An FPGA-based module for multiphoton coincidence counting

David Branning*[a], Mark Beck[b]
[a]Dept. of Physics, Trinity College, 300 Summit St., Hartford, CT 06106;
[b]Dept. of Physics, Whitman College, Walla Walla WA, 99362

ABSTRACT

We present a multi-channel coincidence-counting module for use in quantum optics experiments. The circuit takes up to four TTL pulse inputs and counts either 2-, 3-, or 4-fold coincidences, within a user-selected coincidence time window as short as 12 ns. The module can accurately count eight sets of multi-channel coincidences, for input rates of up to 84 MHz. Due to their low cost and small size, multiple modules can easily be combined to count arbitrary *M*-order coincidences among *N* inputs.

Keywords: coincidence counting, correlation measurements, multiphoton

1. INTRODUCTION

Coincidence counting is the simultaneous detection of two or more particles at different detectors. While this technique is widely used in experimental physics, it plays an especially important role in quantum optics. The coincidence counting of photons is an essential tool for exploring and/or exploiting the nonclassical features of correlated light sources. Many such experiments require only sets of two-fold coincidence measurements, while for others it is necessary to count multi-photon coincidences among many detectors.[1-3]

Historically, the most common method of coincidence counting has used Time-to-Amplitude Converters (TACs), with each TAC adding the capability to count one more pair of photons in coincidence. Multi-photon, or multi-channel coincidence-counting quickly becomes cumbersome and expensive this way, and the maximum coincidence-counting rate is limited by the conversion time required for each start/stop event, typically ~1 μs. In recent years several solutions to these problems have evolved for particular applications, including quantum information processing,[4-6] fluorescence measurements,[7-8] x-ray microscopy,[9] and physics education.[10-11]

Here we present the details of a new multi-channel coincidence-counting module (CCM) that can be built for less than $600 with off-the-shelf integrated circuit components. Starting with up to four TTL signals as inputs, the CCM can register combinations of arbitrary 2-, 3-, or 4-fold coincidences (or single-channel counts), with a coincidence window as short as 12 ns. Eight onboard registers, programmed into a field programmable gate array (FPGA), count the user-defined coincidences for time intervals of between 20 μs and 1 s. The count data is transferred to a personal computer over a USB interface, where the counts are collected, integrated, displayed, and stored to disk via freely-available software.[12-13] Resources for the construction and operation of this CCM, including an assembly guide, an operating manual, and data acquisition software (for LabVIEW or as a standalone executable), may be freely downloaded from our web site.[13]

*david.branning@trincoll.edu; phone 1 860 297-4048

2. DESIGN

2.1 Overview

A block diagram is shown in Figure 1 (a full schematic of the entire circuit is available online).[13]

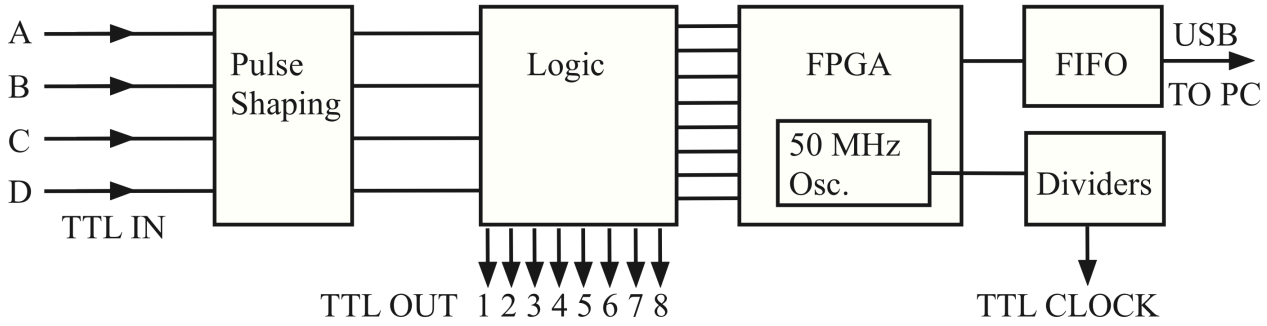

Fig. 1. Block diagram of the CCM architecture.

The inputs A, B, C, and D are 5-volt TTL pulses form single-photon counting modules (SPCMs). Each input to the CCM has a selectable impedance of 50 Ω or 1 kΩ. The input pulses are shortened in duration and then fanned out to form the inputs to eight copies of the coincidence logic circuit in Figure 2. The eight output channels are sent to BNC outputs, and also to the counting registers on the FPGA. The values in the counting registers are sent to the First-In-First-Out (FIFO) buffer where they are bundled into arrays that are read by the computer over a USB interface. Also, the 50 MHz master oscillator of the FPGA is divided down to create a 1 Hz to 10 MHz TTL clock output that can be used to synchronize other equipment or for self-testing of the CCM.

