Mohamed Zanaty[1]
Harvard John A. Paulson School
of Engineering and Applied Sciences,
Harvard University,
Cambridge, MA 02138
e-mail: mzanaty@seas.harvard.edu

Hubert Schneegans
Instant-Lab,
Ecole Polytechnique
Federale de Lausanne,
Neuchatel 2000, Switzerland
e-mail: hubert.schneegans@epfl.ch

Ilan Vardi
Instant-Lab,
Ecole Polytechnique
Federale de Lausanne,
Neuchatel 2000, Switzerland
e-mail: ilan.vardi@epfl.ch

Simon Henein
Instant-Lab,
Ecole Polytechnique
Federale de Lausanne,
Neuchatel 2000, Switzerland
e-mail: simon.henein@epfl.ch

Reconfigurable Logic Gates Based on Programable Multistable Mechanisms

Binary logic gates are building blocks of computing machines, in particular, electronic computers. One variant is the programable logic gate, also known as the reconfigurable logic gate, in which the logical function implemented can be modified. In this paper, we construct a mechanism to implement a reconfigurable logic gate. This mechanism is based on the concept of programable multistable mechanisms which we introduced in previous work. The application of a programable multistable mechanism is superior to the different bistable mechanisms previously used to implement logic gates since a single mechanism can be used to implement several logic functions. Our reconfigurable logic gates use a novel geometric construction where the geometric data depend on the stability behavior of the mechanism. There are 16 binary logic gates and our construction can theoretically produce nine of these and our physical model produces six logical gates. Input and output of the mechanism are displacement and the mechanisms can be combined serially, i.e., output of a mechanism is an input for another. We show that we can implement NOR *and* NAND *gates, so combinations of our mechanism can express any logical function. The mechanism is therefore theoretically universal, i.e., implement any computation. We give an analytic model of the mechanism based on Euler–Bernoulli beam theory to find the geometric data, then validate it using finite element analysis and experimental demonstration.*
[DOI: 10.1115/1.4045970]

Keywords: compliant mechanisms, mechanical computation, mechanism design, mechanism synthesis

1 Introduction and Statement of Results

1.1 Logic Gates. Binary logic gates have two inputs each taking on possible values *True* or *False*, with an output taking on values *True* or *False*. For technical reasons, the usual mathematical formulation is to denote True by the number 1 and False by the number 0 [1]. Logic gates are the classical building blocks of computing machines. Electronic transistors are widely used to implement these logic functions due to the speed and miniaturized size within the nano-regime and simple integration with electronic circuits. However, under extreme conditions occurring in space applications such as high temperature and cosmic radiation, electronic gates can fail. On the other side, mechanical structure is reliable functioning under these conditions [2,3]. This has motivated research into purely mechanical structures to implement logic gates. Switches and rods are a basic example for the implementation of mechanical logic [4]. Mechanical oscillators have recently been utilized as logic devices where the logic output is encoded by oscillator frequency [5]. Bistable mechanisms have also been used to build logic gates [6,7].

There are four unary logic gates and 16 basic binary logic gates [1] and these can be combined to implement any general logical expressions. Of special interest are the NOR and NAND gates which can reproduce any other unary and binary logic gate and are therefore called *universal*. It follows that the implementation of a NOR or NAND gate and the ability to combine gates allow one to implement any computable logic function.

One construction of a binary logic gate is the *programable logic gate*, also known as the *reconfigurable logic gate*, where the logic gate functionality can be modified post-fabrication. This paper will focus on the construction of purely mechanical reconfigurable logic gates.

1.2 Multistable Mechanisms. The mechanisms considered in this paper are *compliant mechanisms* that perform their functions by the deformation of their flexible elements, e.g., slender beams [8,9].

These mechanisms store *strain energy* as they deform, and local energy minima positions correspond to stable states and energy maxima to unstable states. The qualitative behavior of such mechanisms can be described by their energy profile [9]. In particular, a mechanism is called a *multistable mechanism* if it has more than one stable state.

As a bistable mechanism is displaced from one stable state to another stable state, it passes through an unstable state, and before doing so, its stored energy increases. When displaced past the unstable state, the mechanism converts potential energy into kinetic energy, since the system is conservative, and the mechanism displaces to the other stable state [8], see Fig. 1, where the position of the ball indicates the state of the mechanism.

The most basic multistable mechanism is the *bistable mechanism* having two stable states. Examples of these are curved beams [10,11], the four-bar mechanism [12,13], slider crank mechanism [14], and tensural pivots [15]. Bistable mechanisms are well suited for logic computation due to their binary nature. Since they act as switches, they can be considered as mechanical analogues of transistors

1.3 Programable Multistable Mechanisms. The bistable mechanisms applied to logic gates can only implement a single logic function. In order to implement reconfigurable logic gates, we use *programable multistable mechanisms* (PMM) introduced in our previous work [16–18]. These are characterized by having a set of inputs we have named *programing inputs* p_2 and p_1 to modify the mechanism's stability behavior, in particular, the

[1]Corresponding author.

Contributed by the Mechanisms and Robotics Committee of ASME for publication in the JOURNAL OF MECHANISMS AND ROBOTICS. Manuscript received September 21, 2019; final manuscript received January 10, 2020; published online January 14, 2020. Assoc. Editor: Charles Kim.

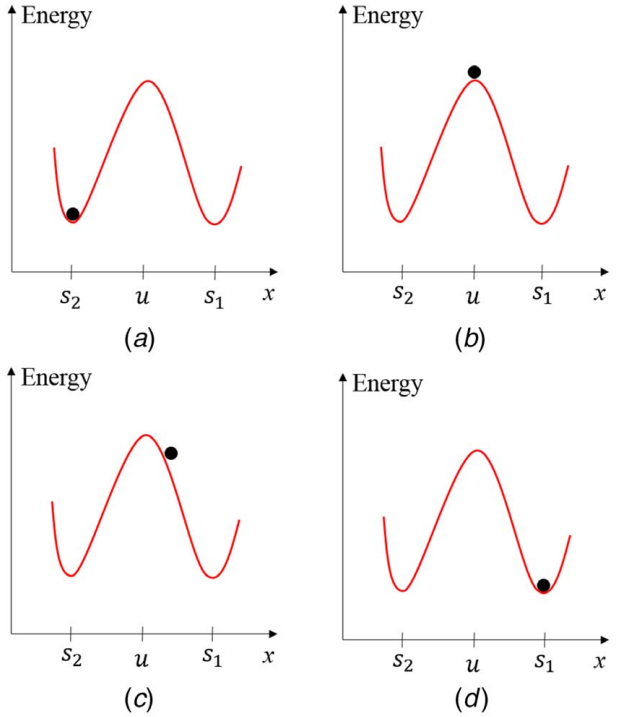

Fig. 1 Strain energy of the displacement-driven bistable mechanism where the mechanism position is denoted by the ball at (*a*) stable state s_2, (*b*) unstable state u, (c) past unstable state u, and (*d*) to other stable state s_1

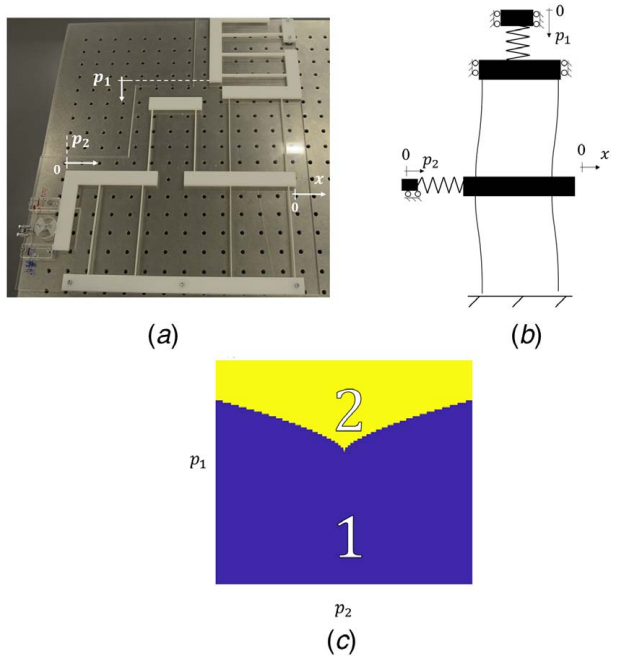

Fig. 2 (*a*) Physically constructed demonstrator of our multistable mechanism implementing logic gates, (*b*) conceptual diagram of our mechanism, (c) programing diagram showing the number of stable states corresponding to programing inputs p_1 and p_2

number of stable states. Figures 2(*a*) and 2(*b*) illustrate our mechanism with programing inputs p_2 and p_1 and output x.

As opposed to bistable mechanisms which are displacement driven, our programable multistable mechanisms produce displacement by modifying the energy profile. One can think of this as moving an object on a carpet by lifting the carpet, i.e., actuation is orthogonal to displacement.

This is illustrated in Fig. 3, where the ball again represents the state of the mechanism. The mechanism is initially bistable with the ball at stable state s_2 (Fig. 3(*a*)). Then, programing inputs p_1 and p_2 are applied to modify the energy profile (Fig. 3(*b*)). At certain values of p_1 and p_2, s_2 becomes unstable as illustrated in Figs. 3(*c*) and 3(*d*) and the mechanism becomes monostable (Figs. 3(*e*) and 3(*f*)). After that, the ball moves to the other stable state s_1, which is now the only stable state (Figs. 3(*e*) and 3(*f*)). One advantage of this method is that actuation is more repeatable, as was applied to the construction of a medical device for delicate eye surgery [18].

The set programing values p_1 and p_2 are partitioned into regions where the mechanism is monostable and bistable. This is described by a *programing diagram*; Fig. 2(*c*) discussed in detail in Sec. 3. This discretization is used to construct logic gates via the geometric method of Sec. 1.4, in particular, regions of monostability will correspond to True, i.e., to output 1 and regions of bistability will correspond to False, i.e., 0.

