Frequency Measurement With Counter in PSoC® 3 / PSoC 5

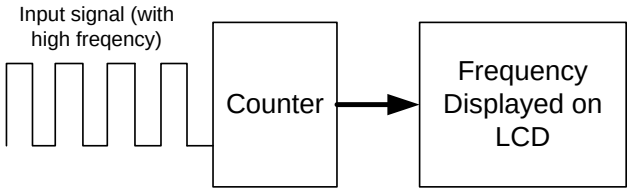
Project Objective

This project demonstrates how to use the counter in PSoC® 3 / PSoC 5 to perform a high-frequency measurement of square waves.

Overview

Frequency measurement is the number of rising edges that occur within one second. To find this number, this project uses a time window and determines the number of counts (rising edges) within that time window. This count is used to find the number of rising edges in one second. The project uses a PWM to generate the time window. The signal whose frequency is to be found is given to the count input of the counter. The count value in the counter within the time window gives a measure of the input signal frequency.

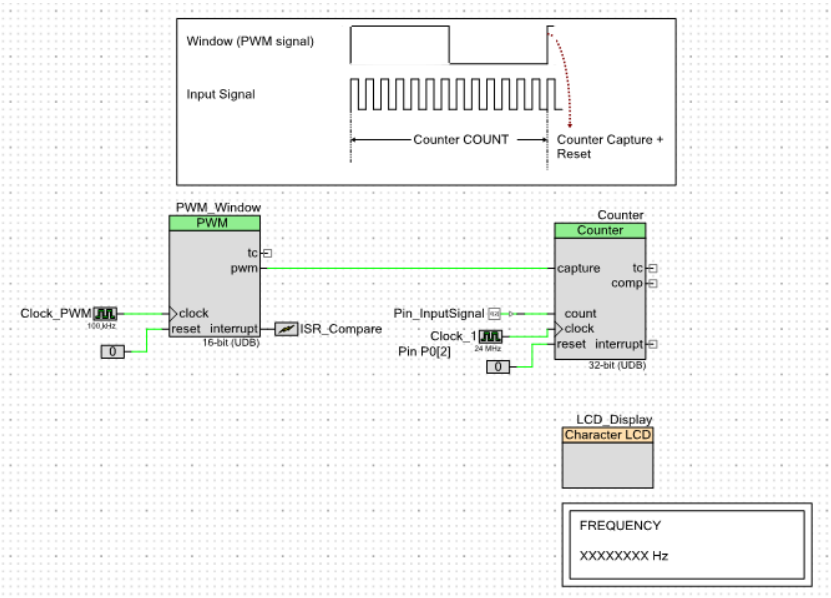
This project can measure only square waves. If you want to measure the frequency of other waveforms, the waveform must be converted to square wave before giving it to the counter.



This project can measure from 20 Hz to 8 MHz. This method is useful in measuring high frequency square waves. For lower frequencies.

