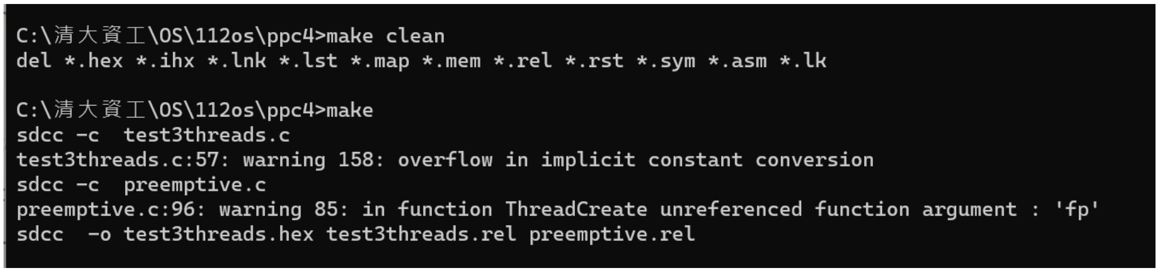
Checkpoint4

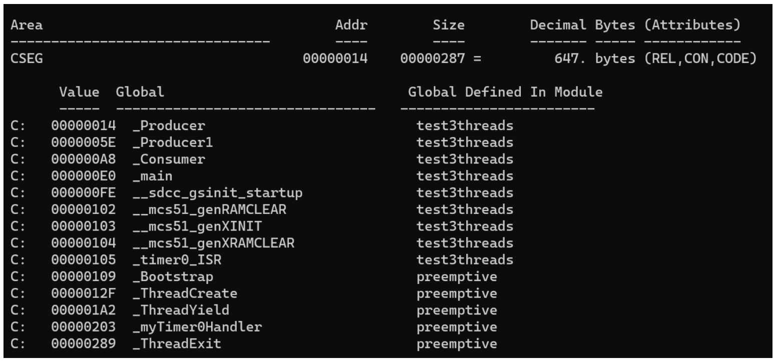
1.Typescript for compilation

I chose to implement on the Windows operating system, so I cannot use the 'rm' command. Here, I opted to use the 'del' command to achieve the same functionality.

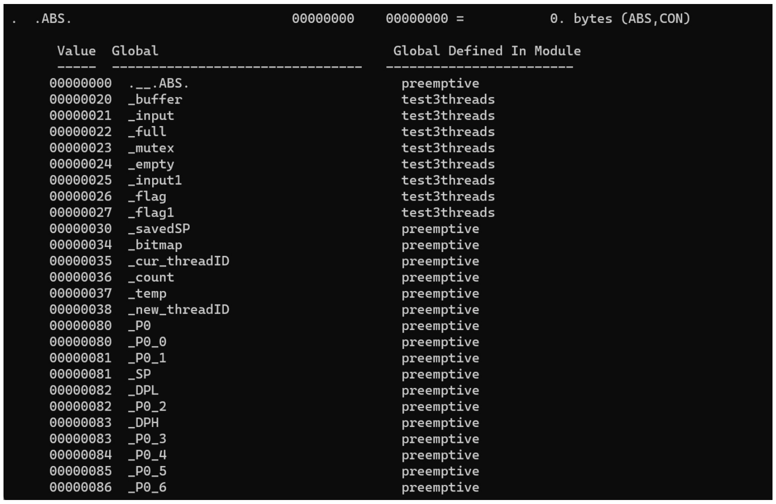


```
C:\清大資工\OS\112os\ppc4>make clean
del *.hex *.ihx *.lnk *.lst *.map *.mem *.rel *.rst *.sym *.asm *.lk

C:\清大資工\OS\112os\ppc4>make
sdcc -c  test3threads.c
test3threads.c:57: warning 158: overflow in implicit constant conversion
sdcc -c  preemptive.c
preemptive.c:96: warning 85: in function ThreadCreate unreferenced function argument : 'fp'
sdcc  -o test3threads.hex test3threads.rel preemptive.rel
```

