



IoT Device Prototyping with STMicroelectronics Nucleo Development Boards

Day 5: Prototyping with the X-NUCLEO-GFX01M1 Graphic Display

Sponsored by



NNNNN -







Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click "Help" or submit a question asking for assistance.
- Participate in 'Attendee Chat' by maximizing the chat widget in your dock.



Sponsored By





Fred Eady

Visit 'Lecturer Profile' in your console for more details.



IoT Device Prototyping with STMicroelectronics Nucleo Development Boards



Sponsored By



Prototype 1: TouchGFX Remote Control Transmitter
 Prototype 2: NUCLEO-G070RB Remote Control Receiver





Sponsored By







Sponsored By





	– a ×
	Properties Interactions +
Interaction1	Interaction1 When hardware button 56 clicked X
Trigger	call virtual function
Hardware button is clicked Hardware button is clicked Screen transition begins Screen transition ends	Interaction2 When hardware button 50 clicked call virtual function
Action Call new virtual function	Interaction3 When hardware button 52 clicked call virtual function
blue_on Can trigger another interaction	Interaction4 When hardware button 54 clicked call virtual function
Interaction Name	
Interaction1	



Sponsored By





		– a ×		
		Properties Interactions		
		+		
Interaction1		Interaction1 When hardware button 56 clicked X		
Trigger		call virtual function		
Hardware button is clicked	~			
Choose button key		Interaction2 When hardware button 50 clicked call virtual function		
56 8	^			
49 1				
50 2		Interaction3 When hardware button 52 clicked		
51 3		call virtual function		
52 4				
53 5		Interaction/		
54 6		When hardware button 54 clicked		
55 7		call virtual function		
56 8				
57 9				
Interaction1				



Sponsored By





		Properties Interacti		
			+	
Interaction1		Interaction1 When hardware button 56 clicked	×	
Trigger		call virtual function		
Hardware button is clicked	~			
Choose button key		Interaction2 When hardware button 50 clicked		
56 8	~			
Action		Interaction3		
Call new virtual function	^	When hardware button 52 clicked call virtual function		
Resize widget				
Rotate Shape Scale Shape Set Language		Interaction4		
		When hardware button 54 clicked call virtual function		
Set text				
Set wildcard				
Show widget				



Sponsored By





	Properties Interaction
	+
Interaction1	Interaction1 When hardware button 56 clicked
rigger	call virtual function
Hardware button is clicked 🛛 🗸 🗸	
hoose button key	Interaction2 When hardware button 50 clicked call virtual function
56 8 ~	
ction	Interaction3
Call new virtual function	When hardware button 52 clicked call virtual function
unction Name	
blue_on	Interaction4
Can trigger another interaction	When hardware button 54 clicked call virtual function
iteraction Name	
Interaction1	



Sponsored By

Step 1: Create the TouchGFX Project Code

STM32G071_NUCLEO (in STM32CubelDE)

- > 👯 Binaries
- > 🔊 Includes
- 🗸 🗁 Application
 - 🗸 🗁 User
 - > 📂 Core
 - 🗸 🗁 generated
 - > 🗟 ApplicationFontProvider.cpp
 - > 🖳 BitmapDatabase.cpp
 - > 🗟 CachedFont.cpp
 - > R Font_verdana_10_4bpp_0.cpp
 - > 🙀 Font_verdana_15_4bpp_0.cpp
 - Font_verdana_20_4bpp_0.cpp
 - > 🙀 Font_verdana_40_4bpp_0.cpp
 - > 🙀 FontCache.cpp
 - > 🔒 FrontendApplicationBase.cpp
 - > 🔒 GeneratedFont.cpp
 - > 🙀 Kerning_verdana_10_4bpp.cpp
 - > 🛃 Kerning_verdana_15_4bpp.cpp
 - Kerning_verdana_20_4bpp.cpp
 Kerning_verdana_40_4bpp.cpp
 - Kerning_verdana_40_4bpp.cpp
 - > 🔒 LanguageGb.cpp
 - Screen1ViewBase.cpp
 - Table_verdana_10_4bpp.cpp
 - > R Table_verdana_15_4bpp.cpp
 - > R Table_verdana_20_4bpp.cpp
 - > A Table_verdana_40_4bpp.cpp
 - > 🖳 Texts.cpp
 - > 🖳 TypedTextDatabase.cpp
 - > 🗟 UnmappedDataFont.cpp
 - > 🔁 gui
 - > 📂 Startup
 - > > > TouchGFX
- 🔉 👝 Debug
- > 👝 Drivers
- ME STM32G071_NUCLEO.ioc
- STM32G071_NUCLEO Debug.launch
- STM32G071RBTX_FLASH.Id