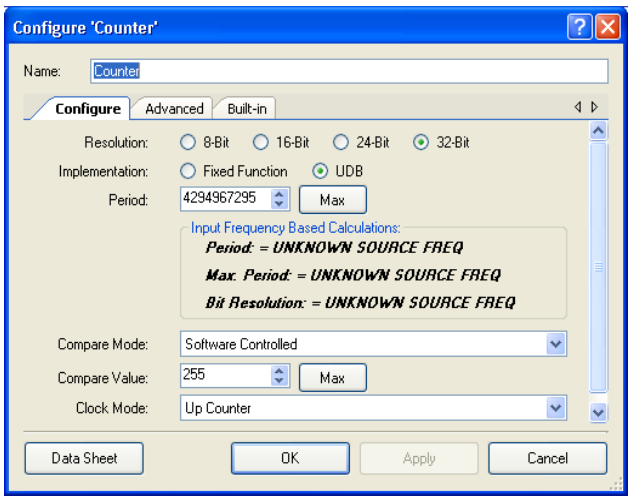
Top Design

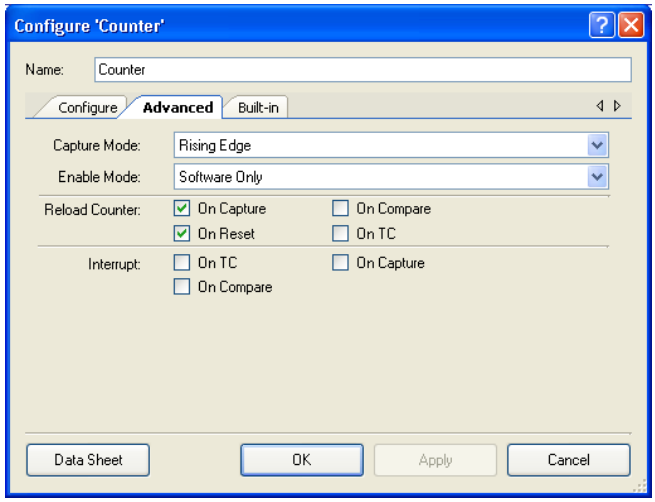
The following figure illustrates the components and their routing.



Component Configuration

Pin Placement





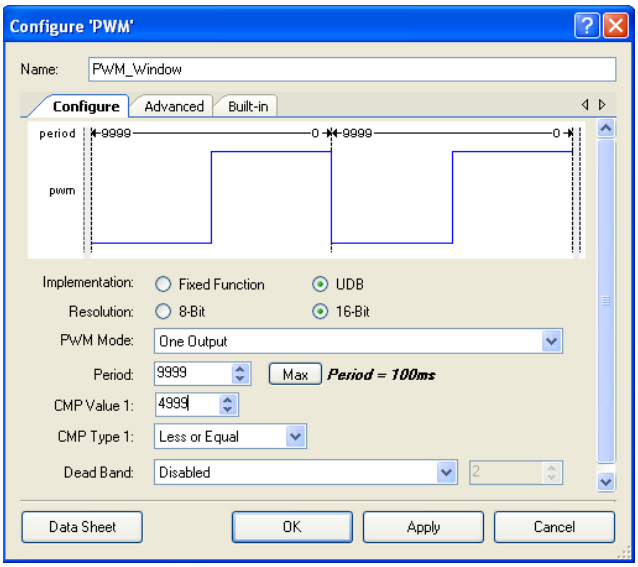
The clock mode of the counter is set to “Up Counter”. The counter component has two inputs: “count” and “clock”. The signal whose input frequency is to be measured is connected to “count” input and count input is synchronized to clock input, which is set to 24 MHz.. The counter is configured as an up counter because counting from 0 makes the calculation easier. When the counter is configured as a down counter, to get the exact count value, decrement the captured value from the counter period before proceeding with calculations.

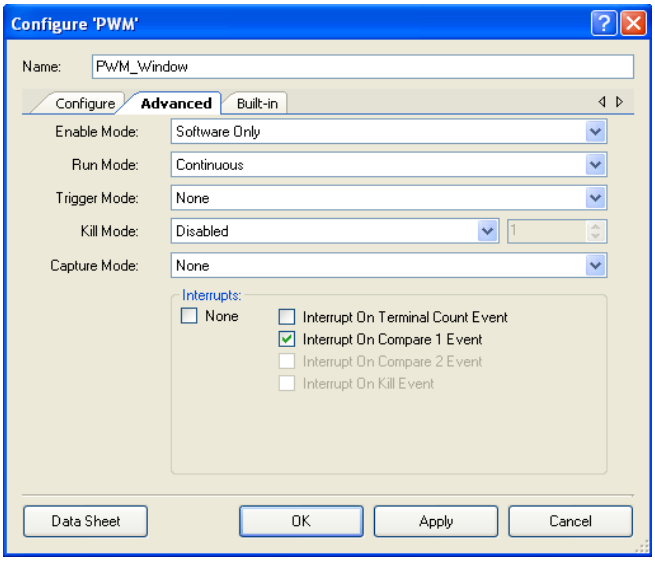
- Up counter: Counts for 100 ms time window = Capture value
- Down counter: Counts for 100 ms time window = Counter period – Capture value

The **Capture Mode** is configured for **Rising Edge**. This is used to detect the time window, which is generated by the PWM signal.

