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.

RENESAS

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.

SH2A Family

High Speed Analog Data Collection using MTU2, ADC and DMAC

Introduction

The purpose of this application note is to show how to connect multiple peripherals in an application to "off-load" the simple work of periodically triggering the ADC and collecting the data into a buffer.

In many applications it is necessary to collect Analog data from Sensors in real-time and process using complex filtering. Since many filters require multiple samples before any reasonable output can be expected, it is desirable to avoid all of the overhead of having interrupts and the CPU handling this data. It is preferred to handle one interrupt when the desired sample buffer size is ready. This Application note shows how to use 3 peripherals (MTU2, ADC and DMAC) in concert to collect two channels of analog data simultaneously and store in a memory buffer at rates up to ~300 kHz.

Target Device

SH7286 (R5F72867FPV)

NOTE: This application note can be applied to other SH2/SH2A devices with minor changes. Basically, it will apply to any SH2/SH2A device with the MTU2, 12 Bit ADC and DMAC, for example the SH7137.

Contents

1. Demo Requirements

The following items are required in order to build and run this demonstration.

1.1 Required Renesas Development Tools (software)

The following tools and their versions are required for building the demonstration and following the tutorial. Evaluation editions of these tools are all available for download from our website.

- High-performance Embedded Workshop (HEW) Version 4.05.xx.xx (or later)
  http://www.renesas.com/download

  Web Search Keyword: "High-performance Embedded Workshop"

- Renesas SuperH RISC Engine Standard Toolchain Version 9.3. Release 00 (or later)
  http://www.renesas.com/download

  Web Search Keyword: "SuperH Compiler"

1.2 Required Hardware for Demo

The following items are needed for the demo below.

- RSK for SH7286 (includes E10A for Starter Kits).
  http://www.renesas.com/rsk

- 2 Analog Signal Sources (i.e. Signal Generators, etc…)

1.3 Demonstration Software

The Demonstration software is intended for use with the Renesas RSK board. You will also need two (2) analog signal sources to generate data to be sampled. More detailed documentation on the RSK hardware is included with the kit or can be downloaded from the Renesas Web site.

2. Building the Code

The demonstration software can be built by opening the HEW Workspace (hw_adc_sample.hws) and building the code using the "**Build All**" icon . The resulting 'hw_adc_sample.abs' file in the 'Debug' directory can then be downloaded using the E10A and HEW.

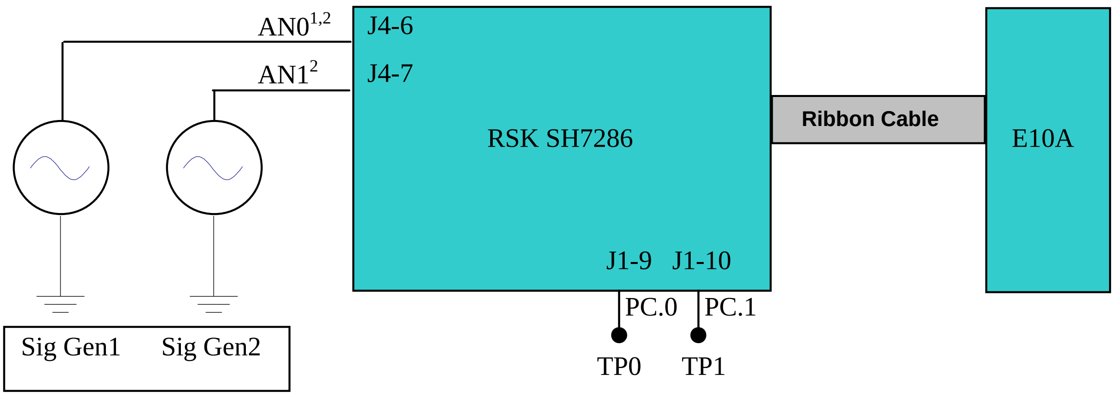
3. Hardware Set-up Description

The demonstration is setup to collect data on ADC Unit 0, channels 0 and 1. The Hardware should be setup as shown in figure 3.1. There are some limitations to the RSK as delivered from the factory. The considerations for this demonstration are:

- AN0 is connected to the potentiometer
- All AN channels have 3K/.01µF filters for protection.

