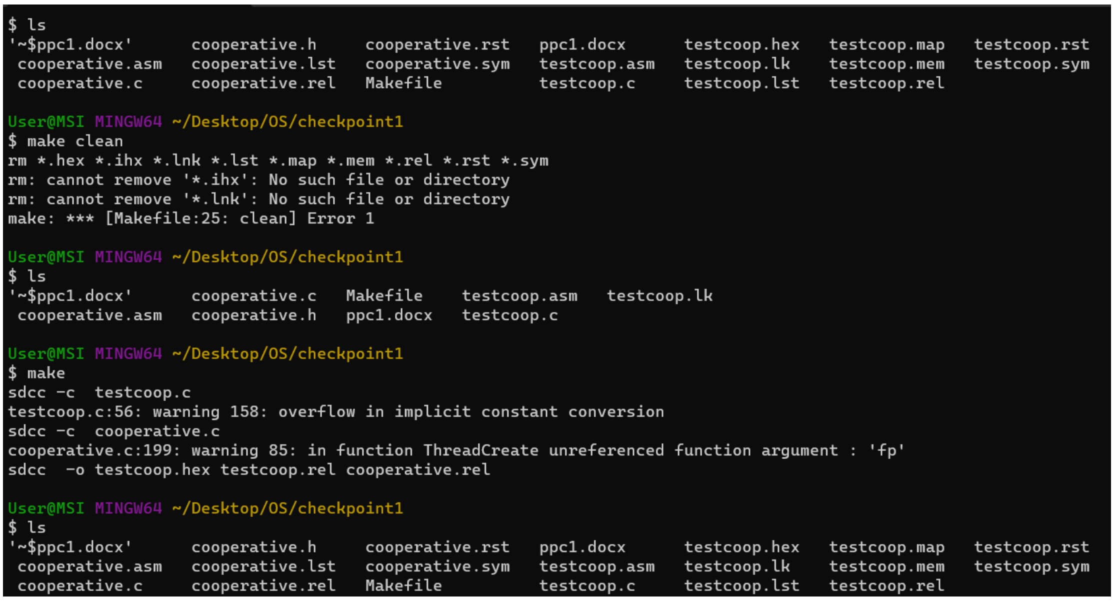
Name: 吳嘉濬 Student ID: 109021115

3.1

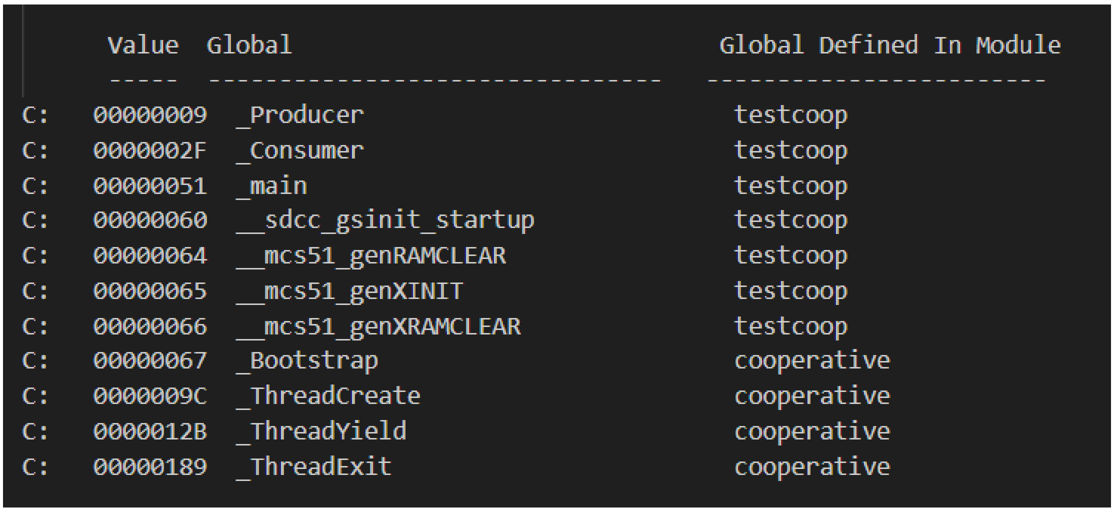


```
$ ls
'~$ppc1.docx'      cooperative.h     cooperative.rst   ppc1.docx        testcoop.hex   testcoop.map   testcoop.rst
 cooperative.asm   cooperative.lst   cooperative.sym   testcoop.asm     testcoop.lk    testcoop.mem   testcoop.sym
 cooperative.c     cooperative.rel   Makefile          testcoop.c       testcoop.lst   testcoop.rel

User@MSI MINGW64 ~/Desktop/OS/checkpoint1
$ make clean
rm *.hex *.ihx *.lnk *.lst *.map *.mem *.rel *.rst *.sym
rm: cannot remove '*.ihx': No such file or directory
rm: cannot remove '*.lnk': No such file or directory
make: *** [Makefile:25: clean] Error 1

User@MSI MINGW64 ~/Desktop/OS/checkpoint1
$ ls
'~$ppc1.docx'      cooperative.c   Makefile    testcoop.asm   testcoop.lk
 cooperative.asm   cooperative.h   ppc1.docx   testcoop.c

User@MSI MINGW64 ~/Desktop/OS/checkpoint1
$ make
sdcc -c  testcoop.c
testcoop.c:56: warning 158: overflow in implicit constant conversion
sdcc -c  cooperative.c
cooperative.c:199: warning 85: in function ThreadCreate unreferenced function argument : 'fp'
sdcc  -o testcoop.hex testcoop.rel cooperative.rel

User@MSI MINGW64 ~/Desktop/OS/checkpoint1
$ ls
'~$ppc1.docx'      cooperative.h     cooperative.rst   ppc1.docx        testcoop.hex   testcoop.map   testcoop.rst
 cooperative.asm   cooperative.lst   cooperative.sym   testcoop.asm     testcoop.lk    testcoop.mem   testcoop.sym
 cooperative.c     cooperative.rel   Makefile          testcoop.c       testcoop.lst   testcoop.rel
```

After compiling, files such as testcoop.map, testcoop.hex… has been generated.