Reload Counter is set to **On Capture** and **On Reset**. The counter reloads when the capture is detected, and fresh count starts for the next time window. The counter is also reloaded on reset to prevent any false counts outside the time window.

PWM Window



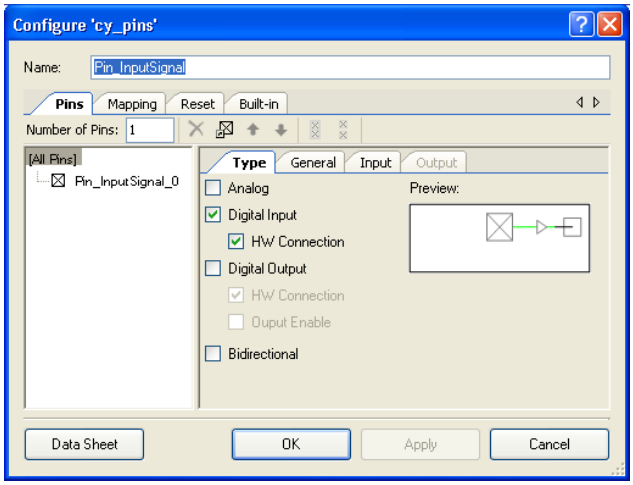


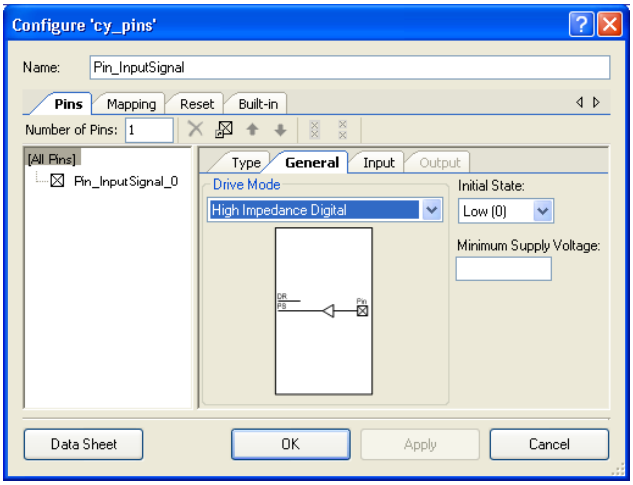
The PWM is designed using UDB. The Fixed function PWM also can also be used. The PWM output is used to generate the time window required for the count. This PWM output acts as the input to the counter capture signal. The PWM period is configured for 9999 which results in (9999 + 1) * 100 kHz = 100.00 ms.

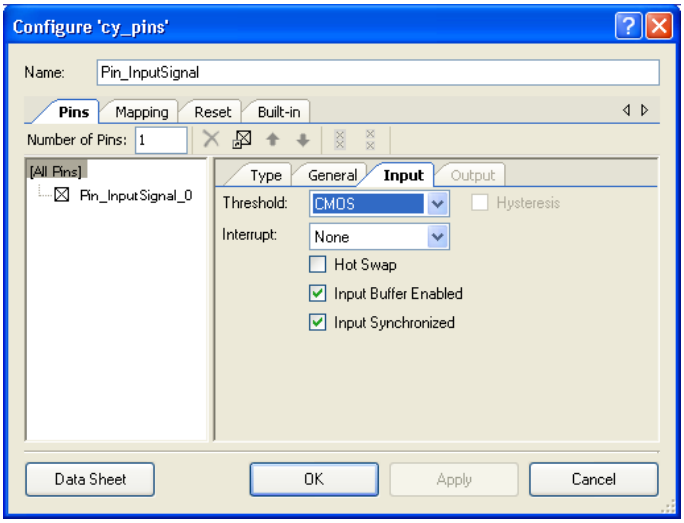
PWM is configured for a compare type of ‘Less or Equal’ and compare value of ‘4999’. This generates a pulse with a duty cycle of 50 percent and time period of 100 ms.

