Report Number: WUCSE-2002-25

2002-08-14

Usage of the Statistics Counter Plus Component in Networking Hardware Modules

Michael Attig and John W. Lockwood

The Statistics Counter Plus is a more generalized version of the Statistics Counter circuit developed earlier this year. When an event occurs multiple times, the user may hold the increment event signal asserted while the event occurs. As before, 256 separate events can be tracked.

2002-25

Usage of the Statistics Counter Plus Component in Networking Hardware Modules

Authors: Attig, Michael; Lockwood, J. W.

August 14, 2002

Abstract: The Statistics Counter Plus is a more generalized version of the Statistics Counter circuit developed earlier this year. When an event occurs multiple times, the user may hold the increment event signal asserted while the event occurs. As before, 256 separate events can be tracked.

Usage of the Statistics Counter Plus Component in Networking Hardware Modules

Michael Attig
John W. Lockwood

Department of Computer Science
Applied Research Lab
Washington University
1 Brookings Drive, Box 1045
Saint Louis, MO 63130

WUCS-2002-25
`http://www.arl.wustl.edu/arl/projects/fpx`

August 14, 2002

Abstract

The Statistics Counter Plus is a more generalized version of the Statistics Counter circuit developed earlier this year. When an event occurs multiple times, the user may hold the increment event signal asserted while the event occurs. As before, 256 separate events can be tracked.

1 Introduction

The Field programmable Port Extender (FPX) [1][2] is a reconfigurable network platform that uses control cells to read and write to memory. To manage and debug a complex networking module, a way to count events is necessary. The Statistics Counter Plus is a component that can record and store up to 256 events. The Statistics Counter Plus provides more generality to the user than the original statistics counter [3]. When the same event occurs multiple times in succession, the user is able to hold the increment request signal asserted indefinitely. The Statistics Counter Plus will correctly count the number of events, and it will update the counter number held in block RAM.

2 Interface

The interface, shown in Figure 1, of the Statistics Counter Plus is identical to the Statistics Counter, aside from re-named input signals. The Statistics Counter Plus is ready to begin tracking events once *cntr_ready* is asserted.

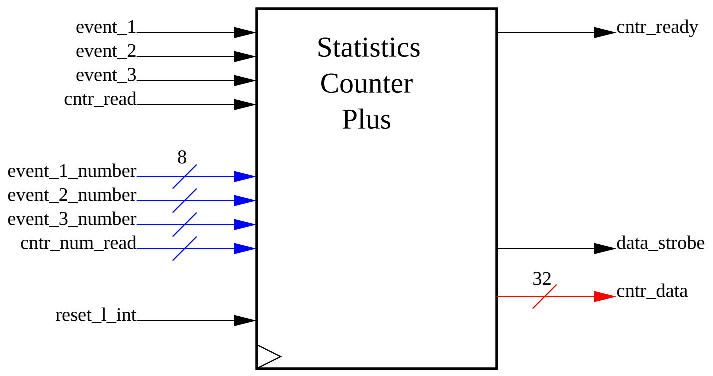

Figure 1: The Statistics Counter Plus interface consists of three event increment requests, a read request, three event counter numbers, a read counter number, a counter ready signal, a data arrival signal, and a counter data signal.

Three separate events can be tracked simultaneously. This is accomplished by asserting the appropriate *event_#* signal and placing the event number on the corresponding *event_#_number*. An event number can be read as well. Simply pulse the *cntr_read* signal and place the event number to be read on *cntr_num_read*. Three to six clock ticks after the *cntr_read* pulse, the appropriate data will be returned on *cntr_data* accompanied by a *data_strobe* pulse. An event number may only be used on a single *event_#_number*. If the same event number is used on multiple event lines, the Statistics Counter Plus will not function correctly.

2.1 Input Specifications

The signals *event_1*, *event_2*, and *event_3* are used to record an event. In order to do so, the *event_#* signal is asserted high and the corresponding 8-bit event number is placed on *event_#_number*. The *event_#_number* needs to remain valid while *event_#* is asserted. *Event_1*, *event_2*, and *event_3* can be asserted at any time relative to each other. However, a spacing of four clock cycles needs to separate events on the same *event_#* signals. To switch events on an *event_#_number*, *event_#* must be de-asserted for four clock cycles. Refer to Figure 2 for a timing diagram showing common usage.

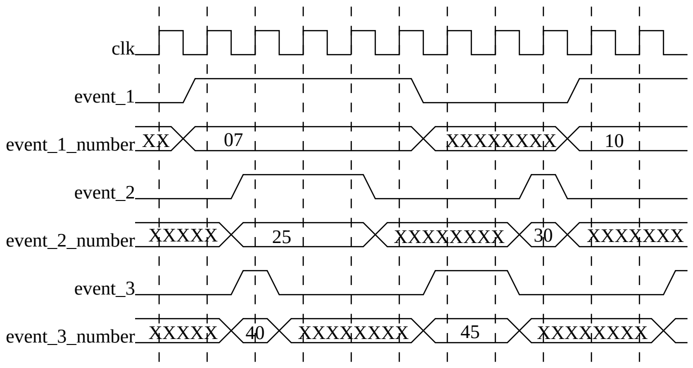

Figure 2: Five events occur for event number x07. To record these, *event_1* is asserted for 5 clock cycles, and the event number is placed on *event_1_number*. Four cycles after *event_1* is de-asserted, *event_1* is asserted again. This time it is recording event number x10. Note that *event_1*, *event_2*, and *event_3* are independent. They can be asserted at any time relative to each other.

