

# Renesas RA Family

# **Getting Started with GUIX Thermostat Application**

# Introduction

This application, which is a Thermostat application, provides a reference for developing complex multithreaded applications with a touch screen graphical Human Machine Interface (HMI) by using Renesas FSP and Azure RTOS GUIX. It describes steps to create a basic GUIX for FSP, integrates touch driver, handles multiple hardware accesses, system updates, and event handling.

This application is developed using the Renesas RA Flexible Software Package (FSP), which provides a quick and versatile way to build secure connected Internet of Things (IoT) devices using the Renesas RA family of Arm microcontrollers (MCUs). RA FSP provides production ready peripheral drivers to take advantage of the RA FSP ecosystem along with Azure RTOS GUIX library and Azure RTOS. In addition, FSP also provides Ethernet, USB, File System and other middleware stacks as well. This powerful suite of tools provides a comprehensive, integrated framework for rapid development of complex embedded applications.

This application note assumes that you are familiar with the concepts associated with writing multi-threaded applications under a Real Time Operating System (RTOS) environment, such as Azure RTOS. This application note makes use of RTOS features such as threads and semaphores. Prior experience in using Azure RTOS would be helpful for easy understanding of the provided application project. For more detailed information on Azure RTOS features, refer to the Azure RTOS User Manual.

The Graphics application is developed using the Renesas e<sup>2</sup> studio Integrated Solution Development Environment (IDE). e<sup>2</sup> studio is integrate with the FSP platform installer, which can be downloaded from Renesas website. The intuitive configurators and code generators in e<sup>2</sup> studio and FSP will help the application developers in creating such complex multi-threaded graphics applications very quickly. This application note walks you through all the necessary steps in creating, building and running a complex graphics project, including the following:

- Board setup.
- Install tools.
- Build and run application.
- Azure RTOS GUIX Studio project integration.
- Setup Azure RTOS GUIX Studio project.
- Add Touch Driver.
- Create FSP GUIX project.
- Hardware Setup.
- Using the General Purpose Timer to drive a PWM backlight control signal.

# **Required Resources**

#### **Development tools and software**

- e<sup>2</sup> studio IDE 2025-04.1
- Renesas Flexible Software Package (FSP) v6.0.0
- Azure RTOS GUIX Studio V6.4.0.0

#### Hardware

• <u>Renesas EK-RA6M3G</u> kit (RA6M3 MCU Group)

# **Reference Manuals**

- RA Flexible Software Package Documentation Release v6.0.0
- Azure RTOS GUIX and GUIX Studio v6.4.0.0
- Renesas RA6M3 Group User's Manual Rev.1.20
- EK-RA6M3G-v1.0 Schematics



# Contents

1.	Installing Tools	3
1.1	Overview	3
1.2	Procedural Steps	3
2.	Create Application Note Project	5
2.1	Overview	5
2.2	Procedural Steps	6
3.	Using GUIX Widget Timer to Trigger a Screen Transition	21
3.1	Overview	21
3.2	Procedural Steps	21
4.	Add Touch Driver to Thermostat_GUIX_EK_RA6M3G Project	23
4.1	Overview	23
4.2	Procedural Steps	23
5.	Control LCD Backlight	31
5. 5.1	Control LCD Backlight	
	-	
5.1	Overview	31 31
5.1 5.2	Overview Procedural Steps	31 31 35
5.1 5.2 6.	Overview Procedural Steps Update Date/Time and Temperature	31 31 35 35
5.1 5.2 6. 6.1	Overview Procedural Steps Update Date/Time and Temperature Overview	
<ol> <li>5.1</li> <li>5.2</li> <li>6.</li> <li>6.1</li> <li>6.2</li> </ol>	Overview Procedural Steps Update Date/Time and Temperature Overview Procedural Steps	
<ol> <li>5.1</li> <li>5.2</li> <li>6.</li> <li>6.1</li> <li>6.2</li> <li>7.</li> </ol>	Overview Procedural Steps Update Date/Time and Temperature Overview Procedural Steps Setting Date/Time in A Full Function Project	
<ol> <li>5.1</li> <li>5.2</li> <li>6.</li> <li>6.1</li> <li>6.2</li> <li>7.</li> <li>7.1</li> </ol>	Overview Procedural Steps Update Date/Time and Temperature Overview Procedural Steps Setting Date/Time in A Full Function Project Overview	



# 1. Installing Tools

# 1.1 Overview

In this section you will copy the application note (AN) materials to your PC and install  $e^2$  studio v2025-04.1/FSP v6.0.0 and Azure RTOS GUIX Studio v6.4.0.0.

# 1.2 Procedural Steps

- 1. If you already have e<sup>2</sup> studio v2025-04.1 with FSP v6.0.0, you can skip this step. Otherwise, you can download <u>RA Flexible Software Package (FSP)</u>.
- 2. You can get <u>Azure RTOS GUIX Studio v6.4.0.0</u> or greater. If it goes well, you will see the window in the next step on the web browser.

Note: It needs Microsoft Store to work on your PC to install Azure RTOS GUIX Studio.

3. Click Get to start installing Azure RTOS GUIX Studio.



Figure 1. Clicking Get to Start Installing Azure RTOS GUIX

4. Click Open Microsoft Store to continue installing Azure RTOS GUIX Studio.

Open Microsoft Store?	
https://www.microsoft.com wants to open this application.	
Open Microsoft Store Cancel	

#### Figure 2. Clicking Open Microsoft Store

5. Click **Install** to continue. A window shows up to ask for a Microsoft account, which is seen in the next step.







6. Ignore it by clicking "X" on the top-right to close this pop-up window and continue Azure RTOS GUIX Studio installation.



#### Figure 4. Closing Pop-up Window to Continue Installing Azure RTOS GUIX

7. Downloading and installation of Azure RTOS GUIX Studio starts.



Figure 5. Starting of Downloading and Installation

8. Click Launch to launch Azure RTOS GUIX Studio.

Wish list

Figure 6. Launching to Start Azure RTOS GUIX Studio



# 9. Azure RTOS GUIX Studio launched.

G Azure RTOS	GUIX Studio	6.4.0.0								-		$\times$
Project Edit Ins												
-		¥	目	Û	-	10	<u>In</u>	훕	88×	-	∎₫	<b>-</b>
🔒 Project \	/iew	-										
喧 No Project												
Propertie	es View	-										

#### Figure 7. Azure RTOS GUIX Studio Launched

10. Close Azure RTOS GUIX Studio, for now, you will open it again later.

# 2. Create Application Note Project

# 2.1 Overview

In this section, you will create a project to which you will add pre-written source code and integrate it with a pre-created Azure RTOS GUIX Studio project.



# 2.2 Procedural Steps

1. Create a new **Renesas RA C/C++ project**. Name it as Thermostat\_GUIX\_EK\_RA6M3G.

💽 New C/C++ Proj				— [	_	×
Templates for Re	nesas RA Project					
	Create an e. for Renesas Renesas Create an F.	RA.	ic library C/C+ on enesas RA com	prising a	or c	
Renesas RA C/C++ Proj Renesas RA C/C++ Pro Project Name and Locatic	ject	Next >	Finish		Cancel	×
Project name Thermostat_GUIX_EK_R Use default location Location:	A6M3G	Thermos	at_GUIX_EK_RA6M:	3G	Browse.	
You can download more	Renesas packs here					
?		< Back	lext > F	Finish	Cancel	

Figure 8. Creating New RA C/C++ Project



#### 2. Set board to EK-RA6M3G.

Renesas RA	A C/C++ Project C/C++ Project ools Selection	
Device Selecti FSP Version: Board: Device: Core: Language:		Evaluation kit with Graphics for RA6M3 MCU Group Visit <u>https://www.renesas.com/ra/ek-ra6m3g</u> to get kit user's manual, quick
IDE Project Ty e <sup>2</sup> studio ma Toolchains LLVM Embec GNU ARM E 13.2.1.arm-1	naged build ~	
?		< Back Next > Finish Cancel