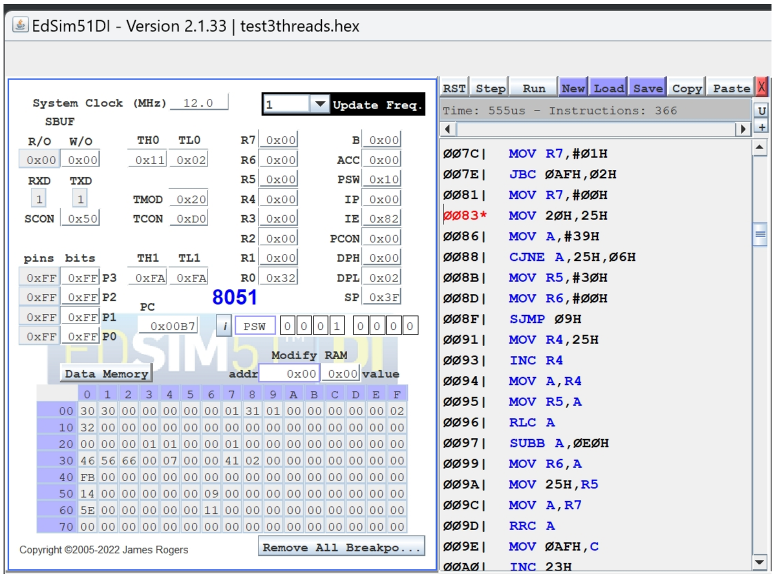
2-1.Take screenshots when the Producer1 and Producer2 running and show semaphore changes.



```
Area                                    Addr        Size        Decimal Bytes (Attributes)
--------------------------------        ----        ----        ------- ----- ------------
CSEG                                00000014    00000287 =        647. bytes (REL,CON,CODE)

      Value  Global                              Global Defined In Module
      -----  --------------------------------   ------------------------
C:   00000014  _Producer                           test3threads
C:   0000005E  _Producer1                          test3threads
C:   000000A8  _Consumer                           test3threads
C:   000000E0  _main                               test3threads
C:   000000FE  __sdcc_gsinit_startup               test3threads
C:   00000102  __mcs51_genRAMCLEAR                 test3threads
C:   00000103  __mcs51_genXINIT                    test3threads
C:   00000104  __mcs51_genXRAMCLEAR                test3threads
C:   00000105  _timer0_ISR                         test3threads
C:   00000109  _Bootstrap                          preemptive
C:   0000012F  _ThreadCreate                       preemptive
C:   000001A2  _ThreadYield                        preemptive
C:   00000203  _myTimer0Handler                    preemptive
C:   00000289  _ThreadExit                         preemptive
```



```
.  .ABS.                              00000000    00000000 =           0. bytes (ABS,CON)

      Value  Global                              Global Defined In Module
      -----  --------------------------------   ------------------------
      00000000  .__.ABS.                            preemptive
      00000020  _buffer                             test3threads
      00000021  _input                              test3threads
      00000022  _full                               test3threads
      00000023  _mutex                              test3threads
      00000024  _empty                              test3threads
      00000025  _input1                             test3threads
      00000026  _flag                               test3threads
      00000027  _flag1                              test3threads
      00000030  _savedSP                            preemptive
      00000034  _bitmap                             preemptive
      00000035  _cur_threadID                       preemptive
      00000036  _count                              preemptive
      00000037  _temp                               preemptive
      00000038  _new_threadID                       preemptive
      00000080  _P0                                 preemptive
      00000080  _P0_0                               preemptive
      00000081  _P0_1                               preemptive
      00000081  _SP                                 preemptive
      00000082  _DPL                                preemptive
      00000082  _P0_2                               preemptive
      00000083  _DPH                                preemptive
      00000083  _P0_3                               preemptive
      00000084  _P0_4                               preemptive
      00000085  _P0_5                               preemptive
      00000086  _P0_6                               preemptive
```