- The time window generated by PWM = (9999 (period configured) + 1 (1 cycle for the period to count to 0) = 10000 * 100 kHz = 100 ms

Pin Input Signal







The **Pins** tab shows that the pin is a digital input. Because the pin is controlled by hardware, HW Connection is enabled. If HW Connection is disabled, the pin can be read only by firmware. The **General** tab shows that the pin is configured for High Impedance Digital drive mode.

Operation

This project uses PWM and Counter components.

- The input signal whose frequency is to be measured is given as the “count” input to counter. The counter increments for every rising edge at the input.

  Note This project can be used only for measurement of square wave inputs. When the input is of some other waveform that does not have a rising edge (for example, sine wave or ramp), you must convert the input wave to square wave before giving it to the counter. The other waveforms can be converted to square wave using a comparator.

- The counter count is observed after every time window.
- The time window is generated using a PWM.

- The following steps show the operation of the PWM:
  1. PWM is started.
  2. PWM begins to count from 9999 (period of PWM).
  3. PWM output is configured for a compare value of '4999' and a compare type of 'Less than or Equal'.
  4. When the count reaches 4999, the compare type is 'True'. Therefore, PWM output generates a high level.
  5. The PWM continues with the high level until the count reaches '0'.
  6. When the count reaches '0' (terminal count), the count register is reloaded and PWM starts counting from 9999. This generates a PWM output that has a pulse width of 5000 PWM clock cycles = 5000 * 100 kHz = 50 ms.
- The project requires a 50-ms pulse for the capture signal for the following reasons:
  - The 50 ms pulse is given to the capture signal of the counter.
  - When the rising edge is encountered at the capture signal, the counter captures the count value and reloads the counter (configured in the counter).
  - A proper capture occurs only when there are at least two rising edges in the Count input when the Capture input is high. (If this is limited to 1 rising edge, the asynchronous occurrence of the Capture input with respect to the Count input might result in missing the Capture signal.) Because of this, when the capture input is high for 50 ms, you can measure frequencies down to 20 Hz (period = 50 ms). Therefore, this project measures frequencies down to 20 Hz.
  - The upper limit of the frequency is set to 8 MHz.
- The Counter Capture value is the number of counts within 100 ms (time window).
- This count value is used to calculate the number of counts per second.
- This count gives the frequency of the input signal. The frequency is calculated in Hz.
- The calculated frequency is displayed on the LCD in hexadecimal.
- Frequency of calculation:
  - The calculation is performed whenever the Compare event occurs. An interrupt is generated in the PWM for every Compare event.
  - The interrupt occurs every 100 ms.
  - The frequency calculation done after every interrupt helps to update the frequency for every 100 ms.
- The ISR_Compare code sets a flag whenever the interrupt occurs. This flag is polled in *main.c* to do the calculation.
- In *ISR_Compare.c*, as shown in the following code, **PWM_Window_ReadStatusRegister** is called to clear the Interrupt bit of the PWM. If the interrupt bit is not cleared, the interrupt will occur continuously:
-

```
void ISR_Compare_Interrupt(void)
{
        /* `#START ISR_Compare_Interrupt` */
        /* Call the ReadStatusRegister to clear the Interrupt Status bit */
      PWM_Window_ReadStatusRegister();
      /* Set the flag to indicate occurence of the interrupt */
      compare_occured = 1;
      /* `#END` */
}
```

  - Refer to the code in *main.c* for the frequency calculation.

Hardware Connections

The project is tested with PSoC Development Kit CY8CKIT-001.

