

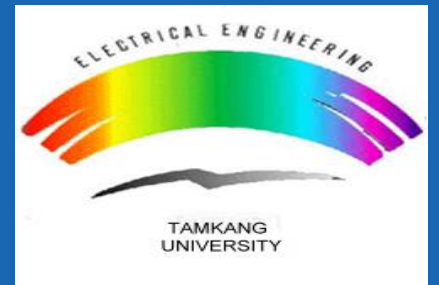
# 第10次實習課-電資

學生：林培瑋

2024 Advanced Mixed-Operation System (AMOS) Lab.



**Tamkang University**  
**Department of Electrical and Computer Engineering**  
No.151, Yingzhuan Rd., Tamsui Dist., New Taipei City 25137, Taiwan (R.O.C.)

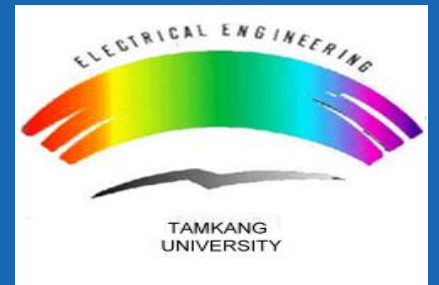


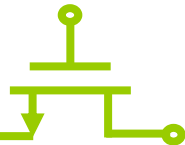
# 期中上機考

**2024 Advanced Mixed-Operation System (AMOS) Lab.**



**Tamkang University**  
**Department of Electrical and Computer Engineering**  
No.151, Yingzhuan Rd., Tamsui Dist., New Taipei City 25137, Taiwan (R.O.C.)





- ❖ 1.(1)(a)(9%)
- ❖ 1.(1)(b)(9%)
- ❖ 1.(1)(c)(9%)
- ❖ 1.(2)(a)(9%)
- ❖ 1.(2)(b)(9%)
- ❖ 1.(2)(c)(9%)
- ❖ 1.(3)(a)(9%)
- ❖ 1.(3)(b)(9%)
- ❖ 1.(3)(c)(9%)
- ❖ 2.(1)(9%)
- ❖ 2.(2)(10%)

- (1) put necessary Keil Tool DEBUG window screenshots to show your program and execution results including highlighted necessary initial assumptions and subsequent memory, register and stack changes,
- (2) comment student ID+your English name in every screenshots, and
- (3) put reports into one word file named by student\_ID+your\_name.

→補繳分數 = 原始分數\*0.8

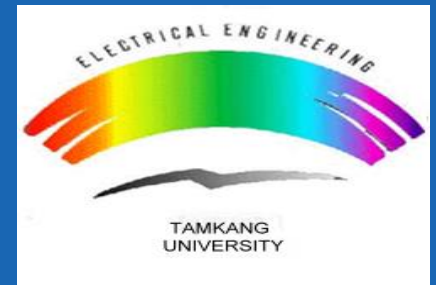


1. Rewrite the UART Program in Sec. 16.2.5 by using **full descending stack** for subroutine **UARTconfig** and **empty descending stack** for subroutine **Transmit** (both with initial stack pointer 0x40000020, to STM and LDM in the subroutine)

2024 Advanced Mixed-Operation System (AMOS) Lab.



**Tamkang University**  
**Department of Electrical and Computer Engineering**  
No.151, Yingzhuan Rd., Tamsui Dist., New Taipei City 25137, Taiwan (R.O.C.)



(1) (a) to configure the UART 5 data bits, even parity, 2 stop bits, a Baud rate if the UART is to generate a serial signal at a Baud rate of 12800 Baud using 48 MHz, and show the results in the window of UART0 after execution.

The screenshot displays the Keil uVision IDE interface during the execution of a program. The main window shows the assembly code for `TEST1-1-a.s`. The code includes instructions for loading characters, comparing them to null, and sending them to the UART. The `UARTConfig` routine is highlighted, showing the configuration of the UART0 register (U0LCR) with 5 data bits, even parity, and 2 stop bits. The baud rate is set to 12800 using the `U0DL` register.

The `Universal Asynchronous Receive Transmitt 0 (UART0)` configuration window is open, showing the following settings:

- Line Control: U0LCR: 0x1C, Word Length: 5 bits, Stop Bits: 2, Parity: Even Parity, Parity Enable: ☒
- Interrupt Enable: U0IER: 0x00, RBR IE: ☐, THRE IE: ☐, Rx Line Status IE: ☐
- Divisor Latch: U0DLL: 0xEA, U0DLM: 0x00, Baudrate: 12800
- Line Status: U0LSR: 0x60, Receiver Data Ready (RDR): ☐, Overrun Error (OE): ☐, Parity Error (PE): ☐, Framing Error (FE): ☐, Break Interrupt (BI): ☐, Tx Holding Register Empty (THRE): ☒, Transmitter Empty (TEMT): ☒, Error in Rx FIFO (RXFE): ☐
- Interrupt ID & FIFO Control: U0IIR/FCR: 0x01, FIFO Enable: ☐, Interrupt: None, Rx Trigger: Level 0 (1), Rx FIFO Reset: ☐, Tx FIFO Reset: ☐
- Receiver & Transmitter Registers: U0RBR/THR: 0x00
- Scratch Pad Register: U0SCR: 0x00

The `Registers` window shows the current state of the registers, with `R0` containing `0x00000000` and `R1` containing `0x00000000`. The `Command` window shows the command `Running with Code Size Limit: 32K` and the load path `Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\TEST1-1-a\Obj`.

The `UART #1` window shows the UART output, which is currently empty.

The `小算盤` (Calculator) window is open, showing the calculation  $48000000 \div 12800 = 16$  and the result `234`.

(1) (b) calculate the system clock frequency from the window of UART0 in (a).



Divisor Latch

U0DLL: 0xEA

U0DLM: 0x00

Baudrate: 801

小算盤

程式設計人員

$234 \times 16 \times 801 =$

**2,998,944**

HEX 2D C2A0

DEC 2,998,944

OCT 13 341 240

BIN 0010 1101 1100 0010 1010 0000

QWORD MS M

位元 位元移位

A << >> CE

B ( ) % ÷

C 7 8 9 ×

D 4 5 6 -

E 1 2 3 +

F +/- 0 . =

(1) (c) to configure the UART 8 data bits, odd parity, 1 stop bits, a Baud rate if the UART is to generate a serial signal at a Baud rate of 12800 Baud **using Keil Tool LPC 2104 CPU frequency** and show the above results in the window of UART0 after execution.



**Registers**

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000000
CPSR	0x000000D3
SFSR	0x00000000

Mode: Supervisor

**Disassembly**

```

0x00000000 E59FD084 LDR R13, [PC, #0x0084]
10: BL UARTConfig ; initialize/configure UART0
0x00000004 EB000005 BL 0x00000020
11: LDR r1, = CharData ; starting address of characters
12:
13: Loop
0x00000008 E59FD084 LDR R13, [PC, #0x0084]

```

**TEST1-1.c.s**

```

25: ; are set to 8 bits, no parity and 1 stop bit.
26: ; Registers used:
27: ; r5 - scratch register
28: ; r6 - scratch register
29: ; inputs: none
30: ; outputs: none
31:
32: ; full descending stack
33: UARTConfig
34:
35: STMDB sp!, {r5,r6,lr}
36:
37: LDR r5, = PINSEL0 ; base address of register
38: LDR r6, [r5] ; get contents
39: BIC r6, r6, #0xF ; clear out lower nibble
40: ORR r6, r6, #0x5 ; sets P0.0 to Tx0 and P0.1 to Rx0
41: STR r6, [r5] ; r/modify/w back to register
42:
43: LDR r5, = UOSTART ; 0b1000 1011 = 0x8B
44: MOV r6, #0x8B ; set 8 bits, odd parity, 1 stop bit
45: STRB r6, [r5, #LCR0] ; write control byte to LCR
46:
47: MOV r6, #0xF ; 12800 baud @3 MHz VPB clock
48: STRB r6, [r5] ; store control byte
49:
50: MOV r6, #0xB ; set DLAB = 0
51: STRB r6, [r5, #LCR0] ; Tx and Rx buffers set up
52:
53: LDMIA sp!, {r5,r6,pc}
54: Subroutine Transmit
55: ; This routine puts one byte into the UART
56: ; for transmitting.
57: Register used:

```

**Universal Asynchronous Receive Transmit 0 (UART0)**

**Line Control**

UOLCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

**Interrupt Enable**

UIOIER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

**Divisor Latch**

UODLL: 0x0F

UODLM: 0x00

Baudrate: 12500

**Line Status**

UOLSR: 0x60

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☒ Tx Holding Register Empty (THRE)

☒ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

**Interrupt ID & FIFO Control**

UIOIR/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

**Receiver & Transmitter Registers**

UORBR/THR: 0x00

**Scratch Pad Register**

UOSCR: 0x00

**小算盤**

標準

234.375 ÷ 16 =

**14.6484375**

MC MR M+ M- MS Mv

% CE C <

1/x x² √ √

7 8 9 ×

4 5 6 -

1 2 3 +

+/- 0 . =

**Command**

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVTOFILE DEFINE DIR

**UART #1**

TKU-ECE 612450097 LIN

Call Stack + Locals | UART #1 | Memory 1

Real-Time Agent: Not in target | Simulation | t1: 80.28116683 sec | L9 C:1 | CAP NUM SCRL OVR R/W



(2) to include the declaration of the string “(ID-Name)-Midterm Exam in Spring 2024!” as variable **StudentData**. Use calls to subroutine **Transmit** to do the following 3 steps

