Supporting Information

Barish *et al.* 10.1073/pnas.0808736106

SI Text

DNA Sequence Design. The design of the "tall rectangle" DNA origami is detailed in literature (1); it is composed of a 7,249-base long scaffold strand (single-stranded M13mp18; New England Biolabs) and 224 short (32-base) staple strands. Here, only 192 of the original staple strands were used. Sixteen strands along the left side of the origami were omitted so that the disordered loops of scaffold that remain prevent stacking interactions between origami. Another 16 strands along the right side were omitted to leave sites for the binding of tile adapter strands. All tile adapter strands and tile strands were designed using programs written in MATLAB, available at: www.dna.caltech.edu/DNAdesign/. These sequences were optimized to minimize the degree of complementarity between subsequences that are not intended to hybridize. Sticky-end sequences were designed to have similar hybridization energies. All strand sequences used in this study, including the tall rectangle origami staples that are identical to the original design (1), are listed in the *SI Appendix*.

Sample Preparation. Except for the scaffold strand and the 192 staple strands, all oligos were PAGE purified (Integrated DNA Technologies). The concentration of the scaffold strand was 10 nM, the staple strands 50 nM, and tile adapter strands 100 nM, except in the case of some binary counter experiments in which the scaffold concentration was 5 nM. (With 5–10 nM seed concentrations, seeded ribbons attached to origami were typically 0.5–1 μm long. In experiments not shown here, by decreasing the scaffold concentration to 1 nM, ribbons 5 μm long could often be observed.) Tile, tile adapter, and scaffold strands were quantitated with a UV spectrophotometer (Biophotometer; Eppendorf), whereas the concentration of staple strands were assumed to be that quoted by the manufacturer, without further quantitation. Concentrations of quantitated strands were adjusted to be within 10% of that intended and, based on previous experience, we expect the concentrations of staple strands to fall within ±10% of that intended. Note that it is not important that the concentration of the staple strands be accurate; origami folding is highly insensitive to the relative concentrations of staple strands as long as there is a few-fold excess of each staple strand over the scaffold (1). For the Variable-Width and Copy experiments all tile strand concentrations were 100 nM except for the strands in the repeatable (0/1)-blocks. Variable-Width experiments were performed in two different ways. In one set of "equal stoichiometry" experiments (Figs. S7 *C–F* and S8), the concentration of each 0-block strand was fixed at 100 nM like the rest of the tile strands. In another set of "normalized stoichiometry" experiments, the concentration of each 0-block strand was proportional to the number of 0-blocks in the desired ribbons (Fig. 3 and Fig. S7 *G–N*). Thus, in normalized stoichiometry experiments, the concentration of each 0-block strand was 200 nM for 8-wide ribbons, 300 nM for 10-wide ribbons, and 400 nM for 12-wide ribbons. For Copy experiments, the concentrations of the strands in the 0- or 1-blocks were adjusted to be proportional to the number of bits of a particular type that were to be copied. For example, for the "000000" ribbon, 600 nM of each 0-block strand was used. Similarly for the "010000" ribbon, 500 nM of each 0-block strand and 100 nM of each 1-block strand were used. And similarly for the "100001" ribbon, 400 nM of each 0-block strand and 200 nM of each 1-block strand was used. For binary counter experiments, all tile strand concentrations were 50 nM except for strands in the 0- or 1-repeatable blocks, which were each 250 nM.

Immediately before annealing, all strands (from 50 μM stocks) were mixed in together in 1× TAE/Mg^{2+} buffer (40 mM Tris-Acetate, 1 mM EDTA, 12.5 mM Mg Acetate). All water for buffers and dilution was purified by a Milli-Q unit (Millipore). For samples that were later to be ligated, 10× T4 DNA Ligase Reaction Buffer concentrate (NEB) was added to achieve a final concentration of 1× T4 DNA Ligase Reaction Buffer (50 mM Tris·HCl, 10 mM $MgCl_2$, 1 mM ATP, 10 mM DTT, 25 g/ml BSA; this is in addition to the 1× TAE/Mg^{2+} concentrations) and T4 Polynucleotide Kinase (NEB) was added to the strand mix (before annealing) to achieve a final concentration of 0.6 units/μL.

Annealing Protocols. All samples were annealed by using an Eppendorf Mastercycler. For samples that were going to be ligated later, a 1-h incubation at 37 °C was performed before annealing to allow T4 Polynucleotide Kinase to phosphorylate the 5′ termini of all oligos in solution. Annealing of the sample was performed in 4 stages. First, the temperature was kept at 90 °C for 5 min, to disrupt any intermolecular binding interactions or secondary structure. Second, the temperature was decreased linearly from 90 °C to 40 °C over 50 min (1 °C/min); this allowed tiles and the origami seed to self-assemble but did not allow tiles to associate with origami or to nucleate ribbons. Third, the temperature was decreased linearly from 40 °C to 25 °C over 15 h (1 °C/hour). During this stage, the temperature passed through the melting temperature of the ribbon. Ribbons nucleated on origami seeds roughly at this temperature, before unseeded ribbons had a chance to nucleate. The temperature decrease was presumably slow enough for the majority of free monomers to be added to the ribbon near the melting temperature (which decreases as free monomer tile concentrations decrease). Under this condition, addition of a monomer by two sticky ends was favored over the addition of a tile by a single sticky end and growth proceeded with few errors. Fourth, the temperature was decreased linearly from 25 °C to 20 °C over 5 min.

Sample Ligation and Purification. When unligated ribbons were deposited on a mica surface, many small structures (which we interpret as broken ribbons) were observed. Ligation with T4 DNA ligase seals nicks in the DNA backbone across sticky ends, between staple strands, and between component strands of the double tiles. In principle, the result is that very long strands should weave back and forth in ligated seeds and ribbons and at the interface between them. These long strands should have a higher melting temperature, require more untangling to dissociate and should make the nucleated structures more mechanically stable. Qualitatively, ligation and filter-based cleanup of samples appears to greatly decrease the number of observed fragments and to increase the number of long ribbons observed. As described above, samples to be ligated were phosphorylated before annealing. After annealing, 10 μL of a sample was diluted using 10× T4 DNA Ligase Reaction buffer (NEB), 10× TAE/Mg^{2+} buffer, and T4 DNA Ligase (NEB) to yield a 100-μL volume with a 1× final concentration of T4 DNA Ligase Reaction Buffer (50 mM Tris·HCl, 10 mM $MgCl_2$, 1 mM ATP, 10 mM DTT, 25 g/mL BSA), 1× TAE/Mg^{2+} buffer, and 20 units/μL of T4 DNA Ligase. The reaction mixture was then incubated for 8 or more hours at room temperature.

After ligation, the following cleanup steps were performed: (*i*) the 100-μL ligation reaction mixture was mixed with a 300-μL volume of TAE/Mg^{2+} buffer and gently vortexed, (*ii*) the sample was spun at 1,000 × g for 12 min at 4 °C in a YM-100 centrifugal spin filter (100-kDa nominal molecular mass limit; Millipore) leaving ≈20 μL retained by the filter, (*iii*) to facilitate sample recovery, the solution retained by the filter was pipetted gently up and down 5–6 times over the filter membrane, (*iv*) 200 μL of TAE/Mg^{2+} buffer was added to the same YM-100 filter, (*v*) the pipetting step from step *ii* was repeated, (*vi*) another 200 μL of TAE/Mg^{2+} buffer was added, bringing the total volume to ≈420 μL, (*vii*) the sample was centrifuged again at 1,000 × g for 6 min at 4 °C using the same filter, (*viii*) the pipetting step from step *iii* was repeated on the remaining 70–100 μL of solution, (*ix*) the spin filter was removed and placed upside-down in a fresh tube, (*x*) a microcentrifuge was used to transfer the sample to a fresh tube. The net effect of ligation and cleanup produced samples that were 7–10 times more dilute than samples that were annealed only (taking only dilution into account; some material may have been lost during filtering). The cleanup procedure also served to remove glycerol (present in the enzyme mixes) that otherwise would interfere with AFM imaging; the procedure may also remove or reduce the concentration of free tile monomers (≈45 kDa), T4 DNA ligase particles (≈68 kDa), and possibly some T4 polynucleotide kinase particles (≈132 kDa), but this has not been verified.

AFM Imaging and Statistics. AFM imaging was performed by using a Digital Instruments Multimode AFM (with a Nanoscope IIIa controller) in tapping mode under TAE/Mg^{2+} buffer. After annealing (or annealing, ligation, and cleanup), 5 μL of each sample was deposited directly on freshly cleaved mica and then covered by a 40-μL layer of TAE/Mg^{2+} buffer. For Variable-Width and Copy tile set statistics, an additional mixing step (to ensure homogeneity of the samples) was performed: The entire sample volume was pipetted (at normal speed) up and down 3–4 times just before deposition. The effectiveness of this mixing step was not measured; it is perhaps unnecessary.

Statistics for Variable-Width and Copy experiments were taken on samples that were unligated, because we were unsure whether the ligation and clean-up procedure affected the distribution of crystal types and sizes. In unligated samples, a large fraction of the ribbons were not attached to the origami seeds, indicating that they had fragmented during sample handling (in contrast to the ligated samples, which remained intact). For normalized stoichiometry Variable-Width statistics (Fig. 3), for each sample, >75 images (1 × 1 μm) were taken at nonoverlapping locations chosen at random to avoid sampling bias. Each sample's measurement tallied between 23,000 and 145,000 tiles (example images in Fig. S7 *H–J* and *L–N*). For equal stoichiometry Variable-Width statistics (Fig. S8), for each sample, >10 images were similarly taken. Each sample encompassed between 10,000 and 14,000 tiles (example images in Fig. S7 *B* and *D–F*). Once the data were collected, all of the tiles in the images were identified and hand-counted to calculate the percentage of total tiles that were in ribbons of each width. Only structures with well-defined ribbon edges were counted. Because the samples on which we measured Variable-Width statistics were unligated, this excluded small fragments, as can be seen in the background of Fig. S7 *E*, *H*, *I*, and *M*. Larger, two-dimensional lattices with ill-defined boundaries were also uncounted. Such lattices occurred almost entirely in unseeded samples in which they can be relatively common; we estimate that they account for ≈20% of the tiles measured in Fig. S7 *L* and *M*. In such samples, the coincidence of a lack of seeds and high concentration of 0-block tiles apparently allowed nucleation and growth of lattices. We note that in seeded samples, even with very high concentrations of 0-block tiles such as the "000000" Copy experiments, formation of lattices was greatly suppressed, with few or no such lattices observed. Finally, aggregation of ribbons occasionally made counting difficult. In most cases, however, ribbon edges could be used to segment the aggregates. Methods for taking statistics on Copy ribbons were similar: ≈75 images, each 1 × 1 μm in size, were collected for a unligated sample of "011010" Copy ribbons within 12 h of sample preparation. Statistics on errors for nucleation and copying are given in Fig. S9. Extensive statistics were not taken for Binary Counter ribbons, however, of 11 steric matching Binary Counters and 8 normal Binary Counters that were imaged at high resolution, all 19 nucleated correctly. Unlike for the Variable-Width and Copy tile set statistics, for the Binary Counter ribbons, we imaged interesting locations rather than locations chosen at random, so we cannot guarantee that the sample of crystals imaged at high resolution was not biased toward well-formed crystals.

Error correction by Tile Set Logic: Proofreading, Self-Healing, and Nucleation Barriers. The DNA tile sets used in this article incorporate several features that use logic (rather than chemistry or molecular structure per se) to reduce assembly errors. The design principles derive from an understanding of the processes by which errors occur. Consider growth of a crystal by one tile at a time. If every attachment is thermodynamically favorable—that is, if it attaches by at least two sticky ends—then growth is correct according to the tile set design. However, tiles will frequently arrive and attach by a single sticky end—an unfavorable attachment. In algorithmic self-assembly, such tiles often will be incorrect. (For example, at a site where a 0-block tile should attach by a green and a blue sticky end, instead the corresponding 1-block tile could attach by just the green sticky end, which it shares.) Fortunately, detachment rates depend strongly on the number of base pairs broken, so tiles attached by a single sticky end fall off much faster than those attached by two or more sticky ends. So long as the incorrect tiles falls off before any other tiles arrive, no permanent mistake in algorithmic growth is made. In many tile sets, however, an incorrect tile can be locked in by subsequent favorable tile attachments, and crystal growth proceeds with the incorrect information—i.e., a growth error has occurred. In contrast, other tile sets have the property that after an initial unfavorable attachment, there is no immediately subsequent tile that can attach favorably—thus, at least two unfavorable attachments must occur in immediate succession without either tile falling off before they can be locked in. That is, two rare events must occur in quick succession before an error can be made, rather than just one. As a result of this difference, the error rate in the second type of tile set is predicted to be approximately the square of the error rate in the first type, under comparable growth conditions—e.g., compare a 4% error rate with a 0.16% error rate (2). Remarkably, essentially any type of tile set of the first type can be transformed into a related (and larger) tile set of the second type that creates the same pattern, but with the lower error rate. This technique—which designs tile sets so that all information is represented redundantly and all logical calculations are performed redundantly—is known as the proofreading transformation (2) and has recently been shown to greatly decrease copying error rates in DNA tile crystals (S. H. Park, personal communication).

