

# The Design of N-ary Mechanical Puzzles

by Goh Pit Khiam



Chinese Rings

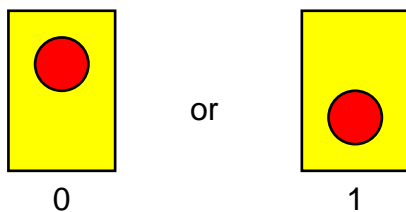


Tower of Hanoi

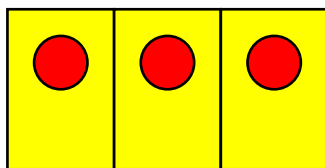
In an earlier article on The Design of Mechanical Puzzles based on the Gray Code published in Cubism For Fun, I posed the question of whether an n-ary puzzle can be designed without the long key piece similar to that found on the Chinese Rings puzzle. This article is a record of my journey to answer that question.

## Introduction

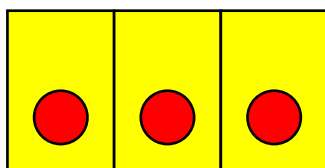
Suppose a puzzle has several uniform stages, each of which can be in one of two possible states:



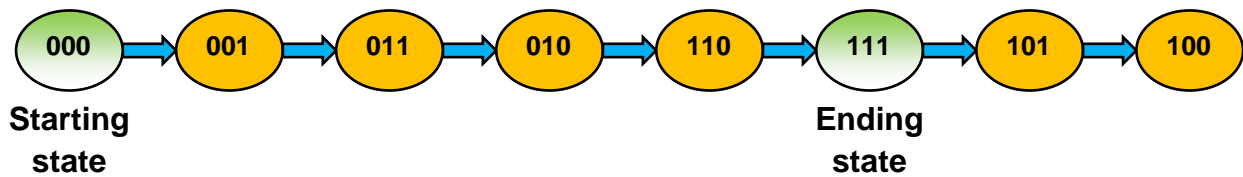
Assume that there are 3 stages in the puzzle and the puzzle is in the unsolved state when all the circles are up which we label as the state 000:



The puzzle is solved when all the circles are down and in the state 111:



If the mechanism in the puzzle enforces the stages to move through all possible combinations, for example:



the number of states in the puzzle would be  $2^3$ . In the above case, only 6 states are used in the solution since 101 and 100 are not used in the solution. If the ending state had been 100 instead of 111, there would be exactly  $2^3-1$  moves for the solution. Since we would like one part of the puzzle to move at one time to maximise the number of moves, it is natural for the states to go through the Gray code sequence where only one digit changes incrementally from one state to the next.

To generalize, an  $n$ -ary puzzle is one where each stage has  $n$  states, thus giving a solution requiring  $n^m-1$  moves for a puzzle with  $m$  stages.

In most  $n$ -ary puzzles like the Chinese Rings and more recent inventions like SpinOut, Binary Burr, Binary Key, Crazy Elephant Dance, Ternary Burr and TernKey, there is a long key piece which coordinates the movements of the stages. This is a drawback in the design because the key piece gets longer as more stages are added.

There is another problem related to the definition of the  $n$ -ary puzzle as pointed out by Goetz Schwandtner. I had also observed this at the time when I designed the Ternary Burr based on Markus Gotz idea in his Crazy Elephant Dance puzzle and had a short discussion with Markus on this. Although there are 3 states in each stage of the puzzle, the number of moves in the solution is only asymptotic to  $2^m$ . This is because many of the states lead to dead-ends and are not on the solution path.

There are thus two ways to define the  $n$ -ary nature of the puzzle:

- (a) the number of states in each stage of the puzzle.
- (b) the smallest number of moves required for the solution and how this is related to the number of stages in the puzzle.

## Representation of an m-stage n-ary puzzle

The state of an m-stage n-ary puzzle can be represented by an m-digit number:

$$q_{m-1}q_{m-2} \dots q_1q_0$$

where

$$0 \leq q_r \leq n - 1 \quad \text{for} \quad 0 \leq r \leq m - 1$$

By complying with the following rules depending on whether n is even or odd, all the base n Gray codes can be generated:

If n is even,

$$q_k \text{ can transit between } 2p \text{ and } 2p+1 \text{ where } \{p \in Z \mid 0 \leq p \leq \lfloor \frac{n}{2} \rfloor - 1\}$$

$$\text{iff } q_{k-1} = n - 1 \text{ and } q_r = 0 \text{ for } \{r \in Z \mid 0 \leq r \leq k - 2\}$$

$$q_k \text{ can transit between } 2p+1 \text{ and } 2p+2 \text{ where } \{p \in Z \mid 0 \leq p \leq \lfloor \frac{n}{2} \rfloor - 2\}$$

$$\text{iff } q_r = 0 \text{ for } \{r \in Z \mid 0 \leq r \leq k - 1\}$$

If n is odd,

$$q_k \text{ can transit between } 2p \text{ and } 2p+1 \text{ where } \{p \in Z \mid 0 \leq p \leq \lfloor \frac{n}{2} \rfloor - 1\}$$

$$\text{iff } q_r = n - 1 \text{ for } \{r \in Z \mid 0 \leq r \leq k - 1\}$$

$$q_k \text{ can transit between } 2p+1 \text{ and } 2p+2 \text{ where } \{p \in Z \mid 0 \leq p \leq \lfloor \frac{n}{2} \rfloor - 1\}$$

$$\text{iff } q_r = 0 \text{ for } \{r \in Z \mid 0 \leq r \leq k - 1\}$$

## A Ternary Example

Consider the three digit ternary Gray code sequence:

0	0	0
0	0	1
0	0	2
0	1	2
0	1	1
0	1	0
0	2	0
0	2	1
0	2	2
1	2	2
1	2	1
1	2	0
1	1	0
1	1	1
1	1	2
1	0	2
1	0	1
1	0	0
2	0	0
2	0	1
2	0	2
2	1	2
2	1	1
2	1	0
2	2	0
2	2	1
2	2	2

The leftmost digit changes between 0 and 1 only when the digits on the right are all 2's.

0	0	0
0	0	1
0	0	2
0	1	2
0	1	1
0	1	0
0	2	0
0	2	1
0	2	2

1	2	2
1	2	1
1	2	0
1	1	0
1	1	1
1	1	2
1	0	2
1	0	1
1	0	0

2	0	0
2	0	1
2	0	2
2	1	2
2	1	1
2	1	0
2	2	0
2	2	1
2	2	2

The leftmost digit changes between **1** and **2** only when the digits on the right are all **0**'s.

### A Quaternary Example

Consider now a three digit quaternary Gray code sequence:

0	0	0
0	0	1
0	0	2
0	0	3
0	1	3
0	1	2
0	1	1
0	1	0
0	2	0
0	2	1
0	2	2
0	2	3
0	3	3
0	3	2
0	3	1
0	3	0

1	3	0
1	3	1
1	3	2
1	3	3
1	2	3
1	2	2
1	2	1
1	2	0
1	1	0
1	1	1
1	1	2
1	1	3
1	0	3
1	0	2
1	0	1
1	0	0

2	0	0
2	0	1
2	0	2
2	0	3
2	1	3
2	1	2
2	1	1
2	1	0
2	2	0
2	2	1
2	2	2
2	2	3
2	3	3
2	3	2
2	3	1
2	3	0

3	3	0
3	3	1
3	3	2
3	3	3
3	2	3
3	2	2
3	2	1
3	2	0
3	1	0
3	1	1
3	1	2
3	1	3
3	0	3
3	0	2
3	0	1
3	0	0

The leftmost digit changes between **0** and **1** when the digits on its right are **3** followed by **0**'s.

0	0	0
0	0	1
0	0	2
0	0	3
0	1	3
0	1	2
0	1	1
0	1	0
0	2	0
0	2	1
0	2	2
0	2	3
0	3	3
0	3	2
0	3	1
0	3	0

1	3	0
1	3	1
1	3	2
1	3	3
1	2	3
1	2	2
1	2	1
1	2	0
1	1	0
1	1	1
1	1	2
1	1	3
1	0	3
1	0	2
1	0	1
1	0	0

2	0	0
2	0	1
2	0	2
2	0	3
2	1	3
2	1	2
2	1	1
2	1	0
2	2	0
2	2	1
2	2	2
2	2	3
2	3	3
2	3	2
2	3	1
2	3	0

3	3	0
3	3	1
3	3	2
3	3	3
3	2	3
3	2	2
3	2	1
3	2	0
3	1	0
3	1	1
3	1	2
3	1	3
3	0	3
3	0	2
3	0	1
3	0	0

The leftmost digit changes between **1** and **2** when the digits on its right are **0**'s.

0	0	0
0	0	1
0	0	2
0	0	3
0	1	3
0	1	2
0	1	1
0	1	0
0	2	0
0	2	1
0	2	2
0	2	3
0	3	3
0	3	2
0	3	1
0	3	0

1	3	0
1	3	1
1	3	2
1	3	3
1	2	3
1	2	2
1	2	1
1	2	0
1	1	0
1	1	1
1	1	2
1	1	3
1	0	3
1	0	2
1	0	1
1	0	0

2	0	0
2	0	1
2	0	2
2	0	3
2	1	3
2	1	2
2	1	1
2	1	0
2	2	0
2	2	1
2	2	2
2	2	3
2	3	3
2	3	2
2	3	1
2	3	0

3	3	0
3	3	1
3	3	2
3	3	3
3	2	3
3	2	2
3	2	1
3	2	0
3	1	0
3	1	1
3	1	2
3	1	3
3	0	3
3	0	2
3	0	1
3	0	0

The leftmost digit changes between **2** and **3** when the digits on its right are **3** followed by **0**'s.