Remark. In this paper, we use the term "programing" to refer to the modification of the stability behavior of the mechanism. This has no relation to stored computer programs in computability.

1.4 Geometric Construction of Reconfigurable Logic Gates. In this section, we present a novel geometric construction of a reconfigurable logic gate represented by block diagram in Fig. 4(*a*). The stability behavior of our PMM mechanism will provide the geometric data to implement this method.

Given a region Ω in the plane and positive constants α_2 and α_1, one can construct a reconfigurable logic gate $f_{r_2,r_1}(i_2, i_1)$, where r_2 and r_1 are values that reconfigure the gate and i_2 and i_1 are the binary logical inputs taking values 0 or 1.

For each choice of reconfiguration values r_2 and r_1, one applies the logical values i_2 and i_1 using r_2 and r_1 so that logical input i_j corresponds to coordinate $r_j + (i_j - 1/2)\alpha_j$. More explicitly, this says that $i_2 = 0$ corresponds to $r_2 - \alpha_2/2$ and $i_2 = 1$ corresponds to $r_2 + \alpha_2/2$, and similarly, $i_1 = 0$ corresponds to $r_1 - \alpha_1/2$ and $i_1 = 1$ corresponds to $r_1 + \alpha_1/2$. Geometrically, this creates a rectangle centered at (r_2, r_1) with sides of length α_2 and α_1, and whose four corners correspond to the four possible values of (i_2, i_1) (see Fig. 4(*b*)).

We then define a logic gate

$$f_{r_2,r_1} = \tau[(r_2 + (i_2 - 1/2)\alpha_2,\ r_1 + (i_1 - 1/2)\alpha_1) \in \Omega] \qquad (1)$$

where τ is the truth value of a logical expression ε

$$\tau[\varepsilon] = \begin{cases} 1 & \text{if } \varepsilon \text{ is True} \\ 0 & \text{if } \varepsilon \text{ is False} \end{cases}$$

Geometrically, the logic gate $f_{r_2,r_1}(i_2, i_1)$ returns 1 if and only if the corner of the rectangle corresponding to (i_2, i_1) lies inside the region Ω (see Fig. 4(*c*)).

The 16 possible binary logical gates are not all necessarily implementable, as the possible gates depend on the region Ω. For example, only 14 of these are possible when Ω is the interior of a circular disc; the two functions corresponding to the rectangle having only opposite corners inside the disc are geometrically impossible; these are $f_6 = 0110$ and $f_9 = 1001$ using the notation of Table 1.

1.5 Reconfigurable Logic Gates Generated by Programable Multistable Mechanisms. In this paper, reconfigurable logic gates will be generated by the geometric method of Sec. 1.4 where the region Ω corresponds to a *programing diagram* described in more detail in Sec. 3. These charts will have the property that the region Ω is the part of the plane lying below the single-valued function $p_1 = \Phi(p_2)$. This means that the truth value True

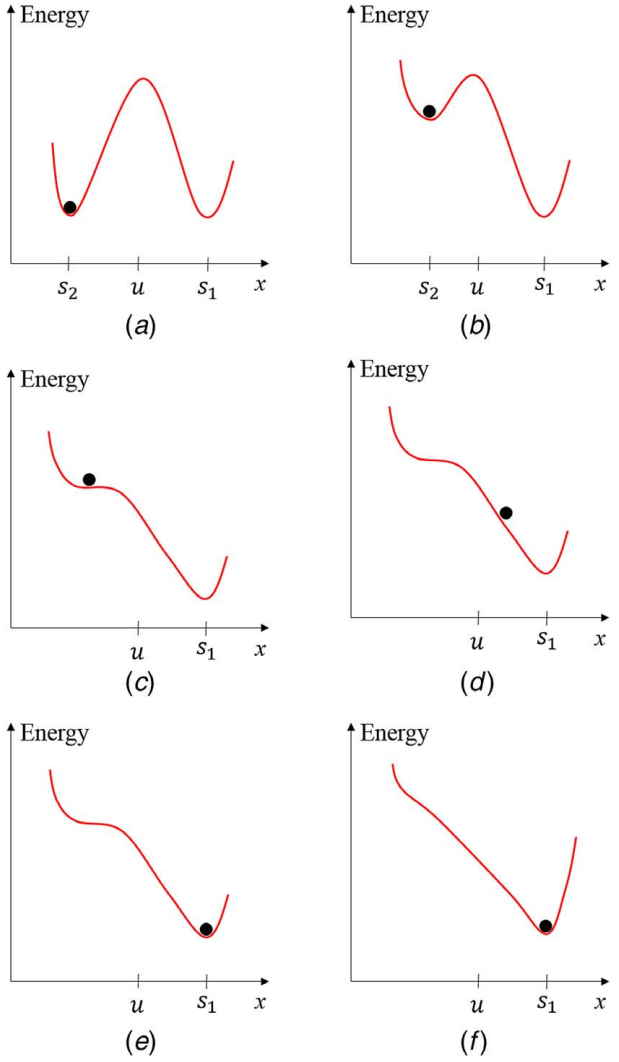

Fig. 3 Strain energy of the one-degree-of-freedom programable multistable mechanism for different values of p_1 and p_2 as the mechanism switches from bistability to monostability, where the ball denotes the position of the mechanism: (*a*) bistable with ball at stable state s_2, (*b*) the stable state s_2 has a higher energy level, (c) stable state s_2 vanishes and the ball starts moving to single stable state s_1, (*d*) ball moving toward the other stable state s_1, (*e*) the ball reaches the stable state, s_1, and (*f*) ball remains at stable state s_1

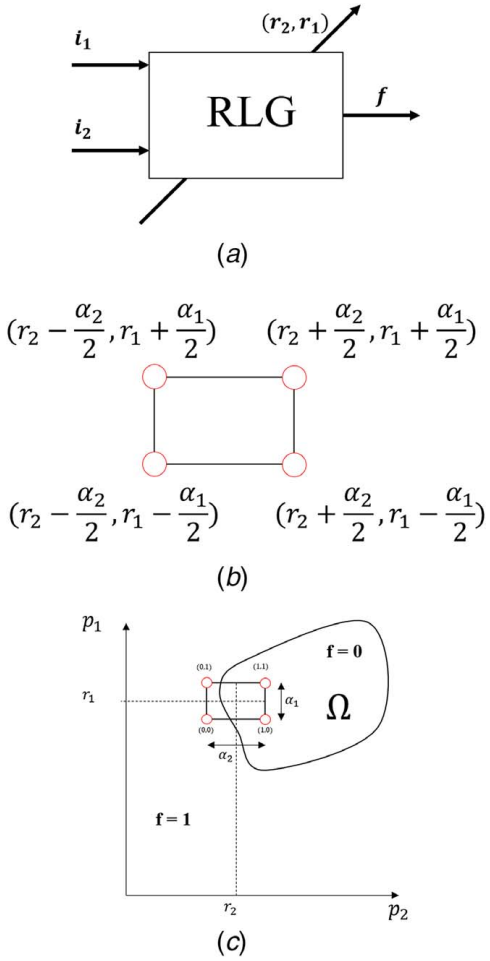

Fig. 4 (*a*) Block diagram representation of reconfigurable logic gate where i_1 and i_2 are the logic operands and r_1 and r_2 determine the logical operation and f is the logic output, (*b*) four possible logic input states, (c) region Ω with output True when corner of the rectangle is outside Ω

corresponds to being below the graph $(p_2, \Phi(p_2))$ in the p_2 and p_1 plane (see Fig. 5).

In the case of a region defined by a single-valued function Φ, the definition of a reconfigurable logic gate given by Eq. (1) becomes

$$f_{r_2,r_1}(i_2, i_1) = \tau[r_1 + (i_1 - 1/2)\alpha_1 < \Phi(r_2 + (i_2 - 1/2)\alpha_2)] \quad (2)$$

The definition of Eq. (2) states that $f_{r_2,r_1}(i_2, i_1)$ has output 1 if and only if $r_1 + (i_1 - 1/2)\alpha_1$ is less than $\Phi(r_2 + (i_2 - 1/2)\alpha_2)$.

The output f corresponds to a displacement x, as illustrated in Fig. 2. When the mechanism is bistable, then it has energy profile as illustrated in Fig. 3(*b*) and is in stable state s_2, corresponding to zero displacement $x = 0$. On the other hand, when the mechanism is monostable, then its energy profile corresponds to Fig. 3(*d*) where the mechanism is in the single stable state s_1 corresponding to a displacement $x > \beta$, where β is a constant. The mechanism is designed so that $\beta > \alpha_1$ and $\beta > \alpha_2$; this property guarantees that the output of one of our mechanisms can be applied as input for another of our mechanisms, i.e., logical expressions can be constructed out of our logic gates.

Table 1 Implementable functions

		i_1	1	0	1	0
		i_2	1	1	0	0
Logical expression	Binary code					
False	f_0		0	0	0	0
$\neg(i_1 \vee i_2)$	f_1		0	0	0	1
$\neg i_2$	f_3		0	0	1	1
$\neg(i_1 \leftarrow i_2)$	f_4		0	1	0	0
$\neg i_1$	f_5		0	1	0	1
$\neg(i_1 \wedge i_2)$	f_7		0	1	1	1
i_2	f_{12}		1	1	0	0
$i_1 \rightarrow i_2$	f_{13}		1	1	0	1
True	f_{15}		1	1	1	1

The number of possible logic gates generated by our PMMs using this geometric method is limited by the fact that

$$f(i_2, 1) = 1 \Rightarrow f(i_2, 0) = 1$$

which simply states that if $r_1 + \alpha_1/2 < \Phi(r_2 + (i_2 - 1/2)\alpha_2)$, then $r_1 - \alpha_1/2 < \Phi(r_2 + (i_2 - 1/2)\alpha_2)$. Geometrically, this says that if a top corner of the rectangle is below the graph, then the bottom corner just under it is also below the graph.