The Copy tile set was designed by applying a variant of the proofreading transformation in which a 2 × 2 block of tiles represents a single bit of information. Thus, each bit is encoded redundantly in multiple tiles rather than in a single tile; consequently, multiple sticky-end mismatches are required to flip a bit to the wrong value, rather than a single mismatch. Let us examine this in detail. In order for the Copy tile set to be able to copy arbitrary bit strings, there must be a choice of tiles at every bit position. This means that between blocks representing "0" or "1" bits, there are identical sticky ends (see Fig. S2, green sticky ends), along the lattice direction in which information is not passed (the top-left to bottom-right diagonal). Growth occurs in a zig-zag fashion back and forth along this diagonal. Consider a tile being added during growth from top left to bottom right, and let it be the leftmost tile of a block, the first tile of a block to be added—this makes it a Z9 or ZB tile. Consider that the position being copied has a 0. Then it presents "N" and "0" sticky ends for binding. A Z9 tile from a 0-block could bind at this position by two sticky ends; this represents correct growth and would initiate growth of a 0-block. But a ZB tile from a 1-block could also bind at this position by a single "N" sticky end and initiate growth of an incorrect 1-block. However, because the next open site on the ribbon also presents a sticky end encoding "0", the next tile of the 1-block (the ZDV tile) can only bind by a single sticky end. In fact, once a single error of the type described for the ZB tile has occurred, no tile in the tile set can bind the next site by more than a single bond. This property is the proofreading property: no single error can be locked in place by the correct and favorable addition of another tile. Thus, the copying error rate using this Copy tile set is expected to be much smaller than the error rate in a more naively designed tile set that copies bit strings encoded using a single tile for each bit.

The Binary Counter tile set also transmits information during growth, but it does not fully employ proofreading principles. During much of the growth process—for example, during the COPY layer growth—the redundancy of the bit encoding translates into effective proofreading, such that a single unfavorable attachment cannot be extended by further favorable attachments. However, during the COUNT layer growth, there are several points where a single unfavorable attachment may be undetectable by further favorable attachments. Referring to Fig. 2, consider a growth site where the leftmost tile in a Carry Bit 0-Block could attach favorably by both left sticky ends. Suppose instead, the leftmost tile in a Carry Bit 1-Block arrives and attaches unfavorably by only its upper left sticky end, the lower left sticky end being a mismatch. Can this tile be locked in by a subsequent favorable attachment? To do so, the next tile must match a "red 6" sticky end (reading the second tile encoding this bit of the counter) and a "red 7" sticky end (which is unique to the Carry Bit 1-Block). But there is no tile that matches both these sticky ends; the bottom tiles of the 1-Block, Carry Bit 0-Block, and Carry Bit 1-Block each match only one of the sticky ends and therefore can attach only by a second unfavorable attachment. Thus, the proofreading principle comes into play, and this error is unlikely to occur. The intuition is that this error involved a mismatch during propagation of a redundantly encoded bit, and hence could be corrected. In contrast, suppose the initial unfavorable attachment is the leftmost tile of the 1-Block, matching on its lower left sticky end and mismatching on its upper left sticky end. In this case, the bottom tile of the 1-Block will now be able to attach favorably, locking in the erroneous tile. No proofreading takes place. The intuition is that this error involved misreading the carry bit, which is not encoded redundantly in this tile set. Thus, one would not expect reduced error rates during the COUNT layer assembly steps. (It is possible to design tile sets for binary counting that employ proofreading for all transmitted information, and therefore should grow with reduced error rates (2), but all known examples require more tile types, which is why we did not implement them here.)

The Variable Width and Copy tile sets have another property which helps them grow properly during initial growth from the seed: they are self-healing (3). Self-healing is a property that allows crystals to grow back correctly after removal of a chunk of tiles, including chunks in the interior. Specifically, a self-assembled structure is self-healing to a specified class of damages if (*i*) no matter which single tile is removed, there is a unique tile that can attach favorably in that location, i.e., that matches at least two sticky ends, and (*ii*) for any removed chunk of the specified class, there is at least one site where a tile can be reattached favorably. The conclusion is that if a chunk is removed, any regrowth that does occur will be correct, and the entire removed region will eventually regrow. Although self-healing is a trivial property for periodic crystals, many algorithmic tile sets do not have this property, and regrowth is error-prone and/or gets stuck before all tiles have been replaced. But it is straightforward to show that the Variable-Width and Copy tile sets are self-healing for any damage that leaves intact a path of tiles from one side of the ribbon to the other, including both double tiles. This fact is relevant to proper growth from an origami seed. Normally algorithmic self-assembly has a canonical growth direction for correct growth, but often self-healing tile sets allow crystals to grow in directions or orders of addition that are different from the original initial growth order. For zig-zag ribbons made from Variable-Width or Copy tile sets, the canonical growth direction and growth order is that described here for zig-zag growth. Self-healing is helpful during growth from origami seeds because the initial growth from the seed is not constrained to the canonical zig-zag growth order. One can think of the edge of the origami seed as the edge of a broken ribbon, one that presents a set of sticky ends that are not in the geometry of usual zig-zag growth. Because the Variable-Width and Copy tile sets are self-healing, the set of sticky ends on the origami edge still allows a sequence of favorable tile attachments that will allow the restoration of a normal zig-zag geometry and zig-zag growth. This is what allows origami seeds to nucleate Variable-Width and Copy ribbons correctly. Furthermore, it conceptually separates crystal growth into a nucleation phase during which a triangular region of tiles grows, and a growth phase during which the zig-zag order is respected.

The Binary Counter tile set is not self-healing, but nonetheless growth from the seed is logically deterministic and proceeds without error under ideal conditions. Being closely related to the Copy tile set, one could say that it is almost self-healing; there are only two pairs of tiles that lead to violations of property (i). Referring still to Fig. 2, the bottom tile of the 0-Block and the bottom tile of the Carry Bit 1-Block have the same two lower sticky ends, whereas the bottom tile of the 1-Block and the bottom tile of the Carry Bit 1-Block have the same two right sticky ends. Therefore, if one tile of a pair is removed from a fully grown ribbon, the other tile can refill that site by attaching favorably, making two sticky-end bonds. (The two resulting mismatches are assumed to result in no energetic cost.) Thus, the Binary Counter tile set can be seen to violate the self-healing property. Nonetheless, these being the only violations, we can see that they do not lead to nondeterminism tile choice during exclusively favorable growth from the seed. Specifically, during the nucleation phase there are multiple sites where a tile can favorably attach—resulting in nondeterministic order for the tile additions—but we claim that tiles attach always by either their left two sticky ends or by their bottom two sticky ends. This excludes the possibility that the second pair can cause trouble. Furthermore, we claim that it is only in the ZAG/COPY layer that tiles can attach by their lower two sticky ends. This excludes the possibility that the first pair can cause trouble, because the first pair only occur in the ZIG/COUNT layer. To see that our claims are correct, note that in the absence of double tiles, the nucleation phase triangle fills out by tiles attaching exclusively by their left sticky ends. In this case, ZIG/COUNT layers are constructed in the same direction as they would be built during zig-zag growth (from upper left to lower right), but ZAG/COPY layers are constructed in the opposite direction as in zig-zag growth. Nonetheless, because no two tiles have the same pair of left sticky ends, tile choice is deterministic at each site. When a lower (magenta) double tile attaches by only its left sticky ends, it creates a site where a ZAG/COPY layer tile can attach by its lower sticky ends, allowing a layer of ZAG/COPY tiles to grow along a leftward and upward diagonal (the normal direction for zig-zag growth). This occurs only for the ZAG/COPY layer. Since there are no pairs of tiles involving the ZAG/COPY layer that violate the self-healing property, no errors can occur during exclusively favorable growth on this layer. As for the ZIG/COUNT layer, all tiles that attach at these sites must attach by their left sticky ends, for which tile choice is deterministic. In conclusion, exclusively favorable growth during the nucleation phase using the Binary Counter tile set will produce a pattern with no mismatch errors. Care is still required to design a seed that correctly specifies an initial counter value other than 00001, because during the nucleation phase the counting process can still take place. There are two simple solutions for designing a seed whose first layer encodes the number n. (*i*) Lay out the pattern for binary counting and cut it along a vertical line directly in front of the desired initial number; the sticky ends along that vertical line, which may contain carry bits, can be placed on the seed. In this case, the first full layer of zig-zag growth (the layer that contains the bit string 00001 in the crystals we demonstrated) will contain a bit string that is different from the bit string that appears directly on the seed. (*ii*) Prepare a seed containing the desired integer, and do not include any carry bits in the seed's tile adapter pattern. The bit string will be copied verbatim within the nucleation phase growth, and counting will commence during the first full layer of zig-zag growth. This second method is demonstrated in Fig. S11.

The tile sets also contain logical features designed to prevent the spontaneous nucleation of unseeded assemblies. The basic design principle, as in proofreading, is that although unfavorable attachments can lead to errors, they rarely stay attached long enough for additional tiles to arrive and lock the error in place—so we exploit a logical design that ensures that erroneous growth cannot cause trouble unless multiple unfavorable attachments occur in succession. In the case of spontaneous nucleation errors, the concern is that in the absence of a seed, enough tiles could attach to each other that they form a supercritical structure that can continue to grow by purely favorable attachments. Any full-width fragment of a tile ribbon (containing both a top double tile and a bottom double tile) would be such a supercritical structure, because it could continue zig-zag growth indefinitely. Conversely, any fragment that is both less than full width and cannot grow to full width by a favorable attachment would be a subcritical structure because it cannot continue to grow indefinitely by zig-zag growth. Roughly, the smallest number of unfavorable attachments needed to construct a supercritical assembly from individual tiles determines the extent of the barrier to spontaneous nucleation; the spontaneous nucleation rate is expected to decrease exponentially in the number of unfavorable attachments. This design principle has been mathematically analyzed in detail (4) and demonstrated experimentally with DNA tiles (5). The Nucleation Barrier tiles, taken from that work, were designed to ensure that all ribbon types have at least a minimum width. In principle, even greater reduction of spontaneous unseeded nucleation could be achieved using a larger nucleation barrier block.

Unfavorable attachments can lead to an additional type of error, called facet nucleation errors, during the growth of seeded structures. Whereas in the above discussion of proofreading, we generally assumed that incorrect tiles unfavorably attached at a site where a different tile should have favorably attached, unfavorable attachments can in fact occur anywhere. In zig-zag ribbons, where there is a single site on the growth facet at which favorable attachment can occur, most unfavorable attachments are presumed to occur elsewhere on the facet. Interestingly, in the Variable-Width and Copy tile sets (but not the Binary Counter tile set), such unfavorable attachments will always attach the correct tile type, because the single sticky end by which they are attaching must be the one that conveys the bit string information to be copied, and there is no other information to be conveyed. Therefore, facet nucleation errors are not expected to contribute to the ultimate copying error rate. However, in the Binary Counter tile set, a facet nucleation error will "guess" the value of the carry bit, getting it wrong half the time. Thus, the measured growth error rate in the Binary Counter experiments puts a limit on the rate at which facet nucleation errors occur. In an improvement to the simple proofreading transformations discussed above, facet nucleation errors can also be effectively eliminated by logical design of the tile set (6), which has been experimentally demonstrated (7), but such logic was not incorporated into the tile sets used in this work.

