บทความ ARM7 เขียนโปรแกรมกับ ARM7
ตอนที่ 3 ทดลองโปรแกรมหลอดแอลอีดี

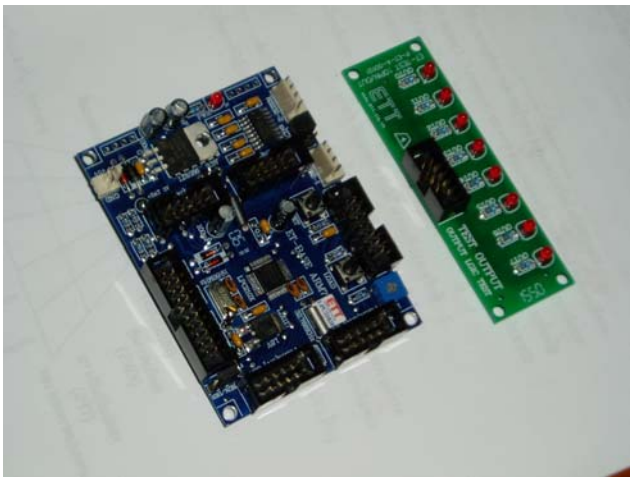
ศุภชัย บุศราทิจ
คณะเทคโนโลยีสารสนเทศ
มหาวิทยาลัยราชภัฏเพชรบุรี

จากบทความครั้งที่แล้วเราได้รู้จักกับคำสั่งของอาร์ม7 (ARM7) กันเรียบร้อยแล้ว ในบทความนี้เป็นการเรียนรู้เกี่ยวกับพอร์ตไอ/โอ (i/o port) ของอาร์ม7 แต่ก่อนอื่นผมต้องขออภัยที่เขียนบทความได้ล่าช้ากว่าที่ควรจะเป็น ทั้งนี้เนื่องจากช่วงเดือนที่ผ่านมานั้นมีหลายเรื่องราวประดังเข้ามาพร้อมๆกัน ไม่ว่าจะเป็นเรื่องการจากไปของโน๊ตบุ๊ก IBM ที่อยู่ร่วมกันมากว่า 2 ปี เรื่องของการสอนที่ผมต้องสอนถึง 5 วิชา คือ ดิสครีตและโครงสร้าง การวิจัยดำเนินงาน การจัดการความมั่นคงสารสนเทศ หลักความมั่นคงคอมพิวเตอร์และไซเบอร์ และพฤติกรรมการสอน แค่เตรียมสอนก็หัวปั่นเลยทีเดียว นอกจากนี้ผมต้องปั่นภาคนิพนธ์ให้จบการศึกษา (ปริญญาโทสาขาเทคโนโลยีและยุทธศาสตร์สารสนเทศ) ซึ่งตอนนี้ก็จัดการเรียบร้อยแล้ว เหลือเก็บตกพวกแก้ไขความผิดพลาดในการพิมพ์เอกสาร พร้อมกันนี้ก็เข้าศึกษาต่อระดับปริญญาเอกในสาขาเทคโนโลยีคุณภาพ ต่อเนื่องกันไป พอซาๆ จากเรื่องงานก็มาเขียนบทความกันต่อ และคงเขียนต่อไปเรื่อยๆ จนกว่าทางอีทีทีจะเบื่อหน้าผมกันไปข้างนึงครับ (ฮ่ะๆๆ ล้อเล่นนะครับ)

อุปกรณ์

ในบทความนี้ผมใช้อุปกรณ์ประกอบการทดลองดังนี้

1. บอร์ด ET-Base ARM7 LPC2103 (www.etteam.com/product/ARM/et-base_arm2103.htm)
2. บอร์ด ET-TEST 10P/OUT (www.etteam.com/product/13A02.html)
3. บอร์ด ET-USB/RS232 (www.etteam.com/product/12A28.html) สำหรับแปลงพอร์ต USB ให้เป็น RS232
4. คอมพิวเตอร์เอเซอร์แอสไพร์ 5051 (ควักเงินซื้ออีกแล้วครับ ฮ่ะๆ)



รูปที่ 1 (ซ้าย) ET-USB/RS232 (ขวา) ET-Base ARM7 LPC2103 และ ET-TEST 10P/OUT

ซอฟต์แวร์

โปรแกรมที่ผมใช้ประกอบการทดลองก็มีดังนี้ครับ

1. โปรแกรม LPC2000 Flash ของบริษัทฟิลิปส์ใช้สำหรับโปรแกรมชิพของอาร์ม 7

2. โปรแกรม Keil-ARM7 รุ่น 3.0a (รุ่นทดลอง) สำหรับเขียนภาษาซี ซึ่งถ้าสนใจรายละเอียดการใช้งานให้อ่านจากเอกสารต่อไปนี้ครับ

