SPC58NE84 eMIOS example

Quick Guide

Version 1.0, March, 2018

Table of Contents

1. Terms & Abbreviations

PWM
: Pulse Width Modulation.

eMIOS
: Enhanced Modular IO Subsystem

PIT
: Periodic Interrupt Timer

IOP
: I/O processor

BSP
: Board Startup Package

GPIO
: General Purpose Input Output

crt0
: 'C' run-time environment initialization code

HAL
: Hardware Abstraction Layer

EVB
: Evaluation Board

NC
: Not Configured

uC
: Microcontroller

ISR
: Interrupt Service Routine

2. Introduction

This example is a small functional project designed to simplify an evaluation phase of the certain peripheral on the microcontroller. It is built on the c-startup example providing necessary low-level functions like a startup code, a minimalistic hardware abstraction or predefined memory partitioning. On top of this low-level implementation, it provides the peripheral example code running on single core.

2.1. Prerequisites

- SPC58NE84 device
- An evaluation board (SPC57xxMB + SPC58xxADPT292S)
- HighTec Development Suite, version: 4.9.3.0

2.2. HW resources

Hardware resources used by this example:

HW unit	channel / I/O pin	function
GPIO	PD[15]	Core [2] eMIOS[0] CH[0]
GPIO	PC[9]	Core [2] eMIOS[0] CH[1]
eMIOS[0]	Channel [0]	Core [2] PWM
eMIOS[0]	Channel [1]	Core [2] periodic interrupt
default resources from the c-startup example		
GPIO	PA [0]	Core [0] LED control
GPIO	PA [1]	Core [1] LED control
GPIO	PA [2]	Core [2] LED control
PIT 0	Channel [0]	Core [0] periodic interrupt
PIT 0	Channel [1]	Core [1] periodic interrupt
PIT 0	Channel [2]	Core [2] periodic interrupt

Tab. 1. HW resources

3. Example overview

The peripheral generates the PWM signal with a synchronous update of the duty cycle within eMIOS module. The duty cycle of the generated PWM periodically sweeps in the range from 1% up to 99% and backward. The PWM signal is routed to the external pins.

3.1. System Description

The figure below shows the system view of the example:

