

Report Checkpoint 3

Jonathan Sutedjo 鄭安良 111006207

```
/cygdrive/c/Users/Jonat/Documents/!NTHU Classes/Operating System/Fin Pr/J...  —  □  X
preemptive.c:274: warning 85: in function ThreadCreate unreferenced function arg
ument : 'fp'
sdcc -o testpreempt.hex testpreempt.rel preemptive.rel

Jonat@LAPTOP-7F30BD0F /cygdrive/c/Users/Jonat/Documents/!NTHU Classes/Operating
System/Fin Pr/Jon/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: *** [Makefile:25: clean] Error 1

Jonat@LAPTOP-7F30BD0F /cygdrive/c/Users/Jonat/Documents/!NTHU Classes/Operating
System/Fin Pr/Jon/ppc3
$ make
sdcc -c testpreempt.c
sdcc -c preemptive.c
preemptive.c:274: warning 85: in function ThreadCreate unreferenced function arg
ument : 'fp'
sdcc -o testpreempt.hex testpreempt.rel preemptive.rel

Jonat@LAPTOP-7F30BD0F /cygdrive/c/Users/Jonat/Documents/!NTHU Classes/Operating
System/Fin Pr/Jon/ppc3
$
```

```

8      CS 3423 Fall 2018
9      */
10
11 #ifndef __PREEMPTIVE_H__
12 #define __PREEMPTIVE_H__
13
14 #define MAXTHREADS 4 /* not including the scheduler */
15 /* the scheduler does not take up a thread of its own */
16
17
18
19 #define CNAME(s) _ ## s
20 #define CNAMELABEL(label) label ## $
21
22 #define SemaphoreWait(s){ \
23     SemaphoreWaitBody(s, __COUNTER__) \
24 }
25
26
27 #define SemaphoreSignal(s){ \
28     __asm \
29     INC CNAME(s) \
30     __endasm; \
31 }
32
33 #define SemaphoreWaitBody(S, label) \
34 { __asm \
35 CNAMELABEL(label): MOV ACC, CNAME(S) \
36                   JZ CNAMELABEL(label) \
37                   JB ACC.7, CNAMELABEL(label) \
38                   dec CNAME(S) \
39                   __endasm; }
40
41
42
43 typedef char ThreadID;
44 typedef void (*FunctionPtr)(void);
45
46 ThreadID ThreadCreate(FunctionPtr);
47 void ThreadYield(void);
48 void ThreadExit(void);
49
50 #endif // __COOPERATIVE_H__

```

In the preemptive.h file, I make changes to it for the semaphore by adding the SemaphoreWait which is to call the SemaphoreWaitBody and pass the parameter with the unique integer __COUNTER__. In the SemaphoreWaitBody, it first move the value of S, then check if the ACC is equal to zero or not, the PC will jump back to MOV ACC, CNAME(S) if it is equal to zero. This will make a loop. In the condition that it is not equal then it will check the 7th bit of ACC if it is 1 or not, where this 7th bit is it to know if it is a negative number or not. If it is 1 then it will jump back to the top to be a loop, but if it is not then the PC will continue and the S will be decrease by 1. Here there is also the SemaphoreSignal which is to increase the S by 1.

```
__data __at (0x35) char TheChar;  
__data __at (0x25) char mutex;  
__data __at (0x26) char full;  
__data __at (0x27) char empty;  
__data __at (0x3A) char head;  
__data __at (0x3B) char tail;  
__data __at (0x3D) char SharedBuffer[3];
```

```
✓ void SemaphoreCreate(char *s, char n){  
    EA = 0;  
    *s = n;  
    EA = 1;  
}
```

In the testpreempt.c file, I add mutex, full, empty, tail, and head, where the tail and head is for the shared buffer. Then The var to keep the A-Z is the “TheChar”.

