Lab 3 – Liquid Crystal Display (LCD)

Graduate Teaching Assistant:

Francisco E. Fernandes Jr.

feferna@okstate.edu

Khuong Vinh Nguyen

Khuong.v.nguyen@okstate.edu

School of Electrical and Computer Engineering

Oklahoma State University

Spring 2019

Lab 3 – Schedule and Objectives

- **Pre-lab Assignment (10 points):**
  - **For Monday labs:** Due on April 01, 2019 (week 1).
  - **For Wednesday labs:** Due on April 03, 2019 (week 1).
- **Lab Demo Questions (10 points):**
  - **For Monday labs:** Due on April 15, 2019 (week 3).
  - **For Wednesday labs:** Due on April 17, 2019 (week 3).

Lab 3 will take a total of three weeks!

- **First Objective (50 points):**
  - **Due date (week 2):**
    - **For Monday labs:** April 08, 2019.
    - **For Wednesday labs:** April 10, 2019.
  - Write a C program to display the first six letters of your last name in the LCD.
- **Second Objective (14 points):**
  - **Due date (week 3):**
    - **For Monday labs:** April 15, 2019.
    - **For Wednesday labs:** April 17, 2019.
  - Create a generic LCD driver in C to display any letter in any display position.

Lab 3 – Schedule

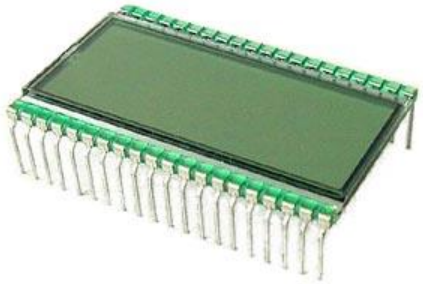
Description	Points	Due date for Monday labs	Due data for Wednesday labs
Pre-lab assignment	10 points	Apr. 01	Apr. 03
Attendance and Class Participation	8 points	Apr. 01, 08, 15	Apr. 03, 10, 17
Code organization	8 points	N/A	N/A
Lab demo questions	10 points	Apr. 15	Apr. 17
First Objective	**50 points**	**Apr. 08**	**Apr. 10**
Second objective	**14 points**	**Apr. 15**	**Apr. 17**
Total:	**100 points**		

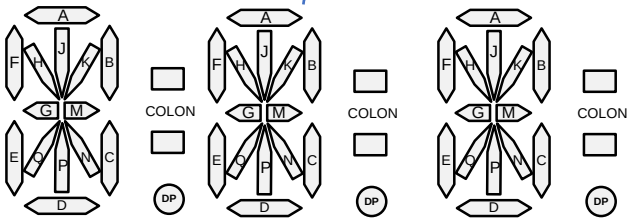
One more thing…

- **Partial credits will be given in this lab!**
  - The TAs will answer your theoretical questions about the lab.
  - However, if the TA write any line of code in your lab, you will receive partial credits!
    - This includes correcting bugs in your code!!!!

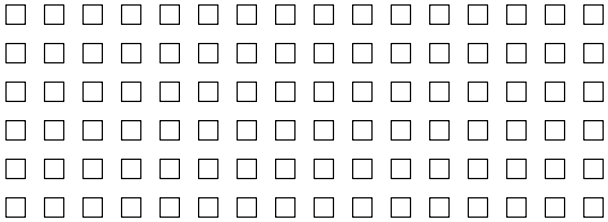
- **To ensure that no code is being copied between students, the TA will randomly ask the meaning of any line of code in your code!**
  - If the student doesn't know the answer, it is because that line of code was copied from someone else!
    - In this case, the student will receive partial credits!
  - If the student doesn't know the meaning of multiple lines of code, it means all of his or her code was copied from someone else!
    - In this case, an **F** will be given to the entire lab!