Note that the rules for digit change apply to every digit even though only the leftmost digit is shown in both the above ternary and quaternary examples.

## Some Notable Designs

Two designs which I find to be particularly interesting are Jean-Claude Constantin's Kugellager and Bob Hearn's Sliding Piece Puzzle.

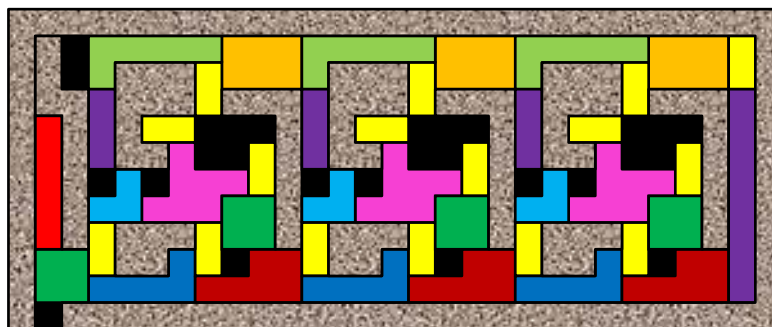
### Jean-Claude Constantin's Kugellager

Jean-Claude Constantin came up with an excellent puzzle called Kugellager:



It is a base-5 n-ary puzzle as each stage has 5 states. This puzzle is quite different in its mechanism compared to the other n-ary puzzles. Moreover, the number of moves in the solution is truly asymptotic to  $5^m$  as all states are used in the solution path. Unfortunately, it still requires a long key piece.

### Bob Hearn's Sliding Piece Puzzle



Bob Hearn designed this sliding piece puzzle where the objective is to slide the red rectangular piece on the left out from the box. In a discussion with Goetz on his compendium of n-ary puzzles, he asked if I could solve Bob Hearn's sliding piece puzzle on BurrTools because he had trouble defining the pieces so that BurrTools could solve it in a reasonable time. I was only able to solve a 2-stage version. The 3-stage version simply took too long no matter how the problem and pieces were defined or coloured. The solution to a two-stage version of the puzzle showed that many of the pieces moved together and there wasn't a need to have them as separate pieces. By "bandaging" them into a single piece, the combinatorial explosion would be reduced. I modified my sliding piece solver program to solve the "bandaged" 3-stage puzzle. To my surprise, the solution was not ternary as we had expected but turned out to have a straightforward linear solution. I informed Goetz about this and he was able to verify this manually using Lego. This was reported to

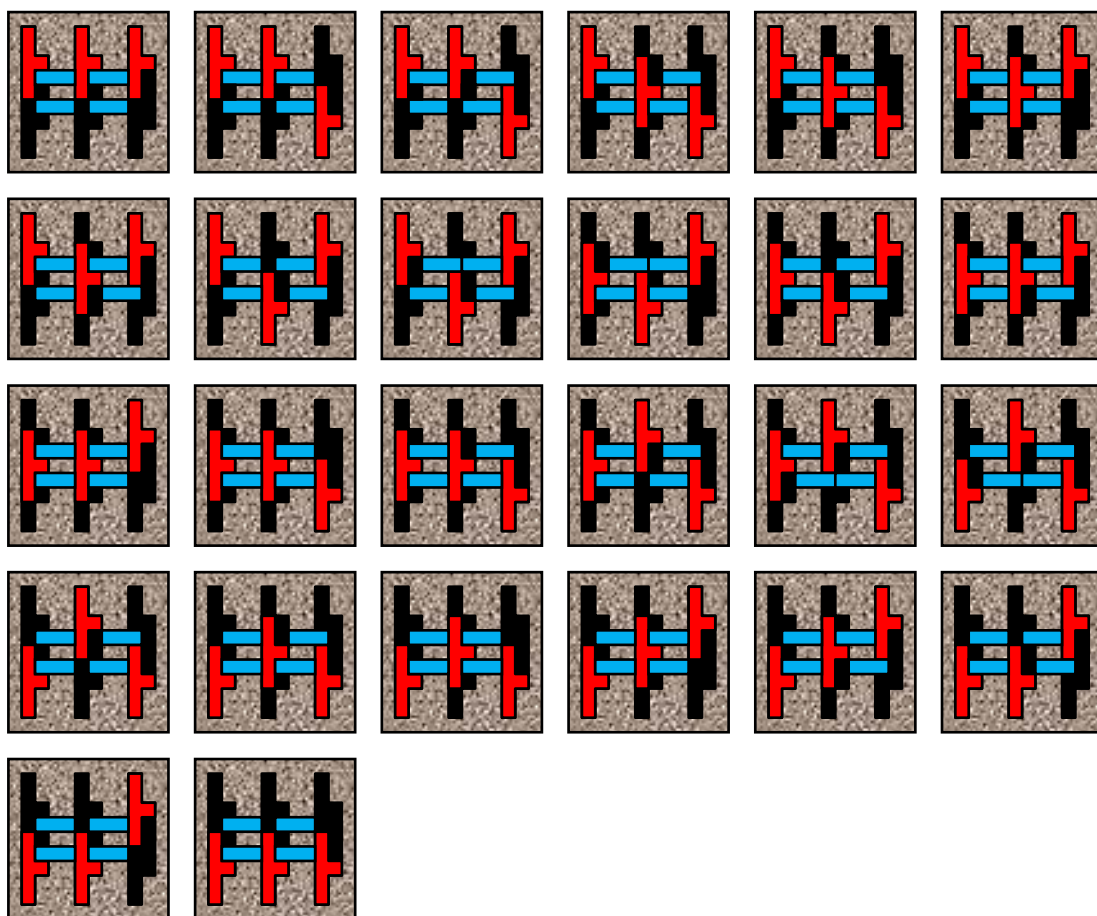
Bob in a hope that a fix could be found. When I first saw this puzzle, I liked it because it has a uniform structure.

### The Return of Tern-Key (or ReTern-Key)

In trying to find a 2D ternary puzzle that did not require the long key piece, I went back to the Tern-Key design. If the key piece was absent, the resulting puzzle had a solution length asymptotic to only  $m^2$ . The number of moves  $T_m$  for solving an  $m$ -stage puzzle is given by:

$$T_m = (2m - 1)^2$$

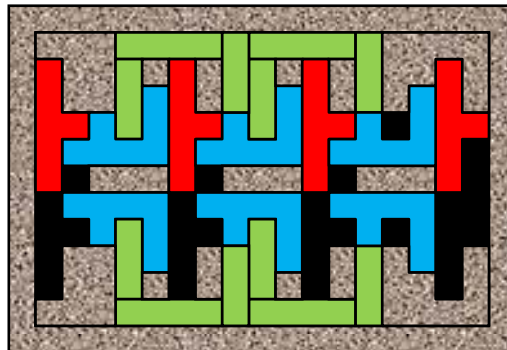
For example, a 3 stage puzzle requires only 25 moves to solve:



One way to solve this problem is to restrict the movements of the pieces so that:

- (a) A stage can change state between 0 and 1 if and only if all the stages to its right are at state 2
- (b) A stage can change state between 1 and 2 if and only if all the stages to its right are at state 0

Adding pieces to enforce the two rules result in a puzzle that is truly ternary – there are 3 states per stage and the number of moves for the solution is asymptotic to  $3^m$  if the number of stages is  $m$ . I call this puzzle the ReTern Key:



If the number of stages is  $m$ , the number of moves  $T_m$  is given by

$$T_m = 3T_{m-1} + 16(m - 2) + 6$$

The following table shows the relationship between  $m$  and  $T_m$  for some small  $m$ :

$m$	$T_m$
1	1
2	9
3	49
4	185
5	609
6	1897
7	5777
8	17433
9	52417
10	157385

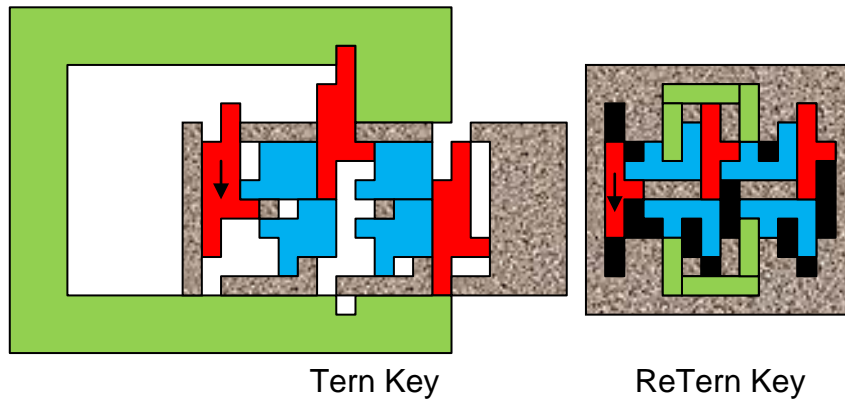
The closed-form expression for  $T_m$  is given by:

$$T_m = 8(3^{m-1}) - 8m + 1$$



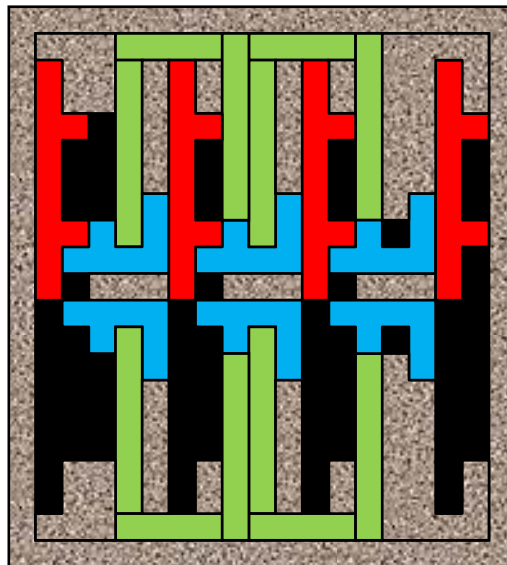
### Tern Key and ReTern Key

The solutions to Tern Key and ReTern Key are different. For the three stage puzzle, the first difference is shown in the following diagram. For the Tern Key puzzle, the leftmost piece can move from state 1 to 2 even though the pieces on its right are in states 0 and 2 respectively. For the ReTern Key puzzle, this can only happen when both the pieces on the right are in state 0.



