Memristor-Based 4:2 Compressor Cells Design

Amirali Amirsoleimani†, Majid Ahmadi†, Mehri Teimoory, Arash Ahmadi†
{amirsol, ahmadi, aahmadi}@uwindsor.ca {mehri_teimoory}@yahoo.com
†Research Center of Integrated Micro Systems,
Department of Electrical and Computer Engineering, University of Windsor, Windsor, Ontario, Canada

***Abstract*— Memristor-based arithmetic circuits promise new alternatives for their conventional CMOS-based peers due to memristor's scalability and non-volatility features. In-memory memristor-based calculations become extensively attractive as it can be a solution to tackle memory bottleneck problems and also an ingredient for future beyond Von-Neumann computer architectures. In this paper material implication-based designs for 4:2 compressor cells using memristor devices are presented. A physical model is applied to determine real switching speed of memristive device. The proposed parallel design promises good speed performance with considerably less area than conventional CMOS designs. Finally a comparison has been made between the proposed memristor-based and CMOS-based designs in terms of number of applied devices per cell and delay.**

***Index Terms* – Memristor, Computer Arithmetic, Material Implication, Logic, CMOS.**

I. INTRODUCTION

CMOS technology scaling is reaching to its limits based on the Moore's law beyond 2020. Recently discovered memristor devices [1] have attracted intensified research interests due to their nanoscale features to replace conventional CMOS devices. Some of the important application for these two terminal resistive switching devices are memory and logic implementation. These devices are also used in neural [12] and arithmetic circuits [2], [6], [8] and [11]. Since memristor is capable of performing logic and memory operations it can be utilized for in-memory computation architectures which is beyond classical Von-Neumann architecture.

Various memristor-based logic implementation techniques are presented. Hybrid memristor/CMOS-based logic in [3] is based on gaining the advantage of in silicon utilization. In this logic the voltage defines the logic value and it cannot store last logic state. Pure memristor-based logic implementations, e.g. Memristor Aided Logic (MAGIC) [4] and Material Implication Logic (IMPLY) [5], are merging memory and logic block in the same unit. In these logics unlike hybrid memristor/CMOS-based logic, memristance of the output memristor shows output logic state. In memristive implication logic Boolean variables are considered by High Resistance State (HRS) and Low Resistance State (LRS) of binary memristor device where HRS and LRS states are considered for logic 0 and 1 respectively. The memristor-based IMPLY logic is a sequential process. Some memristor-based arithmetic circuits based on material implication logic were designed in [6], [8] and [11].

In this paper memristor-based 4:2 compressor cell based on material implementation logic is implemented and compared with existing IMPLY-based designs. 4:2 compressor cells are key components in partial product reduction tree of parallel multipliers.

Rest of the paper is organized as follows. In Section II memristor device and its functionality is defined. The logic implementation with material implication is defined in section III. The proposed design of memristor-based 4:2 compressor cell based on material implication logic is presented with its logic analysis in section IV. Finally in section V conclusion and remarks are provided.

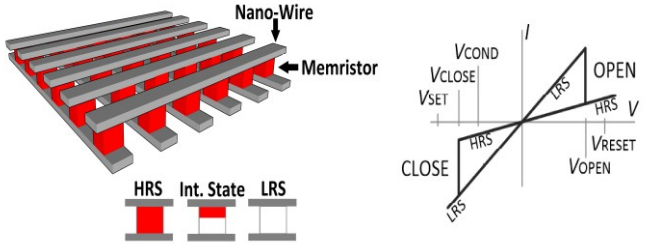
II. MEMRISTOR DEVICE AND ITS FUNCTIONALITY

Transition oxide-based memristor devices are ion-migration-induced redox-based resistive switches. This kind of devices are belong to Valence Change Mechanism (VCM) class [7] and their functionality is based on anion migrations along the device. During SET process in VCM devices, oxygen ions or vacancies (O^{2-}) migrate toward anode and oxygen deficient region is generated along insulator film. Then the electrons emitted from cathode are trapped by transition metal cations. The reduction in valence states of transition metal cations turns the oxide region conductive. At the anode, oxygen ions are oxidized and oxygen atoms are accumulated or released as oxygen gas. The VCM device changes its state to LRS at the end of the SET process.

