RX Family

DAC Module Using Firmware Integration Technology

R01AN1818EJ0300
Rev. 3.00
Feb. 28, 2017

Introduction

This module supports the DAC peripheral on the RX111, RX113, RX130, RX210, RX230, RX231, RX23T, RX24T, RX24U, RX63N, RX631, RX64M, RX65N, RX651 and RX71M. The API is identical for the 8-, 10-, and 12-bit converters. Channels are operated individually and all hardware features are supported.

Target Devices

The following is a list of devices that are currently supported by this API:

- **RX111, RX113 Groups**
- **RX130 Group**
- **RX210 Group**
- **RX230, RX231 Groups**
- **RX23T Group**
- **RX24T Group**
- **RX24U Group**
- **RX631, RX63N Groups**
- **RX64M Group**
- **RX651, RX65N Groups**
- **RX71M Group**

When using this application note with other Renesas MCUs, careful evaluation is recommended after making modifications to comply with the alternate MCU.

Related Documents

- Firmware Integration Technology User's Manual (R01AN1833)
- Board Support Package Firmware Integration Technology Module (R01AN1685)
- Adding Firmware Integration Technology Modules to Projects (R01AN1723)
- Adding Firmware Integration Technology Modules to CS+ Projects (R01AN1826)

Contents

1. Overview

This DAC driver supports the DAC peripheral on the RX111, RX113, RX130, RX210, RX230, RX231, RX23T, RX24T, RX24U, RX63N, RX631, RX64M, RX65N, RX651 and RX71M. The hardware functionality is detailed in the D/A Converter chapter in the User's Manual: Hardware for each MCU.

Data to convert to analog may be either left or right justified, and channels can be output independently. MCU-specific hardware features are also supported. This includes selecting the reference voltage, synchronizing conversions with the ADC peripheral, disabling conversions when the output is disabled, and enabling an internal amplifier for larger loads.

2. API Information

The sample code in this application note has been run and confirmed under the following conditions.

2.1 Hardware Requirements

This driver requires that your MCU support the following features:

- DA, DAa, R12DA, or R12DAA digital-to-analog converters.

2.2 Hardware Resource Requirements

This section details the hardware peripherals that this driver requires. Unless explicitly stated, these resources must be reserved for the driver, and the user cannot use them.

2.2.1 DA, DAa, R12DA, R12DAA

This driver makes use of all features on these peripherals.

2.2.2 GPIO

This driver utilizes port pins corresponding to each individual channel. These pins may not be used for GPIO.

2.3 Software Requirements

This driver is dependent upon the following packages:

- Renesas Board Support Package (r_bsp).

2.4 Limitations

No software limitations.

2.5 Supported Toolchains

This driver is tested and working with the following toolchains:

- Renesas RX Toolchain v.2.02.00 (RX111, RX113, RX210, RX231, RX631, RX63N, RX64M, RX71M)
- Renesas RX Toolchain v.2.03.00 (RX130, RX23T, RX230, RX24T)
- Renesas RX Toolchain v.2.05.00 (RX24U, RX651, RX65N)
- Renesas RX Toolchain v.2.06.00 (RX24U)

2.6 Header Files

All API calls and their supporting interface definitions are located in “r_dac_rx_if.h”.

Build-time configuration options are selected or defined in the file “r_dac_rx_config.h”.

Both of these files should be included by the user’s application.

2.7 Integer Types

This project uses ANSI C99 “Exact width integer types” in order to make the code clearer and more portable. These types are defined in *stdint.h*.

2.8 Configuration Overview

This driver uses the equate DAC_CFG_PARAM_CHECKING_ENABLE to remove parameter checking from the API functions and reduce overall code size.

Configuration options in *r_dac_rx_config.h*	
`#define DAC_CFG_PARAM_CHECKING_ENABLE 1`	If this equate is set to 1, parameter checking is included in the build. If the equate is set to 0, the parameter checking is omitted from the build and code size is reduced. Setting this equate to BSP_CFG_PARAM_CHECKING_ENABLE utilizes the system default setting.

