To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of 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. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. 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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may 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.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

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

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, please evaluate the safety of the final products or system 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. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

User's Manual

Renesas Starter Kit for RSKRX610

Software Manual

Renesas 32-Bit Microcomputer
RX Family

Renesas Electronics
www.renesas.com

Rev.1.01 2010.03

Disclaimer

This software is supplied by Renesas Technology Corp. and is only intended for use with Renesas products. No other uses are authorized.

This software is owned by Renesas Technology Corp. and is protected under all applicable laws, including copyright laws.

THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT. ALL SUCH WARRANTIES ARE EXPRESSLY DISCLAIMED.

TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS TECHNOLOGY CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR ITS AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Renesas reserves the right, without notice, to make changes to this software and to discontinue the availability of this software. By using this software, you agree to the additional terms and conditions found by accessing the following link:
http://www.renesas.com/disclaimer

Table of Contents

Table of Figures

Chapter 1. Preface

Cautions

This document may be, wholly or partially, subject to change without notice.

Trademarks

All brand or product names used in this manual are trademarks or registered trademarks of their respective companies or organisations.

Copyright

Website: www.renesas.com

Glossary

ADC	Analogue / Digital Converter	I/O	Input / Output
API	Application Programming Interface	PDG	Peripheral Driver Generator
CMT	Compare Match Timer	PDL	Peripheral Driver Library
CPG	Clock Pulse Generator	PLL	Phased-Locked Loop
CRC	Cyclic Redundancy Check	PWM	Pulse Width Modulation
DMAC	Direct Memory Access Controller	RSK	Renesas Starter Kit
DTC	Data Transfer Controller	SCI	Serial Communincations Inteface
FCU	Flash Control Unit	TPU	Timer Pulse Unit
HEW	High performance Embedded Workshop	WDT	Watchdog Timer

Chapter 2. Introduction

This document explains the sample code for the RSK Renesas Starter Kit.

It explains by text and diagrams the functionality of the sample code and its interaction with the Renesas Peripheral Driver Library (RPDL).

The Renesas Peripheral Driver Library (hereinafter “this library”) is based upon a unified API for the microcontrollers made by Renesas Technology.

The latest version of this API can be found in the Documentation section of the PDG site (http://www.renesas.com/pdg).

Chapter 3. RSK Sample Code Concept

The diagram below shows the basic structure of all the RSK sample code:

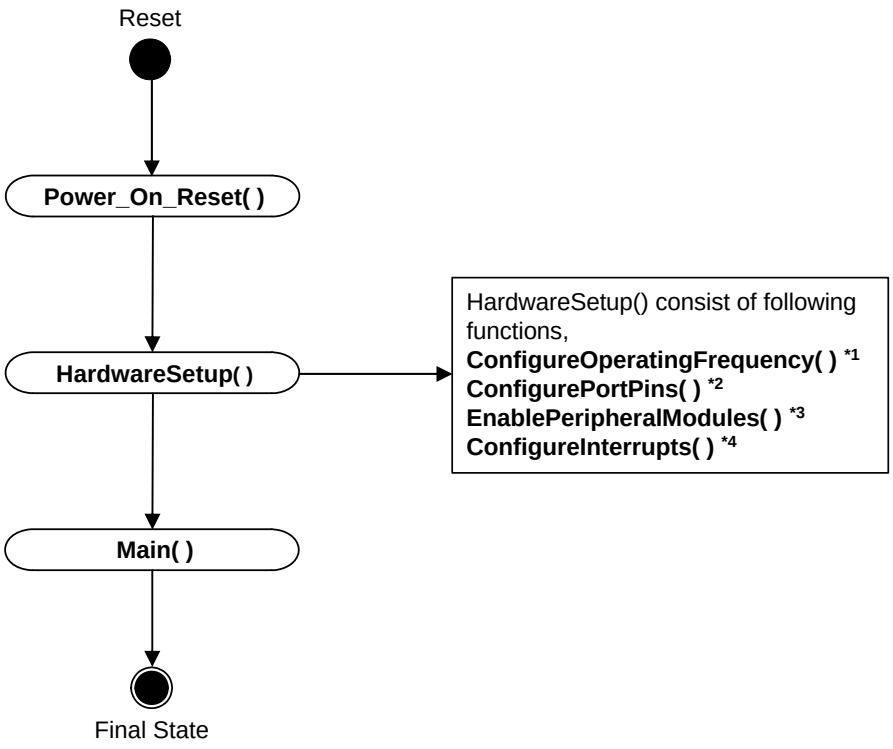

Figure 1 RSK Sample Code

1- This function initialises the CPU main clock, the real time clock and the PLL.

2- This function configures the CPU port pins as input and output. It also determines the initial state of the output port pin.

3- This function enables or disables the peripheral modules of the CPU.

4- This function configures the CPU interrupts.

3.1. List of Sample code

You can click on the sample code title for more description.

	Sample Code	Description
1	Tutorial	Demonstrates usage of the debugger and hardware
2	ADC_Oneshot	Demonstrates usage of the 10bit ADC module, in one-shot mode
3	ADC_Repeat	Demonstrates usage of the 10bit ADC in repeat mode.
4	Async_Serial	Demonstrates usage of the Serial Communications Interface module, in asynchronous mode
5	CRC	Demonstrates usage of the Cyclic Redundancy Check module.
6	Data_Flash	Demonstrates usage of Flash Control Unit, in Data Flash mode.
7	DMAC	Demonstrates usage of the Direct Memory Access Controller module
8	DTC	Demonstrates usage of the Data Transfer Controller module
9	Power_Down	Demonstrates usage of the Low Power Consumption MCU Mode
10	PWM_Mode	Demonstrates usage of the Timer Pulse Unit module, in PWM mode.
11	Sync_Serial	Demonstrates usage of the Serial Communications Interface module, in synchronous mode.
12	Timer_Capture	Demonstrates usage of the Timer, in external capture mode.
13	Timer_Compare	Demonstrates usage of the Compare Match Timer module.
14	Timer_Event	Demonstrates usage of the Timer module, in event mode
15	Timer_Mode	Demonstrates usage of the Timer module
16	WDT	Demonstrates usage of the Watch Dog Timer.

Table 3-1 List of sample code

3.1.1. Tutorial

The tutorial demonstrates the use of the debugger and the hardware. The tutorial code is common for all RSK's.

This code will call three main functions to demonstrate port pin control, interrupt usage and C variable initialization. These functions are shown below.