1. Rothemund PWK (2006) Folding DNA to create nanoscale shapes and patterns. *Nature* 440:297–302.
2. Winfree E, Bekbolatov R (2004) Proofreading tile sets: Error-correction for algorithmic self-assembly. *DNA Computing 9*, eds Chen J, Reif J (Springer, Berlin), Vol LNCS 2943, pp 126–144.
3. Winfree E (2006) Self-healing tile sets. In *Nanotechnology: Science and Computation*, eds Chen J, Jonoska N, Rozenberg G (Springer, Berlin), pp 55–78.
4. Schulman R, Winfree E (2005) Programmable control of Nucleation for algorithmic self-assembly. *DNA Computing 10*, eds Ferretti C, Mauri G, Zandron C (Springer, Berlin), Vol LNCS 3384, pp 319–328.
5. Schulman R, Winfree E (2007) Synthesis of crystals with a programmable kinetic barrier to nucleation. *Proc Natl Acad Sci USA* 104:15236–15241.
6. Chen HL, Goel A (2005) Error free self-assembly using error prone tiles. *DNA Computing 10*, eds Ferretti C, Mauri G, Zandron C (Springer, Berlin), Vol LNCS 3384, pp 62–75.
7. Chen HL, Schulman R, Goel A, Winfree E (2007) Reducing facet nucleation during algorithmic self-assembly. *Nano Lett* 7:2913–2919.

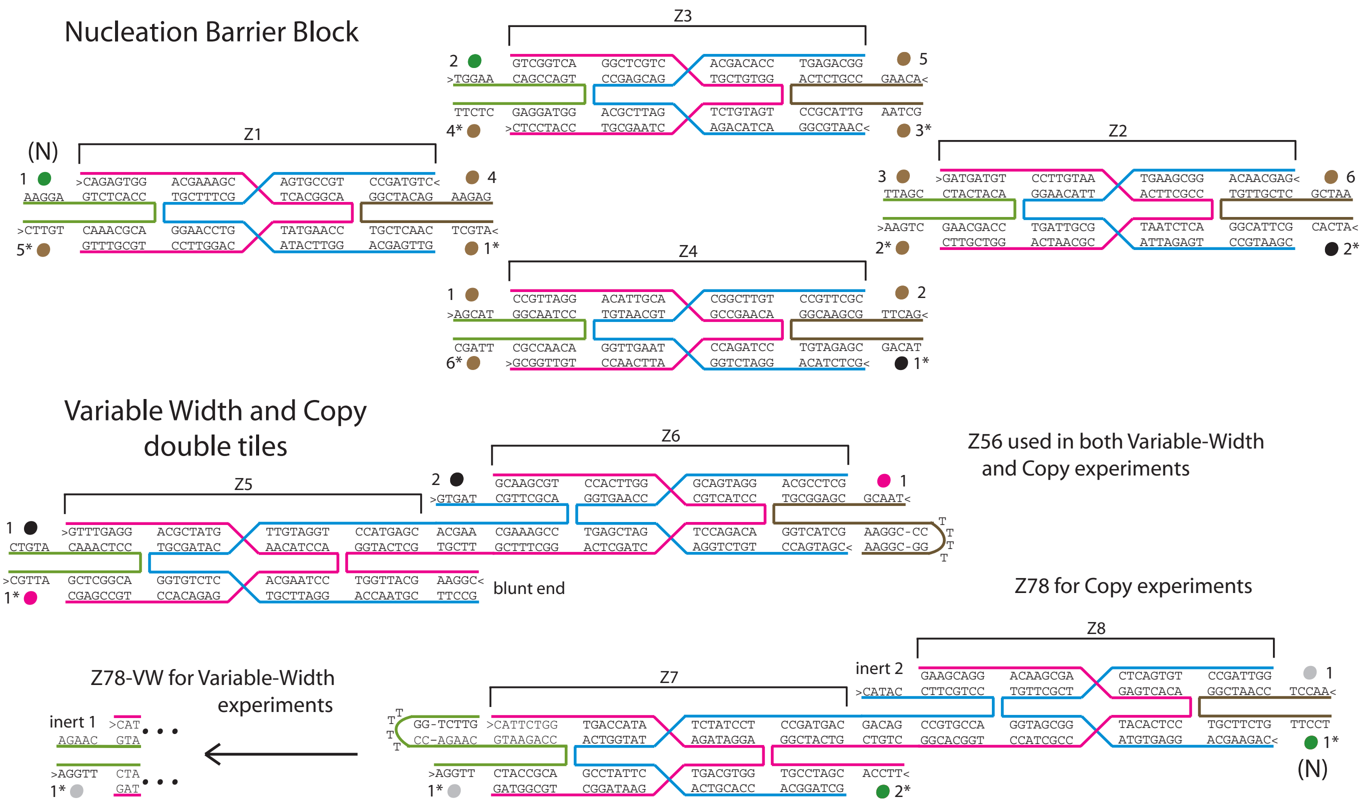

Nucleation Barrier Block
Z3
2 GTCGGTCA GGCTCGTC ACGACACC TGAGACGG 5
>TGGAA CAGCCAGT CCGAGCAG TGCTGTGG ACTCTGCC GAACA<
TTCTC GAGGATGG ACGCTTAG TCTGTAGT CCGCATTG AATCG
4* >CTCCTACC TGCGAATC AGACATCA GGCGTAAC< 3*
(N)
Z1
1 >CAGAGTGG ACGAAAGC AGTGCCGT CCGATGTC< 4
AAGGA GTCTCACC TGCTTTCG TCACGGCA GGCTACAG AAGAG
>CTTGT CAAACGCA GGAACCTG TATGAACC TGCTCAAC TCGTA<
5* GTTTGCGT CCTTGGAC ATACTTGG ACGAGTTG 1*
Z2
3 >GATGATGT CCTTGTAA TGAAGCGG ACAACGAG< 6
TTAGC CTACTACA GGAACATT ACTTCGCC TGTTGCTC GCTAA
>AAGTC GAACGACC TGATTGCG TAATCTCA GGCATTCG CACTA<
2* CTTGCTGG ACTAACGC ATTAGAGT CCGTAAGC 2*
Z4
1 CCGTTAGG ACATTGCA CGGCTTGT CCGTTCGC 2
>AGCAT GGCAATCC TGTAACGT GCCGAACA GGCAAGCG TTCAG<
CGATT CGCCAACA GGTTGAAT CCAGATCC TGTAGAGC GACAT
6* >GCGGTTGT CCAACTTA GGTCTAGG ACATCTCG< 1*
Variable Width and Copy double tiles
Z6
2 GCAAGCGT CCACTTGG GCAGTAGG ACGCCTCG 1
>GTGAT CGTTCGCA GGTGAACC CGTCATCC TGCGGAGC GCAAT<
Z56 used in both Variable-Width and Copy experiments
Z5
1 >GTTTGAGG ACGCTATG TTGTAGGT CCATGAGC ACGAA CGAAAGCC TGAGCTAG TCCAGACA GGTCATCG AAGGC-CC
CTGTA CAAACTCC TGCGATAC AACATCCA GGTACTCG TGCTT GCTTTCGG ACTCGATC AGGTCTGT CCAGTAGC< AAGGC-GG
T T T T
>CGTTA GCTCGGCA GGTGTCTC ACGAATCC TGGTTACG AAGGC<
1* CGAGCCGT CCACAGAG TGCTTAGG ACCAATGC TTCCG
blunt end
Z78 for Copy experiments
Z8
inert 2 GAAGCAGG ACAAGCGA CTCAGTGT CCGATTGG 1
>CATAC CTTCGTCC TGTTCGCT GAGTCACA GGCTAACC TCCAA<
Z78-VW for Variable-Width experiments
Z7
T T T T
GG-TCTTG >CATTCTGG TGACCATA TCTATCCT CCGATGAC GACAG CCGTGCCA GGTAGCGG TACACTCC TGCTTCTG TTCCT
CC-AGAAC GTAAGACC ACTGGTAT AGATAGGA GGCTACTG CTGTC GGCACGGT CCATCGCC ATGTGAGG ACGAAGAC< 1*
>AGGTT CTACCGCA GCCTATTC TGACGTGG TGCCTAGC ACCTT<
1* GATGGCGT CGGATAAG ACTGCACC ACGGATCG 2*
(N)
inert 1 >CAT • • •
AGAAC GTA
>AGGTT CTA
1* GAT • • •

Fig. S1. Tile diagrams for Nucleation Barrier tiles and Variable-Width and Copy double tiles. Arrows (> or <) indicate 5′ ends of strands. Double tiles (structurally two covalently fused tiles) contain hairpins replacing some sticky ends to prevent stacking interactions between the edges of ribbon assemblies. Except for certain named sticky ends (e.g., ''inert 1''), sticky ends are identified by number and color, printed above or below each sticky-end sequence. An asterisk denotes that a sticky end is complementary to the sticky end of the same number and color without the asterisk. The number of a sticky end indicates its relative geometric position within a block of tiles or a double tile. The color of a sticky end indicates either the block to which the sticky end belongs or the intended logical role of that sticky end in the design. For example, sticky ends marked by brown dots represent nucleation barrier-specific sticky ends. Sticky ends marked by black dots are shared only between the Z56 double tile and one edge of the nucleation barrier block: Their role is to enforce a minimum 4-wide ribbon width, guaranteeing that at least the nucleation block occurs. Green dots on the edges of the nucleation barrier block and the Z78 tile indicate sticky ends that are also common to both 0- and 1-blocks (Fig. S2). Thus, 0- and 1-blocks of tiles can be juxtaposed next to each other in any combination: This allows arbitrary patterns to be copied between the nucleation barrier and the Z78 double tile. A variant of the Z78 double tile (Z78-VW) with an extra sticky end (''inert 1'') in place of a hairpin was used for Variable-Width experiments to help nucleation on the seed. Except in this special case, sticky ends labeled ''inert 1'' and ''inert 2'' are intended to remain unbound and have sequences designed to minimize complementarity with other sticky ends. (N) designates a sticky end used to represent the value 0 or ''no carry'' for the carry bit when these tiles are used in Binary Counter experiments.

