Serine Integrase-Based Event Recording in *E. coli*

Thesis by
Andrey Shur

In Partial Fulfillment of the Requirements for the
Degree of
Doctor of Philosophy

CALIFORNIA INSTITUTE OF TECHNOLOGY
Pasadena, California

2022
Defended 12/13/2021

Andrey Shur
ORCID: 0000-0001-9372-6713

ACKNOWLEDGEMENTS

I would like to thank Richard and all members of the Murray lab, and some members of the Elowitz lab for emotional and intellectual support during my time here. Also I would like to thank the Resnick Institute, Caltech, NIH, and ICB for funding me.

ABSTRACT

DNA is a unique molecule that has evolved to serve as the genetic material for life. It seems straightforward to consider this molecule not only as a wonder of the natural world but as a tool for information storage and retrieval. Bacteria have evolved to conserve DNA, but bacteriophages have evolved to specifically integrate their genomes using integrases. In response to viruses, bacteria have evolved the RNA-guided nuclease Cas9 to destroy viral DNA before it can be integrated. The fruits of these evolutionary pressures prove useful to the researcher interested in easily editing DNA. In this work, we have engineered a genetic circuit that can enact specific and controlled genetic changes in response to changing small molecule concentrations. Known DNA sequences can be repeatedly integrated into a synthetic array such that their identity and order encodes information about past small molecule concentrations that the cell has experienced. To accomplish this, we use catalytically inactive CRISPR-Cas9 (dCas9) to bind to and block attachment sites for the integrase Bxb1. Through the co-expression of dCas9 and guide RNA, Bxb1 can be directed to integrate one of two engineered "ink" plasmids, which correspond to two orthogonal small molecule inducers whose presence or absence as a function of time can be recorded with this system. Integrase sites present on these plasmids are found to not participate in intramolecular "deletion" reactions if closer than 100 bp. Guide RNAs overlapping integrase attachment sites are found to effectively block integrase activity at those sites if the overlap is equal to 9 or 19 base pairs. Other overlap values, including forward or reverse binding result in ineffective integrase activity repression. We develop 8 orthogonal guide RNA sequences capable of binding to and repressing integrase activity at the attP site. Plasmid multimers are sequenced using Oxford Nanopore sequencing and found to follow population-level predictions of event record identity. Single DNA states are found insufficient for identifying past history of events; an ensemble of DNA states at the population level must be used. A modular modeling framework is developed (Global enumeration) to describe this system, and integrated with the existing chemical reaction network creation automation software BioCRNpyler. The modeling framework developed here automatically creates chemical reaction networks based on typical linear DNA-based synthetic biology "genetic constructs" and predicts transcripts and proteins produced based on simple transcription/translation rules. Integrase-based recombination events can also be predicted in a recursive way.

PUBLISHED CONTENT AND CONTRIBUTIONS

[1] L. N. Merk, A. S. Shur, A. Pandey, R. M. Murray, and L. N. Green. Engineering logical inflammation sensing circuit for gut modulation. *bioRxiv*, 2020. doi: 10.1101/2020.11.10.377085. URL https://www.biorxiv.org/content/early/2020/11/10/2020.11.10.377085. A.S. served as a mentor and assisted in lab work in creating and testing AND gate genetic constructs, culturing Nissle 1917, developing assays to test AND gate function, and general advising. A.S. also contributed in the role of planning experiments and advising the overall direction of the project.

[2] W. Poole, A. Pandey, A. Shur, Z. A. Tuza, and R. M. Murray. Biocrnpyler: Compiling chemical reaction networks from biomolecular parts in diverse contexts. *bioRxiv*, 2020. doi: 10.1101/2020.08.02.233478. URL https://www.biorxiv.org/content/early/2020/08/03/2020.08.02.233478. A.S. contributed code for performing local enumeration, global enumeration, TxTl explorer, Global component enumerator, and plotting functionality. A.S. made figure 5 and associated description.

[3] A. Shur and R. M. Murray. Repressing integrase attachment site operation with crispr-cas9 in *E. coli*. *bioRxiv*, 2017. doi: 10.1101/110254. URL https://www.biorxiv.org/content/early/2017/02/21/110254. A.S. performed experimental work and wrote manuscript.

[4] A. Shur and R. M. Murray. Proof of concept continuous event logging in living cells. *bioRxiv*, 2020. doi: 10.1101/225151. URL https://www.biorxiv.org/content/early/2020/05/28/225151. A.S. performed experimental work and wrote manuscript.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 The Computer Analogy

DNA has been known as a naturally occurring biological molecule since 1869, but its ability to contain information was discovered relatively recently. We now know that DNA is the core genetic material of almost all life on earth. The amazing realization that DNA could contain hereditary information in 1952, to the elucidation of the genetic code in 1961, to the first human genome published in 2001, represents a herculean effort to understand this new treasure trove of information. Alongside these discoveries, the technologies of DNA synthesis, genetic transformation, and sequencing have allowed the modern age of molecular and synthetic biology. The analogy between living cells and computers is another emergence from these discoveries, as it is now known that life is a process of reading and interpreting commands that are somehow encoded in the sequence of these nucleic acids.

If we make this analogy, then we must deal with the consequence that it should be our goal as synthetic biologists to understand the programming language of these computers, and subsequently make them behave as we see fit. After all, the thing that distinguishes a synthetic biologist from other types of biologist is that we are trying to figure out not just what biology does, but what biology *can do*. So then we find ourselves asking the subsequent question: what do we want these biological computers to do? Eradication of diseases, clean food and material synthesis, and cheap recycling are solid goals. But how can those functions be encoded in DNA? Much like the invention of steel did not immediately lead to skyscrapers and suspension bridges, there must be intermediate steps. We can imagine that a lofty goal such as curing all diseases is equivalent in complexity to creating the internet. But first, we must create at least one functional computer, and be able to easily interface with it. These, as any early computer scientist can attest, are not easy tasks. Our added difficulty is the fact that DNA is very small. We do not really have the right tools to interface with it, except those that have naturally evolved to do so. Therefore, in order to use the programming language of life, we have to know how it works. As many novices in programming soon learn, it is easier to learn a programming language by trying to write a program, than it is to read

all the documentation. Especially, as with living systems (and many open source coding projects), such documentation does not exist.

A key discovery in the field of molecular biology has been the fact that chemically synthesized DNA polymers can be introduced into cells and function essentially the same as the naturally occurring DNA. This means a molecular biologist must first separately synthesize their "code," and then insert it into a living organism. Only then can the function of this code be determined. This process seems slow and cumbersome, considering that cells contain all the machinery needed for making and modifying DNA already, yet it was the crucial innovation that enabled synthetic biology as a field. Indeed, the machinery of the cell can make and modify DNA, but it has evolved specifically to replicate DNA as exactly as it possibly can, not to make arbitrary changes that a researcher desires. As much as cells are like computers and DNA is like machine code, there is no analogy to "users".

1.2 Viral Integrases

In the cellular world there are no authorized users, but there are "hackers" which cells must defend themselves against. Viruses exist solely to co-opt the machinery of life to their own wishes, much like synthetic biologists. So it is no wonder that we take many of the most useful tools from viral genetic material. Viral promoters [42], lysis proteins [41], terminators [14], RNA binding proteins [56], reverse transcriptases [16], and integrases [19] have all become synthetic biology tools which we cannot live without.

The first recombinase discovered was *cre* [71], from the bacteriophage P1. The *cre* recombinase is a tyrosine recombinase and proceeds through a Holliday junction intermediate, where each double strand of DNA only ever has a single strand broken at a single time (see Figure 1.1A). Several important features make this enzyme and its binding site, known as loxP, useful. First, loxP is fairly small; less than 40 bp in length [17]. Second, the site is directional, owing to the 8 bp variable region in the middle. This variable region must match with a compatible sequence to make recombination possible. Thus, the experimenter is relatively free to design this sequence such that there are orthogonal loxP sites. Upon the enzyme's discovery many key genetic manipulations were made possible, particularly the selective deletion, insertion, or cassette exchange [11] of genetic elements.

One particular limitation with *cre*, and the reason why inversion is not a common reaction done with *cre*, is that the enzyme reaction is not particularly directional.

If two loxP sites are compatible, then recombination between them can proceed regardless of whether they already have been recombined or not. This is not a problem when you are trying to delete a gene, since the deleted fragment becomes a detached circle and does not get replicated. In the case of insertion the lack of directionality can present a problem, but because of this, cassette exchange was developed, which starts and leaves the genome with two incompatible loxP sites, and therefore does not risk deleting out the gene which was inserted. To allow inversion reactions, the directionality problem has been partially addressed by the development of the lox66 and lox71 sites [84]. The idea of these directional sites is that they are each singly mutated, but when recombined, the resulting site is doubly mutated. Thus, the doubly mutated site is significantly less active.

Since the discovery of *cre*, many other proteins have been discovered that can mediate genetic modification of various types [68]. Particularly interesting are the serine integrases, as compared with tyrosine integrases of which *cre* is a member. Serine integrases do not make a Holliday junction (see Figure 1.1B) and instead introduce a staggered double strand break into DNA, creating a two base 3' overhang. They then rotate the broken DNA pieces until overhangs hybridize, and are re-ligated. Serine integrases are capable of cleavage and ligation, whereas enzymes like *cre* require a separate ligase.

Another critical difference is that after recombination, the sites cannot be recombined again. This "forward only" behavior allows sharply directional recombination events that were not easy to do using the original tyrosine recombinases. However, since *cre* was first, there are now established mouse and fly and other animal lines that have been engineered to contain an inducible *cre* gene, allowing very easy construction of test constructs, and explains why ostensibly inferior tyrosine recombinases are still heavily used.

It is valuable to step back and consider the idea of the recombinase again from the point of view of the computer analogy. With the advent of recombinases, we now have a fairly effective "find and replace" function for DNA. However, the string you are allowed to "find" is still quite limited. And, the sequence that will be "replaced" must be provided in physical form, either by transformation or because it is already in the genome in a different place. The problem being solved is still that of a user interface. Through the use of recombinases we have gained the ability to flip a switch external to the organism of interest, adding a chemical inducer or increasing the temperature for example, and enact a change at the DNA level. For now, the

change is very simple, just the activation of a dormant gene or the deletion of an active one, but this simple interface allows a lot of interesting questions to be asked.

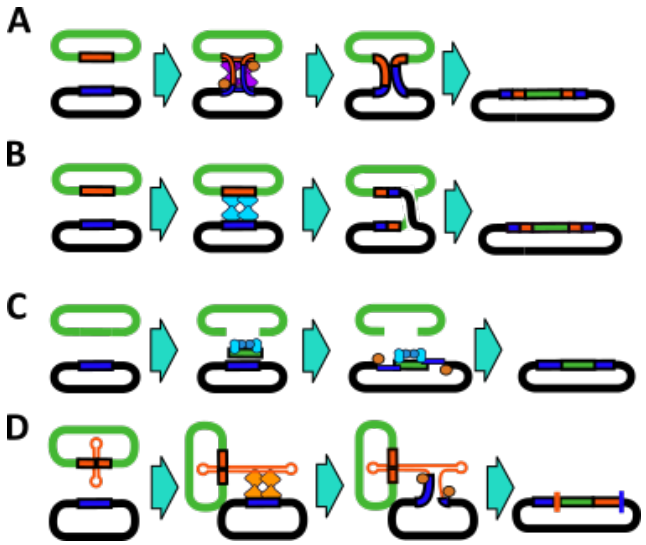


Figure 1.1: Different methods of natural DNA modification. (A) Tyrosine integrase mechanism such as *cre*. Two loxP sites (orange and blue lines) are bound by a tetramer of integrase molecules. The integrase molecules induce a Holliday junction in the DNA, which must get resolved by host ligases (brown circles) to produce a recombined sequence [17]. (B) Serine integrase mechanism such as Bxb1. AttP and attB (orange and blue lines) are bound by a tetramer of integrase molecules. The integrase molecules make double strand breaks in the DNA, and through a rotation of two of the subunits within the tetramer, reposition the half sites into the recombined arrangement, where they are subsequently ligated by the integrase enzyme [24]. (C) CRISPR spacer acquisition mechanism. A tetravalent complex of cas1 and cas2 binds a short double stranded DNA fragment. The protein bound fragment is incorporated into the beginning of the CRISPR array by sequential ligation to the top and bottom strand, such that a single direct repeat becomes two single stranded direct repeats, one on either side of the newly added spacer. Finally, host factors repair the single stranded region into double stranded DNA [54]. (D) Integron mechanism. AttC (orange) adopts a cruciform configuration that binds to the integron protein and is recombined with the attI (blue) sequence through a Holliday junction intermediate. Host replication is required to resolve the resulting structure into a normal double stranded DNA molecule [20].

Part of the reason why recombinases are so effective in providing this interface, as opposed to the ordinary homology-directed repair, is because of how fast and efficiently they act. An integrase protein is entirely nonfunctional when absent, and can react to completion within minutes when present [81]. One can imagine an experiment where a gene deletion is intended, and therefore that gene is flanked

by integrase sites. Upon integrase induction, it is essential that the gene is deleted in every somatic cell, and not only in some of them, as the phenotype of interest might not manifest if the deletion rate was too low. For this reason we can see the difficulty presented to intrepid genome miners or protein engineers. Perhaps there is an enzyme out there which performs a very interesting DNA-altering function. But, if that enzyme is slow or ineffective, it may not end up being useful for the most interesting experiments.

1.3 CRISPR

Recently, a new eye has been trained to proteins that can enact DNA changes, since the elucidation of the CRISPR [18] mechanism of action. In contrast to the many enzymes devoted to maintaining DNA integrity, there are many whose sole purpose is to disrupt DNA in specific, usually pathogenic ways. The enzyme Cas9, for example, is essential for some bacteria to be resistant to bacteriophages. The way it does this is to target very specific bacteriophage sequences for destruction. This allows the enzyme to avoid the host genome, which should be left intact. Because phages evolve quickly, however, the enzyme has evolved to be easily re-programmable, by means of a "guide RNA." These RNAs are encoded in the bacterial genome, and serve as a most wanted list of phages seen in the recent past. If any phages whose sequences are in this array ever make a return appearance, their DNA is quickly destroyed. If a phage waits long enough, however, its own sequences can become so old in this array that the bacterium "forgets" that phage has ever existed.

This mechanism has several very interesting features. A programmable DNA binding enzyme seems perfectly suited for allowing researchers to effectively interface with DNA. The ability to program specific binding has allowed a new wave of specific gene deletion, activation, or modification experiments with many unique advantages over previous methods. There is no doubt that Cas9 is a useful enzyme, from the synthetic biology point of view, but it still has a few limitations. First, the enzyme is programmed by binding to RNA, so you must still somehow produce the RNA sequence of interest in vivo. This does not decrease the need for DNA synthesis or transformation or any of the other "standard" molecular biology techniques, although the scientist does not need to be very worried (some worry is still needed) about the secondary structure of the guide RNA sequence [59]. Second, the enzyme can only cut the DNA, and other types of changes are more difficult, although much work is currently being done on this [5, 21, 26, 27, 35, 36, 52]. In general, since Cas9 is RNA programmable, you always need to somehow change

the RNA sequence in order to control the enzyme. One can imagine the enzyme is a user interface of sorts, which converts a DNA sequence, through RNA, into protein-DNA binding.

Another use of the Cas9 enzyme has recently emerged as a fundamentally unique way of editing DNA. A Cas9 enzyme with only one functional DNA strand cutting domain is fused to a reverse transcriptase protein. This fusion protein is still an RNA-guided DNA nicking enzyme. When bound, this protein induces a single strand break. An extended guide RNA can then hybridize with this nicked single stranded DNA, and create a perfect substrate for reverse transcriptase extension. This technique is now known as prime editing [5]. Utilizing this method, single bases adjacent to guide RNA binding sites can be edited and even entire serine integrase attachment sites can be inserted [32]. This is an incredibly interesting technology from the point of view of developing an effective user interface. However, as this technology is so recent, there are still many optimizations to be made before the prime editor can be as effective at creating inducible DNA changes as serine integrase. The primary limitation is that there is a limit to how big the inserted DNA can be. This limitation is partially ameliorated by the insertion of a serine integrase site, which subsequently allows a serine integrase-mediated insertion of a bigger sequence [32]. Another limitation is the error rate of reverse transcriptase and DNA repair means that repeated edits might result in a build-up of errors, much as has been observed in self-targeting guide RNA experiments [21]. There is no doubt, however, that prime editing has extreme potential when it comes to DNA editing.

The second interesting feature of CRISPR is the record of previous phage encounters. In bacteria, the proteins in the Cas pathway perform a series of DNA processing steps which result in a chronological record of phage-specific sequences [54]. This system is remarkable because it is a sort of Lamarckian evolution. Bacteria containing this system can enact specific heritable changes to their own genome, based on events that happened during their lives (phage encounters), which convey advantages to their offspring (phage resistance). The idea of making deliberate mutations to one's own genome is extremely interesting because it is exactly what molecular biology aims to do, and runs counter to the main goal of most DNA-repair and replication enzymes. This system has not been considered very useful so far because synthetic DNA is so widespread. Why try to clumsily force a bacterium to change its own genome when you can synthesize exactly the sequence desired by chemical means? The CRISPR system also does not allow the creation of arbitrary sequences. Short sequences

known as protospacers are extracted from bacteriophage DNA and sequentially integrated, flanked by repeats, which arise from copies of a specific region of the host's DNA (see Figure 1.1C). Even if it was possible to control the timing of protospacer insertion, it would be difficult to control their exact sequence, since the enzymes insert whatever DNA they can find. But even if the specific sequence of inserted spacers cannot be controlled, useful information can still be extracted from the array. By modulating the copy number of a plasmid, for example, the probability that a new protospacer will come from that plasmid can be modulated, a technique that has been used to create a molecular recording system [63, 64].

A genetically encoded system that can modify itself is a natural stepping stone to having a "user-interfacable" organism. Going back to the computer analogy, the current state of synthetic biology is like writing computer code without having a computer. The exponential growth in computer technology in the last few decades was only able to occur because technological advancements in computers allowed easier technological advancement in computers. To create a situation like this in synthetic biology, there must be a way for synthetic biological organisms to allow easier development of synthetic biological organisms. The CRISPR enzymes are not suited for the role of being a user interface because they have evolved to work slowly and randomly. This makes sense given their function: the enzymes act slowly to prevent a bacterium from shredding its own genome for the sake of having a highly active immune system. That is not to say that these enzymes could not be great building blocks for such a system, or that it is impossible to evolve them to act as we desire.

1.4 Integrons

Considering the idea of "users" of our biological computers, it comes to mind that bacteria are capable of exchanging genetic information in a way distinct from sexual reproduction. These exchanges of genetic information can lead to bacteria rapidly acquiring antibioitic or other resistance [29]. Thus it may be true that genes have evolved that facilitate the process of bacterial DNA acquisition, and perhaps some of these genes could be useful for synthetic biologists. As discussed previously, we may consider viruses as hackers, since they intend to take over the bacterial machinery and bend it to their will, something that the bacterial machinery actively tries to prevent. However, plasmids are in many ways similar to viruses, except in some cases less nefarious. Some plasmids are capable of transferring themselves between hosts, and even confer evolutionary advantages in the way of resistance

genes to those hosts [72, 79].

Bacteria have many methods of acquiring plasmids, from natural competence to conjugation. However, there is another mechanism which has particular interest to our discussion because it is in some ways a natural method of genetic engineering. This mechanism is known as the integron [29, 45]. The basic idea of the integron is similar to the CRISPR. Pieces of DNA are acquired and inserted sequentially in a genetic locus, with the newest piece toward one end and steadily older pieces of DNA farther away from the insertion site (see Figure 1.1D). The difference is the nature of DNA pieces that get incorporated, and the mechanism of incorporation. Integrons selectively integrate protein coding genes. Most genes are integrated in the sense direction, and a promoter at the beginning of the integron makes one long mRNA containing all the genes that have been integrated, with the most recent one being first. This part is very similar to CRISPR, which is also transcribed, but it is particularly interesting that entire protein coding genes are somehow chosen for collection into the integron, whereas the CRISPR incorporates random DNA into the protospacers.

The mechanism of the integron is fascinating and serves as a major inspiration for the work presented here. Something like the integron, a known locus where input genes and pathways are assembled, surely is something which could allow more "user-friendly" synthetic biology in the future. Of course the inevitable problem is that the integron needs input DNA as well, which is perfectly logical for a bacterium that does not have the foresight to design its own protein sequences de novo.

1.5 Event Recording

So far the discussion has focused on the idea of intentionally inducing desired DNA modifications through outside stimuli. This discussion then begs further difficult questions such as, "What modifications should be made this way?" or, "Why should DNA modifications need to be inducible?" The idea of event recording is simpler in the sense that it does not really matter what the DNA modifications are, as long as they can be detected later. In a sense, an event recorder is the first step to creating a "DNA editing user interface." After all, if the user's inputs are to be considered events, then a functioning user interface would enact changes in response to those events, thereby effectively creating a record.

Event recording is also something that can be interesting and useful on its own. DNA already serves as a record for the divergence of species from a common ancestor,

and in this way has been a critical resource for biologists and taxonomists since the discovery that it contains heritable information. In this case, the "events" being recorded are the random changes in a DNA sequence as a result of mutations. It is fairly straight forward to calculate the amount of time that passed since two similar sequences diverged from a common ancestor, based on the number of differences between the two sequences and a known mutation rate [78]. The mutation rate is a critical parameter that determines the limitations of this type of record. If the mutation rate is too low, then there will not be enough differences between two sequences to get an accurate time estimate, and similarly there will be too many differences if the rate is too high. Several such "lineage tracing" methods have been developed that make use of the high, targeted DNA mutation rates offered by Cas9[22] and integrase [15].

The advantage of recombinases over other DNA editing methods is that the changes they make are very predictable and targeted. Thus, simply by arranging recombinase sites on DNA, the order of recombination events can lead to mutually exclusive DNA states. The first study to make use of this idea of mutually exclusive DNA changes used the *cre* recombinase to induce deletions of fluorescent proteins in order to activate a different color of fluorescence expression in what would otherwise be genetically identical nerve cells [43]. Subsequently, serine integrases became widely known and, whereas before, DNA flipping by *cre* was random, it could now be effectively controlled through the expression of integrase or excisionase [9]. The ability to control DNA flipping as well as insertion or deletion meant that mutually exclusive integrase operations could get significantly more compact and sophisticated. So the concept of using integrase operations to enact Boolean logic [10] lead to the idea of the temporal logic gate [31] as well as further integrase logic design software [28] where almost any Boolean logical truth table could be compiled by combining nested, tandem, and otherwise precisely oriented integrase attachment sites. Given that integrase logic seems sufficiently well explored, is there a way to make use of the speed and precision of serine integrases for event recording purposes in a way distinct from using nested integrase sites? This was the motivating question at the heart of the work presented here.

1.6 Simulation Automation

Simulation and engineering go hand in hand when it comes to understanding complex systems. There can be said to be multiple levels of abstraction when it comes to biological simulations. At the lowest level, a series of mathematical equations can

be used to describe the concentrations of chemical species in systems of arbitrary complexity. Slightly more abstract is the idea of a chemical reaction network (CRN), taking advantage of the fact that the same chemical reaction can be approximated mathematically in various ways, this allows the researcher to have some control over the fidelity of the simulation versus expediency of computation. Even higher abstraction levels are usually what is used when discussing the types of biochemical "circuits" that are typically employed by synthetic biologists. For example, it is reasonable to assume in most cases that genes make RNA and that RNA makes protein. So in terms of transcriptional circuits it is common to state that a gene exists, and assume the presence of the cognate RNA and protein, and therefore implicitly assume the presence of the chemical reactions necessary to produce these molecules. Here again, a researcher is given a choice as to which chemical reaction assumption to employ.

When it came to simulating the event recorder presented here, we found that manually enumerating CRNs was impractical given that there were so many possible recombination products. Thus, we chose to develop a generic integrase CRN generation framework to contribute to a growing open source CRN automation package [58]. Many tools exist for simulating chemical reaction networks using the Synthetic Biology Markup Language (SBML) [69, 74]. The critical step, then, is to go from a simple genetic circuit specification to SBML for simulation purposes. To streamline this, certain assumptions must be made about the system and the chemical reactions taking place besides the genes specified by the experimenter. For instance, mRNAs must bind to RNAse before becoming degraded, and the presence of a lot of mRNA could prevent other important cellular mRNAs from being degraded. An effect on degradation like that would not be important in cell extract, where RNA degradation is much slower. Many CRN creators exist [2, 48], but the advantage of our approach [58] is the ease with which the researcher can change the models underlying specific processes without having to rewrite their circuit specification, and achieve a wide range of different CRN complexities.

Chapter 2

INTEGRASE-BASED EVENT RECORDING IN *E. COLI*

2.1 Introduction

Event recording can be divided into two modes of operation. First, the event is detected, and then it is recorded. Living cells naturally respond to stimuli in transient (as well as lasting) ways, and much work has been done to characterize the signal transduction networks behind these responses [76]. Transcriptional upregulation is an extremely common mechanism of response to stimuli in all living cells. Temporary, rapid transcriptional activation is enacted in bacteria through repressors unbinding promoters, or activators binding upstream of promoters and recruiting polymerase. In this work we make use of known promoters with well-characterized activating/repressing proteins [46]. The challenge, then, is how to convert this temporary transcriptional activation to a permanent record. Protein-based memory circuits are among the earliest known transcriptional circuits to be described as having bistable behavior [6]. The bistable switch from lambda phage is capable of hereditary information transfer just through protein concentrations of transcription factors shared from mother to daughter cell. Indeed, this method has been used as an event recorder in gut bacteria [38]. However, protein-based switches can only be read out if the cells are still alive and the number or identity of stable states cannot easily be designed. Previous work using integrases to irreversibly invert or delete pieces of DNA in response to stimuli [9, 31, 60] has surpassed the ability of these protein-based switch systems. A temporary increase in RNA transcription, then, is converted through the action of integrase proteins into a permanent, lasting change in DNA. Phage integrases are extremely useful proteins to employ in this regard because their action to recombine specific DNA sequences is deterministic, fast, and irreversible [67]. The ability to compose integrase sites and easily employ many orthogonal integrases in the same cell allows flexibility and design freedom that is not possible with protein-based switches [82].

The integrase mechanism begins with proteins binding to attachment sites. In the case of serine integrases, attachment sites are short sequences of DNA that can bind a dimer of integrase molecules. An active complex involves two attachment sites, each bound by a dimer of integrases, which in turn are bound together to create an

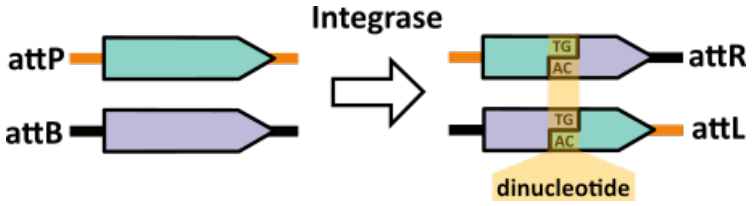


Figure 2.1: Serine integrase recombination. Two sequences (attachment sites) named attB and attP are recombined in a parallel configuration to produce two new attachment sites. Arrow-shaped sites point from 5' to 3'. AttL consists of the left half of attB and the right half of attP, and attR consists of the left half of attP and the right half of attB. A dinucleotide at the center of the attachment site makes sure that the sites can only be recombined if they are both pointing in the same direction.

(attachment site)-(integrase tetramer)-(attachment site) sandwich. For most serine integrases, these attachment sites have a different identity, and the reaction can only proceed between attB and attP, and not any other combination of sites (Figure 2.1). Once this complex is assembled, the integrase tetramer induces a double strand break in the core of each attachment site and allows the broken DNA to rotate around the site of protein binding and re-ligate in a different (or the same, after a 360 degree rotation) orientation, eventually resulting in a recombination reaction. For serine integrases like Bxb1, there is no need for extensive sequence homology in the attachment sites, besides the core dinucleotide. Recombination takes place regardless of what other sequences are adjacent, meaning that the location and orientation of attP and attB free to be engineered to create specific DNA states post-recombination. For example, placing the attP and attB sites on the same piece of DNA, but in opposite orientations will cause a "flip" of the sequence between the attachment sites. Likewise, placing the sites in the same orientation leads to a deletion, as well as other possible configurations (see Figure 3.6). Using two orthogonal integrases in a nested arrangement of sites can create a situation where the order that the two integrases acts results in different DNA end-states that can be distinguished by fluorescence. For example, if integrase 1 acts first, it flips a sequence containing integrase 2's attachment site, which now changes whether integrase 2 will perform a deletion or flip reaction. This mechanism was used in Hsiao et al [31] and has been generalized to allow integrase attachment sites to be composed and nested arbitrarily [28].

However, integrase-based event recorders that flip or delete DNA typically have a limited number of attainable DNA states, meaning that only a few events can be recorded before the memory capacity is "used up." Previous work using Cas9 to

stochastically excise memory units in mouse stem cells [22] and more recent work using integrases [15] have addressed this problem by creating many copies of the "consumable" DNA logic element, such that any one reaction would randomly occur somewhere in this repetitive array, and subsequent reactions can still be recorded by affecting other unaltered array elements. Copying the same DNA logic element many times also has an added benefit that the element chosen by integrase or dCas9 is random, meaning that each lineage of cells has a unique history of which elements were deleted or flipped, allowing lineage tracing. However, lineage information can only be recorded until all such memory units have been used, and so the DNA modification rate must be precisely tuned so that the record is not consumed before the lineage of interest is created. It is desirable, then, to create an "unlimited" record so that precise rate tuning is not necessary.

The CRISPR system represents a natural chronological record [18] of stimuli where pieces of DNA corresponding to phages are inserted into the genome in the order in which the phages were encountered. Phage genomes are chopped into short oligos, which are incorporated into the front of the CRISPR array through the action of the Cas1 and Cas2 proteins. In so doing, the CRISPR system can keep inserting more phage sequences and extending the CRISPR array indefinitely. More recently encountered phages appear closer to the promoter at the front of the CRISPR array, thus those guides are produced in greater abundance than older guides that reside farther down the array. This allows the cell to focus its immune defenses against more pressing threats, while eventually forgetting the faces of long-vanquished foes.

Several groups have endeavored to harness this recording system to create a "DNA tape recorder" circuit. After the function of Cas1 Cas2 proteins was elucidated [54], Church et al showed that seeding the cytoplasm with electroporated oligos [64] would result in those oligos becoming integrated at the beginning of the array. Subsequently, Sheth et al applied the same concept to record the changing copy number of a plasmid, thereby demonstrating the possibility of creating a chronological record of chemical stimuli [63]. In short, over-expression of the spacer acquisition proteins Cas1 and Cas2 leads to a random chunk of DNA to be inserted into the CRISPR array, and by controlling the content of all DNA in a cell, spacer identity can be influenced. Using this system, the DNA tape recorder made by Sheth et al can identify the presence, absence, and chronological order of three different chemicals over a period of four days. An equivalent "DNA flipping" based event recorder would have to have 81 possible DNA states. Theoretically the system would allow

recording to continue indefinitely, but the low rate of spacer acquisition means that eventually the "signal" created by integrating spacers from the plasmid gets diluted out by nonspecific DNA.

2.2 Simulation of the Event Recording System

Overview

Serine integrases catalyze predictable recombination reactions in DNA. In this work we are primarily interested in which DNA species are present at the beginning and end of an "event," which is essentially a time during which integrase is present in the cell. For the purposes of the following simulations, we assume that DNA does not get preferentially degraded or replicated between when it is integrated and when it is sequenced. These assumptions allow us to focus on the consequences of integrase activity and the control over integrase attachment site choice, which is the focus of the work presented here. In general the reaction rates for integrase binding and recombination are not well known, and so the specifics of integrase reaction rates are not the focus here. Instead, we are particularly interested in the effects of integrase reactions on the populations of recombined DNA present inside the cell.

Deterministic simulation shows induction dependant genome array length

To understand the capabilities of a genetic circuit which can continuously integrate plasmids into a genomic locus, we constructed a simulation using BioCRNpyler[58]. BioCRNpyler is a chemical reaction network (CRN) automation tool which allows a flexible simulation complexity given a higher-level object-oriented specification of biochemical parts called Components. In this way we were able to create a relatively simple description of the event recorder circuit and generate a network containing thousands of species and reactions.

