Supplementary Materials for

Synthetic neuromorphic computing in living cells

Luna Rizik†, Loai Danial†, Mouna Habib†, Ron Weiss and Ramez Daniel*

†These authors contributed equally to this work
Correspondence to: ramizda@bm.technion.ac.il

This PDF file includes:

Materials and Methods
Supplementary Text
Figs. S1.1 to S14.158
Tables S1.1 to S9.1
Supplementary References

Materials and Methods

Strains

All plasmids in this work were constructed using basic molecular cloning techniques. *Escherichia coli* 10 β (araD139 D(ara-leu)7697 fhuA lacX74 galK (W80 D(lacZ)M15) mcrA galU recA1 endA1 nupG rpsL (StrR) D(mrr-hsdRMS-mcrBC)).

Plasmid Construction

All the plasmids in this work were constructed using basic molecular cloning techniques(*1*). New England Biolab's (Beverly, MA) restriction endonucleases and Thermo Scientific FastDigest Restriction Enzymes, T4 DNA Ligase, and Taq Polymerase were used. PCRs were carried out with a Bio-Rad S1000™ Thermal Cycler with Dual 48/48 Fast Reaction Modules. Synthetic oligonucleotides were synthesized by Integrated DNA Technologies (Coralville, IA). Plasmids were transformed into *E. coli* 10β, with a standard heat shock protocol (*1*). Plasmids were isolated with Qiagen QIAprep Spin Miniprep Kits (Qiagen, Hilden, Germany) according to the manufacturer's recommendations. Modifications were confirmed by restriction digests. DNA sequencing was done by the Macrogen Sequencing Service (Macrogen Europe, The Netherlands). All devices (promoter-RBS-gene-terminator) were initially assembled in the Lutz and Bujard expression vector pZE11G (*2*) containing ampicillin resistance and the ColE1 origin of replication. Parts are defined as promoters, RBSs, genes, and terminators. Manipulation of different parts of the same type was carried out using the same restriction sites. For example, to change a gene in a device we used Acc65I/KpnI and BamHI/XmaI. To assemble two devices, we used a single restriction site flanking one device and used oligonucleotide primers and PCR to add that restriction site to the 5' and 3' ends of a second device. To assemble multi-devices, we used the Gibson Assembly Master Mix from New England Biolabs to join the DNA fragments (Ipswich, MA), following the manufacturer's instructions. The overlapping inserts were prepared by PCR amplifications using the Phusion High-Fidelity PCR kit (New England Biolabs). Each assembly reaction contained approximately 250 ng of each insert and 100 ng of the linearized vector and incubated at 50°C for 60 min, followed by a transformation into heat shock *E. coli* 10β cells. Colony PCR screening was carried out using forward and reverse primer pairs. Positive clones were sequencing verified. After assembling devices in the ampicillin-resistant ColE1 backbone, antibiotic-resistance genes were changed using AatII and SacI, and the origin of replications was changed with SacI and AvrII. Supplementary Information, Section 16, provides details regarding plasmid maps. The references in Supplementary Information, Section 18, provide details regarding the origin of the plasmids.

Circuit Characterization

Overnight cultures of *E. coli* strains were grown from frozen glycerol stocks at 37^{o}C, in a Shel Labs SSI5 shaking incubator at 250 r.p.m., in 5 ml of Luria–Bertani–Miller medium (Fisher) with appropriate antibiotics: Carbenicillin (50 µg mL^{-1}), Kanamycin (30 µg mL^{-1}), Chloramphenicol (25 µg mL^{-1}). The inducers used were arabinose, IPTG - isopropyl-b-D-1-thiogalactopyranoside, and AHL 3OC6HSL (Sigma-Aldrich). Overnight cultures were diluted 1:100 into 5 ml fresh Luria–Bertani medium with antibiotics and were incubated at 37^{o}C, 250 r.p.m. for 30 min. Cultures (200 µl) were then moved into 96-well plates, combined with inducers, and incubated for 4 h and 20 min in a microplate shaker (37^{o}C, 500 r.p.m.) until they reached an OD_{600nm} ~ 0.4–0.6. At least 10,000 events were recorded in all experiments, and these data were then gated by forward scatter and side scatter using CyExpert 2.2 (Cytoflex S).

The geometric means of the gated fluorescence distributions were calculated using MATLAB. Fluorescence values were based on geometric means of flowcytometry populations from three experiments, each of which corresponded to 10,000 events.

Plate Reader/FACS set-up

GFP fluorescence was quantified by excitation at a wavelength of 484nm and emission at a wavelength of 510nm. mCherry fluorescence was quantified by excitation at 587nm and emission at 610nm. PE-TexasRed filter voltages on a BD LSRFortessa high throughput sampler to measure GFP and mCherry expression levels, respectively. The FACS voltages were adjusted using CyExpert 2.2 software so that the maximum and minimum expression levels could be measured with the same voltage settings. Thus, consistent voltages were used across each entire experiment. The same voltages were used for subsequent repetitions of the same experiment. GFP was excited with a 488nm laser, and mCherry was excited with a 561nm laser. Supplementary Information, Section 14, provides our FACS data.

1. Perceptual computing models

A single layer of the artificial neural networks (ANN) receives multiple linear-scale analog inputs (-∞,+∞). This network (Fig. S1.2): (1) multiplies each input x_i by its corresponding analog scalar n_i, which represents the synaptic weight, (2) sums the multiplication products, $y = \sum x_i n_i$, and (3) contains a non-linear activation function, which is commonly described by a sigmoid function $z = \frac{e^{y}/K}{1+e^{y}/K}$, $z \in [0,1]$. This neural model is known as the *perceptron* (*3*). Asymptotically, the node or the activation function acts as a decision-making function that determines the digital levels corresponding to the analog inputs.

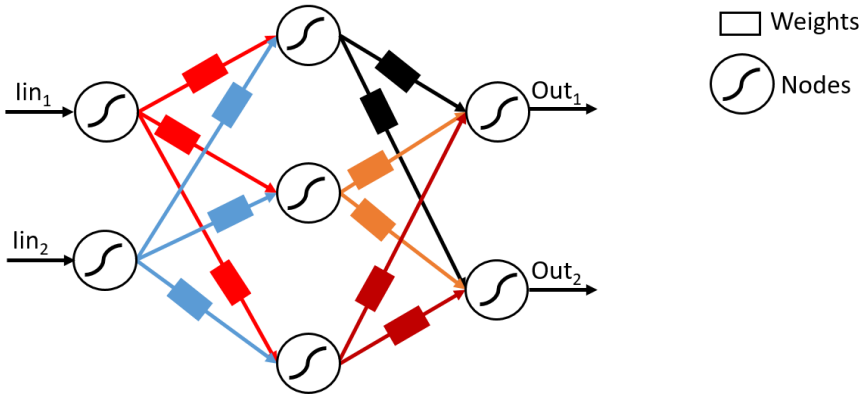


Fig. S1.1. An artificial neural network**.** The non-linear digital elements in the network are called artificial neurons, and are represented as nodes within the graphical abstraction of the network. The strength of each analog signal is called an artificial synapse (weight) and is represented by an edge. The interactions between nodes through the weights lead to the global behavior of the network.

The simulation results of perceptron including the analog signal and the perceptron output signal are shown in Figure S1.2. We present the results in the 2D contour curve and the 3D surface curve. Every method has its benefits. For example, it is simpler to present the analog pattern in 2D contour compared to the 3D surface curve. By contrast to the perceptron output signal, it is better illustrated in the 3D surface curve.

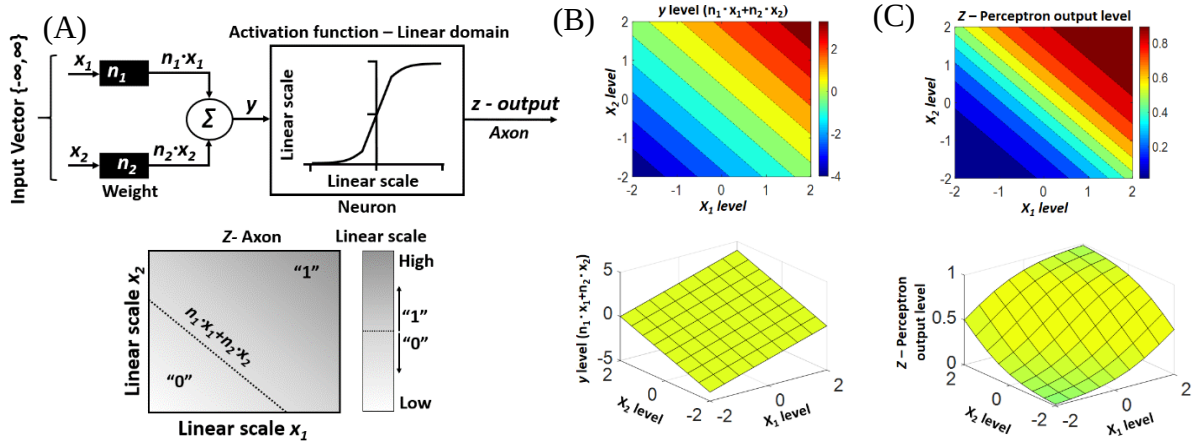


Fig. S1.2. (A) Anatomical structure of abstract perceptron model. The proposed model receives analog inputs processed by analog-weighted elements that collectively interact through non-linear nodes to make an assertive decision. The perceptron model operates in the linear domain, which is widely used as the neuro-processing core in artificial neural networks. (B) Simulation results of the analog signal ($n_1x_1+ n_2x_2$) in 2D contour curve and 3D surface curve. (C) Simulation results of the perceptron signal in 2D contour curve and 3D surface curve.

BOX1: Abstract model of perceptgene

The output of power-law and multiplication circuit can be approximated as:

$$Y = \prod_{i=1}^{N} Y_m \cdot \left(\frac{X_i}{K_{mi}}\right)^{n_i} \quad \text{(I)}$$

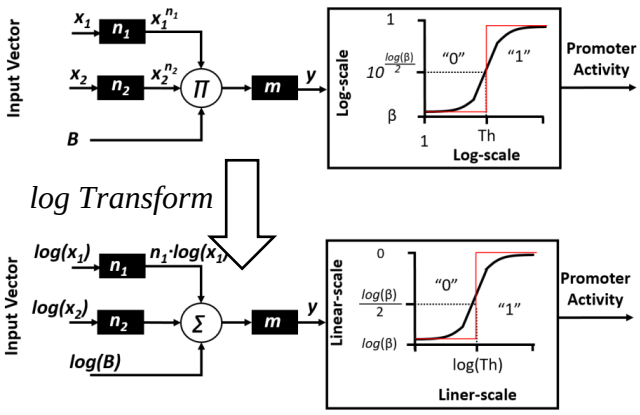
Where X_i is the input concentration, K_{mi} is dissociation constant, or input dynamic range or normalization. Y_m has units of concentration, and it equlas the the maximum level of produced transcription factors (See Supplementary Eq. S2.13). n_i is Hill-coefficient of the input X_i. The promoter activity is initiated when the transcription factor Y binds, and is given by:

$$P_r = \frac{\left(\frac{Y}{K_d}\right)^m + \beta}{1 + \beta + \left(\frac{Y}{K_d}\right)^m} \quad \text{(II)}$$

Where β is the basal level of the promoter, K_d is the dissociation constant of binding Y to promoter, and $\boldsymbol{m}$ is the Hill-coefficient (e.g. number of binding sites within the promoter). The two equations yield an abstract model which is given by:

$$\begin{cases} y = \left(\prod_{i=1}^{N} B \cdot {x_i}^{n_i}\right)^m \\ P_r = \frac{y+\beta}{1+\beta+y} \end{cases} \quad \text{(III)}$$

x_i is normalized input ($1 < x_i < IDR$). Where IDR is the input dynamic range of each input x_i.



The abstract model includes three computational components:

1. **Network Weights (n_i and m): are represented by the effective Hill-coefficients and depend on the biological cooperativity of proteins, number of binding sites in the promoter, the protein quaternary structure (the number of subunits that interact with each other and arrange themselves to form a final protein), design topology, and some cases in other small molecules.**
2. **Bias (B): is determined by the translation/transcription rates, mRNA/protein half-lives, rate of cell growth, binding affinities in protein-protein or protein-DNA reactions. Bias constants are unit-less, $B = Y_m/K_d$.**
3. **Activation functions or network nodes (z_i- the output of each perceptgene layer) which depend on promoter activity and are given by the normalized Michaelis-Menten model with a basal level.**

The basal level has two significant roles in determining the behavior of perceptgene model. First, it preserves the output dynamic range in the logarithmic scale$-\log(\beta)$. Second, it sets the effective threshold of the perceptgene (See Supplementary Information, Section 4).

Two models of perception can be considered in this work, the perceptgene model (Fig. S1.3A) and Michaelis-Menten (MM)-based perceptron model (Fig. S1.3B). Both models include the bindings between transcription factors (TFs) and DNA, with promoter activities modeled as activation functions. The perceptgene model is a logarithmic transformation of the perceptron and it operates in the logarithmic domain. The MM-based perceptron model is similar to the perceptron and operates in the linear domain. The MM-based perceptron model is advantageous in its simple design. For example, the summation will be implemented by expressing common proteins by the inputs (*4*), and the weight n_i is represented as the affinity at ribosome-binding site (*4*). Our analysis showed that such a model requires a much higher Hill coefficient (m) to operate than the perceptgene model.

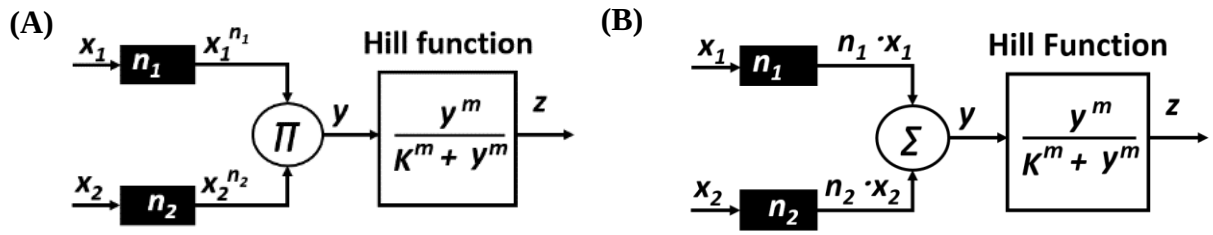


Fig. S1.3. (A) Perceptgene model, (B) Michaelis-Menten (MM)-based perceptron model.

The two systems accept two inputs in the range of $x_L < x_1, x_2 < x_H$. For simplicity, we assume that $n_1 = n_2 = n$, and the basal level *(β)* of every promoter is very low. As in Madar et al.(*5*) and illustrated in Fig. S1.4, for systems that can be described by a Hill function$\frac{y^m}{K^m+y^m}$, we define the output dynamic range (*ODR*) as the difference between the 90% and 10% of the maximal output z_{max} and the input dynamic range (IDR) as the ratio of the input concentrations required for 90% and 10% of the maximal output. For simplicity, we assume that $z_{max} = 1$.

1.1. Perceptgene model

For low-value inputs $x_1 = x_2 = x_L$, we get:

$y_L = {x_L}^n \cdot {x_L}^n$

$z_L = \frac{{y_L}^m}{K^m+{y_L}^m} = 0.1$

$$\Rightarrow \quad n \cdot m \cdot \log(x_L) = 0.5 \cdot \log\left(\frac{1}{9}\right) + \frac{m}{2} \cdot \log(K) \qquad \text{(S1.1)}$$

For high value inputs $x_1 = x_2 = x_H$, we get:

$y_H = {x_H}^n \cdot {x_H}^n$

$z_H = \frac{{y_H}^m}{K^m+{y_H}^m} = 0.9$

$$\Rightarrow \quad n \cdot m \cdot \log(x_H) = 0.5 \cdot \log(9) + \frac{m}{2} \cdot \log(K) \qquad \text{(S1.2)}$$

The $\boldsymbol{IDR}$ can be expressed as (Fig. S4):

$$IDR = \log(x_H) - \log(x_L) \qquad \text{(S1.3)}$$

Substituting Eq. S1.1 and Eq. S1.2 into Eq. S1.3, the IDR of perceptgene is given by:

$$IDR = \frac{1}{n \cdot m} \cdot 0.5 \cdot \log(81) \approx \frac{1}{n \cdot m} \qquad \text{(S1.4)}$$

1.2. Michaelis-Menten (MM)-based perceptron model

For low-value inputs $x_1 = x_2 = x_L$, we get:

$y_L = n \cdot x_L + n \cdot x_L$

$z_L = \frac{{y_L}^m}{K^m + {y_L}^m} = 0.1$

$\Rightarrow \quad m \cdot \log(x_L) = \log\left(\frac{1}{9}\right) + m \cdot \log(K) - m \cdot \log(2) - m \cdot \log(n)$ (S1.5)

For high-value inputs $x_1 = x_2 = x_H$, we get:

$y_H = n \cdot x_H + n \cdot x_H$

$z_H = \frac{{y_H}^m}{K^m + {y_H}^m} = 0.9$

$\Rightarrow \quad m \cdot \log(x_H) = \log(9) + m \cdot \log(K) - m \cdot \log(2) - m \cdot \log(n)$ (S1.6)

Substituting Eq. S1.5 and Eq. S1.6 into Eq. S1.3, the IDR of MM-based perceptron is given by:

$IDR = \frac{\log(81)}{m} \approx \frac{2}{m}$ (S1.7)

The Hill coefficient values ($\boldsymbol{m}, \boldsymbol{n}$) of synthetic biological parts often are between 1 – 2 (*4*, *6*). Therefore, the $\boldsymbol{IDR}$ of the MM-based perceptron is approximated as 1.333 fold (for $m = 1.5$), and the $\boldsymbol{IDR}$ of perceptgene is approximated as 0.667 fold (for $n = 1, m = 1.5$) and 0.333 fold (for $n = 2, m = 1.5$), respectively. Therefore, Eq. S1.4 and Eq. S1.7 show that the MM-based perceptron model requires a higher value of $\boldsymbol{m}$ than the perceptgene model to operate.

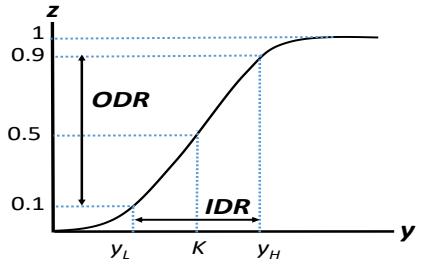


Fig. S1.4. Definition of Input Dynamic Range and Output Dynamic Range.

Figs. S1.5 and S1.6 show the simulation results of the perceptgene model and the MM-based perceptron model for n=1 and 0.5, respectively. The MM-based perceptron model fails to act as a binary classifier (there is no clear separation between "0" and "1" states). By contrast, the perceptgene shows a distinct separation between "0" and "1" states.

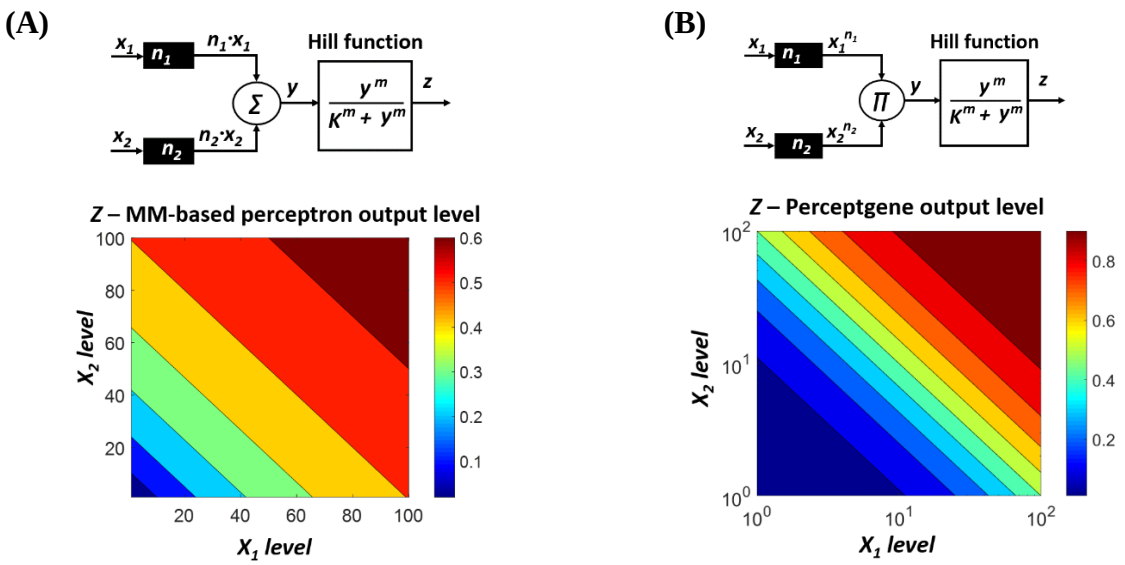


Fig. S1.5. (A) Michaelis-Menten (MM)-based perceptron model that combines linear operations and biochemical reactions. The model operates in the linear domain. Simulation results of the MM-based perceptron model, with $n_1 = n_2 = 1, m = 1$. (B) Simulation results of the perceptgene model, with $n_1 = n_2 = 1, m = 1$.

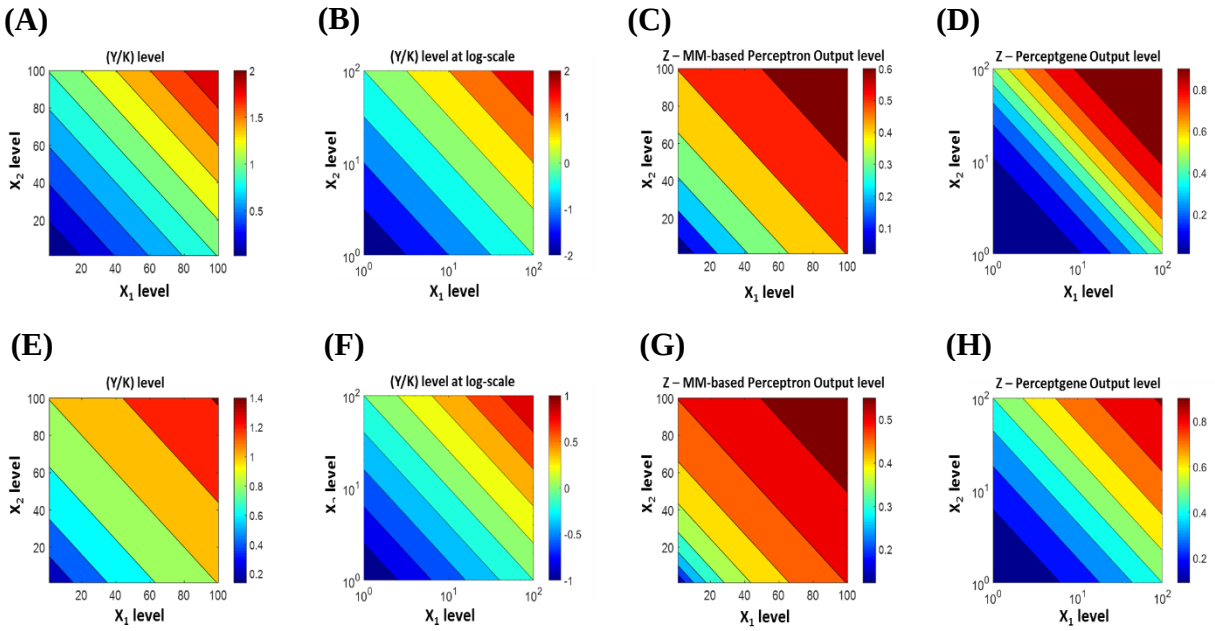


Fig. S1.6. Simulation results of (A) Analog signal (Y/K) for MM-based perceptron model, $n = 1$. (B) Analog signal (Y/K) for perceptgene model, $n = 1$. (C) Output signal for MM-based perceptron model, $n = 1$. (D) Output signal for perceptgene model, $n = 1$. (E) Analog signal (Y/K) for MM-based perceptron model, $n = 0.5$. (F) Analog signal (Y/K) for perceptgene model, $n = 0.5$. (G) Output signal for MM-based perceptron model, $n = 0.5$. (H) Output signal for perceptgene model, $n = 0.5$. In all the simulations, we assumed that $m = 1$.

1.3. 3D-Plane: Simulation of perceptron and perceptgene models include basal level

The perceptron model includes MM as the activation function failed to classify the analog pattern into two non-linear levels even for high hill coefficient in the activation function

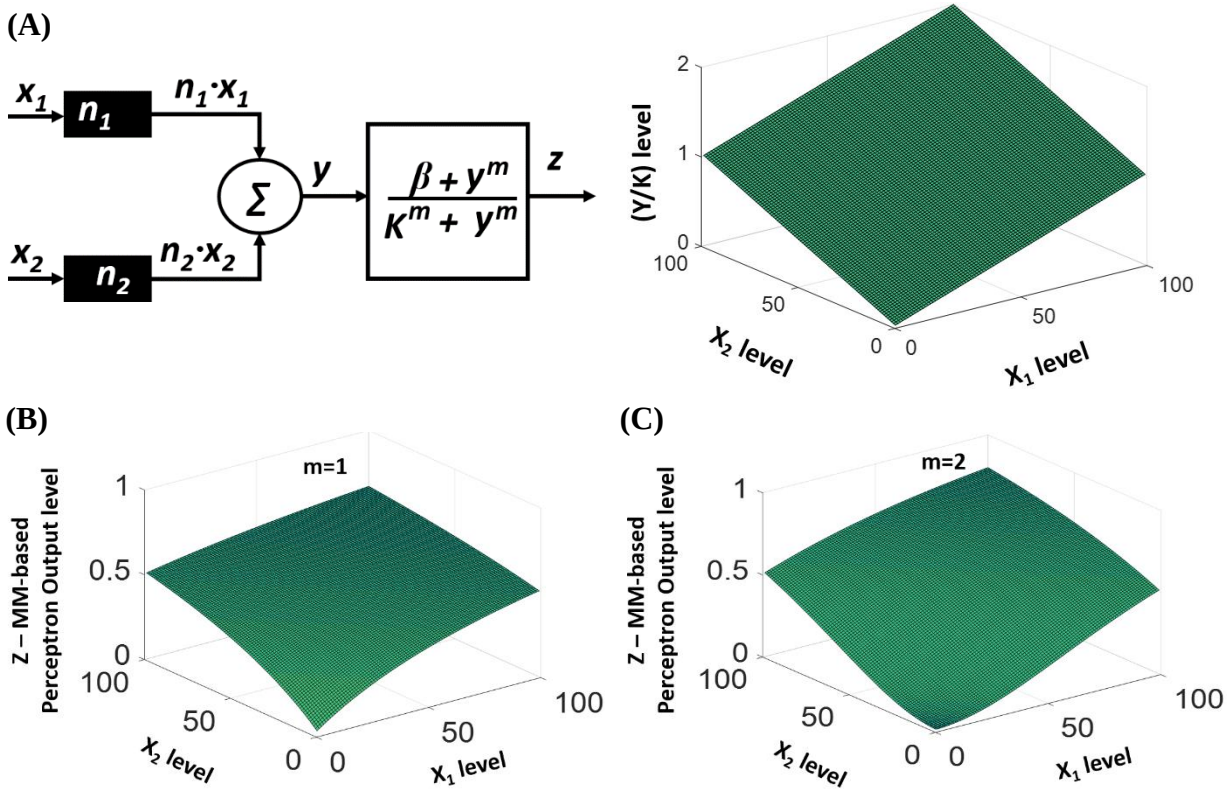


Fig. S1.7. Simulation results of (A) Analog signal (Y/K) for MM-based perceptron. (B) Output signal for modified perceptron model, $m = 1$. (C) Output signal for modified perceptron model, $m = 2$. In all the simulations, we assumed that $n_1 = n_2 = 1$, $\beta = 0.01$.