Screen1ViewBase::Screen1ViewBase()

touchgfx::CanvasWidgetRenderer::setupBuffer(canvasBuffer, CANVAS_BUFFER_SIZE);

__background.setPosition(0, 0, 240, 320); __background.setColor(touchgfx::Color::getColorFromRGB(0, 0, 0));

shape1.setPosition(80, 120, 80, 80); shape1.setOrigin(0.000f, 0.000f); shape1.setScale(1.000f, 1.000f); shape1.setAngle(0.000f); shape1Painter.setColor(touchgfx::Color::getColorFromRGB(10, 26, 252)); shape1.setPainter(shape1Painter); const touchgfx::AbstractShape::ShapePoint<float> shape1Points[4] = { { 40.000f, 0.000f }, { 80.000f }, { 40.000f }, { 0.000f }, { 0.000f } };

const toucng+x::AbstractShape::ShapePoint<+loat> sh shape1.setShape(shape1Points);

textArea1.setXY(87, 91); textArea1.setColor(touchgfx::Color::getColorFromRGB(0, 0, 255)); textArea1.setLinespacing(0); textArea1.setTypedText(touchgfx::TypedText(T___SINGLEUSE_HCVI));

textArea2.setXY(84, 210); textArea2.setColor(touchgfx::Color::getColorFromRGB(0, 0, 255)); textArea2.setLinespacing(0); textArea2.setTypedText(touchgfx::TypedText(T___SINGLEUSE_MZ1W));

textArea3.setXY(9, 151); textArea3.setColor(touchgfx::Color::getColorFromRGB(255, 0, 0)); textArea3.setLinespacing(0); textArea3.setTypedText(touchgfx::TypedText(T___SINGLEUSE_HLDK));

textArea4.setXY(171, 151); textArea4.setColor(touchgfx::Color::getColorFromRGB(255, 0, 0)); textArea4.setLinespacing(0); textArea4.setTypedText(touchgfx::TypedText(T___SINGLEUSE_RL54));

add(__background);

add(shape1); add(textArea1); add(textArea2); add(textArea3); add(textArea4);





> 👫 Binaries > 🔊 Includes > 🕞 Application

🗸 📂 User

> > Core
> > p generated

🗸 🗁 gui

> > > Startup
> > > TouchGFX

MR STM32G071_NUCLEO.ioc

STM32G071RBTX_FLASH.Id

STM32G071_NUCLEO Debug.launch

> 👝 Debug

> 📂 Drivers

> 🔒 FrontendApplication.cpp

stm32g0xx_hal.h

Screen1View::Screen1View()
 Screen1View::blue_off() : void

Screen1View::blue_on() : void

Screen1View::red_off() : void

Screen1View::red_on() : void

Screen1View::setupScreen() : void

Screen1View::tearDownScreen() : vo

gui/screen1_screen/Screen1View.hpp

Screen1Presenter.cpp
 Screen1View.cpp

main.h

> 🔒 Model.cpp

IoT Device Prototyping with STMicroelectronics Nucleo Development Boards Prototyping with the X-NUCLEO-GFX01M1

Sponsored By



Step 1: Create the TouchGFX Project Code

#ifndef SCREEN1VIEW_HPP
#define SCREEN1VIEW_HPP
#define SCREEN1VIEW_HPP

Add This

Code

#include

<gui_generated/screen1_screen/Screen1ViewBase.hpp>
#include <gui/screen1_screen/Screen1Presenter.hpp>

class Screen1View : public Screen1ViewBase

```
{
```

public:

```
Screen1View();
virtual ~Screen1View() {}
virtual void setupScreen();
virtual void tearDownScreen();
virtual void blue_on();
virtual void blue_off();
virtual void red_on();
virtual void red_off();
protected:
};
```