2.2 Coincidence-counting method

The basic method of determining coincidences is shown below. The input signals A, B, C, and D are sent to OR gates, and then to the inputs of a four-way AND gate. The output of the AND gate is true if and only if all four inputs are simultaneously true – that is, if the four detector pulses arrive at the gate at the same time.

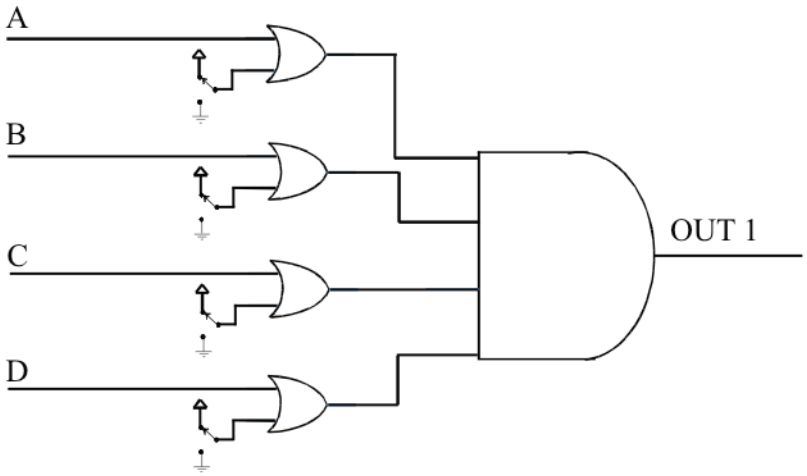

Fig. 2. Four-way AND gate with OR gates on each input. For each input (A, B, C, D) a switch connects one of the OR inputs to 0 or 5 V, so that the input is either included (0 V) or excluded (5 V) from the logic at the AND gate.

The OR gates allow the user to define arbitrary subsets of the four detector signals to be counted in coincidence. The second input of each OR gate is held high or low, as selected by the user with a switch. When the switch for any particular input is high, that input is effectively removed from the coincidence logic. Any inputs with their corresponding switches held low, however, must still arrive simultaneously in order for the AND gate's output to be true. In this fashion, the output of the AND gate can determine any combination of 2-, 3-, or 4-fold coincidences between the four inputs, or simply deliver the single-channel input rate of any one input (by excluding the other three). There are eight 4-input AND gates, and the output of each is sent to the input of a counter, which is implemented on the FPGA. Each counter regularly delivers its recorded number of counts to a PC over a USB interface, and then resets to continue counting.

2.3 Pulse shaping

In order to improve the coincidence time resolution, each of the detector signals first enters a pulse-shaping circuit that reduces its width from the 20-50 ns pulse width typically obtained from commercial SPCMs.[11] A diagram of the pulse-shaping circuit is shown in Figure 3. The pulse shaping is accomplished by using two copies of the same input signal. One copy is time-delayed and inverted with respect to the other copy. Both copies are used as inputs of an AND gate. The output of the AND gate will only be high for the duration of the time delay. The time delays are accomplished by sending the signal through additional gates, e.g., AND gates with one input held high, that delay but do not otherwise alter the signal.

