A Hybrid Communication Topology for Modular Multilevel Converter

Hao Tu, Srdjan Lukic
FREEDM Systems Center
Department of Electrical and Computer Engineering
North Carolina State University, Raleigh, US
Email: {htu, smlukic}@ncsu.edu

***Abstract*—Modular multilevel converter with large number of modules is a potential topology for many applications such as high-voltage DC (HVDC) and solid state transformers (SST). However, the communication between the controller and the modules becomes complex if the number of modules is high. If conventional star topology communication where the controller has direct connection to all the modules is used, the wiring can be cumbersome and the cost can be high. Another possible solution is ring topology communication. Because the modules are connected in series and each module imposes some delay, ring topology has a limited communication speed. In this paper, we proposed a hybrid communication topology to combine star and ring topology. The proposed communication topology reduces the cycle time by 30% compared to ring topology for MMC with 30 modules while increases negligible cost.**

I. Introduction

With the improvement of digital controllers, many power electronics devices are controlled by digital signal processors(DSP) today. Modern DSPs have adequate computational power to implement converter-level control. They feature with float point unit, trigonometric math unit and many peripherals which can greatly reduce the time for prototyping converters. However, a single DSP has limited number of I/Os and computational power which makes it unsuitable for converters with a lot of switches such as MMCs. To take advantage of DSPs, an FPGA is normally attached to form a joint controller for MMCs. An FPGA with high number of I/Os and parallel operation capability is usually served as the interface unit between the DSP and the modules. This communication topology is called star topology as the master controller has direct wire connection to all the modules as shown in Fig.1. In [1]–[4] , an FPGA is used as an I/O expansion unit of DSP to form the joint controller. If the number of modules is too high, the wiring of star topology becomes complex and even impossible . Also, it requires a computationally powerful master controller. The cost of the master controller can increase rapidly with the increase of modules. For example, in [5] and [6], Opal real time simulator from Opal-RT technologies and CompactRIO from National Instruments are used to control MMCs. They are essentially the same as the DSP-FPGA joint controller but with much higher cost. In [7], a hierarchical star topology communication is proposed. Instead of having one single controller, the controllers are divided into three levels: master controller, valve controller and unit controller.

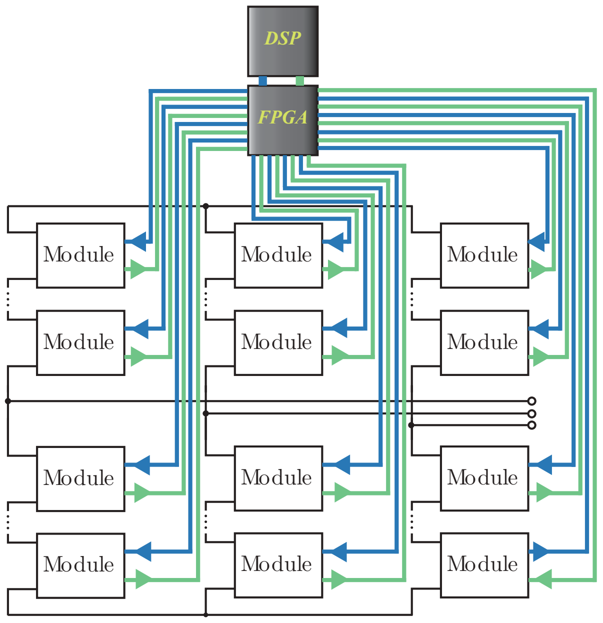

Fig. 1. Star topology communication for MMCs

The control algorithm is distributed to different controllers. This reduced the required computational power and I/Os from the master controller. However, it can still hits the limit as it is a star topology communication essentially. Also, extra effort is needed for synchronizing all the controllers. Another communication topology for MMCs is ring topology [8], [9]. In ring topology each module has a local FPGA controller and they are connected in series as shown in Fig. 2. Thus, only one pair of transceivers are needed for each module. As the number of transceivers are fixed, the system cost is linearly related to the number of modules. Furthermore, part of the control algorithm can be distributed to local controllers reducing the computational burden of the master controller. However, ring topology has a limited communication speed because each module imposes some delay during information exchange. The communication speed will become the bottleneck as devices capable of switching at high frequencies such as SiC or GaN devices become more common. To make use of these new technologies, we would need to update the duty cycles