2.9 Code Size

Typical code sizes associated with this module are listed below. Information is listed for a single representative device of the RX100 Series, RX200 Series, and RX600 Series, respectively.
The ROM (code and constants) and RAM (global data) sizes are determined by the build-time configuration options described in 2.8, Configuration Overview. The table lists reference values when the C compiler's compile options are set to their default values, as described in 2.5, Supported Toolchains. The compile option default values are optimization level: 2, optimization type: for size, and data endianness: little-endian. The code size varies depending on the C compiler version and compile options.

ROM, RAM and Stack Code Sizes

Device	Category	Memory Used		Remarks
		With Parameter Checking	Without Parameter Checking	
RX130	ROM	319 bytes	285 bytes	
	RAM	0 byte	0 byte	
	Maximum stack usage	20 bytes		R_DAC_Open function used
RX231	ROM	338 bytes	289 bytes	
	RAM	0 byte	0 byte	
	Maximum stack usage	20 bytes		R_DAC_Open function used
RX65N	ROM	403 bytes	359 bytes	
	RAM	0 byte	0 byte	
	Maximum stack usage	20 bytes		R_DAC_Open function used

2.10 API Data Structures

The API data structures are located in the file "r_dac_rx_if.h" and discussed in Section 3.

2.11 Adding Driver to Your Project

The FIT module must be added to each project in the e^2 studio.

You can use the FIT plug-in to add the FIT module to your project, or the module can be added manually.

It is recommended to use the FIT plug-in as you can add the module to your project easily and also it will automatically update the include file paths for you.

To add the FIT module using the plug-in, refer to chapter 2. "Adding FIT Modules to e^2 studio Projects Using FIT Plug-In" in the "Adding Firmware Integration Technology Modules to Projects" application note (R01AN1723).

To add the FIT module manually, refer to chapter 3. "Adding FIT Modules to e^2 studio Projects Manually" in the "Adding Firmware Integration Technology Modules to Projects (R01AN1723)"

When using the FIT module, the BSP FIT module also needs to be added. For details on the BSP FIT module, refer to the "Board Support Package Module Using Firmware Integration Technology" application note (R01AN1685).

3. API Functions

3.1 Summary

The following functions are included in this design:

Function	Description
R_DAC_Open()	Applies power to the DAC peripheral, initializes the associated registers, and configures MCU-specific options.
R_DAC_Close()	Removes power to the DAC peripheral.
R_DAC_Write()	Writes data to channel register for conversion.
R_DAC_Control()	Enables or disables channel output. Enables or disabled internal amplifier (RX64M and RX71M).
R_DAC_GetVersion()	Returns at runtime the driver version number.

3.2 Return Values

Below are the different error codes API functions can return. This enum is found in r_dac_rx_if.h along with the API function declarations.

```
/* DAC API ERROR CODE DEFINITIONS */
typedef enum e_dac_err
{
    DAC_SUCCESS=0,
    DAC_ERR_BAD_CHAN,          // non-existent channel number
    DAC_ERR_INVALID_CMD,       // non-existent operation command
    DAC_ERR_INVALID_ARG,       // argument is not valid for parameter
    DAC_ERR_NULL_PTR,          // received null ptr; missing required argument
    DAC_ERR_LOCK_FAILED,       // failed to lock DAC module (module already open)
    DAC_ERR_UNLOCK_FAILED      // failed to unlock DAC module
    DAC_ERR_ADC_NOT_POWERED,   // cannot sync because ADC is not powered
    DAC_ERR_ADC_CONVERTING     // cannot sync because ADC is converting
} dac_err_t;
```

3.3 R_DAC_Open()

This function applies power to the DAC module, initializes the associated registers, and configures MCU-specific options.