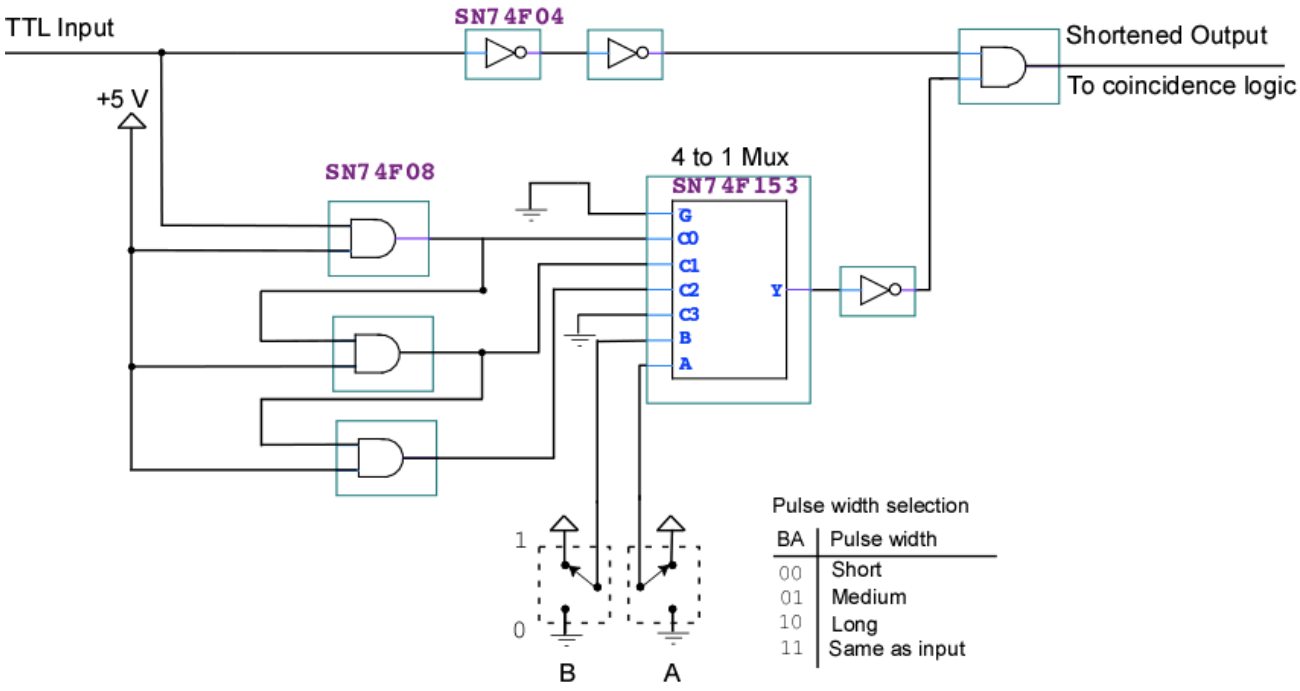

Fig. 3. Pulse-shaping circuit. The input signal (top line) and a time-delayed inverted copy of it are sent to the inputs of an AND gate (at top right). Toggle switches A and B are used with a multiplexer (Mux) to select the width of the shaped pulses of all four inputs, or to bypass the pulse-shaping circuit, leaving the pulse widths largely unchanged.

These manipulations allow for various discrete shortened pulsewidths, to be selected by the user by adjusting the position of two switches (A and B in Figure 3). The pulse-shaping section can also be bypassed, so that the full width of each pulse is passed directly to the logic section.

2.4 TTL channel outputs

In addition to being sent to the FPGA, the output of each 4-way AND gate is also connected to a line driver and a BNC output, providing TTL output pulses which can be monitored or counted externally. By using these output pulses as the inputs to additional CCM's, coincidences among an arbitrarily high number of inputs can be monitored.

2.5 10 MHz Clock output

A TTL clock signal is provided at a BNC output by dividing the FPGA's 50 MHz oscillator down to a user-selectable rate from 10^7 Hz to 1 Hz in decades. Because the TTL clock is derived directly from the master 50 MHz oscillator, it can be easily used to self-test the counting operations of the FPGA: the 10 MHz output should yield precisely 10^7 counts per second with no errors due to lack of synchronization. The 10 MHz clock output can also be used to synchronize other electronic pulse generators with the CCM for testing purposes, or to synchronize other data acquisition equipment in an experiment.

3. IMPLEMENTATION

3.1 F-series logic and FPGA hardware

The circuits of Figures 2 and 3 are implemented using F-series 5V TTL logic gates: these consist of AND gates, OR gates, inverters, multiplexers, and line buffers in the familiar 14-, 16-, or 20- pin DIP packages. The counting registers and USB capabilities are provided by an 80-pin MORPH-IC module from Future Technology Devices International (FTDI), which contains an Altera Acex 1K FPGA and a USB interface with FTDI's FT2232D FIFO buffer.