The perceptgene succeed to classify the analog pattern into two non-linear levels even for low Hill coefficient in the activation function

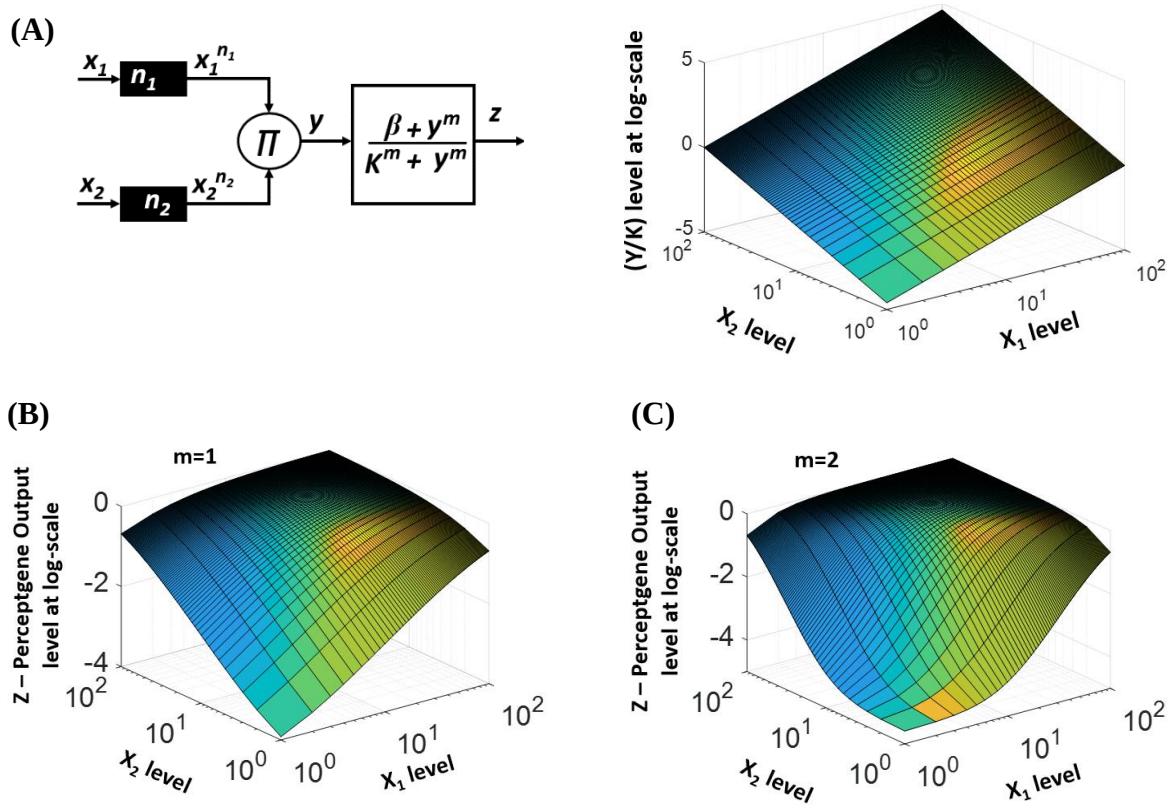


Fig. S1.8. Simulation results of (A) Analog signal (Y/K) for perceptgene. (B) Output signal for perceptgene model, $m = 1$. (C) Output signal for perceptgene model, $m = 2$. In all the simulations, we assumed that $n_1 = n_2 = 1$, $\beta = 0.01$.

1.4. Sensitivity analysis for perceptgene circuit

Sensitivity measures the fold change in the output as a function of the fold change in the input and is given by:

$$S_{Out-In} = \frac{\Delta Out/\langle Out \rangle}{\Delta In/\langle In \rangle} \quad \text{(S1.8)}$$

Where In is the input and Out is the output (Fig. S1.9A). "< >" denotes the mean of the signal. In this section, we analyze the signals' sensitivity that propagate through the perceptgene (Fig. S1.9B). The signals are given by:

$$y = B \cdot \left(\frac{x}{IDR}\right)^n + y_0 + \sigma_y \quad \text{(S1.9.1)}$$

$$z = Z_{max} \cdot \frac{y^m}{y^m + K^m} + z_0 + \sigma_z \quad \text{(S1.9.2)}$$

Where B is the bias with concentration units, $\boldsymbol{IDR}$ is the input dynamic range, x is the perceptgene input (unitless), y_0 is the background signal, z_0 is the promoter basal level, n and m are Hill-coefficients, σ_y and σ_z are random numbers, Z_{max} is the maximum protein expressed in the system.

We calculate the sensitivity of three systems (Fig. S1.9B):

- An analog system maps the analog input signal (x) to collective weighted analog output signal (y) using a power-law and multiplication function:
  $S_{y-x} = \frac{\Delta y/\langle y\rangle}{\Delta x/\langle x\rangle}$ (S1.10)
- A digital system maps analog signal (y) to output levels (z) with a sigmoidal activation function:
  $S_{z-y} = \frac{\Delta z/\langle z\rangle}{\Delta y/\langle y\rangle}$ (S1.11)
- A neuromorphic system combines the analog and digital systems:
  $S_{z-x} = \frac{\Delta z/\langle z\rangle}{\Delta x/\langle x\rangle}$ (S1.12.1)

  $S_{z-x} = \frac{\Delta z/\langle z\rangle}{\Delta y/\langle y\rangle} \cdot \frac{\Delta y/\langle y\rangle}{\Delta x/\langle x\rangle}$ (S1.12.2)

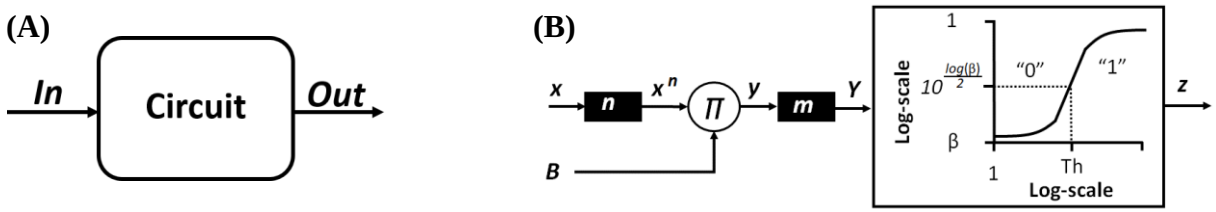


Fig. S1.9. (A) A circuit that has an input In and output Out. (B) Perceptgene with one input (x). y is the collective analog signal and z is the output.

Using Eq. S1.9.1 and S1.9.2, we get:

$\frac{\Delta y}{<y>} = n \cdot (1 - \frac{y_0}{<y>}) \cdot \frac{\Delta x}{<x>}$ (S1.13.1)

$\frac{\Delta z}{<z>} = m \cdot (1 - \frac{z_0}{<z>}) \cdot (1 - \frac{<z>-z_0}{Z_{max}}) \cdot \frac{\Delta y}{<y>}$ (S1.13.2)

Therefore, according to the definition of sensitivity, we get:

$S_{y-x} = n \cdot (1 - \frac{y_0}{<y>})$ (S1.14.1)

$S_{z-y} = m \cdot (1 - \frac{z_0}{<z>}) \cdot (1 - \frac{<z>-z_0}{Z_{max}})$ (S1.14.2)

$S_{z-x} = n \cdot m \cdot (1 - \frac{y_0}{<y>}) \cdot (1 - \frac{z_0}{<z>}) \cdot (1 - \frac{<z>-z_0}{Z_{max}})$ (S1.14.3)

Our simulation results (Fig. S1.10) based on the set of equations S1.14.1- S1.14.3 show the influence of the background level (y_0) on the analog, digital and neuromorphic systems and their sensitivity. When y_0 increases, the effective input dynamic range (IDR_{eff}) decreases (Fig. S1.10A). Based on Eq. S1.9.1 when $y_0 = 0$, the $IDR_{eff} = IDR$, meaning that the effective (actual) input dynamic range is equal to the theoretical input dynamic range. While increasing the background level decreases the sensitivity of the analog system (S_{y-x}, Fig. S1.10D), it almost does not affect the sensitivity of the neuromorphic system (S_{z-x}, Fig. S1.10F). More interestingly, the sensitivity of the neuromorphic system (S_{z-x}, Fig. S1.10F) is higher than the sensitivity of digital system (S_{z-y}, Fig. S1.10E). These results strongly depend on the value of basal promoter level (z_0). In case where $z_0 = 0$, the digital system's sensitivity can be higher than the sensitivity of the

neuromorphic system. However, since synthetic biological parts always have basal levels, we expect that the neuromorphic system's sensitivity will not be affected by the background level as compared to the digital system. Therefore, in conclusion, the main contribution of increasing the background level (y_0) on the performance of a neuromorphic (perceptgene) system is decreasing the input dynamic range.

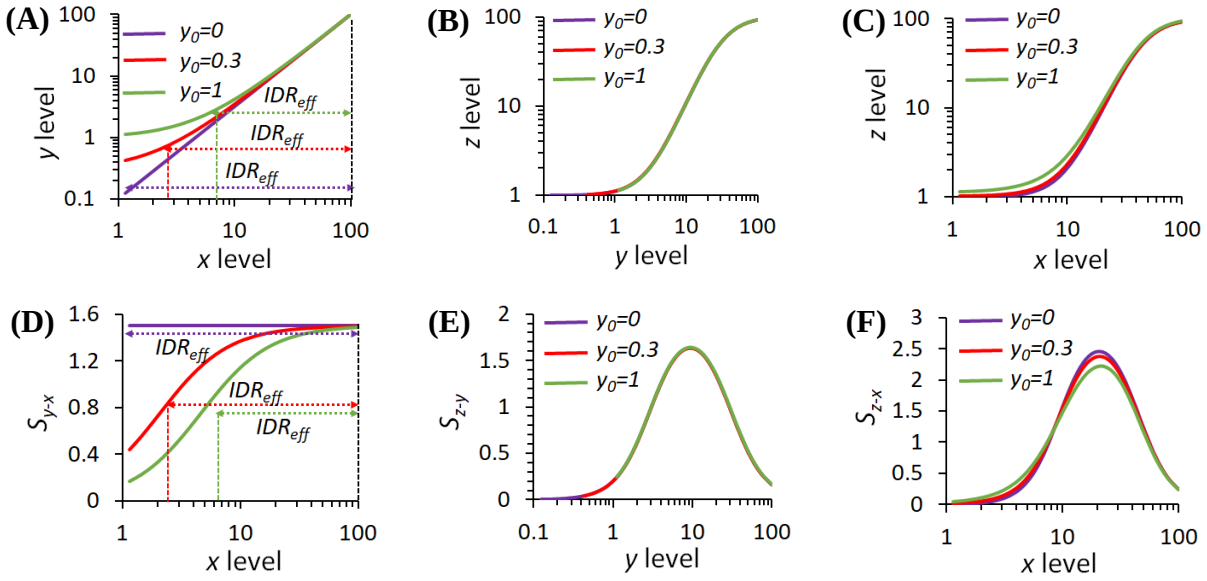


Fig. S1.10. Simulation results show the influence of the background level (y_0) on the signals and sensitivity of perceptgene. (A) Simulation results of a weighted and collective analog signal (y) versus input level (x). (B) Simulation results of the output signal (z) versus collective signal (y). (C) Simulation results of the output signal (z) versus input signal (x). (D) Sensitivity of the analog system (S_{y-x}), where x level is the input and y is the output. (E) Sensitivity of the digital system (S_{z-y}), where y is the input and z is the output. (F) Sensitivity of the neuromorphic system (S_{z-x}), where x is the input and z is the output. IDR_{eff} is the effective input dynamic range when the background signal (y_0) is higher than zero. Simulation parameters: $n = 1.5, IDR = 100, B = 100, Z_{max} = 100, m = 2, z_0 = 1$.

In summary: The analog system maps input levels to output levels using a power-law and multiplication function. The digital system maps input levels to output levels with a sigmoidal activation function in the logarithmic domain. The neuromorphic system combines analog and digital systems to carry out perceptgene computation. Our analysis shows that the analog system's sensitivity is reduced significantly with an increase in basal expression, meaning that analog operation in the low concentration regime is highly prone to errors. The digital system is designed to be insensitive in the low concentration regime, and hence less affected by an increase in basal expression. The neuromorphic system combines the best features of both, namely the ability to perform analog computation while being insensitive to the low concentration regime's noisy aspects, as shown in the sensitivity analysis (Fig. S1. 10).

1.5. Noise analysis for analog signals in perceptron

Here we will evaluate the noise that generates during the signal aggregation in genetic perceptron. If we assume that their "*l*" signals that aggregate and each of them has a noise of ΔX, then the system should satisfy two boundary conditions:

1. $l \cdot \Delta X_L " \overline{X_L}$ (S1.15.1)
2. $\Delta X_H " \overline{X_L}$ (S1.15.2)

Where X_L is defined as a low signal and X_H is defined as a high signal. In the first condition, we request that the total noise that is generated by the "*l*" signals is smaller than the low signal X_L. In the second condition, we inquire that the noise that is generated by the high signal (X_H) is smaller than the low signal itself. Signals often originate from the transport of discrete random carriers in systems; in biology, it is the diffusion of biochemical molecules and proteins. Naturally, these signals propagate with random fluctuations inside the networks. These fluctuations follow the Poisson process and generate shot noise that scales as the square root of the molecular count. Typically, there are two orthogonal sources of noise in any biological system (*7*, *8*). The first source is the intrinsic noise, which is inherently generated by the system itself. The second source is the extrinsic noise, which is generated by random fluctuations in the input or another environmental parameter. Here we consider the influence of the intrinsic noise only on the perceptron. A stochastic model for intrinsic cellular noise may be greater than the Poisson process, with the addition of burst size $(\boldsymbol{b_{int}})$ is given by (*9*):

$$\Delta X_L = \sqrt{(1 + b_{int}) \cdot \overline{X_L}} \quad \text{(S1.16.1)}$$

$$\Delta X_H = \sqrt{(1 + b_{int}) \cdot \overline{X_H}} \quad \text{(S1.16.2)}$$

Substituting Eq. S1.15.1 into Eq.S1.16.1 and Eq. S1.15.2 into Eq.S1.16.2, we obtain:

$$l \cdot \sqrt{(1 + b_{int}) \cdot \overline{X_L}} " \overline{X_L} \quad \rightarrow \quad l " \sqrt{\frac{\overline{X_L}}{(1+b_{int})}} \quad \text{(S1.17.1)}$$

$$\sqrt{(1 + b_{int}) \cdot \overline{X_H}} " \overline{X_L} \quad \rightarrow \quad \overline{X_H} " \frac{\overline{X_L}^2}{(1+b_{int})} \quad \text{(S1.17.2)}$$

Therefore, there is a tradeoff between number of inputs and system accuracy

X_L	b_{int}	Maximum *l*	Maximum X_H
10	0	~3	100
100	0	10	10000
10	3	1	25
100	3	5	2500
100	9	3	1000
1000	9	10	100000

Table S1.1: The relations between the maximum number of inputs that are allowed and burst size in perceptron.

The burst size relies on the translation rate, the number of amino acids (aa) in the synthesized protein, and mRNA half time. Typically, in *Escherichia coli*, the translation rate ranges between 10-20 aa/sec, depending on growth condition (*10*), and mRNA half time is around 3-5 min (*10*). Therefore, the burst size in *Escherichia coli* can be ranged between 3-15.

The main limitation for scaling the perceptgene beyond 2 inputs, is the construct itself.

Table S1.2 List of parameters used in this section

Symbol	Description
m	Hill coefficient of binding transcription factor to promoter
n	Hill coefficient of binding inducer to transcription factor
x_i	Inputs
y	Summation or multiplication of inputs (analog signal)
z_{max}	Maximal output
β	Basal level
S	Sensitivity

Table S1.3 List of abbreviations used in this section

Symbol	Description
MM	Michaelis-Menten
TFs	Transcription factors
IDR	Input dynamic range
ODR	Output dynamic range

2. Biophysical models and analysis of single perceptgene networks

In the following sections, we model the genetic network motifs using biophysical models at steady state *d/dt=0*. Our models involve detailed biochemical reactions, such as the bindings between inducers and transcription factors (TFs), as well as between TFs and promoters that consist of multiple binding sites. Our models focus on the effects of negative and positive feedback loops. These detailed biochemical models can accurately capture the behavior of the various proposed circuit topologies by solely changing the parameters that are expected to vary between experiments (e.g., plasmid copy number). In the models, we assume that the concentration of chemical species is uniformly distributed and the behaviors of our genetic circuits can be analyzed at the steady states.

2.1. Model of auto-negative feedback (ANF) loops and combinatorial promoter:

In this section, we present a model of ANF loops with a combinatorial promoter (Fig. S2.1, Fig. S2.2A), which yields the loops' behavior resembling a power-law and multiplication function. But first, we show that the experimental results of this circuit fit the power-law and multiplication function.

Fitting experimental results of ANF loops and $P_{lacO/tetO}$-based combinatorial promoter to power-law and multiplication function:

$\log(GFP) = c + n_1 \cdot \log(IPTG) + n_2 \cdot \log(aTc)$

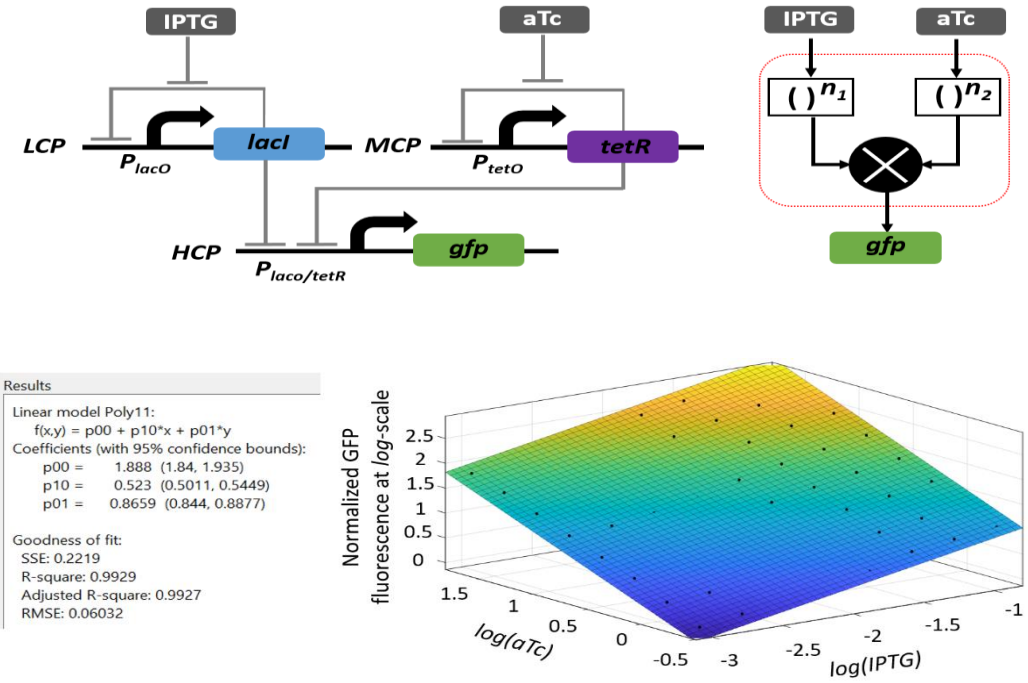


Fig. S2.1. Matlab surface fits the experimental results of P_{lacO} and P_{tetO} ANF loops and combinatorial promoter ($P_{lacO/tetO}$- GFP) to power-law and multiplication function $\log(GFP) = c + n_1 \cdot \log(IPTG) + n_2 \cdot \log(aTc)$. The data appears in Fig. 1C in the main text and is reproduced here for clarity.

The combinatorial promoter in this section includes two binding sites with different repressors. In this system, we assume that:

- The Hill coefficient of binding R_1 repressor to P_1(promoter within ANF) is equal to the Hill coefficient of binding R_1 to $P_{1/2}$(combinatorial promoter), n_1
- The Hill coefficient of binding R_2 repressor to P_2 (promoter within ANF) is equal to the Hill coefficient of binding R_2 to $P_{1/2}$ (combinatorial promoter), n_2
- The binding affinity of R_1 repressor to P_1 (promoter within ANF) is equal to binding affinity to $P_{1/2}$ (combinatorial promoter), K_{d1}
- The binding affinity of R_2 repressor to P_2 (promoter within ANF) is equal to binding affinity to $P_{1/2}$ (combinatorial promoter), K_{d2}
- The Basal levels (β_i) of P_1 , P_2 and $P_{1/2}$ are very low.

The binding of TFs to promoters is modeled according to the Shea-Ackers formalism (*11*, *12*). Therefore, the total level of expressed R_{Ti} $(i = 1,2)$ repressors in the case of ANF loops can be expressed as:

$$R_{Ti} = R_{max_i} \frac{1}{1+\left(\frac{R_i}{K_{di}}\right)^{n_i}} \quad \text{(S2. 1)}$$

Where R_{maxi} is the maximum protein level achieved by P_i which is proportional to (transcription rate × translation rate) × (mRNA half-life × protein half-life). R_i is the level of repressors that are bound to P_i. The induction of the repressors by x_i inducers is given by:

$$R_i = R_{Ti} \cdot f_i(x_i) \quad \text{(S2. 2)}$$

$$f_i(x_i) = \frac{1}{1+\left(\frac{x_i}{K_{mi}}\right)^{h_i}} \quad \text{(S2. 3)}$$

Where K_{mi} dissociation constant and h_i Hill coefficients of binding x_i to R_i. The formed new complex (Inducer-repressor) prevents the repressors from binding to P_1 and P_2. By substituting Eq. S2.1 into Eq. S2.2 we get:

$$\frac{R_i}{K_{di}} \cdot \frac{K_{di}}{f_i(x_i)} = R_{max_i} \frac{1}{1+\left(\frac{R_i}{K_{di}}\right)^{n_i}} \quad \text{(S2. 4)}$$

By developing Eq. 2.4, we get:

$$\left(\frac{R_i}{K_{di}}\right)^{n_i} = R_{max_i} \cdot \frac{f_i(x_i)}{Y_i} - 1 \quad \text{(S2. 5)}$$

The binding states for $P_{1/2}$ combinatorial promoter is shown in fig. S2.2B. The probability for $P_{1/2}$ promoter being in open complex P is described by the following equations(*11–13*):

$$P_{1/2} = \frac{1}{1+\left(\frac{R_1}{K_{d1}}\right)^{n_1}+\left(\frac{R_2}{K_{d2}}\right)^{n_2}+\theta\cdot\left(\frac{R_1}{K_{d1}}\right)^{n_1}\cdot\left(\frac{R_2}{K_{d2}}\right)^{n_2}} \quad \text{(S2. 6)}$$

Then, the expression level of the output protein at steady is given by:

$$Y = Y_{max} \cdot \frac{1}{1+\left(\frac{R_1}{K_{d1}}\right)^{n_1}+\left(\frac{R_2}{K_{d2}}\right)^{n_2}+\theta\cdot\left(\frac{R_1}{K_{d1}}\right)^{n_1}\cdot\left(\frac{R_2}{K_{d2}}\right)^{n_2}} \quad \text{(S2. 7)}$$

Where Y_{max} is the maximum protein level achieved by $P_{1/2}$ promoter. In case that the two repressors do not interfere with their bindings to $P_{1/2}$ promoter ($\theta = 1$), we substitute Eq. S2.5 into Eq. S2.7 and get:

$$Y = Y_{max} \cdot \left(\frac{R_1}{R_{max1}\cdot f_1(x_1)}\right) \cdot \left(\frac{R_2}{R_{max2}\cdot f_2(x_2)}\right) \quad \text{(S2. 8)}$$

Substituting Eq. S2.3 into Eq. S2.8, we get:

$$Y = Y_{max} \cdot \left(\frac{R_{T1}}{R_{max1}}\right) \cdot \left(\frac{R_{T2}}{R_{max2}}\right) \quad \text{(S2. 9)}$$

Therefore, in an ANF loop motif that regulates a combinatorial promoter, with $\theta = 1$, the expressed signal is effectively the multiplication of the two repressors. Substituting Eq. S2.2 into Eq. S2.1, we get:

$$R_{Ti} = R_{max_i} \frac{1}{1+\left(\frac{R_{Ti} \cdot f(x_i)}{K_{di}}\right)^{n_i}}$$

$$R_{Ti} + {R_{Ti}}^{n_i+1} \left(\frac{f(x_i)}{K_{di}}\right)^{n_i} = R_{max_i}$$

$$R_{Ti} \cdot \left(\frac{K_{di}}{f(x_i)}\right)^{n_i} + {R_{Ti}}^{n_i+1} = R_{max_i} \cdot \left(\frac{K_{di}}{f(x_i)}\right)^{n_i} \quad \text{(S2. 10)}$$

When $K_{di}$is very small (high binding affinity between repressors and promoters), we get:

$$R_{Ti} = \left(R_{max_i}\right)^{1/(n_i+1)} \cdot \left(\frac{K_{di}}{f(x_i)}\right)^{n_i/(n_i+1)} \quad \text{(S2. 11)}$$

Substituting Eq. S2.3 into Eq. S2.11:

$$R_{Ti} = \left(R_{max_i}\right)^{1/(n_i+1)} \cdot (K_{di})^{n_i/(n_i+1)} \cdot \left(1 + \left(\frac{x_i}{K_{mi}}\right)^{h_i}\right)^{n_i/(n_i+1)} \quad \text{(S2. 12)}$$

Substituting Eq. S2.12 into Eq. S2.9: we get:

$$Y = Y_{max} \cdot \left(\frac{\left(R_{max_1}\right)^{1/(n_1+1)} \cdot (K_{d1})^{n_1/(n_1+1)} \cdot \left(1 + \left(\frac{x_1}{K_{m1}}\right)^{h_1}\right)^{n_1/(n_1+1)}}{R_{max1}} \right) \cdot \left(\frac{\left(R_{max_2}\right)^{1/(n_2+1)} \cdot (K_{d2})^{n_2/(n_2+1)} \cdot \left(1 + \left(\frac{x_2}{K_{m2}}\right)^{h_2}\right)^{n_2/(n_2+1)}}{R_{max2}} \right)$$

$$\Rightarrow \quad Y = Y_{max} \cdot \left(\frac{K_{d1}}{R_{max_1}}\right)^{n_1/(n_1+1)} \cdot \left(\frac{K_{d2}}{R_{max_2}}\right)^{n_2/(n_{21}+1)} \cdot \left(1 + \left(\frac{x_1}{K_{m1}}\right)^{h_1}\right)^{n_1/(n_1+1)} \cdot \left(1 + \left(\frac{x_2}{K_{m2}}\right)^{h_2}\right)^{n_2/(n_2+1)} \quad \text{(S2. 13)}$$

By applying a logarithmic operation to Eq. S2.13, we get:

$$\log(Y) = \log(Y_{max}) + \frac{n_1}{n_1+1} \cdot log\left(\frac{K_{d1}}{R_{max_1}}\right) + \frac{n_2}{n_2+1} \cdot log\left(\frac{K_{d2}}{R_{max_2}}\right) + \frac{n_1}{n_1+1} \cdot log\left(1 + \left(\frac{x_1}{K_{m1}}\right)^{h_1}\right) + \frac{n_2}{n_2+1} \cdot log\left(1 + \left(\frac{x_2}{K_{m2}}\right)^{h_2}\right) \quad \text{(S2. 14)}$$

We define:

$$\mathrm{C} \equiv \log(Y_{max}) + \frac{n_1}{n_1+1} \cdot log\left(\frac{K_{d1}}{R_{max_1}}\right) + \frac{n_2}{n_2+1} \cdot log\left(\frac{K_{d2}}{R_{max_2}}\right) \quad \text{(S2. 15)}$$

Then, we get:

$$\log(Y) = C + \frac{n_1}{n_1+1} \cdot log\left(1 + \left(\frac{x_1}{K_{m1}}\right)^{h_1}\right) + \frac{n_2}{n_2+1} \cdot log\left(1 + \left(\frac{x_2}{K_{m2}}\right)^{h_2}\right) \quad \text{(S2. 16)}$$

In case that $x_i/K_{mi} > 1$, the ANF loops and combinatorial promoter circuit act as power-law and multiplication function, **and are linearly separable in the *log-log* scale as follows**:

$$\log(Y) = C + \frac{n_1 \cdot h_1}{n_1+1} \cdot log\left(\frac{x_1}{K_{m1}}\right) + \frac{n_2 \cdot h_2}{n_2+1} \cdot log\left(\frac{x_2}{K_{m2}}\right) \quad \text{(S2. 17)}$$

Eq. S2.17 and Fig. S2.3 show that the coefficients of the power-law functions are set by the cooperativity, number of binding sites and Hill coefficients (n_i *and* h_i). Therefore, in our circuit motif, the cooperativity effectively acts as weights in the perceptgene model, equivalent to synaptic weights in the perceptron model.