Format

```
dac_err_t R_DAC_Open(dac_cfg_t * p_cfg);
```

Parameters

p_cfg
Pointer to the configuration structure

Sample structure used for p_cfg:

```
typedef struct st_dac_cfg
{
    bool            fmt_flush_right;            // all MCUs
    bool            sync_with_adc;              // RX113/RX130/RX230/RX231/
                                                // RX24U/RX63N/RX631/RX64M/
                                                // RX65N/RX651/RX71M
    uint8_t         sync_unit;                  // 0 or 1; RX64M/RX71M
                                                // RX65N/RX651
    bool            ch_conv_off_when_output_off; // RX210/RX63N/RX631/RX64M/
                                                 // RX65N/RX651/RX71M
    dac_refv_t      ref_voltage;                // RX113/RX230/RX231
} dac_cfg_t;


typedef enum e_dac_refv  // DAC reference voltage
{
    DAC_REFV_AVVC0_AVSS0    = 1,
    DAC_REFV_INTERNAL_AVSS0 = 3,
    DAC_REFV_VREFH_VREFL    = 6
} dac_refv_t;
```

Return Values

DAC_SUCCESS:	*Successful; DAC initialized*
DAC_ERR_NULL_PTR:	*p_cfg pointer is NULL*
DAC_ERR_LOCK_FAILED:	*Failed to lock DAC module; already opened*
DAC_ERR_INVALID_ARG:	*Invalid unit number for sync_unit*
DAC_ERR_ADC_NOT_POWERED:	*Cannot sync because ADC is not powered*
DAC_ERR_ADC_CONVERTING:	*Cannot sync because ADC is converting*

Properties

Prototyped in file “r_dac_rx_if.h”

Description

This function applies power to the DAC module, initializes the associated registers, and configures MCU-specific options.

Reentrant

No.

Example

```
    dac_err_t  err;
    dac_cfg_t  config;

    /* Initialize RX63N DAC */
    config.fmt_flush_right = true;
    config.sync_with_adc = false;
    config.ch_conv_off_when_output_off = true;
    err = R_DAC_Open(&config);
```

Special Notes:

Data must be left or right justified by the application. The “fmt_flush_right” parameter just tells the DAC how to interpret the data.

To avoid a “DAC_ERR_ADC_CONVERTING”, open the DAC module after the ADC is opened but before scanning has begun on the ADC.

The DAC I/O pins must be configured prior to calling this function. An example initialization follows:

```
    R_BSP_RegisterProtectDisable(BSP_REG_PROTECT_MPC); // unlock

#ifdef BSP_MCU_RX113
    /*
     * Per Note 1 below Table 19.1 Allocation of Pin Functions to Multiple
     * Pins (10/10) in the RX113 Group User's Manual: Hardware:
     *    Select general input (by setting the Bm bits for the given pin in the
     *    PDR and PMR for the given port to 0) for the pin if this pin function
     *    is to be used.
     */
    PORTJ.PDR.BIT.B0 = 0;
    PORTJ.PMR.BIT.B0 = 0;
    PORTJ.PDR.BIT.B2 = 0;
    PORTJ.PMR.BIT.B2 = 0;

  /* Set the pin function for PJ0 & PJ2 to be used as DAC analog output pins. */
    MPC.PJ0PFS.BIT.ASEL = 1;
    MPC.PJ2PFS.BIT.ASEL = 1;

    /*
     * Uncomment the two lines below if you want to use VREFH/VREFL for the DAC
     * reference voltage.
     */
    //MPC.P41PFS.BIT.ASEL = 1;   // Configure P41 as a VREFH analog pin
    //MPC.P42PFS.BIT.ASEL = 1;   // Configure P42 as a VREFL analog pin

#else  /* RX111, RX210, RX63N */

    /* Configure I/O port pins for analog outputs as general input pins.
    PORT0.PDR.BIT.B3 = 0;
    PORT0.PMR.BIT.B3 = 0;
    PORT0.PDR.BIT.B5 = 0;
    PORT0.PMR.BIT.B5 = 0;

  /* Set the pin function for P03 & P05 to be used as DAC analog output pins. */
    MPC.P03PFS.BIT.ASEL = 1;
    MPC.P05PFS.BIT.ASEL = 1;

#endif

    R_BSP_RegisterProtectEnable(BSP_REG_PROTECT_MPC);  // lock
```

Note when using an amplifier
When using an amp, set true in ch_conv_off_when_output_off.