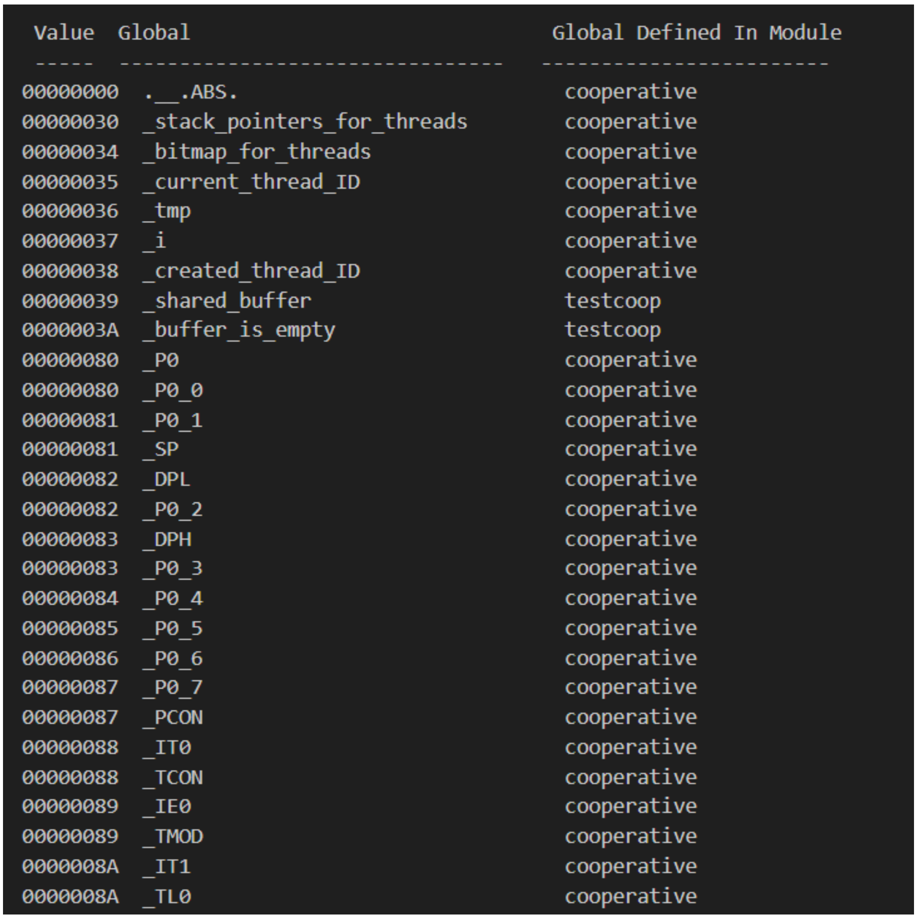
3.2

Some addresses of the functions and variables:



```
      Value  Global                              Global Defined In Module
      -----  --------------------------------   ------------------------
C:   00000009  _Producer                          testcoop
C:   0000002F  _Consumer                          testcoop
C:   00000051  _main                              testcoop
C:   00000060  __sdcc_gsinit_startup              testcoop
C:   00000064  __mcs51_genRAMCLEAR                testcoop
C:   00000065  __mcs51_genXINIT                   testcoop
C:   00000066  __mcs51_genXRAMCLEAR               testcoop
C:   00000067  _Bootstrap                         cooperative
C:   0000009C  _ThreadCreate                      cooperative
C:   0000012B  _ThreadYield                       cooperative
C:   00000189  _ThreadExit                        cooperative
```

We can see address of ThreadCreate(): 0x9C

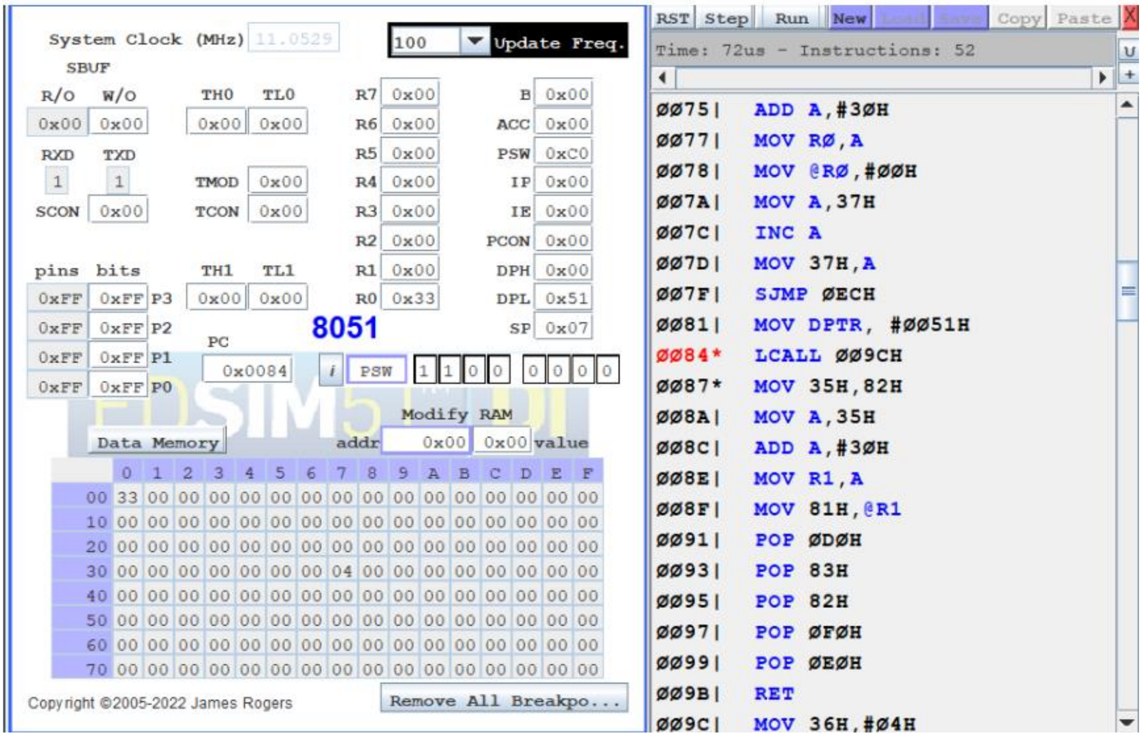


```
 Value  Global                              Global Defined In Module
 -----  --------------------------------   ------------------------
00000000  .__.ABS.                            cooperative
00000030  _stack_pointers_for_threads         cooperative
00000034  _bitmap_for_threads                 cooperative
00000035  _current_thread_ID                  cooperative
00000036  _tmp                                cooperative
00000037  _i                                  cooperative
00000038  _created_thread_ID                  cooperative
00000039  _shared_buffer                      testcoop
0000003A  _buffer_is_empty                    testcoop
00000080  _P0                                 cooperative
00000080  _P0_0                               cooperative
00000081  _P0_1                               cooperative
00000081  _SP                                 cooperative
00000082  _DPL                                cooperative
00000082  _P0_2                               cooperative
00000083  _DPH                                cooperative
00000083  _P0_3                               cooperative
00000084  _P0_4                               cooperative
00000085  _P0_5                               cooperative
00000086  _P0_6                               cooperative
00000087  _P0_7                               cooperative
00000087  _PCON                               cooperative
00000088  _IT0                                cooperative
00000088  _TCON                               cooperative
00000089  _IE0                                cooperative
00000089  _TMOD                               cooperative
0000008A  _IT1                                cooperative
0000008A  _TL0                                cooperative
```

1.

