A Simple PWM generation and Capture on HET module using HALCoGen for TMS570 / RM series of Devices

1. Setup

PWM generated on Pin – 10 and same PWM is CAPTURED on PIN – 30.

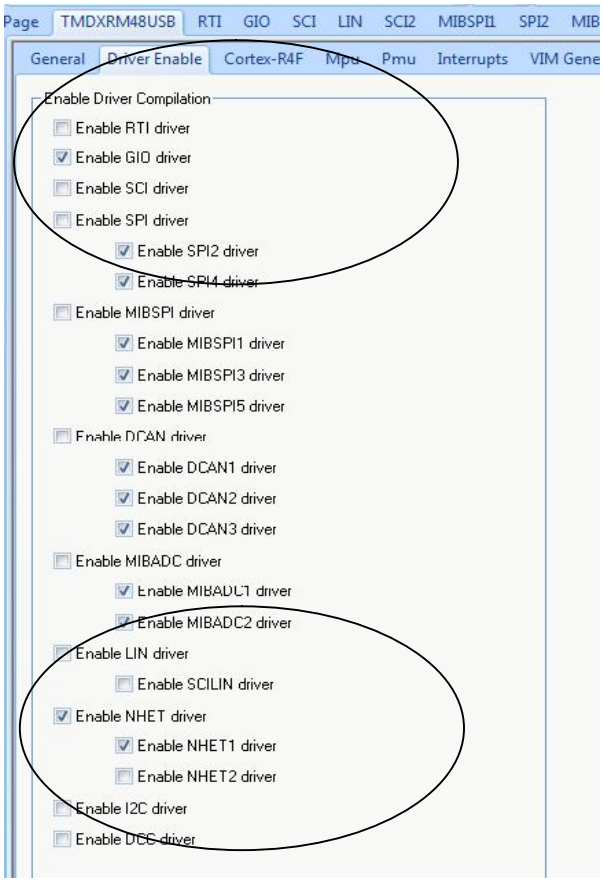
Loop Pin 10 and Pin 30.

The Hardware used for this sample code is RM48L950 series USB The RM48 Hercules USB Development Stick and HALCoGen ver 2.11 → http://www.ti.com/tool/halcogen