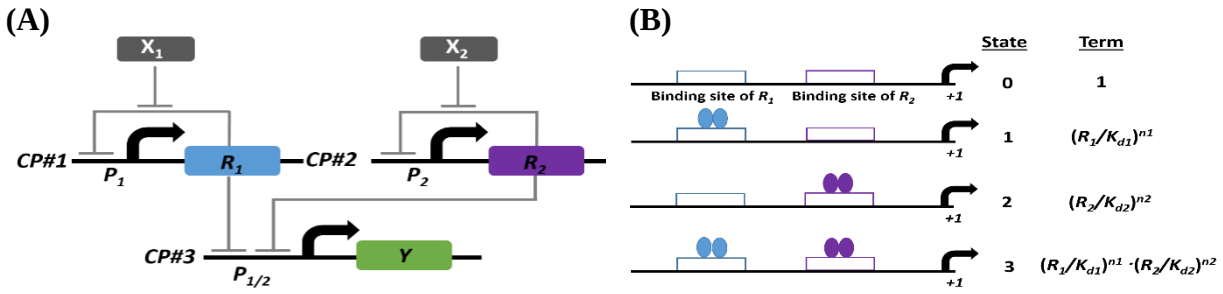


Fig. S2.2. A theoretical model of linearly separable function at the *log-log* scale. (A) ANF loops and combinatorial promoter circuit motif, CP#=Copy number of plasmids. (B) The binding states of $P_{1/2}$ hybrid promoter.

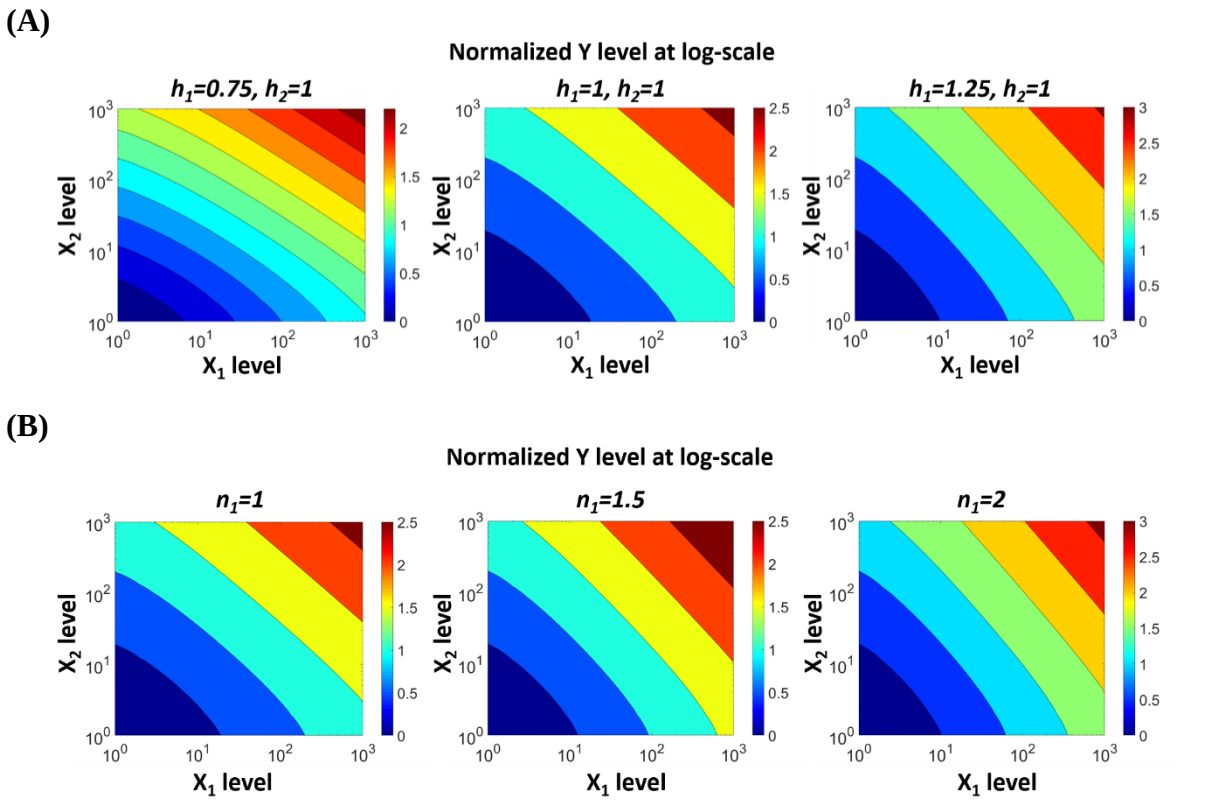


Fig. S2.3. Theoretical results of ANF loops and combinatorial promoter using Eq. S2.3. (A) Effect of h: $K_{m1} = 1, K_{m2} = 1, K_{d1} = 1, K_{d2} = 1, R_{max1} = 1000, R_{max2} = 1, n_1 = 1, n_2 = 1$. (B) Effect of n: $K_{m1} = 1, K_{m2} = 1, K_{d1} = 1, K_{d2} = 1, R_{max1} = 1000, R_{max2} = 1, n_2 = 1$.

Then we tested the influence of other biophysical and design parameters on the behavior of the system as shown in Fig. S2.4. We expanded our models to include other mechanisms such as the basal levels (β_i) of promoters, as well as the asymmetry between the combinatorial promoter ($P_{1/2}$) and ANF promoters (P_1 & P_2) :

- ANF loops that are based on Eq. S2.1 and include β_i:

$$R_{Ti} = R_{max_i} \frac{1}{1+\beta_i+\left(\frac{R_i}{K_{di}}\right)^{n_i}} \quad \text{(S2. 18)}$$

- The induction process that is based on Eq. S2.3 can be described as follows:

$$R_i = R_{Ti} \cdot \frac{1}{1+\left(\frac{x_i}{K_{mi}}\right)^{h_i}} \quad \text{(S2. 19)}$$

- For asymmetric combinatorial promoters, we assume that $K_{d1} \neq K_{d1h}$ and $K_{d2} \neq K_{d2h}$

$$Y = Y_{max} \cdot \frac{1}{1+\left(\frac{R_1}{K_{d1h}}\right)^{n_{1h}}+\left(\frac{R_2}{K_{d2h}}\right)^{n_{2h}}+\theta\cdot\left(\frac{R_1}{K_{d1h}}\right)^{n_{1h}}\cdot\left(\frac{R_2}{K_{d2h}}\right)^{n_{2h}}} \quad \text{(S2. 20)}$$

Fig. S2.4A shows the influence of increasing K_m (the dissociation constant of binding inducer X_i and repressor R_i). When K_m decreases, the input dynamic range increases.

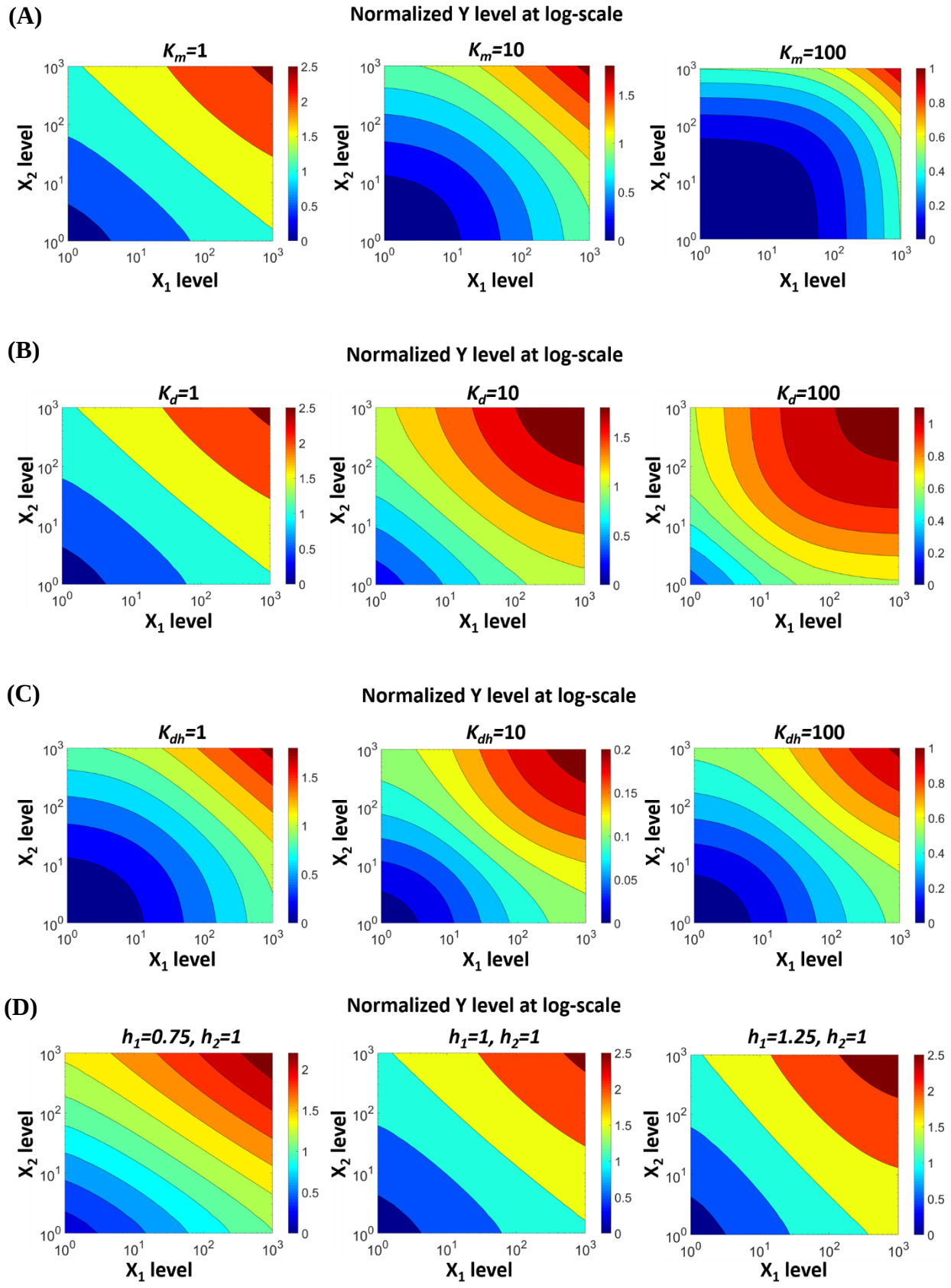
Fig. S2.4B shows the influence of increasing K_{di} (the dissociation constant of binding repressors to P_1 & P_2 promoters, respectively). When K_d increases, the input dynamic range decreases.

Fig. S2.4C shows the influence of increasing K_{dh} (the dissociation constant of binding repressors to $P_{1/2}$ combinatorial promoter), without changing the K_{di} of ANF promoters. Our simulations show that there is a tradeoff between the width of theK_{dh} and the input dynamic range. For example, the input dynamic range with $K_{dh} = 10$ is wider than the input dynamic range with $K_{dh} = 100$. To demonstrate, we cloned the combinatorial promoter on HCP and ANF loops on LCP/MCP, which gave a higher K_{dh} value compared to K_{di}.

Fig. S2.4D shows the influence of increasing h (Hill coefficient of binding inducer to repressors). When h increases, the slope of the log-log scale increases. Therefore, the Hill coefficient acts as a weight, resembling the synaptic weight in the perceptron model. These results match our theoretical model (Eq. S2.23).

Fig. S2.4E shows the influence of increasing n (Hill coefficient of binding R repressor to P_1 & P_2 promoters). When n increases (while n_h of binding repressors to $P_{1/2}$ is constant), the slope of the log-log scale increases. Remarkably the maximum protein level which is achieved by the circuit (R_{max}) has an inverse effect than decreasing K_d.

The interference between the binding of different TFs on combinatorial promoter (*13*) is represented by θ in Eq. S2.6, ($\theta = 1$ when there is no interference). When the binding between one TF and the combinatorial promoter affects the interaction between other TFs, then $\theta < 1$ (Eq. S2.6). Fig. S2.4F shows the influence of increasing θ on the ANF loops and combinatorial promoter circuit. As in Fig. S2.4F, even for a very low θ, which means a high interference between the two repressors, the input-output transfer function of the circuit motif can be fitted to a power-law and multiplication function.


(A)
Normalized Y level at log-scale
K_m=1
K_m=10
K_m=100
(B)
Normalized Y level at log-scale
K_d=1
K_d=10
K_d=100
(C)
Normalized Y level at log-scale
K_{dh}=1
K_{dh}=10
K_{dh}=100
(D)
Normalized Y level at log-scale
h_1=0.75, h_2=1
h_1=1, h_2=1
h_1=1.25, h_2=1
X1 level
X2 level

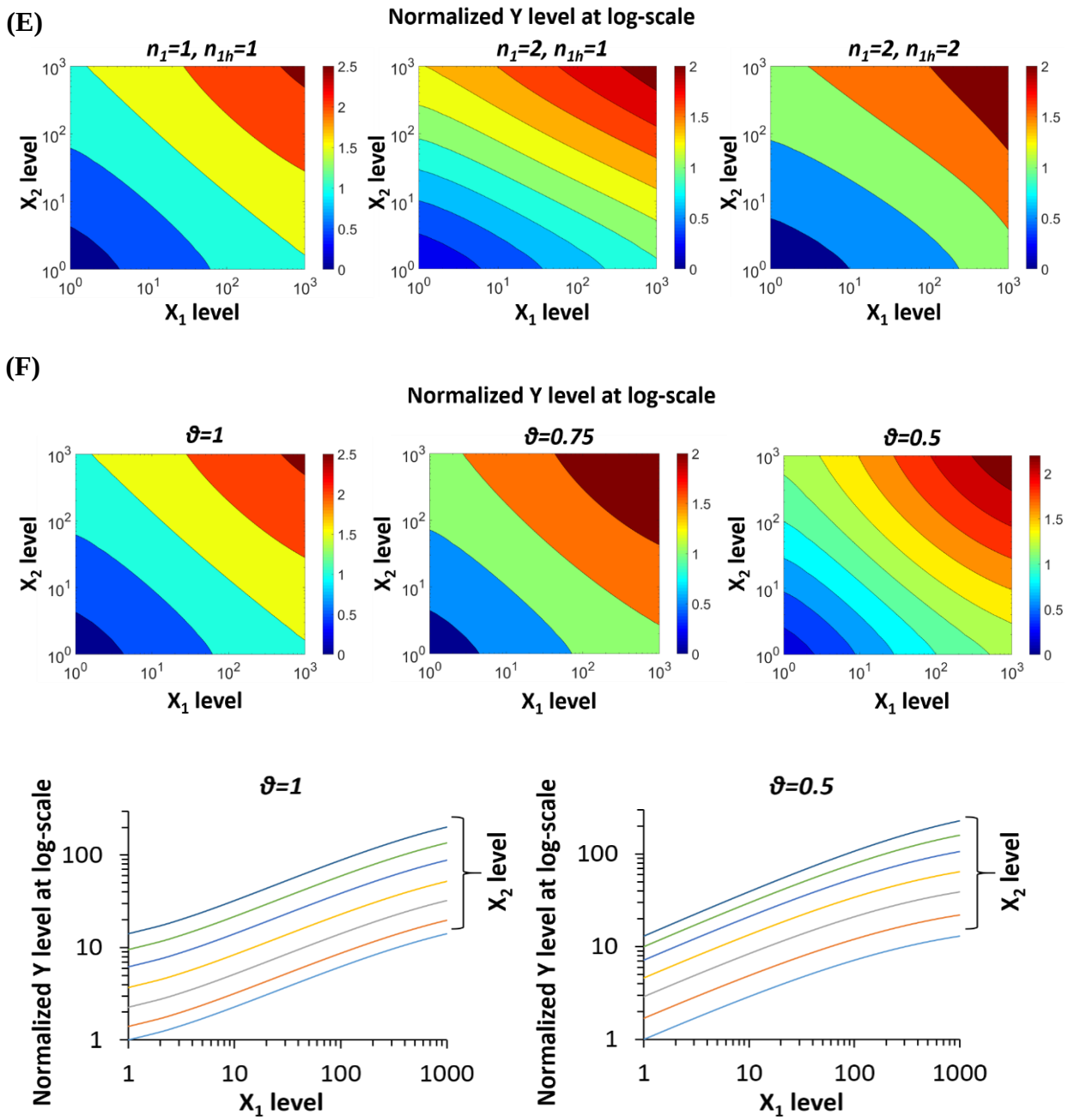


Fig. S2.4. Simulation results of ANF loops and combinatorial promoter circuit motif.

A. Effect of K_m: $K_m = K_{m1} = K_{m2}, h_1 = 1, h_2 = 1, k_{d1} = 1, K_{d2} = 1, K_{d1h} = 1, k_{d2h} = 1, R_{max1} = 1000, R_{max2} = 1, n_1 = 1, n_2 = 1, n_{1h} = 1, n_{2h} = 1, \theta = 1, \beta = 0.001$.

B. Effect of K_d: $K_d = K_{d1} = K_{d2} = K_{d1h} = K_{d2h}, K_{m1} = 1, K_{m2} = 1,\ h_1 = 1, h_2 = 1, R_{max1} = 1000, R_{max2} = 1, n_1 = 1, n_2 = 1, n_{1h} = 1, n_{2h} = 1, \theta = 1, \beta = 0.001$.

C. Effect of K_{dh}: $K_{dh} = K_{d1h} = K_{d2h}, k_{d1} = 1, K_{d2} = 1, K_{m1} = 1, K_{m2} = 1,\ h_1 = 1, h_2 = 1,\ R_{max1} = 1000, R_{max2} = 1, n_1 = 1, n_2 = 1, n_{1h} = 1, n_{2h} = 1, \theta = 1, \beta = 0.001$.

D. Effect of h: $K_{m1} = 1, K_{m2} = 1, k_{d1} = 1, K_{d2} = 1, K_{d1h} = 1, k_{d2h} = 1, R_{max1} = 1000, R_{max2} = 1, n_1 = 1, n_2 = 1, n_{1h} = 1, n_{2h} = 1, \theta = 1, \beta = 0.001$.

E. Effect of n: $K_{m1} = 1, K_{m2} = 1, k_{d1} = 1, K_{d2} = 1, K_{d1h} = 1, k_{d2h} = 1, R_{max1} = 1000, R_{max2} = 1, h_1 = 1, h_2 = 1, n_{1h} = 1, n_{2h} = 1, \theta = 1, \beta = 0.001.$

F. Simulation results show the influence of interference between transcription factors (TFs) on the ANF loops and combinatorial promoter circuit motif.
$K_m = K_{m1} = K_{m2}, h_1 = 1, h_2 = 1, k_{d1} = 1, K_{d2} = 1, K_{d1h} = 1, k_{d2h} = 1, R_{max1} = 1000,\ R_{max2} = 1, n_1 = 1, n_2 = 1, n_{1h} = 1, n_{2h} = 1, \beta = 0.001.$

2.2. Computed transfer function of power-law and multiplication function based on ANF

The simulations are based on Eq. S2.18, Eq. S2.19, and Eq. S2.20. Parameters that were used in simulations:

Based on P_{lacO} within ANF loop – Fig. 1E

$K_{m1} = 0.8, K_{m2} = 1, K_{d1} = 10, K_{d2} = 5, K_{d1h} = 45, K_{d2h} = 4, h_1 = 1, h_2 = 1.4, R_{max1} = 2000, R_{max2} = 3000, \boldsymbol{n_1 = 2}, n_2 = 2, n_{1h} = 1, n_{2h} = 1, \theta = 1, \beta = 0.001$

Based on P_{lacO1} within ANF loop – Fig. 1F

$K_{m1} = 0.8, K_{m2} = 1, K_{d1} = 90, K_{d2} = 6, K_{d1h} = 45, K_{d2h} = 4, h_1 = 1, h_2 = 1.4, R_{max1} = 2000, R_{max2} = 3000, \boldsymbol{n_1 = 1}, n_2 = 2, n_{1h} = 1, n_{2h} = 1, \theta = 1, \beta = 0.001$

The parameters that were used in our simulation fit well with the values that were reported in the literature. For example, the binding dissociation constant of LacI is known to be 10 times larger than of TetR. The interference parameter was set to $\theta = 1$. Furthermore, the maximum level of protein achieved by P_{lacO} in our simulation is smaller than P_{tetO} ($R_{max1} < R_{max2}$), which is consistent with our construction that P_{lacO} is located on LCP and P_{tetO} and is located on MCN. The binding dissociation constant of LacI and TetR to their promoters within the ANF loop (P_{lacO} and P_{tetO}, respectively) is different than their values within the combinatorial promoter, because the promoters were located on different plasmid copy numbers.

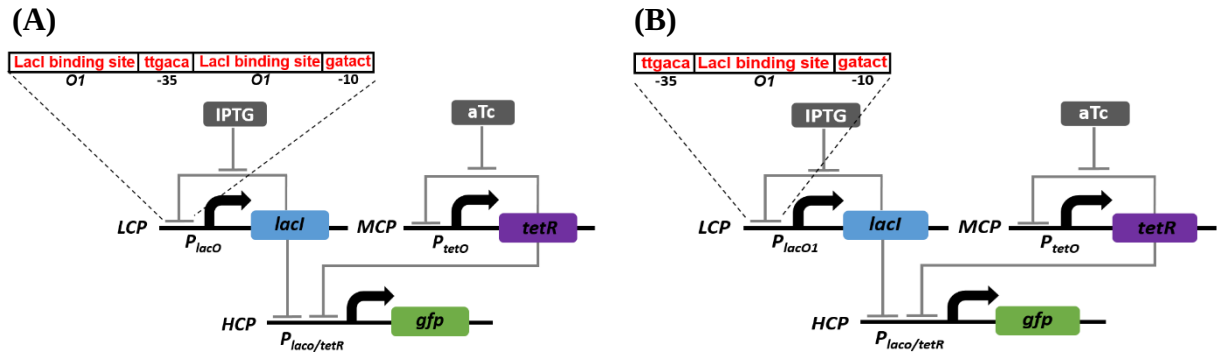


Fig. S2.5. (A) The circuit that was used to produce the computed transfer function from Fig. 1E, power-law and multiplication based on P_{lacO}. (B) The circuit that was used to produce the computed transfer function from Fig. 1F, power-law and multiplication based on P_{lacO1}.

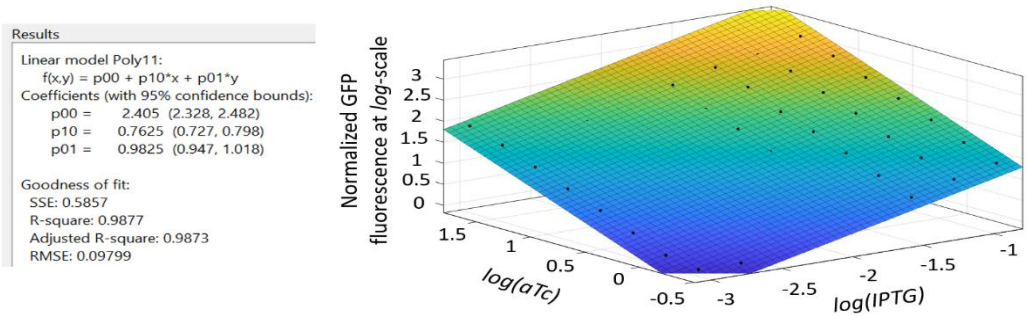


Fig. S2.6. Matlab surface fits the experimental results of P_{lacO1} and P_{tetO} ANF loops and combinatorial promoter ($P_{lacO/tetO}$ - GFP) to power-law and multiplication function $\log(GFP) = c + n_3 \cdot \log(IPTG) + n_4 \cdot \log(aTc)$. The data appears in Fig. 1D in the main text and is reproduced here for clarity.

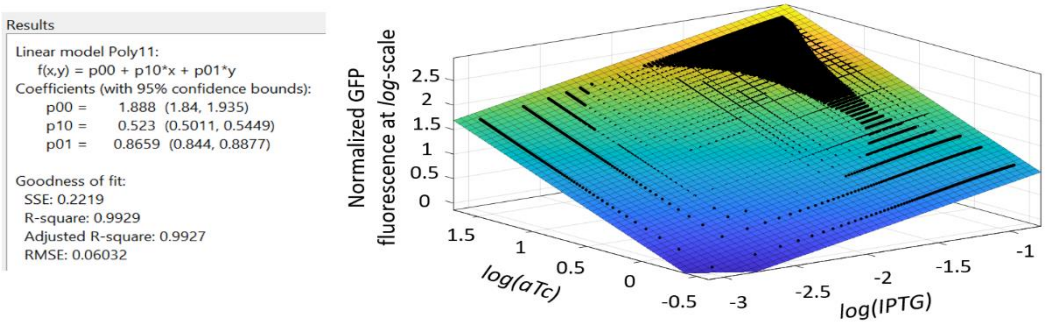


Fig. S2.7. Matlab surface fits the simulation results of P_{lacO} and P_{tetO} ANF loops and combinatorial promoter ($P_{lacO/tetO}$- GFP) to power-law and multiplication function $\log(GFP) = c + n_1 \cdot \log(IPTG) + n_2 \cdot \log(aTc)$. The data appears in Fig. 1E in the main text and is reproduced here for clarity.

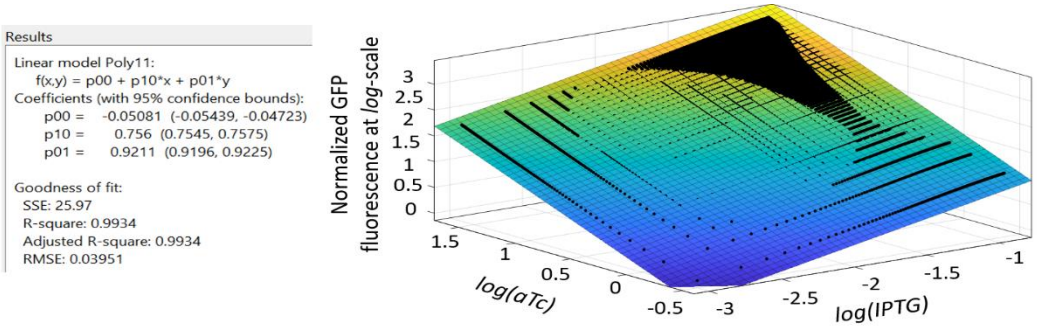


Fig. S2.8. Matlab surface fits the simulation results of P_{lacO1} and P_{tetO} ANF loops and combinatorial promoter ($P_{lacO/tetO}$- GFP) to power-law and multiplication function $\log(GFP) = c + n_3 \cdot \log(IPTG) + n_4 \cdot \log(aTc)$. The data appears in Fig. 1F in the main text and is reproduced here for clarity.