EdSim51DI - Version 2.1.33 | test3threads.hex

System Clock (MHz) 12.0

1 Update Freq.

SBUF R/O 0x00 W/O 0x00

RXD 1 TXD 1

SCON 0x50

TH0 0x11 TL0 0x02

TMOD 0x20

TCON 0xD0

TH1 0xFA TL1 0xFA

R7 0x00
R6 0x00
R5 0x00
R4 0x00
R3 0x00
R2 0x00
R1 0x00
R0 0x32

B 0x00
ACC 0x00
PSW 0x10
IP 0x00
IE 0x82
PCON 0x00
DPH 0x00
DPL 0x02
SP 0x3F

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

8051

PC 0x00B7

PSW 0 0 0 1 0 0 0 0

Modify RAM addr 0x00 0x00 value

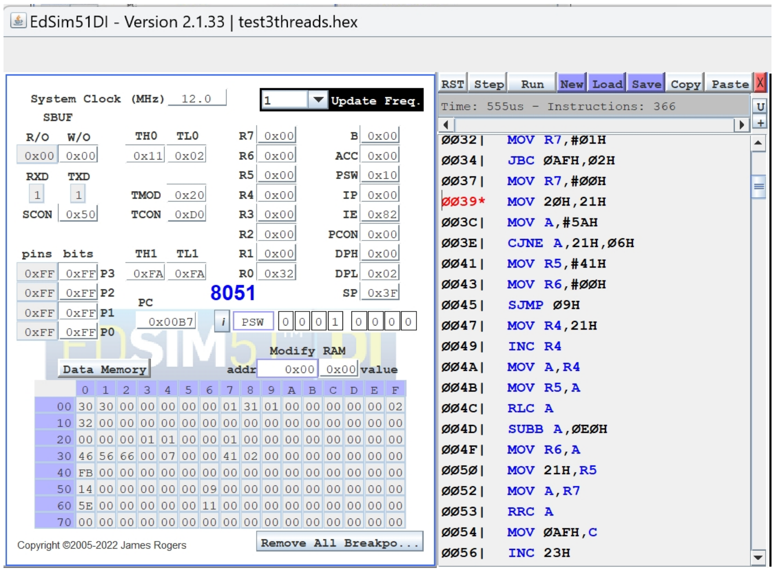
Data Memory

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	30	30	00	00	00	00	00	01	31	01	00	00	00	00	00	02
10	32	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	01	01	00	00	01	00	00	00	00	00	00	00	00
30	46	56	66	00	07	00	00	41	02	00	00	00	00	00	00	00
40	FB	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
50	14	00	00	00	00	00	09	00	00	00	00	00	00	00	00	00
60	5E	00	00	00	00	00	11	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: 555us - Instructions: 366

```
ØØ7C|   MOV R7,#Ø1H
ØØ7E|   JBC ØAFH,Ø2H
ØØ81|   MOV R7,#ØØH
ØØ83*   MOV 2ØH,25H
ØØ86|   MOV A,#39H
ØØ88|   CJNE A,25H,Ø6H
ØØ8B|   MOV R5,#3ØH
ØØ8D|   MOV R6,#ØØH
ØØ8F|   SJMP Ø9H
ØØ91|   MOV R4,25H
ØØ93|   INC R4
ØØ94|   MOV A,R4
ØØ95|   MOV R5,A
ØØ96|   RLC A
ØØ97|   SUBB A,ØEØH
ØØ99|   MOV R6,A
ØØ9A|   MOV 25H,R5
ØØ9C|   MOV A,R7
ØØ9D|   RRC A
ØØ9E|   MOV ØAFH,C
ØØAØ|   INC 23H
```