2. HALCOGEN Configuration

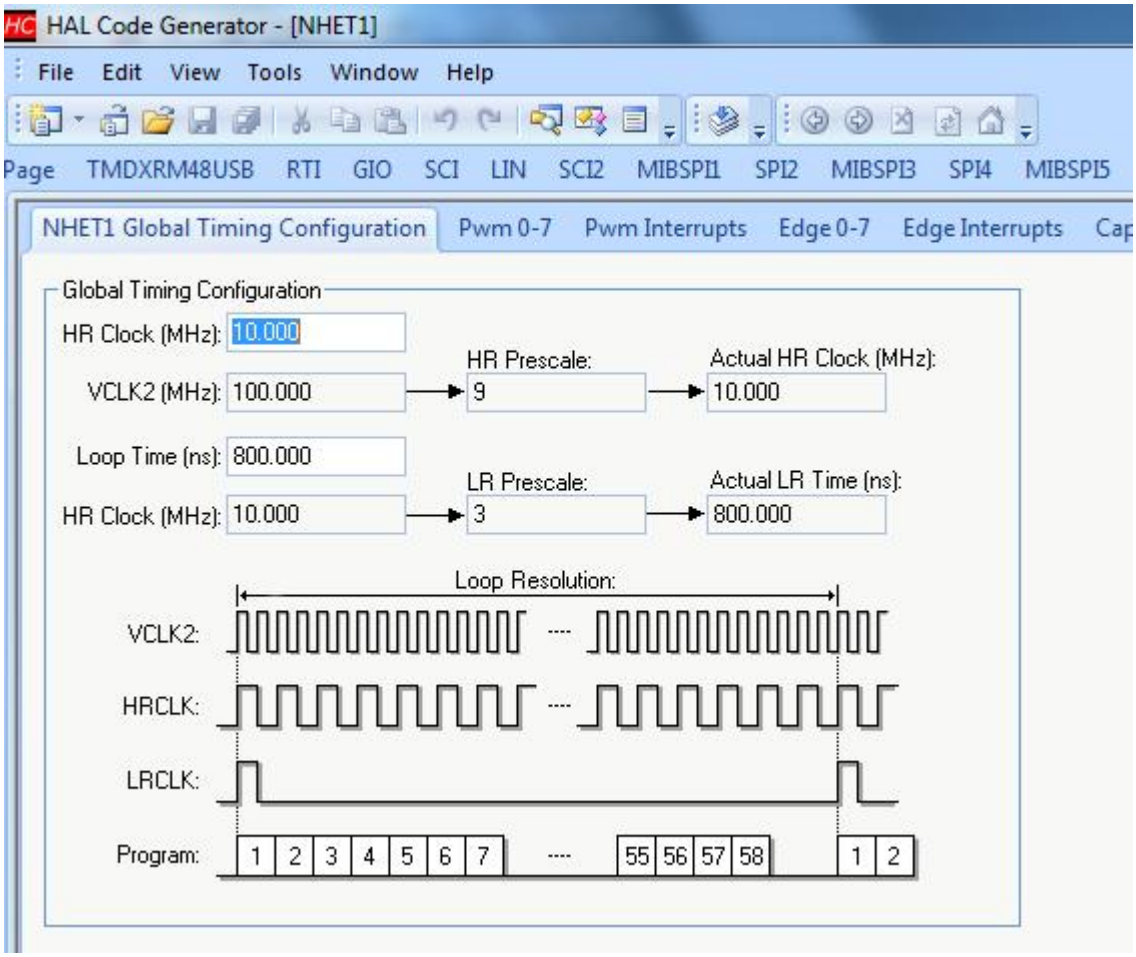
2.1. *Step 1*

- Enable GIO Driver and NHET1 Driver



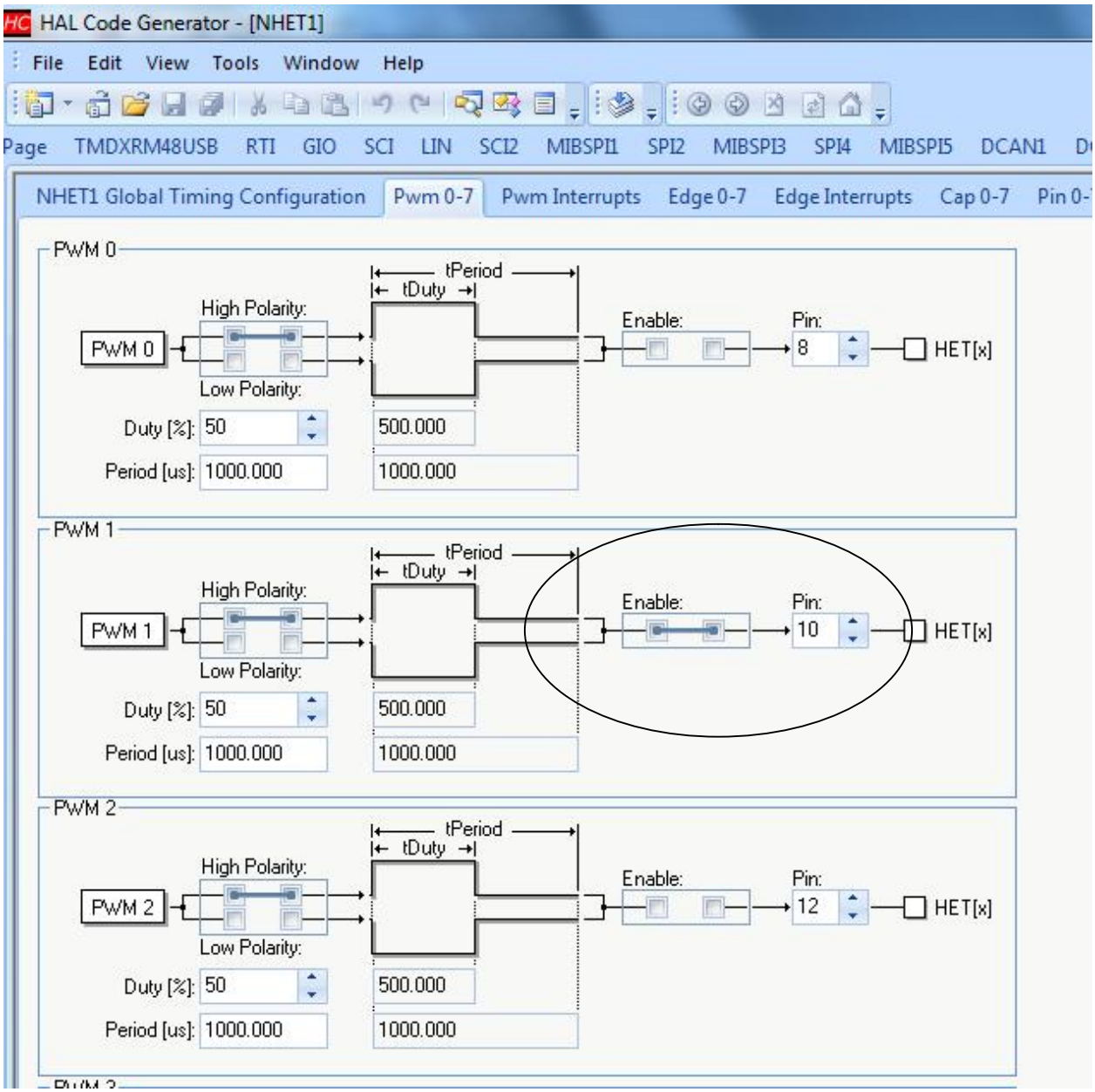