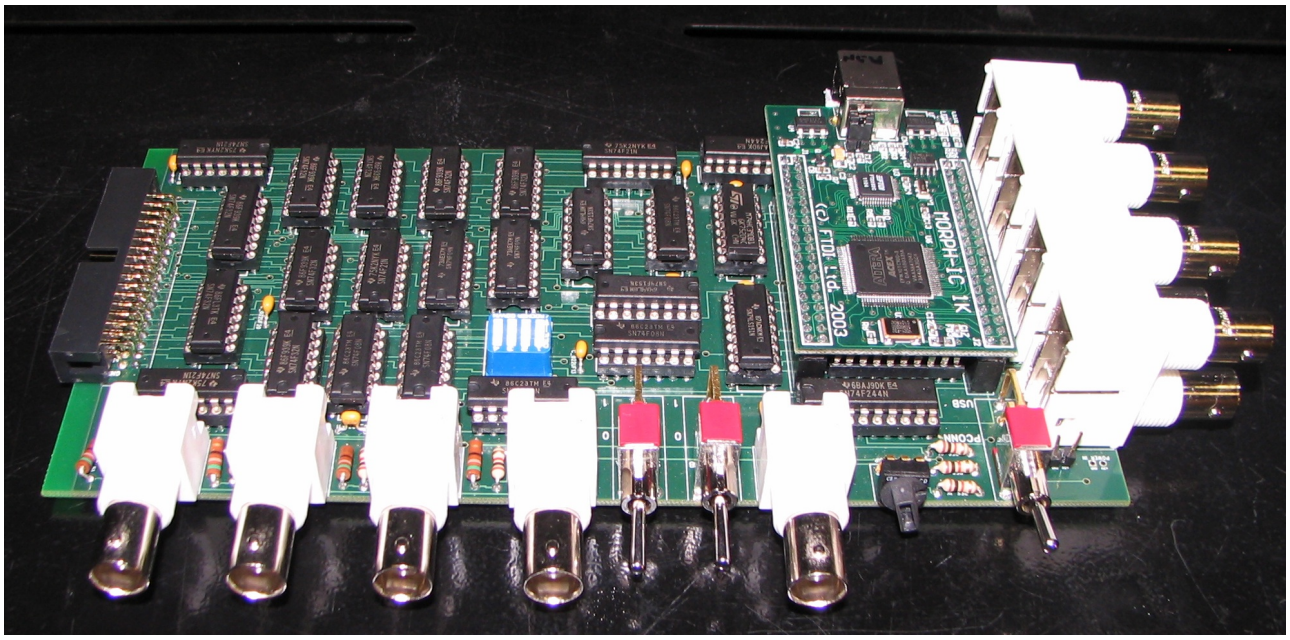

Fig. 4. An assembled logic board. BNC inputs A, B, C, and D are on the lower left. The TTL outputs 1-8 are on the far right. The MORPH-IC module is just to the left of these.

The logic chips and FPGA are mounted on a custom-manufactured 4-layer circuit board as shown in Figure 4. The boards can be manufactured by various online suppliers using the gerber files that are freely available from our web site.[13]

3.2 Pushbutton controls

The switches connected to the OR gates that determine which coincidences are counted are latching pushbuttons, with an embedded orange (590 nm) LED. The switches are double-pole double-throw (DPDT), with one pole used to control the logic and the other used to control the LED. When a switch is depressed, the center pole for the logic is connected to ground and the LED is lit, indicating that the corresponding input is included in the 4-way AND logic.

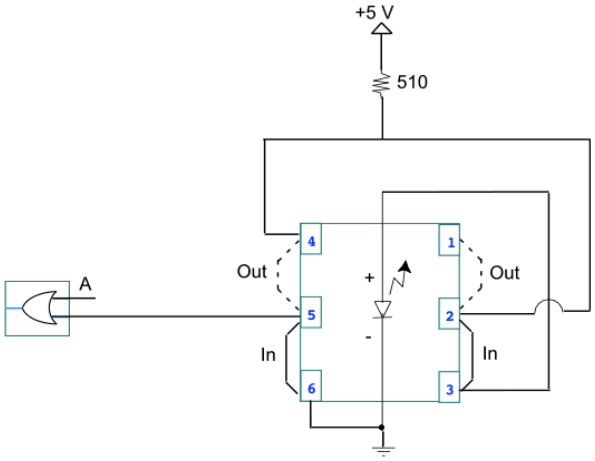

Fig. 5. Pushbutton wiring. When the button is IN, the left center pole (terminal 5) connects one input of the OR gate to ground. This means that input A, at the other OR input, is included in the coincidence circuit (see Fig. 2). To indicate this, the LED is lit by connecting it to +5 V using right terminals 2 and 3. A 510-ohm resistor limits the current through the LED. When the button is OUT, the LED is not lit, and the OR gate terminal is raised to +5 V, removing input A from the coincidence logic.

The switches are arranged in a 4x8 grid as shown in Figure 6. The four rows correspond to the four inputs, and the eight columns correspond to the eight counters. In this way the user can very easily set (and observe) which coincidences are being registered by which counter.

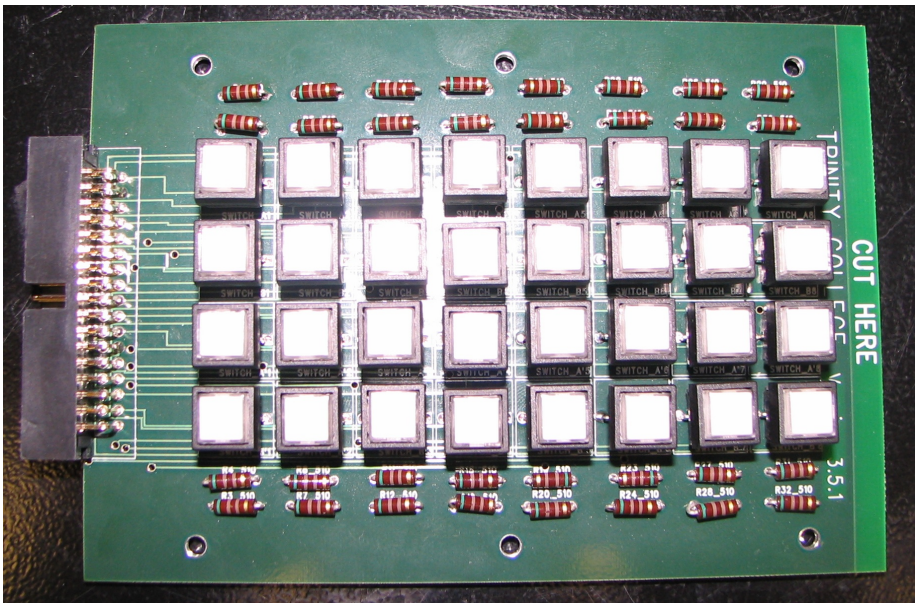

Fig. 6. An assembled control board. The pushbuttons indicate, for each output channel 1-8 (left to right), which of the four inputs A – D (top to bottom) are in the coincidence circuit. The +5 V or 0 V control signals for each OR gate (Fig. 2) are sent to the logic board (Fig. 4) using a ribbon connector which attaches at the left side.