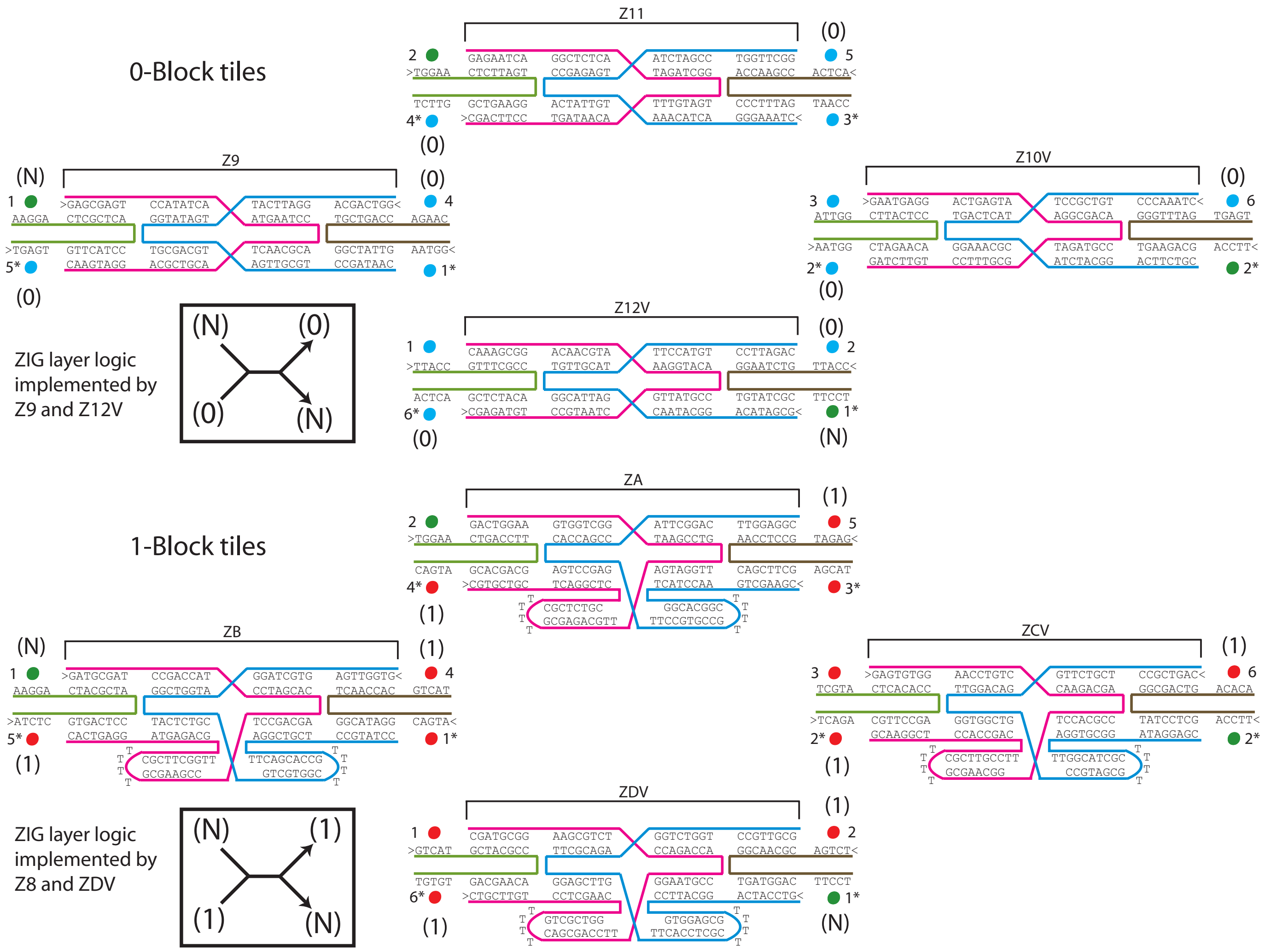

0-Block tiles
Z11
(0)
2
GAGAATCA GGCTCTCA ATCTAGCC TGGTTCGG
5
>TGGAA CTCTTAGT CCGAGAGT TAGATCGG ACCAAGCC ACTCA<
TCTTG GCTGAAGG ACTATTGT TTTGTAGT CCCTTTAG TAACC
4* >CGACTTCC TGATAACA AAACATCA GGGAAATC< 3*
(0)
Z9
(N)
(0)
1 >GAGCGAGT CCATATCA TACTTAGG ACGACTGG< 4
AAGGA CTCGCTCA GGTATAGT ATGAATCC TGCTGACC AGAAC
>TGAGT GTTCATCC TGCGACGT TCAACGCA GGCTATTG AATGG<
5* CAAGTAGG ACGCTGCA AGTTGCGT CCGATAAC 1*
(0)
Z10V
(0)
3 >GAATGAGG ACTGAGTA TCCGCTGT CCCAAATC< 6
ATTGG CTTACTCC TGACTCAT AGGCGACA GGGTTTAG TGAGT
>AATGG CTAGAACA GGAAACGC TAGATGCC TGAAGACG ACCTT<
2* GATCTTGT CCTTTGCG ATCTACGG ACTTCTGC 2*
(0)
ZIG layer logic
implemented by
Z9 and Z12V
(N) (0)
(0) (N)
Z12V
(0)
1 CAAAGCGG ACAACGTA TTCCATGT CCTTAGAC 2
>TTACC GTTTCGCC TGTTGCAT AAGGTACA GGAATCTG TTACC<
ACTCA GCTCTACA GGCATTAG GTTATGCC TGTATCGC TTCCT
6* >CGAGATGT CCGTAATC CAATACGG ACATAGCG< 1*
(0)
(N)
1-Block tiles
ZA
(1)
2 GACTGGAA GTGGTCGG ATTCGGAC TTGGAGGC 5
>TGGAA CTGACCTT CACCAGCC TAAGCCTG AACCTCCG TAGAG<
CAGTA GCACGACG AGTCCGAG AGTAGGTT CAGCTTCG AGCAT
4* >CGTGCTGC TCAGGCTC TCATCCAA GTCGAAGC< 3*
CGCTCTGC GGCACGGC
GCGAGACGTT TTCCGTGCCG
T
(1)
ZB
(N)
(1)
1 >GATGCGAT CCGACCAT GGATCGTG AGTTGGTG< 4
AAGGA CTACGCTA GGCTGGTA CCTAGCAC TCAACCAC GTCAT
>ATCTC GTGACTCC TACTCTGC TCCGACGA GGCATAGG CAGTA<
5* CACTGAGG ATGAGACG AGGCTGCT CCGTATCC 1*
CGCTTCGGTT TTCAGCACCG
GCGAAGCC GTCGTGGC
(1)
ZCV
(1)
3 >GAGTGTGG AACCTGTC GTTCTGCT CCGCTGAC< 6
TCGTA CTCACACC TTGGACAG CAAGACGA GGCGACTG ACACA
>TCAGA CGTTCCGA GGTGGCTG TCCACGCC TATCCTCG ACCTT<
2* GCAAGGCT CCACCGAC AGGTGCGG ATAGGAGC 2*
CGCTTGCCTT TTGGCATCGC
GCGAACGG CCGTAGCG
(1)
ZIG layer logic
implemented by
Z8 and ZDV
(N) (1)
(1) (N)
ZDV
(1)
1 CGATGCGG AAGCGTCT GGTCTGGT CCGTTGCG 2
>GTCAT GCTACGCC TTCGCAGA CCAGACCA GGCAACGC AGTCT<
TGTGT GACGAACA GGAGCTTG GGAATGCC TGATGGAC TTCCT
6* >CTGCTTGT CCTCGAAC CCTTACGG ACTACCTG< 1*
GTCGCTGG GTGGAGCG
CAGCGACCTT TTCACCTCGC
(1)
(N)

Fig. S2. Tile diagrams for the 0- and 1-blocks of the Copy tile set. The 0-block is also the repeatable block for the Variable-Width tile set. Arrows (> or <) indicate the 5′ ends of each strand. Tiles in the 1-block contain protruding internal hairpins to generate height contrast and allow for their identification under AFM. Blue and red dots indicate sticky ends specific to the 0- and 1-blocks, respectively. Green dots indicate sticky ends common to both 0- and 1-blocks that also occur on one edge of the nucleation barrier block and the Z78 double tile. Thus, either a 0- or 1-block can be adjacent to the nucleation barrier block and Z78 tile; further 0- and 1-blocks can be juxtaposed next to each other in any order. Thus, a Copy ribbon starts on one edge with a Z78 double tile, continues through an arbitrary sequence of 0- and 1-blocks (possibly none), and then finishes with a nucleation block (and a Z56 double tile). *Insets* describe how information is processed by ZIG/COUNT layer of the 0- and 1-blocks when they are used in the Binary counter. The left half of the *Inset* indicates information that is input via the top-left and bottom-left sticky ends of the leftmost tile in a block (e.g., Z9 or ZB). The bottom left of the *Inset* indicates the value of the input digit, 0 or 1, from the previous ZAG/COPY layer of the Binary Counter ribbon. The top left indicates the value of the carry bit that is being input ($N = 0$ or ''no carry'' and $C = 1$ or ''carry''). The top right of the *Inset* indicates the computed value of the digit in the current layer (in the current block), and the arrowhead indicates that the digit is output to the next COPY layer. The output digit is the opposite of the input digit if the carry bit is set (C) and is otherwise unchanged. (This is the XOR of the two input bits.) The bottom right indicates the computed value of the carry bit that is propagated along the current layer. The output carry bit is set (C) only if both the input digit is 1 and the input carry bit is set (C). (This is the AND of the two input bits.)

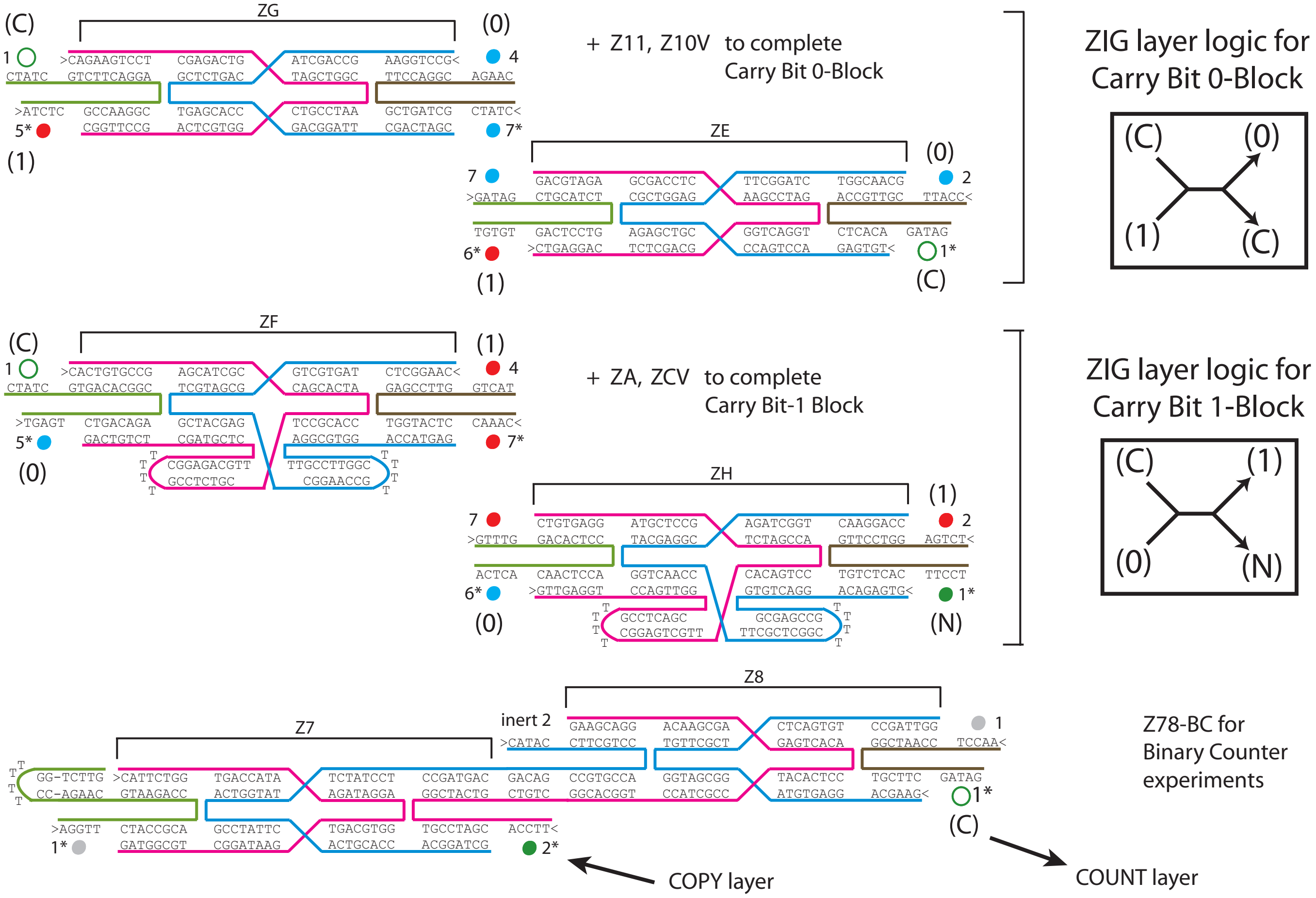

ZG
(C)
(0)
1
>CAGAAGTCCT CGAGACTG ATCGACCG AAGGTCCG<
4
CTATC GTCTTCAGGA GCTCTGAC TAGCTGGC TTCCAGGC AGAAC
>ATCTC GCCAAGGC TGAGCACC CTGCCTAA GCTGATCG CTATC<
5* CGGTTCCG ACTCGTGG GACGGATT CGACTAGC 7*
(1)
+ Z11, Z10V to complete Carry Bit 0-Block
ZE
(0)
7
GACGTAGA GCGACCTC TTCGGATC TGGCAACG
2
>GATAG CTGCATCT CGCTGGAG AAGCCTAG ACCGTTGC TTACC<
TGTGT GACTCCTG AGAGCTGC GGTCAGGT CTCACA GATAG
6* >CTGAGGAC TCTCGACG CCAGTCCA GAGTGT< 1*
(1)
(C)
ZIG layer logic for Carry Bit 0-Block
(C) (0)
(1) (C)
ZF
(C)
(1)
1 >CACTGTGCCG AGCATCGC GTCGTGAT CTCGGAAC< 4
CTATC GTGACACGGC TCGTAGCG CAGCACTA GAGCCTTG GTCAT
>TGAGT CTGACAGA GCTACGAG TCCGCACC TGGTACTC CAAAC<
5* GACTGTCT CGATGCTC AGGCGTGG ACCATGAG 7*
(0)
T T T CGGAGACGTT GCCTCTGC
TTGCCTTGGC CGGAACCG T T T
+ ZA, ZCV to complete Carry Bit-1 Block
ZH
(1)
7 CTGTGAGG ATGCTCCG AGATCGGT CAAGGACC 2
>GTTTG GACACTCC TACGAGGC TCTAGCCA GTTCCTGG AGTCT<
ACTCA CAACTCCA GGTCAACC CACAGTCC TGTCTCAC TTCCT
6* >GTTGAGGT CCAGTTGG GTGTCAGG ACAGAGTG< 1*
(0)
T T T GCCTCAGC CGGAGTCGTT
GCGAGCCG TTCGCTCGGC T T T
(N)
ZIG layer logic for Carry Bit 1-Block
(C) (1)
(0) (N)
Z8
inert 2
GAAGCAGG ACAAGCGA CTCAGTGT CCGATTGG 1
>CATAC CTTCGTCC TGTTCGCT GAGTCACA GGCTAACC TCCAA<
Z7
T T T T GG-TCTTG >CATTCTGG TGACCATA TCTATCCT CCGATGAC GACAG CCGTGCCA GGTAGCGG TACACTCC TGCTTC GATAG
CC-AGAAC GTAAGACC ACTGGTAT AGATAGGA GGCTACTG CTGTC GGCACGGT CCATCGCC ATGTGAGG ACGAAG< 1*
>AGGTT CTACCGCA GCCTATTC TGACGTGG TGCCTAGC ACCTT<
1* GATGGCGT CGGATAAG ACTGCACC ACGGATCG 2*
(C)
COPY layer
Z78-BC for Binary Counter experiments
COUNT layer