The SemaphoreCreate is to put a value in char to the semaphore. Here I use a pointer so that the one that I update is the value from the source address of the semaphore.

```

void Producer(void)
{
    /*
     * [TODO]
     * initialize producer data structure, and then enter
     * an infinite loop (does not return)
     */
    EA = 0;
    TheChar = 'A'-1;
    EA = 1;

    while (1)
    {
        /* [TODO]
         * wait for the buffer to be available,
         * and then write the new data into the buffer */
        SemaphoreWait(empty);
        SemaphoreWait(mutex);
        EA = 0;

        if(TheChar == 'Z'){
            TheChar = 'A';
        }
        else{
            TheChar += 1;
        }
        SharedBuffer[tail] = TheChar;
        tail += 1;
        if(tail == 3){
            tail = 0;
        }

        EA = 1;
        SemaphoreSignal(mutex);
        SemaphoreSignal(full);
    }
}

```

For the producer part, now I use the bounded buffer such that I need to wait for the semaphore to be ready before doing anything to the shared buffer. The producer will need to wait for the “empty” to be not 0 and the “mutex” to be 1 before it can produce more character. After the character is produced of course it is assigned to the SharedBuffer[tail]. After it is done, it will return the value of mutex and signal full. The full signal is so that the consumer know there is new value inside the shared buffer. The mutex is so that other thread can use it.

```

void Consumer(void)
{
    /*
     * [TODO]
     * initialize Tx for polling
     */
    TMOD |= 0x20;
    TH1 = (char)-6;
    SCON = 0x50;
    TR1 = 1;
    TI = 1;

    while (1)
    {
        /*
         * [TODO]
         * wait for new data from producer
         * write data to serial port Tx,
         * poll for Tx to finish writing (TI),
         * then clear the flag
         */

        SemaphoreWait(full);
        SemaphoreWait(mutex);
        EA = 0;

        while (!TI){
        }
        SBUF = SharedBuffer[head];
        TI = 0;
        head += 1;
        if(head == 3){
            head = 0;
        }

        EA = 1;
        SemaphoreSignal(mutex);
        SemaphoreSignal(empty);
    }
}

```

In the consumer part, I use the bounded buffer such that the consumer will have to wait for the producer to signal full to get the new value in the shared buffer. After the signal is received, it will wait for the mutex. Consumer will signal mutex to let other thread run after it is done. It will also signal empty to let the producer know the sharedbuffer have spot available for new characters now.

```

void main(void)
{
    /*
     * [TODO]
     * initialize globals
     */

    SemaphoreCreate(&mutex, 1);
    SemaphoreCreate(&full, 0);
    SemaphoreCreate(&empty, 3);

    head = 0;
    tail = 0;

    SharedBuffer[0] = ' ';
    SharedBuffer[1] = ' ';
    SharedBuffer[2] = ' ';

    /*
     * [TODO]
     * set up Producer and Consumer.
     * Because both are infinite loops, there is no loop
     * in this function and no return.
     */
    ThreadCreate(Producer);
    Consumer();
}

```

Here, I need to first initialize the semaphores by using the SemaphoreCreate for full, empty, and mutex. The value of full will be 0 because it means there are no new characters there. The value for mutex is so that only 1 thread runs at a time. The empty will be 3 because there are 3 slots available for new characters and that is what we want (3-deep).

| Area | Addr | Size | Decimal | Bytes | (Attributes) |
|------|----------|----------------------|---------|--------------------------|------------------|
| CSEG | 00000014 | 000004AF = | 1199. | bytes | (REL, CON, CODE) |
| | Value | Global | | Global Defined In Module | |
| C: | 00000014 | _SemaphoreCreate | | testpreempt | |
| C: | 0000002A | _Producer | | testpreempt | |
| C: | 00000073 | _Consumer | | testpreempt | |
| C: | 000000B8 | _main | | testpreempt | |
| C: | 000000F4 | _sdcc_gsinit_startup | | testpreempt | |
| C: | 000000F8 | _mcs51_genRAMCLEAR | | testpreempt | |
| C: | 000000F9 | _mcs51_genXINIT | | testpreempt | |
| C: | 000000FA | _mcs51_genXRAMCLEAR | | testpreempt | |
| C: | 000000FB | _timer0_ISR | | testpreempt | |
| C: | 000000FF | _Bootstrap | | preemptive | |
| C: | 00000153 | _ThreadCreate | | preemptive | |
| C: | 00000207 | _ThreadYield | | preemptive | |
| C: | 000002E7 | _ThreadExit | | preemptive | |
| C: | 000003CE | _myTimer0Handler | | preemptive | |
| C: | 000004A8 | _gptrput | | _gptrput | |

ASxxxx Linker V03.00/V05.40 + sldd, page 13.

This is some of the map file content.

