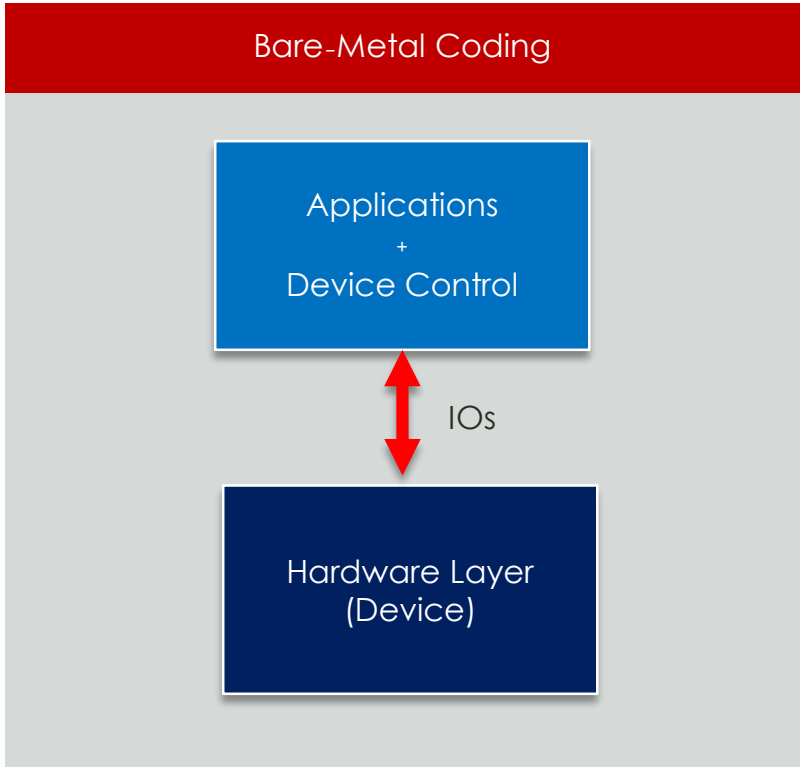
Embedded Systems Design and Applications

Lab3 : Character LCD Device Driver - I

ดร.ปนันท์ โพธิพันธุ์ทอง

Simple Device Control



การเขียนโปรแกรมบนโพรเซสเซอร์โดยตรงไม่ผ่านระบบปฏิบัติการ (Bare-Metal Programming) โดยทั่วไป ผู้พัฒนาอาจรวมโค้ดของแอพพลิเคชันและการควบคุมดีไวซ์เข้าไว้ด้วยกัน เพื่อความสะดวกในการเขียนโปรแกรมเป็นครั้งคราว

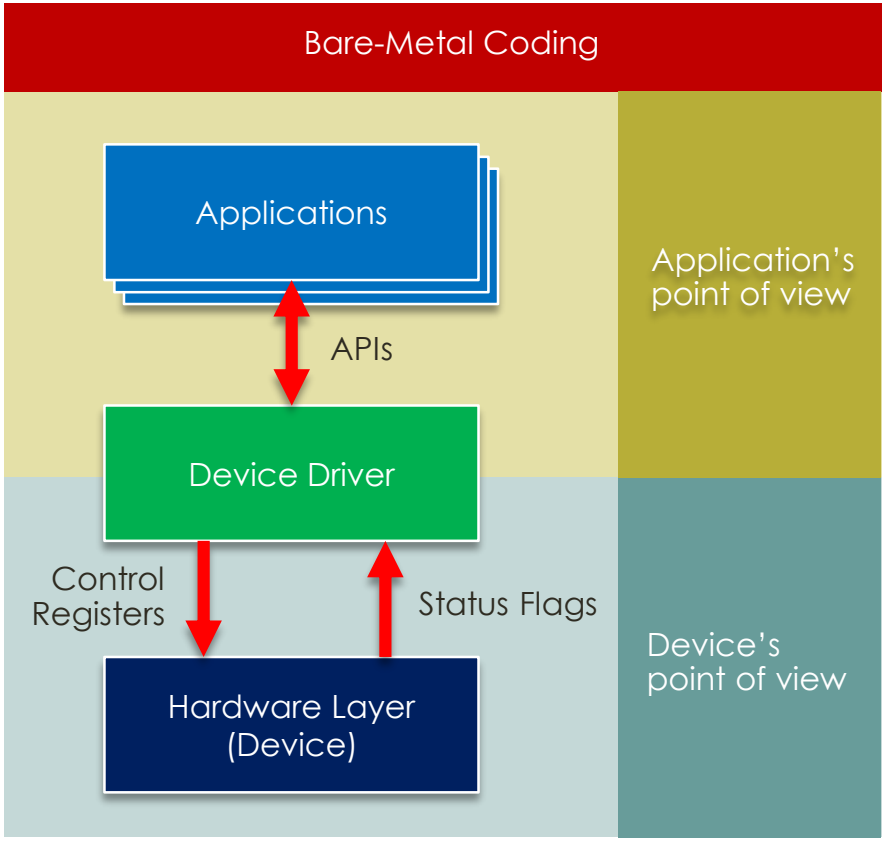
ส่งผลให้แอพพลิเคชั่นยึดโยงกับการควบคุมดีไวซ์ โค้ดที่ได้จึงมีโครงสร้างที่ไม่เหมาะสม (Unstructured Program) ยากต่อการอ่านเพื่อทำความเข้าใจและตรวจสอบการทำงานในภายหลัง อีกทั้งการพอร์ตโค้ดไปใช้งานบนฮาร์ดแวร์ระบบอื่น ๆ ก็เป็นไปได้ยาก