Note when using the A/D converter
When the D/A A/D synchronous conversion (sync_with_adc = true) is enabled, if the A/D converter (1) is to be placed in the module stop state, first, execute the R_DAC_Close function.

Note 1. The intended A/D converter is unit 1 for RX64M/RX651/RX65N/RX71M and unit 2 for RX24U.

3.4 R_DAC_Close()

This function removes power from the DAC peripheral.

Format
```
dac_err_t R_DAC_Close(void);
```

Parameters
none

Return Values
DAC_SUCCESS: *Successful; channels closed*
DAC_ERR_UNLOCK_FAILED: *Failed to unlock DAC module*

Properties
Prototyped in file “r_dac_rx_if.h”

Description
Disables DAC channel output and powers down the peripheral.

Reentrant
No.

Example
```
 :
/* Initialize DAC Peripheral */
err = R_DAC_Open(&config);
 :
 :
/* Shut down DAC Peripheral */
err = R_DAC_Close();
```

Special Notes:
When the D/A A/D synchronous conversion (sync_with_adc = true) is enabled, if the A/D converter (1) is to be placed in the module stop state, first, execute the R_DAC_Close function.

Note 1. The intended A/D converter is unit 1 for RX64M/RX651/RX65N/RX71M and unit 2 for RX24U.

3.5 R_DAC_Write()

This function writes data to channel data register.

Format

```
dac_err_t R_DAC_Write(uint8_t const chan, uint16_t data);
```

Parameters

chan
Channel to write to
data
Data to write

Return Values

DAC_SUCCESS:	*Data written to channel register successfully*
DAC_ERR_BAD_CHAN:	*Non-existent channel number*

Properties

Prototyped in file “r_dac_rx_if.h”

Description

Writes data to the channel register for conversion. Depending upon the MCU, this data may be 8-, 10-, or 12-bits in length. The data must be aligned properly for the selected format before issuing a Write().

Reentrant

Function is re-entrant for different channels.

Example

```
dac_err_t     err;
uint16_t      g_short;
  :
  :
/* Write data for conversion to 0V on channel 1 */
g_short = 0x0000;
err = R_DAC_Write(DAC_CH1, g_short);
```

Special Notes:

None.

3.6 R_DAC_Control()

This function is used to enable/disable channel features.

Format

```
dac_err_t R_DAC_Control(uint8_t const chan, dac_cmd_t const cmd);
```

Parameters

chan
Channel to operate on
cmd
Command to run (see enumeration below)

The *cmd* values are as follows:

```
typedef enum e_dac_cmd
{
    DAC_CMD_OUTPUT_ON,    // Analog output of channel is enabled
    DAC_CMD_OUTPUT_OFF,   // Analog output of channel is disabled

    DAC_CMD_AMP_ON,       // RX64M/RX71M: Gain of 1 amplifier. See Electrical
    DAC_CMD_AMP_OFF,      //   Characteristics in User's Manual: Hardware.

    DAC_CMD_END_ENUM
} dac_cmd_t;
```

Return Values

DAC_SUCCESS:	*Successful; channel initialized*
DAC_ERR_BAD_CHAN:	*Non-existent channel number*
DAC_ERR_INVALID_CMD:	*Invalid command*

Properties

Prototyped in file “r_dac_rx_if.h”

Description

The output of conversion data written in data register by Write() function is enable by OUTPUT command while Amp is enable by AMP command, The output permission must be set after enabling amp.

Reentrant

Function is re-entrant for different channels.