Image Context 1

EdSim51DI - Version 2.1.38 & Dynamic Interface x | testpreempt.hex

System Clock (MHz): 11.0592 | Update Freq: 1

SBUF: R/O W/O TH0 TL0 R7 0x31 B 0x00
0x00 0x00 0x00 0x08 ACC 0x00
R5D TXD 1 1 TMOD 0x00 R4 0x00 PSW 0x00
SCON 0x00 TCON 0x10 R3 0x00 IP 0x00
R2 0x00 PCON 0x00
R1 0x00 DPH 0x00
R0 0x00 DPL 0xB8
SP 0x09

pins bits TH1 TL1
0xFF 0xFF P3 0x00 0x00
0xFF 0xFF P2 0x00 0x00
0xFF 0xFF P1 0x00 0x00
0xFF 0xFF P0 0x00 0x00

Modify RAM
Data Memory
addr 0x00 0x00 value
0 1 2 3 4 5 6 7 8 9 A B C D E F
00 00 00 00 00 00 27 00 31 1D 01 00 00 00 33 01 01
10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
20 01 30 41 4F 00 01 00 03 00 00 00 00 00 00 00 00
30 3F 4F 5F 6F 00 30 4F 00 4B 00 00 31 20 20 20
40 F1 00 27 00 00 00 00 93 00 00 40 31 00 88 00 00
50 2A 00 00 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 ©2005-2024 James Rogers | Remove All Breakpoints

Time: 29us - Instructions: 18

```

ORG 0000H
0000| LJM0 00F4H
0003| RETI
ORG 000BH
000B| LJM0 00FBH
000E| LJM0 00B8H
0011| LJM0 000EH
0014| MOV R5,82H
0016| MOV R6,83H
0018| MOV R7,0F0H
001A| CLR 0AFH
001C| MOV 82H,R5
001E| MOV 83H,R6
0020| MOV 0F0H,R7
0022| MOV A,08H
0024| LCALL 04A8H
0027| SETB 0AFH
0029| RET
002A| CLR 0AFH
002C| MOV 35H,#40H
002F| SETB 0AFH

```

P0.7 1 Display-select Decoder CS|DAC WR
P0.6 1 Keypad Column 2
P0.5 1 Keypad Column 1
P0.4 1 Keypad Column 0
P0.3 1 Keypad Row 3
P0.2 1 Keypad Row 2
P0.1 1 Keypad Row 1
P0.0 1 Keypad Row 0
P1.7 1 LED 7|Seg. dp|DAC DB7|LCD DB7
P1.6 1 LED 6|Seg. g|DAC DB6|LCD DB6
P1.5 1 LED 5|Seg. f|DAC DB5|LCD DB5
P1.4 1 LED 4|Seg. e|DAC DB4|LCD DB4
P1.3 1 LED 3|... d|..DB3|..DB3|.. RS
P1.2 1 LED 2|... c|..DB2|..DB2|LCD E
P1.1 1 LED 1|Seg. b|DAC DB1|LCD DB1
P1.0 1 LED 0|Seg. a|DAC DB0|LCD DB0
P2.7 1 SW 7|ADC DB7
P2.6 1 SW 6|ADC DB6
P2.5 1 SW 5|ADC DB5
P2.4 1 SW 4|ADC DB4
P2.3 1 SW 3|ADC DB3
P2.2 1 SW 2|ADC DB2
P2.1 1 SW 1|ADC DB1
P2.0 1 SW 0|ADC DB0
P3.7 1 ADC RD|Comparator Output
P3.6 1 ADC WR
P3.5 1 Motor Sensor
P3.4 1 Display-select Input 1
P3.3 1 AND Gate Output|Display-se..t 0
P3.2 1 ADC INTR
P3.1 1 Motor Control Bit 1|Ext. UART Rx
P3.0 1 Motor Control Bit 0|Ext. UART Tx

DI | LD
AND Gate Disabled
Key Bounce Disabled
Standard |
7 6 5 4 3 2 1 0
0.0V output
Scope
DAC
BF 0 AC 0x00 IR 0x00 DR 0x00
U No Parity 8-bit UART @ 4800 Baud
Rx Rx Reset
Tx Tx Send
0.0V input
11111111
ADC
MAX
MIN
Motor Enabled