### Extension of ReTern Key to Higher Odd Bases

This puzzle can be extended easily to base-5, base-7 or any higher odd powers. For example, the following is a base-5 version:



Jack Krijnen has derived the recurrence for the generalized base- $n$  ReTern Key puzzle having  $m$ -stages to be:

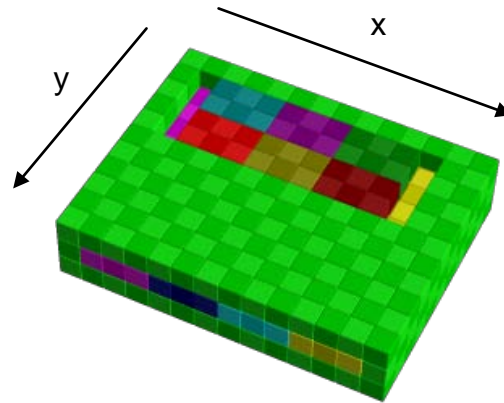
$$T_m = nT_{m-1} + (n - 1)(8m - 13)$$

or a closed-form expression of:

$$T_m = \frac{[4(n+1)n^{m-1} + 5n - 13]}{n-1} - 8m$$

### The NumLock Burr

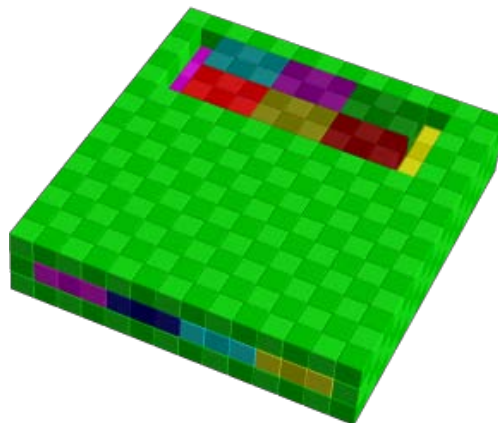
The idea in the ReTern Key can be extended easily to 3D resulting in a burr which has a very simple mechanism. I call this the NumLock Burr, a name to complement the ReTern Key. It turns out that all the pieces used in the puzzle are notchable.



This 4-stage puzzle requires 143 moves to remove the first piece. The number of moves is asymptotic to  $3^m$  for a puzzle with  $m$  stages. The number of stages can be increased indefinitely by increasing the number of stages uniformly in the x-direction. For  $m$  stages, the number of moves to remove the first piece  $T_m$  is given by:

$$T_m = 16(3^{m-2}) - 1$$

Another interesting property of this puzzle is that the base can also be increased to 5, 7, 9 or any higher odd base indefinitely by extending the pieces in the y-direction. For example, the base-5 puzzle with 4 stages is as follows:



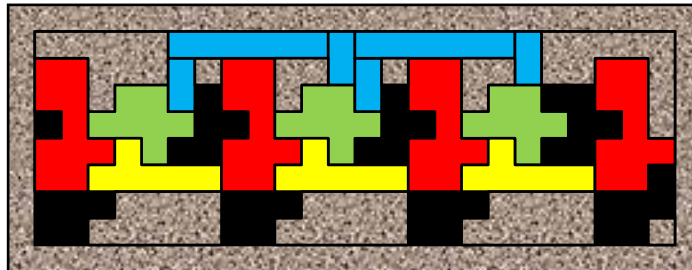
This puzzle requires 599 moves to remove the first piece. For the base-5 puzzle,

$$T_m = 24(5^{m-2}) - 1$$

## Even Bases

The ReTern Key design results in puzzles with odd bases. Using the idea, puzzles can be made with moves asymptotic to  $3^m$ ,  $5^m$ ,  $7^m$  and so on. Is there a similar design which gives puzzles with even bases?

After some thought, the following 2D design was found which I call the Modular Binary Key.

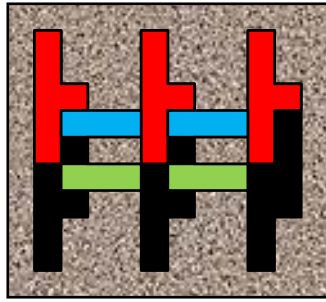


The puzzle is to bring the leftmost red piece to the bottom position, the rest remaining where they are. This four-stage puzzle requires 68 moves to solve. However, I was unable to extend the idea to higher even bases.

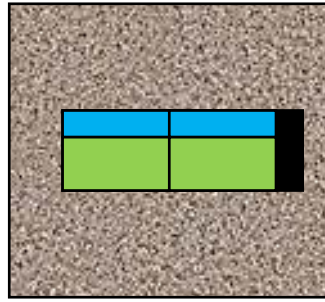
## A Comb Mechanism for Odd and Even Bases

During discussions with Jack Krijnen in December 2013, I found a new comb mechanism that allows both odd and even bases to be implemented. However, the mechanism requires at least 2 layers when implemented as a sliding piece puzzle. The diagrams in the following pages show how the concept is used to design puzzles of different bases.