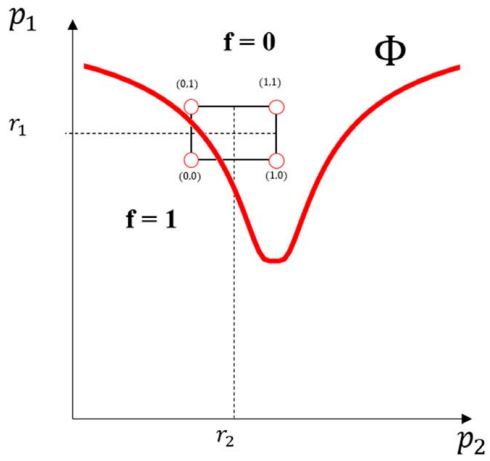

Fig. 5 Single valued function boundary $p_1 = \Phi(p_2)$

This is the only general restriction, and using the notation of Table 1, it forbids all binary codes f_{xx10} and f_{10xx}, that is, the seven functions

$$f_2 = f_{0010}, \quad f_6 = f_{0110}, \quad f_8 = f_{1000},$$

$$f_9 = f_{1001}, \quad f_{10} = f_{1010}, \quad f_{11} = f_{1011}, \quad f_{14} = f_{1110}$$

In general, this method can generate nine logic gates at most and these functions are listed in Table 1. The number of the implemented logic functions depends on value of α_1 and α_2.

The partition Φ generated by the programing diagram of our PPMs can generate all nine of these logic gates. Our physical demonstrator was limited only to four of these logic gates (see Sec. 6).

We note that $f_1 = \neg(i_1 \vee i_2)$ and $f_7 = \neg(i_1 \wedge i_2)$ in Table 1 correspond to a NOR gate and a NAND gate, respectively, since these are universal, our mechanisms can be combined to perform any computation. In particular, our demonstrator implements f_1 (see Sec. 6).

The nine possible functions divided the space of reconfiguration parameters r_1 and r_2 into nine regions; this generates a *reconfiguration diagram* (see Fig. 6). This discrete partition of the reconfiguration space means that the reconfiguration of our logic gates can be done by applying nine discrete inputs. The application of this technique is the subject of our current research.

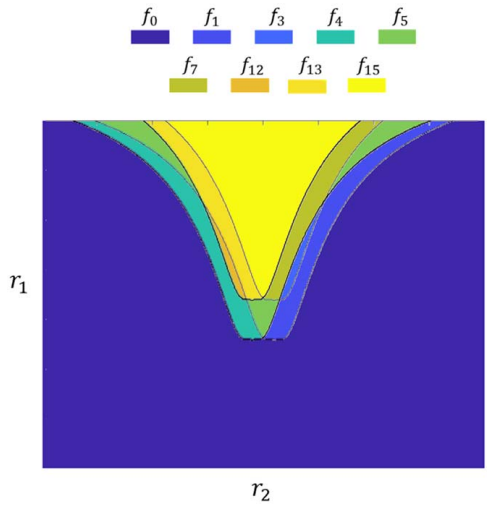

Fig. 6 The space of reconfiguration parameters r_1 and r_2 divided into nine regions corresponding to the nine possible logic gates

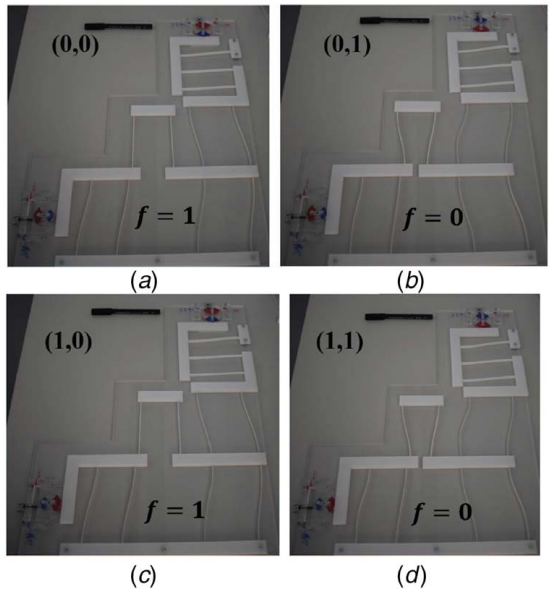

Fig. 7 Mechanism to implement $f = \neg p_1^i$ at $(p_2^i, p_1^i) =$ (a) (0,0), (b) (0,1), (c) (1,0), and (d) (1,1)

1.6 Physical Implementation. Figure 2(*a*) gives the physical implementation of a reconfigurable logic gate studied in this paper. The demonstrator is made of polyoxymethylene using laser cutting. Its programing diagram is given in Fig. 2(*c*). Two different logic functions are illustrated in Figs. 7 and 8. Further details concerning the application of the programing inputs are given in Fig. 19. We ensure that the displacement of the mechanism is within the linear regime; therefore, there is no plastic deformation.

We selected the dimensions of the mechanism such that the maximum stresses during actuation never surpass the admissible stress, which we define as the fatigue limit divided by a safety factor as discussed in Ref. [8].

1.7 Advantages and Limitations. Our implementation of logic gates has the following advantages compared to the state-of-the-art.

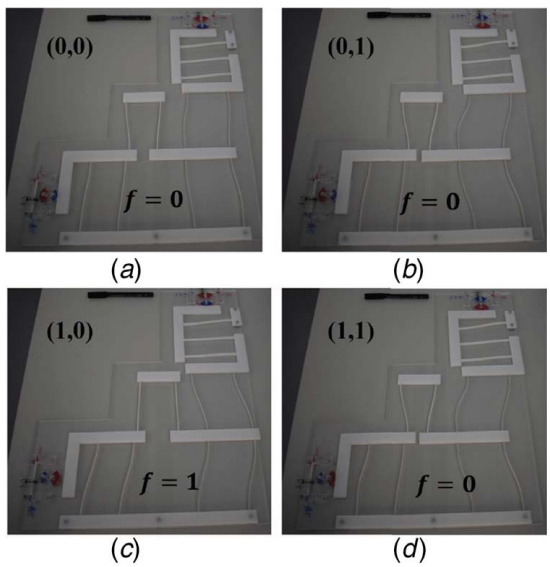

Fig. 8 Mechanism to implement $f = p_2^i \wedge \neg p_1^i$ at $(p_2^i, p_1^i) =$ (a) (0,0), (b) (0,1), (c) (1,0), and (d) (1,1)

(1) Our logic gates are reconfigurable which enable the implementation of different logic functions using the same mechanical structure.
(2) Our logic gates are based on multistable mechanisms so energy is required only during the transition between stable states, i.e., during computation. No energy is required to maintain the stable position of the mechanism.
(3) Our mechanisms store output values, represented by the stable states.
(4) Our mechanical logic gates can be combined without requiring auxiliary mechanisms.
(5) Our logic gates are universal, i.e., any computable function can be implemented using the logic gates of this paper.
(6) The number of the logic operands can be easily increased by adding a spring element for each additional input.

Complex functions can be implemented on combining logic gates. In logic design, two metrics are used to evaluate the performance of combined logic gates, fan out limit and propagation delay.

The fan out limit is the maximum number of logic gate inputs that can be driven by the output of the logic gate. As illustrated in Fig. 10, a small amount of energy is provided by the input E_{in}, represented by the energy difference between unstable state u and stable state s_2, to initiate the execution of the function. Also, the gate releases energy, E_{out}, equivalent to the energy difference between the unstable state, u, and the stable state, s_1, to represent a high output.

Therefore, the limit of the fan out is

$$M = \frac{E_{out}}{E_{in}} \tag{3}$$

assuming that the driven logic gates are identical.

The second parameter is the propagation delay. It is defined as the time taken by a signal to travel from the input of the logic gates to the output. In our case, we can consider our serial combination of programable multistable mechanisms as a transmission line. The propagation delay within bistable transmission lines was addressed in Ref. [19] and it depends on the amount of energy released by each bistable unit and energy dissipation.

In this paper, we will focus only on the implementation of reconfigurable logic gates using programable multistable mechanisms.

1.8 Paper Outline. The paper is organized as follows: Sec. 2 gives a complete description of our proposed reconfigurable logic gate. Section 3 derives an analytical model to calculate the programing diagram by computing mechanism stiffness as a function of the programing inputs. Section 4 provides a description of the effect of modifying the programing inputs on the number of stable states, their position, and their stiffness. Section 5 gives a setup of the FEM simulation of our mechanism. Finally, Sec. 6 describes the logic gates implemented by our physical construction and deduces the reconfiguration diagram.

Whenever possible, we express our results using normalized parameters so that the expressions are independent of the actual physical dimensions of the mechanisms.

2 Elastically Driven Axially Loaded Double Parallelogram Mechanism

The *elastically driven axially loaded double parallelogram mechanism* (EADPM) given in Fig. 9 consists of two slender beams with thickness t, width w, and length ℓ. The two beams are connected by a central block. On one extremity, the beams are fixed and the other extremity is restricted only to the movement in the axial direction of the beams and loaded by a spring of stiffness k_1. The central block is loaded by a spring k_2 in the lateral direction of the beams. The spring k_1 is preloaded by displacement p_1 and the spring k_2 is preloaded by displacement p_2. The displacement p_1 and p_2 are the programing inputs. The position of the central block of the mechanism is given by x.

When $p_1 = 0$ and $p_2 = 0$, the mechanism is monostable as given in Fig. 9. On imposing x, the lateral displacement of the central block increases and spring k_2 is loaded. As the beams deform laterally, the beams move axially loading k_1. On increasing p_1 for $p_2 = 0$, the spring k_1 imposes an axial load on the beams reducing its stiffness as illustrated in Figs. 10(*a*) and 10(*b*). As p_1 increases beyond a critical value p_1^{cr}, the beams buckle and move laterally deforming the spring k_2 and the mechanism is bistable. Figures 9(*b*) and 9(*c*) give the stable positions of the mechanism. These stable states correspond to the logic output.

As p_1 increases beyond p_1^{cr}, the energy barrier between the stable states increases. However, if $p_1 < 0$, the stiffness of the mechanism increases.