3.3 Final Assembly

The logic board (Figure 4) and control board (Figure 6) are connected together with a 34-conductor ribbon cable and housed in a project box as shown in Figure 7.

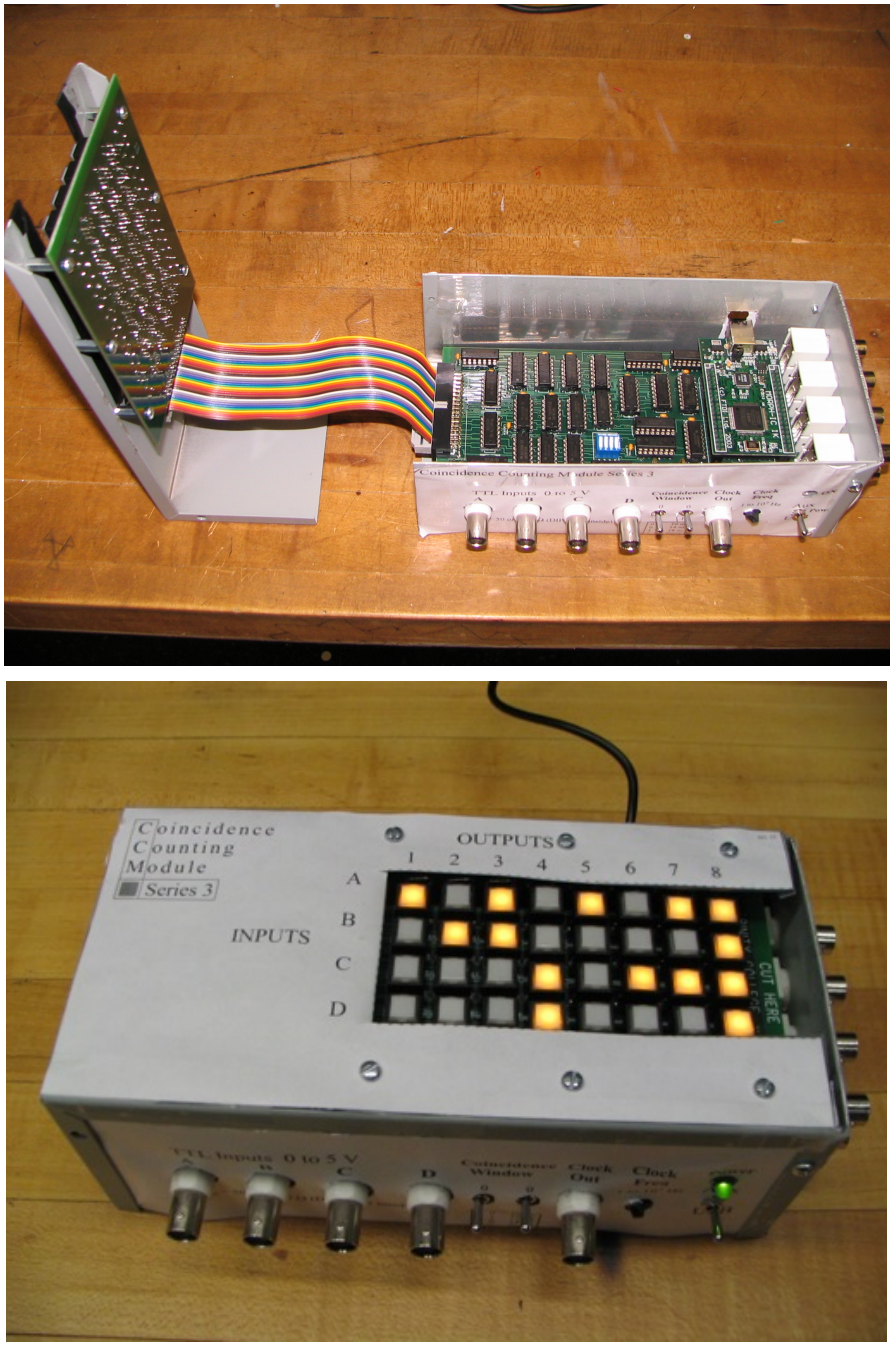

Fig. 7. Final CCM assembly.