2.2.1. A simple mode for Auto-negative feedback

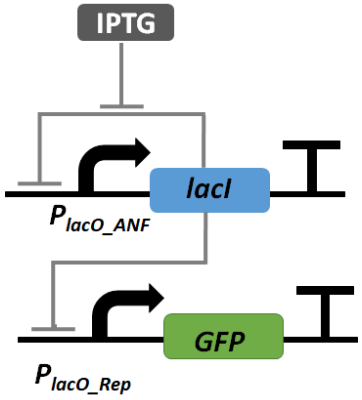


Fig. S2.8.1. Auto-negative feedback loop circuit.

$$LacI_T = \frac{\alpha_{11}\cdot\tau_1}{1+\left(\frac{LacI}{K_d}\right)^{n_{11}}} \quad \text{(S2. 20.1)}$$

α_{11} is the production rate of LacI by the P_{lacO_ANF}
n_{11} is the Hill-coefficient of binding LacI to P_{lacO_ANF} promoter
K_d is the binding affinity of LacI with P_{lacO_ANF} promoter (promoter within ANF), which equals (our assumption) to the binding affinity of LacI with P_{lacO_REP} promoter regulating GFP
τ_1 is LacI half-life

$$LacI = LacI_T \cdot f(IPTG) \quad \text{(S2. 20.2)}$$

$$f(IPTG) = \frac{IPTG^{h_1}}{{K_m}^{h_1}+IPTG^{h_1}}$$

h_1 is the Hill-coefficient of binding IPTG (x_1) to LacI (R_1)

$$GFP = \frac{\alpha_{21}\cdot\tau_2}{1+\left(\frac{LacI}{K_d}\right)^{n_{21}}} \quad \text{(S2. 20.3)}$$

α_{21} is the production rate of GFP by the P_{lacO_Rep}
n_{21} is the Hill-coefficient of binding LacI to P_{lacO_Rep} promoter
τ_2 is GFP half-life

$$LacI_T + {LacI_T}^{n_{11}+1}\left(\frac{LacI_T\cdot f(IPTG)}{K_d}\right)^{n_{11}} = \alpha_{11}\cdot\tau_1 \quad \text{(S2. 20.4)}$$

$$LacI + K_d\cdot\left(\frac{LacI}{K_d}\right)^{n_{11}+1} = f(IPTG)\cdot\alpha_{11}\cdot\tau_1 \quad \text{(S2. 20.5)}$$

For $K_d \ll \alpha_{11}\cdot\tau_{11}$

$$\rightarrow LacI_T = (\alpha_{11}\cdot\tau_1)^{1/(n_{11}+1)}\cdot\left(\frac{K_d}{f(IPTG)}\right)^{\frac{n_{11}}{n_{11}+1}} \quad \text{(S2. 20.6)}$$

$$\rightarrow LacI = K_d\cdot\left(\frac{Z_{max}\cdot f(IPTG)}{K_d}\right)^{1/(n_{11}+1)} \quad \text{(S2. 20.7)}$$

$$\Rightarrow \quad GFP = \frac{\alpha_{21}\cdot\tau_2}{1+\left(\frac{\alpha_{11}\cdot\tau_1\cdot f(IPTG)}{K_d}\right)^{n_{21}/(n_{11}+1)}} \qquad \text{(S2. 20.8)}$$

$$\Rightarrow \quad GFP = \frac{GFP_{max}}{1+\left(\frac{LacI_{max}\cdot f(IPTG)}{K_d}\right)^{n_{21}/(n_{11}+1)}} \qquad \text{(S2. 20.9)}$$

$f(\text{IPTG})$ is the induction function, $GFP_{max} = \alpha_{21}\cdot\tau_2, \quad LacI_{max} = \alpha_{11}\cdot\tau_1$
. In case that $n_{21} = 1$ (e.g., $P_{lacO/tetO}$- GFP), the Hill- coefficient of $\text{IPTG} - \text{GFP}$ is proportional to $1/(n_{11}+1)$. Increasing the n_{11} leads to decrease the Hill coefficient of $\text{IPTG} - \text{GFP}$ transfer function.

2.3. Characterization of P_{BAD} promoter

It has been reported that the behavior of the Arabinose-inducible promoter P_{BAD} is strongly affected by the arabinose concentration (*14*). Using the characteristic of P_{BAD}, we constructed a graded auto-positive feedback (APF) circuit (Fig. S2.9A) to tune the expression level of AraC. The purpose of the graded APF is to increase the dynamic range of P_{lux} which can regulate the AraC level to a very wide range. The analysis for the APF circuit and linearization is provided in the next sections. We added a ssrA degradation tag (*15*) (LAA) to AraC to ensure low basal in the absence of the input (AHL). The circuit was first induced with different concentrations of Arabinose (0.7mM, 0.2mM, 0.07mM and 0.02mM). Then the experimental results (Fig. S2.9B) were fitted to Hill-function ($a\cdot\frac{AHL^{m_{eff}}}{AHL^{m_{eff}}+K_{eff}^{m_{eff}}}+b$). Our experimental results show that in the presence of a low level of Arabinose (0.02mM), the input dynamic range of AHL to P_{BAD} decreased with $m_{eff} = 3, K_{eff} = 120nM, a = 165\ (a.u.), b = 1\ (a.u.)$. Whereas with a high level of Arabinose (0.7mM), the input dynamic range increased with $m_{eff} = 1.5, K_{eff} = 200nM, a = 170\ (a.u.), b = 1\ (a.u.)$. As a result, the Hill-coefficient of binding Arabinose-AraC complex to P_{BAD} increases when Arabinose is decreased. The relation between the effective dissociation constant K_{eff} and m_{eff} is summarized in Fig. S2.9C and can be well fitted using the power-law function ($K_{eff} = a\times {m_{eff}}^{-b}$). The transfer function of P_{BAD} promoter with respect to AraC is shown in Fig. S2.9D. The level of AraC is evaluated by the GFP signal from Fig. S2.13C.

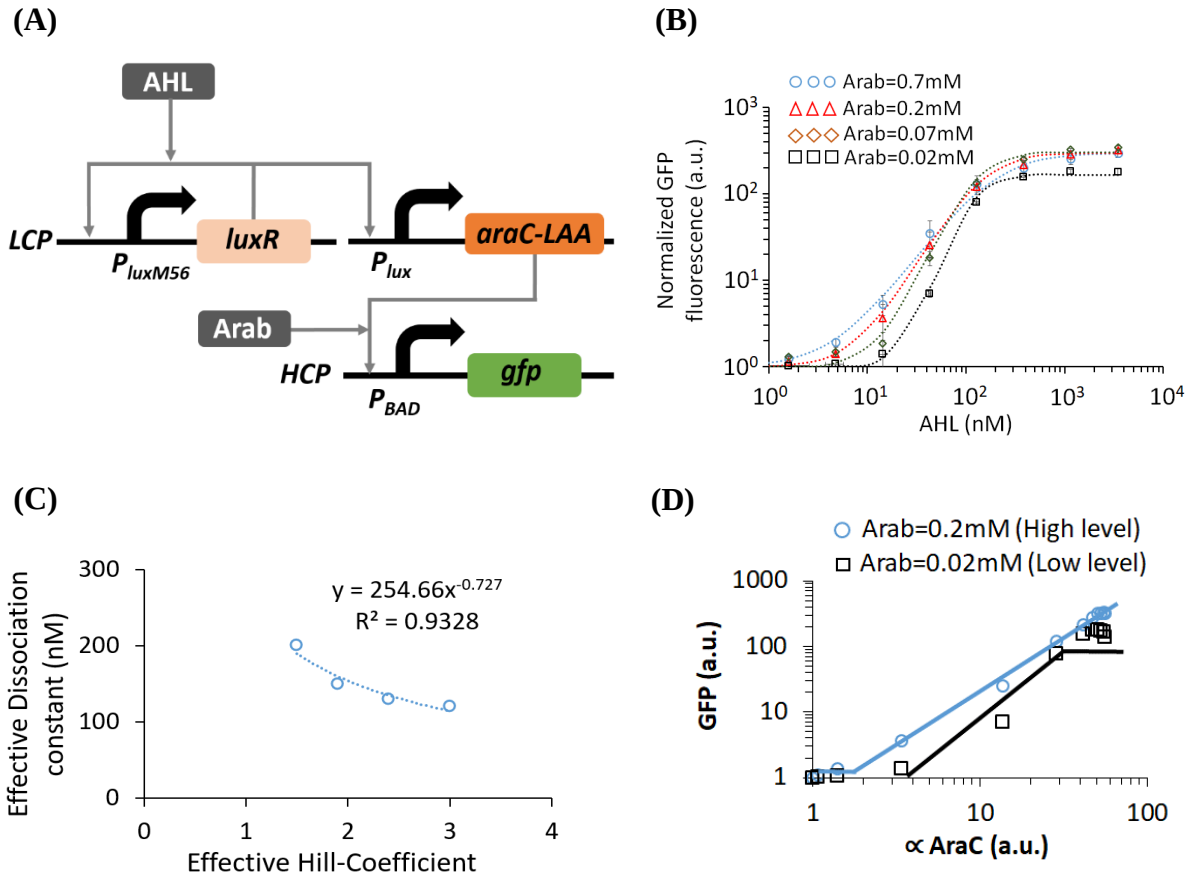


Fig. S2.9. (A) The Characterization of P_{BAD} promoter by tuning the expression level of AarC by mutated APF. (B) Experimental results of AHL – GFP transfer function for a low Arabinose (0.02mM) and a high Arabinose (0.2mM). The dotted lines are a Hill-function fitting. (C) The relation between the effective dissociation constant and effective Hill-coefficient is described by a power-law function. (D) AraC -P_{BAD} the transfer function for low and high Arabinose levels.

<u>Nonlinear fitting models of AraC -based synthetic perceptgene circuit</u>

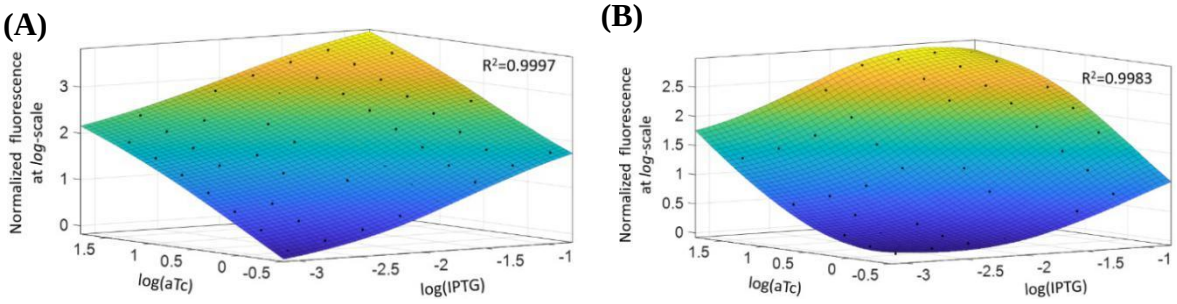


Fig. S2.10. (A) Fitting experimental results of ANF loops and $P_{lacO1/tetO}$-based combinatorial promoter to non-linear models (Quadratic) using locally weighted scatterplot smoothing (Circuit from Fig. 1B, Data based on Fig. 1D). (B) Fitting experimental results of perceptgene based ANF loops and $P_{lacO1/tetO}$ combinatorial promoter to non-linear models (Quadratic) using locally weighted scatterplot smoothing (Circuit from Fig. 1G, Data based on Fig. 1H).

2.4. Model of AraC -based synthetic perceptgene circuit

The P_{BAD} promoter is activated by the AraC -TF when it is induced by arabinose (Arab). The probability of the P_{BAD} promoter being induced by the arabinose-AraC complex is described by (*4*):

$$P = \frac{\frac{AraC_c}{K_{d3}} + \beta_4}{1 + \beta_4 + \frac{AraC_c}{K_{d3}} + \frac{AraC}{K_{d4}}} \quad \text{(S2. 21)}$$

where $AraC_C$ is the concentration of the Arabinose-AraC complex, AraC is the concentration of free AraC −TF, K_{d3} is the dissociation constant for binding of the Arabinose- AraC complex to P_{BAD} promoter, K_{d4} is the dissociation constant for free AraC binding to P_{BAD}, and β_4 is the basal level of P_{BAD} promoter. The concentration of the arabinose- AraC complex is given by (*4*, *16*):

$$AraC_c = AraC_T \cdot \frac{\left(\frac{Arab}{K_{m3}}\right)^{h_3}}{1 + \left(\frac{Arab}{K_{m3}}\right)^{h_3}} \quad \text{(S2. 22)}$$

With, $AraC_T$ is the total concentration of AraC, K_{m3} is the dissociation constant of binding arabinose to AraC and h_3 is the Hill coefficient (~2.8 (*16*)). The concentration of the free AraC is given by:

$$AraC = AraC_T - AraC_c \quad \text{(S2. 23)}$$

In the perceptgene circuit the $AraC_T$ is equal to the output of the power-law and multiplication function (Y) at steady state:

$$AraC_T = Y \quad \text{(S2. 24)}$$

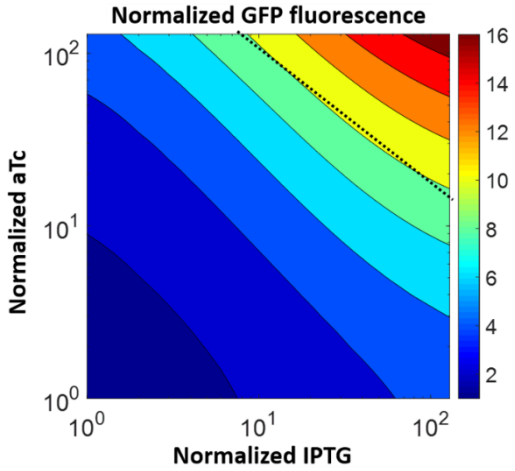


Fig. S2.11. The computed transfer function of synthetic perceptgene based on AraC system. Parameters that were used in simulation: $AraC_T = 30$, $K_{m3} = 0.09$, $K_{d3} = 3$, $K_{d4} = 30$, $h_3 = 2.8$ (*16*), $\beta_4 = 0.002$. The ratio between K_{d3} and K_{d4} fits well to the values that were reported in the literature (*4*).

2.5. Model of APF, ANF loops and combinatorial promoter

In this section, we present a model that describes the behavior of APF, ANF loops with a combinatorial promoter (Fig. S2.12A), which results in power-law and multiplication function:

- The Hill coefficient of binding A activator to P_1 (promoter within APF) is equal to Hill coefficient of binding A to $P_{1/2}$(combinatorial promoter) and are equal to, n_1
- The Hill coefficient of binding R repressor to P_2 (promoter within ANF) is equal to Hill coefficient of binding R to $P_{1/2}$ (combinatorial promoter), n_2
- The binding affinity of A activator to P_1 (within APF) is equal to binding affinity to $P_{1/2}$ (combinatorial promoter), K_{d1}
- The binding affinity of R repressor to P_2 (within ANF) is equal to binding affinity to $P_{1/2}$ (combinatorial promoter), K_{d2}
- The β of $P_{1/2}$ is very low.

The binding states for $P_{1/2}$ combinatorial promoter is shown in Fig. S2.12B. The probability for $P_{1/2}$ promoter being in the open complex is described by the following equations:

$$P_{1/2} = \frac{\left(\frac{A}{K_{d1}}\right)^{n_1}}{1+\left(\frac{A}{K_{d1}}\right)^{n_1}+\left(\frac{R}{K_{d2}}\right)^{n_2}+\theta\cdot\left(\frac{A}{K_{d1}}\right)^{n_1}\cdot\left(\frac{R}{K_{d2}}\right)^{n_2}} \quad \text{(S2. 25)}$$

For simplicity, we assumed that $n_1 = 1$. Then, the expression level of the output protein at a steady state is given by:

$$Z = Z_{max}\cdot\frac{\frac{A}{K_{d1}}}{1+\frac{A}{K_{d1}}+\left(\frac{R}{K_{d2}}\right)^{n_2}+\theta\cdot\frac{A}{K_{d1}}\cdot\left(\frac{R}{K_{d2}}\right)^{n_2}} \quad \text{(S2. 26)}$$

Where Z_{max} is the maximum protein level achieved by $P_{1/2}$ promoter.

A graded Positive feedback model: The first step toward implementation of synthetic power-law and multiplication function in living cells, is to broaden the input dynamic range of genetic synthetic parts. It has shown that a graded PF loop increased the input dynamic range by more than three orders of magnitude (*4*) (Fig. S2.12C). TFs bindings to promoters are modeled according to the Shea-Ackers formalism (*11*, *12*). Therefore, the total level of expressed A activator (Fig. S2.12C) in the case of APF loops can be expressed as:

$$A_T = A_{max}\cdot\frac{\frac{A}{K_{d1}}+\beta_1}{1+\beta_1+\frac{A}{K_{d1}}} \quad \text{(S2. 27)}$$

$$Y = Y_{max}\cdot\frac{\frac{A}{K_{d2}}+\beta_2}{1+\beta_2+\frac{A}{K_{d2}}} \quad \text{(S2. 28)}$$

Where A_{max} and Y_{max} are the maximum protein levels achieved by P_1 and P_2 respectively, β_1 and β_2 are the Basal levels of P_1 and P_1 respectively, and A is the level of activators that are bound to P_1 and P_2. The induction of the activator by x inducers is given by:

$$A = A_T\cdot g(x) \quad \text{(S2. 29)}$$

$$g(x) = \frac{\left(\frac{x}{K_m}\right)^h}{1+\left(\frac{x}{K_m}\right)^h} \quad \text{(S2. 30)}$$

Where K_m is the dissociation constant and h the Hill coefficient of binding inducer to activator. Substituting Eq. S2.29 into Eq. S2.27 we get:

$$A = A_{max} \cdot \frac{\frac{A_T \cdot g(x)}{K_{d1}} + \beta_1}{1 + \beta_1 + \frac{A_T \cdot g(x)}{K_{d1}}} \quad \text{(S2. 31)}$$

$$A_T + A_T \cdot \beta_1 + \frac{{A_T}^2 \cdot g(x)}{K_{d1}} = A_{max} \cdot \frac{A_T \cdot g(x)}{K_{d1}} + A_{max} \cdot \beta_1$$

$$A_T \cdot \frac{K_{d1}}{g(x)} + A_T \cdot \beta_1 \cdot \frac{K_{d1}}{g(x)} + {A_T}^2 = A_{max} \cdot A_T + A_{max} \cdot \beta_1 \cdot \frac{K_{d1}}{g(x)}$$

$$A_T \cdot \left(\frac{K_{d1}}{g(x)} + \beta_1 \cdot \frac{K_{d1}}{g(x)} - A_{max}\right) + {A_T}^2 = A_{max} \cdot \beta_1 \cdot \frac{K_{d1}}{g(x)}$$

$$A_T \cdot \frac{K_{d1}}{g(x)} \cdot \left(1 + \beta_1 - \frac{A_{max} \cdot g(x)}{K_{d1}}\right) + {A_T}^2 = A_{max} \cdot \beta_1 \cdot \frac{K_{d1}}{g(x)} \quad \text{(S2. 32)}$$

In case that $A_{max}/K_{d1}<1$, we can approximate Eq. S2.32 as:

$$A_T \approx \frac{A_{max} \cdot \beta_1}{\left(1 + \beta_1 - \frac{A_{max}}{K_{d1}} \cdot g(x)\right)} \quad \text{(S2. 33)}$$

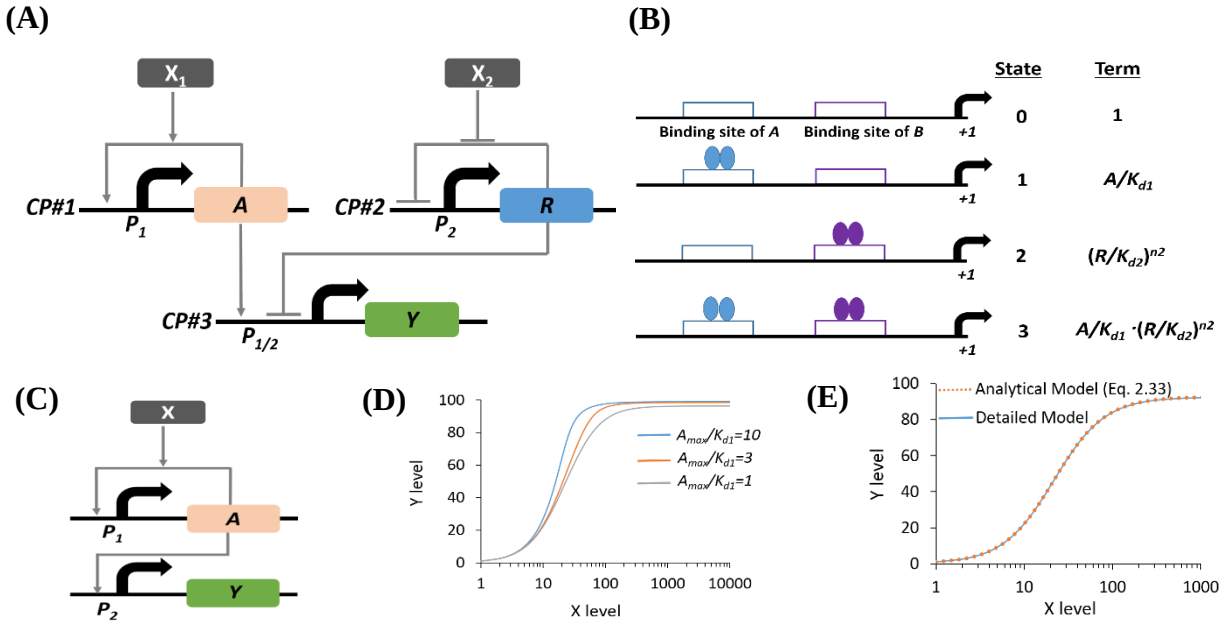


Fig. S2.12. (A) APF and ANF loops combined with a hybrid combinatorial promoter, CP#=Copy number of plasmids. (B) The binding states of $P_{1/2}$ promoter. (C) APF loop circuit. (D) Simulation results of APF loop circuit using a detailed model $K_m = 100\,, m = 1.5, A_{max} = 1000, \beta_1 = 0.1, K_{d2} = 10, Z_{max} = 1, \beta_2 = 0.01$. (E) The detailed and analytical models of APF ($K_d = 3000$).

The Simulation results for the exact model of APF (Eq. S2.27-Eq. S2.30) and the approximated model based on Eq. S2.33 are shown in Fig. S2.12D and 2.12E. When the ratio A_{max}/K_d decreases, the input dynamic range increases.

By applying a logarithmic operation to Eq. S2.33, we get:

$$\log(A_T) \approx log(A_{max} \cdot \beta_1) - log\left(1 + \beta_1 - \frac{A_{max}}{K_{d1}} \cdot g(x)\right) \quad \text{(S2. 34)}$$

By substituting Eq. S2.30 into Eq. S2.34, and assuming that β_1"1, we get:

$$\log(A_T) \approx log(A_{max} \cdot \beta_1) + log\left(1 + \left(\frac{x}{K_m}\right)^h\right) - log\left(1 + \left(\frac{x}{K_m}\right)^h \cdot \left(1 - \frac{A_{max}}{K_{d1}}\right)\right) \quad \text{(S2. 35)}$$

In case that $x/K_m >> 1$ and $\left(\frac{x}{K_m}\right)^h \cdot \left(1-\frac{A_{max}}{K_{d1}}\right) < 1$, we can approximate Eq. S2.35 as:

$$\log(A_T) \approx log(A_{max}\cdot\beta_1) + h\cdot log\left(\frac{x}{K_m}\right) - \left(1-\frac{A_{max}}{K_{d1}}\right)\cdot\left(\frac{x}{K_m}\right)^h \quad \text{(S2. 36)}$$

$$\log(A_T) \approx log(A_{max}\cdot\beta_1) + h\cdot log\left(\frac{x}{K_m}\right) - \left(1-\frac{A_{max}}{K_{d1}}\right)\cdot e^{h\cdot ln\left(\frac{x}{K_m}\right)}$$

$$\log(A_T) \approx log(A_{max}\cdot\beta_1) + h\cdot log\left(\frac{x}{K_m}\right) - \left(1-\frac{A_{max}}{K_{d1}}\right)\cdot\left(1+h\cdot ln\left(\frac{x}{K_m}\right)\right)$$

$$\log(A_T) \approx log(A_{max}\cdot\beta_1) + h\cdot log\left(\frac{x}{K_m}\right) - 2.3\cdot\left(1-\frac{A_{max}}{K_{d1}}\right)\cdot\left(\frac{1}{2.3}+h\cdot log\left(\frac{x}{K_m}\right)\right)$$

$$\log(A_T) \approx log(A_{max}\cdot\beta_1) - \left(1-\frac{A_{max}}{K_{d1}}\right) + \left[1-2.3\cdot\left(1-\frac{A_{max}}{K_{d1}}\right)\right]\cdot h\cdot log\left(\frac{x}{K_m}\right) \quad \text{(S2. 37)}$$

Eq. S2.37 shows that a graded APF loop, when $A_{max}/K_d << 1$, can be approximated as a power-law function. The power-law coefficient is mainly set by Hill coefficient h and the ratio of A_{max}/K_d. The simulation results of Fig. S2.12D show that the power-law coefficient in the case of strong APF ($A_{max}/K_d = 10$) is 1.5, and for a graded APF ($A_{max}/K_d = 1$) is 1.35.

Experimental Results of APF: To test our approach, we first created various synthetic libraries that permute the sequence features affecting DNA binding site affinity. This was achieved by creating random mutations in the TF-DNA binding site sequence within the promoter. The synthetic P_{lux} promoter was selected due to its simple structure (*16*). The promoter consists of a single LuxR binding site upstream to the -35 location. First, we constructed an open-loop gene circuit consisting of two components: a constitutive promoter regulating the expression of the LuxR gene, and a P_{lux} promoter regulating the expression of GFP (Fig. S2.13A). Then we ran a random mutation on the first 7 nucleotides of the LuxR binding site sequence (*17*), resulting in a new promoter called (P_{luxM56}). To test the new promoter, we reconstructed an open-loop and APF circuits with the P_{luxM56} promoter (Fig. S2.13B). The positive feedback circuit consisting of a positive feedback loop based on the a mutated P_{lux} promoter regulates the expression of the LuxR gene and a wild type P_{lux} promoter, which regulates the expression of GFP. As shown in Fig. S2.13C, the mutated promoters (P_{luxM56} in open circuit) exhibited weaker TF-DNA binding than the wild type promoter (P_{lux} in open circuit), with a lower GFP signal and a wider input dynamic range. In particular, the mutated promoter (P_{luxM56}) gives rise a graded APF transfer function with a broad region of linearity for more than four orders of magnitude without losing its magnitude. The measured transfer functions of multiple circuits were fitted using Hill function $a\cdot\frac{\left(\frac{AHL}{k}\right)^{meff}}{\left(1+\left(\frac{AHL}{k}\right)^{meff}\right)}+b$ (Fig. S2.13C). The strength of a PF loop, which is set by the dissociation constant of binding LuxR to P_{lux} or P_{luxM56}, affects the input dynamic range and the effective Hill coefficient meff, as well as the power-law coefficient. Our experimental results showed that the influence of dissociation constant on m_{eff} is much larger than our theoretical analysis. This is because our theoretical analysis is based on Michaelis-Menten model, which assumes that the TF concentration is much larger than promoter concentration. It has been shown that, when these assumptions are violated, detailed biochemical reaction models can capture the behavior of graded APF accurately (*4*). Fig. S2.14 shows the experimental results of $AHL-GFP$ transfer function for other mutated P_{lux} promoter.

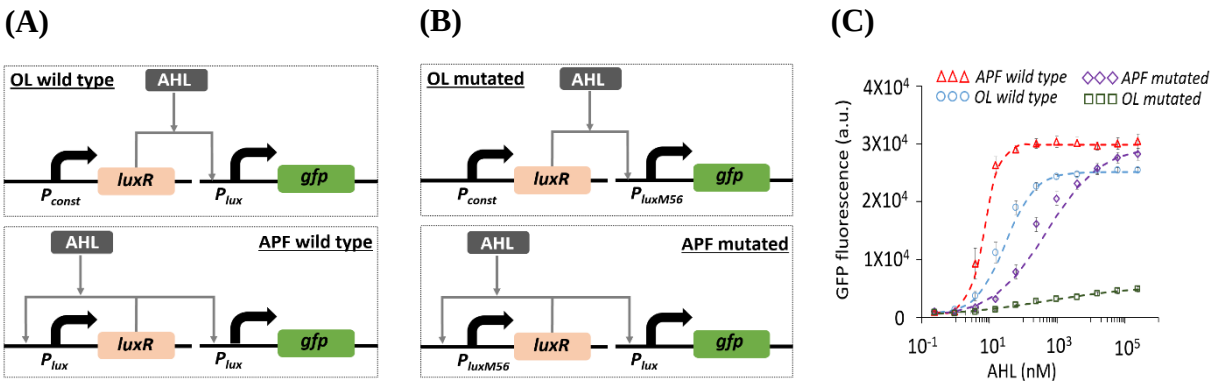


Fig. S2.13. (A) The construction of open-loop (OL) and APF circuits based on P_{lux} promoter. (B) The construction of OL and APF circuits based on mutated P_{lux} promoter (P_{luxM56}). (C) Measured transfer functions of multiple circuits, dots are experimental data, and dashed-line is a Hill function fitting with the below parameters:

OL circuit – Wild type P_{lux}: $K = 30, m_{eff} = 1, a = 25 \times 10^3, b = 600$

APF circuit – Wild type P_{lux}: $K = 7, m_{eff} = 2, a = 30 \times 10^3, b = 800$

OL circuit – Mutated P_{luxM56}: $K = 500, m_{eff} = 0.3, a = 5 \times 10^3, b = 100$

APF circuit – Mutated P_{luxM56}: $K = 500, m_{eff} = 0.5, a = 30 \times 10^3, b = 100$

Fig. S2.14 shows the experimental results of AHL − GFP transfer function for other mutated P_{lux} promoter.