Fig. S3. Tile diagrams for Binary Counter experiments. Arrows (> or <) indicate 5′ ends of strands. To implement the Binary Counter, 4 new single tiles ZE, ZF, ZG, and ZH are added to the Copy tile set, and a new Z78-BC double tile is used. The ZG/ZE and ZF/ZH tile pairs are used in the context of two new types of blocks, a ''Carry Bit 0-Block'' and a ''Carry Bit 1-Block'' as diagrammed in the right-hand side of Fig. 2. Each new block uses two tiles from the original Copy tile set. A Carry Bit 0-Block includes ZG, ZE, Z11, and Z10V; a Carry Bit 1-Block includes ZF, ZH, ZA, and ZCV. *Insets* describe how information is processed by the ZIG/COUNT layer of the Carry Bit 0- and 1-Blocks. The left half of the *Inset* indicates information that is input via the top-left and bottom-left sticky ends of the leftmost tile in a block (e.g., ZG or ZF). The bottom left of the *Inset* indicates the value of the input digit, 0 or 1, from the previous ZAG/COPY layer of the Binary Counter ribbon. The top left indicates the value of the carry bit that is being input ($N = 0$ or ''no carry'' and $C = 1$ or ''carry). The top right of the *Inset* indicates the computed value of the digit in the current layer (in the current block), the arrowhead indicates that the digit is output to the next ZAG/COPY layer. The output digit is the opposite of the input digit if the carry bit is set (C) and is otherwise unchanged. (This is the XOR of the two input bits.) The bottom right indicates the computed value of the carry bit that is propagated along the current layer. The output carry bit is set (C) only if both the input digit is 1 and the input carry bit is set (C). (This is the AND of the two input bits.) The Z78-BC double tile inputs a carry bit of value ''1'' (C) to each new ZIG/COUNT layer to initiate the ripple carry addition. In an effort to lower the binary counter error rate, a ''steric matching'' scheme was implemented for the ''C'' sticky ends. The arms adjacent to input and output ''C'' sticky ends were elongated and truncated, respectively, by 2 nucleotides so that an incorrectly added tile would suffer an additional energetic penalty for having the wrong shape. However, no obvious differences were discernible between experiments done with and without this particular error-reduction scheme. The fact that steric matching did not decrease the error rate suggests that ''facet nucleation errors'' (which occur when tiles bind facets via single bonds) may dominate the error rate (rather than growth errors that would incur a steric mismatch) because the current tile sets do not protect against facet nucleation errors (6, 7).

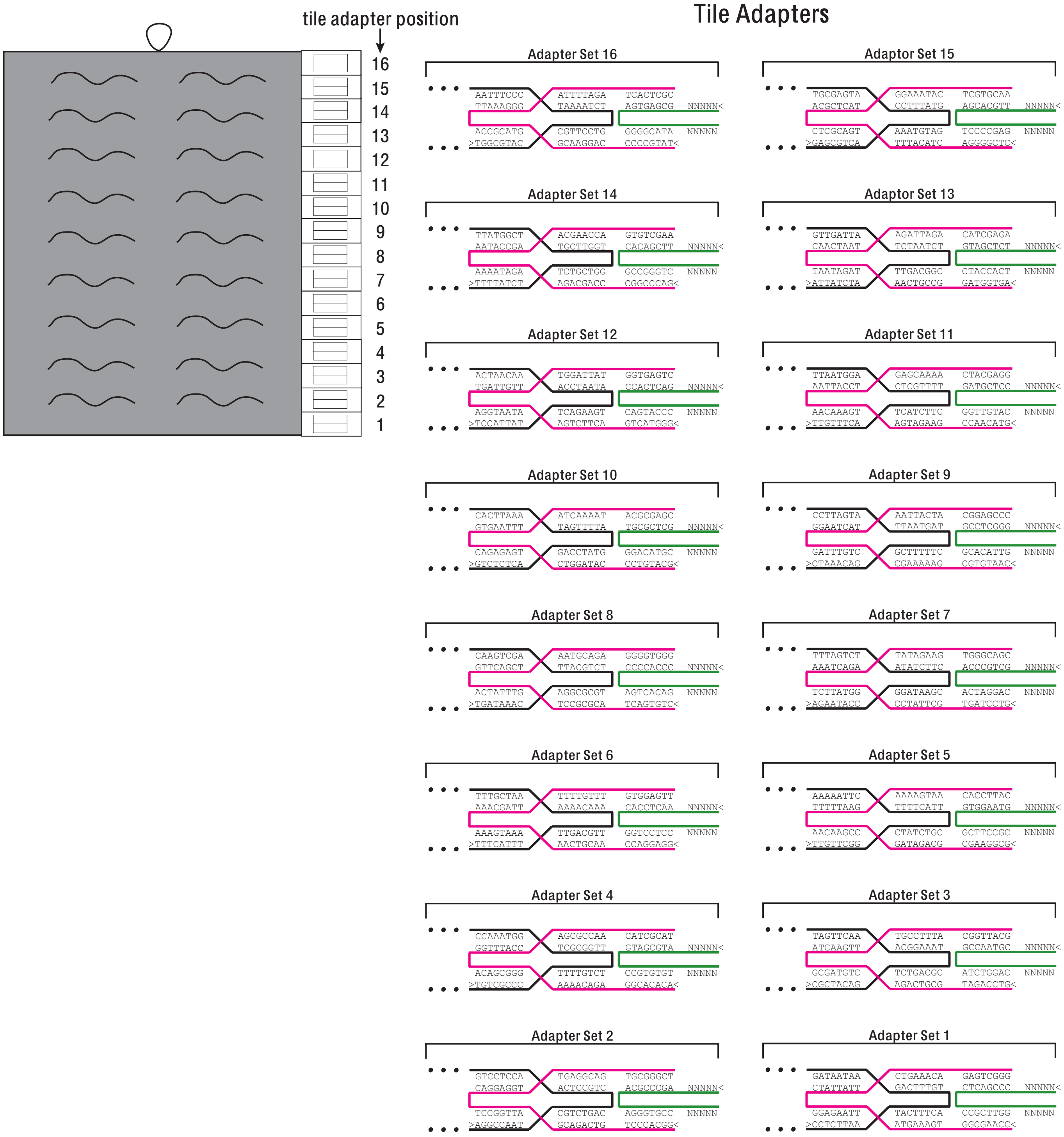

tile adapter position
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
Tile Adapters
Adapter Set 16
AATTTCCC ATTTTAGA TCACTCGC
TTAAAGGG TAAAATCT AGTGAGCG NNNNN<
ACCGCATG CGTTCCTG GGGGCATA NNNNN
>TGGCGTAC GCAAGGAC CCCCGTAT<
Adaptor Set 15
TGCGAGTA GGAAATAC TCGTGCAA
ACGCTCAT CCTTTATG AGCACGTT NNNNN<
CTCGCAGT AAATGTAG TCCCCGAG NNNNN
>GAGCGTCA TTTACATC AGGGGCTC<
Adapter Set 14
TTATGGCT ACGAACCA GTGTCGAA
AATACCGA TGCTTGGT CACAGCTT NNNNN<
AAAATAGA TCTGCTGG GCCGGGTC NNNNN
>TTTTATCT AGACGACC CGGCCCAG<
Adaptor Set 13
GTTGATTA AGATTAGA CATCGAGA
CAACTAAT TCTAATCT GTAGCTCT NNNNN<
TAATAGAT TTGACGGC CTACCACT NNNNN
>ATTATCTA AACTGCCG GATGGTGA<
Adapter Set 12
ACTAACAA TGGATTAT GGTGAGTC
TGATTGTT ACCTAATA CCACTCAG NNNNN<
AGGTAATA TCAGAAGT CAGTACCC NNNNN
>TCCATTAT AGTCTTCA GTCATGGG<
Adapter Set 11
TTAATGGA GAGCAAAA CTACGAGG
AATTACCT CTCGTTTT GATGCTCC NNNNN<
AACAAAGT TCATCTTC GGTTGTAC NNNNN
>TTGTTTCA AGTAGAAG CCAACATG<
Adapter Set 10
CACTTAAA ATCAAAAT ACGCGAGC
GTGAATTT TAGTTTTA TGCGCTCG NNNNN<
CAGAGAGT GACCTATG GGACATGC NNNNN
>GTCTCTCA CTGGATAC CCTGTACG<
Adapter Set 9
CCTTAGTA AATTACTA CGGAGCCC
GGAATCAT TTAATGAT GCCTCGGG NNNNN<
GATTTGTC GCTTTTTC GCACATTG NNNNN
>CTAAACAG CGAAAAAG CGTGTAAC<
Adapter Set 8
CAAGTCGA AATGCAGA GGGGTGGG
GTTCAGCT TTACGTCT CCCCACCC NNNNN<
ACTATTTG AGGCGCGT AGTCACAG NNNNN
>TGATAAAC TCCGCGCA TCAGTGTC<
Adapter Set 7
TTTAGTCT TATAGAAG TGGGCAGC
AAATCAGA ATATCTTC ACCCGTCG NNNNN<
TCTTATGG GGATAAGC ACTAGGAC NNNNN
>AGAATACC CCTATTCG TGATCCTG<
Adapter Set 6
TTTGCTAA TTTTGTTT GTGGAGTT
AAACGATT AAAACAAA CACCTCAA NNNNN<
AAAGTAAA TTGACGTT GGTCCTCC NNNNN
>TTTCATTT AACTGCAA CCAGGAGG<
Adapter Set 5
AAAAATTC AAAAGTAA CACCTTAC
TTTTTAAG TTTTCATT GTGGAATG NNNNN<
AACAAGCC CTATCTGC GCTTCCGC NNNNN
>TTGTTCGG GATAGACG CGAAGGCG<
Adapter Set 4
CCAAATGG AGCGCCAA CATCGCAT
GGTTTACC TCGCGGTT GTAGCGTA NNNNN<
ACAGCGGG TTTTGTCT CCGTGTGT NNNNN
>TGTCGCCC AAAACAGA GGCACACA<
Adapter Set 3
TAGTTCAA TGCCTTTA CGGTTACG
ATCAAGTT ACGGAAAT GCCAATGC NNNNN<
GCGATGTC TCTGACGC ATCTGGAC NNNNN
>CGCTACAG AGACTGCG TAGACCTG<
Adapter Set 2
GTCCTCCA TGAGGCAG TGCGGGCT
CAGGAGGT ACTCCGTC ACGCCCGA NNNNN<
TCCGGTTA CGTCTGAC AGGGTGCC NNNNN
>AGGCCAAT GCAGACTG TCCCACGG<
Adapter Set 1
GATAATAA CTGAAACA GAGTCGGG
CTATTATT GACTTTGT CTCAGCCC NNNNN<
GGAGAATT TACTTTCA CCGCTTGG NNNNN
>CCTCTTAA ATGAAAGT GGCGAACC<

Fig. S4. Diagrams of all 16 tile adapters. Arrows (<) indicate the 5′ ends of each strand. Tile adapters are abstracted as clear boxes with two internal rectangular stripes that can be color-coded to indicate the use of particular sticky ends. Here, no colors are used because all sticky-end sequences are undefined (indicated by the ''NNNNN'' string in the tile sequence diagrams). As in Fig. 1, pink and green are used to represent tile adapter strands, whereas the black is used to represent the scaffold strand for the origami seed. Pink strands bind the scaffold; green strands bind pink strands and present sticky ends.