Sponsored By



STM32G071_NUCLEO (in STM32CubeIDE) Stmartes	<pre>#include <gui screen1_screen="" screen1view.hpp=""> #include "main.h" #include "stm32g0xx_hal.h" extern UART_HandleTypeDef huart3;</gui></pre>	Кеу	Code
 Application Application 	Screen1View::Screen1View() {	Left	'4'
 Core generated gui FrontendApplication.cpp Model.cpp Screen1Presenter.cpp gui/screen1.screen/Screen1View.hpp main.h stm32g0xx_hal.h Screen1View::blue_off() : void Screen1View::blue_off() : void Screen1View::blue_on() : void Screen1View::ed_off() : void Screen1View::etarDownScreen() : void Statup Debug Drivers STM326071.NUCLEO.ioc STM326071.NUCLEO.ioc STM326071RBTX_FLASH.ld 	<pre>} void Screen1View::setupScreen() { Screen1ViewBase::setupScreen(); } void Screen1View::tearDownScreen() { Screen1ViewBase::tearDownScreen(); } void Screen1View::blue_on() { uint8_t txBite = 56; HAL_UART_Transmit(&huart3,&txBite,1,0xFFFF); } void Screen1View::blue_off()</pre>	Right	'6'
		Up	'8'
		Down	'2'
		Center	'5'
		Blue User Button	'0'
	<pre>{ uint8_t txBite = 50; HAL_UART_Transmit(&huart3,&txBite,1,0xFFFF); }</pre>		
	<pre>void Screen1View::red_on() {</pre>	BLUE ON	
	<pre>uint8_t txBite = 52; HAL_UART_Transmit(&huart3,&txBite,1,0xFFFF); }</pre>	RED ON	RED OFF
	<pre>void Screen1View::red_off() { uint8_t txBite = 54; HAL_UART_Transmit(&huart3,&txBite,1,0xFFFF); }</pre>	BLUE OFF	



Sponsored By







Sponsored By



Step 3: Design and Construct the XBee Transmitter







Sponsored By



Step 4: Test the XBee Transmitter





Sponsored By



Step 5: Configure the XBee Receiver Hardware





Sponsored By



Step 6: Design and Construct the XBee Receiver







17



Sponsored By

Step 7: Code the XBee Receiver Application



> 🗁 Inc

- 🗸 🗁 Src
 - > 💼 gpio.c
 - > 💼 main.c
 - > 1 stm32g0xx_hal_msp.c
 - > 💼 stm32g0xx_it.c
 - > 底 syscalls.c
 - > 💽 sysmem.c
 - > ic system_stm32g0xx.c
 - > 🖻 usart.c
- > 📂 Startup
- > 😕 Drivers
- > 📂 Debug
- > 📂 Release
 - mx remote_app_receiver.ioc
 - remote_app_receiver Debug.launch
- STM32G070RBTX_FLASH.Id
- STM32G071_NUCLEO (in STM32CubelDE)

extern UART_HandleTypeDef huart3; #define USART3_RX_BUFFER_SIZE128 #define USART3_RX_BUFFER_MASK (USART3_RX_BUFFER_SIZE - 1) extern uint8_t USART3_RxHead; extern uint8_t USART3_RxTail; extern uint8_t USART3_RxBuffer[USART3_RX_BUFFER_SIZE]; extern uint8_t USART3_RxBuffer[USART3_RX_BUFFER_SIZE];





Sponsored By

Step 7: Code the XBee Receiver Application

void USART3_4_IRQHandler(void)

```
✓ IDE remote_app_receiver
                                           uint32_t cr1its
   🐰 🐰 Binaries
                                           uint32 t cr3its
  > 🔊 Includes
  V 🏳 Core
                                           uint32 t errorflags;
                                           uint16 t data;
    > 🗁 Inc
                                           uint8_t tmphead;
    🗸 🗁 Src
       > 💼 gpio.c
       > 💼 main.c
                                           /* If no error occurs */
         stm32g0xx_hal_msp.c
                                           if (errorflags == 0U)
         stm32g0xx_it.c
         c syscalls.c
       > c sysmem.c
       > c system_stm32g0xx.c
       > .c usart.c
    > 👝 Startup
  > Privers
  > 👝 Debug
  > 📂 Release
                                               // store new index
    MX remote_app_receiver.ioc
       remote_app_receiver Debug.launch
    STM32G070RBTX_FLASH.Id
 STM32G071_NUCLEO (in STM32CubeIDE)
                                           else
```