(a) display **reversely** the string and **continuously** the string (F5)

Registers

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000000
CPSR	0x000000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC	0x00000000
Mode	Supervisor
States	0
Sec	0.00000000

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

TEST1-2.a.s

```

1      AREA UARTDEMO, CODE, READONLY
2  PINSEL0 EQU 0xE002C000 ; controls the function of the pins
3  UOSTART EQU 0xE000C000 ; start of UART0 registers
4  LCR0    EQU 0xC        ; line control register for UART0
5  LSR0    EQU 0x14       ; line status register for UART0
6  RAMSTART EQU 0x40000020 ; start of onboard RAM for 2104
7
8  start
9
10     LDR    sp, = RAMSTART ; set up stack pointer
11     BL     UARTConfig    ; initialize/configure UART0
12
13     LDR    r1, = StudentData ; starting address of characters
14     ADD    r1, #43
15     MOV    r2, #44
16 Loop1
17     LDRB   r0, [r1], #-1 ; load character, increment address
18     CMP    r2, #0        ; null terminated?
19     BLNE   Transmit      ; send character to UART
20     SUB    r2, #1
21     BNE    Loop1        ; continue if not a '0'
22
23 Loop
24     LDR    r1, = StudentData ; starting address of characters
25     LDRB   r0, [r1], #1 ; load character, increment address
26     CMP    r0, #0        ; null terminated?
27     BLNE   Transmit      ; send character to UART
28     BNE    Loop        ; continue if not a '0'
29 done
30     B      done ; otherwise we are done
31
32 ; Subroutine UARTConfig
33 ; This subroutine configures the I/O pins first. It
34 ; then sets up the UART control register. The
35 ; parameters
36 ; are set to 8 bits, no parity and 1 stop bit

```

Universal Asynchronous Receive Transmitter 0 (UART0)

Line Control: UOLCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

Interrupt Enable: UOIER: 0x00

Interrupt ID & FIFO Control: UOIER/FCR: 0x01

Divisor Latch: UODLL: 0x0E

Baudrate: 13392

Command

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\TEST1-2-a\Obj

UART #1

!4202 qnirpS ni maxE mretdiM-NIL-790054216((612450097-LIN)-Midterm Exam in Spring 2024!

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVTOFILE DEFINE DIR Display Enter

Call Stack + Locals

UART #1

Memory 1

Real-Time Agent: Not in target

Simulation

t1: 147.08366675 sec

L:29 C:25

CAP\_NUM SCRL OVR R/W

按F5的結果會全部輸出出來(紅色底線)，因為程式執行到最後，將Transmit Holding Register和Transmit Shift Register的值傳送至UART#1。(有模擬輸出端)



(2) to include the declaration of the string “(ID-Name)-Midterm Exam in Spring 2024!” as variable **StudentData**. Use calls to subroutine **Transmit** to do the following 3 steps

(a) display **reversely** the string and **continuously** the string (F10)

**Registers**

Register	Value
R0	0x00000000
R1	0x000000C1
R2	0xFFFFFFFF
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x40000020
R14 (LR)	0x00000038
R15 (PC)	0x0000003C
CPSR	0x600000D3
SFSR	0x00000000

**UserSystem**

- Fast Interrupt
- Interrupt
- Supervisor**
- Abort
- Undefined

**Internal**

- PC \$ 0x0000003C
- Mode Supervisor
- States 847749
- Sec 0.07064575

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

```

17      CMP     r2, #0          ; null terminated?
18      BLNE    Transmit        ; send character to UART
19      SUB     r2, #1
20      BNE     Loop1           ; continue if not a '0'
21
22      LDR     r1, = StudentData ; starting address of characters
23
24      Loop:   LDRB    r0, [r1], #1 ; load character, increment address
25              CMP     r0, #0          ; null terminated?
26              BLNE    Transmit        ; send character to UART
27              BNE     Loop            ; continue if not a '0'
28
29      done:   B       done           ; otherwise we e done
30
31      ; Subroutine UARTConfig
32      ; This subroutine configures the I/O pins first. It
33      ; then sets up the UART control register. The
34      ; parameters
35      ; are set to 8 bits, no parity and 1 stop bit.
36      ; Registers used:
37      ; r5 - scratch register
38      ; r6 - scratch register
39      ; inputs: none
40      ; outputs: none
41
42      ; full descending stack
43      UARTConfig
44      STMDB    sp!, {r5, r6, lr}
45
46      LDR     r5, = PINSEL0 ; base address of register
47      LDR     r6, [r5] ; get contents
48      BIC     r6, r6, #0xF ; clear out lower nibble
49      ORR     r6, r6, #0x5 ; sets P0.0 to Tx0 and P0.1 to Rx0
50      STR     r6, [r5] ; r/modify/w back to register
51
52      LDR     r5, = U0START ; 0b1000 1011 = 0x8B
53      MOV     r6, #0x8B ; set 8 bits, odd parity, 1 stop bit
54      STRB    r6, [r5, #LCR0] ; write control byte to LCR
55

```

**Universal Asynchronous Receive Transmit 0 (UART0)**

**Line Control**

UOLCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

**Interrupt Enable**

UOIER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

**Divisor Latch**

UODLL: 0x0E

UODLM: 0x00

Baudrate: 13392

**Line Status**

UOLSR: 0x00

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☒ Tx Holding Register Empty (THRE)

☒ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

**Interrupt ID & FIFO Control**

UOIR/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

**Receiver & Transmitter Registers**

UORBR/THR: 0x00

**Scratch Pad Register**

UOSCR: 0x00

**Command**

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\TEST1-2-a\Obj

UART #1

!4202 gnirpS ni maxE mretdiM-NIL-790054216((612450097-LIN)-Midterm Exam in Spring 202

按F10的結果會有兩個值(4、!)無法輸出出來(紅色框框)，因為程式沒有執行到最後，使4、!兩個字元還存在Transmit Holding Register和Transmit Shift Register裡。(有模擬輸出端)

(2) to include the declaration of the string “(ID-Name)-Midterm Exam in Spring 2024!” as variable **StudentData**. Use calls to subroutine **Transmit** to do the following 3 steps

(a) display **reversely** the string and **continuously** the string (F11)

(highlight the stack elements with the related registers stored for subroutines **Receive** and **Transmit**)

**Registers**

Register	Value
R0	0x00000021
R1	0x000000BE
R2	0x0000002C
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x40000014
R14 (LR)	0x00000020
R15 (PC)	0x0000007C
CPSR	0x200000D3
SFSR	0x00000000
UserSystem	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC \$	0x0000007C
Mode	Supervisor
States	55
Sec	0.00000458

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

```

49      ORR     r6, r6, #0x5 ; sets P0.0 to Tx0 and P0.1 to Rx0
50      STR     r6, [r5] ; r/modify/w back to register
51
52      LDR     r5, =U0START ; 0b1000 1011 = 0x8B
53      MOV     r6, #0x8B ; set 8 bits, odd parity, 1 stop bit
54      STRB    r6, [r5, #LCR0] ; write control byte to LCR
55
56      MOV     r6, #0xE ; 12800 baud @3 MHz VPB clock
57      STRB    r6, [r5] ; store control byte
58
59      MOV     r6, #0xB ; set DLAB = 0
60      STRB    r6, [r5, #LCR0] ; Tx and Rx buffers set up
61
62      LDMIA   sp!, {r5, r6, pc}
63
64 ; Subroutine Transmit
65 ; This routine puts one byte into the UART
66 ; for transmitting.
67 ; Register used:
68 ; r5 - scratch
69 ; r6 - scratch
70 ; inputs: r0- byte to transmit
71 ; outputs: none
72 ;
73
74 ; empty descending stack
75 Transmit
76      STMDA   sp!, {r5, r6, lr}
77      LDR     r5, =U0START
78 wait   LDRB   r6, [r5, #LSR0] ; get status of buffer
79      TST     r6, #0x20 ; buffer empty?
80      BEQ     wait ; spin until buffer's empty
81      STR     r0, [r5]
82      LDMIB   sp!, {r5, r6, pc}
83
84 StudentData DCB "(612450097-LIN)-Midterm Exam in Spring 2024!",0
85      END

```

Universal Asynchronous Receive Transmit 0 (UART0)

Line Control

U0LCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

Interrupt Enable

U0IER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

Divisor Latch

U0DLL: 0x0E

U0DLM: 0x00

Baudrate: 13392

Line Status

U0LSR: 0x60

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☒ Tx Holding Register Empty (THRE)

☒ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

Interrupt ID & FIFO Control

U0IIR/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

Receiver & Transmitter Registers

U0RBR/THR: 0x00

Scratch Pad Register

U0SCR: 0x00

**Command**

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\TEST1-2-a\0b\j

**Memory 1**

Address: 0x40000010

Address	Value
0x40000010	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000011	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000012	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000013	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000014	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000015	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000016	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000017	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000018	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000019	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x4000001A	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x4000001B	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x4000001C	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x4000001D	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x4000001E	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x4000001F	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

(2) to include the declaration of the string “(ID-Name)-Midterm Exam in Spring 2024!” as variable **StudentData**. Use calls to subroutine **Transmit** to do the following 3 steps

(a) display **reversely** the string and **continuously** the string (F11)

**Registers**

Register	Value
R0	0x00000032
R1	0x000000BC
R2	0x0000002A
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x40000014
R14 (LR)	0x00000020
<b>R15 (PC)</b>	<b>0x00000088</b>
CPSR	0x000000D3
SFSR	0x00000000