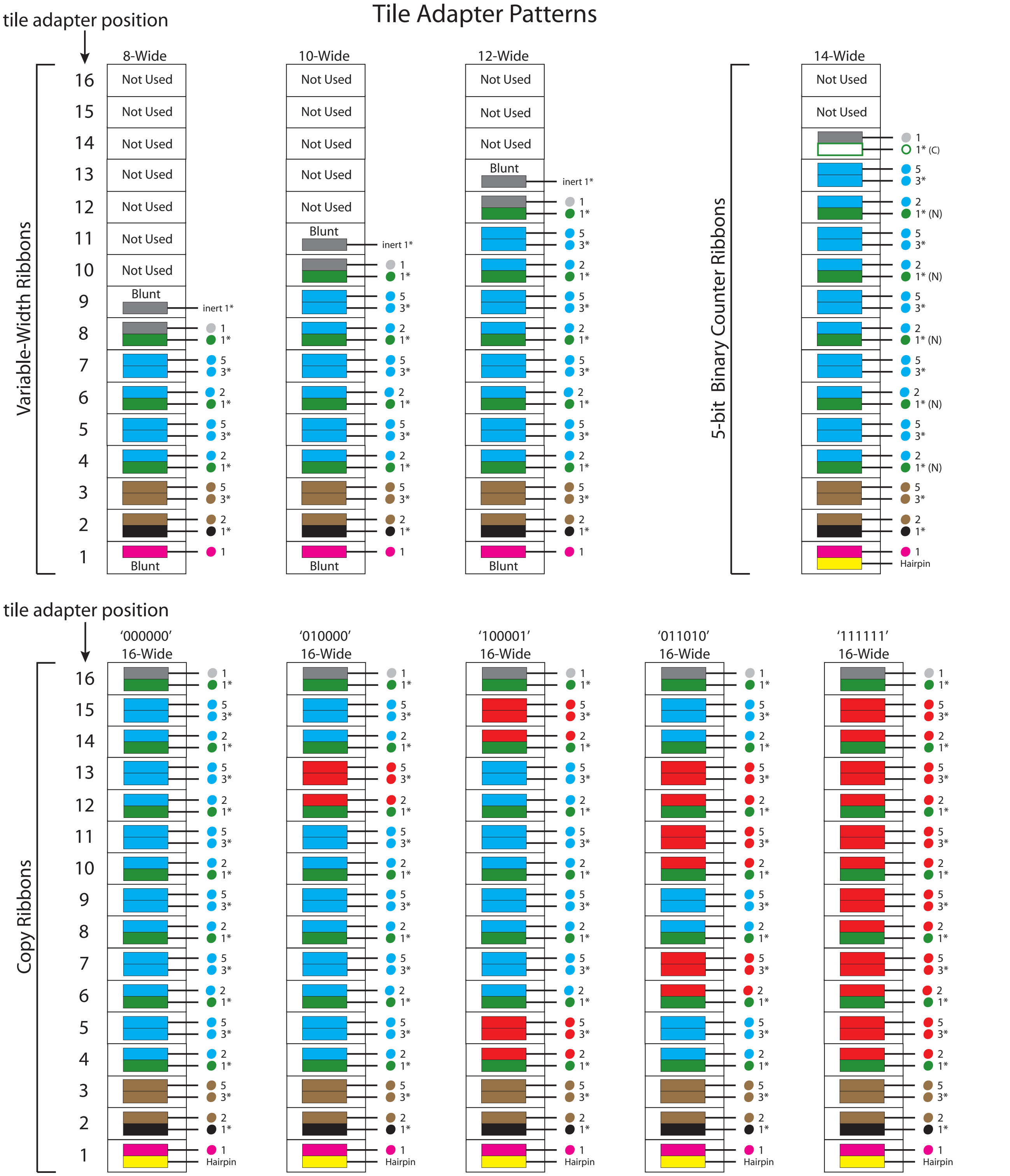


Fig. S5. Tile adapter sticky-end patterns for all ribbon nucleation experiments. Tile adapters are abstracted as clear boxes with two internal rectangular stripes that can be color-coded to indicate the use of particular sticky ends. Sticky-end arrangements for all Variable-Width, Copy, and Binary Counter experiments are explicitly shown on columns of the 16 tile adapters. If a box is labeled ''Not Used,'' the strands for the tile adapter at that position were not added to the final strand mix, leaving a single-stranded loop of the scaffold strand at that site. Copy ribbon seeds ''100001'' and ''011010'' were also used for 6-bit Binary Counter experiments shown in Fig. S11.

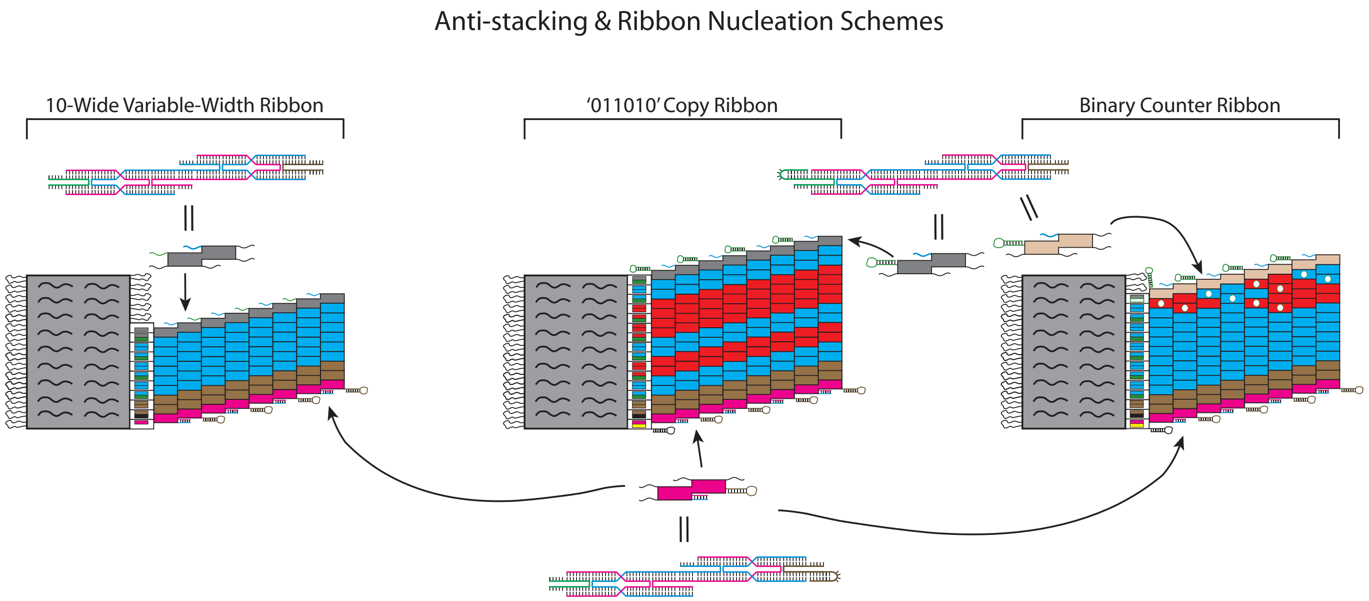
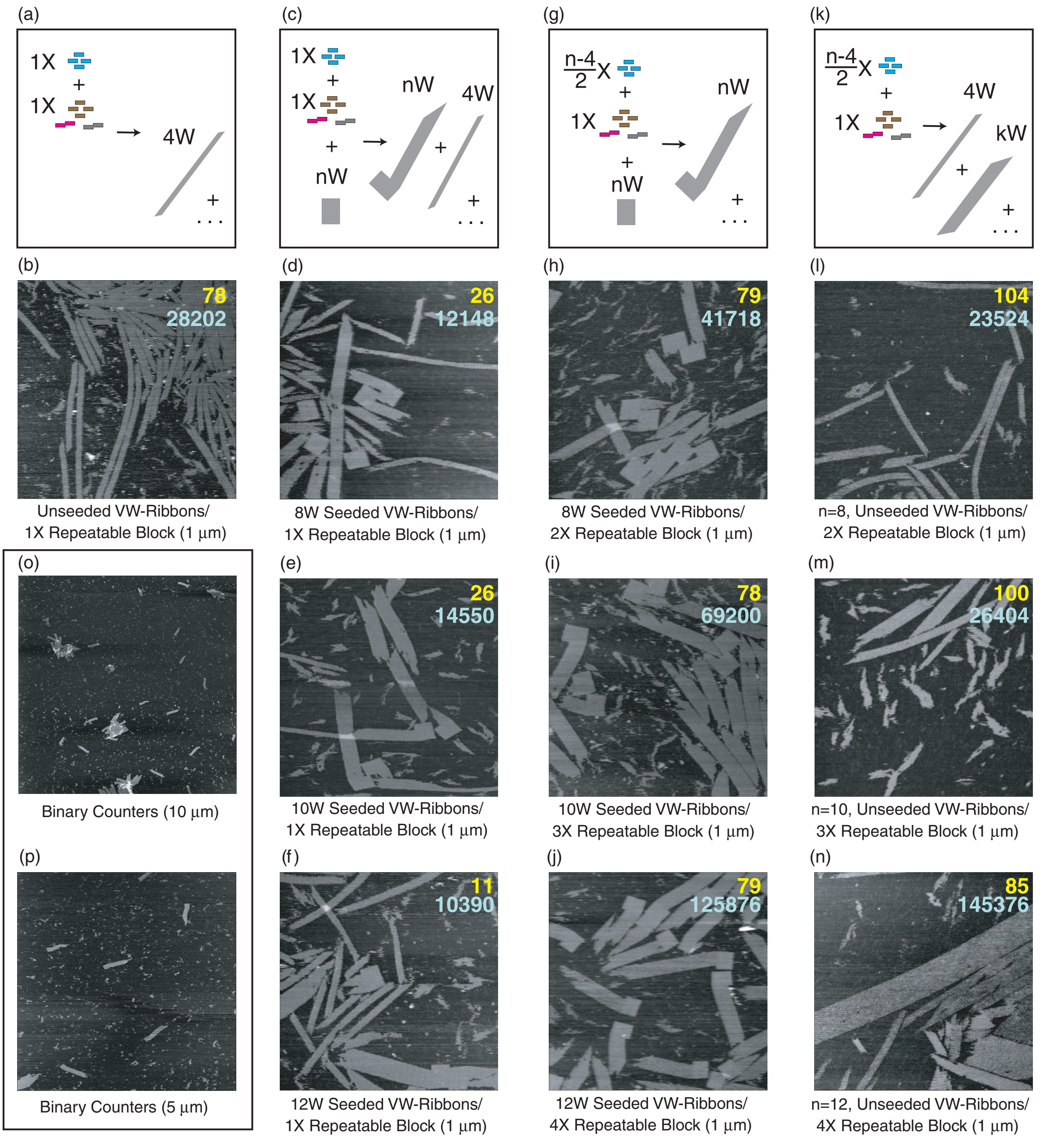

Anti-stacking & Ribbon Nucleation Schemes
10-Wide Variable-Width Ribbon
'011010' Copy Ribbon
Binary Counter Ribbon

Fig. S6. Anti-stacking and ribbon nucleation schemes. Multiple approaches were taken to inhibit π-stacking interactions between blunt ends of DNA helices that can lead to the aggregation of origami seed and ribbons. Thirty of 244 staple strands on the right and left edges of the original tall rectangle origami (described in ref. 1) were not used, so that, at sites where tile adapters are not present, loops of the scaffold strand (black) inhibit stacking interactions between seeds. Furthermore, in all experiments, the bottom Z56 double tile presented alternating blunt ends and hairpins to discourage stacking interactions between ribbons. In Variable-Width experiments, the top double tile (Z78-VW) presented ''inert'' sticky ends with no binding partners at all sites along the top edge of the ribbon. In Copy and Binary Counter experiments, a top double tile was used that presented alternating inert sticky ends and hairpins. However, using this hairpin variant of the top double tile, we observed no clear differences in the number and type of stacking interactions between ribbons. Subjective experience suggests that ''inert'' sticky ends (those who sequence is not complementary to any tile sticky end) should be avoided because they can bind tiles anyway. Blunt ends should be avoided because they can cause stacking between ribbons. Single-stranded loops and hairpins appear to be the best way to terminate helices that are intended to be ''inert.'' A further difference between the Variable-Width and Copy/Binary Counter experiments is that in the Variable-Width experiments, the top double tile (Z78-VW) attaches via two sticky ends to the tile adapters, whereas in the Copy/Binary Counter experiments, the top double tile (Z78/Z78-BC) attaches to the tile adapters by only a single sticky end. In both cases, it is possible to nucleate via a series of exclusively favorable binding steps (with two sticky ends binding simultaneously), but in the Copy/Binary Counter nucleation scheme, the top double tile can bind via a favorable step only after the initial layer of zig-zag growth for the ribbon has completed.