/* USER CODE BEGIN USART3 4 IRQn 0 */ uint32 t isrflags = READ REG(huart3.Instance->ISR); = READ_REG(huart3.Instance->CR1); = READ_REG(huart3.Instance->CR3);

uint16 t dataMask = 0x00FF;

errorflags = (isrflags & (uint32 t)(USART ISR_PE | USART ISR_FE | USART ISR_ORE | USART ISR_NE | USART ISR_RTOF));

```
// UART in mode Receiver ------
//if(((isrflags & USART ISR RXNE) != 0U) && ((cr1its & USART CR1 RXNEIE) != 0U))
if (((isrflags & USART_ISR_RXNE_RXFNE) != 0U)
      && (((cr1its & USART_CR1_RXNEIE_RXFNEIE) != 0U)
          || ((cr3its & USART CR3 RXFTIE) != 0U)))
```

```
// read byte from UART
data = (uint16_t)READ_REG(USART3->RDR);
```

// calculate buffer index tmphead = (USART3 RxHead + 1) & USART3 RX BUFFER MASK; USART3_RxHead = tmphead;

```
if ( tmphead == USART3 RxTail )
```

// ERROR! Receive buffer overflow

```
// store received data in ring buffer
USART3_RxBuf[tmphead] = (uint8_t)(dataMask & data);
```

```
HAL UART CLEAR PEFLAG(&huart3);
____HAL_UART_CLEAR_FEFLAG(&huart3);
___HAL_UART_CLEAR_OREFLAG(&huart3);
```





Sponsored By

Step 7: Code the XBee Receiver Application

int main(void)

HAL Init(); ✓ IDE remote_app_receiver > 👯 Binaries MX GPIO Init(); > 🔊 Includes V 🏳 Core > 🗁 Inc 🗸 🗁 Src > c gpio.c c main.c stm32g0xx_hal_msp.c > c stm32g0xx_it.c > c syscalls.c while (1) > c sysmem.c > c system_stm32g0xx.c > .c usart.c > 👝 Startup > 2 Drivers > 📂 Debug > 📂 Release MX remote_app_receiver.ioc remote_app_receiver Debug.launch break; STM32G070RBTX_FLASH.Id STM32G071_NUCLEO (in STM32CubeIDE)

```
SystemClock Config();
MX USART2 UART Init();
MX USART3 UART Init();
/* USER CODE BEGIN 2 */
HAL GPIO WritePin(XB RESET GPIO Port, XB RESET Pin, GPIO PIN SET);
USART3 RxHead = 0 \times 00;
USART3 RxTail = 0x00;
__HAL_UART_ENABLE_IT(&huart3,UART_IT_RXNE);
/* USER CODE END 2 */
/* USER CODE BEGIN WHILE */
  if(CharInRing())
    HAL Delay(100);
    rxBuf[0] = readring();
    switch(rxBuf[0])
      case 56: //up blue on
        HAL GPIO WritePin(LED BLUE GPIO Port, LED BLUE Pin, GPIO PIN SET);
      case 50: //down blue off
        HAL GPIO WritePin(LED BLUE GPIO Port, LED BLUE Pin, GPIO PIN RESET);
      break;
      case 52: //left red on
        HAL GPIO WritePin(LED RED GPIO Port, LED RED Pin, GPIO PIN SET);
      break;
      case 54: //right red off
        HAL GPIO WritePin(LED RED GPIO Port, LED RED Pin, GPIO PIN RESET);
      break;
```





Sponsored By



Step 8: Fire It All Up!



21





LET'S EAT!!



Thank you for attending!!!

Please consider the resources below:

- Today's Project Download Package
- STM32MP1 TouchGFX
- NUCLEO-G070RB User Manual

To get today's Project Download Package please send an email request to: therealfredeady@gmail.com





Thank You





Same