Integrase reactions had to be simplified to be used in the full event recorder model:

$$attB_{A_1:A_2} + 2I_{Bxb1} \underset{k_{u,int}}{\overset{k_{b,int}}{\rightleftharpoons}} attB_{A_1:A_2} : 2I_{Bxb1} \tag{2.1}$$

$$attP_{B_1:B_2} + 2I_{Bxb1} \underset{k_{u,int}}{\overset{k_{b,int}}{\rightleftharpoons}} attP_{B_1:B_2} : 2I_{Bxb1} \tag{2.2}$$

$$attL_{A_1:B_2} + 2I_{Bxb1} \underset{k_{u,int}}{\overset{k_{b,int}}{\rightleftharpoons}} attL_{A_1:B_2} : 2I_{Bxb1} \tag{2.3}$$

$$attR_{B_1:A_2} + 2I_{Bxb1} \underset{k_{u,int}}{\overset{k_{b,int}}{\rightleftharpoons}} attR_{B_1:A_2} : 2I_{Bxb1} \tag{2.4}$$

$$\begin{aligned} attB_{A_1:A_2} : 2I_{Bxb1} + attP_{B_1:B_2} : 2I_{Bxb1} \xrightarrow{k_{int}} \\ attL_{A_1:B_2} : 2I_{Bxb1} + attR_{B_1:A_2} : 2I_{Bxb1} \end{aligned} \tag{2.5}$$

As seen in Equation (2.1) and Equation (2.2), two integrase molecules bind to one of two approximately 50 bp sequences known as an attachment site, indicated as attB or attP. This is already a slightly simplified version of the real reaction, which would probably involve first one integrase monomer binding, and then recruiting the other one. Other integrases besides Bxb1 can form dimers or tetramers in solution, but Bxb1 does not [25]. Similar binding reactions also occur for attL and attR. Subscripts indicate the identity of the DNA sequences at either end of the attachment sites, so when a recombination occurs between attB and attP to produce attL, the DNA 5' to attB and that 3' to attP are now on the same strand, and similarly with attR.

We chose to simplify this reaction to eliminate intermediate steps where integrase bound to attachment sites would be represented. This simplification stems from an assumption that integrase binding and unbinding is much faster than recombination. This is a reasonable assumption because integrase binding is simple and mechanically similar to protein-protein binding reactions which occur at the speed of diffusion. However, integrase-mediated recombination involves many sequential steps: the dimer-DNA complexes form a tetramer-DNA complex, the DNA is cleaved, the complex rotates, and then the DNA is ligated again [67] (see Figure 2.1). A more simplified mass action form of the integrase recombination reaction can be described as follows:

$$attB_{A_1:A_2} + attP_{B_1:B_2} + 4I_{Bxb1} \xrightarrow{k'_{int}} attL_{A_1:B_2} + attR_{B_1:A_2} + 4I_{Bxb1} \tag{2.6}$$

To account for the integrase binding to attachment sites such as attL and attR that cannot participate in a reaction, the rate of integration k'_{int} should be smaller than the rate shown in Reaction (2.5). Thus in the following simulations we use $k'_{int} = \frac{k_{int}}{N_{attsite}}$ where $N_{attsite}$ is the total number of attachment sites in the initial condition of the simulation. Since integrases only rearrange attachment sites and do not create or destroy them, and we do not simulate plasmid replication, this number is constant throughout the simulation.

A piece of DNA containing integrase sites can undergo three different types of reaction: inversion, deletion, and integration. Inversion occurs when attB and attP are present on the same piece of DNA, in opposite orientations, and results in the DNA in between these two sites being reversed in its orientation. In the context of event recording, inversion has been used to great effect in order to create a permanent, measurable DNA "state" [9, 31, 82]. However, in this work we have endeavored to arrange all integrase sites in the same direction, meaning that only deletion and integration reactions are relevant. A deletion reaction, then, is one where integrase sites oriented in the same direction on the same piece of DNA combine to form a shorter piece of DNA, and a free circular DNA. In the context of this work, such a reaction could happen within a single plasmid $P_{N:attB:attP}$ containing both attB and attP, and result in two "dead end" plasmids $D_{1:attL}$ and $D_{2:attR}$ that cannot participate in future integrase reactions (see Reaction (2.8)). Finally, integration is a reaction where attB and attP are on two separate pieces of DNA. If these DNA reactants are linear, then the products are also two pieces of DNA, where the corresponding parts on either side of attB and attP have exchanged. If one of the reactants is circular, then the result is a single linear piece of DNA where the circular plasmid has been completely integrated, as can happen between the genome site G_{attB} and a barcode-containing plasmid $P_{N:attP}$ (see Reaction (2.7)). Likewise, an integration can also occur between two circular plasmids, what will subsequently be referred to as an "inter-plasmid" reaction, to make one larger circular plasmid, as in reaction (2.9).

$$G_{attB} + P_{N:attP} + 4I_{Bxb1} \xrightarrow{k_{int}} G_{attL:P_N:attR} + 4I_{Bxb1} \tag{2.7}$$

$$P_{12:attB:attP} + 4I_{Bxb1} \xrightarrow{k_{int}} D_{1:attL} + D_{2:attR} + 4I_{Bxb1} \tag{2.8}$$

$$P_{N:attB} + P_{N':attP} + 4I_{Bxb1} \xrightarrow{k_{int}} P_{attL:N:N':attR} + 4I_{Bxb1} \tag{2.9}$$

Given these three rather simple possibilities, an arbitrarily complex model can be created. Since every integrase reaction occurs between attB and attP, a plasmid containing both attB and attP that reacts with another plasmid now creates a new plasmid that, again, contains attB and attP. Likewise, a plasmid reacting with a genome site containing attB would normally result in a non-reactive linear product containing attL and attR only, except that in this work the plasmids contain attB also, such that this reaction then leads to replacing the "consumed" attB with a new one. Thus the more time that integrase is allowed to act on the system, the more new species of DNA are created that can participate in the same three possible types of integration reactions. This presents a problem for a simulation that has

to keep track of a seemingly infinite number of different species and reactions, even though at the beginning almost none of them are present. Similarly, writing all of these species and reactions by hand in order to produce a chemical reaction network that can be simulated [74] becomes impractical rather quickly. Automated CRN generation features of BioCRNpyler[58] are essential to make our simulation possible, but we must also make a choice as to the recursion depth allowed when species are generated. In other words, we will not generate an infinite number of multiply-integrated states, but limit the amount of sequential integration reactions that can occur to any of the species in the model. Conveniently, we have seen that many sequential integrations are relatively rare in vivo, meaning that it is reasonable to neglect their existence.

One critical parameter of a recording system is the range of input values that can be recorded. In our case the presence of integrase determines the rate of integration, but the final amount of integration is the actual record. Thus the total integrase activity, rather than the amount of integrase or duration of induction, is the actual input value. If a recording system is to be useful, then, there should be a distinguishable difference between different input values. In Figure 2.2, the different possible genome sites (with different numbers of plasmids integrated) are represented on the X axis, so a set of dots of the same color is the number and type of genome sites present in the simulation after a specific length of time. Figure 2.2. A indicates the ideal scenario where active integrase leads to plasmids being repeatedly integrated into the genome with no undesired reactions. The average number of integrations in the genome site then continues to increase the longer that the integrase is present. If a population of cells contains this event recorder, a large enough sampling of the genomes of this population should yield a distribution of genome site lengths that is significantly different for a wide range of induction times, what will be referred to as "dynamic range." The other plots (Figure 2.2 B–D) simulate a system where intra-plasmid deletion (B), inter-plasmid integration (C), and both (D) are also possible. These "undesired" reactions effectively allow integration events to go unrecorded, since instead of leading to genome record extension, an integration reaction may now remove a plasmid from being ever able to be integrated, or create a plasmid multimer which is not integrated into the genome site. Multimerized plasmids are still capable of integrating into the genome, although they lead to signal degradation in a different way: an integrase reaction that results in a multimer integration expands the genome site by many plasmid instances instead of one at a time, as it would if only a single plasmid got integrated at a time. Interestingly, it seems that the

recording ability of the system at lower input values (visualized as the difference between $t = 1$ and $t = 30$) is largely unaffected by the undesired reactions.

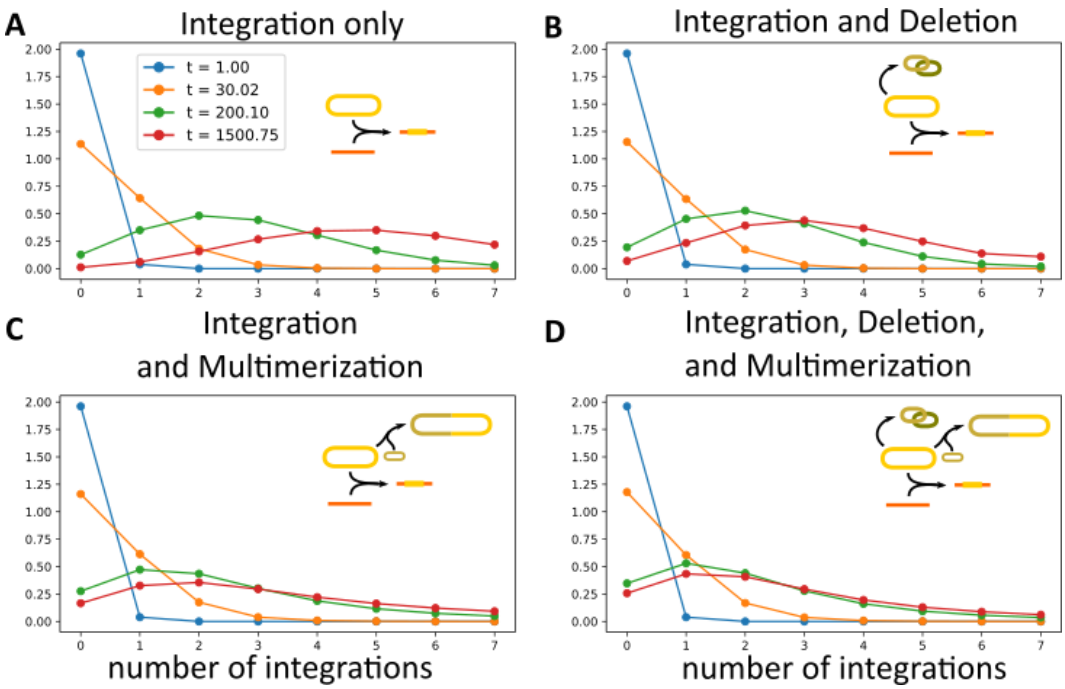


Figure 2.2: Simulation of integrase site expansion by repeated plasmid integration. Horizontal axis shows different numbers of times that the plasmid is integrated into the genome site, and the vertical axis is molecules. (A) Integration reactions allowed only, so a plasmid containing attB and attP can only react with the genome which contains attB. (B) Integration and deletion reactions are allowed, so now the attB and attP sites on the plasmid can react intramolecularly to yield a "dead end" deleted product. (C) Integration and multimerization reactions are allowed; now the plasmid can react with other copies of itself to form a multimeric plasmid, and the original plasmid as well as the multimers can integrate into the genome, but intramolecular "deletion" reactions do not happen. (D) all three reactions are allowed. In all cases, simulations were started with 10 plasmids, 2 genomes, and 5 integrase molecules. For full simulation parameters, see Table 2.2.

Multiple Ink Plasmids

The ability to record the presence of a single stimulus is interesting, but has been achieved using much more elegant genetic circuits before [9, 31]. The advantage of the work presented here is that it can be easily expanded to recording two or more stimuli without significant changes to the overall design. DNA is integrated repeatedly into a recording locus, but the source of DNA is a set of plasmids known as "ink" plasmids. By adding an orthogonal ink plasmid, we should be able to record the order and relative magnitude of two events occurring in sequence. Along with the

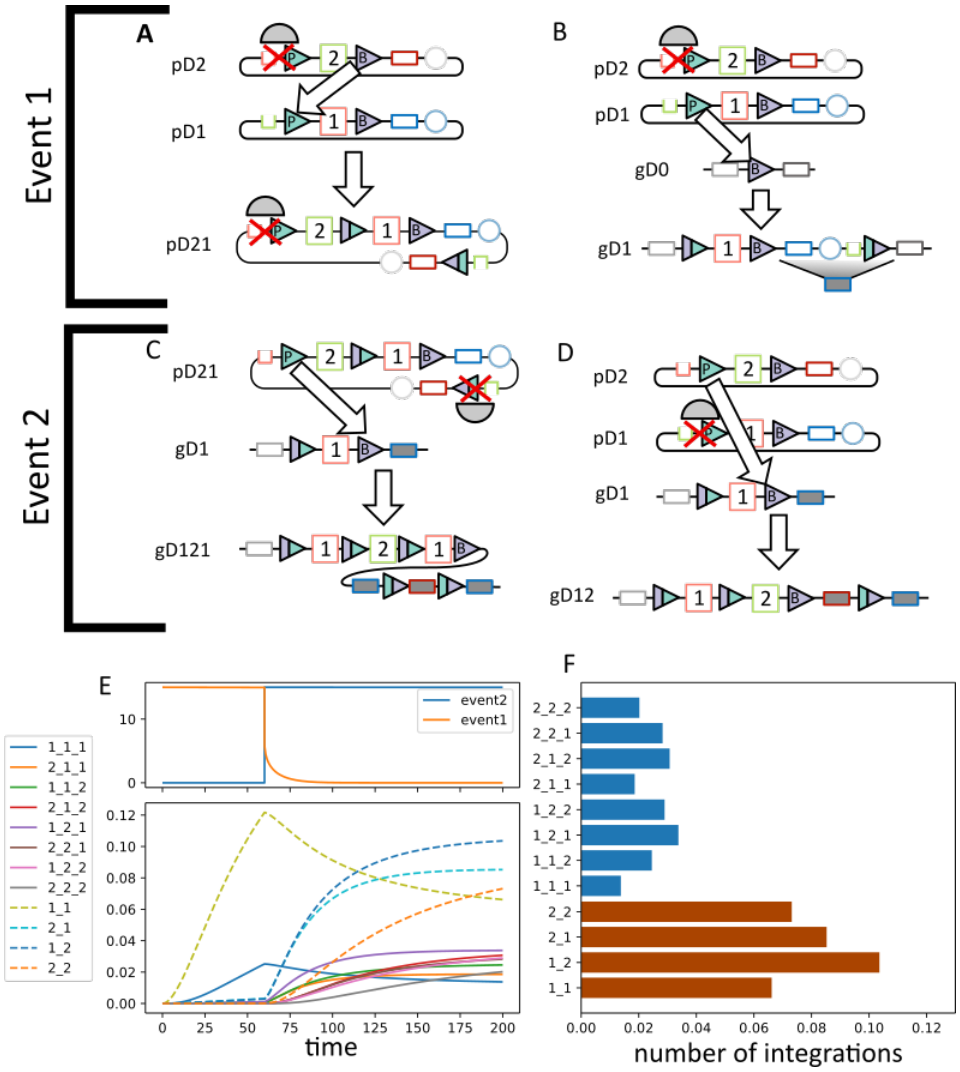


Figure 2.3: Sequential integration simulation. (A) integrase reaction resulting from two ink plasmids recombining to form a dimer, while plasmid 2's attP site is blocked. (B) integrase reaction resulting from the unblocked ink plasmid integrating into the genome. Grey square with blue outline represents shorthand for DNA sequences derived from pD1. Likewise, grey square with red outline represents shorthand for sequences derived from pD2. (C) Integrase reaction resulting from ink plasmid dimer (product of part A above) integrating into the genome after plasmid 2's attP site becomes unblocked. (D) Integrase reaction resulting from ink plasmid 2 integrated into the genome, which already contains a record from event 1. (E) Time series simulation of the genome site of an event recorder experiencing event 1 followed by event 2. Only 2-length (dashed) and 3-length (solid) genomes are plotted for clarity. Initially, the genome record "1_1" is the most prevalent, indicating that mostly plasmid 1 integrates and plasmid 2 is blocked during "event 1." When "event 2" is present and event 1 is removed, 1_2_1 becomes the most common 3-length genome instead of 1_1_2 as one may expect. (F) endpoint amount of each genome type of length 3 and 2, from the simulation presented in E. Blue are genomes of length 3, orange are genomes of length 2.

addition of a new ink plasmid there is also a new possible "unproductive" reaction: integration of the two plasmids with each other. Introducing more orthogonal plasmids might then lead to a diminished useful recording period. Ink plasmids are orthogonal in the sense that the integrase can be driven to react with one or the other. The mechanism for this integrase control is achieved by competitively blocking integrase activity through the binding of another molecule. This blocking molecule is specific to a certain ink plasmid and can be induced in the presence of outside stimuli that will be recorded.

To investigate the limitations of a two ink plasmid event recorder we constructed a simulation and subjected it to a changing inducer profile (Figure 2.3). Ink plasmids are integrated into the genome site at a specific location, meaning that newer plasmids are present closer to the attB site and older integrations are farther away. For example, a genetic record identity of "1_1_2" has the plasmid 1 barcode first, followed by plasmid 1 again, and most recently plasmid 2. Intuitively it would follow that a genetic record containing a sequence of "1_2," that is to say, plasmid 1 followed by plasmid 2, would be the most commonly found genetic record sequence when the "event 1" stimulus is presented to the circuit, followed by "event 2." Simulations predict that this is not the full story, however.

First, the rate of integration is such that if we induce event 1 for long enough to make sure that most records have at least one integration of plasmid 1, there will not be many free ink plasmids left to record future instances of event 1. This would be essentially depleting the "storage" of the event recorder for a single event, which would result in a very short usable recording time. Thus we are better off aiming for a scenario where most records have zero integrations, and therefore a fairly low chance of an existing record containing a "plasmid 1" to meet with a "plasmid 2" integration. This situation could be alleviated by waiting between events to allow the copy number of the depleted ink plasmid to regenerate. There is no real reason why ink plasmids would regenerate, however, since plasmid copy numbers are maintained by the number of origins present in the cell, and integrase action does not destroy the origins of the plasmids that got integrated, instead creating multimer plasmids with multiple origins.

Second, there is nothing preventing the two orthogonal ink plasmids from reacting with each other. In fact, this inter-plasmid reaction is more likely to happen than the plasmid-genome integration, since the ink plasmids are present in higher copy number than the genome recording site. The mechanism we have chosen to use for

Reaction	Rate
$G_{attB} + P_{N:attP} + 4I_{Bxb1} \xrightarrow{k_{int}} G_{attL:P_N:attR} + 4I_{Bxb1}$	$k_{int}[G_{attB}][P_{N:attP}][I_{Bxb1}]^4$
$P_{N:attB:attP} + 4I_{Bxb1} \xrightarrow{k_{int}} D_{N:attL} + D_{N:attR} + 4I_{Bxb1}$	$k_{int}[P_{N:attB:attP}][I_{Bxb1}]^4$
$P_{N_1:attB} + P_{attP_1:N'} + 4I_{Bxb1} \xrightarrow{2k_{int}} P_{N_1:attL:N_2:attR} + 4I_{Bxb1}$	$2k_{int}[P_{N_1:attB}][P_{attP_1:N'}][I_{Bxb1}]^4$
$P_{N:attP} + E_{N'} \underset{k_b}{\overset{k_u}{\rightleftharpoons}} P_{N:attP} : E_{N'}$	$k_f = k_b[P_{N:attP}][E_{N'}]$ $k_r = k_u[P_{N:attP} : E_{N'}]$
$P_{N:attB} : E_{N'} + P_{N:attP} \xrightarrow{k_{int}} P_{attL:N:N:attR} : E_{N'}$	$k_{int}[P_{N:attB} : E_{N'}][P_{N:attP}]$

Table 2.1: Reactions used in two-plasmid event recorder simulation.

plasmid integration control is by blocking and occlusion of the attP attachment site present on the plasmid. This blocking would not affect the attB site on the plasmid, which should remain fully functional and unblocked in case it has become the new integration site on the genome. Another ink plasmid has no way of "knowing" if an attB site is on the genome or on a plasmid and so should integrate with either without difficulty.

In the case of two ink plasmids where one of them has the attP site blocked, the other orthogonal plasmid is free to have its attP site react with the attB present on the blocked plasmid, as seen in Figure 2.3 A. This will happen at the same time as the integration with the genome (Figure 2.3 B). The product from this integration will be a multimer ink plasmid containing a hybrid of the two orthogonal ink plasmids. If the repression of attP site changes, this new hybrid plasmid will be allowed to react with the genome, thus leading to a non-intuitive record containing "121" integration, alongside more intuitive records containing "12" integrations (Figure 2.3 C and D). In order to fully understand the nature of genomic records, it is insufficient to simply read the order of the barcodes present, it seems essential to consider all possible integrations and to measure the concentration of all present barcode sequences.

Stochastic simulation reveals variability of integration length

Simulations presented in the previous section are deterministic, continuous simulations which are not sensitive to the effects of small copy numbers as present in real biological systems. In our system, the use of the genome as a recording

Parameter	Value
k_b	100
k_u	10
k_{int}	0.0004
k_{deg}	0.1

Table 2.2: Parameters used in two-plasmid event recorder simulation.

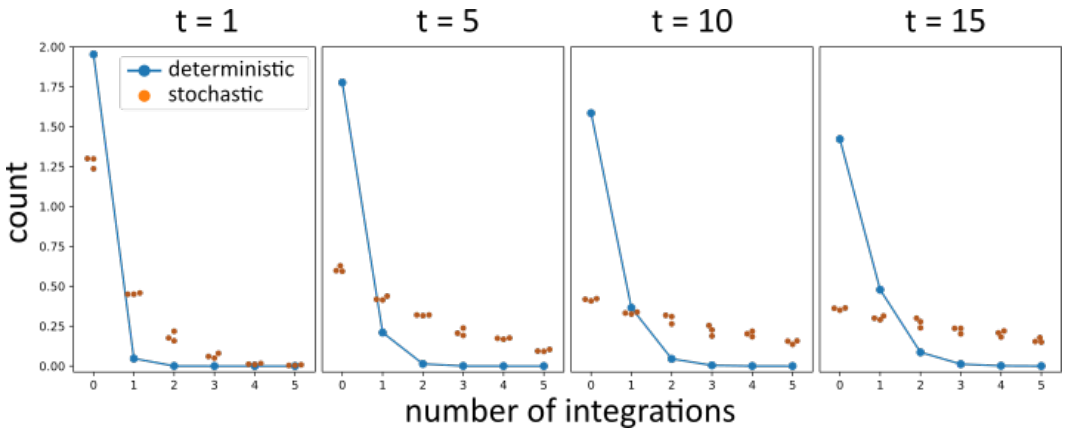


Figure 2.4: Stochastic simulation of repeated plasmid integration. Intermolecular and deletion reactions are allowed, similar to Figure 2.2D. Orange dots represent an average endpoint (at time indicated at the top of the graph) of 500 stochastic simulations. Blue connected dots represent endpoints of a deterministic simulation. Both simulations were started with 12 plasmids, 2 genomes, and 5 integrases. Because of low copy number, stochastic simulations result in more integrations per unit time than the deterministic case.

site necessitates the consideration of small copy number effects since the average genome copy number in *E. coli* is ~2 [57] and the copy number of p15a plasmids is ~20 [44]. In the stochastic regime we can see that the dynamic range is even less pronounced: comparing Figure 2.4 to Figure 2.2 D, we see that the records are already indistinguishable after $t = 30$ whereas the deterministic simulation suggested that it would be difficult to distinguish anything above $t = 200$. Thus any dynamic range estimations we can deduce from deterministic simulations are likely to be 10x higher than in the real system.

An important goal of event recording is the ability to reconstruct the nature of the event from the record. In order for this to be possible, the record created by a specific event must be distinguishable from that created by another, similar event. For this reason we have discussed the idea of dynamic range. A stochastic

simulation then gives us another way to look at this range, and that is to consider the natural variation inherent in the event recording mechanism. The stochastic nature of integrase activity creates noise in the record which would put further limits on the effectiveness of event reconstruction. If two different event sequences create a similar but distinguishable record as judged by the deterministic simulation, it may well be that the noise in the stochastic regime may render the two records indistinguishable.

If we assume that the event recorder is functional, then it is reasonable to say that each cell that encounters a set of inducer conditions creates a random genetic record which is somehow influenced by these inducers. The more cells whose records are sequenced, the greater confidence can be obtained about the shape of this distribution. Thus, to avoid the noise inherent in integrase-based event recording, we can simply sequence more cells. However, cells are also growing and dividing and in so doing they multiply their genetic material, which contains the record. This leads to numerous cells having the same record sequence because they came from the same parent, not because they independently arrived at the same sequence through a set of integrase reactions. This fact is useful for lineage tracing, but since cells in general make few integrations, and at most only two options will exist for what gets integrated, the likelihood that two records can appear the same even though they came from different lineages is fairly high. This makes the event recorder unfit for lineage tracing, and instead means that lineages of cells will confound measurements of record abundance in a population. Nevertheless, we must use population measurements of genetic records because single records are too random and similar between different event sequences.

2.3 Event Recorder Design

Our event recorder design consists of three components (Figure 2.5): a set of "ink plasmids" that serves as a source of DNA for integration, a "recording circuit" utilizing Bxb1 integrase that converts stimulus detection into plasmid integration, and an "integration site" where information is stored in the genome.

An ink plasmid's purpose is to provide raw material for Bxb1 integrase to act upon. As such, these plasmids are intended to have only the minimal components necessary for replication and integrase activity. Both attP and attB sites must be present on ink plasmids, as the attP site is used for integrating with the attB site on the genome, and the attB site is there to replace the consumed genomic attB site to allow for the

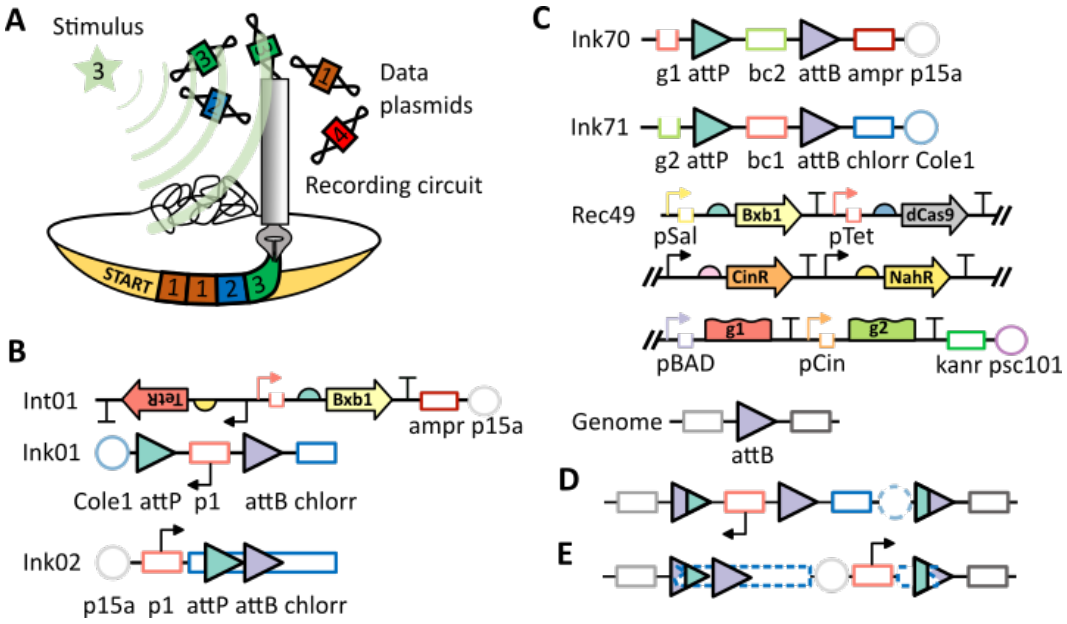


Figure 2.5: Event recorder concept (A) conceptual representation of the genome event logging circuit. Ink plasmids are selected by the recording circuit to be inserted into the integration site, directed by an external stimulus. (B) Constructs used to test sequential integration. Arrows are promoters, half-circles are ribosome binding sites (RBS), fat arrows are protein coding sequences, Ts are terminators, circles are origins of replication, and squares represent antibiotic resistance. Bxb1 is the integrase used, driven by pSal promoter which is activated by NahR. Black promoters are constitutive. Triangles are integrase sites. Genome site used was the same as in C. (C) Constructs used for two-ink plasmid event recorder system. Rectangles labeled "bc1" or "bc2" are sequences used to identify which plasmid was integrated. U-shapes labeled "g1" and "g2" are guide RNA binding sites. Wavy-edged boxes are guide RNA sequences. Cells used in this case were DH5αZ1, which contain TetR and AraC, which are essential transcription factors for pTet and pSal, respectively. Double slash represents a continuation of the same plasmid onto the next line. Plasmid Rec48 is the same as Rec49 except with the identity of g1 and g2 swapped. (D) Product of integrating plasmid Ink01 into the genome; non-functional ColE1 origin shown as dashed circle. (E) Product of integrating Ink02 into the genome; non-functional ChlorR shown as dashed broken rectangle.

next integration. Ink plasmids also contain a unique sequence between the attB and attP sites which serves as a barcode for identifying which plasmid was integrated. The first plasmids we tested for repeated integration are depicted in Figure 2.5 B, called Ink01 and Ink02. Ink01 contains a sequence in between attP and attB which also contains a promoter. This promoter is designed to replace the promoter present at the 5' end of the ColE1 origin. The motivation behind this design is that once the plasmid is integrated by recombining attP with attB on the genome,

it would be linearized such that this promoter would end up pointing away from the origin of replication (Figure 2.5 D). In so doing we intended to minimize the activity of the ColE1 origin once it was present on the genome. We used a strong promoter and so this design led to high plasmid copy numbers [55]. Ink01 and Ink02 were abandoned in favor of designs depicted in Figure 2.5 C. We abandoned this promoter-containing design for a few reasons. First, in order to allow more complete binding of the attP site by dCas9 in the two-plasmid system, we needed a lower plasmid copy number. Second, plasmid-plasmid integration reactions could happen, and would create a multimer plasmid with many copies of this promoter all pointing in the same direction. Multi-promoter containing multimers had an even higher replication rate than single plasmids containing this promoter and that led to cells where multimer plasmids were building up too much.

Ink02 is designed so that once integrated, the promoter would no longer drive chloramphenicol resistance (ChlorR) (Figure 2.5 E). The idea of this plasmid is to make sure that the ink plasmid kept being replicated, since only unintegrated plasmids would have functional chloramphenicol resistance. This plasmid was also abandoned because it did not result in significant ink plasmid retention. Also, this design did not have space between attP and attB for a plasmid-specific barcode, since attP and attB have to be in the coding frame of the chloramphenicol resistance gene.

To allow two ink plasmids to be used in the same cell at the same time we developed an orthogonal ink plasmid containing the P15A origin, and a different barcode and guide RNA binding site (Ink70 and Ink71, Figure 2.5 C).

The recording circuit is designed to inducibly express integrase and/or guide RNAs. The biggest problem we faced when designing this plasmid was dealing with leaky promoters. The salicylate-induced promoter pSal [46] was chosen to drive integrase expression because it had the lowest leak out of the ones we tested. We found that integrase expression was the most sensitive to leak, since the slightest amount of integrase was enough to cause integration. A low copy number plasmid origin was also essential for minimizing integrase leak, so we initially used P15A (~20 copies per cell, Figure 2.5 B) but later moved to pSC101 (Figure 2.5 C as it is lower copy (~10) and we wanted to use the shorter P15A origin for the Ink71 plasmid. We needed to include four inducible promoters on this plasmid for control of integrase, dCas9, and two guide RNAs. We chose pTet (induced by anhydrotetracycline, hereafter referred to as aTc) for dCas9 as we found that it was sufficiently able to

control dCas9 expression tightly and express the protein at a high enough level for easily detectable function. Pcin and PBAD were chosen for the guide RNAs as the molecules that these promoters respond to (arabinose for PBAD and 3OHC14-HSL for pCin, hereafter referred to as Cin AHL) could be found in the environment [13, 62] and might be interesting to record.

The genome site must contain an attB sequence suitable for plasmid integration, surrounded by known primer binding sites that can be used for recording site PCR and readout. To integrate this site we used the POSIP KH system [70] into the genome of DH5α-Z1 cells [44] which already contain transcription factors TetR and AraC for control of pTet and pBAD promoters. Primers on either side of the site must be designed such that they are not present in the genome, so they can be used to PCR the genetic record. We made use of the Unique Nucleotide Sequences [77], as well as designing custom non-genomic sequences by generating random nucleotides and then checking them for genome binding using primer3[37].

The intention was that our system could offer several advantages over other "tape recorder" like event recording system, such as that developed by Sheth et al [63]. First, phage integrases are much more active and their recognition site is more well-defined than that of cas1-2, which allows our system to react faster than cas1-2 while being less toxic, since phage integrases will not interact with the *E. coli* genome if their cognate attachment site is not present. Second, our system can allow integration of any size of DNA fragment, which can lead to wider applications such as stimulus-directed pathway assembly or programmed integration of promoters and other active genetic elements. We envision these systems being used to produce "molecular sentinels"—bacteria that can be seeded in a river or a waste treatment plant or a gut microbiome to record chemicals or proteins present over time in a much less obtrusive way than using conventional means.

