

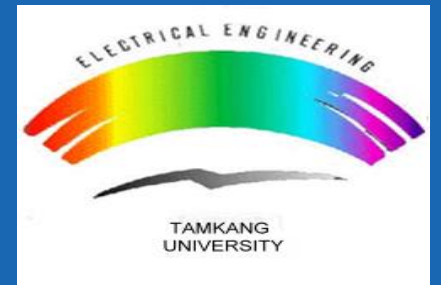
第04次實習課

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.)



P347 16.2.4 Writing the Data to the UART



16.2.4 WRITING THE DATA TO THE UART

Now that the UART is configured to send and receive data, we can try writing some data out of the part. In this case, we'll send some character data—the short message “Watson. Come quickly!” The subroutine for this task is written so that the calling routine can send a single character at a time. When the subroutine receives the character, **it's placed into the transmit buffer**, but only after the processor checks to ensure the previous character has been transmitted. Who's reading this data? In the simulation tools, there is a **serial window that can accept data from a UART**, driving the necessary handshake lines that are normally attached to the receiver. The assembly code for our transmitter routine looks like the following:

```

LDR    r5, =U0START
wait   LDRB    r6, [r5, #LSR0]    ; get status of buffer
      CMP     r6, #0x20          0b0010 0000 ; buffer empty?
      BEQ     NE wait            ; spin until buffer's empty
      STRB    r0, [r5]
  
```

同步機制

錯誤寫法 (因為其他未使用到的位元初始值為0，才可以使用CMP。)

建議使用TST 搭配 EQ

UART 0													
0xE000C000	U0RBR (DLAB=0)	U0 Receiver buffer register	8-bit data								RO	un-defined	
	U0THR (DLAB=0)	U0 Transmit holding register	8-bit data								WO	NA	
	U0DLL (DLAB=1)	U0 Divisor latch LSB	8-bit data								R/W	0x01	
0xE000C004	U0IER (DLAB=0)	U0 Interrupt enable register	0	0	0	0	0	En. Rx Line Status Int.	Enable THRE Int.	En. Rx Data Av.Int.	R/W	0	
	U0DLM (DLAB=1)	U0 Divisor latch LSB	8 bit data								R/W	0	
0xE000C008	U0IIR	U0 Interrupt ID register	FIFOs Enabled	0	0	IIR3	IIR2	IIR1	IIR0		RO	0x01	
	U0FCR	U0 FIFO control register	Rx Trigger	-	-	-	U0 Tx FIFO Reset	U0 Rx FIFO Reset	U0 FIFO Enable		WO	0	
0xE000C00C	U0LCR	U0 Line control register	DLAB	Set break	Stick parity	Even parity select	Parity enable	Nm. of stop bits	Word length select		R/W	0	
0xE000C014	U0LSR	U0 Line status register	Rx FIFO Error	TEMT	THRE	BI	FE	PE	OE	DR	RO	0x60	
0xE000C01C	U0LSR	U0 Scratch pad register	8-bit data								R/W	0	

