第03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.)

設定傳輸格式

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.)

P346 程式複習

The next step to configuring the UART is to set the number of data bits, the parity, and the number of stop bits. Again, the starting address of the UART0 configuration register, 0xE000C000, is loaded into a general register to be used as a base address. The LCR and LSR registers can be accessed using a pre-indexed addressing scheme, where the offsets are equated to known values at the beginning of the final routine. Here, LCR0 would be equated to 0xC, and for our write routine, LSR0 would be equated to 0x14. Since these are 8-bit registers, they must be accessed using STRB and LDRB instructions. The rest of the configuration code is below.

題目：initialize UART to have 8-bit character length , no-parity, 1-stop-bit formet of line.

組合語言：

Write 0x83 into a byte at memory address 0xE000C00C

```
LDR     r5, =U0START
MOV     r6, #0x83          ; set 8 bits, no parity, 1 stop bit
STRB    r6, [r5, #LCR0]    ; write control byte to LCR
MOV     r6, #0x61          ; 9600 baud @15 MHz VPB clock
STRB    r6, [r5]           ; store control byte
```

```
LDR r5, =0xE000C00C
MOV  r6, #0x83
STRB r6, [r5]
```

FIGURE 16.3 & TABLE 16.1

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

FIGURE 16.3 Memory map of UART0 on the LPC2104. (From LPC2106/2105/2104 User Manual NXP Semiconductors, September 2003. With permission.)

→set 8 bits, no parity, 1 stop bit

→0b10000011 = 0x83

TABLE 16.1
UART Configuration Bits in the Control Register

	U0LCR	Function	Description	Reset Value
11	1:0	Word Length Select	00:5-bit character length 01:6-bit character length 10:7-bit character length 11:8-bit character length	0
0	2	Stop Bit Select	0:1 stop bit 1:2 stop bits (1.5 if U0LCR[1:0] = 00)	0
0	3	Parity Enable	0: Disable parity generation and checking 1: Enable parity generation and checking	0
00	5:4	Parity select	00: Odd parity 01: Even parity 10: Forced "1" stick parity 11: Forced "0" stick parity	0
(無關填0)				
0	6	Break Control	0: Disable break transmission 1: Enable break transmission. Output pin UART0 TxD is forced to logic 0 when U0LCR6 is actively high	0
1	7	Divisor Latch Access Bit	0: Disable access to divisor latches 1: Enable access to divisor latches	0

設定傳輸速率

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.)

P346 程式(Initialize UART)

在 UART 通信中，
波特率（即傳輸速率）是通過系統時鐘的頻率(clock cycles/second)除以除數鎖存器(divisor latch)(clock cycles/bit)中的值來確定的。
這個除數值決定了數據的傳送或接收速度。

設定 transmission format

(因為資料傳輸都是"bit stream")

transmission speed

```
LDR     r5, =U0START
MOV     r6, #0x83           ; set 8 bits, no parity, 1 stop bit
STRB    r6, [r5, #LCR0]     ; write control byte to LCR
MOV     r6, #0x61(divisor latch) 9600 baud @15 MHz VPB clock
STRB    r6, [r5]            ; store control byte
```

9600 bps(bits/second) → ? clock cycles/bit(因為電腦只看得懂clock cycles)

15 MHz = 15000000 clock cycles/second (clock frequency)

→ 15000000(clock cycles/second) * (1/9600)(seconds/bit)

= 15000000/9600 = 1562.5 (clock cycles/bit)

問題：只有8-bit register

1562/16 = 97.625 = 0x61 (shift right 4 bits)

(注：如果原本就在0~255之間，則不須除以16。)

FIGURE 16.3 & TABLE 16.1

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

Operation (開機完成後使用)

Initialization (BIOS)