The user should refer to the RSK schematic and change these according to the use of channel 0 and the frequencies of the signals you desire to sample.

CAUTION: Your analog sources should not violate the input specifications of the SH7286 ADC channels found in the Hardware Manual. For example maximum V_{AN} is listed as -0.3 to AVCC+0.3. See Absolute Maximum Ratings and ADC characteristics section of SH7286 Hardware manual.



Notes: 1. To use AN0 you should disconnect it from potentiometer by removing 0Ω resistor R87

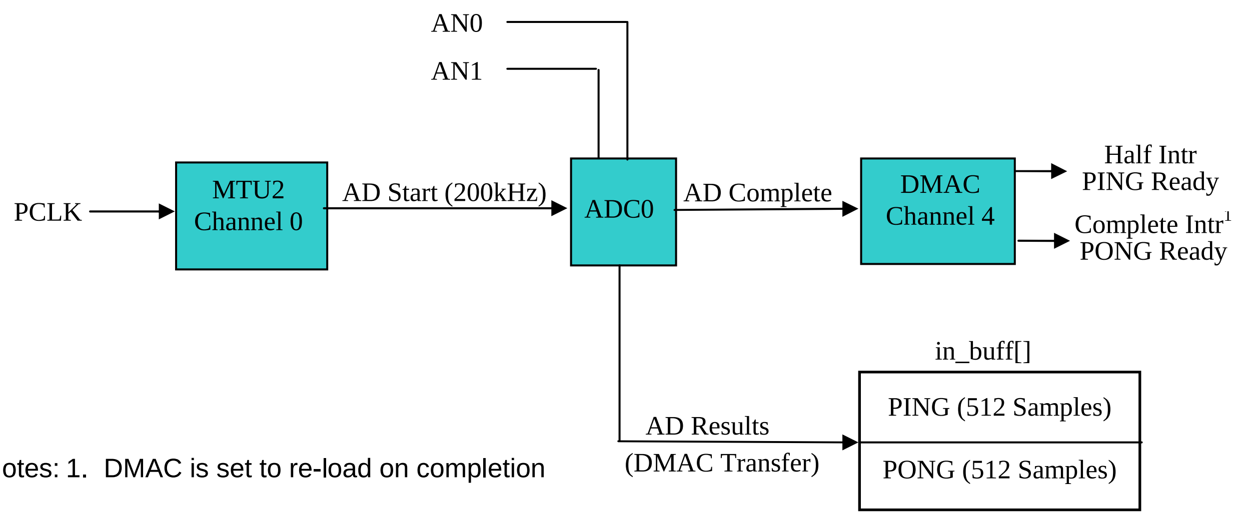
2. Do not exceed the input specification of these ADC channels
To sample higher frequencies you may need to change input filters of the RSK

Figure 3.1 Hardware Setup Block Diagram

4. Software Demo Description

The following sections describe the Software's operation. The demo does NOT do anything with the collected data, but rather the user can insert code in main() where the buffers are signaled as ready.

Please refer to Figure 4.1 for details on the peripheral signaling flow.



Notes: 1. DMAC is set to re-load on completion

Figure 4.1 Peripheral Signaling Flow

4.1 PING/PONG Buffer Usage

Many applications utilize complex filters to extract the required information from the Analog signals. These filters typically take multiple input samples to get usable outputs. In addition, these filters may take some extensive compute time. In this demonstration, we utilize a PING/PONG buffer scheme. We collect data into the PING buffer until it reaches the desired sample size. The software signals that the PING buffer is ready and continues to collect the signals in the PONG buffer. This gives the software enough samples and processing time to operate (i.e. run the filters) on the PING buffer while the hardware continues to collect data in the PONG buffer. Compute time allowed is then:

Sample Rate Period * MIN_SAMPLE_SIZE before the next buffer is ready.