The signal *cntr_read* must only be pulsed. At the time of the pulse, the event number to be read must be placed on *cntr_num_read*. Again, a separation of four clock cycles must be between successive read pulses. Refer to Figure 3 for a timing diagram showing proper usage.

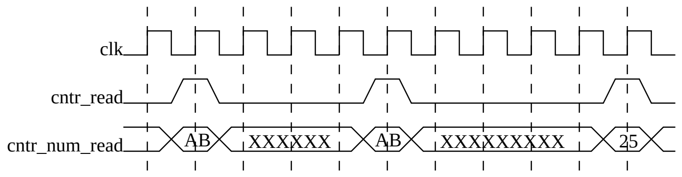

Figure 3: *Cntr_read* pulses must have a minimum separation of four clock cycles. Note that the first two pulses, although reading the same event number, still must have a separation of four clock cycles.

2.2 Output Specifications

Between 3 and 6 clock cycles after a *cntr_read* pulse, the counter value will be returned. The arrival of data on *cntr_data* is signaled by a *data_strobe* pulse. The value on *cntr_data* is only valid during the *data_strobe* pulse. The waveform in Figure 4 shows the best-case and worst-case latency.

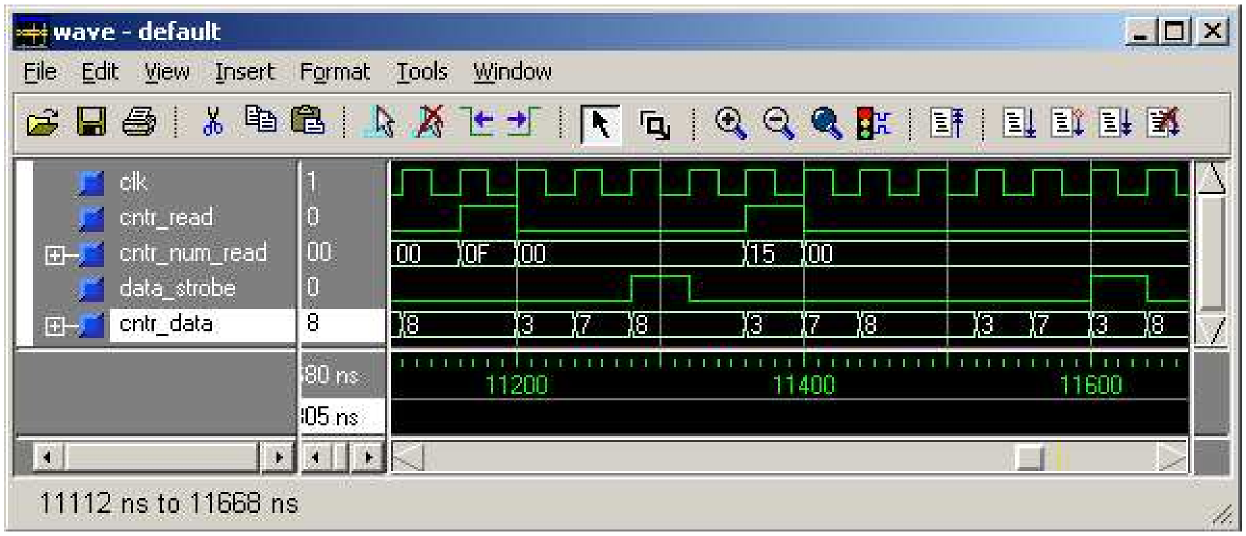

Figure 4: The number of clock cycles from a *cntr_read* pulse to a *data_strobe* pulse varies from three to six. This waveform shows the three cycle and six cycle latency.

3 Block Diagram

The Statistics Counter Plus has three more components than the original statistics counter. The event requestor is responsible for interfacing with the old statistics counter in that it pulses the increment request signals at the appropriate time. Three 3-bit flip-flops have been added to store the amount to add to the counter value. The 32-bit increment has been replaced by an adder which adds a 32-bit number and 3-bit number. These new components can be seen in Figure 5.