2.2. *Step 2*

- NHET1 Tab – Global Timing Configuration



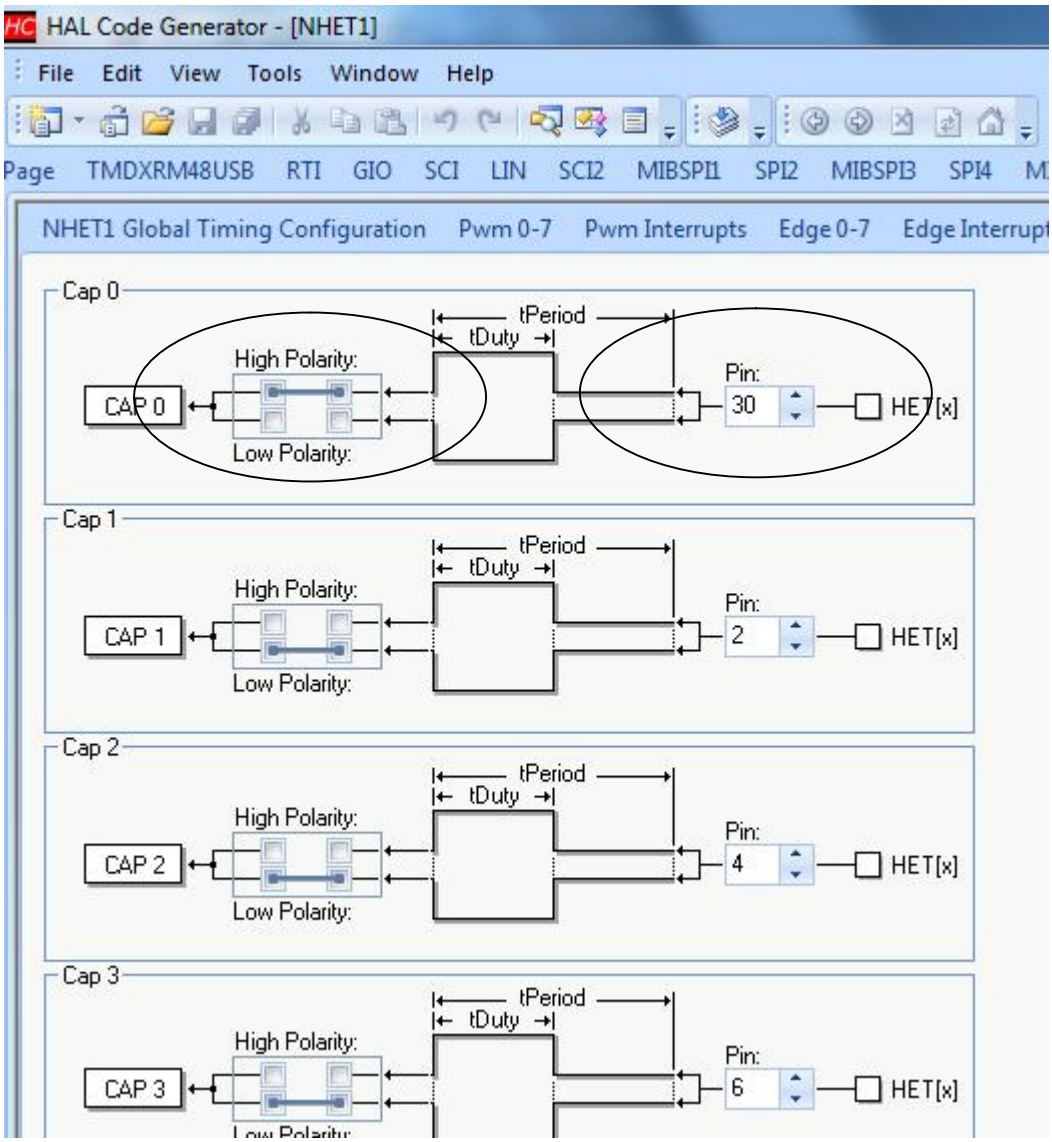
2.3. *Step 3*

- Enable PWM1 selecting PIN 