Types of LCD





Segment LCD

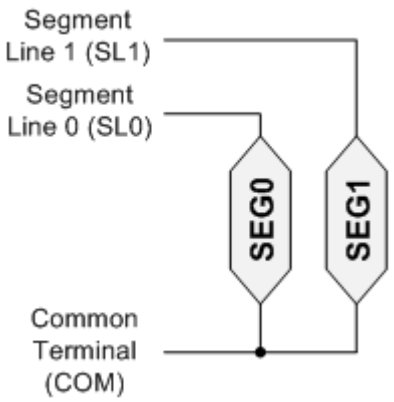


Dot Matrix LCD

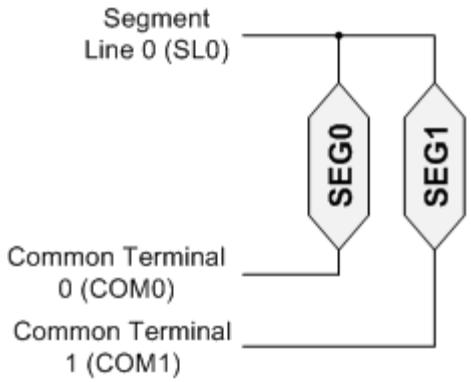
Multiplexed LCD drive

Duty Ratio

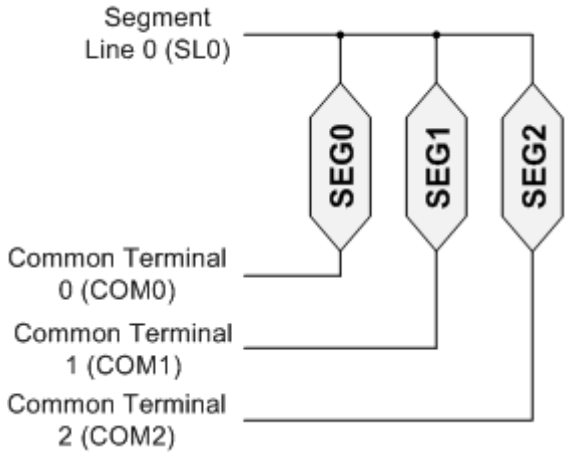
- how long each segment is activated during each frame