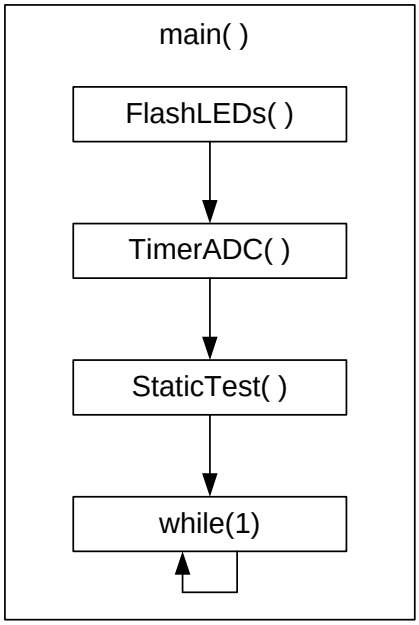

Figure 2 Functions used in Tutorial code

3.1.1.1. Sequence Diagram

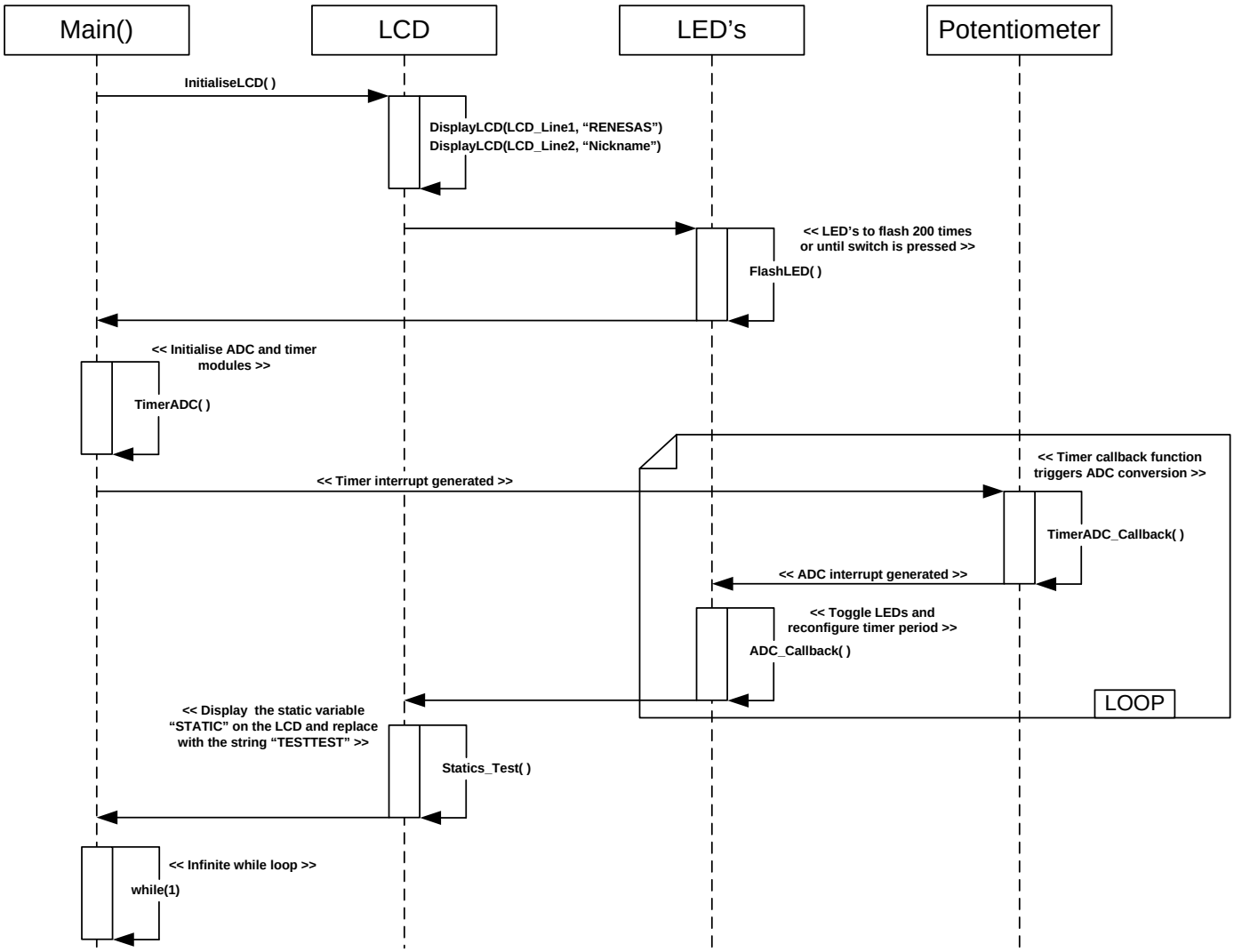

Figure 3 Tutorial Sequence Diagram

3.1.1.2. Description

1- The tutorial code initializes the LCD module and displays “Renesas” on Line-1 and the RSK nickname on Line-2.

2- All the LED’s start flashing. The LED’s flash 200 times or until switch is pressed.

3- After initial 200 flashes LED’s flashing, the flash rate is now controlled by the potentiometer (RV1).

4- At the same time the LCD will display a variable “STATIC” which is then replaced by “TESTTEST” to demonstrate the static variable test.

	Function	RPDL API Function
1	FlashLEDs	R_TMR_CreatePeriodic
2	TMR_Callback	R_IO_PORT_Modify
3	TimerADC	R_INTC_ControlExtInterrupt
4	StartTimer	R_TMR_CreateUnit
5	StartADC	R_ADC_10_Create
6	TimerADC_callback	R_ADC_10_Control
7	ADC_callback	R_ADC_10_Read
		R_TMR_ControlUnit

Table 3-2 RPDL API function list for Tutorial code

3.1.2. ADC_OneShot

This sample code demonstrates the use of on-chip ADC module for one shot conversion.

3.1.2.1. Sequence Diagram

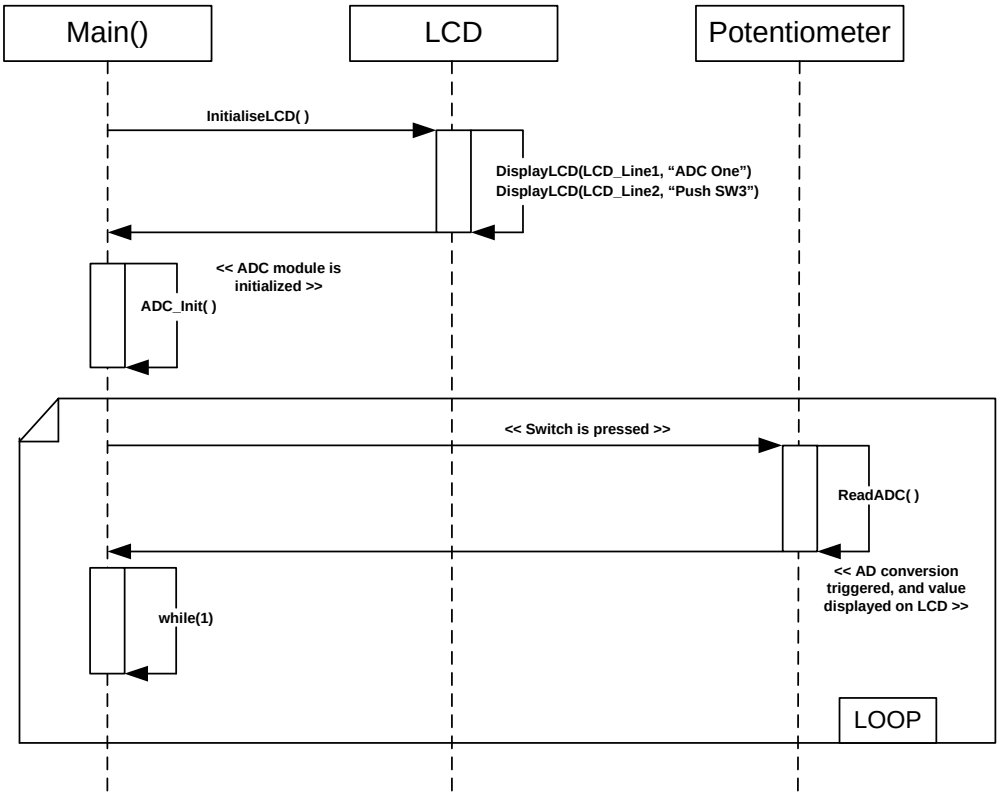

Figure 4 ADC_OneShot Sequence Diagram

3.1.2.2. Description

1- ADC_OneShot sample code initializes the LCD module and display “ADC One” on Line-1 and “Push SW3” on Line-2.

2- The ADC channel is configured for one shot conversion.

3- The result is stored in the “ad_value” variable and user may examine the AD conversion result in “ad_value” using HEW C watch window.

4- The debug LCD also displays the result of AD conversion in hex format.

5- The AD conversion result is shown on LCD when switch is pressed.

	Function	RPDL API Function
1	ADC_Init( )	R_ADC_10_Create
2	ReadADC()	R_ADC_10_Control R_ADC_10_Read

Table 3-3 RPDL API function list for ADC_OneShot

3.1.3. ADC_Repeat

This sample code demonstrates the use of on-chip ADC module for repeat conversion.

3.1.3.1. Sequence Diagram

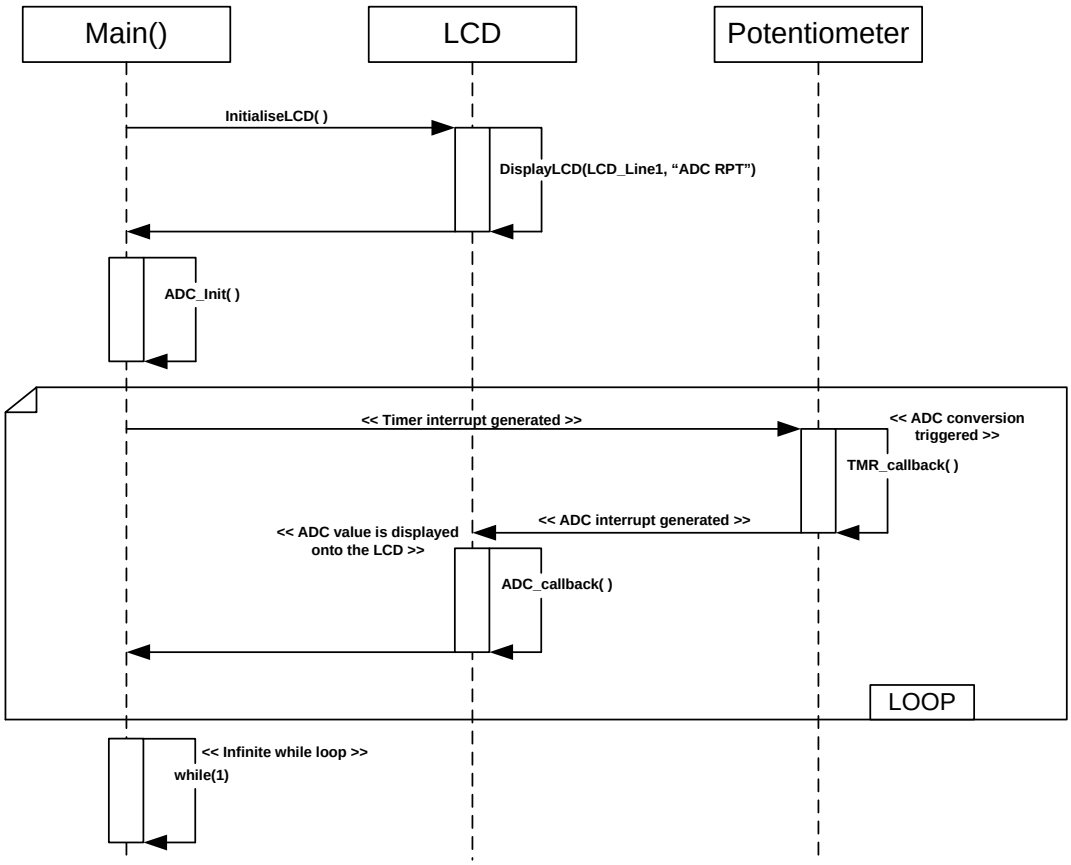

Figure 5 ADC_Repeat Sequence Diagram

3.1.3.2. Description

1- The ADC_Repeat sample code initializes the LCD module and displays “ADC RPT” on Line-1 and AD conversion result on Line-2.

2- The function, ADC_Init, configures the ADC module for repeat conversion; and configures the timer module to generate periodical interrupts.