bit 5(Transmit Holding Register Empty) -> if 1:empty ; 0:not empty。

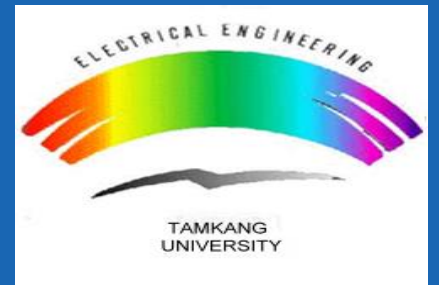
→類似於C語言中的put character

Keil Tool-UARTConfig

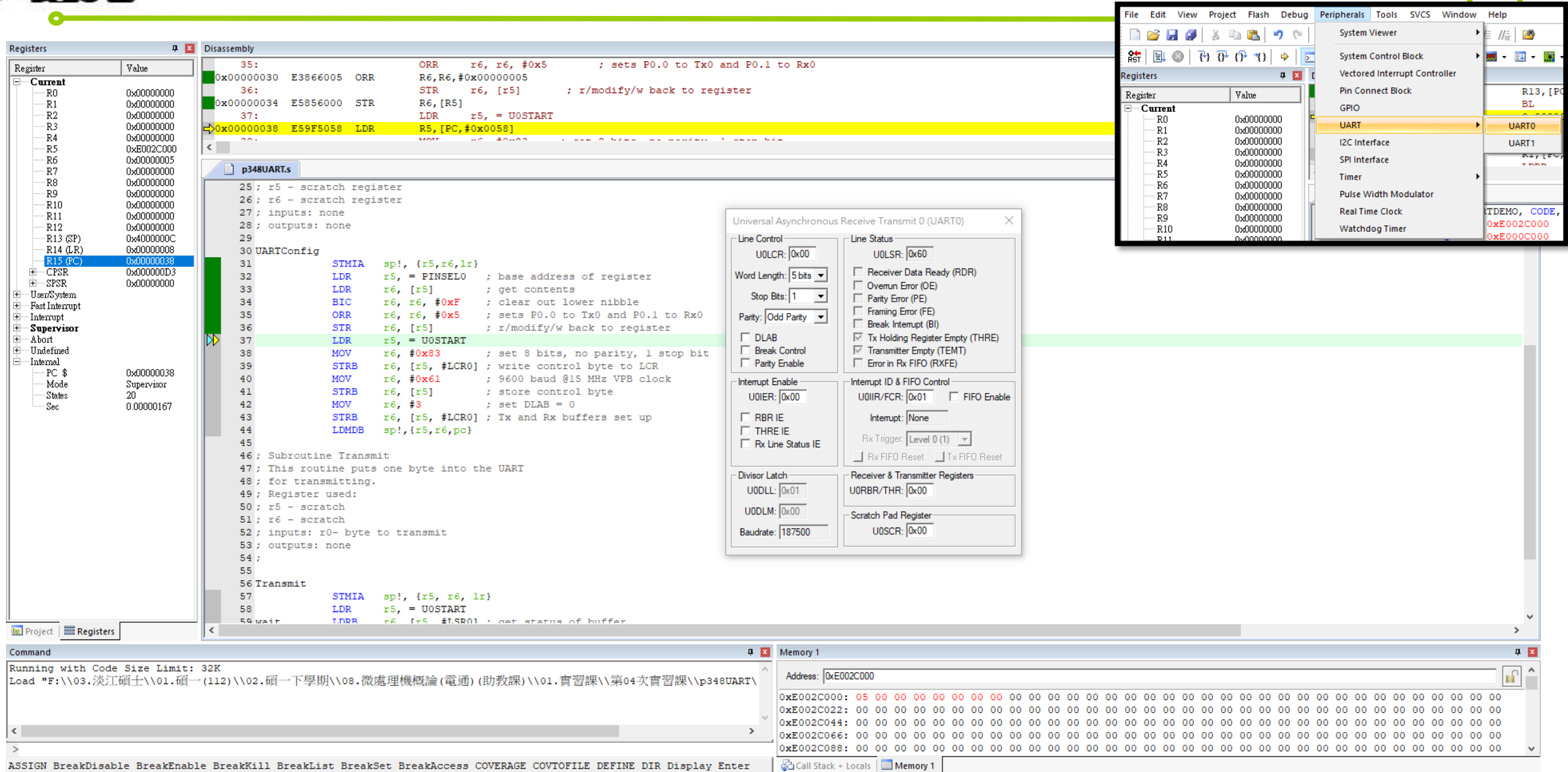
2024 Advanced Mixed-Operation System (AMOS) Lab.



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



Keil Tool-P348(16.2.5 Putting the Code Together)

