**Partitioning of a 2-bit hash function across 65 communicating cells**

Jai Padmakumar1 and Christopher A. Voigt1, \*

1 Synthetic Biology Center, Department of Biological Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA

Potential reviewers (round 1):

Round 2:

\*Correspondence and request for materials should be addressed to C.A.V. (cavoigt@gmail.com)

**Key words:** Synthetic Biology, Systems Biology, genetic circuit design automation, cryptography, bitcoin, blockchain, computer aided design, Boolean logic

**Abstract**

Cell populations can perform powerful distributed computing. Here, we introduce a paradigm for dividing a large circuit across cells as well as the extensible sensors, logic gates, and cell-cell communication channels required to reconstruct the subcircuits in their genomes. These tools were demonstrated by re-coding a hash function that performs one iteration towards generating a 2-bit hash value. This algorithm is related to MD5, an early predecessor to the cryptographic functions underlying cryptocurrency. Implementation requires 110 logic gates, which we partition to 65 strains of *Escherichia coli*, requiring a total of ?? bp of synthetic DNA be integrated into their genomes. The strains are experimentally verified to be able to integrate their assigned input signals, process this information correctly, and propagate the result to the cell in the next layer. This work captures a major step towards harnessing the computational potential of living cells, whether it is to obtain programmable control of biological processes or to implement highly-parallelized algorithms to identify solutions to hard computational problems.

**Introduction**

The complexity of the natural world, from the development of body plans to the computational power of the brain, arise from distributed computation performed by many communicating cells, each implementing relatively simple operations [CITE]. If this computational capacity can be harnessed, cell populations will have the potential to solve hard, energy intensive problems that are highly parallelizable [CITE]. One example is the hash algorithm, used in cryptography and made famous by cryptocurrencies, that is used to render data unreadable by mapping data of arbitrary size (“a message”) to a bit array of fixed size (the “hash”). Currently, to mine a bitcoin, the SH256 hash algorithm is used and only approximately one out of 1020 independent hashes will be successful, requiring repetitive independent calculations [CITE]. Beyond solving computational problems, fully realizing the potential of engineered biology will require programming cell communities to collectively compute or coordinate their actions.

Single cells can be programmed to execute a desired function using genetic circuits [CITE]. They require the careful balancing of interacting regulators and the assembly of many DNA parts. The Cello automation software simplifies circuit design, where a user specifies the function using a high-level language (Verilog) that is mapped to a DNA sequence [CITE]. Logic minimization algorithms deconstruct the circuit into gates and assigns regulators to each gate from a pre-characterized set. The size of a circuit that can be put into one cell is limited. One reason is that gate functions are performed by freely diffusing molecules that can cross-react with each other’s targets, leading to efforts to identify “orthogonal” sets [CITE]. Second, the expression of many regulatory genes metabolically burdens the cell, and this leads to growth defects, failed predictions and evolutionary breakage [CITE]. In practice, these issues have limited the circuit size to 12 regulatory proteins [CITE]. Methods to reduce burden include the use of genomic landing pads, rather than plasmids, and borrowing paradigms from control theory [CITE].

Genetic NOT/NOR gates are common because they are easy to encode in DNA and can be used to build more complex functions [CITE]. They require a single repressor that can be implemented using proteins or RNA, orthogonal libraries of which enable a circuit to be built by connecting gates [CITE]. However, these classes of regulators have various problems in their use, such as the presence of ligand-binding domains, large operators, off-target interactions, high required expression, interference with host gene expression, repetitive domain structure and resource sharing [CITE]. In contrast, phage genomes encode small repressors, such as the well-studied CI repressor, that has been used in many early synthetic biology projects [CITE]. They are small, bind tightly to compact and well-defined DNA operators and do not interfere with host expression. Repressors from different phage tend to be orthogonal [CITE] and directed evolution has been shown to be able to diversify CI to bind to different operators [CITE]. To date, there has not been an effort to build a library of gates based on diverse phage repressors.

Distributed computing is a powerful approach to problem solving, where multiple cells collaborate by communicating the states of their circuits [CITE]. Communication is performed by chemical signals, where the circuit output of one cell is transmitted by a “sender device” (enzyme that makes the chemical) and the next cell responds through a “receiver device” (regulator that is activated by chemical) that serves as a circuit input [CITE]. Communicating cells that individually perform simple logic has led to a wealth of computational functions, including edge detection and pattern formation [xxx]. Distributed computing can also be used to divide a circuit too large for a single cell across multiple cells, which reduces the burden on any single cell, allows cells to be reused in different circuits without genetic manipulation, and improves robustness by requiring consensus. Early in the field, circuit size was highly constrained, so in 2011, we designed strains of *E. coli* to carry single NOR gates that communicate to produce all 2-input Boolean logic [CITE. Distributed computing has also been applied to build a 2x2 maze-solving circuit (6 *E. coli* strains with single AND gates)[CITE], multiplexors (*e.g*., 12 gates distributed across 2 *E. coli* strains)[CITE], bistable memory (e.g., 5 *S. cerevisiae* containing single gates)[CITE], a comparator (10 *S. cerevisiae* strains containing single gates) [CITE], a band-stop filter (5 *S. cerevisiae* strains containing single gates)[CITE], and adders (*e.g*., 22 gates distributed across 9 mammalian strains) [CITE]. A hard constraint in these systems is the number of orthogonal communication signals, which after considerable effort has been expanded to 4 through directed evolution and repurposing secondary metabolism [CITE]. These circuits were sufficiently small where the gate partitioning could be performed easily by hand.

In electronic circuit design, a common task is to divide a circuit into subcircuits; for example, to distribute circuits too large for one chassis (modules, chips or boards) across multiple chassis [CITE]. Partitioning algorithms convert the circuit into a graph and divide the node (gates) across a fixed number of chassis while minimizing the edges (wires) spanning chassis [CITE]. Many variations of these algorithms and corresponding software tools have been developed, but a shared feature is that they keep the number of systems fixed [CITE]. In contrast, when dividing a large genetic circuit across cells, the gates per cell and number of signaling molecules are hard constraints, whereas the number of cells (chassis) is variable. This constraint is so profound that it skews which two-input gates are preferred for biological circuit design [CITE]. To this end, we have developed partitioning algorithms that constrain the number of gates per cell and the number of cell-cell communication signals, while allowing the number of cells to float [CITE].

In this manuscript, we demonstrate the partitioning of a hash function into individual computing units that are encoded within *E. coli* genomes. The 128-bit MD5 (“message digest”) hash algorithm is a predecessor to SH256 that is not suitable for cryptographic functions, but is commonly applied to other problems, such as verifying the integrity of data after transfer [CITE]. Here, we use a version that [does what?] and produces a 2-bit hash. We write this algorithm as a 16-input 2-output circuit consisting of ?? NOR/NOT gates. Computational tools are applied to partition these gates into ?? cells, conforming to constraints on the gates per cell and the signaling molecules to facilitate the propagation of a signal. To build the circuits, we mine ?? phage repressors from the literature and characterize them as gates, use the 12 inducible systems from *E. coli* Marionette [CITE] and the OC6-AHL, pC-HSL and DAPG sender:receiver devices [CITE]. The circuits are computationally designed and encoded in the genome of each *E. coli* strain, requiring up to ?? recombinant genes and ?? kb DNA in a single cell. Collectively, this requires 0.? Mb of synthetic DNA to be introduced into the strains, which is on the scale of small bacterial genomes [CITE]. The strains are constructed, and the circuit functions are verified, both within each cell and that the information can propagate between pairs of cells.