Figure 9. Setting Board to EK-RA6M3G

3. Select Preceding project or Smart Bundle.

Renesas RA C/C++						
Preceding Project or S	mart Bundle Selection					
None	Choose this option when creatir	ng a project for the pri	imary processor core (no	preceding project c	r Smart Bundle).	
O Preceding Project:						
	Choose this option if you have a for Secure Software Debug (SSE the target device. Should you w the Device Lifecycle state as nee	D) state will be generat ish to test Non-Secure	ted where applicable and	both secure and nc	n-secure images will b	e downloaded to
O Smart Bundle:						
	Resolved location:					
				Workspace	File System	Variables
	Choose this option if you only h	ano accors to a pro-p			1	
	Choose this option if you only h should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)
Preceding Project/Sma	should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)
Preceding Project/Sma	should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)
FSP version Toolchain	should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)
FSP version Toolchain Toolchain version	should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)
FSP version Toolchain	should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)
FSP version Toolchain Toolchain version Board	should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)
FSP version Toolchain Toolchain version Board Device	should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)
FSP version Toolchain Toolchain version Board Device Core	should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)
FSP version Toolchain Toolchain version Board Device Core	should be obtained from the Tru be generated and only the non-	ustZone Secure Projec	t developer. A debug co	aining TrustZone sec nfiguration for Non-	ure code. The Smart Bu	undle file (*.sbd)

Figure 10. Preceding project or Smart Bundle Selection



#### 4. Select Azure RTOS ThreadX (v6.4.0+fsp.6.0.0).

Renesas RA C/C++ Project	
Build Artifact and RTOS Selection	
Build Artifact Selection	RTOS Selection
Executable	Azure RTOS ThreadX (v6.4.0+fsp.6.0.0)
<ul> <li>Project builds to an executable file</li> </ul>	
<ul> <li>Static Library</li> <li>Project builds to a static library file</li> </ul>	
<ul> <li>Executable Using an RA Static Library</li> <li>Project builds to an executable file</li> </ul>	
<ul> <li>Project uses an existing RA static library project</li> </ul>	
2	

Figure 11. Selecting Azure RTOS ThreadX

5. Use Azure RTOS ThreadX-Minimal template.
Renesas RA C/C++ Project
Renesas RA C/C++ Project
Project Template Selection
Project Template Selection
Azure RTOS ThreadX - Blinky Azure RTOS ThreadX project that includes BSP and will blink LEDs if available. This project will initialize the MCU using the BSP. ThreadX will also be initialized and a single thread to blink the LEDs will be started. [Renesas.RA.6.0.0.pack]
Azure RTOS ThreadX - Minimal     Empty ThreadX FSP project with no threads. This project will initialize the MCU using the BSP.     [Renesas.RA.6.0.0.pack]
Code Generation Settings
Use Renesas Code Formatter
Back     Next >     Finish     Cancel

Figure 12. Selecting Azure RTOS ThreadX – Minimal Template



#### 6. Open project configuration, go to **BSP** tab, change Heap size to 0x2000.

File Edit Source Refactor Navigate Search Project Ren	iesas Views Run	Renesas AI	Window Help			
🔚 🐯 🕶 🔦 🎋 🕶 💁 🕶 🚱 🗉						
Project Explorer ×	🔅 [Thermosta	GUIX_EK_RA6	M3G] FSP Configur	ation $ imes$		
✓	Board Sup	port Packa	age Configura	ntion		
> 🔊 includes > 😂 ra						
> 😂 ra_gen > 😂 src	Device Selec	tion				
<ul> <li>&gt; ⇒ Debug</li> <li>&gt; ⇒ script</li> <li>☆ configuration.xml</li> <li>〕 JLinkLog.log</li> <li>ⓐ ra_cfg.txt</li> <li>※ Thermostat_GUIX_EK_RA6M3G Debug_Flat.launch</li> <li>? Developer Assistance</li> </ul>	FSP version Board: Device: Core: RTOS: Summary BSP	EK-RA6M30 R7FA6M3AH CM4 Azure RTOS	H3CFC ThreadX	w     m.s     Linker Sections	Visit https://www.	a Graphics for RA6M3 MCU Group renesas.com/ra/ek-ra6m3g to get xample projects, etc.
	Problems	Console	Properties ×	Smart Browser	Smart Manual	Progress Search
	EK-RA6M30	6				
	Settings	operty RA Common			Value	
		Main stack	size (bytes)		0x400	
		Heap size (			0x2000	
			r Secondary XIP		Disabled	
		MCU Vcc (r	mV)		3300	

Figure 13. Changing Heap size (bytes) in Project Configuration

# 7. You can remove **Azure RTOS ThreadX Port** if you want.

🔯 *[Thermostat_GUIX_EK_RA6M3G] FSP Configuration $ imes$		
Stacks Configuration		Generate Project Content
Threads     Image: New Thread     Image: Remove       Image: Weight of the second sec	Azure RTOS ThreadX Port (rm_threadx_port) Stacks	Stack > ≜ Extend Stack > ■ Remove
<ul> <li>g ioport I/O Port (r ioport)</li> <li>Azure RTOS ThreadX Port (rm_threadx_port)</li> </ul>	# Azure RTOS ThreadX Port (rm_threadx_port)	
	<b>i</b>	
Objects New Object > 🕢 Remove		
ummary BSP Clocks Pins Interrupts Event Links Stacks	Components	

#### Figure 14. Remove Azure RTOS ThreadX Port

8. Add a **New Thread** and name it as **System Thread** with the following settings.



Stacks	Configuration				
Threads	🕢 New Thread 🙀 Remove	System	n Thread Stacks		🗿 New Stack 🗄
	AL/Common g_ioport I/O Port Driver on r_ioport /stem Thread	0	Add stacks to the selected t clipboard.	hread by using the 'New Stack >' t	oolbar button (above),
Objects	🛃 New Object > 📓 Rem	nove			
Summary	BSP Clocks Pins Interrupts Event Links Stack	ks Components			
🔜 Pin Cor	aflicts Properties 🕅	_ ·			
	flicts Properties 🛛		I		
Pin Cor System			I		
System			Value		
	Thread				
System	<b>Fhread</b> Property				
System	Fhread Property ✔ Common	•			
System	Thread Property ✓ Common → General	•			
System	Thread Property Common General Timer	•			
System	Thread Property Common General Timer Trace				
System	Thread Property Common Seneral Timer Trace Performance				
System	Thread Property Common General Timer Trace Performance RA				
System	Thread Property Common General Timer Trace Performance RA Interrupts				
System	Thread  Property  ✓ Common  > General  > Timer  > Trace  > Performance  > RA  > Interrupts  ✓ Thread		Value		
System	Thread  Property  ✓ Common  > General  > Timer  > Trace  > Performance  > RA  > Interrupts  ✓ Thread Symbol		Value system_thread		
System '	Thread Property Common General Timer Trace RA Interrupts Thread Symbol Name		Value system_thread System Thread		
System	Thread Property Common General Timer Trace Proformance RA Interrupts Thread Symbol Name Stack size (bytes)		Value system_thread System Thread 1024		

Figure 15. Adding a New Thread and Naming it System Thread

# 9. Add Azure RTOS GUIX to System Thread.

🐡 *[Thermostat_GUIX_EK_RA6M3G] FS	P Configurat	ion $ imes$				- 0	$^{5}$ FSP Visualization $ imes$ $\square$
Stacks Configuration				Generate P	O roject	Content	<b>i</b>
Threads New Thread Remove g_ioport I/O Port (r_iop Azure RIOS ThreadX Pc System Thread C	System 1	<b>Thread Stacks</b> Add stacks to the selected thread by using the from the clipboard.	New Stack	AI Analog Audio Bootloader CapTouch Connectivity DSP	>	iemove here	RENESAS
Objects       New Object >         @ Remove       Remove         Summary       BSP       Clocks       Pins       Interrupts         Browser       Console       Search       Image: Ima				Graphics Input Monitoring Motor Networking Power Security	> > > > >	<ul> <li>D/AV</li> <li>D/AV</li> <li>Graph</li> <li>JPEG</li> </ul>	RTOS GUIX E 2D (r_drw) E 2D Port Interface (r_drw) nics LCD (r_glcdc) Codec (r_jpeg) el Data Capture (r_pdc)