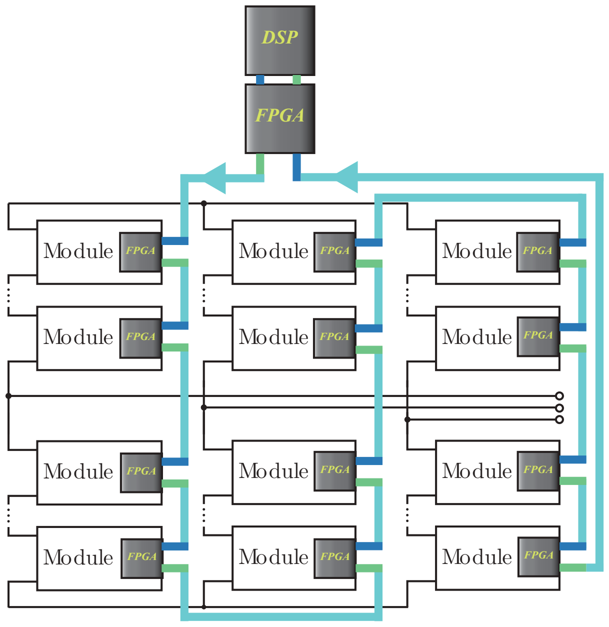

Fig. 2. Ring topology communication for MMCs

more often which requires a high speed communication link. In this paper, we propose a hybrid communication topology to combine star and ring topology. The proposed topology has the advantage of ring topology such as fixed number of transceivers and low cost while improving the communication speed.

The rest of the paper is arranged as follows. Section II reviews the basic communication principle of ring topology communication. Section III introduces the proposed hybrid communication topology. Section IV compares the performance of hybrid topology with ring topology. The hardware implementation and experiment results are in Section V. Section VI concludes the paper.

II. Ring Topology Communication

Fig. 3 shows simplified ring topology communication network. The master controller and all the modules are connected in series. One module's transmitter is connected to the subsequent module's receiver. For ring topology communication to work a dedicated protocol has to be implemented. Two typical ring communication protocols for power electronics systems are PES Net [10] and SyCCo bus [11] developed by Virgina Tech and ETH Zurich, respectively. In [12], the performances of the two protocols are compared. As SyCCo bus shows a better performance in terms of minimal cycle time and synchronization accuracy, we will use SyCCo bus as an example but any ring topology communication protocol can be applied.

In SyCCo bus, the master controller will send out a data frame, as shown in Fig.4, every switch cycle. This frame contains information for all modules. The control byte at the start of the frame indicates the frame type. Then follow N

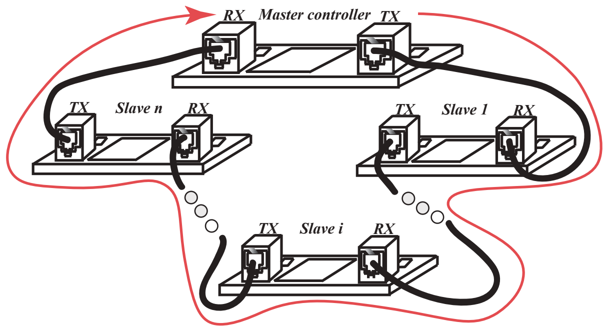

Fig. 3. Ring topology communication network

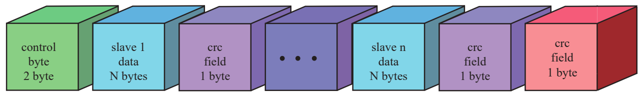

Fig. 4. Data frame defined by SyCCo bus

bytes data for slave 1 and 1 byte of CRC for the data. When slave 1 receives this frame, it will extract the data, replace them with N byte of measurements and then send the frame to the second slave. The second slave will do the same operation with its own data when it receives this frame from slave 1. Finally when master receives the frame from the last slave, all the data are replaced with slaves' local measurement. From its operation principle, we can see the cycle time T_{sw} of SyCCo bus consists of three parts,

