鍾維聖 110062261

Programming Project Checkpoint 1 Report

Typescript for Compiling

```
wilbertallen@MacBook-Air-2016 ppc2 % make clean
rm *.hex *.ihx *.lnk *.lst *.map *.mem *.rel *.rst *.sym
rm: *.ihx: No such file or directory
rm: *.lnk: No such file or directory
make: *** [clean] Error 1
wilbertallen@MacBook-Air-2016 ppc2 % make
sdcc -c  testpreempt.c
sdcc -c  preemptive.c
preemptive.c:220: warning 85: in function ThreadCreate unreferenced function argument : 'fp'
preemptive.c:285: warning 283: function declarator with no prototype
sdcc  -o testpreempt.hex testpreempt.rel preemptive.rel
wilbertallen@MacBook-Air-2016 ppc2 %
```

Fig.1 Typescript for compiling using the given makefile

Before Each ThreadCreate Call

Based on my understanding, ThreadCreate is called 2 times, one is for ThreadCreate(main), and the other is for ThreadCreate(Producer)

ThreadCreate(main);

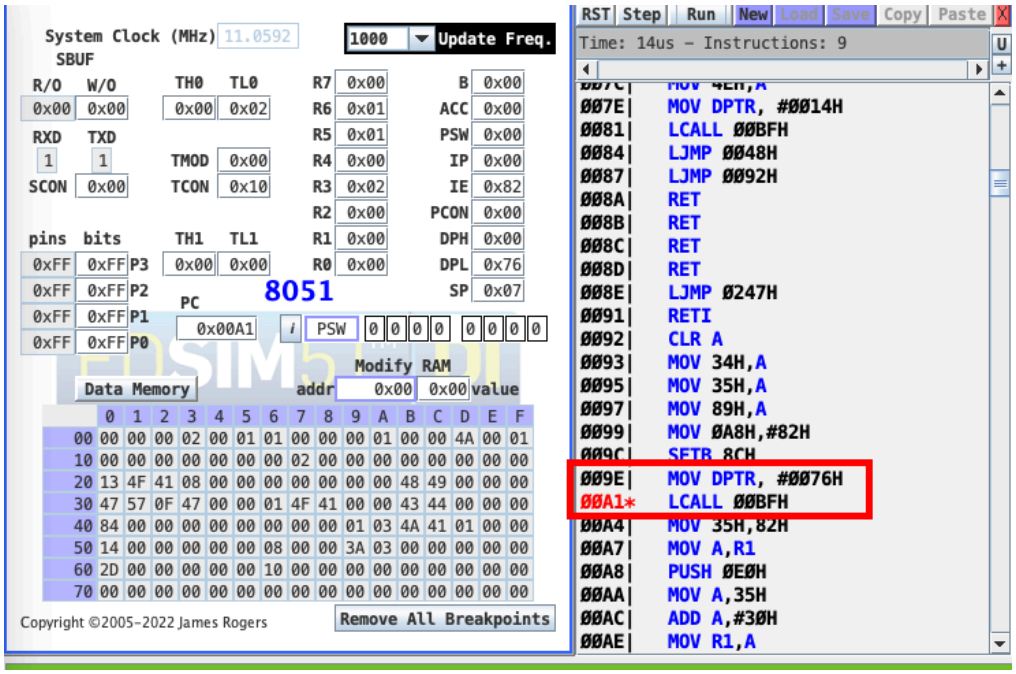


Fig2.1 Screenshot before ThreadCreate(main)

When the breakpoint reached LCALL, we can see that the address of main which is 0x76 can be seen in the DPL. It will then be pushed into SP, which will change 0x07 to 0x09.

ThreadCreate(Producer);