3- The timer period interrupt calls the callback function, TMR_callback. This function triggers an ADC conversion.

4- When the ADC conversion is complete; an interrupt is generated, which calls the callback function ADC_callback. This function fetches the ADC value, and displays it onto the LCD.

5- The result is stored in the “usADC_Result” variable and user may examine the AD conversion result in “usADC_Result” using HEW C watch window.

	Function	RPDL API Function
1	ADC_Init( )	R_ADC_10_Create R_TMR_CreatePeriodic
2	TMR_callback( )	R_ADC_10_Control
3	ADC_callback( )	R_ADC_10_Read

Table 3-4 RPDL API function list for ADC_Repeat

3.1.4. Async_Serial

This sample code demonstrates an asynchronous serial communication using the on-chip serial interface module.

3.1.4.1. Sequence Diagram

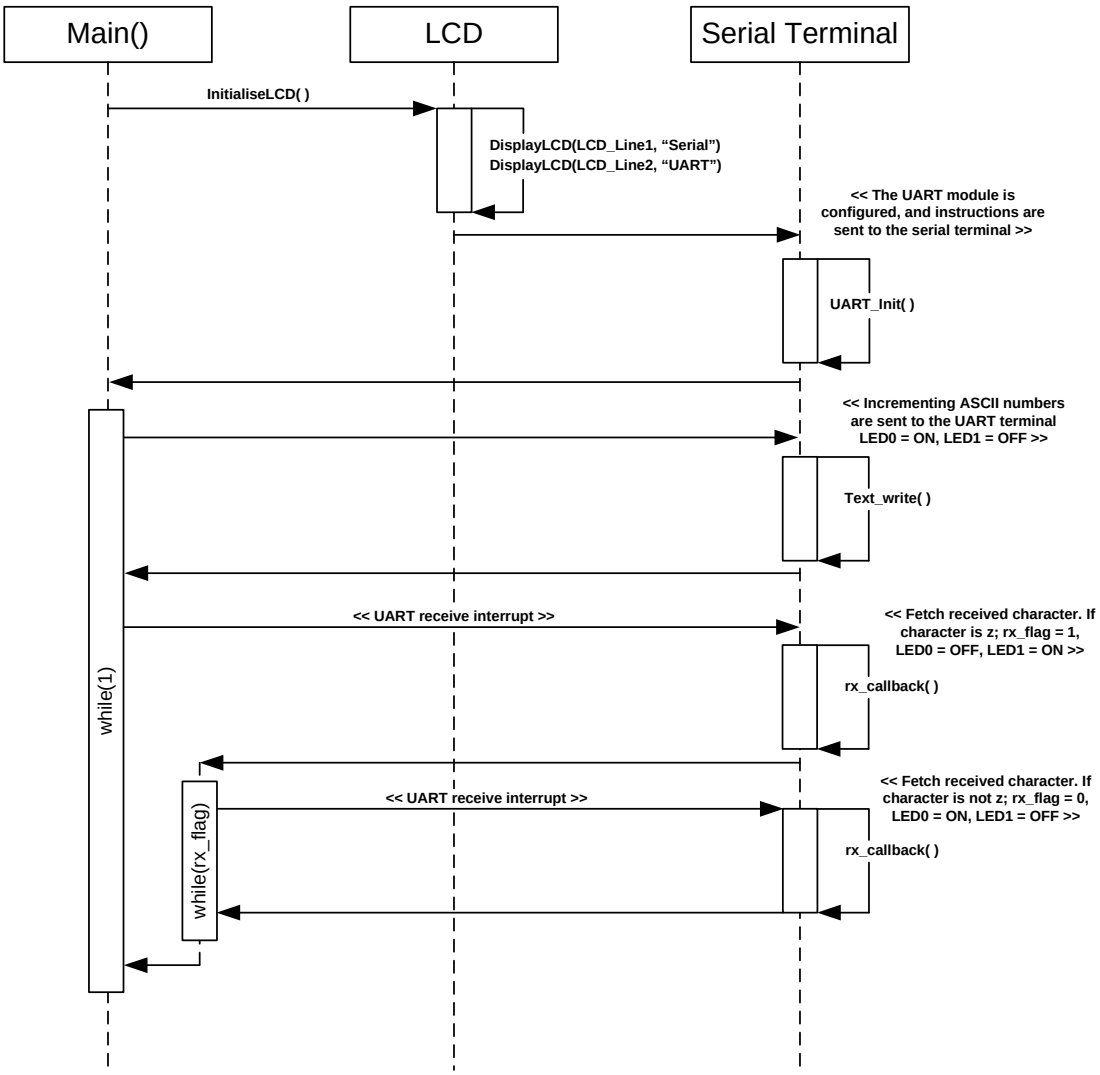

Figure 6 Async_Serial Sequence Diagram

3.1.4.2. Description

1- The Serial_Uart sample code initializes the LCD module and displays “Serial” on Line-1 and “Uart” on Line-2.

2- The function UART_Init configures the UART module for asynchronous transmission, and sends instructions to the serial terminal.

3- The Text_write function is called, and sends incrementing ASCII numbers (0 to 9) serial terminal window repeatedly.

4- When the user presses a key in the serial terminal, the rx_callback function is executed by interrupt. If the inputted key is a ‘z’, the callback function sets the rx_flag to 1.

5- The program waits in a while loop, whilst the rx_flag is 1.

6- When the user presses a key into the serial terminal again; the rx_callback function will set the rx_flag back to 0 if the inputted key is not ‘z’.

7- Once the rx_flag is set back to 0, the program continues to run through the main while(1) loop; and continues to send incrementing ASCII numbers to the serial terminal.

	Function	RPDL API Function
1	Uart_Init( )	R_IO_PORT_Set R_SCI_Create R_SCI_Receive R_SCI_Send
2	Text_write( )	R_SCI_Send R_IO_PORT_Write
3	rx_callback( )	R_SCI_Receive

Table 3-5 RPDL API function list for Async_Serial

3.1.5. CRC

This sample code configures the cyclic redundancy check unit to perform a checksum calculation of an arbitrary user keyboard press.

3.1.5.1. Sequence Diagram

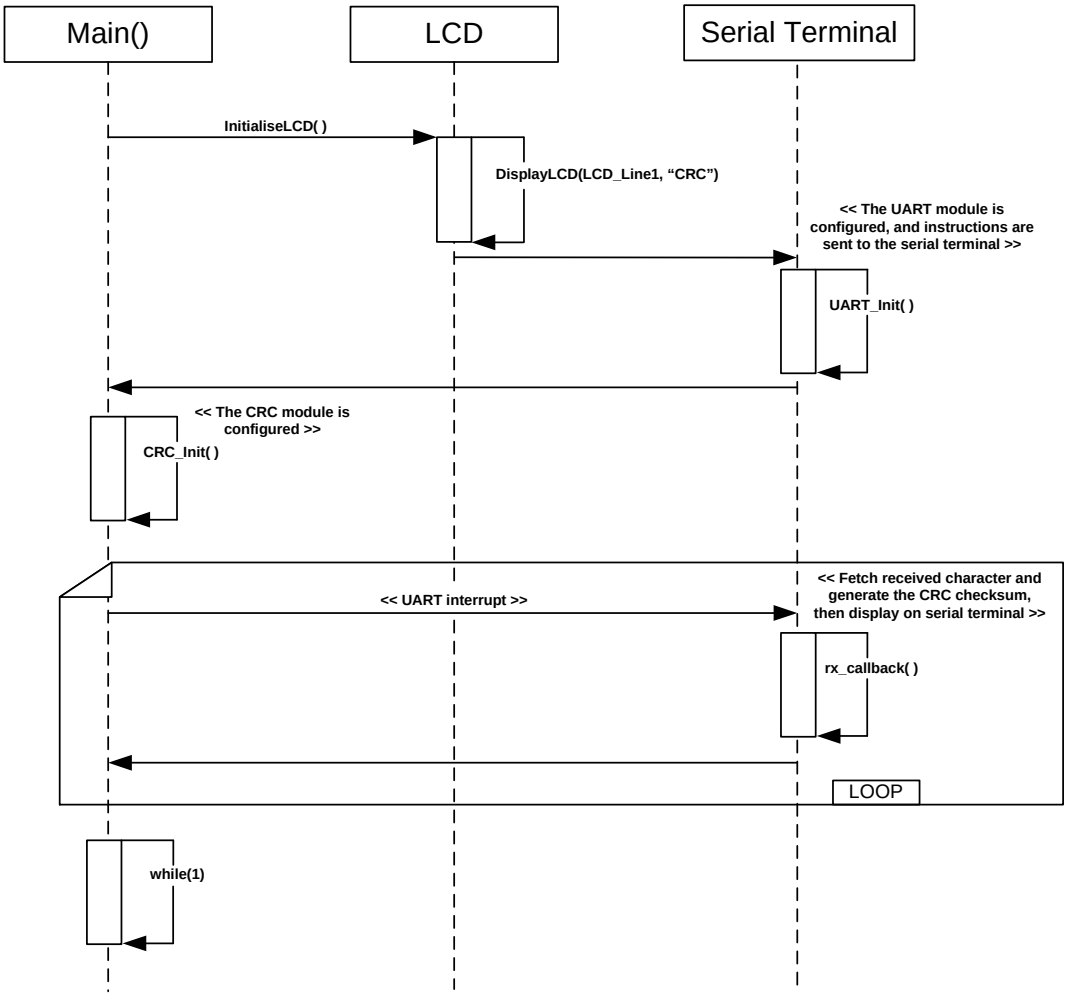

Figure 7 CRC Sequence Diagram

3.1.5.2. Description

1- The UART unit is initialised in the UART_Init function call.