$$T_{\mathrm{sw}} \geq T_{\mathrm{DSP}} + T_{\mathrm{data}} + T_{\mathrm{tran}} \tag{1}$$

where T_{DSP} is the time for master controller to calculate control algorithm, T_{data} is the time to send the data frame, T_{tran} is the transmission delay from the master controller to the last slave. T_{DSP} is independent of the communication speed. T_{data} only depends on the bit rate and the length of data frame. This is absolute minimal time needed for ring communication topology to transmit the data frame. T_{tran} is the sum of delays caused by signal propagation, FPGA logic and transceivers. Compared to the last two components, the signal propagation delay is negligible. The FPGA logic delay depends on the implementation and is proportional to the number of slaves. The transceiver delay is caused the transceiver chips used and also proportional to the number of slaves.

The transmission delay in a communication chain with 30 slaves is responsible for about 40% of the cycle time [12]. One way to reduce the transmission delay is to use faster transceiver chips. However, this could greatly increase the cost of communication system as every slave's transceiver chips need to be upgraded.

III. Hybrid Topology Communication

To decrease the transmission delay thus improve the communication speed, a hybrid topology which combines the

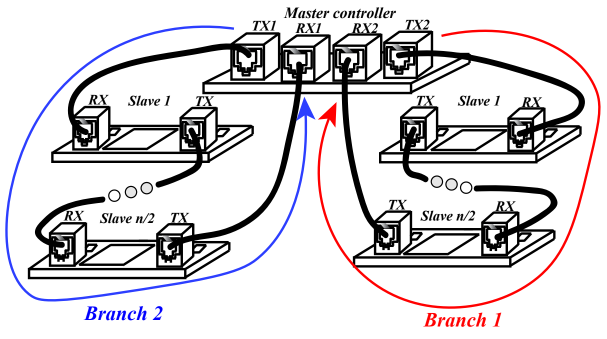

Fig. 5. Hybrid topology communication network

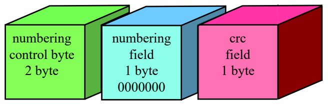

Fig. 6. Numbering frame sent out from the master controller

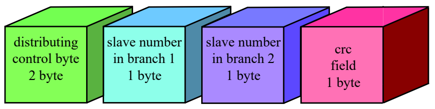
distributing control byte 2 byte
slave number in branch 1 1 byte
slave number in branch 2 1 byte
crc field 1 byte

Fig. 7. Numbering distributing frame sent out from the master controller

star and ring topology is proposed. The proposed hybrid communication topology is shown in Fig. 5. To imitate the parallel operation of star topology, a second branch is added. The two branches run in parallel while the slaves in each branch are connected in series. Therefore, the number of slaves in each is halved compared to the original single ring topology. The operating principle of the hybrid topology is introduced in this section.

After the system is powered up, the master goes through a process to initialize the system. The master controller sends a numbering frame to each branch as shown in Fig. 6. The numbering field is zero out of master controller. The slave receives the numbering frame will take the numbering field as its ID. Then, it will increase this number by one and send out this frame to the next slave in chain. The following slaves do the same thing to the frame and each slave is assigned an ID according to its position in the branch. When the numbering frame goes back to the the master controller, its numbering field is the number of slaves in the branch. The number of slaves in each branch is used by the next frame, distributing frame.

