EXAMPLE NAME: CCU4_SLICE_CONFIG_EXAMPLE_XMC47

EXAMPLE REVISION HISTORY:
1.0 Initial Version

OVERVIEW:
This example shows timer concatenation functionality of CCU4 unit. The necessity of using timer concatenation comes in case when low frequency with high resolution is required. For this reason, in this example PWM signal is generated with the time period of 1 second with the resolution of 6.944 ns. Due to the fact that each timer is 16 bit wide, it would be possible only to generate time intervals of 0.455104 ms with such a high resolution. To generate greater time intervals, timer concatenation is used. Additionally, to demonstrate the usage of shadow transfers, compare register values are constantly updated to generate duty cycle from 10-90% with 10% increments.

DESCRIPTION:
To calculate the values used by the period and compare register following algorithm is used:

```
time_interval = 1. / frequency
count = time_interval / tick_resolution

msb = 0;
while True:
    count = count / 2
    msb = msb + 1
    if (count < 65535):
        break

higher_period_value = (1 << msb) - 1
lower_period_value = int(count)

temp_period  = ((higher_period_value + 1) * (lower_period_value + 1)) + 1
compare_value = ((100 - duty_cycle) * (temp_period)) / 100
lower_compare_value = int(compare_value % lower_period_value)
higher_compare_value= int(compare_value / lower_period_value)
```

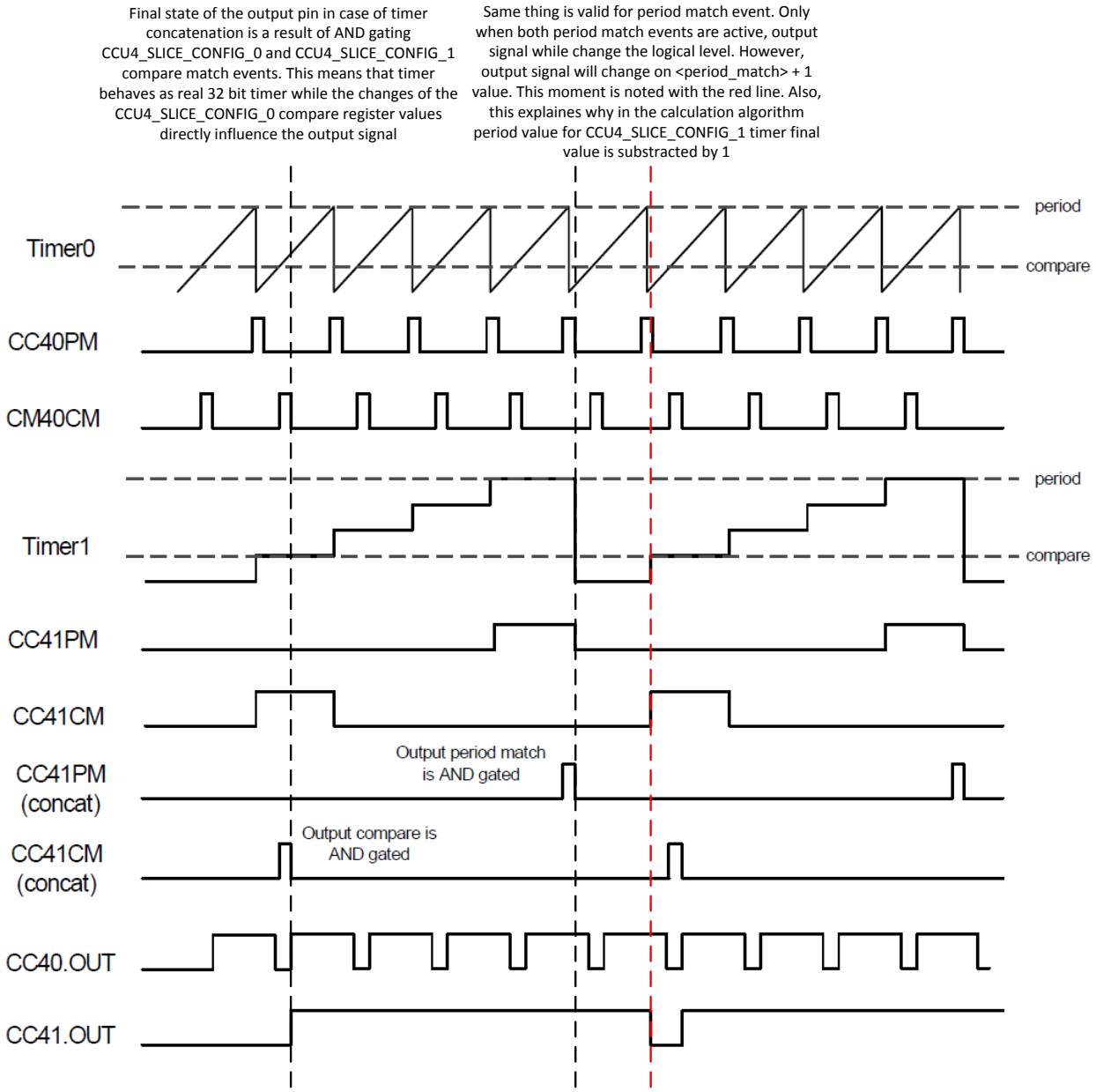
This is just the main logic behind the algorithm while the fully usable Python script is provided inside the project. Script expects as input parameters frequency [Hz], XMC device (used for resolution calculation), required resolution [ns] (based on the clock frequency first frequency that satisfies given requirement will be selected), duty cycle [%].