What is the device-driver?

ดีไวซ์ไดร์เวอร์ คือ “**ซอฟต์แวร์โมดูลที่ทำให้เกิดฟังก์ชันในการควบคุมอินพุตและเอาต์พุต**” ดีไวซ์ไดร์เวอร์ใช้ รีจีสเตอร์ควบคุมการทำงาน (Control Register) และแฟล็กสถานะ (Status Flag) ในการจัดการและควบคุมการทำงานของดีไวซ์

โดยทั่วไปแล้วผู้พัฒนาดีไวซ์ไดร์เวอร์สำหรับอุปกรณ์ใด ๆ ก็ตาม จำเป็นต้องมีความรู้ในเชิงลึกเกี่ยวกับอุปกรณ์ตัวนั้น ๆ ในทางกลับกันผู้ใช้งานดีไวซ์ไดร์เวอร์ไม่จำเป็นต้องทราบรายละเอียดใด ๆ ทางฮาร์ดแวร์ของดีไวซ์นั้นเลย ก็สามารถใช้งานดีไวซ์ได้อย่างสะดวกผ่าน APIs (Application Programming Interface) ที่ดีไวซ์ไดร์เวอร์จัดเตรียมไว้ให้

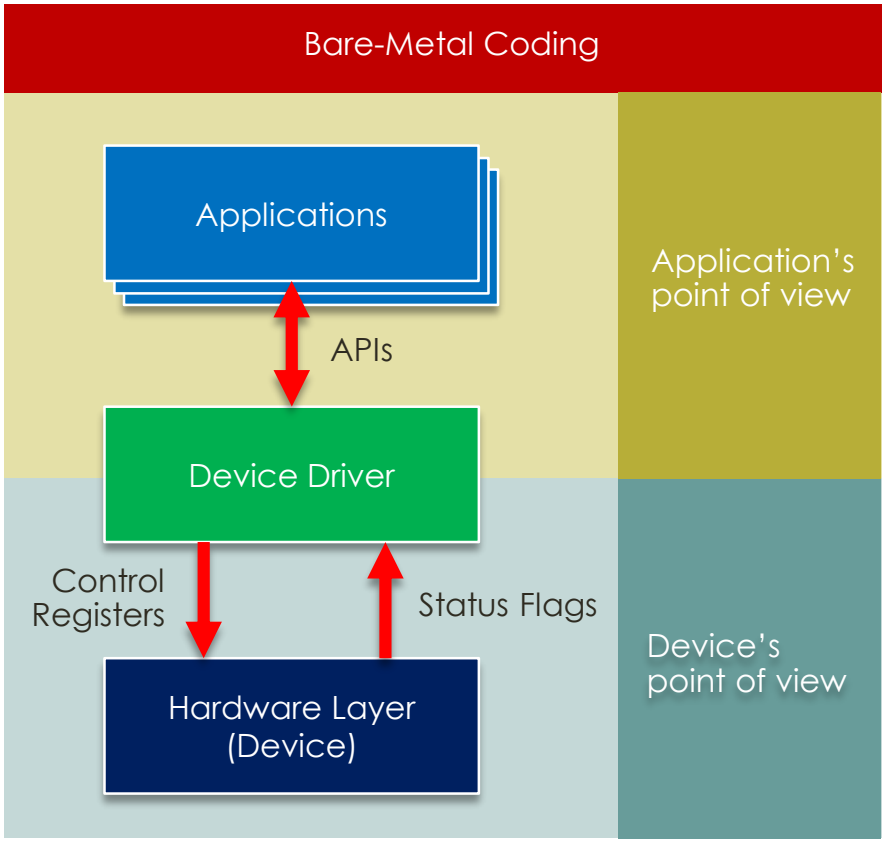
Device Driver : Application's Point of View