The paper labels, which also serve as guide templates for drilling and cutting the project box, are available as part of the assembly guide for the CCM.[13]

3.4 FPGA counter operation

The FPGA is configured by flashing a pre-written and compiled VHDL program onto it over USB. This program creates eight (or six) independent counting registers from cells in the FPGA, with 16 bits (or 20 bits) available in each channel register. The number stored in each counting register is incremented on the leading edge of each TTL pulse from the 4-input AND gate. After a user-defined counting time (20 μs to 1 sec) has elapsed, the value in each counting register is copied to a storage register, and the counting registers are reset to zero. While the counting registers begin incrementing again, the storage register values are written into the FIFO buffer. After a predefined number of storage values are written to the buffer, they are transferred in a block to an array in the computer RAM via USB. The sets of count values in this array are then integrated for a user defined time interval, displayed on the computer monitor, and/or stored to hard disk; these tasks, and the flashing of the VHDL program, are managed by a LabVIEW routine that is freely available.[12]

3.5 Blind cycles

The transfer of the counting register values to the storage registers within the FPGA occupies one cycle of the FPGA's 50 MHz master oscillator; during this 0.2 μs time interval, the counting registers cannot be incremented, and are therefore "blind" to the arrival of any new TTL pulses. One such "blind cycle" will occur after each counting time interval has elapsed; thus, for an elapsed time T, the true duration of active data acquisition time is

$$T_{active} = T\left[1 - R/(50\ MHz)\right] \tag{1}$$

where R is the (user-selected) rate of data acquisition. The available values for R range from 1 Hz to 50 kHz.

4. PERFORMANCE

4.1 Pulse-shaping

The pulse-shaping circuit of Figure 3 was tested with 3-V TTL signals from a function generator. For this input (Channel A), the shortened pulses have durations of 7.5, 9.0, or 11.5 ns (± 0.5 ns) measured full width at half maximum, while the "11" setting creates a signal that is ~10 ns longer than the input pulse.

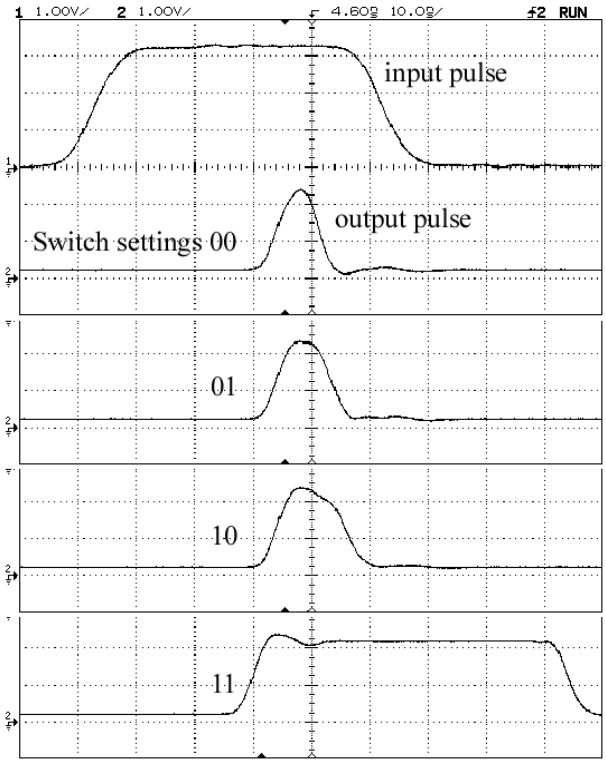

Fig. 8. Pulse profiles before and after the pulse-shaping circuit, for different settings of the A and B toggle switches.

4.2 Counting periodic pulses

The CCM was tested with a TTL pulse generator and was able to count coincidences at fixed frequencies of up to 37 MHz without losses. To achieve this, the pulse generator was phase-locked to the FPGA's master oscillator using the CCM's 10 MHz clock output, and a phase offset was added to prevent input pulses from coinciding with the blind cycles. Above 37 MHz, the blind cycles could not be avoided, and exactly R counts per second were missing from the totals in the counting registers. Above 74 MHz, exactly $2R$ counts per second were missing. The totals remained stable up to 84 MHz; above this input rate, the coincidences fluctuated, and ultimately fell to zero at 147 MHz, as successive pulses overlapped within the rise/fall times of the AND gates.

Because the coincidences and the single-channel counts are sent to the FPGA through different logic gates, they may arrive at the counting register inputs at slightly different times due to chip-to-chip variations in the rise and fall times of the gates. For periodic pulse trains that are synchronized to the CCM, it may turn out that some of the single-channel counts arrive at the FPGA during a blind cycle, while all of coincidence counts avoid them. This leads to the odd result that more coincidences are counted than single-channel events. This artifact of the FPGA counting routines does not affect the single-channel and coincidence count pulses that are produced at the TTL outputs. It can be overcome by phase-shifting the pulse train relative to the 10 MHz clock signal.

4.3 8-fold coincidences

To test the scalability with multiple modules, the phase-locked pulses from the generator were fanned out to eight copies, and delivered to the inputs of two separate CCM's. The 4-way coincidence TTL output from each CCM was fed to a third CCM, which counted them in coincidence as shown in Figure 9.