Example

```
    dac_cfg_t     config;
    dac_err_t     err;

    /* Initialize RX64M,RX71M DAC */
    config.fmt_flush_right = true;
    config.sync_with_adc = true;
    config.sync_unit = 1;
    config.ch_conv_off_when_output_off = true;

    err = R_DAC_Open(&config);

    /* Write data for 0V on channel 0 */
    err = R_DAC_Write(DAC_CH0, 0x0);

    /* Drive a larger load */
    err = R_DAC_Control(DAC_CH0, DAC_CMD_AMP_ON);
    /*  It is necessary to wait more than 3us.  */
```

```
/* Output converted data */
err = R_DAC_Control(DAC_CH0, DAC_CMD_OUTPUT_ON);

/* Write data for 3.3V on channel 0 */
err = R_DAC_Write(DAC_CH0, 0x0FFF);
```

Special Notes:

Amp output is generated after R_DAC_Write(DAC_CHx, 0x0) is executed.
When amp out is in use (DAC_CMD_AMP_ON command running), set true in ch_conv_off_when_output_off.
When using an amp, follow the process below.

1. Execute DAC_CMD_AMP_ON command in R_DAC_Control function.
2. Execute DAC_CMD_OUTPUT_ON command in R_DAC_Control function
3. Wait more than 3.3us
4. Write D/A output value in R_DAC_Write function.

The DAC_CMD_OUTPUT_ON and DAC_CMD_OUTPUT_OFF commands must be executed while the A/D converter (1) to be synchronized with is stopped when the D/A A/D synchronous conversion is enabled.

Note 1. For RX64M/RX651/RX65N/RX71M, unit 1 of the A/D converter is to be stopped, and for RX24U, unit 2 is to be stopped. The other MCUs do not need to specify the unit to be stopped since they only have one unit.

3.7 R_DAC_GetVersion()

This function returns the driver version number at runtime.

Format
```
uint32_t R_DAC_GetVersion(void)
```

Parameters
None

Return Values
Version number.

Properties
Prototyped in file “r_dac_rx.h”

Description
Returns the version of this module. The version number is encoded such that the top 2 bytes are the major version number and the bottom 2 bytes are the minor version number.

Reentrant
Yes

Example
```
uint32_t   version;
  :
version = R_DAC_GetVersion();
```

Special Notes:
This function is inline using the “#pragma inline” directive

4. Demo Projects

Demo projects are complete stand-alone programs. They include function main() that utilizes the module and its dependent modules (e.g. r_bsp). The standard naming convention for the demo project is <module>_demo_<board> where <module> is the peripheral acronym (e.g. s12ad, cmt, sci) and the <board> is the standard RSK (e.g. rskrx113). For example, s12ad FIT module demo project for RSKRX113 will be named as s12ad_demo_rskrx113. Similarly the exported .zip file will be <module>_demo_<board>.zip. For the same example, the zipped export/import file will be named as s12ad_demo_rskrx113.zip

4.1 dac_demo_rskrx113

This is a simple demo of the RX113 D/A Converter (R12DAA) for the RSKRX113 starter kit (FIT module "r_dac_rx"). The demo uses the r_dac_rx API to enable, configure, and write to the DAC. A continuous loop is entered in which low, medium, and high data values are written to DAC channel 1 for 1 second each. LED 0 (green) is lit when a low value is written, LED 1 (orange) is lit when a medium value is written, and LED 2 (red) is lit when a high value is written. See the "Notes for Measuring the DAC Channel 0/1 Output Signals" section below for details on accessing and configuring the DAC output channel signals and reference voltages on the RSKRX113 board.

Setup and Execution

1. Compile and download the sample code.
2. Attach multimeter leads or oscilloscope probes to the DAC channel output pin(s) if desired.
3. Click 'Reset Go' to start the software. If PC stops at Main, press F8 to resume.
4. Set breakpoints and watch global variables

Boards Supported

RSKRX113

Notes For Measuring the DAC Channel 0/1 Output Signals

- DAC channel 0 (DA0) output uses I/O Port PJ0 which maps to pin 2 of the MCU.

  Per the RSKRX113 schematic, DA0 shares pin 2 with switch 1 (SW1). To use pin 2 for the DA0 analog output signal a 0 ohm resistor needs to be moved from R241 to R239. Once this has been done DA0 can be accessed via header JA1_13 or J1_2. Note that once this link configuration change is made SW1 will not be usable.