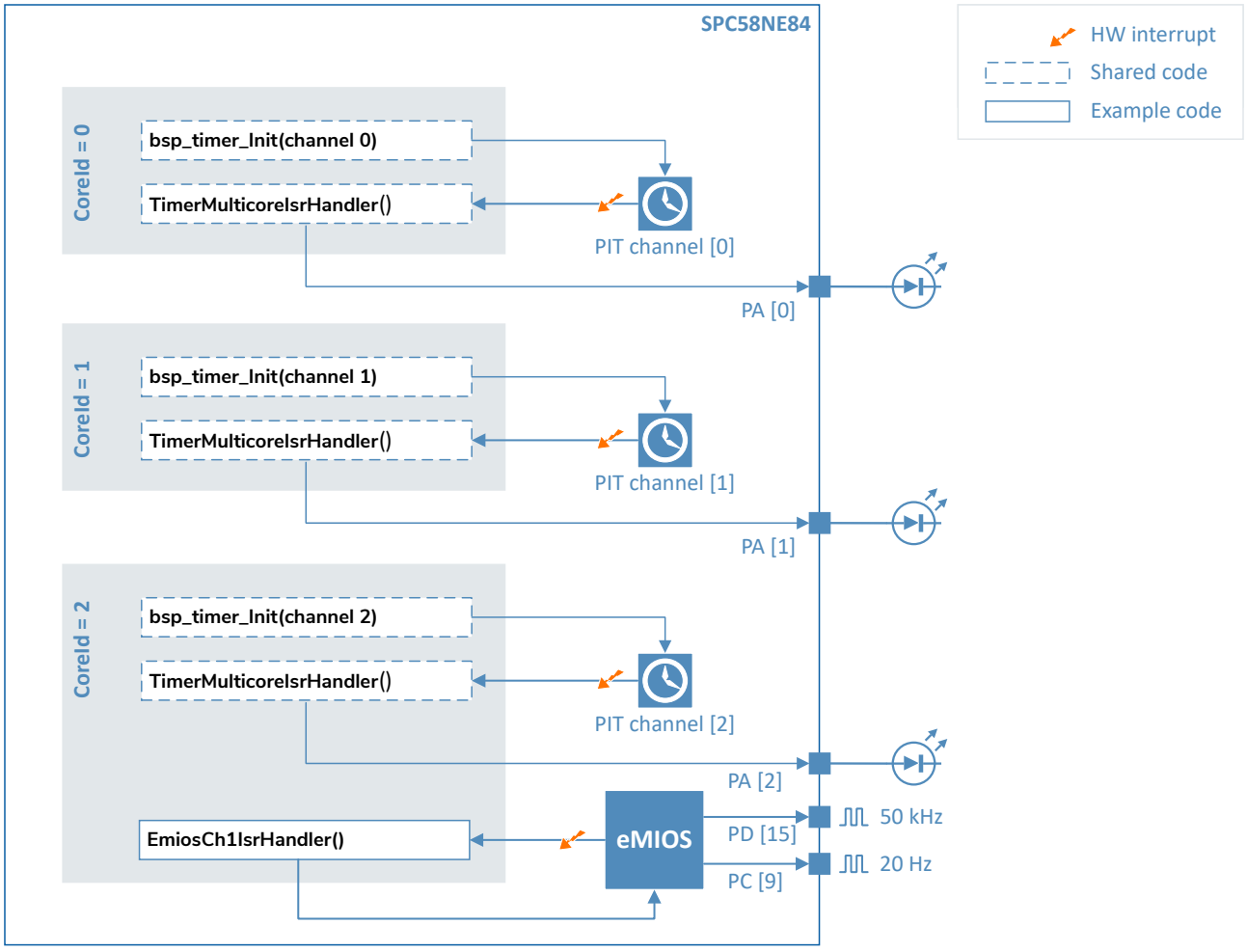

Fig. 1. System block diagram

The peripheral code runs on a single core. Other cores run the default functionality defined by SPC5xx_c_startup.

The example initializes two channels of the eMIOS_0 module.

- The channel 0 (CH0) generates the output PWM signal with the static frequency of 50kHz and with a dynamically changed duty cycle.
- The channel 1 (CH1) ensures a periodical update of the PWM duty cycle

The eMIOS_0 CH1 timer generates an interrupt every 50ms in which PWM duty cycle is updated. One duty cycle increment means 1% and the sweep range is from 1% up to 99%.

The synchronous update of the duty cycle is ensured by writing a new value to shadow registers `A2` and `B2` without preceding disable of the update mechanism. The new duty cycle is applied in the period following the period where the update of register is done.

3.2. Application Flow

The `UC_CORE_IOP` core controls the eMIOS peripheral.

