M480 PLL and PWM Clock Setting

Example Code Introduction for 32-bit NuMicro® Family

Information

Application	Two applications that can list clock setting to generate target PLL and PWM settings.
BSP Version	N/A
Hardware	N/A

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1 Function Description

1.1 Introduction

NuMicro® M480 series consist of one PLL sourced from HIRC or HXT, allows CPU operation up to the maximum CPU frequency without the need for a high-frequency crystal. And the M480 can generate PWM waveform from BPWM and EPWM.
These two example codes list the PLL and BPWM/EPWM settings can generate the target output frequency from designated input clock frequency.

1.2 Principle

These two example codes loop through possible settings and print out the settings that can generate target frequency.

1.3 Demo Result

After execute the program, a terminal window will pop up and shows the available setting. User can then press any key to close the window.
Below is the output of PLL_CLOCK using 12MHz clock source to generate 192MHz PLL output.

```
FOUT=192000000.0, NO=2.000000, NR=2.000000 NF=32.000000
FOUT=192000000.0, NO=2.000000, NR=3.000000 NF=48.000000
```

Below is the output of PWM_CLOCK using 192MHz clock source to generate 8MHz PWM output. BPWM and EPWM supports different counter types. So the settings of different counter types are listed separately.

```
Up Counter/Down counter
PWMOUT= 8000000.0, PRESCLE=1.000000, PERIOD=24.000000
PWMOUT= 8000000.0, PRESCLE=2.000000, PERIOD=12.000000
PWMOUT= 8000000.0, PRESCLE=3.000000, PERIOD=8.000000
PWMOUT= 8000000.0, PRESCLE=4.000000, PERIOD=6.000000
PWMOUT= 8000000.0, PRESCLE=6.000000, PERIOD=4.000000
PWMOUT= 8000000.0, PRESCLE=8.000000, PERIOD=3.000000
PWMOUT= 8000000.0, PRESCLE=12.000000, PERIOD=2.000000
Up-Down Counter
PWMOUT= 8000000.0, PRESCLE=1.000000, PERIOD=12.000000
PWMOUT= 8000000.0, PRESCLE=2.000000, PERIOD=6.000000
PWMOUT= 8000000.0, PRESCLE=3.000000, PERIOD=4.000000
PWMOUT= 8000000.0, PRESCLE=4.000000, PERIOD=3.000000
PWMOUT= 8000000.0, PRESCLE=6.000000, PERIOD=2.000000
```

2 Code Description

PLL_CLOCK sample searches for the settings that can generate the target output frequency. Below is the piece of source code that searches for possible settings.

```
for(dNR = 1.0; dNR < 32.0; dNR = dNR + 1.0)
{
    // FREF = FIN / NR, 4MHz <= FREF <= 8MHz
    dFREF = FIN / dNR;
    if((dFREF < 4000000.0) || (dFREF > 8000000.0))
        continue;
    for(dNF = 2.0; dNF < 257.0; dNF = dNF + 1.0)
    {
        // FVCO = FREF * 2 * NF, 200MHz <= FVCO <= 500MHz
        dFVCO = dFREF * 2.0 * dNF;
        if((dFVCO < 200000000.0) || (dFVCO > 500000000.0))
            continue;
        for(dNO = 1.0; dNO <= 4.0; dNO = dNO * 2.0)
        {
            // FOUT = FVCO / NO, 50MHz <= FOUT <= 500MHz
            dFOUT = dFVCO / dNO;
            if((dFOUT < 50000000.0) || (dFOUT > 500000000.0))
                continue;
            if(dFOUT == TARGET)
                printf("==>FOUT=%10.1f, NO=%f, NR=%f NF=%f\n", dFOUT, dNO, dNR, dNF);
        }
    }
}
```

PWM_CLOCK sample searches for the settings that can generate the target output frequency. Following piece of source code is for Up-Counter type and Down-Counter type. Source code for Up-Down Counter type is identical except the line that calculates output frequency.

```
for(dPRESCALE = 1.0; dPRESCALE < 4096.0; dPRESCALE = dPRESCALE + 1.0)
{
    for(dPERIOD = 2.0; dPERIOD < 65536.0; dPERIOD = dPERIOD + 1.0)
    {
        dOUT = PWMCLK / (dPRESCALE * dPERIOD);
        if (dOUT == PWMOUT)
        {
            printf("PWMOUT=%10.1f, PRESCLE=%f, PERIOD=%f \n\r", PWMOUT, dPRESCALE,
dPERIOD);
        }
        else if (dOUT < PWMOUT)
```

```
        {
            break;
        }
    }
}
```

3 Software and Hardware Environment

Software Environment

- BSP version
  - N/A
- IDE version
  - Visual Studio 2017

Hardware Environment

- Circuit components
  - N/A
- Diagram
  - N/A

4 Directory Information

- EC_M480_PWM_and_PLL_Clock_Setting_V1.00
  - PLL_CLOCK
    - PLL_CLOCK — Source file of PLL_CLOCK example code.
  - PWM_CLOCK
    - PWM_CLOCK — Source file of PWM_CLOCK example code

5 How to Execute Example Code

1. Browsing into sample code folder by Directory Information (section 4) and double click project file PLL_CLOCK.sln or PWM_CLOCK.sln.
2. Build the project by pressing Shift + Ctrl + B.
3. Double click the generated executable file.

6 Revision History

Date	Revision	Description
Sept. 17, 2019	1.00	1. Initially issued.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.