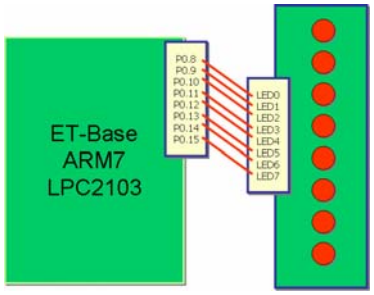
2.1 http://www.etteam.com/product/ARM/01/exKeil.pdf เป็นตัวอย่างการเขียนโปรแกรม ของอีทีที

2.2 http://www.etteam.com/product/ARM/01/qStartARM.pdf เป็นเอกสารสำหรับอาร์ม 7 เบื้องต้น ของอีทีที

3. ที่ขาดไม่ได้คือไมโครซอฟต์วินโดวส์ XP Pro SP2 (รหัสเป็นของจริง แต่ลงใหม่ไม่ได้แล้วเนื่องจากลงหลายครั้ง ไมโครซอฟต์ไม่ยอมให้ activated ผมเลยต้องใช้ของปลอมแล้วเปลี่ยนรหัสเป็นของจริง... งงไหมครับ :D)

วงจรการเชื่อมต่อ

ในการทดลองนี้ผมเชื่อมต่อบอร์ด ET-TEST 10P/OUT เข้ากับพอร์ต P0.8-P0.15 ของอาร์ม 7 ดังรูปด้านล่าง



รูปที่ 2 การเชื่อมต่อ ET-Base ARM7 LPC 2103 เข้ากับ ET-TEST 10P/OUT

พอร์ตของอาร์ม 7

อาร์ม 7 มีพอร์ตเอนกประสงค์ (GPIO: general purpose input/output) ให้ผู้เขียนโปรแกรมใช้งานได้ 2 พอร์ต คือ พอร์ต0 (P0) และพอร์ต1 (P1) ทั้งสองพอร์ตนี้จะถูกอ้างอิงด้วยเรจิสเตอร์ขนาด 32 บิต แต่ในการใช้งานจริง P0 ของบอร์ด ET-Base ARM7 LPC2103 นั้นเราสามารถใช้งานได้เพียง 30 บิต คือ ขา P0.0-P0.25 และ P0.27-P0.30 ส่วน P1 นั้นสามารถใช้ได้ 16 บิต คือ P1.16-P1.31

ในการสั่งงานนั้นจะต้องกำหนดการทำงานของแต่ละขาของพอร์ตผ่านทางเรจิสเตอร์ PINSEL0, PINSEL1 และ PINSEL2 ซึ่งมีรายละเอียดดังต่อไปนี้

PINSEL0 (ค่าหลังรีเซ็ตคือ $00000000000000000000000_2$)

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
P0.15		P0.14		P0.13		P0.12		P0.11		P0.10		P0.9		P0.8		P0.7		P0.6		P0.5		P0.4		P0.3		P0.2		P0.1		P0.0	

จากการจัดเรียงบิตใน PINSEL0 จะพบว่าในแต่ละบิตของพอร์ต P0 นั้นประกอบไปด้วย 2 บิต ทั้งนี้เนื่องจาก เราสามารถที่จะกำหนดภาวการณ์ทำงานของแต่ละบิตของ P0 ได้ ดังรายละเอียดต่อไปนี้ครับ

P0.0
- 00 = ทำงานเป็น GPIO P0.0
- 01 = เป็น TxD ของ UART0
- 10 = ทำงานเป็น PWM1
- 11 = กันเอาไว้

P0.1
- 00 = ทำงานเป็น GPIO P0.1
- 01 = เป็น RxD ของ UART0
- 10 = ทำงานเป็น PWM3
- 11 = ทำหน้าที่เป็น EINT0

สิ่งที่เราจะต้องระวังเป็นอย่างมากคือ ในการใช้ P0 นั้น เราจะต้องกำหนดค่าของ P0.0 และ P0.1 เป็น 01 กับ 01 เสมอ เนื่องจากวงจรของ ET-Base ARM7 LPC1203 นั้นต่อสองขานี้เข้ากับวงจร RS232 ซึ่งถ้ากำหนดค่าผิดพลาดอาจจะมีผลกับบอร์ดได้ครับ

P0.2
- 00 = ทำงานเป็น GPIO P0.2
- 01 = เป็นขา SCL ของการเชื่อมต่อผ่าน I^2C
- 10 = ทำงานเป็น Capture 0.0 ของ TIMER0
- 11 = กันเอาไว้