The programing input p_2 does not affect the stiffness of the beams. However, it modifies the mechanism overall stability behavior. To study the effect of p_2 on the stability response of the mechanism, there are two qualitative distinct cases depending on p_1.

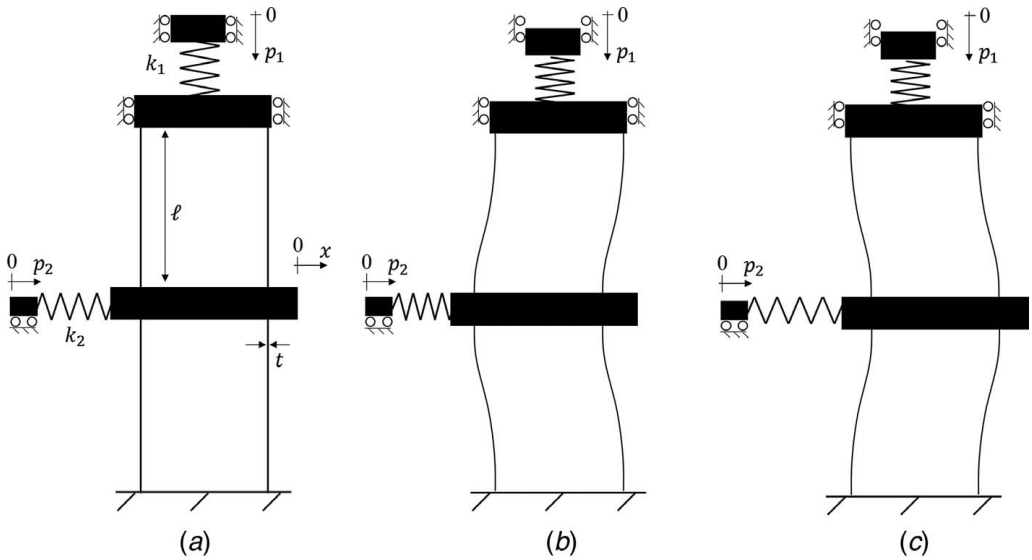

Fig. 9 An EADPM for $p_2 = 0$ where (*a*) stable state at $p_1 = 0$, (*b*) and (c) stable states for $|p_2| < p_2^{cr}$ and $p_1 > p_1^{cr}$, corresponding to high output and low output of a logic operation

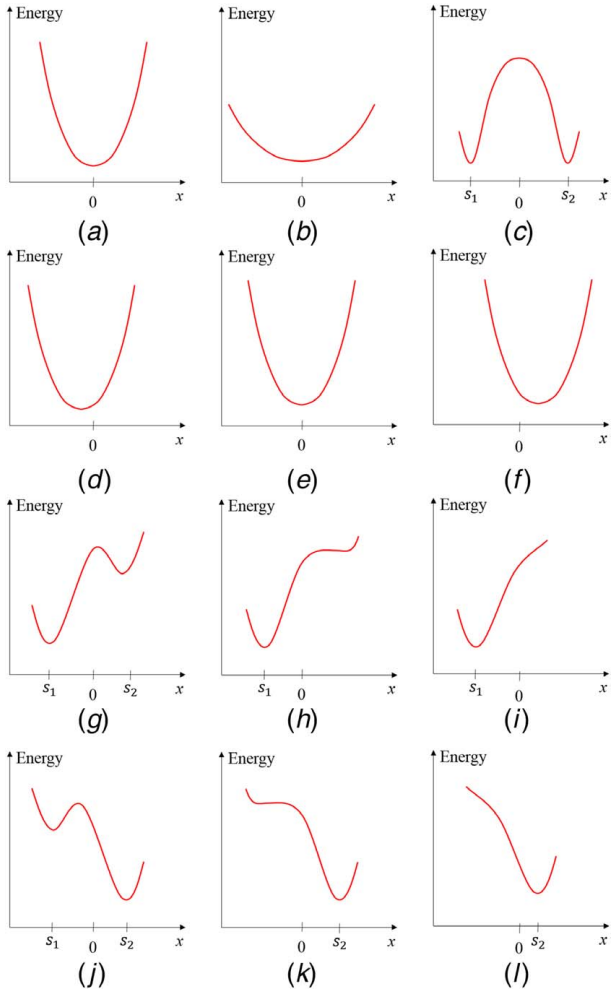

Fig. 10 Energy profiles of EADPM at (a) $p_1 = 0$, $p_2 = 0$; (b) $p_1 = p_1^{cr}, p_2 = 0$; (c) $p_1 > p_1^{cr}, p_2 = 0$; (d) $p_1 < p_1^{cr}, p_2 < 0$; (e) $p_1 < p_1^{cr}, p_2 = 0$; (f) $p_1 < p_1^{cr}, p_2 > 0$; (g) $p_1 > p_1^{cr}, 0 > p_2 > -p_2^{cr}$; (h) $p_1 > p_1^{cr}, p_2 = -p_2^{cr}$; (i) $p_1 > p_1^{cr}, p_2 < -p_2^{cr}$; (j) $p_1 > p_1^{cr}, 0 < p_2 < p_2^{cr}$; (k) $p_1 > p_1^{cr}, p_2 = p_2^{cr}$; and (l) $p_1 > p_1^{cr}, p_2 > p_2^{cr}$

Case 1: $p_1 < p_1^{cr}$

The EADPM is monostable where the position of the stable state depends on p_2. At $p_2 = 0$, the mechanism has one stable position at $x = 0$. When $p_2 > 0$, the stable position will be at $x > 0$. Otherwise, the stable position will be at $x < 0$ as illustrated by the energy profiles given in Figs. 10(*d*)–10(*f*).

Case 2: $p_1 > p_1^{cr}$

If $p_2 = 0$, the mechanism is bistable. As p_2 changes, the energy barrier will vary. There is a value at which $|p_2| > p_2^{cr}$, a stable and unstable state merge. Therefore, the energy barrier vanishes and the mechanism is monostable. The strain energy of the mechanism for different values of p_2 when $p_1 > p_1^{cr}$ are given in Figs. 10(*g*)–10(*l*).

Table 2 summarizes the different stability regions as functions of p_1 and p_2 where "–" indicates no impact.

Remark II: The mechanism exhibits zero stiffness behavior at $p_1 = p_1^{cr}$ or $p_2 = p_2^{cr}$. The mechanism is a zero force mechanism at $p_1 = p_1^{cr}$ and a nonzero force mechanism when $p_2 = p_2^{cr}$.

Remark III: The value of p_2^{cr} is a function of p_1. The higher p_1 is, the higher p_2^{cr} will be. When $p_1 < p_1^{cr}$, p_2^{cr} does not exist.

3 Analytical Model

The goal of the model is to find the stability boundaries at which the mechanism switches between monostability and bistability, which are essential for finding the relation between the logic operation and the programing inputs. This requires a qualitative description of the stability behavior of the EADPM. Our model is based on the following assumptions:

(1) The beams are slender, i.e., the beam thickness is 100 times lower than the length. Therefore, there is no shear within the beam.
(2) The beam displacement is within its intermediate range.
(3) The material used is a linear elastic material with Young's modulus E.
(4) The elastic axial extension of the beams is neglected.

Figure 9(*a*) gives the main variables representing the dimensions of the mechanism used in the model and Table 3 gives the value of these dimensions. The stiffness of the double parallelogram mechanism can be written as [20]

$$k_p = \frac{48EI}{\ell^3}\left(1 - \frac{N}{N_0}\right) \tag{4}$$

where

$$N_0 = \frac{2\pi^2 EI}{\ell^2}, \quad I = \frac{wt^3}{12}$$

The axial load exerted on the beams is

$$N = k_1(p_1 - \lambda)$$

where λ is the axial displacement of the beam

$$\lambda = \frac{6x^2}{5\ell}$$

Substituting back into the equation

$$k_p = \frac{48EI}{\ell^3}\left(1 - \frac{k_1(p_1 - \frac{6x^2}{5\ell})}{N_0}\right)$$

The effective stiffness of the mechanism is equivalent to two springs, the beams denoted by k_p and k_2, connected in parallel. Therefore, the total required force is

$$f = k_p x - k_2(p_2 - x) \tag{5}$$

We normalize the force f by the factor EI/ℓ^2. This leads to the following equation:

$$\hat{f} = \frac{f\ell^2}{EI} = \frac{288}{5}\gamma\hat{x}^3 + (48 - 48\gamma\hat{p}_1 + \hat{k}_2)\hat{x} - \hat{k}_2\hat{p}_2 \tag{6}$$

Table 2 Stability regions as functions of p_1 and p_2

Number of stable states	p_1	p_2
1	$p_1 < p_1^{cr}$	–
1	$p_1 > p_1^{cr}$	$\lvert p_2\rvert > p_2^{cr}$
2	$p_1 > p_1^{cr}$	$\lvert p_2\rvert < p_2^{cr}$

Table 3 Dimensions used for FEM simulation

Parameter	Value	Parameter	Value
ℓ	120 (mm)	t	0.2 (mm)
w	10 (mm)	E	210 (GPa)
k_1	5 (kN/m)	k_2	30 (N/m)

where

$$\hat{x}=\frac{x}{\ell},\quad \hat{p}_1=\frac{p_1}{\ell},\quad \hat{p}_2=\frac{p_2}{\ell},\quad \gamma=\frac{k_1\ell}{N_0},\quad \hat{k}_2=\frac{k_2\ell^3}{EI}$$

The roots of equation determine the number of stable states of the mechanism. On normalizing the equation by the first coefficient, it can be rewritten as

$$\hat{x}^3+\left(\frac{5}{6\gamma}-\frac{5}{6}\hat{p}_1+\frac{5\hat{k}_2}{288\gamma}\right)x-\frac{5\hat{k}_2\hat{p}_2}{288\gamma}=0 \tag{7}$$

The number of the real roots is determined by the discriminant which can be written as

$$\Delta=-4\left(\frac{5}{6\gamma}-\frac{5\hat{p}_1}{6}+\frac{5\hat{k}_2}{288\gamma}\right)^3-27\left(\frac{5\hat{k}_2\hat{p}_2}{288\gamma}\right)^2 \tag{8}$$

In order to have more than one root, i.e., multistability, the discriminate Δ needs to be positive. This condition can be written as

$$\hat{p}_2<\left|\frac{576\gamma}{15\sqrt{3}\hat{k}_2}\left(\frac{5\hat{p}_1}{6}-\frac{5}{6\gamma}-\frac{5\hat{k}_2}{288\gamma}\right)^{3/2}\right| \tag{9}$$

This equation represents the stability boundary between monostable and bistable configurations. This is equivalent to the function Φ discussed in Sec. 1. In order to have real value, the following condition has to be satisfied:

$$\hat{p}_1>\left(\frac{1}{\gamma}+\frac{\hat{k}_2}{48}\right) \tag{10}$$

Therefore, once both conditions in Eqs. (10) and (9) are satisfied, the mechanism exhibits bistability behavior.