A distributing frame is shown in Fig. 7. This frame contains the number of slaves in both branches. The slave receiving this frame will store number of slaves of the branches into registers. This information will be used for synchronization.

After the distributing frames from both branches go back, the master controller will send out a delay measuring frame. This frame will collect the delay of each slave has for forwarding the frames. For each slave, it measures the time interval between they receiving and they sending out frames.

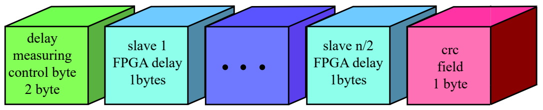
delay measuring control byte 2 byte
slave 1 FPGA delay 1bytes
• • •
slave n/2 FPGA delay 1bytes
crc field 1 byte

Fig. 8. Delay measuring frame received by the master controller

This is time interval is the delay caused by the FPGA logic. When a slave receives delay measuring frame, it will insert its measured delay in the measuring frame based on its slave ID. When this frames goes back to the master controller, it contains the delay of all slaves in that branch as shown in Fig. 8. During this process, the master controller is also measuring the time interval between it sending out and receiving delay measuring frame. With those time interval measured, the average delay of each slave can be calculated [12].

After the average delay of each slave is calculated, it is sent to the slaves by a delay distributing frame. This information will be used for synchronization.

After the delay distributing frame goes back to the master controller, the initialization process is finished. The master controller can start normal operation. As the each protocol has a different frame structure when sending information from the master to the slaves, SyCCo bus is used as the example protocol to illustrate the operation of the hybrid topology communication. The master controller sends out two data frames to branch 1 and branch 2, respectively. As explain in the last section, the data frame contains information for all the slaves in that branch. The slave extracts the data that are addressed to it, replaces the field with its measurements and forwards it to the next slave. After the slaves receive their data, the data will be stored in a shadow register. They will not become valid until the synchronization event.

The synchronization event happens when the last slave in the whole system receives its data. It can be either the last slave in branch 1 or the last slave in branch 2 depending on the delay in the branches. This can be calculated using the information distributed by the frames in the initialization process. Each slave estimates the time intervals between it finishing receiving the data frame and the last slave in both branches receiving the data frame. The larger time interval is used as the delay for the data to become valid. In this way, all the slaves in the whole system are synchronized to the moment that last slave receives its data.

Compared to the ring communication topology, the hybrid topology achieves faster communication speed with very small increase in hardware cost. It requires only two extra transceivers on the master controller. From the implementation point of view, the slaves have almost the same logic as ring communication topology. The master controller needs to accommodate to two branches accordingly. With two branches run in parallel, not only the transmission delay T_{tran} but also the frame time T_{data} can be reduced. Because the data frame for each branch only has about half the length as a single data frame for all slaves.

IV. Comparison between Ring and Hybrid Topology Communication

In this section we will compare the performance of ring and hybrid topology communication in term of cycle time using SyCCo bus as their protocol. Two case studies are conducted. In the first case, an MMC with 30 slaves is assumed and each slave has 2 bytes of data. In the second case, the MMC is assumed to have 30 slaves but each slave only has 9 bytes of data.

A. Cycle time for ring topology communication

For minimal information exchange between the master controller and the slave in an MMC, 2 bytes of data for each slave are enough. Duty cycle is sent from the master to the slave while module voltage measurement is sent from the slave to the master.

For SyCCo bus, the frame time can be calculated by,

$$T_{\text{data}} = (3 + \sum_{i=1}^{n} (N_{\text{byte}}(i) + 1))T_{\text{byte}} \tag{2}$$

where n is the number of slaves, $N_{\text{byte}}(i)$ is the number of bytes of data for slave i, T_{byte} is the time for transmitting 1 byte data. With 100 Mbps, $T_{\text{byte}} = 80$ ns. For MMC with 30 slaves and 2 bytes data for each slave, the frame is calculated by (2)

$$T_{\text{data}} = 7.44\ \mu s \tag{3}$$

As will be shown in section V, T_{tran} is around 640 ns per slave. For 30 slaves the transmission delay T_{tran} is 19.84 μs. One extra delay is caused by the master controller.