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

```

48      BIC      r6, r6, #0xF ; clear out lower nibble
49      ORR      r6, r6, #0x5 ; sets P0.0 to Tx0 and P0.1 to Rx0
50      STR      r6, [r5] ; r/modify/w back to register
51
52      LDR      r5, =U0START ; 0b1000 1011 = 0x8B
53      MOV      r6, #0x8B ; set 8 bits, odd parity, 1 stop bit
54      STRB     r6, [r5, #LCR0] ; write control byte to LCR
55
56      MOV      r6, #0xE ; 12800 baud @3 MHz VPB clock
57      STRB     r6, [r5] ; store control byte
58
59      MOV      r6, #0xB ; set DLAB = 0
60      STRB     r6, [r5, #LCR0] ; Tx and Rx buffers set up
61
62      LDMIA     sp!, {r5, r6, pc}
63
64 ; Subroutine Transmit
65 ; This routine puts one byte into the UART
66 ; for transmitting.
67 ; Register used:
68 ; r5 - scratch
69 ; r6 - scratch
70 ; inputs: r0- byte to transmit
71 ; outputs: none
72 ;
73
74 ; empty descending stack
75 Transmit
76      STMDA     sp!, {r5, r6, lr}
77      LDR      r5, =U0START
78 wait   LDRB     r6, [r5, #LSR0] ; get status of buffer
79      TST      r6, #0x20 ; buffer empty?
80      BEQ      wait ; spin until buffer's empty
81      STRB     r0, [r5]
82      LDMIB     sp!, {r5, r6, pc}
83
84 StudentData DCB "(612450097-LIN)-Midterm Exam in Spring 2024!",0
85      END
    
```

**Universal Asynchronous Receive Transmit 0 (UART0)**

**Line Control**

U0LCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

**Interrupt Enable**

U0IER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

**Divisor Latch**

U0DLL: 0x0E

U0DLM: 0x00

Baudrate: 13392

**Line Status**

U0LSR: 0x00

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☒ Tx Holding Register Empty (THRE)

☒ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

**Interrupt ID & FIFO Control**

U0IIR/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

**Receiver & Transmitter Registers**

U0RBR/THR: 0x00

**Scratch Pad Register**

U0SCR: 0x00

**Command**

Running with Code Size Limit: 32K

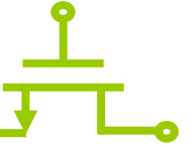
Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\TEST1-2-a\Obj

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR Display Enter

**UART #1**

按F11的結果會全部無法輸出出來，因為沒有模擬輸出端。

(2) (b) display reversely the string characters in multiple-of-3 positions (assuming the last character starts from position 0) (F5)



**Registers**

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000000
CPSR	0x000000D3
SFSR	0x00000000

UserSystem  
Fast Interrupt  
Interrupt  
Supervisor  
Abort  
Undefined  
Internal  
PC \$  
Mode  
States  
Sec

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

```

1  AREA UARTDEMO, CODE, READONLY
2  PINSEL0 EQU 0xE002C000 ; controls the function of the pins
3  UOSTART EQU 0xE000C000 ; start of UART0 registers
4  LCR0 EQU 0xC ; line control register for UART0
5  LSRR EQU 0x14 ; line status register for UART0
6  RAMSTART EQU 0x40000020 ; start of onboard RAM for 2104
7  ENTRY
8
9  start
10     LDR sp, = RAMSTART ; set up stack pointer
11     BL UARTConfig ; initialize/configure UART0
12
13     LDR r1, = StudentData ; starting address of characters
14     ADD r1, #43
15     SUB r1, #3
16     MOV r2, #14
17
18 Loop1
19     LDRB r0, [r1], #-3 ; load character, increment address
20     CMP r2, #0 ; null terminated?
21     BLNE Transmit ; send character to UART
22     SUB r2, #1
23     BNE Loop1 ; continue if not a '0'
24
25 done B done ; otherwise we e done
26
27 ; Subroutine UARTConfig
28 ; This subroutine configures the I/O pins first. It
29 ; then sets up the UART control register. The
30 ; parameters
31 ; are set to 8 bits, no parity and 1 stop bit.
32 ; Registers used:
33 ; r5 - scratch register
34 ; r6 - scratch register
35 ; inputs: none
36 ; outputs: none
37
38 ; full descending stack
39 UARTConfig
40     STMB sp!, {r5,r6,lr}
    
```

**Universal Asynchronous Receive Transmit 0 (UART0)**

**Line Control**

U0LCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

**Interrupt Enable**

U0IER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

**Divisor Latch**

U0DLL: 0x0E

U0DLM: 0x00

Baudrate: 13392

**Line Status**

U0LSR: 0x60

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☒ Tx Holding Register Empty (THRE)

☒ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

**Interrupt ID & FIFO Control**

U0IIR/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

**Receiver & Transmitter Registers**

U0RBR/THR: 0x00

**Scratch Pad Register**

U0SCR: 0x00

**Command**

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\TEST1-2-b\Obj

UART #1

0gr xmtMN-046

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR Display Enter

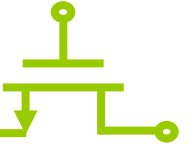
Real-Time Agent: Not in target

Simulation

t1: 29.64308333 sec

L7 C:18

CAP NUM SCRL OVR R/W



**Registers**

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000000
CPSR	0x000000D3
SFSR	0x00000000
UserSystem	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC	0x00000000
Mode	Supervisor
States	0
Sec	0.00000000

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

```

7      ENIR
8      start
9      LDR    sp, = RAMSTART ; set up stack pointer
10     BL     UARTConfig    ; initialize/configure UART0
11
12     ; !
13     LDR    r1, = StudentData ; starting address of characters
14     ADD    r1, #43
15     LDRB   r0, [r1]        ; load character, increment address
16     BL     Transmit        ; send character to UART
17
18     ; 2024
19     LDR    r1, = StudentData ; starting address of characters
20     ADD    r1, #39
21     MOV    r2, #4
22 Loop1
23     LDRB   r0, [r1],#1      ; load character, increment address
24     CMP    r2,#0           ; null terminated?
25     BLNE   Transmit        ; send character to UART
26     SUB    r2, #1
27     BNE    Loop1          ; continue if not a '0'
28
29     ; blank
30     LDR    r1, = StudentData ; starting address of characters
31     ADD    r1, #38
32     LDRB   r0, [r1]        ; load character, increment address
33     BL     Transmit        ; send character to UART
34
35     ; Spring
36     LDR    r1, = StudentData ; starting address of characters
37     ADD    r1, #32
38     MOV    r2, #6
39 Loop2
40     LDRB   r0, [r1],#1      ; load character, increment address
41     CMP    r2,#0           ; null terminated?
42     BLNE   Transmit        ; send character to UART
43     SUB    r2, #1
44     BNE    Loop2          ; continue if not a '0'
45

```

**Universal Asynchronous Receive Transmit 0 (UART0)**

**Line Control**

UOLCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

**Interrupt Enable**

UOIER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

**Divisor Latch**

UODLL: 0x0E

UODLM: 0x00

Baudrate: 13392

**Line Status**

UOLSR: 0x60

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☒ Tx Holding Register Empty (THRE)

☒ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

**Interrupt ID & FIFO Control**

UOIR/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

**Receiver & Transmitter Registers**

UORBR/THR: 0x00

**Scratch Pad Register**

UOSCR: 0x00

**Command**

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\TEST1-2-c\Obj

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVTOFILE DEFINE DIR Display Enter

**UART #1**

!2024 Spring in Exam Midterm-)LIN-612450097(

Call Stack + Locals | UART #1 | Memory 1

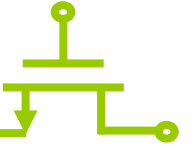
Real-Time Agent: Not in target | Simulation | t1: 39.12091683 sec | L9 C:1 | CAP NUM SCRL OVR R/W

提示：把單字、空格、符號分開來處理。  
(須找到每個單字、空格、符號的起始位置)



(3) to include subroutines **Receive** (using **empty ascending stack** with initial stack pointer 0x40000020, to STM and LDM in the subroutine) to receive an **error-free** byte data from the receiver buffer register to R1.

(a) to copy the string (variable **StudentData**) **reversely** to memory starting from address 0x40000070.



The screenshot displays the Keil uVision IDE interface during a simulation. The main window shows assembly code for a program that initializes UART0, sets up a stack pointer, and implements two loops: Loop1 for transmitting a string and Loop2 for receiving data.