Figure 16. Adding Azure RTOS GUIX to System Thread



10. Property settings for the g\_display0 Graphics LCD on r\_glcdc.

Problem	is Co	nsole Properties × Smart Browse	er Smart Manual	Progress	Search		
g_displa	y0 Grapł	nics LCD (r_glcdc)					
Settings	Property			Value			
-	> Ge						
API Info	> Int	errupts					
	💙 Inp	put					
		Graphics Layer 1					
		✓ General					
		Enabled		Yes			
		Horizontal size		480			
		Vertical size Horizontal position		272			
		Vertical position		0			
		Color format		RGB565 (16-b	(tit)		
		Line descending mode		Disabled			
		✓ Framebuffer					
		Framebuffer name		fb_backgroun	d		
		Number of framebuffers		2			
		Section for framebuffer allocation		.bss			
		> Line Repeat > Fading					
	_display	<b>0 Graphics LCD (r_glcdc)</b> Property				Value	
	-	> General					
A	PI Info	> Interrupts					
		> Input					
		✓ Output					
		✓ Timing					_
		Horizontal total cycles				525	
		Horizontal active video cy	/cles			480	
		Horizontal back porch cy	cles			40	
		Horizontal sync signal cyc	cles			1	
		Horizontal sync signal po				Low activ	/e
		Vertical total lines				316	1
		Vertical active video lines				272	
		Vertical back porch lines				8	
		Vertical sync signal lines				1	
		Vertical sync signal polari	ty			Low activ	/e
		Data Enable Signal Polarit				High acti	ve
		Sync edge				Falling ed	
		✓ Format					<u> </u>
		Color format				16bits R	GB565
		Color order				RGB	
		Endian				Little end	lian
		Endidit				and chie	

Figure 17. Graphics LCD Display Driver Configuration

11. Configure the Hsync, Vsync, Data Enable pins and clock as shown.

_aispia	/0 Graphics LCD (r_glcdc)	
Settings	Property VTCON	Value
API Info	Hsync pin select	LCD_TCON0
	Vsync pin select	LCD_TCON1
	Data enable (DE) pin select	LCD_TCON2
	Panel clock source	Internal clock (GLCDCLK)
	Panel clock division ratio	1/24
	> Color Correction	
	> Dithering	

# Figure 18. TCON Properties

12. In Pin Configuration, change P603's mode to Output mode (Initial high) to enable LCD panel backlight.



RA6M3G-EK.pincfg	<b>∨</b> <u>Ma</u>	nage configurations	Generate data: g_bsp_pin_cfg				
Pin Selection	∃ ⊡ ↓ª₂	Pin Configuration					
Type filter text		Name	Value	Link			
✓ ✓ P6	•	Symbolic Name					
✓ P600		Comment					
✓ P601		Mode	Output mode (Initial High)				
✓ P602		Pull up	None				
✓ P603		Drive Capacity	Low				
P604		Output type	CMOS				
P605		✓ Input/Output					
P606		P603	V GPIO	$\Rightarrow$			
P607							
✓ P608							
✓ P609							
✓ P610		Module name: P603					
✓ P611	~	Port Capabilities: BUS0: D13 GPT7: GTI					

Figure 19. Changing Pin Configuration Mode to Enable LCD Panel Backlight

# O

13. In RA Configurator, click Generate Project Content to generate project content. Make sure project is active,

click to build the project. It may take a long period of time to finish building an Azure RTOS/GUIX project on your PC.



14. Copy Azure RTOS GUIX Studio project to e<sup>2</sup> studio project (Themostat\_GUIX\_EK\_RA6M3G) by copying "guix\_studio" folder in the application note (AN) folder (FSP\_GUIX\_Thermostat) and pasting it in the Themostat\_GUIX\_EK\_RA6M3G project.

Name	Date modified	Туре	:
2.23	7/1/2021 11:10 AM	File folder	
2.28	7/1/2021 11:10 AM	File folder	
3.1	7/1/2021 11:10 AM	File folder	
4.11	7/1/2021 11:10 AM	File folder	
5.6	7/1/2021 11:10 AM	File folder	
6.6	7/1/2021 11:10 AM	File folder	
completed_project	7/1/2021 11:11 AM	File folder	
guix_studio	7/1/2021 11:10 AM	File folder	
touch_ft5x06	7/1/2021 11:10 AM	File folder	
ile Edit Navigate Search Project Re	New		>
🗐 📎 <b>- 🌾 -</b> 🞺 🗠 🏪 🌼 :	Go Into		
	Open in New Window	N	
	Open in New Window Show In		Alt+Shift+W >
			Alt+Shift+W > Ctrl+C
	Show In		
Thermostat_GUIX_EK_RA6M3G	Show In Copy		Ctrl+C
Froject Explorer ×         Image: State of the state	Show In Copy Paste		Ctrl+C Ctrl+V
<ul> <li>Project Explorer ×</li> <li>S Thermostat_GUIX_EK_RA6M3G</li> <li>S Binaries</li> <li>Binaries</li> <li>Includes</li> </ul>	Show In Copy Paste Delete		Ctrl+C Ctrl+V Delete
Froject Explorer ×         Image: State of the state	Show In Copy Paste Delete Source		Ctrl+C Ctrl+V Delete
<ul> <li>Project Explorer ×</li> <li>Thermostat_GUIX_EK_RA6M3G</li> <li>S Includes</li> <li>S ra</li> <li>S ra</li> <li>S ra_gen</li> <li>S rc</li> <li>Debug</li> </ul>	Show In Copy Paste Delete Source Move Rename		Ctrl+C Ctrl+V Delete >
<ul> <li>Project Explorer ×</li> <li>Project Explorer ×<td>Show In Copy Paste Delete Source Move Rename Import</td><td></td><td>Ctrl+C Ctrl+V Delete &gt;</td></li></ul>	Show In Copy Paste Delete Source Move Rename Import		Ctrl+C Ctrl+V Delete >
<ul> <li>Project Explorer ×</li> <li>Thermostat_GUIX_EK_RA6M3G</li> <li>S Includes</li> <li>Includes</li> <li>Includes<!--</td--><td>Show In Copy Paste Delete Source Move Rename Import</td><td></td><td>Ctrl+C Ctrl+V Delete &gt; F2</td></li></ul>	Show In Copy Paste Delete Source Move Rename Import		Ctrl+C Ctrl+V Delete > F2
<ul> <li>Project Explorer ×</li> <li>Thermostat_GUIX_EK_RA6M3G</li> <li>S Includes</li> <li>S ra_gen</li> <li>S rc</li> <li>Pebug</li> <li>Pobuld</li> <li>ra_cfg</li> <li>script</li> </ul>	Show In Copy Paste Delete Source Move Rename Import Export		Ctrl+C Ctrl+V Delete > F2
<ul> <li>Project Explorer ×</li> <li>Project Explorer ×<td>Show In Copy Paste Delete Source Move Rename Import Export Renesas FSP Export</td><td></td><td>Ctrl+C Ctrl+V Delete &gt;</td></li></ul>	Show In Copy Paste Delete Source Move Rename Import Export Renesas FSP Export		Ctrl+C Ctrl+V Delete >
<ul> <li>Froject Explorer ×</li> <li>Froject Explorer ×</li> <li>From Stat_GUIX_EK_RA6M3G</li> <li>From Sinces</li> <li>From Sinces<td>Show In Copy Paste Delete Source Move Rename Import Export Renesas FSP Export Build Project</td><td></td><td>Ctrl+C Ctrl+V Delete &gt; F2</td></li></ul>	Show In Copy Paste Delete Source Move Rename Import Export Renesas FSP Export Build Project		Ctrl+C Ctrl+V Delete > F2

Figure 20. Copying the Azure RTOS GUIX Studio Project to e<sup>2</sup> studio



15. GUIX Studio project is now in Thermostat\_GUIX\_EK\_RA6M3G project. In e<sup>2</sup> studio, right-click the "guix\_studio" folder and exclude it from the build since it contains the Azure GUIX Studio project, which will not be built by FSP.

