Student ID : 111062307, Name : 陳大佑

```
__data __at (0x30) char saved_SP[MAXTHREADS];
__data __at (0x34) ThreadID cur_ID;
__data __at (0x35) char bitmap_ID;
__data __at (0x36) char temp_SP;        // temporary save the SP
__data __at (0x37) ThreadID new_ID;
```

```
__data __at (0x38) char head;
__data __at (0x39) char tail;
__data __at (0x3A) char nextChar;
__data __at (0x3B) char buffer[3];

__data __at (0x20) Semaphore mutex;
__data __at (0x21) Semaphore full;
__data __at (0x22) Semaphore empty;
```

Set some parameters on manually allocated memory.

```
/* Semaphore API */
#define CNAME(s) _ ## s
#define L(label) label ## $
#define SemaphoreCreate(s, n) s = n
#define SemaphoreWaitBody(S, label)          \
    {                                        \
        __asm                                \
        L(label):  MOV A, CNAME(S)           \
                   JZ L(label)               \
                   JB ACC.7, L(label)        \
                   DEC CNAME(S)              \
        __endasm;                            \
    }
#define SemaphoreWait(S) SemaphoreWaitBody(S, __COUNTER__)
#define SemaphoreSignal(S)                   \
    __asm                                    \
        INC CNAME(S)                         \
    __endasm;
```

Label : for loop to check the semaphore's value repeatedly.

MOV A, CNAME(S): Copies the value of the semaphore S into the accumulator A.

JZ L(label) : If A == 0, it jumps back to the label L(label)

JB ACC.7, L(label) : Checks the highest bit (bit 7) of the accumulator A / jump to L

DEC CNAME(S) : Decrements the value of semaphore S by 1

Screenshots for compliation :

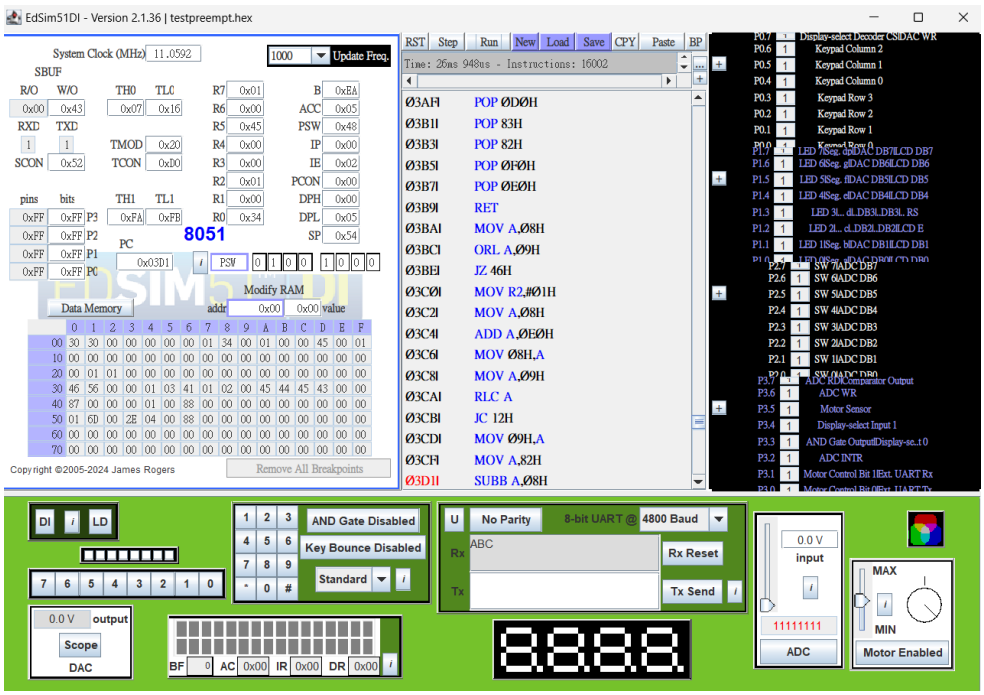
```
dylan@LAPTOP-S5FOLV5V ~/os/ppc3
$ make clean
rm *.hex *.ihx *.lnk *.lst *.map *.mem *.rel *.rst *.sym *.asm *.lk
rm: cannot remove '*.ihx': No such file or directory
rm: cannot remove '*.lnk': No such file or directory
make: *** [clean] Error 1

dylan@LAPTOP-S5FOLV5V ~/os/ppc3
$ make
sdcc -c  testpreempt.c
sdcc -c  preemptive.c
preemptive.c:161: warning 85: in function ThreadCreate unreferenced function argument : 'fp'
sdcc  -o testpreempt.hex testpreempt.rel preemptive.rel
```

Screenshots and explanation:

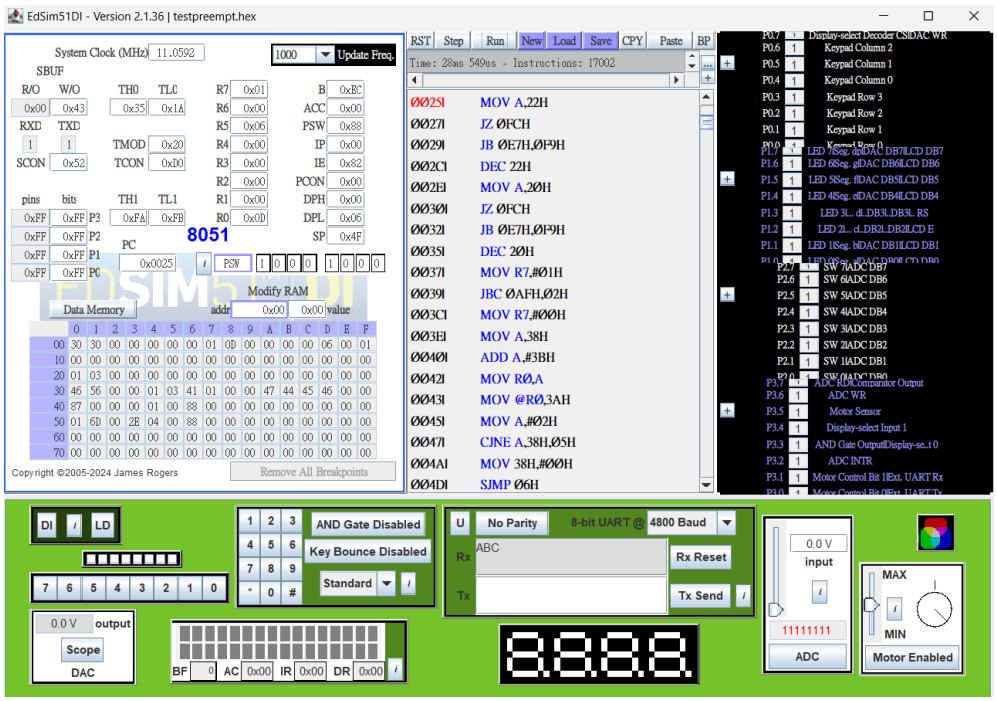
Cur_ID (34H) is 1, which means producer is running.

And now our mutex = 0, full = 1, empty = 1 。



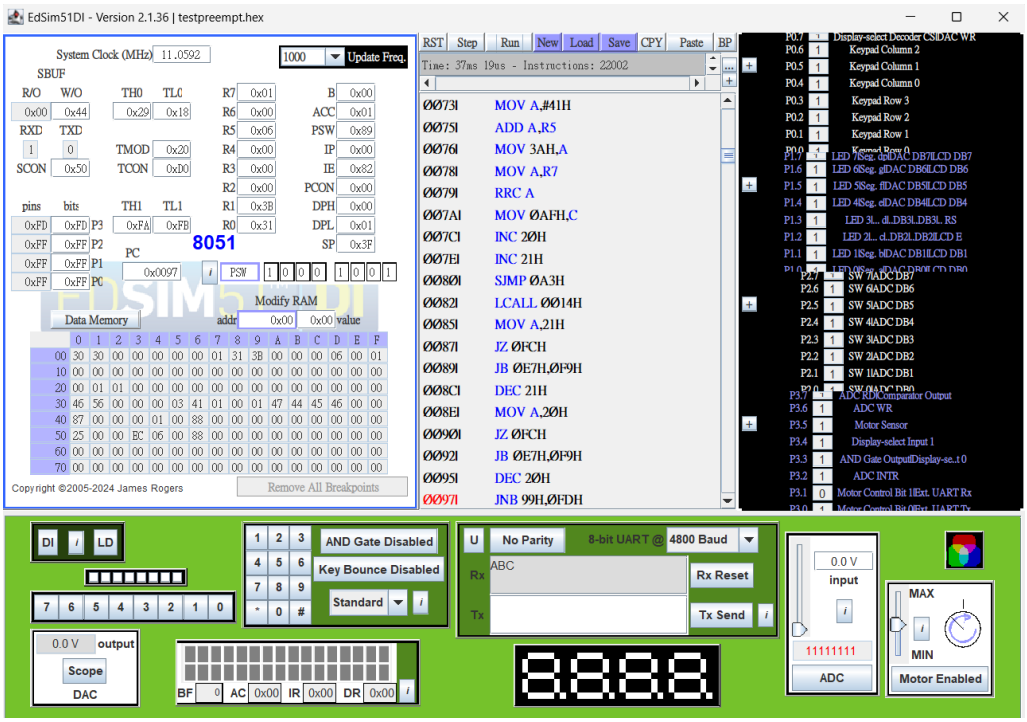
After 1000 iteration, cur_ID(34H) is still 1

But now our mutex = 1, full = 3, empty = 0 。 Which means semaphore changes



Cur_ID (34H) is 0, which means consumer is running.

And now our mutex = 0, full = 1, empty = 1 。



After 1000 iteration, cur_ID(34H) is still 0

But now our mutex = 0, full = 0, empty = 2 。 Which means semaphore changes