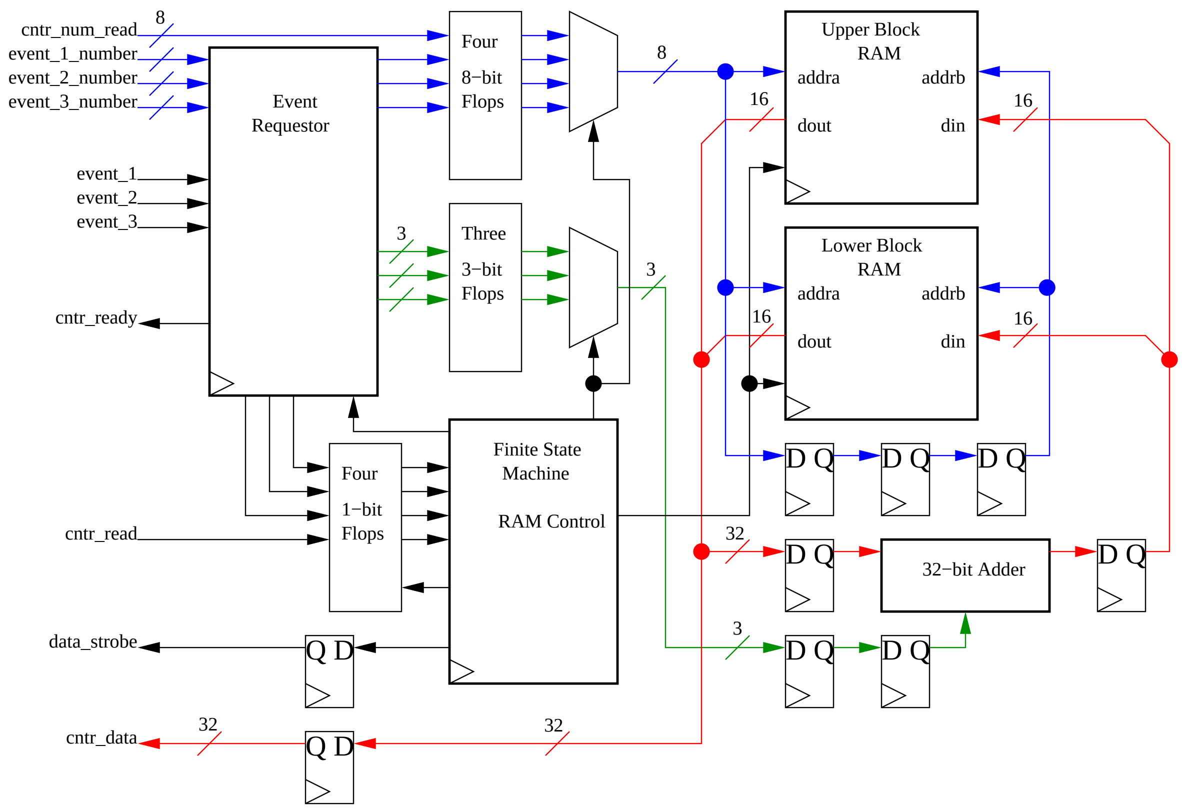

Figure 5: A block diagram of the Statistics Counter Plus. Three 3-bit flip-flops, an event requestor, and a 3-bit adder have been added to create the Statistics Counter Plus.

The event requestor counts the number of events on each *event_#* signal, and it maintains a count until it can update the value in block memory. During an increment request pulse, the internal counter is reset either to zero or one. It is reset to zero if the corresponding *event_#* signal is de-asserted and one if asserted. When the event requestor makes an increment request, it supplies the amount to add and the event number that is to be updated.

4 Example Usage

The Statistics Counter Plus is currently being used in the fast CCP [4] to count four different types of events. The CCP uses the Statistics Counter Plus to count the number of cells arriving on each VCI, the number of SRAM reads on each VCI, the number of SRAM writes on each VCI, and the total number of control cells that pass through the module. The fast CCP uses the lower 6-bits of the VCI, and sets bits 7 and 6 to ”00” for counting a cell, ”01” for reads, and ”10” for writes. When counting the total number of cells, event number 1100000b is used.

5 Code Location

The vhdl source code and documentation for this module can be found in the CVS tree at: /project/arl/fpx/cvsroot/FPX_ROOT/RAD/INFRASTRUCTURE/STAT_MOD_PLUS

6 Conclusion

The Statistics Counter Plus is capable of tracking up to 256 distinct events. As opposed to the old interface in which you could only increment an event every four clock cycles, the Statistics Counter Plus allows the user to keep the increment request signal asserted as long as the event is occurring. However, if a different event is to be tracked on the same increment request line, a four clock cycle delay is still necessary to separate the two events. The Statistics Counter Plus is a useful circuit when tracking events that occur more than once in a span of four clock cycles.

References

[1] J. W. Lockwood, J. S. Turner, and D. E. Taylor, “Field programmable port extender (FPX) for distributed routing and queuing,” in *ACM International Symposium on Field Programmable Gate Arrays (FPGA’2000)*, (Monterey, CA, USA), pp. 137–144, Feb. 2000.

[2] J. W. Lockwood, N. Naufel, J. S. Turner, and D. E. Taylor, “Reprogrammable Network Packet Processing on the Field Programmable Port Extender (FPX),” in *ACM International Symposium on Field Programmable Gate Arrays (FPGA’2001)*, (Monterey, CA, USA), pp. 87–93, Feb. 2001.

[3] M. E. Attig and J. W. Lockwood, “Statistic Counter for Networking Hardware Modules,” tech. rep., WUCS-02-20, Washington University, Department of Computer Science, July 2002.

[4] M. E. Attig and J. W. Lockwood, “Implementation of a Pipelined Control Cell Processor,” tech. rep., WUCS-02-24, Washington University, Department of Computer Science, Aug. 2002.