For example if the selected frequency is 1 Hz, selected device is XMC4700, required resolution is 10 ns, and duty cycle is 90%, following settings should be applied as an input to period and compare register:

```
================================================================================
PRESCALER INITIAL VALUE FOR ALL APPS = 1 Tick resolution = 6.94444444444e-09s
--------------------------------------------------------------------------------
CCU4_SLICE_CONFIG_0 period value = 35156(0x8954)
CCU4_SLICE_CONFIG_1 period value = 4095(0xfff)
--------------------------------------------------------------------------------
CCU4_SLICE_CONFIG_0 compare value = 21503(0x53ff)
CCU4_SLICE_CONFIG_1 compare value = 409(0x199)
```

In this example values for the duty cycle in the range of 10% to 90% are already calculated and written in “compare_values” array.

Finally, it should be noted that high precision using very low values of tick resolution can still be achieved even with the timer concatenation. The reason behind it is the hardware design of the compare and period match events. This means that even small changes to the CCU4_SLICE_CONFIG_0 period/compare register value will have the certain impact on the final output signal. For more details regarding the way compare and period match events are handled in case of timer concatenation please refer to following figure.

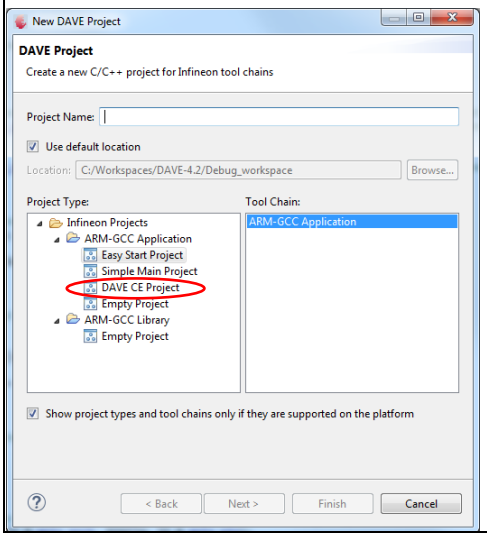
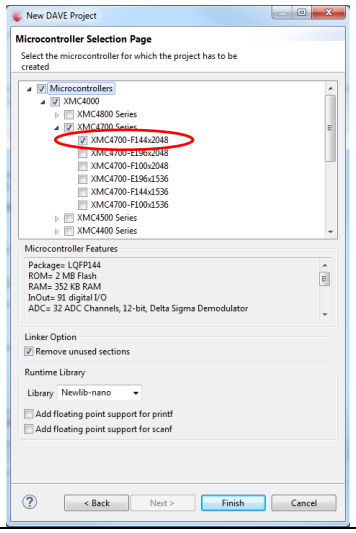


REQUIRMENTS:

Boards required: XMC4700/XMC4800 Relax Kit

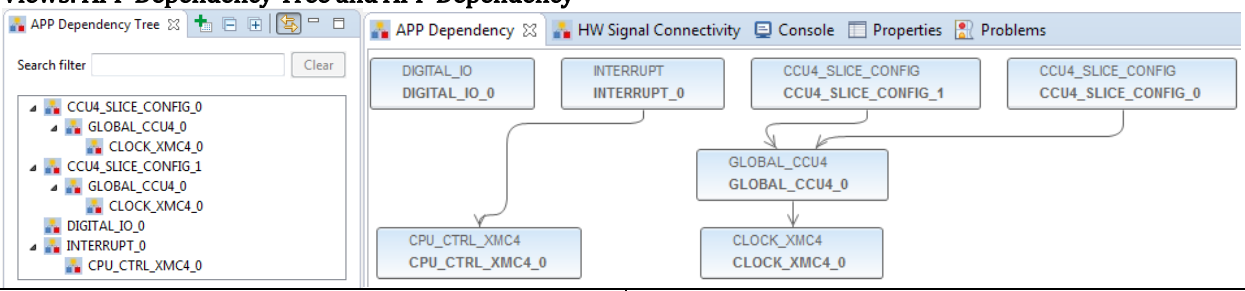
HOW TO CREATE THE PROJECT:

1. Open the DAVE CE and use  "Add IDE New Project Wizards" found in the toolbar to add a new DAVE project.

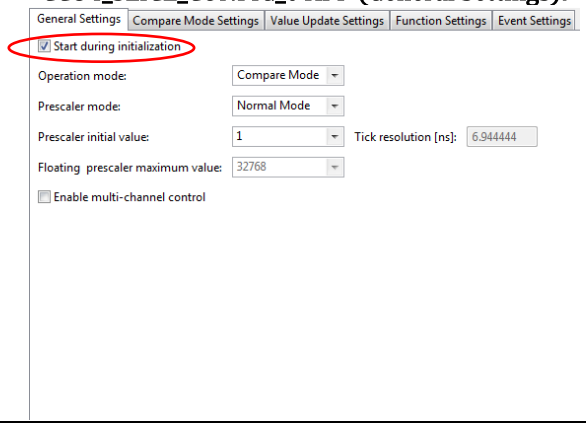
Enter project name Select "Project type → DAVE CE Project"	Select "Microcontrollers → XMC4000 → XMC4700/XMC4800 Series → XMC4700-F144x2048/ XMC4800-F144x2048
	

2. Use the  "Add New APP" found in the toolbar to add two (2) instances of CCU4_SLICE_CONFIG APP, one instance of DIGITAL_IO APP, and one instance of INTERRUPT APP. Configure the APP instances with the following configurations.

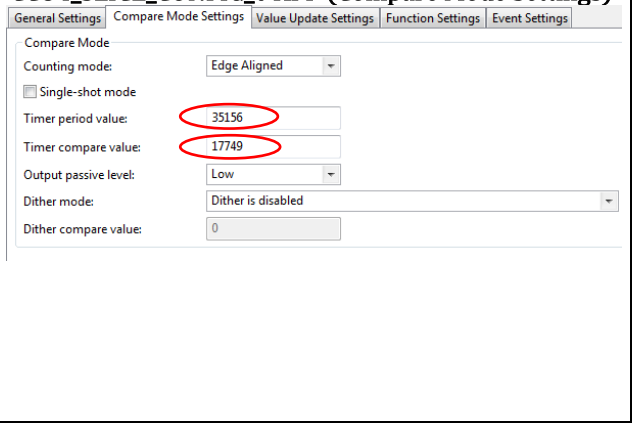
Views: APP Dependency Tree and APP Dependency



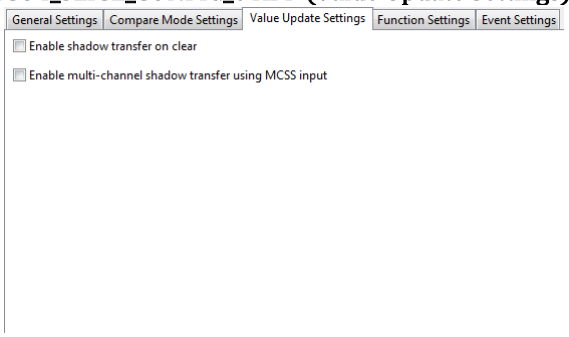
CCU4_SLICE_CONFIG_0 APP (General Settings):