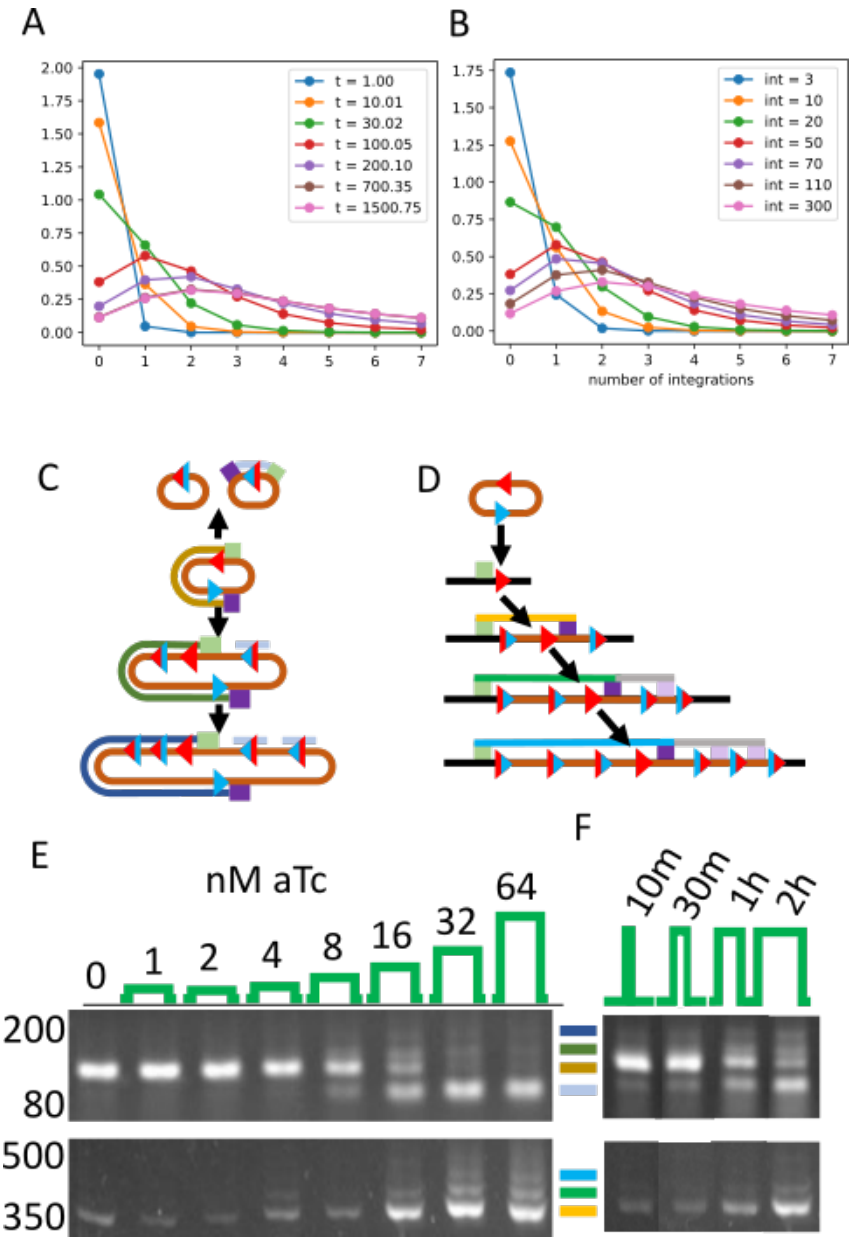


Figure 2.6: Continuous integration proof of concept. Caption on next page.

Figure 2.6: (A) Plasmid continuous integration simulation depicting amounts of different numbers of integrations as integrase is induced for longer. Simulated using $plasmid = 12$, $genome = 2$, $integrase = 5$. (B) Same as in A except induction time is constant at $t = 10$, and integrase molecule number is varied. (C) Schematic of plasmid-plasmid integrations, and different PCR products that can be obtained using the same two primers. Triangles represent integrase sites, with blue = attP, red = attB, and two color triangles representing non-functional attL or attR sites. Colored rectangles represent primer sites, and lines between colored rectangles are different PCR products. (D) Schematic of genome integrations, and PCR products obtained to determine integration amount. PCR products represented as colored lines used to determine integration amount. Gray lines represent possible PCR products that are not seen because they are much longer (and therefore the shorter products would dominate in a PCR reaction). (E) Agarose gel of PCR from cell cultures which were induced for 1 hour given different quantities of aTc from 1 to 64 nM. PCR product sizes are indicated with colored rectangles corresponding to colored lines from C and D. DNA ladder size indicated on the left. Top gel shows PCR products from plasmid multimers, and bottom gel shows PCR products from genomes. Different primers were used from the same cultures to obtain different PCR products. (F) Same as in E, except now cells were induced with 16 nM aTc for differing amounts of time.

2.4 Experimental Validation of Sequential Integration as a Way of Recording Events

A proof of concept sequential integration strain was made using three components: an "ink plasmid" with both attB and attP sites, an inducible integrase cassette, and a genomically integrated attB site. Integrase induction was inducible by addition of aTc. Upon addition of different quantities of aTc for one hour, increasing amounts of plasmid-plasmid recombination and plasmid-genome integration are visualized by PCR. Similarly, induction with the same amount of aTc, but for differing lengths of time, leads to similar changes in genome integration amount and plasmid-plasmid integration amount, as predicted by the simulation in Figure 2.6. As expected, the identity of the genome-specific PCR products was confirmed to contain multiple copies of the ink plasmid using Nanopore sequencing.

Plasmid-plasmid reactions and plasmid-genome integration rates seem to be similar, based on PCR banding. As time or induction amount was increased, more integration happened, although there was no visible difference until 16 nM induction for 1 hour. 300 nM is generally considered a maximal induction level for pTet, so the fact that we saw integration at less than 1/10th that amount suggests that in the real system, integrase is active at fairly low concentrations, as the simulation predicts. Both plasmid and genome PCR products showed visibly increased integration at the same induction level, which may be because there was not enough granularity in induction levels, or that the rate of plasmid-plasmid reactions is equivalent to the rate of genome-plasmid reactions. The simulation suggests that the rate of plasmid-plasmid reactions should be greater, purely because there are more plasmids than genomes. In this construct, we are using the ColE1 origin of replication, which is regulated at ~50-70 copies per cell [44], and should be significantly higher than the number of genomes per cell.

Repeated plasmid integration into the genome would create long multiply-repeated DNA regions that could cause stress, and undesired recombination. Cells were grown for 12 hours after induction in liquid culture before Nanopore sequencing, to determine that integrations in excess of 10x could be detected (Figure 2.7). In any case, integrations do not need to be stable in living cells to be detected using PCR, as even extracellular DNA from lysed cells could be amplified and measured. Even though these integrations create a highly repetitive genome region, there are no highly expressed genes present on these repeated elements, so it seems unlikely that there would be an excessive burden from these genome arrays. The presence of

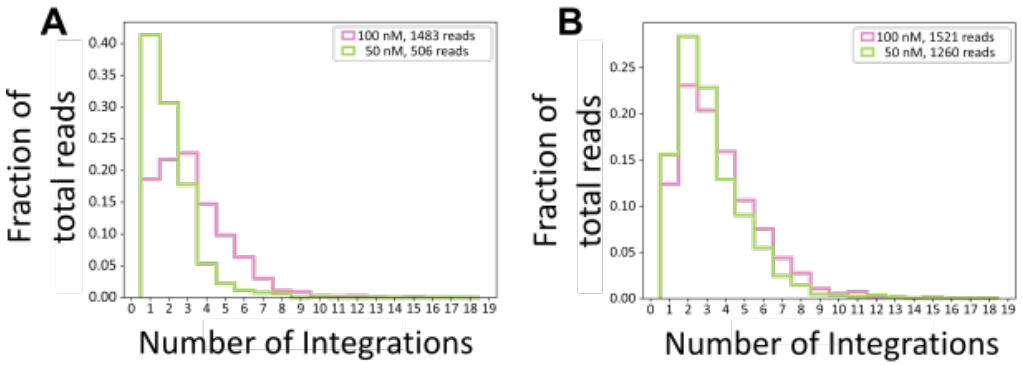


Figure 2.7: Long read sequencing quantification of integrations using Ink02 (A) or Ink01 (B). Genomic PCR products of sequencing reactions as depicted in Figure 2.6 D. Pink line represents histogram with 100 nM of aTc, green line with 50 nM aTc. Since we used primers spanning the genome-insert sequence, we could not sequence or compare integrated plasmids to unintegrated in this experiment. Thus, the minimum integration number that could be seen in this experiment is 1. Histograms were normalized so the fill area is equal to 1.

multiple copies of the ColE1 origin on the genome post-integration, however, could lead to instability. Ink01 has a ColE1 origin such that an integrase site was located between the RNAII promoter and the rest of the origin. Thus, after integration the promoter that drives replication would end up facing the wrong way, when integrated on the genome (Figure 2.5 D). However, these plasmids lead to an overabundance of multimers, since the multimers would end up with multiple promoters pointed into the origin, and thus replicate faster (see Figure 2.8). So we did not pursue methods of alleviating this multi-origin problem, resorting instead to simply sampling the cultures soon after induction to prevent the accumulation of any genome instability sequences.

2.5 Intramolecular Deletion Reactions Are Inhibited Through Context-Sensitive Plasmid Design

To create the best genomic record of integrations, we should prevent any integration reactions that do not lead to genomically integrated plasmids. Bxb1 integrase recombines attachment sites regardless of how they are oriented or what pieces of DNA they are located on. Ink plasmids must contain both attP and attB integrase sites so they can integrate into the attB site on the genome and provide a functional attB site that can be used for future integration. This co-localization of attB and attP sites can lead to intra-molecular recombination, that results in the region in between the sites becoming deleted from the plasmid, the sites then becoming converted

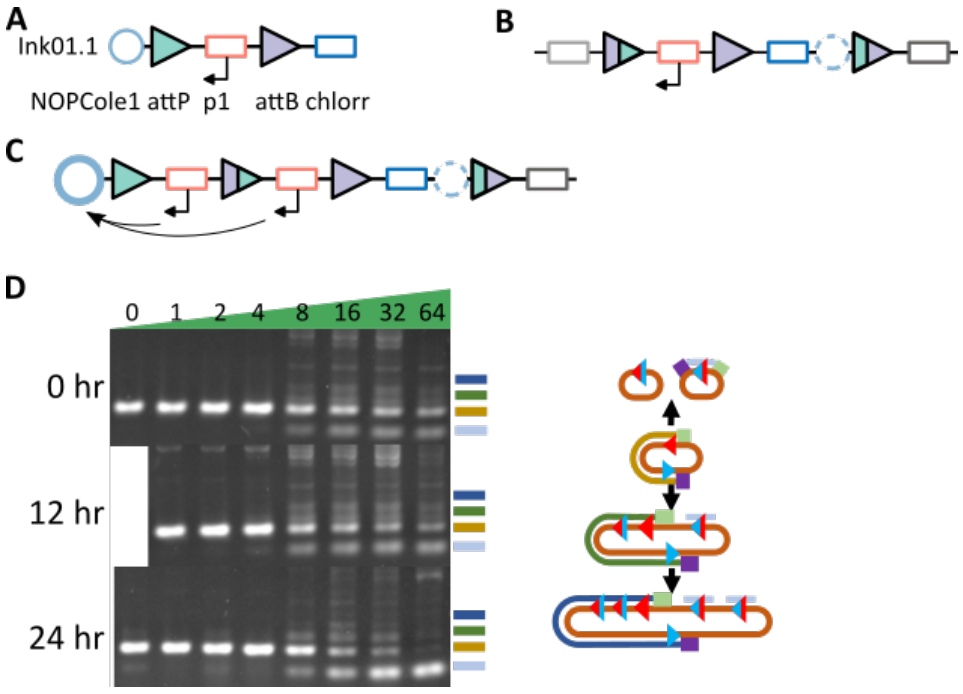


Figure 2.8: Multimer overgrowth with split origin. (A) Schematic of Ink01.1 plasmid, with promoter p1 pointing into NOPColE1, a ColE1 origin with no promoter. (B) Schematic of Ink01.1 integrated into the genome, with nonfunctional origin with dashed border. (C) Schematic of Ink01.1 plasmid, integrated with another copy of itself. This time, two promoters point into one NOPColE1 origin (heavy blue circle), while another one is non functional, with zero promoters (dashed circle) (D) PCR of integrated plasmids, with indicated nM of aTc induction. Cells were grown for 3 hours before inducer addition, then two hours after inducer addition, then diluted and allowed to grow for various lengths of time as indicated before PCR. Longer growth results in no change or enrichment of higher size bands to the detriment of lower size bands. Bands indicate PCR product made from plasmid multimers; grey means a deleted sequence, brown a normal unintegrated plasmid, green a dimer, and blue a trimer.

into attL and attR attachment sites that cannot be integrated (see Figure 2.2 B). Thus, any plasmids that undergo these reactions essentially turn into “dead end” products which are still replicated, but cannot participate in event recording. This intramolecular reaction might be preventable through design, however, because the integrase only recombines attB and attP if they can be arranged into a parallel arrangement [25]. It might be possible to create a plasmid where the sites are too close together to allow a DNA conformation that can bring the sites into the proper parallel 3d orientation.

To investigate if it is possible to create a construct that prevents intramolecular

deletion by having attB and attP too close together, we constructed a test substrate that, when recombined, leads to in-frame, fluorescent GFP (Figure 2.9 A). To favor intramolecular integrations, the reaction was done in cell extract, where DNA is 10 times more dilute than in bacteria. Then, the results of this ex vivo recombination reaction were transformed into cells, and GFP positive colonies were counted (Figure 2.9 B). Constructs containing a spacer between integrase sites less than 100 bp (measured from the edges of the integrase sites) had significantly less deletion activity, as indicated by the decreased fraction of GFP positive colonies. The shorter the better, but there was a definitive drop-off at 100 bp, which is close to the DNA persistence length of 150 bp [23].

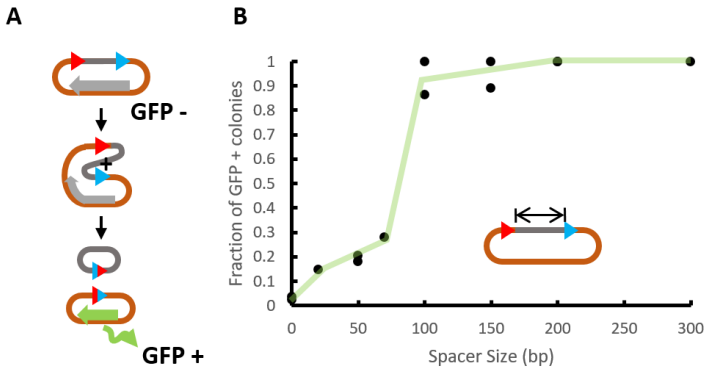


Figure 2.9: Minimal site spacing required for intramolecular integration. Plasmids containing a variable length spacer are incubated with integrase-expressing plasmid in cell-free extract. After incubation for 20 hours, plasmids are purified and transformed into competent cells such that each cell gets one or zero plasmids. Recombined plasmids yield green colonies, while un-recombined plasmids do not express GFP. Colonies were counted and the results are plotted. Green line is purely for visualization.

2.6 Decreased Copy Number ColE1 Plasmid Origin

We sought to use a low copy number P15a origin for an ink plasmid for several reasons. First, it meant that the rate of integration was lower, which makes the system more tolerant of leaky integrase regulation. Second, the dCas9 integrase blocking is more effectively able to block all available attP sites when there are less of those sites to block (because the copy number of the plasmid is low). But, a second ink plasmid must have an orthogonal origin of replication or else the relative copy number of the two plasmids will be unpredictable, and introduce unwanted noise. Thus, we wanted to use the ColE1 origin since PSC101 was already in use

for the integrase controller Rec49. So we sought to create a ColE1 origin with comparable copy number to the P15a origin by using error-prone PCR.

To construct this origin we created an error-prone PCR product from a wild-type ColE1 origin and cloned it into a plasmid with constitutive RFP. Then we picked colonies which had approximately the same fluorescence as a very similar plasmid that contained P15a instead of ColE1 (schematic shown in Figure 2.10A). Copy number of the new origin was estimated by comparing plasmid extraction yield from the same number of cells with a P15a plasmid or the wild-type ColE1 sequence. This origin was subsequently used for all experiments where two ink plasmids were used.

2.7 Control of Integrase Activity Using dCas9

Catalytically non-functional Cas9 (dCas9) can be used as a programmable DNA binding protein [18]. Cas9 is known to bind strongly to DNA, and affect the function of other DNA-binding proteins such as RNA polymerase. We sought to use this binding activity to out-compete the binding of integrase to attachment sites specified by different guide RNAs. As well as needing a cognate gRNA, dCas9 requires the presence of a protospacer adjacent motif (PAM) sequence to the 3' of the guide sequence [12]. In the case of SpCas9, this PAM is NGG. A test construct was made such that two identical attP sites are presented on the same vector, but only one of them has NGG in close proximity to the integrase attachment site. This allows a guide RNA to be designed that only binds to one of the two sites. An integrase blocking experiment was performed in TX-TL extract [73], in which dCas9, guide RNA, and integrase are expressed from linear DNA. When binding of dCas9 is activated by the production of guide RNA, integrase activity at one of two identical attachment sites is decreased (see Figure (2.11)). One interesting observation is that if more guide RNAs are being produced, less integrase activity is repressed at the repressed attachment site. Increasing guide RNA production in extract is enacted by adding more DNA from which the guide RNA is transcribed. This decrease in integrase repression may then indicate that 1nM of guide RNA-producing DNA already makes enough guide RNA to saturate dCas9, and adding more simply occupies RNA polymerase in such a way that less dCas9 is made, and less RNP complex can be assembled in order to repress integration.

It is difficult to predict the effectiveness of guide RNA sequences without empirical observations, so a variety of different guide RNAs were designed and tested (Fig-

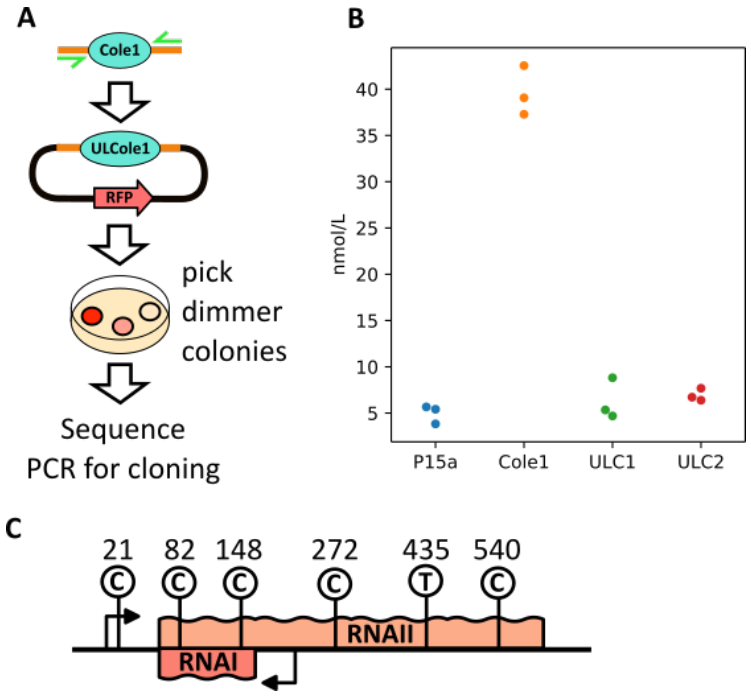


Figure 2.10: Error-Prone PCR method for constructing ultra low copy ColE1 origin. (A) Schematic of error prone cloning protocol. First, The origin is amplified in the presence of Mn^{2+} to produce the error-prone product, and then cloned into a plasmid expressing constitutive RFP. Then, colonies are picked with a brightness level similar to another plasmid which has a P15a origin. The colony is grown again and mini-prepped for use as a template and for sequencing. (B) A comparison of miniprep yields from wild-type ColE1, P15a, and two different colonies of Ultra Low Copy ColE1 (ULC1 and ULC2), which was created by this protocol. Miniprep yields were obtained from four separate 1 mL cultures of *E. coli* grown overnight to the same Optical Density. (C) schematic of mutations observed in ULC1. ULC2 had different mutations, but was not used in favor of ULC1. Letters in circles represent nucleotides which have replaced existing ones, with numbers on top indicating their position relative to the first nucletide of the RNAII promoter (far left, pointing right). RNAII and RNAI are both indicated as wavy-bordered rectangles, with RNAII on top reading 5' to 3' left to right and reverse RNAI on the bottom.

ure 2.12). Some guide RNAs, such as L2 (the same one tested in Figure 2.11) and B1, are extremely effective at shutting down integrase site activity. Other guides, such as M1, M2, and L1, have lower effectiveness. The lower effectiveness of M1 and M2 when compared to B1 is difficult to explain, but it shows that guide RNAs with a wide range of repression strength can be designed, in case such a binding strength range is desired.

In order to utilize two ink plasmids to record two events, we would need to develop

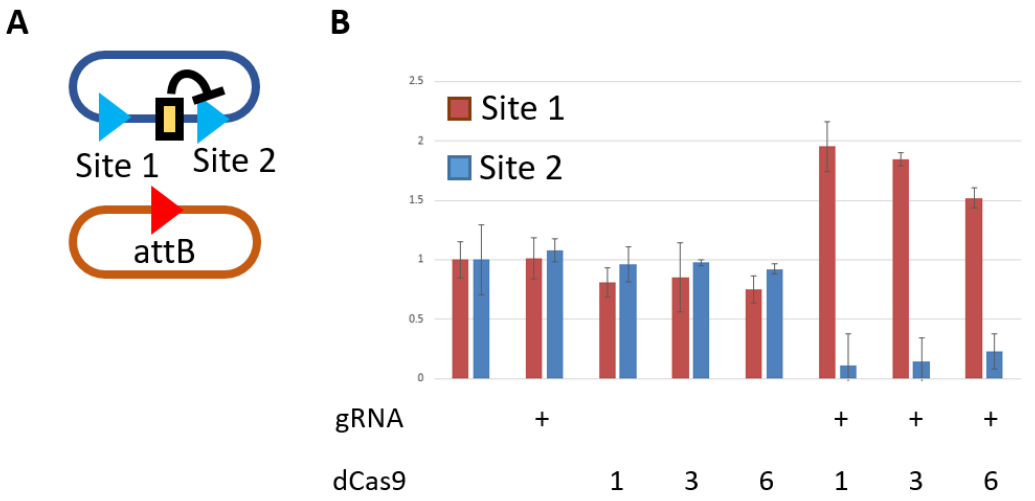


Figure 2.11: Integrase activity can be inhibited by the presence of dCas9 and a guide RNA designed to overlap one of two identical attP sites. (A) binding of dCas9 prevents integrase from binding and performing recombination at site 2. Without dCas9, either site should be able to react with attB. (B) Expressed in TX-TL, integrase reacts at sites 1 and 2 at approximately the same rate if gRNA or cas9 are added independently, but when both components are present, site 2 (blue bars) is recombined significantly less often than site 1 (red bars). Three different concentrations of gRNA are used: 1, 3, and 6 nM. dCas9 and integrase are present at 1 nM. These are concentrations of a linear DNA that expresses protein or RNA in all cases.

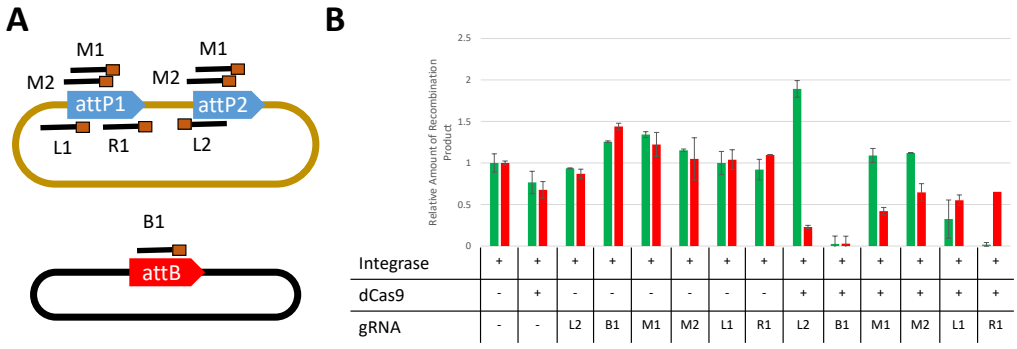


Figure 2.12: Testing of additional guide RNAs. (A) Map of guide RNA locations. Brown box represents the location of the NGG PAM sequence for each gRNA. M1 and M2 bind to both attP sites. (B) Repression effects from each guide. Red bars represent integration from site 1, and green bars represent integration from site 2.

two guide RNAs which can block integrase activity by specifically binding to one of two attP sites. Since we cannot change the sequence of attP or risk affecting integrase binding, we decided to use a partially overlapping guide RNA design. If the guide RNA partially overlaps the integrase site, then integrase might be prevented from full site access, but at the same time, a portion of the guide RNA binding sequence would be outside the attP sequence, and therefore could be used to design orthogonality. Since guide RNA sequences can tolerate some mismatches in binding sequence, we decided to investigate the minimum amount of overlap necessary for dCas9 to block integrase function. That way, we could design a guide RNA that had the most possible nucleotides outside the attP site, and therefore the highest chance to be specific to one attP site or another. To this end, we tested 8 guide RNAs, 5 that had the PAM sequence facing away from the integrase site, and 3 with the PAM sequence facing towards the integrase site (Figure 2.13).

Out of 8 guide RNAs tested we found that only two achieved maximum attachment site blocking, and three had intermediate activity. One of the two strongly blocking guide RNAs tested had only two nucleotides outside the attP sequence. For subsequent experiments, we focused on using a guide RNA design which overlapped the attP site by 9 nucleotides, leaving 11 nucleotides outside the integrase site, which could be altered to create orthogonal sequences.

Next we established a repertoire of orthogonal guide RNA sequences and evaluated their abilities to repress integrase activity. Using the 9 nt overlapping, 11 nt outside guide RNA position, 8 orthogonal sequences were designed which should allow up to eight Bxb1 attachment sites to be simultaneously controlled by guide RNAs. Repression of integrase activity was measured using the same attP site selection plasmid as before, where an unblocked site allowed GFP expression, and the site leading to RFP expression is the one that could be blocked. If the guide RNA was effective, cells would turn only green, whereas if no guide RNA was present the integrase could choose either the GFP or the RFP-creating attachment site. Each guide RNA has a cognate reporter plasmid, meaning that the plasmid contains the binding site for that specific guide RNA. Thus we could co-transform cognate and non-cognate guide RNA and reporter plasmid pairs, and thereby evaluate how well the “wrong” guide could repress integrase activity. Out of 9 guide sequences tested, 8 showed good orthogonality, and two (guide 0 and guide 4) were chosen to be used for the two-ink plasmid experiment.

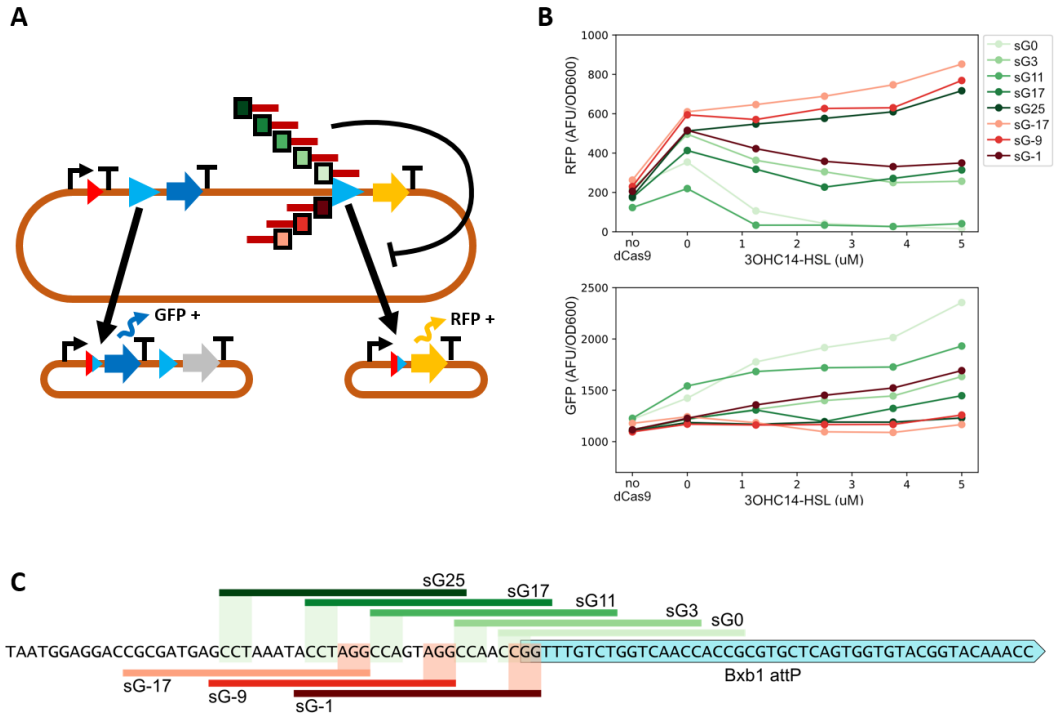


Figure 2.13: Guide RNA overlap optimization. (A) Construct schematic for spacing optimization experiment. 8 guide RNAs are designed such that they bind at different locations adjacent to the Bxb1 attP site (cyan triangle) upstream of RFP (yellow arrow). Guide RNA binding should lead to decreased RFP expression because plasmids that are recombined into the RFP expressing configuration are less numerous if the guide RNA dCas9 complex is in fact interfering with the integrase's ability to recombine plasmids into the RFP+ state. (B) Endpoint fluorescence values of cells exposed to different amounts of inducer that leads to guide RNA expression. dCas9 is driven by pTet promoter (induced with 300 nM aTc), and integrase by pSal promoter (induced with 100 uM Salicylate). Guide RNA sequences (legend) are depicted in C. (C) Guide RNA sequences used for spacing optimization. Shaded squares outline PAM sequence, colored lines indicate guide sequence. G<number> denotes the number of nucleotides between the end of the attP site and the 3'-most base pair of the guide RNA. For example, sG3 binds to the PAM sequence CCA (reverse complement is TGG) and so the 3'-most nucleotides of the guide bind to CCA, which is three base pairs outside of the attP site. Negative numbers indicate that the guide binding is on the opposite strand.

2.8 Integrase Controller Characterization

To evaluate the ability of the chosen promoters to produce guide RNAs at a level that could be useful for recording events, we utilized a reporter construct to test integration repression (Figure 2.15 E). A plasmid contains attB and two attP sites. Normally the plasmid produces no fluorescent proteins, as the constitutive promoter

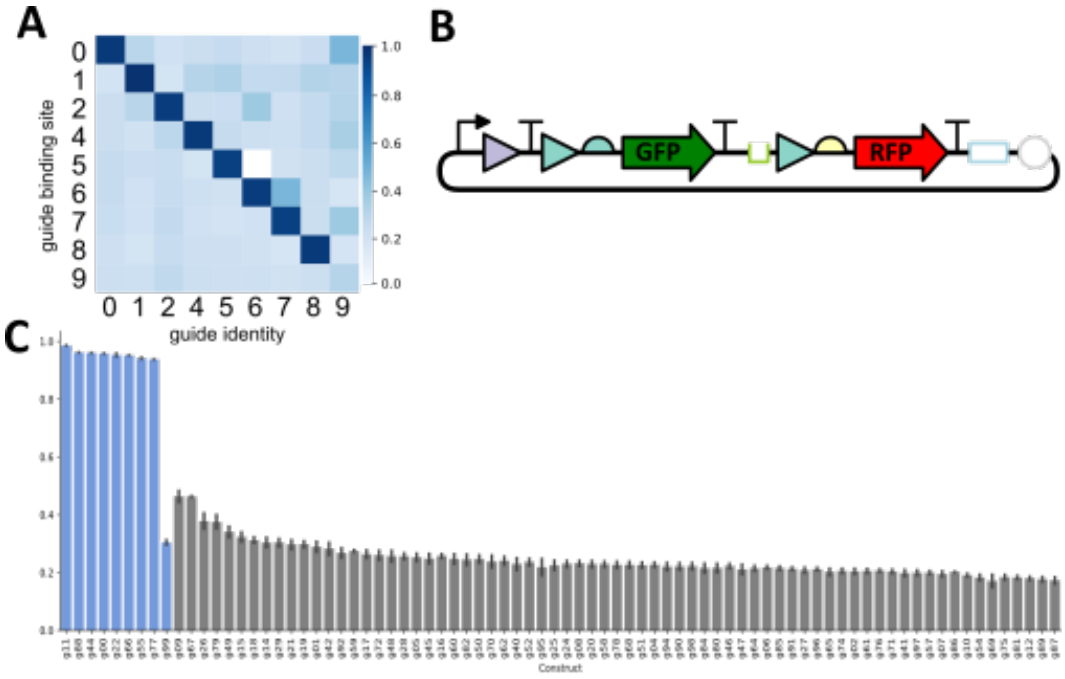


Figure 2.14: gRNA orthogonality (A) Orthogonality heat map. Plasmids expressing guide RNA 0-9 were co-transformed with plasmids containing binding sites 0-9. Most of the time the integrase would integrate the binding site which is supposed to be blocked, as indicated by a light color which represents a low gfp/total fluorescence. However, cognate guide RNA and binding site pairs integrated the unblocked site, which lead to GFP expression and a high gfp/total fluorescence ratio, which is indicated by a dark blue color. (B) Same results as in A but represented as a bar chart which shows an error bar plotting the results of two technical replicates. (C) A schematic of the construct used. Integrase sites are red triangles (attB) and blue triangles (attP). Guide RNA binding site is indicated by a blue square, next to the attP site in front of RFP. Thus, when gRNA dCas9 complex binds there, the integrase should integrate the site in front of GFP, leading to GFP expression.

would read through attB and into a terminator. But, integrase can recombine attB and one of the two attPs on the plasmid to delete the terminator, and allow RNA polymerase read-through into either GFP or RFP, depending on which attP was chosen. Without dCas9 action, some ratio of GFP to RFP fluorescence will be created, as the integrase chooses unimpeded between either attP site on the reporter plasmid. Leaky guide RNA expression from the arabinose promoter causes the point with zero inducer concentration to appear closer to the maximum arabinose endpoints.