See section 5.3 Buffer Size Considerations for the calculations for this demo.

4.2 MTU2 Setup

The MTU2 is setup to create a periodic rate to the ADC Unit 0 to start a conversion. The sampling rate is set in MTU2.TGRA (compare A). This compare bit is used because it can trigger ADC start.

The rate is based off the definition for sampling frequency in the common_def.h file. These are based on MACROs which define the CPU Clock, I clock, B clock and P clock (see the SH7286 hardware manual for details on the clock structure). The maximum sample rate should not exceed the limits of the microcontroller. These limits are based on the various clock rates in the system, the number of channels converted, etc…. . For this demo, the SH7286 is running at maximum clock on ADC and converting two channels, so the limits are:

~251kHz with ADBYPSCR.SH=1 and ADBYPSCR.OFC=0 (S/H, offset correction ON)

```
#define     SAMPLING_FREQ 200000L   // 200.0 kHz
```

NOTE: If different crystal is installed the user will need to change the macros in common_def.h that sets those numbers. The PCLK driving the MTU2 basically is sourced by the CPU clock through some divider chains, so they are related.

4.3 ADC Setup

This demo uses ADC controller 0. This converter was chosen because it has the ability to trigger the DMAC. It is set for a single scan with Channels 0 and 1 selected to convert. The individual Sample and Holds are enabled so Channels 0 and 1 are sampled simultaneously (i.e. time coherent). The Offset correction is turned on, so there are extra clocks required for the Offset correction and thus limits the maximum rate (see MTU2 setup section above).

The ADC clock is programmed to maximum, 50MHz in the case of the default RSK for SH7286

4.4 DMAC Setup

We chose to use channel 4 of the DMAC controller. Channel 4 is the "subset" channel (i.e. channel 0 - 3 have some minor additional features that were not required when transferring data from ADC to internal memory).

NOTE: Although any DMAC channel can be used to do this ADC transfer, the designer must evaluate the system requirements when choosing channels to use. For example if you need to move data externally using single cycle transfers, these are only supported by channels 0-3. So it can be seen these should be allocated first.

The DMA is setup to operate as follows:

- Triggered by ADC converter 0 complete
- One longword transfer per request
- 1024 total transfers (MIN_SAMPLE_SIZE*2)
- Interrupt when Half complete
- Interrupt when complete
- Auto reload on completion of count

5. Demo Debug/Runtime Hints, Tips and Tricks

The following section gives some tips on using this demo application. .

5.1 STOPPING Data Collection in Break

As pointed out in earlier in this document, the data collection does not stop until told to do so by the processor. The easiest way to stop data collection is to just shutdown MTU2 Channel 0. The demo software provides a couple of methods to do this.

It can be stopped by a function call to stop_data_collection().

It can be stopped at buffer boundaries by placing an eventpoint or breakpoint at the nop() at line 180 in main(). The user adds stop_now to the watch window and sets it for automatic update (R). The user then sets the stop_now variable to a 1 and the program will break at the NOP and will be stopped on "buffer boundary" for viewing a full buffer.

5.2 Viewing Collected Data

The collected data can be viewed in a number of ways, but the two methods that will give the user a feel for "what does the data look like?" are by Viewing Graphic or dumping the memory buffer into an Excel file and charting.

NOTE: The following data has been normalized from 12 Bit ADC to the range 0 to 32768 for easy viewing in the Waveform viewer. Done by code in main() at lines 176 and 177.

```
in_buff[i].sample1 *= 8;      // so we can see on graph
in_buff[i].sample2 *= 8;      // so we can see on graph
```

5.2.1 Viewing Graphic

Using Menus: View → Graphic → Waveform

Fill in Dialog box as shown in figure 5.1.