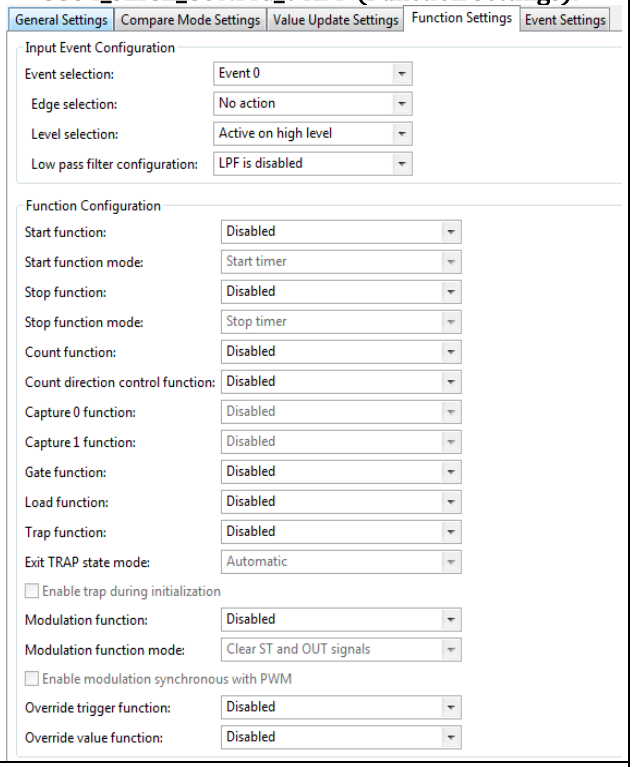
CCU4_SLICE_CONFIG_0 APP (Compare Mode Settings)



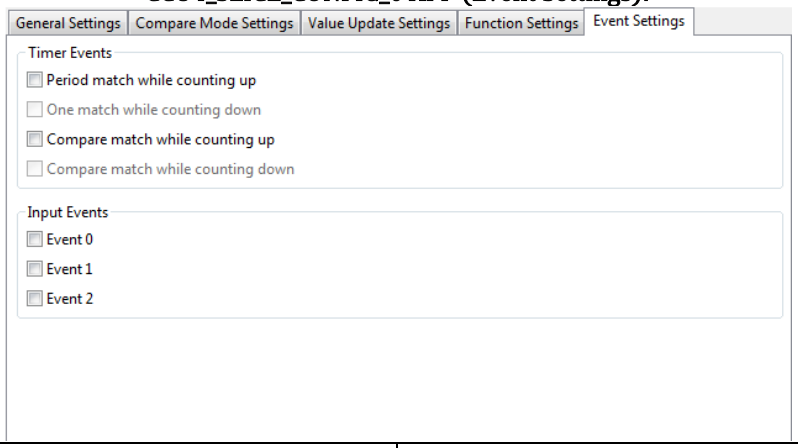
CCU4_SLICE_CONFIG_0 APP (Value Update Settings):



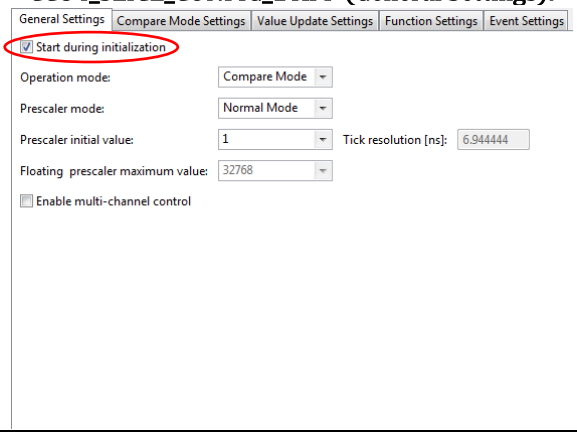
CCU4_SLICE_CONFIG_0 APP (Function Settings):



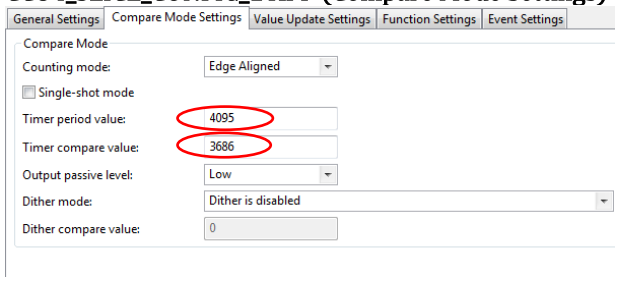
CCU4_SLICE_CONFIG_0 APP (Event Settings):