Duty = 1



Duty = 1/2



Duty = 1/3

$$Duty\ Ratio = \frac{1}{Number\ of\ Common\ Terminals}$$

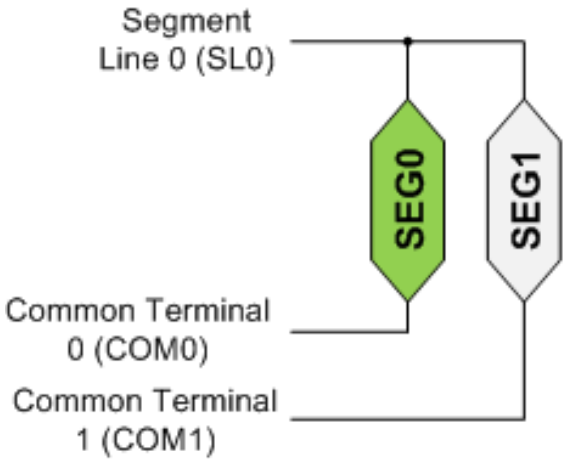
Drive Bias

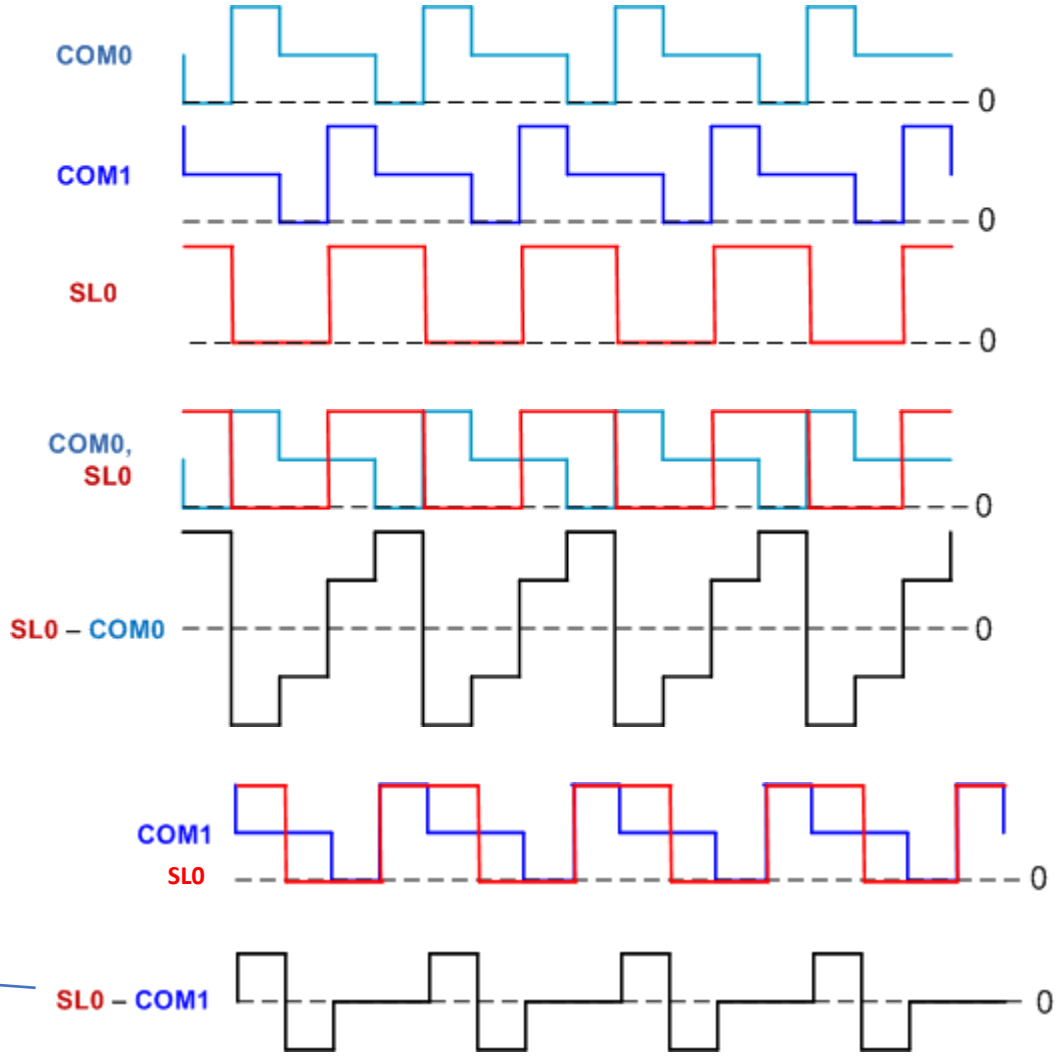
- the number of voltage levels used