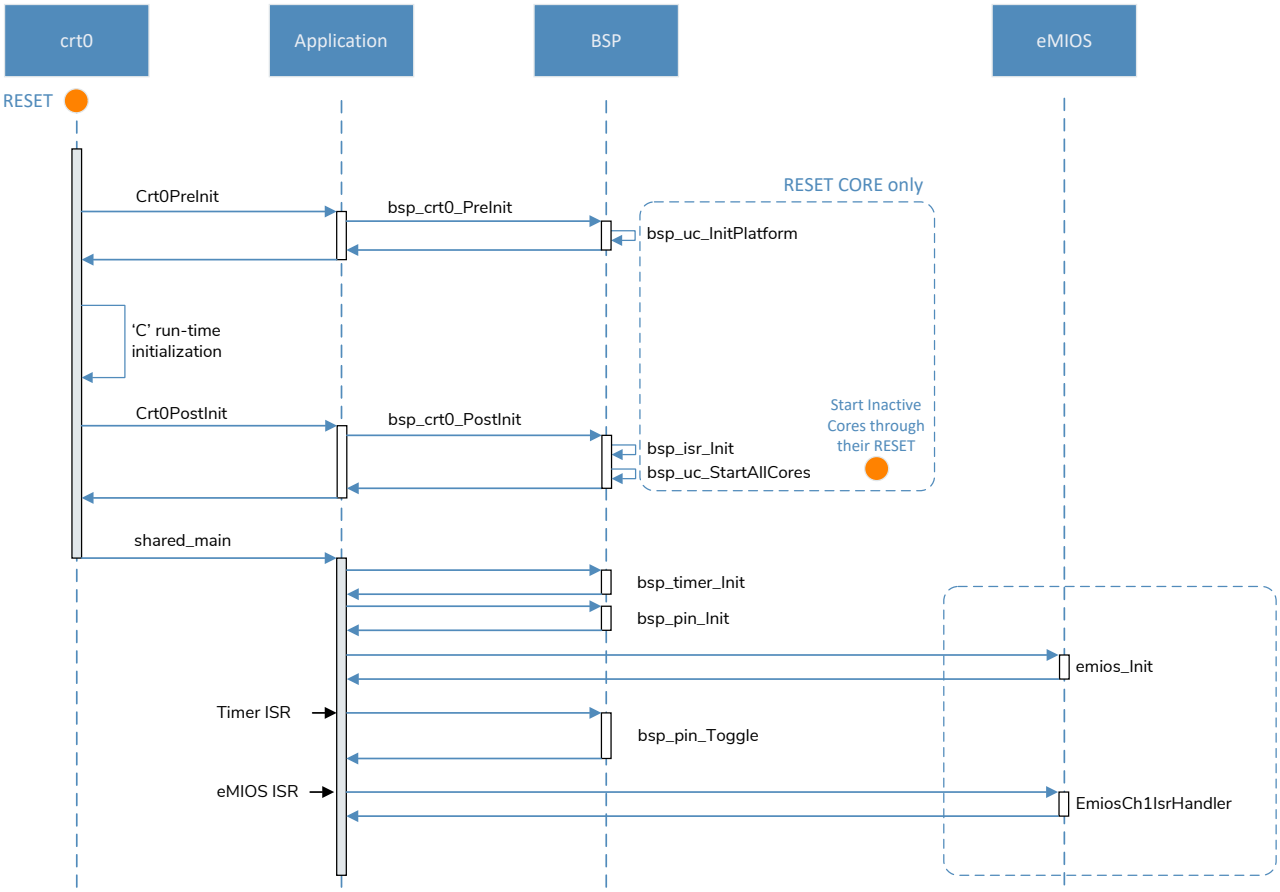

Fig. 2. Simplified individual core execution flow

3.3. Interrupt Handling

Implemented interrupt handling uses low-level BSP functions, for example to register each interrupt service routine (`bsp_isr_RegisterHandler`).

The eMIOS_0 initialization function assigns the `"EMIOS_GFR[F0,F1]"` interrupt vector and sets the interrupt vector priority.

Code 1. /app/emios.c

```
bsp_isr_RegisterHandler(UC_EMIOS_CH1_ISR, 10, EmiosCh1IsrHandler);
```

The example uses the SW interrupt mode.

3.4. eMIOS module configuration

The eMIOS module is configured with these parameters:

- eMIOS channels in Output Pulse Width and Frequency Modulation Buffered (OPWFMB) mode
- f_{eMIOS} = 100MHz
- f_{eMIOS_GLB} = 5MHz
- CH[0]$f_{EMIOSCNT}$ = f_{eMIOS_GLB} / 1 = 5MHz
- CH[1]$f_{EMIOSCNT}$ = f_{eMIOS_GLB} / 4 = 1.25MHz

for more details see `emios_Init()` function below.