→set 8 bits, no parity, 1 stop bit

→0b10000011 = 0x83

TABLE 16.1
UART Configuration Bits in the Control Register

U0LCR		Function	Description	Reset Value
11	1:0	Word Length Select	00:5-bit character length 01:6-bit character length 10:7-bit character length 11:8-bit character length	0
0	2	Stop Bit Select	0:1 stop bit 1:2 stop bits (1.5 if U0LCR[1:0] = 00)	0
0	3	Parity Enable	0: Disable parity generation and checking 1: Enable parity generation and checking	0
00 (無關填0)	5:4	Parity select	00: Odd parity 01: Even parity 10: Forced "1" stick parity 11: Forced "0" stick parity	0
0	6	Break Control	0: Disable break transmission 1: Enable break transmission. Output pin UART0 TxD is forced to logic 0 when U0LCR6 is actively high	0
1	7	Divisor Latch Access Bit	0: Disable access to divisor latches 1: Enable access to divisor latches 打開divisor latch功能	0

設定完速率，將功能恢復成Tx、Rx

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 程式(設定成Tx、Rx功能)

```
MOV     r6, #3            ; set DLAB = 0
STRB    r6, [r5, #LCR0]   ; Tx and Rx buffers set up
```

→準備開機完成之後要運用Tx、Rx功能。
(將功能恢復成Tx、Rx)

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

U0LCR	Function	Description	Reset Value
1:0	Word Length Select	00:5-bit character length 01:6-bit character length 10:7-bit character length 11:8-bit character length	0
2	Stop Bit Select	0:1 stop bit 1:2 stop bits (1.5 if U0LCR[1:0] = 00)	0
3	Parity Enable	0: Disable parity generation and checking 1: Enable parity generation and checking	0
5:4	Parity select	00: Odd parity 01: Even parity 10: Forced “l” stick parity 11: Forced “0” stick parity	0
6	Break Control	0: Disable break transmission 1: Enable break transmission. Output pin UART0 TxD is forced to logic 0 when U0LCR6 is actively high	0
7	Divisor Latch Access Bit	0: Disable access to divisor latches 1: Enable access to divisor latches	0

設定腳位功能為Tx、Rx

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.)