Address of ThreadCreate(): 0x9C

Before the function call of ThreadCreate(main) in Bootstrap(). 0x84



System Clock (MHz) 11.0529

100 Update Freq.

SBUF R/O 0x00 W/O 0x00

RXD 1 TXD 1

SCON 0x00

TH0 0x00 TL0 0x00

TMOD 0x00 TCON 0x00

TH1 0x00 TL1 0x00

pins bits: P3 0xFF 0xFF; P2 0xFF 0xFF; P1 0xFF 0xFF; P0 0xFF 0xFF

PC 0x0084

R7 0x00, R6 0x00, R5 0x00, R4 0x00, R3 0x00, R2 0x00, R1 0x00, R0 0x33

B 0x00, ACC 0x00, PSW 0xC0, IP 0x00, IE 0x00, PCON 0x00, DPH 0x00, DPL 0x51, SP 0x07

PSW 1 1 0 0 0 0 0 0

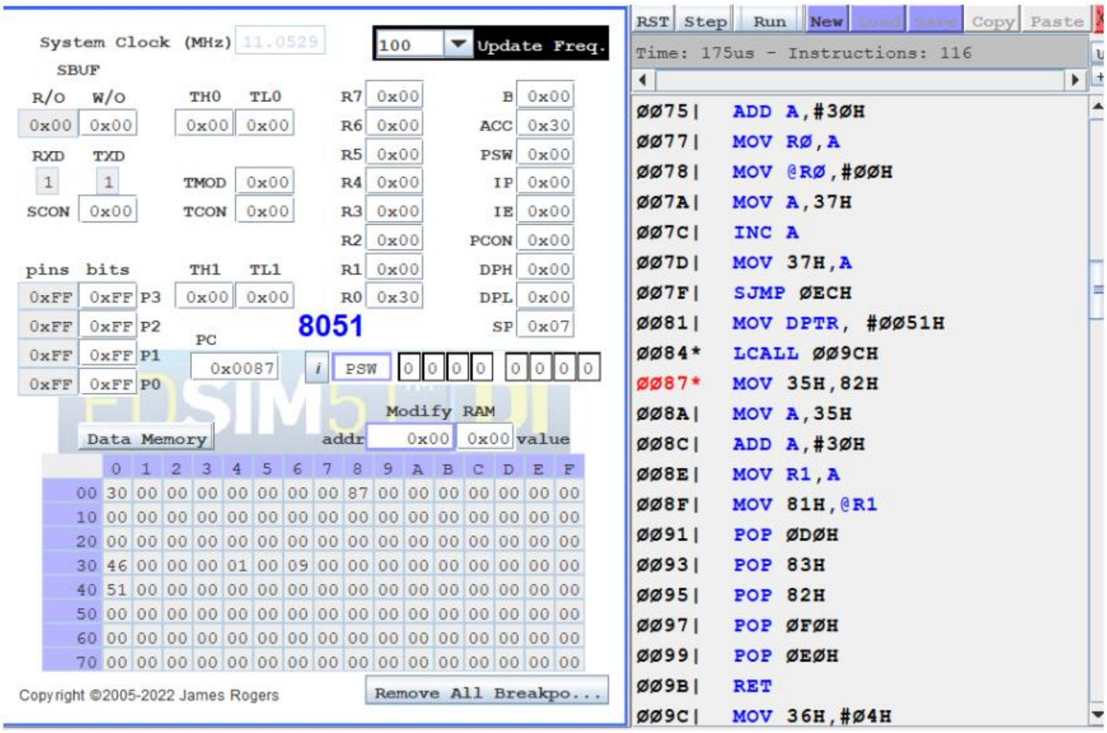
addr 0x00 0x00 value

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	33	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
30	00	00	00	00	00	00	00	04	00	00	00	00	00	00	00	00
40	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Time: 72us - Instructions: 52

```
ØØ75|   ADD A,#3ØH
ØØ77|   MOV RØ,A
ØØ78|   MOV @RØ,#ØØH
ØØ7A|   MOV A,37H
ØØ7C|   INC A
ØØ7D|   MOV 37H,A
ØØ7F|   SJMP ØECH
ØØ81|   MOV DPTR, #ØØ51H
ØØ84*   LCALL ØØ9CH
ØØ87*   MOV 35H,82H
ØØ8A|   MOV A,35H
ØØ8C|   ADD A,#3ØH
ØØ8E|   MOV R1,A
ØØ8F|   MOV 81H,@R1
ØØ91|   POP ØDØH
ØØ93|   POP 83H
ØØ95|   POP 82H
ØØ97|   POP ØFØH
ØØ99|   POP ØEØH
ØØ9B|   RET
ØØ9C|   MOV 36H,#Ø4H
```

After returning from the ThreadCreate function. 0x87



System Clock (MHz) 11.0529
100 Update Freq.
RST Step Run New Load Save Copy Paste
8051
Data Memory
Modify RAM
Remove All Breakpo...
Copyright ©2005-2022 James Rogers

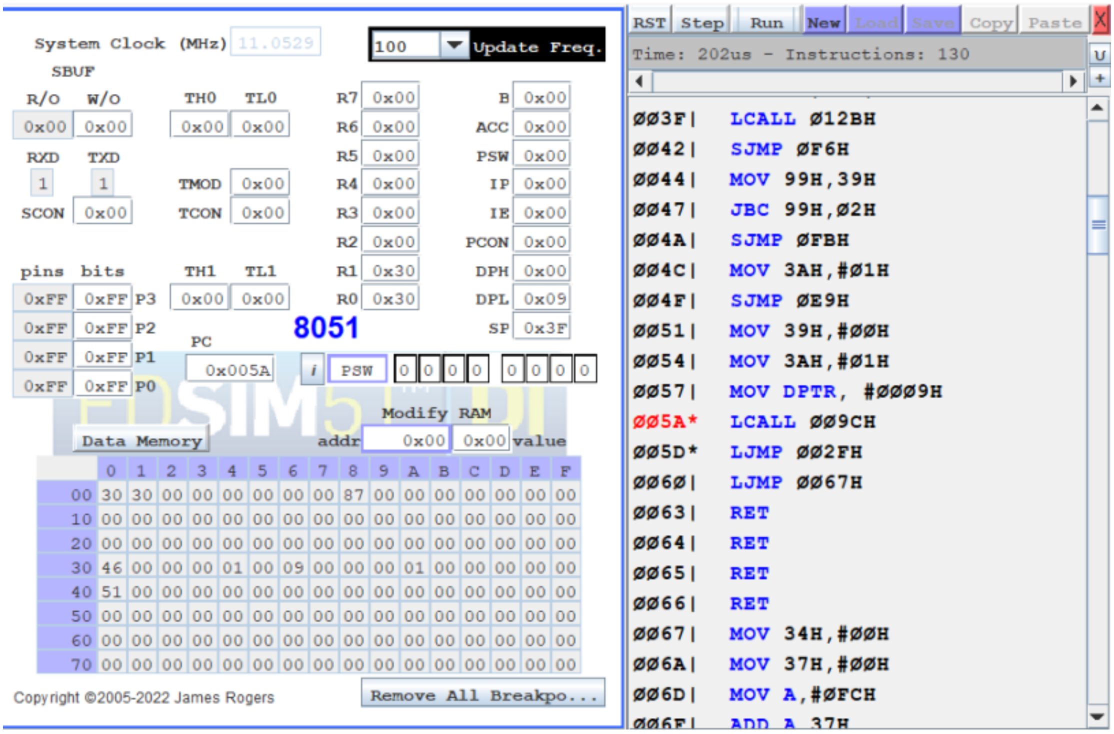
```
SBUF
R/O 0x00   W/O 0x00
RXD 1      TXD 1
SCON 0x00
TH0 0x00   TL0 0x00
TMOD 0x00
TCON 0x00
TH1 0x00   TL1 0x00
pins  bits
0xFF  0xFF P3
0xFF  0xFF P2
0xFF  0xFF P1
0xFF  0xFF P0
R7 0x00    B 0x00
R6 0x00    ACC 0x30
R5 0x00    PSW 0x00
R4 0x00    IP 0x00
R3 0x00    IE 0x00
R2 0x00    PCON 0x00
R1 0x00    DPH 0x00
R0 0x30    DPL 0x00
           SP 0x07
PC 0x0087
PSW 0 0 0 0  0 0 0 0
addr 0x00  0x00 value
```

```
    0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
00 30 00 00 00 00 00 00 00 87 00 00 00 00 00 00 00
10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
30 46 00 00 00 01 00 09 00 00 00 00 00 00 00 00 00
40 51 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
Time: 175us - Instructions: 116
ØØ75|  ADD A,#3ØH
ØØ77|  MOV RØ,A
ØØ78|  MOV @RØ,#ØØH
ØØ7A|  MOV A,37H
ØØ7C|  INC A
ØØ7D|  MOV 37H,A
ØØ7F|  SJMP ØECH
ØØ81|  MOV DPTR, #ØØ51H
ØØ84*  LCALL ØØ9CH
ØØ87*  MOV 35H,82H
ØØ8A|  MOV A,35H
ØØ8C|  ADD A,#3ØH
ØØ8E|  MOV R1,A
ØØ8F|  MOV 81H,@R1
ØØ91|  POP ØDØH
ØØ93|  POP 83H
ØØ95|  POP 82H
ØØ97|  POP ØFØH
ØØ99|  POP ØEØH
ØØ9B|  RET
ØØ9C|  MOV 36H,#Ø4H
```