<https://crypto.stackexchange.com/questions/18612/how-is-sha1-different-from-md5?rq=1>

(^use for context relating to current algorithms?)

**Results**

Circuit partitioning

Implementation of a very large circuit in living cells requires dividing the circuit across a number of communicating cells, where each cell performs a small portion of the full circuit calculation. To implement a circuit in a cell, genetic logic gates are implemented using transcriptional repressors. Cell-cell communication can be achieved using by converting a transcriptional output in a “sender” cell into expression of a diffusible small molecule that can subsequently be sensed in one cell and converted to a transcriptional output by a “receiver” cell. Since we are limited in the number of both logic gates that can be placed into a single cell and the number of cell-cell communication channels available, each subcircuit must fit into acceptable design parameters. Once a circuit is divided into smaller subcircuits, the inputs and outputs to each cell must be assigned to orthogonal cell-cell communication channels. Here we designed a partitioning strategy to divide a large circuit into smaller subcircuits and assign cell-cell communication channels given: 1) the maximum number of gates that can be placed into a single cell, 2) the maximum number of cell-cell communication channels that can be placed into a single cell, and 3) the total number of cell-cell communication channels available.

To generate the subcircuits, we divided the larger circuit into smaller subcircuits that contain at most the maximum number of gates per cell specified by the user (eight in our case) and at most the maximum number cell-cell communications channels per cell (three in our case). The number of cell-cell communication channels required for a given subcircuit is equivalent to the number wires entering and exiting a given subcircuit (i.e. those spanning 2 or more cells). First, a gate containing an output is selected. If there are multiple outputs, a gate is arbitrarily chosen. Upstream gates are then greedily added to a subcircuit until either the maximum number of gates or channels is reached. In general, there are multiple different alternatives that can be taken that satisfy the above criteria and the subcircuit with the most gates is taken. This process is then repeated until all gates are assigned to a subcircuit.

The next step is assigning cell-cell communication channels to each subcircuit. All subcircuits must contain unique channels. In the previous step, all subcircuits were divided such that they required 3 or less cell-cell communication signals.. To assign cell-cell communication channels a cell requiring no quorum outputs is chosen and channels are randomly assigned to the (incoming) wires of that cell. Next an upstream cell is chosen and wires are assigned to meet the criteria. This is repeated until all wires are assigned or it is no longer possible to assign wires to orthogonal channels. If an assignment is not possible, we can choose a different initial assignment of cell-cell communication channels. If that fails, the subcircuit that cannot be assigned to unique channels is divided into 2 smaller subcircuits [explain in more detail??] and the process is repeated again. Note that despite each individual cell only requiring three communication channels, the circuit as a whole requires four to be implemented (Figure 1).

NOT gate construction with phage repressors

Construction of large circuits requires a set of NOT gates that display high dynamic ranges, similar response functions, low cellular toxicity, and no cross-talk when expressed from the genome. used NOT gates based on to construct circuits with up to 8 gates on plasmids and 4 gates on the genome An initial attempt to move a set of 10 TetR NOT gates from plasmids to the genome ultimately led to a set of 6 NOT gates with variable performance (worst performing gate had a fold change of 9, making it effectively unusable in circuit). TetR proteins are often sensors of various compounds, making them unpredictable in different environments and preventing them from being used as gates in applications requiring their sensor activity. For example, of the 6 TetR NOT gates move to the genome, 5 have known small-molecule regulators, ranging from unsaturated fatty acids to multidrug binding sites. Additionally PhlF (best NOT gate with a dynamic range >500 fold) and BetI, are used as sensors in the Marionette sensor array, a set of 12 high performance, orthogonal sensors all in a single strain previously developed by our lab. Usage of these proteins as NOT gates makes them incompatible with this sensor array. PhlF is particular valuable as it can be used as a cell-cell signalling system (Fig 2D).

To these issues, we curated a set of 12 phage repressors homologous to the classic cI repressor from lambda phage with documented DNA binding sites and repressible constitutive promoters. Due to their function in phage life-cycle, cI-like repressors have features making them ideal for use as NOT gates. They have evolved to repress their cognate promoters efficiently with minimal host burden, are known to be cooperative, and bind to DNA tightly with cI having a kD of ~5nM. These proteins lack small molecule binding sites, making them both highly composable with other work and giving them a higher likelihood of working in different applications and environments. Additionally, these proteins are carried on phages and are not natively found in bacterial genomes. The proteins have similar secondary structures with an N-terminal helix-turn-helix DNA binding domain and (typically) a C-terminal domain that facilitates oligomerization and doubles as a RecA-dependent autocleavage domain (Fig 2A, left side). The autocleavage site is present in a linker region connecting the N and C-terminal domains, functioning as a DNA damage sensor and can be activated upon UV damage to the bacterial genome but this activity can be straightforwardly abolished if desired. 3 repressors in the set (JR8, JR9, JR11) do not contain this domain entirely and are ~100 amino acids in length, approximately half of the size of other cI repressors (Fig 2A, left side). JR10 and JR12 contain C-terminal domains but are not homologous to any PFAM domains. cI repressors bind to single operator sites as dimers and then tetramerize after binding to increase repression by RNA polymerase (RNAP) occlusion. WT phage promoters typically have 1 or more operator sites overlapping the -35 and -10 boxes, increasing repression (Fig 2A, right). Additionally, operator sites are spaced near multiples 10.5bp apart, promoting tetramerization by ensuring individual dimers bind to DNA on the same side.

To build NOT gates, we designed constructs consisting of a PTet-PTac tandem promoter driving expression of a repressor insulated with an upstream ribozyme and a strong downstream terminator. The output promoter drives a standardized YFP expression cassette (Fig 2B, Supplemental circuit datashseets). We then characterize the response function by inducing with IPTG and aTc and using flow cytometry to measure steady-state YFP expression (Fig 2B). The YFP expression cassette was placed upstream of the repressor to alleviate an issue in measuring response functions where transcriptional readthrough of the downstream repressor terminator knocked the repressor off DNA, resulting in a characteristic uptick in the response function at high induction levels (Supplementary Figure 1). The NOT gates were assayed by integrating into the attB2 landing pad of a previously designed *E. coli* MG1655 containing 2 empty genomic landing pads and a sensor array with 7 sensors integrated into a 3rd landing pad (YJP\_MKC173, Fig 2B). [mention upstream spacer sequences and operator spacing for optimal binding]

After assaying NOT gates as described above, the result is a response function where the X-axis in units of inducer concentration and Y-axis is in arbitrary units (AU). In order to use the gates for genetic circuit design, we converted the inputs and outputs into units of RNAP polymerase flux (RNAP/s) so that the gates had standardized input/output units. To convert to RNAP/s, we first measure the response functions in terms of a relative unit of promoter flux (RPUg) determined by calculating the relative fluorescence from a promoter of interest to a standard reference promoter, BBa\_J23101 (materials and methods). The promoter flux from BBa\_J23101 was empirically determined using single cell microscope approaches to be 0.029 RNAP/s/DNA. Multiplying the RPUg value by both the promoter flux and the landing pad copy number yields units of RNAP/s. To convert the input units into RNAP/s, the same method is employed but using separate measurements from a PTet-PTac tandem promoter driving YFP.