P346 FIGURE 16.4 & P347 TABLE 16.2

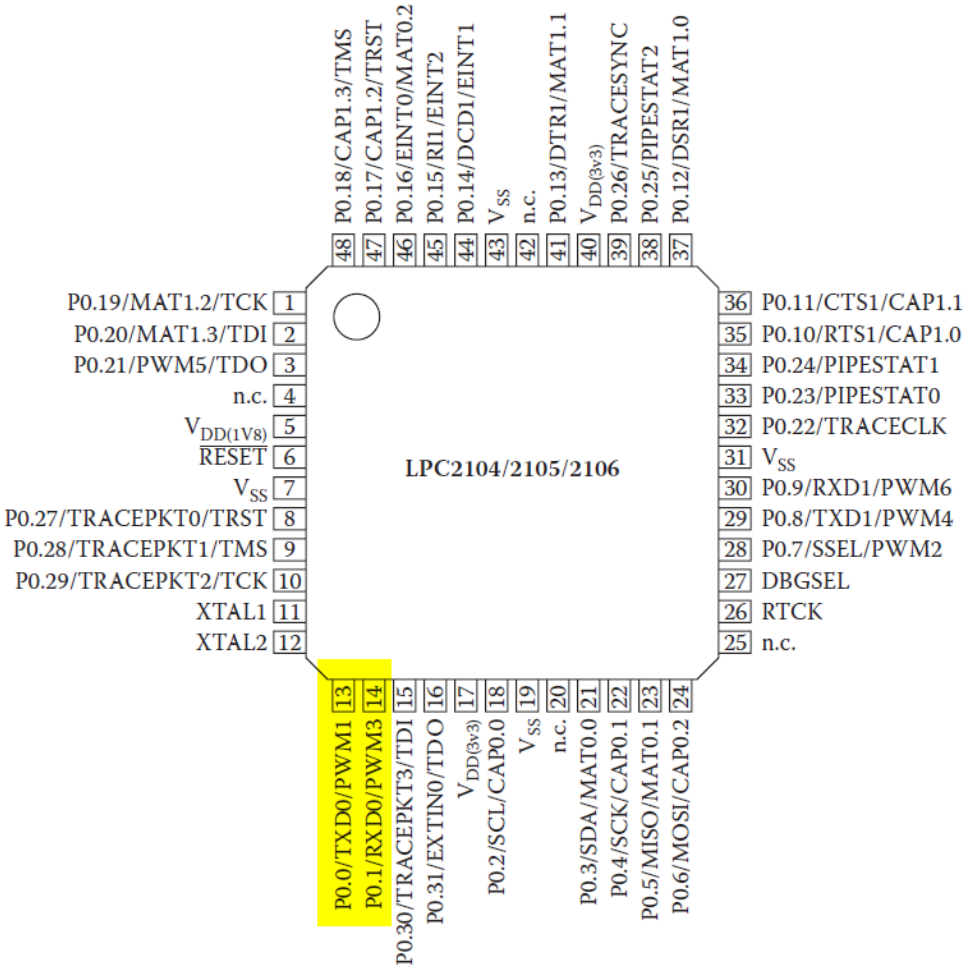

TABLE 16.2
PINSEL0 Register for Pin Configurations

PINSEL0	Pin Name	Function When 00	Function When 01	Function When 10	Function When 11	Reset Value
1:0	P0.0	GPIO Port 0.0	TxD (UART 0)	PWM1	Reserved	0
3:2	P0.1	GPIO Port 0.1	RxD (UART 0)	PWM3	Reserved	0
5:4	P0.2	GPIO Port 0.2	SCL (I^2C)	Capture 0.0 (Timer 0)	Reserved	0
7:6	P0.3	GPIO Port 0.3	SDA (I^2C)	Match 0.0 (Timer 0)	Reserved	0
9:8	P0.4	GPIO Port 0.4	SCK (SPI)	Capture 0.1 (Timer 0)	Reserved	0
11:10	P0.5	GPIO Port 0.5	MISO (SPI)	Match 0.1 (Timer 0)	Reserved	0
13:12	P0.6	GPIO Port 0.6	MOSI (SPI)	Capture 0.2 (Timer 0)	Reserved	0
15:14	P0.7	GPIO Port 0.7	SSEL (SPI)	PWM2	Reserved	0
17:16	P0.8	GPIO Port 0.8	TxD UART 1	PWM4	Reserved	0
19:18	P0.9	GPIO Port 0.9	RxD (UART 1)	PWM6	Reserved	0
21:20	P0.10	GPIO Port 0.10	RTS (UART 1)	Capture 1.0 (Timer 1)	Reserved	0
23:22	P0.11	GPIO Port 0.11	CTS (UART 1)	Capture 1.1 (Timer 1)	Reserved	0
25:24	P0.12	GPIO Port 0.12	DSR (UART 1)	Match 1.0 (Timer 1)	Reserved	0
27:26	P0.13	GPIO Port 0.13	DTR (UART 1)	Match 1.1 (Timer 1)	Reserved	0
29:28	P0.14	GPIO Port 0.14	CD (UART 1)	EINT1	Reserved	0
31:30	P0.15	GPIO Port 0.15	RI (UART1)	EINT2	Reserved	0

Source: From LPC2106/2105/2104 User Manual NXP Semiconductors, September 2003. With permission.

P345 程式(設定腳位功能)

16.2.3 Configuring the UART

To demonstrate how easy it is to talk to a peripheral, we'll write a short block of code that does two things: it calls a subroutine to configure our UART, and it then sends a short message through the UART for the Keil tools to read. If you look at the external pins of the LPC2104, shown in Figure 16.4, you will notice that they are multiplexed pins, meaning that the function of the pin itself is configurable. This allows a package to reduce the pin count, but the programmer must configure the pins to use them. So the first order of business is to set up the LPC2104 so that pins P0.0 and P0.1 become our transmit and receive pins, Tx0 and Rx0, respectively. To do this, we load the address of the pin configuration register, shown in Table 16.2, into a general register, where PINSEL0 is equated to 0xE002C000. Using a read-modify-write sequence (good practice when you don't want to disturb other configuration or status bits), PINSEL0[1:0] and PINSEL0[3:2] are set to 0b01. The assembly would look like the following:

```
        LDR     r5,=PINSEL0     ; base address of register
Read    LDR     r6,[r5]         ; get contents
Modify  BIC     r6,r6,#0xF      ; clear out lower nibble 4 bits
        ORR     r6,r6,#0x5      ; sets P0.0 to Tx0 and P0.1 to Rx0
Write   STR     r6, [r5]        ; r/modify/w back to register
```

組合語言：Write 0b0101 (5) into bits 3~0 of register PINSEL0, the remaining(other) bits unchanged.