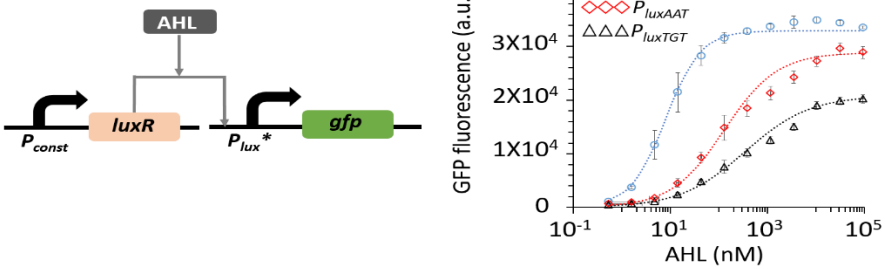


Fig. S2.14. AHL − GFP transfer function of mutated P_{lux} promoters (P_{luxAAT}, P_{luxTGT}). Dots are experimental data, and the dashed line is a Hill function fitting.

Power-law and multiplication function based on ANF and graded APF loops: The combination of graded APF and ANF loops with a combinatorial promoter is shown in Fig. S2.12A. A set of equations that describes the behavior of this system at a steady state is given by:

- APF loop equation:

$$A_{T1} = A_{max1} \cdot \frac{\frac{A}{K_{d1}} + \beta_1}{1 + \beta_1 + \frac{A}{K_{d1}}} \qquad \text{(S2. 38)}$$

- Induction of X_1 (AHL) and Y_1 (LuxR) equation (*4*):

$$A \approx A_T \frac{{X_1}^2}{{X_1}^2 + 2 \cdot X_1 \cdot A_T + {K_{m1}}^2} \qquad \text{(S2. 39)}$$

- ANF loops equation:

$$R_T = R_{max}\frac{1}{1+\beta_2+\left(\frac{R}{K_{d2}}\right)^{n_2}} \quad \text{(S2. 40)}$$

- Induction of X_2 and R:

$$R = R_T \cdot \frac{1}{1+\left(\frac{x_2}{K_{m2}}\right)^{h_2}} \quad \text{(S2. 41)}$$

- The a ctivity of combinatorial promoter:

$$P_{12} = \frac{\frac{A}{K_{dh1}}+\beta_{12}}{1+\beta_{12}+\frac{A}{K_{dh1}}+\left(\frac{R}{K_{dh2}}\right)^{n_2}+\theta\cdot\frac{A}{K_{dh1}}\cdot\left(\frac{R}{K_{dh2}}\right)^{n_2}} \quad \text{(S2. 42)}$$

Where β_{12} is the basal level of combinatorial promoter.

- The mass balance between the complex and the promoter; the sum of free TF concentration and complex concentration is equal to the total concentration of TF ($A + Complex = A_T$):

$$A + P_{T1}\cdot\frac{\frac{A}{K_{d1}}+\beta_1}{1+\beta_1+\frac{A}{K_{d1}}} + P_{T12}\cdot\frac{\frac{A}{K_{dh1}}+\theta\cdot\frac{A}{K_{dh1}}\cdot\left(\frac{R}{K_{dh2}}\right)^{n_2}}{1+\frac{A}{K_{dh1}}+\left(\frac{R}{K_{dh2}}\right)^{n_2}+\theta\cdot\frac{A}{K_{dh1}}\cdot\left(\frac{R}{K_{dh2}}\right)^{n_2}} = A_T \quad \text{(S2. 43)}$$

P_{T1} and P_{T12} are the total concentration of P_1 and P_{12} promoters. For simplicity, we assumed that $\theta \approx 1$, and $P_{T12} >> P_{T1}$. Therefore, we can write:

$$A + P_{T12}\cdot\frac{\frac{A}{K_{d1h}}}{1+\frac{A}{K_{d1h}}} = A_T \quad \text{(S2. 44)}$$

$$A^2 + A\cdot(K_{d1h} + P_{T12} - A) = Y_A \cdot K_{d1h} \quad \text{(S2. 45)}$$

Simulation results of $P_{lux/tetO}$-based power-law and multiplication function (Fig. 2C): Parameters were used: $K_{m1} = 1, K_{m2} = 1, k_{d1} = 200, K_{d2} = 5, K_{d1h} = 10, k_{d2h} = 4, h_1 = 1, h_2 = 1.4, A_{max} = 200, R_{max} = 500,\ n_2 = 2, n_{1h} = 1, n_{2h} = 1, \theta = 1, P_{T12} = 30,\ \beta_{12} = 0.001,\ \beta_1 = 0.002,\ \beta_2 = 0.001.$

Now we will show that the experimental results of this circuit fit power law and multiplication function:

Fitting experimental results of APF, ANF loops and $P_{lux/tetO}$-based combinatorial promoter to a power-law and multiplication function.

$\log(mCherry) = c + n_5 \cdot \log(AHL) + n_6 \cdot \log(aTc)$

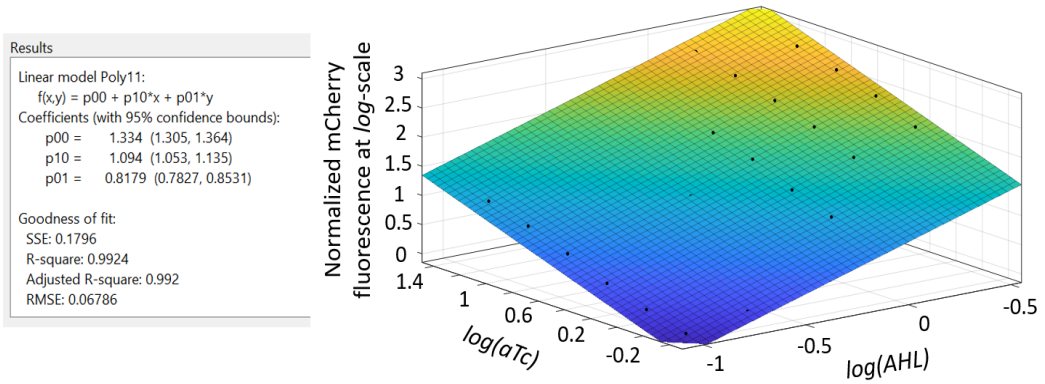


Fig. S2.15. Matlab surface fits the experimental results of APF (P_{luxTGT}) and ANF (P_{tetO}) loops and combinatorial promoter ($P_{lux/tetO}$- mCherry) to power-law and multiplication function. The data appears in Fig. 2B in the main text and is reproduced here for clarity.

Fitting simulation results of APF, ANF loops and $P_{lux/tetO}$-based combinatorial promoter to a power-law and multiplication function.

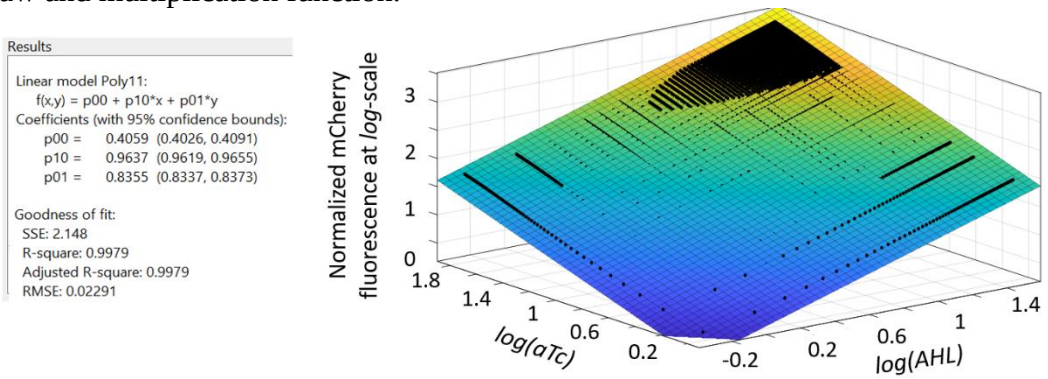


Fig. S2.16. Matlab surface fits the simulation results of APF (P_{luxTGT}) and ANF (P_{tetO}) loops and combinatorial promoter ($P_{lux/tetO}$- mCherry) to power-law and multiplication function. The data appears in Fig. 2C in the main text and is reproduced here for clarity.

Simulation results of $P_{lux/tetO}$-based perceptgene circuit (Fig. 2D) are shown in Fig. S2.17. We used the same equations and parameters that describe the P_{BAD} system (Eq. S2.21-S2.24), and assumed that $AraC_T$ is proportional to the output of $P_{lux/tetO}$-based power-law and multiplication function.

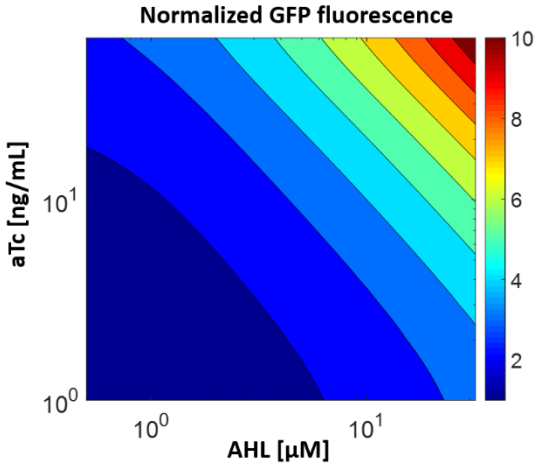


Fig. S2.17. The computed transfer function of synthetic perceptgene based on AraC system. Parameters that were used in simulation: $AraC_T = 8, K_{d3} = 3, K_{d4} = 30, m_3 = 2.8$ (*4*, *16*). The ratio between K_{d3} and K_{d4} fits well to the values that were reported on literatures(*4*, *16*).

Further analysis of 3D surface plot which compares the power-law and multiplication circuit based on ANF and APF loop versus perceptgene is provided:

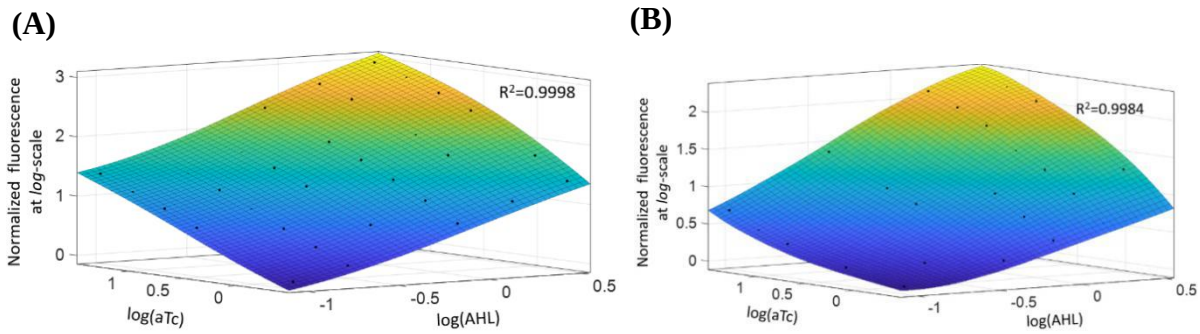


Fig. S.2.18. (A) Fitting experimental results of APF, ANF loops and $P_{lux/tetO}$ -based combinatorial promoter to non-linear models (Quadratic) using locally weighted scatterplot smoothing (Circuit from Fig. 2A, Data based on Fig. 2B). (B) Fitting experimental results of perceptgene based APF, ANF loops and $P_{lux/tetO}$-based combinatorial promoter to non-linear models (Quadratic) using locally weighted scatterplot smoothing (Circuit from Fig. 2D, Data based on Fig. 2E).

Table S2.1 List of parameters used in this section

Symbol	Description
R_i	The level of repressors which are bound to P_i
A	The level of activator which are bound to P_1
X_i	The level of repressors/activator which are bound to P_i
$P_{\ i}$	Promoter within ANF
$P_{\ 1/2}$	Vombinatorial promoter
β_i	Basal level of promoter
n_i	Hill coefficient of binding of repressor/activator to $P_{\ 1}$ & $P_{\ 2}$ promoter
K_{di}	Dissociation constant of binding repressor/activator to $P_{\ 1}$ & $P_{\ 2}$ promoter
R_{Ti}	Total level of expressed R_i
R_{max_i}	The maximum protein level achieved by P_i
A_T	Total level of expressed A
A_{max}	The maximum protein level achieved by P_1
x_i	Inducer
$K_{\ mi}$	Dissociation constant of binding x_i to Y_i
$h_{\ i}$	Hill coefficients of binding x_i to Y_i
Y	The expression level of the output protein
Y_{max}	The maximum protein level achieved by $P_{1/2}$ promoter
θ	Combinatorial promoter binding interfere
K_{dh}	Dissociation constant of binding repressors to $P_{\ 1/2}$combinatorial promoter
n_h	Hill coefficient of binding repressors to $P_{\ 1/2}$ combinatorial promoter
$AraC_C$	Concentration of the Arabinose-$AraC$ complex
$AraC_T$	The total concentration of $AraC$
m_{eff}	Effective Hill coefficient

β_{12}	Basal level of $P_{1/2}$ combinatorial promoter
P_{Ti}	Total concentration of P_1
P_{T12}	Total concentration of P_{12}
K_{eff}	Effective dissociation constant

Table S2.2 List of abbreviations used in this section

Symbol	Description
TFs	transcription factors
ANF	auto-negative feedback
P_{BAD}	AraC promoter is activated by the $AraC$ when it is induced by arabinose (Arab)
OL	open-loop
APF	auto-positive feedback
AHL	Free N-(β-Ketocaproyl)-L-homoserine Lactone $3OC_6HSL$ concentration
Arab	Free arabinose concentration
IPTG	Free Isopropyl 1-β-D-1-thio galactopyranoside concentration
aTc	Free anhydrotetracycline
P_{lux}	LuxR promoter is activated by the $LuxR$ when it is induced by AHL
P_{lacO}	LacI promoter is activated by the $LacI - IPTG$
P_{TetO}	TetR promoter is activated by the $TetR - aTc$

3. Smooth logical functions

Minimum and maximum functions perform logical operations ("if" loop) to a set of analog/digital numbers. While smooth minimum and maximum functions perform the analog operation to a set of analog numbers (x_i). To implement smooth logical functions between two analog numbers, we used rectifier activation functions (Fig. S3.1), which is widely used in artificial neural networks, and is given by (*18*, *19*):

$$\min(u_{01}, u) + f_{max} = \begin{cases} u + f_{max} & u < u_{01} \\ f_{max} & u > u_{01} \end{cases} \quad \text{(S3.1.1)}$$

$$\max(u_{02}, u) + f_{min} = \begin{cases} u + f_{min} & u > u_{02} \\ f_{min} & u < u_{02} \end{cases} \quad \text{(S3.1.2)}$$

We first will prove two mathematical identities:
(1) $min(u_{01}, x + y) = \min(u_{01} - y, x) + y$
Computing the left side:

1. For $u_{01} - (x + y) < 0$, we obtain $min(u_{01}, x + y) = u_{01}$
2. For $u_{01} - (x + y) > 0$, we obtain $min(u_{01}, x + y) = x + y$

Computing the right side:

1. For $u_{01} - (x + y) < 0$, we obtain $\min(u_{01} - y, x) = u_{01} - y$, and, thus $\min(u_{01} - y, x) + y = u_{01}$
2. For $u_{01} - (x + y) > 0$, we obtain $\min(u_{01} - y, x) = x$, and, thus $\min(u_{01} - y, x) + y = x + y$

Therefore, we obtain that the left side and the right side are equal in all the conditions.

(2) $max(u_{02}, x + y) = \max(u_{02} - y, x) + y$
Computing the left side:

3. For $u_{02} - (x + y) < 0$, we obtain $max(u_{02}, x + y) = x + y$
4. For $u_{02} - (x + y) > 0$, we obtain $min(u_{02}, x + y) = u_{02}$

Computing the right side:

3. For $u_{02} - (x + y) < 0$, we obtain $\min(u_{02} - y, x) = x$, and, thus $\min(u_{02} - y, x) + y = x + y$
4. For $u_{02} - (x + y) > 0$, we obtain $\min(u_{02} - y, x) = u_{02} - y$, and, thus $\min(u_{02} - y, x) + y = u_{02}$

Therefore, we obtain that the left side and the right side are equal in all the conditions.

An analytical expression that approximately describes Eqs. S3.1 can be given by (*20*):

$$S_\alpha(u_{0i}, u) = (u - u_0) \cdot \frac{e^{\alpha \cdot (u - u_0)}}{1 + e^{\alpha \cdot (u - u_0)}} + f \quad \text{(S3.2.1)}$$

When $\alpha < 0$, Eq. S3.3.1 finds the smooth minimum between u_0 *and* u

$$S_{-|\alpha|}(u_{01},u)=\frac{(u-u_{01})}{1+e^{+|\alpha|\cdot(u-u_{01})}}+f_{max} \quad , \quad \text{when} \quad \alpha\to-\infty \quad \rightarrow \quad S_{-|\alpha|}(u_{01},u)=\min(u_{01},u) \qquad \text{(S3.2.2)}$$

When $\alpha > 0$, Eq. S3.3 finds the smooth maximum between u_0 *and* u

$$S_{|\alpha|}(u_{02},u)=\frac{(u-u_{02})}{1+e^{-|\alpha|\cdot(u-u_{02})}}+f_{min}, \text{ when } \alpha\to\infty \rightarrow S_{|\alpha|}(u_{02},u)=\max(u_{02},u) \qquad \text{(S3.2.3)}$$

The P_{BAD} promoter can exhibit $S_{-10}=min\{u_{01},u\}$ for high Arabinose level, and $S_4=max\{u_{02},u\}$ for low Arabinose level (Fig. S.3.1C). This data is based on Fig. S.2.9.

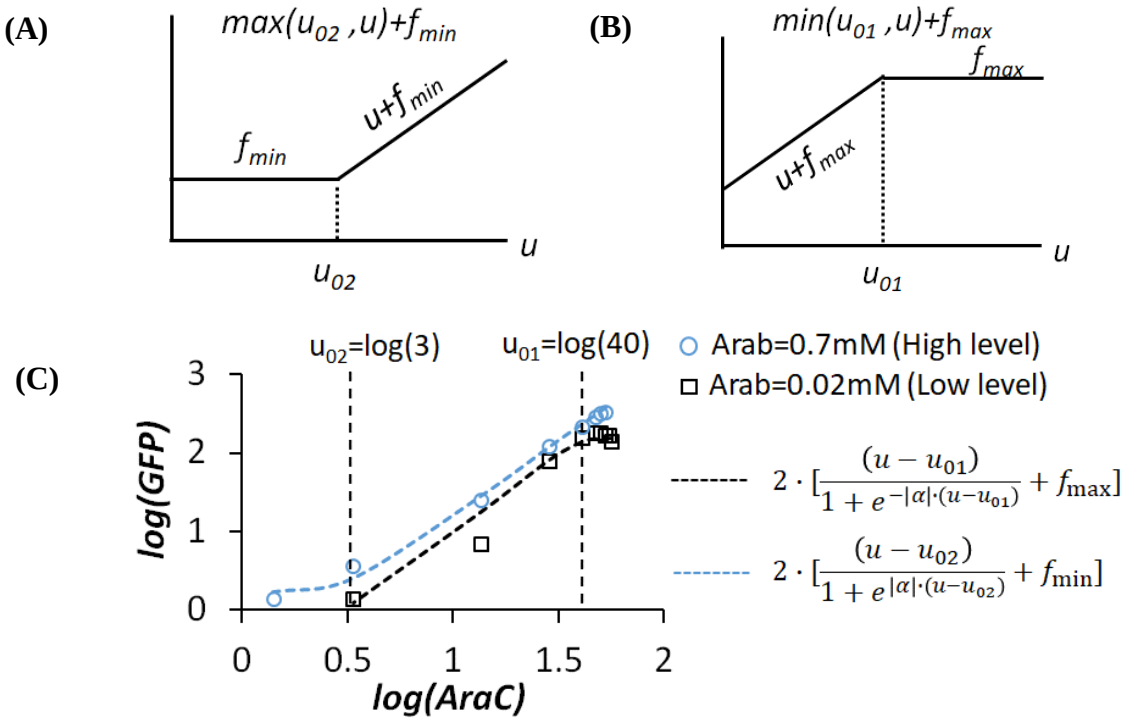


Fig. S3.1. (A) negative-rectifier activation function. (B) positive-rectifier activation function. (C) Fitting the induced P_{BAD} with low Arabinose (0.02mM) to negative rectifier Eq. S3.2.2 ($\alpha=-10, u_{01}=\log(40)$, f_{max}=0.56, and with high Arabinose (0.2mM) to positive rectifier Eq. S3.2.3 ($\alpha=4, u_{02}=\log(3)$, f_{min}=0.1. This data is based on Fig. S.2.9.

Analyzing log-transformed negative rectifier:
For simplicity, we assumed that $u_0=0, x=r, and\ y=-v$. First, we plotted the function $\min(0, u=r-v)+cont$ (Fig. S3.1.1A). Then, we subtracted the function output by the input v which brings all the plots to the same initial point: $\min(0,r-v)+v+const$ (Fig. S3.1.1B). In the next step, we graphed the function $\min(r,v)$ (Fig. S3.1.1C), and by comparing Fig. S3.1.1B to Fig. S3.1.1C, we concluded that these two graphs are equivalent. Therefore, eventually, the function $\min(0,r-v)+cont$, which can be implemented by a negative rectifier with a threshold that is controlled by the second analog input can be used to compute the minimum between two analog numbers. For further study of the relation between perceptron and minim function, we

plotted a 2-input perceptron using a sigmoid function $\frac{e^{w\cdot(r+v-c)}}{1+e^{w\cdot(r+v-c)}}$ (Fig. S3.1.1.D), and a 2-input perceptron that is normalized by second input ($\frac{e^{w\cdot(r+v-c)}}{1+e^{w\cdot(r+v-c)}} - v$) that brings all the curves to the same initial point. In our simulation, we designed the sigmoid to act as a negative rectifier in the operating dynamic range $w = 7$ and large $c = 7$. We obtained that $\min(0, r - v) + cont$ and perceptron have similar behaviors, however, in an opposite dependency concerning the second input.

Then, we plotted the perceptgene of two inputs $\frac{bais\cdot r^{n_3}\cdot v^{n_4}+\beta}{1+bais\cdot r^{n_1}\cdot v^{n_2}}$ (Fig. S3.1.1D, only in this case we assumed that $n_3 = n_4 = 1.5, bais = 0.001, \beta = 0.01$).

For operating in partial swing, the perceptgene can be approximated as shifted and biased log-transformed negative rectifier (NR). The NF receives the collective analog signal $k_1 \cdot \log(r) + k_2 \cdot \log(v) + \log(AraC_{max})$, where $AraC_{max}$ is the maximum AraC level. Using Eq. S3.2.2, when the bias depends linearly on the *v* level,
the output of the NR:

$$f_{NR} = \frac{k_1\cdot\log(r)+k_2\cdot\log(v)+\log(AraC_{max})-\log(U_{o1})}{1+\left(\frac{b}{U_{o1}}\cdot r^{k_1}\cdot v^{k_2}\right)^{|\alpha|}} + 2\cdot w_1\cdot\log(v) + const \qquad \text{(S3.3.3)}$$

Where $u_{01} = \log(U_{o1})$,

(1) when: $\alpha \ll -1$, and $\frac{AraC_{max}}{U_{o1}}\cdot r^{k_1}\cdot v^{k_2} \ll 1$ ➔ $f_{NR} = k_1\cdot\log(r) + k_2\cdot\log(v) + \log(AraC_{max}) - \log(U_{o1}) + 2\cdot w_1\cdot\log(v) + const$. In this case, we can claim that the f_{NR} is equal to the analog argument $bias\cdot r^{n_3}\cdot v^{n_4}$ at the log-scale, where n_3 and n_4 are the weights of the two inputs as calculated by the power-law and multiplication function.

(2) when: $\alpha \ll -1$, and $\frac{AraC_{max}}{U_{o1}}\cdot r^{k_1}\cdot v^{k_2} \gg 1$ ➔ $f_{NR} = 2\cdot w_1\cdot\log(v) + const$

In the case of the P_{BAD} as shown in Fig. S3.1C, since the promoter activity is approximated as twice the negative rectifier, we expect that

⇨ $2\cdot k_1 = n_3$,
⇨ $2\cdot(k_2 + w_1) = n_4$
⇨ $\log(bias) = \log(AraC_{max}) - \log(U_{o1})$

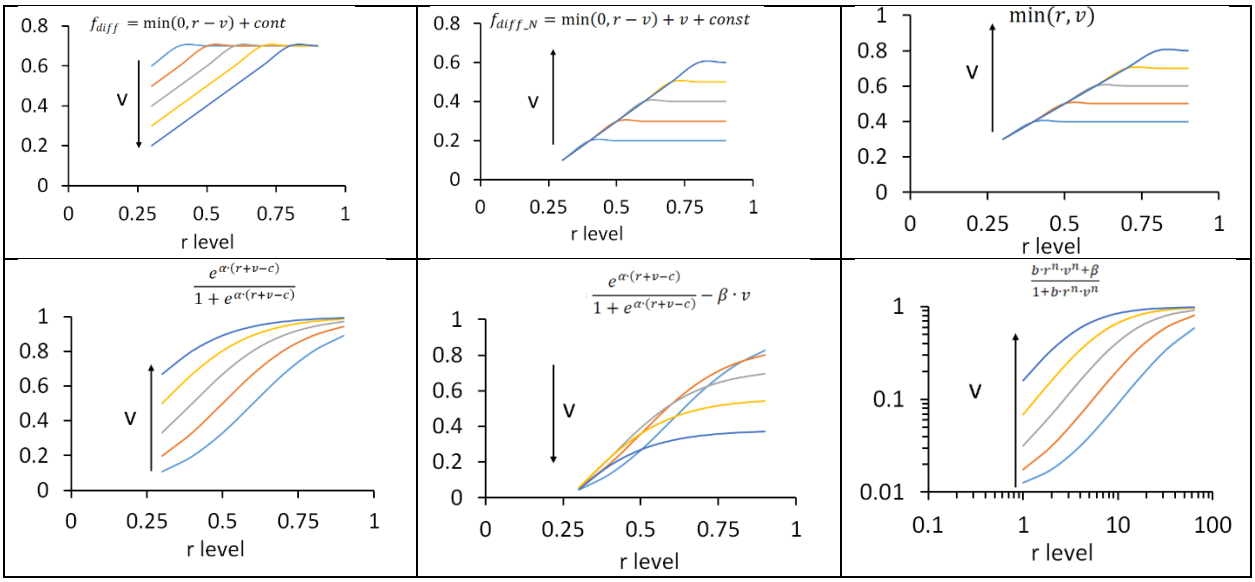


Fig. S3.1.1. Calculations of several functions related to smooth minimum calculations

First, we normalized the measured data of three circuits from Fig. 1H and Fig. 2E, and Fig. 2H, by the minimum level achieved for each circuit. Second, we transformed Eq. S3.3 operates in the linear-scale, to the logarithmic-scale, by a logarithmic operation to the normalized data. Then, we normalized the inputs for every circuit by its input dynamic range value (x_i/IDR_i).