$$Bias = \frac{1}{Number\ of\ Voltage\ Levels\ - 1}$$

Multiplexed LCD drive

Duty Ratio = ½, SEG0 ON, SEG1 OFF



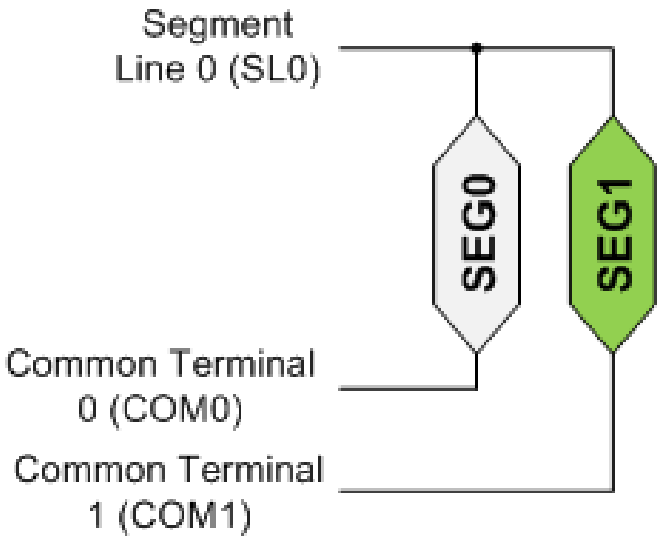


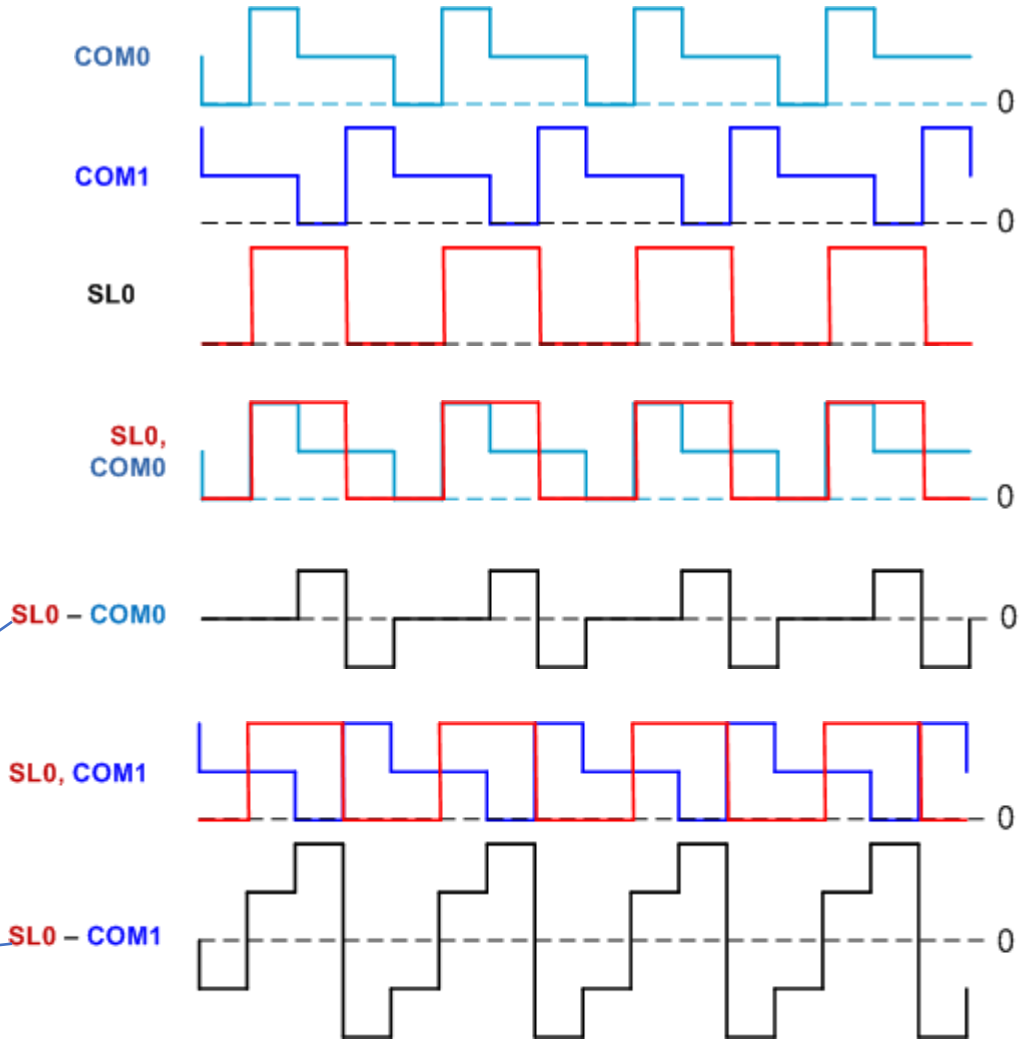
SEG0 is switched on and off quickly.

SEG1 will never turn on.