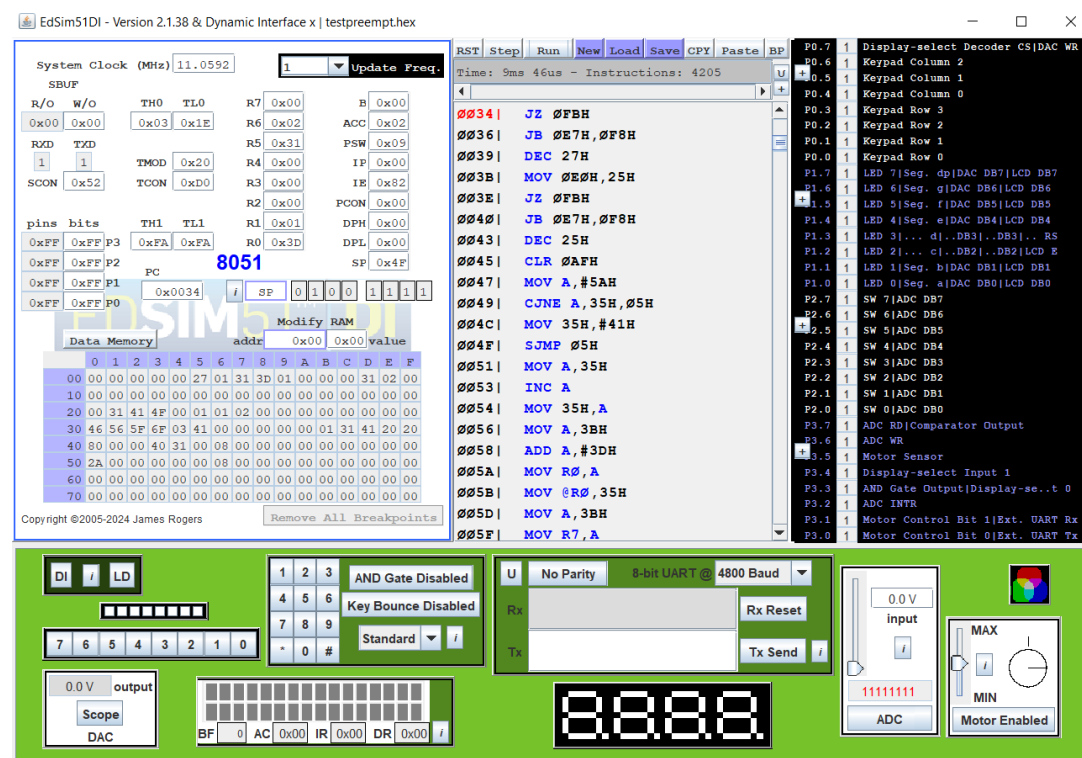
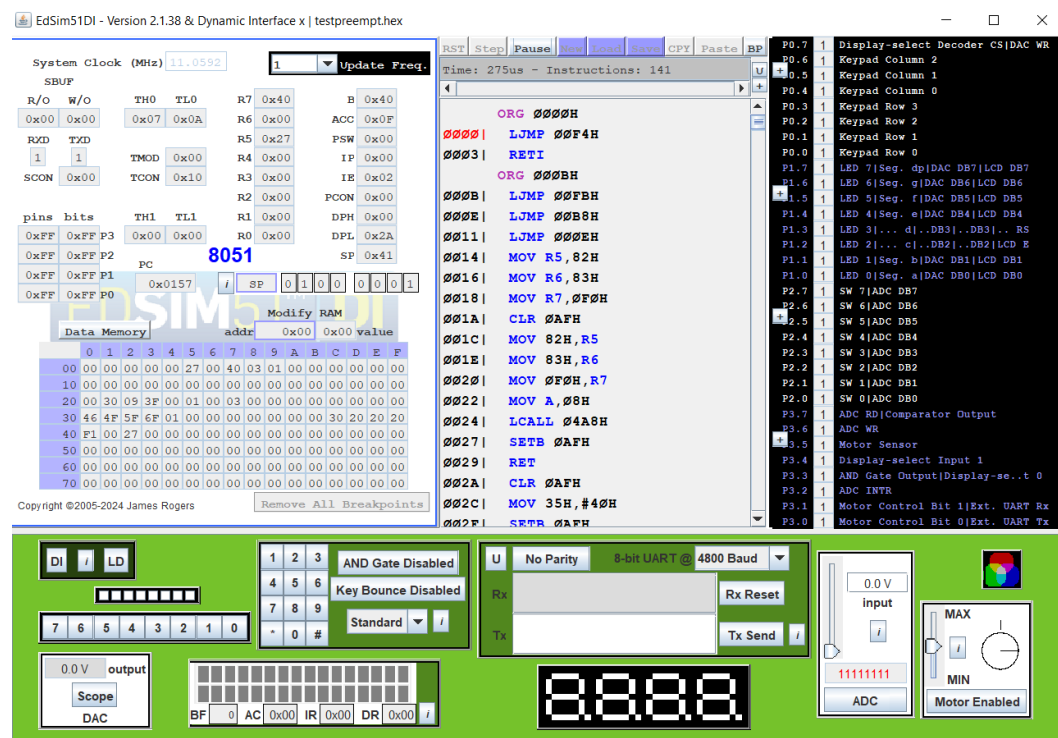