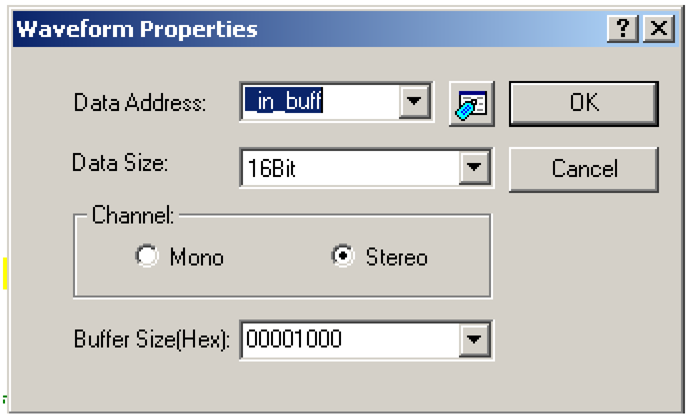


Figure 5.1: Waveform Dialog Box

Figure 5.2 shows the data of two different frequency sine waves. In this case we sampled a 5kHz and a 3kHz sine wave at the input of the ADC channels.

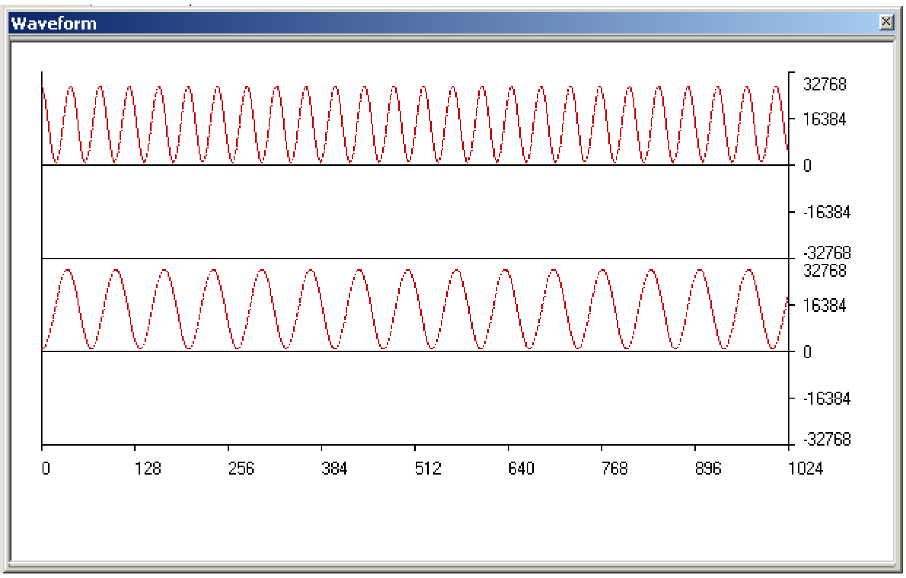


Figure 5.2: Waveform Window

5.2.2 Viewing Using Excel

Using Menus: View ➔ CPU ➔ Memory

Set the Memory Dialog as Shown in Figure 5.3.

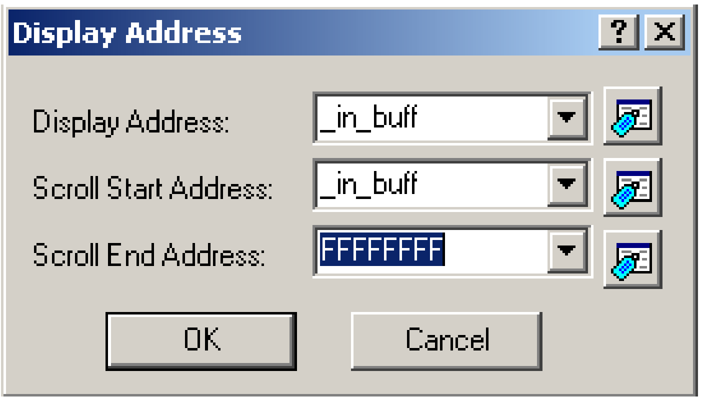


Figure 5.3: Memory Window Dialog

Using right clicks in the memory window set it as follows:

- Data Length – 2 bytes
- Radix – Decimal
- Column – 2
- Code – 16 bit fixed

Right Click the memory window and “Save the Memory Contents” to a file setting the dialog box as shown in figure 5.4.