**Base 3**

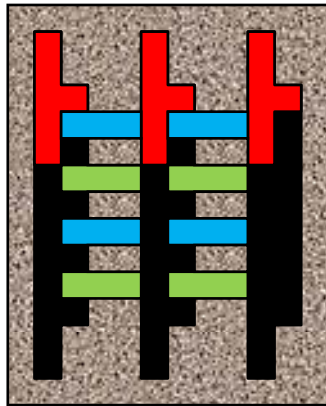


Top Layer

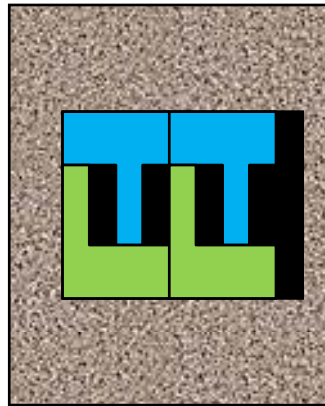


Bottom Layer

**Base 5**

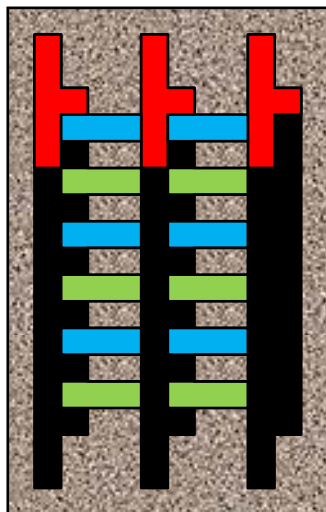


Top Layer

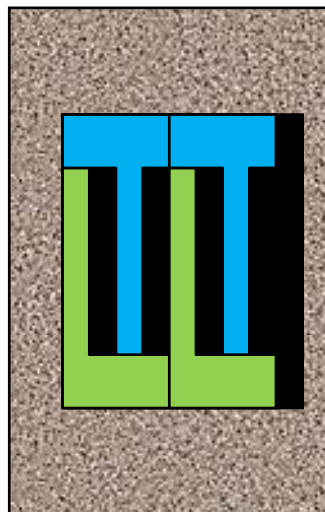


Bottom Layer

**Base 7**

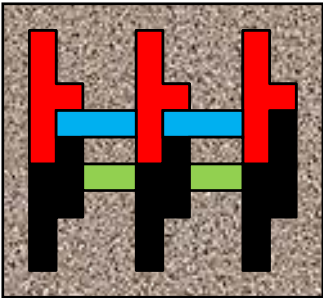


Top Layer

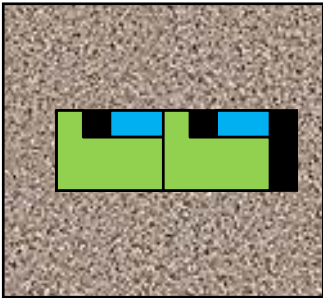


Bottom Layer

**Base 2**

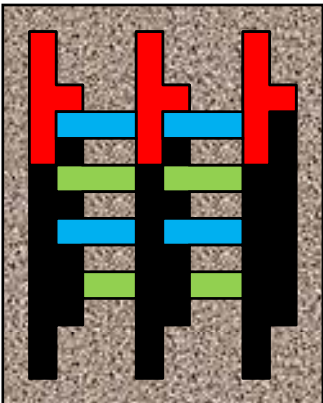


Top Layer

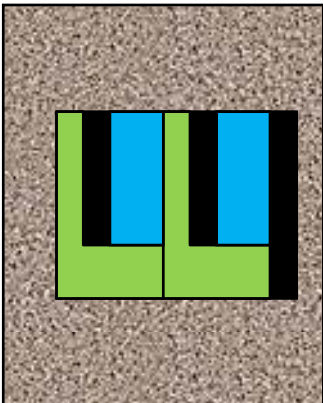


Bottom Layer

**Base 4**

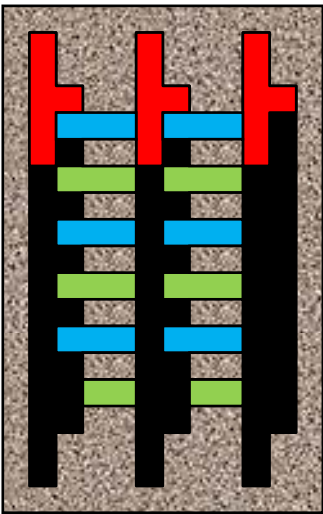


Top Layer

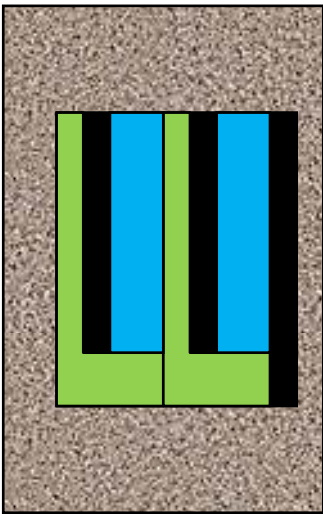


Bottom Layer

**Base 6**



Top Layer

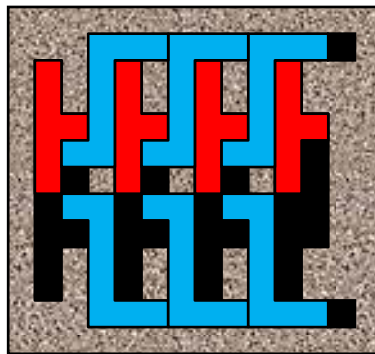


Bottom Layer

At the time when I came up with these 2-layer designs, I was unable to make a completely 2D puzzle out of this idea nor to extend this idea to burrs. As in the NumLock Burr, the pieces only go in two orthogonal directions. This is because if there were pieces going in the third orthogonal axis, these pieces will get longer as the number of pieces increase, much like the long key piece we were trying to eliminate in the design of these puzzles.

### ReTern Key Again

In December 2013, the ReTern Key design was simplified to the following:



The minimum number of moves to solve this puzzle is given by:

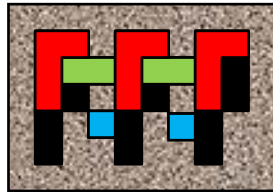
$$T_m = 5(3^{m-1}) - 2m - 2$$

where  $m$  is the number of stages.

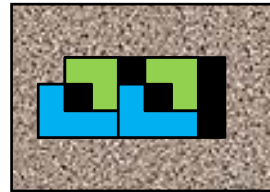
## Separating the Combs

A few days after the simplified ReTern Key was found, I realised that the comb structure could be separated as follows:

### Base 2

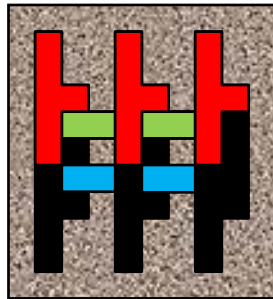


Top Layer

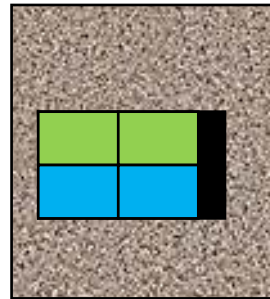


Bottom Layer

### Base 3

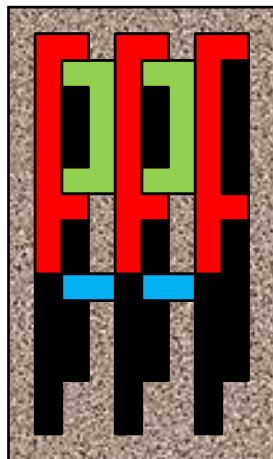


Top Layer

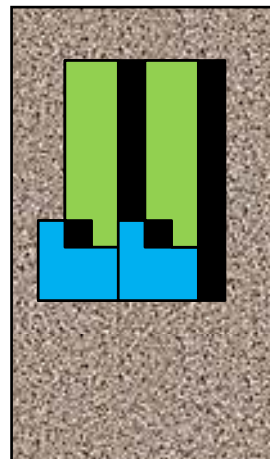


Bottom Layer

### Base 4

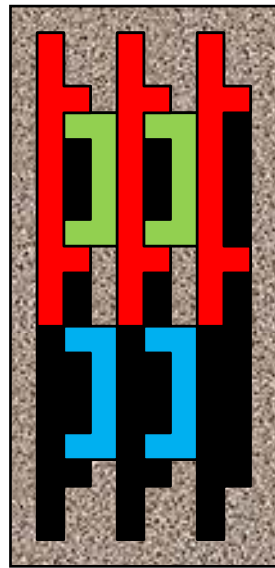


Top Layer

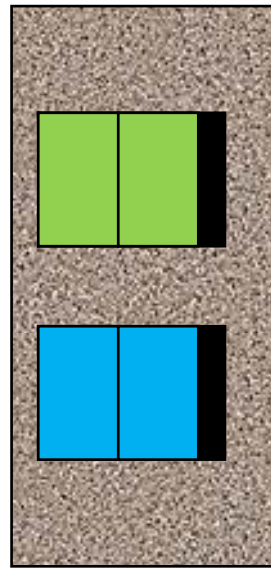


Bottom Layer

**Base 5**

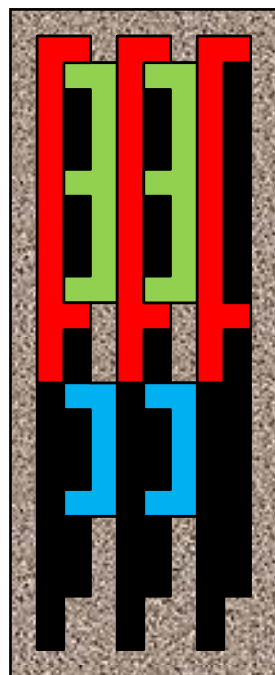


Top Layer

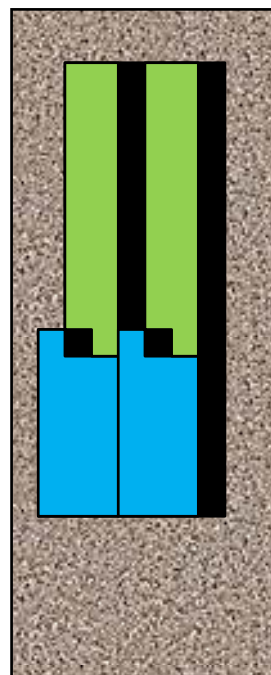


Bottom Layer

**Base 6**



Top Layer



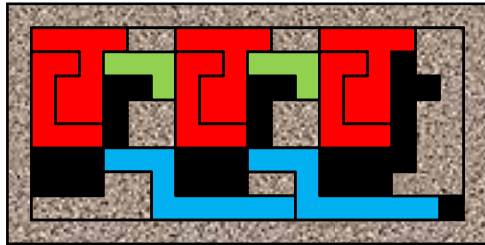
Bottom Layer

Perhaps this new structure would be simpler to implement in 2D?

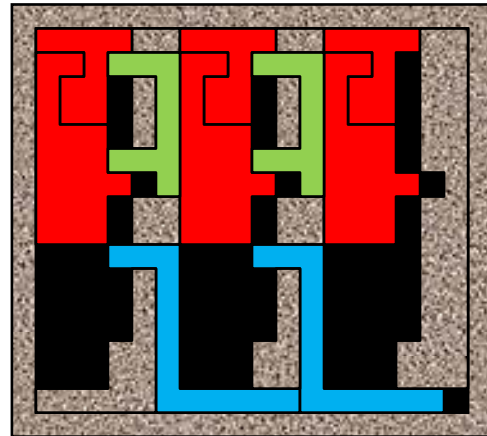


## 2D N-ary Puzzles with Even Bases

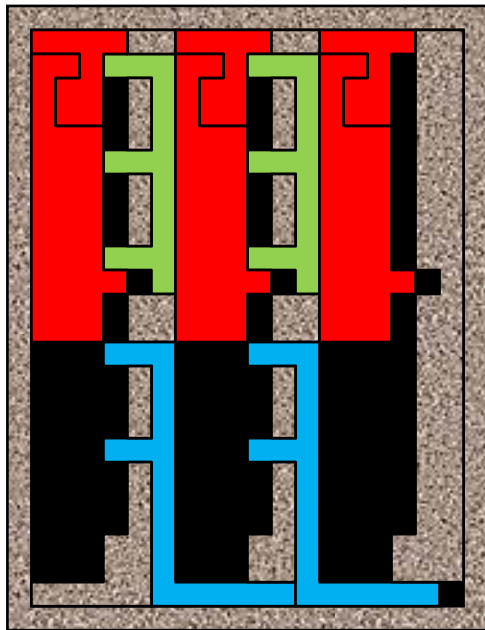
During a Saturday morning on 14 December 2013, I finally had an idea to make the comb structure work entirely in 2D:



Base 2



Base 4

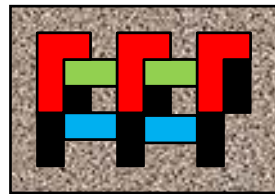


Base 6

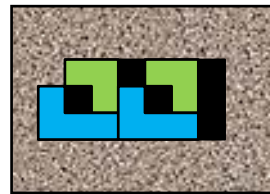
## Designs by Jack Krijnen

In discussions with Jack Krijnen, he simplified the 2-layer designs further and made them more uniform and elegant as follows:

### Base 2

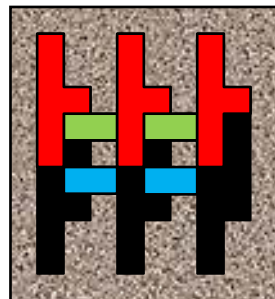


Top Layer

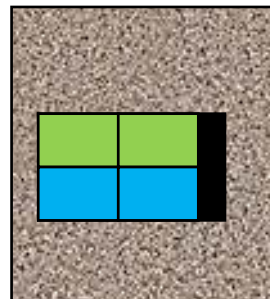


Bottom Layer

### Base 3

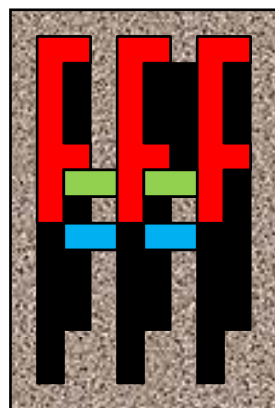


Top Layer

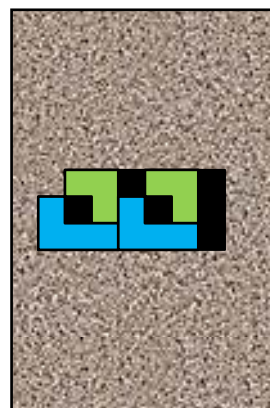


Bottom Layer

### Base 4

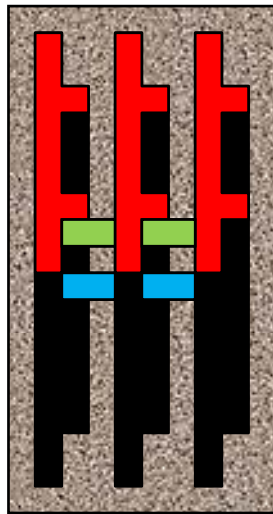


Top Layer

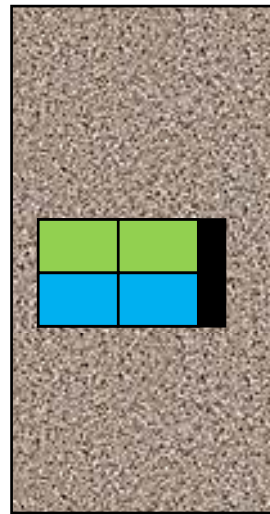


Bottom Layer

**Base 5**

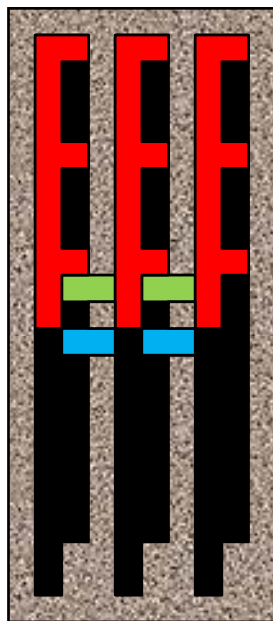


Top Layer

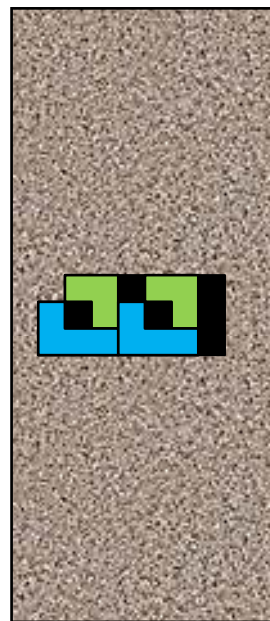


Bottom Layer

**Base 6**



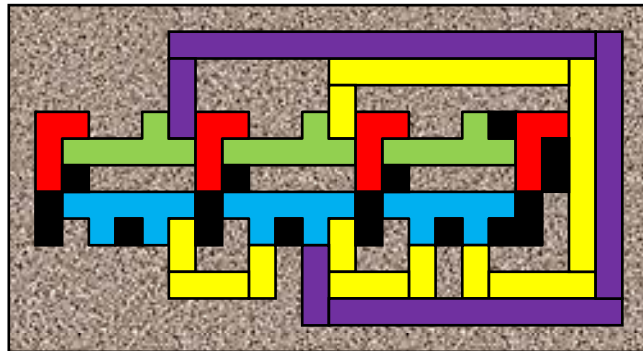
Top Layer



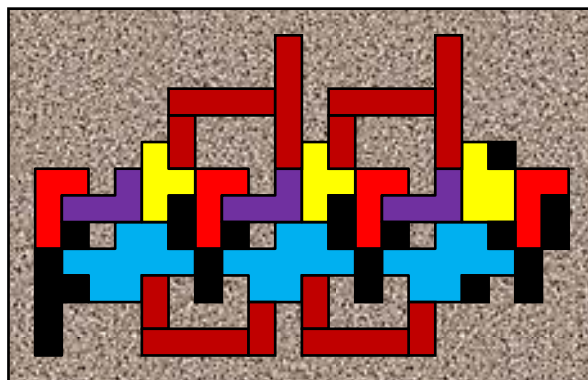
Bottom Layer

In these designs, only the red pieces get longer for higher bases.

Jack also came up with some ingenious 2D mechanisms for the even-based puzzles. The following diagrams show his 2D binary designs, which he called Long Run and Crossing respectively:



Long Run



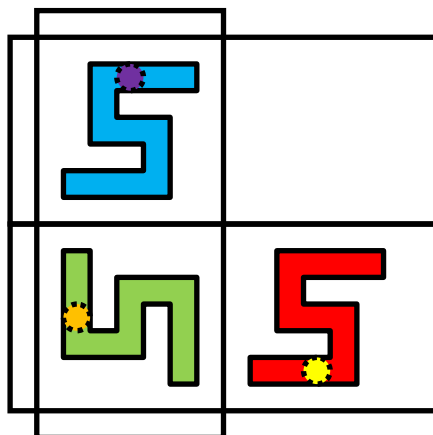
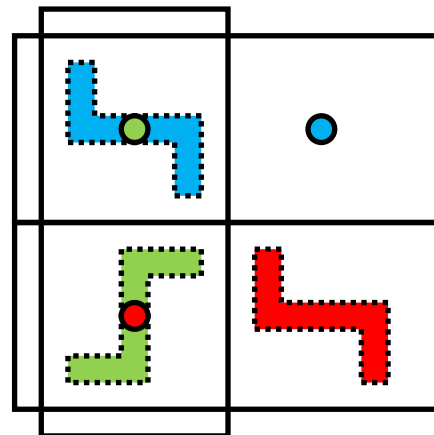
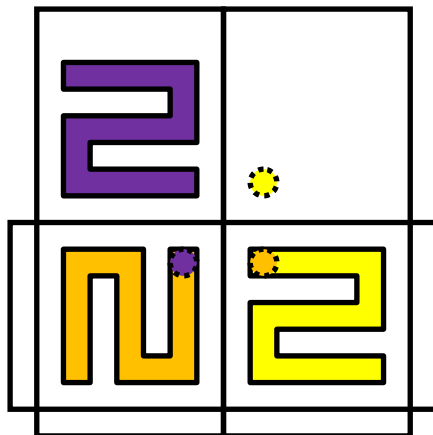
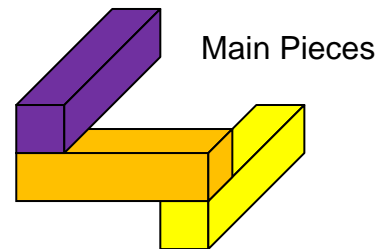
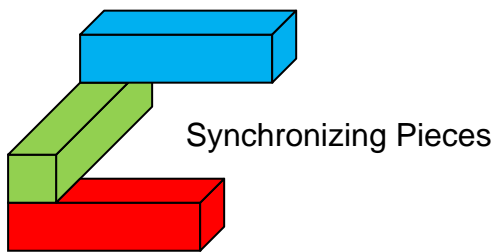
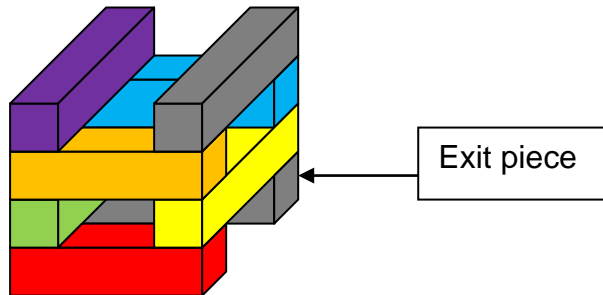
Crossing

Many thanks to Jack also for sending me an excellent even-based puzzle he made himself:



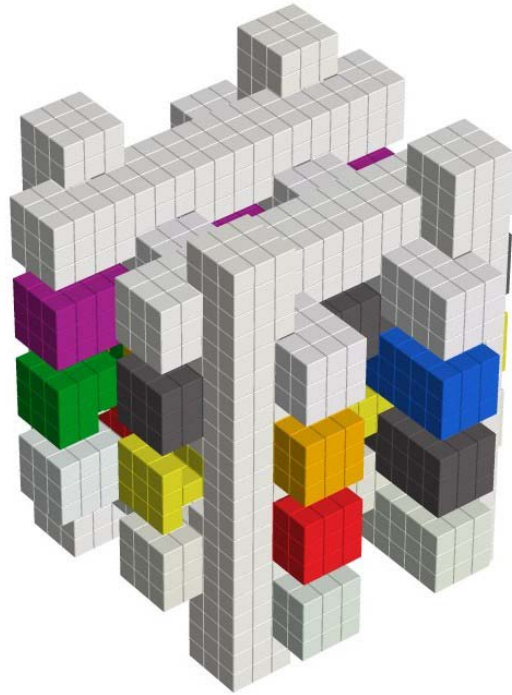
## The Double Helix

In early 2014, I suggested to Jack Krijnen about the possibility of extending the new ideas that we had to a tower burr. We took different paths of inquiry. My explorations took me finally to the Double Helix Burr. The name comes from the two intertwining spirals of pieces – the main ternary sequence pieces and the synchronizing pieces.