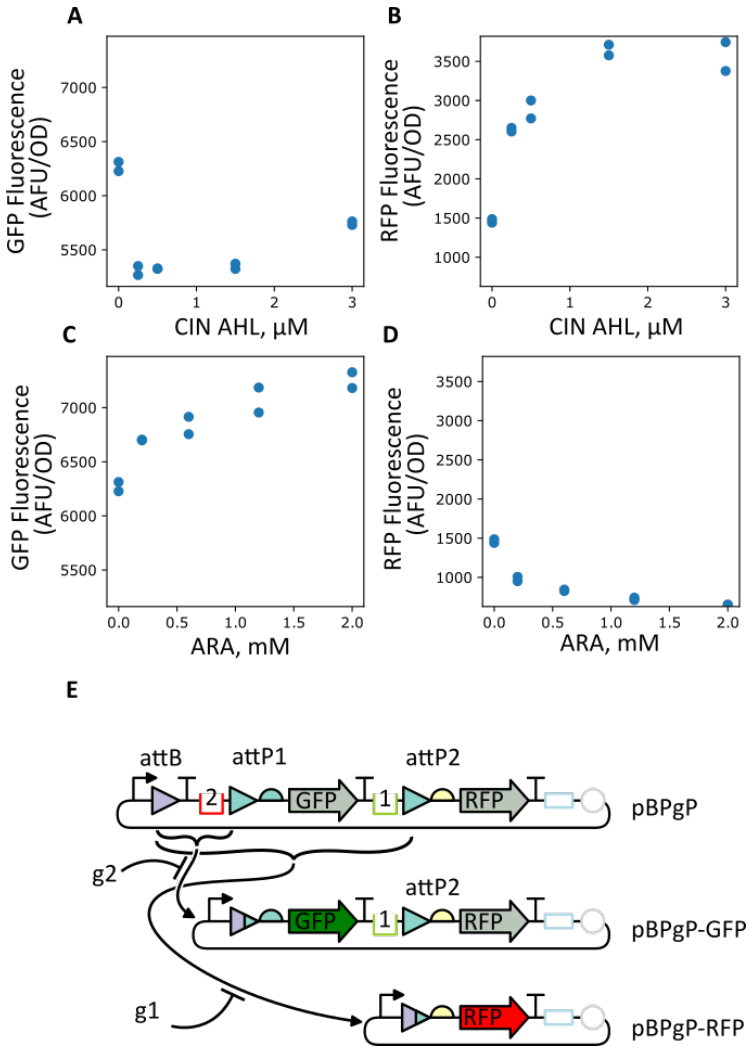


Figure 2.15: Attachment site repression inducer titration (A) GFP fluorescence of bulk cultures grown with labeled concentration of CIN AHL and also 100 μM Sal, and 300 nM aTc for 24 hours. Integrase controller Rec48 was used in this experiment, meaning that g1 is produced from an arabinose-responsive promoter and g2 from a CIN AHL-responsive promoter. (B) RFP fluorescence of the same cultures from A. (C) GFP fluorescence of bulk cultures grown with labeled concentration of arabinose (ARA). (D) RFP fluorescence of the same cultures from C. (E) Schematic of the reporter construct used to test Rec48 activity. In the unintegrated form, pBPgP cannot produce fluorescent proteins. If the integrase chooses attP1, the terminator in front of GFP is excised and pBPgP-GFP is formed. Conversely, if attP2 is chosen, the terminator and GFP are chopped out, and RFP can be produced from the resulting plasmid pBPgP-RFP.

When dCas9 and guide RNA are added, integrase activity becomes biased in the expected direction and the amount of GFP or RFP fluorescence at the end of culture growth is correspondingly greater or less than without gRNA expression. It is

important to note that the effects on fluorescent protein production do not have anything to do with dCas9 binding. The plasmid is constructed such that the bound dCas9 ribonucleoparticle (RNP) does not interfere with the promoter which is required for expression of GFP or RFP. Since the gRNA binding site would be on the left of the attP site, that region is cut out by the recombination event, which allows the promoter to read through into the fluorescent protein. Ideally, we would be able to count the number of molecules that are recombined into the GFP or RFP form. However, since plasmids exist in multiple copies inside each cell, this would be a difficult task even with flow cytometry. One could re-transform plasmids following integrase induction and count fluorescent colonies.

This "GFP or RFP" reporter plasmid also functions as an analog event recorder, as the level of guide RNA induction is "remembered" by the number of plasmids that have been permanently altered to produce GFP or RFP (Figure 2.15). Even after diluting cells into inducer-free media and growing again, the amount of fluorescence and thus the amount of GFP or RFP plasmids in the population is preserved (Figure 2.16). Discernible fluorescence values are produced by this system up to at least half the maximal inducer concentrations used in this experiment.

Cas9 is capable of fairly rapid binding of DNA, reaching 90% target site occupation within 10 minutes [34], but unbinding is substantially slower, taking about 40-50 minutes for 90% to unbind at 37°C [34]. This slow unbinding time means that an event recorder constructed using Cas9 may not be able to effectively record events that take place within a short period of time. To investigate the impact of slow dCas9 kinetics on event recorder performance we again made use of the "GFP or RFP" reporter plasmid pBPgP (Figure 2.17). When gRNA production is induced, ribonucleoprotein complexes (RNPs) of gRNA and dCas9 are formed, which then bind to the DNA of interest. When a different gRNA is subsequently expressed, we must wait for new RNPs to be formed, and likewise for the existing RNPs to unbind and be diluted out before the overall dCas9 repression effect "switches" to behave like the new guide RNA, as opposed to a combination of both. This gRNA switching is also relevant in considering how quickly the event recorder "forgets" that an event has happened, and how quickly two events can sequentially happen and still be recorded.

When guide RNAs are induced one after the other, we might expect some time during which the bound RNP from the first induction is still active when the second induction occurs. Delaying the second guide RNA induction should then allow more

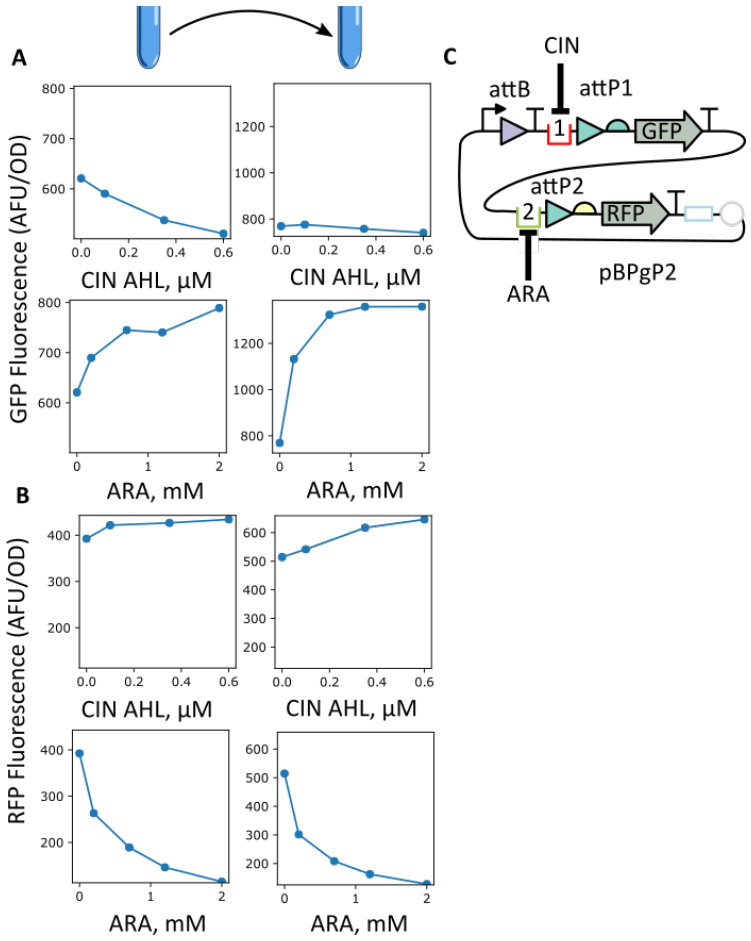


Figure 2.16: Maintenance of fluorescent protein expression after removing inducers. (A) GFP fluorescence of bulk cultures grown with labeled concentration of CIN AHL and also 20 μM Sal, and 60 nM aTc for 24 hours. Integrase controller Rec49 was used in this experiment, meaning that g2 is produced from an arabinose-responsive promoter and g1 from a CIN AHL-responsive promoter. Left column shows final culture fluorescence values in a tube with inducer, and right column shows final culture fluorescence after cells were diluted 1:100 into a well with no inducer, and grown again to saturation for 20 hours. (B) RFP fluorescence of the same cultures from A. (C) Rec49 and pBPgP2 were used in this experiment, so g1 is produced from the CIN AHL promoter and blocks GFP integration, and g2 from the arabinose (ARA) promoter and blocks RFP integration. See Figure 2.15.

of the previously loaded RNP to block integrase activity. We chose to investigate this question by enacting a sequence of inducer additions and culture dilutions to achieve the following culture trajectory: First, a guide RNA is induced along with dCas9 for 3 hours. Then, cultures are diluted 1:100 into a well containing integrase and dCas9 inducer (but no guide inducer). Another inducer is then added to this new well some time later to activate production of the second guide RNA (Figure

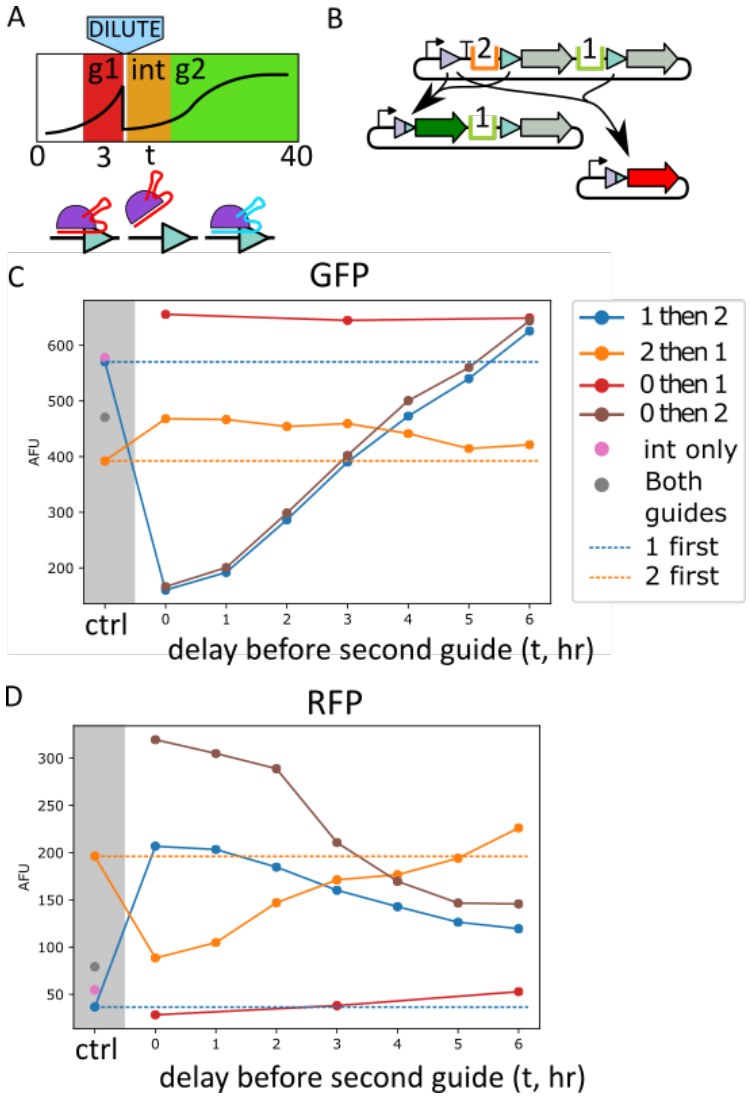


Figure 2.17: gRNA switching experiment. (A) Experiment schematic: cells were grown until log phase at which point inducers were added to express the first guide RNA and dCas9 for 3 hours. Then, the culture was diluted 1:100 into a new well containing integrase inducer. This culture was allowed to grow for some number of hours from 0-6 before adding inducer for the second guide RNA. Endpoint fluorescence is collected after 40 hours total. Black line represents OD600 curve. Cartoons below the graph represent a schematic of the expected behaviour: dCas9 binding, then unbinding, then the second guide binding. (B) reporter construct used in this experiment. Plasmid is the same as in Figure 2.15 E. (C) GFP fluorescence/OD at the endpoint of the cultures. "1 then 2" means guide 1 followed by guide 2, "2 then 1" means 2 followed by 1, etc. "0" means no guide inducer was added. Leftmost set of dots are control wells where no second guide inducer was added after cultures were diluted. (D) same as C except showing RFP fluorescence/OD.

2.17 A). Because the integrase is activated immediately after dilution, the integrase effectively "records" the detachment of this RNP before the second guide is activated. The longer the integrase is allowed to act before the second guide inducer is added, the more time during the "detaching" state is recorded. The record in this case is the amount of RFP or GFP positive plasmid formed from non-fluorescent precursor (Figure 2.17 B).

Induction of the second guide RNA at t = 5+ hrs is essentially the same as never inducing the second guide RNA (Figure 2.17 C and D, comparing solid and dashed lines). This suggests that most of the integrase recording activity is concluded after 5 hours, at this level of integrase induction. It is possible that a lower level of integrase induction would allow recording for longer time. Induction of the second guide RNA at t = 0 creates a strong bias in GFP versus RFP production, but not quite as much as if there is no first guide RNA induced. Inducing guide 1 only (red lines, Figure 2.17 C and D) produces a steady effect that does not diminish if guide 1 induction is delayed. The same is not true for guide 2, which shows a steady increase of GFP and decrease of RFP as guide induction is delayed. These observations are consistant with guide 1 leakage. Since guide 1 is present at the beginning anyway, delaying guide 1 induction does not have a strong effect, whereas delaying guide 2 induction means that the low amount of guide 1 present at the beginning can exert its effect for a longer time. Another interesting observation is that inducing guide 2 first followed by guide 1 is the same (at time = 0) as inducing both guides followed by dilution into media where no guides are induced. This suggests that guide 2 remains active even after it is no longer induced. The same is not true of guide 1, as switching from guide 1 to guide 2 results in lower GFP and higher RFP level (effects created by the activity of guide 2) than going from both guides to none.

It is clear that guide residency is important, but slow guide switching will not severly impact the function of the event recorder. Even though a previously induced guide stays active after diluting into a different inducer, freshly induced guide still generally dominates this effect. The level of GFP produced after switching to guide 2 induction from nothing is the same as when we switch from guide 1 to guide 2, presumably as a result of guide 1 leakage. However, RFP production is not the same. We would expect that RFP and GFP production would be mirrored, since a plasmid can only become GFP or RFP, but a third option is for a plasmid to have both attP sites blocked, and it seems in the case of "1 then 2," the residency of guide 1 results in more plasmids with both sites blocked than the effects of leaky guide 1

expression as seen in the “0 then 2” cases.

The dCas9 repression system is effective when attB and attP are on the same plasmid, but in the full event recorder system we propose placing the attB site used for event recording on the genome, which would be present at significantly lower copy number than plasmids. In addition, two plasmids will be used that might have slightly different copy number. Integrase site repression must be complete enough that random differences in plasmid copy number do not create noise in the recordings. In the next section we investigate the ability of the integrase site repression mechanism in the context of the full event recorder system.

2.9 In Vivo Event Recorder Characterization

In order to record order and identity of two events using this system, we make use of two ink plasmids containing unique barcodes. The integration of an entire plasmid results in extension of the recording array by one barcode unit. Ideally the presence of external stimuli of some kind would result in more of one or the other ink plasmid becoming integrated into the recording array. We developed a construct that uses arabinose and Cin AHL to induce the transcription of guide RNAs that repress integration of either of the two different ink plasmids. In this section, plasmid Rec48 is used, which has g1 and g2 reversed from that pictured in Rec49 as seen in Figure 2.5. In Rec48, arabinose induces production of gRNA 1, that represses the integration of plasmid 2, that leads to more integration of plasmid 1, and vice versa with Cin AHL. Thus, adding arabinose produces gRNA 1 and leads to longer stretches of plasmid 1 relative to plasmid 2. Conversely, adding Cin AHL produces gRNA 2, and leads to more integration of plasmid 2.

To characterize event recorder activity, we grew cells in media containing arabinose or Cin AHL, and then subjected purified plasmids to Nanopore long read sequencing. Long reads are essential to understand the order and identity of barcodes in the recording array, as even one barcode and attachment site repeat is ~150 bp in size. We chose to utilize the plasmids themselves as the recording array in the following experiments, as this way we can get an unbiased measurement of how many plasmids integrated and how many remained whole. Plasmid-plasmid integrations are also capable of event recording, as when integrase is induced in the presence of two ink plasmids, the plasmids will recombine with each other and produce multimers with barcodes arranged in a particular order. Ink plasmids contain an attB site and an attP site which is partially overlapped with a guide RNA binding site. Thus, each

plasmid can accept any other plasmid integrating at its attB site, but can also itself be blocked from integrating through dCas9 binding to its attP site.

Plasmid-Plasmid Integrations Controlled by dCas9

Each ink plasmid is capable of integrating with another ink plasmid in one of two ways. First, the attP site of the first plasmid can react with the attB site of the second plasmid. Second, the attB site of the first plasmid can react with the attP site of the second plasmid. The products of these reactions are distinguishable in that the unique barcodes present on the newly created multimer plasmid can be arranged in a different order, depending on which of the two reactions occurred. A schematic of plasmid-plasmid reactions is presented in Figure 2.19 A. When two ink plasmids are thus recombined, the plasmid whose attP site was unblocked appears to the right. Thus, if plasmid 2's attP site was unblocked, and integrated with plasmid 1's attB site, we would produce plasmid 12. Otherwise, the converse reaction (plasmid 1's attP site and plasmid 2's attB site) would yield plasmid 21.

A proof of concept experiment was performed to evaluate the viability of using dCas9-based integrase attachment site blocking to affect the order of barcodes present in ink plasmid multimers. Cells were transformed with integrase recording plasmid Rec48 (g1 and g2 swapped relative to Rec49 as pictured in Figure 2.5 A) Ink70, and Ink71. Cells were grown in media containing some mixture of Sodium Salicylate (integrase inducer) aTc (dCas9 inducer), cin AHL (g1 inducer) and arabinose (g2 inducer) (Figure 2.19). Cells were grown for 30 hours until reaching stationary phase. After culture growth, DNA was extracted from each well using a standard miniprep kit, and subjected to Nanopore rapid barcoding library prep (RBK004 kit). Reads were aligned to expected genomes constructed from expected DNA sequences after integration of up to four barcodes.

Reads obtained for each barcode ranged from ~10,000 to 500 at the lowest. Of these, usually more than 25% counted as plasmids with no integration, and a steadily decreasing amount up to 5 integrations (6 barcodes) that was aligned (Figure 2.18). It is interesting to note that induction of integrase only leads to the least integrations of all the conditions. This is a consistent trend we have seen among many experiments (see Figure 2.13). This effect is opposite to the commonly seen "resource limit" effect that is obtained when a large protein such as dCas9 consumes all cellular resources, and all other proteins (such as integrase) are seen to be expressed at a lower level. Such an increase in integrase production can be caused by transcrip-

tional interference, since all the genes in the integrase control circuit are present in close proximity on the same plasmid. Active promoters have been shown to up-regulate upstream promoters [83], and so perhaps this is the reason why activating dCas9 production up-regulates the upstream integrase production. For this reason also it would seem that comparing integrase only conditions with those in which integrase and dCas9 are induced would have different integrase production levels and therefore confound the results. It may, however, be possible to decrease the integrase induction level in conditions where dCas9 is activated to the point where the integrase activity is the same as the condition where only integrase is induced. Alternatively, a restructuring of the construct into an "opposed" configuration where the two constructs point at each other could also work [83].

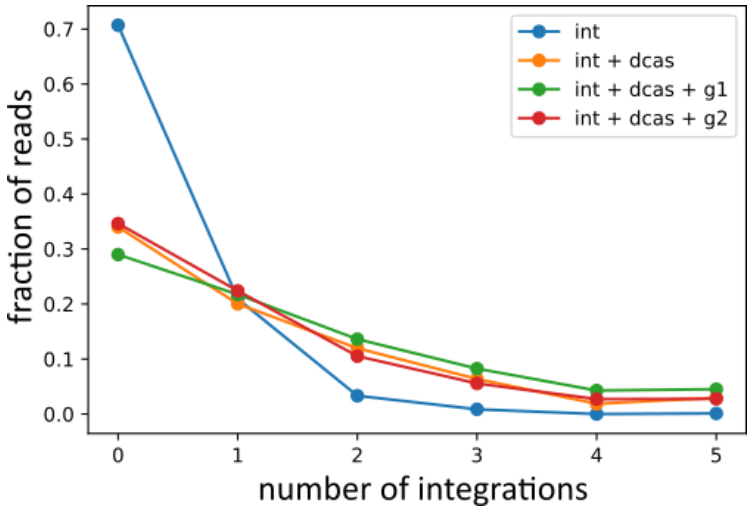


Figure 2.18: Fractional number of reads indicative of integration numbers. Blue line shows integrase induced only; orange line is integrase + dCas9 induced; green line is integrase, dCas9, and g1 induced; and red line is integrase, dCas9, and g2 induced. Zero integrations represents unreacted plasmid. One integration means two barcodes present on a plasmid, since one plasmid integrated once into another one, and so on. Four cell culture conditions are listed, same as in Figure 2.19. Longer integrations occur when dCas9 is induced, indicating transcriptional context effects between dCas9 and integrase.

When neither of the two guide RNA promoters are induced, the system behaves as though guide 2 is induced, as indicated by the alignment counts shown in Figure 2.19. The simplest explanation for this behavior is that the arabinose promoter is leaky, and the leaky level of g2 behaves the same as full induction g2, in the absence of a competing guide RNA. Once g1 is induced, however, integration of the plasmid containing barcode 2 is strongly inhibited, as seen by the lack of reads

containing "barcode 2" sequences to the right of "barcode 1" sequences as shown in Figure 2.19. Likewise, there does seem to be a difference between no guide and g2 induction (despite the leak) as seen in read counts of 122, which is almost non-existent in every condition except for the one where g2 is induced. Also, the amount of 12 genome content is highest in the g2 induction condition, as expected if barcode 1 integration is blocked by the presence of g2.

Barcode 1 seemed disproportionately more likely to become integrated with itself to produce multimer polymers containing many barcode 1 units. In addition, the amount of integrated barcode 2 observed seemed significantly lower than barcode 1. It seems intuitive that this problem stems from the difference in ink plasmid copy number. We attempted to create a low copy number version of the cole1 origin of replication to more closely match the copy number of P15a (see Figure 2.10), and indeed the origin from the construct named "ULC1" was used throughout ink plasmid experiments. However, it is clear that even this reduced copy number plasmid is still present at a higher copy number than the other ink plasmid. The question we must ask next, then, is how critical is this copy number difference?

Effect of Copy Number On Two-Ink Plasmid Event Recorder

After confirming that the integrase site repression works when the sites are on different ink plasmids, we sought to understand the limitations of the system when it comes to plasmid copy number. As mentioned above, we saw that integration of barcode 1 was disproportionately higher than barcode 2, and attributed this to a different copy number between the two plasmids containing these barcodes. To understand this further, we constructed a simulation to explore the effects of copy number on the event recorder's ability to effectively record events. ColE1 plasmid copy number has a natural variability in cells up to ±18% of the average [80] and our modified cole1 origin likely does not maintain the same copy number as the P15A origin used in plasmid Ink70 (the one containing barcode 2, see Figure 2.5). Thus, it would be important to know how different these copy numbers are allowed to be before the event recorder becomes dysfunctional.

The model used to perform these simulations is as described above in Section 2.2. To investigate the effect of plasmid copy number on event recorder behavior we looked at the relative amount of four-plasmid species that can be reached through the fewest number of integrations, and undergo interesting changes in concentration when inducer conditions are changed. We settled on tracking the concentration

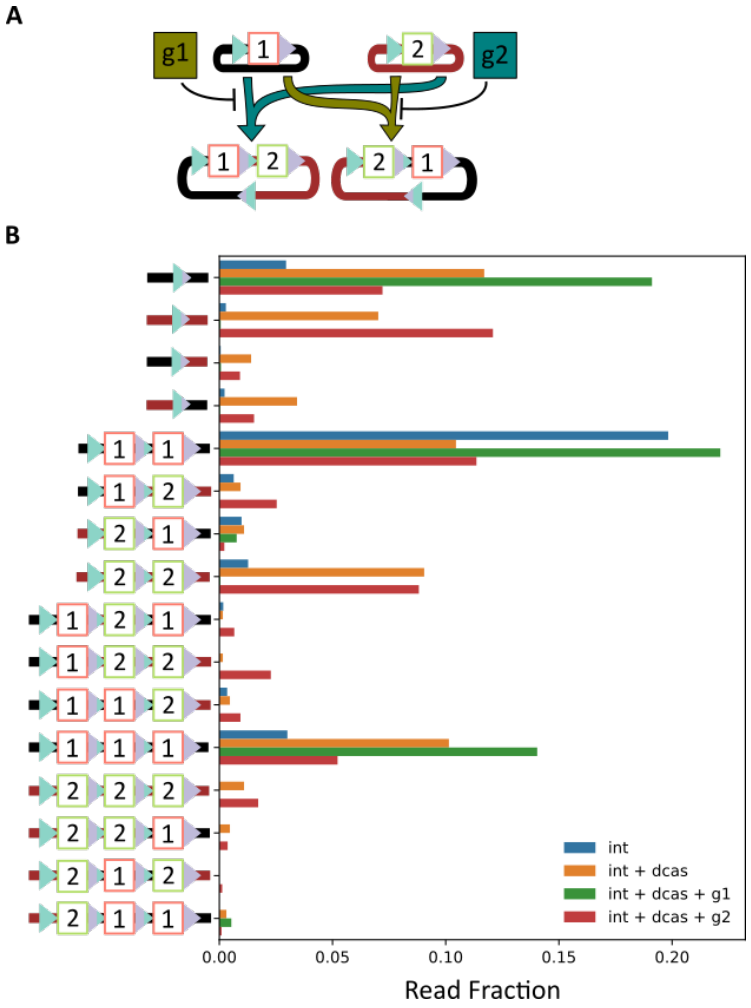


Figure 2.19: Plasmid-plasmid integrations as measured by Nanopore sequencing. (A) Schematic of plasmid-plasmid integrations. Plasmids depicted as ovals. Blue triangle represents attP site, mauve triangle is attB site with a numbered square representing the plasmid-specific barcode. Filled arrows represent integration reactions, flat-headed arrows represent dCas9 repression of integration. G1 represses plasmid 2 integrating into plasmid 1's attB site, and therefore when g1 is present, plasmids with the barcode order "2,1" should be over-represented. Likewise, presence of g2 leads to plasmids having barcode order "1,2." B) Quantification of Nanopore sequencing results. Raw reads were aligned to short genomes containing sequences as represented in the small schematics to the left of the bar graph. Colored lines represent DNA from plasmid 1 or plasmid 2 (black or red, respectively). Multi-colored triangles represent recombined sites. With triangle point to the left: cyan left with mauve right is attR, mauve left with cyan right is attL. Cells were grown in microplate wells containing inducers for integrase production only, integrase and dCas9, and integrase, dCas9, and one of the two guide RNAs. Bars represent the read fraction of total that were counted to align to the specified genomes.

changes of plasmids containing barcode 1 followed by barcode 2, “12,” barcode 2 followed by barcode 1 “21” as well as “121” and “212.” Simulation was run for 200 time units with the first event active for the first 60 time units followed by the second event for the rest of the time, or the same event the entire time (Figure 2.20). The ideal event recorder configuration involves the same copy number of each plasmid, and in that case we see that the expected multimer is the most prevalent at the endpoint.

The plasmid 21 is seen as the most prevalent when g1 is present, which is expected because that would indicate that the attP site on plasmid 2 is blocked, allowing plasmid 1 to integrate into the attB site in plasmid 2 (Figure 2.20 A). As the copy number ratio is increased, fewer integrations are made. At first this may seem counter-intuitive but it is because the actual amount of plasmids present in the simulation is also increasing. That means that there are more attachment sites, but the number of integrases is not increasing proportionately. Thus, with a greater number of integrase sites, we would expect to see that there is a decreased chance that compatible sites (attP of plasmid 1 and attB of plasmid 2 for example) would be properly occupied by integrase. Likewise, you would expect that the reaction that forms plasmid 21 performs most optimally when plasmid 2 and plasmid 1 are present at equal amounts.

A similar trend is observed when g2 is present (Figure 2.20 B), although now we see an increase in plasmid 121 product when the copy number ratio is increased. This also makes sense as the increase in plasmid 1 means that plasmid 1 is less effectively repressed by the presence of g2, and therefore can integrate into the attB site on newly created plasmid 12 to produce 121. A similar reaction can occur to produce 112, and we see a similar increase in plasmid 112 (not shown). While the absolute amounts of plasmids change when the copy number ratio is increased, the relative rank ordering of the plasmids stays the same when a single guide RNA stimulus is presented. Even at a copy number ratio of 4, plasmid 21 is still the most prevalent species when g1 is applied, and likewise for plasmid 12 in the g2 case. The same is not true when the guide RNA condition is changed.

The beauty of event recording is that the record reflects not just the current inducer but also the history of inducers that were seen before. Thus, we would expect that the final amount of the four multimer plasmids would be somehow distinguishable between different histories. For that reason we have created a simulation where g1 is swapped for g2 after 60 time units, and likewise for g2 being swapped for g1 (Figure

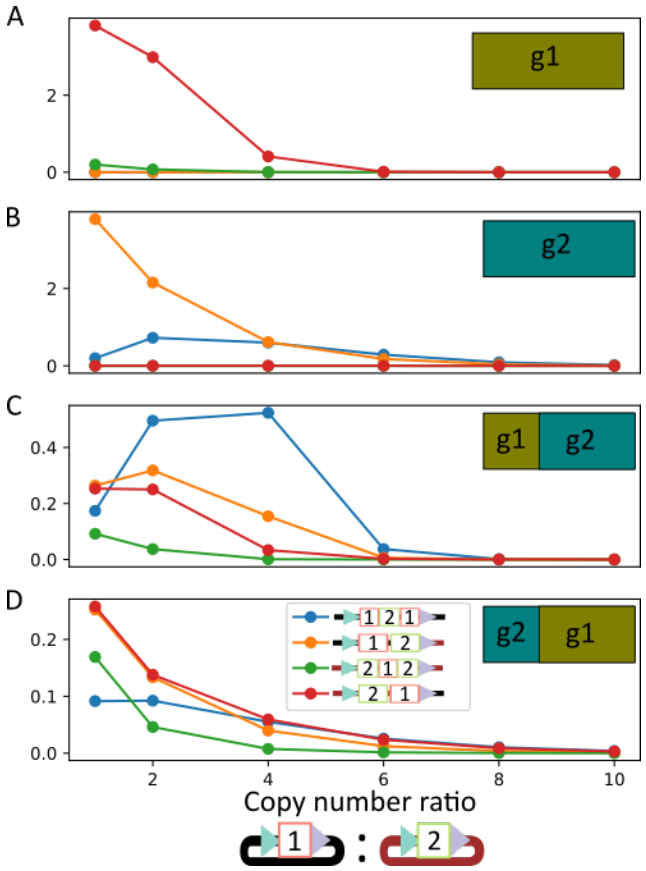


Figure 2.20: Simulated event recorder as a function of copy number differences between ink plasmids. (A) Final amounts of four interesting plasmid species at the end of a simulation ran for 200 time units, with only guide 1 present. Horizontal axis shows plasmid 2 to plasmid 1 ratio, and Y axis shows molecular units. Copy number is increased by increasing the number of plasmid 1 at the start of the simulation while keeping the number of plasmid 2 constant at 20. That means that for copy number ratio = 10, there are 220 plasmids at the start of the simulation, whereas at copy number ratio = 1, there are only 40. (B) Final values of interesting plasmid species at the end of 200 time units (endpoints) with only guide 2 present. (C) Endpoints where the guide present is swapped at t = 60 time units from g1 to g2. (D) Endpoints where the guide is swapped from g2 to g1.