การแยกการควมคุมดีไวซ์ออกจากแอพพลิเคชั่น ในรูปแบบของ "ดีไวซ์ไดรเวอร์" มีข้อดีดังต่อไปนี้

- แอพพลิเคชั่นมากกว่า 1 ตัวสามารถใช้ดีไวซ์ร่วมกันได้ (ทำการซิงโครไนเซชั่นระหว่างแอพฯ)
- การที่แอพพลิเคชั่นไม่ยึดติดอยู่กับดีไวซ์ ทำให้สามารถพอร์ตแอพพลิเคชั่นไปใช้งานบนฮาร์ดแวร์อื่น ๆ ได้ง่าย
- แอพพลิเคชั่นไม่ได้รับผลกระทบเมื่อมีการเปลี่ยนดีไวซ์
- ผู้พัฒนาแอพพลิเคชั่นไม่มีความจำเป็นต้องทราบรายละเอียดการทำงานของดีไวซ์ฮาร์ดแวร์ในเชิงลึก ทำแค่เพียงการใช้งานดีไวซ์ผ่าน APIs ของดีไวซ์ไดร์เวอร์เท่านั้น
- เวลาในการพัฒนาแอพพลิเคชั่นลดลง เนื่องจากผู้พัฒนาไม่จำเป็นต้องเขียนดีไวซ์ไดร์เวอร์เอง

Device Driver : Device's Point of View



ดีไวซ์ไดร์เวอร์ คือ ซอฟต์แวร์โมดูลที่ควบคุมการทำงานของดีไวซ์ ผ่าน Control Registers และตรวจสอบสถานะของดีไวซ์ผ่าน Status Flags

ผู้พัฒนาดีไวซ์ไดร์เวอร์จำเป็นต้องมีความรู้ความเข้าใจในเชิงลึกเกี่ยวกับดีไวซ์ที่กำลังทำงานด้วย

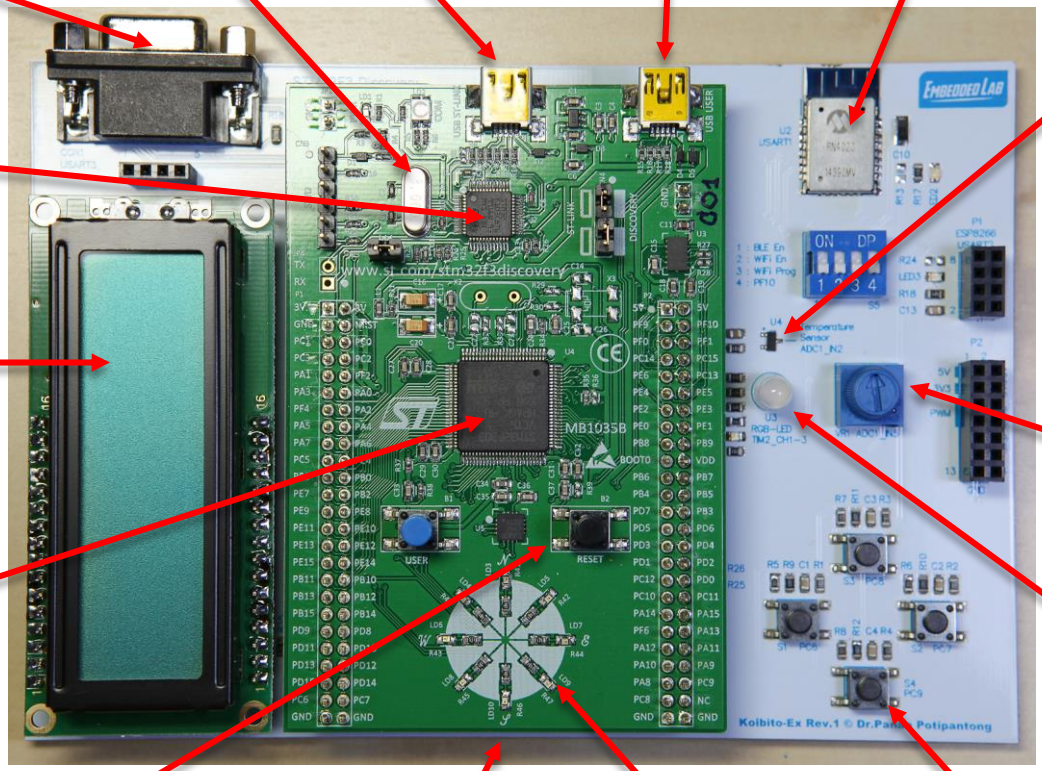
ผู้พัฒนาดีไวซ์ไดร์เวอร์ต้องจัดเตรียม APIs ฟังชั่นที่ครอบคลุมการทำงานของดีไวซ์ในทุกด้าน เพื่อให้แอพพลิเคชั่นสามารถเรียกใช้งานได้อย่างสะดวก