陷 Project Explorer 🗦		New Go Into	>	] FSP Co	nfiguration $ imes$
∨ 📂 Thermostat_GI		Show In	Alt+Shift+W >		
> 🐝 Binaries		Сору	Ctrl+C		
> 🗊 Includes	Ē	Paste	Ctrl+V		
> 😕 ra	×	Delete	Delete		✓ Manage configura
> 🐸 ra_gen		Move			
> 😕 src		Rename	F2	∃↓ªz	Pin Configuration
> 🗁 Debug	പ്പ	Import			Name
> 🗁 build	4	Export			Symbolic Name
> 🗁 guix_studio		Build Project	Ctrl+B	^	Comment
> 🗁 ra_cfg	ন্থ্য	Refresh	F5		Mode
> 🧀 script 💮 configuration	_	Source	>		Dull up
i ra_cfg.txt					
Thermostat		Build Targets	> 		Module name: P603
> ⑦ Developer A		Resource Configurations	>		ude from Build
<		Run As	>		et to Default ent Links   Stacks   Compor
	1	Debug As	> ×		
		Exclude object(s) from build in th			
		Debug	The following configurations		
		✓ Release			
		Sel	ect All Deselect All		

Figure 21. Excluding guix\_studio Folder from the Build



16. Get to Thermostat\_GUIX\_EK\_RA6M3G project folder by right clicking the e<sup>2</sup> studio project and select "System Explorer" as shown below.

**	Debug As	>
	Team	>
	Compare With	>
	Restore from Local History	
	MISRA-C	>
1	C/C++ Project Settings	Ctrl+Alt+P
	Save build settings report	
	Change Device	
*	Run C/C++ Code Analysis	
1	System Explorer	
65.	Command Prompt	
	Validate	
	Configure	>
	Source	>
	Properties	Alt+Enter

Figure 22. Selecting System Explorer

17. Open **thermostat.gpx** project file in "**guix\_studio>GNU**" sub-folder in your Thermostat\_GUIX\_EK\_RA6M3G folder. If you have several GUIX Studio versions in your system, make sure you choose the right one, which is **v6.4.0.0 or later**.



Figure 23. Opening the Project File

18. This GUIX Studio project has a complete design of this Thermostat application. The next several steps describe the process to generate resources, application code and integrate them with an e<sup>2</sup> studio project.

G Azure RTOS GUIX Stu	dio 6.4.(	.0 - th	ermostat.gxp	)							_		$\times$
Project Edit Insert Con	figure	Help											
🖬 🖻 🖻 🗶 🖻	Ô 🖹	-	편 쇼 출	H-	D C	1 G (s	ÈΘ	⊕ ∎	I ► - <sup>x</sup>	' (?)			
🔒 Project View		-									📮 Theme:	"theme	_1"‡
🗅 thermostat											🕞 Colors		+
i≘⊡ display i=⊡ default_folder											T Fonts		+
⊞ - 📑 Help ⊞ - 🚍 MainPage				3.8 °C		20, 2025 mostat			0:00 AM	Ĺ	Pixelma	ps	+
■ 🔁 Settings ■ 🔁 Splash		1	AUTO	When the air condition	fan is set to AUT oning unit tells it	O it will only turn to.	on when eithe	er the heater or	the		♦ Strings		+
🗄 🕂 🗂 Thermostat			ON System:	whether or	not the heater a	t will continue to o and air conditionin	ng units are or	n.					
III Properties View	N	-	OFF	If the syste to ON then	m is OFF then al n it will not shut	I heating and coo off when the syste	ling units are im is off.	off. If your fan i	s set				
Widget Type	window	^	COOL	When the needed to	system is set to ( meet the "Set to	COOL the air cond of temperature.	litioning unit 1	nill turn on whe	en.				
Widget Name	Help		HEAT	When the meet the "	system is set to i Set to" temperat	HEAT the heating ure.	unit will turn o	on when needer	d to				
Widget Id	ID_HEL									┢			
User Data													
Left	0												
Тор	0												
Width	480	$\checkmark$											

Figure 24. GUIX Studio Thermostat Application View



19. The Azure RTOS GUIX Studio project consists of 5 screens, including Splash, Main Page, Settings, Thermostat and Help from top to bottom:

Rene	Ver : 1.0.0
	•
March 20, 2025 Thursday 23.8°C March 20, 1	09:30:00 AM
Indeor: 23.8°C Set To: 33.0°C Fan: System: AUTO OFF	O Settings
23.8°C March 20, 3 Year Month 2021 MAY Variable MAY	Day Display 11 Set Date
23.8°C March 20, 2 Thermostat Indoor: <b>23.</b> Fan: Syst	8°C 5€t To: 5,33, 0°€
AUTO C 23.8°C March 20,3 Fan:	FF Help
System:	
to ON then it will not shut off whe	ne air conditioning unit will turn on when
HEAT When the system is set to HEAT th meet the "Set to" temperature.	e heating unit will turn on when needed to

Figure 25. Azure RTOS GUIX Studio Project Screens



20.	Click "Configure→Project/Display"	and confirm the following settings.
-----	-----------------------------------	-------------------------------------

🗘 Configure Project	×	
Directories Source Files//src/guix_gen	browse	
Header Files//src/guix_gen	browse	
Target CPU Renesas RA V Advanced Settings	browse	
Toolchain GNU V big endian		
Additional Headers	nsert Before	
Number of Displays 1 GUIX Library Version 6 . 4	- 0 - nor Patch	
Display Configuration		
Display Number 1 Name display		
x resolution 480 pixels y resolution 272 pixels		
1 bpp grayscale 1:5:5:5 form	at	
2 bpp     invert polarity     44:4:4 form     4 bpp	at	
8 bpp     reverse byte order     3:3:2 format		
16 bpp     packed format		
○ 24 bpp     □ allocate canvas memory     Rotation:     None       ○ 32 bpp     □     □     □     □	~	
Number of Palette Mode Anti-aliased Text Colors: 8 ~		
Cancel	Save	

Figure 26. Configure Project Settings

21. Go back e<sup>2</sup> studio project (Thermostat\_GUIX\_EK\_RA6M3G), right click "**src**", then select "**New**→**Folder**" and create a folder named "**guix\_gen**".

				🕲 New Folder	— 🗆 X
				Folder Create a new folder resource.	
				Enter or select the parent folder:	
				Thermostat_GUIX_EK_RA6M3G/src	
File Edit Navig	gate Search Project R New Go Into	Renesas Views Run Renesas Al V	Project	<ul> <li>☆ ☆</li> <li>☆ Thermostat_GUIX_EK_RA6M3G [Debug]</li> <li>☆ settings</li> <li>&gt; ☆ Debug</li> <li>&gt; ☆ guix_studio</li> <li>&gt; ☆ ra</li> <li>&gt; ▷ a_cfg</li> <li>&gt; cript</li> <li>&gt; src</li> </ul>	
- Floject L	Open in New Window				
🗸 📂 Thern	Show In		File from Template		
> 🐝 Bin 📗	Сору	Ctrl+C	Folder	Folder name: guix_gen	
> 🔊 Incl 👔	Paste	Guilt	Class		
> 😕 ra 🔰	-		Header File	Advanced >>	
> 😕 ra (	Source		Source File Source Folder		
> 🔑 src > 🗁 Del	Move				
> 🕞 bui	Rename		Example		
> 🗁 gui 🚬	Import		Other Ctrl+N	Fini	ish Cancel

Figure 27. Creating a "guix\_gen" in e<sup>2</sup>studio Project



22. Confirm "guix\_gen" is created before moving to next step.



Figure 28. Confirming Creation of "guix\_gen"

23. In Azure RTOS GUIX Studio, click **Project→Generate All Output Files** to generate resource files, header files and source files of this GUIX design.