- DAC channel 1 (DA1) output uses I/O Port PJ2 which maps to physical pin 100.

  Per the RSKRX113 schematic, DA1 can be accessed via header JA1_14.

  DA1 can also be accessed via J4_25.

- GROUND can be accessed via JA1_2 (pin 4 next to it is also a ground pin)
- The RX113 supports three possible DAC reference voltages via the DAVREFCR register:

  1. AVCC0/AVSS0

     On the RSKRX113 board, AVCC0/AVSS0 are connected to UC_VCC (typ. 3.3V) and GROUND, respectively.

  2. Internal reference voltage/AVSS0

     Typically 1.4V and GROUND, respectively.

  3. VREFH/VREFL

     On the RSKRX113 board, VREFH/VREFL are not connected. They go to J4_18 (CON_VREFH) and J4_17 (CON_VREFL), respectively. In order to use VREFH/VREFL as the DAC reference voltage:

     - I/O pins P41 & P42 must be configured via the MPC as analog pins.
     - VREFH/VREEFL must be connected to high/low supply voltages:

       Refer to the "DAC Configuration" section of the RSKRX113 User's Manual (R20UT2762EJ0100) for details on the option links for configuring these signals.

- An alternative option is:

  Connect J3_12 (GROUND) to J4_17 (CON_VREFL)

  Connect J3_10 (UC_VCC, 3.3V) to J4_18 (CON_VREFH)

4.2 dac_demo_rskrx231

This is a simple demo of the RX231 D/A Converter (R12DAA) for the RSKRX231 starter kit (FIT module "r_dac_rx"). The demo uses the r_dac_rx API to enable, configure, and write to the DAC. A continuous loop is entered in which low, medium, and high data values are written to DAC channel 1 for 1 second each. LED 0 (green) is lit when a low value is written, LED 1 (orange) is lit when a medium value is written, and LED 2 (red) is lit when a high value is written. See the "Operation" notes below for details on accessing and configuring the DAC output channel signals and reference voltages on the RSKRX231 board.

Setup and Execution

1. Compile and download the sample code.
2. Attach multimeter leads or oscilloscope probes to the DAC channel output pin(s) if desired.
3. Click 'Reset Go' to start the software. If PC stops at Main, press F8 to resume.
4. Set breakpoints and watch global variables

Boards Supported

RSKRX231

Notes For Measuring the DAC Channel 0/1 Output Signals

- DAC channel 0 (DA0) output uses I/O Port P03 which maps to pin 2 of the MCU. It can be accessed via J1_2.
- DAC channel 1 (DA1) output uses I/O Port P05 which maps to pin 100 of the MCU. It can be accessed via JA1_14 (or J4_25).
- GROUND can be accessed via JA1_2 (pin 4 next to it is also a ground pin)
- The RX231 supports three possible DAC reference voltages via the DAVREFCR register:

  1. AVCC0/AVSS0

     On the RSKRX231 board, AVCC0/AVSS0 are connected to UC_VCC (typ. 3.3V) and GROUND, respectively.

  2. Internal reference voltage/AVSS0

     Typically 1.4V and GROUND, respectively.

  3. VREFH/VREFL

     On the RSKRX231 board, VREFH/VREFL are connected to UC_VCC (typ. 3.3V) and GND, respectively. To connect an external reference voltage, CON_VREFH (J1_1) and CON_VREFL (J1_3) can be used after moving the following 0 ohm resistors:

     - For CON_VREFH move R68 to R67
     - For CON_VREFL move R65 to R66

4.3 dac_demo_rskrx64m

This is a simple demo of the RX64M D/A Converter (R12DA) for the RSKRX64M starter kit (FIT module "r_dac_rx"). The demo uses the r_dac_rx API to enable, configure, and write to the DAC. A continuous loop is entered in which low, medium, and high data values are written to DAC channel 0 for 1 second each. LED 1 is lit when a low value is written, LED 2 is lit when a medium value is written, and LED 3 is lit when a high value is written. See the "Operation" notes below for details on accessing and configuring the DAC output channel signals on the RSKRX64M board.