Character LCD Device Driver

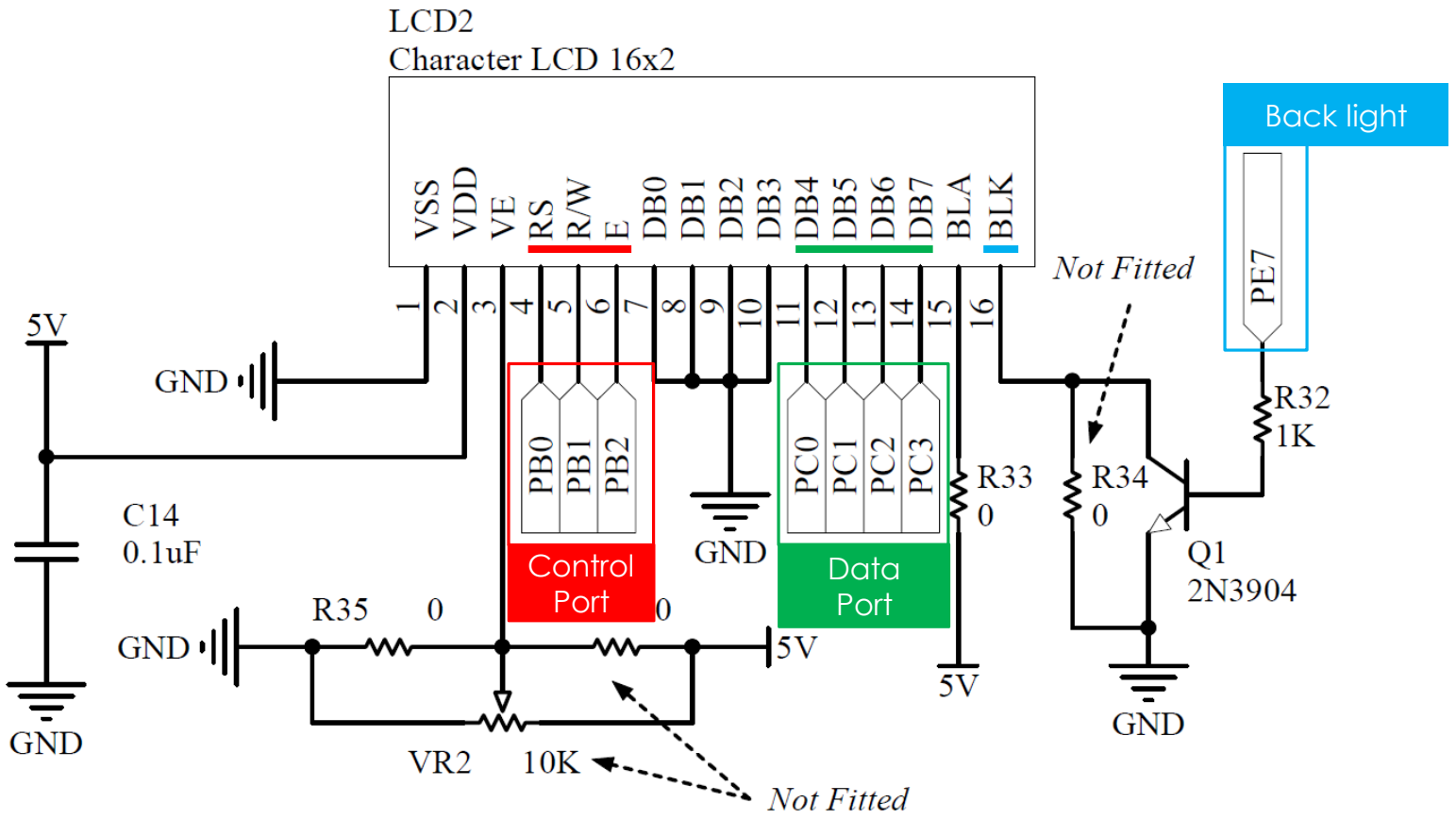
จอแอลซีดีแบบตัวอักษร (Character LCD) ถือว่าเป็นดีไวซ์พื้นฐานชนิดหนึ่งสำหรับระบบสมองกลฝังตัว ถูกใช้ในการแสดงผลข้อมูลที่เป็นตัวอักษรเป็นหลัก การควบคุมการทำงานจอแอลซีดีแบบตัวอักษรนั้น นักศึกษาจำเป็นต้องทราบรายละเอียดทางฮาร์ดแวร์และการควบคุมรีจีสเตอร์ต่าง ๆ เพื่อให้สามารถพัฒนาดีไวซ์ไดร์เวอร์ให้ทำงานได้อย่างมีประสิทธิภาพ