If we assume $T_{\text{DSP}} = 10\ \mu s$, the cycle time can be calculated by (1),

$$T_{\text{sw,ring,2byte}} \geq 37.28\ \mu s \tag{4}$$

Repeat the same process we can get the cycle time for the second case where each slave has 9 bytes data,

$$T_{\text{sw,ring,9byte}} \geq 54.08\ \mu s \tag{5}$$

B. Cycle time for hybrid topology communication

For hybrid topology communication, each data frames contains data for $\frac{n}{2}$ slaves. The slave data fields only has half the length as that in the ring communication topology. However, the overheads such as control byte and CRC check for the data frame are the same. Using equation (2) with $n = 15$ for each branch, the frame time can be calculated as,

So the frame time is only half of that in ring topology. For

$$T_{\text{data}} = 3.84\ \mu s \tag{6}$$

For the transmission delay, as each branch only has half of the slaves, the transmission delay is calculated by,

$$T_{\text{tran}} = 0.64(\frac{n}{2} + 1) = 10.24\ \mu s \tag{7}$$

where one extra delay is caused by the master controller. Then, use (1) to calculate the cycle time for 2 bytes and 9 bytes data for each slave, respectively,

TABLE I
Cycle time components for ring and hybrid topology

	2 bytes per slave		9 bytes per slave	
Topology	Ring	Hybrid	Ring	Hybrid
$T_{\text{DSP}}(\mu\text{s})$	10		10	
$T_{\text{data}}(\mu\text{s})$	7.44	3.84	24.24	12.24
$T_{\text{tran}}(\mu\text{s})$	19.84	10.24	19.84	10.24
$T_{\text{sw}}(\mu\text{s})$	37.28	24.08	54.08	32.48

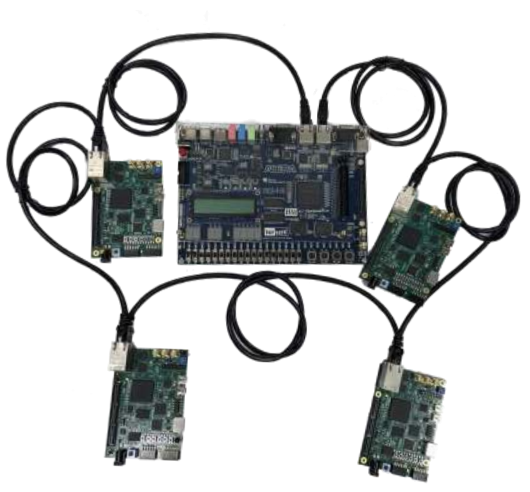
Fig. 9. Hardware setup for ring topology

$$T_{\text{sw,hybrid,2byte}} \geq 23.44\ \mu s \tag{8}$$

$$T_{\text{sw,hybrid,9byte}} \geq 30.64\ \mu s \tag{9}$$

The time components for ring and hybrid topology are listed in Table I. The transmission delay and data frame time in hybrid topology is cut to almost half of that in ring topology. This corresponds to 35% reduction in cycle time when each slave has 2 bytes data and 40% when each slave has 9 bytes data. It can be observed that hybrid topology offers more improvement over ring topology when 1. each slave requires more data 2. the number of slaves is larger.

V. Hardware Implementation and Experiment Results

A. Ring topology

In the lab a ring topology communication network with four slaves has been implemented. The Verilog source code is made open-source in [13]. The experiment setup is shown in Fig. 9.

The master controller is Terasic DE2-115 board with a Cyclone EP4CE115 FPGA. The slaves are implemented using Intel MAX 10 FPGA board. Both boards have two Ethernet transceiver chips, Marvell 88e1111. The transceiver chips work in 100 Mb/s MII mode. In this mode, 4 bits of data are transmitted or received with a 25 MHz clock. Therefore, 4 bits of data(one nibble) are sent at the same time with 40 ns bit time.