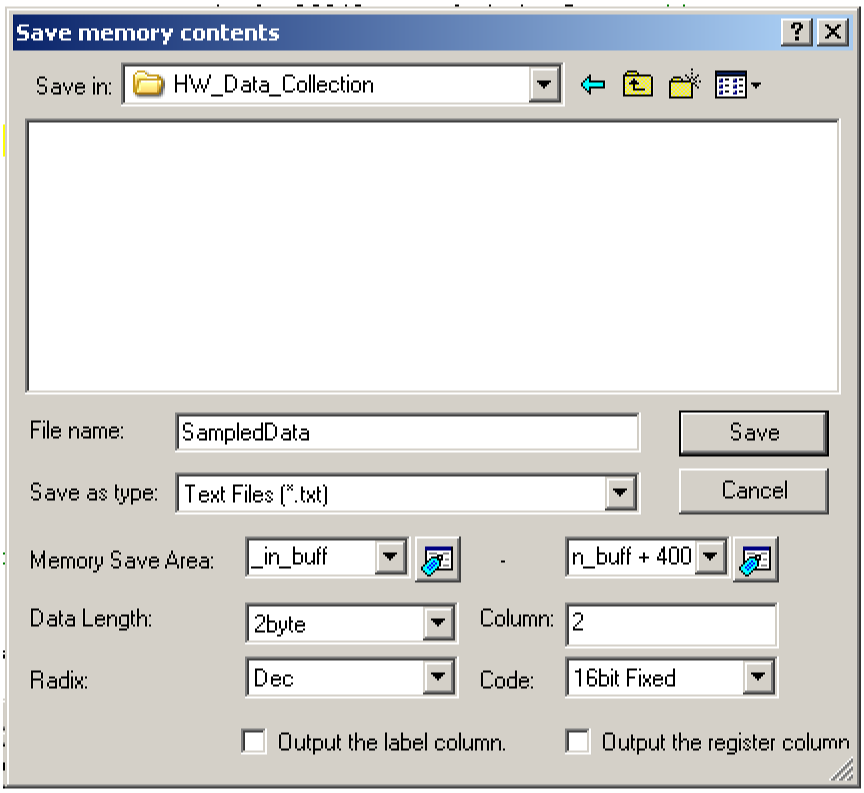


Figure 5.4: Save Memory Dialog

NOTE: Memory range to save is _in_buff to _in_buff + 400.

You can now import the text file into Excel and plot either columns 1 and 2 or 3 and 4. The resulting plot will look something like Figure 5.5 for the two sine waves captured in this demo.

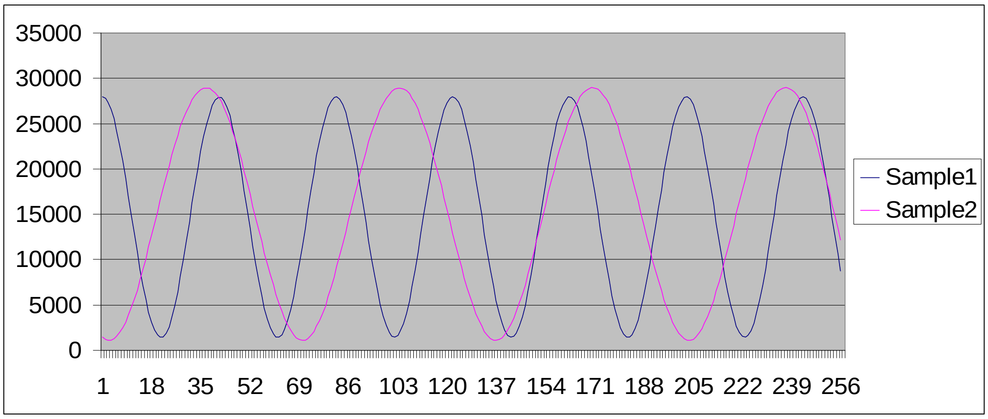


Figure 5.5: Sample Excel Chart

5.3 Buffer Size Considerations

The data rate at which you collect data along with the size of the buffer will affect the processing time you have in between the signaling of a "buffer ready" status. An example of this, is the rate and buffer size used in this demo software.