By the screenshot, we can find out that thread 0 has been created since the bitmap for threads(0x34) has been changed to 1. So in terms of the stack, we could check from address 0x40. Moreover, in 0x30, the value of stack_pointers_for_threads[0] is 0x46(address of the top of the stack for thread 1). So we could know that, during the execution of CreateThread(main), SP has been moved to 0x3F, then 7 variables has been pushed into the stack. DPL, DPH (return addresses to resume the thread), 4 zeros for ACC, B, DPL, DPH, and PSW has been pushed in the stack in order, which corresponds to the values in 0x40, 0x41, 0x42, 0x43, 0x44, 0x45, 0x46. That’s why the stack_pointers_for_threads[0] has the value 0x46.

Before the function call of ThreadCreate(Producer) in main(). 0x5A



Time: 202us - Instructions: 130

SBUF R/O	SBUF W/O	TH0	TL0	R7	B
0x00	0x00	0x00	0x00	0x00	0x00

RXD 1 TXD 1 SCON 0x00 TMOD 0x00 TCON 0x00 TH1 0x00 TL1 0x00

R6 0x00 R5 0x00 R4 0x00 R3 0x00 R2 0x00 R1 0x30 R0 0x30

ACC 0x00 PSW 0x00 IP 0x00 IE 0x00 PCON 0x00 DPH 0x00 DPL 0x09 SP 0x3F

pins	bits	
0xFF	0xFF	P3
0xFF	0xFF	P2
0xFF	0xFF	P1
0xFF	0xFF	P0

PC 0x005A PSW 0 0 0 0 0 0 0 0

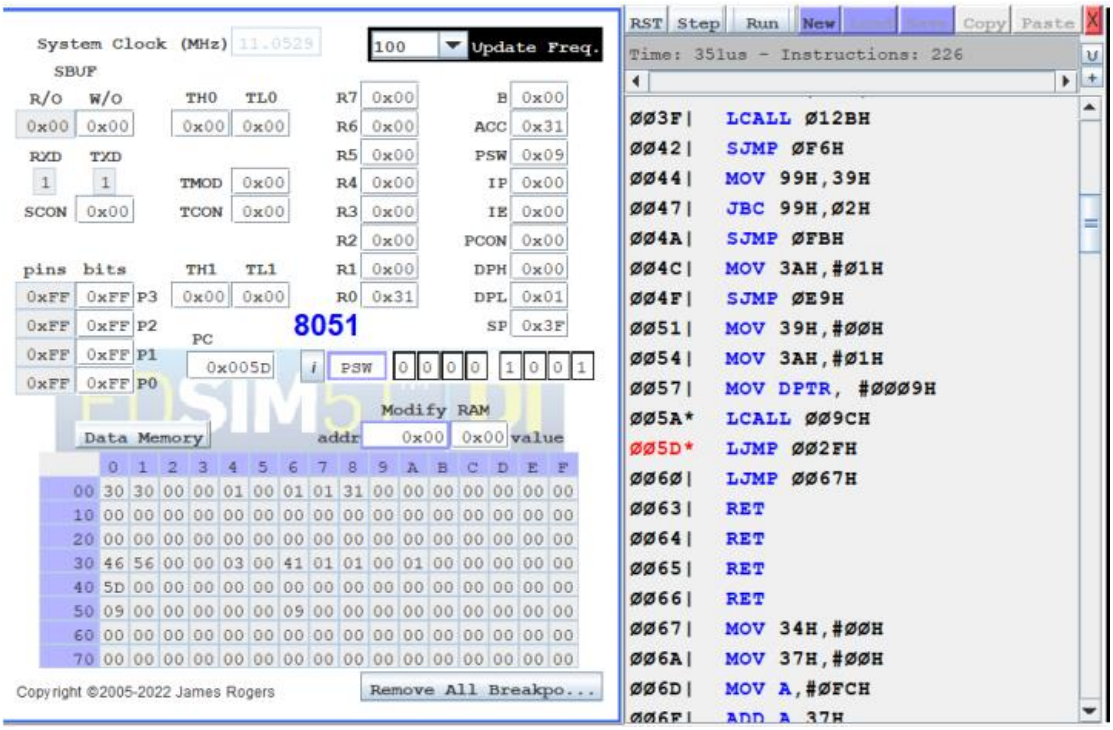
addr 0x00 0x00 value

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	30	30	00	00	00	00	00	00	87	00	00	00	00	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
30	46	00	00	00	01	00	09	00	00	00	01	00	00	00	00	00
40	51	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
50	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Copyright ©2005-2022 James Rogers

```
ØØ3F|  LCALL ØØ12BH
ØØ42|  SJMP ØF6H
ØØ44|  MOV 99H,39H
ØØ47|  JBC 99H,Ø2H
ØØ4A|  SJMP ØFBH
ØØ4C|  MOV 3AH,#Ø1H
ØØ4F|  SJMP ØE9H
ØØ51|  MOV 39H,#ØØH
ØØ54|  MOV 3AH,#Ø1H
ØØ57|  MOV DPTR, #ØØØ9H
ØØ5A*  LCALL ØØ9CH
ØØ5D*  LJMP ØØ2FH
ØØ6Ø|  LJMP ØØ67H
ØØ63|  RET
ØØ64|  RET
ØØ65|  RET
ØØ66|  RET
ØØ67|  MOV 34H,#ØØH
ØØ6A|  MOV 37H,#ØØH
ØØ6D|  MOV A,#ØFCH
ØØ6F|  ADD A,37H
```