In the experiment two bytes data are assigned for each slave. The results are shown in Fig. 10. The blue and green

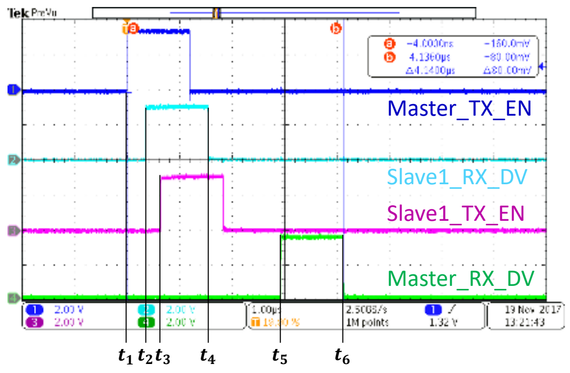

Fig. 10. Master transmitting enable and receiving valid signal for hybrid topology

waveforms are the transmitting enable and receiving valid signal of the master controller, respectively. The purple and cyan waveform are the transmitting enable and receiving valid signal of first slave in the chain, respectively. When the transmitting enable signal is asserted, the data frame is transmitted at a rate of 40 ns per nibble. When the receiving valid signal is asserted, the frame is received at the same rate. The length of the data frame can be calculated by the length of transmitting enable(receiving valid) signal. As shown in the Fig. 10, the master transmitting enable signal is asserted for 1.2 μs. This corresponds to a data frame with 15 bytes.

After the master starts transmitting the data frame at t_1, the receiving valid signal of the first slave is asserted at t_2. The time interval $t_2 - t_1$ is the delay caused the transmitter (of the master) and receiver(of the slave). It is around 360 ns. At t_3 the slave starts transmitting. The time interval $t_3 - t_2$ is the delay caused by the FPGA logic. Its length depends on the specific implementation. It is around 280 in current implementation. Together from t_3 to t_1, it is the delay cause by a slave assuming the master controller and the slave have the same transceiver chip. It is around 640 ns. This means adding one extra slave would introduce another 640 ns delay. This limits the speed of ring communication topology. During t_3 and t_4, the transmitting enable and receiving valid of the first slave are asserted at the same time, the slave is transmitting the data frame to the next slave while receiving the rest of the frame from the master.

At t_5 the master controller starts to receive and it finishes receiving the data frame at t_6. The cycle time is the time interval between the master controller starts to send data frame and it finishes receiving the data frame. It is calculated by $t_6 - t_1 = 4.14$ μs.

Fig. 11 shows the synchronization signals of all the slaves. The blue, cyan, purple and green waveforms are synchronization signals of the four slaves, respectively. When the signal transions from de-asserted to asserted, the newly received data become valid.