Example: MIN_SAMPLE_SIZE= 512 samples collected at a rate of 200kHz. Therefore you will get a BUFFER Ready status (PING or PONG) every (1/200kHz)*512 = 5µS*512 = 2.56mS. So you have 2.56mS to act on the 512 samples of data before the next buffer is ready.

So it is important to validate that whatever you need to do with the data buffer (Filters, normalization, etc…) you have sufficient compute time to avoid the data collection buffer being overwritten by the incoming data (i.e. overwriting of the collected data by the hardware before the software has had time to "digest" it).

5.4 Buffer Usage Considerations

The data that has been collected obviously must be processed. We have indicated the two locations where you can add this data processing to the demo application. This data processing may be as simple as converting from raw ADC values to scaled values representing motor current, voltage, etc… to complex filtering. The points are indicated by the following comments in the code in main.c:

```
//****************** Insert User PING buffer operations here ******************************//
//****************** or signal other task that buffer is ready ****************************//
```

Or for the PONG buffer

```
//****************** Insert User PONG buffer operations here ******************************//
//****************** or signal other task that buffer is ready ****************************//
```

Since the hardware data collection does not stop, care should be taken that the data processing does not exceed the time calculated in section 6.3

5.5 Test Point Usage

The Demo software provides two GPIO which can be used as test points for triggering or measurement points. These are declared (along with macros for setting and clearing) in the file custom_bsp.h as follows:

```
#define TRIGGER_PORT          PC.DR.BIT.B0          // PC0 data bit
#define TRIGGER_PORT_PFC      PFC.PCCRL1.BIT.PC0MD  // Pin Function
#define TRIGGER_PORT_DIR      PFC.PCIORL.BIT.B0     // Pin direction (IN/OUT)

#define TRIGGER_PORT1         PC.DR.BIT.B1          // PC0 data bit
#define TRIGGER_PORT1_PFC     PFC.PCCRL1.BIT.PC1MD  // Pin Function
#define TRIGGER_PORT1_DIR     PFC.PCIORL.BIT.B1     // Pin direction (IN/OUT)

#define SET_TRIGGER_PORT(ON_OR_OFF)   (TRIGGER_PORT = ON_OR_OFF)
#define SET_TRIGGER_PORT1(ON_OR_OFF)  (TRIGGER_PORT1 = ON_OR_OFF)
```

The demo code as shipped is using the first test point, TRIGGER_PORT to indicate when PING buffer is ready (0 → 1 transition) and when PONG buffer is ready (1 → 0 transition). The trigger port is set and cleared at lines 160 and 186 respectively. Scope shot shown in figure 5.6.

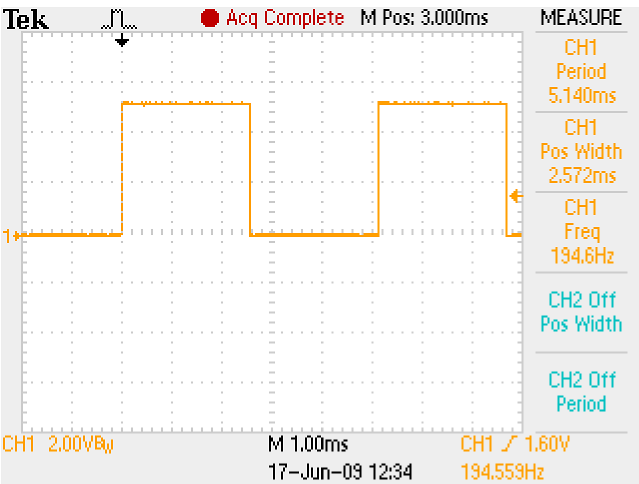


Figure 5.6: Trigger port 1

6. Limitations of Testing

This demo software combines the use of multiple peripherals and thus contains example code for these peripherals.