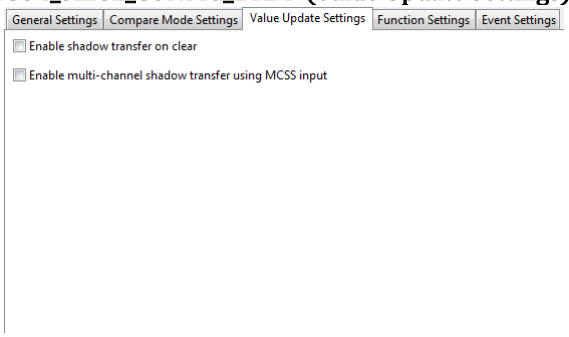
CCU4_SLICE_CONFIG_1 APP (General Settings):



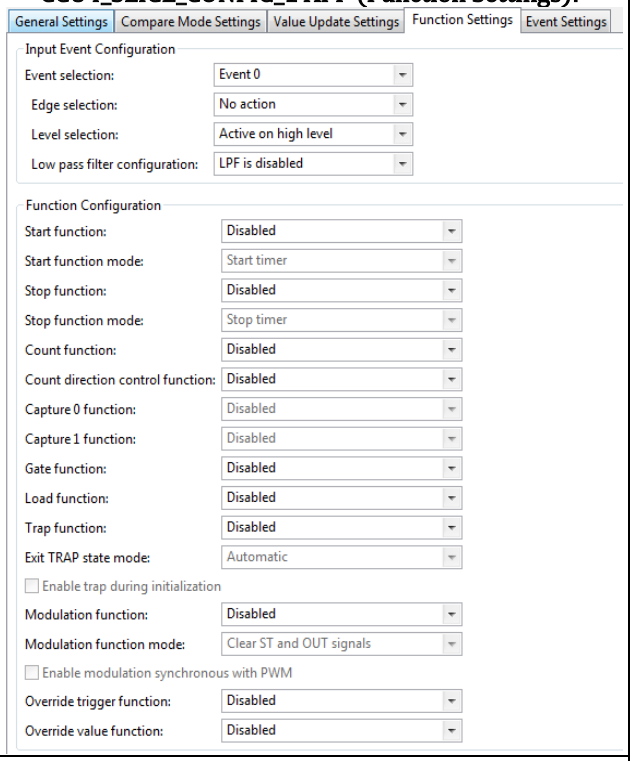
CCU4_SLICE_CONFIG_1 APP (Compare Mode Settings)



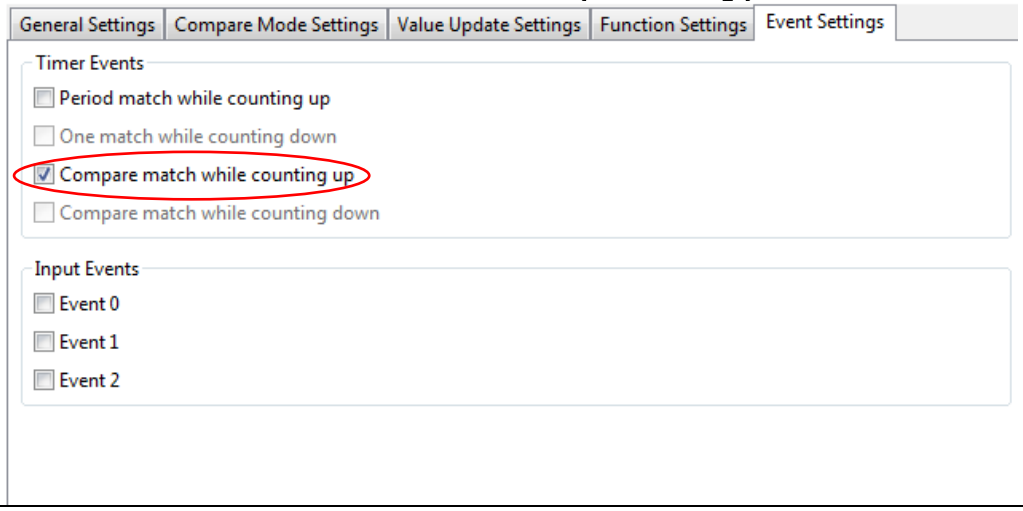
CCU4_SLICE_CONFIG_1 APP (Value Update Settings):



