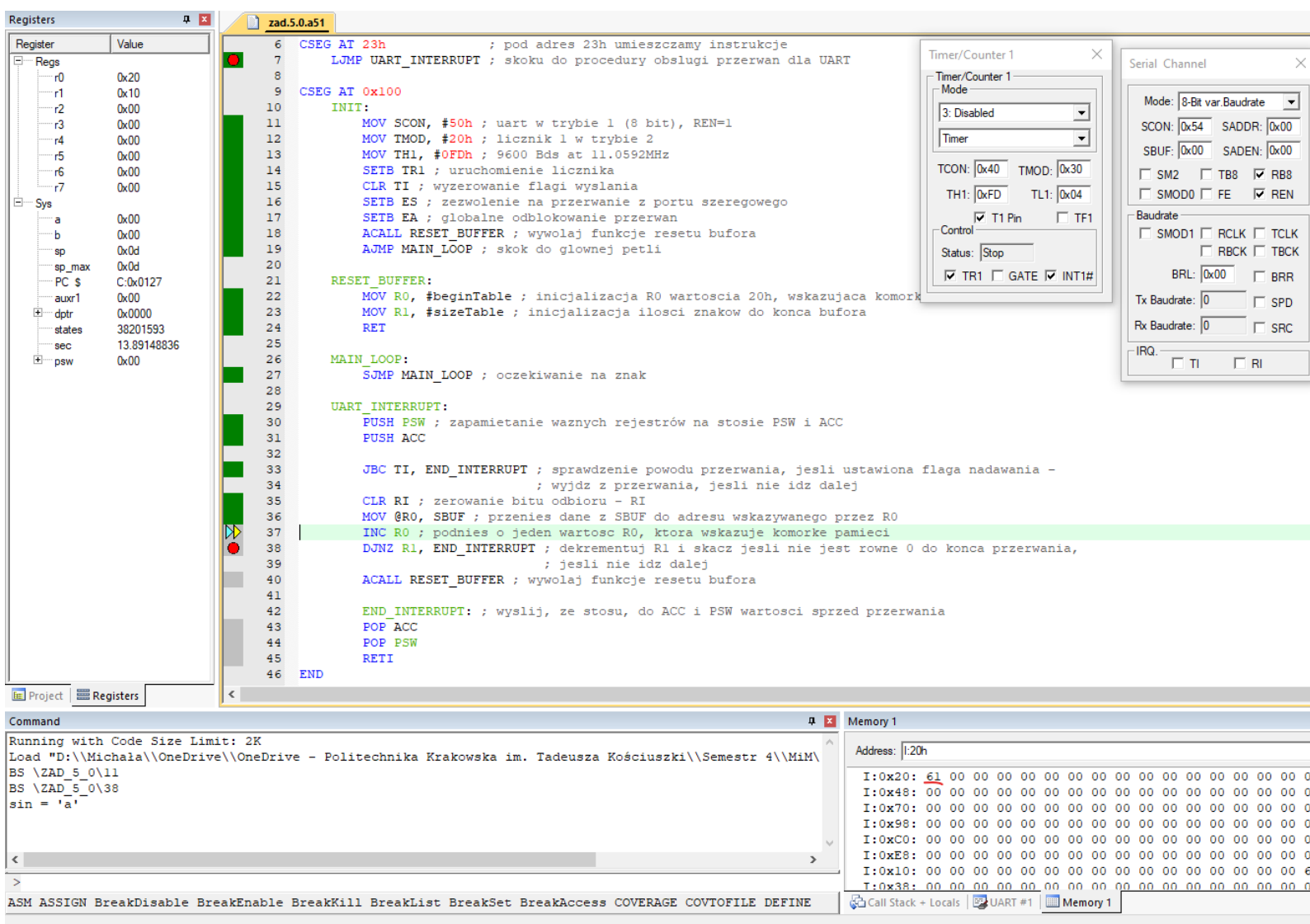
8. Następuję zerowanie bitu odbioru RI;

The screenshot shows the Keil IDE interface with the following components:

- Registers Window (Left):** Displays the state of various registers including r0-r7, Sys, a, b, sp, sp_max, PC, auxr1, dptr, states, sec, and psw.
- Main Editor:** Contains assembly code for 'zad.5.0.a51'. The code includes an interrupt service routine 'UART_INTERRUPT' where the RI bit is cleared using the instruction 'CLR RI'. The code also includes an 'INIT' section and a 'MAIN_LOOP'.
- Timer/Counter 1 Window (Right):** Shows the configuration for Timer/Counter 1, including Mode (3: Disabled), TCON, TMOD, TH1, TL1, and Control settings.
- Serial Channel Window (Right):** Shows the configuration for the Serial Channel, including Mode (8-Bit var.Baudrate), SCON, SADDR, SBUF, SADDEN, and Baudrate settings.
- Command Window (Bottom Left):** Shows the command 'sin = 'a'' and other assembly-related commands.
- Memory Window (Bottom Right):** Shows the memory dump for address 0x20h, displaying the contents of memory locations from 0x20 to 0x38.

Rys. 8. Zerowanie bitu RI

9. Dalej następuje przenoszenie danych z rejestru SBUF, czyli znaku wprowadzonego przez użytkownika, do adresu wskazywanego przez rejestr R0;



Rys. 9. Przenoszenie znaku z SBUF do pamięci RAM

10. W następnym kroku program inkrementuje R0, by wskazywał kolejną komórkę bufora cyklicznego (w pamięci RAM);

The screenshot displays the Keil uVision IDE interface. The main window shows the assembly code for 'zad.5.0.a51'. The code is as follows:

```

6 CSEG AT 23h ; pod adres 23h umieszczamy instrukcje
7 LUMP UART_INTERRUPT ; skoku do procedury obsługi przerwan dla UART
8
9 CSEG AT 0x100
10 INIT:
11 MOV SCON, #50h ; uart w trybie 1 (8 bit), REN=1
12 MOV TMOD, #20h ; licznik 1 w trybie 2
13 MOV TH1, #0FDh ; 9600 Bds at 11.0592MHz
14 SETB TR1 ; uruchomienie licznika
15 CLR TI ; wyzerowanie flagi wyslania
16 SETB ES ; zezwolenie na przerwanie z portu szeregowego
17 SETB EA ; globalne odblokowanie przerwan
18 ACALL RESET_BUFFER ; wywołaj funkcje resetu bufora
19 AJMP MAIN_LOOP ; skok do glownej petli
20
21 RESET_BUFFER:
22 MOV R0, #beginTable ; inicjalizacja R0 wartoscia 20h, wskazujaca komorke
23 MOV R1, #sizeTable ; inicjalizacja ilosci znakow do konca bufora
24 RET
25
26 MAIN_LOOP:
27 SJMP MAIN_LOOP ; oczekiwanie na znak
28
29 UART_INTERRUPT:
30 PUSH PSW ; zapamietanie waznych rejestrów na stosie PSW i ACC
31 PUSH ACC
32
33 JBC TI, END_INTERRUPT ; sprawdzenie powodu przerwania, jesli ustawiona flaga nadawania -
34 ; wyjdź z przerwania, jesli nie idź dalej
35 CLR RI ; zerowanie bitu odbioru - RI
36 MOV @R0, SBUF ; przenies dane z SBUF do adresu wskazywanego przez R0
37 INC R0 ; podnies o jeden wartosc R0, ktora wskazuje komorke pamieci
38 DJNZ R1, END_INTERRUPT ; dekrementuj R1 i skacz jesli nie jest rowne 0 do konca przerwania,
39 ; jesli nie idź dalej
40 ACALL RESET_BUFFER ; wywołaj funkcje resetu bufora
41
42 END_INTERRUPT: ; wyslij, ze stosu, do ACC i PSW wartosci sprzed przerwania
43 POP ACC
44 POP PSW
45 RETI
46 END

```

The 'Timer/Counter 1' window shows the following configuration:

- Mode: 3: Disabled
- Timer: 1
- TCN: 0x40, TMOD: 0x30
- TH1: 0xFD, TL1: 0x04
- Control: ☒ T1 Pin, ☐ TF1
- Status: Stop
- ☒ TR1, ☐ GATE, ☒ INT1#