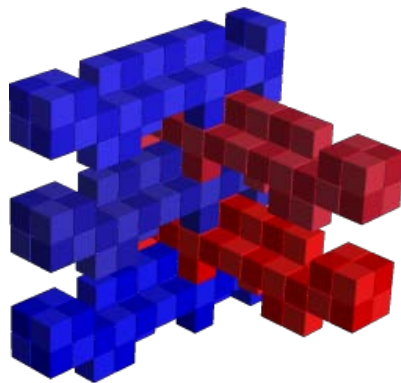
Piece Interactions

This is a 3-stage puzzle which requires 75 moves to remove the first piece. I used a cage of similar pieces with 3x3 cross sections to create a “visible” burr. The pieces creating the ternary sequence is sandwiched in the middle and the rest of the cage comes apart once the ternary sequence is complete. This puzzle requires 2 pieces per stage.



### One Piece Per Stage – The Power Tower Burr

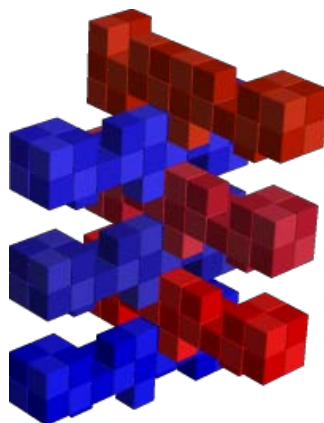
Meanwhile, Jack had the idea of doing away with the additional synchronizing pieces altogether. He started by looking for a binary puzzle but the search wasn't fruitful. However, when he turned his attention to a ternary puzzle, his efforts paid off. Over the weekend on 24 March 2014, he found very simple burr pieces which allowed a ternary puzzle to be implemented, with only one piece per stage! There are only two types of mirror-image pieces used in the puzzle and the solution is unique. The following shows an example of Jack's 5-stage ternary caged burr with 237 moves to remove the first piece.



This design is easily extended to all other higher odd bases.

### Even Bases

While experimenting with the pieces that Jack came up with for the ternary sequence, I shortened the pieces to see what would turn out. The result was a binary Gray-code sequence! There are also only two types of mirror-image pieces used in the puzzle and the solution is unique. The following shows the pieces for a 6-stage binary caged-burr. Again, the design is easily extended to all other higher even bases.



For the odd-based puzzles, the rest of the pieces come off easily after the first piece is removed. The even-based puzzles are even more interesting. Once the first piece is removed, the rest of the pieces are at their starting positions. This means that we now have to go through another n-ary sequence to solve a similar puzzle with one stage less. This continues until the last piece is removed.

The number of moves required to remove the first piece for an m-stage puzzle with n states is given by the closed form expression:

$$T_m = \frac{2(n^m - 1)}{n - 1} - m$$

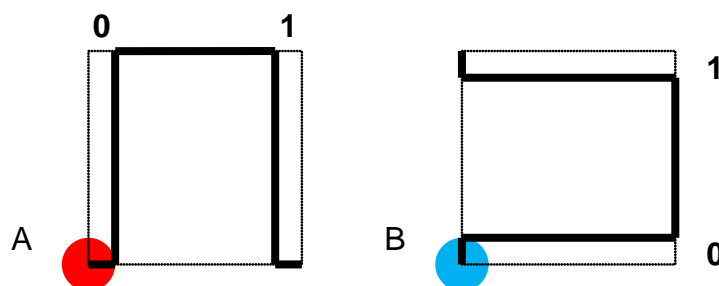
This is the same for both the even-based and odd-based puzzles.

This burr puzzle design is called Power Tower and is a joint discovery by Jack Krijnen and myself. The following shows some binary and ternary Power Towers made by Jack.



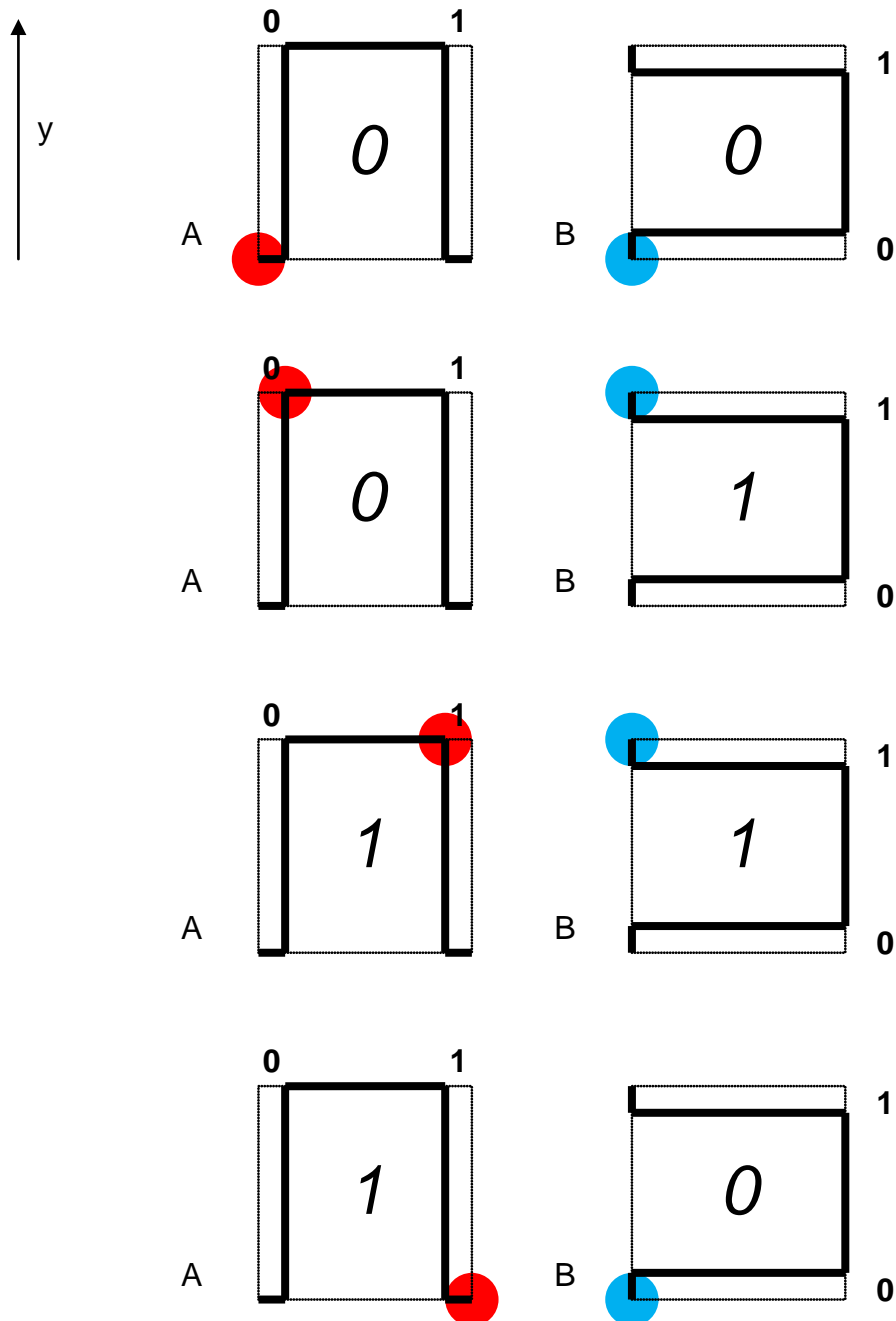
### A General N-ary Mechanism

On 28 March 2014, I discovered that the n-ary mechanism found by Jack Krijnen for odd bases could also be made to work for even bases. The idea can be visualized using paths on a square. Imagine two squares on which are inscribed paths with congruent shapes except for a 90 degree rotation:

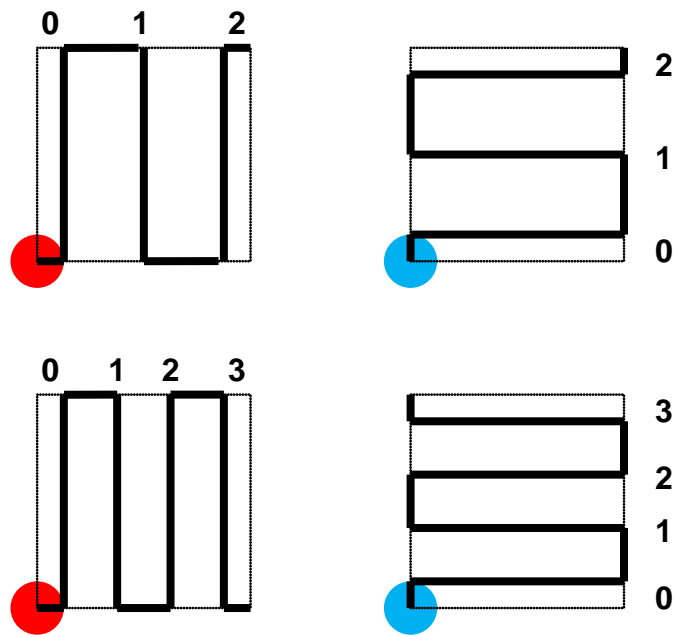




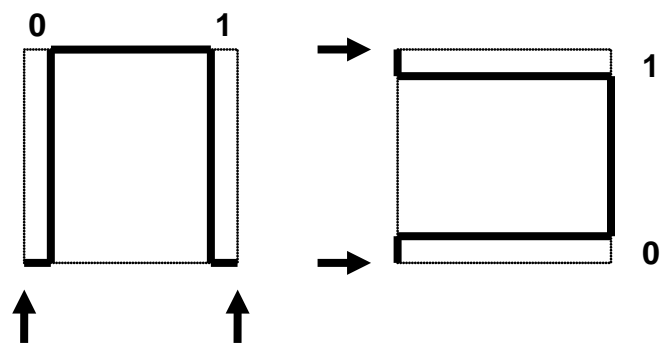
The paths go from 0 to 1. We want to get marker A to move from 0 to 1 in the left square. The constraint is that there is a marker B in the right square whose y-coordinate must always match that of marker A. We see that the states A and B will go through the binary Gray code:



This mechanism can be extended to any base. For example, the ternary and quaternary mechanisms are shown:



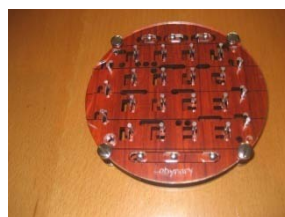
The key to Jack's discovery lies in the short endpoints in the path as indicated by the arrows which enforces the n-ary sequence.



Many designs so far do not have this critical feature which results in the number of moves being asymptotic to  $m^2$  instead of  $n^m$  where  $m$  is the number of stages and  $n$  is the number of states per stage. Examples of such puzzles include Oskar's Frequency Doubler and Jean-Claude Constantin's Labynary and N522 puzzles:



Frequency Doubler



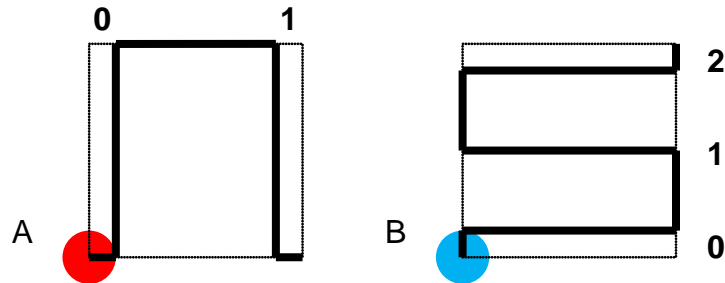
Labynary



N522

### Mixed Base Gray Codes

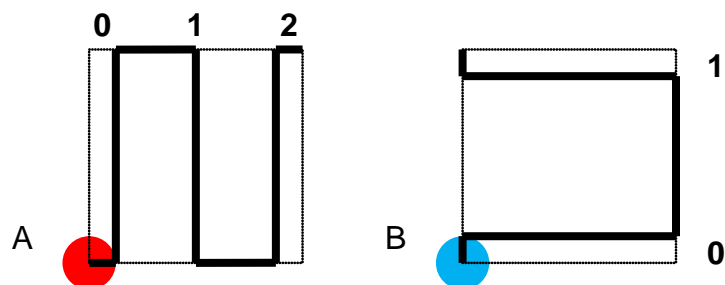
The above general Gray code mechanism can be applied to mixed base Gray codes. For example, if the leftmost digit has two states while the rightmost digit has three states:



This gives the following Gray code sequence:

0	0
0	1
0	2
1	2
1	1
1	0

On the other hand, if the leftmost digit has three states and the rightmost digit has two states:

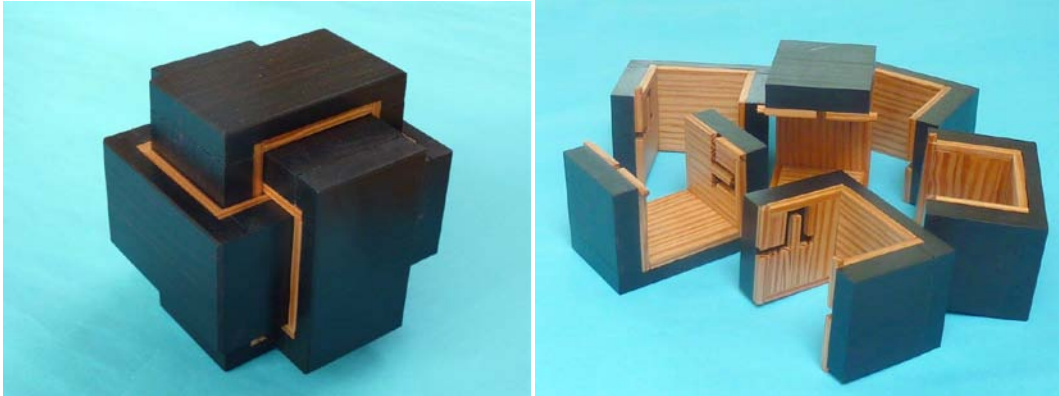


In this case, the Gray code sequence is:

0	0
0	1
1	1
1	0
2	0
2	1

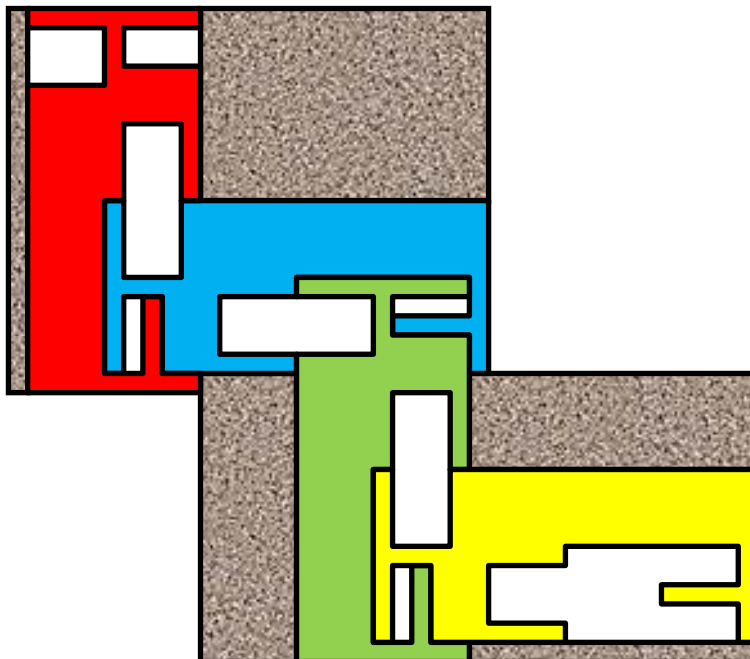
## The Power Box

A secret-opening box can be designed using the new n-ary mechanism with the external shape of Akio Kamei's Expansion Box. It is called the Power Box in general. The binary version is named Binary Box. Jack Krijnen made a prototype of it:



## Back to 2D

After coming so far, a question remains – Is a 2D n-ary puzzle possible with only one piece per stage? On the evening of 31 Mar 2014, I found the answer to this question. A simple binary puzzle is shown below whose objective is to remove the four pieces from the 2D frame.



The scheme can, of course, be easily extended to higher bases of any parity.

### Mixed Base Power Tower

In the month of July 2014, Jack wrote to say that he was playing with designs for the Power Tower with mixed bases but had some problems getting them to work. A few days later, the pieces were modified to work for any mixed based Power Tower. The pieces for both the even and the odd bases now follow an identical structure.

The number of moves required to remove the first piece for an  $m$ -stage puzzle with mixed bases can be given.

Let  $n = (n_{m-1}, \dots, n_1, n_0)$  be the bases from top to bottom in the tower and let  $T_m$  be the number of moves to free the first piece.

Then:

$$T_m = S_{m-1} + m - 2, \quad m \geq 1$$

where  $S_{m-1}$  is recursively defined by:

$$\begin{aligned} S_0 &= 2 \\ S_i &= n_i S_{i-1} + 2(n_i - 1)(i - 1), \quad 1 \leq i \leq m-1. \end{aligned}$$

Jack also formulated the Gray code for mixed base puzzles using short endpoints in general terms.

Let  $q_{m-1}, q_{m-2}, \dots, q_1, q_0$  represent the states of the  $m$  stages.

Then:

1.  $q_0$  can always transit between state 0 and 1, 1 and 2, etc (note: in the case of the Power Tower the intermediate states are passed on the move; the bottom piece always shows a binary behaviour, regardless its actual base).
2.  $q_k$  ( $k \geq 1$ ) can transit between 1 and 2, 3 and 4, etc  
iff  $q_k$  is followed by 0's only.
3.  $q_k$  ( $k \geq 1$ ) can transit between 0 and 1, 2 and 3, etc  
iff  $q_k$  is followed by an arbitrary number ( $\geq 0$ ) of odd-based stages in the high state ( $n_i - 1$ ), which in turn is optionally followed by exactly one even-based stage in the high state followed by an arbitrary number ( $\geq 0$ ) of (odd- or even-based) stages in the 0 state.