2- Program instructions are sent to the UART terminal.

3- The CRC unit is initialised in the CRC_Init function call.

4- When the user enters a key into the UART terminal, the rx_callback function is called.

5- The rx_callback function generates a checksum value and sends it to the UART terminal

6- The code loops back to step 4, and waits for another user key press.

	Function	RPDL API Function
1	UART_Init( )	R_IO_PORT_Set R_SCI_Create R_SCI_Receive R_SCI_Send
2	CRC_Init( )	R_CRC_Create
3	rx_callback( )	R_CRC_Write R_CRC_Read

Table 3-6 RPDL API function list for CRC

3.1.6. Data_Flash

This sample code configures the MCU to perform Data Flash writes, based on the value attained from the ADC.

3.1.6.1. Sequence Diagram

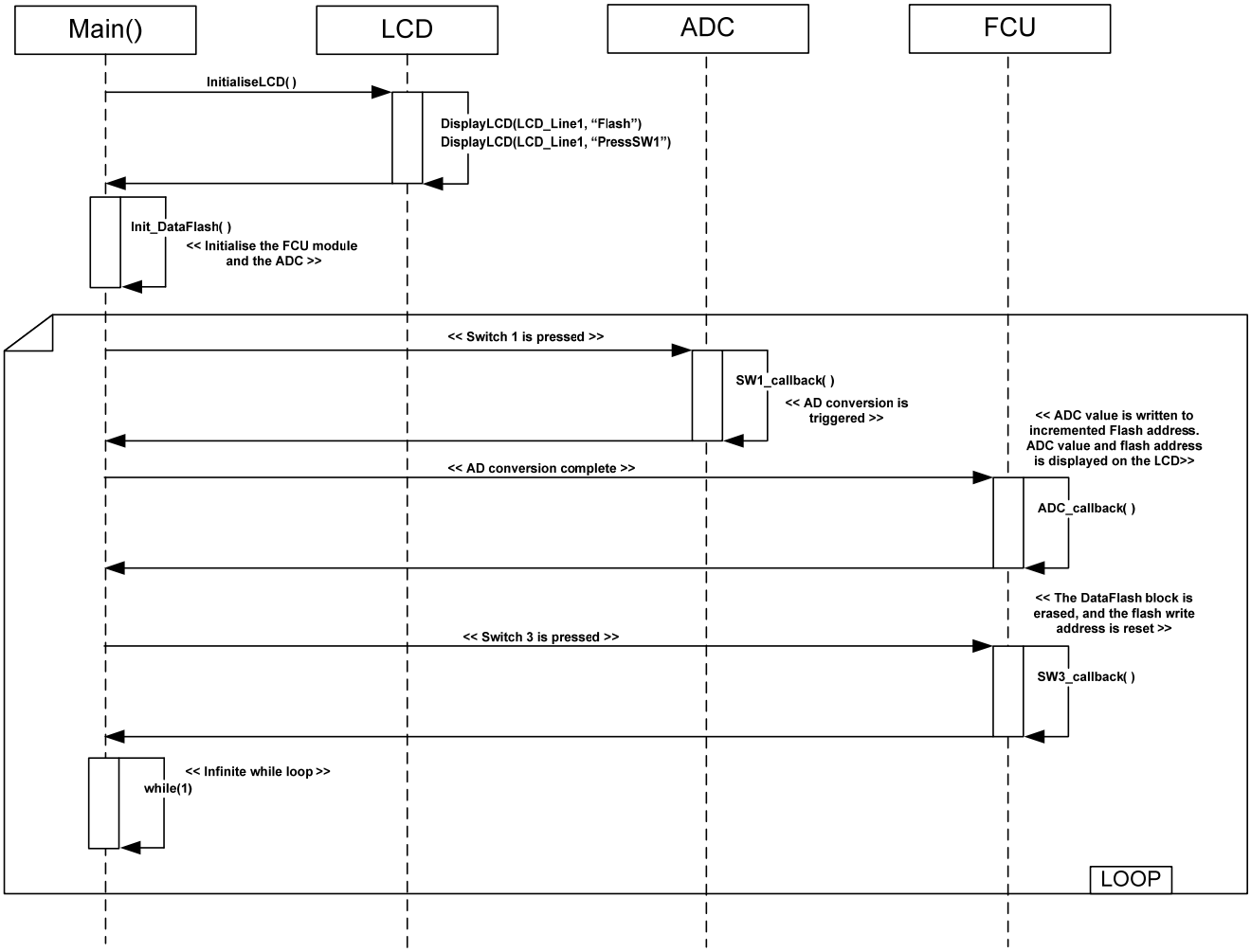

Figure 8 Data_Flash Sequence Diagram

3.1.6.2. Description

1- The FCU and ADC modules are initialised in the Init_DataFlash function call.

2- When the user presses switch 1 the callback function, SW1_callback, is executed.

3- The function, SW1_callback, triggers the AD conversion.

4- When the AD conversion completes, the function ADC_callback is executed.

5- The function, ADC_callback, writes the ADC value to an incremented DataFlash address, and displays both the ADC value and the memory address location on the LCD.

6- Pressing SW3 erases the DataFlash block, and resets the flash address write location.

7- The program returns to an infinite while loop. When the user presses switch 1 again, the process is repeated from step 2.

	Function	RPDL API Function
1	Init_DataFlash	R_ADC_10_Create
2	SW1_callback	R_ADC_10_Control
3	ADC_callback	R_ADC_10_Read

Table 3-7 RPDL API function list for Data_Flash

Note: The Data_Flash sample uses a non-RPDL API to perform flash operations. This middleware API uses the following function naming convention: R_FlashErase instead of the RPDL style: R_CRC_Create, which includes the peripheral group (CRC). Further details of the flash middleware API can be found in the application note, *Flash API Application Note.*

3.1.7. DMAC

This sample code configures the DMAC unit to perform a data transfer to the global variable, gDMA_DataBuff.

3.1.7.1. Sequence Diagram

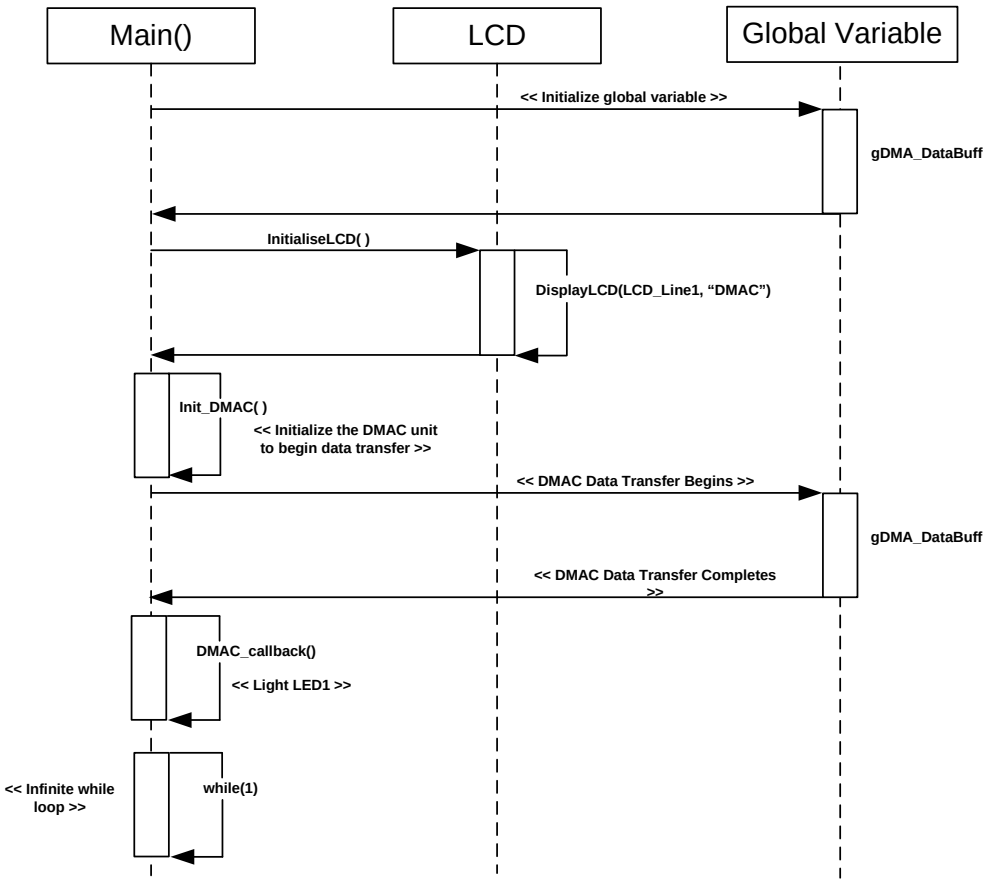

Figure 9 DMAC Sequence Diagram

3.1.7.2. Description

1- The DMAC unit is started within the Init_DMAC function call.

2- The DMA transfer fills the global array gDMA_DataBuff with data from the source array DMA_DataSource.

3- After transfer completion the callback function, DMAC_callback, lights LED1.

4- The contents of the desintation array gDMA_DataBuff can be read as alternating character words of 0xFF and 0xEE.

	Function	RPDL API Function
1	Init_DMAC( )	R_DMAC_Create R_DMAC_Control
2	DMAC_callback	R_IO_PORT_Modify

Table 3-8 RPDL API function list for DMAC

3.1.8. DTC

This sample code demonstrates usage of the DTC module, by performing an AD conversion on the AD pot and transfer the data across to a global variable.