EdSim51DI - Version 2.1.33 | test3threads.hex

System Clock (MHz) 12.0

1 Update Freq.

SBUF R/O 0x00 W/O 0x00

RXD 1 TXD 1

SCON 0x50

TH0 0x11 TL0 0x02

TMOD 0x20

TCON 0xD0

TH1 0xFA TL1 0xFA

R7 0x00
R6 0x00
R5 0x00
R4 0x00
R3 0x00
R2 0x00
R1 0x00
R0 0x32

B 0x00
ACC 0x00
PSW 0x10
IP 0x00
IE 0x82
PCON 0x00
DPH 0x00
DPL 0x02
SP 0x3F

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

8051

PC 0x00B7

PSW 0 0 0 1 0 0 0 0

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	00	00	00	01	31	01	00	00	00	00	00	02
10	32	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	01	01	00	00	01	00	00	00	00	00	00	00	00
30	46	56	66	00	07	00	00	41	02	00	00	00	00	00	00	00
40	FB	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
50	14	00	00	00	00	00	09	00	00	00	00	00	00	00	00	00
60	5E	00	00	00	00	00	11	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: 555us - Instructions: 366

```
ØØ32|   MOV R7,#Ø1H
ØØ34|   JBC ØAFH,Ø2H
ØØ37|   MOV R7,#ØØH
ØØ39*   MOV 2ØH,21H
ØØ3C|   MOV A,#5AH
ØØ3E|   CJNE A,21H,Ø6H
ØØ41|   MOV R5,#41H
ØØ43|   MOV R6,#ØØH
ØØ45|   SJMP Ø9H
ØØ47|   MOV R4,21H
ØØ49|   INC R4
ØØ4A|   MOV A,R4
ØØ4B|   MOV R5,A
ØØ4C|   RLC A
ØØ4D|   SUBB A,ØEØH
ØØ4F|   MOV R6,A
ØØ5Ø|   MOV 21H,R5
ØØ52|   MOV A,R7
ØØ53|   RRC A
ØØ54|   MOV ØAFH,C
ØØ56|   INC 23H
```

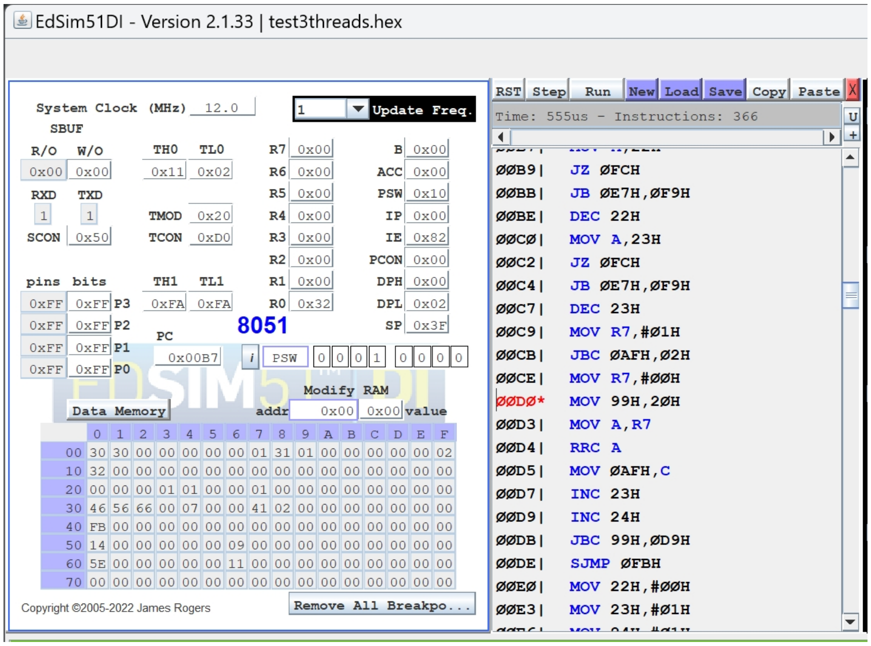
1.At 0083 , executing MOV 20H,25H in the producer(buffer←_input1), the values of semaphores(full,empty,mutex) (at 0x22, 0x24 ,0x23) is changed (0,0,0).