Code 2. /app/emios.c

```
void emios_Init(void)
{
    /* Global initialization eMIOS */
    EMIOS_0.MCR.B.GPREN = 0x0; /* Prescaler disabled (no clock) and prescaler
                                  counter is cleared */
    EMIOS_0.MCR.B.GPRE = EMIOS_EMIOSMCR_GPRE; /* Global Prescaler */

        /*Configuration Channel 0 */
    EMIOS_0.CH[0].C.B.UCPREN = 0; /* Prescaler disabled (no clock) */
    EMIOS_0.CH[0].C.B.MODE = 0x1U; /* GPIO mode, the registers A1 and B1 can
                                      be set in this mode */
    EMIOS_0.CH[0].C.B.UCPRE = EMIOS_CH0_CTRL_CLK_SRC_SR;
    EMIOS_0.CH[0].C.B.BSL = 0x3; /* Bus Select: UC internal counter*/

    EMIOS_0.CH[0].A.R = EMIOS_CH0_PERIODE_TICKS/2;

    EMIOS_0.CH[0].B.R = EMIOS_CH0_PERIODE_TICKS;

    /*Configuration Channel 1 */
    EMIOS_0.CH[1].C.B.UCPREN = 0; /* Prescaler disabled (no clock) */
    EMIOS_0.CH[1].C.B.MODE = 0x1U; /* GPIO mode, the registers A1 and B1 can
                                      be set in this mode */
    EMIOS_0.CH[1].C.B.UCPRE = EMIOS_CH1_CTRL_CLK_SRC_SR;
    EMIOS_0.CH[1].C.B.BSL = 0x3;    /* Bus Select: UC internal counter*/
    EMIOS_0.CH[1].C.B.FEN = 0x1U;   /* FLAG Enable bit, interrupt enable */

    EMIOS_0.CH[1].A.R = ((EMIOS_CH1_PERIODE_TICKS * EMIOS_CH1_DUTY)/100);

    EMIOS_0.CH[1].B.R = EMIOS_CH1_PERIODE_TICKS;

    /* Set PWM mode */
    EMIOS_0.CH[0].C.B.MODE = 0x58U; /* Output Pulse Width and Frequency
                                       Modulation Buffered (OPWFMB) mode */
    EMIOS_0.CH[1].C.B.MODE = 0x58U; /* Output Pulse Width and Frequency
                                       Modulation Buffered (OPWFMB) mode */

    /* Double buffer register A2 and B2 configuration for CH0 and CH1*/
    EMIOS_0.CH[0].A.R = EMIOS_CH0_PERIODE_TICKS/2;
        EMIOS_0.CH[0].B.R = EMIOS_CH0_PERIODE_TICKS;

    EMIOS_0.CH[1].A.R = ((EMIOS_CH1_PERIODE_TICKS * EMIOS_CH1_DUTY)/100);
    EMIOS_0.CH[1].B.R = EMIOS_CH1_PERIODE_TICKS;

...

    /* Internal and global clock enable */
    EMIOS_0.CH[0].C.B.UCPREN = 1;
    EMIOS_0.CH[1].C.B.UCPREN = 1;
    EMIOS_0.MCR.B.GPREN = 1;
    EMIOS_0.MCR.B.GTBE = 1;
}
```

4. Example Import & Build

Follow these steps to import an example project to the HighTec IDE environment:

1. From menu **File** → **Import** → **General** choose an option `Existing Projects into Workspace`
2. Browse for your project location
3. Select project
4. Leave the `copy to the workspace option` empty
5. Click Finish

Activate the project from menu **Project** → **Set Active Project**.

Build the project from the menu **Project** → **Build Project**.

The output binary file is located under the `_irom_build` folder.

Appendix A: Document References

[1] "SPC5x c-startup - 'C' run-time initialization", HighTec EDV Systeme GmbH, 2018

[2] "SPC5x c-startup - Linker file template", HighTec EDV Systeme GmbH, 2018

[3] "SPC5x c-startup - Hardware Abstraction Layer", HighTec EDV Systeme GmbH, 2018

Appendix B: Release Notes

Version	Date	Changes to the previous version
1.0	March 2018	Initial version - based on SPC58NE84 c-startup example V2.2

HighTec EDV-Systeme GmbH
Feldmannstrasse 98, D-66119 Saarbrücken
info@hightec-rt.com
+49-681-92613-16
www.hightec-rt.com