After optimizing the repressor RBS (Materials and Methods) and, if necessary, optimizing promoters (data not shown), we obtained a set of 12 NOT gates displaying large dynamic ranges (low, median, high = 24, 76, 144), moderate cooperativities (low, median, high hill coefficient = 1.5, 2.1, 3.2) and no cross-talk (Fig 2C, Supplementary Figures 2 and 3). The WT promoters for repressors JR3, JR4, JR9, and JR12 showed poor performance initially and were replaced by synthetic versions. For PJR3, PJR9, and PJR12, we placed 1 or more operators into the background of a synthetic Anderson promoter, ensuring that at least 1 operator overlapped either the -35 and -10 boxes and that the center to center distance between operators was close to a multiple of 10.5bp (Fig 2A, right). For PJR4, the WT promoter contained only a single operator and we simply placed an additional operator downstream of the -10 box (Fig 2A, right). The gates here showed slightly lower cooperativities overall than the previously designed TetR-based gates on the genome and plasmids (median hill coefficients = 2.6 and 2.7, respectively). Interestingly, much higher cooperativities, with hill coefficients up to 4.4, have been achieved with TetR repressors by fusing an additional C-terminal dimerization domain from cI (or homologous repressors) but were not observed here. In terms of dynamic range, the gates also outperformed a set of orthogonal cI repressors generated using directed evolution (find some numbers for their best case) [cite cI PACE paper]. Additionally, we assessed the relative impact of each repressor on growth by measuring relative OD600 at various repressor induction levels (Supp figures N-Z). Only 3 gates displayed toxicity. Repressors JR10 and JR12 showed ~20% and ~30% decrease in growth at max induction (Supp figure XX) while repressor JR9 showed slight toxicity (~10% decrease in growth at max induction) with the strongest RBS, j9R3 (Supplemental gate datasheets).

Construction of sender cell

Next, we developed a set cell-cell signaling channels, which are required for communication between cells in our distributed computation framework. We required four orthogonal communication channels, each consisting of a defined synthesis pathway expressed in a “sender” converting promoter flux into concentration of a diffusible small molecule and a transcriptional sensor expressed in a “receiver” cell that can convert the small molecule concentration back into promoter flux. To this end, we constructed 4 sender:receiver devices based on four molecules: OC6 (Lux), OHC14 (Cin), pC-HSL (Rpa), and DAPG. These molecules were chosen for their relative ease of production and the availability of high performance sensors within the Marionette sensor array (Supplementary Figures 4 and 5).

Our “sender” devices consisted of an insulated cassette composed an a PTac promoter, ribozyme, one or more synthesis genes, and a downstream terminator (Fig 2D). Our “receiver” cells (rLux, rCin, rRpa, rDAPG) consisted of an inducible promoter responsive to given sender device and output YFP cassette (Fig 2D). To simplify downstream circuit and receiver cell construction, we moved the Marionette sensor array, which already included optimized sensors for OC6, OHC12, and DAPG, into an E. coli MG1655 delta AraC strain containing 3 empty landing pads and placed it outside of the landing pads (JAI\_MKC300), freeing up all the landing pads for integration (Figure 2D, Supplementary Figures 4 and 5, Materials and Methods). Note the RpaR sensor was not in this array and was cloned alongside any constructs that required it (Figure 2D). All sender and receiver constructs were integrated into the genomic attB2 landing pad, except for the DAPG synthesis device which was placed on a p15a plasmid to enable sufficient expression levels (Fig 2D).

To characterize our sender:receiver channels, sender cells were inoculated into M9 media and grown into stationary, induced overnight with varying amounts of IPTG, then diluted and cultured for four more hours before collecting the supernatant (Materials and Methods). The supernatant was then mixed with receiver cells to assay the amount of molecule left in the media and generate a response function for each sender:receiver pair (Materials and Methods).

We were able to generate good response functions for expressing Lux, Cin, Rpa, and DAPG with each channel displaying a >50 fold dynamic range (Figure 2E, Supplementary Figure 6). However, we initially found synthesis of a pC-HSL in M9 to be too low to be usable (Supplementary Figure 7). pC-HSL is synthesized via 3-step pathway where the first step results in conversion of Tyrosine to p-Coumarate. To optimize production, we tested different concentrations of exogenously added p-Coumarate to boost overall yield of pC-HSL (Supplementary Figure 7). Ultimately, we found 100nM of the precursor gave optimal expression and used this concentration in all experiments requiring pC-HSL production (Fig 2E). To synthesize DAPG, we used a previously characterized plasmid with no modification other than swapping the promoter as necessary [cite chunbo lou paper].

binary ss a total of to hash a single 32-bit message. The 64 iterations are fourof 16 iterations (Supplementary Figure 8) The calculations performed in but the size of each input is reduced from 32-bits in the original implementation to 2-bits in our implementation. (note these values must be held constant to generate the same hash from a given input)2 additional parameters that are pre-computed constants, S and T. S and T are tables containing 64 2-bit values, where a particular value is used at each given iteration. , one of the 16 message chunks, and the appropriate values of S and T are passed to a function which performs a series of Boolean computations (Supplementary Figure 8). At the end of the iteration, a 2-bit value is generated and inputted into the subsequent iteration as the B value. The values for A, C, and D are also updated based on their initial values (Supplementary Figure 8). chosen toIf the initial message was 32-bits or less, then the values of A, B, C, D are concatenated together to produce a final 8-bit hash after the 64th iteration. If the initial message was longer than 32-bits, the entire 64 iteration process is repeated with the next portion of the message except the initial values for A, B, C, D are set to final values of from the first 32-bit message.

. outputs one 2-bit value representing the updated B parameter (Supplementary Figure 8)We note that in our implementation in living cells, we test our design by characterizing all possible inputs for each subcircuit, rather than identifying ideal initial values for A, B, C, D, S, and T. thus, we demonstrate our circuit implementation operates successfully regardless of the particular values chosen.

To generate a circuit diagram implementing the hashing algorithm, we used a logic synthesis tool called Yosys to synthesize a circuit from a (behavioral?) Verilog description of the MD5 algorithm. We used Yosys to perform both logic minimization and map the circuit to abstract NOT/NOR gates with individual wires (Materials and Methods). These steps resulted in a 16 input (8 2-bit inputs), 2 output (1 2-bit output) circuit with a total of 131 gates (Supplementary Figure 9). This circuit was then partitioned into 65 subcircuits, each individually implementable in living cells (Figure 1, Supplementary Figure 9, Materials and Methods).