6.1.1 DMAC Driver

The demo code for the application note contains a mini-DMA Driver. It supports the programming of multiple channels on the DMAC, but all permutations have not been tested. It has only been used to the extent required by this application note, which is DMAC Channel 4 triggered by ADC0 Complete, Half-interrupt enables, Transfer Complete Interrupt enable and Auto-reload.

6.1.2 ADC Driver

The demo code for the application note contains a mini-ADC Driver. It supports the programming of multiple converters and channels on the ADC, but all permutations have not been tested. It has only been used to the extent required by this application note, that is ADC0, channels 0 and 1 enabled for conversion, single sweep, and AD Start triggered by MTU2 channel 0.

6.1.3 MTU2 Channel 0 Usage

MTU2 channel 0 was chosen to trigger the ADC, but any of the MTU2 or MTU2S channels that can create an AD_START can be used. In addition, you can save the timer and use /ADTRG (ADC trigger input) to start the process if you have some fixed rate hardware signal related to the analog sampling.

6.1.4 RSK Code

Much of the demo is based on RSK code and as such should be very familiar to those who have built tutorials on the RSK. We have not endeavored to strip any of this so some functions are available to enhance this demo code (i.e. switch and LCD functions), but are not extensively used.

CMT0/LED1: CMT0 is programmed for 500mS rate and its interrupt service routine toggles LED1. This is used to indicate that the program is running.

LCD: The LCD is initialized as in RSK tutorials and a SPLASH screen is displayed before the Data collection is actually running. Beyond that, the LCD may be used to post other messages if desired.

6.1.5 Reducing / Expanding the concept

This demo is written to fulfill a very specific data collection concept, two analog channels at a high rate and then post processed by some numerical function. To reduce this should be a very simple process (not undertaken here). Simply reduce the sample buffer to one entry (sample1) of MIN_SAMPLE_SIZE, set the DMAC to transfer one word instead of long word.

To expand the concept is just a little more difficult. This dual channel data collection works because we use some tricks in the process. We set the DMAC to do one transfer in response to a request from the ADC, we set the transfer size to a longword. When the DMAC does the single transfer it is broken into two words internally and since ADC result register 0 is adjacent to ADC result register 1, we get both channel 0 and channel 1 results transferred to memory (in_buff.sample1 and in_buff.sample2 respectively).

The Concept can be extended by using the DTC to perform a more complex transfer from the ADC result registers to the memory buffer. In addition, the DTC can be triggered by ADC1 so this concept could be extended to the full number of ADC channels in the microcontroller. Of course the total data collection rate will be based on the ADC Clock, the number of channels and the size of the data buffer. This exercise is left to the reader.

7. References

SH7280 Group Hardware Manual, REJ09B0393-0100

Renesas Start Kit 2 for SH7286 Documents

- User's Manual, REG10J0147-0100
- Schematic, REG99J0037 Rev 2
- Quick Start Guide

SH7280 Group Application Notes

- Data Transfer to On-chip Peripheral Modules with DMAC, REJ06B0778-0100 Rev 1.00
- DMAC Dual Address Mode, REJ06B0767-0100 Rev 1.00

SH7137 Group Application Notes

- 12-Bit A/D Converter: Example of Settings for Conversion in Single-Scan Mode, REJ06B0746-0100 Rev 1.00
- 12-Bit A/D Converter: Example of Settings for Conversion in Continuous-Scan Mode, REJ06B0747-0100 Rev 1.00

8. Glossary

ADC – Analog to Digital Converter

CMT – Compare Match Timer

CPU – Central Processing Unit

DMAC – Direct Memory Access Controller

DTC – Data Transfer Controller

GPIO – General Purpose Input/Output

LCD – Liquid Crystal Display

MTU2 – Multifunction Timer Unit 2

PCLK – Peripheral Clock

RSK - Renesas Starter Kit

Website and Support

Renesas Technology Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/inquiry
csc@renesas.com

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Sep.29.09	—	First edition issued