Multiplexed LCD drive

Duty Ratio = ½, SEG0 OFF, SEG1 ON

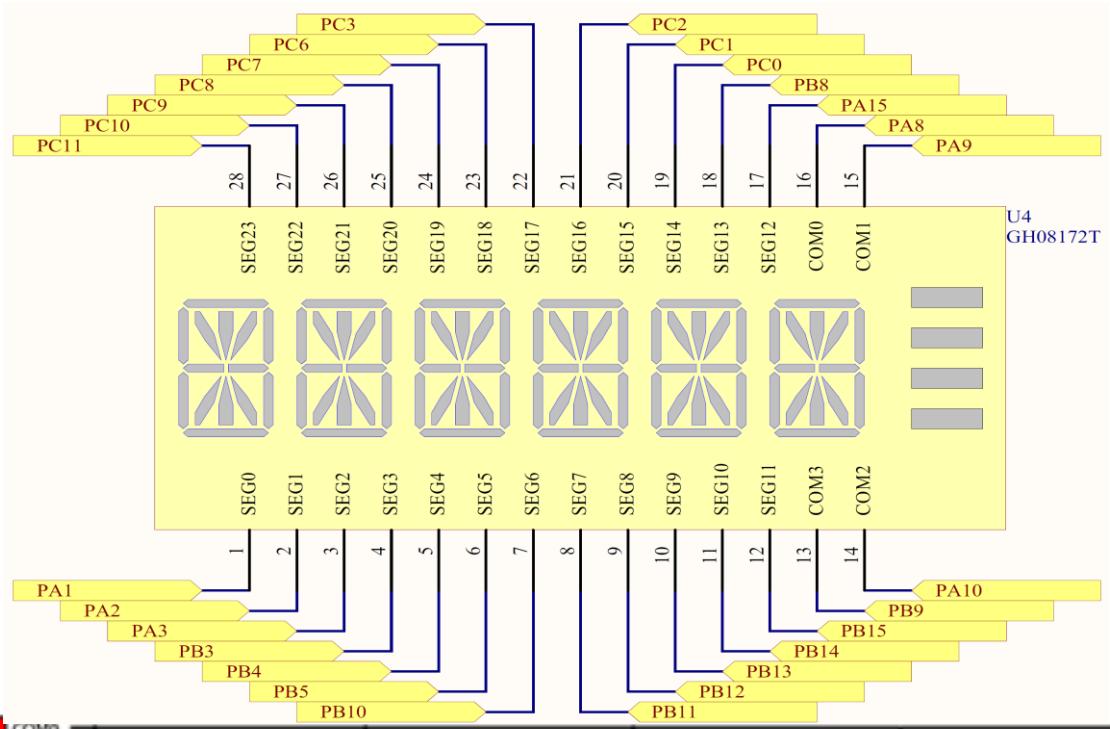


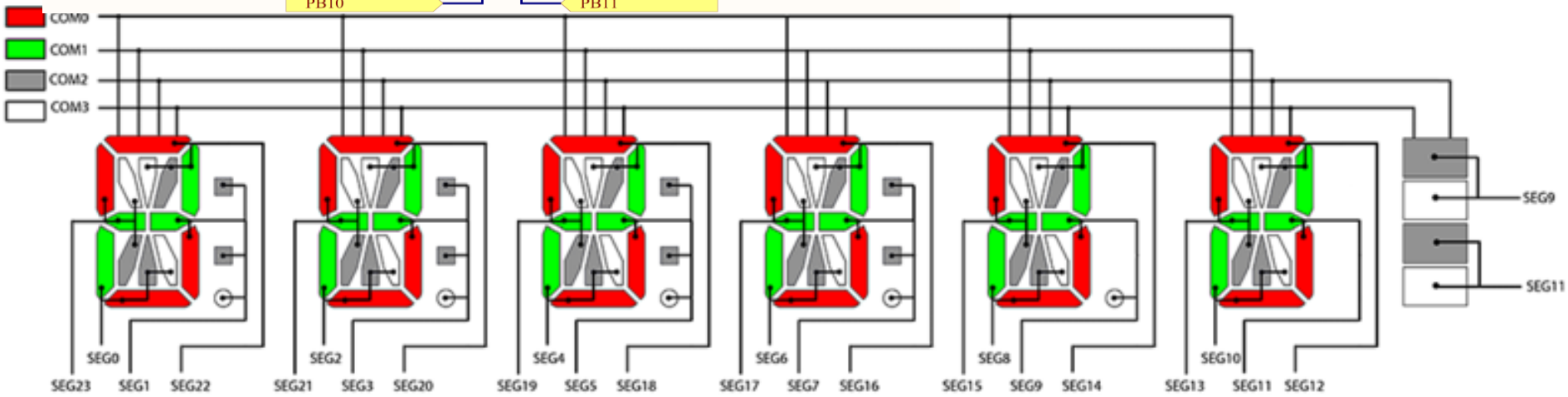


SEG0 will never turn on!

SEG1 is switched on and off quickly.

LCD on the ST32L4 Discovery Kit





Duty Ratio = ¼

24 segment lines (SEG0-SEG23)

4 common terminals (COM0-COM3)