Unseeded VW-Ribbons/ 1X Repeatable Block (1 μm)

8W Seeded VW-Ribbons/ 1X Repeatable Block (1 μm)

8W Seeded VW-Ribbons/ 2X Repeatable Block (1 μm)

n=8, Unseeded VW-Ribbons/ 2X Repeatable Block (1 μm)

Binary Counters (10 μm)

10W Seeded VW-Ribbons/ 1X Repeatable Block (1 μm)

10W Seeded VW-Ribbons/ 3X Repeatable Block (1 μm)

n=10, Unseeded VW-Ribbons/ 3X Repeatable Block (1 μm)

Binary Counters (5 μm)

12W Seeded VW-Ribbons/ 1X Repeatable Block (1 μm)

12W Seeded VW-Ribbons/ 4X Repeatable Block (1 μm)

n=12, Unseeded VW-Ribbons/ 4X Repeatable Block (1 μm)

Fig. S7. Examples of unligated Variable-Width ribbon images used for nucleation statistics and wide-field views of ligated Binary Counter ribbons. (*a*, *c*, *g*, and *k*) In schematic form, the Variable-Width experiment that was performed and the majority products of the reaction for comparison with images below showing experimental data. nX indicates that *n* × 100 nM of each of a particular tile's strands were used. In *d*–*f* and *h*–*j*, nW indicates that a particular origami seed encoded a ribbon *n* tiles wide, for *n* = 8, 10, and 12. For *l*–*n*, concentrations of tiles were made to match those of *h*–*j*, and *n* is given explicitly because there were no seeds. (*d*–*f*) Example data from the equal stoichiometry Variable-Width experiment reported in Fig. S8. (*b*) Example data from the unseeded control experiment for the equal stoichiometry Variable-Width experiment. (*h*–*n*) Seeded (*h*–*j*) and unseeded (*l*–*n*) controls give example data for the normalized stoichiometry Variable-Width experiment reported in Fig. 3. In *l*–*n*, kW means that ribbons of a variety of widths *k* were produced, although the maximum *k* increased with *n*. No large lattices made from the 0-block tiles were observed in seeded samples *h*–*j*, but in the unseeded samples (*l*–*n*), with high concentrations (200–400 nM) of 0-block tiles, such lattices were relatively common; in *l* and *n*, we estimate that they contained ≈20% of tiles. In each sample image, the yellow number refers to the number of 1 × 1-μm AFM images that were examined to produce statistics, and the blue number refers to the total number of tiles counted to arrive at a given statistic. (*o* and *p*) Wide-field views of Binary Counter ribbons.

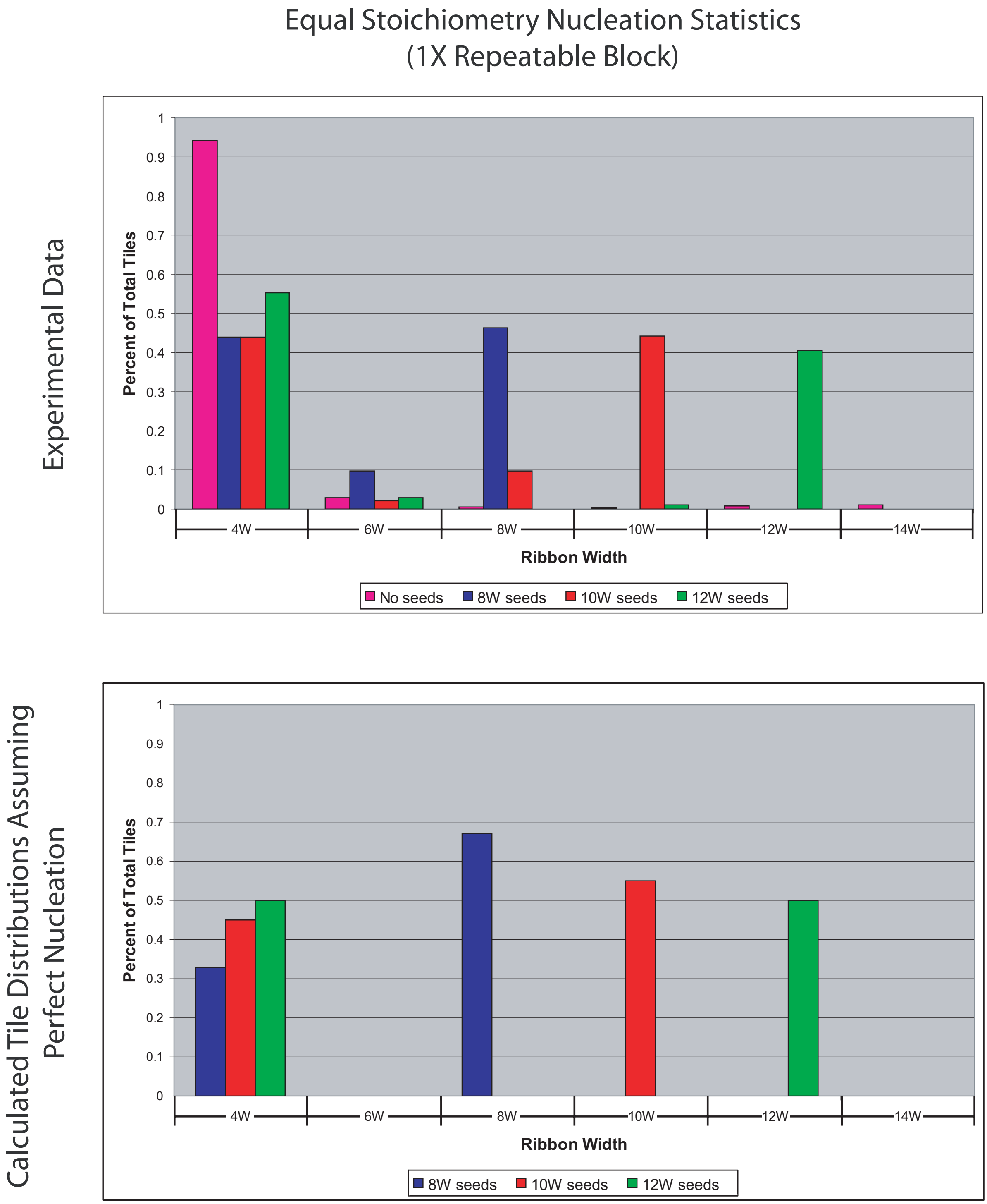


Fig. S8. Statistics for unseeded Variable-Width ribbons and Variable-Width ribbons nucleated by 8-, 10-, and 12-wide origami seeds using tiles at equal stoichiometry (100 nM). Here, each tile adapter strand was at 100 nM, each staple was at 50 nM, and the scaffold was at 10 nM; samples were unligated. ''Experimental Data'' shows the experimentally observed tile distribution in ribbons with widths ranging from 4 to14 tiles. N = 12,147; 14,550; and 10,390 tiles, respectively, for the nucleated samples and N = 28,202 for the unnucleated sample. ''Calculated Tile Distributions Assuming Perfect Nucleation'' shows the tile distribution that would theoretically occur if all ribbons were correctly nucleated and grew until free monomers for the 0-block were exhausted, subsequently allowing only 4-wide ribbons to grow using the remaining nucleation barrier tiles and double tiles. Discrepancies between experimental data and perfect nucleation estimates are partially caused by errors that decrease the width of nucleated ribbons as they nucleate or grow, such as lattice errors and premature reversal errors (see Figs. S9 and S10). The high prevalence of spontaneously nucleated 4-wide ribbons (composed of ''leftover'' nucleation barrier and double tiles) is the major difference between these equal stoichiometry experiments and the normalized stoichiometry experiments reported in Fig. 3; the results of these two experiments were otherwise very similar.

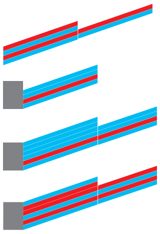
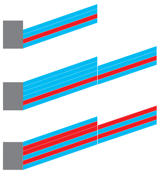
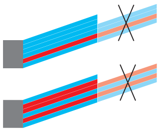
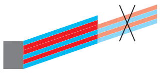
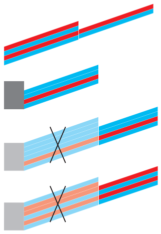
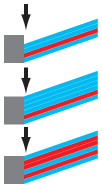
models show the types of ribbon considered in each column; emphasis shows which parts were counted. error rates are expressed: per bit / per layer	**a** all apparently nucleated ribbons 	**b** ribbons attached to origami 	**c** ribbons attached to origami, full width 	**d** ribbons attached to origami, correct nucleation 	**e** < 6-bit ribbons from **a** 	**f** first layer stats for structures in **b** 
1 → 0 copying errors 1 → 0 copying error rate (%)	51 .21 / .54	42 .21 / .53	29 .19 / .56	17 .18 / .54	14 .25 / .44	252 15.6 / 46.5
0 → 1 copying errors 0 → 1 copying error rate (%)	11 .046 / .12	8 .039 / .10	6 .038 / .12	2 .021 / .064	3 .054 / .093	7 .43 / 1.3
total copying errors total copying error rate (%)	62 .13 / .66	50 .12 / .62	35 .11 / .67	19 .10 / .60	17 .15 / .53	259 8.0 / 47.5
lattice errors lattice error rate (%)	33 .069 / .35	25 .061 / .31	14 .045 / .27	9 .048 / .29	13 .12 / .40	17 .53 / 3.1
premature reversal errors prem. rev. error rate (%)	189 .39 / 2.0	169 .42 / 2.1	145 .46 / 2.8	89 .47 / 2.8	28 .25 / .87	17 .53 / 3.1
total layers total bits copied	9384 48118	8007 40725	5220 31320	3141 18846	3220 11134	545 3231
zero positions one positions						1620 1611
total seeded ribbons total nucleated correctly correct nucleation rate (%)						545 303 55.6

Fig. S9. Statistics on unligated Copy ribbon error rates. Copy ribbons were nucleated from seeds encoding the bit pattern ''011010'' by using 300 nM concentrations of each 0-block strand, 300 nM concentrations of each 1-block strand, 100 nM concentrations of each of the rest of the tile strands, and 10 nM concentrations of the scaffold strand. A variety of error rates were collected on subpopulations of the observed structures; models show the range of structures in each subpopulation. Column **a** includes all structures that were ''apparently nucleated'': all copy ribbons that were attached to origami as well as copy ribbons that started at least 4 bits wide (i.e., 12 tiles wide; based on our experience with other experiments, this assumes that the occurrence of spontaneously nucleated 12-wide ribbons was very low). Column **b** includes all ribbons that were attached to origami, regardless of whether they nucleated with incorrect patterns or nucleated with less than full width. Column **c** includes ribbons that are both attached to origami and full width (i.e., they encoded 6 bits and were 16 tiles wide). Column **d** includes only ribbons that are simultaneously attached to origami, full width, and nucleated with the correct pattern. The populations of structures in **a**–**d** are subsets of decreasing size: **a**⊃**b**⊃**c**⊃**d**. For **a**–**d**, counting of errors began after the first layer, i.e., errors at the origami–first layer interface were ignored; for **a**–**e**, parts of the ribbons that were not counted have been deemphasized. Counting of errors for structures in **a** and **b** was performed on the entire ribbon, whereas counting of errors for structures in **c** and **d** was performed only up to the point that any ribbon narrowed and became less than full width. Column **e** gives statistics for less than full width, so it tallies the part of ribbons from **a** that had <6 bits (<16 tiles wide). Column **f** tallies errors in the first layer (arrows) of all ribbons attached to origami (the same structures as **b**). This allows a comparison of the error rates for nucleation and for copying in ribbons away from the nucleus; some rates change drastically; others do not. Errors are reported in both absolute numbers of observed errors and the rates (as a percentage) of the total number of observed bits and growth layers: layers are 1 tile thick, and each bit comprises 2 tiles. Per-bit rates and per-layer (or equivalently, per-seed for **f**) rates are separated by a ''/''. Copying errors (changes in the identity of a bit) are separated into 1 → 0 and 0 → 1 errors to highlight the great asymmetry in these error rates, which is presumably due to an energetic cost for association incurred by the hairpins on the 1-block tiles. Both rates decrease by approximately a factor of 10 (per layer or per bit) when considering growth away from the first layer. (Note that the total copying error rate is not simply an average of the two kinds of copying error rates, because the denominator changes from the number of 0 or 1 bits to the total number of bits.) Two kinds of errors cause ribbons to narrow and represent fewer bits. One, a ''lattice error,'' occurs when there is a geometric defect in the lattice as shown in Fig. S10. The second, a ''premature reversal error,'' occurs when a double tile binds where a tile from a 0- or 1-block should go, and this causes a premature reversal of growth direction that terminates the row. Premature reversal errors are the most common type of error (including copy errors) by a factor of at least 3 when considering growth away from the nucleus (in the first layer, lattice and premature reversal errors happen at nominally equal rates). For example, when comparing **b** with **f**, 40% of lattice defects occurred in the first layer of growth, whereas only 9% of premature reversal errors occurred in the first layer. Lattice errors decrease drastically in rate away from the first layer (by a factor of 10 per layer), whereas the rate of premature reversal errors decreases by only 1/3.