After returning from the ThreadCreate function. 0x5D



Time: 351us - Instructions: 226

SBUF R/O	SBUF W/O	TH0	TL0	R7	B
0x00	0x00	0x00	0x00	0x00	0x00

RXD 1 TXD 1 SCON 0x00 TMOD 0x00 TCON 0x00 TH1 0x00 TL1 0x00

R6 0x00 R5 0x00 R4 0x00 R3 0x00 R2 0x00 R1 0x00 R0 0x31

ACC 0x31 PSW 0x09 IP 0x00 IE 0x00 PCON 0x00 DPH 0x00 DPL 0x01 SP 0x3F

pins	bits	
0xFF	0xFF	P3
0xFF	0xFF	P2
0xFF	0xFF	P1
0xFF	0xFF	P0

PC 0x005D PSW 0 0 0 0 1 0 0 1

addr 0x00 0x00 value

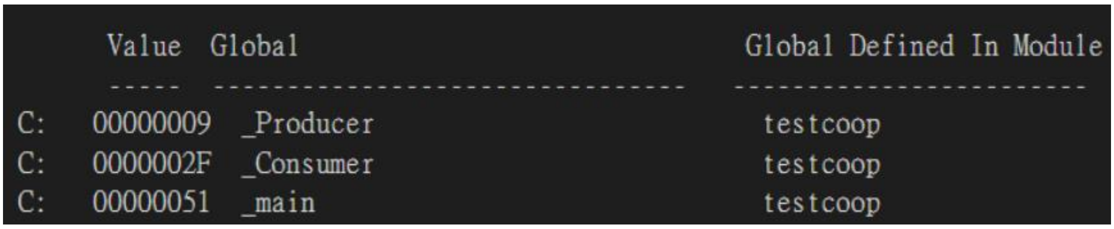
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	30	30	00	00	01	00	01	01	31	00	00	00	00	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
30	46	56	00	00	03	00	41	01	01	00	01	00	00	00	00	00
40	5D	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
50	09	00	00	00	00	00	09	00	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Copyright ©2005-2022 James Rogers

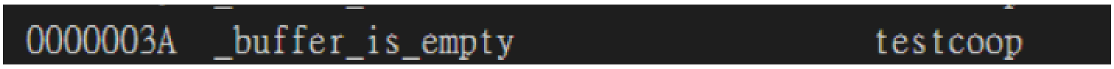
```
ØØ3F|  LCALL ØØ12BH
ØØ42|  SJMP ØF6H
ØØ44|  MOV 99H,39H
ØØ47|  JBC 99H,Ø2H
ØØ4A|  SJMP ØFBH
ØØ4C|  MOV 3AH,#Ø1H
ØØ4F|  SJMP ØE9H
ØØ51|  MOV 39H,#ØØH
ØØ54|  MOV 3AH,#Ø1H
ØØ57|  MOV DPTR, #ØØØ9H
ØØ5A*  LCALL ØØ9CH
ØØ5D*  LJMP ØØ2FH
ØØ6Ø|  LJMP ØØ67H
ØØ63|  RET
ØØ64|  RET
ØØ65|  RET
ØØ66|  RET
ØØ67|  MOV 34H,#ØØH
ØØ6A|  MOV 37H,#ØØH
ØØ6D|  MOV A,#ØFCH
ØØ6F|  ADD A,37H
```

bitmap_for_threads has been updated to 3, means that thread 1 has been created. Similar to the ThreadCreate(main) function explained above, stack_pointers_for_threads[1] gets the value 0x56(address of the top of the stack for thread 1), means that 7 variables has been pushed into the stack for thread 1. DPL, DPH, 4 zeros for ACC, B, DPL, DPH, and PSW are in address 0x50, 0x51, 0x52, 0x53, 0x54, 0x55, 0x56 respectively.

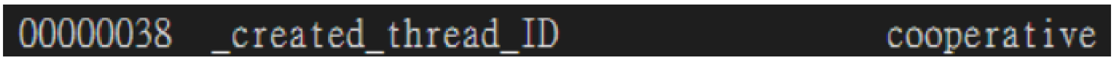
2. We could know that Producer() is running for several reasons:



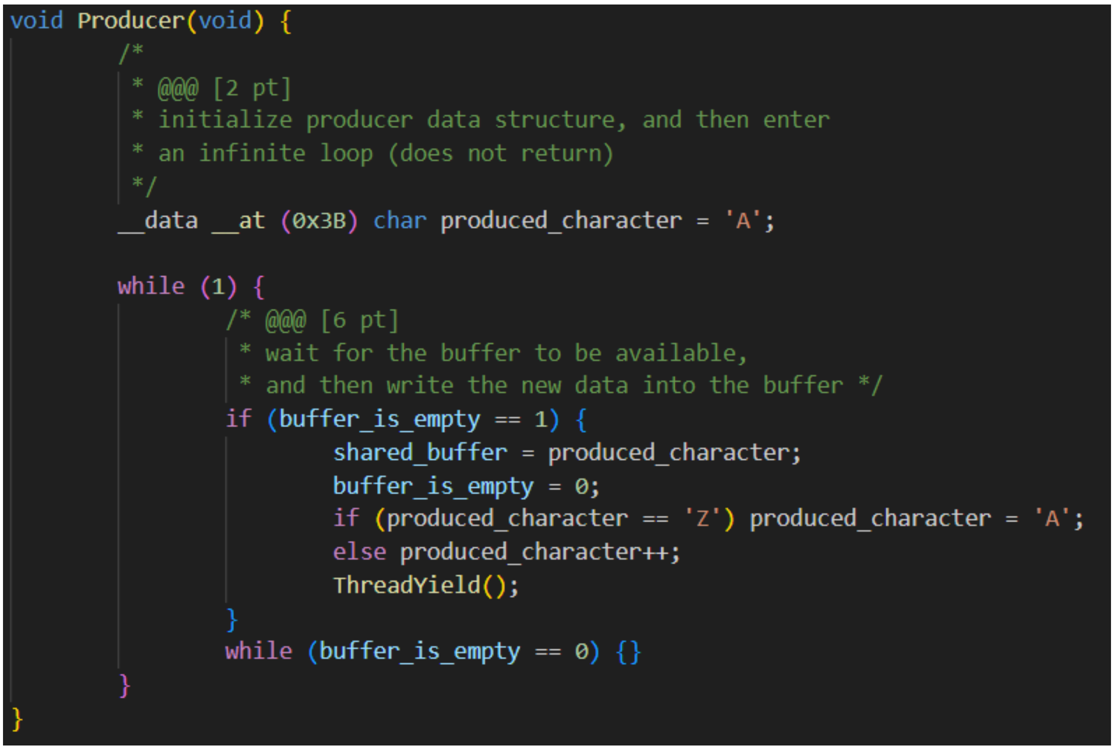
```
     Value   Global                             Global Defined In Module
     -----   --------------------------------   ------------------------
C:   00000009  _Producer                           testcoop
C:   0000002F  _Consumer                           testcoop
C:   00000051  _main                               testcoop
```



```
 0000003A  _buffer_is_empty                    testcoop
```