The positions of the stable states, s_1 and s_2, and unstable state, u, are the roots of the cubic polynomial given in Eq. (6), where the mechanism position at the roots x_{s1} and x_{s2} corresponding to s_1 and s_2, respectively, and the mechanism position of the root x_u corresponds to u.

The stiffness of the mechanism is the first derivative of the force, given in Eq. (6), with respect to $\hat{x}$ such that

$$\hat{k}=\frac{864}{5}\gamma\hat{x}^2+48-48\gamma\hat{p}_1+\hat{k}_2 \tag{11}$$

The stiffness of the mechanism at stable and unstable states can be evaluated at the position of the roots s_1, s_2, and u.

Remark IV: The stiffness of the mechanism is independent of displacement p_2.

4 Mechanism Behavior With Respect to Programming Input

The stability behavior of a PMM can be determined by calculating its strain energy, which depends on the value of p_1 and p_2. Figure 11 illustrates the normalized force for different values of p_1 and p_2. As discussed in our previous work [16], the qualitative behavior of PMM can be described in terms of the programing diagram, bifurcation diagrams, and stiffness maps. These diagrams are essential to illustrate the operation of mechanism as reconfigurable logic gate as it will be given in Sec. 6.

4.1 Programing Diagram. The programing diagram gives the relation between the number of stable states and p_1 and p_2, as illustrated in Fig. 12(*a*). The values of p_1^{cr} and p_2^{cr} representing the curve Φ are highlighted in Fig. 12(*b*). At $p_2=0$, the mechanism is bistable when $p_1>p_1^{cr}$. Otherwise, it is a monostable mechanism.

If $|p_2|<p_2^{cr}$, the mechanism is bistable for $p_1>p_1^{cr}$. The values of p_1^{cr} and p_2^{cr} representing the stability boundaries are given by Eqs. (10) and (9).

To describe the impact of p_1 and p_2 on the mechanism stability behavior qualitatively, we use both bifurcation and stiffness diagrams.

4.2 Bifurcation Diagram. A bifurcation diagram gives the relation between the position of stable and unstable states with the programing inputs. Since we have two programing inputs, we fix one of the programing inputs and vary the other to simplify the analysis of the mechanism behavior.

Figure 13 illustrates the impact of p_1 on the position of the stable and unstable states for given p_2. At $p_2=0$, the mechanism acts as buckled beams. One stable state exists, s_1. At $p_1=p_1^{cr}$, pitchfork bifurcation occurs. The stable state becomes unstable u and bifurcates into two stable states s_1 and s_2.

For non-zero $|p_2|<p_2^{cr}$, a saddle-node bifurcation occurs instead of a pitchfork bifurcation, at which a stable state and an unstable state emerge. For $p_2<0$, the bifurcation occurs at $x>0$ as shown in Fig. 13(*b*). Otherwise, the bifurcation node occurs for $p_2>0$ at $x<0$ as given in Fig. 13(*c*). The value of x at which bifurcation occurs depends on p_2. The higher $|p_2|$, the higher $|x|$.

In Fig. 14, the bifurcation diagram of stable and unstable states with p_2 for different values of p_1 is shown. If $p_1<p_1^{cr}$, the mechanism has one stable state s_1. As p_2 increases, the position of the stable state varies linearly. As p_1 increases, the relation between the position of stable states becomes nonlinear due to the softening effects of the beam under compression.

At $p_1=p_1^{cr}$, bifurcation occurs; two stable states and one unstable state emerge at $p_2=0$ as given in Fig. 14(*b*). As p_1 increases, the position of the bifurcation nodes varies. It should be noted that the bifurcation node is the value of p_2^{cr}, where the mechanism switches from bistability to monostability.

If we assume that the mechanism is positioned at s_2 in Figs. 13(*b*) and 13(*c*), on decreasing p_1, the mechanism position will slightly change until p_1 reaches the bifurcation node. Then, the stable state s_2 vanishes and the mechanism will switch to s_1 releasing energy. The energy released and distance traveled by the mechanism on switching depend on p_2.

Similarly, p_2 can be used to switch the mechanism between its stable states. We assume that the mechanism is placed at s_2 position given in Fig. 14(*c*). As p_2 increases, the mechanism slightly moves. Once $p_2>p_2^{cr}$, s_2 vanishes and the mechanism snaps from s_2 to s_1.

4.3 Stiffness Map. The stiffness map gives the relation between the stiffness of stable states, s_1 and s_2, and unstable state u versus the programing inputs p_1 and p_2 as illustrated in Fig. 15.

In the case of the stable state, s_1, it does exist for all values of p_1 and p_2. The stiffness is always positive, implying that the position is stable. p_1 has a higher impact on the stiffness. For large negative values p_1, the stiffness of EADPM is high and it decreases with increasing p_1. It reaches zero at $p_1=p_1^{cr}$ when $p_2=0$. After that, increasing p_1 increases stiffness. The stiffness of s_1 is slightly affected by p_2.

In the case of the unstable state, u, it exists only when the mechanism is bistable and its stiffness is always negative. Similar to s_1, the stiffness magnitude increases on increasing p_1 and it is slightly impacted by p_2 variations.

For the stable state, s_2, it does exist only when the mechanism is bistable. The stiffness varies significantly with p_1 and slightly with p_2. As p_1 increases, the stiffness increases.

5 Numerical Verification

COMSOL was used to verify our analytical computations. Table 3 gives the dimensions of the mechanism used in the simulation. The solid mechanics module is used with the linear elastic material model.

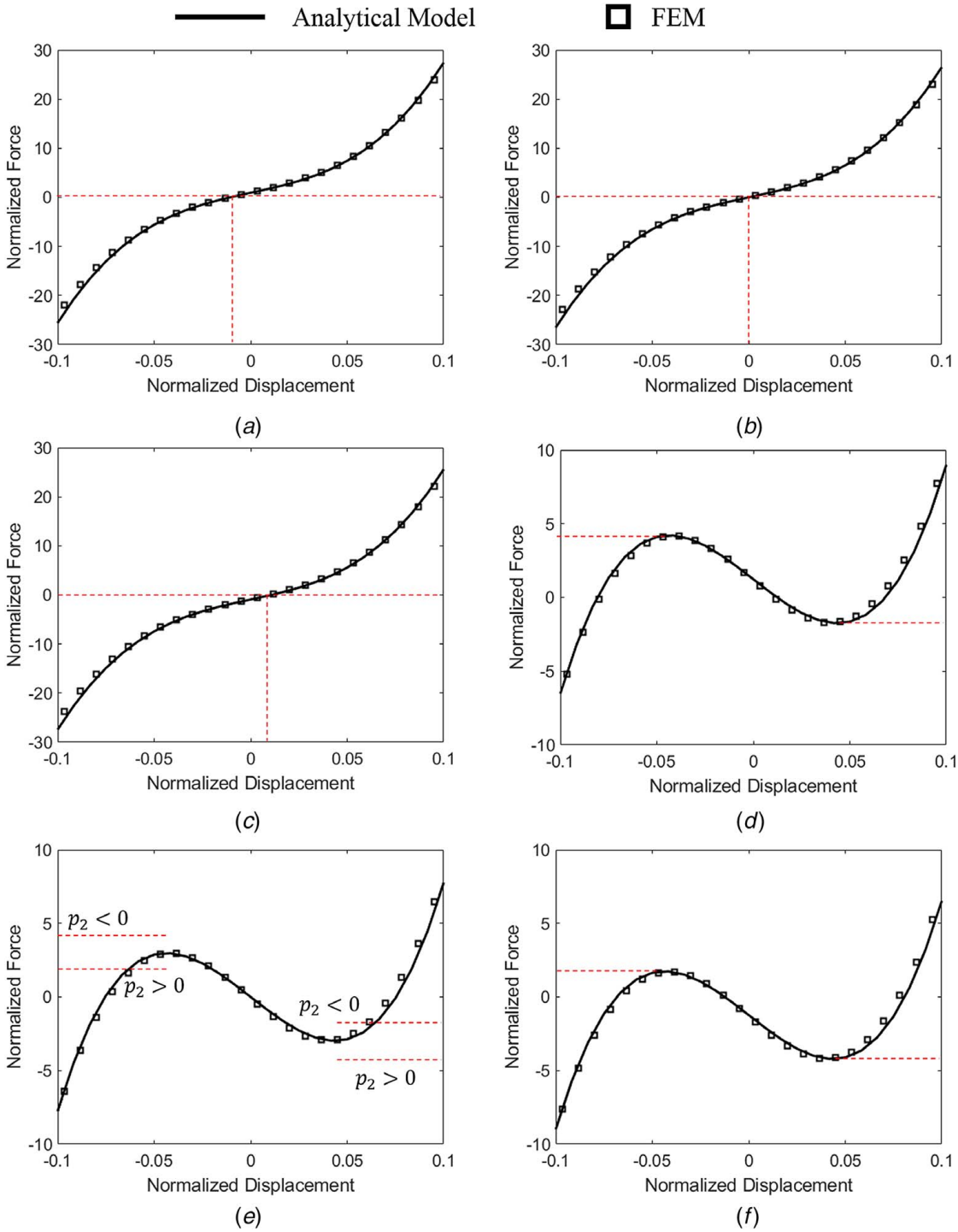

Fig. 11 Numerical verification of the model at (*a*) $p_1 = 0$ (mm), $p_2 = -3$ (mm); (*b*) $p_1 = 0$ (mm), $p_2 = 0$ (mm); (c) $p_1 = 0$ (mm), $p_2 = 3$ (mm); (*d*) $p_1 = 1.0$ (mm), $p_2 = -4$ (mm); (e) $p_1 = 1.0$ (mm), $p_2 = 0$ (mm); and (*f*) $p_1 = 1.0$ (mm), $p_2 = 4$ (mm)

The programing inputs were applied as prescribed displacement. A parametric sweep was performed. For each step, the strain energy was calculated where geometric non-linearity was considered. A mesh convergence test was performed to verify the validity of the solution.