P0.3
- 00 = ทำงานเป็น GPIO P0.3
- 01 = เป็นขา SDA ของการเชื่อมต่อผ่าน I^2C
- 10 = ทำงานเป็น Match 0.0 ของ TIMER0
- 11 = EINT1

P0.4
- 00 = ทำงานเป็น GPIO P0.4
- 01 = เป็นขา SCK ของการเชื่อมต่อแบบ SPI0
- 10 = ทำงานเป็น Capture 0.1 ของ TIMER0
- 11 = กันเอาไว้

P0.5
- 00 = ทำงานเป็น GPIO P0.5
- 01 = เป็นขา MISO ของการเชื่อมต่อผ่าน SPI0
- 10 = ทำงานเป็น Match 0.1 ของ TIMER0
- 11 = กันเอาไว้

P0.6

- 00 = ทำงานเป็น GPIO P0.6
- 01 = เป็นขา MOSI ของการเชื่อมต่อผ่าน SPI0
- 10 = ทำงานเป็น Capture 0.2 ของ TIMER0
- 11 = กันเอาไว้

P0.7

- 00 = ทำงานเป็น GPIO P0.7
- 01 = เป็นขา SSEL ของการเชื่อมต่อผ่าน SPI0
- 10 = ทำงานเป็น Capture 0.2 ของ TIMER0
- 11 = กันเอาไว้

P0.8

- 00 = ทำงานเป็น GPIO P0.8
- 01 = เป็นขา TxD ของการเชื่อมต่อ UART1
- 10 = ทำงานเป็น PWM4
- 11 = กันเอาไว้

P0.9

- 00 = ทำงานเป็น GPIO P0.9
- 01 = เป็นขา RxD ของการเชื่อมต่อ UART1
- 10 = ทำงานเป็น PWM6
- 11 = EINT3

P0.10

- 00 = ทำงานเป็น GPIO P0.10
- 01 = เป็นขา RTS ของการเชื่อมต่อ UART1
- 10 = ทำงานเป็น Capture 1.0 ของ TIMER1
- 11 = กันเอาไว้

P0.11

- 00 = ทำงานเป็น GPIO P0.11
- 01 = เป็นขา CTS ของการเชื่อมต่อ UART1
- 10 = ทำงานเป็น Capture 1.1 ของ TIMER1
- 11 = กันเอาไว้

P0.12

- 00 = ทำงานเป็น GPIO P0.12
- 01 = เป็นขา DSR ของการเชื่อมต่อ UART1
- 10 = ทำงานเป็น Match 1.0 ของ TIMER1
- 11 = กันเอาไว้

P0.13
- 00 = ทำงานเป็น GPIO P0.13
- 01 = เป็นขา TxD ของการเชื่อมต่อ UART1
- 10 = ทำงานเป็น PWM4
- 11 = กันเอาไว้

P0.14
- 00 = ทำงานเป็น GPIO P0.14
- 01 = เป็นขา CD ของการเชื่อมต่อ UART1
- 10 = EINT1
- 11 = กันเอาไว้

P0.15
- 00 = ทำงานเป็น GPIO P0.15
- 01 = เป็นขา RI ของการเชื่อมต่อ UART1
- 10 = EINT2
- 11 = กันเอาไว้

ในตารางต่อไปนี้จะเป็นของ PINSEL1 ครับ

PINSEL1 (ค่าหลังรีเซ็ตคือ $00010101010000000000000_2$)

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
		P0.30		P0.29		P0.28		P0.27				P0.25		P0.24		P0.23		P0.22		P0.21		P0.20		P0.19		P0.18		P0.17		P0.16	

รายละเอียดของภาวการณ์ทำงานของแต่ละบิตของ P0.16-P0.30 เป็นดังต่อไปนี้ครับ

P0.16
- 00 = ทำงานเป็น GPIO P0.16
- 01 = EINT0
- 10 = เป็นขา Match 2.0 ของ TIMER0
- 11 = เป็นขา Capture 0.2 ของ TIMER0

P0.17
- 00 = ทำงานเป็น GPIO P0.17
- 01 = เป็นขา Capture 1.2 ของ TIMER1
- 10 = เป็นขา SCK ของการเชื่อมต่อ SPI1
- 11 = เป็นขา Match 1.2 ของ TIMER1

P0.18

00	=	ทำงานเป็น GPIO P0.18
01	=	เป็นขา Capture 1.3 ของ TIMER1
10	=	เป็นขา MISO ของการเชื่อมต่อ SPI1
11	=	เป็นขา Match 1.3 ของ TIMER1