Project Edit Insert Configure H	Help	Displays 
New Project	Ctrl+N	theme_1
Open Project	Ctrl+O	color table
Save Project	Ctrl+S	pixelmap table
Save Project As	Shift+Ctrl+S	
Close Project		
Import Project		
Recent Projects	>	Specify Resource File Name
Generate All Output Files		☐ binary mode
Generate Resource Files		Binary Mode File Format S-Record OBinary Memory Offset: 0000
Generate Specification Files		
Exit	Alt+F4	Cancel Generate

Figure 29. Clicking Generate All Output Files

Click Generate to generate all output files. If succeeded, you will see below notification.

A Notification	×
All Output files have been updated	
	ОК





24. All output files are now in "guix\_gen" folder.



Figure 31. Location of Output Files

25. In the Azure RTOS GUIX Studio Project, click "Splash" and pick up "Widget Name" and "Event Function" definitions. These definitions are used to create a screen and handle it in the e2studio/FSP project. The other windows have similar definitions.

🔒 Project Vie	2014			
🗅 thermostat				
🖶 🖳 display				
🖮 🔤 default_fol	lder			
🖽 📑 Help				
🗈 🗂 MainPa				
B-C Setting				
B				
🗄 📑 Thermo	ostat			
Deservations	10			
E Properties				
Widget Type	window	^	+	
Widget Name	Splash			Ver : 1.0.0
Widget Id				
User Data				
Left	0			
Тор	0			
Width	480			
Height	272			
Border	No Border	$\sim$	March 20, 2025	
Transparent			Thursday	09:30:00 AM
Draw Selected				+
Enabled				
Accepts Focus				
Runtime Allocate				
Normal fill	WINDOW_FILL	~		
Selected fill	WINDOW_FILL	~		
Disabled fill	WINDOW_FILL	~		
Template				
Visible At Startup				
Draw Function				
the second se	splashscreen_event			
Event Function	spidshisereen_event			

Figure 32. Definitions in the Azure RTOS GUIX Studio Project



- 26. Copy and replace the files in "src" folder in e<sup>2</sup> studio project with the files in "2.23" folder in the AN folder:
  - hmi\_event\_handler.c
  - system\_thread\_entry.c

**Build** Thermostat\_GUIX\_EK\_RA6M3G project you will see several warnings, but we will address them in later steps.

27. **Code highlight:** The following example creates a screen based on Widget Name in GUIX project and attached it to the root window. In this case, it is the "Splash" screen. Refer to system\_thread\_entry.c for more details.

```
/* Create the widget and attached to root window.*/
gx_err = gx_studio_named_widget_create("Splash", (GX_WIDGET *) p_root, (GX_WIDGET **) &p_splash_screen);
if(GX_SUCCESS != gx_err)
{
     APP_ERR_TRAP(FSP_ERR_ASSERTION);
}
```

28. **Code highlight:** An event function associated with a screen needs to be defined to handle events on that screen. Refer to hmi\_event\_handler.c for more details. All event functions are empty at this point.



29. Get your EK-RA6M3G ready to run the project. Connect LCD board to **Graphics Expansion** connector on EK-RA6M3 as shown below.



Figure 33. Connecting LCD Board to Graphics Expansion Connector of EK-RA6M3



- 30. Connect EK-RA6M3G kit to your PC using **J10**. **Download and Run** Thermostat\_GUIX\_EK\_RA6M3G project, you will see a black screen.
- 31. Add the following code to **splashscreen\_event** function in **hmi\_event\_handler.c** to show Splash screen. **Build** the e<sup>2</sup> studio project.

```
switch (event_ptr->gx_event_type)
{
    case GX_EVENT_SHOW:
        gx_err = gx_window_event_process(widget, event_ptr);
        if(GX_SUCCESS != gx_err) {
            while(1);
        }
            break;
    default:
        gx_err = gx_window_event_process(widget, event_ptr);
        if(GX_SUCCESS != gx_err) {
            while(1);
        }
        break;
    }
```

}

Please refer to splashscreen\_event function in hmi\_event\_handler.c in "**2.28**" folder in the AN folder. 32. **Download and Run** the project, you will see the Splash screen on LCD panel.



Figure 34. Splash Screen View on LCD

# 3. Using GUIX Widget Timer to Trigger a Screen Transition

# 3.1 Overview

In this section, you will implement a simple use of GUIX Widget timer, which is to trigger a screen transition.

# 3.2 Procedural Steps

1. Copy and replace the files in "**src**" folder in e<sup>2</sup> studio project with the files in "**3.1**" folder in the AN folder:

hmi\_event\_handler.c

system\_thread\_entry.c



2. **Code highlight**: The following code in splashscreen\_event function starts a GUIX Widget timer and trigger a screen transition that hides Splash screen and shows Main Page screen.

```
switch (event ptr->gx event type)
    {
        case GX EVENT TIMER:
            gx system timer stop(widget, 10);
            toggle screen(p mainpage screen,p splash screen);
            break;
        case GX EVENT SHOW:
            gx system timer start(widget, 10 , SPLASH TIMEOUT,
SPLASH TIMEOUT);
            gx err = gx window event process (widget, event ptr);
            if(GX SUCCESS != gx err) {
                while(1);
            }
            break;
        default:
            gx err = gx window event process(widget, event ptr);
            if(GX SUCCESS != gx err) {
                while(1);
            }
            break;
    }
```

3. **Build, Download, and Run** the project, you will see the transition from Splash screen to Main Page screen in about 3 seconds.



Figure 35. Main Page Screen



# 4. Add Touch Driver to Thermostat\_GUIX\_EK\_RA6M3G Project

# 4.1 Overview

In this section, you will add the ft5x06 touch driver to the project to handle touch events on LCD panel.

# 4.2 Procedural Steps

1. In Thermostat\_GUIX\_EK\_RA6M3G project, create a folder by right-clicking "**src**", then select "New-→Folder".

Enter or selec	the narest fe	Idan			
	GUIX_EK_RA6				
se 🔁 .se De 🔁 (	ttings bug ix_studio .cfg gen ipt	_EK_RA6M3G [I	Debug]		
Folder name:	touch ft5v0f	5			
rolaer name.	>	,			

Figure 36. Creating New Folder in Thermostat\_GUIX\_EK\_RA6M3G Project Click Finish to create "touch\_ft5x06" folder.



2. Copy touch\_ft5x06.c and touch\_ft5x06.h from "touch\_ft5x06" folder in the Lab folder to the one in e<sup>2</sup> studio project.



Figure 37. Copying files to the e<sup>2</sup> studio Project

3. Open project configuration and create **Touch Thread** with the settings below.

Threads	🕢 New Thread 🛍 Remove 📄 Touch Threa	ad Stacks
<b>₩</b> ▼		ld stacks to the selected m the clipboard.
Objects	New Object > 🕡 Remove	
-		
Summary	BSP Clocks Pins Interrupts Event Links Stacks Components	
Propert	ies 🔀 😰 Problems  📮 Console 🁒 Smart Browser 🔑 Smart Manual 🛷 Search	n 🚺 Memory
Touch T	aread	
Touch II	neud	
Settings	Property	Value
-	✓ Common	
	> General	
	> Timer	
	> Trace	
	> Trace > Performance	
	> Trace > Performance > RA	
	<ul> <li>&gt; Trace</li> <li>&gt; Performance</li> <li>&gt; RA</li> <li>&gt; Interrupts</li> </ul>	
	<ul> <li>&gt; Trace</li> <li>&gt; Performance</li> <li>&gt; RA</li> <li>&gt; Interrupts</li> <li>Y Thread</li> </ul>	lauch durad
	<ul> <li>&gt; Trace</li> <li>&gt; Performance</li> <li>&gt; RA</li> <li>&gt; Interrupts</li> <li>Y Thread</li> <li>Symbol</li> </ul>	touch_thread
	<ul> <li>&gt; Trace</li> <li>&gt; Performance</li> <li>&gt; RA</li> <li>&gt; Interrupts</li> <li>Y Thread</li> <li>Symbol</li> <li>Name</li> </ul>	Touch Thread
	<ul> <li>&gt; Trace</li> <li>&gt; Performance</li> <li>&gt; RA</li> <li>&gt; Interrupts</li> <li>&gt; Thread</li> <li>Symbol</li> <li>Name</li> <li>Stack size (bytes)</li> </ul>	Touch Thread 1024
	<ul> <li>&gt; Trace</li> <li>&gt; Performance</li> <li>&gt; RA</li> <li>&gt; Interrupts</li> <li>Y Thread</li> <li>Symbol</li> <li>Name</li> </ul>	Touch Thread

Figure 38. Creating Touch Thread



- 4. The pins marked in red below are used for touch panel controller on the LCD board:
  - IRQ0 interrupt (P206) is used to trigger touch events.
  - I2C channel 2 (P512, P511) is used to read and write data to the touch controller. P304 is used to reset the touch controller.