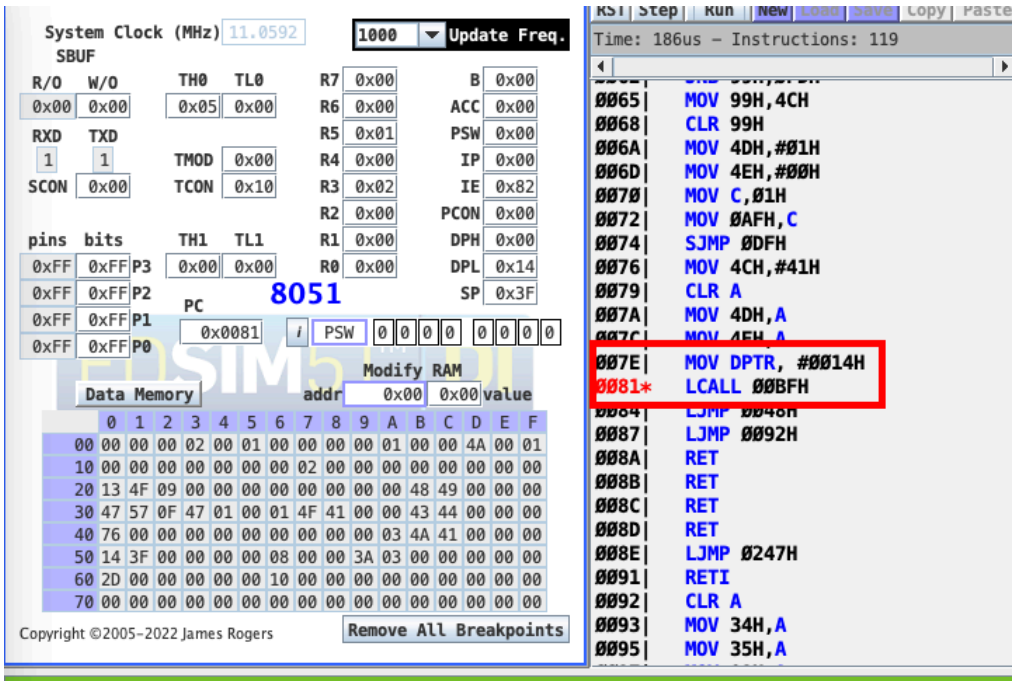


Fig.2.2 Screenshot before ThreadCreate(Producer)

Same goes for ThreadCreate(Producer). At the breakpoint, the address of Producer which is 0x14 is pushed into the Stack Pointer, which will result in a change in the SP from 0x3F to 0x41.

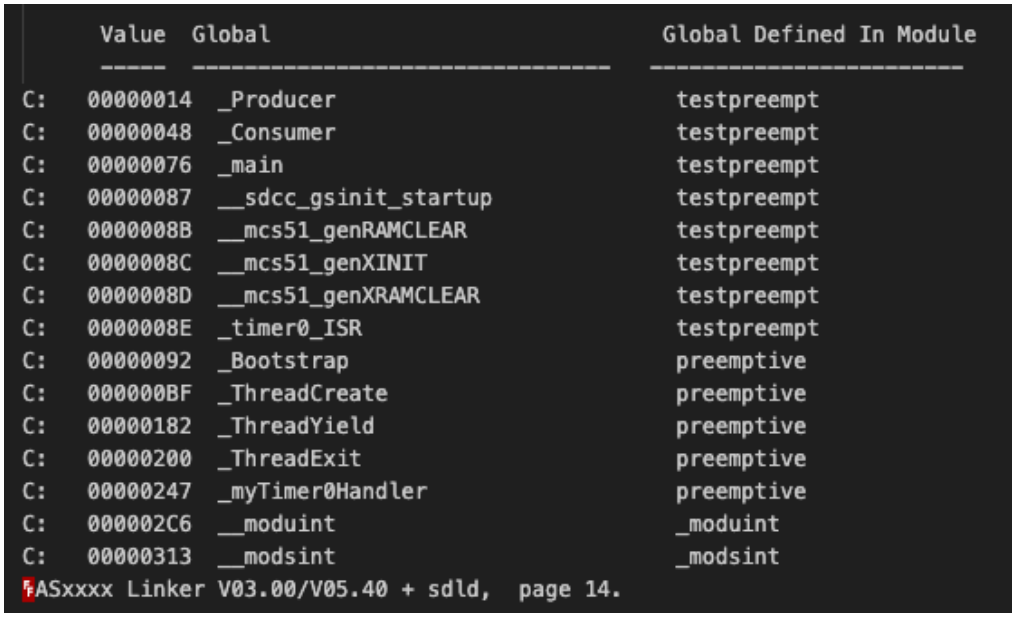
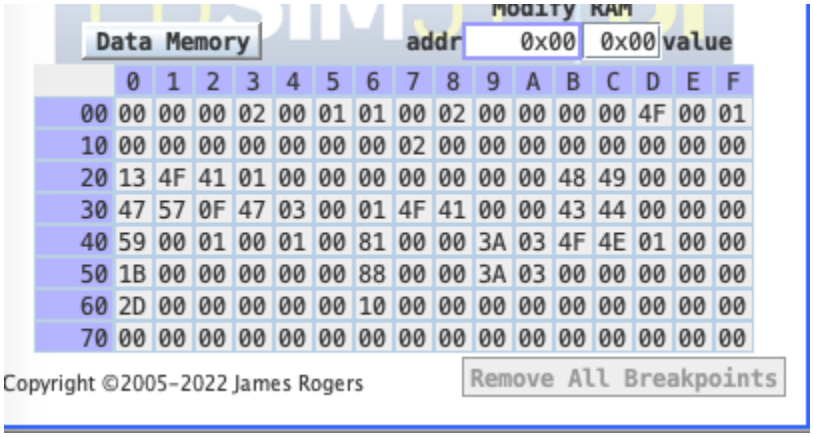