10 and Polarity as High



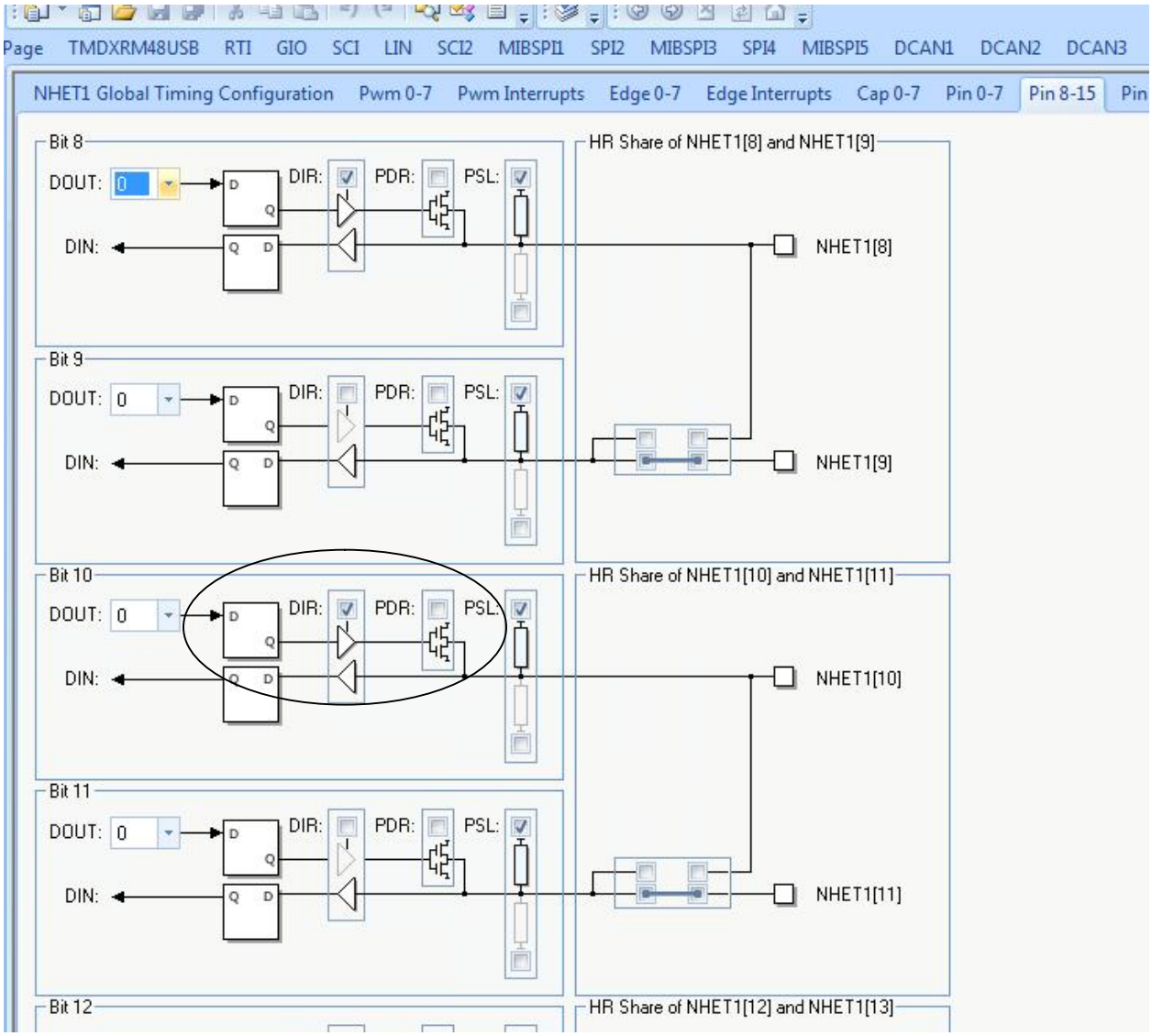
2.4. *Step 4*

- Enable CAP 0 selecting PIN 30 and Polarity as High