**Registers Window:**

Register	Value
R0	0x00000000
R1	0x000000E3
R2	0x00000000
R3	0x4000009C
R4	0x00000000
R5	0x00000028
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x40000020
R14 (LR)	0x00000008
R15 (PC)	0x00000034
CPSR	0x600000D3
SFSR	0x00000000
UserSystem	
Past Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC \$	0x00000034
Mode	Supervisor
Status	574
Sec	0.00004783

**Assembly Code:**

```
offset 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
offset(re) 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
string ( 6 1 2 4 5 0 0 9 7 - L I N ) - M i d t e r m E x a m i n S p r i n g 2 0 2 4 !
ascii(HEX) 28 36 31 32 34 35 30 30 39 37 2D 4C 49 4E 29 2D 4D 69 64 74 65 72 6D 45 78 61 6D 69 6E 53 70 72 69 6E 67 32 30 32 34 21

3 U0START EQU 0xE000C000 ; start of UART0 registers
4 LCRO EQU 0xC ; line control register for UART0
5 LSRO EQU 0x14 ; line status register for UART0
6 RAMSTART EQU 0x40000020 ; start of onboard RAM for 2104
7
8 ENTRY
9 start
10 LDR sp, =RAMSTART ; set up stack pointer
11 BL UARTConfig ; initialize/configure UART0
12 ; 1-(3)-(a)
13 LDR r3, =0x40000070
14 LDR r1, =StudentData ; starting address of characters
15 ADD r1, #43
16 MOV r2, #44
17 MOV r4, #0
18 Loop
19 LDRB r5, [r1], #-1
20 STRB r5, [r3], #1
21 SUB r2, #1
22 CMP r2, #0
23 STRB r4, [r3]
24 BNE Loop
25
26 ; 1-(3)-(b)
27 LDR r1, =0x40000078
28 MOV r2, #10
29 Loop1
30 LDRB r0, [r1], #1 ; load character, increment address
31 CMP r2, #0 ; null terminated?
32 BLNE Transmit ; send character to UART
33 SUB r2, #1
34 BNE Loop1 ; continue if not a '0'
35
36 ; 1-(3)-(c)
37 MOV r9, #20
38 LDR r1, =0x400000A0
39 Loop2
40 BL Receive
```

**Memory Window:**

Address: 0x40000070

0x40000070:	21 34 32 30 32 20 67 6E 69 72 70 53 20 6E 69 20 6D 61 78 45 20 6D 72 65 74 64 69 4D 2D 29 4E 49 4C 2D
0x40000092:	37 39 30 30 35 34 32 31 36 28 00
0x400000B4:	00 00
0x400000D6:	00 00

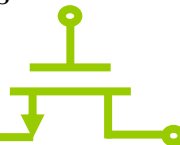
**Command Window:**

Running with Code Size Limit: 32K  
Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\TEST1-3\Objec

**Simulation Status:**

Real-Time Agent: Target Stopped | Simulation | t1: 0.00004783 sec | L:27 C:1 | CAP NUM SCRL OVR R/W

(3) (b) to use calls to subroutine **Transmit** to display a sequence of **10** characters at memory address **0x40000078** in the **UART #1** window after program execution by using **F5 (Run)**.



**Registers**

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000000
CPSR	0x000000D3
SFSR	0x00000000
UserSystem	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC \$	0x00000000
Mode	Supervisor
States	0
Sec	0.00000000

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

```

1  AREA UARTDEMO, CODE, READONLY
2  PINSEL0 EQU 0xE002C000 ; controls the function of the pins
3  UOSTART EQU 0xE000C000 ; start of UART0 registers
4  LCR0 EQU 0xC ; line control register for UART0
5  LSR0 EQU 0x14 ; line status register for UART0
6  RAMSTART EQU 0x40000020 ; start of onboard RAM for 2104
7  ENTRY
8
9  start
10 LDR sp, = RAMSTART ; set up stack pointer
11 BL UARTConfig ; initialize/configure UART0
12
13 ; 1-(3)-(a)
14 LDR r3, =0x40000070
15 LDR r1, = StudentData ; starting address of characters
16 ADD r1, #43
17 MOV r2, #44
18 MOV r4, #0
19
20 Loop
21 LDRB r5, [r1], #-1
22 STRB r5, [r3], #1
23 SUB r2, #1
24 CMP r2, #0
25 BNE Loop
26
27 ; 1-(3)-(b)
28 LDR r1, =0x40000078
29 MOV r2, #10
30
31 Loop1
32 LDRB r0, [r1], #1 ; load character, increment address
33 CMP r2, #0 ; null terminated?
34 BLNE Transmit ; send character to UART
35 SUB r2, #1
36 BNE Loop1 ; continue if not a '0'
37
38 ; 1-(3)-(c)
39 MOV r9, #20
40 LDR r1, =0x400000A0

```

**Universal Asynchronous Receive Transmit 0 (UART0)**

**Line Control**

UOLCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

**Interrupt Enable**

UIER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

**Divisor Latch**

UODLL: 0x0E

UODLM: 0x00

Baudrate: 13392

**Line Status**

UOLSR: 0x60

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☒ Tx Holding Register Empty (THRE)

☒ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

**Interrupt ID & FIFO Control**

UIIR/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

**Receiver & Transmitter Registers**

UORBR/THR: 0x00

**Scratch Pad Register**

UOSCR: 0x00

**UART #1**

lrpS ni ma

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\TEST1-3\Objec

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR Display Enter

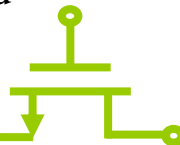
Call Stack + Locals | UART #1 | Memory 1

Real-Time Agent: Not in target | Simulation | t1: 64.43034475 sec | L9 C:1 | CAP NUM SCRL OVR R/W

按F5的結果會全部輸出出來(紅色底線)，因為程式執行到最後，將Transmit Holding Register和Transmit Shift Register的值傳送至UART#1。(有模擬輸出端)



(3) (c) to use calls to subroutine **Receive** to receive a sequence of **error-free 20** characters from the UART0 and put them in memory starting from address 0x400000A0.(F10未進Receive)



Registers

Register	Value
R0	0x00000028
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000014
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x40000020
R14 (LR)	0x00000048
R15 (PC)	0x00000058
CPSR	0x000000D3
SFSR	0x00000000
PC \$	0x00000058
Mode	Supervisor
States	79477
Sec	0.00662308

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

TEST1-3.s

```

24      BNE      Loop
25
26      ; 1-(3)-(b)
27      LDR      r1, =0x40000078
28      MOV      r2, #10
29 Loop1
30      LDRB     r0, [r1], #1 ; load character, increment address
31      CMP      r2, #0 ; null terminated?
32      BLNE     Transmit ; send character to UART
33      SUB      r2, #1
34      BNE      Loop1 ; continue if not a '0'
35
36      ; 1-(3)-(c)
37      MOV      r9, #20
38      LDR      r0, =0x400000A0
39 Loop2
40      BL       Receive
41      STRB     r1, [r0], #1
42      SUBS     r9, r9, #1
43      BNE      Loop2
44
45 done    B      done ; otherwise we e done
46
47 ; Subroutine UARTConfig
48 ; This subroutine configures the I/O pins first. It
49 ; then sets up the UART control register. The
50 ; parameters
51 ; are set to 8 bits, no parity and 1 stop bit.
52 ; Registers used:
53 ; r5 - scratch register
54 ; r6 - scratch register
55 ; inputs: none
56 ; outputs: none
                
```

Universal Asynchronous Receive Transmitter 0 (UART0)

Line Control

UOLCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

Interrupt Enable

UOIER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

Divisor Latch

UODLL: 0x0E

UODLM: 0x00

Baudrate: 13392

Line Status

UOLSR: 0x00

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☐ Tx Holding Register Empty (THRE)

☐ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

Interrupt ID & FIFO Control

UOIER/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

Receiver & Transmitter Registers

UORBR/THR: 0x00

Scratch Pad Register

UOSCR: 0x00

按F10的結果會有兩個值(m、a)無法輸出出來(紅色框框)，因為程式沒有執行到最後，使m、a兩個字元還存在Transmit Holding Register和Transmit Shift Register裡。(有模擬輸出端)

Command

Running with Code Size Limit: 32K

Load "F:\\03.淡江碩士\\01.碩一(112)\\02.碩一下學期\\07.微處理機概論(電資)(助教課)\\06.期中考\\02.解答\\

UART #1

irpS ni

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR

Real-Time Agent: Target Stopped Simulation t1: 0.00662308 sec L:40 C:1 CAP NUM SCRL OVR R/W

P. W. LIN

2024 Tamkang University

16

(3) (c) to use calls to subroutine **Receive** to receive a sequence of **error-free 20** characters from the UART0 and put them in memory starting from address 0x400000A0.(F10)



Registers

Register	Value
R0	0x00000028
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000014
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000020
R10	0x00000048
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000058
CPSR	0x000000D3
SFSR	0x00000000

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

**Assembly Code (TEST1-3.s):**

```

18 Loop
19     LDRB    r5, [r1], #-1
20     STRB    r5, [r3], #1
21     SUB     r2, #1
22     CMP     r2, #0
23     STRB    r4, [r3]
24     BNE     Loop
25
26     ; 1-(3)-(b)
27     LDR     r1, =0x40000078
28     MOV     r2, #10
29
30 Loop1
31     LDRB    r0, [r1], #1 ; load character, increment address
32     CMP     r2, #0 ; null terminated?
33     BLNE    Transmit ; send character to UART
34     SUB     r2, #1
35     BNE     Loop1 ; continue if not a '0'
36
37     ; 1-(3)-(c)
38     MOV     r9, #20
39     LDR     r0, =0x400000A0
40 Loop2
41     BL      Receive
42     STRB    r1, [r0], #1
43     SUBS    r9, r9, #1
44     BNE     Loop2
45 done    B     done ; otherwise we e done
46
47 ; Subroutine UARTConfig
48 ; This subroutine configures the I/O pins first. It
49 ; then sets up the UART control register. The
50 ; parameters

```

**Universal Asynchronous Receive Transmitter 0 (UART0) Configuration:**

- Line Control: UOLCR: 0x0B, Word Length: 8 bits, Stop Bits: 1, Parity: Odd Parity
- Line Status: UOLSR: 0x60, ☒ Tx Holding Register Empty (THRE), ☒ Transmitter Empty (TEMT)
- Interrupt Enable: UOIER: 0x00, ☐ RBR IE, ☐ THRE IE, ☐ Rx Line Status IE
- Divisor Latch: UODLL: 0x0E, UODLM: 0x00, Baudrate: 13392
- Receiver & Transmitter Registers: UORBR/THR: 0x00, UOSCR: 0x00