6 digits and **4** bars

96 pixel segments

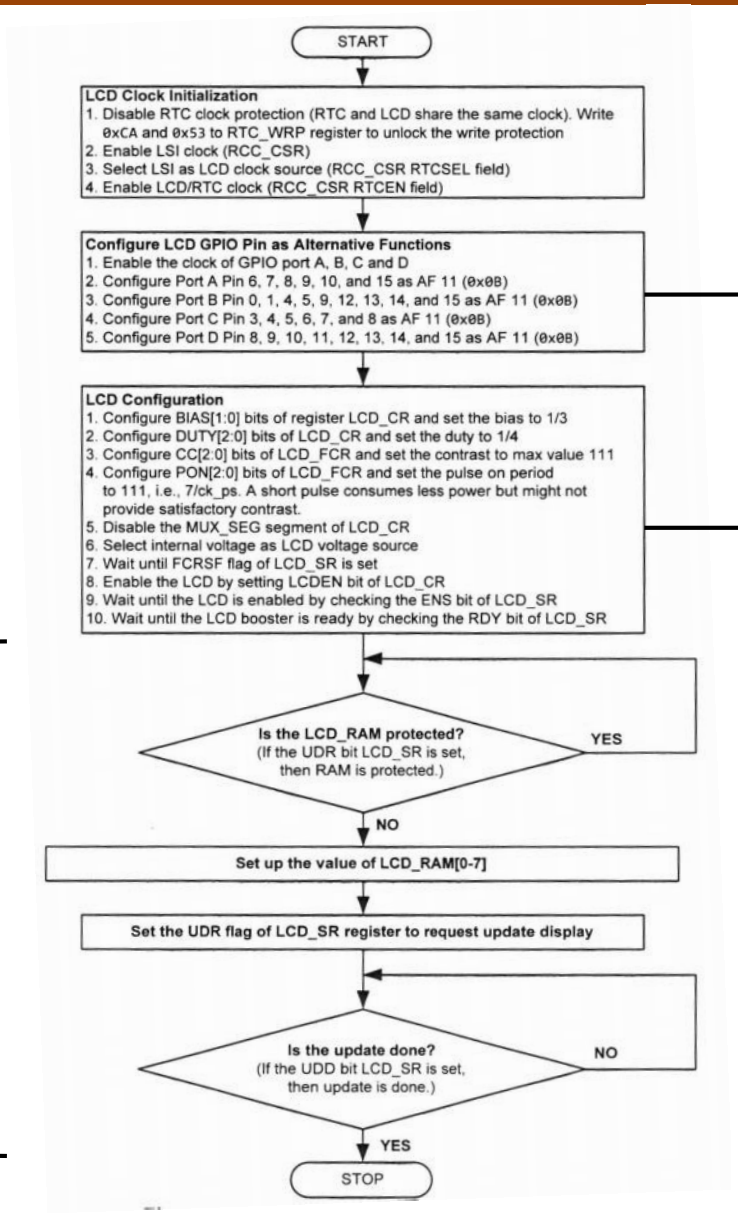
Lab Assignment – First Objective

- **Write a C program to display the first six letters of your last name in the LCD.**
  - A startup code is provided on D2L under Lab 3 section (filename: *Lab 3 – Startup Code.zip*) containing the following files: **LCD.c**, **LCD.h**, **main.c**, and **stm32l476.h**.
    - **Download** and **extract** the startup code.
    - Create a new C Project using System Workbench for STM32 IDE.
    - Move the files **main.c** and **LCD.c** to your project's **src** folder.
    - Move the files **LCD.h** and **stm32l476xx.h** to your project's **inc** folder.
  - **For the first objective, all your code should be written in the *LCD.c* file.**
  - **You are required to complete four functions:**
    - ***LCD_PIN_Init***(): enables GPIO clocks and configures GPIO pins as the alternative function 11 (Pre-Lab, Questions 1 to 4) .
    - ***LCD_Configure***(): performs the LCD configuration in the flowchart (Pre-Lab, Question 5).
    - ***LCD_Display_Name***(): display the first six letters of your last name by setting up the LCD_RAM registers (Pre-Lab, Question 6).

Lab Assignment – Second Objective

- **Create a generic LCD driver in C to display any letter in any LCD position.**
  - You are required to complete **LCD_WriteChar()** function located in the **LCD.c** file.
  - This objective is **VERY CHALLENGING!**
  - TAs will not provide help with this part!
  - **COPIED CODE WILL BE PUNISHED WITH AN F IN THE ENTIRE LAB!**
  - The **textbook section 17.3.2** can help you with this part!

Lab flowchart



LCD_PIN_Init()
Due today!

LCD_Configure()
Due next class!

LCD_Display_Name()
Due next class!

LCD_PIN_Init()

1. Enable the clock of GPIO port A, B, C and D.