The screenshot displays the Keil MDK-ARM v5.26.0.0 development environment. The main window shows the disassembly of the file `p348UART.s`. The assembly code is as follows:

```

35:      ORR     r6, r6, #0x5      ; sets P0.0 to Tx0 and P0.1 to Rx0
0x00000030 E3866005 ORR     R6,R6,#0x00000005
36:      STR     r6, [r5]        ; r/modify/w back to register
0x00000034 E5856000 STR     R6,[R5]
37:      LDR     r5, =U0START
0x00000038 E59F5058 LDR     R5,[PC,#0x0058]
38:      MOV     r6, #0x83      ; set 8 bits, no parity, 1 stop bit
39:      STRB    r6, [r5, #LCR0] ; write control byte to LCR
40:      MOV     r6, #0x61      ; 9600 baud @15 MHz VPB clock
41:      STRB    r6, [r5]        ; store control byte
42:      MOV     r6, #3         ; set DLAB = 0
43:      STRB    r6, [r5, #LCR0] ; Tx and Rx buffers set up
44:      LDMDMB  sp!, {r5, r6, pc}
45:
46: ; Subroutine Transmit
47: ; This routine puts one byte into the UART
48: ; for transmitting.
49: ; Register used:
50: ; r5 - scratch
51: ; r6 - scratch
52: ; inputs: r0- byte to transmit
53: ; outputs: none
54:
55:
56 Transmit
57:      STMIA   sp!, {r5, r6, lr}
58:      LDR     r5, =U0START
59 wait  LDRB    r6, [r5, #LSR0] ; get status of buffer

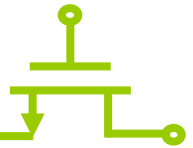
```

The `UART0` configuration dialog is open, showing the following settings:

- Line Control:** UOLCR: 0x00, Word Length: 5 bits, Stop Bits: 1, Parity: Odd Parity.
- 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 Enable:** UIER: 0x00, RBR IE, THRE IE, Rx Line Status IE.
- Interrupt ID & FIFO Control:** UIIR/FCR: 0x01, FIFO Enable, Interrupt: None, Rx Trigger: Level 0 (1), Rx FIFO Reset, Tx FIFO Reset.
- Divisor Latch:** UODLL: 0x01, UODLM: 0x00, Baudrate: 187500.
- Receiver & Transmitter Registers:** UORBR/THR: 0x00.
- Scratch Pad Register:** UOSCR: 0x00.

The `Registers` window shows the current state of the registers, with `R15 (PC)` highlighted at address `0x00000038`. The `Memory` window shows the memory view starting at address `0xE002C000`.

Keil Tool-P348(16.2.5 Putting the Code Together)



Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0xE000C000
R6	0x00000083
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x4000000C
R14 (LR)	0x00000008
R15 (PC)	0x00000044
CPSR	0x000000D3
SFPR	0x00000000

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

```

40:      MOV     r6, #0x61      ; 9600 baud @15 MHz VPB clock
41:      STRB    r6, [r5]      ; store control byte
42:      MOV     r6, #3        ; set DLAB = 0
43:      MOV     R6, #0x00000003
44:      LDMDB   sp!, {r5, r6, pc}
45:
46: ; Subroutine Transmit
47: ; This routine puts one byte into the UART
48: ; for transmitting.
49: ; Register used:
50: ; r5 - scratch
51: ; r6 - scratch
52: ; inputs: r0- byte to transmit
53: ; outputs: none
54:
55:
56 Transmit
57      STMIA    sp!, {r5, r6, lr}
58      LDR     r5, = UOSTART
59 wait    LDRB   r6, [r5, #LSR0] ; get status of buffer

```

Line Control
U0LCR: 0x83
Word Length: 8 bits
Stop Bits: 1
Parity: Odd Parity
☒ DLAB
☐ Break Control
☐ Parity Enable

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 Enable
U0IER: 0x00
☐ RBR IE
☐ THRE IE
☐ Rx Line Status IE

Interrupt ID & FIFO Control
U0IIR/FCR: 0x01
FIFO Enable
Interrupt: None
Rx Trigger: Level 0 (1)
☐ Rx FIFO Reset
☐ Tx FIFO Reset

Divisor Latch
U0DLL: 0x01
U0DLM: 0x00
Baudrate: 187500

Receiver & Transmitter Registers
U0RBR/THR: 0x00
Scratch Pad Register
U0SCR: 0x00

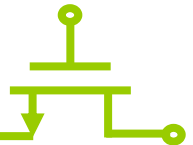
Running with Code Size Limit: 32K
Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\08.微處理機概論(電通)(助教課)\01.實習課\第04次實習課\p348UART\

Address: 0xE000C00C
0xE000C00C: 83 00 00 00 00 00 00 00 60 00
0xE000C02E: 00 00 80 00
0xE000C050: 00
0xE000C072: 00
0xE000C094: 00