P0.19

00	=	ทำงานเป็น GPIO P0.19
01	=	เป็นขา Match 1.2 ของ TIMER1
10	=	เป็นขา MOSO ของการเชื่อมต่อ SPI1
11	=	เป็นขา Match 1.3 ของ TIMER1

P0.20

00	=	ทำงานเป็น GPIO P0.20
01	=	เป็นขา Match 1.3 ของ TIMER1
10	=	เป็นขา SSEL ของการเชื่อมต่อ SPI1
11	=	EINT3

P0.21

00	=	ทำงานเป็น GPIO P0.21
01	=	เป็นขา PWM5
10	=	กันเอาไว้
11	=	เป็นขา Capture 1.3 ของ TIMER1

P0.22

00	=	ทำงานเป็น GPIO P0.22
01	=	กันเอาไว้
10	=	เป็นขา Capture 0.0 ของ TIMER0
11	=	เป็นขา Match 0.0 ของ TIMER0

P0.23

00	=	ทำงานเป็น GPIO P0.23
01	=	เป็นขา RD2 ของ CAN Controller2
10	=	กันเอาไว้
11	=	กันเอาไว้

P0.24

00	=	ทำงานเป็น GPIO P0.24
01	=	เป็นขา TD2 ของ CAN Controller 2
10	=	กันเอาไว้
11	=	กันเอาไว้

P0.25
- 00 = ทำงานเป็น GPIO P0.25
- 01 = เป็นขา RD1 ของ CAN Controller 1
- 10 = กันเอาไว้
- 11 = กันเอาไว้

P0.27
- 00 = ทำงานเป็น GPIO P0.27
- 01 = เป็นขา AIN0 ของตัวแปลง A/D
- 10 = เป็นขา Capture 0.1 ของ TIMER0
- 11 = เป็นขา Match 0.1 ของ TIMER0

P0.28
- 00 = ทำงานเป็น GPIO P0.28
- 01 = เป็นขา AIN1 ของตัวแปลง A/D
- 10 = เป็นขา Match 0.2 ของ TIMER0
- 11 = เป็นขา Match 0.2 ของ TIMER0

P0.29
- 00 = ทำงานเป็น GPIO P0.29
- 01 = เป็นขา AIN2 ของตัวแปลง A/D
- 10 = เป็นขา Match 0.3 ของ TIMER0
- 11 = เป็นขา Match 0.3 ของ TIMER0

P0.30
- 00 = ทำงานเป็น GPIO P0.30
- 01 = เป็นขา AIN3 ของตัวแปลง A/D
- 10 = EINT3
- 11 = เป็นขา Capture 0.0 ของ TIMER0

และตารางต่อไปนี้เป็นตารางของ PINSEL2 มีรายการแตกต่างจาก PINSEL0 และ PINSEL1 เนื่องจากในการกำหนดค่านั้นเราสามารถกำหนดภาวการณ์ทำงานได้ 4 ลักษณะดังตารางถัดไป

PINSEL2 (ค่าหลังรีเซ็ตคือ $00000000000000000000000_2$)

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
																												GPIO			

GPIO
- 00 = P1.26-P1.31 ทำงานเป็น GPIO P1.26 ถึง P1.31
  P1.16-P1.25 ทำงานเป็น GPIO P1.16 ถึง P1.31
- 01 = P1.26-P1.31 ใช้เป็นพอร์ต Debug
  P1.16-P1.25 ทำงานเป็น GPIO P1.16 ถึง P1.31
- 10 = P1.26-P1.31 ทำงานเป็น GPIO P1.26 ถึง P1.31
  P1.16-P1.25 ใช้งานเป็นพอร์ต Trace
- 11 = P1.26-P1.31 ใช้เป็นพอร์ต Debug
  P1.16-P1.25 ใช้งานเป็นพอร์ต Trace

ทั้งนี้นอกจาก PINSEL0, PINSEL1 และ PINSEL2 แล้วยังมีเรจิสเตอร์สำหรับควบคุมการทำงานของ GPIO อีก นั่นคือ IOPIN, IOSET, IOCLR และ IODIR โดยมีหน้าที่ดังตารางต่อไปนี้