Genetic circuit design and construction

To convert an abstract circuit diagram to a DNA sequence, transcriptional gates must be assigned to each NOT/NOR gate such that their inputs and outputs match to perform the desired circuit function. A previously developed software tool called Cello performs the full stack required to convert a circuit description (in Verilog) to a DNA sequence, including logic synthesis and mapping to NOT/NOR gates, gate assignment, and circuit simulation. The user simply inputs a set of gates with defined response functions, transcriptional sensors that can sense various inputs, and Verilog code. Cello then uses a simulated annealing algorithm to identify an optimal gate assignment by optimizing the circuit’s worst case behavior (specifically, the ratio of the lowest ON state to highest OFF state).

The subcircuits generated by the partitioning stage often connect to more than one cell and therefore require one or more outputs. However, Cello currently does not have the capability to handle multi-output circuits. To map each subcircuit to a DNA sequence, we first wrote descriptions of each subcircuit as structural Verilog. We then ran each subcircuit through Yosys again to further minimize each individual subcircuit and substituted OR gates wherever possible (in a genetic context, OR gates can be implemented without any repressors if they are the final gate in a circuit)[cite Cello, Cello2.0, Jonghyeon paper]. These steps reduced the total number of gates to 110, generating the final fully minimized MD5 circuit implemented in cells (Fig 4a, Supplementary Figure 10). To perform the gate assignment, we use custom Python scripts that implement the same gate assignment algorithm performed by Cello but modified it to handle multi-output circuits (Materials and Methods).

To implement the genetic circuits, we first cloned each circuit into attB2 and attB7 landing pads of *E. coli* MG1655 (JAI\_MKC300). To monitor the performance of the circuit, one or more fluorophores were added to the circuit (Fig 3B, Supplemental Figure Circuit Datasheets). After validating circuit performance, we cloned the cell-cell signalling pathways into the attB5 landing pad under control of the required gate promoter (Figures 2B and 3B). For example, in circuit g6, PJR1 controls both the circuit output YFP and the pC-HSL production cassette (Figures 3B and 3C). In some cases, the orientation of gates was manually changed to reduce effects from transcriptional readthrough. We developed a streamlined protocol to simultaneously perform double integrations into the attB2 and attB7 landing pads directly from an overnight culture by adding a plasmid expressing both the *int2* and *int7* integrases to JAI\_MKC300 (Materials and Methods). Subsequent integrations into the attB5 landing pad were achieved via conjugation and always performed after an integration into attB2 or attB7, enabling selection of transconjugants (Materials and Methods).

The largest subcircuit implemented required a total of 8 gates, 2 inputs, and 3 outputs employing a total of 21 regulatory genes (with total of 30XX something synthetic genes in total) spanning >30kb of synthetic DNA (Figures 3A, 3B, and Supplemental Datasheet). The circuit contains 3 fluorescent proteins to simultaneously measure the promoter activities from the 3 outputs. Initially, we were unable to clone the attB2 payload through plasmids due to toxicity. To circumvent this, we divided the attB2 payload across three plasmids with low copy pSC101 origins. We then performed a four-piece Golden Gate assembly to assemble the three into the attB2 R6K integration vector in a 20uL reaction. We dialyzed the assembly to remove salt and electroporated the full 20uL of assembly into JAI\_MKC300 (Materials and Methods). We then integrated the attB7 payload using our normal protocol (Materials and Methods). We then validated the circuit performed as predicted across all states by flow cytometry (Figure 3C). Next, we integrated the OC6 and pC-HSL synthase genes into the attB5 landing pad (Figure 3B). Finally, we transformed the DAPG production pathway, carried on a p15a plasmid, into the same strain. In adding the DAPG production plasmid to this and other circuits, we discovered context dependent toxicity that caused significant instability leading to mutations in both the circuit and/or DAPG synthase genes. To address this, we lowered the copy number of the DAPG production plasmid by replacing the p15a origin with a pSC101 origin. Remarkably, despite only lowering the copy number ~2x, from 9 (p15a) to 4 (pSC1010), the toxicity was completely alleviated [cite bin’s paper]. Using this strain we proceeded to test the circuit with cell-cell signalling. To characterize the circuit, the strain was first grown overnight with appropriate inducers to get the circuit into each of the 4 states (Materials and Methods). This step was necessary to address issues with timing and hazard states that appear as the circuit computes the correct output. Since the produced molecules accumulate in the media without degradation, any transient promoter activity could cause false outputs. After culturing overnight, the cells were then diluted into a larger volume of media with appropriate inducers and grown for 4 hours before collecting the supernatant (Materials and Methods). The rLux, rRpa, and rDAPG receiver cells were then cultured in the collected supernatant to analyze the cell-cell signalling function (Materials and Methods). The circuit outputs performed as expected across all states (Fig 3C). To the best of our knowledge, this represents the largest transcriptional circuit implemented to date.

To implement the full MD5, we constructed XX unique circuits to distribute 131 gates circuit across 65 cells (Figure 4A, Supplemental circuit datasheets). The circuits themselves all performed as predicted, particularly in the ON states [add pearsons and spearmans correlations] (Figure 3D). Likely due to the small number of sender:receiver channels available, the circuits spanned a wide range of required gates, regulatory proteins, and DNA sizes (Figures 3E and 4A). The largest circuit we had previously built on the genome contained 8 regulatory proteins while the largest we had implemented using plasmids contained 13 regulatory proteins. We constructed a number of circuits with >8 and >13 regulatory proteins, all implemented on the genome. We profiled every cell across all input states and demonstrated that the cell-cell signalling channels worked as expected (Figure 4B).

Chris comments:

Introduce the algorithm more thoroughly when it is first mentioned. What it does, what the inputs outputs are, etc. The different types of computations that you referred to in your email.

They also tend to have to be highly expressed, with up to 103 repressors per cell, and the homologues have highly variable response functions. The classic lutz and bujard inducible systems produce ~3000 LacI tetramers and ~7000 TetR dimers (NAR, 1997, 25: 1997). The binding of TetR to TetO is 6 x 109 M-1 (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC373327/)

CI binds tightly to a short DNA operator sequence, requiring approximately ?? proteins for repression. The KD of CI is 5 nM (Koblan and Ackers, 1991).

Good to compare resonse functions with those published here.

Note TetR use in marionette array with citation as part of justification for phage repressors.

RPU and RPUg – a little intro when describing for the first time. Include conversion to RNAP/s. Cite Drew Endy’s paper, Shaobin’s paper and Yongjin’s paper.

Brodel paper and the avoidance of CI regulators found through directed evolution [CITE]. Leads to non-specific DNA binding that can lead to interference and toxicity. TetR variant evolution by PACE. Also the CI variants are both activators and repressors and we don’t want the activator function.

When describing the selection of the quorum systems, need to be specific regarding why these three. I think many will wonder why we didn’t select others given Alvin’s paper, Chunbo’s paper and the Polizzi paper.