Figure 39. Pins Used in Touch Panel Controller (marked in red)

5. Since the IRQ0 (P206) needs a pull-up to function properly with the LCD board, do not change the setting that was done by FSP, as shown below.

elect Pin Configuration		Export to C	🔚 Export to CSV file 🛛 🖺 Configure Pin Driver Warnings			
RA6M3G-EK.pincfg	✓ <u>Ma</u>	Gener	Generate data: g_bsp_pin_cfg			
in Selection	⊞ ⊟ ↓ <mark>ª</mark> z	Pin Configuration				
Type filter text		Name	Value	Link		
✓ ✓ P2	^	Symbolic Name				
P200		Comment				
P201		Mode	Input mode			
✓ P202		Pull up	input pull-up			
✓ P203		IRQ	IRQ0-DS			
✓ P204		Drive Capacity	Low			
✓ P205		Output type	CMOS			
✓ P206		Input/Output				
P207		P206	🗸 GPIO	$\rightarrow$		
✓ P208						
✓ P209						
✓ P210						
✓ P211		Module name: P2	206			
✓ P212		Port Capabilities: Bl	JS0: WAIT			
✓ P213	¥	C C	ISU0: TS01			

Figure 40. FSP Setting



6. In e<sup>2</sup> studio project configuration, add **External IRQ Driver on r\_icu** to **Touch Thread** with the following settings.

<ul> <li></li></ul>		uch_irq External IRQ (r_ic g_touch_irq External IRQ (r_icu)	•	
Problem	s Console Properties $ imes$ Smart Browser	Smart Manual Se	arch	
a touch	irq External IRQ (r_icu)			
2				
Settings	Property Common		Value	
API Info	Parameter Checking		Default (BSP)	
	<ul> <li>Module g_touch_irg External IRQ (r_icu)</li> </ul>			
	Name		g_touch_irq	
	Channel		0	
	Trigger		Falling	
	Digital Filtering		Enabled	
	Filter Source		PCLK filter	
	Digital Filtering Sample Clock (Only valid when Dig	ital Filtering is Enabled)	PCLK / 64	
	Callback		touch_irq_cb	
	Pin Interrupt Priority		Priority 5	
	✓ Pins			
	IRQ00		None	

Figure 41. Adding External IRQ Driver on r\_icu to Touch Thread

7. In project configuration, add I2C Master Driver on r\_iic\_master to Touch Thread with below settings.

Stacks C	onfiguration			Generate Project Co
V 🕸 Tou	New Thread New Thread Duch ing External IRQ (r_jcu) Duch ing External IRQ (r_jcu) New Object > Remove SP Clocks Pins Interrupts Event Links Link	Touch Thread Stacks	Nr     g_J2c_touch I2C Master (r_ii      g_J2c_transfer     (r_drc) IIC2 TXI (fransfer     (r_drc) IIC2 TXI (fransmit     data empty)	ew Stack > 🔮 Extend Stack > 🕷 Ren ic_master)
Problems	Console Properties X Smart	Browser Smart Manual		
a i2c tou	ch I2C Master (r iic master)			
Settings API info	Property Parameter Checking DTC on Transmission and Reception 10-bit slave addressing Channel Rate Channel Rate Custom Rate (bps) Rise Time (ns) Fall Time (ns) Duty Cycle (%) Slave Address Address Address Address Address Address Interrupt Priority Level Plins SDA SCL	iter)	Value Default (85P) Enabled Jisabled 2 Fast-mode 0 120 120 120 50 0 X38 7-Sit Short Mode Enabled touch_12c_callback Priority 6	

Figure 42. Adding I2C Master Driver on r\_iic\_master to Touch Thread



8. In project configuration, add **Touch Semaphore** as shown below. We use this semaphore to signal the Touch thread when a touch event occurs. The Touch thread then sends the touch event to GUIX.

Stacks Configuration	Generate Project Conte
Threads       New Thread       Remove         Image: System Thread       Image: System Thread         Image: System	Touch Thread Stacks       Image: Remove         Image: Remove       Image: Remove         Image:
Summary BSP Clocks Pins Interrupts Event Links	s Stacks Components Smart Browser 🔑 Smart Manual 🔗 Search 🚺 Memory
g_new_semaphore0 Semaphore	Smarcolowsci 🦏 Smarciwandar 🧳 Search 😈 Wellioly
Settings Property Name Symbol Initial count	Value Touch Semaphore g_touch_semaphore 0

Figure 43. Adding Touch Semaphore

9. In project configuration, add **I2C Semaphore** as shown below. This semaphore is used in the ft5x06 driver to trigger data reading when a touch-panel interrupt occurs.

Stacks Configuration	Generate Project Content
Threads 🔄 New Thread 🚯 Remove 📄	Touch Thread Stacks
Inreads     New Inread     Inreads     Inreads     Inreads       Inreads     Inreads     Inreads     Inreads     Inreads	
Summary BSP Clocks Pins Interrupts Event Li	
Properties × Reproblems Console     G_new_semaphore0 Semaphore	ቅ Smart Browser 🔑 Smart Manual 🔗 Search 🔋 Memory
Settings Property Name	Value I2C Semaphore
Symbol Initial count	g_i2c_semaphore 0

Figure 44. Adding I2C Semaphore



- 10. In RA Configurator, click Generate Project Content to generate project content.
- 11. Copy and replace the files in "**src**" folder in the e<sup>2</sup> studio project with the files in "**4.11**" folder in the AN folder:
  - hmi\_event\_handler.c
  - system\_thread\_entry.c
  - touch\_thread\_entry.c

#### 12. Code highlight: Below code in touch\_thread\_entry.c get touch data and send touch event to GUIX.

```
/* Get touch data from the FT5X06 */
ft5x06_payload_get(&touch_data);
/* Send touch data*/
if(1 == touch_data.num_points)
{
    gxe.gx_event_payload.gx_event_pointdata.gx_point_x
    gxe.gx_event_payload.gx_event_pointdata.gx_point_y
```

= touch\_data.point[0].x; = touch\_data.point[0].y;

```
gxe.gx_event_type = GX_EVENT_PEN_DOWN;
```

gx\_system\_event\_send(&gxe);

```
}
```

```
else if (GX_EVENT_PEN_DOWN == gxe.gx_event_type)
```

```
{
```

```
gxe.gx_event_type = GX_EVENT_PEN_UP;
```

# gx\_system\_event\_send(&gxe);

}

13. All the screens designed in the Azure RTOS GUIX Studio project are now created in system\_thread\_entry.c





The code marked in red in hmi\_event\_handler.c handle touch event when Thermostat button and Settings button are clicked. Refer to hmi\_event\_handler.c for more details.





14. **Build, Download, and Run** the e<sup>2</sup> studio project. Then, you will be able to go back and forth from the Main Page screen to Thermostat screen and Settings screen using **Thermostat** and **Settings** buttons on Main Page screen and "**Back**" button on the other two screens.