微處理機概論：Set Pin 13.14 of UART to function as Tx and Rx
p0.0 p0.1

把所有程式放一起

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.)

P348 16.2.5 Putting the Code Together

```
            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
            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
```

P348 16.2.5 Putting the Code Together

第一:設定腳位功能為Tx、Rx
第二:設定目前的要求至LCR暫存器
(Line Control Register, 0xE000C000)
第三:將要設定的速度存入指定暫存器
(0xE000C000)
第四:將功能恢復為Tx、Rx

```
; r5 - scratch register
; r6 - scratch register
; inputs: none
; outputs: none

UARTConfig
        STMIA  sp!, {r5,r6,lr}
        LDR    r5,=PINSEL0      ; base address of register
        LDR    r6,[r5]          ; get contents
        BIC    r6,r6,#0xF       ; clear out lower nibble
        ORR    r6,r6,#0x5       ; sets P0.0 to Tx0 and P0.1 to Rx0
        STR    r6, [r5]         ; r/modify/w back to register
        LDR    r5,=U0START
        MOV    r6, #0x83        ; set 8 bits, no parity, 1 stop bit
        STRB   r6, [r5, #LCR0]  ; write control byte to LCR
        MOV    r6, #0x61        ; 9600 baud @15 MHz VPB clock
```

BL UARTConfig ;LR = return address
Link Register

Memory-Mapped Peripherals 349

```
        STRB   r6, [r5]         ; store control byte
        MOV    r6, #3           ; set DLAB = 0
        STRB   r6, [r5, #LCR0]  ; Tx and Rx buffers set up
        LDMDB  sp!,{r5,r6,pc}

; Subroutine Transmit
; This routine puts one byte into the UART
; for transmitting.
; Register used:
; r5 - scratch
; r6 - scratch
; inputs: r0- byte to transmit
; outputs: none
;
```

BX LR ;MOV PC, LR
Program Counter

Caller-saved vs. Callee saved

Caller Saved Registers（呼叫者保存寄存器）：通常是r0到r3。呼叫者需要在調用子程序之前將這些寄存器的值保存到堆疊中，並在調用後恢復這些值。

Callee Saved Registers（被呼叫者保存寄存器）：通常是r4到r11。被呼叫的子程序需要在進行寄存器操作之前將這些寄存器的值保存到堆疊中，然後在返回之前恢復這些值。

(在副程式有用到不需要暫存、還原)
Caller-saved registers：r0~r3, r12

r13(SP)、r14(LR)、r15(PC)
沒有規範

348 ARM Assembly Language

Callee-saved registers：r4~r11
(在副程式有用到要暫存、還原)

ex:employer、employee

16.2.5 Putting the Code Together

Now that we have one subroutine to set up our UART and another to send a character, the remaining code will be responsible for reading a sentence from memory, one character at a time, and calling the subroutine to transmit it. A small loop will read a character from memory and test to see whether it is the null terminator for a string, i.e., the value 0. If so, the loop terminates. Otherwise, the subroutine Transmit is called. **-> Arm Architecture Procedure Call Standard(跟副程式呼叫有關的標準) 協定(ex：2G、3G、4G、5G...)**

The AAPCS allows registers r0 through r3 to be corruptible, and we've used registers r5 and r6 in our subroutines. While the code could be written without having to stack any registers (left as an exercise), we'll go ahead and set the stack pointer to the start of RAM, which is address 0x40000000. The registers used in our subroutines can then be saved off. The code below is a complete routine.

P348 16.2.5 Putting the Code Together

```
            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'


 Transmit

              STMIA   sp!,{r5,r6,lr}
              LDR     r5,=U0START
 wait         LDRB    r6,[r5,#LSR0]      ; get status of buffer
              CMP     r6,#0x20           ; buffer empty?
              BEQ     wait               ; spin until buffer's empty
              STRB    r0,[r5]
              LDMDB   sp!,{r5,r6,pc}
 CharData
              DCB     "Watson. Come quickly!",0
              END
```

Q&A

Thanks for your attention !!