**Command Window:**

```

Running with Code Size Limit: 32K
Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\

```

**UART #1 Window:**

```

lrxS ni ma

```

按F10的結果會全部輸出出來(紅色框框)，因為程式重新存取LSR0暫存器，使m、a兩個字元從Transmit Holding Register和Transmit Shift Register裡輸出到UART#1視窗。(有模擬輸出端)

而程式則會在Receive副函式中無窮迴圈，因為PDR值皆為0，因此無法接收任何字元。

(3) (c) to use calls to subroutine **Receive** to receive a sequence of **error-free 20** characters from the UART0 and put them in memory starting from address 0x400000A0.(F11)  
(highlight the stack elements with the related registers stored for subroutines Receive and Transmit)



**Registers**

Register	Value
R0	0x00000028
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x4000002C
R14 (LR)	0x00000040
R15 (PC)	0x000000A8
CPSR	0x000000D3
SFSR	0x00000000
UserSystem	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC \$	0x000000A8
Mode	Supervisor
States	585
Sec	0.00004875

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

**TEST1-3.s**

```

84: r5 - scratch
85: r6 - scratch
86: inputs: r0- byte to transmit
87: outputs: none
88:
89:
90: empty descending stack
91 Transmit
92     STMDA sp!, {r5, r6, lr}
93     LDR r5, =U0START
94 wait
95     LDRB r6, [r5, #LSR0]; get status of buffer
96     TST r6, #0x20; buffer empty?
97     BEQ wait; spin until buffer's empty
98     STRB r0, [r5]
99     LDMIB sp!, {r5, r6, pc}
100: empty ascending stack
101 Receive
102     STMDA sp!, {r5, r6, lr}
103     LDR r5, =U0START
104 wait1
105     LDRB r6, [r5, #LSR0]
106     TST r6, #1
107     BEQ wait1
108     TST r6, #0xE
109     LDRB r1, [r5]
110     BNE wait1
111     LDMDB sp!, {r5, r6, pc}
112
113 StudentData
114     DCB "(612450097-LIN)-Midterm Exam in Spring 2024!",0
115     END
          
```

**Universal Asynchronous Receive Transmitter 0 (UART0)**

**Line Control**

U0LCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

**Interrupt Enable**

U0IER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

**Divisor Latch**

U0DLL: 0x0E

U0DLM: 0x00

Baudrate: 13392

**Line Status**

U0LSR: 0x60

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☒ Tx Holding Register Empty (THRE)

☒ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

**Interrupt ID & FIFO Control**

U0IIR/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

**Receiver & Transmitter Registers**

U0RBR/THR: 0x00

**Scratch Pad Register**

U0SCR: 0x00

**Memory 1**

Address: 0x40000020

Address	R5	R6	LR
0x40000020:	28 00 00 00	00 00 00 00	40 00 00 00
0x4000003D:	00 00 00 00	00 00 00 00	00 00 00 00
0x4000005A:	00 00 00 00	00 00 00 00	00 00 00 00
0x40000077:	6E 69 72 70	53 20 6E 69	20 6D 61 78
0x40000094:	30 30 35 34	32 31 36 28	00 00 00 00

**Command**

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\

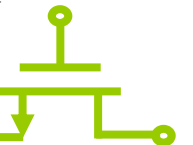
**ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR**

Real-Time Agent: Target Stopped

Simulation

t1: 0.00004875 sec L:103 C:1 CAP NUM SCRL OVR R/W

(3) (c) to use calls to subroutine **Receive** to receive a sequence of **error-free 20** characters from the UART0 and put them in memory starting from address 0x400000A0.(F11)



**Registers**

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x40000014
R14 (LR)	0x00000048
<b>R15 (PC)</b>	<b>0x000000AC</b>
CPSR	0x000000D3
SFSR	0x00000000

offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
offset(re)	43	42	41	40	39	38	37	36	35	34	33	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
string	(	6	1	2	4	5	0	0	9	7	-	L	I	N	)	-	M	i	d	t	e	r	m		E	x	a	m		i	n		S	p	r	i	n	g		2	0	2	4	!
ascii(HEX)	28	36	31	32	34	35	30	30	39	37	2D	4C	49	4E	29	2D	4D	69	64	74	65	72	6D		45	78	61	6D		69	6E		53	70	72	69	6E	67		32	30	32	34	21

**TEST1-3.s**

```

84: r5 - scratch
85: r6 - scratch
86: inputs: r0- byte to transmit
87: outputs: none
88:
89:
90: empty descending stack
91 Transmit
92
93         SIMDA    sp!, {r5, r6, lr}
          LDR      r5, =U0START
94 wait    LDRB     r6, [r5, #LSR0] ; get status of buffer
95         TST      r6, #0x20      ; buffer empty?
96         BEQ      wait          ; spin until buffer's empty
97         STRB     r0, [r5]
98         LDMIB    sp!, {r5, r6, pc}
99
100: empty ascending stack
101 Receive
102
103         SIMIA    sp!, {r5, r6, lr}
          LDR      r5, =U0START
104 wait1
105
106         LDRB     r6, [r5, #LSR0]
107         TST      r6, #1
108         BEQ      wait1
109         TST      r6, #0xE
110         LDRB     r1, [r5]
111         BNE      wait1
112         LDMDB    sp!, {r5, r6, pc}
113 StudentData
114         DCB      "(612450097-LIN)-Midterm Exam in Spring 2024!",0
115         END

```

**Universal Asynchronous Receive Transmitter 0 (UART0)**

**Line Control**

U0LCR: 0x0B

Word Length: 8 bits

Stop Bits: 1

Parity: Odd Parity

☐ DLAB

☐ Break Control

☒ Parity Enable

**Interrupt Enable**

U0IER: 0x00

☐ RBR IE

☐ THRE IE

☐ Rx Line Status IE

**Divisor Latch**

U0DLL: 0x0E

U0DLM: 0x00

Baudrate: 13392

**Line Status**

U0LSR: 0x00

☐ Receiver Data Ready (RDR)

☐ Overrun Error (OE)

☐ Parity Error (PE)

☐ Framing Error (FE)

☐ Break Interrupt (BI)

☒ Tx Holding Register Empty (THRE)

☒ Transmitter Empty (TEMT)

☐ Error in Rx FIFO (RXFE)

**Interrupt ID & FIFO Control**

U0IIR/FCR: 0x01 ☐ FIFO Enable

Interrupt: None

Rx Trigger: Level 0 (1)

☐ Rx FIFO Reset ☐ Tx FIFO Reset

**Receiver & Transmitter Registers**

U0RBR/THR: 0x00

**Scratch Pad Register**

U0SCR: 0x00

**Command**

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\07.微處理機概論(電資)(助教課)\06.期中考\02.解答\

**UART #1**


按F11的結果會全部無法輸出出來，因為沒有模擬輸出端。

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR

Call Stack + Locals | **UART #1** | Memory 1

Real-Time Agent: Target Stopped | Simulation | t1: 0.00007158 sec | L:94 C:1 | CAP NUM SCRL OVR R/W

P. W. LIN



2024 Tamkang University

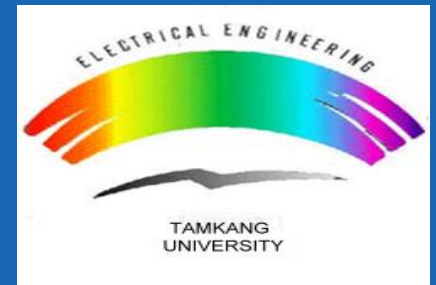
19

2. Rewrite Program 15-1 to include the following 2 declarations and  
MSG\_with\_Error DCB “DIVIDE-BY-0 Happened!”, 0  
MSG\_without\_Error DCB “DIVIDE-BY-0 Not Happened!”, 0

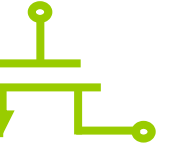
**2024 Advanced Mixed-Operation System (AMOS) Lab.**



**Tamkang University**  
**Department of Electrical and Computer Engineering**  
No.151, Yingzhuang Rd., Tamsui Dist., New Taipei City 25137, Taiwan (R.O.C.)