2.20 C and D). Relative amounts of the four tracked plasmids are significantly different in these "event" conditions versus the conditions where the same inducer was present the entire time. It is clear, however, that different copy numbers result in vastly different multimer plasmid concentrations. For example, at a copy number ratio of 2 and above, the plasmid 121 becomes the most prevalent plasmid in the case of the event history where g2 follows g1 (Figure 2.20 C). Again it is fairly straight forward to understand why this happens: as we increase the amount of

plasmid 1 monomers in solution by increasing the copy number ratio, the repression of the plasmid 1 attP site becomes weaker, and since that is what prevents buildup of plasmid 121 (addition of the terminal "1" requires a free attP site on plasmid 1), we are essentially releasing inhibition for formation of 121 in that condition. In the condition where the final inducer is g1 though, as in Figure 2.20 D, we see that releasing the inhibition on plasmid 1 does not make as much of a drastic change. This can be explained by the fact that plasmid 1 is inhibited for only the beginning of the event series, and what we see is mostly the effect of increasing copy number during the presence of g1, as in Figure 2.20 A which does not seem to have much effect. Therefore, we can conclude that increasing the copy number of a specific ink plasmid only affects the events recorded which require that plasmid to be blocked.

Response of Event Recorder to Different Event Duration

While it is difficult to adjust the copy number of the ink plasmids in vivo, it is important to keep in mind the effect that copy number can have on the functionality of the event recorder, and how to interpret the results of in vivo data taking into account the fact that the relative copy number of the plasmids could be different than expected. For example, with increased copy number, integrations happen less often, and the higher copy plasmid is harder to repress. Another important metric to characterize is the response of the event recorder to different event durations. One can imagine there is a minimum event that can be detected, and that such an event can be understood in terms of induction amount and induction duration.

We sought to investigate minimal induction duration, since it seemed more thematic to an event recorder to measure induction in the time rather than the concentration domain. To accomplish this measurement we grew cultures of cells in inducer-containing media and then diluted a small aliquot of cell-containing media into another well containing a different set of inducers after a certain amount of time. Again we tracked the concentration of plasmids 12, 21, 121, and 212. Plasmids up to length 6 (5 integrations) could be tracked using our Nanopore sequencing followed by putative genome alignment strategy (Figure 2.18), but for simplicity only these four plasmids are presented here.

Two competing effects can confound results in these experiments. First, either g1 or g2 can be leaky. This means that during a time when we expect g1 to be induced, g2 also acts as though it is partially induced, or vice versa. It is clear that g2 is leaky from the fact that cultures grown with only integrase and dCas9 induced appear

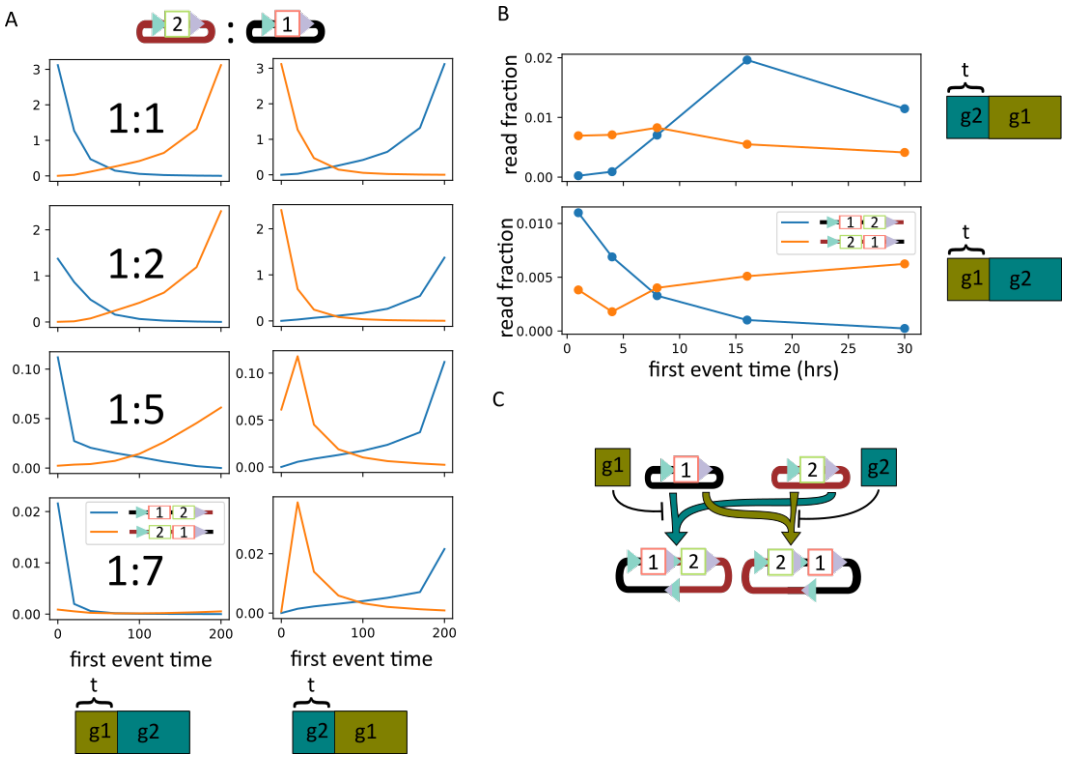


Figure 2.21: Event recorder final state after experiencing a sequence of events. (A) Simulated final state of event recorder plasmid concentrations after a sequence of events. Either g1 or g2 is present first which switches to the other inducer after a certain time. Left column shows g1 followed by g2, right column shows g2 followed by g1. Blue line represents the count of plasmids having the "12" barcode arrangement, and orange line represents the count of plasmids having the "21" barcode arrangement. From top to bottom, plasmid copy number is skewed towards the barcode 1 containing plasmid by increasing the initial amount of plasmid 1, while keeping plasmid 2 constant. (B) Event recorder experiment in live cells. Rec48 integrase controller plasmid is used in cells grown in M9CA glycerol media. Cells are outgrown in media without inducer and then transferred to media containing either arabinose or cin AHL. Then, after some amount of time as indicated on the graph, cells from this new well are diluted into another well containing the other inducer, where they are left overnight. Inducer for g2 is arabinose, and inducer for g1 is cin AHL. Top graph has g2 first, g1 second. Bottom graph has g1 first, g2 second. (C) schematic showing the integration steps necessary to create these plasmids. AttB site on plasmid 1 can react with attP site on plasmid 2 to produce 12, and attB site on plasmid 2 can react with attP site on plasmid 1 to produce 21. Thick arrows indicate the integration activity, and are colored depending on which path is preferred if the cognate guide RNA is active.

similar to when g2 is also induced (Figure 2.19). A second form of confusion can come from the fact that one of the two ink plasmids exists at a higher copy number,

and therefore is more likely to integrate and more difficult to repress. We can clearly see that this is the case with plasmid 1, as there is a huge quantity of 11 and 111 plasmids observed compared to very few 22 or 222 (Figure 2.19). As these effects are opposed, leaky g2 will tend to repress the integration of plasmid 1, and we can be satisfied that we have done as much as we can to mitigate these confounding effects with the constructs we have.

When the simulated event recorder is presented with an event sequence of g1 followed by g2, there exists a critical point where the relative amount of plasmid 12 and 21 cross, and then the other of the two becomes the dominant species. This point is a sort of midpoint of the event recorder, where the amount of induction before and after switching to the other guide RNA is the same. This is because we assume that while g1 is active, production of plasmid 21 dominates, but the opposite occurs when g2 is active. When the plasmids are present at equal copy number, this point occurs before the half way mark of the simulation (Figure 2.21A), and exists roughly in the middle of the simulated time period despite a skewed copy number ratio up to 1:7. The same cannot be said when looking at plasmid 121 and 212, which also exhibit the same sort of crossing point phenomenon (Figure 2.22 A). However, if the copy number is skewed past 1:5 this crossing point disappears. Thus, when we consider the data obtained from cells (Figure 2.21 B and Figure 2.22 B), it is clear that the cells are showing a very similar behavior to the simulation, with a crossing point for plasmids 12 and 21 occurring at 8 hours of induction, but no visible crossing point for 121 and 212 presumably because the copy number imbalance heavily favors the formation of 121 versus 212.

It is interesting to note that there is a clear and definite pathway for plasmid 12 to form only when g2 is present, and likewise that plasmid 21 forms only when g1 is present (Figure 2.21 C). However, plasmid 121 can form if the cells see a sequence of g2 followed by g1, but also with a sequence of g1 followed by g2 (Figure 2.22 C). If we intend to create a system which can effectively record chronological information, it is important to consider the implications. This means that plasmid 121 can form under conditions which could indicate either the chronological sequence g1, g2 or g2, g1. Thus, is the presence of plasmid 121 not really recording any useful information? The logical conclusion is that, by itself, the presence of this plasmid is not definitive proof that any event order happened, but it is the context of this plasmid along with the relative amounts of other multimers that must inform what event order history led to its creation. If plasmid 121 is observed in the context of

a large amount of plasmid 12, we can infer that it is likely that it formed from the addition of a single plasmid 1 to the attB site of plasmid 12, and thus, that the true sequence was g1, g2. On the other hand, if plasmid 121 is seen with a large quantity of plasmid 21, we might consider the opposite. Indeed, plasmid 121 and 212 do seem to be responsive to the order and duration of guide RNA presence (Figure 2.22 A).

Distinguishing Different Events

An event record is only as good as the ability to distinguish which events lead to the production of which record. In the case of the plasmid-plasmid event recorder, the record consists of plasmids that have integrations in different order, but also the ensemble of all such plasmids present in a population of cells. Integrase site blocking is not perfect, which means that sometimes plasmids that should be blocked still get integrated. In addition to this source of noise, records formed on different plasmids can integrate together, meaning the order of new barcode addition is not always logical.

Despite these confounding factors, we can still try to understand if the timing of an inducer pulse produces a noticably different record. To address this question we simulated a pulse of inducer in the middle of a constant amount of the other inducer. This constitutes a "pulse" event, distinguished from the "step" event that was simulated in Figure 2.21 and 2.22. In the pulse simulation, time before the pulse is plotted on the X axis and amounts of different plasmid states are plotted on the Y axis (Figure 2.23). $T = 0$ on these graphs indicates a step induction going from g1 to g2, whereas the other time points have a delay before addition of g1. In general differences in endpoint concentrations when the delay is added are not great, indicating that the event recorder is not particularly good at distinguishing this type of event from the step event as portrayed previously. In addition, the dynamic range of these simulations is also seen as essentially reaching a plateau after $t = 50$. This plateau indicates that $t = 50$ is the limit to which a pulse delay can be detected. This is in contrast to the step induction, which seemed to have a dynamic range stretching throughout the simulated time all the way to $t = 200$. From this we can conclude that the plasmid-plasmid event recorder would be much less capable of recording pulse events than step events. As plasmid copy number is increased, the event recorder also becomes less effective as we see that the endpoint number of all multimer plasmids decreases.

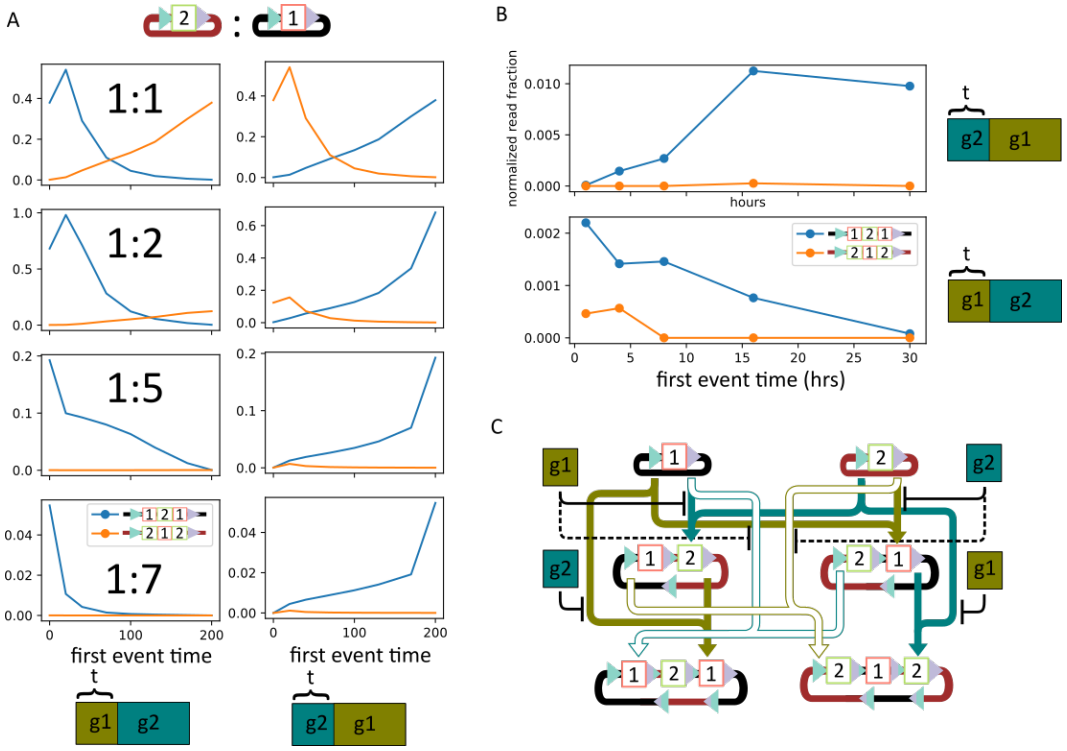


Figure 2.22: Longer plasmid integration sequence after event recorder experiences sequence of events. (A) Simulated final state of event recorder plasmid concentrations after a sequence of events. Either g1 or g2 is present first which switches to the other inducer after a certain time. Left column shows g1 followed by g2, right column shows g2 followed by g1. Blue line represents the count of plasmids having the "121" barcode arrangement, and orange line represents the count of plasmids having the "212" barcode arrangement. From top to bottom, plasmid copy number is skewed toward the barcode 1 containing plasmid by increasing the initial amount of plasmid 1, while keeping plasmid 2 constant. (B) Event recorder experiment in live cells, same as Figure 2.21 (C) Schematic showing the integration steps necessary to create these plasmids. AttB site on plasmid 1 can react with attP site on plasmid 2 to produce 12, and attB site on plasmid 2 can react with attP site on plasmid 1 to produce 21. After 21 and 12 are produced, depending on which side the integration occurs will determine whether 212 or 121 is produced next. If plasmid 1 is not blocked by the presence of g2, then 12 will combine with plasmid 1 to produce plasmid 121 (solid brown arrow). However, plasmid 21 can also react with the attB site on plasmid 1 to produce plasmid 121, if it is not blocked by the presence of g1 (open teal arrow). Likewise, plasmid 212 can be created from the interaction of the attB site on plasmid 21 and the attP site on plasmid 2 (right side solid teal arrow) if g1 is not present, but also from plasmid 12's attP site reacting with plasmid 2's attB site in the absence of g2 (open brown arrow).

We discuss the idea that different multimer states can have different trajectories over time, indicating that the event recorder is producing a specific record that is

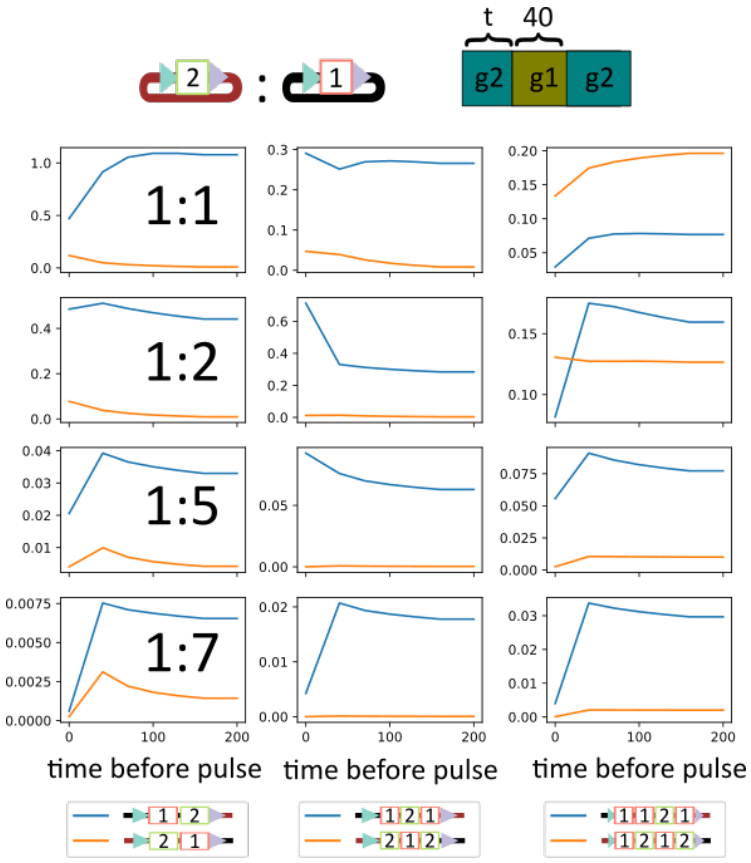


Figure 2.23: Simulated event recorder response to inducer pulse. Endpoint amount of multimer plasmid following different inducer pulses. Total simulation time was 200 time units and inducer pulse was 40 units long with a varying time before the start of the pulse. At $t = 0$, the inducer trajectory is the same as $t = 40$ in Figure 2.21 and 2.22. Any deviation (up or down) from the leftmost value on each plot indicates a different plasmid state that could be used to distinguish a pulse event from a step event. Subset of plasmid multimer types were chosen to be plotted. Plasmid 12/21 and plasmid 121/212 as before, as well as plasmid 1121/1212 were chosen. Rows from top to bottom have increasing plasmid 1 amount, leading to the depicted ratios of ink plasmids.

indicative of a particular past series of events. However, it is very difficult to know what such a record would look like, because of all the confounding factors we have discussed previously. For this reason, we have opted to compare multimer states which are fairly small (consist of few barcodes) to see if any difference in final multimer state can be perceived at all. In this sense the work presented here is still at the "proof of concept" stage.

2.10 Conclusion

A novel event recorder with an unlimited theoretical capacity for information storage has been developed in this work. Serine integrases, specifically, Bxb1 integrase, are well studied recombinases that were chosen to be used in the event recorder developed in this work because of their high activity and well-defined recombination reaction pathway.

As a result of specific integrase recombination between attB and attP sites, several challenges had to be overcome. First, in order to allow sequential integration, attB sites present on the recording locus must be regenerated when they are destroyed by recombination. Thus, placing attB and attP on integrated plasmids, ("ink" plasmids) was essential. Placing attB and attP closer than 100 bp away from each other greatly diminishes intramolecular deletion reactions, which is essential to preventing a plasmid containing both attB and attP from immediately undergoing deletion reactions and not being able to participate in integration reactions. Second, we attempted to structure a plasmid such that after recombination, the origin of replication was no longer functional, but this resulted in multimer plasmids having extremely active origins of replication so this idea was abandoned. Third, we needed two orthogonal plasmid origins with equal, fairly low copy number to allow event recording with two different "ink" plasmids. We developed a lower copy ColE1 origin by using error-prone PCR. Sequential integration of a single plasmid containing attB and attP was confirmed in vivo.

We also developed a dCas9-based competitive binding system for controlling integrase activity without changing attachment site sequence. Overlapping guide RNA sequence with attP attachment site led to an effective "integrase blocking" activity which was active even if 11 base pairs of guide RNA sequence existed outside the attachment site. We developed 8 functional orthogonal guide RNA sequences which could be used for repressing integrase activity with little cross reactivity. We also tested using two different guide RNAs for repressing either of two integrase sites in the same cell to confirm inducible dCas9-based repression in vivo.

Nanopore long-read sequencing was used to sequence multimer plasmids, which were found to also be capable of containing event records, as determined by simulation. We encountered several confounding factors when using event records to back out the sequence of events which led to producing those records. First, when looking at genome records, the existence of plasmid multimers means that a certain genome record (for example, barcode 1 followed by barcode 2) can be created in the

instance of either event 1 happening first followed by event 2, or event 2 followed by event 1 at lower frequency. Thus in order to have a full picture of what event sequence resulted in what record, we must evaluate the ensemble of records present in the entire population. Comparing the incidence of barcode 1 followed by barcode 2 versus barcode 2 followed by barcode 1 yields a pattern that varies predictably with a changing event sequence. Likewise for barcode sequences 121 and 212, although in general we detected very low levels of barcode 212, consistent with the copy number of barcode 1 still being significantly higher than barcode 2, despite the error prone PCR low copy ColE1 origin.

2.11 Materials and Methods

Simulations

Chemical reaction networks were prepared using BioCRNpyler [58] and simulated using BioSCRAPE [74]. Parameters used for chemical reaction network rates are as presented in Table 2.2. Generic integrase reactions used are as presented in Table 2.1. Python code used to generate the genetic parts and CRNs for the simulations is reproduced in Appendix A.

Cell Strains

Cells used were DH5alpha Z1 [44]. Genome site constructs were made by Gibson assembly into SpeI-KpnI digested pOSIP KH or pOSIP KO or KH [70], followed by genome integration and pE-FLP excision protocol as described.

Constructs

Construct design was done using Geneious (Biomatters, Auckland, New Zealand) and using a custom-made Gibson/Golden Gate design program written in Python [1]. Bxb1 integrase sequence was amplified from the Dual-recombinase-controller vector, which was provided by Drew Endy (Addgene plasmid # 44456) [9]. Deactivated Cas9 was amplified from pAN-PTet-dCas9, which was a gift from Christopher Voigt (Addgene plasmid # 62244) [51]. Recording circuit plasmids were assembled with Golden Gate followed by Gibson assembly as described [30]. Parts for inducible promoters were modified from plasmids obtained from Christopher Voigt [46]. pSal, NahR, pTet, pCin, CinR, and pBAD were amplified from source material with primers to add BsaI cut sites using compatible sequences [30] before being used for assembly. Full construct sequences can be found in the supplemental documents of Shur and Murray 2020[66].

Ribosome binding site for Bxb1 integrase was designed using random screening. A degenerate oligo was used to create a library of clones containing different RBS strengths (see Table B.1. Parenthesis in sequence indicate BsaI cut site locations used for Golden Gate cloning [33]. Correct clone was chosen by streaking cells containing random clones on LB agar plates containing or not containing inducer, and observing GFP activity of a plasmid containing a flippable promoter driving GFP expression (pVHct1) generously provided by Victoria Hsiao.

Bacteria Culturing

For static inducer conditions, cells were grown in M9CA minimal media (Teknova) overnight, then diluted 1:20 into fresh M9CA minimal media containing inducers. For event timing experiments where inducers were changed, cells from an overnight culture in M9CA minimal media were diluted 1:20 into fresh M9CA minimal media. When the cells reached linear growth phase at 0.5 OD, they were diluted again 1:10 into fresh media containing inducers. After a set amount of time, aliquots from these inducer-containing cultures were diluted 1:100 into media containing a different set of inducers. Total culture volume used was either 200 μL in round-well cylindrical glass-bottom plates (Brooks Automation) or 400 μL in square-well glass-bottom plates. For static inducer experiments, inducers were added 100x concentrated to each well followed by media manually, or using an Echo 525 liquid handler (Labcyte). For changing inducer experiments, inducers were added into media before hand and initial outgrowth to linear phase was monitored using an automated plate reader (Biotek). Once OD 0.5 was reached, cells were diluted appropriately using a Starlet liquid handler (Hamilton). Liquid handler protocol is described in Appendix C.

Inducers used were 1–100 nM Anhydrotetracycline (Sigma), 0.2 uM Sodium Salicylate (Sigma), 0.2% Arabinose (Teknova), 1 mM Isopropyl-beta-D-thiogalactoside (Sigma), or 1-5uM 3OHC14-HSL (Sigma 51481). Inducer concentrations were mixed to 100x concentration either in water or in DMSO if the set of inducers included 3OHC14-HSL. Then, 2 or 4 μL was added to each well of a 96-well plate, with media up to the total volume, 200 or 400 μL respectively, added on top. For automated inducer dispensation using Echo 525, inducers were made to 500x concentration, then placed in an Echo qualified 96-well source plate. After droplets of inducer were deposited in each well of a 96-well plate containing round or square wells, the appropriate volume of M9CA media containing diluted cells was added on top.

Extract

For experiments performed using extract, TX-TL extract was prepared using a French press as described previously [73]. For 10-μL reactions, pOR1OR2 (from Addgene plasmid # 45789[3])-containing integrase and/or dCas9 linear PCR amplicons were used at a concentration of 5 nM. Integrase-site containing plasmids were used at a concentration of 1 nM for attB or attP containing constructs. Reactions were allowed to incubate in a plate reader (Biotek) for 20 hours and then either subjected to qPCR using primers P1F (TCTTGCTCAGGCGCAATCA), P2F (CGATGCGCCAGAGTTGTTTC) and BR (CAACGCTACCTTTGCCATGT). Pairs used were P1F BR and P2F BR, to check if Site 1 or Site 2 had integrated, respectively (see Figure 2.11 and 2.12). In other extract experiments, results of extract reaction were subjected to Qiaquick PCR purification kit (Qiagen) and then transformed into JM109 competent cells (Agilent). The next day, GFP or RFP positive colonies were counted using a fluorescent stereomicroscope (Olympus MVX10).

Sequencing

Event recorder sequencing was performed using the MinION (Oxford Nanopore, Oxford). After cells had grown under time varying inducer conditions through the use of the Hamilton Starlet liquid handling robot in conjunction with the Biotek plate reader, small aliquots of culture were extracted and used as a template for a PCR reaction using UINTF (TGTGGCCTCTGATTGGTGTC) and U21R (TCCGTCTACGAACTCCCAGC) or U22R (GCTTGGATTCTGCGTTTGTT). Alternatively CARBR (GGATCTAGGTGAAGATCCTTTTTGA) or CHLORR (TATTCTGCCTCCCAGAGCCT) was used. These primers were augmented with a 5' extension containing Nanopore barcode sequences 1–15[50]. PCR was performed using Q5 DNA polymerase (NEB) as described using 25–32 cycles and an extension time of one minute. After PCR, the amplicons were cleaned and ligated by following the LSK109 protocol as described [49]. Sequencing data was acquired and processed using the MinKnow software and basecalled using guppy 5.0.7 with the "high accuracy" mode. After sequencing and barcode splitting, reads were aligned to genomes generated from predicted recombined sequences using minimap2[39]. Primary alignments were then counted for each genome and divided by the total number of reads binned into that barcode. Predicted genomes were generated only up to 6 barcode units.

Chapter 3

INTEGRASE-RELATED SIMULATION AUTOMATION IN BIOCRNPYLER

3.1 BioCRNpyler Overview

Chemical reaction networks (CRNs) are an established way to represent bio-chemical reactions for the purposes of simulation. Many existing bio-chemical modeling packages allow simulation and parameter estimation using chemical reaction networks [4]. One complication with chemical reaction networks is that in order to accurately replicate biochemical behaviors, a CRN must contain a large number of mass action reactions, or a smaller number of reactions with complicated rate laws. Complicated rate laws often require fine-tuning many parameters and often are not compatible with standard ODE solvers. Realistic behavior can also be achieved using a large number of simple mass action reactions, but constructing large chemical reaction networks by hand is tedious and error prone. The role of BioCRNpyler[58] is to automate large network creation by the application of several assumptions and rules common to biological transcriptional circuits.

The motivation for creating these software tools is to automate the simulation of integrase reactions. One example of a common synthetic biology relevant CRN made by BioCRNpyler is a gene which produces RNA and eventually fluorescent protein (see Figure 3.1A and Figure 3.3). Experimentally, the fluorescent protein (and not the RNA or DNA) can be measured easily, and so it makes sense to simulate the dynamics of such a protein production system in order to understand something about the properties of the components that cannot be measured. A promoter or ribosome binding site (RBS) can be characterized by measuring the level of fluorescent protein that is made, and in various ways comparing the experimental values to simulated protein from such a CRN. To create a CRN, using BioCRNpyler, that represents a single protein production gene, the promoter and RBS (and relevant rate constants) must be specified. BioCRNpyler allows different mechanisms for transcription or translation, which allows simulation at different levels of complexity, with or without degradation or dilution.

In the process of simulating synthetic biology-relevant regulation networks, one might start with a relatively simple gene such as the one discussed above, and

progress to a more complicated gene network where one protein product represses another gene (Figure 3.1B), and interesting dynamics can thus be created. The basic mechanism by which complicated regulation dynamics can be realized is that the amount of a certain species, such as polymerase-bound DNA, is changing over time, and that leads to changes in other species such as RNA and eventually protein. In many circuits, the composition of the DNA genes or the concentration of them does not really change throughout the course of the simulation. Hard-coding the identity of genes (gene A produces RNA A which produces protein A) works well for many synthetic biology relevant networks where DNA identity and concentration do not change.

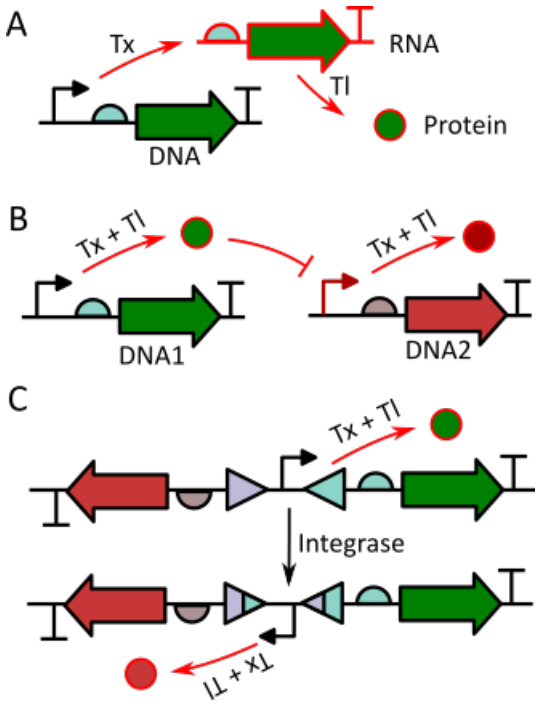


Figure 3.1: Increasing complexity of simulation reactions. (A) A simple gene (DNA) that produces an RNA using transcription (Tx) and then that RNA produces a protein using translation (Tl). A promoter is shown as a bent arrow, an RBS as a half circle, a protein-coding gene as a thick arrow, and a terminator as a "T." (B) A gene regulatory network consisting of one gene whose protein product represses the promoter of another gene. (C) An integrase circuit, which has a protein leading to the production of the green gene product, but then the promoter is "flipped" to point towards the red product through the action of the integrase.

Integrases can dynamically alter the composition of a gene by enacting DNA recombination (Figure 3.1C). A promoter which would produce an RNA that contains a certain RBS might be recombined so that it is producing a different RNA containing

a different RBS. For this reason, it is not enough to specify which promoter makes which RNA, but also to specify which RNA would be produced after the recombination event. Thus, the user would have to specify all possible DNA configurations by hand, which could lead to errors. The aim of the work presented here is to facilitate the automated population of promoter-RNA-protein relationships given linear lists of DNA parts, and automated enumeration of all the possible linear DNA part lists that can be reached by successive or alternative integrase reactions.

3.2 BioCRNpyler Structure

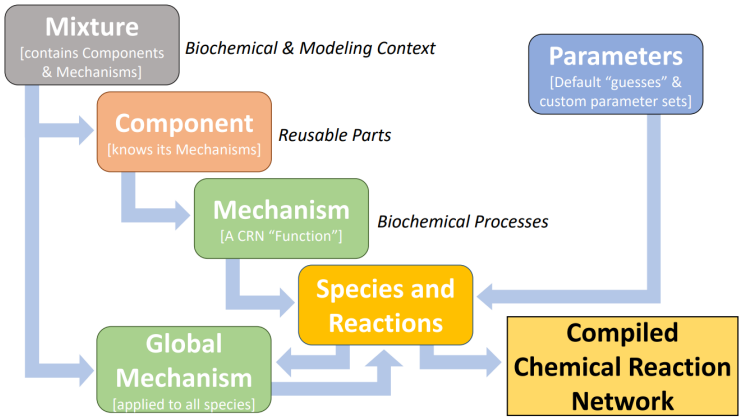


Figure 3.2: The hierarchical organization of Python classes in the BioCRNpyler. Arrows represent direction of compilation: from high-level design specifications (`Components`) in a modeling context (`Mixtures`) and biochemical processes (`Mechanisms`) to a CRN representation.

In BioCRNpyler, a hierarchy (Figure 3.2) of Python classes is used to organize different elements of an abstracted CRN. At the lowest level, `Species` and `Reactions` are used to represent nodes and transitions between nodes in the CRN. `Mechanisms` are a higher level construction which contains the logic for assembling `Species` and `Reactions`. For example, a transcription mechanism can be made so that it will create a `Reaction` where RNA polymerase binds to DNA, and another `Reaction` where RNA polymerase unbinds from DNA together with a newly made RNA. Some `Species` such as RNA polymerase are general and used in most transcription mechanisms, but specific species such as DNA and RNA need to be specified externally. The logic to assemble together `Species`, `Reactions`, and `Mechanisms` is contained within `Components`. `Components` such as a promoter can be used to specify that a certain DNA can create a certain RNA. Then, during compilation, the proper transcription `Mechanism` is combined with the proper DNA and RNA

`Species` to produce a correct CRN. In certain cases even higher level classes are needed than `Components`. Multiple `Components` and general mechanisms such as transcription and translation are in turn contained within a `Mixture`.