Setup and Execution

1. Compile and download the sample code.
2. Attach multimeter leads or oscilloscope probes to the DAC channel output pin(s) if desired.
3. Click ‘Reset Go’ to start the software. If PC stops at Main, press F8 to resume.
4. Set breakpoints and watch global variables

Boards Supported

RSKRX64M

Notes For Measuring the DAC Channel 0/1 Output Signals

- DAC channel 0 (DA0) output uses I/O Port P03 which maps to pin 4 of the MCU.

  Per the RSKRX64M schematic, DA0 shares pin 4 with LED 0. To use pin 4 for the DA0 analog output signal a 0 ohm resistor needs to be moved from R277 to R189. Once this has been done DA0 can be accessed via header JA1_13. Note that once this link configuration change is made LED 0 will not be usable.

- DAC channel 1 (DA1) output uses I/O Port P05 which maps to pin 2 of the MCU.

  Per the RSKRX64M schematic, DA1 shares pin 2 with LED 1. To use pin 2 for the DA1 analog output signal a 0 ohm resistor needs to be moved from R280 to R188. Once this has been done DA1 can be accessed via header JA1_14. Note that once this link configuration change is made LED 1 will not be usable.

- On the RSKRX64M board the DA reference voltage AVCC1 (VREFH) and AVSS1 (VREFL) are connected to UC_VCC (typ. 3.3V) and GROUND, respectively.

4.4 dac_demo_rskrx71m

This is a simple demo of the RX71M D/A Converter (R12DA) for the RSKRX71M starter kit (FIT module “r_dac_rx”). The demo uses the r_dac_rx API to enable, configure, and write to the DAC. A continuous loop is entered in which low, medium, and high data values are written to DAC channel 0 for 1 second each. LED 1 is lit when a low value is written, LED 2 is lit when a medium value is written, and LED 3 is lit when a high value is written. See the “Operation” notes below for details on accessing and configuring the DAC output channel signals on the RSKRX71M board.

Setup and Execution

1. Compile and download the sample code.
2. Attach multimeter leads or oscilloscope probes to the DAC channel output pin(s) if desired.
3. Click ‘Reset Go’ to start the software. If PC stops at Main, press F8 to resume.
4. Set breakpoints and watch global variables

Boards Supported

RSKRX71M

Notes For Measuring the DAC Channel 0/1 Output Signals

- DAC channel 0 (DA0) output uses I/O Port P03 which maps to pin 4 of the MCU.

  Per the RSKRX71M schematic, DA0 shares pin 4 with LED 0. To use pin 4 for the DA0 analog output signal a 0 ohm resistor needs to be moved from R277 to R189. Once this has been done DA0 can be accessed via header JA1_13. Note that once this link configuration change is made LED 0 will not be usable.

- DAC channel 1 (DA1) output uses I/O Port P05 which maps to pin 2 of the MCU.

  Per the RSKRX71M schematic, DA1 shares pin 2 with LED 1. To use pin 2 for the DA1 analog output signal a 0 ohm resistor needs to be moved from R280 to R188. Once this has been done DA1 can be accessed via header JA1_14. Note that once this link configuration change is made LED 1 will not be usable.

- On the RSKRX71M board the DA reference voltage AVCC1 (VREFH) and AVSS1 (VREFL) are connected to UC_VCC (typ. 3.3V) and GROUND, respectively.

4.5 Adding a Demo to a Workspace

Demo projects are found in the FITDemos subdirectory of the distribution file for this application note. To add a demo project to a workspace, select File>Import>General>Existing Projects into Workspace, then click “Next”. From the Import Projects dialog, choose the “Select archive file” radio button. “Browse” to the FITDemos subdirectory, select the desired demo zip file, then click “Finish”.

Related Technical Updates

This module reflects the content of the following technical updates.