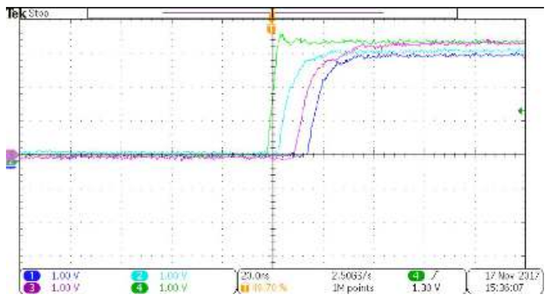

Fig. 11. Synchronization signal of 4 slaves for hybrid topology

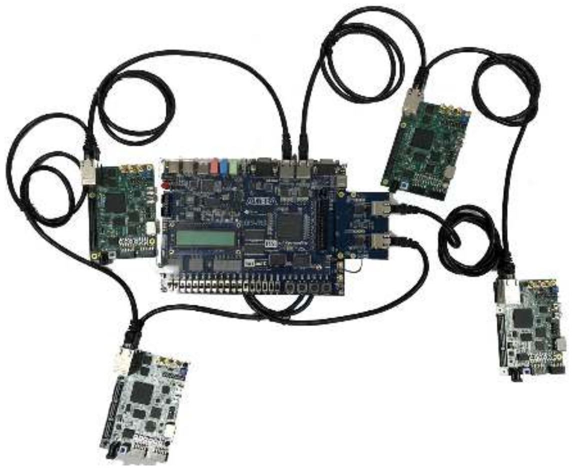

Fig. 12. Hardware setup of hybrid topology

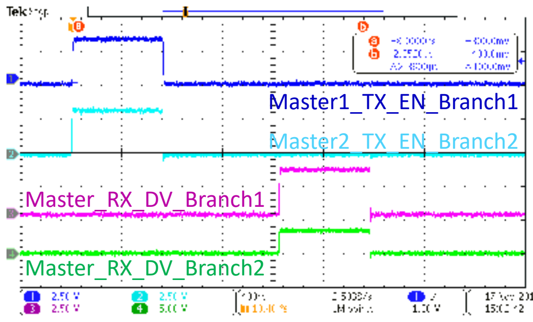

Fig. 13. Master transmitting enable and receiving valid signal for hybrid topology

B. Hybrid topology

The experiment setup for hybrid topology is shown in Fig. 12. Again, The master controller is Terasic DE2-115 board while the slaves are implemented using Intel MAX 10 FPGA board. An extension daughter card with two Marvell 88E1111 transceivers and RJ45 ports added to the master controller. The master controller is connected to two branches. Each branch has two slaves with two bytes of data for each slave.

The results are shown in Fig. 13. The blue and cyan waveforms are the transmitting enable of the two branches,

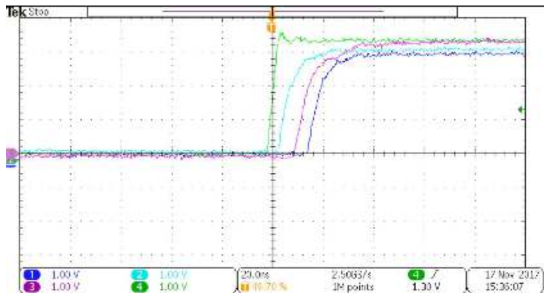

Fig. 14. Synchronization signal of 4 slaves for hybrid topology

TABLE II
FPGA LE USAGE FOR RING AND HYBRID TOPOLOGY

	ring topology	hybrid topology
Master	1121	2072
Slave	867	867

respectively. The purple and green waveform are the receiving valid signal of the two branches, respectively. As shown in the Fig. 13, the master transmitting enable signal for each branch is asserted for 720 ns. This corresponds to a data frame with 9 bytes. As the slave delay is the same as that in the ring topology, it is not repeated here.

The cycle time for hybrid topology is 2.36 μs. This is 43% reduction compared to single ring topology. The synchronization performance is shown in Fig. 14.

During the test, the maximum synchronization error can as high as 80 ns. This is because the transceiver chip Marvell 88E1111 on the boards has a variable transmitting and receiving latency. As indicating in the datasheet, the latency can fluctuate between 280 ns to 320 ns. As the synchronization methods in section III only compensates the average delay, any non-deterministic latency will result in an error in the synchronization event. The solution is to use transceiver chips with deterministic latency. This can improve the synchronization accuracy to ± 5 ns [11].

Finally the logic element(LE) used in the FPGA are compared. As shown in table II. For the master controller, the LE usage is almost doubled. However, it is still a small portion of the FPGA. It will not affect the other functions of the FPGA such as control algorithm calculation. For the slaves, the LE usage is the same in those two communication topologies.