Image Context 4

EdSim51DI - Version 2.1.38 & Dynamic Interface x | testpreempt.hex

System Clock (MHz) 11.0592 | Update Freq. 1

SBUF

| R/O | W/O | TH0 | TL0 | R7 | B |
|------|------|------|------|------|------|
| 0x00 | 0x00 | 0x05 | 0x04 | 0x01 | 0x00 |

RXD TXD

| R/O | W/O | TH0 | TL0 | R7 | B |
|-----|-----|------|------|------|------|
| 1 | 1 | 0x20 | 0x00 | 0x01 | 0x00 |

SCON 0x52 TCON 0xD0

pins bits

| TH1 | TL1 | R7 | B |
|------|------|------|------|
| 0xFF | 0xFF | 0x01 | 0x00 |

PC 0x0034

Modify RAM

| addr | 0x00 | 0x00 | value |
|------|------|------|-------|
| 0 | 0 | 0 | 0 |

Data Memory

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |

Copyright ©2005-2024 James Rogers

Remove All Breakpoints

RST Step Run New Load Save CPY Paste BP

Time: 9ms 87us - Instructions: 4233

```
0034 | JZ 0FBH
0036 | JB 0E7H,0F8H
0039 | DEC 27H
003B | MOV 0E0H,25H
003E | JZ 0FBH
0040 | JB 0E7H,0F8H
0043 | DEC 25H
0045 | CLR 0AFH
0047 | MOV A,#5AH
0049 | CJNE A,35H,05H
004C | MOV 35H,#41H
004F | SJMP 05H
0051 | MOV A,35H
0053 | INC A
0054 | MOV 35H,A
0056 | MOV A,3BH
0058 | ADD A,#3DH
005A | MOV R0,A
005B | MOV @R0,35H
005D | MOV A,3BH
005F | MOV R7,A
```

P0.7 1 Display-select Decoder CS|DAC WR
P0.6 1 Keypad Column 2
P0.5 1 Keypad Column 1
P0.4 1 Keypad Column 0
P0.3 1 Keypad Row 3
P0.2 1 Keypad Row 2
P0.1 1 Keypad Row 1
P0.0 1 Keypad Row 0
P1.7 1 LED 7|Seg. dp|DAC DB7|LCD DB7
P1.6 1 LED 6|Seg. g|DAC DB6|LCD DB6
P1.5 1 LED 5|Seg. f|DAC DB5|LCD DB5
P1.4 1 LED 4|Seg. e|DAC DB4|LCD DB4
P1.3 1 LED 3|... d|..DB3|..DB3|.. RS
P1.2 1 LED 2|... c|..DB2|..DB2|LCD E
P1.1 1 LED 1|Seg. b|DAC DB1|LCD DB1
P1.0 1 LED 0|Seg. a|DAC DB0|LCD DB0
P2.7 1 SW 7|ADC DB7
P2.6 1 SW 6|ADC DB6
P2.5 1 SW 5|ADC DB5
P2.4 1 SW 4|ADC DB4
P2.3 1 SW 3|ADC DB3
P2.2 1 SW 2|ADC DB2
P2.1 1 SW 1|ADC DB1
P2.0 1 SW 0|ADC DB0
P3.7 1 ADC RD|Comparator Output
P3.6 1 ADC WR
P3.5 1 Motor Sensor
P3.4 1 Display-select Input 1
P3.3 1 AND Gate Output|Display-se..t 0
P3.2 1 ADC INTR
P3.1 1 Motor Control Bit 1|Ext. UART Rx
P3.0 1 Motor Control Bit 0|Ext. UART Tx