**Discussion**

This work demonstrates the design of a genetically heterogeneous set of cells, where each cell is treated as a computational unit, to perform operations too large to be carried by a single cell. A combination of new software tools, scalable gates and communication signals and genome editing has allowed us to simultaneously push the size of a circuit that can be carried in one cell as well as the number of cells. Here, we divide a circuit requiring ?? NOR gates distributed across ?? cells, with the largest circuit in one cell being ?? kb and ?? regulators. This is a marked improvement over our previous record to build the Texas Instruments SN74LS49 chip controlling the LCD calculator display that required 7 non-interacting strains of *E. coli* with up to 12 regulators (0.074 Mb DNA). The ability to construct larger circuits in cells and to include more cells in a circuit design is increasing rapid due to rapid declines in the costs of DNA synthesis and strain construction (approximately ??-fold faster than Moore’s Law describing transitions per chip) [CITE]. Note that the primary goal of this research is not to beat electronic computers, but rather gain computational control over what biology is able to build and do, as evidenced in the natural world [CITE]. However, as our ability to build circuits that operate in and across cells increases, the question is begged whether there is a point where cellular computing would be able to outperform electronic circuits with respect to certain classes of tasks [CITE].

The first question is how far the computational complexity of an individual cell can be pushed. Because gate functions are performed by molecular interactions (*e.g*., protein binding to DNA) and there is degeneracy in these interactions (e.g., a protein binds to more than one DNA sequence), the number of regulators is constrained to hundreds by bounds imposed by information theory [CITE]. This limit is still out-of-reach because of burdens imposed by recombinant gene expression, although we are in a period of rapid growth because of a better understanding of its origins and mitigation strategies [CITE]. In this work, one cell carries up to ?? recombinant genes (*E. coli* g?) without imposing a large growth defect. However, our use of regulatory proteins – as 2-input NOR gates – is remarkably computationally inefficient [CITE]. The number of required gates is dramatically reduced by expanding the gate types (e.g., AND) and the number of inputs per gate [CITE]. Compressing transcriptional layers by having logic operations performed by regulator binding patterns at a promoter also decreases circuit size; for example, the circuit in *E. coli* g? (Figure 3a) could be compressed from ?? kb to 3 kb. Shifting from digital to analog computing can reduce the number of genetic parts by orders-of-magnitude; for example, only three regulators can perform addition, ratiometric and power-law functions [CITE].

The second question is how many cells can be connected to perform distributed computing. The most difficult step is the passaging of information between cells. Indeed, early justification for encoding single gates in cells is that they could be reconfigured without additional genetic modification, however because of a lack of communication signals, this limits the total circuit size to 4-5 gates [CITE] and we would need ?? orthogonal signals for the hash function described in this manuscript. There is also the problem of how to organize cells in space to reliably and efficiently pass signals. In our hands, when grown in unstructured liquid cultures, the result is very sensitive to initial conditions and fluctuations [CITE]. Microfluidic devices or small chemostats could be used to house individual strains or consortia of a few strains, but they tend to have to be designed on-off for a particular circuit topology, which loses the potential advantages of parallelized cellular computation [CITE]. Signals can be transmitted by colonies on plates [CITE] or as cultures in plate wells separated by permeable walls [CITE]. However, this is spatially constraining, limiting the number of cells with which one cell can communicate and bypassing layers becomes technically challenging. Across all these methods, it requires long times (sometimes 10s of hours) and many cells to build up enough signal to transmit the information between strains. Engineered mammalian cells have been used to transmit signals through receptors that trigger upon direct physical cell contact [CITE], neurons are naturally able to extend ??? to make contacts with many cells separated by large physical distances, and quorum signals are transmitted in soil biofilms between single cells [CITE]. Fully realizing the computational potential of a cell population will likely require the ability to transmit signals through contact [CITE], grow or print “brain-like” structures to transmit information [CITE] and amplify signals to reduce the number of cells required to transmit a message [CITE].

Computer aided design (CAD) will be essential for designing distributed computing, both in determining the properties of agents required to perform a function and in the design of the DNA to be carried by each agent. In this manuscript, we first converted the desired circuit to NOR gates using logic minimization algorithms. Then, we developed a partitioning algorithm that constrains the number of gates per cell and the communication signals while allowing the number of cells to vary. Instead of simply counting the number of gates, or regulatory genes, this criterion could be improved by directly calculating the material (metabolites) or energy (ATP) usage as the threshold. The partitioning algorithm leads to a list of cells with assigned logic operations, the DNA for which can then be designed by Cello. This workflow is limited to Boolean digital logic circuits. Ultimately, we will need design automation software for asynchronous and amorphous computing. There is software for simulating cell growth, communication and circuits as “agents” [CITE]. But there is not software to go in the other direction, where a desired computation is broken down to the required set of agents.

Cell computation may ultimately outperform electronic computers for specific classes of algorithms that lend themselves to repetitive, independent calculations that can be performed by multiple CPUs and then retrieved and synthesized to obtain a solution [CITE]. While we have selected a hash algorithm as a proof-of-principle, whether cellular computing is appropriate for this class of algorithms, and cryptography in general, remains an open question. [Typical energy consumption for mining hardware is 1000 megahashes/J. It takes an estimated 1,500 kWh to mine one bitcoin; enough energy to power a U.S. household for 13 years. [xxx]] [Conceptually, how many cells required for SH256 algorithms. Energy and power usage and time estimation]. However, it is important to consider that there are dozens of hash algorithms other than SH256 and MD5 that are used for various cryptocurrencies and other applications. All of these have been designed for use with an electronic CPU. One can imagine designing cryptography algorithms specific for cell-based computers that make use of their highly-parallelized asynchronous and amorphous structure. These approaches would have to overcome the reduction in processing speed, with the fastest between cells being the 1 ms pules in the brain, 10?-fold slower than an electronic computer. Coupled with advances in programming cells, this approach could lead to a new era of living cryptography.

* How many input states? Logic verification (pseudoverification)
* What is the potential computation in a gm of bacteria?
  + This is cool though looking at a mL of culture is more interesting for the cell-based computer argument b/c it gets pretty close to rivalling modern day CPUs.
* Scalability of phage repressors
* Timescale separation of inside cell computation versus cell-to-cell computation.
* Simulations. Are there key thresholds that make a major shift in complexity?
* Optogenetics could be used as an external clock or to spatially isolate or activate clusters of cells [CITE].
* Logic minimization simultaneous with partitioning [CITE].
* Timing is important. Intracellular logic takes ~20 min per layer. Extracellular wires take hours. The activation of a lux sensor is 2-200 nM and the production rate is 0.03 nM/hr by colonies on plates (Edge detector paper).
* An advantage over molecular computing, for example that which relies on DNA hybridization, is that the cells enable the sequestration of gates as well as ease in connection to biologically relevant sensors and actuators.
* There are analogies with the design of low power electrical circuits, where clusters of asynchronous and analog circuits are embedded within a larger clocked digital circuit, separated by latches [CITE].
* Interestingly lambda repressor is limited by coding theory to a predicted 77 variants in a single cell [xxx]

**Materials and methods**

**Add protein secondary structure**

**Partitioning**

**Circuit design/minimization after partitioning**

**Alternate fluorophore conversion to rpug**

Strains and media:

All routine cloning and characterization of plasmids containing non-R6K plasmid origins was performed in *E. coli* NEB10β competent cells (NEB C3019I). All cloning of plasmids containing R6K plasmid origins was done using E. coli JTK164A (Kittleson et al 2011) or E. coli TransforMaxTM EC100DTM pir+ (Lucigen CP09500). All genomically integrated constructs were characterized in E. coli MG1655 (YJP\_MKC174, Park et al 2020) or E. coli MG1655 ΔAraC (derivative of YJP\_MKC173, Park et al 2020). LB-miller liquid media (Difco 244620) and LB-miller media + 2% Bacto Agar (Difco 244620) plates were used for all routine cloning and culturing. Unless otherwise noted, all routine cloning was performed at 30°C. M9 media (1x M9 salts (Difco 248510), 2mM MgSO4, 100μM CaCl2, 0.2% Casamino acids (BD 223050), 0.4% glucose, 0.34 mg/mL Vitamin B1 (Alfa Aesar A19560)) was used for all assays unless otherwise noted. SOC recovery media (NEB B9020S) was used to recover all transformations. Antibiotics were used at the following concentrations: 50 ug/mL kanamycin (Kan, GoldBio K-120-10), 100ug/mL carbenicillin (Carb, GoldBio C-103-5), 5ug/mL tetracycline (Tet, GoldBio T-101-25), 50ug/mL spectinomycin (Spec, GoldBio S-140-5), 20ug/mL gentamicin (Gm, Enzo Lifesciences 380-003-G001), and 25ug/mL chloramphenicol (Cm, Alfa Aesar B20841).

Chemical inducers

Cells were induced with the following chemicals: isopropyl β -d-1-thiogalactopyranoside (IPTG, GoldBio I2481C), anhydrotetracycline (aTc, Sigma 37919), cuminic acid (Cuma, Sigma 268402), vanillic acid (Van, Sigma 94770), l-arabinose (Ara, Sigma A3256), 3-oxohexanoyl-homoserine lactone (Lux/OC6, Millipore Sigma K3007), 3-hydroxytetradecanoyl-homoserine lactone (Cin/OHC14, Sigma 51481), 2,4-diacetylphophloroglucinol (DAPG, Santa Cruz Biotechnology sc-206518), p-coumaroyl-homoserine lactone (Rpa, Millipore Sigma 07077), p-coumarate (precursor, Sigma C9008)

AraC knockout protocol:

The strain used to characterize the circuits is E. coli MG1655 cells (NCBI U00096.3) with three empty landing pads inserted in their genome (YJP\_MKC173) as previously described (yongjin’s paper). The native araC gene on the genome of YJP\_MKC173 cells was knocked out of the using a CRISPR-Cas9 assisted λ-RED recombineering plasmid (pAF\_1013). Briefly, overnight culture of YJP\_MKC173 cells were diluted 1:100 in LB (BD Difco 244620) and cultured at 37˚C until OD600 of 0.4 was reached. Cells were made electrocompetent by washing three times with 50 mL of 10% glycerol solution at 4˚C. Next, 200 ng of pAF.1013 plasmid was mixed with cells, electroporated (Eppendorf Electroporator 2510) at 2500V, rescued at 30˚C for 1 hr, plated on LB-agar plates containing 50 µg/mL kanamycin and incubated overnight at 30˚C. Next, overnight culture of pAF\_1013-containing YJP\_MKC173 cells were diluted 1:100 in LB with 50 µg/mL kanamycin, cultured at 30˚C to 0.4 OD600, heat shocked to 42˚C for 20 mins to induce the expression of the Gam, Beta, Exo genes, chilled on ice for 20 minutes and then made electrocompetent, as described above. 1 µg of linear DNA oligo, containing a total of 100 bp (tgcaatatggacaattggtttcttctctgaatggtgggagtatgaaaagttaattggtaacgaatcagacaattgacggcttgacggagtagcatagggt) homology arms for the DNA region flanking the araC gene on the genome, was dialyzed with 200 ng of gRNA encoding plasmid for 1 hr and transformed to cells by electroporation. Cells were rescued in NEB Stable Outgrowth Medium (NEB B9035S) + 0.2% L-arabinose for 1 hr at 30˚C and plated on LB-agar with 50 µg/mL kanamycin and 100 µg/mL carbenicillin. The cells were incubated at 30˚C overnight. Native araC was confirmed with colony PCR. The plasmids in YJP\_MKC173∆araC cells were cured by growing the cells in LB without antibiotics at 37˚C overnight, then adding 1 µL of saturated cells with 1 µL of the nucleoside 6-(β-D-2-Deoxyribofuranosyl)-3,4-dihydro-8H-pyrimido-[4,5-c][1,2]oxazin-7-one (dP; BioSearch Technologies PY7270-B050) at 1µM final concentration to 100 µL of LB in a PCR tube and growing the cells for 8 hours at 37˚C. Cells were streaked on LB-agar plates, 8 colonies were picked and checked for susceptibility to kanamycin and carbenicillin, and glycerol stocked.

Phage transduction of marionette cluster into landing pad strain

To construct a strain containing 3 empty landing pads with the MG1655 marionette sensor array in strain sAJM.1506 (Meyer et al 2019) using the protocol described in Meyer et al. Overnight cultures of sAJM.1506 and of MG1655 ΔAraC were grown in 1mL. 50μL of the sAJM.1506 overnight culture was added to 5mL LB + 0.2% glucose + 5mM CaCl2 and cultured at 37°C, 250 RPM for 30 minutes. 100μL phage P1vir- lysate was added to the culture. The culture then continued shaking at 37°C, 250 RPM for 3 hours until cell lysis. 50μL of chloroform was added and the cultured was shaken again for 5 minutes under the same conditions. The culture was then pelleted at 18,000g for 2 minutes and the supernatant was filter sterilized with a 0.2 μM filter. 1.5mL of the int299 overnight culture was then pelleted at 4000g for 3 mins and resuspended in 750μL of a 5mM MgSO4 + 10mM CaCl2 buffer. 1μL, 10μL, and 100μL of the marionette sensor phage was then mixed 100μL int299. The mix was incubated at room temperature for 30 minutes, 1mL LB + 200μL 1M NA-citrate was added to the mix, and the mix was cultured for 1 hour at 37°C, 250 RPM. Cells were then pelleted and resuspended in 100μL LB and plated at 37°C overnight on plates containing 5ug/mL chloramphenicol + 5mM sodium citrate. The next day, single colonies were picked and streaked on plates containing 12.5ug/mL chloramphenicol + 10mM sodium citrate. The transduction of the full marionette cluster and maintenance of the integrity of 3 landing pads was verified via PCR and sequencing.