The RESET procedure requires inverse voltage bias to device. The oxygen atoms gathered in anode are ionized and start to drift back as oxygen ions along the insulator film. At the end of the process of RESET the device is switched to HRS. Nano-crossbar layers of memristive devices can be placed over CMOS platform. Memristor logic states depends on amount and direction of charge passing through it. The crossbar memristor structure is displayed in Fig. 1. The voltages of V_{SET} and V_{RESET} should be greater than effective threshold voltage $|V_{OPEN}|$ and $|V_{CLOSE}|$ of VCM device respectively. Physical model [8] described TiO_2/TiO_{2-x} devices by dividing them into three regions: the conductive region, the transition region, and the insulating region.

The switching time for SET procedure in TiO_2/TiO_{2-x} memristor to change device state from HRS to LRS is determined by [8],

$$t_{SET} = \frac{C_0^2}{2\gamma_t \beta V \lambda} . \quad (1)$$



(a) (b)

Figure 1: (a) Memristor-based crossbar structure. (b) Ideal VCM memristor *I-V* curve with different voltages for memristor switching between LRS and HRS.

978-1-4799-5341-7/16/$31.00 ©2016 IEEE

The parameters γ_t and λ are the electron generating coefficient in the transition region and transition region length respectively. The parameter V is the voltage across device. Other parameters, C_0 and β, can be determined by,

$$C_0 = \frac{\lambda}{2} + \left(\frac{n_c}{n_i}\right)\left(D - \frac{\lambda}{2}\right) \quad (2)$$

$$\beta = \frac{1}{4}\left(\frac{n_c}{n_i} - 1\right)^3 + \frac{2}{3}\left(\frac{n_c}{n_i} - 1\right)^2 + \frac{1}{2}\left(\frac{n_c}{n_i} - 1\right). \quad (3)$$

Here n_i and n_c are ions concentration in insulation and conduction regions respectively. Also D is thickness of device film. The RESET procedure switching time from LRS to HRS is [8],

$$t_{RESET} = \frac{\left[\left(1 - \frac{n_c}{n_i}\right)D + 2C_0\right]\left(1 - \frac{n_c}{n_i}\right)D}{2\gamma_t \beta V \lambda} \quad (4)$$

III. Material Implication Logic with Memristor

One of the specific realization of stateful logic with memristive device is material implication (IMP) [5]. Memristors can execute IMP logic as one of the fundamental Boolean logic operations. Memristor-based IMP gate is a simple circuit comprising of two electrically connected memristor devices and one grounded resistor. The IMP statement is defined as,

$$f = a \text{ IMP } b \cong a \to b \cong (\sim a) \vee b, \quad (5)$$

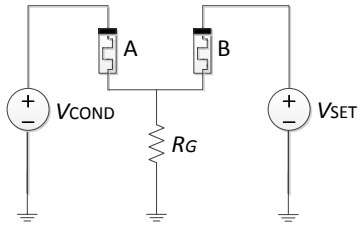
where f is only false if a is true and b is false. The IMP logic gate with memristor device is depicted in Fig. 2. Memristor A is considered as input memristor and memristor B is utilized as input/output device. The IMPLY logic with memristor has three main steps. Initializing the memristive devices based on the inputs is the first step. Then applying V_{COND} to memristor A and V_{SET} to memristor B simultaneously. The last step is reading out memristance of memristor B. In initialization of memristor devices each device based on the input value should be set to logic 0 (HRS) or logic 1 (LRS). The truth table of material implication logic and schematic of the IMP gate is shown in Fig. 2. The V_{COND} voltage magnitude is considered lower than V_{SET} ($|V_{COND}| < |V_{SET}|$). After this stage the output should be read by applying V_{READ} voltage pulse to the output memristor B. The resistance of R_G should be considered between R_{LRS} and R_{HRS} ($R_{LRS} < R_G < R_{HRS}$). Also two V_{COND} and V_{SET} voltages are applied to the doped sides of A and B memristors respectively.

The sequence of applying voltages to each memristive devices of the IMP gate in each stage for case 1 ($a = 0, b = 0$) is showed in Table. 1. In this case as it is stated in case 1 both devices should be adjusted to HRS initially. Therefore at first memristive device A should be switched with RESET pulse by duration of t_{RESET}. The same procedure is repeated again for device B in next step. At this point two devices are already switched to HRS state. Then V_{COND} and V_{SET} are imposed to A and B memristors. The voltage over the memristor B is determined by,

$$V_B = \frac{R_{HRS} + R_G}{R_{HRS} + 2R_G} V_{SET} - \frac{R_G}{R_{HRS} + 2R_G} V_{COND} \cong V_{SET}. \quad (6)$$

Case	A	B	A IMP B
1	HRS (0)	HRS (0)	LRS (1)
2	HRS (0)	LRS (1)	LRS (1)
3	LRS (1)	HRS (0)	HRS (0)
4	LRS (1)	LRS (1)	LRS (1)

(a)



(b)

Figure 2: (a) IMPLY truth table. (b) Schematic of memristor-based IMPLY logic gate.