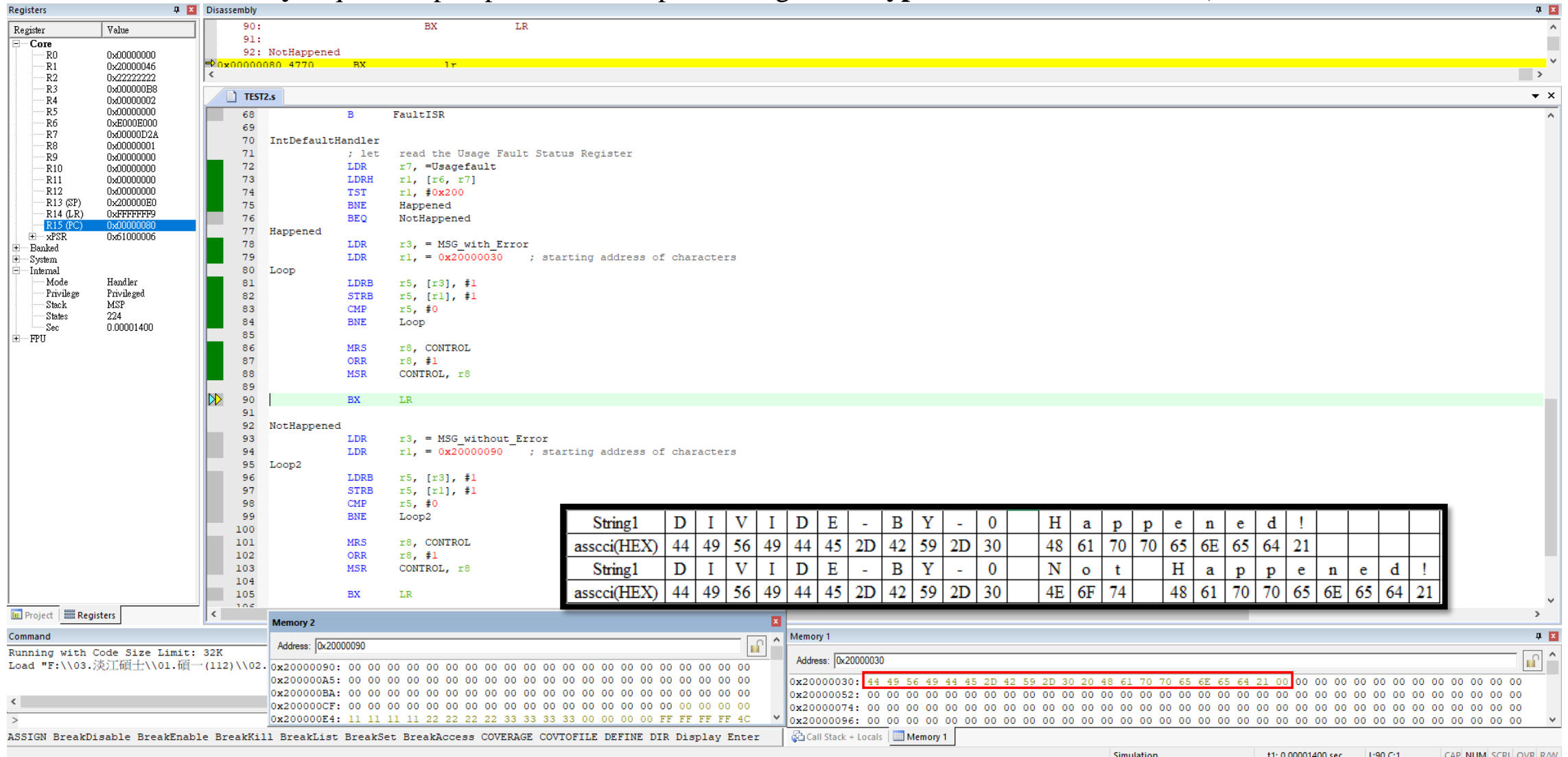






0x1C	xPSR
0x18	ReturnAddress
0x14	LR (R14)
0x10	R12
0x0C	R3
0x08	R2
0x04	R1
0x00	R0

(1) check the usage fault status register, write string MSG\_with\_Error to memory with starting address 0x20000030 if a divide-by-zero has taken place, and write string MSG\_without\_Error to memory with starting address 0x20000090 if a divide-by-zero has not taken place.(Be sure to show the first step (stacking) in the entry sequence upon processor exception and give the type of the stack used here.)



The screenshot shows a debugger interface with the following components:

- Registers:** A list of registers (R0-R15, xPSR) and their values. R15 (PC) is highlighted with the value 0x00000080.
- Disassembly:** Assembly code for the `TEST2.s` file. The code includes a `FaultISR` function that checks the Usage Fault Status Register (UFSR) and writes strings to memory based on the result. The current instruction is `BX LR` at address 0x20000090.
- Memory 2:** A window showing memory contents starting at address 0x20000090. The first few bytes are highlighted in red.
- Memory 1:** A window showing memory contents starting at address 0x20000030. The first few bytes are highlighted in red.
- String Tables:** Two tables showing the ASCII values of the strings `MSG_with_Error` and `MSG_without_Error`.

String1	D	I	V	I	D	E	-	B	Y	-	0		H	a	p	p	e	n	e	d	!				
ascii(HEX)	44	49	56	49	44	45	2D	42	59	2D	30		48	61	70	70	65	6E	65	64	21				

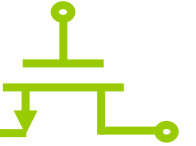
String1	D	I	V	I	D	E	-	B	Y	-	0		N	o	t		H	a	p	p	e	n	e	d	!
ascii(HEX)	44	49	56	49	44	45	2D	42	59	2D	30		4E	6F	74		48	61	70	70	65	6E	65	64	21





(2) to switch modes and show the mode changes

(a) from **privileged thread mode** to **unprivileged thread mode**(1/2)



Registers

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000100
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x0000001C
xPSR	0x01000000

Disassembly

```
29: MRS r8, CONTROL
0x0000001C F3EF8814 MRS r8,CONTROL
30: ORR r8, #1
0x00000020 F0480A01 ORR r8,r8,#0x01
```

TEST2.s

```
17 DCD StackMem + Stack ; Top of Stack
18 DCD Reset_Handler ; Reset Handler
19 DCD NmiISR ; NMI Handler
20 DCD FaultISR ; Hard Fault Handler
21 DCD IntDefaultHandler ; MPU Fault Handler
22 DCD IntDefaultHandler ; Bus Fault Handler
23 DCD IntDefaultHandler ; Usage Fault Handler
24 EXPORT Reset_Handler
25 ENTRY
26
27 Reset_Handler
28
29 MRS r8, CONTROL
30 ORR r8, #1
31 MSR CONTROL, r8
32 ; enable the divide-by-zero trap
33 ; located in the NVIC
34 ; base: 0xE000E000
35 ; offset: 0xD14
36 ; bit: 4
37 ; ADR r11, StackMem
38 LDR r6, =NVICBase
39 LDR r7, =Divby2
40 LDR r1, [r6, r7]
41 ORR r1, #0x10 ; enable bit 4
42 STR r1, [r6, r7]
43 ; now turn on the usage fault exception
44 LDR r7, =SYSHNDCTRL ; p. 163
45 LDR r1, [r6, r7]
46 ORR r1, #0x40000
47 STR r1, [r6, r7]
48 ; try out a divide by 2 then a divide by 0!
49 MOV r0, #0
50 MOV r1, #0x11111111
51 MOV r2, #0x22222222
52 MOV r3, #0x33333333
53 ; this divide works just fine
54 UDIV r4, r2, r1
55 ; this divide takes an exception
```

Memory 2

Address: 0x20000090

Memory 1

Address: 0x20000030

Simulation

t1: 0.00000000 sec L:29 C:1 CAP NUM SCRL OVR R/V



(2) to switch modes and show the mode changes

(a) from **privileged thread mode** to **unprivileged thread mode**(2/2)



**Registers**

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000001
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000100
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x00000028
xPSR	0x01000000

**Disassembly**

```

38:      LDR    r6, =NVICBase
0x00000028: F04F26E0  MOV    r6, #0xE000E000
39:      LDR    r7, =Divby2
0x0000002C: F6405714  MOVW   r7, #0xD14
26:
27: Reset_Handler
28:
29:      MRS    r8, CONTROL
30:      ORR    r8, #1
31:      MSR    CONTROL, r8
32:      ; enable the divide-by-zero trap
33:      ; located in the NVIC
34:      ; base: 0xE000E000
35:      ; offset: 0xD14
36:      ; bit: 4
37:      ; ADDR    r11, StackMem
38:      LDR    r6, =NVICBase
39:      LDR    r7, =Divby2
40:      LDR    r1, [r6, r7]
41:      ORR    r1, #0x10      ; enable bit 4
42:      STR    r1, [r6, r7]
43:      ; now turn on the usage fault exception
44:      ; p. 163
45:      LDR    r7, =SYSHNDCTRL
46:      LDR    r1, [r6, r7]
47:      ORR    r1, #0x400000
48:      STR    r1, [r6, r7]
49:      ; try out a divide by 2 then a divide by 0!
50:      MOV    r0, #0
51:      MOV    r1, #0x11111111
52:      MOV    r2, #0x22222222
53:      MOV    r3, #0x33333333
54:      ; this divide works just fine
55:      UDIV   r4, r2, r1
56:      ; this divide takes an exception
57:      UDIV   r5, r3, r0
58:      ; Exception Entry:
59:      ; 1. Stacking
60:      ; 2. Interrupt Vector Lookup
61:      ; 3. LR
62:      Exit
63:      B      Exit
64:

```

**Memory 2**

Address: 0x20000090

```

0x20000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000A5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000BA: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000CF: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000E4: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

**Memory 1**

Address: 0x20000030

```

0x20000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000052: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000074: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000096: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