The number of energy minima represents the number of stable states. The mechanism reaction force and stiffness were calculated as the first and second derivatives of the strain energy with respect to the displacement. Figure 19 gives the deformation of our logic gate based on the finite element analysis.

6 Logic Computation

The EADPM can be used as a reconfigurable logic device as illustrated in Fig. 16. The boundary condition of the mechanism is modified by imposing displacement, p_1 and p_2, representing the programing input. As discussed in Sec. 1, the programing inputs have two components, the reconfiguration component, r_1 and r_2, and the logic operand i_1 and i_2. The number of the stable states of the mechanism can be either "1" or "2" depending on the values of r_1 and r_2 with respect to the critical function, Φ, represented by p_1^{cr} and p_2^{cr}.

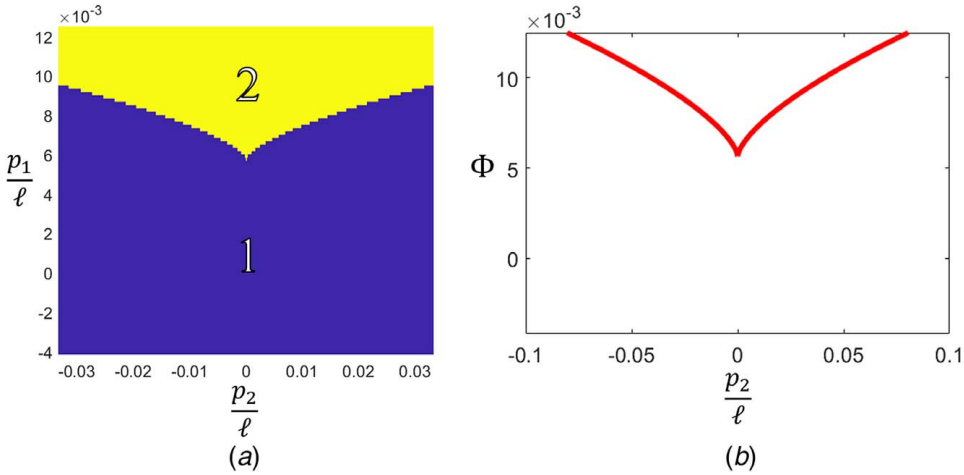

Fig. 12 (*a*) Programing diagram and (*b*) stability boundaries, representing the function Φ used for logic computation

The displacement r_1 and r_2 set the mechanism as bistable close to the stability boundary, as given in Fig. 16(*b*) and the mechanism is placed at s_2. On applying i_1 and/or i_2, the mechanism switches from bistability to monostability as given in Fig. 16(*c*). We denote the output "0," if the mechanism maintains its state s_2, i.e., bistable. The output is "1" if the mechanism is monostable and its stable state s_2 vanishes. As we change the values of r_1 and r_2, the mechanism snaps and energy release will occur for different combinations of i_1 and i_2. Therefore, the implemented logic function is changed.

As shown in Fig. 17, the number of implementable logic functions depends on the value of α_1 and α_2. Therefore, we sweep r_1 and r_2 for given values of i_1 and i_2 and possible logic functions are found using our analytical model. This is done as follows:

(1) Check that the mechanism is bistable at r_1 and r_2.
(2) If the mechanism is bistable, we calculate the discriminant from Eq. (8) for the four possible values of the input.
(3) If the discriminant for any of the four values is not positive, it indicates monostability and the output is "1" for this value.

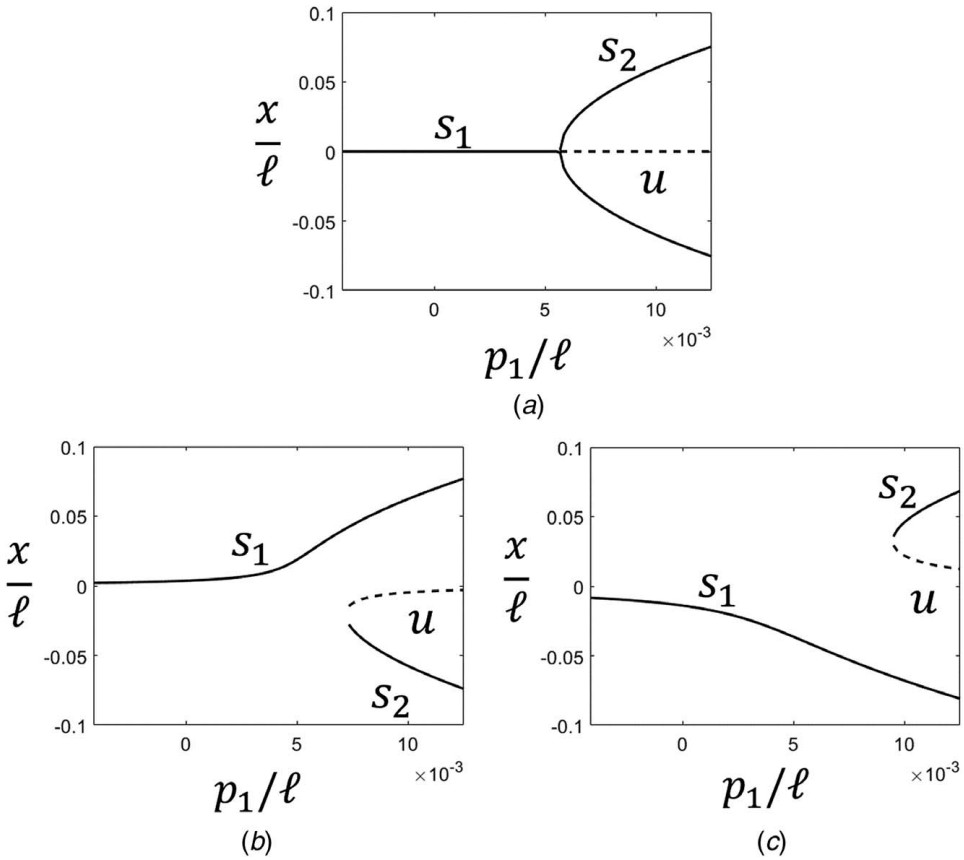

Fig. 13 Bifurcation diagram for stable and unstable states as a function of p_1 at (*a*) $p_2 = 0$ (mm), (*b*) $p_2 = 1$ (mm), and (c) $p_2 = -1$ (mm)

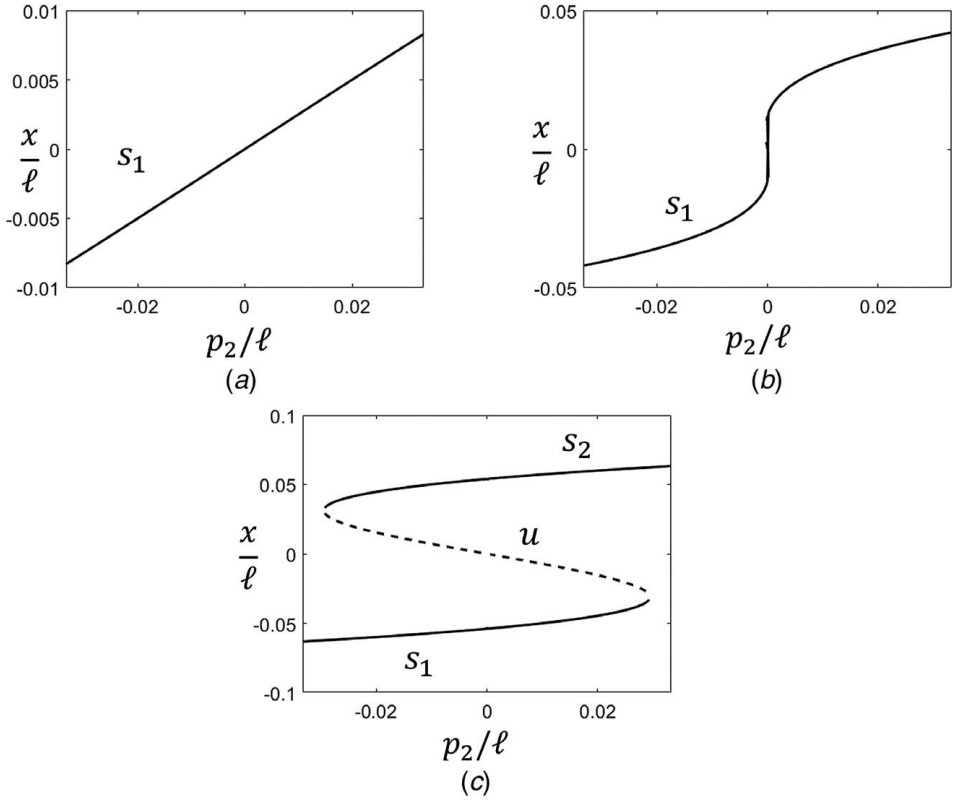

Fig. 14 Bifurcation diagram for stable and unstable states as a function of p_2 at (*a*) $p_1 = 0.2$ (mm), (*b*) $p_1 = 0.7$ (mm), and (*c*) $p_1 = 1$ (mm)