โดยทั่วไปจอแอลซีดีแบบตัวอักษรจะถูกควบคุมด้วย Controller ที่มีต้นแบบมาจากชิพ HD44780 ของฮิตาชิ ทำให้เราสามารถใช้งานจอแอลซีดีแบบตัวอักษรรุ่นอื่น ๆ ได้โดยใช้การวิธีการควบคุมแบบเดียวกัน

Koibito-Ex Development Board



Koibito-Ex LCD Interface



LCD Controller Instructions (1)

Instruction	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Exec. Time
Clear display	0	0	0	0	0	0	0	0	0	1	1.53 ms
Return home	0	0	0	0	0	0	0	0	1	-	1.53 ms
Entry mode set	0	0	0	0	0	0	0	1	I/D	S	39 us
Display on/off control	0	0	0	0	0	0	1	D	C	B	39 us
Cursor or Display shift	0	0	0	0	0	1	S/C	R/L	-	-	39 us
Function set	0	0	0	0	1	DL	N	F	-	-	39 us
Set DDRAM address	0	0	1	7-bits DDRAM Address							39 us
Write data to DDRAM	1	0	8-bits ASCII Data								43 us

* ASCII : American Standard Code for Information Interchange

* - : Don't care

LCD Controller Instructions (2)

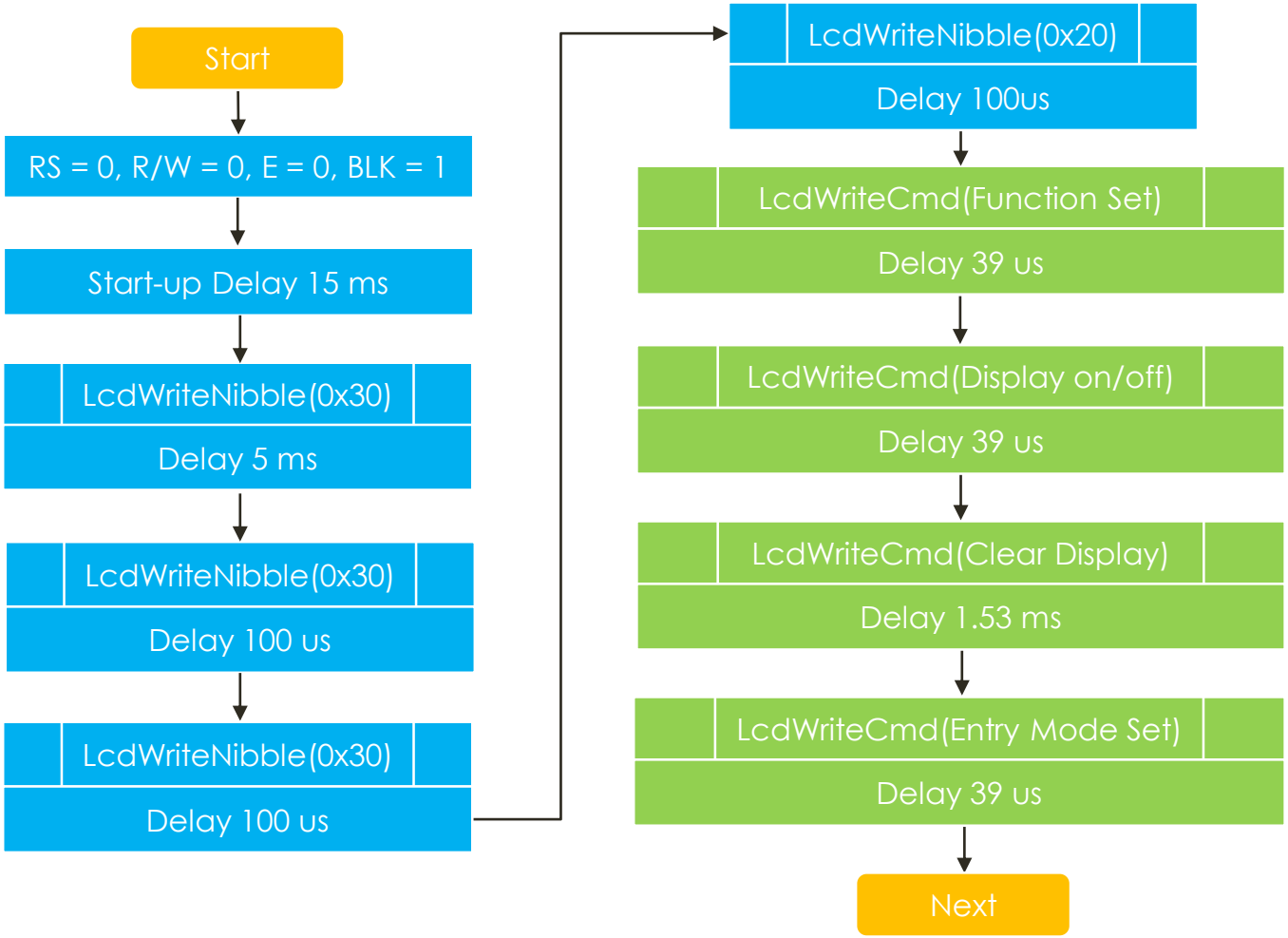
	0	1
DL	4-bit Interface *	8-bit interface
N	1 line	2 lines *
F	5x7 dots *	5x10 dots
D	Display off	Display on
C	Cursor off	Cursor on
B	Cursor blink off	Cursor blink on
I/D	Decrement cursor position	Increment cursor position *
S	No display shift *	Display shift