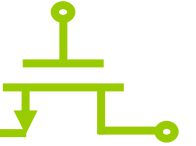
```

**Command**

Running with Code Size Limit: 32K  
Load "F:\03.淡江碩士\01.碩一(112)\02.

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR Display Enter

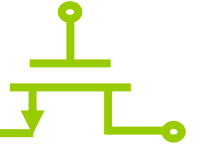
Simulation t1: 0.00000025 sec L:38 C:1 CAP NUM SCRL OVR R/V



25

(2) to switch modes and show the mode changes

(b) from **privileged thread mode** to **privileged handler mode**(2/2)



Register

Value

Register	Value
R0	0x00000000
R1	0x11111111
R2	0x22222222
R3	0x33333333
R4	0x00000002
R5	0x00000000
R6	0xE000E000
R7	0x0000D24
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x200000E0
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x00000056
xPSR	0x01000006

Banked

System

Internal

Mode

Privilege

Stack

States

Sec

Handler

Privileged

MSF

35

0.00000219

FPU

Disassembly

72: LDR r7, =Usagefault

0x00000056 F640572A MOVW r7, #0xD2A

73: LDRH r1, [r6, r7]

0x0000005A 5BFF1 LDRH r1, [r6, r7]

TEST2.s

60 ; 3. LR

61 Exit

62 B Exit

63

64 NmiISR

65 B NmiISR

66

67 FaultISR

68 B FaultISR

69

70 IntDefaultHandler

71 ; let read the Usage Fault Status Register

72 LDR r7, =Usagefault

73 LDRH r1, [r6, r7]

74 TST r1, #0x200

75 BNE Happened

76 BEQ NotHappened

77 Happened

78 LDR r3, =MSG\_with\_Error

79 LDR r1, =0x20000030 ; starting address of characters

80 Loop

81 LDRB r5, [r3], #1

82 STRB r5, [r1], #1

83 CMP r5, #0

84 BNE Loop

85

86 MRS r8, CONTROL

87 ORR r8, #1

88 MSR CONTROL, r8

89

90 BX LR

91

92 NotHappened

93 LDR r3, =MSG\_without\_Error

94 LDR r1, =0x20000090 ; starting address of characters

95 Loop2

96 LDRB r5, [r3], #1

97 STRB r5, [r1], #1

98 CMP r5, #0

99 BNE Loop2

100

Memory 2

Address: 0x20000090

0x20000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000A5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000BA: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000CF: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000E4: 11 11 11 11 22 22 22 22 33 33 33 33 00 00 00 FF FF FF 4C

Memory 1

Address: 0x20000030

0x20000030: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x20000052: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x20000074: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x20000096: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Command

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.

ASSIGN

BreakDisable

BreakEnable

BreakKill

BreakList

BreakSet

BreakAccess

COVERAGE

COVTOFILE

DEFINE

DIR

Display

Enter

Simulation

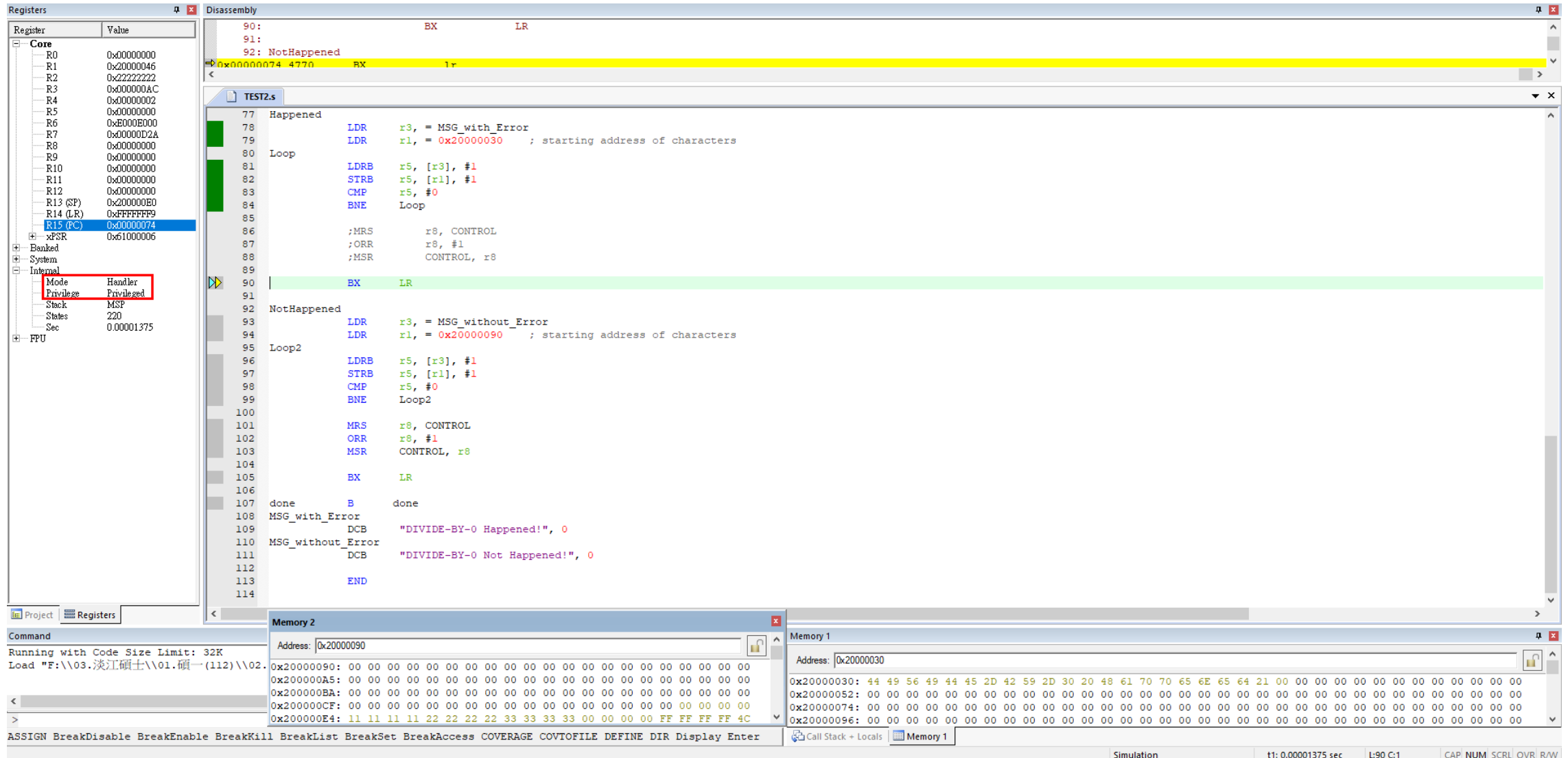
t1: 0.00000219 sec

L72 C:1

CAP NUM SCRL OVR R/W

(2) to switch modes and show the mode changes

(c) from **privileged handler mode** to **privileged thread mode**(1/2)

**Registers**

Register	Value
R0	0x00000000
R1	0x20000046
R2	0x22222222
R3	0x000000AC
R4	0x00000002
R5	0x00000000
R6	0xE000E000
R7	0x0000D2A
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x200000E0
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x00000074
xPSR	0x61000006

**Disassembly**

```

90:          BX      LR
91:
92: NotHappened
93:          LDR      r3, =MSG_without_Error
94:          LDR      r1, =0x20000090 ; starting address of characters
95: Loop2
96:          LDRB     r5, [r3], #1
97:          STRB     r5, [r1], #1
98:          CMP      r5, #0
99:          BNE      Loop2
100:
101:          MRS      r8, CONTROL
102:          ORR      r8, #1
103:          MSR      CONTROL, r8
104:
105:          BX      LR
106:
107: done      B       done
108: MSG_with_Error
109:          DCB      "DIVIDE-BY-0 Happened!", 0
110: MSG_without_Error
111:          DCB      "DIVIDE-BY-0 Not Happened!", 0
112:
113:          END
114:

```

**Memory 2**

Address: 0x20000090

```

0x20000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000A5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000BA: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000CF: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000E4: 11 11 11 11 22 22 22 22 33 33 33 33 00 00 00 00

```

**Memory 1**

Address: 0x20000030

```

0x20000030: 44 49 56 49 44 45 2D 42 59 2D 30 20 48 61 70 70 65 6E 65 64 21 00 00 00 00 00 00 00 00 00 00
0x20000052: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000074: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000096: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

**Command**

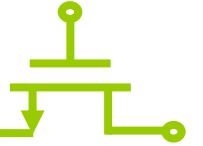
Running with Code Size Limit: 32K  
Load "F:\03.淡江碩士\01.碩一(112)\02.

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVTOFILE DEFINE DIR Display Enter

Simulation t1: 0.00001375 sec L:90 C:1 CAP NUM SCRL OVR R/V

(2) to switch modes and show the mode changes

(c) from **privileged handler mode** to **privileged thread mode**(2/2)



Register

Value

Register	Value
R0	0x00000000
R1	0x11111111
R2	0x22222222
R3	0x33333333
R4	0x00000002
R5	0x00000000
R6	0xE000E000
R7	0x0000D2A
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000100
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x0000004C
xPSR	0x01000000

Banked

System

Internal

Mode

Thread

Privilege

Privileged

Stack

MSF

States

231

Sec

0.00001444

Project

Registers

56: UDIV r5, r3, r0

57: ; Exception Entry:

58: ; 1. Stacking

59: ; 2. Interrupt Vector Lookup

TEST2.s

44 LDR r7, =SYSHNDCTRL ; p. 163

45 LDR r1, [r6, r7]

46 ORR r1, #0x40000

47 STR r1, [r6, r7]

48 ; try out a divide by 2 then a divide by 0!