Figure 45. Navigating between Main Page Screen and Thermostat Screen



# 5. Control LCD Backlight

# 5.1 Overview

In this section, you will use a PWM output pin of a GPT timer to control the intensity (brightness) of LCD backlight.

# 5.2 Procedural Steps

1. In LCD board schematics below, the LCD\_BLEN signal, which is connected to the P603 on the RA6M3 MCU, is configured in PWM mode to control the intensity of LCD backlight.



Figure 46. LCD Board Schematic

2. To configure P603 in PWM output mode, we disable it in Pin Configuration at first. **Save this change before moving to the next step.** 

Select Pin Configuration			📑 Export to CSV fil	e 🔚 Configure Pin Driver	Warnings
RA6M3G-EK.pincfg	✓ Mar	nage configurations	🗹 Generate o	ata: g_bsp_pin_cfg	
Pin Selection	₽ ₽↓ <mark>a</mark> z	Pin Configuration			
Type filter text	^	Name Symbolic Name Comment	Value	Link	
> * P4 > * P5 * * P6		Mode Pull up	Disabled None		
<ul> <li>✓ P600</li> <li>✓ P601</li> </ul>		Drive Capacity Output type V Input/Output	CMOS		
✓ P602 P603 P604		P603	None	$\Rightarrow$	
P605 P606 P607					
<ul> <li>✓ P608</li> <li>✓ P609</li> <li>✓ P610</li> </ul>			3 0: D13_DQ13 7: GTIOCA		

Figure 47. Disabling P603 in Pin Configuration



#### 3. In Pin Configuration, set P603 as GPT7 GTIOCA output.



#### Figure 48. Setting P603 as GPT7 GTIOCA Output in Pin Configuration

4. In project configuration, add Timer Driver on r\_gpt to System Thread with below settings.

100		g_timer_PWM Timer, General PWM (r_gpt) Stacks					
	L/Common ^	g_timer_PWM Timer,					
	stem Thread	General PWM (r gpt)					
4	Azure RTOS GUIX						
$\oplus$	g_timer_PWM Timer, General PWM (r_gpt)	<b>(i)</b>					
		<u> </u>					
Objects	된 New Object > 🛍 Remove						
<b>a</b> i2a	semaphore Semaphore						
• <u>g_izc_</u>	semaphore Semaphore v						
Summary	BSP Clocks Pins Interrupts Event Links Linker Sections Stacks	omponents					
Problem	s Console Properties × Smart Browser Sma	Manual Search					
a timer	PWM Timer, General PWM (r_gpt)						
9							
Settings	Property	Value					
API Info	✓ Common						
AFTINO	Parameter Checking	Default (BSP)					
	Pin Output Support	Enabled					
	Write Protect Enable	Disabled					
	<ul> <li>Module g_timer_PWM Timer, General PWM (r_gpt)</li> </ul>						
	✓ General						
	Compare Match						
	Name	g_timer_PWM					
	Channel	7					
	Mode	Saw-wave PWM					
	Period	100000					
	Period Unit	Nanoseconds					
	✓ Output						
	> Custom Waveform						
	Duty Cycle Percent (only applicable in PWM mode)	10					
	GTIOCA Output Enabled	True					
	GTIOCA Stop Level	Pin Level Low					
	GTIOCB Output Enabled	False					
	GTIOCB Stop Level	Pin Level Low					

Figure 49. Adding Timer Driver on r\_gpt to System Thread

Even though the duty cycle of PWM output is purposely set to **10%** here, it will be changed to **50%** later in the code.

# O

5. In RA Configurator, click Generate Project Content to generate project content.



- 6. Copy and replace the files in "**src**" folder in e<sup>2</sup> studio project with the files in "**5.6**" folder in the AN folder:
  - hmi\_event\_handler.c
  - system\_thread\_entry.c
  - brightness.c
  - brightness.h
  - system\_api.h
  - system\_cfg.h
- 7. brightness\_up and brightness\_down functions in brightness.c are used to set the PWM duty cycle, as shown below:

```
/* Get the current period setting. */
R_GPT_InfoGet(&g_timer_PWM_ctrl, &info);
/* Calculate the desired duty cycle based on the current period. */
duty_cycle_count = (uint32_t) ((info.period_counts *
brightness)/GPT_PWM_MAX_PERCENT);
err = R_GPT_DutyCycleSet(&g_timer_PWM_ctrl, duty_cycle_count,
GPT IO PIN GTIOCA);
```

8. Looking at gpt\_timer\_PWM\_Setup function in system\_thread\_entry.c, you will see brightness (duty cycle of PWM output) is set to 50 percent.





9. **Build, Download, and Run** the e<sup>2</sup> studio project. By clicking the **Settings** button on **Main Page** screen, you can access **Settings** screen.



Figure 50. Settings Button on Main Page Screen

10. PWM output measured on pin P603 with brightness is set to 50%.



Figure 51. PWM Output on P603 at 50% Brightness

11. Click "**Display**" menu on **Settings** screen, you can use "**Up**" and "**Down**" buttons to change the brightness of LCD backlight.



Figure 52. Display on Settings Screen

12. PWM output measured on pin P603 after changing brightness to **65%**.



Figure 53. PWM Output on P603 at 65% Brightness



# 6. Update Date/Time and Temperature

# 6.1 Overview

In this section, you will enable RTC controller as a timekeeper and one ADC channel to read the MCU die's temperature sensor and use it as Thermostat temperature data.

# 6.2 Procedural Steps

1. In project configuration, create **Temperature Time Thread**.

Threads		Temper	ature Time Thread	d Stacks
<ul> <li>♥</li> <li>Sys</li> <li>♥</li> </ul>	L/Common g_ioport I/O Port (r_ioport) stem Thread Azure RTOS GUIX g_timer_PWM Timer, General PWM (r_gpt) uch Thread g_touch_irq External IRQ (r_icu) g_i2c_touch I2C Master (r_iic_master) mperature Time Thread		Add stacks to the clipboard.	e selected
• g_i2c_	New Object > Remove     Remove     Semaphore Semaphore     Semaphore Semaphore     BSP Clocks Pins Interrupts Event Links     Semaphore     Semaphore			🔗 Sear
Tempera	ture Time Thread			
Settings	Property > Common Thread Symbol Name Stack size (bytes) Priority	-	ture_time_thread ture Time Thread	
	Auto start Time slicing interval (ticks)	Enabled 10		
	Time siding interval (ucks)	10		

Figure 54. Create Temperature Time Thread

2. In project configuration, add RTC Driver on g\_rtc to Temperature Time Thread.



Threads	🕢 New Thread ᇵ Remove  📄	Temperature Time Thread Stacks
<ul> <li>♥</li> <li>♥</li></ul>	IJ/Common g_ioport I/O Port (r_ioport) stem Thread Azure RTOS GUIX g_timer_PWM Timer, General PWM (r_gpt) uch Thread g_touch_irq External IRQ (r_icu) g_i2c_touch I2C Master (r_iic_master) nperature Time Thread g_rtc Realtime Clock (r_rtc)	g_rtc Realtime Clock (r_rtc)
<ul> <li>g_i2c_</li> <li>Summary</li> <li>Properti</li> </ul>	ch_semaphore Semaphore semaphore Semaphore BSP Clocks Pins Interrupts Event Links Stac ies X Problems Console Smart ealtime Clock (r rtc)	
Settings	Property > Common	Value
API Info	Module g_rtc0 Realtime Clock (r_rtc) Name Clock Source Frequency Comparision Value (LOCO) Automatic Adjustment Mode Automatic Adjustment Period Adjustment Type (Plus-Minus) Error Adjustment Value Callback	g_tc LOCO 255 Enabled 10 Seconds NONE 0 time_update_callback
	Alarm Interrupt Priority Period Interrupt Priority	Disabled Priority 9