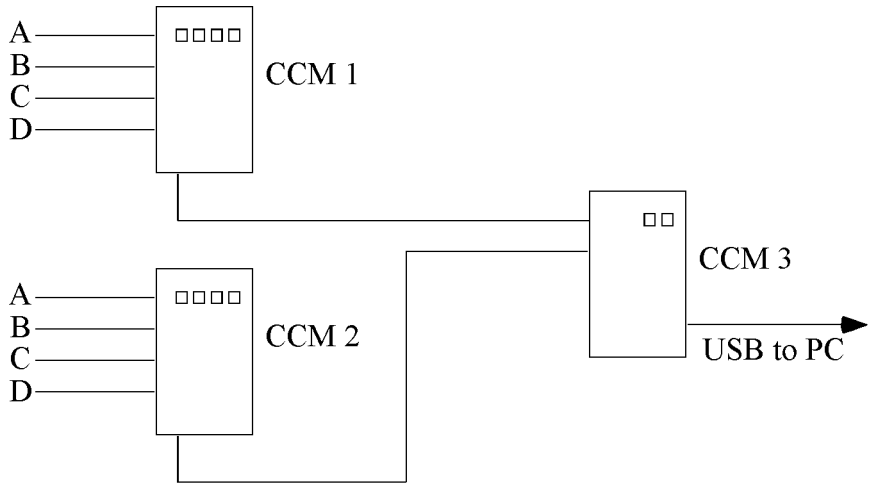

Fig.9. Configuration of 3 CCM's used to count 8-fold coincidences.

In this manner the third CCM was able to register up to eight-fold coincidence counts at rates of up to 30 MHz (the limit of the fan-out), without losses.

4.4 Counting pseudo-random pulses

The CCM was also tested with pulses from a linear feedback shift register (LFSR), which generated a pseudo-random binary TTL output with controllable mean rates of up to 10 MHz. Figure 10a shows the single-channel response of the CCM for all four input channels. The pseudo-random input pulses were counted independently with external 50-MHz counters The CCM is observed to precisely count the input pulses, all the way up to the maximum output rate of the LFSR.

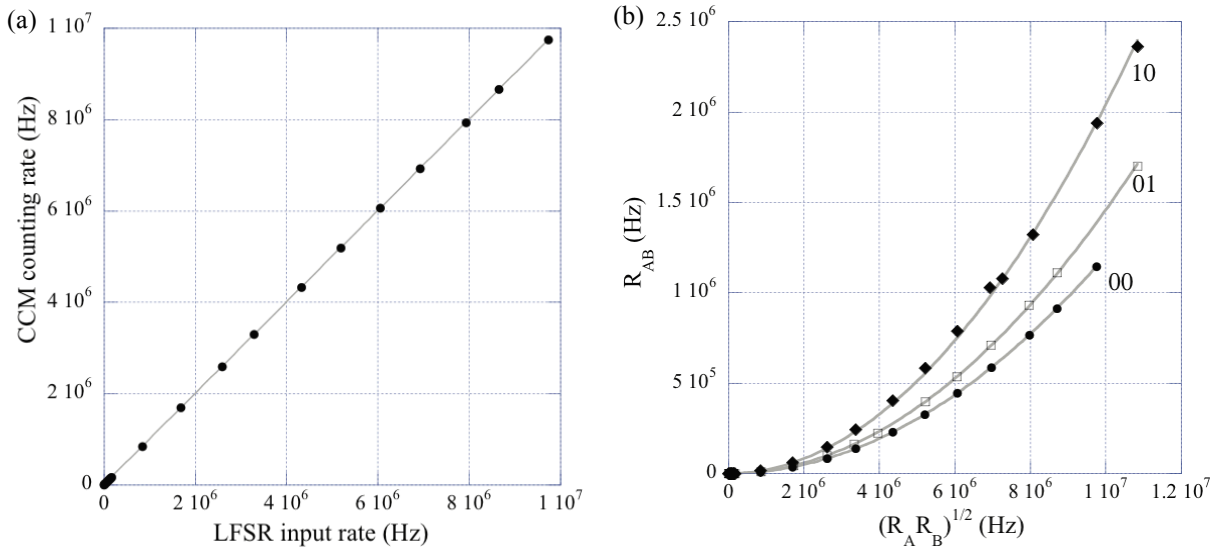

Fig. 10. (a) Mean single-channel counting rate in the CCM versus mean input pulse rate from an LFSR, acquired during 10-second intervals. A least-squares fit (solid line) of the form $y = mx$ yielded $m = 1.002 \pm 0.002$. (b) Coincidence rates R_{AB} in the CCM for pseudo-random input rates R_A and R_B on channels A and B, as a function of $x = \sqrt{R_A R_B}$, for pulse-width settings 00, 01, and 10. The solid lines are fits of the form $y = \tau_c x^2$ in accord with Eq. (2). Similar results were observed in the other input channels

The coincidence times were measured using two independent LFSR's on pairs of inputs. For randomly-arriving pulses with mean rates R_A and R_B in inputs A and B, the coincidence rate R_{AB} is given by

$$R_{AB} = \tau_c R_A R_B . \qquad (2)$$

where τ_c is the coincidence time, equal to twice the pulse duration τ minus a small amount necessary for sufficient overlap.[14] Single-parameter fits of the data to Eq. (2), shown in Figure 10b, yielded the values $\tau_c = 12.033 \pm 0.006$ ns, 14.56 ± 0.02 ns, and 20.38 ± 0.09 ns for the toggle positions 00, 01, and 10.