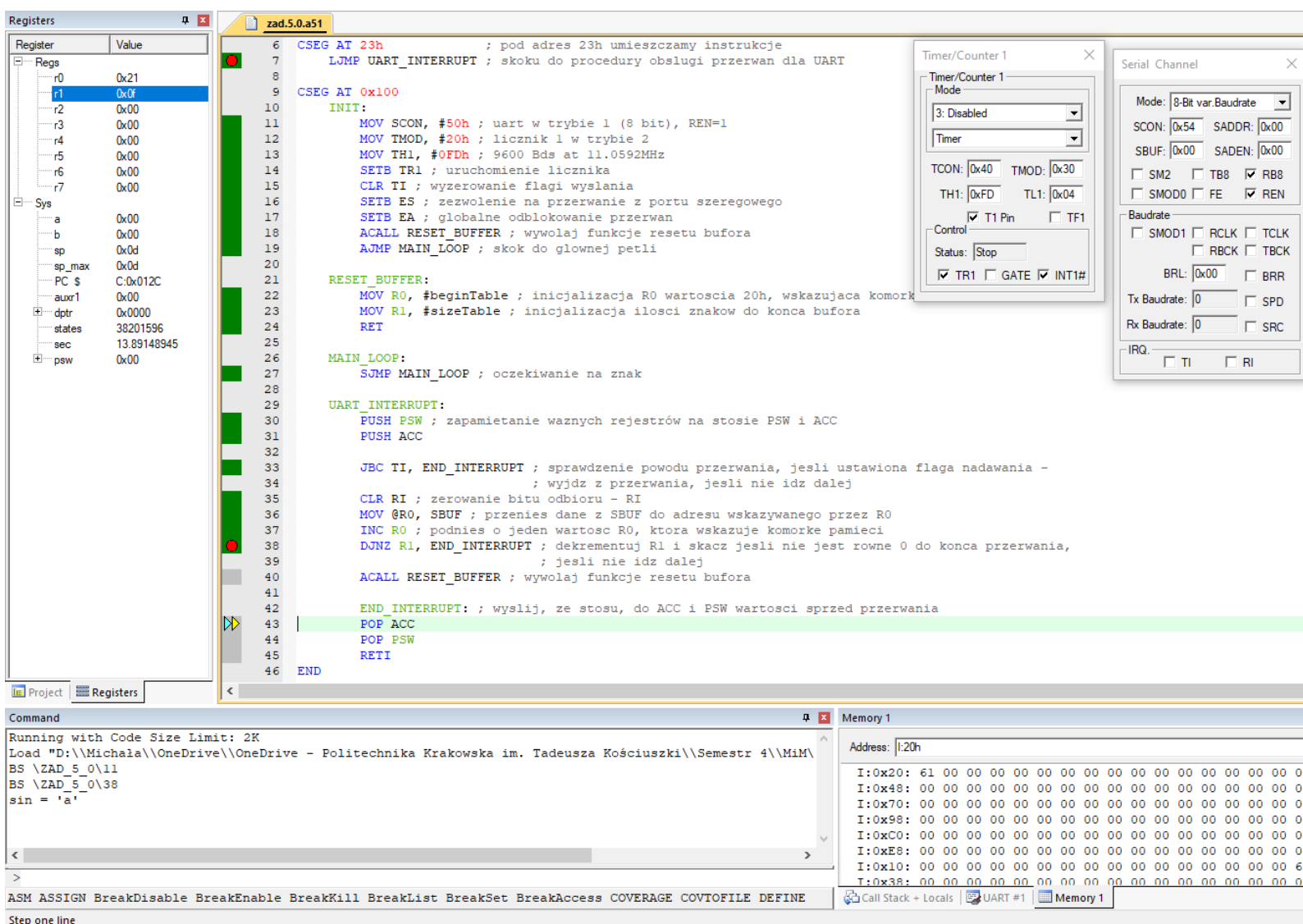
The 'Serial Channel' window shows the following configuration:

- Mode: 8-Bit var.Baudrate
- SCON: 0x54, SADDR: 0x00
- SBUF: 0x00, SADDN: 0x00
- ☐ SM2, ☐ TB8, ☒ RB8
- ☐ SMOD0, ☐ FE, ☒ REN
- Baudrate: ☐ SMOD1, ☐ RCLK, ☐ TCLK, ☐ RBCK, ☐ TBCK
- BRL: 0x00, ☐ BRR
- Tx Baudrate: 0, ☐ SPD
- Rx Baudrate: 0, ☐ SRC
- IRQ: ☐ TI, ☐ RI

The 'Command' window shows the command 'sin = 'a'' being executed. The 'Memory' window shows the contents of memory addresses starting from 0x20.