1. Smooth minimum function – $P_{lacO/tetO}$-based Perceptgene circuit. Fig. S3.2A shows the experimental data of Fig. 1H in the linear-scale, which is well matched to Eq. S3.3:
   Where:
   $u = 0.4 \cdot x_1 + 0.2 \cdot x_2 - 1$
   $b = 0.3 \cdot x_2 + 0.6$
   $\alpha = -10 < 0$ ➔ Soft minimum
   $x_1 = \log(IPTG/IDR_3),\ x_2 = \log(aTc/IDR_4)$

   We can also build the model that that b depends on x_1 $(IPTG)$

$$f_1 = S_{-10}(0,0.4 \cdot x_1 + 0.2 \cdot x_2 - 1) + 0.3 \cdot x_2 + 0.6 \quad \text{(S3.4.1)}$$

Where f_1 is the normalized data of Fig. 1H. We can write:

$$f_1 = min(0,0.4 \cdot x_1 + 0.2 \cdot x_2 - 1) + 0.3 \cdot x_2 + 0.6 \quad \text{(S3.4.2)}$$

$$= \begin{cases} 0.4 \cdot x_1 + 0.5 \cdot x_2 - 0.4 & 0.4 \cdot x_1 + 0.2 \cdot x_2 - 1 < 0 \\ 0.3 \cdot x_2 + 0.6 & 0.4 \cdot\ x_1 + 0.2 \cdot x_2 - 1 > 0 \end{cases}$$

A general formula to Eq. S3.4 is:

$$f_1 = min(0, k_1 \cdot x_1 + k_2 \cdot x_2 - \gamma) + 2 \cdot w_1 \cdot x_2 + a_2 \quad \text{(S3.5)}$$

$$= \begin{cases} k_1 \cdot x_1 + k_2 \cdot x_2 - \gamma + 2 \cdot w_1 \cdot x_2 + a_2 & k_1 \cdot x_1 + k_2 \cdot x_2 - \gamma < 0 \\ 2 \cdot w_1 \cdot x_2 + a_2 & k_1 \cdot x_1 + k_2 \cdot x_2 - \gamma > 0 \end{cases}$$

We can write Eq. S3.5 as:

$$f_2 = f_1 - 2 \cdot w_1 \cdot x_2 + a_2) = \begin{cases} k_1 \cdot x_1 + k_2 \cdot x_2 - \gamma & k_1 \cdot x_1 + k_2 \cdot x_2 - \gamma < 0 \\ 0 & k_1 \cdot x_1 + k_2 \cdot x_2 - \gamma > 0 \end{cases}$$

$$f_3 = f_1 - (k_2 \cdot x_2 - \gamma) - (2 \cdot w_1 \cdot x_2 + a_2) = \begin{cases} k_1 \cdot x_1 & k_1 \cdot x_1 < \gamma - k_2 \cdot x_2 \\ \gamma - k_2 \cdot x_2 & k_1 \cdot x_1 > \gamma - k_2 \cdot x_2 \end{cases} \quad \text{(S3.6)}$$

Eq. S3.6 can be viewed as a smooth minimum logical function between two analog numbers that are proportional to system inputs x_1 and x_2 (Fig. S3.2B):

$$f_3 = \min(\gamma - k_2 \cdot x_2, k_1 \cdot x_1) = \begin{cases} k_1 \cdot x_1 & k_1 \cdot x_1 < \gamma - k_2 \cdot x_2 \\ \gamma - k_2 \cdot x_2 & k_1 \cdot x_1 > \gamma - k_2 \cdot x_2 \end{cases} \quad \text{(S3.7)}$$

Fig. S3.2B and Table in Fig, S3.2C give similar results with a little bit of difference, and this is because that each one is calculated in a different way.

The IPTG weight $n_3 = 0.75$ (Table S.4.2), and $k_1 = 0.4$ ➔ $n_3 = 1.875 \cdot k_1$ as we expected
The aTc weight $n_4 = 1$ (Table S.4.2), and $k_2 = 0.2$, $w_1 = 0.3$ ➔ $n_4 = 2 \cdot (k_2 + w_1)$ as we expected.

For a more general case: an ideal minimum function is observed when $min(r, v) = min(0, r - v) + v$. Thus, if we assume that $r = k_1 \cdot x_1$ and $v = \gamma - k_2 \cdot x_2$, we can find that the perceptgene computes:

$$min(0, r - v) + v + 0.5 \cdot x_2 - 0.4 ==> min(r, v) + 0.5 \cdot x_2 - 0.4 \quad \text{(S3.7.1)}$$

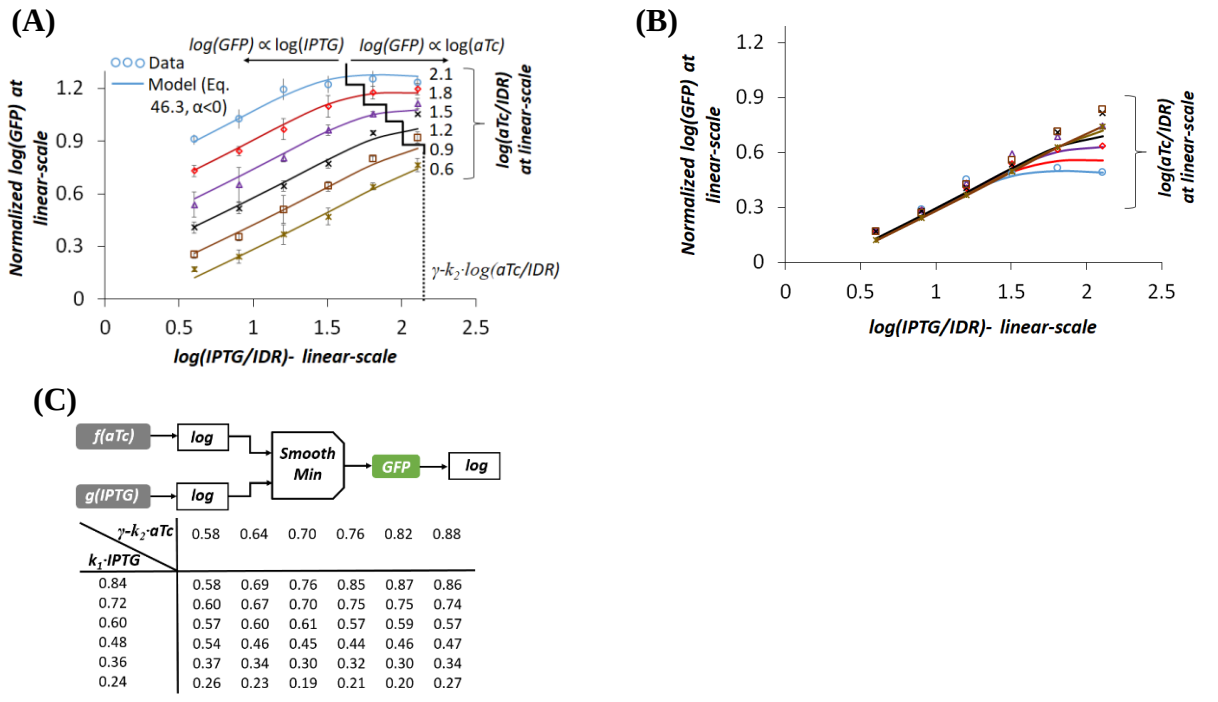


Fig. S3.2. The perceptgene circuit calculates the smooth minimum between the analog inputs ($\log(IPTG)$ *and* $\log(aTc)$). (A) the raw data, Where $\gamma = 0.5 \cdot \log(B_2)$, $k_2 = 0.5 \cdot (n_4 - w_1)$. (B) the raw data after bringing all the curves to the same initial minimum point, Similar to Fig. S3.1.1. (C) Raw data using Eq. S3.7.

We then calculated the errors between the transformed data of the perceptgene circuit and the ideal data of smooth functions.

Table S3.1. Transformed experimental data (D_{exp}) of Fig. 1H (Fig. S3.2)

Experimental Results	**0.58**	**0.64**	**0.70**	**0.76**	**0.82**	**0.88**
0.84	0.58	0.69	0.76	0.85	0.87	0.86
0.72	0.60	0.67	0.70	0.75	0.75	0.74
0.60	0.57	0.60	0.61	0.57	0.59	0.57
0.48	0.54	0.46	0.45	0.44	0.46	0.47
0.36	0.37	0.34	0.30	0.32	0.30	0.34
0.24	0.26	0.23	0.19	0.21	0.20	0.27

Table S3.2. Ideal (Expected) data for minimum function (D_{min-1}) based on Table S3.1

Expected MIN	**0.58**	**0.64**	**0.70**	**0.76**	**0.82**	**0.88**
0.84	0.58	0.64	0.70	0.76	0.82	0.84
0.72	0.58	0.64	0.70	0.72	0.72	0.72
0.60	0.58	0.60	0.60	0.60	0.60	0.60
0.48	0.48	0.48	0.48	0.48	0.48	0.48
0.36	0.36	0.36	0.36	0.36	0.36	0.36
0.24	0.24	0.24	0.24	0.24	0.24	0.24

Table S3.3. Calculation of the error between measurement data (Table S3.1) and Ideal data for the minimum function (Table S3.2) using:

$$Error_i = abs\left(\frac{(D_{exp} - D_{min-1})}{D_{min-1}}\right)$$

Error(i)	**0.58**	**0.64**	**0.70**	**0.76**	**0.82**	**0.88**
0.84	0.001622052	0.086931	0.089926	0.122127	0.06111	0
0.72	0.040504146	0.054275	0.007044	0.036687	0.037627	0.029814
0.60	0.017410123	0.007701	0.015816	0.050489	0.011948	0.049949
0.48	0.127994603	0.031549	0.059475	0.077629	0.042186	0.025321
0.36	0.441393616	0.312972	0.158067	0.219982	0.163661	0.317077
0.24	0.013971729	0.124459	0.282187	0.205529	0.217792	0.038107

The error for the experiment is calculated as the:

$$Error = 100 \times \frac{1}{N}\sum_{i=1}^{N} Error_i$$

➔ 10%

Table S3.4. Ideal data for maximum function (D_{max-1}) based on Table S3.1

Expected MAX	**0.58**	**0.64**	**0.70**	**0.76**	**0.82**	**0.88**
0.84	0.84	0.84	0.84	0.84	0.84	0.88
0.72	0.72	0.72	0.72	0.76	0.82	0.88
0.60	0.60	0.64	0.70	0.76	0.82	0.88
0.48	0.58	0.64	0.70	0.76	0.82	0.88
0.36	0.58	0.64	0.70	0.76	0.82	0.88
0.24	0.58	0.64	0.70	0.76	0.82	0.88

Table S3.5. Calculation of the error between measurement data (Table S3.1) and Ideal data for the maximum function (Table S3.4) using:

$$Error_i = abs\left(\frac{(D_{exp} - D_{max-1})}{D_{min-1}}\right)$$

Error(i)	**0.58**	**0.64**	**0.70**	**0.76**	**0.82**	**0.88**
0.84	0.310123265	0.173461	0.093065	0.014157	0.035065	0.019391
0.72	0.163899995	0.064676	0.02237	0.017875	0.088913	0.157425
0.60	0.050163119	0.067919	0.128017	0.250386	0.277035	0.352238
0.48	0.066487225	0.272256	0.354118	0.41745	0.439328	0.468357
0.36	0.353858034	0.465573	0.569227	0.582638	0.631034	0.610863
0.24	0.557987327	0.643623	0.732991	0.728207	0.751983	0.693287

The error for the experiment is calculated using Eq. S3.25 ➔ 32%

Table S3.6. Ideal data for the average function (D_{ave-1}) based on Table S3.1

Expected Average	**0.58**	**0.64**	**0.70**	**0.76**	**0.82**	**0.88**
0.84	0.71	0.74	0.77	0.80	0.83	0.86
0.72	0.65	0.68	0.71	0.74	0.77	0.80
0.60	0.59	0.62	0.65	0.68	0.71	0.74
0.48	0.53	0.56	0.59	0.62	0.65	0.68
0.36	0.47	0.50	0.53	0.56	0.59	0.62
0.24	0.41	0.44	0.47	0.50	0.53	0.56

Table S3.7. Calculation the error between measurement data (Table S3.1) and Ideal data for average function (Table S3.6) using:

$$Error_i = abs\left(\frac{(D_{exp} - D_{ave-1})}{D_{min-1}}\right)$$

Error(i)	**0.58**	**0.64**	**0.70**	**0.76**	**0.82**	**0.88**
0.84	0.1846358	0.06281	0.011806	0.063495	0.046108	0.001974
0.72	0.074591661	0.010556	0.009607	0.007547	0.030918	0.074359
0.60	0.034569811	0.040348	0.063066	0.162957	0.165873	0.230545
0.48	0.021381523	0.170205	0.235228	0.28638	0.293251	0.312616
0.36	0.202458539	0.317255	0.432002	0.433788	0.487469	0.448044
0.24	0.374242173	0.482392	0.602825	0.586875	0.61635	0.518199

The error for the experiment is calculated using Eq. S3.25 ➔ 22%

Table S3.8. summarized the data

Std err (experimental vs. expected)	Smooth Min (expected)	Smooth Avg (expected)	Smooth Max (expected)
Smooth Min (experimental)	10%	22%	32%

Analyzing log-transformed positive rectifier:
For simplicity, we assumed that $u_0 = 0, x = r, and\ y = -v$. First, we plotted the function $\max(0, u = r - v) + const$ (Fig. S3.3.1A). Then, we subtracted the function output by the input v which brings all the plots to the same initial point: $\max(0, r - v) + v + const$ (Fig. S3.3.1B). In the next step, we graphed the function $\max(r, v)$ (Fig. S3.3.1C), and by comparing Fig. S3.3.1B to Fig. S3.3.1C, we concluded that these two graphs are equivalent. Therefore, eventually, the function $\max(0, r - v) + const$, which can be implemented by a positive rectifier with a threshold that is controlled by the second analog input, can be used to compute the maximum between two analog numbers. For further study of the relation between perceptron and maximum function, we plotted a 2-input perceptron using a sigmoid function $\frac{e^{\alpha\cdot(r+v-c)}}{1+e^{\alpha\cdot(r+v-c)}}$ (Fig. S3.3.1D), and a 2-input perceptron that is normalized by second input $(\frac{e^{\alpha\cdot(r+v-c)}}{1+e^{\alpha\cdot(r+v-c)}} - v)$ that brings all the curves to the same initial point. In our simulation, we designed the sigmoid to act as a positive rectifier in the operating dynamic range $\alpha = 7$ and large $c = 1.5$. We obtained that $\max(0, r - v) + const$ and perceptron have similar behavior, however, in an opposite relation with the second input.
Then, we plotted the perceptgene of two inputs $\frac{bias\cdot r^{n_1}\cdot v^{n_2} + \beta}{1 + bias\cdot r^{n_1}\cdot v^{n_2}}$ (Fig. S3.1.1D, only in this case we assumed that $n_1 = n_2 = 1.5, bias = 0.00005, \beta = 0.01$).

For operating in partial swing, the perceptgene can be approximated as shifted and biased log-transformed positive rectifier (PR). The PF receives the collective analog signal $k_1 \cdot \log(r) + k_2 \cdot \log(v) + \log(AraC_{max})$, where $AraC_{max}$ is the maximum AraC level. Using Eq. S3.2.3, when the bias depends linearly on the *v* level, the output of the PR:

$$f_{NR} = \frac{k_1 \cdot \log(r) + k_2 \cdot \log(v) + \log(AraC_{max}) - \log(U_{o2})}{1 + \left(\frac{AraC_{max}}{U_{o2}} \cdot r^{k_1} \cdot v^{k_2}\right)^{-|\alpha|}} + 2 \cdot w_2 \cdot \log(r) + const \qquad \text{(S3.8)}$$

Where $u_{02} = \log(U_{o2})$,

(3) when: $\alpha \gg 1$, and $\frac{AraC_{max}}{U_{o2}} \cdot r^{k_1} \cdot v^{k_2} \gg 1$ $\square$ $f_{PR} = k_1 \cdot \log(r) + k_2 \cdot \log(v) + \log(AraC_{max}) - \log(U_{o2}) + 2 \cdot w_2 \cdot \log(r) + const$. In this case, we can claim that the f_{PR} is equal to the analog argument $bias \cdot r^{n_5} \cdot v^{n_6}$ at the log-scale., where n_5 and n_6 are the are the weights of the two inputs as calculated by the power-law and multiplication function.

(4)

⇨ $k_1 = n_1$,

⇨ $k_2 + k_3 = n_2$

(5) $B = b$when: $\alpha \gg 1$, and $\frac{AraC_{max}}{U_{o2}} \cdot r^{k_1} \cdot v^{k_2} \ll 1$ ➔ $f_{PR} = 2 \cdot w_2 \cdot \log(r) + const$

In case of the P_{BAD} as shown in Fig. S3.1C, since the promoter activity is approximated as twice the positive rectifier, we expect that

⇨ $2 \cdot k_2 = n_6$,

⇨ $2 \cdot (k_1 + w_2) = n_5$

⇨ $\log(bias) = \log(AraC_{max}) - \log(U_{o2})$

Smooth maximum function – $P_{lux/tetO}$-based Perceptgene circuit. Fig. S3.3A. shows the experimental data of Fig. 2E in the linear-scale, is well matched to Eq. S3.3, where

$u_1 = 0.25 \cdot x_1 + 0.38 \cdot x_2 - 0.2$

$b = 0.25 \cdot x_1 + 0.1$

$\alpha = 4 > 0$ ➔ soft maximum

$x_1 = \log(AHL/IDR_5), x_2 = \log(aTc/IDR_6)$

We can also build the model that b depends on x_2 (aTc)

$$f_1 = S_4(0.25 \cdot x_1 + 0.38 \cdot x_2 - 0.2) + 0.22 \cdot x_1 + 0.1 \qquad \text{(S3.8.1)}$$

Where f_1 is the normalized data of Fig. 2E. We can write:

$$f_1 = max(0, 0.25 \cdot x_1 + 0.38 \cdot x_2 - 0.2) + 0.22 \cdot x_1 + 0.1 \qquad \text{(S3.8.2)}$$

$$= \begin{cases} 0.25 \cdot x_1 + 0.38 \cdot x_2 - 0.2 & 0.25 \cdot x_1 + 0.38 \cdot x_2 - 0.2 > 0 \\ 0.22 \cdot x_1 + 0.1 & 0.25 \cdot x_1 + 0.38 \cdot x_2 - 0.2 < 0 \end{cases}$$

Similar to Eq. S3.7 we can write Eq. S3.8 as:

$$f_1 = \max(k_2 \cdot x_2, \gamma - k_1 \cdot x_1) = f_1 - (k_1 \cdot x_1 - \gamma) - (2 \cdot w_2 \cdot x_1 + a_2)$$
$$= \begin{cases} k_1 \cdot x_1 & k_1 \cdot x_1 > \gamma - k_2 \cdot x_2 \\ \gamma - k_2 \cdot x_2 & k_1 \cdot x_1 < \gamma - k_2 \cdot x_2 \end{cases} \quad \text{(S3.9)}$$

Eq. S3.9 can be viewed as a smooth maximum logical function between two analog numbers that are proportional to system inputs x_1 and x_2 (Fig. S3.3B).

The aTc weight $n_6 = 0.8$, and $k_2 = 0.38$ ➔ $n_6 \approx 2 \cdot k_2$
The AHL weigh $n_5 = 1$, and $k_1 = 0.25$, $w_2 = 0.22$ ➔ $n_5 \approx 2 \cdot (k_1 + w_2)$

For a more general case: an ideal maximum function is observed when $max(r, v) = max(0, r - v) + v$. Thus, if we assume that $r = k_2 \cdot x_2$ and $v = \gamma - k_1 \cdot x_1$, we can find that the perceptgene computes:
$max(0, r - v) + v + 0.47 \cdot x_1 - 0.1$➔ $max(r, v) + 0.47 \cdot x_1 - 0.1$ (S3.9.1)

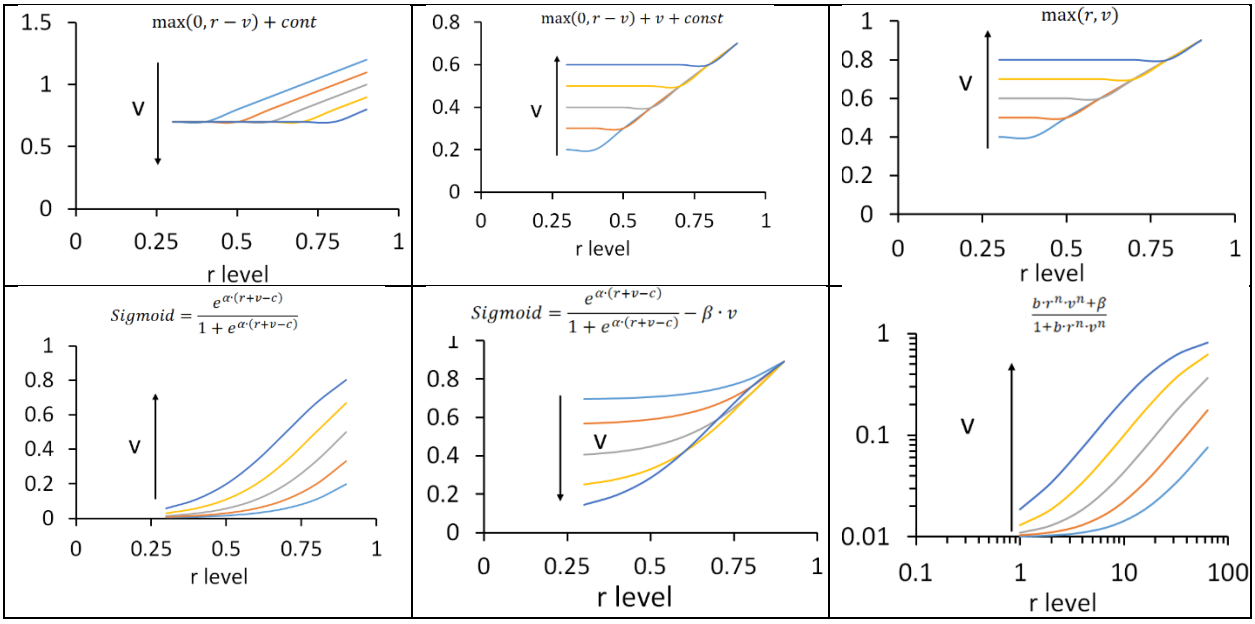


Fig. S3.3.1. Calculations of several functions related to smooth maximum calculations.

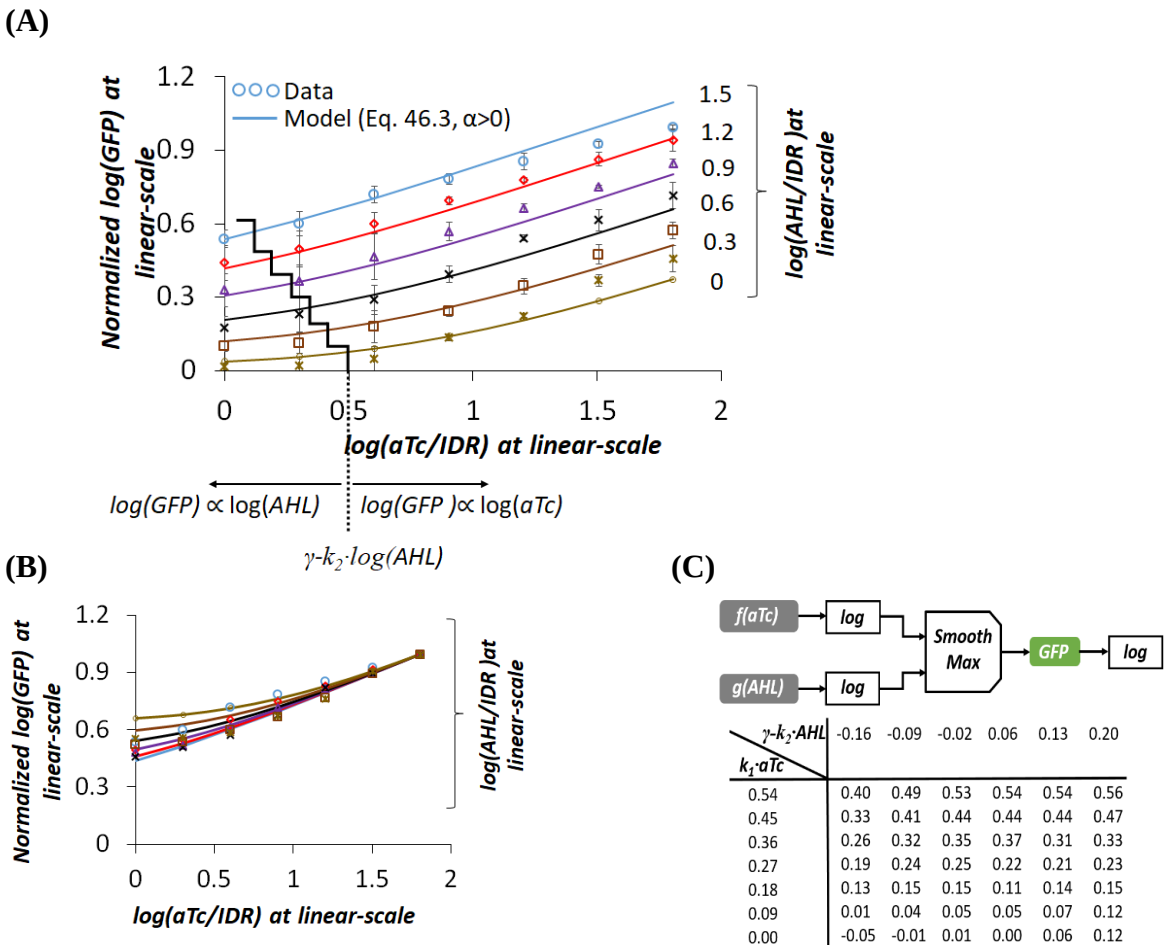


Fig. S3.3.2. The perceptgene circuit calculates the smooth maximum between the analog inputs $(\log(AHL), \log(aTc))$. (A) the raw data, where $\gamma = 0.5 \cdot \log(B_3)$, $k_2 = 0.5 \cdot (n_5 - w_2$ (B) the raw data after bringing all the curves to same maximum point, Similar to Fig. S3.3.1. (C) Raw data using Eq. S3.9.