Figure 55. Adding RTC Driver on g\_rtc to Temperature Time Thread

3. In project configuration, add **ADC Driver on r\_adc** to **System Thread**.

<ul> <li>Te</li> <li>Te</li> <li>Te</li> <li>Te</li> <li>Te</li> </ul>	uch Thread		
g_adc Al	DC (r_adc)		
Settings	Property	Value	
API Info	✓ Common		
	Parameter Checking	Default (BSP)	
	<ul> <li>Module g_adc ADC (r_adc)</li> </ul>		
	✓ General		
	Name	g_adc	
	Unit	0	
	Resolution	12-Bit	
	Alignment	Right	
	Clear after read	On	
	Mode	Continuous Scan	
	Double-trigger	Disabled	
	> Input		
	> Interrupts		
	> Extra		

Figure 56. Adding ADC Driver on r\_adc to System Thread



#### 4. Select **Temperature Sensor** as input source for g\_adc module.

g_adc ADC (r_adc)	
Settings Property	Value
API Info	
Parameter Checking	Default (BSP)
<ul> <li>Module g_adc ADC (r_adc)</li> </ul>	
> General	
✓ Input	
<ul> <li>Channel Scan Mask (channel availability varies by</li> </ul>	MCU)
Channel 0	
Channel 1	
Channel 2	
Channel 3	
Channel 4	
Channel 5	
Channel 6	
Channel 7	
Channel 8	
Channel 9	
Channel 10	
Channel 11	
Channel 12	
Channel 13	
Channel 14	
Channel 15	
Channel 16	
Channel 17	
Channel 18	
Channel 19	
Channel 20	
Channel 21	
Channel 22	
Channel 23	
Channel 24	
Channel 25	
Channel 26	
Channel 27	
Channel 28	
Temperature Sensor	
Internal Reference Voltage	
<ul> <li>Group B Scan Mask (channel availability varies by</li> </ul>	
<ul> <li>Addition/Averaging Mask (channel availability varies by</li> </ul>	

Figure 57. Selecting Temperature Sensor as Input Source for g\_adc

5. Create **g\_timer\_semaphore** with the following settings. We use this semaphore to trigger the date and time update every second.

Objects       Image: New Object > Image: Remove         Image: g_i2c_semaphore Semaphore       Image: g_display0 Display Driver on r_glcdc         Image: g_timer_semaphore Semaphore       Image: g_display0 Display Driver on r_glcdc         Image: g_timer_semaphore Semaphore       Image: g_display0 Display Driver on r_glcdc         Image: g_timer_semaphore Semaphore       Image: g_display0 Display Driver on r_glcdc					+ g_jpe Drive	
Summary       BSP       Clocks       Pins       Interrupts       Event Links       Stacks       Components         Problems       Console       Properties       Smart Browser       Smart Manual       Debug         g_timer_semaphore       Semaphore						
Settings Property Name Symbol Initial count				Value Timer Semaphor g_timer_semaph 0		

Figure 58. Creating g\_timer\_semaphore



O

- 6. In RA Configurator, click Generate Project Content to generate project content.
- 7. Copy and replace the files in **"src"** folder in e<sup>2</sup> studio project with the files in **"6.6"** folder in the Lab folder:
  - hmi\_event\_handler.c
  - system\_thread\_entry.c
  - system\_time.c
  - system\_time.h
  - system\_api.h
- 8. In System Thread, date/time data and temperature data get updated every second. It then sends out events to trigger GUIX updates.

```
while (1)
{
    /* Wait for RTC interrupt. */
    status = tx_semaphore_get(&g_timer_semaphore, TX_WAIT_FOREVER);
    if(TX_SUCCESS != status)
    {
        APP_ERR_TRAP(FSP_ERR_ASSERTION);
    /* Get date, time */
    R_RTC_CalendarTimeGet(&g_rtc_ctrl, &g_gui_state.time);
       Send GUIX event to update
    send_hmi_message(GXEVENT_MSG_TIME_UPDATE);;
    /* Delay and update temperature*/
    tx_thread_sleep (10);
    /* Read die temperature */
    err = R_ADC_Read(&g_adc_ctrl, ADC_CHANNEL_TEMPERATURE, &adc_temp_data);
       Handle erro
    if (FSP_SUCCESS != err)
    {
        APP_ERR_TRAP(err);
    }
    /* Conversion of ADC temperature in celsius */
    g_gui_state.temp_c = ADCTEMP_AS_C(adc_temp_data);
    /* Send GUIX event to update time */
send_hmi_message(GXEVENT_MSG_UPDATE_TEMPERATURE);
    tx thread sleep (1);
```

9. The following is an example of handling temperature and time update events in the Main Page screen event handler.





10. **Build, Download, and Run** the e<sup>2</sup> studio project. You will see time and temperature get updated every second.





# 7. Setting Date/Time in A Full Function Project

# 7.1 Overview

In this section, you will import and run the complete Thermostat project that enables the settings of date and time. Upon user press date and time buttons on the settings screen, a message will be sent to the system thread to update the date and time, then the system thread will send a GUIX event to trigger time display update on screens.

# 7.2 Procedural Steps

 You can try the completed project in "completed\_project" folder that has a full function Thermostat application. Use "Rename & Import Existing C/C++ Project into Workspace" feature of Import menu in e<sup>2</sup> studio to do so since you already had a project with the same in the workspace.

			🕲 Import - 🗆 🗙
			Select
			Rename and Import and Existing C/C++ Project into the workspace
		- 15 - P	Select an import wizard:
File	Edit Navigate Search Project		type filter text
	New Cite	Alt+Shift+N >	V 🗁 General
~	Open File		Generation      Generation
	Open Projects from File System Recent Files	>	😂 Existing Projects into Workspace
			🗅 File System
	Close Editor	Ctrl+W	Preferences     Projects from Folder or Archive
	Close All Editors	Ctrl+Shift+W	🖉 Rename & Import Existing C/C++ Project into Workspace
	Save	Ctrl+S	Renesas CS+ Project for CA78K0R/CA78K0 Renesas CS+ Project for CC-RX and CC-RL
	Save As		> C/C++
B	Save All	Ctrl+Shift+S	> ≽ İnstall
	Revert		> 🗁 Oomph
	Move		> 🗁 Team
	Rename	F2	> > Tracing
5	Refresh	F5	> 🗁 XML
	Convert Line Delimiters To	>	
Ð	Print	Ctrl+P	
2	Import Import		
4	Export		
	Properties	Alt+Enter	
	Switch Workspace	>	
	Restart		
	Exit		 

Figure 60. Rename & Import Existing C/C++ Project into Workspace on Import Menu

# 8. Website and Support

Visit the following URLs to learn about key elements of the RA family, download components and related documentation, and get support:



RA Product Information RA Product Support Forum RA Flexible Software Package Renesas Support renesas.com/ra renesas.com/ra/forum renesas.com/FSP renesas.com/support



# **Revision History**

		Descript	Description			
Rev.	Date	Page	Summary			
1.00	Mar.31.23	—	Initial release			
1.01	Jul.17.23	—	Updated for FSP v4.4.0.			
1.02	Jun.03.24	—	Updated for FSP v5.2.0.			
1.03	Jul.08.25	—	Updated for FSP v6.0.0.			



# General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

#### 1. Precaution against Electrostatic Discharge (ESD)

A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

#### 2. Processing at power-on

The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power reaches the level at which resetting is specified.

3. Input of signal during power-off state

Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.

4. Handling of unused pins

Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.

5. Clock signals

After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.

#### 6. Voltage application waveform at input pin

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between  $V_{IL}$  (Max.) and  $V_{IH}$  (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between  $V_{IL}$  (Max.) and  $V_{IH}$  (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between  $V_{IL}$  (Max.) and  $V_{IH}$  (Min.).

7. Prohibition of access to reserved addresses

Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

8. Differences between products

Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

#### Notice

- Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
- Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
- 3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
- 4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
- 5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
- 6. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

"Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.

- 7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
- 8. When using Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
- 9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
- 10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
- 11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
- 12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
- This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
   Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas
- Electronics products.
- (Note1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
- (Note2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

# **Corporate Headquarters**

TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

#### www.renesas.com

# Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

# **Contact information**

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit: <u>www.renesas.com/contact/</u>.