3.1.8.1. Sequence Diagram

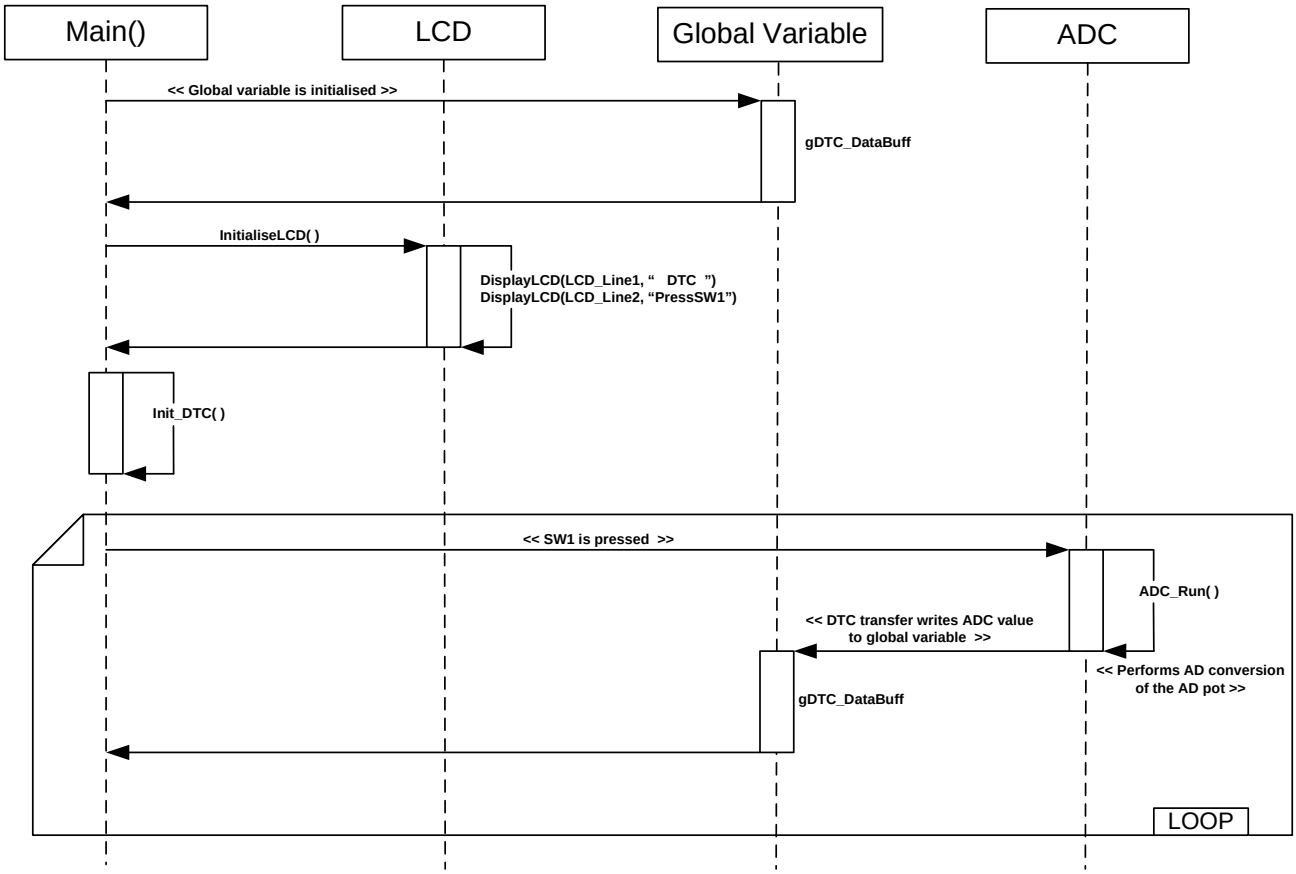

Figure 10 DTC Sequence Diagram

3.1.8.2. Description

1- The global variable, gDTC_DataBuff is initialised, and instructions are displayed onto the LCD.

2- The Init_DTC function executes, and initialises the DTC to trigger after a successful AD conversion.

3- The user presses switch 1, which raises an interrupt which starts the ADC.

4- When the AD conversion completes, the DTC transfer the Ad value to all 1024 elements of the global variable array gDTC_DataBuff.

5- The program waits in an infinite while loop. When a user presses SW1 again, the program loops back to step 3.

	Function	RPDL API Function
1	Init_DTC( )	R_DTC_Set R_DTC_Create R_DTC_Control R_ADC_10_Create
2	ADC_Run( )	R_ADC_10_Control

Table 3-9 RPDL API function list for DTC

3.1.9. Power_Down

This sample code configures MCU to enter flash the user LEDs in normal operation, and to enter sleep mode when switch 1 is pressed. Any switch will then wake the MCU and the user LEDs will recommence flashing.

3.1.9.1. Sequence Diagram

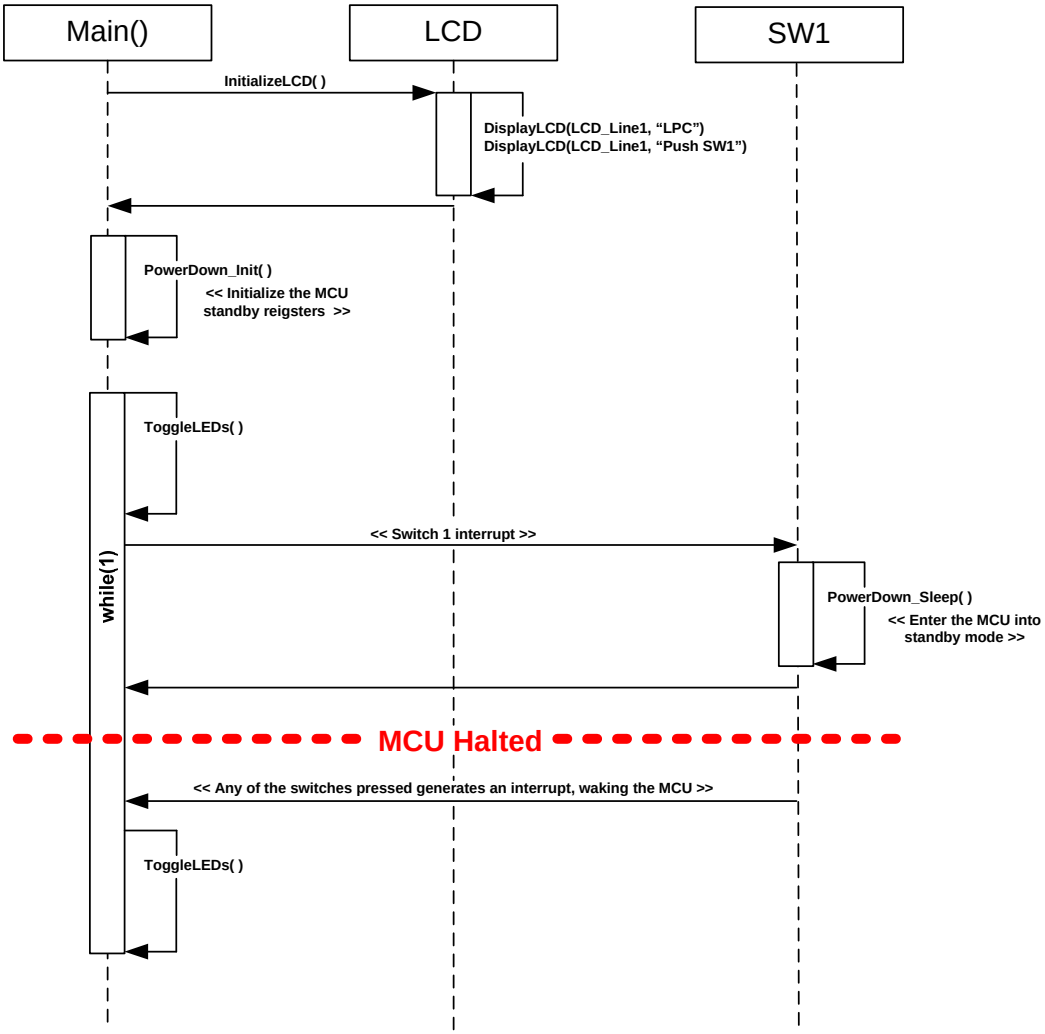

Figure 11 Power_Down Sequence Diagram

3.1.9.2. Description

1- The Power_Down sample code initializes the LCD module and displays “LPC” (Low Power Consumption) on Line-1 and “PressSW1” on Line-2.

2- This application enters an infinite while loop which calls the function ToggleLEDs every iteration, to cause the user LEDs to flash.

3- When the user presses switch 1, the function PowerDown_Sleep is called, which puts the MCU into sleep mode – thus stopping all processes, including the flash LEDs.

4- When a user presses either of the switches again, the MCU will wake from sleep mode, and continue to flash the LEDs.

	Function	RPDL API Function
1	PowerDown_Init( )	R_LPC_Create
2	ToggleLEDs( )	R_IO_PORT_Modify
3	PowerDown_Sleep	R_IO_PORT_Write
		R_LPC_Control

Table 3-10 RPDL API function list for PWM

3.1.10. PWM_Mode

This sample code configures the timer to generate 1KHz waveform, with a constantly changing duty cycle. The duty cycle begins at 10% and gradually increases to 90%, then cycles back to 10% continuously until the user presses switch 1 and freezes the duty cycle.