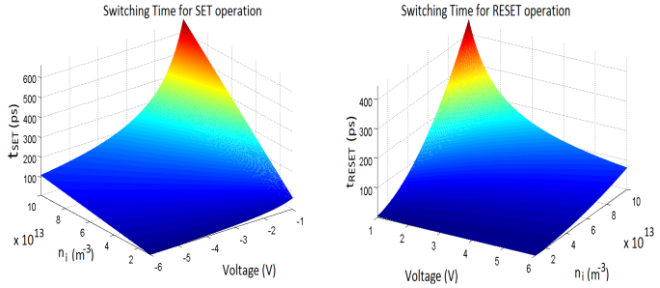
Table. 1: Material implication logic applied voltage sequence for case 1.

Memristors	Init. A	Init. B	A IMP B	Read B
A	V_{RESET}	0	V_{COND}	0
B	0	V_{RESET}	V_{SET}	V_{READ}

This voltage is enough to switch memristor B from HRS to LRS state. By considering the parameters in Table. 2 for device simulated in [8], -5.5 V for SET voltage and 5.5 V for RESET voltage over the device, minimum t_{SET} and t_{RESET} are about 70 ps and 25 ps respectively. The SET and RESET time behavior toward sweeping voltage over device and ions concentration in insulation region (n_i) parameters are displayed in Fig. 3. As it can be seen the worst speed performance is obtained in red region where n_i is about 10^{14} cm^{-3} and voltage is low ($|V| < 2$ V).

IV. Memristor-Based 4:2 Compressor Cell Designs

4:2 compressor cells are commonly utilized in high performance arithmetic systems and they are basic blocks in multipliers architecture. Parallel multipliers are consist of three main blocks: partial product generator, partial product reduction tree and final fast adder. Designing partial product tree is one of the key determining factors for improving speed, area and power consumption of multiplier. Several low power and high speed 4:2 compressor cell designs were presented by various digital logic style with conventional CMOS technology [9]. Memristor-based 4:2 compressor cell design can be applied in memristive crossbar architectures for signal processing application. The 4:2 compressor is a five-input/three-output



(a) (b)

Figure 3: (a) SET time behavior by sweeping voltage and n_i. (b) RESET time behavior by sweeping voltage and n_i.

Table. 2: Memristor parameters for physical model.

Parameter	Value	Parameter	Value	Parameter	Value
D (nm)	35	e (c)	1.602e-19	γ_t	2.3e-06
λ_0 (nm)	0.05D	w_0 (nm)	0.15D	γ_i	1e-06
A (μm^2)	0.25	n_c (m^{-3})	8.75e22	n_i (m^{-3})	3e13

device. It takes five equally weighted inputs (X_1, X_2, X_3, X_4, C_{IN}) and produces a sum-bit (S), a carry-bit (C) and a carry-propagate-bit (C_{OUT}). The Boolean expressions for outputs of 4:2 compressors are,

$$S = X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus C_{IN} \quad (7)$$

$$C = (X_1 \oplus X_2 \oplus X_3 \oplus X_4).C_{IN} + \neg(X_1 \oplus X_2 \oplus X_3 \oplus X_4).X_4 \quad (8)$$

$$C_{OUT} = (X_1 \oplus X_2).X_3 + \neg(X_1 \oplus X_2).X_1 . \quad (9)$$

Two primitive representations of 4:2 compressor cell are illustrated in Fig. 4. First design is comprised of two cascaded Full-Adder (FA). Further minimization of this design produces another representation comprising of four XOR gates and two multiplexers (MUX). Since with combination of FALSE and IMPLY logic, complete logic can be created, all basic logic gates can be implemented by applying IMPLY logic gate [3]. For implementing 4:2 compressor with two cascaded FA design two memristor-based full-adder implemented in [5] and [10] are utilized for two separate designs. The design with [5] full-adder requires 8 memristors and 58 computational cycles to complete its task. While by utilizing [10] full-adder with the same quantity of memristors, 44 computational steps are needed to finish its operation.