2.Because the producer is in the critical section,and hasn't produced an input. Also, it hasn't finished producing ,so the flag (26H)and flag1(27H) are 0. After it finish, it will set flag1(27H) to 1

1-1.At 0039 , executing MOV 20H,21H in the producer1(buffer←_input), the values of semaphores(full,empty,mutex) (at 0x22, 0x24 ,0x23) is changed (0,0,0).

2-1.Because the producer1 is in the critical section,and hasn't produced an input. Also, it hasn't finished producing ,so the flag (26H)and flag1(27H) are 0. After it finish, it will set flag(26H) to 1..

2-2.Take screenshots when the Consumer is running and show semaphore changes



EdSim51DI - Version 2.1.33 | test3threads.hex

Time: 555us - Instructions: 366

System Clock (MHz) 12.0

Update Freq. 1

SBUF R/O	SBUF W/O	TH0	TL0	TH1	TL1
0x00	0x00	0x11	0x02	0xFA	0xFA

RXD 1, TXD 1, SCON 0x50, TMOD 0x20, TCON 0xD0

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

R7	0x00	B	0x00
R6	0x00	ACC	0x00
R5	0x00	PSW	0x10
R4	0x00	IP	0x00
R3	0x00	IE	0x82
R2	0x00	PCON	0x00
R1	0x00	DPH	0x00
R0	0x32	DPL	0x02
		SP	0x3F

PC 0x00B7

PSW 0 0 0 1 0 0 0 0

Modify RAM addr 0x00 value 0x00

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	30	30	00	00	00	00	00	01	31	01	00	00	00	00	00	02
10	32	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
20	00	00	00	01	01	00	00	01	00	00	00	00	00	00	00	00
30	46	56	66	00	07	00	00	41	02	00	00	00	00	00	00	00
40	FB	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
50	14	00	00	00	00	00	09	00	00	00	00	00	00	00	00	00
60	5E	00	00	00	00	00	11	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

```
ØØB9|   JZ ØFCH
ØØBB|   JB ØE7H,ØF9H
ØØBE|   DEC 22H
ØØCØ|   MOV A,23H
ØØC2|   JZ ØFCH
ØØC4|   JB ØE7H,ØF9H
ØØC7|   DEC 23H
ØØC9|   MOV R7,#Ø1H
ØØCB|   JBC ØAFH,Ø2H
ØØCE|   MOV R7,#ØØH
ØØDØ*   MOV 99H,2ØH
ØØD3|   MOV A,R7
ØØD4|   RRC A
ØØD5|   MOV ØAFH,C
ØØD7|   INC 23H
ØØD9|   INC 24H
ØØDB|   JBC 99H,ØD9H
ØØDE|   SJMP ØFBH
ØØEØ|   MOV 22H,#ØØH
ØØE3|   MOV 23H,#Ø1H
```

Copyright ©2005-2022 James Rogers

1.At 00D0,executing MOV 99H, 20H in the consumer(SBUF←buffer), the values of semaphores(full,empty,mutex)(at 0x22, 0x24 ,0x23) is changed (0,0,0).

2.Because the consumer is in the critical section,and hasn't consumed an input. Besides, one of the semaphore flag(26H) or flag1(27H) is changed from 0 to 1 and the other remains 0. The one whose flag becomes 1 will be the next producer to produce.

2-3.Show and explain UART output to show the unfair version, if any, and the fair version.

1.If using the Round-Robin scheduling policy, the producer executing after the consumer will always fill the buffer. Therefore, the other producer will never get the chance to produce.

2.Adding a semaphore flag to each producer and making them wait for their own flag, then signaling the other one after they finish, will result in a fair version.