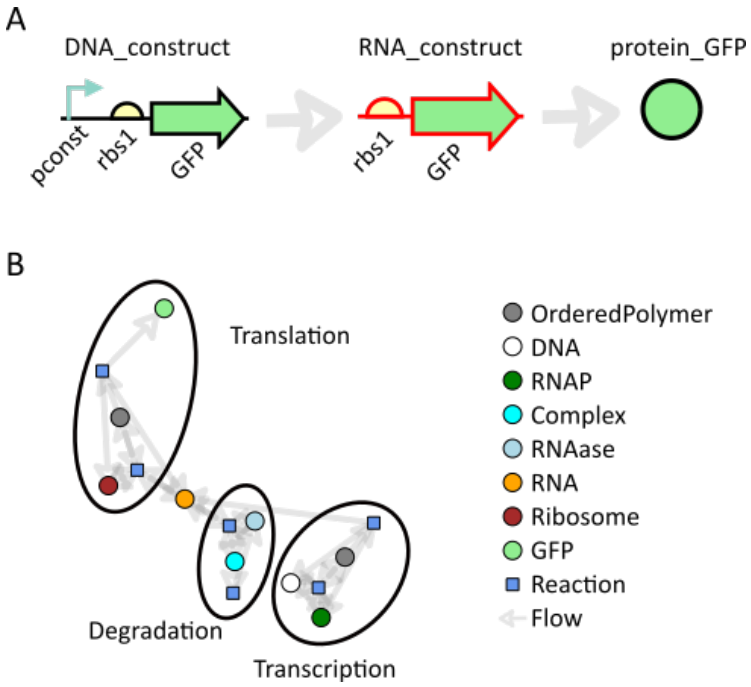


Figure 3.3: Simple transcription translation simulation. (A) A schematic representation of a simple transcription translation reaction. A `DNA_construct` produces an `RNA_construct` through transcription and an `RNA_construct` produces a `Species` (with `material_type` protein) through translation. (B) CRN generated from above simple schematic. Squares and circles represent reactions and species, respectively. Gray arrows show the direction of flow, for example a species that is a product of a reaction would exist at the end of an arrow emitted from a square. Clusters of species and reactions produced by different mechanisms are circled. The complexity of each of these mechanisms (and thus the organization and content of these clusters) can be changed by selecting which mechanism is to be used in the entire `Mixture` or in specific `Components`.

In some cases, it is also necessary to have a `Component` that contains other `Components` in a special organization. A `DNA_construct` is an example of such a `Component`, because it must contain fragments of DNA and know about their order. For this purpose, `Components` can also be `DNA_parts`, which know about their position and direction inside `DNA_constructs`.

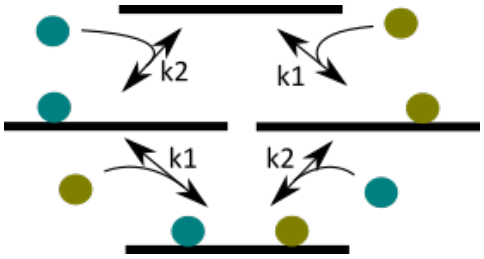


Figure 3.4: Two independent binding reactions. Two species (colored circles) can bind to a piece of DNA (black line) independently. Each species must be able to react with unbound DNA and also bound DNA. Because the binding is independent, the rate constant for binding is unchanged whether each species reacts with a bound or unbound DNA.

3.3 DNA Construct

Gene circuits in general, as constructed by synthetic biologists, exist in a linear fashion on a piece of DNA. Promoters drive production of RNAs immediately downstream, and the RNAs produced usually contain a single or a series of open reading frames (ORFs) which, in combination with upstream RBSes, encode proteins. The goal of the `DNA_construct` class is to contain a linear sequence of genetic parts that could be present in a circular or linear chromosome. The topology of the underlying DNA is important because one of the main assumptions in `DNA_construct` is that the parts (or `Components` as they are called in BioCRNpyler) interact with each other in sequence. `Promoters` produce RNA containing the parts immediately downstream until hitting a `Terminator`, and `RBSes` produce protein as long as both protein and RBS are present in order, in the correct orientation, on an RNA.

In BioCRNpyler, a `Promoter` is a container that brings together a combination of `Species`, `Mechanisms`, and `DNA`. The general reaction schematic for a transcription reaction (when using `TxTlMixture`) is as follows:

$$D_{DNA} + P_{RNAP} \underset{k_u}{\overset{k_b}{\rightleftharpoons}} D_{DNA} : P_{RNAP} \tag{3.1}$$

$$D_{DNA} : P_{RNAP} \xrightarrow{k_{tx}} D_{DNA} + P_{RNAP} + R_{RNA} \tag{3.2}$$

When dealing with simple DNA species that contain only a single promoter and produce only a single RNA, reactions like the above can be easily generated when a user creates a `Promoter("mypromoter",dna=DNA("mydna"))`. The goal of `DNA_construct`, however, is to allow an arbitrary ordering and assemblage of parts. Thus it should be possible for a single piece of DNA to contain many

promoters acting independently. This presents a logistical problem to BioCRNpyler in that a variety of bound species must be automatically created. If two seperate DNA molecules existed with independent promoters, it would be sufficient to simply populate reactions as indicated above in Equation (3.1). But, if we imagine both promoters present on the same piece of DNA, it would be possible to have a species where both RNAPs are bound at the same time, as in Figure 3.4. We will refer to this as *combinatorial binding*.

3.4 Local Component Enumeration

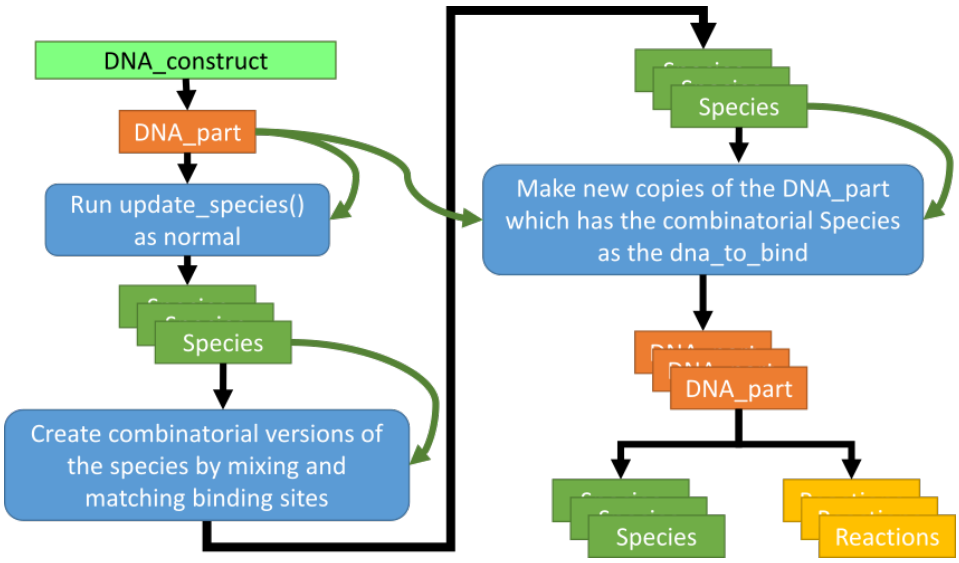


Figure 3.5: Combinatorial Enumeration. The function `update_species()` is called twice in combinatorial enumeration. First, each `DNA_part` is asked to `update_species` while binding to an "empty" version of the `DNA_construct` underlying species. This generates a set of possible bound states with each active `DNA_part` in isolation (first stack of green "Species" boxes). Then, these `ComplexSpecies` are recombined: for example, a complex at position 1 can be combined on the same molecule with a complex on position 2 (second stack of green "Species" boxes). Once these combinatorial complexes are made, a copy of each active `DNA_part` is made, each containing a different combinatorial complex. Finally, these copied `DNA_parts` are made to `update_species()` and `update_reactions()`, thus generating all the proper species and reactions in between the combinatorially recombined versions of the original `DNA_construct`. Black arrows represent logical flow, and green arrows represent information flow. For example, information from the `DNA_parts` present in the original `DNA_construct` is combined with the new `Species` generated from the combinatorial enumeration to produce the copied `DNA_parts` which are essential for producing the proper `Species` and `Reactions`.

Combinatorial binding occurs when at least two sites on one species can be bound independently by different species. The identity of the species which performs the binding is not important; the only thing that matters is that it is possible for any of the independent binding sites to be occupied and none of the sites is affected by any of the other ones. Our goal in implementing combinatorial binding in BioCRNpyler was to re-use the existing interface for DNA parts like `Promoter`s, and allow them to be used with complicated DNA species that can combinatorially bind multiple promoters. Another consequence of combinatorial binding is that it is essential to keep track of which binding site is occupied. For example, in the condition where two promoters exist on one DNA, it is important to know which of them is bound by RNA polymerase in the case that only one promoter is bound, because that determines which RNA will be produced. For this reason we have created the `OrderedPolymer` class, which serves as a way to organize species with multiple binding sites. An `OrderedPolymer` is a list where each element has a reference to its location within the list and also a link back to the parent. This is important because of the interaction with `Component`s such as `Promoter`s. Since a `Promoter` must accept a DNA `Species` as input, now we can use an element from the `OrderedPolymer` class (called an `OrderedMonomer`) as an input. Thus, a `Promoter` can use this `OrderedMonomer` in exactly the same way as a standard DNA species would have been used, but now any `ComplexSpecies` created as a result of `Mechanism`s contained within that `Promoter` will be created in the proper binding position of the parent `OrderedPolymer`. Likewise all the parts contained in a `DNA_construct` are also stored in an `OrderedPolymer`. Simply using an `OrderedPolymer` does not accomplish combinatorial binding. Given a `DNA_construct` that contains multiple active `Component`s, we allow each `Component` to generate the bound species relevant to its `Mechanism`, then create all combinatorial combinations of these bound species. This means, for example, if Promoter 1 yields a `ComplexSpecies` where RNAP binds to position 1, and Promoter 2 yields a `ComplexSpecies` where RNAP binds to position 2, there is a possible combinatorial `ComplexSpecies` where both RNAPs are bound. Once this combinatorial species is generated, it can be fed back into the active `DNA_part`s (`Component`s are also `DNA_part`s) in order to generate the proper reactions and species. The logical flow of this process is illustrated in Figure 3.5.

Given that all the `Component`s contained within a `DNA_construct` have had all relevant variables properly populated, this process will always yield the correct species and reactions for a given `DNA_construct`. A different algorithm, which we

call TxTl_explorer, is used to populate Components properly, according to the central dogma. This method iterates sequentially along the linear DNA_construct in order, tabulating all Components that exist between Promoters and Terminators and thus end up contained in RNAs, as well as CDS components which exist following RBS components. TxTl_explorer operates in the forward and reverse directions, as well as being able to cross from one side of a DNA_construct back to the beginning in case we are trying to represent a circular plasmid.

Within each Mixture, multiple DNA_constructs may be present. Combinatorial enumeration is able to properly create the necessary species and reactions pertaining to a single DNA_construct, and the DNA_parts contained within. A necessary consequence of combinatorial enumeration, however, is the creation of RNA_constructs. These are similar linear arrangements of parts except now an RBS fills the role of a Promoter. Thus, during CRN compilation, DNA_constructs lead to the creation of RNA_constructs. Once species and reactions are created for all DNA_constructs in the mixture and all RNA_constructs that started life in the mixture as well as those that are generated, nothing else needs to be done. Thus we call this combinatorial enumeration "local" because each DNA_construct only knows about itself, and there is no potential for infinite recursion. This is in contrast with integrases, as presented in the next section.

3.5 Integrase Sites and Global Component Enumeration

Integrases are capable of rearranging DNA based on the identity and layout of DNA sequences called attachment sites. The mechanism by which different biochemical integrases undertake their reactions can be different, but the outcome is usually similar: a recombination event occurs between two attachment sites. In the case of serine integrases such as Bxb1, two different sites known as attB and attP react to form attL and attR, recombining the DNA in the process. From the point of view of what the recombination actually does to the sequences being recombined, five basic types of integration are possible, as illustrated in Figure 3.6.

Since DNA_constructs represent linear sequences of parts, the transformations they undergo as a result of integrase activity are fairly straightforward, and lead to the creation of new DNA_constructs. These new DNA_constructs might have different RNA or protein products, owing to the fact that promoters might now be arranged differently with regard to terminators or protein coding genes might now be oriented in forward or reverse directions, and these eventualities are handled by local

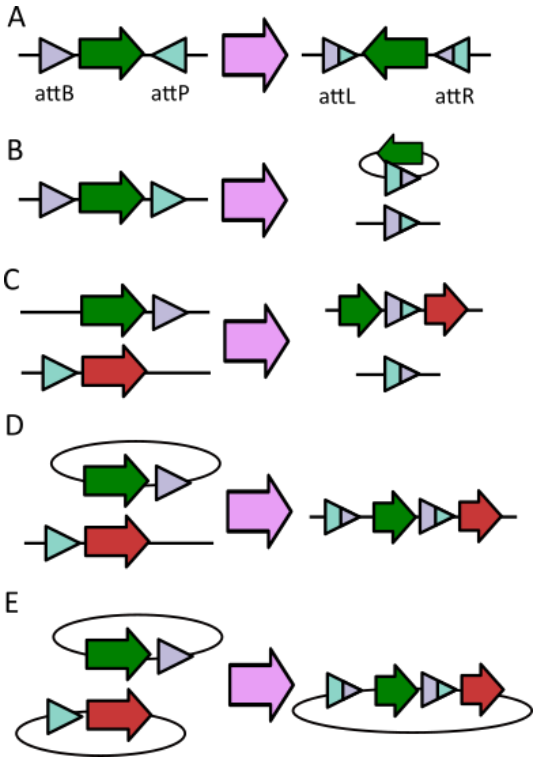


Figure 3.6: Topology changes induced by integrase. Triangles represent integrase sites, black line represents DNA, thick arrow illustrates the direction of the DNA. (A) Flipping: integrase sites on the same piece of DNA are pointed at each other and the DNA between the integrase sites is inverted in direction. (B) Deletion: integrase sites on the same piece of DNA are pointed in the same direction, and the DNA between the integrase sites is excised into a circular fragment. (C) Integration between linear DNA: integrase sites are present on two different pieces of DNA, which are both linear. This results in two pieces of DNA containing the recombined sites, that are also both linear. (D) Integration of a circular DNA: integrase sites are present on two different pieces of DNA, but one of them is circular. This results in one linear piece of DNA containing the recombined pieces. (E) Integration of two circular DNAs: integrase sites are present on two different pieces of DNA, and both are circular. This results in one circular piece of DNA with essentially the same organization as D.

enumeration, as presented in the previous section. The challenge to BioCRNpyler, and ultimately to the user, is when to stop generating these new constructs. One might consider a simple case such as that presented in Figure 3.6A. Integrase sites facing each other lead to a portion of DNA being "flipped." Due to the nature of serine integrases, the reaction does not proceed in the reverse direction, so it seems like generating a single new `DNA_construct` with the proper sequences flipped should be the end.

However, in a real cell there could be multiple copies of the same DNA sequence, whether that DNA exists as a plasmid, a cell is replicating and thus has multiple genomes, or the cell was transformed with many linear copies of the same DNA. Thus integrase could perform a recombination reaction between attB and attP on two different pieces of DNA. We call this type of reaction "intermolecular" to distinguish it from the "intramolecular" flip reaction which would occur between two sites present on the same piece of DNA. Thus even a simple flip reaction can lead to infinite possible `DNA_construct`s, and for this reason we have included a variety of flags that allow users to decide whether intermolecular reactions are allowed, and sequentially how many times the pool of `DNA_construct`s in a mixture are interrogated for new `DNA_construct`s.

Once BioCRNpyler has enumerated all the possible integrase reactions that can occur, it is important to make sure that the resulting CRN includes the proper reactions to connect together the generation and destruction of these new `DNA_construct`s. This is more sophisticated than normal reaction and species generation for a couple of reasons. First, the species that are involved with integrase reactions are in general already generated. Reactant and product `DNA_construct`s have already been generated as part of the first step which is enumerating possible integrase reactions. When an integrase recombines a piece of DNA, we assume that the intervening DNA goes along for the ride, and any complexes and bound proteins present there should not be affected by the action of the integrase. This means, for example, that if a promoter is to be flipped, then any polymerase that happened to be bound to that promoter could get flipped as well. Thus, the correct CRN should contain a reaction going from unbound DNA to unbound, flipped DNA, but also from RNAP-bound DNA to RNAP-bound flipped DNA.

As we mentioned above, in local component enumeration, a copy is made of each `DNA_part` which contains a species representing a different bound state of the rest of the DNA molecule. Integrase sites are likewise copied, but in a way integrase sites are not independent as all other `DNA_part`s are assumed to be, since they care about the state of their partner site. In the case of an intermolecular reaction, an integrase site must care about all the different combinatorially bound states of the other `DNA_construct` that it will integrate with, and create the proper reactions. For this reason we call this step "global enumeration," and the process is a bit more complicated.

As indicated in Figure 3.7, a recursive step allows `DNA_construct`s to create

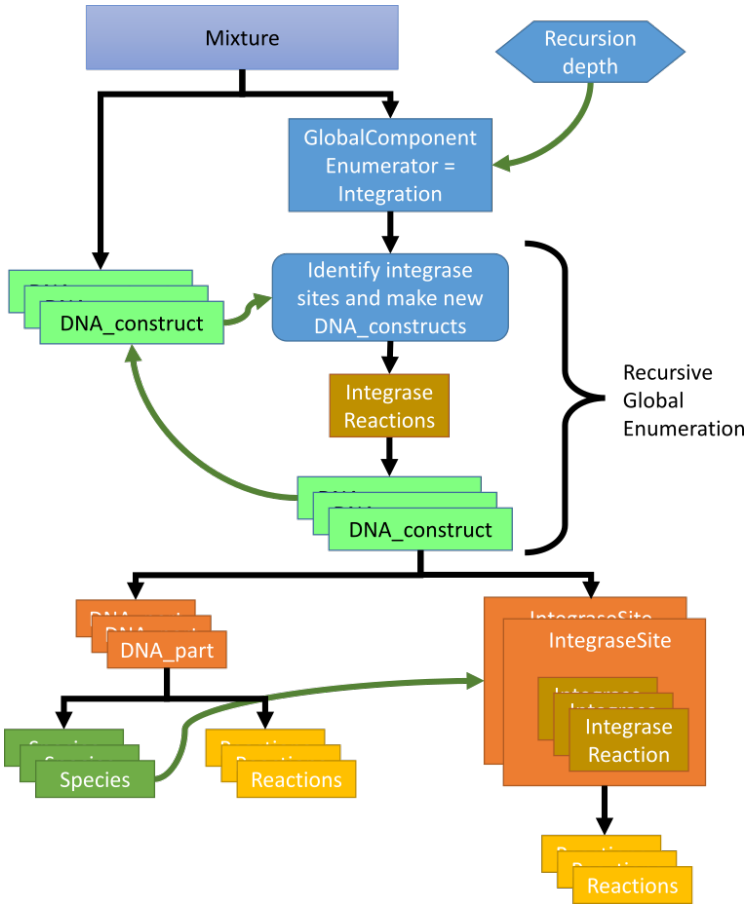


Figure 3.7: Global enumeration. A mixture (blue box at the top) starts by containing `DNA_constructs` and a `GlobalComponentEnumerator`. A user-entered parameter of recursion depth is fed into the `GlobalComponentEnumerator` to limit how many times new `DNA_constructs` will be generated. The global enumeration loop is indicated by the curly brackets. All the `DNA_constructs` in the mixture are evaluated by the `GlobalComponentEnumerator` to determine if they will participate in integrase reactions. Multiple different `GlobalComponentEnumerators` can be added to represent different integrases or other processes like splicing. If a new `DNA_construct` is generated by the `GlobalComponentEnumerator`, that information is saved in the `IntegraseSites` that participated in the reaction (in the case of integrase `GlobalComponentEnumerator`). Once the recursion depth is satisfied, remaining `DNA_constructs` undergo local component enumeration. As part of this process, species and reactions are generated, and `IntegraseSites` make use of these combinatorially bound species to create the proper reactions between species that integrases would allow.

new `DNA_constructs` using the logic embedded in an integrase-specific `GlobalComponentEnumerator`. As part of this process, the integrase reactions (flip, deletion, integration, etc) which will take place are recorded within `IntegraseSites`

which are a type of DNA_part. Then, during local component enumeration, these saved integrase reactions are used to perform integrase reactions on species containing other bound species, which were generated as part of local component enumeration. These recombined species are used to create reactions that involve integrase. No new species are generated using the integrase GlobalComponentEnumerator, as the integrase binding to integrase sites has already been taken care of during local component enumeration. Real integrases have an intermediate step where a tetramer of integrase molecules creates a bridge between the two attachment sites, and this type of reaction would be a good improvement to add in future work. The recursion inherent in global component enumeration reveals that even relatively simple integrase-containing constructs can lead to very complicated populations of different DNA sequences inside cells. It may be possible to ignore this complexity, as in the flipping example it is likely that intramolecular recombination is much more favorable than intermolecular reactions. It may also be possible to harness this complexity, as we aim to do with the event recorder system presented in the previous chapter. In any case, the ability to easily generate these very complicated CRNs allows integrase-containing systems to be studied more carefully, and "undesired" or "unexpected" integrase reactions may be more easily predicted and accounted for.

3.6 Promoter Flipping Example

A relatively straightforward construct to simulate is one where integrase activity causes an inversion of a constitutive promoter. The DNA construct is made such that compatible integrase sites are facing each other on either side of a promoter. Then, integrase activity causes the sites to flip the DNA in between them. To simulate such a construct using BioCRNpyler, we start with the construct specification. We define the individual parts as DNA_parts, and then combine them into a DNA_construct in the form of the initial, unflipped promoter sequence (see Figure 3.8 A). We may want to limit the simulation to only consider intramolecular reactions, and ignore multiple flipping constructs reacting with each other or any other such complications. In order to do that, we must specify an attribute called "no_inter" on any part in the DNA_construct. Thus, any DNA_constructs that contain a part having this attribute will also not participate in intramolecular reactions. Thus, if new DNA_constructs are formed after recombination, they will still remember that intermolecular reactions are forbidden. After the initial DNA_construct is specified, the compilation process begins, using local enumer-

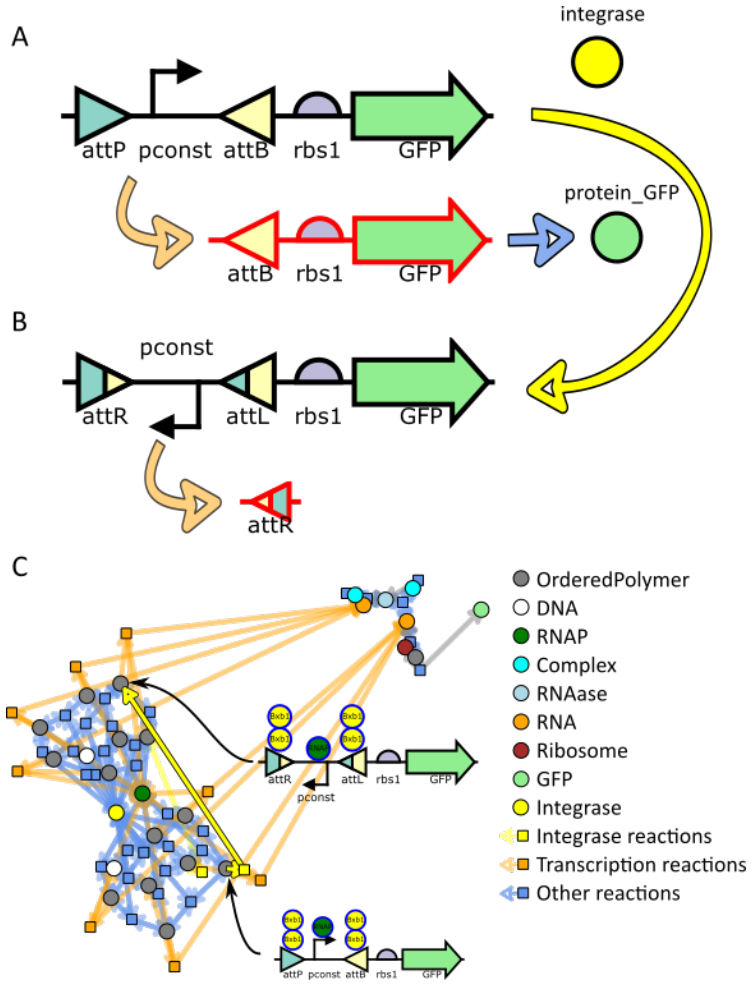


Figure 3.8: Integrase-containing CRN. (A) Input `DNA_construct` containing integrase sites, represented by triangles labeled with attP and attB. This `DNA_construct` contains a `Promoter` and so it can transcribe an RNA, which eventually leads to protein. (B) Recombined `DNA_construct` generated by global component enumeration. A "flip" reaction between the two integrase sites can be used to reverse the promoter to produce a different RNA, which now does not lead to protein. (C) CRN diagram of this system. Reaction flow is color coded for readability, and one integrase reaction is highlighted. Two blobs of highly connected species and reactions represent the two `DNA_construct`s, each of which undergoes many binding reactions. One particular `OrderedPolymerSpecies` contains integrase bound to both attachment sites, and RNAP bound to the promoter. This complex is recombined, through an integrase reaction, into the "flipped" product species, where the RNAP is still bound to the promoter, but the promoter now faces in the opposite direction.

ation to create an `RNA_construct` containing the `DNA_part`s that make up the mRNA made in the forward direction (Figure 3.8 A). Since the integrase global enumerator was added to the model, global enumeration is also used to create a

new DNA_construct that represents the flipped version of the input construct (see Figure 3.8 B). Again, this flipped construct also creates an mRNA, but this time the mRNA contains only the attR site since the promoter is facing the other direction.

Finally, the full CRN graph is represented in Figure 3.8 C. This CRN has many species and many states which can lead to mRNA production, indicated in the plot with orange arrows. This is because integrase can bind to the attachment sites independently, and so the promoter can be active whether the attP site is bound, the attB site is bound, none of them are bound, or both of them are bound. The same is true for the flipped construct, with attL and attR. This presents an opportunity for any sort of transcriptional interference to be applied. If we were interested in having the bound integrase decrease the rate of transcription through it, now the species exist in the model to allow these modified rates to be applied. Another interesting thing is that the bound state of "pconst" does not affect integrase activity. Two integrase reactions are thus possible: flipping the promoter when it is bound to RNAP and also when it is not bound. These reactions are indicated in the CRN graph with yellow arrows in Figure 3.8 C. Thus from an intial input of five DNA_parts arranged into a DNA_construct, a CRN with 1 new DNA_construct, 26 species and 40 reactions is created. By simply removing the "no_inter" attribute, an arbitrarily large number of DNA_constructs and species could be generated from just this simple starting construct (and the idea that the construct can react with another copy of itself).

In using these simulation features we found that finer control was sometimes desired for the integrase reactions. For example, sometimes it is desirable to say that a particular type of integrase reaction (from the list of types in Figure 3.6) should be prevented or allowed. To do this, one need only specify an IntegraseRule containing an argument such as allow_deletion, allow_integration, or allow_inversion. If any of these are set to False, then the integrase will never perform that specific reaction. Such a feature can be useful when implementing resolvases or invertases which actually do only perform one of the several types of integrase reactions. Another useful feature is the ability to ignore integrase binding. Part of the reason why there were so many species and reactions in Figure 3.8 C is that integrases binding to integrase sites rapidly increase combinatorial complexity. It is important to consider integrase binding as integrases that bind to attL or attR are in effect sequestered away from being useful participants in reactions. To disable binding, one must replace the Mechanism inside the attachment site DNA_part

with the `EnzymeIntegration` mechanism, which does not create a bound species. One must also indicate that integrase attachment sites should not bind integrases by setting the `integrase_binding=False` argument. These code examples are illustrated in Appendix A.

The idea of integrase being able to react with DNA fragments regardless of protein binding state in between the integrase sites is interesting. One can imagine a hypothetical scenario where an integrase creates a deletion between two sites, thereby excising a circular piece of DNA. But, if RNA polymerase happens to be transcribing in the sequence between integrase sites which gets deleted, then that RNAP could become trapped on this circular DNA, transcribing around the horn until dissociation. Of course an event like this is likely rare, and once it happens, it can not readily happen again since the deletion needed to produce this hypothetical circle has already taken place. Currently, BioCRNpyler does not simulate RNAP traversing sequences that are being transcribed, but this suggests it would be a useful addition.

3.7 Recursion Depth Example

Global component enumeration allows `DNA_construct`s to create more `DNA_construct`s as a result of integrase activity, which in this case is implemented as a `GlobalComponentEnumerator`. Since global component enumeration is recursive, it is necessary to add a "recursion depth" argument that determines how many times global components can be enumerated. In a real cell, there would realistically be no limit to such recursion, because serine integrase is indifferent to the context of attachment sites. However, when we create the CRN for simulations, the more species and reactions there are, the longer it takes to simulate. When plasmids recombine, the result is multimer plasmids. In order to attain plasmids made up of more than two copies of a plasmid, multiple sequential integration events would have to happen. The probability of sequential integration events is thus significantly lower for each "level" of integration, which is equivalent to each depth of recursion. Thus at some point, deeply recursed plasmid multimers will be unlikely enough in solution that they may as well be ignored.

Another consequence of global component enumeration is the rapid increase in CRN size as recursion depth is increased. At each level of recursion, the number of `DNA_construct`s can go up exponentially, and each construct would have more and more binding sites and more and more species would need to be generated to

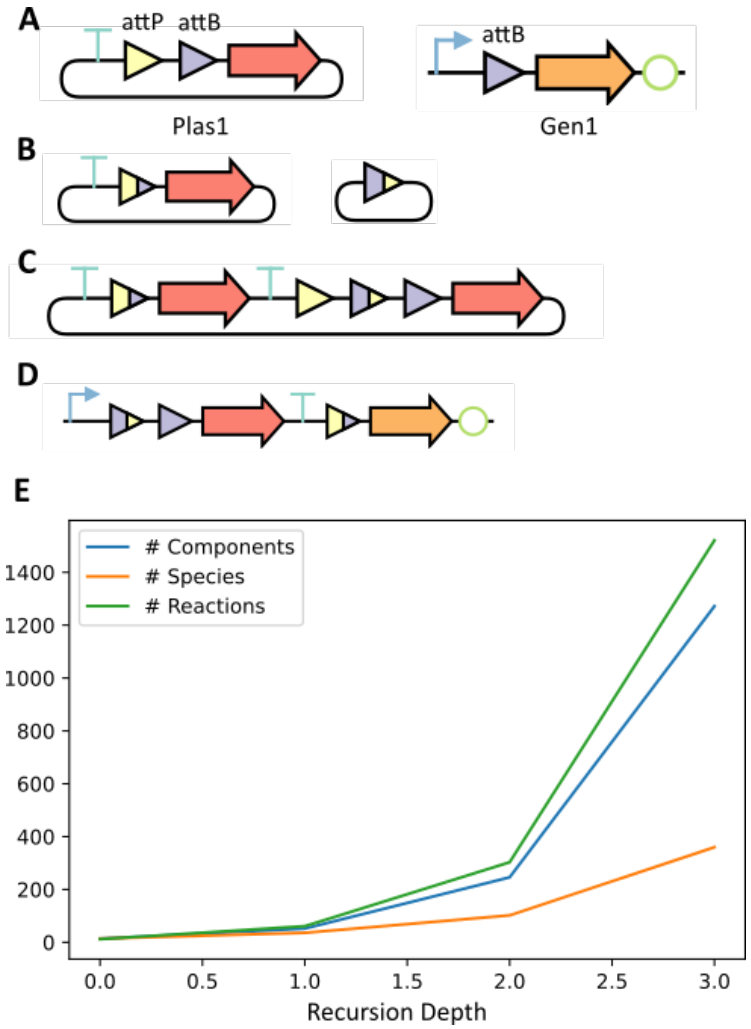


Figure 3.9: Example of continuous integration recursion depth. (A) Input `DNA_constructs` containing integrase sites, represented by triangles labeled with attP and attB. One `DNA_construct` is a circular plasmid which is allowed to integrate with itself, and the other is intended to represent the genome site, having only attB. (B) Integrated `DNA_constructs` generated by global component enumeration as a result of a "deletion" reaction between the attB and attP of Plas1. (C) Integrated `DNA_construct` generated by global component enumeration as a result of an "integration" reaction between the attB and attP of two different copies of Plas1. (D) Integrated `DNA_construct` generated by global component enumeration as a result of an "integration" reaction between the attP of Plas1 and attB of Gen1. (E) Plot of species, reactions, and components generated based on recursion depth. The greater the recursion depth the more the products of recombination are allowed to recombine to produce more `DNA_constructs`.

account for partially bound states. This is illustrated in Figure 3.9: starting for a relatively starting material containing a single plasmid and a single genome, the first

recursion depth generates 4 additional DNA_constructs, two of which (Figure 3.9 C and D) are themselves capable of integration. So even though we started with three integrase binding sites, after one recursion we have a total of 12 integrase sites, 6 of which can still be integrated. The speed of this scaling is illustrated in Figure 3.9 E, where the number of components, species, and reactions in the CRN goes up by almost five fold for every new depth of recursion. Of course most of the species and reactions in this case are binding and unbinding of integrase, as we saw in Figure 3.8, where a large number of reactions and species were present but only two of the reactions were driven by integrase. Thus, if we are only interested in integrase reactions, it may be prudent to ignore integrase site binding in favor of simulating the integration reactions and identity of DNA_constructs generated only.