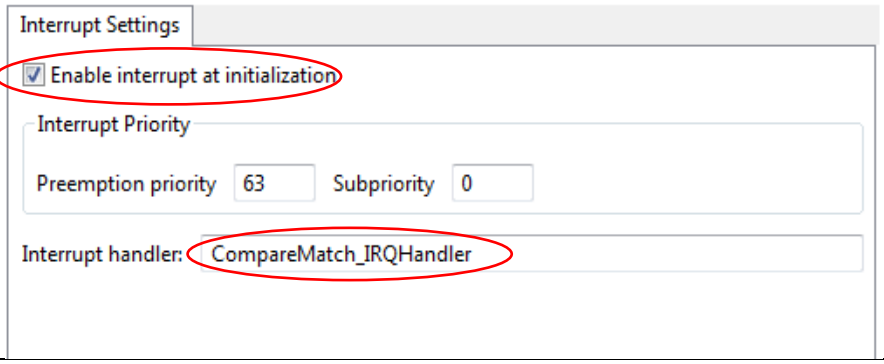
CCU4_SLICE_CONFIG_1 APP (Function Settings):



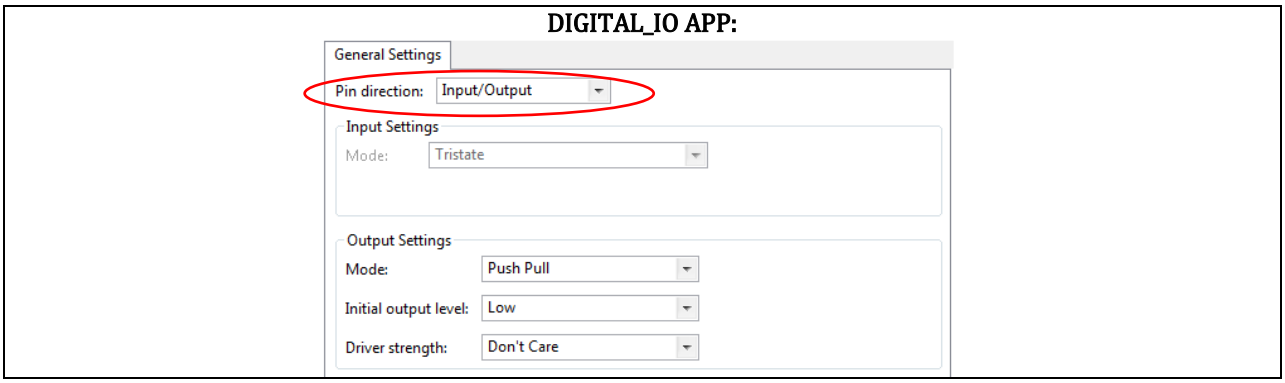
CCU4_SLICE_CONFIG_1 APP (Event Settings):



INTERRUPT APP:

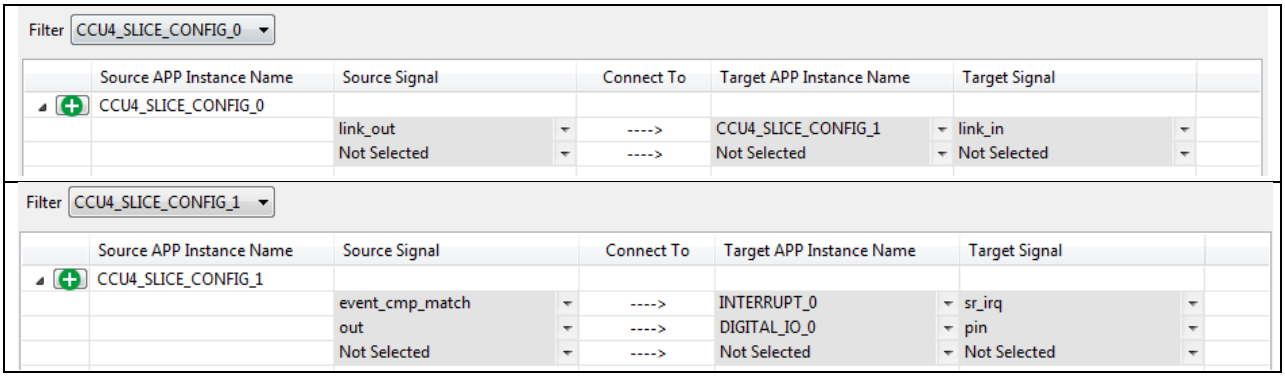


DIGITAL_IO APP:

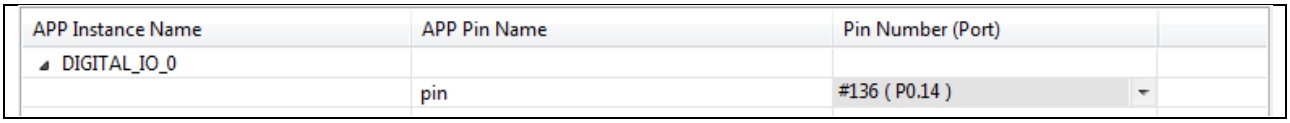


3. Use the "HW Signal Connections" (right click on the APP in App Dependency view and select "HW Signal Connections") to configure the signal connections for both CCU4_SLICE_CONFIG APPs.

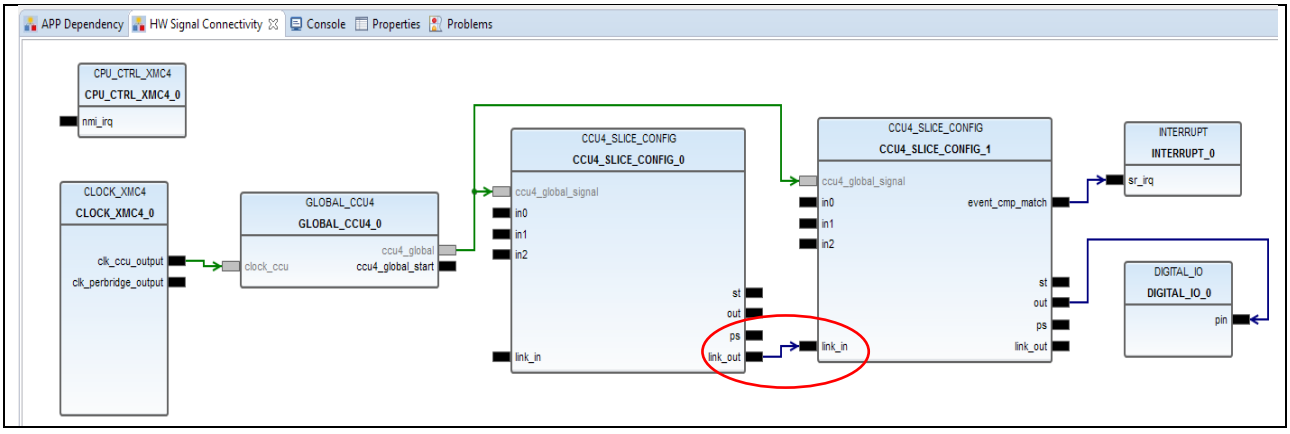
   To use the timer concatenation it is necessary to connect "link_out" signal of the CCU4_SLICE_CONFIG_0 APP to the "link_in" signal of the CCU4_SLICE_CONFIG_1 APP.



4. Use the  "Manual Pin Allocator" (found in the toolbar) to configure the pin for the DIGITAL_IO APP.



5. Verify that "HW Signal Connectivity" view (located next to the "APP Dependency" view) resembles the following figure. Note also the visual representation of the timer concatenation in red circle.



6. Generate the code for the configurations made using  and change “main.c” accordingly.
7. Build () and download to the microcontroller ().

HOW TO TEST:
Download and run the demo in the microcontroller.

OBSERVATIONS:
Connect the oscilloscope probe to the pin 0.14 on the XMC4700/XMC4800 Relax Kit. Generated signal will continuously change the duty cycle from 10-90% by increments of 10%.

HINTS HOW TO MIGRATE EXAMPLE TO OTHER DEVICES:
Applicable to every XMC device. Select other XMC device as shown in step 1 and follow additional steps. Additionally, values given as a period and compare timer register values needs to be recalculated while they depend on the tick resolution/CPU frequency (144 MHz for the XMC4700/XMC4800 microcontrollers). For this purpose, provided Python script can be used.

EXAMPLE PROJECT LIMITATIONS/KNOWN ISSUES:
Not applicable

REFERENCES:
XMC4700/XMC4800 Reference Manual