4.5 Counting pulses from random photon events

The coincidence times were also measured using two SPCMs and scattered light from a laser (which should produce independent, random streams of photons at the two detectors), yielding values of $\tau_c = 12.140 \pm 0.007$ ns, 14.133 ± 0.008 ns, and 21.47 ± 0.014 ns via Eq. (2). These coincidence times differ slightly from those measured with the LFSR's, due to differences in the input pulse heights and shapes from the SPCMs.

The measured values of τ_c from both methods are consistent with the times that we would expect, given the duration of the output pulses from the pulse shaping circuit. Note that because of chip-to-chip variations in the CCM components, the coincidence time may vary slightly for coincidences between different pairs of detectors, but does not vary over time for a fixed pair of detectors.

5. CONCLUSION

For applications where time-tagging of individual photon detections is not needed, our CCM offers some attractive features. It takes four inputs, and determines user selectable 2-, 3-, or 4-fold coincidences (or single-channel counts) on eight counting channels. The CCM has a high maximum count rate of 84 MHz, and its coincidence resolution is as low as 12 ns. Furthermore, several CCMs can be cascaded together to count arbitrary *M*-order coincidences among *N* inputs. Because of its small size, low cost, and intuitive user interface, the CCM is also well-suited to undergraduate physics laboratories.

6. ACKNOWLEDGEMENTS

We thank David Ahlgren, Sagar Bhandari, John Bower, Brandon Clary, Adam Katcher, Sarthak Khanal, Larry North, Steve Petkovsek, Young Ho Shin, Wayne Strange, and Jared Zimmerman for help with design, assembly and testing. This work was supported by NASA through the Connecticut Space Grant College Consortium

REFERENCES

[1] J. Gea-Banacloche, "Optical Realizations of Quantum Teleportation," Prog. Optics 46, 311-353 (2004).
[2] Kok, P., Munro, W. J., Ralph, T. C., Dowling, J. P., Dowling, and Milburn, G. J., "Linear optical quantum computing with photonic qubits," Rev. Mod. Phys 79, 135-174 (2007).
[3] Scarani, V., Bechmann-Pasquinucci, H., Cerf, N., Dusek, M., Lütkenhaus, N., and Peev, M., "The security of practical quantum key distribution," Rev. Mod. Phys. 81, 1301-1350 (2009).
[4] Gaertner, S., Weinfurter, H., and Kurtsiefer, C., "Fast and compact multichannel photon coincidence unit for quantum information processing," Rev. Sci. Instrum. 76, 123108 (2005).
[5] Stumner, A., "Coincidence Counter," http://www.physics.utoronto.ca/~astummer/pub/mirror/Projects/index.html
[6] Polyakov, S., Migdall, A., Nam, S. W., "FPGA-based multicoincidence recipe and software," (November 24 2009), http://www.nist.gov/pml/div684/grp03/multicoincidence.cfm
[7] Felekyan, S., Kuhnemuth, R., Kudryavtsev, V., Sandhagen, C., Becker, W., and Siedel, C. A. M., "Full correlation from picoseconds to seconds by time-resolved and time-correlated single photon detection," Rev. Sci. Instrum. 76, 083104 (2005).
[8] Wahl, M., Rahn, H-J., Gregor, I., Erdmann, R., and Enderlein, J., "Dead-time optimized time-correlated photon counting instrument with synchronized, independent timing channels," Rev. Sci. Instrum. 78, 033106 (2007).
[9] Acremann, Y., Chembrolu, V., Strachan, J.P., Tyliszczak, T., and Stohr, J., "Software defined photon counting system for time resolved x-ray experiments," Rev. Sci. Instrum. 78, 014702 (2007).
[10] Dehlinger, D., and Mitchell, M. W., "Entangled photon apparatus for the undergraduate laboratory," Am. J. Phys. 70, 898-901 (2002).
[11] Branning, D., Bhandari, S., and Beck, M., "Low-cost coincidence-counting electronics for undergraduate quantum optics," Am. J. Phys 77, 667-670 (2009).
[12] Branning, D., Khanal, S., Shin, Y.H., Clary, B., and Beck, M., "Note: Scalable multiphoton coincidence-counting electronics," Rev. Sci. Instrum. 82, 016102 (2011)
[13] Branning, D., http://www.trincoll.edu/~dbrannin/Coincidence%20Counting/CoincidenceHome.htm
[14] Eckart, C., and Shonka, "Accidental Coincidences in Counter Circuits," Phys. Rev. 53, 752-756 (1938).