For designing 4:2 compressor with MUX/XOR design serial and parallel implementations can be applied. In the MUX/XOR design two blocks of XOR and MUX are required for implementation. The XOR block designed in [10] is utilized in two XOR/MUX designs of 4:2 compressor cells. The IMPLY-based implementation of applied XOR gate is [10],

$$A \oplus B = (A \text{ IMP } B) \text{ IMP } \neg(\neg A \text{ IMP } \neg B) . \quad (10)$$

This XOR design can be implemented by 4 memristors in 9 computational steps. For MUX circuit previously in [11] a 2 to 1 multiplexer with 13 computational steps (including initialization of devices) is designed with 6 memristors. Here a memristor-based MUX circuit with material implication logic is proposed by 4 memristors and 7 computational steps. By considering initialization of input memristors it needs 10 computational steps to complete its task. The proposed design utilizes memristors A and X as input devices. Also memristor B used as both input and output memristor. An additional memristor Y is applied for storing the values and performing FALSE operation during the procedure. The computational steps and memristor's logic states for proposed MUX are displayed in Table. 3 and Table. 4 respectively. In first stage logic statements of $\neg B + \neg X$ and $\neg A + X$ should be generated.

This can be done in three steps. First FALSE operation should apply on Y device to set it to HRS (logic 0). Then ¬X is stored in Y by utilizing IMPLY operation between X and Y device. Subsequently by using IMPLY operation between B and Y the target logic statement of $\neg B + \neg X$ is stored in Y device. Finally with IMPLY operation between A and X devices $\neg A + X$ logic statement is stored in device X. Since an output of MUX device is $A.\neg X + B.X$, two stored logic statements in X and Y should be inverted first. For this issue FALSE operation is applied to memristor B to change its state to HRS (logic 0). Then inverse of the statement stored in Y is generated by IMPLY operation between Y and B and it is stored in B at the end of the step. The stored logic statement in B is $B.X$. In step 7 by IMPLY operation between X and B the final logic statement stored in B becomes

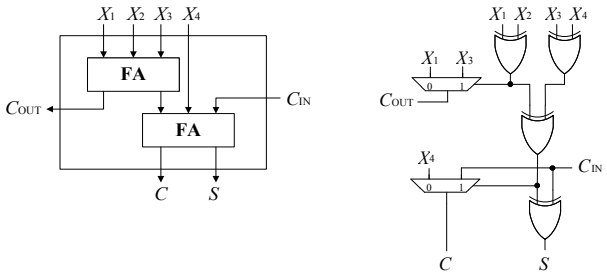


Figure 4: (a) FA design for 4:2 compressor. (b) XOR/MUX design.

$A.\neg X + B.X$. For serial implementation of 4:2 compressor XOR/MUX design, two different designs are proposed. First design has used the MUX circuit presented in [11] and the second design applied proposed MUX circuit. In XOR/MUX design with MUX circuit [11] 8 memristor devices are used and this design requires 64 computational steps. Second serial XOR/MUX design with proposed MUX circuit requires less devices and it has lower delay in comparison with first design. It needs 7 memristor devices for completing its task in 52 computational steps. Although these designs are efficient in terms of area and number of applied devices they are slower by considering current memristor technology and switching speed in comparison with conventional CMOS designs. For this purpose parallel implementation can be applied for increasing the speed of computations. The parallel memristor-based XOR/MUX 4:2 compressor cell design requires 11 memristor devices and it requires 26 computational step to finish its task as it is defined in Table. 5. In this design at first two XOR gates are implemented in parallel and it takes 9 steps for implementing them. The initialization of the input memristors is first step for X_1, X_2, X_3 and X_4 devices. Also all FALSE operations for 5 auxiliary memristors, M_0, M_1, M_2, M_3 and M_4 are completed in first step in parallel. After that XOR and MUX implementations are done in parallel. This stage requires 9 computational step since XOR design last 9 steps with initialization which is 2 cycles more than proposed parallel MUX. The output of the MUX is stored in X_3 device in step 17

Table. 3: Memristor-based MUX computational steps.

Step	Logic	Value
1	FALSE (Y)	Y = 0
2	X IMP Y	Y = ¬*X*
3	B IMP Y	Y = ¬*B* + ¬*X*
4	A IMP X	X = ¬*A* + *X*
5	FALSE(B)	B = 0
6	Y IMP B	B = ¬Y = *B.X*
7	X IMP B	B = ¬X + (*B.X*) = *A.*¬*X* + *B.X*

Table. 4: Memristor' logic states for proposed MUX in each step.