3.1.10.1. Sequence Diagram

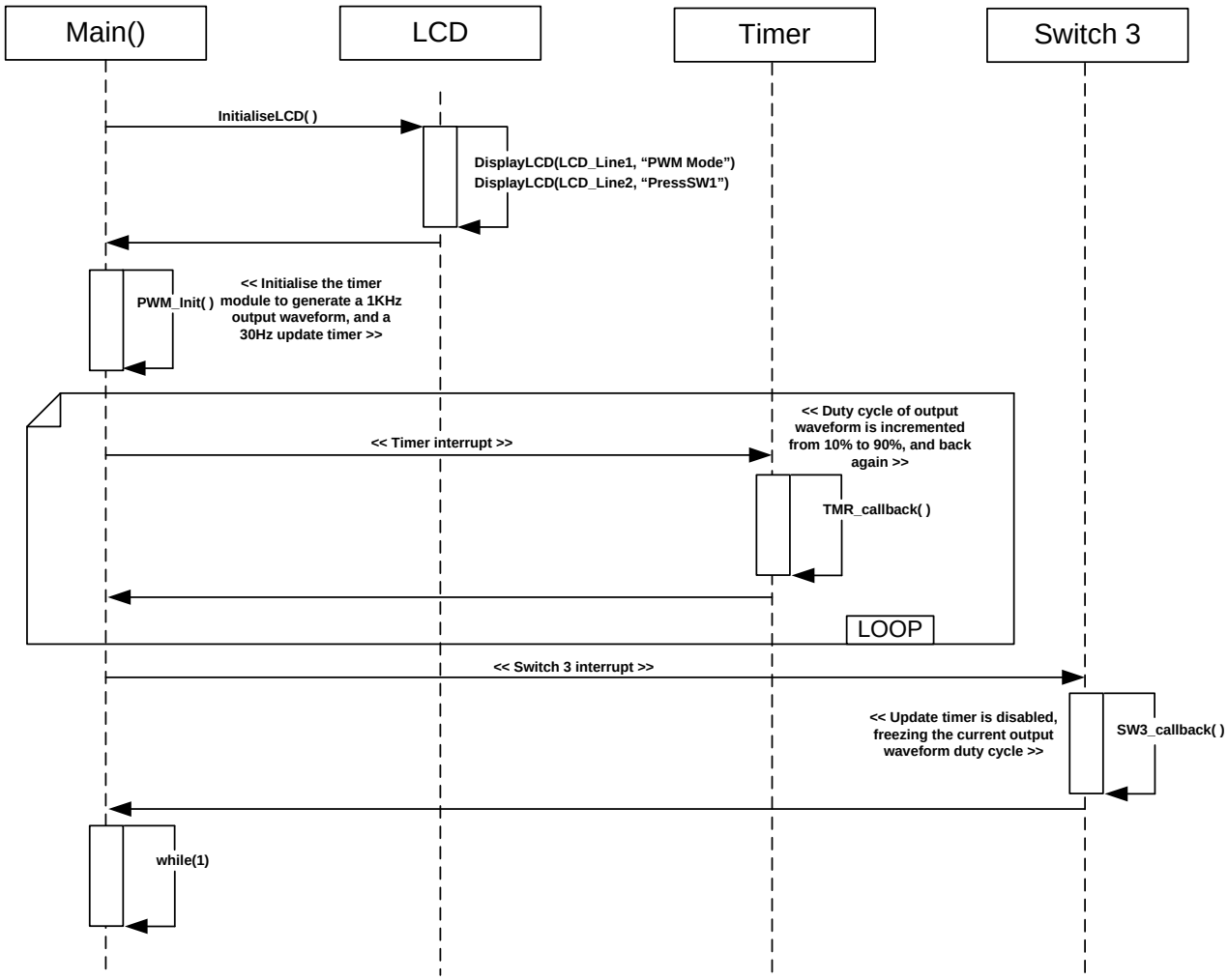

Figure 12 PWM_Mode Sequence Diagram

3.1.10.2. Description

1- The PWM_Mode sample code initializes the LCD module and displays “PWM Mode” on Line-1 and “PressSW1” on Line-2.

2- The function, PWM_Init, configures the timer module to generate a 1KHz output waveform and a 30Hz update waveform.

3- The output waveform can be seen using oscilloscope, connected to pin JA2-20.

4- The update waveform timer generates an interrupt each period, and calls the callback function, TMR_callback. The callback function increments the duty cycle of the output waveform from 10% to 90%, and then back to 10% repeatedly.

5- Pressing SW3 generates an interrupt, which calls the callback function, SW3_callback. This function disables the update waveform, thus freezing the output waveform’s duty cycle at its current value.

	Function	RPDL API Function
1	PWM_Init( )	R_TMR_CreatePeriodic
2	TMR_callback( )	R_TMR_ControlPeriodic
3	SW3_callback( )	R_TMR_ControlChannel

Table 3-11 RPDL API function list for PWM

3.1.11. Sync_Serial

This sample code configures SCI unit to perform a synchronous loop back data transfer. The ASCII string “Renesas” is sent from one SCI unit to another. The SCI receive results are then displayed onto the LCD.

3.1.11.1. Sequence Diagram

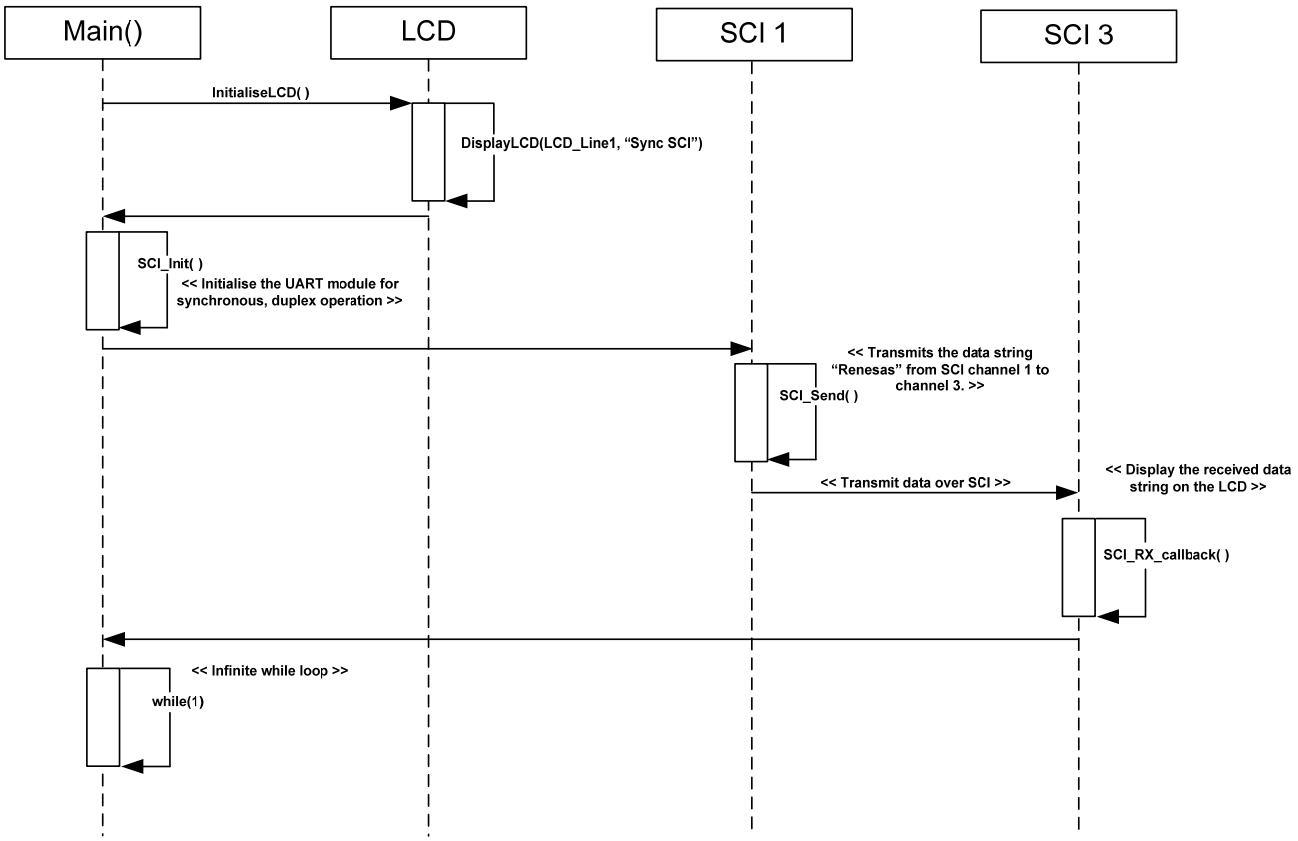

Figure 13 Sync_Serial Sequence Diagram

3.1.11.2. Description

1- The Sync_Serial sample code initializes the LCD module and displays “Sync SCI” on Line-1.

2- The program initializes two SCI channels in synchronous mode, and then transmits the data string ‘Renesas’ across from one to the other.

3- The callback function SCI_RX_callback is executed when the correct number of data bytes has been received by the SCI channel. The function displays the data string onto the LCD.

	Function	RPDL API Function
1	SCI_Init	R_SCI_Create R_SCI_Receive
2	SCI_Send	R_IO_PORT_Modify R_SCI_Send
3	SCI_RX_callback	R_IO_PORT_Modify

Table 3-12 RPDL API function list for PWM

3.1.12. Timer_Capture

This sample code configures the timer to toggle on and off by pressing SW3. The count value is displayed onto the LCD and then reset, each time the timer is toggled off.

