Sorry i decided not to write and further waste my time here since its a only a miracle that will make Touchgfx working with custom LCD interface , but still to take you on right direction i am writing this so that it may be beneficial for rest of professional who are intending to use Touchgfx

See below clarifications:

1. This "https://support.touchgfx.com/docs/development/touchgfx-hal-development/scenarios/scenarios-fmc" is same useless link after reading which i decided to evaluate Touchgfx for one of my project. This is the same single page document which i referred in my first post. This page seems to be written so poorly and by trainee engineer with Gun pointing to his head with threatened to complete this help pay by end of the day. It looks more like to be an assignment rather than help file which is approved without quality check.
    1. First of all this page says you only need to modify 2 or 3 functions along with your own lcd driver. This page even does not talk about in which file this functions resides. So that is up to developer to identify where this functions among the 100s of file that cubemx and touchgfx generates. It would be much more handy reference if it could have been there.
    2. Are touchgfx designer and ST fellows yourself are confident enough that just going thru this user will be able to implement Touchgfx ?
    3. There are may sudo code are are left on user , for example documents says "developers to implement a custom TouchGFX Application Tick driver that signals OSWrappers::signalVSync() to unblock the TouchGFX Engine" now
    4. what's that Tick driver ? Are you expect newbie to know this by default, even your document does not talks about it much. Also where to place in in main.c or where? God only knows
    5. Further OSWrappers::signalVSync() function is of c++ file class , for a newbies it is difficult to call it in c.
    6. Why don't you create a function in touchgfx framework to call this OSWrappers::signalVSync(). User will call this after particular interval.
    7. You refer lvgl gu implementation / porting method to further refine your code. It will definatly help you to make integration simpler
2. Nearly all TFT LCD weather spi, fmc , parallel etc. have wide range of library available , all such library necessary implements Setpixel function which takes x,y and color from user and set that on display. So function `void  TouchGFXHAL::flushFrameBuffer(const Rect& rect)` needs to be further expanded and broken down to accept only Setpixel function.

    There are many issues to implement `TouchGFXHAL` function even though we know how to interact with our display. For example I have 16 bit color format R565 for my lcd and frame buffer is declared as

    uint32_t frameBuf[(128 * 160 * 2 + 3) / 4]
    LOCATION_ATTRIBUTE("TouchGFX_Framebuffer");

    Now how to retrieve 16bit color from uint32_t is unanswered even that need some computing like bit shift and consume time simplest way is to declare frameBuf as a array of uint1 ie same as color format which makes it easy to read it.

    However you have narrated procedure to retrieve and set color from frambuffer at https://support.touchgfx.com/docs/basic-concepts/framebuffer

```
uint32_t pixelAddress = x + y * WIDTH;
framebuffer[ pixelAddress ] = newColor;
uint32_t pixelAddress = x + y * WIDTH;
framebuffer[ pixelAddress ] = darken( framebuffer[ pixelAddress ] );
```

Have you ever tried this code, what is `darken` seems to be not because it's simply does not works. So you have left much more for user to create mess with it . At least instead of pseudo code you should provide working code snippet.

You should implement `void  TouchGFXHAL::flushFrameBuffer(const Rect& rect)` function as below

i. Put a loop to set GRAM pixel by pixel.
ii. Inside loop make a function to retrieve color as per color format ie 16bit, 24bit etc
iii. And ask user to implement setpixel function and simply pass this data to their own lcd.

Following routine will explain it further

See below demo code 1

```
void  TouchGFXHAL::flushFrameBuffer(const Rect& rect)

{

// Display test pixel by pixel
   for(int x = 0; x < LCD_WIDTH; x++) {
      for(int y = 0; y < LCD_HEIGHT; y++) {
         uint16_t color565 = test_img_128x128[y][x];
         // fix endiness
         // implemented by touchgfx as per color bits
         Uint16 color565 = GetcolorfromFramebusfferat(x,y);
         LCD_DrawPixel(x, y, color565);   // To be implemented by user
      }
   }
}
```

Demo code 2 will be for bulk transfer to lcd

```
void  TouchGFXHAL::flushFrameBuffer(const Rect& rect)
{
          // Touchgfx routinr=e to convert 32bit frame buffer to desired color bit conversion for
          example here // 32bit buffer to array of bytes as data
          LCD_DrawImage(uint16_t x, uint16_t y, uint16_t w, uint16_t h, uint8_t* data) ; // To be
          implemented by user
}
```