NOT gate plasmid integrations into MG1655 Landing Pad strain with sensor array:

Chemically competent MG1655 possessing a sensor array integrated in att5 (YJPMKC174, Park et al 2020) and carrying a temperature sensitive plasmid expressing int2 (plasmid g69) were prepared by culturing cells overnight in LB + Spec at 30°C. Cells were then made chemically competent following the standard protocol using the Zymo Mix and Go kit (Zymo Research T3001). 100μL aliquots were frozen at -80°C for later use. For plasmid integrations, 100ng of plasmid was used and cells were plated at 37°C overnight with appropriate antibiotics. To cure the int2 integrase plasmid, a single colony was picked, restreaked on LB agar plates with appropriate antibiotics, and grown overnight. 8-16 single colonies were then picked into LB + Spec and grown overnight to test for Spec resistance.

attB2 and attB7 landing pad integrations:

Integrations into att2 and att7 landing pads of MG1655 ΔAraC with the Marionette sensor array (int300 this work, sensor array previously described in Meyer et al 2019) were performed by first transforming a plasmid constitutively expressing the int2 and int7 integrases on a temperature sensitive backbone (plasmid g365) to make strain int324. Electrocompetent cells were prepared by first inoculating cells into 1mL LB + Spec from glycerol stocks and culturing cells into stationary phase at 30°C, 250 RPM. Cells were then pelleted at room temperature by spinning at 4000g for 2-3 mins, washed 3x in 1mL H2O, and resuspended in 400μL H2O. 200μL of cells were mixed with 100ng of the appropriate plasmid(s) and electroporated at 2500V in 2MM cuvettes (VWR 89047-208). Immediately following electroporation, 800μL SOC was added and cells were recovered for 1 hour at 30°C at 250 RPM (New Brunswick Scientific Innova 44 shaker). After this, the entire volume was plated on appropriate antibiotics and grown overnight at 37°C. The same protocol was used for both single and double plasmid integrations.

attB5 landing pad integrations:

Integrations into the att5 landing pad were conjugated using triparental mating into target int300 derivatives already carrying Kan and Carb markers from a prior integration. Overnight cultures of pRK2013 (helper plasmid, Cm, Ryu 2019), plasmid g575 (plYJP053 in EC100D, Park et al 2020, R6K vector expressing int2 + int7 + int5, Carb), the target strain (Kan and Carb), and the att5 payload of interest (Tet) were inoculated from glycerol stocks into 1mL LB + antibiotics and cultured overnight at 30°C, 250 RPM. 50μL of each culture was mixed, pelleted at 4000g for 3 minutes, washed 1x with 500μL LB, and resuspended in 10μL LB. The 10μL resuspension was then spotted on an LB agar plate and left at room temperature for 7-8 hours. To select for transconjugants, the spot was then streaked onto LB + Tet + Kan plates and grown overnight at 37°C .

Genome NOT gate response functions characterization:

Strains were streaked from glycerol stocks onto plates LB agar + Kan plates. Single colonies were picked and cultured overnight in 400μL M9. The cultures were then back diluted 1:100 into 400μL M9 and grown for 1.5 hours . Cultures were then back diluted 1:1000 into 400μL M9 + appropriate inducer and cultured for 4.5 hours. 50μL of cells were then diluted into 180μL PBS + 1 g/L kanamycin for flow cytometry analysis. All steps were performed in 2mL 96-deep-well-plates (USA Scientific) + AeraSeal film (Excel Scientific) at 37°C shaken at 900 RPM (InforsHT Multitron Pro shaker incubator).

Plasmid NOT gate response functions characterization:

Strains were streaked from glycerol stocks onto LB agar + Kan plates cells. Single colonies were picked and grown overnight in 150μL M9 + Kan at 37°C shaking at 1000 RPM in shallow bottom NuncTM 96-well plates (Thermo Scientific 249662) using an ELMI plate shaker. Cells were diluted 1:100 and grown for 1.5 hours in 150μL M9 + Kan, then diluted 1:625 into M9 + Kan + appropriate inducers and grown for 5 hours. 50μL of cells were then diluted into 180μL PBS + 1g/L kanamycin for flow cytometry analysis.

NOT gate growth impact characterization:

Strains were streaked from glycerol stocks onto plates LB agar + Kan plates. Single colonies were inoculated overnight in 400μL M9 in 2mL 96-deep-well-plates + AeraSeal film at 37°C, 900 RPM (InforsHT Multitron Pro shaker incubator). The cultures were then back diluted 1:100 into 400μL M9 and grown for 1.5 hours. Cultures were then back diluted again 1:1000 into M9 + appropriate inducer and cultured for 5.5 hours. The optical density at 600nm (OD600) was then measured and a relative growth rate was calculated. For OD measurements, 200μL of culture was transferred to NuncTM 96-well plates with optically clear bottoms (Thermo Scientific, 165305). The plates were then placed into a Synergy H1 plate reader (BioTek Instruments) for absorbance measurements. Relative growth at a given induction level was normalized to the OD of uninduced cultures and calculated with the following equation: (

Cross talk characterization:

MG1655 strains were plated for single colonies on LB agar + Kan + Carb plates, inoculated into 150μL M9 in shallow bottom 96-well plates, and cultured overnight at 37°C, 1000 RPM on an ELMI shaker. Cells were then diluted 1:100 and cultured for 1 hour. Cells were then diluted 1:1000 into M9 with and without 1000μM IPTG and grown for 4.5 hours. 50μL of cells were then diluted into 180μL PBS + 1mg/mL and fluorescence was measured by flow cytometry. Median population values were extracted, autofluorescence measured with strain int300 was subtracted, and fold change was calculated as a ratio of the fluorescence in the 0μM IPTG and 1000μM IPTG induction conditions.

Construction of sender devices:

Need info from Haorong.

Circuit and quorum production response function characterization:

Strains were streaked from glycerol stocks onto LB agar plates containing appropriate antibiotics and incubated at 37°C overnight. Single colonies of both sensor and circuit strains were inoculated into 100μL M9 or M9 + Gm (as appropriate) and cultured in shallow bottom plates at 37°C, 1000 RPM until stationary phase. Cells were then diluted 1:2500 into M9 or M9 + Gm with appropriate inducers and cultured in shallow bottom cultured in shallow bottom plates at 37°C, 1000 RPM overnight. In the morning, 1μL of cells was diluted into 300μL PBS + 1g/L Kan for flow cytometry. Cells were then diluted 1:1000 into 1mL M9 or M9 + Gm in 2mL deep-well plates and cultured for 4 hours at 37°C, 900 RPM. Plates were then spun at 4500g for 10 minutes at room temperature to pellet cells. 500μL of supernatant was filtered to ensure removal of all cells. OC6, pC-HSL, and DAPG producing samples were filtered using either cellulose acetate 96-well filter plates (Cytiva Life Sciences 7700-2808) or regenerated cellulose 96-well filter plates (Chrom Tech 96F-RC020). OHC12 producing samples were always filtered using regenerated cellulose filters. For circuits producing pC-HSL, 100nM of p-Coumarate was added [supplier]. Overnight cultures of the appropriate sensor cells were then diluted 1:1000 into 150μL of the collected supernatant (containing the produced quorum molecule) and cultured for 3 hours at 37°C, 1000 RPM in shallow bottom plates. 50μL of cells were then diluted into 180μL PBS + 1g/L Kan and analyzed via flow cytometry.

Circuit simulation

Verilog code for the MD5 combinatorial algorithm (<https://www.ietf.org/rfc/rfc1321.txt>) was adapted from <https://github.com/stass/md5_core/blob/master/md5_core.v>. The circuit design was synthesized, minimized to reduce the total number of gates, and mapped to NOT/NOR gates using Yosys with the commands in the “commands\_md5\_opt.ys” file. After partitioning, wiring diagrams for individual cells were generated by writing individual structural Verilog files for each cell and, if possible, minimized further using Yosys before genetic circuit implementation.

