# Programming Counter/Timer Tasks for DriverLINX Supported Boards in LabVIEW

Below is shown a typical wiring diagram from LabVIEW for a counter/timer (CT) task for a DriverLINX supported board.

A bundle is used to control which CT task type (Square Wave Generation, Event Counting, Frequency Measurement, Pulse Generation, etc.) will be performed as well as the timing parameters of the task (frequency of the Square Wave, etc.).
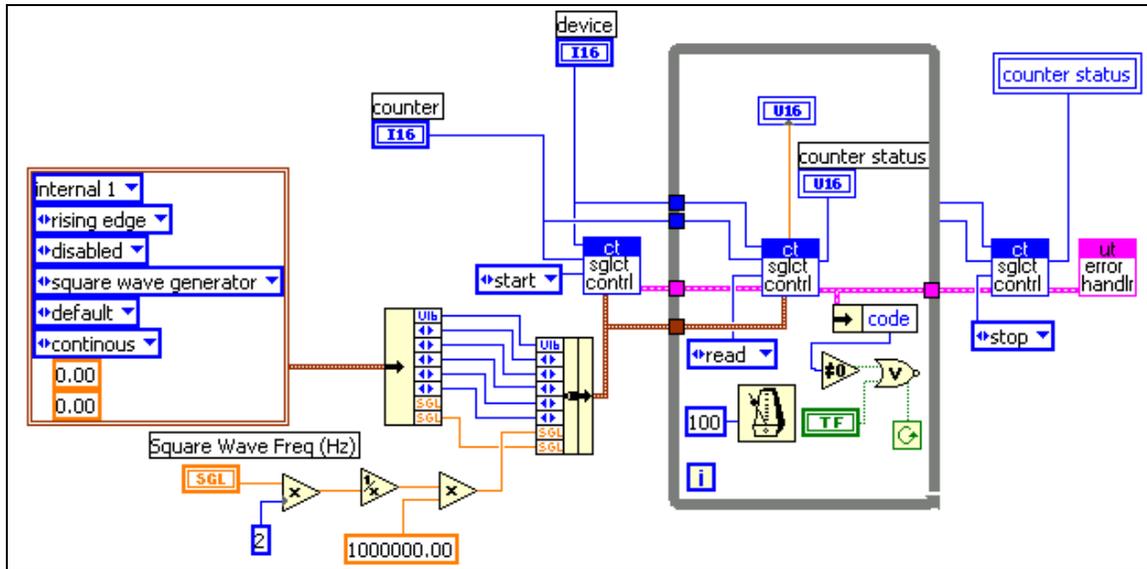
The names of the eight values within the bundle are, in top to bottom order: clock, clock edge, gate, mode, output, pulses, period and onCount.

Clock: pacing source either internal or external
Clock Edge: rising or falling edge of Clock (not used)
Gate: enable or disable gating. If enabled, select level or edge gating
Mode: one of the Task based CT operations implemented by DriverLINX
Output: behavior of the output pin of the selected CT channel
Pulses: either single or continuous
Period: controls timing. Mode dependent. In LabVIEW, is expressed in microseconds.
OnCount: controls timing of delays or pulse duration. Mode dependent. In LabVIEW, is expressed in microseconds.

One typical technique used is to unbundled and then re-bundle the information. As the wiring diagram above shows, this allows placement of a single, more intuitive control on the application's front panel (Square Wave Freq). The user can input the desired frequency of the square wave on the front panel. The scaling for time expressed in microseconds required by the VIs implementation can be done on the wiring diagram.

With the exception of clock edge, these values correspond to the ActiveX or DLL API of the DriverLINX driver that would be used from Visual Basic or C/C++ languages as shown below.

VB code:

```
With DriverLINXSR1
.Evt_Tim_rateClock = clock
' clock edge of LV bundle has no corresponding property
.Evt_Tim_rateGate = gate
.Evt_Tim_rateMode = mode
.Evt_Tim_rateOutput = output
.Evt_Tim_ratePulses = pulses
.Evt_Tim_ratePeriod = Period
.Evt_Tim_rateOnCount = onCount
End With
```

C/C++ code:

```
m_pSR->timing.u.rateEvent.clock = clock;
// clock edge of LV bundle has no corresponding member
m_pSR->timing.u.rateEvent.gate = gate;
m_pSR->timing.u.rateEvent.mode = mode;
m_pSR->timing.u.rateEvent.pulses = pulses;
m_pSR->timing.u.rateEvent.pulses |= OUTPUTHITOGGLED;   // output
m_pSR->timing.u.rateEvent.period = period;
m_pSR->timing.u.rateEvent.onCount = onCount;
```

NOTE: These VB and C/C++ code fragments do not show all the necessary code. For example, opening the driver, closing the driver and hardware initialization steps are not shown.

If a functional VB or C++ example exists for a given mode of operation for a board, then that code could be used as a guideline for how the bundle in the LabVIEW program should be configured to carry out the same operation.