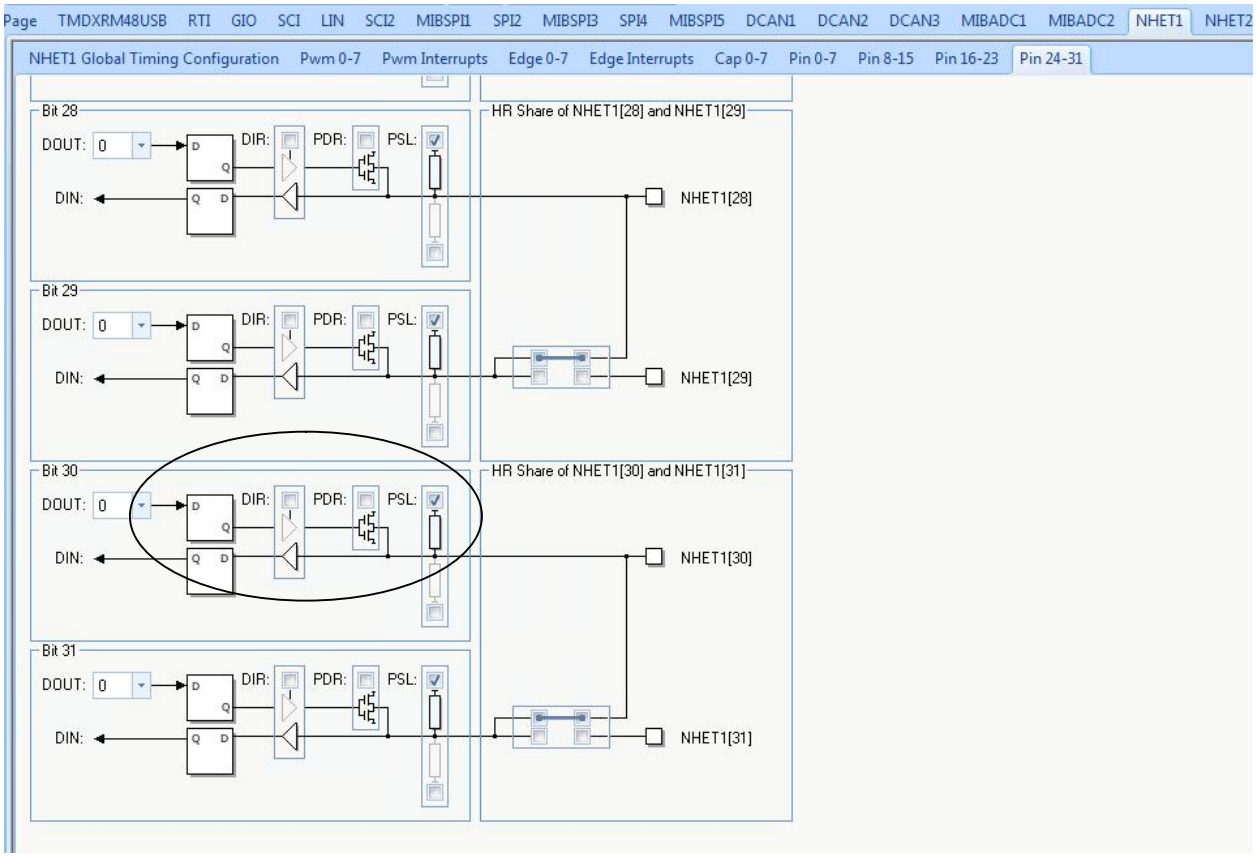
2.5. *Step 5*

- Enable PIN 10 (PWM1 generating pin) as output pin.
- HR Share feature is not used. (Can use for better resolution – It is out of scope of this doc)



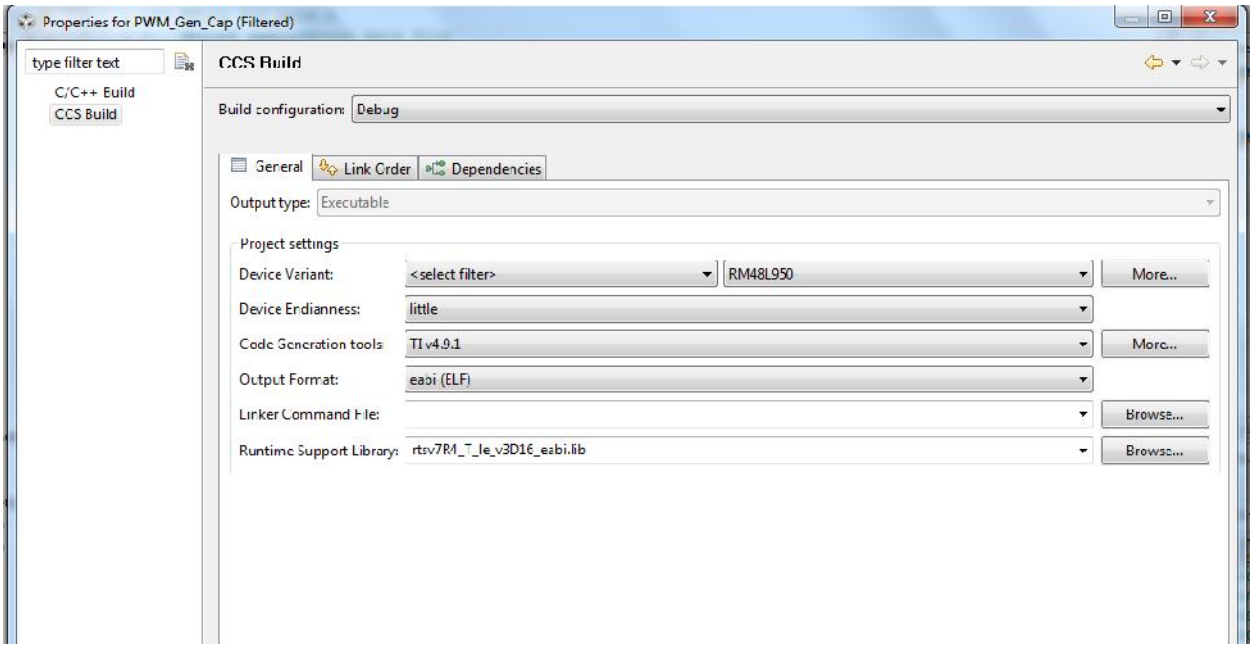
2.6. *Step 6*

- Enable PIN 30 (CAP 0 pin) direction as Input pin.