Genetic circuit gate assignment

Genetic circuits were designed using the methods previously described in Nielsen 2016. Custom Python code was written for each individual circuit to perform gate assignment based on the Yosys designed circuit. For gate assignment, the same simulated annealing algorithm as described in Nielsen 2016 was implemented in Python. NOT gate parameter fits and input sensor data measured from the marionette sensor array was used for gate assignment. Because some circuits have multiple outputs, a modified circuit score describing the worst case circuit behavior was used to rank gate assignments. For each output, the ratio between lowest predicted ON state and highest predicted OFF state is calculated and the circuit score is defined as the lowest score across outputs.

Hill equation curve fitting

All fits were performed using the SciPy Python package scipy.optimize.curve\_fit() method. Response functions from NOT gates were fit to the following Hill equation: while response functions from activators were fit to .

RNAP/s conversion

To convert fluorescence values to physical units, fluorescence values were first converted to RPUg (Park et al 2020). To convert to RPUg, cells containing RPU reference promoter (YJP\_MKC254) were cultured with the same protocol as the cells of interest. Fluorescence was converted RPUg by dividing fluorescence values by the fluorescence observed from YJP\_MKC254 (autofluorescence was subtracted all samples). Shao et al 2021 determined the RNAP flux in MG1655 of the RPU reference promoter to be 0.029 RNAP/s/DNA and Park et al 2020 estimated the DNA copy number of the landing pad containing the RPU standard to be 3.5. Multiplying the two yields a scaling factor of 1 RPUg to 0.102 RNAP/s.

Flow Cytometry:

Fluorescence measurements were taken on BD LSRII Fortessa flow cytometer with the HTS attachment (BD Biosciences). Samples were prepared as described in other sections. Samples were run at a flow rate 0.5μL/sec and 2.0μL/sec and 30,000 events were captured. The FITC, PE-Texas Red, and Pacific Blue channels were used to collect data for YFP, RFP, and BFP, respectively. The Cytoflow Python package (<https://github.com/cytoflow/cytoflow> ) was used for processing FCS 3.0 files and gating cells. FSC, SSC, FITC, PE-Texas Red, and Pacific Blue voltages were set to 750V, 300V, 450V, 600V, and 418V, respectively. Medians were extracted from each sample.

**Acknowledgements**

**Figure Captions**

**Figure 1:**

1. Overview of partitioning scheme. The initial criteria were a maximum of eight gates and three cell-cell signalling channels per cell with 4 channels total. The first stage divides the gates into subcircuits that can be implemented in single cells. Shaded boxes represent individual cells. The second stage assigns cell-cell signalling channels to wires that cross cell boundaries. Each color represents a different communication channel.

**Figure 2:**

1. The left schematic shows all 12 repressors aligned by the first helix of their DNA binding HTH motif. Hatched boxes represent Pfam Peptidase S24 domains. Gray boxes represent the individual putative DNA binding helices for the HTH motif. The right schematic shows a diagram of each repressors cognate promoter. Colored boxes represent operator sites. White boxes represent the -35 and -10 sequences of the promoter while the gray boxes represent 15bp spacer sequences (Materials and Methods).
2. Schematic a NOT gate. The entire NOT gate was integrated in the attB2 landing pad. Note the orientation of the repressor and output constructs have been reversed for clarity (see Gate datasheets).
3. Response functions for each gate are shown on the left. The following 12 concentrations of inducer(s) were used: 0, 10, 20, 30, 40, 50, 70, 100, 150, 200, 1000, and 1000uM IPTG + 200nM ATC. Fluorescence measurements were converted to RNAP/s and fit to a hill equation as described in the Materials and Methods. For promoter cross-talk measurements, the repressor and output promoter were cloned into the attB7 and attB2 landing pads, respectively (Supplementary Figure XX). Strains were induced with 0uM and 1000uM IPTG. Fluorescence units were converted to RPUg and a ratio of the output in the 0uM and 1000uM induction conditions is shown.
4. Schematic of sender-receiver devices. All constructs were assayed using the attB2 landing pad with the marionette sensor array. Note the RpaRAM sensor was not present in the marionette sensor array and cloned into strains as needed. To characterize the sender receiver devices, enzymes were cloned downstream of a PTac promoter and induced with the same concentrations as the NOT gates (except for the aTc condition, 11 points total). Characterization protocols, fluorescence conversion to RNAP/s, and hill equation fitting was performed as described in the Materials and Methods.

**Figure 3:**

1. Circuit diagram of the largest single circuit in the MD5 circuit. The circuit contains two inputs, eight gates, and produces three output molecules: pC-HSL, DAPG, and OC6.
2. Genetic circuit diagram. The circuit components were integrated into the attB2 and attB7 landing pads, including three fluorophores controlled by the same promoters as the three communication channels. NOR gates were implemented by integrating 2 copies of the gate into separate landing pads, each controlled by an appropriate promoter. The OC6 and pC-HSL production cassettes were integrated into the attB5 landing pad. The DAPG production pathway was placed on a plasmid containing a pSC101 origin of replication. Note the marionette sensor array is not present in a landing pad.
3. Circuit characterization data. The left plot shows the predicted promoter activity (gray bars) and fluorescence characterization of each of the 3 fluorophores in the circuit (points). The left 4 bars represent YFP, middle 4 bars represent RFP, and right 4 bars represent BFP. The right plot shows the output from the circuit measured from the output receiver cells with bars representing mean promoter activity. Black bars at the top show expected ON states. Cells were induced with 200nM aTc and 100uM Cuma as appropriate.
4. The scatterplot shows the predicted vs. measured circuit outputs for each individual state of every unique cell in the MD5 circuit (pearsons correlation = ).
5. The top histogram shows the total amount of synthetic DNA in each unique cell. Bins represent 2.5kb of DNA. The bottom histogram shows the total number of regulatory genes involved in the circuit of each unique cell. The count is the sum of the sensors used in the circuit, the total number of gates, total number of fluorophores, and total number of genes involved for synthesis of the quorum outputs. Bins represent 2 genes.

**Figure 4:**

1. The schematic shows the complete, fully minimized circuit diagram for the MD5 circuit implemented. Shaded boxes represent circuits in individual cells (colors have no meaning and are just to aide in visualization). Any gates not within a shaded box were implemented as a single cell containing that gate. Black lines represent inputs to the circuit as a whole, implemented using non-quorum chemical inducers. The 4 colored lines represent the cell-cell signaling inputs/outputs for the circuit. Blue, purple, green, and orange lines represent OC6, OHC12, pC-HSL, and DAPG respectively. The subscript i denotes a single bit of each 2-bit value.
2. Characterization of the cell-cell signalling as measured via the receiver cells for each cell in the MD5 circuit is shown. The bars are colored by quorum signal with output measured in RNAP/s (legend in top right shaded box). The plots are spatially organized to approximate the correct position in the circuit diagram. As before, lines represent either inputs to the whole circuit or cell-cell communication signals and bars represent the mean of 3 independent replicates taken on different days. Letters in the top right of each plot show cells that are identical to each other.

**References**