DI / LD

AND Gate Disabled

Key Bounce Disabled

Standard

U No Parity 8-bit UART @ 4800 Baud

Rx Rx Reset

Tx Tx Send

0.0V output

Scope DAC

BF 0 AC 0x00 IR 0x00 DR 0x00

0.0V input

ADC

MAX MIN

Motor Enabled

Image Context 5

EdSim51DI - Version 2.1.38 & Dynamic Interface x | testpreempt.hex

System Clock (MHz) 11.0592 | Update Freq. 1

SBUF

| R/O | W/O | TH0 | TL0 | R7 | B |
|------|------|------|------|------|------|
| 0x00 | 0x00 | 0x06 | 0x10 | 0x02 | 0x00 |

RXD TXD

| R/O | W/O | TH0 | TL0 | R7 | B |
|-----|-----|------|------|------|------|
| 1 | 1 | 0x20 | 0x00 | 0x01 | 0x00 |

SCON 0x52 TCON 0xD0

pins bits

| TH1 | TL1 | R7 | B |
|------|------|------|------|
| 0xFF | 0xFF | 0x01 | 0x00 |

PC 0x0034

Modify RAM

| addr | 0x00 | 0x00 | value |
|------|------|------|-------|
| 0 | 0 | 0 | 0 |

Data Memory

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |

Copyright ©2005-2024 James Rogers

Remove All Breakpoints

RST Step Run New Load Save CPY Paste BP

Time: 9ms 135us - Instructions: 4264

```
0034 | JZ 0FBH
0036 | JB 0E7H,0F8H
0039 | DEC 27H
003B | MOV 0E0H,25H
003E | JZ 0FBH
0040 | JB 0E7H,0F8H
0043 | DEC 25H
0045 | CLR 0AFH
0047 | MOV A,#5AH
0049 | CJNE A,35H,05H
004C | MOV 35H,#41H
004F | SJMP 05H
0051 | MOV A,35H
0053 | INC A
0054 | MOV 35H,A
0056 | MOV A,3BH
0058 | ADD A,#3DH
005A | MOV R0,A
005B | MOV @R0,35H
005D | MOV A,3BH
005F | MOV R7,A
```

P0.7 1 Display-select Decoder CS|DAC WR
P0.6 1 Keypad Column 2
P0.5 1 Keypad Column 1
P0.4 1 Keypad Column 0
P0.3 1 Keypad Row 3
P0.2 1 Keypad Row 2
P0.1 1 Keypad Row 1
P0.0 1 Keypad Row 0
P1.7 1 LED 7|Seg. dp|DAC DB7|LCD DB7
P1.6 1 LED 6|Seg. g|DAC DB6|LCD DB6
P1.5 1 LED 5|Seg. f|DAC DB5|LCD DB5
P1.4 1 LED 4|Seg. e|DAC DB4|LCD DB4
P1.3 1 LED 3|... d|..DB3|..DB3|.. RS
P1.2 1 LED 2|... c|..DB2|..DB2|LCD E
P1.1 1 LED 1|Seg. b|DAC DB1|LCD DB1
P1.0 1 LED 0|Seg. a|DAC DB0|LCD DB0
P2.7 1 SW 7|ADC DB7
P2.6 1 SW 6|ADC DB6
P2.5 1 SW 5|ADC DB5
P2.4 1 SW 4|ADC DB4
P2.3 1 SW 3|ADC DB3
P2.2 1 SW 2|ADC DB2
P2.1 1 SW 1|ADC DB1
P2.0 1 SW 0|ADC DB0
P3.7 1 ADC RD|Comparator Output
P3.6 1 ADC WR
P3.5 1 Motor Sensor
P3.4 1 Display-select Input 1
P3.3 1 AND Gate Output|Display-se..t 0
P3.2 1 ADC INTR
P3.1 1 Motor Control Bit 1|Ext. UART Rx
P3.0 1 Motor Control Bit 0|Ext. UART Tx

DI / LD

AND Gate Disabled

Key Bounce Disabled

Standard

U No Parity 8-bit UART @ 4800 Baud

Rx Rx Reset

Tx Tx Send

0.0V output

Scope DAC

BF 0 AC 0x00 IR 0x00 DR 0x00

0.0V input

ADC

MAX MIN

Motor Enabled

From Image context 1-5, we can see that the value of full will be increase by 1 every time a new character is produced by the producer. While the full is increased by 1, the empty will decrease by 1.