3. CCS Pjt creation

- HALCoGen generates the code once “Generate code” Button is clicked.
- Create CCS pjt for the device by selecting the correct device variant;
- Add all the Source files generated by the HALCoGen tool to the pjt file



4. Sample code Interpretation

1. Replace the sys_main.c with the below code ..

2. Compile, download and run the code.

3. Monitor the Structures

Duty_Period1 – Read before capture hence it should return 0.

Duty_Period2 – Contain the duty and period selected in the GUI

Duty_Period3 – Contain the duty and period selected in the GUI

Duty_Period4 – Contain the duty and period which are in the code.

```
/****************************************************************************/

/** @file sys_main.c
*   @brief Application main file
*   @date 04.October.2011
*   @version 1.02.000
*
*   This file contains an empty main function,
*   which can be used for the application.
*/

/* (c) Texas Instruments 2009-2011, All rights reserved. */

/* USER CODE BEGIN (0) */
/* USER CODE END */



/* Include Files */

#include "sys_common.h"
#include "system.h"

/* USER CODE BEGIN (1) */
#include <stdio.h>
#include "gio.h"
#include "sys_esm.h"
#include "nhet.h"
#include "dcc.h"
/* USER CODE END */


/** @fn void main(void)
*   @brief Application main function
```

```
*   @note This function is empty by default.
*
*   This function is called after startup.
*   The user can use this function to implement the application.
*/

/* USER CODE BEGIN (2) */
nhetSIGNAL_t Duty_Period1,Duty_Period2,Duty_Period3,Duty_Period4 ;
nhetSIGNAL_t Set_Duty_Period1;
/* USER CODE END */

void main(void)
{
/* USER CODE BEGIN (3) */
    int i;

    gioInit();
    gioSetDirection(nhetPORT1, 0xBFFFFFFF); // 30 Input, 10 Output
    gioSetBit(nhetPORT1, 10, 1);
    while(gioGetBit(nhetPORT1, 30) == 0);
    gioSetBit(nhetPORT1, 10, 0);

    Duty_Period1 = capGetSignal(nhetRAM1,cap0);

    nhetInit();

    for(i=0;i<0x10000;i++); // Simple Delay

    Duty_Period2 = capGetSignal(nhetRAM1,cap0);

    for(i=0;i<0x10000;i++); // Simple Delay

    Duty_Period3 = capGetSignal(nhetRAM1,cap0);

    pwmStop(nhetRAM1, pwm1);

    Set_Duty_Period1.duty = 75;
    Set_Duty_Period1.period = 2000;

    pwmSetSignal(nhetRAM1, pwm1, Set_Duty_Period1);
    //pwmSetDuty(nhetRAM1, pwm1, 75);

    pwmStart(nhetRAM1, pwm1);

    for(i=0;i<0x10000;i++); // Simple Delay

    Duty_Period4 = capGetSignal(nhetRAM1,cap0);

    for(i=0;i<0x10000;i++); // Simple Delay

/* USER CODE END */
}


/* USER CODE BEGIN (4) */
void esmGroup1Notification(uint32_t channel){}
void esmGroup2Notification(uint32_t channel){}
```

```
void gioNotification(int bit){}
void pwmNotification(nhetBASE_t * nhetREG,uint32_t pwm, uint32_t
notification){}
void edgeNotification(nhetBASE_t * nhetREG,uint32_t edge){}
void dccNotification(dccBASE_t  *dcc,uint32_t flags){}
/* USER CODE END */


   /*****************************************************************************/
```