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

Real-Time Agent: Target Stopped
Simulation
t1: 0.00000217 sec
L:3 C:31
CAP NUM SCRL OVR R/W

Keil Tool-P348(16.2.5 Putting the Code Together)



Registers

Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0xE000C000
R6	0x00000061
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x4000000C
R14 (LR)	0x00000008
R15 (PC)	0x0000004C
CPSR	0x000000D3
SFPR	0x00000000

Disassembly

```

40: 0x00000044 E3A06061 MOV     R6, #0x61      ; 9600 baud @15 MHz VPB clock
41: 0x00000048 E5C56000 STRB    R6, [R5]      ; store control byte
42: 0x0000004C E3A06003 MOV     R6, #0x00000003 ; set DLAB = 0
43: 0x0000004E E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
44: 0x00000050 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
45: 0x00000052 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
46: 0x00000054 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
47: 0x00000056 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
48: 0x00000058 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
49: 0x0000005A E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
50: 0x0000005C E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
51: 0x0000005E E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
52: 0x00000060 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
53: 0x00000062 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
54: 0x00000064 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
55: 0x00000066 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
56: 0x00000068 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
57: 0x0000006A E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
58: 0x0000006C E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
59: 0x0000006E E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
60: 0x00000070 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
61: 0x00000072 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
62: 0x00000074 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
63: 0x00000076 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
64: 0x00000078 E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
65: 0x0000007A E5C56000 STRB    R6, [R5]      ; Tx and Rx buffers set up
        
```

p348UART.s

```

31: STMIA    sp!, {r5, r6, lr}
32: LDR      r5, = PINSEL0 ; base address of register
33: LDR      r6, [r5]      ; get contents
34: BIC      r6, r6, #0xF  ; clear out lower nibble
35: ORR      r6, r6, #0x5  ; sets P0.0 to Tx0 and P0.1 to Rx0
36: STR      r6, [r5]      ; r/modify/w back to register
37: LDR      r5, = UOSTART
38: MOV      r6, #0x83     ; set 8 bits, no parity, 1 stop bit
39: STRB     r6, [r5, #LCR0] ; write control byte to LCR
40: MOV      r6, #0x61     ; 9600 baud @15 MHz VPB clock
41: STRB     r6, [r5]      ; store control byte
42: MOV      r6, #3        ; set DLAB = 0
43: STRB     r6, [r5, #LCR0] ; Tx and Rx buffers set up
44: LDMDB    sp!, {r5, r6, pc}
45:
46: ; Subroutine Transmit
47: ; This routine puts one byte into the UART
48: ; for transmitting.
49: ; Register used:
50: ; r5 - scratch
51: ; r6 - scratch
52: ; inputs: r0- byte to transmit
53: ; outputs: none
54: ;
55:
56: Transmit
57: STMIA    sp!, {r5, r6, lr}
58: LDR      r5, = UOSTART
59: wait:   LDRB    r6, [r5, #LSR0] ; get status of buffer
60: CMP      r6, #0x20 ; buffer empty?
61: BEQ      wait ; spin until buffer's empty
62: STRB     r0, [r5]
63: LDMDB    sp!, {r5, r6, pc}
64: CharData
65: DCB      "Watson, Come quickly!"
        
```

Universal Asynchronous Receive Transmit 0 (UART0)

Line Control

U0LCR: 0x83

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: 0x61

U0DLM: 0x00

Baudrate: 1932

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\08.微處理機概論(電通)(助教課)\01.實習課\第04次實習課\p348UART\

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

Memory 1

Address: 0xE000C000

0xE000C000: 61 00 00 00 00 00 00 00 01 00 00 00 83 00 00 00 60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0xE000C022: 00 00 00 00 00 00 10 00 00 00 00 00 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

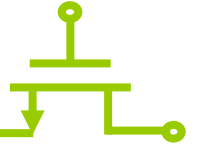
0xE000C044: 00

0xE000C066: 00

0xE000C088: 00

→可以自行算出Clock Frequency。

Keil Tool-P348(16.2.5 Putting the Code Together)