Similar calculations of error were performed to the maximum circuit**.**

Table S3.9. summarized our results

Std err (experimental vs. expected)	Smooth Min (expected)	Smooth Avg (expected)	Smooth Max (expected)
Smooth Max (experimental)	470%	256%	23%

A simple model for the average function:

The output of the power-law and multiplication circuit (Fig. S3.4A) is given by:

$$Y = Y_N \left(\frac{AHL}{IDR_1}\right)^{n_7} \cdot \left(\frac{IPTG}{IDR_2}\right)^{n_8} \quad \text{(S3.10)}$$

Where Y_N is a normalized parameter with a unit of concentration, IDR_i are the input dynamic ranges. Applying log operation, we get:

$$\log(Y) = \log(Y_N) + n_7 \cdot log\left(\frac{AHL}{IDR_1}\right) + n_8 \cdot log\left(\frac{IPTG}{IDR_2}\right) \quad \text{(S3.11)}$$

In case that n_7=n_8=0.5, we get:

$$Out = \log(Y) - \log(Y_N) = \frac{log\left(\frac{AHL}{IDR_1}\right)+log\left(\frac{IPTG}{IDR_2}\right)}{2} \quad \text{(S3.12)}$$

Therefore, conceptually, we can implement average only with the power-law and multiplication, without the need for activation function. However, it is challenging to obtain IPTG weight around 0.5. We can solve this challenge by applying a linear activation function using AraC, P_{BAD} and Arabinose (Fig. S3.4B). Equations that describe circuit from Fig. S3.4B are:

$$AraC_T = AraC_N \left(\frac{AHL}{IDR_1}\right)^{n_7} \cdot \left(\frac{IPTG}{IDR_2}\right)^{n_8} \quad \text{(S3.13.1)}$$

$$GFP = GFP_{max} \frac{\frac{AraC_T}{K_{d3}}+\beta_4}{1+\frac{AraC_T}{K_{d3}}} \quad \text{(S3.13.2)}$$

Eq. S3.13.2 is driven from Eq. S2.21, and Eq. S2.22, where the arabinose concentration is very high (Arab>>K_{m3}), β_4 <<1, is the basal level of P_{BAD} promoter, GFP_{max} is the maximum GFP achieved by P_{BAD} promoter, $AraC_N$ is corresponding to Y_N. The Eq. S3.13.2, or the P_{BAD} promoter can operate in two linear ranges:

1. $\frac{AraC_T}{K_{d3}} \ll 1$, in this range, we can approximate Eq. S3.13.2 as:

$$GFP = GFP_{max}\left(\frac{AraC_T}{K_{d3}} + \beta_4\right) \quad \text{(S3.13.3)}$$

Substituting Eq. S3.13.1 into Eq. S3.13.3, we get:

$$\frac{GFP}{GFP_{max}} = \left(\frac{AraC_N}{K_{d3}}\right) \cdot \left(\frac{AHL}{IDR_1}\right)^{n_7} \cdot \left(\frac{IPTG}{IDR_2}\right)^{n_8} + \beta_4 \quad \text{(S3.14)}$$

Applying a log-operation into 3.14 and in case that $n = n_7 = n_8$, we get:

$$log\left(\frac{GFP}{GFP_{max}} - \beta_4\right) - log\left(\frac{AraC_N}{K_{d3}}\right) = n\left(log\left(\frac{AHL}{IDR_1}\right) + log\left(\frac{IPTG}{IDR_2}\right)\right) \quad \text{(S3.15)}$$

In this working range, the linear activation function based on P_{BAD} could not solve the challenge of achieving low weights. These are two solutions, (1) working with an activation function with very low hill coefficient, which is not simple to create. (2) working with another working range:

2. Second analog working range: $\beta_4 < \frac{AraC_T}{K_{d3}} < 1$. For simplicity, in this analysis, we neglect the basal level, and applying a log operation to Eq. S3.13.2, we get:

$$\log\left(\frac{GFP}{GFP_{max}}\right) = \log\left(\frac{AraC_T}{K_{d3}}\right) - \log\left(1 + \frac{AraC_T}{K_{d3}}\right) \quad \text{(S3.16)}$$

The slope in the log-log is equal:

$$m_{eff} \equiv \frac{d\left[\log\left(\frac{GFP}{GFP_{max}}\right)\right]}{d\left[log\left(\frac{AraC_T}{K_{d3}}\right)\right]} = 1 - \frac{\frac{AraC_T}{K_{d3}}}{1+\frac{AraC_T}{K_{d3}}} \quad \text{(S3.17)}$$

Our goal now is to approximate Eq. S3.17 when $\beta_4 < \frac{AraC_T}{K_{d3}} < 1$.

By applying Taylor series around $\frac{AraC_T}{K_{d3}} \approx 1$, we get (Fig. S3.5):

$$\frac{\frac{AraC_T}{K_{d3}}}{1+\frac{AraC_T}{K_{d3}}} \approx \frac{1}{2} + \frac{1}{4} \cdot ln(\frac{AraC_T}{K_{d3}}) \quad \text{(S3.18)}$$

Substituting Eq. S3.18 into Eq. S3.17, we get

$$m_{eff} = \frac{1}{2} - \frac{1}{4} \cdot ln(\frac{AraC_T}{K_{d3}}) \quad \text{(S3.19.1)}$$

$$GFP = GFP_{max} \cdot \left(\frac{AraC_T}{K_{d3}}\right)^{m_{eff}} \quad \text{(S3.19.2)}$$

Substituting Eq. S3.13.1 into Eq. S3.19.2, we get:

$$\frac{GFP}{GFP_{max}} = GFP_{max} \cdot \left(\frac{AraC_N}{K_{d3}}\right)^{m_{eff}} \cdot \left(\frac{AHL}{IDR_1}\right)^{n_7 \cdot m_{eff}} \cdot \left(\frac{IPTG}{IDR_2}\right)^{n_8 \cdot m_{eff}} \quad \text{(S3.20)}$$

Applying log operation, we get:

$$\log\left(\frac{GFP}{GFP_{max}}\right) - m_{eff} \cdot log\left(\frac{AraC_N}{K_{d3}}\right) = n_7 \cdot m_{eff} \cdot log\left(\frac{AHL}{IDR_1}\right) + n_8 \cdot m_{eff} \cdot log\left(\frac{IPTG}{IDR_2}\right) \quad \text{(S3.21)}$$

So that Eq. S3.21 computes the average, we require:

1. $n_7 = n_8 \equiv n$
2. $n \cdot m_{eff} = 0.5$ (Substituting from Eq. S3.19.1)

⇨ $n \cdot \left(\frac{1}{2} - \frac{1}{4} \cdot ln(\frac{AraC_T}{K_{d3}})\right) = 0.5$

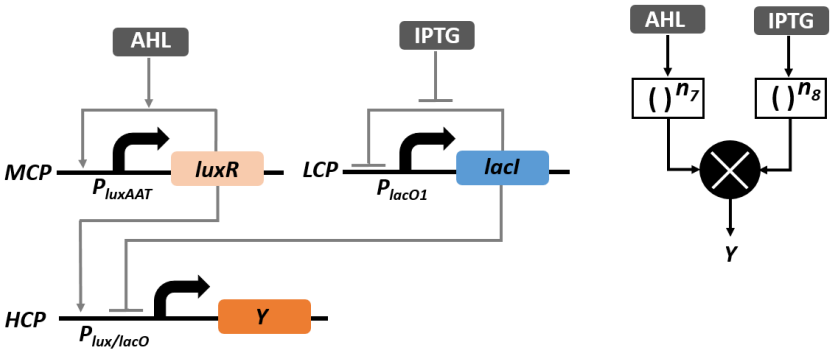
⇨ $\log\left(\frac{AraC_T}{K_{d3}}\right) = 2 \cdot \left(1 - \frac{1}{n}\right)$

⇨ $\frac{AraC_T}{K_{d3}} = e^{2 \cdot \left(1 - \frac{1}{n}\right)}$ (S3.22)

Numbers: $n = 1.65, m = 1$ (These numbers are based on our measurements Fig. S4.6)
➔ **AraC** $\approx$ **2K**, $m_{eff} = 0.33$ .Under these conditions:

$$\log\left(\frac{GFP}{GFP_{max}}\right) - Const = 0.5 \cdot log\left(\frac{AHL}{IDR_1}\right) + 0.5 \cdot log\left(\frac{IPTG}{IDR_2}\right) \quad \text{(S3.23)}$$

(A)



(B)

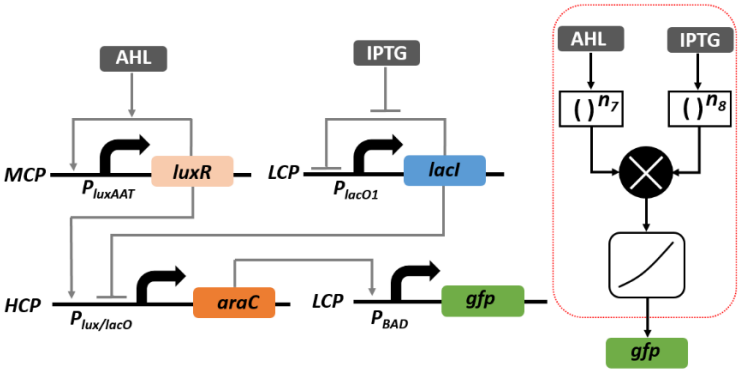


Fig. S3.4. (A) APF (P_{luxAAT}), ANF (P_{lacO1}), and $P_{lux/lacO}$-based Combinatorial promoter. (B) Preceptgene based on linear activation function to implement average.

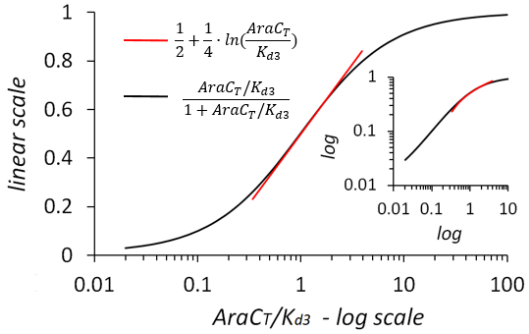


Fig. S3.5. The approximation of Eq. S3.18, the inset Fig S3.5. is a representation in the log-log scale.

Experimental results of APF (P_{luxAAT}), ANF (P_{lacO1}) and $P_{lux/lacO}$-based Combinatorial promoter:

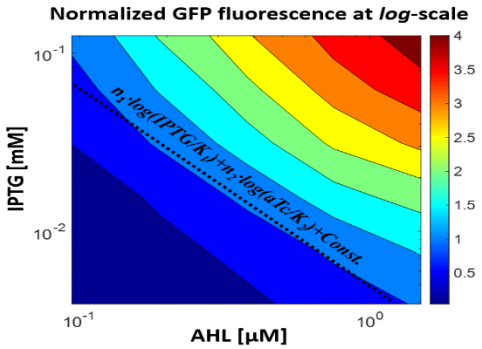


Fig. S3.6. Experimental results of circuit Fig. S3.4. The data represent means calculated from three experiments.

Fitting experimental results of P_{luxAAT}-based APF, P_{lacO1}-based ANF loops and $P_{lux/lacO}$-based combinatorial promoter to power-law and multiplication function ($\log(GFP) = c + n_7 \cdot \log(AHL) + n_8 \cdot \log(IPTG)$)

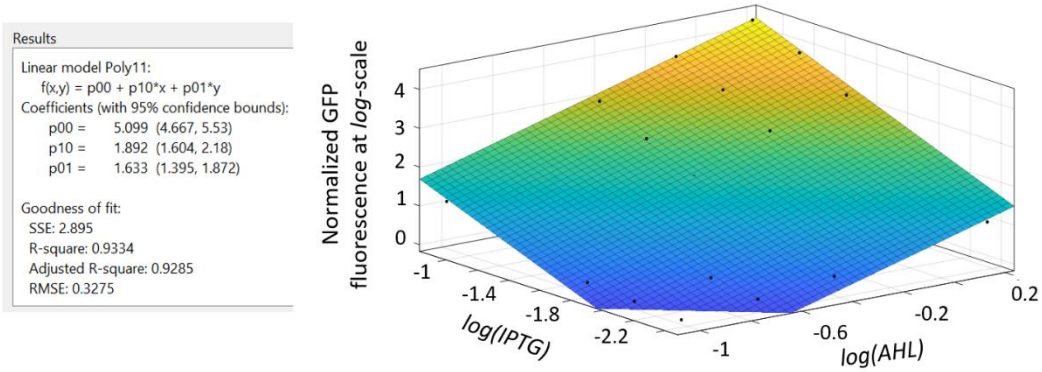


Fig. S3.7. Matlab surface fits the experimental results of APF (P_{luxAAT}) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function.

2. Average function – $P_{lux/lacO}$-based Perceptgene circuit. To implement the average function between two analog numbers, we used a linear activation function (Fig. S3.8A). A linear activation function is a special case of Eq. S3.3 with $\alpha \approx 0$. Fig. S3.8B shows the experimental data of Fig. 2H in the linear-scale, which is well matched to Eq. S3.3 with $\alpha \approx 0$ (Fig. S3.6B):
   $f_1 = 0.55 \cdot x_1 + 0.51 \cdot x_2 - 0.25$

⇨ $f_1 + 0.25 \cong \frac{x_1 + x_2}{2}$

The power-law and multiplication functions set the input dynamic range of the smooth logic functions. For example, the average function has an IDR of $\log(32)$ order of magnitude for IPTG and the multiplication function has a IDR of $\log(16)$ order of magnitude for AHL.

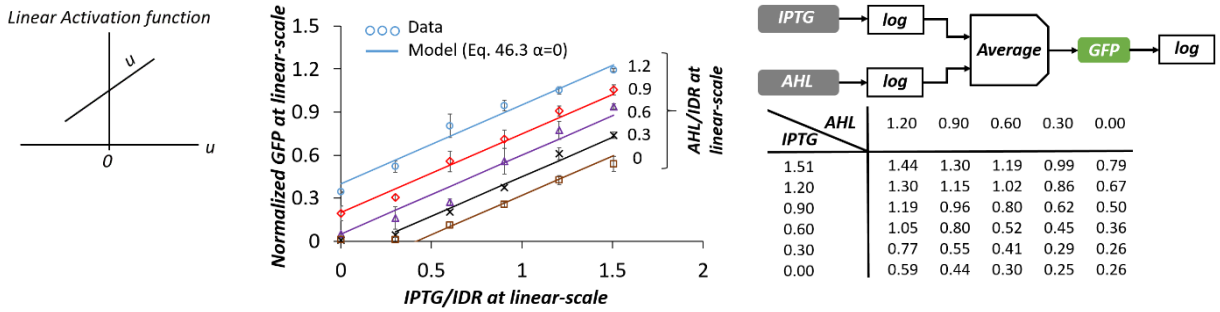


Fig. S3.8. The perceptgene circuit calculates the average between the analog inputs $(\log(AHL), \log(IPTG))$.

Similar calculations of error were performed on the average circuits.

Table S3.10 summarized our results

Std err (experimental vs. expected)	Smooth Min (expected)	Smooth Avg (expected)	Smooth Max (expected)
Smooth Avg (experimental)	66%	8.5%	24%

Table S3.11 summarized all the results

Std err (experimental vs. expected)	Smooth Min (expected)	Smooth Avg (expected)	Smooth Max (expected)	Circuits
Smooth Min (experimental)	10%	22%	32%	Inputs: IPTG, aTc ANF (P_{lacO1}), ANF (P_{tetO}), combinatorial promoter ($P_{lacO/tetO}$), AraC /P_{BAD}-GFP Arabinose-low Circuit – Fig. 1G Results: Fig. 1H Transformed results: Fig. 1J

Smooth Max (experimental)	470%	256%	23%	Inputs: AHL, aTc APF (P_{luxTGT}), ANF (P_{tetO}), combinatorial promoter ($P_{lux/tetO}$), AraC /P_{BAD}-GFP Arabinose-high Circuit – Fig. 2D Results: Fig. 2E Transformed results: Fig. 2F
Smooth Avg (experimental)	66%	8.5%	24%	Inputs: AHL, IPTG APF (P_{luxAAT}), ANF (P_{lacO}), combinatorial promoter ($P_{lux/tetO}$), AraC /P_{BAD}-GFP Arabinose-high Circuit – Fig. 2G Results: Fig. 2H Transformed results: Fig. 2I

Table S3.12 List of abbreviations used in this section

Symbol	Description
x_i	Analog numbers
$P_{lacO/tetO}$	combinatorial promoter
IPTG	Free Isopropyl 1-β-D-1-thio galactopyranoside concentration
aTc	Free anhydrotetracycline
IDR	Input dynamic range
$P_{lux/tetO}$	combinatorial promoter
$P_{lux/lacO}$	combinatorial promoter

4. Calculations of parameters for a single perceptgene

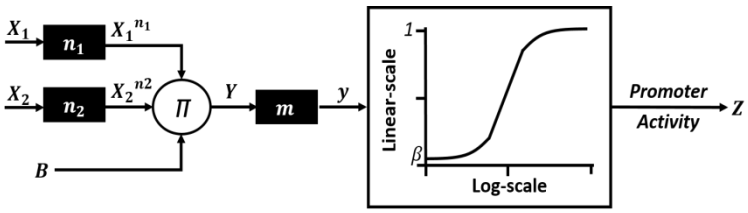


Fig. S4.1. Basic structure of perceptgenes.

The output of power-law and multiplication circuit can be approximated as:

$$Y = \prod_{i=1}^{N} Y_m \cdot \left(\frac{X_i}{K_{mi}}\right)^{n_i} \quad \text{(S4.1)}$$

Where X_i is the input concentration, K_{mi} is the dissociation constant or scalar normalization or input dynamic range. Y_m has units of concentration, and it equals the maximum level of produced transcription factors. n_i is Hill-coefficient of the input X_i. The operation range of the circuit is defined $X_{Li} < X_i < X_{Hi}$, where$IDR = \log(\frac{X_{Hi}}{X_{Li}})$. Then we can rewrite Eq. S4.1 as:

$$Y = \prod_{i=1}^{N} Y_m \cdot \left(\frac{X_i}{X_{Li}}\right)^{n_i} \cdot \left(\frac{X_{Li}}{K_{mi}}\right)^{n_i} \quad \text{(S4.2)}$$

Assuming that $X_{Li} \approx K_{mi}$, relevant parameters used in models are listed in Table S4.1

System [Inpu1, Input2]	[IPTG, aTc]	[AHL, aTc]	[AHL, IPTG]
X_{L1}	$1\ \mu M$	$90\ nM$	$90\ nM$
K_{m1}	$1\ \mu M$ (*21*)	$125\ nM$ (*16*)	$125\ nM$
X_{L2}	$0.4\ ng/mL$	$0.4\ ng/mL$	$1\ \mu M$
K_{m2}	$1.7\ ng/mL$ (*16*)	$1.7\ ng/mL$	$1\ \mu M$

Table S4.1. The Lowest input values were used in our circuits, and the dissociation constants for binding IPTG -LacI, AHL -LuxR, and aTc -TetR

The measured signal of the power-law and multiplication circuit is given:

$$GFP \approx \xi \cdot \prod_{i=1}^{N} Y_m \cdot \left(\frac{X_i}{X_{Li}}\right)^{n_i} \quad \text{(S4.3)}$$

Where ξ is the efficiency of converting GFP molecules to optical signals. The minimum measured signal achieved is when $X_i = X_{Li}$:

$$GFP_{min} = \xi \cdot \prod_{i=1}^{N} Y_m \cdot \left(\frac{X_{Li}}{K_{mi}}\right)^{n_i} \quad \text{(S4.4)}$$

Then, the normalized signal of the power-law and multiplication circuit is given by:

$$GFP_N = \frac{GFP}{GFP_{min}} = \prod_{i=1}^{N} \left(\frac{X_i}{X_{Li}}\right)^{n_i}$$

$$\log(GFP_N) = n_i \cdot \sum_{i=1}^{N} \left[\log\left(\frac{x_i}{x_{Li}}\right)\right] \quad \text{(S4.5)}$$

The promoter activity is initiated when the transcription factor Y binds and is given by:

$$P_r = \frac{\left(\frac{Y}{K_d}\right)^m + \beta}{1+\beta+\left(\frac{Y}{K_d}\right)^m} \quad \text{(S4.6)}$$

β is the basal level of the promoter, K_d is the dissociation constant of binding Y to promoter, and $\boldsymbol{m}$ is the Hill-coefficient (number of binding sites within the promoter). Substituting Eq. S4.2 into Eq. S4.6 gives:

$$P_r = \frac{\left(\frac{\prod_{i=1}^{N} Y_m \cdot \left(\frac{X_i}{X_{Li}}\right)^{n_i}}{K_d}\right)^m + \beta}{1+\beta+\left(\frac{\prod_{i=1}^{N} Y_m \cdot \left(\frac{X_i}{X_{Li}}\right)^{n_i}}{K_d}\right)^m}$$

$$P_r = \frac{\left(\prod_{i=1}^{N} \frac{Y_m}{K_d} \cdot \left(\frac{X_i}{X_{Li}}\right)^{n_i}\right)^m + \beta}{1+\beta+\left(\prod_{i=1}^{N} \frac{Y_m}{K_d} \cdot \left(\frac{X_i}{X_{Li}}\right)^{n_i}\right)^m}$$

$$y = \left(\prod_{i=1}^{N} B \cdot {x_i}^{n_i}\right)^m \quad \text{(S4.7)}$$

$$P_r = \frac{y+\beta}{1+\beta+y} \quad \text{(S4.8)}$$

where $x_i = X_i/X_{Li}$, and $B = Y_m/K_d$

The minimum promoter activity is $P_{min} \approx \beta$, and the maximum promoter activity is $P_{max} \approx 1$. The perceptgene is designed as a modular, meaning that the output of the first layer acts as the input of the second layer. Also, the decision at the perceptgene output should be made at the logarithmic-scale. Therefore, we normalized the promoter activity by the basal level:

$1 \le P_r \le 1/\beta$ ➔ linear-scale/log transform: $0 \le \log(P_r) \le -\log(\beta)$ (S4.9)

In analogy to perceptron, we approximated the activation function as a step function:

$$\begin{cases} m \cdot n_i \cdot \sum_{i=1}^{N} x_i - m \cdot B \ge \log(Th) & \log(P_r) = -\log(\beta) \\ \text{Otherwise} & \log(P_r) = 0 \end{cases} \quad \text{(S4.10)}$$

We define the *Th* is the effective threshold of the activation function and is set by the Basel level (Fig. S4.2):

$$10^{log(\beta)/2} = \frac{Th+\beta}{1+Th+\beta} \quad \text{(S4.11)}$$

$$Th = \frac{10^{log(\beta)/2} - \beta}{1 - 10^{log(\beta)/2}}$$

The measured signal of the perceptgene circuit in steady state is given:

$$GFP = \xi \cdot GFP_{max} \cdot P_r \quad \text{(S4.12)}$$

where $\mathrm{GFP_{max}}$ is the maximum GFP achieved by the promoter. Then the normalized signal of the preceptgene circuit is given by:

$$GFP_N = \frac{GFP}{GFP_{min}} = 1 + \frac{\xi \cdot GFP_{max} \cdot P_r}{\xi \cdot GFP_{max} \cdot (B^{N+m} + \beta)} \quad \text{(S4.13)}$$

Here $\mathrm{GFP_{min}}$ is the minimum GFP achieved by the perceptgene ($x_i = 1$). The maximum fold change of the perceptgene ($\mathrm{GFP_{N\_max}}$) is achieved when the normalized inputs equal to IDR (input dynamic range), then we can assume that the promoter activity is approximately 1. Therefore, the maximum fold change is given by:

$$GFP_{N\_max} = \frac{1}{B^m + \beta}$$

$$B^m = \frac{1-\beta}{GFP_{N\_max}} \approx \frac{1}{GFP_{N\_max}} \quad \text{(S4.14)}$$

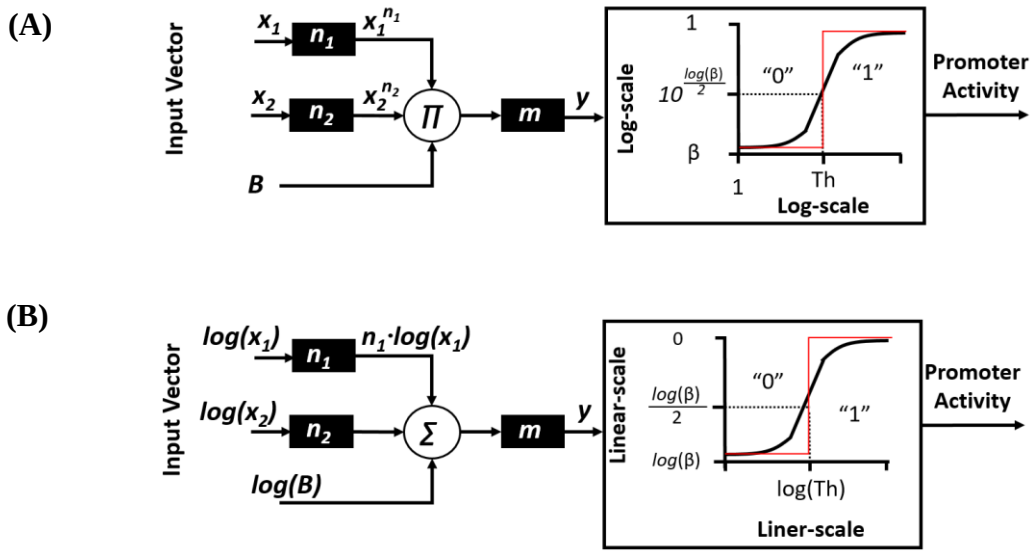


Fig. S4.2. Description of the perceptgene model in the (A) logarithmic-domain and (B) linear-domain.

In this work, we used the fundamental properties of ANNs to create genetic circuits that encoded the calculations of smooth maximum, the smooth minimum, and the average of two analog inputs (Supplementary analysis, Section 3). In our implementation the cooperativity, represented by the Hill coefficient, acts as a weight (m, n_i), and the node threshold is set by the basal level. Additionally, the bias is set by a linear function of the translation/transcription rates, the mRNA/protein half-life and cell growth rate divided by the binding affinities of protein-protein/protein-DNA reactions (Supplementary Information, BOX1). Correspondingly, the logarithmic equivalent product of $m \cdot \sum_i n_i \cdot (X_i/IDR_i) + Bias > Threshold$ sets the operation type shown in Table S7 (*e.g.*, the maximum function uses a positive rectifier activation, the minimum function uses a negative rectifier function, and the average function uses a linear activation function; Supplementary Figs. 3.1, 3.4). To enable different operation types, we controlled the Hill coefficient of the P_{BAD} promoter (m) by adjusting the arabinose concentration (Supplementary Fig. S2.11).

Table S4.2. shows the calculations used to define the operation of each perceptgene circuit

System	[IPTG, aTc]	[aTc, AHL]	[AHL, IPTG]
n_1	0.75	0.8	1.9
n_2	1	1	1.65
IDR_1 at log-scale	$\log(128) = 2.1$	$\log(64) = 1.8$	$\log(16) = 1.2$
IDR_2 at log-scale	$\log(128) = 2.1$	$\log(32) = 1.5$	$\log(32) = 1.5$
M	2	1	1
Maximum Fold Change	16	10	14

Bias at log-scale (B)	$\log(1/16^{1/2}) = -0.6$	$\log(1/10) = -1$	$\log(1/14) = -1.1$
β – Basal level	0.001	0.001	0.001
log(Th)	-1.5	-1.5	-1.5
$B \cdot m - Th$	-2.7	-2.5	-2.6
$n_1 \cdot m_1 \cdot IDR_1 + B \cdot m - Th$	0.45	-1	-0.3
$n_2 \cdot m_2 \cdot IDR_2 + B \cdot m - Th$	1.5	-1	-0.2
$n_1 \cdot m_1 \cdot IDR_1 + n_2 \cdot m_2 \cdot IDR_2 + B \cdot m - Th$	4.5	0.45	2.2
Operation type	Minimum	Maximum	Average

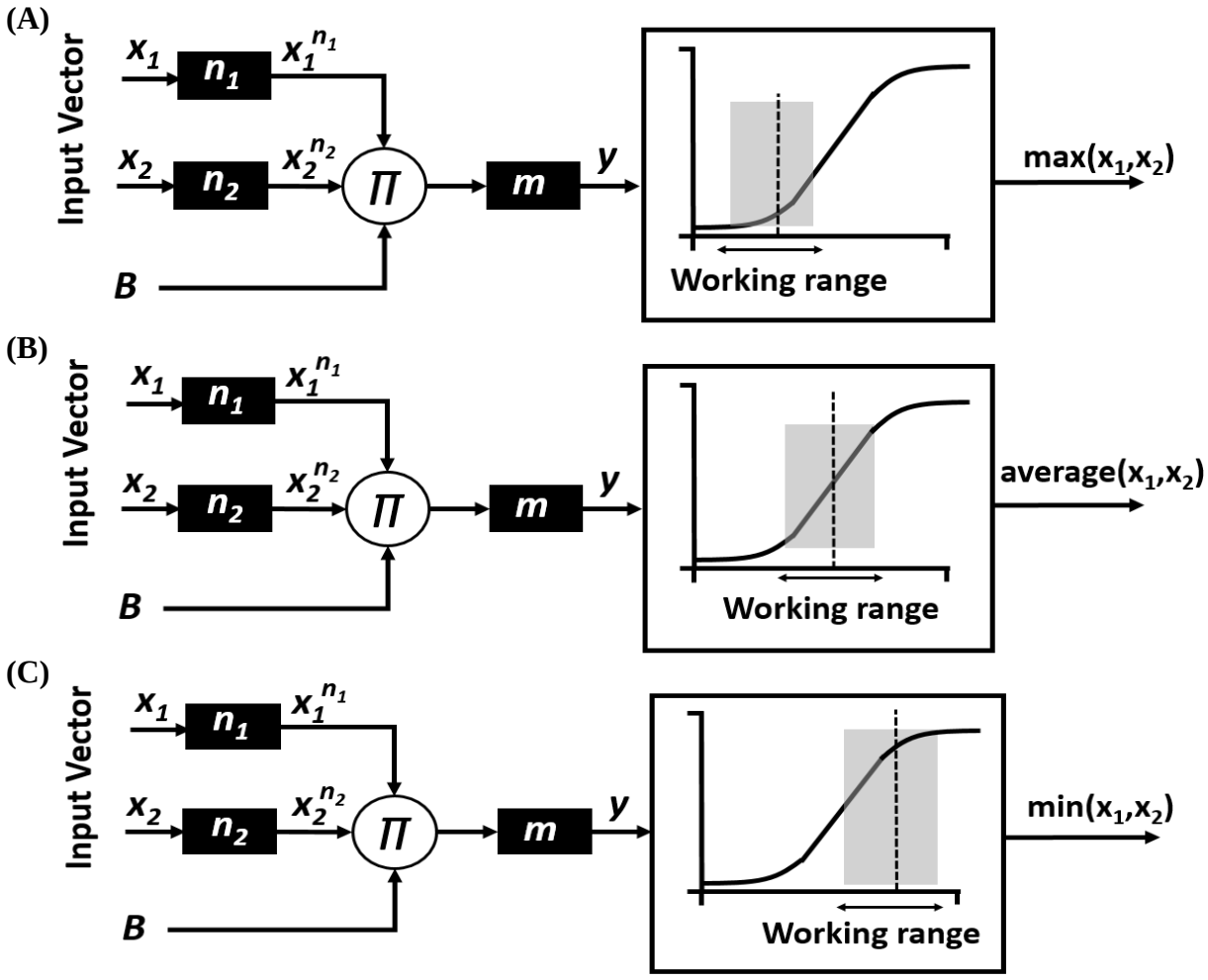


Fig. S4.3. Operation type of perceptgene. (A) Maximum, (B) average, and (C) minimum.