EdSim51DI - Version 2.1.38 & Dynamic Instruction | test|preempt.hex

System Clock (MHz) 11.0592 Update Freq. 1

SBUF

R/O W/O TH0 TL0 R7 0x01 B 0x40

0x00 0x42 0x68 0x1C

RMD TXD

1 0 TMOD 0x20

SCON 0x50 TCON 0xD0

pins bits TH1 TL1

0xF0 0xF0 P3 0xFA 0xFE

0xFF 0xFF P2

0xFF 0xFF P1

0xFF 0xFF P0

PC 0x0096

8051

Modify RAM

Data Memory

| addr | 0x00 | 0x00 | value |
|------|------|------|-------|
| 0 | 0 | 1 | 2 |
| 1 | 0 | 0 | 0 |
| 2 | 0 | 0 | 0 |
| 3 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 |
| 9 | 0 | 0 | 0 |
| A | 0 | 0 | 0 |
| B | 0 | 0 | 0 |
| C | 0 | 0 | 0 |
| D | 0 | 0 | 0 |
| E | 0 | 0 | 0 |
| F | 0 | 0 | 0 |
| 10 | 0 | 0 | 0 |
| 11 | 0 | 0 | 0 |
| 12 | 0 | 0 | 0 |
| 13 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 |
| 17 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 |
| 19 | 0 | 0 | 0 |
| 20 | 0 | 0 | 0 |
| 21 | 0 | 0 | 0 |
| 22 | 0 | 0 | 0 |
| 23 | 0 | 0 | 0 |
| 24 | 0 | 0 | 0 |
| 25 | 0 | 0 | 0 |
| 26 | 0 | 0 | 0 |
| 27 | 0 | 0 | 0 |
| 28 | 0 | 0 | 0 |
| 29 | 0 | 0 | 0 |
| 30 | 0 | 0 | 0 |
| 31 | 0 | 0 | 0 |
| 32 | 0 | 0 | 0 |
| 33 | 0 | 0 | 0 |
| 34 | 0 | 0 | 0 |
| 35 | 0 | 0 | 0 |
| 36 | 0 | 0 | 0 |
| 37 | 0 | 0 | 0 |
| 38 | 0 | 0 | 0 |
| 39 | 0 | 0 | 0 |
| 40 | 0 | 0 | 0 |
| 41 | 0 | 0 | 0 |
| 42 | 0 | 0 | 0 |
| 43 | 0 | 0 | 0 |
| 44 | 0 | 0 | 0 |
| 45 | 0 | 0 | 0 |
| 46 | 0 | 0 | 0 |
| 47 | 0 | 0 | 0 |
| 48 | 0 | 0 | 0 |
| 49 | 0 | 0 | 0 |
| 50 | 0 | 0 | 0 |
| 51 | 0 | 0 | 0 |
| 52 | 0 | 0 | 0 |
| 53 | 0 | 0 | 0 |
| 54 | 0 | 0 | 0 |
| 55 | 0 | 0 | 0 |
| 56 | 0 | 0 | 0 |
| 57 | 0 | 0 | 0 |
| 58 | 0 | 0 | 0 |
| 59 | 0 | 0 | 0 |
| 60 | 0 | 0 | 0 |
| 61 | 0 | 0 | 0 |
| 62 | 0 | 0 | 0 |
| 63 | 0 | 0 | 0 |
| 64 | 0 | 0 | 0 |
| 65 | 0 | 0 | 0 |
| 66 | 0 | 0 | 0 |
| 67 | 0 | 0 | 0 |
| 68 | 0 | 0 | 0 |
| 69 | 0 | 0 | 0 |
| 70 | 0 | 0 | 0 |