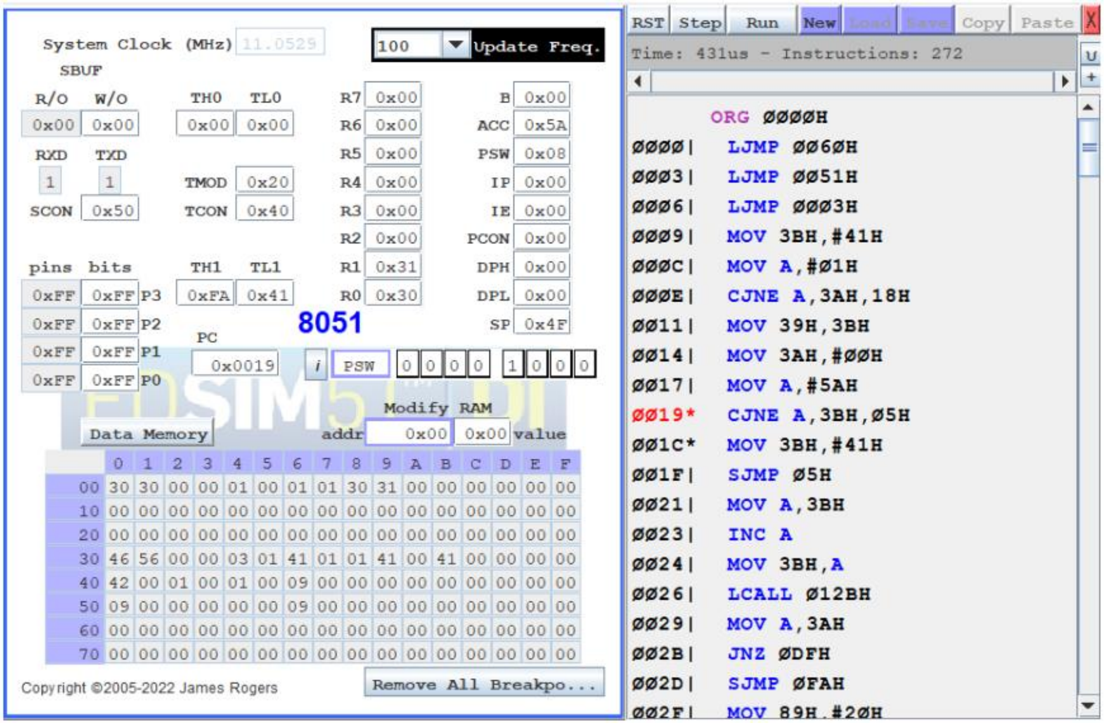
```
00000038  _created_thread_ID                   cooperative
```



```
void Producer(void) {
        /*
         * @@@ [2 pt]
         * initialize producer data structure, and then enter
         * an infinite loop (does not return)
         */
        __data __at (0x3B) char produced_character = 'A';

        while (1) {
                /* @@@ [6 pt]
                 * wait for the buffer to be available,
                 * and then write the new data into the buffer */
                if (buffer_is_empty == 1) {
                        shared_buffer = produced_character;
                        buffer_is_empty = 0;
                        if (produced_character == 'Z') produced_character = 'A';
                        else produced_character++;
                        ThreadYield();
                }
                while (buffer_is_empty == 0) {}
        }
}
```

Variable “produced_character” whose address is at 0x3B.

0x19: value of “produced_character”(at address 0x3B) is 0x41.



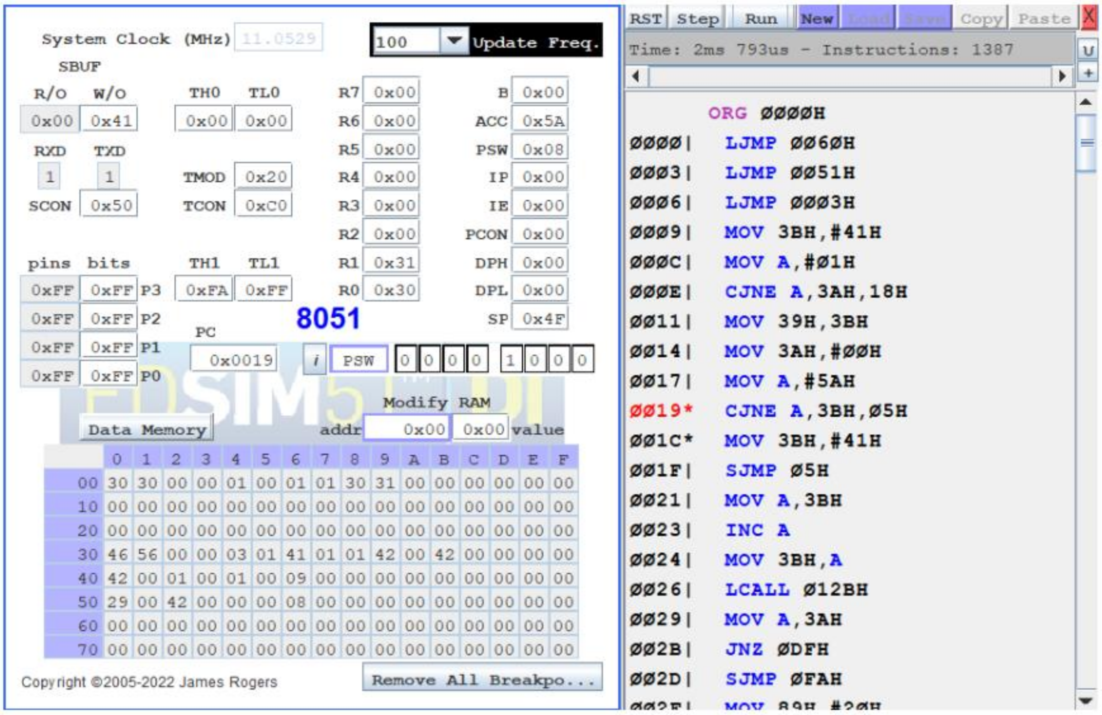
```
System Clock (MHz) 11.0529        100  Update Freq.
SBUF
R/O   W/O      TH0   TL0      R7 0x00        B 0x00
0x00  0x00     0x00  0x00     R6 0x00      ACC 0x5A
RXD   TXD                     R5 0x00      PSW 0x08
1     1        TMOD  0x20     R4 0x00       IP 0x00
SCON  0x50     TCON  0x40     R3 0x00       IE 0x00
                              R2 0x00     PCON 0x00
pins  bits     TH1   TL1      R1 0x31      DPH 0x00
0xFF  0xFF P3  0xFA  0x41     R0 0x30      DPL 0x00
0xFF  0xFF P2                 8051          SP 0x4F
0xFF  0xFF P1  PC
0xFF  0xFF P0  0x0019      i  PSW  0 0 0 0  1 0 0 0
                                  Modify RAM
               addr 0x00  0x00 value
```