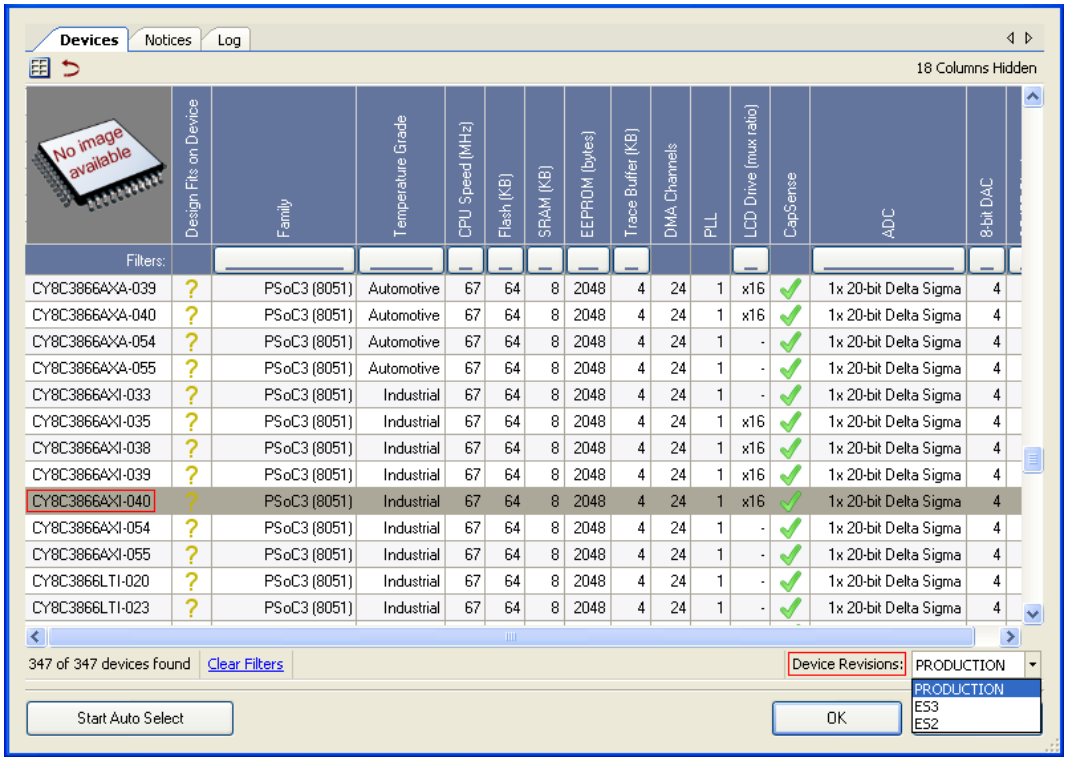
To use the kit with the jumpers in their default state, refer to the *PSoC Development Kit Board Guide* supplied with the kit:

1. Place the Character LCD on P18 of the DVK.
2. Enable the LCD power by placing the jumper J12 to the ON position.
3. Connect the function generator output to Pin ‘P0[0]’ available in the P19 header in the DVK. This is the input to the counter.
4. Set the function generator as follows:
    - Load impedance: Hi Z
    - Frequency: Frequency from 20Hz to 8 MHz
    - Voltage levels:
        - Low level – 0 V
        - High level – V_{ddd} (V_{ddd} is the limit set in the DVK for the Digital Voltage)

Output

Use the device selector window (**Project** > **Device Selector**) in PSoC Creator to select the appropriate device and device revision.

If you are using a PSoC3 device with production revision (such as CY8C3866AXI-040), use the following selection.



Similarly, select appropriate device number to work with PSoC5 Device family (such as CY8C5588AXI-060).

Note For engineering samples, device revision is marked on the package as part of the device number. Production silicon does not have ES marking.

Build the project and program the PSoC 3 device. The project is designed to display the input frequency as a hexadecimal number in Hz.

Set the frequency in the Function generator to 1 MHz. The LCD display should show nearly 0x F4240 (the hex value for 1 MHz).

Change the frequencies between 20 Hz and 8 MHz to see the frequency display on the LCD.