2. Configure PA 6, 7, 8, 9, 10, 15 as Alternative Function 11 (0x0B).
3. Configure PB 0, 1, 4, 5, 9, 12, 13, 14, 15 as Alternative Function 11 (0x0B).
4. Configure PC 3, 4, 5, 6, 7, 8 as Alternative Function 11 (0x0B).
5. Configure PD 8, 9, 10, 11, 12, 13, 14, 15 as Alternative Function 11 (0x0B).

```
GPIOx->MODER &= ~(MASK);
GPIOx->MODER |=  MASK;

GPIOx->AFR[0] &= ~MASK;
GPIOx->AFR[0] |= MASK;

GPIOx->AFR[1] &= ~MASK;
GPIOx->AFR[1] |= MASK;

// The GPIO output speed can be set to “low speed”
GPIOx->OSPEEDR &= ~(MASK);

// GPIOx Push-Pull: No pull-up, no pull-down (00)
GPIOx->PUPDR &= ~MASK;
```

This function is DUE TODAY!

LCD_Configure()

```
// 1. Configure BIAS[1:0] bits of register LCD_SR and set the bias to 1/3
LCD->CR ; //BIAS[1:0]: 00=1/4; 01=1/2; 10=1/3

// 2. Configure DUTY[2:0] bits of LCD_CR and set the duty to 1/4
LCD->CR ; //DUTY[2:0]: 000=Static; 001=1/2; 010=1/3; 011=1/4; 100=1/8

// 3. Configure CC[2:0] bits of LCD_FCR and set the contrast to max value 111
LCD->FCR ;

// 4. Configure PON[2:0] bits of LCD_FCR and set the pulse on period to 111.
LCD->FCR ;     // PON[2:0] = 0x111

// 5. Diable the MUX_SEG segment of LCD_CR
LCD->CR ;

// 6. Select internal voltage as LCD voltage source
LCD->CR ; // 0 = internal source, 1 = external source (VLCD pin)

// 7. Wait until FCRSF flag of LCD_SR is set
while ((LCD->SR & MASK) == 0); // Wait until FCRSF flag is set

// 8. Enable the LCD by setting LCDEN bit of LCD_CR
LCD->CR ;

// 9. Wait until the LCD is enabled by checking the ENS bit of LCD_SR
while ((LCD->SR & MASK) == 0); // ENS is set by hardware automatically

// 10. Wait until the LCD booster is ready by checking the RDY bit of LCD_SR
while ((LCD->SR & MASK) == 0); // Loop until step-up converter is ready to provide the correct voltage.
```

This function is NEXT CLASS!