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	30	30	00	00	01	00	01	01	30	31	00	00	00	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
30	46	56	00	00	03	01	41	01	01	41	00	41	00	00	00	00
40	42	00	01	00	01	00	09	00	00	00	00	00	00	00	00	00
50	09	00	00	00	00	00	09	00	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Copyright ©2005-2022 James Rogers

```
Time: 431us - Instructions: 272

      ORG  ØØØØH
ØØØØ|  LJMP  ØØ6ØH
ØØØ3|  LJMP  ØØ51H
ØØØ6|  LJMP  ØØØ3H
ØØØ9|  MOV  3BH,#41H
ØØØC|  MOV  A,#Ø1H
ØØØE|  CJNE  A,3AH,18H
ØØ11|  MOV  39H,3BH
ØØ14|  MOV  3AH,#ØØH
ØØ17|  MOV  A,#5AH
ØØ19*  CJNE  A,3BH,Ø5H
ØØ1C*  MOV  3BH,#41H
ØØ1F|  SJMP  Ø5H
ØØ21|  MOV  A,3BH
ØØ23|  INC  A
ØØ24|  MOV  3BH,A
ØØ26|  LCALL  Ø12BH
ØØ29|  MOV  A,3AH
ØØ2B|  JNZ  ØDFH
ØØ2D|  SJMP  ØFAH
ØØ2F|  MOV  89H,#2ØH
```

0x19: value of “produced_character”(at address 0x3B) gets changed to 0x42.



```
System Clock (MHz) 11.0529        100  Update Freq.
SBUF
R/O   W/O      TH0   TL0      R7 0x00        B 0x00
0x00  0x41     0x00  0x00     R6 0x00      ACC 0x5A
RXD   TXD                     R5 0x00      PSW 0x08
1     1        TMOD  0x20     R4 0x00       IP 0x00
SCON  0x50     TCON  0xC0     R3 0x00       IE 0x00
                              R2 0x00     PCON 0x00
pins  bits     TH1   TL1      R1 0x31      DPH 0x00
0xFF  0xFF P3  0xFA  0xFF     R0 0x30      DPL 0x00
0xFF  0xFF P2                 8051          SP 0x4F
0xFF  0xFF P1  PC
0xFF  0xFF P0  0x0019      i  PSW  0 0 0 0  1 0 0 0
                                  Modify RAM
               addr 0x00  0x00 value
```

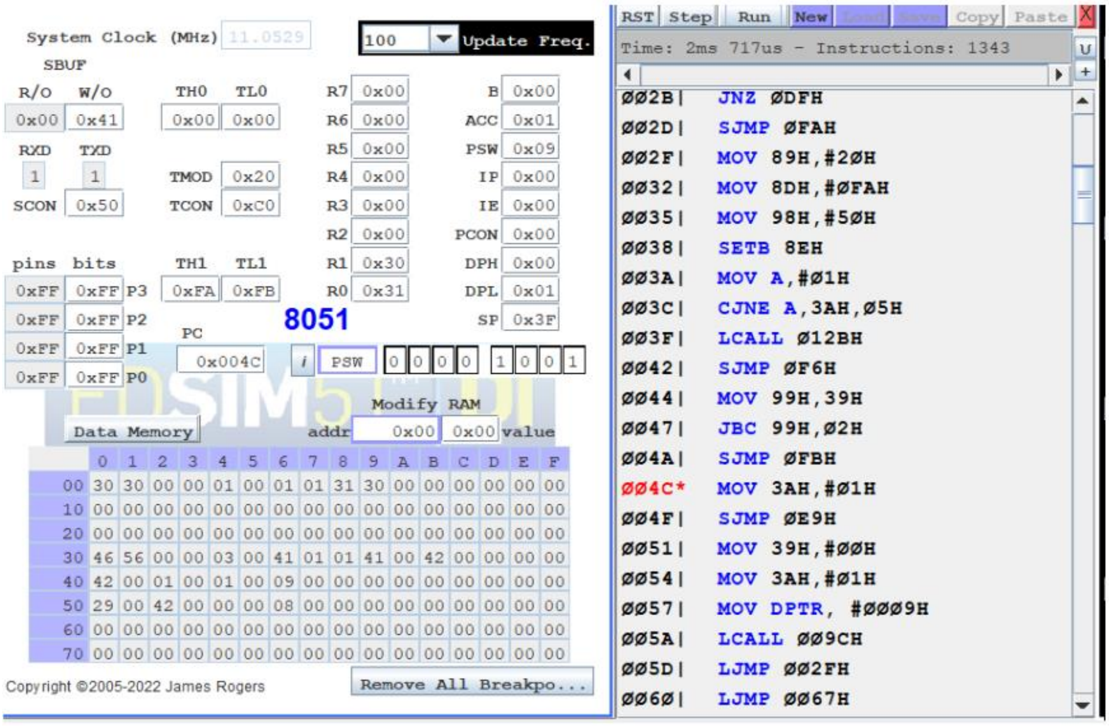
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	30	30	00	00	01	00	01	01	30	31	00	00	00	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
30	46	56	00	00	03	01	41	01	01	42	00	42	00	00	00	00
40	42	00	01	00	01	00	09	00	00	00	00	00	00	00	00	00
50	29	00	42	00	00	00	08	00	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Copyright ©2005-2022 James Rogers

```
Time: 2ms 793us - Instructions: 1387

      ORG  ØØØØH
ØØØØ|  LJMP  ØØ6ØH
ØØØ3|  LJMP  ØØ51H
ØØØ6|  LJMP  ØØØ3H
ØØØ9|  MOV  3BH,#41H
ØØØC|  MOV  A,#Ø1H
ØØØE|  CJNE  A,3AH,18H
ØØ11|  MOV  39H,3BH
ØØ14|  MOV  3AH,#ØØH
ØØ17|  MOV  A,#5AH
ØØ19*  CJNE  A,3BH,Ø5H
ØØ1C*  MOV  3BH,#41H
ØØ1F|  SJMP  Ø5H
ØØ21|  MOV  A,3BH
ØØ23|  INC  A
ØØ24|  MOV  3BH,A
ØØ26|  LCALL  Ø12BH
ØØ29|  MOV  A,3AH
ØØ2B|  JNZ  ØDFH
ØØ2D|  SJMP  ØFAH
ØØ2F|  MOV  89H,#2ØH
```

First, we could recognize that Producer() is currently running by the address of the function since the address of Producer() starts from 0x09. Second, we could check the value of 0x35 in memory, which is the ID of the currently running thread. We find out that the currently running thread is 1, which is correct. Third, the value of the variable “produced_character” gets changed only in the function Producer(). So if we saw the value in memory 0x3B has changed, we could know that the current thread is running the Producer() function. For example, the screenshots above shows that the value of 0x3B has changed from 0x41 to 0x42, which means that function Producer() is running now.

3. We could know that Consumer() is running for several reasons:
0x4C: variable “buffer_is_empty” is 0



System Clock (MHz) 11.0529 | 100 Update Freq.