Fig.2.3 Function addresses value

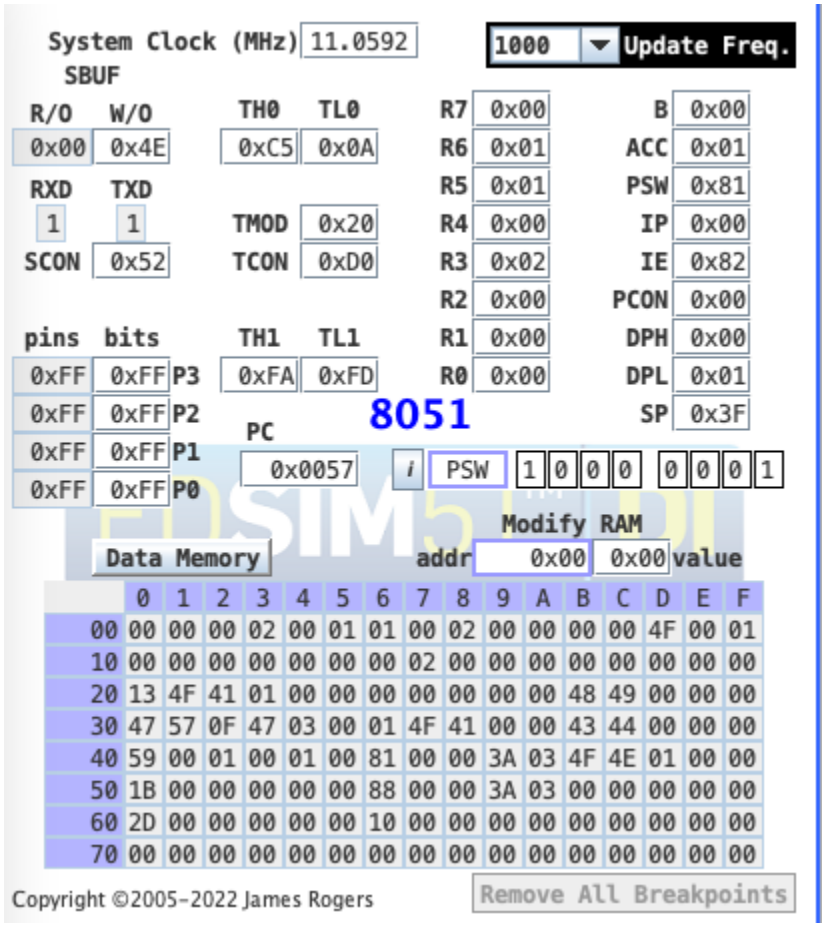
Running Producer



We can see that the Producer function is running by observing the addresses of *shared_buff* and *next_buff*. In my implementation, I store *shared_buff* on 0x4C and *next_buff* on 0x4B. At each iteration, the value from 0x4B will be copied to 0x4C. At the above example, we know that the current value of the shared buffer is 4E which is HEX for the ASCII character 'N', and the next buffer has a value of 4F, which translates to 'O'.

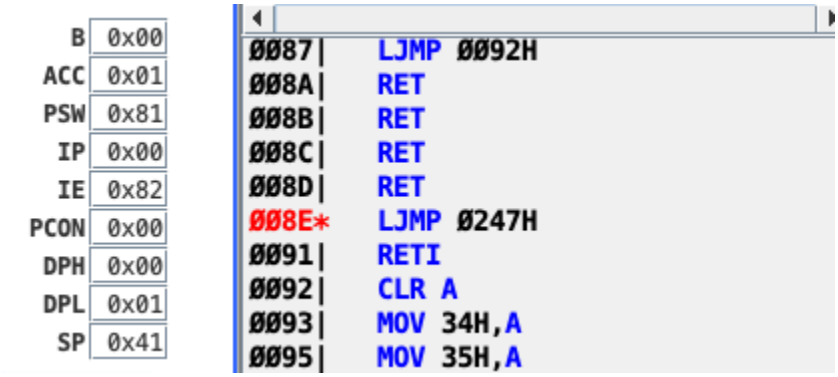
Link of additional producer video : [Drive]

Running Consumer

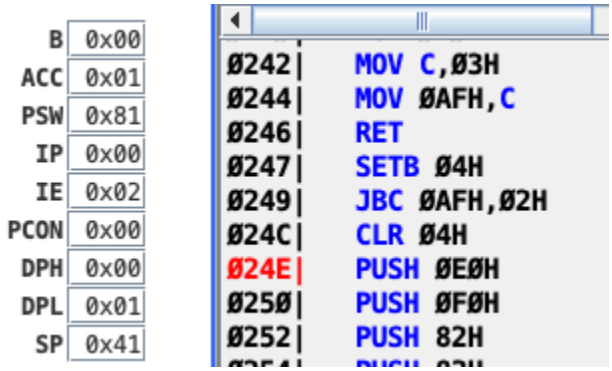


We can know Consumer is running by observing the SBUF. SBUF is currently writing out 0x4E to the received_data part on Edsim, from the shared_buffer.

Interrupt



When the timer interrupt (0x8E) is triggered, the code jumps to 0x247, which is the address for the function myTimer0Handler(), described by the LJMP above.



A few steps later, we can see that an interrupt has happened by observing the IE that changes from 0x82 to 0x02. We can tell that the interrupt is triggering on a regular basis by simply observing the changes in the IE value.