LCD_Configure()

Use the **LCD Register Map** in order to find the correct bit positions!

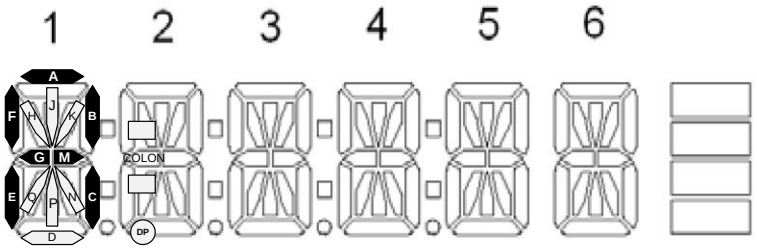
Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	LCD_CR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	BUFEN	MUX_SEG	BIAS[1:0]		DUTY [2:0]			VSEL	LCDEN
	Reset value																									0	0	0	0	0	0	0	0
0x04	LCD_FCR	Res.	Res.	Res.	Res.	Res.	Res.	PS[3:0]				DIV[3:0]				BLINK[1:0]		BLINKF[2:0]			CC [2:0]			DEAD [2:0]			PON [2:0]			UDDIE	Res.	SOFIE	HD
	Reset value							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0
0x08	LCD_SR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	FCRSF	RDY	UDD	UDR	SOF	ENS
	Reset value																											1	0	0	0	0	0
0x0C	LCD_CLR	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	UDDC	Res.	SOFC	Res.
	Reset value																													0		0	

LCD_Display_Name()

```
// Is the LCD_RAM protected?
// If the UDR bit in LCD_SR is set, then RAM is protected
while ((LCD->SR & MASK) != 0); // Wait for Update Display Request Bit

// Set up the value of LCD_RAM[0-7] with your last name
LCD->RAM[0] ;
LCD->RAM[1] ;
LCD->RAM[2] ;
LCD->RAM[3] ;
LCD->RAM[4] ;
LCD->RAM[5] ;
LCD->RAM[6] ;
LCD->RAM[7] ;

// Set the UDR flag of LCD_SR register to request update display
LCD->SR |= MASK;

// Is the update done?
// If the UDD bit in LCD_SR is set, then update is done.
while ((LCD->SR & MASK) == 0); // Wait for update display done
```

This function is NEXT CLASS!

LCD_Display_Name()




```
// Set up the value of LCD_RAM[0-7] with your last name
    LCD->RAM[0] |= 0x00C00018;
    LCD->RAM[2] |= 0x00C00008;
```

	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
LCD_RAM[0]	4E	4G	3M	3B		6G	5M	5B	1M	1B					6E		3E	3G	2M	2B			6B	6M		2E	2G	1E	1G			
LCD_RAM[1]																													5E	5G	4M	4B
LCD_RAM[2]	4D	4F	3C	3A		6F	5C	5A	1C	1A					6D		3D	3F	2C	2A			6A	6C		2D	2F	1D	1F			
LCD_RAM[3]																													5D	5F	4C	4A
LCD_RAM[4]	4P	4Q	3 Col	3K		6Q	3 Bar	5K	1 Col	1K					6P		3P	3Q	2 Col	2K			6K	1 Bar		2P	2Q	1P	1Q			
LCD_RAM[5]																													5P	5Q	4 Col	4K
LCD_RAM[6]	4N	4H	3 DP	3J		6H	2 Bar	5J	1 DP	1J					6N		3N	3H	2 DP	2J			6J	0 Bar		2N	2H	1N	1H			
LCD_RAM[7]																													5N	5H	4 DP	4J