Remove All Breakpoints

RST Step Run New Load Save CPY Paste BP

Time: 21ms 440us - Instructions: 9966

0065 | CJNE A, 3BH, 03H

0068 | MOV 3BH, #00H

006B | SETB 0AFH

006D | INC 25H

006F | INC 26H

0071 | SJMP 0BEH

0073 | ORL 89H, #20H

0076 | MOV 8DH, #0FAH

0079 | MOV 98H, #50H

007C | SETB 8EH

007E | SETB 99H

0080 | MOV 0E0H, 26H

0083 | JZ 0FBH

0085 | JB 0E7H, 0F8H

0088 | DEC 26H

008A | MOV 0E0H, 25H

008D | JZ 0FBH

008F | JB 0E7H, 0F8H

0092 | DEC 25H

0094 | CLR 0AFH

0096 | JNB 99H, 0FDH

0097 |

0098 |

0099 |

009A |

009B |

009C |

009D |

009E |

009F |

00A0 |

00A1 |

00A2 |

00A3 |

00A4 |

00A5 |

00A6 |

00A7 |

00A8 |

00A9 |

00AA |

00AB |

00AC |

00AD |

00AE |

00AF |

Image context 7

EdSim51DI - Version 2.1.38 & Dynamic Interface x | testpreempt.hex

System Clock (MHz): 11.0592 | 100 | Update Freq.

SBUF

| | | | | | | | |
|------|------|------|------|----|------|------|------|
| R/O | W/O | TH0 | TL0 | R7 | 0x02 | B | 0x40 |
| 0x00 | 0x43 | 0xC0 | 0x12 | R6 | 0x01 | ACC | 0x00 |
| RxD | TxD | TMOD | 0x20 | R5 | 0x33 | PSW | 0x08 |
| 1 | 1 | TCOD | 0x20 | R4 | 0x00 | IP | 0x00 |
| SCON | 0x52 | TCOD | 0xD0 | R3 | 0x00 | IE | 0x82 |
| | | | | R2 | 0x00 | PCON | 0x00 |
| | | | | R1 | 0x3F | DPH | 0x00 |
| | | | | R0 | 0x3F | DPL | 0x31 |
| | | | | | | SP | 0x3F |

pins bits TH1 TL1

| | | | | |
|------|------|----|------|------|
| 0xFF | 0xFF | P3 | 0xFA | 0xFC |
| 0xFF | 0xFF | P2 | | |
| 0xFF | 0xFF | P1 | | |
| 0xFF | 0xFF | P0 | | |

8051

Data Memory

| | | | | | | | | | | | | | | | |
|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| 00 | 00 | 00 | 00 | 00 | 00 | 27 | 01 | 31 | 3F | 00 | 00 | 00 | 33 | 01 | 02 |
| 10 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 20 | 00 | 30 | 41 | 4F | 00 | 01 | 00 | 03 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 30 | 46 | 56 | 5F | 6F | 03 | 43 | 00 | 00 | 00 | 00 | 00 | 31 | 41 | 42 | 43 |
| 40 | 80 | 00 | 00 | 40 | 31 | 00 | 08 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 50 | 34 | 00 | 00 | 00 | 00 | 00 | 08 | 00 | 00 | 00 | 00 | 00 | 00 | 00 | 00 |
| 60 | 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 |

Copyright ©2005-2024 James Rogers

Remove All Breakpoints

Assembly Code:

```

0065 | CJNE A, 3BH, 03H
0068 | MOV 3BH, #00H
006B | SETB 0AFH
006D | INC 25H
006F | INC 26H
0071 | SJMP 0BEH
0073 | ORL 89H, #20H
0076 | MOV 8DH, #0FAH
0079 | MOV 98H, #50H
007C | SETB 8EH
007E | SETB 99H
0080 | MOV 0E0H, 26H
0083 | JZ 0FBH
0085 | JB 0E7H, 0F8H
0088 | DEC 26H
008A | MOV 0E0H, 25H
008D | JZ 0FBH
008F | JB 0E7H, 0F8H
0092 | DEC 25H
0094 | CLR 0AFH
0096 | JNB 99H, 0FDH
    
```

Control Panel:

- DI, LD
- AND Gate Disabled
- Key Bounce Disabled
- Standard
- 0 0 #
- 0.0V output
- Scope DAC
- BF 0 AC 0x00 IR 0x00 DR 0x00
- UART: No Parity, 8-bit UART @ 4800 Baud
- Rx: ABC
- Rx Reset
- Tx Send
- 0.0V input
- 11111111
- ADC
- MAX MIN
- Motor Enabled

From image context 5-7 we can see that the full will decrease by 1 every time when it print out 1 character. The empty will increase by 1 also when the full is decreasing.