49 MOV r0, #0

50 MOV r1, #0x11111111

51 MOV r2, #0x22222222

52 MOV r3, #0x33333333

53 ; this divide works just fine

54 UDIV r4, r2, r1

55 ; this divide takes an exception

56 UDIV r5, r3, r0

57 ; Exception Entry:

58 ; 1. Stacking

59 ; 2. Interrupt Vector Lookup

60 ; 3. LR

61 Exit

62 B Exit

63

64 NmiISR

65 B NmiISR

66

67 FaultISR

68 B FaultISR

69

70 IntDefaultHandler

71 ; let read the Usage Fault Status Register

72 LDR r7, =Usagefault

73 LDRH r1, [r6, r7]

74 TST r1, #0x200

75 BNE Happened

76 BEQ NotHappened

77 Happened

78 LDR r3, =MSG\_with\_Error

79 LDR r1, =0x20000030 ; starting address of characters

80 Loop

81 LDRB r5, [r3], #1

82 STNB r5, [r1], #1

Memory 2

Address: 0x20000090

0x20000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000A5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000BA: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000CF: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000E4: 11 11 11 11 22 22 22 22 33 33 33 33 00 00 00 00

Memory 1

Address: 0x20000030

0x20000030: 44 49 56 49 44 45 2D 42 59 2D 30 20 48 61 70 70 65 6E 65 64 21 00 00 00 00 00 00 00 00 00 00 00 00

0x20000052: 00

0x20000074: 00

0x20000096: 00

Command

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVERAGE COVTOFILE DEFINE DIR Display Enter

Call Stack + Locals

Memory 1

Simulation

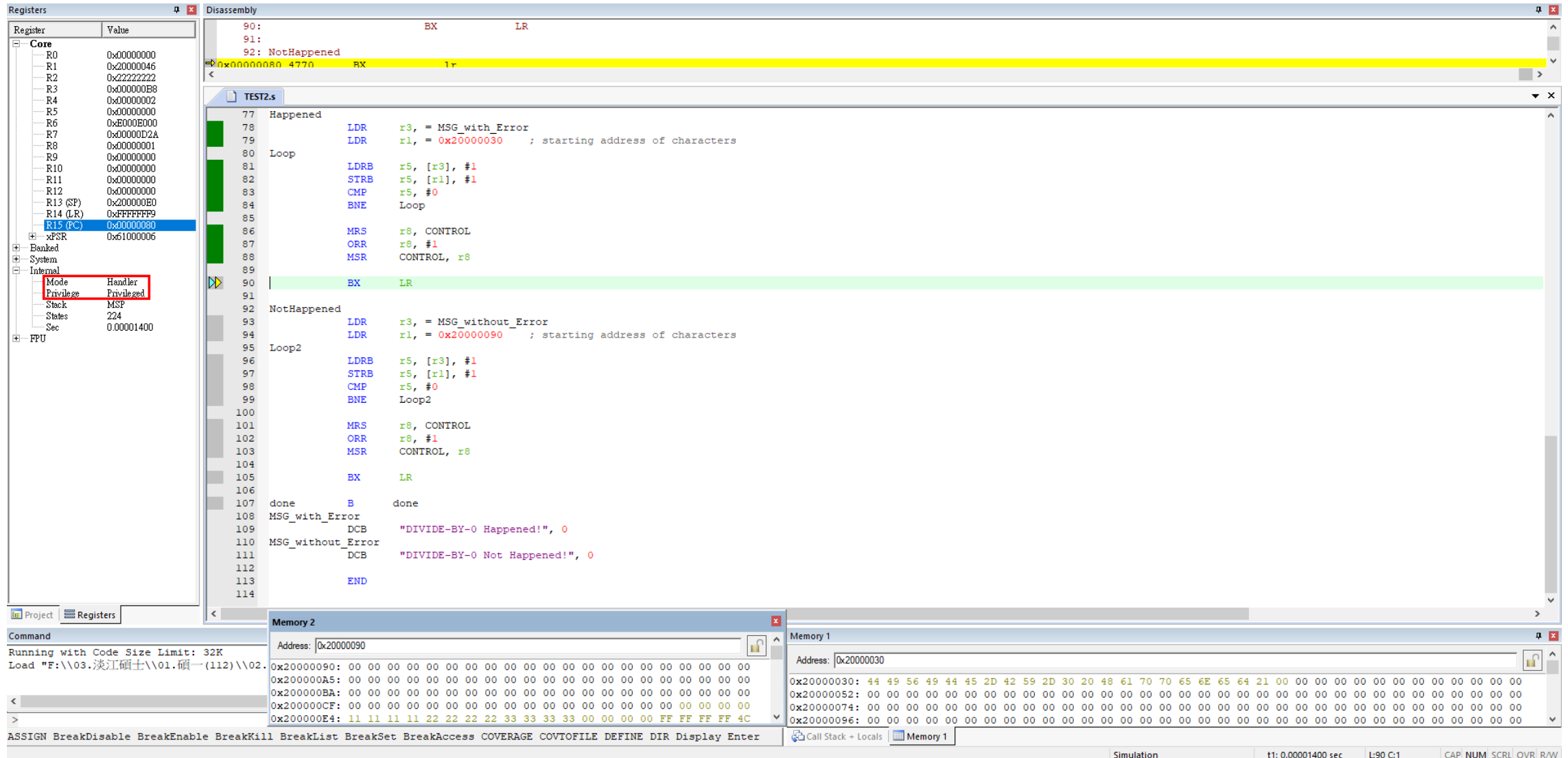
t1: 0.00001444 sec

L56 C:1

CAP NUM SCRL OVR R/V

(2) to switch modes and show the mode changes

(d) from **privileged handler mode** to **unprivileged thread mode**(1/2)

**Registers**

Register	Value
R0	0x00000000
R1	0x20000046
R2	0x22222222
R3	0x000000B8
R4	0x00000002
R5	0x00000000
R6	0xE000E000
R7	0x0000D2A
R8	0x00000001
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x200000E0
R14 (LR)	0xFFFFFFFF
<b>R15 (PC)</b>	<b>0x00000080</b>
xPSR	0x61000006

**Disassembly**

```

90:          BX      LR
91:
92: NotHappened
93: 0x00000080 4770  BX      LR
94:
95:
96:
97:
98:
99:
100:
101:
102:
103:
104:
105:
106:
107: done      B      done
108: MSG_with_Error
109:          DCB      "DIVIDE-BY-0 Happened!", 0
110: MSG_without_Error
111:          DCB      "DIVIDE-BY-0 Not Happened!", 0
112:
113:          END
114:

```

**Memory 2**

Address: 0x20000090

```

0x20000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000A5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000BA: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000CF: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x200000E4: 11 11 11 11 22 22 22 22 33 33 33 33 00 00 00 00

```

**Memory 1**

Address: 0x20000030

```

0x20000030: 44 49 56 49 44 45 2D 42 59 2D 30 20 48 61 70 70 65 6E 65 64 21 00 00 00 00 00 00 00 00 00 00 00 00
0x20000052: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000074: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x20000096: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

**Command**

Running with Code Size Limit: 32K  
Load "F:\03.淡江碩士\01.碩一(112)\02.

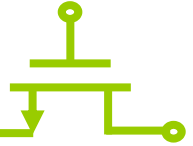
**Simulation**

t1: 0.00001400 sec | L:90 C:1 | CAP NUM SCRL OVR R/V



(2) to switch modes and show the mode changes

(d) from **privileged handler mode** to **unprivileged thread mode**(2/2)



Register

Value

R0	0x00000000
R1	0x11111111
R2	0x22222222
R3	0x33333333
R4	0x00000002
R5	0x00000000
R6	0xE000E000
R7	0x0000D2A
R8	0x00000001
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x20000100
R14 (LR)	0xFFFFFFFF
R15 (PC)	0x0000004C
xPSR	0x01000000

Banked

System

Internal

Mode

Thread

Privileged

Unprivileged

Stack

MSF

States

235

Sec

0.00001469

Project

Registers

56: UDIV r5, r3, r0

57: ; Exception Entry:

58: ; 1. Stacking

59: ; 2. Interrupt Vector Lookup

TEST2.s

44 LDR r7, =SYSHNDCTRL ; p. 163

45 LDR r1, [r6, r7]

46 ORR r1, #0x40000

47 STR r1, [r6, r7]

48 ; try out a divide by 2 then a divide by 0!

49 MOV r0, #0

50 MOV r1, #0x11111111

51 MOV r2, #0x22222222

52 MOV r3, #0x33333333

53 ; this divide works just fine

54 UDIV r4, r2, r1

55 ; this divide takes an exception

56 UDIV r5, r3, r0

57 ; Exception Entry:

58 ; 1. Stacking

59 ; 2. Interrupt Vector Lookup

60 ; 3. LR

61 Exit B Exit

62

63

64 NmiISR B NmiISR

65

66

67 FaultISR B FaultISR

68

69

70 IntDefaultHandler

71 ; let read the Usage Fault Status Register

72 LDR r7, =Usagefault

73 LDRH r1, [r6, r7]

74 TST r1, #0x200

75 BNE Happened

76 BEQ NotHappened

77 Happened

78 LDR r3, =MSG\_with\_Error

79 LDR r1, =0x20000030 ; starting address of characters

80 Loop

81 LDRB r5, [r3], #1

82 STNB r5, [r1], #1

Memory 2

Address: 0x20000090

0x20000090: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000A5: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000BA: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000CF: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x200000E4: 11 11 11 11 22 22 22 22 33 33 33 33 00 00 00 00 FF FF FF FF 4C

Memory 1

Address: 0x20000030

0x20000030: 44 49 56 49 44 45 2D 42 59 2D 30 20 48 61 70 70 65 6E 65 64 21 00 00 00 00 00 00 00 00 00 00 00 00

0x20000052: 00

0x20000074: 00

0x20000096: 00

Command

Running with Code Size Limit: 32K

Load "F:\03.淡江碩士\01.碩一(112)\02.

ASSIGN BreakDisable BreakEnable BreakKill BreakList BreakSet BreakAccess COVTOFILE DEFINE DIR Display Enter

Simulation

t1: 0.00001469 sec

L56 C:1

CAP NUM SCRL OVR R/V



*Thanks for your attention !!*