One could imagine a different simulation framework where the species and reactions are generated only if they occur in the simulation, rather than all at the beginning. This is not how BioCRNpyler is built and indeed not how the SBML to simulator pipeline is capable of operating, but the idea might be a good compromise for recursive systems such as discussed in this work. As is, the GlobalComponentEnumerator framework is extremely useful both for predicting the output of integrase reactions and for automatically generating large sized CRNs.

3.8 Conclusion

CRNs are commonly used for simulating biological processes and reactions. Generating CRNs with enough detail to fully capture complex biological processes can be difficult and time consuming. Automating CRN creation with BioCRNpyler allows the modeler to specify the level of complexity for specific mechanisms and sub-components of the model to facilitate rapidly analyzing models with different levels of sophistication. By applying common assumptions and known mechanisms to known processes, a higher level modeling language can be achieved. Now, by using BioCRNpyler, the modeler does not need to specify, for instance, that a DNA bound to polymerase makes RNA, but only that a promoter exists and points in a certain direction. Utilizing the DNA_construct interface, modelers need only specify the sequence and direction of DNA components with known mechanisms and interactions. Automated enumeration of species and reactions for high level specifications allows more simple user interaction and more rapid model creation and evaluation.

Another consequence of automating `DNA_construct` enumeration is automatic enumeration of integrase reaction products. From a single input construct, infinite new constructs can be automatically created based on known rules of integrase recombination. To prevent the production of excessively large CRNs, a recursion depth limit determines the number of times that integrase-based recombination is allowed to happen in succession. In order to predict the products of integrase reactions a user must specify recombination sites and integrase reactions possible in their system, but also the recursion depth to which new recombined constructs are simulated. Through the use of specific flags, certain integrase reactions can also be prevented such as deletion, flipping, or intermolecular versus intramolecular reactions.

The advent of these automated high level CRN-creation functions allows very easy model creation at arbitrary levels of complexity and opens the door to creating very large CRNs that can then be used for model validation and parameter inference.

Chapter 4

FUTURE WORK

A molecular event recorder offers a unique way to capture the world from the point of view of a single bacterium. Many mysteries of life at the single cell level can be unraveled if such an event recorder could be built and operated effectively. It may even be possible that traditional scientific instruments may not be necessary if single cells could effectively communicate their molecular state to the macroscopic scientist. Cellular developmental decision-making, reproduction, mutation, and environmental response could all be elucidated if effective molecular event recorders were possible.

The reality is that molecular event recorder technology is far from this dream. The work presented here shows progress towards a molecular event recorder, yet the fully functioning ideal may look completely different from the system we have developed. Perhaps a different integrase, different method of DNA generation, or different mechanism of integrase control would lead to an overall more capable event recording system. Several other works present promising alternative systems that can also record events [5, 15, 64, 75]. The question we must ask, as bio-engineers, is: what features are desired in such an event recording system?

4.1 Continuous Event Recorder

We imagine two classes of usage for an event recording device. First, an event recorder could be used to create an inert record of unknown events. This is similar to a tape recorder in functionality: The tape is barely changed, but enough so that a sensitive tape player could reconstruct the events that lead to the tape's changes. The second type of event recorder would be more like a programmable computer. Certain defined events could be utilized to make desired functional changes to the recording medium that predictably change the behavior of the device containing the record. Fortunately, DNA can fulfill both a role as an inert storage medium and an active "program memory," depending on the sequence.

In the tape recorder, the changes enacted by the recording system are in a way not important, as long as they can be read out. Since in our system, DNA sequencing is used as the readout, any predictable change to DNA would satisfy this requirement.

Likewise, systems that depend on spacer insertion or base editing could perform identically, from the point of view of event recording and read-out. There are other challenges to consider: Is the change heritable? Can the system record magnitude separately from duration? How many different "events" can be recorded using a certain type of genetic change? How long can the recorder be functional before all the "space" is consumed? How much is the system sensitive to "noise"? The answers to these questions are inherent to the technology as well as the implementation. Integrases, for example, can allow a piece of DNA to be flipped or deleted, thereby enacting a very noticeable genetic change in a precise region.

Event recorders built with this technology tend to be sensitive yet limited, as the DNA usually can only be "flipped" once before no further integrase action is possible. This specific challenge is what we attempted to overcome in this work. By allowing continuous integration, the initial sequence of the recording site theoretically does not limit the length of recordings that can be enacted. However, a practical limit still exists on the maximum recording length, and in our system that is set by the number of plasmids present. Since these plasmids are used to create the record, they form a sort of limited ink pool that can be exhausted. Other works have tried to overcome this issue by producing DNA substrate via reverse transcription [7] or simply transforming in more DNA [64, 65].

One important challenge that we have yet to overcome is that of unwanted reactions. The most efficient event recorder would enact changes to DNA only when instructed to, and only in the location which will later be read out. If we consider an event recorder where plasmids are integrated into the genome, it seems clear that any integration events that do not involve the genome site are undesired. One major flaw in our system is the presence of both attP and attB integrase sites on the same plasmid. It is necessary to replace the destroyed attB site on the genome post-integration, but this also means that the integrase knows no difference between integrating with an attB on another plasmid or on the genome. This leads to a sort of confounding effect where a record builds up in multiple places. For example, if we allow barcode 2 to integrate, it may well integrate into a plasmid and form a multimer plasmid 12, or into the genome to expand the memory array there by one "barcode 2" unit. However, plasmid 12 can subsequently also become integrated into the genome, and thus a single integration event leads to a genome expansion by two barcode units. In the short term this is not a big problem, since the concentration of plasmid 12 is small initially and the concentration of individual plasmid 1 or plasmid 2 is

comparatively large. But, this confusion imposes another limit on the maximum duration of event recorder function.

One way we have chosen to try to overcome this issue is to eschew the genome in favor of the plasmids themselves. The order that the plasmids recombine with each other should, in theory, produce a record just as well as the genome does. This does, however, have many of the same limitations as the genome does. Multimer plasmids can still integrate with multimer plasmids, and because each plasmid has two sites for integration, as opposed to the genome's one, it becomes a bit more difficult to piece together the order of events from looking at the order of barcodes on a single plasmid. Instead we found that it was necessary to look at the relative concentration of a series of multimer plasmids.

Integrons

The core issue is that the DNA sequence which directs the integrase to action is present in multiple locations. If there was some way to separate the targeting system from the integrating system, we could imagine an event recorder which acts like the "integration only" simulation from Figure 2.2 A. There is in fact a natural system that functions somewhat like this, besides the CRISPR array, known as an integron [29]. Integrons are found in a wide range of hosts, on transposons, and particularly in *Vibrio*. The basic idea is that a DNA sequence known as attI acts as the integration site, and sequences containing attC can be repeatedly inserted at attI without destruction of the attI itself (Figure 4.1 A).

On the surface, this system seems to offer a natural answer to many problems we inadequately addressed in this work. Sequences are integrated into the array only and not anywhere else. Sequences in the array are already ordered according to chronology. Extremely long arrays have been found in nature and therefore must be possible to create [29]. However, it is still not perfect. In order for DNA to be integrated into the integron it must be single stranded [53]. After sequences get integrated, they can still be excised because they still contain attC[8]. The rate of integron activity is quite low. And, importantly, the source of the DNA for integron addition is still not well defined. The natural system is thought to act primarily on linear single-stranded DNA taken up from the environment or conjugation. It is possible to get a plasmid to become integrated into attI in a synthetic system (Figure 4.1 B, C), but in our hands the rate of integration was substantially slower than Bxb1. We were never able to see more than one integration.

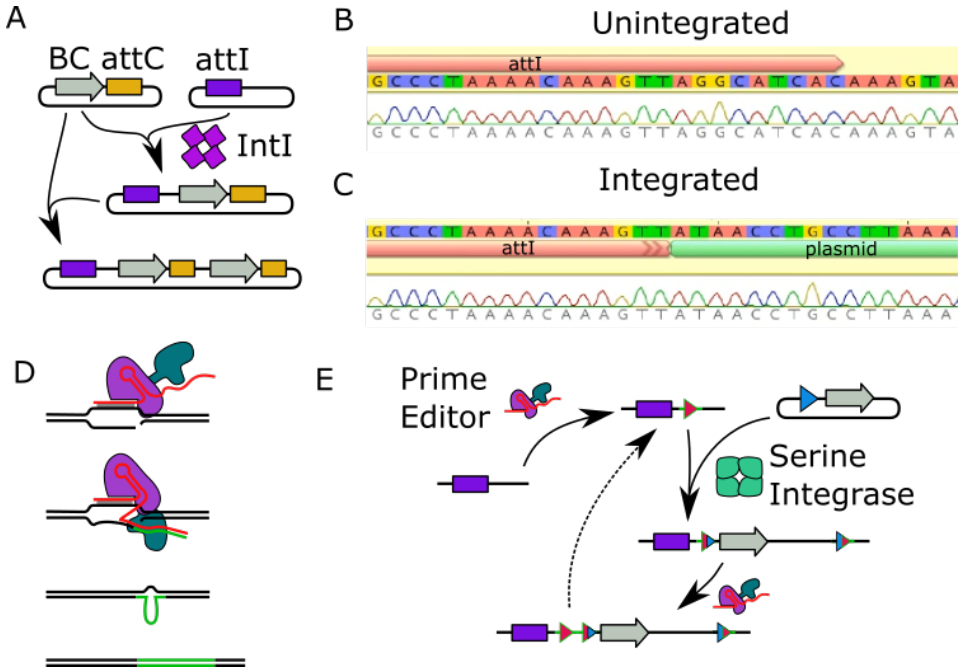


Figure 4.1: Alternative event recorder schema. (A) Integron event recorder. The integron integrase IntI recombines between attI and attC to insert the attC containing sequence after attI. This process is by itself continuous; more attC-containing plasmids can continue to be sequentially inserted, with the newest ones being inserted closer to the "attI." (B) Sequence of the 3' end of attI, with nothing integrated. (C) Sequence of the 3' end of attI, with the attC-containing plasmid integrated, demonstrating a successful plasmid-plasmid integration reaction performed by the IntI system we tested. (D) Prime editor mechanism of action. First a nickase cas9 (purple blob) nicks the DNA at the right position, the extended sequence on the pegRNA hybridizes with the free nicked DNA, then the fused reverse transcriptase extends the hybrid to produce reverse-transcribed DNA (green line). This reverse-transcribed DNA is added only to one strand and can be an insertion. After replication, the insertion is copied to make a double stranded insertion. (E) Prime editor schema using the insertion mechanism to add an integrase attB site (red triangle). Ink plasmid containing attP site (blue triangle) can then integrate with the inserted attB site, and a new attB site eventually gets inserted again, repeating the cycle.

Prime Editor

Recent publications have shown that reverse transcriptase fused to nicking Cas9 could be used to edit DNA adjacent to a guide RNA cut site [5] (Figure 4.1 D). This has recently been expanded to adding entire integrase attachment sites using the "PASTE" system [32]. Using a system like this seems to be an easy way to overcome the "unwanted reactions" inherent having an ink plasmid that contains both attB and attP sites. If it was possible to inducibly add an integrase site, ink plasmids could

be made such that they have only attP sites, and the attB was added using this DNA editing system (Figure 4.1 E) There are, however, still issues to overcome. In PASTE, RNP and DNA are transfected into cells where they become active. The RNP contains RNA that is reverse transcribed into an integrase attachment site, and the DNA has the integrase attP site as our ink plasmid would have. Because the guide RNA containing the added attachment site is never seen in its DNA form, one never has to worry about integration into the guide RNA containing gene. In a putative event recorder built with this platform, integration into the guide RNA containing DNA sequence would be a significant problem. After all, the template for attachment site addition is itself just an attachment site. If that template exists in DNA form, to be transcribed into a guide RNA which is then used for reverse transcription, there is nothing to stop the integrase from reacting with the site in the template. Something like a self-splicing intron, which is a DNA sequence that removes itself from RNA, could be integrated into the middle of the attachment site sequence and therefore prevent this from happening.

If we consider just the act of DNA editing, however, it is possible to imagine an event recorder that does not use integrases at all. Prime-editing is capable of inserting bases in a way that should allow continuous recording of information. Even though prime editing has a far lower insertion/deletion (indel) rate than homology-directed recombination [5], that rate is not zero. If we intend to create a system relying on continuous editing of a genetic region, indel formation could halt this process early. However, multiple alternative pegRNAs could be used, recording simultaneously at multiple sites. Such a system might be much more easily scalable than the integrase-based system presented here. It is clear though that further gains in prime editing efficiency are needed before such a system can be seriously considered.

4.2 Integrase Site Control

The ability to control integrase activity by competitively binding dCas9 to integrase attachment sites is a novel integrase regulation method we have developed in this work. Another group has recently arrived at a similar system using repressor binding sites flanking integrase attachment sites [40]. Typically integrase activity is controlled by production of the integrase itself or a recombinase directionality factor (RDF) [9]. Usually when an integrase is active, it can recombine all cognate attachment sites, and so to make different recombination events, different integrase sites must be used [82]. While there are many orthogonal integrases that have been discovered, they have different efficiencies and idiosyncrasies. For example, it is

possible that one might want to use a certain integrase site because it has a promoter or does not have stop codons or any other reason. Then, it becomes very useful to have an alternative integrase site control method such as the one described here.

In this work we have tested using dCas9 on one side of the integrase site to prevent integrase activity, and the effect achieved can be described as measurable but not complete integrase recombination prevention. However, one can imagine placing two guide RNA binding sites, one on either side of the attachment site would be much more effective. Likewise, there is no reason why the method described here of overlapping guide RNA binding with integrase attachment site cannot be applied to many other integrases and attachment sites, thereby creating a huge library of repressible integrase components.

A natural application of repressible integration is the creation of designed population ratios. An integrase that performs one *or* another recombination, subject to bias via guide RNA expression, could make for an easily adjustable population switch. Since the change is permanent, one can imagine creating permanent "cell differentiation" strains which are readily tunable to a desired ratio. Because the guide RNA repression can be used to control sites from the same integrase, it would also be relatively straight forward to create a "selector" circuit where one integrase attB site can be integrated into one of an array of attP sites, depending on any conditions or logic that are desired.

Given that integrase repression is possible, one might consider that integrase activation could be achieved as well. Indeed, integrases bound to attL or attR are not active [61], unless RDF is present. One can imagine a construct where RDF is fused to dCas9, and therefore integrases can be activated by binding this RDF fusion nearby. Activation and repression of integrase sites can possibly be enacted by the same enzyme, depending on the orientation of binding or the nature of sites which are bound. After all, integrases bound to RDF do not recombine attB and attP, so this may be an alternative way to repress integrase activity as well.

4.3 Modular Simulation Automation

Synthetic biologists often create linear assemblies of DNA constructs that interact in a transcriptional regime. A synthetic DNA will contain promoter, ribosome binding site, and protein coding gene and in turn will produce RNA followed by protein. Usually that means the RNA and protein concentration will change over time, and these dynamic changes make up the system behavior that the synthetic biologist

is interested in. By expressing transcription factors or mRNA binding proteins, production rates of RNA or protein can be affected by downstream products.

Integrases offer another avenue of regulation, because they can cause recombination events and allow the presence of integrase proteins to affect the concentration of DNA assemblies. An integrase can cause an assembly of DNA parts to become rearranged into a new assembly of DNA parts, and this assembly would now be called a `DNA_construct`. Keeping track of and enumerating these types of rearrangements was the goal of the simulation automation work presented here. Specifically, we developed a series of algorithms that effectively generates a set of `DNA_construct`s that results from integrase-based recombination.

One could imagine a different type of recombination that can behave in a similar fashion. Messenger RNA splicing is one example of a recombinase-like reaction that occurs on the RNA level. Certain specific sites are recombined to produce new mRNAs. The interesting thing about splicing is that it often leads to different protein subunits. Thus far, in `DNA_construct` we have kept track of proteins as monolithic units that are either expressed or not. However, spliced mRNAs would require the development of additional logic for protein domains, that could inform specific protein function. For example, an mRNA could choose between a domain that binds protein A or protein B. Thus, the resulting protein would have to have an altered binding partner, but still the same catalytic effect (such as removing phosphate).

Another avenue for future development would be the addition of more sophisticated transcriptional logic. Currently, parts in a `DNA_construct` that are downstream of a promoter are transcribed, and when a terminator is reached, virtual transcription stops. In a real cell, transcriptional and translational interference can lead to interactions between neighboring parts. For example, a terminator that is being actively transcribed can affect the activity of a downstream promoter[83]. Likewise, a stop codon and ribosome binding site in close proximity results in translational coupling [47]. Since we already evaluate the orientation and proximity of genetic elements in a `DNA_construct`, it should be fairly straightforward to automatically create these context-sensitive connections between neighboring DNA parts.

To better integrate simulation into the synthetic biology workflow, it would also be useful to integrate genetic construct simulation with feature detection and DNA design. Often, DNA constructs are assembled by combining a set of known parts in a defined order. If we could connect DNA sequence, functionality, and parameters, a synthetic biologist could perform both physical DNA design and simulation at

the same time. Since the DNA sequence contains more information than just what type of promoter a certain part contains, these DNA sequence-centric parts could be updated over time as more features are identified. Perhaps a particular integrase site is found to have a cryptic promoter, then that DNA sequence component can be updated to contain a promoter function, and simulations made with that DNA sequence would more closely resemble reality. Likewise, if a certain DNA sequence only forms after recombination, then not only would an integrase be capable of rearranging the order and connectivity of DNA parts, but also create new parts generated from parsing the DNA sequence itself.

BIBLIOGRAPHY

[1] murraylab_tools software package. `https://github.com/biocircuits/murraylab_tools`. Online; accessed 13 December 2021.

[2] SIMbiology MATLAB package. `https://www.mathworks.com/products/simbiology.html`.

[3] Addgene. pBEST plasmid. `http://n2t.net/addgene:45789`.

[4] U. Alon. *An Introduction to Systems Biology: Design Principles of Biological Circuits*. CRC press, 2019.

[5] A. V. Anzalone, P. B. Randolph, J. R. Davis, A. A. Sousa, L. W. Koblan, J. M. Levy, P. J. Chen, C. Wilson, G. A. Newby, A. Raguram, and D. R. Liu. Search-and-replace genome editing without double-strand breaks or donor DNA. *Nature*, 576(7785):149–157, 2019.

[6] A. Arkin, J. Ross, and H. H. McAdams. Stochastic kinetic analysis of developmental pathway bifurcation in phage λ-infected *Escherichia coli* cells. *Genetics*, 149(4):1633–1648, aug 1998.

[7] S. Bhattarai-Kline, E. Lockshin, M. G. Schubert, J. Nivala, G. Church, and S. L. Shipman. Reconstructing transcriptional histories by crispr acquisition of retron-based genetic barcodes. *bioRxiv*, 2021.

[8] D. Bikard, S. Julié-Galau, G. Cambray, and D. Mazel. The synthetic integron: an in vivo genetic shuffling device. 38(15):e153–e153, June 2010.

[9] J. Bonnet, P. Subsoontorn, and D. Endy. Rewritable digital data storage in live cells via engineered control of recombination directionality. *Proceedings of the National Academy of Sciences of the United States of America*, 109(23):8884–9, Jun 2012.

[10] J. Bonnet, P. Yin, M. E. Ortiz, P. Subsoontorn, and D. Endy. Amplifying Genetic Logic Gates. *Science*, 340(6132):599–603, May 2013.

[11] C. S. Branda and S. M. Dymecki. Talking about a revolution: The impact of site-specific recombinases on genetic analyses in mice. *Developmental Cell*, 6(1):7–28, 2004.

[12] R. Cencic, H. Miura, A. Malina, F. Robert, S. Ethier, T. M. Schmeing, J. Dostie, and J. Pelletier. Protospacer Adjacent Motif (PAM)-Distal Sequences Engage CRISPR Cas9 DNA Target Cleavage. *PLoS ONE*, 9(10):e109213, 2014.

[13] M. V. Cheshire and S. J. Thompson. Configuration of soil arabinose. *Biochemical Journal*, 129(2):19P–19P, September 1972.

[14] D. A. Chetverina, P. V. Elizar'ev, P. G. Georgiev, and M. M. Erokhin. SV40 transcription terminators stabilize the activity of regulatory elements in Drosophila melanogaster. *Doklady Biochemistry and Biophysics*, 463(1):251–254, Jul 2015.

[15] K. K. Chow, M. W. Budde, A. A. Granados, M. Cabrera, S. Yoon, S. Cho, T. Huang, N. Koulena, K. L. Frieda, L. Cai, C. Lois, and M. B. Elowitz. Imaging cell lineage with a synthetic digital recording system. *Science*, 372(6538):eabb3099, 2021.

[16] J. M. Coffin and H. Fan. The Discovery of Reverse Transcriptase. *Annual Review of Virology*, 3(1):29–51, Sep 2016.

[17] N. L. Craig. The mechanism of conservative site-specific recombination. *Annual Review of Genetics*, 22(1):77–105, 1988. PMID: 3071260.

[18] J. A. Doudna and E. Charpentier. The new frontier of genome engineering with CRISPR-Cas9. *Science*, 346(6213):1258096–1258096, Nov 2014.

[19] M. G. Durrant, A. Fanton, J. Tycko, M. Hinks, S. S. Chandrasekaran, N. T. Perry, J. Schaepe, P. P. Du, P. Lotfy, M. C. Bassik, L. Bintu, A. S. Bhatt, and P. D. Hsu. Large-scale discovery of recombinases for integrating dna into the human genome. *bioRxiv*, 2021.

[20] J. A. Escudero, C. Loot, A. Nivina, and D. Mazel. The integron: Adaptation on demand. *Microbiol. Spectr.*, 3(2):MDNA3–0019–2014, April 2015.

[21] F. Farzadfard and T. K. Lu. Emerging applications for DNA writers and molecular recorders. *Science*, 361(6405):870–875, 2018.

[22] K. L. Frieda, J. M. Linton, S. Hormoz, J. Choi, K. K. Chow, Z. S. Singer, M. W. Budde, M. B. Elowitz, and L. Cai. Synthetic recording and in situ readout of lineage information in single cells. *Nature*, 541(7635):107–111, 2017.

[23] H. G. Garcia, P. Grayson, L. Han, M. Inamdar, J. Kondev, P. C. Nelson, R. Phillips, J. Widom, and P. A. Wiggins. Biological consequences of tightly bent DNA: the other life of a macromolecular celebrity. *Biopolymers*, 85(2):115–130, February 2007.

[24] P. Ghosh, L. A. Bibb, and G. F. Hatfull. Two-step site selection for serine-integrase-mediated excision: DNA-directed integrase conformation and central dinucleotide proofreading. *Proceedings of the National Academy of Sciences*, 105(9):3238–3243, 2008.

[25] P. Ghosh, N. R. Pannunzio, G. F. Hatfull, and M. Gottesman. Synapsis in phage Bxb1 integration: Selection mechanism for the correct pair of recombination sites. *Journal of Molecular Biology*, 349(2):331–348, 2005.

[26] L. A. Gilbert, M. H. Larson, L. Morsut, Z. Liu, G. A. Brar, S. E. Torres, N. Stern-Ginossar, O. Brandman, E. H. Whitehead, J. A. Doudna, W. a. Lim, J. S. Weissman, and L. S. Qi. XCRISPR-mediated modular RNA-guided regulation of transcription in eukaryotes. *Cell*, 154(2):442–451, 2013.

[27] J. P. Guilinger, D. B. Thompson, and D. R. Liu. Fusion of catalytically inactive Cas9 to FokI nuclease improves the specificity of genome modification. *Nature biotechnology*, 32(6):577–82, 2014.

[28] S. Guiziou, P. Mayonove, and J. Bonnet. Hierarchical composition of reliable recombinase logic devices. *Nature Communications*, 10(1):456, Dec 2019.

[29] R. M. Hall and C. M. Collis. Antibiotic resistance in gram-negative bacteria: the role of gene cassettes and integrons. 1(2):109–119, January 1998.

[30] A. D. Halleran, A. Swaminathan, and R. M Murray. Single Day Construction of Multigene Circuits with 3G Assembly. *ACS Synthetic Biology*, 7:1477–1480, 2018.

[31] V. Hsiao, Y. Hori, P. W. K. Rothemund, and R. M. Murray. A population-based temporal logic gate for timing and recording chemical events. *Molecular Systems Biology*, 557:1–17, 2016.

[32] E. I. Ioannidi, M. T. N. Yarnall, C. Schmitt-Ulms, R. N. Krajeski, J. Lim, L. Villiger, W. Zhou, K. Jiang, N. Roberts, L. Zhang, C. A. Vakulskas, J. A. Walker, A. P. Kadina, A. E. Zepeda, K. Holden, J. S. Gootenberg, and O. O. Abudayyeh. Drag-and-drop genome insertion without dna cleavage with crispr-directed integrases. *bioRxiv*, 2021.

[33] S. V. Iverson, T. L. Haddock, J. Beal, and D. M. Densmore. CIDAR MoClo: Improved MoClo assembly standard and new e. coli part library enable rapid combinatorial design for synthetic and traditional biology. *ACS Synthetic Biology*, 5(1):99–103, November 2015.

[34] D. L. Jones, P. Leroy, C. Unoson, D. Fange, V. Ćurić, M. J. Lawson, and J. Elf. Kinetics of dCas9 target search in Escherichia coli. *Science*, 357(6358):1420–1424, Sep 2017.

[35] S. E. Klompe, P. L.H. Vo, T. S. Halpin-Healy, and S. H. Sternberg. Transposon-encoded CRISPR–Cas systems direct RNA-guided DNA integration. *Nature*, (March), 2019.

[36] A. C. Komor, K. T. Zhao, M. S. Packer, N. M. Gaudelli, A. L. Waterbury, L. W. Koblan, Y. B. Kim, A. H. Badran, and D. R. Liu. Improved base excision repair inhibition and bacteriophage Mu Gam protein yields C:G-to-T:A base editors with higher efficiency and product purity. *Science Advances*, 3(8):1–10, 2017.

[37] T. Koressaar and M. Remm. Enhancements and modifications of primer design program Primer3. *Bioinformatics*, 23(10):1289–1291, 03 2007.

[38] J. W. Kotula, S. J. Kerns, L. A. Shaket, L. Siraj, J. J. Collins, J. C. Way, and P. A. Silver. Programmable bacteria detect and record an environmental signal in the mammalian gut. *Proceedings of the National Academy of Sciences*, 111(13):4838–4843, apr 2014.

[39] H. Li. Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100, Aug 2017.

[40] J. Li, Y. Zhang, and Y. Zong. A novel design of transcriptional factor-mediated dynamic control of dna recombination. *bioRxiv*, 2020.

[41] M. J. Liao, M. O. Din, L. Tsimring, and J. Hasty. Rock-paper-scissors: Engineered population dynamics increase genetic stability. *Science*, 365(6457):1045–1049, Sep 2019.

[42] A. Lieber, U. Kiessling, and M. Strauss. High level gene expression in mammalian cells by a nuclear T7-phage RNA polymerase. *Nucleic Acids Research*, 17(21):8485–8493, 1989.

[43] J. Livet, T. A. Weissman, H. Kang, R. W. Draft, J. Lu, R. A. Bennis, J. R. Sanes, and J. W. Lichtman. Transgenic strategies for combinatorial expression of fluorescent proteins in the nervous system. *Nature*, 450(7166):56–62, November 2007.

[44] R. Lutz and H. Bujard. Independent and tight regulation of transcriptional units in escherichia coli via the LacR/O, the TetR/O and AraC/I1-I2 regulatory elements. *Nucleic Acids Research*, 25(6):1203–1210, 1997.

[45] D. Mazel, B. Dychinco, V. A. Webb, and J. Davies. A distinctive class of integron in the vibrio cholerae genome. *Science*, 280(5363):605–608, 1998.

[46] A. J. Meyer, T. H. Segall-Shapiro, and C. A. Voigt. Marionette: E. coli containing 12 highly-optimized small molecule sensors. *bioRxiv*, pages 1–27, 2018.

[47] V. K. Mutalik, J. C. Guimaraes, G. Cambray, Colin Lam, M. J. Christoffersen, Q. Mai, A. B. Tran, M. Paull, J. D. Keasling, A. P. Arkin, and D. Endy. Precise and reliable gene expression via standard transcription and translation initiation elements. *Nature Methods*, 10(4):354–360, 2013.

[48] C. J. Myers, N. Barker, K. Jones, H. Kuwahara, C. Madsen, and N. D. Nguyen. iBioSim: a tool for the analysis and design of genetic circuits. *Bioinformatics*, 25(21):2848–2849, 07 2009.

[49] Oxford Nanopore. LSK109 Nanopore protocol. `https://community.nanoporetech.com/protocols/low-input-DNA-PCR-sqk-lsk109/v/lwp_9087_v109_revj_14aug2019`. Online; accessed 13 December 2021.

[50] Oxford Nanopore. RBK110 Nanopore protocol. https://community.nanoporetech.com/protocols/rapid-barcoding-kit-96-sqk-rbk110-96/v/rbk_9126_v110_revg_24mar2021. Online; accessed 13 December 2021.

[51] A. A. Nielsen and C. A. Voigt. Multi-input CRISPR/Cas genetic circuits that interface host regulatory networks. *Molecular Systems Biology*, 10(11):763–763, Nov 2014.

[52] K. Nishida, T. Arazoe, N. Yachie, S. Banno, M. Kakimoto, M. Tabata, M. Mochizuki, A. Miyabe, M. Araki, K. Y. Hara, Z. Shimatani, and A. Kondo. Targeted nucleotide editing using hybrid prokaryotic and vertebrate adaptive immune systems. *Science*, 353(6305), 2016.

[53] A. Nivina, M. S. Grieb, C. Loot, D. Bikard, J. Cury, L. Shehata, J. Bernardes, and D. Mazel. Structure-specific DNA recombination sites: Design, validation, and machine learning–based refinement. 6(30), July 2020.

[54] J. K. Nuñez, A. S.Y. Lee, A. Engelman, and J. A. Doudna. Integrase-mediated spacer acquisition during CRISPR-Cas adaptive immunity. *Nature*, 519(7542):193–198, 2015.

[55] Piotr P. and Zygmunt C. Effect of altered efficiency of the RNA i and RNA II promoters on in vivo replication of ColE1-like plasmids in escherichia coli. *Molecular and General Genetics MGG*, 194(1-2):227–231, April 1984.

[56] D.S. Peabody. The RNA binding site of bacteriophage MS2 coat protein. *The EMBO Journal*, 12(2):595–600, Feb 1993.

[57] V. Pecoraro, K. Zerulla, C. Lange, and J. Soppa. Quantification of ploidy in proteobacteria revealed the existence of monoploid, (mero-)oligoploid and polyploid species. *PloS one*, 6(1):e16392, Jan 2011.

[58] W. Poole, A. Pandey, A. Shur, Z. A. Tuza, and R. M. Murray. Biocrnpyler: Compiling chemical reaction networks from biomolecular parts in diverse contexts. *bioRxiv*, 2020. A.S. contributed code for performing local enumeration, global enumeration, TxTl explorer, Global component enumerator, and plotting functionality. A.S. made figure 5 and associated description.

[59] L. S. Qi, M. H. Larson, L. A. Gilbert, J. A. Doudna, J. S. Weissman, A. P. Arkin, and W. A. Lim. Repurposing CRISPR as an RNA-Guided Platform for Sequence-Specific Control of Gene Expression. *Cell*, 152(5):1173–1183, Feb 2013.

[60] N. Roquet, A. P. Soleimany, A. C. Ferris, S. Aaronson, and T. K. Lu. Synthetic recombinase-based state machines in living cells. *Science*, 353(6297):aad8559–aad8559, 2016.

[61] K. Rutherford, P. Yuan, K. Perry, R. Sharp, and G. P. Van Duyne. Attachment site recognition and regulation of directionality by the serine integrases. *Nucleic Acids Research*, 41(17):8341–8356, July 2013.

[62] M. Sanchez-Contreras, W. D. Bauer, M. Gao, J. B. Robinson, and J. A. Downie. Quorum-sensing regulation in rhizobia and its role in symbiotic interactions with legumes. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 362(1483):1149–1163, March 2007.