Register	Value
R0	0x00000000
R1	0x00000000
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0xE000C000
R6	0x00000003
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x4000000C
R14 (LR)	0x00000008
R15 (PC)	0x00000054
CPSR	0x000000D3
SPSR	0x00000000
UserSystem	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC \$	0x00000054
Mode	Supervisor
States	32
Sec	0.00000267

```

44:          LDMDMB    sp!, {r5, r6, pc}
45:
46: ; Subroutine Transmit
47: ; This routine puts one byte into the UART
48: ; for transmitting.
49: ; Register used:
50: ; r5 - scratch
51: ; r6 - scratch
52: ; inputs: r0- byte to transmit
53: ; outputs: none
54:
55
56 Transmit
57          STMIA     sp!, {r5, r6, lr}
58          LDR       r5, = U0START
59 wait     LDRB      r6, [r5, #LSR0] ; get status of buffer
60          CMP       r6, #0x20      ; buffer empty?
61          BEQ       wait           ; spin until buffer's empty
62          STRB      r0, [r5]
63          LDMDMB    sp!, {r5, r6, pc}
64 CharData
65          DCB       "Watson. Come quickly!", 0
66          END

```

Universal Asynchronous Receive Transmit 0 (UART0)

Line Control
U0LCR: 0x03
Word Length: 8 bits
Stop Bits: 1
Parity: Odd Parity
☐ DLAB
☐ Break Control
☐ Parity Enable

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 Enable
U0IER: 0x00
☐ RBR IE
☐ THRE IE
☐ Rx Line Status IE

Interrupt ID & FIFO Control
U0IIR/FCR: 0x01 ☐ FIFO Enable
Interrupt: None
Rx Trigger: Level 0 (1)
☐ Rx FIFO Reset ☐ Tx FIFO Reset

Divisor Latch
U0DLL: 0x61
U0DLM: 0x00
Baudrate: 1932

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\08.微處理機概論(電通)(助教課)\01.實習課\第04次實習課\p348UART\

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

Memory 1

Address: 0xE000C00C

0xE000C00C:	03 00 00 00 00 00 00 00 60 00
0xE000C02E:	00 00 80 00
0xE000C050:	00 00
0xE000C072:	00 00
0xE000C094:	00 00

Call Stack + Locals Memory 1

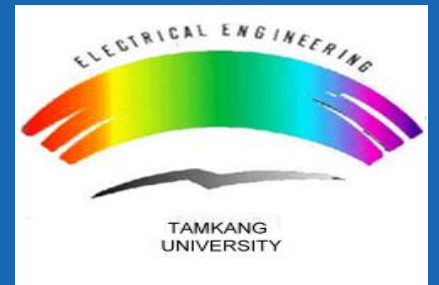
Real-Time Agent: Target Stopped Simulation t1: 0.00000267 sec L:44 C:1 CAP NUM SCRL OVR R/W