VI. CONCLUSION

In this paper, two communication typologies, namely star and ring topology, for MMCs are reviewed. To combine their advantages, a hybrid communication topology is proposed. The hybrid topology has two communication branches in parallel while the modules inside each branch are connected in series. Theoretical analysis shows the proposed topology can reduce the cycle time of the ring topology by 30% for MMC with 30 modules while increasing negligible cost. The performance of ring and hybrid topology are compared through experiment. The experiment results show that for 4 slaves and 2 bytes data for each slave, the cycle time reduction is 43%.

ACKNOWLEDGMENT

This work was supported by the National Science Foundation (NSF) under Grants No. 1608929 and 1610074.

REFERENCES

[1] J. Wang, J. Liang, F. Gao, X. Dong, C. Wang, and B. Zhao, "A closed-loop time-domain analysis method for modular multilevel converter," *IEEE Transactions on Power Electronics*, vol. 32, no. 10, pp. 7494–7508, 2017.

[2] J. Kucka, D. Karwatzki, L. Baruschka, and A. Mertens, "Modular multilevel converter with magnetically coupled branch inductors," *IEEE Transactions on Power Electronics*, vol. 32, no. 9, pp. 6767–6777, 2017.

[3] Y. S. Kumar and G. Poddar, "Control of medium-voltage ac motor drive for wide speed range using modular multilevel converter," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 4, pp. 2742–2749, 2017.

[4] M. Espinoza, R. Cárdenas, M. Díaz, and J. C. Clare, "An enhanced *dq*-based vector control system for modular multilevel converters feeding variable-speed drives," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 4, pp. 2620–2630, 2017.

[5] S. M. Goetz, Z. Li, X. Liang, C. Zhang, S. M. Lukic, and A. V. Peterchev, "Control of modular multilevel converter with parallel connectivityapplication to battery systems," *IEEE Transactions on Power Electronics*, vol. 32, no. 11, pp. 8381–8392, 2017.

[6] M. Abdelsalam, M. I. Marei, and S. B. Tennakoon, "An integrated control strategy with fault detection and tolerant control capability based on capacitor voltage estimation for modular multilevel converters," *IEEE Transactions on Industry Applications*, vol. 53, no. 3, pp. 2840–2851, 2017.

[7] B. Fan, Y. Li, K. Wang, Z. Zheng, and L. Xu, "Hierarchical system design and control of an mmc-based power-electronic transformer," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 238–247, 2017.

[8] C. Toh and L. Norum, "A high speed control network synchronization jitter evaluation for embedded monitoring and control in modular multilevel converter," in *PowerTech (POWERTECH), 2013 IEEE Grenoble*. IEEE, 2013, pp. 1–6.

[9] A. Hillers, H. Tu, and J. Biela, "Central control and distributed protection of the dsbc and dscc modular multilevel converters," in *Energy Conversion Congress and Exposition (ECCE), 2016 IEEE*. IEEE, 2016, pp. 1–7.

[10] I. Milosavljevic, Z. Ye, D. Boroyevich, and C. Holton, "Analysis of converter operation with phase-leg control in daisy-chained or ring-type structure," in *Power Electronics Specialists Conference, 1999. PESC 99. 30th Annual IEEE*, vol. 2. IEEE, 1999, pp. 1216–1221.

[11] C. Carstensen, R. Christen, H. Vollenweider, R. Stark, and J. Biela, "A converter control field bus protocol for power electronic systems with a synchronization accuracy of±5ns," in *Power Electronics and Applications (EPE'15 ECCE-Europe), 2015 17th European Conference on*. IEEE, 2015, pp. 1–10.

[12] H. Tu and S. Lukic, "Comparative study of pes net and sycco bus: Communication protocols for modular multilevel converter," in *Energy Conversion Congress and Exposition (ECCE), 2017 IEEE*. IEEE, 2017, pp. 1487–1492.

[13] H. Tu. (2017, June) Verilog source code for mmc communication. [Online]. Available: https://github.com/htuNCSU/MmcCommunicationVerilog