3.1.12.1. Sequence Diagram

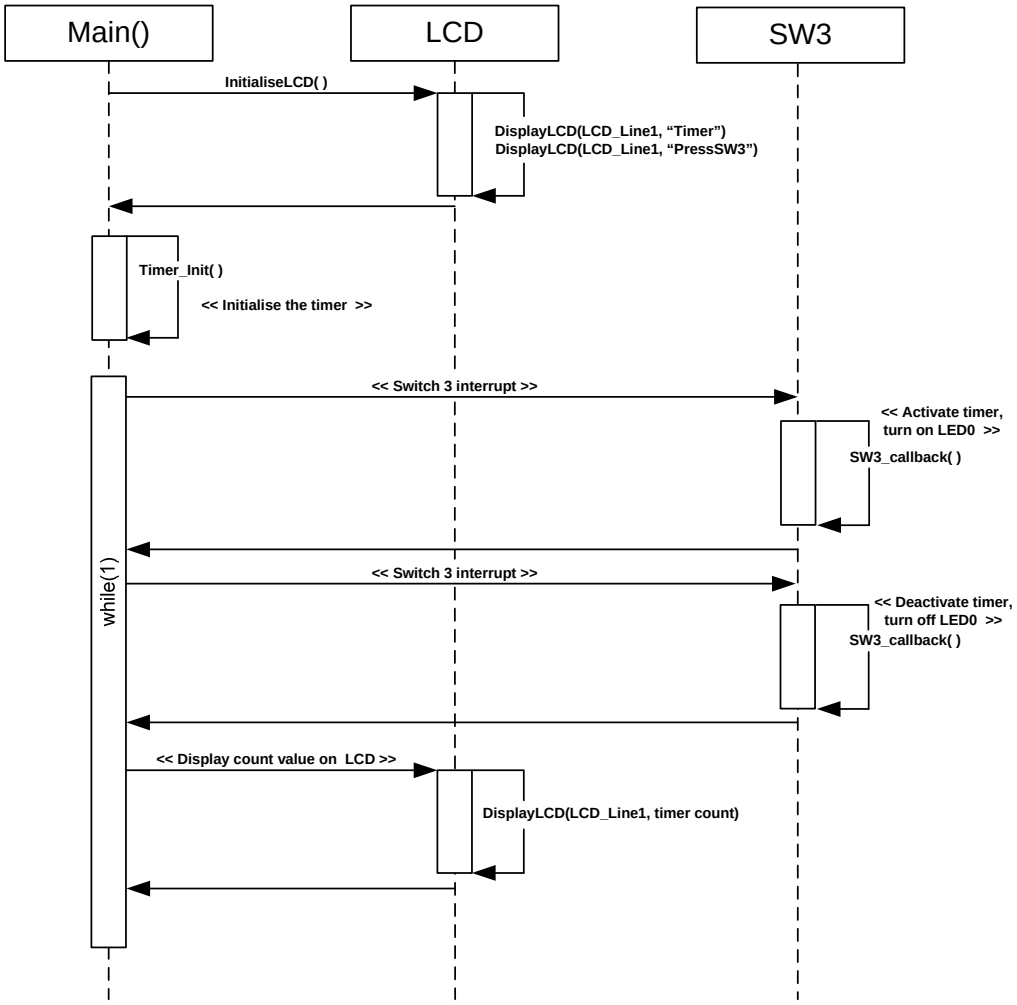

Figure 14 Timer_Capture Sequence Diagram

3.1.12.2. Description

1- The timer module is initialised in the Timer_Init function call.

2- The user presses switch 3, which calls the SW3_callback function.

3- In this instance, the SW3_callback function starts the timer and lights LED0.

4- The user presses switch 3 again, which calls the SW3_callback function.

5- In this instance, the SW3_callback function stops the timer, turns off LED0 and displays the current count value to the LCD.

6- The program loops back to step 2.

	Function	RPDL API Function
1	Timer_Init( )	R_TMR_CreatePeriodic R_TMR_ControlUnit
2	SW3_callback	R_TMR_ReadUnit R_TMR_ControlUnit R_IO_PORT_Modify

Table 3-13 RPDL API function list for PWM

3.1.13. Timer_Compare

This sample code configures the timer to toggle on and off by pressing SW3. The count value is displayed onto the LCD and then reset, each time the timer is toggled off.

3.1.13.1. Sequence Diagram

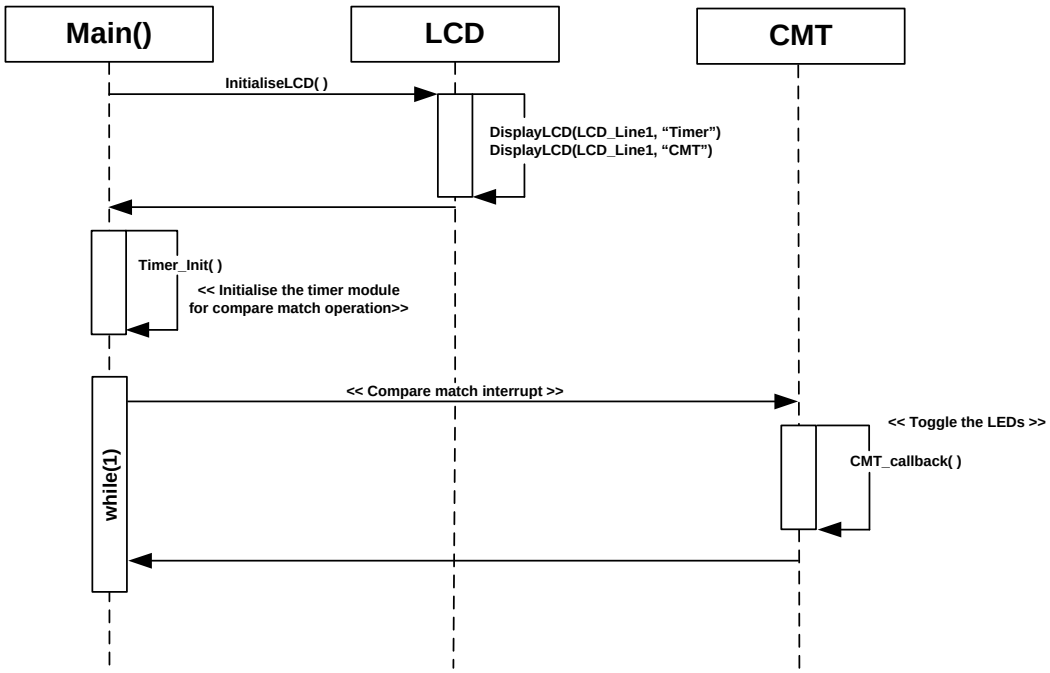

Figure 15 Timer_Compare Sequence Diagram

3.1.13.2. Description

1- The timer module is initialised in the Init_Timer function call.

2- When a compare match occurs, the CMT_callback function is called.

3- The CMT_callback toggles the user LEDs.

	Function	RPDL API Function
1	Timer_Init( )	R_TMR_CreateChannel
2	CMT_callback( )	R_IO_PORT_Modify

Table 3-14 RPDL API function list for Timer_CMT

3.1.14. Timer_Event

This sample code demonstrates the use of the external clock driven timer. The sample increments a count value as the user presses a switch.

3.1.14.1. Sequence Diagram

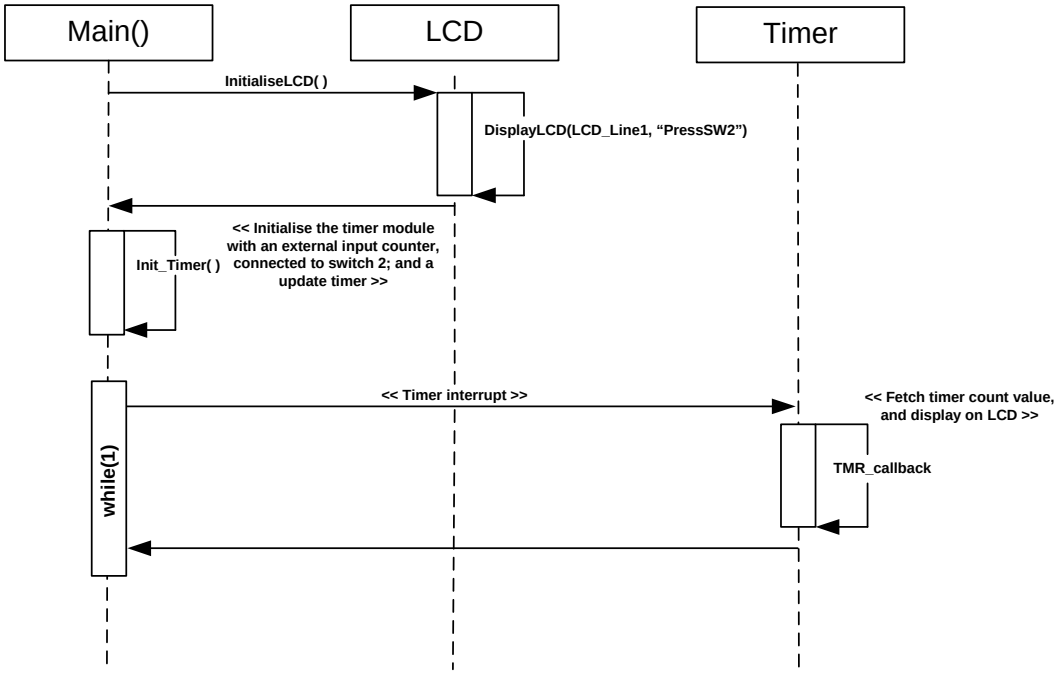

Figure 16 Timer_Event Sequence Diagram

3.1.14.2. Description

1- The timer module is initialised in the Init_Timer function call. The function configures a counter, connected to switch 2; and an update timer.