Keil Tool-Transmit

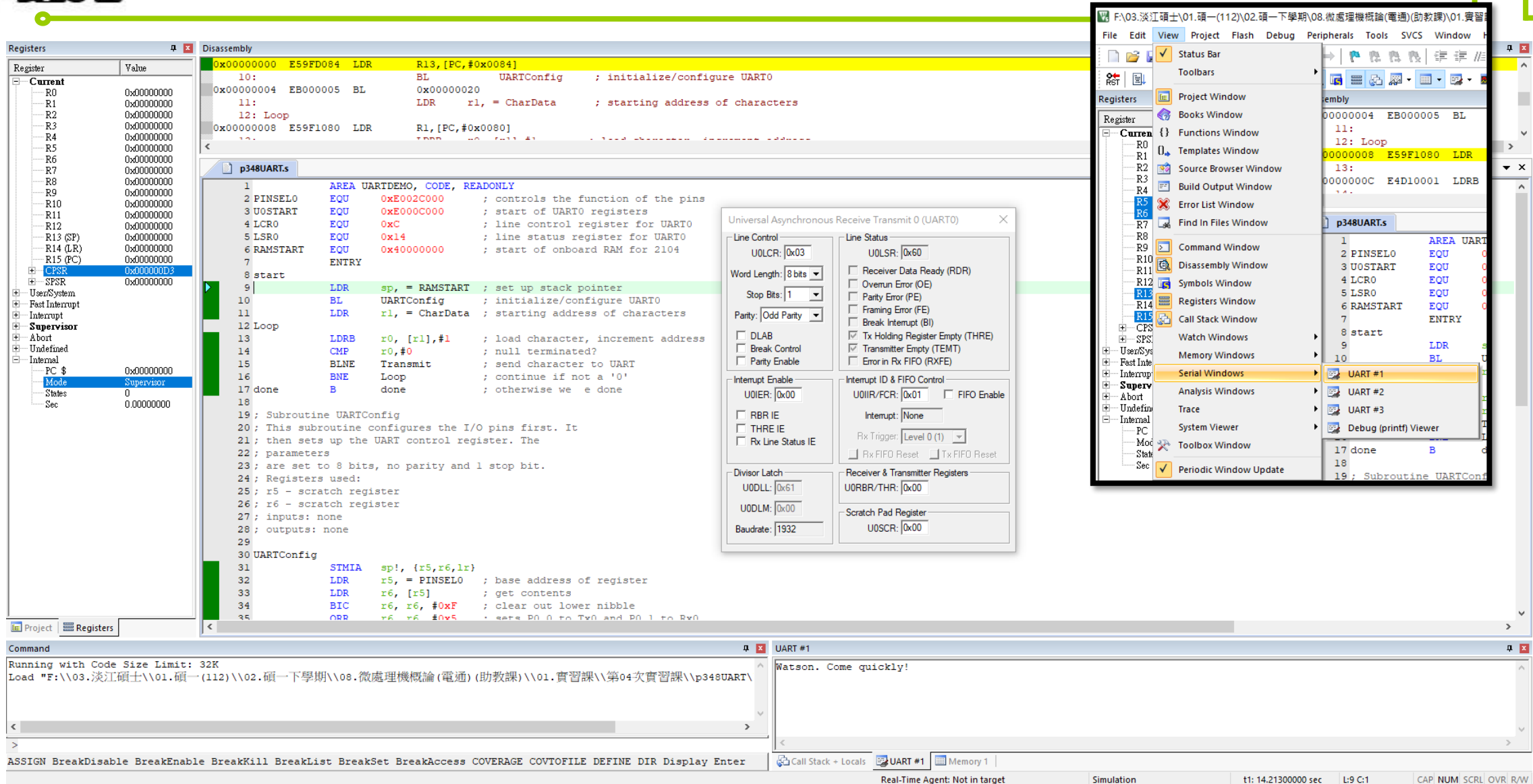
2024 Advanced Mixed-Operation System (AMOS) Lab.



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



Keil Tool-P348(16.2.5 Putting the Code Together)

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
UserSystem	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC \$	0x00000000
Mode	Supervisor
States	0
Sec	0.00000000

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: Loop
0x00000008 E59F1080 LDR R1,[PC,#0x0080]

```

p348UART.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 0x40000000 ; start of onboard RAM for 2104
7 ENTRY
8 start
9 LDR sp, = RAMSTART ; set up stack pointer
10 BL UARTConfig ; initialize/configure UART0
11 LDR r1, = CharData ; starting address of characters
12 Loop
13 LDRB r0, [r1],#1 ; load character, increment address
14 CMP r0,#0 ; null terminated?
15 BLNE Transmit ; send character to UART
16 BNE Loop ; continue if not a '0'
17 done B ; otherwise we are done
18
19 ; Subroutine UARTConfig
20 ; This subroutine configures the I/O pins first. It
21 ; then sets up the UART control register. The
22 ; parameters
23 ; are set to 8 bits, no parity and 1 stop bit.
24 ; Registers used:
25 ; r5 - scratch register
26 ; r6 - scratch register
27 ; inputs: none
28 ; outputs: none
29
30 UARTConfig
31 STMIA sp!, {r5,r6,r1}
32 LDR r5, = PINSEL0 ; base address of register
33 LDR r6, [r5] ; get contents
34 BIC r6, r6, #0x0F ; clear out lower nibble
35 ORR r6, r6, #0x5 ; sets P0.0 to Tx0 and P0.1 to Rx0

```

Universal Asynchronous Receive Transmit 0 (UART0)

Line Control: U0LCR: 0x03, Word Length: 8 bits, Stop Bits: 1, Parity: Odd Parity

Line Status: U0LSR: 0x60

Interrupt Enable: U0IER: 0x00, RBR IE, THRE IE, Rx Line Status IE

Interrupt ID & FIFO Control: U0IIR/FCR: 0x01, Interrupt: None, Rx Trigger: Level 0 (1), Rx FIFO Reset, Tx FIFO Reset

Receiver & Transmitter Registers: U0RBR/THR: 0x00, U0DLM: 0x00, U0DLH: 0x00, Baudrate: 1932

Serial Windows

- UART #1
- UART #2
- UART #3
- Debug (printf) Viewer

UART #1

```

Watson. Come quickly!