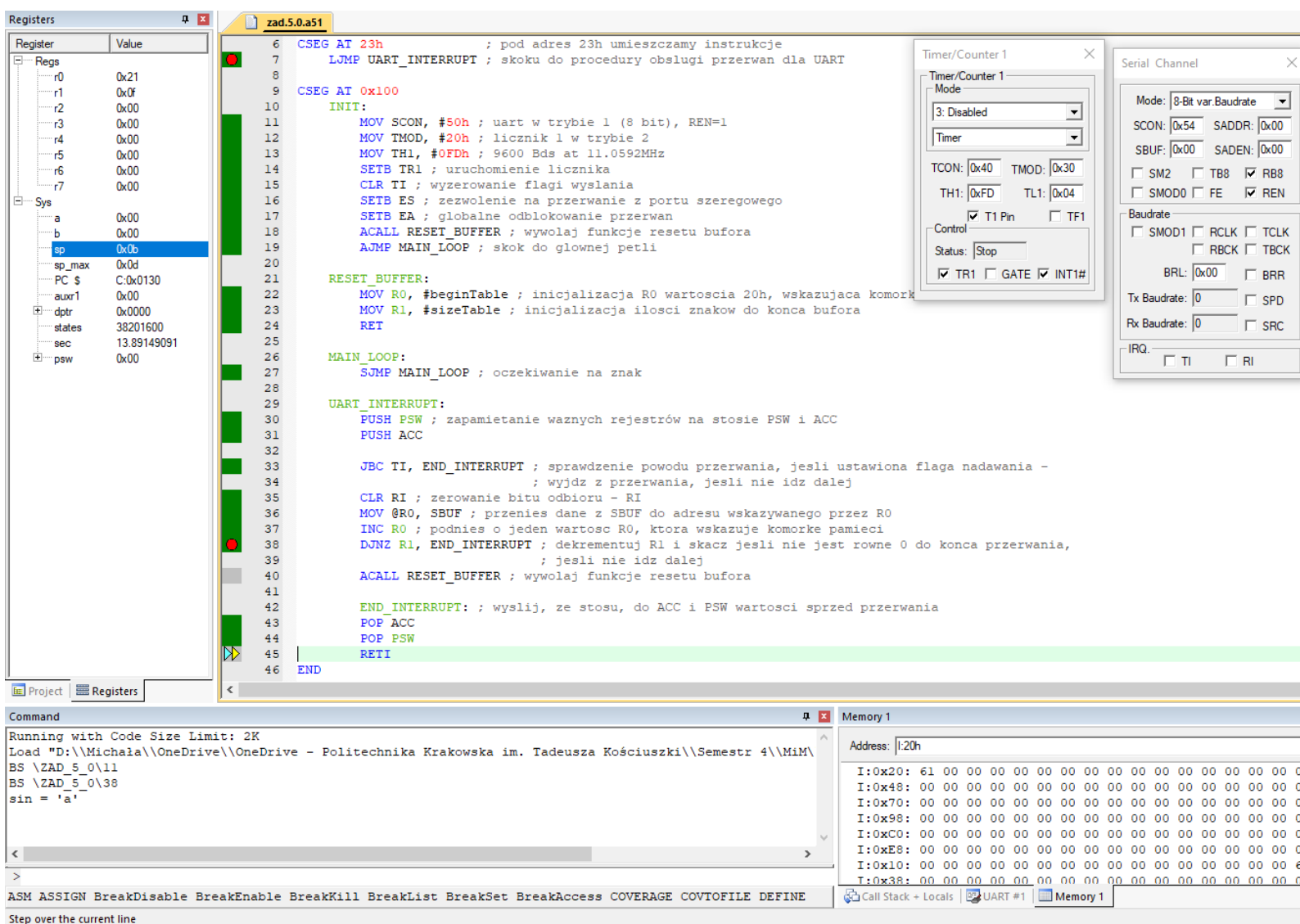
Rys. 10. Inkrementacja R0

11. Dalej algorytm dekrementuje rejestr R1 oraz sprawdza, czy jest równy zero, jeśli tak, oznacza to, że osiągnęliśmy koniec bufora i wywoływana jest funkcja resetująca rejestry R0 i R1 (opis w punkcie 13). Jeśli nie, program idzie do obsługi zakończenia przerwania;