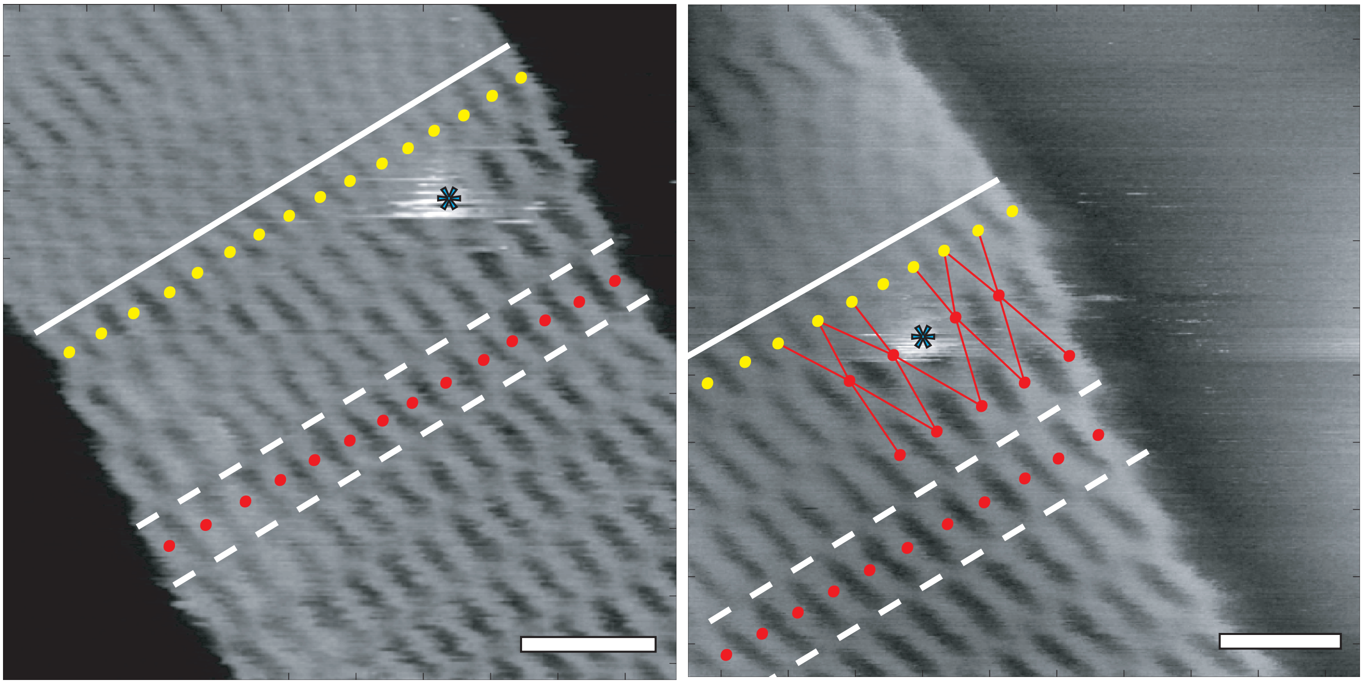


Fig. S10. Example of a lattice defect during nucleation in a ''000000'' Copy ribbon. (Scale bars, 20 nm.) Yellow dots indicate the center positions of the 16 tile adapters on an origami seed specifying a 16-wide ribbon, whereas red dots indicate the center positions of tiles in the 14-wide ribbon that is nucleated. A solid white line marks the edge of the origami seed at the base of the tile adapters: Dashed white lines bound the third layer of tiles in the ribbon. The lattice defect is indicated by an asterisk. Connections between red dots (*Right*) show how sticky-end pairing interactions coordinate around the lattice defect to cause the ribbon to shrink by 2 tiles (the size of a 0- or 1-block) from its programmed width of 16 tiles. Tile arms are evidently flexible enough to accommodate this kind of error. Similar lattice defects during nucleation have been observed for all types of ribbon used in this work. The mechanism responsible for the lattice defect is unknown, but we hypothesize that it is due to the mismatch in the interhelix spacing of the origami and the lattice constant for the DNA tile ribbon. Even though both structures encompass 32 DNA helices, they have different patterns and frequencies of crossovers, which change the forces that balance the electrostatic repulsion between helices. This appears to affect their relative size in the direction perpendicular to the helix axes, as marked by the solid and white dashed lines for the origami and ribbon, respectively. The interhelix spacing in the 32-helix origami used here (with 1.5 turns between adjacent crossovers) is ≈1 nm, which gives a length (along the white solid line) for the origami of ≈95 nm. The lattice spacing for tile lattices has been measured at ≈6–7 nm per tile, which would give a width of 96–112 nm for a 16-wide ribbon along the white dashed line). There is a noticeable ''flare'' at the junction between many origami and the ribbons that they nucleate, indicating that a relaxed 16-tile-wide ribbon is indeed wider that an origami is tall. For example, see Fig. 4, which shows noticeable curvature at the origami-ribbon junction for the ''000000'' ribbon. When measured from Fig. 4, the ratio of sizes is ≈1.12, which means that if the origami is 95 nm, then the ribbon is ≈106 nm (absolute AFM measurements are probably no better than 5%). A fix for this problem might be to use an origami with 1.5-nm interhelix spacing (as appears to be true for origami with 2.5-turn spacing between adjacent crossovers): This approach should yield an ≈110.5-nm-tall, 32-helix origami seed. We note that width-shrinking is also observed to occur later on in growth, due both to lattice defects like that shown here and a much more prevalent type of error that we term a ''premature reversal error.'' Statistics on lattice defects and premature reversal errors, both during nucleation and for later growth, are given in Fig. S9.

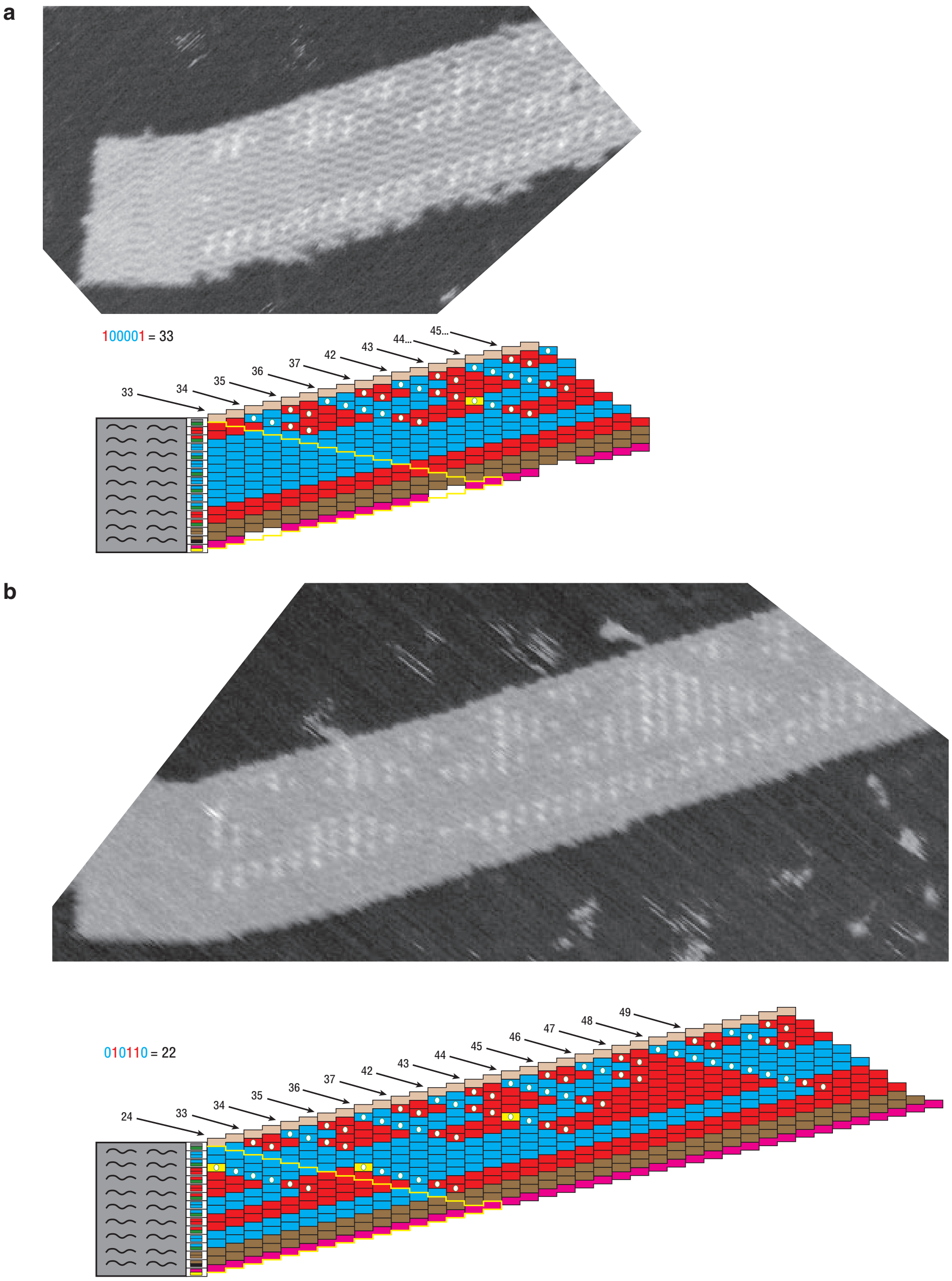


Fig. S11. Binary Counter crystals nucleated with alternative initial values. (*a*) Using a seed with the ''100001''-bit string to create a binary counting pattern that starts at 33, using tiles without the shape modifications for the carry bit. The seed is identical to the one used for the Copy tile set with the same bit string. The yellow outline delineates nucleation phase growth, i.e., the initial series of tiles that can be favorably attached by their left sticky ends, prior to zig-zag growth. A single error occurred during zig-zag growth, shown in yellow. In this experiment, it appears that we used a faulty preparation of the lower (magenta) double-tile molecules; it seems unlikely that proper zig-zag growth was occurring. Therefore, we presume that the ZAG/COPY layer growth was not initiated by the lower double tile but, rather, occurred by unfavorable attachment of tiles and facet nucleation. Thankfully, facet nucleation on the ZAG/COPY layer does not cause bit copying errors. Because the overall error rate is low, we presume that in this crystal the ZIG/COUNT layer initiated properly from the upper double tile; facet nucleation of the ZIG/COUNT layer would be likely to result in a counting error. (*b*) Using a seed with the ''010110'' bit string to create a binary counting pattern that starts at 22, using tiles with shape modifications to encode the carry bit. The seed is identical to the one used for the Copy tile set with the same bit string. (Note that here, the string is read from bottom to top, whereas there, it was read from top to bottom and written ''011010.'') There was 1 growth error during nucleation and 2 growth errors during zig-zag growth. The growth error rates for these two crystals are compatible with what we have observed in the ''00001''-nucleated Binary Counter crystals, and the nucleation errors are compatible with the measurements for Copy crystals.

Other Supporting Information Files

SI Appendix (PDF)