```

Command

```

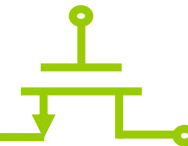
Running with Code Size Limit: 32K
Load "F:\03.淡江碩士\01.碩一(112)\02.碩一下學期\08.微處理機概論(電通)(助教課)\01.實習課\第04次實習課\p348UART\

```







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

Real-Time Agent: Not in target | **Simulation** | **t1: 14.21300000 sec** | **L:9 C:1** | **CAP NUM SCRL OVR R/W**

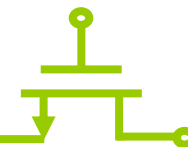
Debug : F11 vs. F10 vs. F5



- ❖ F11(Step)：單步執行。(一行指令一行指令執行)
- ❖ F10(Step Over)：不進副函式，但副函式的結果會呈現出來。
(把副函式當成一步)
- ❖ F5(Run)：整個程式執行完的結果。

	Run	F5
	Stop	
	Step	F11
	Step Over	F10
	Step Out	Ctrl+F11
	Run to Cursor Line	Ctrl+F10

練習：反向印出字串



```

                AREA UARTDEMO, CODE, READONLY
PINSEL0 EQU    0xE002C000    ; controls the function of the pins
U0START EQU    0xE000C000    ; start of UART0 registers
LCR0 EQU    0xC              ; line control register for UART0
LSR0 EQU    0x14             ; line status register for UART0
RAMSTART EQU    0x40000000    ; start of onboard RAM for 2104

```

ENTRY

start

→先自行計算最後一個字元的位置，依序往前取值。(設COUNTER)

```

LDR    sp, =RAMSTART    ; set up stack pointer
BL     UARTConfig        ; initialize/configure UART0

```

```

LDR    r1, =CharData    ; starting address of characters

Loop
LDRB   r0, [r1], #1      ; load character, increment address
CMP    r0, #0            ; null terminated?
BLNE   Transmit          ; send character to UART
BNE    Loop              ; continue if not a '0'

done   B     done         ; otherwise we're done

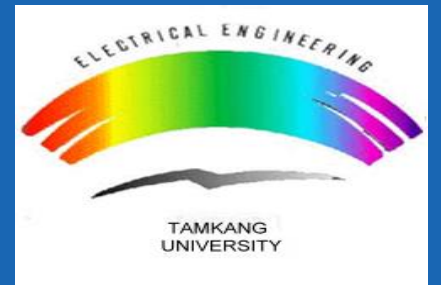
```

Receive

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.)



Reading the data from the UART buffer



Receive

```
STMIA    sp!, {r5, r6, lr}
LDR      r5, = U0START
```

wait

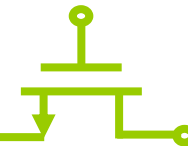
```
LDRB     r6, [r5, #LSR0] ; get status of buffer
TST      r6, #1           ; DR: Data Ready?
BEQ      wait            ; spin until data ready
STRB     r0, [r5]
LDMDB    sp!, {r5, r6, pc}
```