Memristors	**A**	**B**				**X**		**Y**		
Steps	In_A	In_B	5	6	**7**	In_X	4	1	2	3
Case 1	0	0	0	0	**0**	0	1	0	1	1
Case 2	0	0	0	0	**0**	1	1	0	0	1
Case 3	0	1	0	0	**0**	0	1	0	1	1
Case 4	0	1	0	1	**1**	1	1	0	0	0
Case 5	1	0	0	0	**1**	0	0	0	1	1
Case 6	1	0	0	0	**0**	1	1	0	0	1
Case 7	1	1	0	0	**1**	0	0	0	1	1
Case 8	1	1	0	1	**1**	1	1	0	0	0
Value		**Output**								

as C_{OUT} of 4:2 compressor. In step 18 C_{IN} is applied to initialize C_1 and C_2 devices and FALSE operation is done for X_2, M_0, M_1, M_2, M_3 and M_4. In the last stage XOR and MUX blocks are implemented in parallel to extract *C* and *S*. At the end of the procedure in step 26 the device C_2 is stored *C* and M_0 is saved *S*. By comparing proposed 4:2 compressor design in terms of delay with conventional CMOS designs with transmission gate and pass transistor logic in [9] the proposed parallel design showed acceptable speed performance. The delay of the circuits is determined by the time difference from when the inputs reach 10% of their swing until the output signal reaches 90% of its full potential measure. The worst case delay based on device parameter mentioned in Table. 2 is 1.8 ns. Also in terms of area this model only applied 11 memristors while CMOS-based compressors with transmission gate and pass transistor logic are utilized 32 and 30 transistors respectively. The comparison for delay and device quantity per cell of the mentioned memristor-based designs with conventional CMOS designs are displayed in Fig. 5 and Fig. 6 respectively.

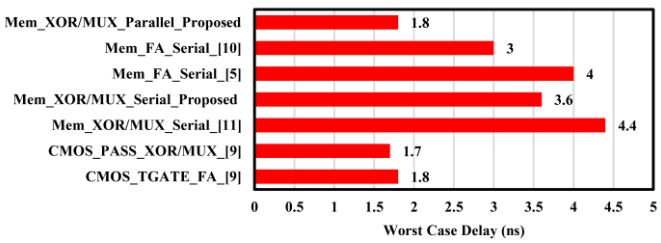


Fig. 5: Designs worst case delay comparison.

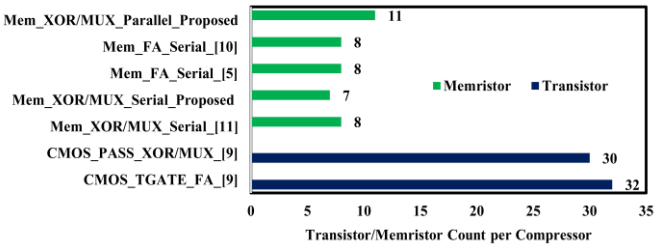


Fig. 6: Designs transistor/memristor count per compressor comparison.

V. Conclusion

In this paper several memristor-based implementations by material implication logic for XOR/MUX and Full-adder based representation of 4:2 compressor are presented. The physical model of a VCM device is utilized to determine the SET and RESET time. The parallel implementation with memristor offers comparable speed with considerably less area in comparison with conventional CMOS-based peers that utilized pass transistor and transmission gate logic. Also all proposed memristor-based serial and parallel implementations of 4:2 compressor cells are compared with CMOS based design in terms of delay and number of device per cell.

References

[1] D. B. Strukov, G. S.Snider, D. R. Stewart, and R. S. Williams, "The Missing Memristor Found," Nature, Vol. 453, pp. 80-83, May 2008.

[2] K. Bickerstaff, E. E. Swartzlander, "Memristor-based arithmetic," in Signals, Systems and Computers (ASILOMAR), 2010 Conference Record of the Forty Fourth Asilomar Conference on , vol., no., pp.1173-1177, 7-10 Nov. 2010.

[3] S. Kvatinsky, N. Wald, G. Satat, E. G. Friedman, A. Kolodny, and U. C. Weiser, "MRL – Memristor Ratioed Logic," Proceedings of the International Cellular Nanoscale Networks and their Applications, pp. 1-6, August 2012.