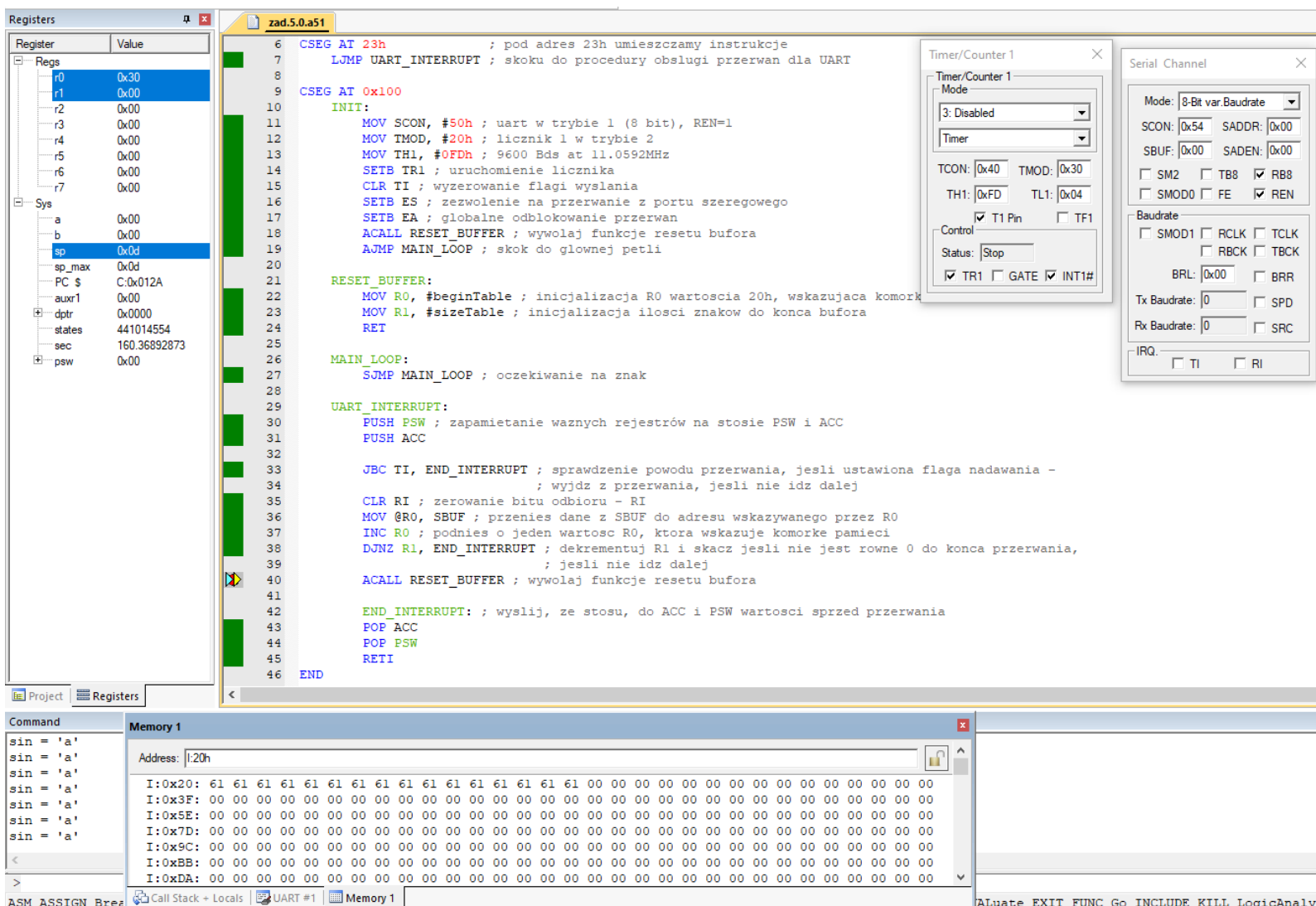
Rys. 11. Sprawdzenie warunku końca bufora cyklicznego z dekrementacją rejestru R1

12. W trakcie procedury zakończenia przerwania, program ściąga ze stosu wartości rejestrów PSW i ACC sprzed rozpoczęcia przerwania oraz wychodzi z niego.



Rys. 12. Przywrócenie wartości rejestrów ACC i PSW ze stosu do rejestrów

13. Jeśli w trakcie działania programu bufor cykliczny zapełni się, wskaźnik na rejestr R0 zostanie zresetowany do wartości domyślnej (początek bufora), a do rejestru R1 zostanie załadowana długość bufora cyklicznego.