* Recommended value

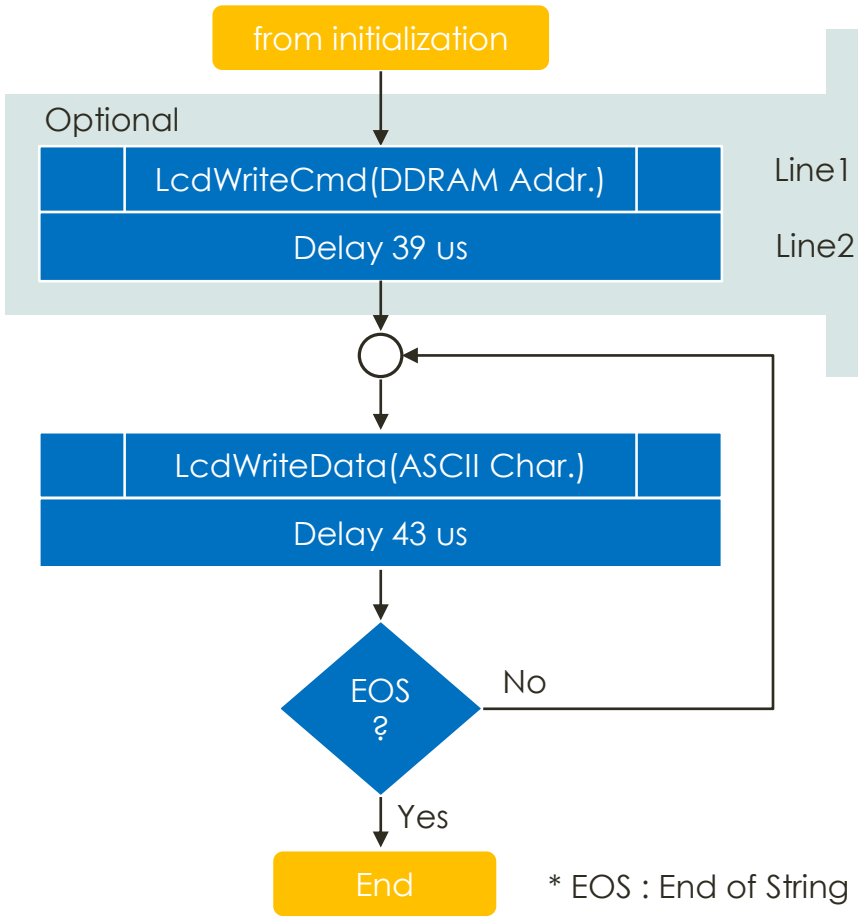
Device Initialization

Device Configuration

LCD Initialization Flow



LCD Write Character Flow

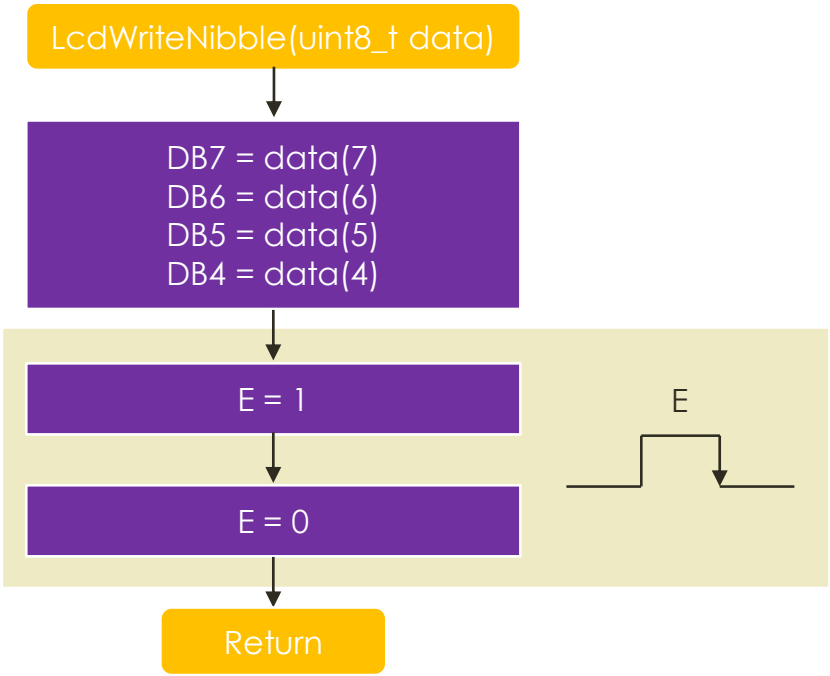