UART 0

0xE000C000	U0RBR (DLAB=0)	U0 Receiver buffer register	8-bit data								RO	un-defined
	U0THR (DLAB=0)	U0 Transmit holding register	8-bit data								WO	NA
	U0DLL (DLAB=1)	U0 Divisor latch LSB	8-bit data								R/W	0x01
0xE000C004	U0IER (DLAB=0)	U0 Interrupt enable register	0	0	0	0	0	En. Rx Line Status Int.	Enable THRE Int.	En. Rx Data Av.Int.	R/W	0
	U0DLM (DLAB=1)	U0 Divisor latch LSB	8 bit data								R/W	0
0xE000C008	U0IIR	U0 Interrupt ID register	FIFOs Enabled	0	0	IIR3	IIR2	IIR1	IIR0		RO	0x01
	U0FCR	U0 FIFO control register	Rx Trigger	-	-	-	U0 Tx FIFO Reset	U0 Rx FIFO Reset	U0 FIFO Enable		WO	0
0xE000C00C	U0LCR	U0 Line control register	DLAB	Set break	Stick parity	Even parity select	Parity enable	Nm. of stop bits	Word length select		R/W	0
0xE000C014	U0LSR	U0 Line status register	Rx FIFO Error	TEMT	THRE	BI	FE	PE	OE	DR	RO	0x60
0xE000C01C	U0LSR	U0 Scratch pad register	8-bit data								R/W	0



FE vs. PE vs. OE



- ❖ FE : Frame Error, PE : Parity Error, OE : Overrun Error
- ❖ Read bits 3, 2 and 1 of a word at memory address 0xE000C014(LSR)
 - If they are all 0 then END
 - Otherwise, then go back to wait

1. Frame Error (FE) - 幀錯誤：

- 幀錯誤發生在接收器無法正確識別通信中的開始位和停止位之間的數據位時。在UART通信中，每個數據字節都有一個開始位和一個或多個停止位，用於標誌數據字節的開始和結束。如果接收器在接收數據時無法準確同步，就會導致幀錯誤。
- 幀錯誤可能由於多種因素引起，例如通信速率設置不正確、噪聲干擾或線路問題。當出現幀錯誤時，接收器通常會將錯誤標誌設置為FE，並且可能會觸發相應的中斷來通知系統。

2. Parity Error (PE) - 奇偶校驗錯誤：

- 奇偶校驗是一種用於檢測數據錯誤的方法。在某些UART設置中，可以配置奇偶校驗位，以使數據位的總數（包括奇偶校驗位）為奇數或偶數。在接收端，當接收到的數據的奇偶性與預期的不一致時，就會發生奇偶校驗錯誤。
- 奇偶校驗錯誤通常表示接收到的數據可能已經被損壞或出錯。這可能是由於通信中的噪聲、干擾或其他原因引起的。

3. Overrun Error (OE) - 溢出錯誤：

- 溢出錯誤發生在UART接收緩衝區無法及時處理接收到的數據時。當接收器無法處理數據流的速度，導致接收緩衝區溢出時，就會發生溢出錯誤。
- 溢出錯誤通常由於接收數據的速度超過系統處理數據的速度而引起。這可能是因為數據傳輸速率太快、接收端處理能力不足或者其他系統性能問題。當發生溢出錯誤時，可能會丟失部分數據，這可能會導致通信錯誤或故障。

UART 0													
0xE000C000	U0RBR (DLAB=0)	U0 Receiver buffer register	8-bit data								RO	un-defined	
	U0THR (DLAB=0)	U0 Transmit holding register	8-bit data								WO	NA	
	U0DLL (DLAB=1)	U0 Divisor latch LSB	8-bit data								R/W	0x01	
0xE000C004	U0IER (DLAB=0)	U0 Interrupt enable register	0	0	0	0	0	En. Rx Line Status Int.	Enable THRE Int.	En. Rx Data Av.Int.	R/W	0	
	U0DLM (DLAB=1)	U0 Divisor latch LSB	8 bit data								R/W	0	
0xE000C008	U0IIR	U0 Interrupt ID register	FIFOs Enabled	0	0	IIR3	IIR2	IIR1	IIR0		RO	0x01	
	U0FCR	U0 FIFO control register	Rx Trigger	-	-	-	U0 Tx FIFO Reset	U0 Rx FIFO Reset	U0 FIFO Enable		WO	0	
0xE000C00C	U0LCR	U0 Line control register	DLAB	Set break	Stick parity	Even parity select	Parity enable	Nm. of stop bits	Word length select		R/W	0	
0xE000C014	U0LSR	U0 Line status register	Rx FIFO Error	TEMT	THRE	BI	FE	PE	OE	DR	RO	0x60	
0xE000C01C	U0LSR	U0 Scratch pad register	8-bit data								R/W	0	

→參考技術手冊：<https://reurl.cc/yY8MV2>

Q&A

Thanks for your attention !!