Rys. 13. Zapełnienie bufora cyklicznego

Registers

Register	Value
r0	0x21
r1	0x0f
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
a	0x00
b	0x00
sp	0x09
sp_max	0x0f
PC \$	C:0x011A
auxr1	0x00
dpnr	0x0000
states	441014592
sec	160.36894255
psw	0x00

zad.5.0.a51

```

6 CSEG AT 23h ; pod adres 23h umieszczamy instrukcje
7 LJMP UART_INTERRUPT ; skoku do procedury obsługi przerw dla UART
8
9 CSEG AT 0x100
10 INIT:
11 MOV SCON, #50h ; uart w trybie 1 (8 bit), REN=1
12 MOV TMOD, #20h ; licznik 1 w trybie 2
13 MOV TH1, #0FDh ; 9600 Bds at 11.0592MHz
14 SETB TR1 ; uruchomienie licznika
15 CLR TI ; wyzerowanie flagi wysłania
16 SETB ES ; zezwolenie na przerwanie z portu szeregowego
17 SETB EA ; globalne odblokowanie przerw
18 ACALL RESET_BUFFER ; wywołaj funkcję resetu bufora
19 AJMP MAIN_LOOP ; skok do głównej petli
20
21 RESET_BUFFER:
22 MOV R0, #beginTable ; inicjalizacja R0 wartoscia 20h, wskazująca komórkę
23 MOV R1, #sizeTable ; inicjalizacja ilości znaków do końca bufora
24 RET
25
26 MAIN_LOOP:
27 SJMP MAIN_LOOP ; oczekiwanie na znak
28
29 UART_INTERRUPT:
30 PUSH PSW ; zapamiętanie ważnych rejestrów na stosie PSW i ACC
31 PUSH ACC
32
33 JBC TI, END_INTERRUPT ; sprawdzenie powodu przerwania, jeśli ustawiona
34 ; wyjdź z przerwania, jeśli nie idź dalej
35 CLR RI ; zerowanie bitu odbioru - RI
36 MOV @R0, SBUF ; przenies dane z SBUF do adresu wskazywanego przez R0
37 INC R0 ; podnieś o jeden wartość R0, która wskazuje komórki pamięci
38 DJNZ R1, END_INTERRUPT ; dekrementuj R1 i skacz jeśli nie jest równe 0
39 ; jeśli nie idź dalej
40 ACALL RESET_BUFFER ; wywołaj funkcję resetu bufora
41
42 END_INTERRUPT: ; wyslij, ze stosu, do ACC i PSW wartości sprzed przerwa
43 POP ACC
44 POP PSW
45 RETI
46 END

```

Timer/Counter 1

Mode

3: Disabled

Timer

TCON: 0x04

TMOD: 0x30

TH1: 0xFD

TL1: 0x04

☒ T1 Pin

☐ TF1

Status: Stop

☒ TR1

☐ GATE

☒ INT1#

Serial Channel

Mode: 8-Bit var.Baudrate

SCON: 0x54

SADDR: 0x00

SBUF: 0x00

SADEN: 0x00

☐ SM2

☐ TB8

☒ RB8

☐ SMOD0

☐ FE

☒ REN

Baudrate

☐ SMOD1
☐ RCLK
☐ TCLK
☐ RBCK
☐ TBCK

BRL: 0x00

☐ BRR

Tx Baudrate: 0

☐ SPD

Rx Baudrate: 0

☐ SRC

IRQ:

☐ TI

☐ RI

Project Registers

Command

```

sin = 'a'
sin = 'a'
sin = 'a'
sin = 'a'
sin = 'a'
sin = 'a'
sin = 'z'

```

Memory 1

Address: 0x20h

I:0x20:	7A 61 61 61 61 61 61 61 61 61 61 61 61 61 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
I:0x3F:	00 00
I:0x5E:	00 00
I:0x7D:	00 00
I:0x9C:	00 00
I:0xBB:	00 00
I:0xDA:	00 00

ASM ASSIGN Bree

Call Stack + Locals

UART #1

Memory 1

ALuate EXIT FUNC

Rys. 14. Powrót na początek bufora cyklicznego, po zapełnieniu

4. Podsumowanie i wnioski:

Komunikacja z urządzeniami zewnętrznymi nie jest skomplikowana dla badanego kontrolera. Dzięki wsparciu dla standardu UART, możliwe jest sparowanie mikrokontrolera z szeroką gamą urządzeń peryferyjnych.