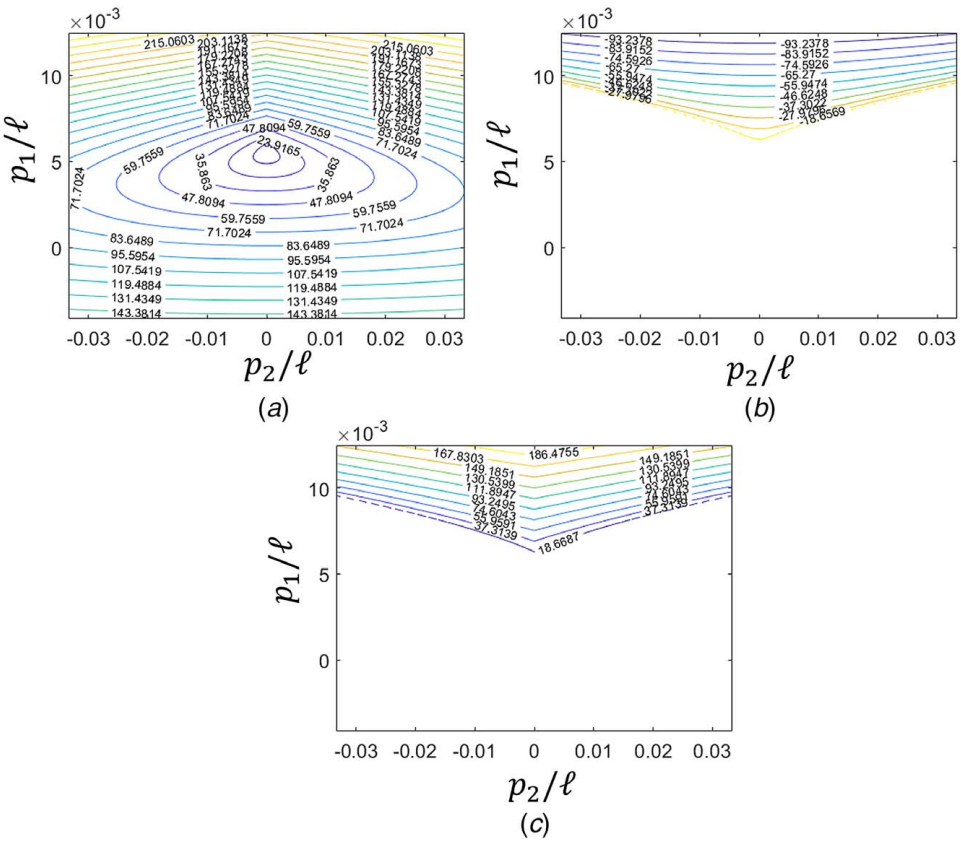

Fig. 15 Stiffness maps as a function of the programing inputs for (*a*) s_1, (*b*) u, and (*c*) s_2

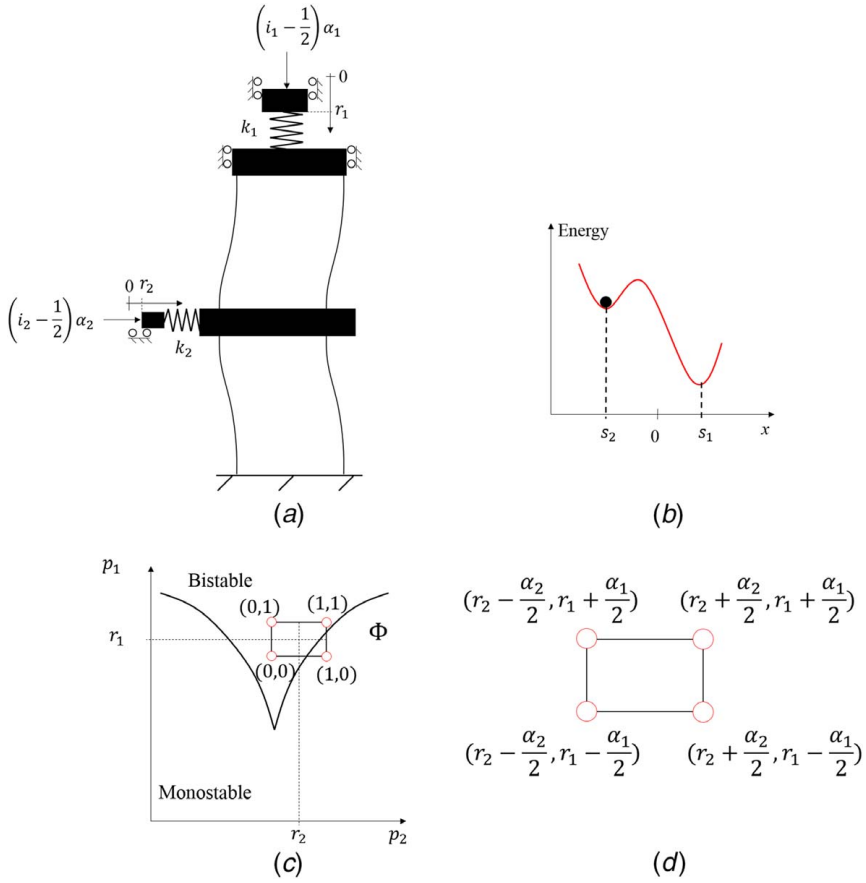

Fig. 16 Configuration of the EADPM for implementing logic operations: (*a*) mechanism, (*b*) energy profile corresponding to the mechanism deformation, (c) input values on the programing diagram, and (*d*) rectangle representing the four discrete values of the operands

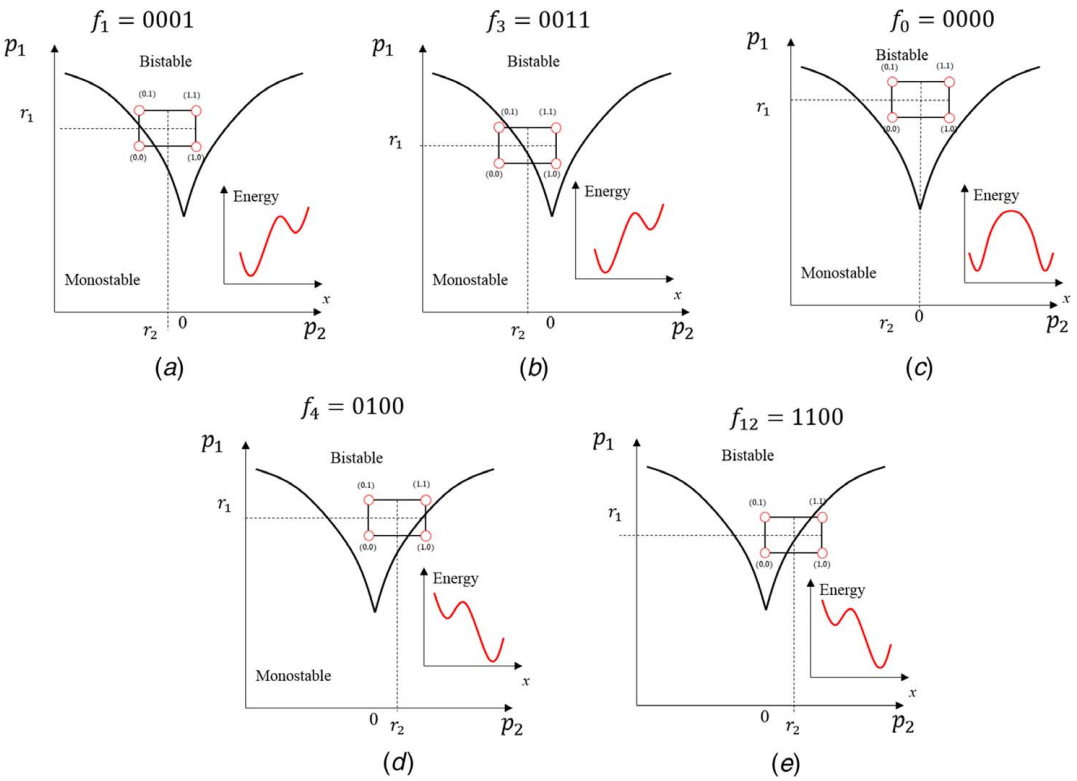

Fig. 17 Possible logic functions implemented by the EADPM for different values of r_1 and r_2: (*a*) NOT (i_2 OR i_1), (*b*) NOT i_2, (c) False, (*d*) i_2 OR (NOT i_1), and (e) NOT i_2

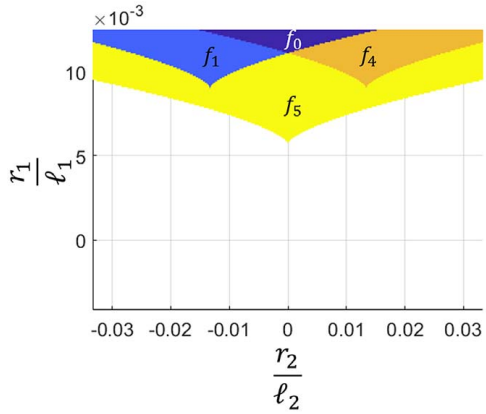

Fig. 18 Reconfiguration diagram for $\alpha_1 = 1.6$ (mm) and $\alpha_2 = 6$ (mm) based on the analytical model

(4) A reconfiguration diagram relating the implementable function with the set component of programing input, r_1 and r_2, is calculated.

Figure 18 shows the reconfiguration diagram for certain values i_1 and i_2 where the functions f_i are given in Table 1.

We built a demonstrator of the EADPM out of plastic to illustrate the different logic functions. Figure 19 shows the mechanism deformation based on numerical and experimental demonstration. The programing inputs are applied via a slider and a cam, where the position of the sliders gives the set component, r_1 and r_2, and the position of the cams represents the logic operands, i_1 and i_2. We verified that f_0, f_4, and f_5 are implementable when r_1 is positive. This is illustrated by the timing diagram in Fig. 20.