[4] Kvatinsky, Shahar, Dmitry Belousov, Slavik Liman, Guy Satat, N. Wald, E. G. Friedman, A. Kolodny, and U. C. Weiser. "MAGIC–memristor aided LoGIC." IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 62, pp. 786–790, 2014.

[5] S. Kvatinsky, N. Wald, G. Satat, E. G. Friedman, A. Kolodny, and U. C. Weiser, "Memristor-based material implication (imply) logic: Design principles and methodologies," IEEE Transactions on Very Large Scale Integration (VLSI), vol. 22, pp. 2054–2066, 2013.

[6] M. Teimoory, A. Amirsoleimani, A. Ahmadi, S. Alirezaee, S. Salimpour, M. Ahmadi, "Memristor-Based Linear Feedback Shift Register Based on Material Implication Logic" IEEE European Conference on Circuit Theory and Design (ECCTD), 2015.

[7] R. Waser, R. Dittmann, G. Staikov, and K. Szot, "Redox-based resistive switching memories–nanoionic mechanisms, prospects, and challenges," Advanced Materials, no. 21, pp. 2632–2663, 2009.

[8] L. Zhang , Z. Chen , J. J. Yang , B. Wysocki , N. McDonald and Y. Chen "A compact modeling of TiO_2-TiO_{2-x} memristor", Appl. Phys. Lett., vol. 102, no. 15, pp.153503 2013.

[9] M. Howard, P. Mokrian, M. Ahmadi, and W. C. Miller, "Power and delay analysis of 4:2 compressor cells," CAS-FEST of IEEE International Symposium on Circuits and Sysems (ISCAS), May 2005.

[10] M. Teimoory, A. Amirsoleimani, J. Shamsi, A. Ahmadi, S. Alirezaee, M. Ahmadi, "Optimized implementation of memristor-based full adder by material implication logic," Electronics, Circuits and Systems (ICECS), 21st IEEE International Conference on, pp.562-565, Dec. 2014.

[11] H. Owlia, P. Keshavarzi and A. Rezai. A novel digital logic implementation approach on nanocrossbar arrays using memristor-based multiplexers. Microelectronics journal. Elsevier, 45:597-603, 2014.

[12] J. Shamsi, A. Amirsoleimani, S. Mirzakuchaki, M. Ahmadi, "Modular Neuron Comprises of Memristor-based Synapse," Neural Computing and Applications, pp. 1-11, 2015.

Table. 5: Computations are shown for each step of memristor-based 4:2 compressor operation.

Steps	XOR1	XOR2	Steps	XOR3	MUX1	Steps	MUX2	XOR4
1	$M_{0,1,2,3,4} = 0$, set inputs ($X_{1,2,3,4}$)		**10**	M_0 imp M_1		**19**	X_4 imp M_1	
2	X_1 imp M_0	X_3 imp M_2	**11**	M_0 imp X_4	M_1 imp X_2	**20**	M_1 imp X_2	X_4 imp M_0
3	X_2 imp M_1	X_4 imp M_3	**12**	M_2 imp M_3	X_2 imp M_4	**21**	X_2 imp M_3	C_1 imp M_4
4	X_1 imp X_2	X_3 imp X_4	**13**	M_0 imp M_2	X_3 imp M_4	**22**	C_2 imp M_3	X_4 imp C_1
5	M_0 imp M_1	M_2 imp M_3	**14**	X_4 imp M_3	X_1 imp X_2	**23**	X_1 imp X_2	M_0 imp M_4
6	$M_0 = 0$	$M_2 = 0$	**15**	$X_4 = 0$	$X_3 = 0$	**24**	$C_2 = 0$	$M_0 = 0$
7	M_1 imp M_0	M_3 imp M_2	**16**	M_3 imp X_4	M_4 imp X_3	**25**	M_3 imp C_2	M_4 imp M_0
8	X_2 imp M_0	X_4 imp M_2	**17**	M_2 imp X_4	X_2 imp X_3 = $\boldsymbol{C_{OUT}}$	**26**	X_2 imp C_2 = $\boldsymbol{C}$	C_1 imp M_0 = $\boldsymbol{S}$
9	$M_{0,1,2,3,4} = 0$, $X_2 = 0$, $X_4 = 0$		**18**	$M_{0,1,2,3,4} = 0$, $X_2 = 0$, set $C_{1,2} = C_{IN}$				