None

Website and Support

Renesas Electronics Website

http://www.renesas.com/

Inquiries

http://www.renesas.com/contact

Revision Record

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	Nov.15.13	—	First edition issued
2.00	Apr.02.14	—	Updated for new API and RX210 & RX63N/631 support
2.10	Sep.08.14	—	Added RX64M support
2.20	Jan.20.15	—	Added RX113 support
2.30	Mar.19.15	—	Added RX71M support
2.40	Jun.30.15	—	Added RX231 support
2.50	Sep.30.15	—	Added RX23T support
2.60	Oct.1.15	—	Added RX130 support
2.70	Dec.1.15	—	Added RX230 support
		1, 5	Changed the document number for the “Board Support Package Firmware Integration Technology Module” application note.
		3	Changed the description in section 2.
		3	Removed “DAA” from the required peripheral lists in sections 2.1 and 2.2.
		7, 11	Modified some code examples shown in the Parameters and Example in sections 3.3 and 3.6.
		14	Added “4. Demo Projects”.
2.80	Feb.1.16	—	Added RX24T support
		18	Added “Related Technical Updates”.
2.91	Oct.1.16	—	Added RX65N Group support
		5	ROM,RAM and stack Code Sizes description change
		8	Added The notice when using an amplifier
		11, 12	Description, Example change, Special Notes addition
3.00	Feb.28.17	—	Added support for the RX24T (including ROM 512 KB version) and RX24U Groups.
		3	Added RXC v2.06.00 to “2.5 Supported Toolchains”.
		9, 10, 13	3.3 R_DAC_Open(), 3.4 R_DAC_Close(), and 3.6 R_DAC_Control(): Added the note when enabling the D/A A/D synchronous conversion in the Special Notes.
		Program	For RX64M, RX71M, and RX65N, the code has been modified to set unit 0 as the unit to be synchronized with and also generate an error when the D/A A/D synchronous conversion is enabled.

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. 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has 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. Moreover, 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.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the 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

1. 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.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other disputes 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, drawing, chart, program, algorithm, 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 not alter, modify, copy, or otherwise misappropriate 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, copy or otherwise misappropriation of Renesas Electronics products.
5. 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; and 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.

   Renesas Electronics products are neither intended nor 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 damages (space and 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 third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. When using the 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 radiation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions or failure or accident arising out of the use of Renesas Electronics products beyond such specified ranges.
7. 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. Further, Renesas Electronics products are not subject to radiation resistance design. Please ensure to implement safety measures to guard them against the possibility of bodily injury, injury or damage caused by fire, and social damage in the event of 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 by your own responsibility as warranty for your products/system. Because the evaluation of microcomputer software alone is very difficult and not practical, please evaluate the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please investigate applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive carefully and sufficiently and use 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.
9. 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 not use Renesas Electronics products or technologies for (1) any purpose relating to the development, design, manufacture, use, stockpiling, etc., of weapons of mass destruction, such as nuclear weapons, chemical weapons, or biological weapons, or missiles (including unmanned aerial vehicles (UAVs)) for delivering such weapons, (2) any purpose relating to the development, design, manufacture, or use of conventional weapons, or (3) any other purpose of disturbing international peace and security, and you shall not sell, export, lease, transfer, or release Renesas Electronics products or technologies to any third party whether directly or indirectly with knowledge or reason to know that the third party or any other party will engage in the activities described above. When exporting, selling, transferring, etc., Renesas Electronics products or technologies, you shall comply with any applicable export control laws and regulations promulgated and administered by the governments of the countries asserting jurisdiction over the parties or transactions.
10. Please acknowledge and agree that you shall bear all the losses and damages which are incurred from the misuse or violation of the terms and conditions described in this document, including this notice, and hold Renesas Electronics harmless, if such misuse or violation results from your resale or making Renesas Electronics products available any third party.
11. This document shall not be reprinted, 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.

(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.

(Rev.3.0-1 November 2016)

RENESAS

SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL II Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 6.0