7 Future Work

We plan to use the discretization afforded by the reconfiguration diagram to construct a PMM with purely discrete reconfiguration input. We also plan to design new PMMs having different programing maps such that more of the possible 16 logical gates can be realized by the same mechanism. We will then combine mechanisms in order to implement more complex computations.

8 Conclusion

This paper provides new concepts to implement mechanical computation. We present a novel geometric method to construct a reconfigurable logic gate. We further construct a new programable multistable mechanism (PMM), the EADPM, to generate the geometric data for a reconfigurable logic gate. Based on our analytical calculations, 9 of 16 possible implementable logic operations can be constructed using this mechanism. We gave a complete qualitative analysis of the mechanism's stability behavior. The analytical

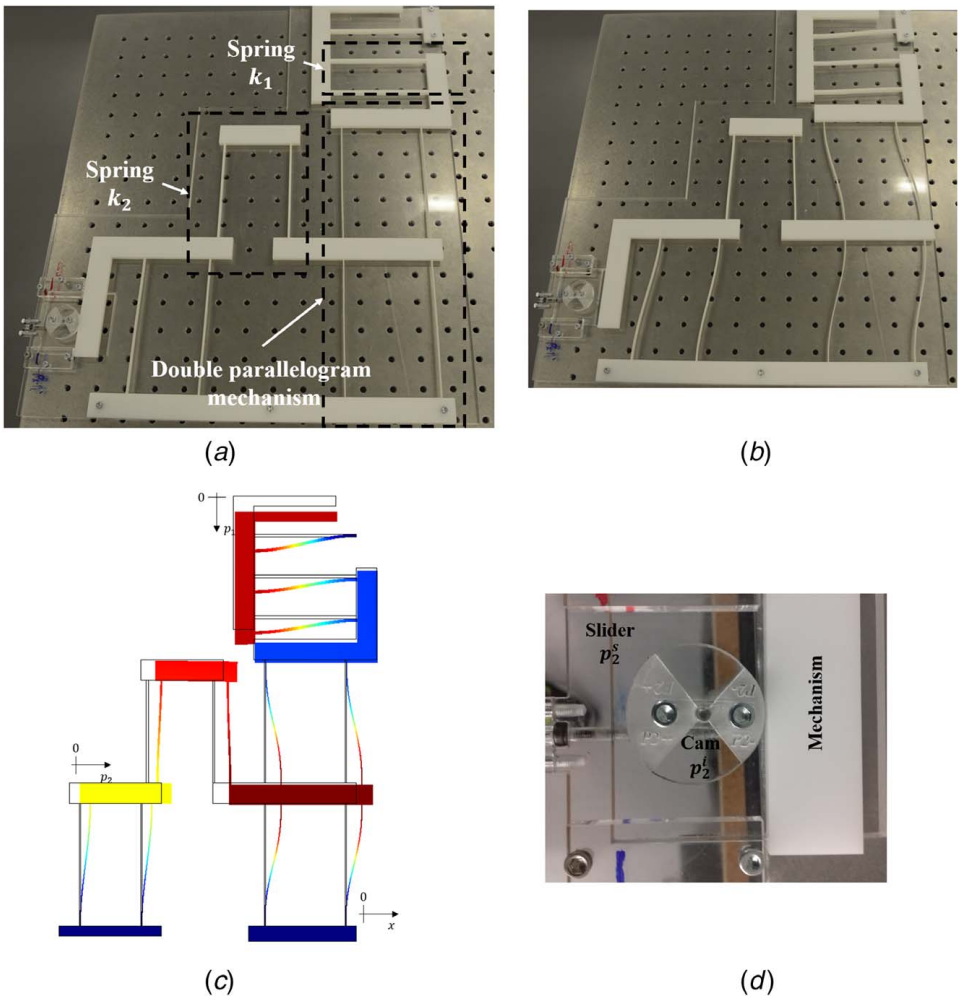

Fig. 19 (*a*) Physical demonstrator, (*b*) deformed in a stable state, (*c*) FEM simulation of the mechanism deformation, and (*d*) cam and slider for imposing i_2 and r_2, respectively

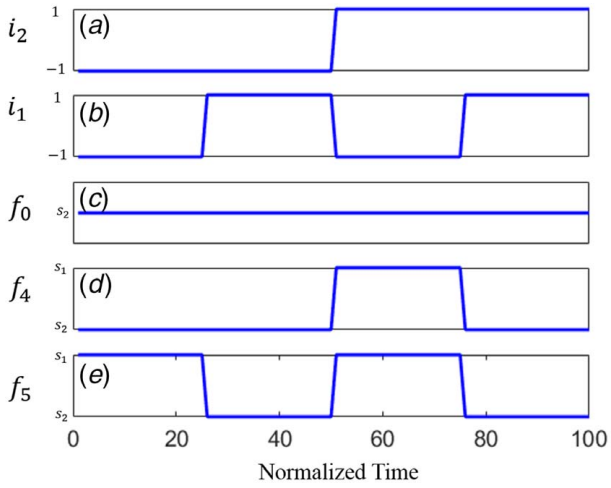

Fig. 20 Timing diagram based on experimental demonstration for (*a*) i_1, (*b*) i_2, (c) f_0 at $r_1 = 12$ (mm), $r_2 = 5$ (mm); (*d*) f_4 at $r_1 = 8$ (mm), $r_2 = 5$ (mm), and (e) f_5 at $r_1 = 7$ (mm), $r_2 = 5$ (mm)

computations were verified using numerical simulations and experimental demonstration.

Acknowledgment

M.Z. acknowledges the financial support provided by the Swiss National Science Foundation (Grant No. P2ELP2 184497).

References

[1] Mano, M. M., 1991, *Digital Design*, Prentice-Hall, Englewood Cliffs, NJ.
[2] Henein, S., Spanoudakis, P., Droz, S., Myklebust, L. I., and Onillon, E., 2003, "Flexure Pivot for Aerospace Mechanisms," 10th European Space Mechanisms and Tribology Symposium (ESMATS), San Sebastian, Spain, Sept. 24–26, pp. 285–288.
[3] Henein, S., Spanoudakis, P., Schwab, P., Giriens, L., Lisowski, L., Onillon, E., and Myklebust, L. I., 2004, "Mechanical Slit Mask Mechanism for the James Webb Space Telescope Spectrometer," Proc. SPIE 5487 Optical, Infrared, and Millimeter Space Telescopes, Glasgow, UK, Oct. 12, Vol. 5487, International Society for Optics and Photonics, pp. 765–776.
[4] Merkle, R. C., 1993, "Two Types of Mechanical Reversible Logic," Nanotechnology, **4**(2), p. 114.
[5] Hafiz, M., Kosuru, L., and Younis, M. I., 2016, "Microelectromechanical Reprogrammable Logic Device," Nat. Commun., **7**(1), pp. 1–9.
[6] Jiang, Y., Korpas, L. M., and Raney, J. R., 2019, "Bifurcation-Based Embodied Logic and Autonomous Actuation," Nat. Commun., **10**(1), p. 128.
[7] Song, Y., Panas, R. M., Chizari, S., Shaw, L. A., Jackson, J. A., Hopkins, J. B., and Pascall, A. J., 2019, "Additively Manufacturable Micro-Mechanical Logic Gates," Nat. Commun., **10**(1), p. 882.
[8] Henein, S., Cosandier, F., Rubbert, L., and Richard, K., 2017, *The Art of Flexure Mechanism Design*, EPFL Press, Lausanne, Switzerland.
[9] Howell, L. L., 2001, *Compliant Mechanisms*, John Wiley & Sons, New York.
[10] Qiu, J., Lang, J. H., and Slocum, A. H., 2004, "A Curved-Beam Bistable Mechanism," J. Microelectromech. Syst., **13**(2), pp. 137–146.
[11] Oh, Y. S., and Kota, S., 2009, "Synthesis of Multistable Equilibrium Compliant Mechanisms Using Combinations of Bistable Mechanisms," ASME J. Mech. Des., **131**(2), p. 021002.
[12] Jensen, B. D., and Howell, L. L., 2003, "Identification of Compliant Pseudo-Rigid-Body Four-Link Mechanism Configurations Resulting in Bistable Behavior," ASME J. Mech. Des., **125**(4), pp. 701–708.
[13] Jensen, B., Howell, L., and Salmon, L., 1999, "Design of Two-Link, In-Plane, Bistable Compliant Micro-Mechanisms," ASME J. Mech. Des., **121**(3), pp. 416–423.
[14] Jensen, B. D., and Howell, L. L., 2004, "Bistable Configurations of Compliant Mechanisms Modeled Using Four Links and Translational Joints," ASME J. Mech. Des., **126**(4), pp. 657–666.
[15] Wilcox, D. L., and Howell, L. L., 2005, "Fully Compliant Tensural Bistable Micromechanisms (FTBM)," J. Microelectromech. Syst., **14**(6), pp. 1223–1235.
[16] Zanaty, M., Vardi, I., and Henein, S., 2018, "Programmable Multistable Mechanisms: Synthesis and Modeling," ASME J. Mech. Des., **140**(4), p. 042301.
[17] Zanaty, M., and Henein, S., 2018, "Experimental Characterization of a T-Shaped Programmable Multistable Mechanism," ASME J. Mech. Des., **140**(9), p. 092301.
[18] Zanaty, M., Fussinger, T., Rogg, A., Lovera, A., Lambelet, D., Vardi, I., Wolfensberger, T. J., Baur, C., and Henein, S., 2019, "Programmable Multistable Mechanisms for Safe Surgical Puncturing," ASME J. Med. Dev., **13**(2), p. 021002.
[19] Raney, J. R., Nadkarni, N., Daraio, C., Kochmann, D. M., Lewis, J. A., and Bertoldi, K., 2016, "Stable Propagation of Mechanical Signals in Soft Media Using Stored Elastic Energy," Proc. Natl. Acad. Sci. U.S.A., **113**(35), pp. 9722–9727.
[20] Henein, S., 2001, *Conception des guidages flexibles*, PPUR presses polytechniques, Lausanne, Switzerland.