เรจิสเตอร์	การทำงาน	ความหมาย
IOPIN0	อ่าน	อ่านสภานะของพอร์ต P0
IOPIN1	อ่าน	อ่านสถานะของพอร์ต P1
IOSET0	เขียน/อ่าน	กำหนดบิตที่ระบุของพอร์ต P0 ให้เป็น 1
IOSET1	เขียน/อ่าน	กำหนดบิตที่ระบุของพอร์ต P1 ให้เป็น 1
IOCLR0	เขียน/อ่าน	กำหนดบิตที่ระบุของพอร์ต P0 ให้เป็น 0
IOCLR1	เขียน/อ่าน	กำหนดบิตที่ระบุของพอร์ต P1 ให้เป็น 0
IODIR0	เขียน	กำหนดบิตที่ระบุว่าเป็น 1 ของพอร์ต P0 เป็นส่วนนำออก
IODIR1	เขียน	กำหนดบิตที่ระบุว่าเป็น 1 ของพอร์ต P1 เป็นส่วนนำออก

จากข้อมูลทั้งหมดทำให้เราทราบว่าการติดต่อกับพอร์ตภายนอกนั้นเราจะต้องกำหนดค่าการทำงานของแต่ละบิตของ P0 กับ P1 ได้อย่างไร และเราได้รู้จักคำสั่งพื้นฐานอีก 8 คำสั่ง ที่ใช้สำหรับอ่านค่าสถานะของพอร์ต กำหนดค่าบิตของพอร์ตให้เป็น1 กำหนดค่าบิตของพอร์ตให้เป็น0 และกำหนดบิตของพอร์ตให้ทำงานเป็นส่วนนำเข้าหรือส่งออก

เพื่อความเข้าใจผมขอยกตัวอย่างง่ายๆให้พอมองเห็นภาพ เช่น ถ้าเราต้องการกำหนดให้ P0.8 ถึง P0.15 เป็นพอร์ตส่งออก (output port) เราจะต้องกำหนดค่าดังนี้

```
PINSEL0 = 0x00000005; /* กำหนดให้ P0.0 ทำงานเป็น TxD และ P0.1 ทำงานเป็น RxD*/
IODIR0 = 0x0000FF00; /* 0000 0000 0000 0000-1111 1111 0000 0000 */
```

ถ้าเราต้องการหลอดแอลอีดีให้มีการติด/ดับสลับกันไปจะสามารถเขียนขั้นตอนได้ดังนี้

```
IOSET0 = 0x0000FF00;
หน่วงเวลา
IOCLR0 = 0x0000FF00;
หน่วงเวลา
```

ตัวอย่างโปรแกรม

จากทั้งหมดเมื่อเรานำมารวมกันเป็นโปรแกรมจะสามารถเขียนออกมาได้อย่างนี้

```
#include <LPC21xx.h>

void delay(int nCnt)
{
    int cnt, cnt2;
    for (cnt=0; cnt<nCnt; cnt++) {
         for (cnt2=0; cnt2<1000; cnt2++);
    }
}

int main(void)
{
    PINSEL0 = 0x00000005;
    IODIR0 = 0x0000FF00;
    for (;;) {
        IOSET0 = 0x0000FF00;
        delay(5000);
        IOCLR0 = 0x0000FF00;
        Delay(5000);
    }
}
```

สรุป

จากบทความนี้คงพอจะเป็นข้อมูลพื้นฐานได้ว่าเราจะสามารถกำหนดสภาวะการทำงานของพอร์ต และสั่งงานการทำงานของพอร์ตได้อย่างไร ในครั้งหน้าเราจะเพิ่มการเชื่อมต่อกับสวิทช์ร่วมกับหลอดแอลอีดีเพื่อสั่งให้แอลอีดีทำงานได้หลากหลายขึ้น โดยอาศัยตัวอย่างของการเขียนในครั้งนี้เป็นพื้นฐาน

สุดท้ายต้องขอขอบคุณทางบริษัทอีทีทีที่ยังคงสนับสนุนการทำงานของผม และขอขอบคุณครอบครัวที่ให้กำลังใจผมตลอดมา วันนี้ผมก็ต้องทำงานหนักมากขึ้น แต่ผมจะเขียนบทความให้ได้ทุกเดือน เพราะสิ่งที่ผมทำอยู่นี้เป็นความใฝ่ฝันของผม เพียงแต่เมื่อเราเติบโตขึ้น อะไรต่อมิอะไรก็นำพาเราไปทางโน้นทางนี้จนเราแทบจะไม่ได้อยู่บนทางเดินที่เราชอบ แต่อย่างไรก็ดีถึงผมจะต้องเดินออกนอกทางเดินที่ผมใฝ่ฝัน ผมก็ยังมีอีทีทีเป็นแหล่งเผยแพร่ความรู้ที่ผมพยายามศึกษา ถึงมันจะไม่มากมาย ไม่ยาก ไม่ลึกลับซับซ้อน แต่คงพอจะเป็นประโยชน์บ้างสำหรับคนเริ่มต้น หรือคนที่สนใจ