LCD 16x2 DDRAM Address (Hex)

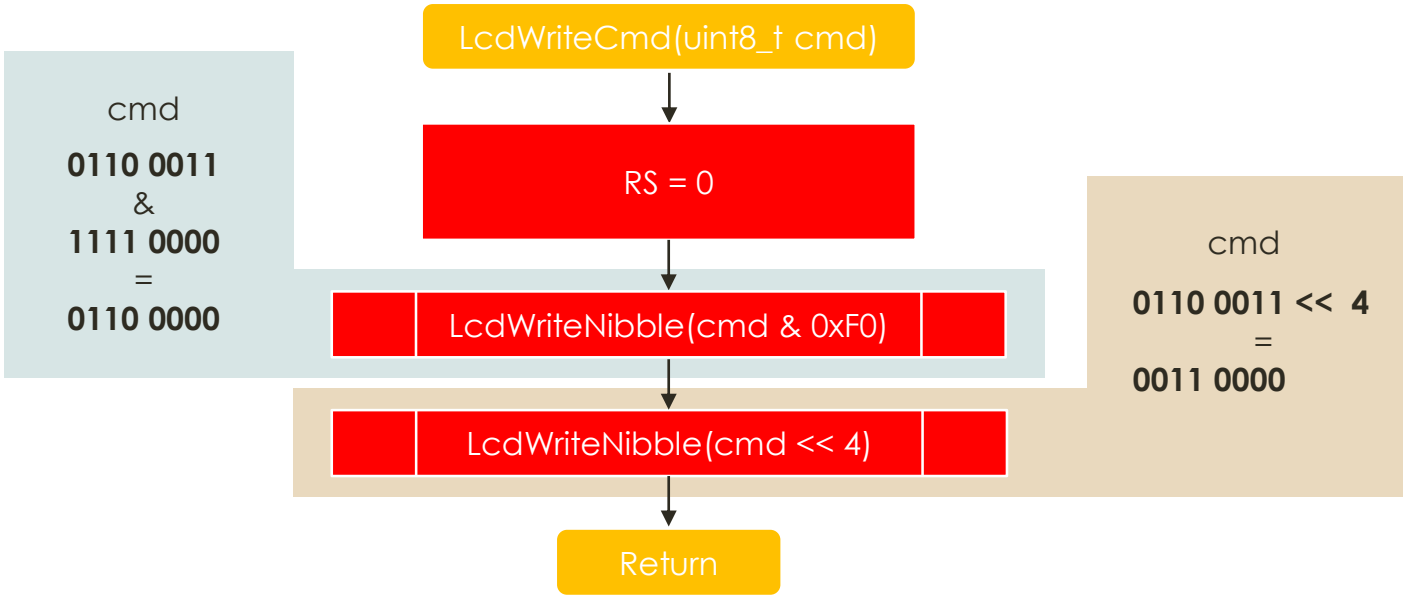
Line1	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
Line2	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F