2- When the period of the update timer elapses, an interrupt is generated. The interrupt exectures the callback function, TMR_callback

3- When a compare match occurs, the CMT_callback function is called.

4- The CMT_callback toggles the user LEDs.

	Function	RPDL API Function
1	Init_Timer( )	R_TMR_CreateChannel R_TMR_CreatePeriodic
2	TMR_callback( )	R_TMR_ReadChannel

Table 3-15 RPDL API function list for Timer_Event

3.1.15. Timer_Mode

This sample code configures the timer to generate 1KHz waveform. The waveform can be seen on the oscilloscope.

3.1.15.1. Sequence Diagram

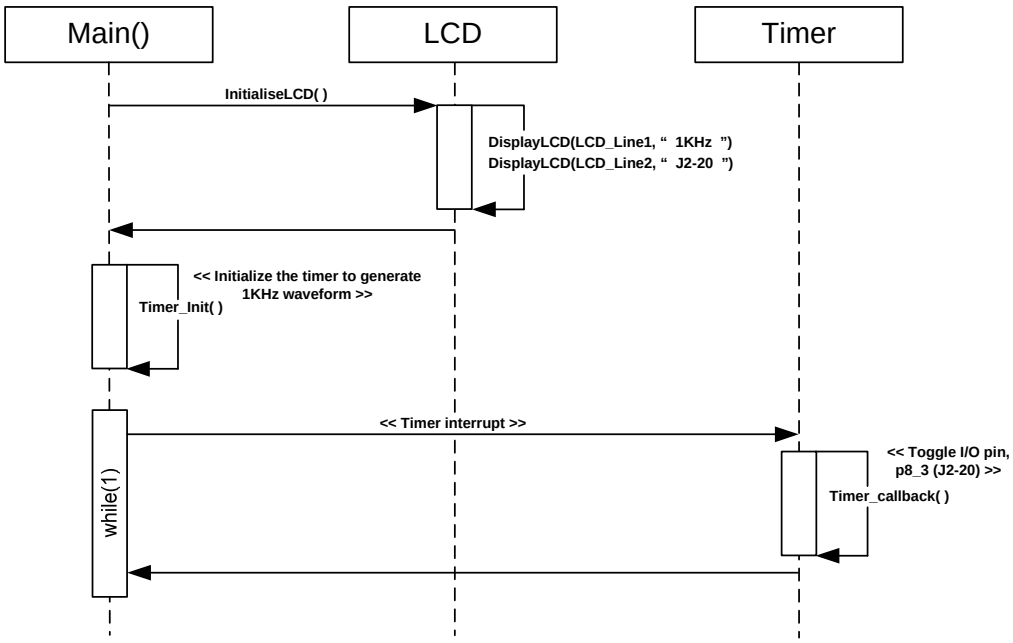

Figure 17 Timer_Mode Sequence Diagram

3.1.15.2. Description

1- The Timer_Mode sample code initializes the LCD module and displays “1KHz” on Line-1 and “J2-20” on Line-2.

2- The function, Timer_Init, initialises the timer module to produce a 1KHz interrupt.

3- When the timer period elapses, a timer interrupt is generated. The interrupt calls the callback function, Timer_callback. This function toggles the I/O port P8_3, accessible via J2-20.

4- The waveform can be seen using oscilloscope, attached to J2-20.

	Function	RPDL API Function
1	Timer_Init( )	R_TMR_CreatePeriodic
2	Timer_callback( )	R_IO_PORT_Modify

Table 3-16 RPDL API function list for DMAC

3.1.16. WDT

This sample code demonstrates the usage of the watchdog timer functionality; by resetting the WDT timer count regularly, at a rate controlled by the potentiometer. When the timer is too slow, the WDT times out and the LEDs stop flashing.

3.1.16.1. Sequence Diagram

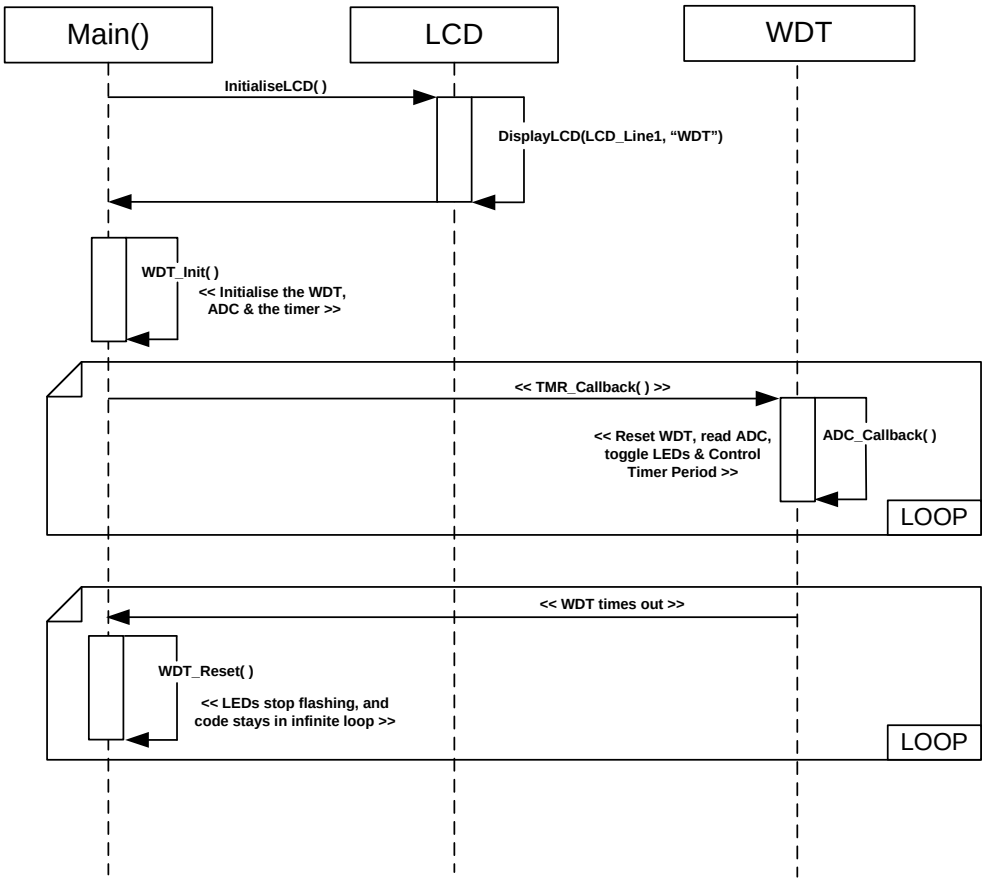

Figure 18 WDT Sequence Diagram

3.1.16.2. Description

1- The ADC, timer and WDT are initialised in the WDT_Init function call.

2- The timer exectutes the callback function, TMR_Callback every period.

3- The callback function starts an AD conversion, which calls the callback function ADC_Callback when complete.

4- The ADC_Callback function reads the value of the ADC, and varies the timer frequency with it. The function also toggles the LEDs and resets the count value of the WDT.

5- Steps 2 to 4 repeat as long as the timer frequency is high enough to prevent the WDT from timing out.

6- When the WDT times out, the callback function WDT_Reset is called.

7- The WDT_Reset function keeps the LEDs light solidly, and stays in an infinite loop.

	Function	RPDL API Function
1	WDT_Init( )	R_TMR_CreateUnit R_WDT_Create R_ADC_10_Create
2	TMR_Callback( )	R_ADC_10_Control
3	ADC_Callback( )	R_WDT_Control R_ADC_10_Read R_IO_PORT_Modify R_TMR_ControlUnit
4	WDT_Rest( )	R_IO_PORT_Modify

Table 3-17 RPDL API function list for WDT

Chapter 4. Additional Information

For details on how to use High-performance Embedded Workshop (HEW), refer to the HEW manual available on the CD or installed in the Manual Navigator.

For information about the RX610 microcontrollers refer to the *RX610 Group Hardware Manual*

For information about the RX610 assembly language, refer to the *RX610 Software Manual*

For information about the E1 Emulator, please refer to the *RX Family E1/E20 Emulator User's Manual*

Online technical support and information is available at: www.renesas.com/renesas_starter_kits

Technical Contact Details

America: techsupport.rta@renesas.com

Europe: tools.support.eu@renesas.com

Japan: csc@renesas.com

General information on Renesas Microcontrollers can be found on the Renesas website at: www.renesas.com

Revision History	Renesas Starter Kit for RX610 Software Manual

Rev.	Date	Description	
		Page/Section	Summary
1.0	10.03.2010	ALL	First Issue
1.01	19.03.2010	ALL	QA Fixes

Renesas Starter Kit

Software Manual

Publication Date Rev.1.01 19.MAR.2010

Published y: Renesas Technology Europe Ltd.
Duke's Meadow, Millboard Road, Bourne End
Buckinghamshire SL8 5FH, United Kingdom

Renesas Starter Kit for RSKRX610
Software Manual

Renesas Electronics Corporation
1753, Shimonumabe, Nakahara-ku, Kawasaki-shi, Kanagawa 211-8668 Japan

REJ10J2017-0101