3. You said "you are working on an F429-DISCO SPI Application Template" this is seriously wrong thing. There are certain templates are available like what you said regarding example

of STM32L496-DISCO Application template. There are many issues with this which you will understand when you start thinking from user perceptive rather than developers mind set

1. How can one understand exactly what is change in your provided template wrt original touchgfx and cubemx code. It is bit difficult to understand the code flow when it is written by third person. Even one try to do it with text compare tool each and every time cubemx and touchgfx updated versions comes it generates the code in differently so it become further difficult to identify code that needs to be change wrt original generated code.
2. Using F429-DISCO board is itself a wrong start. When a newbie sees your demo where you select the board and load default initialisation it creates mess for all board functionality initialisation code. Now in practically no one professional developer is going to use your development board for their final product still today all development boards are more or less used by youtubers and trainers for giving trainings. I really wonder cost of your certain development boars are upto $1000. And if you create demo of touchgfx who will purchase it just for testing your demo since it is unusable for his customised product. Even if try to prepare design in line with your development board all spare parts you have used will cost $5000 with shipping from various countries. And I doubt it will be made available for small quantity and that with custom duties. On other hand there are many cheap development boards are available from Chinese or Japanese supplier (rather real talented developers) in which they use all functionality of your development board with cheaper and easily available components like Sram, Nand flash etc. So basically I mean to say that you should always use bare minimum development board for creating such demo like nucleon board with manual configuration in cubemx.
3. To amplify above let us start discussing about one of my product I designed in past for medical industry for giving treatment to certain physically challenged people. I had options as shortlisted below
   i. Option 1 : Use Stm32F4 / STM32F7 board + LTDC + Touchscreen which attracts purchase of your development bard and then test it on bench for project requirement and the based on that develop new design (time cost of $200 as design cost) + PCB layout design ($100) + PCB fabrication ($ 200) + Components ($100) + RGB display with touch screen ($100) + Assembly cost ($50). **So Total Cost = $850**
   ii. **Option 2 :** Same as option 1 except laptop for gui
   iii. **Option 3 :** Standard Arduino Mega board ($10) + Cheapest yet powerful core to duo intel based touchscreen laptop ($ 200) with ESP8266 ($2) + HC05 ($2) Bluetooth module for wireless integration with board
   iv. **Option 4 :** same as option 3 Arduino Mega ($10) + 10.1” android tablet ($50 china) + Freelancer gui developer ($100) using adobe XD + with ESP8266 ($2) + HC05 ($2) Bluetooth module for wireless integration with board

   And the winner is Arduino Mega2560 + touchscreen laptop with tablet app only as good will from our end. See drastic cost reduction why to use complicated st board based design.

Just to note a cheaper android based laptop can easily create a stunning gui based app with iot boards even with more functionality of touchgfx at much lower cost and yet easily replicable and portable. Isn't it?

4. My suggestion to you is use cheaper bare development board for demonstrating since any one can afford to have it at low cost. I have following suggestion which are widely available and cheaper
   i. Development board STM32 ARM Cortex M4 STM32F407ZGT6 development board STM32F4 core board

      https://www.aliexpress.com/item/32827402179.html?spm=a2g0s.9042311.0.0.27424c4dpwoMIS

   ii. Parallel LCD **2.4 inch TFT LCD Shield**

      https://www.aliexpress.com/item/32895445316.html?spm=a2g0s.9042311.0.0.27424c4dZePrfQ

      For user HAL LCD interface library FMC based refer

      https://www.youtube.com/watch?v=NUErX4dx2Tw&list=PLfExI9i0v1sn_lQjCFJHrDSpvZ8F2CpkA&index=21

   iii. SPI LCD **1.8 inch TFT LCD Module LCD Screen SPI**

      https://www.aliexpress.com/item/32946727857.html?spm=a2g0s.9042311.0.0.27424c4dZePrfQ

      For user HAL LCD interface library SPI based refer

      https://www.youtube.com/watch?v=iJMhan6tvuA&feature=youtu.be

      https://www.youtube.com/watch?v=HziYB11Hu-I&feature=youtu.be

5. You should make step by step tutorial and videos of integrating Touchgfx using cubemx for following scenarios to help other
   i. STM32F4 + FMC display with No OS (Not using RTOS)
   ii. STM32F4 + FMC display with OS (using RTOS)
   iii. STM32F4 + SPI display with No OS (Not using RTOS)
   iv. STM32F4 + SPI display with OS (using RTOS)