The following 4-stage example with bases 3322 illustrates this Gray code. Between each bracket is the rule applied for the transition.

```

0000 (1) 0001 (3) 0011 (1) 0010 (3) 0110 (1) 0111 (3)
0101 (1) 0100 (2) 0200 (1) 0201 (3) 0211 (1) 0210 (3)
1210 (1) 1211 (3) 1201 (1) 1200 (2) 1100 (1) 1101 (3)
1111 (1) 1110 (3) 1010 (1) 1011 (3) 1001 (1) 1000 (2)
2000 (1) 2001 (3) 2011 (1) 2010 (3) 2110 (1) 2111 (3)
2101 (1) 2100 (2) 2200 (1) 2201 (3) 2211 (1) 2210

```

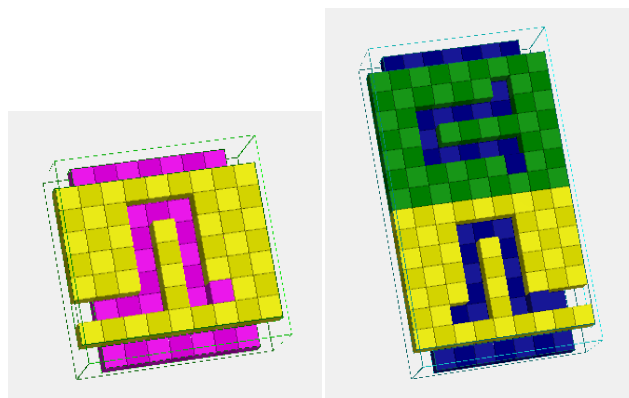
It shows a total of 36 state combinations as expected  $3 \times 3 \times 2 \times 2$ , all different. So it is a complete n-ary sequence.

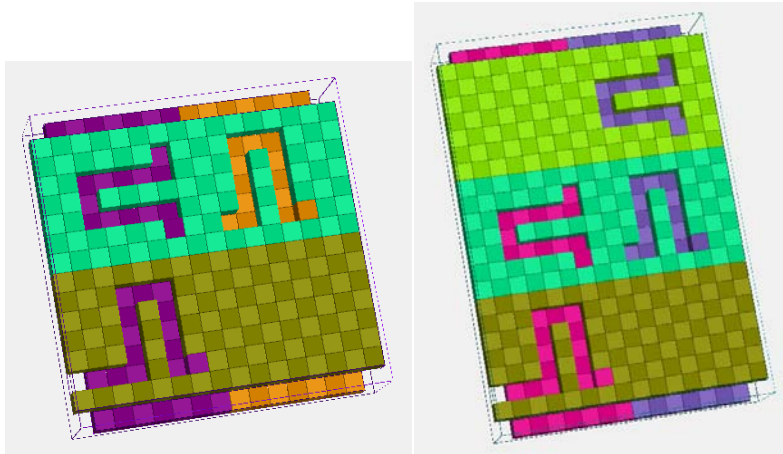


Mixed-Base Power Tower with base 2, 3 and 4 pieces

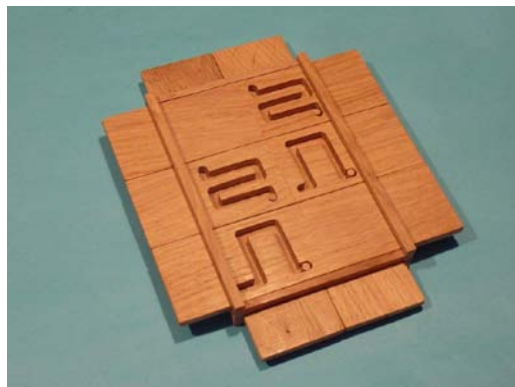
### Slots and Pins

With the new n-ary mechanism, puzzles based on the slots and pins mechanism can be designed which are truly n-ary. The following shows some simple binary puzzles. These puzzles give 5, 13, 29 and 61 moves for puzzles having one to four stages:



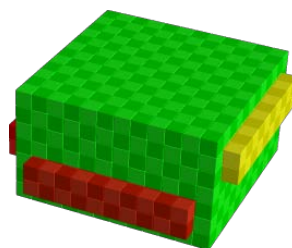


Such puzzles can be easily designed using the other bases. Bases may be mixed also. There is, however, an interesting difference between Slots and Pins and the Power Tower: where the bottom piece in the Tower always behaves binary. In Slots and Pins, the corresponding top right slot does behave according to its own base, with a stop at each state. The following shows a simple mixed-base Slots and Pins puzzle made by Jack Krijnen.



### **Racktangle**

After playing with the Mixed-Base Power Tower that Jack sent me, I decided to design a Slots and Pins type of puzzle in a similar way. The pieces are plates arranged in a spiral, each plate having a pin which interacts with the adjacent plate. With a special solid plate for the least significant digit, three base-2 plates and three base-3 plates, 8 different puzzles can be played. These are also the only solutions possible for fitting the plates into the box.



### Simplification of the Mixed-Base Power Tower

During discussions between Jack and myself, I found that the pieces in the Power Tower can be shortened by one unit to make it behave more like the Slots and Pins. Doing so also removes extraneous solutions and increases the level slightly.

For the new design, the number of moves to free the first piece becomes:

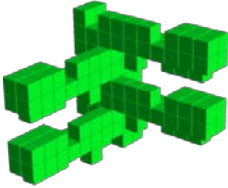
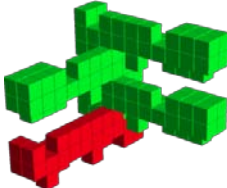
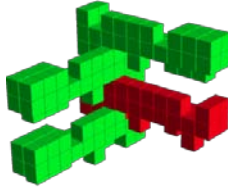
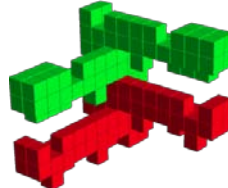
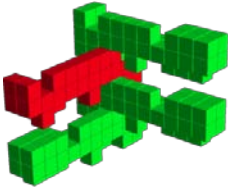
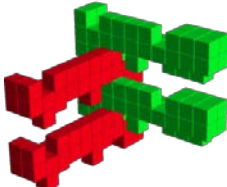
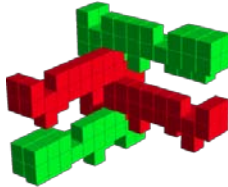

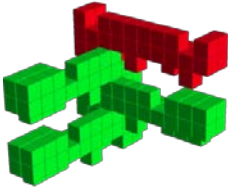
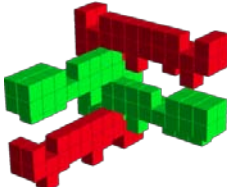
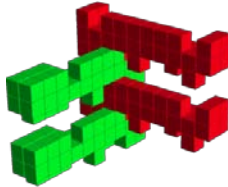
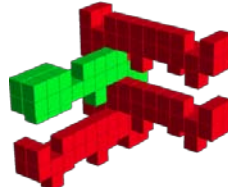
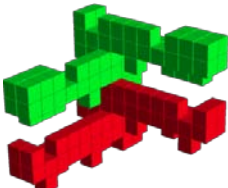
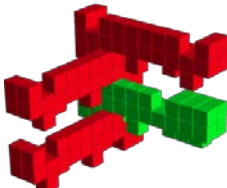
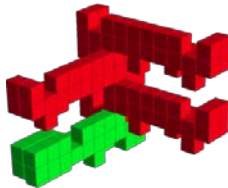

$$T_m = S_{m-1} + 2m - 3, \quad m \geq 1$$

where  $S_{m-1}$  is recursively defined by:

$$S_0 = 2$$

$$S_i = n_i S_{i-1} + 2(n_i - 1)(i - 1), \quad 1 \leq i \leq m - 1$$

The following shows the 16 different 4-Piece Mixed-Base Power Tower puzzles using the simplified Base-2 and Base-3 pieces:

 2222	 2223	 2232	 2233
 2322	 2323	 2332	 2333
 3222	 3223	 3232	 3233
 3322	 3323	 3332	 3333

In reality, there are only 8 distinct puzzles since the least significant digit has only two states regardless of whether the piece is base-2 or base-3.



## Conclusions

The Return Key and NumLock Burr designs show that it is possible to design an  $n$ -ary puzzle without the long key piece. These designs have the property that an  $m$ -stage puzzle requires number of moves asymptotic to  $n^m$  to solve. The ReTern Key shows that this is possible even if the puzzle is completely two-dimensional. They can even be easily extended to any odd base.

The Modular Binary Key design shows this can also be done in an even base, in this case, binary. With the new comb mechanism that I discovered, both odd and even bases can now be implemented. These puzzles can also be designed completely in 2D. The collaboration between Jack Krijnen and myself have simplified these designs and finally yielded the Power Tower, a truly  $n$ -ary burr with only one piece per stage without the need of any additional synchronizing pieces. The puzzle uses only two types of mirror-image pieces and the solution is unique. The burr design is applicable to both odd and even bases. I am surprised and delighted at the many discoveries made with this line of inquiry into the Chinese Rings puzzle, an ancient puzzle with still so many treasures to uncover.

## Acknowledgements

Special thanks to Dan Feldman for encouraging me to write my first article in CFF on mechanical puzzles based on Gray codes. It was in that article that I asked the question of whether such puzzles can be designed without the need of a long key piece to coordinate the piece movements. That question led to the discoveries in the current article. Thanks also to Goetz Schwandtner for the many discussions on closed-form expressions for the number of moves for such puzzles and also the writing of software for solving them. Goetz also gave many good suggestions to improve this article. Not forgetting Andreas Röver for making his excellent BurrTools program freely available to puzzle designers and which has been indispensable in this study of  $n$ -ary puzzles. Most of all, I am very grateful for the collaboration with Jack Krijnen. There has been much mutual inspiration since the days of 18-piece burrs and the exploration into  $n$ -ary puzzles has been no different.