[63] R. U. Sheth, S. S. Yim, F. L. Wu, and H. H. Wang. Multiplex recording of cellular events over time on CRISPR biological tape. *Science*, 358(6369):1457–1461, Dec 2017.

[64] S. L. Shipman, J. Nivala, J. D. Macklis, and G. M. Church. Molecular recordings by directed CRISPR spacer acquisition. *Science*, 353(6298):aaf1175–aaf1175, 2016.

[65] S. L. Shipman, J. Nivala, J. D. Macklis, and G. M. Church. CRISPR–cas encoding of a digital movie into the genomes of a population of living bacteria. *Nature*, 547(7663):345–349, Jul 2017.

[66] A. Shur and R. M. Murray. Proof of concept continuous event logging in living cells. *bioRxiv*, 2020. A.S. performed experimental work and wrote manuscript.

[67] S. Singh, P. Ghosh, and G. F. Hatfull. Attachment Site Selection and Identity in Bxb1 Serine Integrase-Mediated Site-Specific Recombination. *PLoS Genetics*, 9(5), 2013.

[68] M. C. M. Smith. Conservative site-specific recombination. In *Encyclopedia of Biological Chemistry*, pages 555–561. Elsevier, 2013.

[69] E. T. Somogyi, J. Bouteiller, J. A. Glazier, M. König, J. K. Medley, M. H. Swat, and H. M. Sauro. libroadrunner: a high performance sbml simulation and analysis library. *Bioinformatics*, 31(20):3315–3321, 06 2015.

[70] F. St-Pierre, L. Cui, D. G. Priest, D. Endy, I. B. Dodd, and K. E. Shearwin. One-step cloning and chromosomal integration of DNA. *ACS Synthetic Biology*, 2(9):537–541, 2013.

[71] N. Sternberg and D. Hamilton. Bacteriophage p1 site-specific recombination: I. recombination between loxp sites. *Journal of Molecular Biology*, 150(4):467–486, 1981.

[72] D. Sun, K. Jeannot, Y. Xiao, and C. W. Knapp. Editorial: Horizontal gene transfer mediated bacterial antibiotic resistance. *Frontiers in Microbiology*, 10:1933, 2019.

[73] Z. Z. Sun, E. Yeung, C. A. Hayes, V. Noireaux, and R. M. Murray. Linear DNA for Rapid Prototyping of Synthetic Biological Circuits in an Escherichia coli Based TX-TL Cell-Free System. *ACS Synthetic Biology*, 3(6):387–397, Jun 2014.

[74] A. Swaminathan, W. Poole, V. Hsiao, and R. M. Murray. Fast and flexible simulation and parameter estimation for synthetic biology using bioscrape. *bioRxiv*, 2019.

[75] W. Tang and D. R. Liu. Rewritable multi-event analog recording in bacterial and mammalian cells. *Science*, 10(February):eaap8992, 2018.

[76] R. Tecon and J. R. Van Der Meer. Bacterial biosensors for measuring availability of environmental pollutants. *Sensors*, 8(7):4062–4080, 2008.

[77] J. P. Torella, F. Lienert, C. R. Boehm, J. Chen, J. C. Way, and P. A. Silver. Unique nucleotide sequence–guided assembly of repetitive DNA parts for synthetic biology applications. *Nature Protocols*, 9(9):2075–2089, August 2014.

[78] C. Van Der Wal and S. Y. W. Ho. Molecular clock. In Shoba Ranganathan, Michael Gribskov, Kenta Nakai, and Christian Schönbach, editors, *Encyclopedia of Bioinformatics and Computational Biology*, pages 719–726. Academic Press, Oxford, 2019.

[79] C. J. H. von Wintersdorff, J. Penders, J. M. van Niekerk, N. D. Mills, S. Majumder, L. B. van Alphen, P. H. M. Savelkoul, and P. F. G. Wolffs. Dissemination of antimicrobial resistance in microbial ecosystems through horizontal gene transfer. *Frontiers in Microbiology*, 7:173, 2016.

[80] J. Wong Ng, D. Chatenay, J. Robert, and M. G. Poirier. Plasmid copy number noise in monoclonal populations of bacteria. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 81(1):1–8, 2010.

[81] J. Xu and Y. Zhu. A rapid in vitro method to flip back the double-floxed inverted open reading frame in a plasmid. *BMC Biotechnology*, 18(1), August 2018.

[82] L. Yang, A. A. Nielsen, J. Fernandez-Rodriguez, C. J. McClune, M. T. Laub, T. K. Lu, and C. A. Voigt. Permanent genetic memory with >1-byte capacity. *Nat Methods*, 11(12):1261–1266, Dec 2014.

[83] E. Yeung, A. J. Dy, K. B. Martin, A. H. Ng, D. Del Vecchio, J. L. Beck, J. J. Collins, and R. M. Murray. Biophysical Constraints Arising from Compositional Context in Synthetic Gene Networks. *Cell Systems*, 5(1):11–24.e12, 2017.

[84] Z. Zhang. Cre recombinase-mediated inversion using lox66 and lox71: method to introduce conditional point mutations into the CREB-binding protein. *Nucleic Acids Research*, 30(17):90e–90, September 2002.

Appendix A

SIMULATION CODE

```
class BlockableIntegraseSite(IntegraseSite):
    def __init__(self,name,blocker,site_type="attB",\
        integrase="int1",dinucleotide=1,\
            no_stop_codons=None,integrase_binding=True,\
            **keywords):
        IntegraseSite.__init__(self,name,site_type,\
            integrase,dinucleotide,no_stop_codons,\
            integrase_binding,**keywords)
        self.blocker = Component.set_species(blocker)
        if(integrase_binding):
            self.binders += [self.blocker]
        else:
            self.binders = [self.blocker]
    def update_species(self):
        return DNABindingSite.update_species(self)
    def update_reactions(self):
        int_rxns = IntegraseSite.update_reactions(self)
        if(not self.integrase_binding):
            int_rxns += DNABindingSite.update_reactions(self)
        return int_rxns
```

Listing A.1: Blockable integrase site part definition

```
    #species used to represent "events"
    e1spec = Species("event1",material_type="protein")
    e2spec = Species("event2",material_type="protein")
    #genetic parts
    gen_ori = Origin("genome")
    gen_ori.attributes = ["no_inter"] #no inter-
             #molecular reactions with this part
    bc1 = UserDefined("bc1")
    bc2 = UserDefined("bc2")
    plas_ori = Origin("plas")
    attP1 = BlockableIntegraseSite("attP1",e1spec,\
                        "attP",integrase="Bxb1",\
                        integrase_binding=False)
    attP2 = BlockableIntegraseSite("attP2",e2spec,\
                        "attP",integrase="Bxb1",\
```

```
                        integrase_binding=False)
attP1.add_mechanism(EnzymeIntegration( \
                    integrase="Bxb1"), \
                    "integration",overwrite=True)
attP2.add_mechanism(EnzymeIntegration(\
                    integrase="Bxb1"),\
                    "integration",overwrite=True)
attB = IntegraseSite("attB","attB",\
                    integrase="Bxb1",\
                    integrase_binding=False)
attB.add_mechanism(EnzymeIntegration(\
                    integrase="Bxb1"),\
                    "integration",overwrite=True)
bxb1_mechanism = IntegraseRule("Bxb1",\
                    reactions={("attB","attP"):\
                    "attL",("attP","attB"):"attR"})
bxb1 = Integrase_Enumerator("Bxb1",\
            int_mechanisms={"Bxb1":bxb1_mechanism})
#constructs below
plas1 = DNA_construct([attB,bc2,attP1,\
                    plas_ori],circular=True)
plas2 = DNA_construct([attB,bc1,attP2,\
                    plas_ori],circular=True)
genome = DNA_construct([attB,gen_ori])
```

Listing A.2: Code used to generate genetic parts

```
#dummy species to allow "degrading" events
e1deg = Species("event1deg",material_type="protein")
e2deg = Species("event2deg",material_type="protein")
deg1 = Reaction.from_massaction([e1spec,e1deg],\
            [e1deg], k_forward=parameters["kdeg"])
deg2 = Reaction.from_massaction([e2spec,e2deg],\
            [e2deg], k_forward=parameters["kdeg"])

genome_mixture = TxTlExtract(name = "txtl", \
            parameters = parameters,\
        components = [plas1,plas2, genome],\
        global_component_enumerators=[bxb1],\
        global_recursion_depth=3)
genome_CRN,genome_components = \
        genome_mixture.compile_crn(\
        return_enumerated_components=True)

```

```
    plasonly_mixture = TxTlExtract(name = "txtl", \
                parameters = parameters,\
                components = [plas1,plas2],\
            global_component_enumerators=[bxb1],\
            global_recursion_depth=3)
    plasonly_CRN,plasmid_components = \
        plasonly_mixture.compile_crn(\
        return_enumerated_components=True)
```

Listing A.3: Code used to generate CRNs

```
plas_ori = Origin("plas")
plas_ori.attributes = ["no_inter"]

bxb1_nodel_mechanism = IntegraseRule("Bxb1",\
    reactions={("attB","attP"):"attL",\
    ("attP","attB"):"attR"},\
    allow_deletion=False)

bxb1_nodel = Integrase_Enumerator("Bxb1",\
    int_mechanisms={"Bxb1":bxb1_nodel_mechanism})
plas = DNA_construct([attB,attP,plas_ori],\
    circular=True)

genome = DNA_construct([attB,gen_ori])

myMixture_nodel = TxTlExtract(name = "txtl",\
    parameters = parameters, components = \
    [plas,genome],global_component_enumerators=\
    [bxb1_nodel],global_recursion_depth=8)
myCRN_nodel,components = \
    myMixture_nodel.compile_crn(\
    return_enumerated_components=True)

myMixture = TxTlExtract(name = "txtl", \
    parameters = parameters, components = \
    [plas,genome],global_component_enumerators=\
    [bxb1],global_recursion_depth=recursion_depth)

myCRN,components = \
    myMixture.compile_crn(\
    return_enumerated_components=True)
```

Listing A.4: Code used to generate CRNs missing some integrase reactions

Appendix B

SEQUENCES

Guide RNA sequences		
	Sequence	**Figure**
L2	ACCAGACAAACCACGACATT	2.12
B1	GGCCGGCTTGTCGACGACGG	2.12
M1	GTTTGTCTGGTCAACCACCG	2.12
M2	GTCAACCACCGCGGTCTCAG	2.12
L1	CTCATGGTTCGTGGTTTGTC	2.12
R1	TACAAACCCAAGCTCCAACA	2.12
B1	GGCCGGCTTGTCGACGACGG	2.12
sG0	CGGTGGTTGACCAGACAAAC	2.13
sG3	GGTTGACCAGACAAACCGGT	2.13
sG11	AGACAAACCGGTTGGCCTAC	2.13
sG17	ACCGGTTGGCCTACTGGCCT	2.13
sG25	GCCTACTGGCCTAGGTATTT	2.13
sG-1	ACCTAGGCCAGTAGGCCAAC	2.13
sG-9	GCCTAAATACCTAGGCCAGT	2.13
sG-17	CGCGATGAGCCTAAATACCT	2.13

Guide RNA sequences (continued)		
	Sequence	**Figure**
oG0	AGACAAACCTCACTCGATCA	2.14
oG1	AGACAAACCTAGATTGGGCA	2.14
oG2	AGACAAACCAGCCCCCCTAC	2.14
oG4	AGACAAACCGTTTTGGGGCC	2.14
oG5	AGACAAACCGGCCAAAATAC	2.14
oG6	AGACAAACCTTTCGCGCAAA	2.14
oG7	AGACAAACCTTGCAGCTGAC	2.14
oG8	AGACAAACCACACGACTTGA	2.14
oG9	AGACAAACCCGACGGGGACG	2.14
g1	AGACAAACCTCACTCGATCA	2.15-2.22
g2	AGACAAACCGTTTTGGGGCC	2.14-2.22

Other Sequences	
Name	**Sequence**
attP	GGTTTGTCTGGTCAACCACCGCGTGCTCAGTGGTGTACGGTACAAACC
attB	GGCTTGTCGACGACGGCGTGCTCCGTCGTCAGGATCAT
attL	GGCTTGTCGACGACGGCGTGCTCAGTGGTGTACGGTACAAACC
attR	GGTTTGTCTGGTCAACCACCGCGTGCTCCGTCGTCAGGATCAT
gRNA scaffold	GTTTTAGAGCTAGAAATAGCAAGTTAAAATAAGGCTAGTCCGTTATCAACTTGAAAAAGTGGCACCGAGTCGGTGC
RBS library	GGGTCTCA(TACT)GAAAGANNNGANNNACTA(AATG)AGAGACCCATGACG
overlap experiment gRNA landing pad	GCTTAATGGAGGACCGCGATGAGCCTAAATACCTAGGCCAGTAGGCCAACCGGTTTGTCTGGTCAACCACCGCGTGCTCAGTGGTGTACGGTACAAACC
ULCCole1	TTGAGATCCTTTTTTTCTGCCCGTAATCTGCTGCTTGCAAACAAAAAAACCACCGCTACCAGCGGTGGTTTGTTTGCCGGACCAAGAGCTACCAACTCTTTTTCCGAAGGTAACTGGCTTCAGCAGAGCGCAGATACCAAATACTGTCCTTCTAGTGTAGCCGTAGTTAGGCCACCACTTCAAGAACTCTGTAGCACCGCCTACATACCTCGCTCTGCTAATCCTGTTACCAGTGGCTGCTGCCAGTGGCGATAAGTCGTGTCTTACCGGGCTGGACTCAAGACGATAGTTACCGGATAAGGCGCAGCGGTCGGGCTGAACGGGGGGTTCGTGCACACAGCCCAGCTTGGAGCGAACGACCTACACCGAACTGAGATACCTACAGCGTGAGCTATGAGAAAGCGCCACGCTTCCCGAAGGGAGAAAGGCGGACATGTATCCGGTAAGCGGCAGGGTCGGAACAGGAGAGCGCACGAGGGAGCTTCCAGGGGGAAACGCCTGGTATCTTTATAGTCCTGTCGGGTTTCGCCACCTCTGACCTGAGCGTCGATTTTTGTGATGCTCGTCAGGGGGGCGGAGCCTATGGAAAAACGCCAGCAACGCGGCCTTTTTACGGTT

Table B.1: Various important sequences

Appendix C

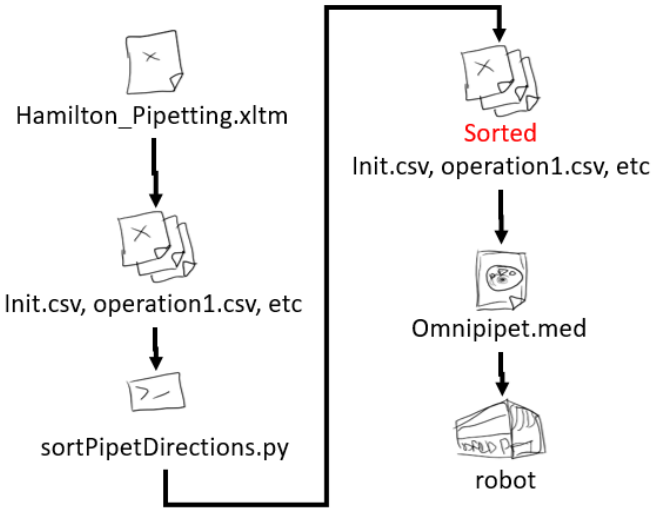
HAMILTON PROGRAM

Andrey Shur	Created	7/17/17	Updated	11/22/21

Using the Omnipipet method

Abstract

The omnipipet.med file is designed to make Hamilton Starlet programming as easy as editing an excel spreadsheet. The Hamilton program reads through a master csv file called the “Init” file, which points to other csv files that contain specific pipetting commands, or tells it to load up a biotek program. A macro-enabled excel template has been built to streamline programming the Hamilton using this interface. The omnipipet.med program supports a small subset of commands from all the possible things the Hamilton could do: pipetting, aliquotting (from one tube into many), mixing, placing into and taking out of the biotek, and running a python script. More commands can be implemented fairly easily, but so far the omnipipet method does not support: pooling (from many tubes into one), rapid pipetting, putting different liquids in the same tip, and other advanced Hamilton features.

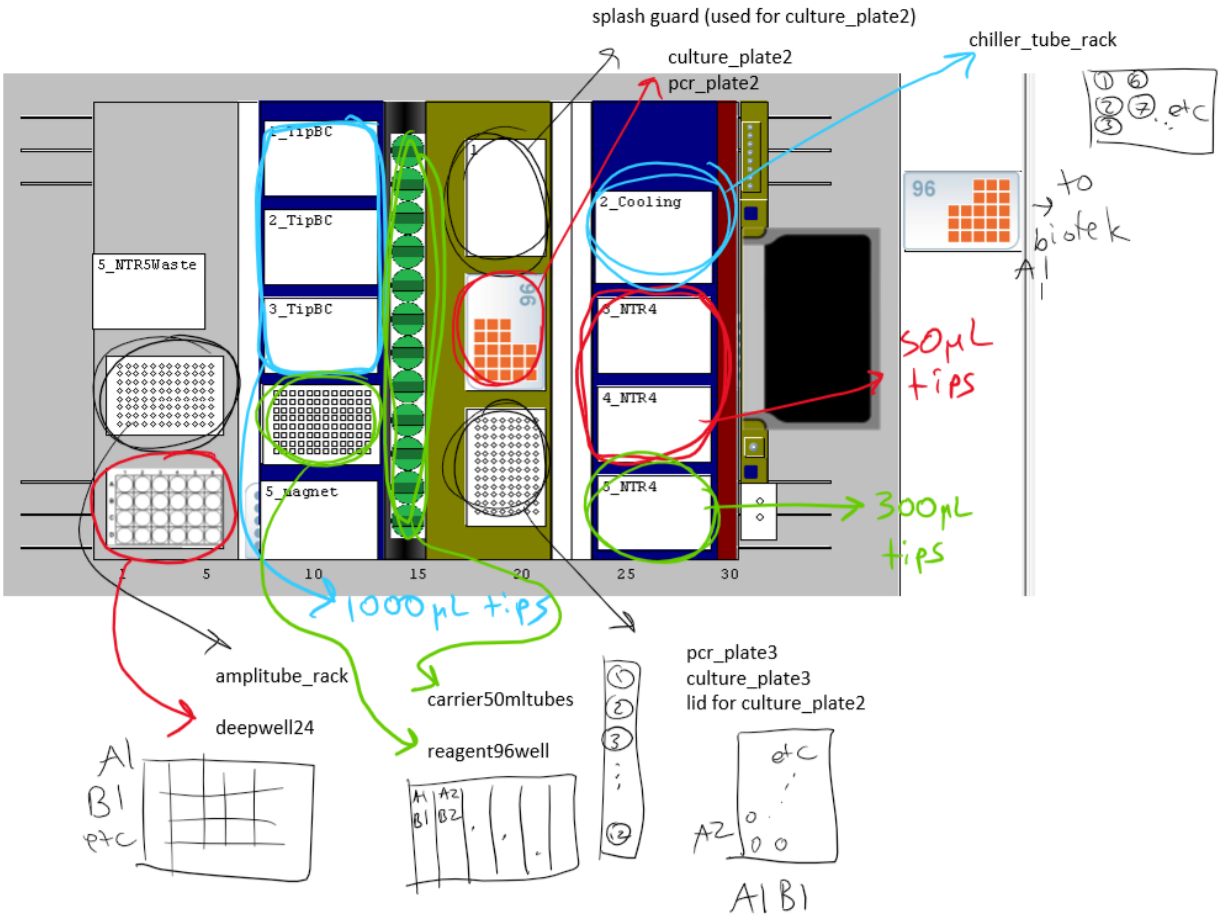


Schematic

The schematic of files generated and used by the **omnipipet.med** program. First the user makes their pipetting protocol using the macro-enabled template **Hamilton_pipetting.xltm.** After the user has finished editing the protocol, they will export the contents of the template into a set of .csv files, headed by Init.csv. When the Hamilton program is run, it calls **sortPipetDirections.py** which is a python script that tidies up those csv files and sorts the pipetting operations to make them more efficient. Finally, the **omnipipet.med** program will read the .csv files in the order denoted in **Init.csv**, which results in the robot moving and pipetting liquid.

Understanding the Hamilton

The Hamilton has pre-defined locations for a variety of useful *labwares*. For example a 96 well plate is a type of labware called a *rack*, and it is full of 96 wells which are called *containers*. This 96 well *rack* can be placed on a few different *carriers*, which are the larger removable units slotted inside the Hamilton's working volume. All of this is represented as a **layout** file. Within the layout file associated with the omnipipet method, the following *labwares* are defined:



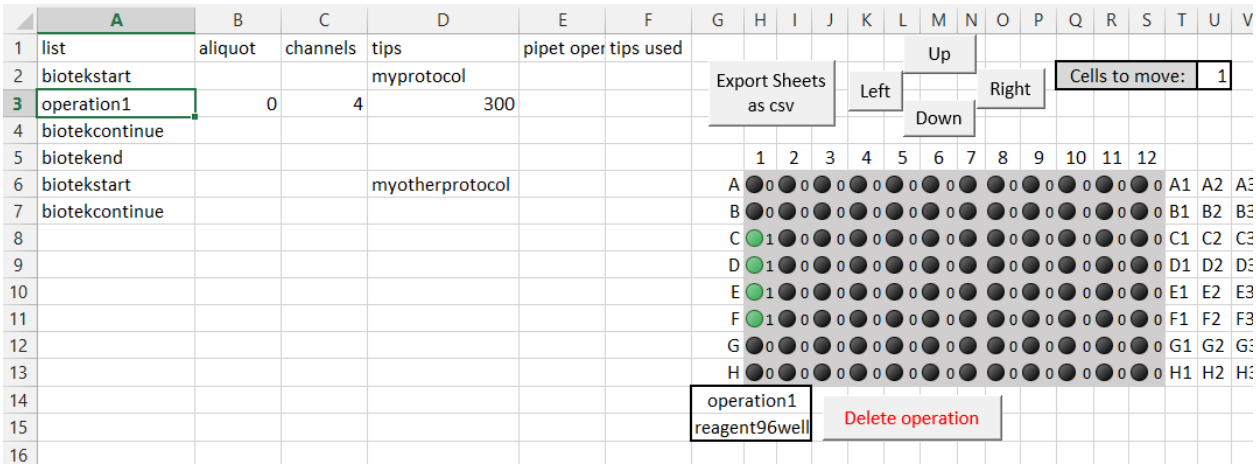
labware	wells	Description
pcr_plate{2,3}	96; A1-H12	120 ul max volume PCR plate
culture_plate{2,3}	96; A1-H12	200 ul max volume round well matriplate. If you are using culture_plate2, the machine will automatically remove the lid and place the splash guard on the plate before pipetting into it.

deepwell24	24; A1-D6	8mL max volume deep well 24 well plate
amplitube_rack	96; A1-H12	Rack for putting PCR tubes. Make sure your tubes are open!!
reagent96well	96; A1-H12	This is intended for use with the "reservoir" racks that have connected wells. We have racks with wells connected in columns (A1, B1, C1 are the same liquid) as well as rows (A1,A2,A3 are the same liquid). Each "well" holds about 5 ml
chiller_tube_rack	25; 1-25	This is a 1.7 mL Eppendorf tube rack. The tubes are numbered in columns, so 1-5 is the first column, 6-10 is the second column, etc. This rack can refrigerate down to 4 C. It must be plugged in first in the bottom left of the robot, and then the temperature is set with the scroll wheel on the rack itself.
carrier50mltubes	12; 1-12	This is a carrier which holds 50ml falcon tubes. The tubes are numbered 1-12, with 12 being close to the front of the machine. These containers also have lids, and the tabs must all be oriented horizontally. You don't have to use the lids, but the machine will try to remove lids anyway.

Biotek Commands

The Biotek protocols are stored in murraylab/ashur/Hamilton/readfile/biotekProtocols

In general the biotek commands are used by typing them into the “list” column in the Hamilton_pipetting template. Place the name of the biotek file in the “tips” column.



An example protocol that uses the biotek commands. First, you start a biotek program called “myprotocol.prt”, which is located in the murraylab/ashur/Hamilton/readfile/biotekProtocols/ folder. Then, you perform a pipetting operation called operation1, which involves adding 2 ul of liquid from pcr tube in position A1 to positions C1, D1, E1, and F1 of “reagent96well”. Then you place the plate into the biotek and run it. You end that protocol, then start and run another protocol called “myotherprotocol.prt”

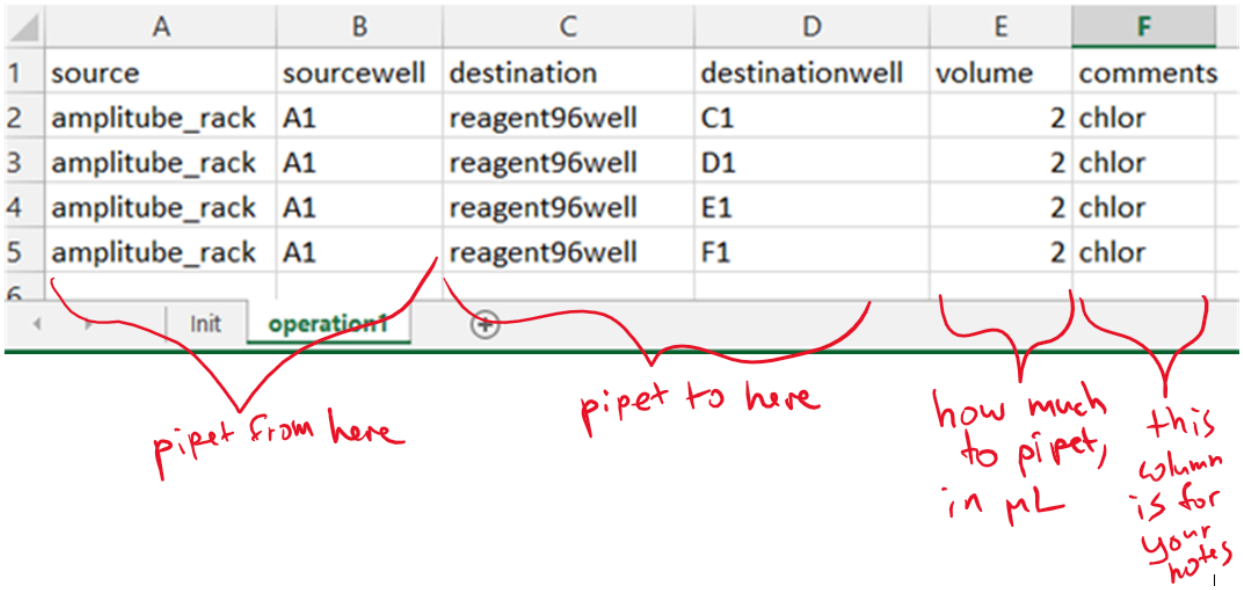
Below is a list of all the possible biotek commands. Remember, the “argument” goes in the “tips” column!

command	Argument	What it does
biotekstart	protocol	Tells the Biotek software to load up the specified protocol, save a new experiment file in murraylab\ashur\hamilton\readfile\plateData\ and keep that experiment open. No physical action is done by the Hamilton! Protocols have to be located in murraylab\ashur\hamilton\readfile\biotekProtocols and must have the extension “.prt” after the text you typed in the “tips” cell.

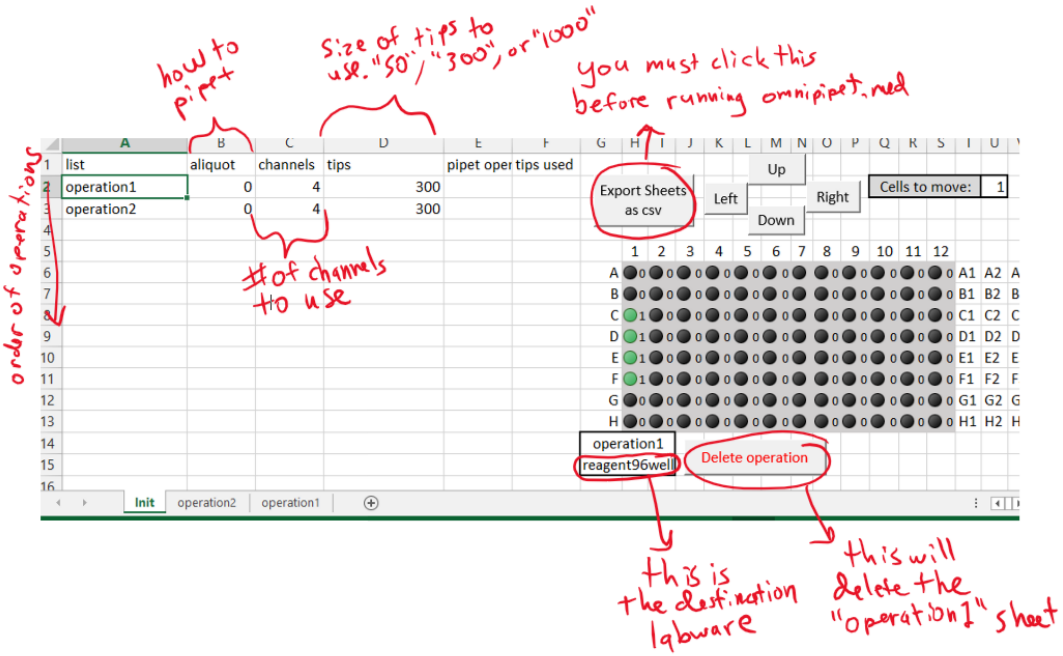
biotekcontinue	none	Tells the biotek to open the plate tray. Places *culture_plate2* into the biotek plate tray. Then starts the protocol, with the specified name of "plate1"/"plate1". If your protocol has the "discontinuous kinetic" option, then this will append data to the file. **Otherwise**, it will **overwrite** the data in your file.
biotekend	none	This tells the biotek to close the currently open experiment. You MUST have this in your method before the next "biotekstart" or else it will throw an error and you will be sad.
biotekloadplate	none	Put a plate from position 2 into the biotek
biotekremoveplate	none	Take a plate from the biotek tray and put it in position 2

Using the Hamilton_pipetting template.

1. Double click on the template, which can be found in murraylab/!ExcelTemplates
2. Click on the operation1 sheet. Here you will define a single pipetting operation.
    a. Define your desired pipetting operation by typing the appropriate source and destination labwares, as well as the desired wells.



3. Click on the Init sheet. When you select the well that says “operation1” you should see the correct wells light up green in the diagram on the right.



a. After the name of your pipetting operation under the “list” column, there are three more important columns.

Column	Accepted values	What they mean
aliquot	“0”,”1”,”1.1”	This column tells the robot how to utilize tips in this protocol. **0**: This means use tips only once, drawing up exactly the volume necessary to pipet, and pipetting it into the liquid, then throwing out the tip. **1**: This means to use a small amount of tips, denoted by the “channels” well, to dispense to a large amount of wells. In this case the tip does not touch the surface of the liquid, but “jets” the liquid into the wells from some height. This only works with more than 10 ul of liquid, and only with the 300 or 1000 ul tips. **1.1**: The same as using “1”, but the tip touches the destination liquid. Useful for

		pipetting 1 ul into a lot of wells, but will probably contaminate the source well.
channels	1-8	This determines how many channels the robot will use if aliquot is not equal to zero. If it is equal to zero, then the Hamilton will use all 8 channels regardless of what you put here.
tips	“50”,”300”,”1000”	There are three different tip sizes. 50 ul, 300 ul, and 1000 ul. Make sure your tip size corresponds with the maximum volume your pipetting operation calls for. If not, then the robot will throw an error in the middle of your run and you will be sad!

4. If you want to duplicate your pipetting operation, you can do so by using the controls in the top right. Type in the correct value in the well that says **Cells to move**, then hit one of the buttons that says **Up** or **Down** or **Left** or **Right**. You should see a new sheet created, where all the wells have been moved over by the correct amount. Double check by selecting the new operation and seeing the wells you want light up.
5. Click “Export Sheets as csv”. This creates a bunch of csv files in murraylab/ashur/Hamilton/readfile/pipetDirections. If this doesn’t work then make sure you are on a windows machine and the lac drive is mounted under Z:\. You can also write the protocol on your own machine, and click this button when you are at the Hamilton computer.
6. Open the Hamilton method editor and open the *omnipipet.med* method.
7. Press the traffic light icon at the top named “run control”
8. Press the green “play” button at the top to run your method!!

Warning: the next few screens will go by quickly so you have to pay attention!

9. The python script will run. This should show a terminal window that stays open for five seconds.
    a. If the terminal window quits quickly, this means an error has happened. Double check that you did everything right in your excel file and export to csv again.

10. The Hamilton will allow you to tell it how many tips there are. Click anywhere except the timer icon, then select how many tips you see in each of the different screens. First it asks for 1000 ul tips, then 50 ul, then 300 ul. You should calculate how many tips your run requires, and make sure there are enough.
    a. If there are not, then the robot will stop and wait for tips. This may happen at any time, especially in the middle of the night.