* EOS : End of String

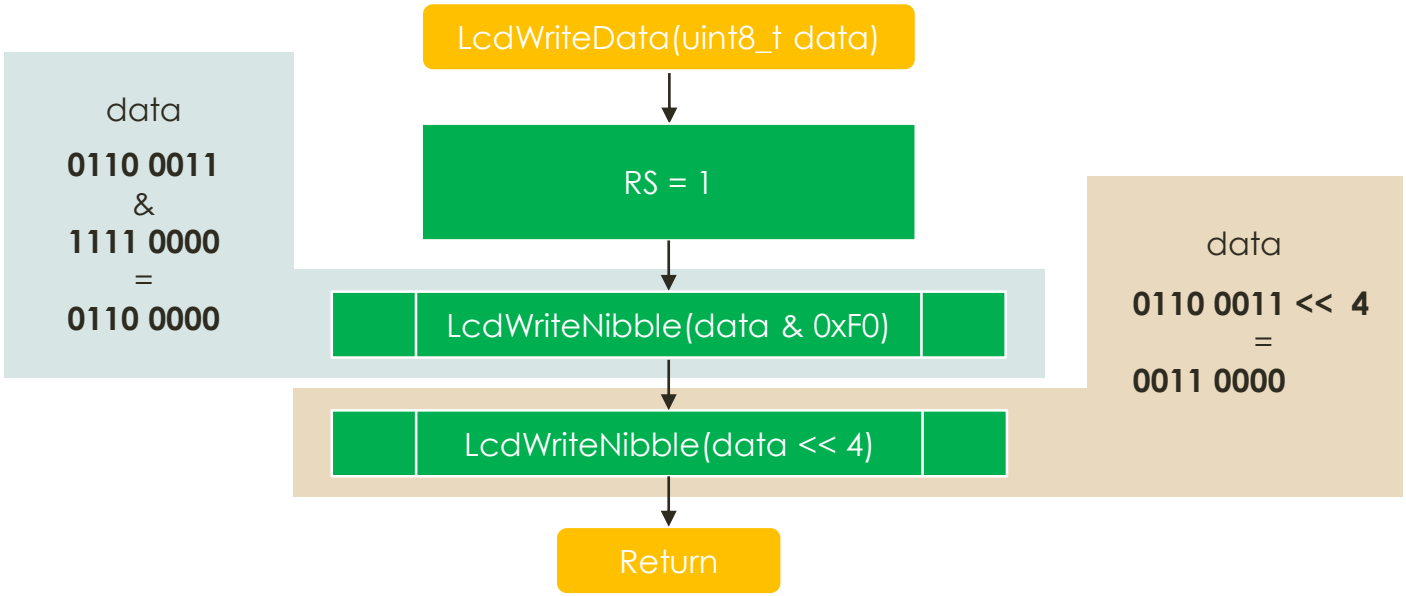
LcdWriteNibble function



LcdWriteCmd function



LcdWriteData function



C Coding Tips and Tricks #3

เราสามารถใช้ #define ประกาศค่าคงที่ (Constants) ที่ใช้ในโปรแกรม โดยตั้งชื่อให้มีความหมายสอดคล้องกับการใช้งาน เช่น

```
#define   TRUE    (1)
#define   FALSE   (0)
#define   LCD2_CTR_GPIO_PORT   GPIO_PORTB
```

นอกจากนี้สามารถยังใช้ #define สร้าง Macro เพื่อเป็นตัวแทน Expressions ที่ซับซ้อนได้ เช่น

```
#define   RS(x)   HAL_GPIO_WritePin(LCD2_CTR_GPIO_PORT, LCD2_RS_GPIO_PIN, x)
```