SBUF R/O 0x00 W/O 0x41 | RXD 1 TXD 1 | SCON 0x50

TH0 0x00 TL0 0x00 | TMOD 0x20 | TCON 0xC0 | TH1 0xFA TL1 0xFB

pins bits: P3 0xFF 0xFF | P2 0xFF 0xFF | P1 0xFF 0xFF | P0 0xFF 0xFF

R7 0x00 | R6 0x00 | R5 0x00 | R4 0x00 | R3 0x00 | R2 0x00 | R1 0x30 | R0 0x31

B 0x00 | ACC 0x01 | PSW 0x09 | IP 0x00 | IE 0x00 | PCON 0x00 | DPH 0x00 | DPL 0x01 | SP 0x3F

8051 | PC 0x004C | PSW 0 0 0 0 1 0 0 1

Modify RAM addr 0x00 0x00 value

Data Memory

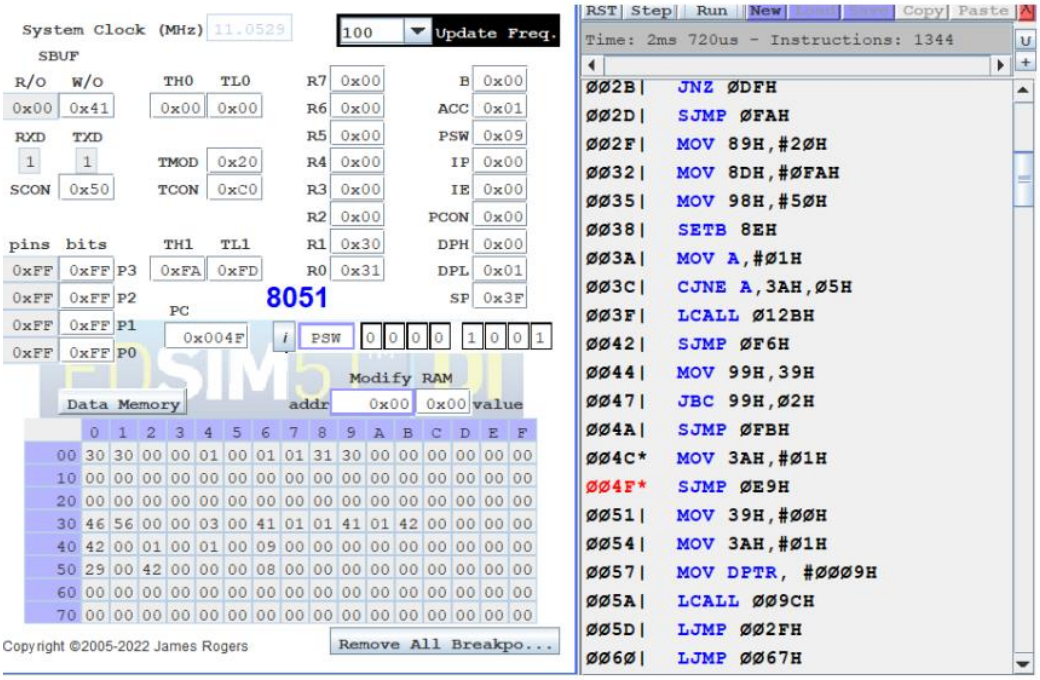
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	30	30	00	00	01	00	01	01	31	30	00	00	00	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
30	46	56	00	00	03	00	41	01	01	41	00	42	00	00	00	00
40	42	00	01	00	01	00	09	00	00	00	00	00	00	00	00	00
50	29	00	42	00	00	00	08	00	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Copyright ©2005-2022 James Rogers

Time: 2ms 717us - Instructions: 1343

```
002B|  JNZ 0DFH
002D|  SJMP 0FAH
002F|  MOV 89H,#20H
0032|  MOV 8DH,#0FAH
0035|  MOV 98H,#50H
0038|  SETB 8EH
003A|  MOV A,#01H
003C|  CJNE A,3AH,05H
003F|  LCALL 012BH
0042|  SJMP 0F6H
0044|  MOV 99H,39H
0047|  JBC 99H,02H
004A|  SJMP 0FBH
004C*  MOV 3AH,#01H
004F|  SJMP 0E9H
0051|  MOV 39H,#00H
0054|  MOV 3AH,#01H
0057|  MOV DPTR, #0009H
005A|  LCALL 009CH
005D|  LJMP 002FH
0060|  LJMP 0067H
```

0x4F: variable “buffer_is_empty” gets changed to 1



System Clock (MHz) 11.0529 | 100 Update Freq.

SBUF R/O 0x00 W/O 0x41 | RXD 1 TXD 1 | SCON 0x50

TH0 0x00 TL0 0x00 | TMOD 0x20 | TCON 0xC0 | TH1 0xFA TL1 0xFD

pins bits: P3 0xFF 0xFF | P2 0xFF 0xFF | P1 0xFF 0xFF | P0 0xFF 0xFF

R7 0x00 | R6 0x00 | R5 0x00 | R4 0x00 | R3 0x00 | R2 0x00 | R1 0x30 | R0 0x31

B 0x00 | ACC 0x01 | PSW 0x09 | IP 0x00 | IE 0x00 | PCON 0x00 | DPH 0x00 | DPL 0x01 | SP 0x3F

8051 | PC 0x004F | PSW 0 0 0 0 1 0 0 1

Modify RAM addr 0x00 0x00 value

Data Memory

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	30	30	00	00	01	00	01	01	31	30	00	00	00	00	00	00
10	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
30	46	56	00	00	03	00	41	01	01	41	01	42	00	00	00	00
40	42	00	01	00	01	00	09	00	00	00	00	00	00	00	00	00
50	29	00	42	00	00	00	08	00	00	00	00	00	00	00	00	00
60	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
70	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Copyright ©2005-2022 James Rogers

Time: 2ms 720us - Instructions: 1344

```
002B|  JNZ 0DFH
002D|  SJMP 0FAH
002F|  MOV 89H,#20H
0032|  MOV 8DH,#0FAH
0035|  MOV 98H,#50H
0038|  SETB 8EH
003A|  MOV A,#01H
003C|  CJNE A,3AH,05H
003F|  LCALL 012BH
0042|  SJMP 0F6H
0044|  MOV 99H,39H
0047|  JBC 99H,02H
004A|  SJMP 0FBH
004C*  MOV 3AH,#01H
004F*  SJMP 0E9H
0051|  MOV 39H,#00H
0054|  MOV 3AH,#01H
0057|  MOV DPTR, #0009H
005A|  LCALL 009CH
005D|  LJMP 002FH
0060|  LJMP 0067H
```

First, we could recognize by the address of the function since Consumer starts from address 0x3F. Second, we could check the value of 0x35, which is the ID of the current running thread. We could find out that the current running thread is 0, which is correct. Third, the variable “buffer_is_empty” gets changed to 1 while the variable gets changed to 0 at the Producer() part. So when swapping between the two threads(Consumer and Producer), if we saw the value in address 0x3A gets changed from 0 to 1(the screenshots above), then we could know that the thread of Consumer() is running now.