Fitting experimental results of the power-law and multiplication functions with respect to their input dynamic ranges ($IDRs$):

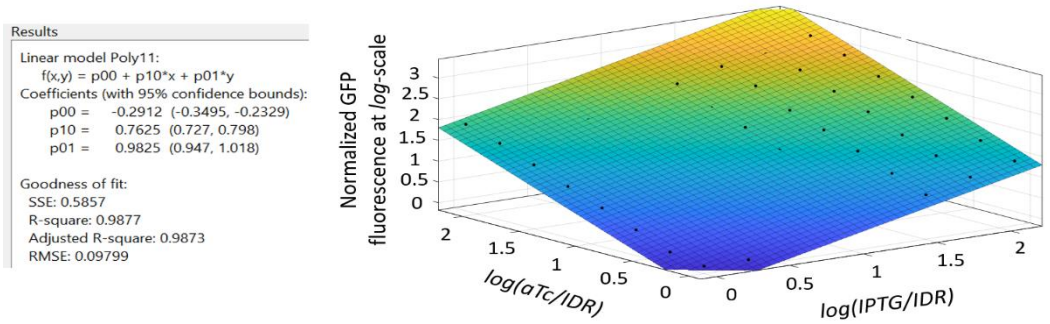


Fig. S4.4. Matlab surface fits the experimental results of P_{lacO1} and P_{tetO} ANF loops and combinatorial promoter ($P_{lacO/tetO}$-GFP) to power-law and multiplication function with respect to IDR axis.

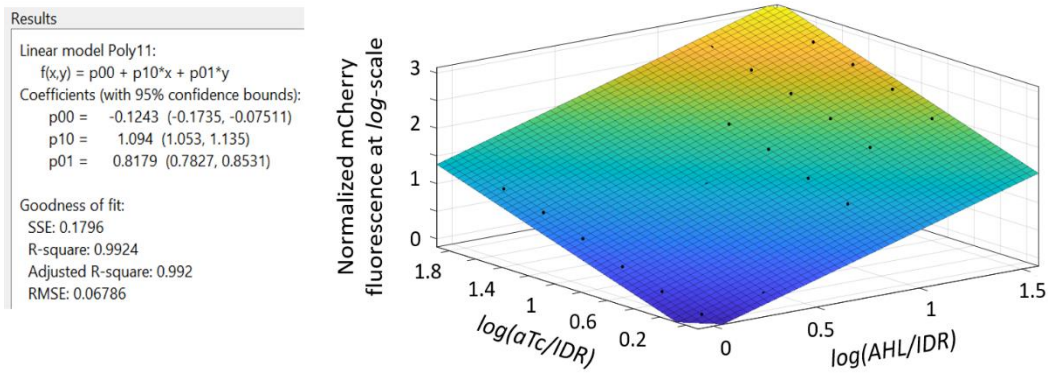


Fig. S4.5. Matlab surface fits the experimental results of APF (P_{luxTGT}) and ANF (P_{tetO}) loops and combinatorial promoter ($P_{lux/tetO}$- mCherry) to power-law and multiplication function with respect to IDR axis.

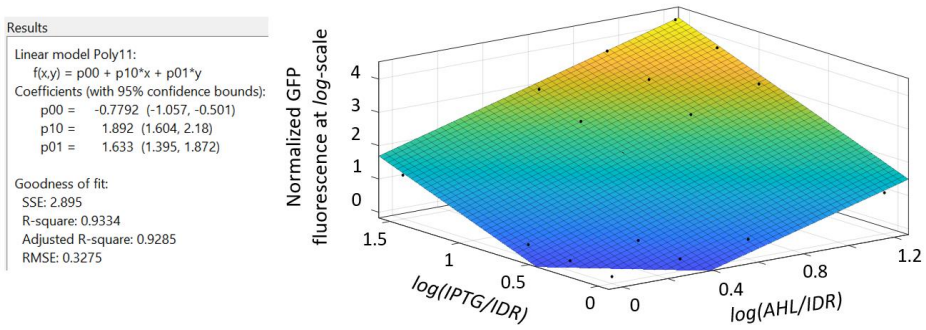


Fig. S4.6. Matlab surface fits the experimental results of APF (P_{luxAAT}) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function with respect to IDR axis.

Circuits that calculate minimum and maximum input values can be used to construct conjunction and disjunction logic functions (Fig. S4.7). For example, the minimum between {“0”, “1”} is 0, i.e., “0” AND “1”; the maximum between {“0”, “1”} is 1, i.e., “0” OR “1”. Therefore, we can apply maximum/minimum perceptgene-based circuits to implement logical computation functions in living cells. **Controlling the weights can be achieved via processes such as splitting proteins** (*22–25*)**.**

x_1	x_2	$AND=min(x_1,x_2)$	$OR=max(x_1,x_2)$
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

Fig. S4.7. Truth table of AND/OR logic gates demonstrate minimum/maximum functions, respectively.

5. Design and model of multilayer perceptgene networks

In this section we present a biophysical model that describes the behavior of multilayer perceptgene network at steady state. We show that biophysical models can be described in a similar fashion to neural networks with three components: (1) weights that are represented by Hill-coefficients and cooperativity, (2) bias constants that are proportional to the ratio of the total synthesized proteins and promoter binding affinities and (3) activation functions that are represented by a promoter activity and are described by Michaelis-Menten kinetics (Supplementary information, Section 1, Box1).

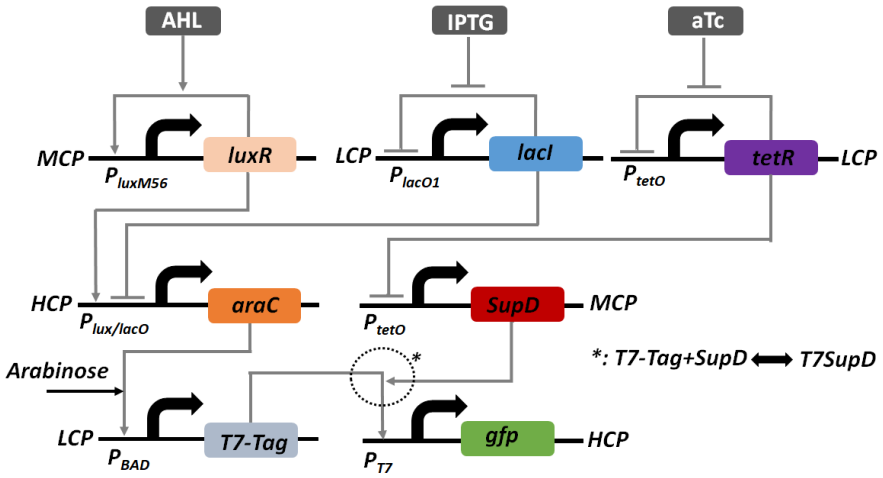


Fig. S5.1. Multilayer perceptgene network accepts three analog inputs (AHL, IPTG and aTc).

In the first layer, the AraC protein is regulated by a circuit consists of a graded APF, an ANF loop and a combinatorial promoter ($P_{lux/lacO}$). The first layer circuit displays a power-law and multiplication function (Fig. S.5.5-5.7). The APF loop is induced by AHL and the ANF loop is induced by IPTG. A weak mutated P_{luxM56} promoter was used in the APF part to broaden the IDR of AHL (Fig. S2.13). The activity of AraC protein upon inducers AHL and IPTG is described by:

$$y_1 = y_{m1} \cdot \left(\frac{AHL}{K_{m1}}\right)^{n_1} \cdot \left(\frac{IPTG}{K_{m2}}\right)^{n_2} \quad \text{(S5.1)}$$

Where y_{m1} has units of concentration, and it depends on the binding affinity between transcription factors and the corresponding promoter, as well as the maximum level of transcription factor (Eq. S2.13). The experimental results of $P_{lux/lacO}$-based power-law and multiplication circuit are shown in Fig. S.5.5-5.7, which are well fitted using Eq. S5.1.

The AraC proteins are expressed as the output of the first layer. Subsequently, they interact with the P_{BAD} promoter, which further regulates the T7-RNA polymerase. The activity of the T7-RNA can be modeled in the following way

$$z_1 = z_{m1} \frac{(y_1/K_1)^{m_1} + \beta_1}{1 + \beta_1 + (y_1/K_1)^{m_1}} \quad \text{(S5.2)}$$

$$K_1 = a \cdot {m_1}^{-b} \quad \text{(S5.3)}$$

Where K_1 is the dissociation constant of Arabinose- AraC complex binding to P_{BAD} promoter, β_1 is the basal level of P_{BAD}, and m_1 is the effective Hill-coefficient. We have shown that K_1 and m_1 can be tunable by the Arabinose concentration level (Fig. S2.9) with a power-law relation.

The amber suppressor tRNA $supD$ is regulated by P_{tetO}-based ANF, and is given by:

$$y_2 = y_{m2} \cdot \left(\frac{aTc}{K_{m3}}\right)^{n_3} \quad \text{(S5.4)}$$

where y_{m2} has units of concentration. The experimental results of P_{tetO}-based ANF are shown in Fig. S.5.8**,** which are fitted well to a power-law function using Eq. S5.1. The biochemical binding reaction between T7 RNA polymerase and $supD$ is given by(*26*):

$$T7_{RNA} + supD \leftrightarrow T7_{RNA}SupD \quad \text{(S5.5.1)}$$

$$T7_{RNA}SupD + supD \leftrightarrow T7_{RNA}SupD_2 \quad \text{(S5.5.2)}$$

A simple solution to the set of Eq. S5.5.1 and Eq. S5.5.2 at the steady-state gives:

$$y_3 = y_{m3} \cdot \frac{(y_2/K_2)^{m_2} \cdot K_3 \cdot (z_1/K_3)^{m_3}}{K_4} \quad \text{(S5.5.3)}$$

Here we assumed that there are two amber stop codons in the open reading frame, which should lead to $m_2 = 2$ (*26*). However, as the first biochemical reaction in Eq. S5.5 ($T7_{RNA} + supD \leftrightarrow T7_{RNA}SupD$) can also bind to T7 promoter with a probability larger than zero, the effective value of m_2 can be reduced to less than 2. m_3 depends on the protein quaternary structure (the number of subunits that interact with each other and arrange themselves to form a final structure of the protein). Since the T7 RNA polymerase is a single subunit (*26*, *27*), $m_3 = 1$. K_2, K_3 and K_4 are the dissociation constants of biochemical reactions in Eq. S5.5. The T7 RNA polymerase was regulated by a ribosome binding sequence with a very low binding affinity (BBa_B0031) (*4*, *28*). The binding of $T7_{RNA}SupD_2$ complex to T7 promoter, actives it and expresses GFP. This process demonstrates the output of the second perceptgene layer, and is given by:

$$z_2 = z_{m2} \frac{(y_3/K_5)^{m_4} + \beta_2}{1 + \beta_2 + (y_3/K_5)^{m_4}} \quad \text{(S5.6)}$$

Where K_5 is the dissociation constant $T7_{RNA}SupD_2$ complex to P_{T7} , β_2 is the basal level of P_{T7}, and m_4 is the effective Hill-coefficient. Rewriting the set of Eq. S5.1-Eq. S5.6, gives:

$$y_1 = \left(B_1 \cdot \left(\frac{AHL}{K_{m1}}\right)^{n_1} \cdot \left(\frac{IPTG}{K_{m2}}\right)^{n_2}\right)^{m_1} \quad \text{(S5.7)}$$

$$z_1 = \frac{y_1 + \beta_1}{1 + \beta_1 + y_1} \quad \text{(S5.8)}$$

$$y_2 = \left(B_2 \cdot \left(\frac{x_3}{K_{m3}}\right)^{n_3}\right)^{m_2} \quad \text{(S5.9)}$$

$$y_3 = (B_4 \cdot y_2 \cdot (B_3 \cdot z_1)^{m_3})^{m_4} \quad \text{(S5.10)}$$

$$z_2 = \frac{y_3 + \beta_2}{1 + \beta_2 + y_3} \quad \text{(S5.11)}$$

Where: $B_4 \equiv \frac{y_{m3}}{K_5} \cdot \frac{K_3}{K_4}$, $B_3 \equiv \frac{z_{m1}}{K_3}$, $B_2 \equiv \frac{y_{m2}}{K_2}$, $B_1 \equiv \frac{y_{m1}}{a} \cdot {m_1}^b$

An abstract model of the set of Eq. S5.7-Eq. S5.11 is shown in Fig. S5.2, and is built from three computational components:

(1) Network Weights (n_i and m_i): are represented by effective Hill-coefficients, which depend on biological cooperativities of protein interactions and protein quaternary structure (the number of subunits that interact with each other and arrange themselves to form a final protein).

(2) Bias constants (B_i): are represented by translation/transcription rates, mRNA/protein half-lives, rates of cell growth, binding affinities in protein-protein or protein-DNA interactions. Bias constant is unit-less.

(3) Activation functions or network nodes (z_i- the output of each perceptgene) which are represented by a promoter activity and is given by a normalized Michaelis-Menten model $y_i/(1+y_i)$.

As in Fig. S5.2, each input is normalized and fed into a power-law function with an exponent n_i. Then the product is multiplied with a bias constant, giving rise to an analog signal (y_i). Applying a logarithmic operation to the signal, we get: $log(y_i) = m_l \cdot log(B_i) + m_l \cdot \sum_j n_j \cdot log(I_{ni})$ (I_{ni} is the normalized input). The bias acts as a reference level.

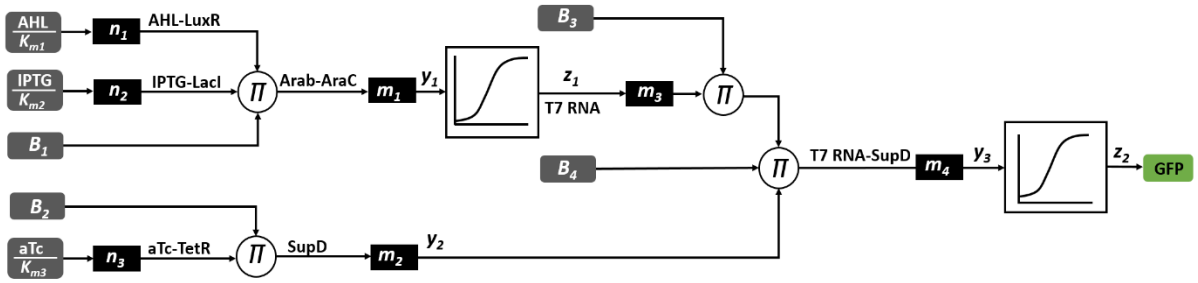


Fig. S5.2. Abstract model of Multilayer perceptgene in analogy to abstract models of artificial neural networks. Comparing to Fig. 3B, the only difference is that B_3 is included in B_4.

5.1. Design of 3-input majority function

The 3-input **majority function** (also called the **median operator**) describes a logic function from three inputs to one output. The output is a high- "1" if and only if the majority of the inputs are high- "1". Otherwise, the output is a low- "0". The majority function can be found in various applications such as adders and subtractors (*29*). The truth table and the implementation of the 3-input majority function using a logic gate design are shown in Fig. S5.3A and B. The early works with artificial neural networks were based on a linear threshold unit (LTU) and were targeted to serve as a computational model that can implement any Boolean logic function (*30*). The implementation of a 3-input majority function (Fig. S5.1) using a two-layer perceptgene network is based on using principles of artificial neural networks (Fig. S5.2). First, we solved the set of Eq. S5.7-Eq. S5.11, and then converted them to the linear domain using a logarithmic transformation. Subsequently, an activation function was applied to the transformed output. For simplicity, we approximated the activation function of the last layer as a step function in the log-scale:

$$z_2 = \begin{cases} 10 & y_3 \geq 1 \\ 1 & y_3 < 1 \end{cases} \quad \text{(S5.12)}$$

And the activation function $[(y_1+\beta_1)/(1+y_1+\beta_1)]$ in the hidden layer was approximated (e.g. Fig. S5.3C) as:

$$z_1 = \begin{cases} 1 & y_1 \geq \alpha \\ \beta_1 < z_1 < 1 & 1/\alpha \leq y_1 < \alpha \\ \beta_1 & y_1 < 1/\alpha \end{cases} \quad \text{(S5.13)}$$

- We assumed that the normalized input (x_i) is between 1 and 10
- We define:

  $x_1 \equiv \frac{AHL}{K_{m1}}$, $x_2 \equiv \frac{IPTG}{K_{m2}}$, $x_3 \equiv \frac{aTc}{K_{m3}}$

  $A_1 \equiv log\left(\frac{y_{m1}}{a}\right)$, $\gamma \equiv \log(\alpha)$, $b_1 \equiv \log(\beta_1)$

  $A_5 \equiv \log(B_4) + m_2 \cdot \log(B_2) + m_3 \cdot \log(B_3)$

  $B_5 \equiv 10^{A_5} = B_4 \cdot {B_2}^{m_2} \cdot {B_3}^{m_3}$

$B_1 \equiv 10^{A_1} \cdot {m_1}^b$

A simplified model of the set of equations shows that the network consists of two-layers perceptgene, as shown in Fig. S5.3D.

State 0: $x_1 = 1, x_2 = 1, x_3 = 1,\ z_2 = 1;$

We require that $y_1 = {B_1}^{m_1} < 1/\alpha$

➔ $m_1 \cdot \log(B_1) < -\log(\alpha)$

➔ $m_1 \cdot \left(\log\left(\frac{y_{m1}}{a}\right) + b \cdot \log(m_1)\right) < -\log(\alpha)$

⇨ $m_1 \cdot (A_1 + b \cdot \log(m_1)) < -\gamma$ (S5.14.1)

We require that $z_1 = \beta_1$

$y_2 = {B_2}^{m_2}$

$y_3 = (B_4 \cdot {B_2}^{m_2} \cdot (B_3 \cdot \beta_1)^{m_3})^{m_4} < 1$

➔ $m_4 \cdot (\log(B_4) + m_2 \cdot \log(B_2) + m_3 \cdot \log(B_3) + m_3 \log(\beta_1)) < 0$

⇨ $m_4 \cdot (A_5 + m_3 \cdot b_1) < 0$ (S5.14.2)

State 1: $x_1 = 1, x_2 = 1, x_3 = 10,\ z_2 = 1;$

We require that $y_1 = {B_1}^{m_1} < 1/\alpha$

⇨ $m_1 \cdot (A_1 + b \cdot \log(m_1)) < -\gamma$

We require that $z_1 = \beta_1$

$y_2 = {B_2}^{m_2} \cdot 10^{n_3 m_2}$

$y_3 = (B_4 \cdot {B_2}^{m_2} \cdot 10^{n_3 m_2} \cdot (B_3 \cdot z_1)^{m_3})^{m_4} < 1$

➔ $m_4 \cdot (\log(B_4) + m_2 \cdot \log(B_2) + m_3 \cdot \log(B_3) + m_2 \cdot n_3 + m_3 \log(\beta_1)) < 0$

⇨ $m_4 \cdot (A_5 + m_2 \cdot n_3 + m_3 \cdot b_1) < 0$ (S5.14.3)

State 2: $x_1 = 1, x_2 = 10, x_3 = 1,\ z_2 = 1;$

We require that $1/\alpha < y_1 < \alpha$:

$y_1 = (B_1 \cdot (10)^{n_2})^{m_1}$

è $1/\alpha < (B_1 \cdot (10)^{n_2})^{m_1} < \alpha$

➔ $-\log(\alpha) < m_1 \cdot (\log(B_1) + n_2) < \log(\alpha)$

⇨ $-\gamma < m_1 \cdot (A_1 + b \cdot \log(m_1) + n_2) < \gamma$ (S5.14.4)

We require that: $\beta_1 < z_1 < 1$

$y_2 = {B_2}^{m_2}$

$y_3 = (B_4 \cdot {B_2}^{m_2} \cdot (B_3 \cdot z_1)^{m_3})^{m_4} < 1$

➔ $m_4 \cdot (\log(B_4) + m_2 \cdot \log(B_2) + m_3 \cdot \log(B_3) + m_3 \log(z_1)) < 0$

⇨ $m_4 \cdot (A_5 + m_3 \log(z_1)) < 0$ (S5.14.5)

State 3: $x_1 = 1, x_2 = 10, x_3 = 10,\ z_2 = 10;$

We require that $1/\alpha < y_1 < \alpha$:

$y_1 = (B_1 \cdot (10)^{n_2})^{m_1}$

$-\gamma < m_1 \cdot (A_1 + b \cdot \log(m_1) + n_2) < \gamma$

We require that: $\beta_1 < z_1 < 1$

$y_2 = {B_2}^{m_2} \cdot 10^{n_3 m_2}$

$y_3 = (B_4 \cdot {B_2}^{m_2} \cdot 10^{n_3 m_2} \cdot (B_3 \cdot z_1)^{m_3})^{m_4} \geq 1$

➔ $m_4 \cdot (\log(B_4) + m_2 \cdot \log(B_2) + m_3 \cdot \log(B_3) + m_2 \cdot n_3 + m_3 \log(z_1)) \geq 0$

⇨ $m_4 \cdot (A_5 + m_2 \cdot n_3 + m_3 \log(z_1)) \geq 0$ (S5.14.6)

State 4: $x_1 = 10, x_2 = 1, x_3 = 1,\ z_2 = 1;$

We require that $1/\alpha < y_1 < \alpha$:

$y_1 = (B_1 \cdot (10)^{n_1})^{m_1}$

➔ $1/\alpha < (B_1 \cdot (10)^{n_1})^{m_1} < \alpha$

➔ $-\log(\alpha) < m_1 \cdot (\log(B_1) + n_1) < \log(\alpha)$

⇨ $-\gamma < m_1 \cdot (A_1 + b \cdot \log(m_1) + n_1) < \gamma$ (S5.14.7)

We require that: $\beta_1 < z_1 < 1$

$y_2 = {B_2}^{m_2}$

$y_3 = (B_4 \cdot {B_2}^{m_2} \cdot (B_3 \cdot z_1)^{m_3})^{m_4} < 1$

⇨ $m_4 \cdot (A_5 + m_3 \log(z_1)) < 0$

State 5: $x_1 = 10, x_2 = 1, x_3 = 10,\ z_2 = 10;$

We require that $1/\alpha < y_1 < \alpha$:

$y_1 = (B_1 \cdot (10)^{n_1})^{m_1}$

⇨ $-\gamma < m_1 \cdot (A_1 + b \cdot \log(m_1) + n_1) < \gamma$

We require that: $\beta_1 < z_1 < 1$

$y_2 = {B_2}^{m_2} \cdot 10^{n_3 m_2}$

$y_3 = (B_4 \cdot {B_2}^{m_2} \cdot 10^{n_3 m_2} \cdot (B_3 \cdot z_1)^{m_3})^{m_4} \geq 1$

➔ $m_4 \cdot (\log(B_4) + m_2 \cdot \log(B_2) + m_3 \cdot \log(B_3) + m_2 \cdot n_3 + m_3 \log(z_1)) \geq 0$

⇨ $m_4 \cdot (A_5 + m_2 \cdot n_3 + m_3 \log(z_1)) \geq 0$ (S5.14.8)

State 6: $x_1 = 10, x_2 = 10, x_3 = 1,\ z_2 = 10;$

We require that $y_1 > \alpha$

➔ $y_1 = (B_1 \cdot (10)^{n_1} \cdot (10)^{n_2})^{m_1} > \alpha$

➔ $m_1 \cdot (\log(B_1) + n_1 + n_2) > \log(\alpha)$

⇨ $m_1 \cdot (A_1 + b \cdot \log(m_1) + n_1 + n_2) > \gamma$ (S5.14.9)

We require that $z_1 = 1$

$y_2 = {B_2}^{m_2}$

$y_3 = (B_4 \cdot {B_2}^{m_2} \cdot (B_3)^{m_3})^{m_4} \geq 1$

➔ $m_4 \cdot (\log(B_4) + m_2 \cdot \log(B_2) + m_3 \cdot \log(B_3)) \geq 0$

⇨ $m_4 \cdot (A_5) \geq 0$ (S5.14.10)

State 7: $x_1 = 10, x_2 = 10, x_3 = 10,\ z_2 = 10;$

We require that $y_1 > \alpha$

$y_1 = (B_1 \cdot (10)^{n_1} \cdot (10)^{n_2})^{m_1} \geq \alpha$

⇨ $m_1 \cdot (A_1 + b \cdot \log(m_1) + n_1 + n_2) > \gamma$

We require that $z_1 = 1$

$y_2 = {B_2}^{m_2} \cdot 10^{n_3 m_2}$

$y_3 = (B_4 \cdot {B_2}^{m_2} \cdot 10^{n_3 m_2} \cdot (B_3)^{m_3})^{m_4} \geq 1$

➔ $m_4 \cdot (\log(B_4) + m_2 \cdot \log(B_2) + m_3 \cdot \log(B_3) + m_2 \cdot n_3) \geq 0$

⇨ $m_4 \cdot (A_5 + m_2 \cdot n_3) \geq 0$ (S5.14.11)

Summary: The design conditions of A_1 are set by:

I. $A_1 < -\frac{\gamma}{m_1} - b \cdot \log(m_1)$

II. $-\frac{\gamma}{m_1} - b \cdot \log(m_1) - n_2 < A_1 < \frac{\gamma}{m_1} - b \cdot \log(m_1) - n_2$

III. $A_1 > \frac{\gamma}{m_1} - b \cdot \log(m_1) - n_1 - n_2$

Therefore, $A_{1min} < A_1 < A_{1max}$ (S5.15.1)

where:

$A_{1max} = min\left\{-\frac{\gamma}{m_1} - b \cdot \log(m_1), \frac{\gamma}{m_1} - b \cdot \log(m_1) - \max(n_1, n_2)\right\}$ (S5.15.2)

$A_{1min} = max\left\{\frac{\gamma}{m_1} - b \cdot \log(m_1) - n_1 - n_2, -\frac{\gamma}{m_1} - b \cdot \log(m_1) - \min(n_1, n_2)\right\}$ (S5.15.3)

We require that $A_{min} < A_{max}$. To gain deeper insights into the effects of A_1 on the behavior of the network, we consider two asymptotic cases:

1. $\gamma \ll 0$; a step function is used as the activation function in the hidden layer
   $A_{1max} = -b \cdot \log(m_1) - \max(n_1, n_2)$
   $A_{1min} = -b \cdot \log(m_1) - \min(n_1, n_2)$ (S5.16.1)
   $A_{1max} < A_{1min}$, unachievable condition. Therefore, we cannot implement a majority function using two perceptgene layers with a step function in the hidden layer
2. $\gamma \gg 0$, an analog function is used as the activation function in the hidden layer
   $A_{1max} = -\frac{\gamma}{m_1} - b \cdot \log(m_1)$
   $A_{1min} = \frac{\gamma}{m_1} - b \cdot \log(m_1) - n_1 - n_2$
   $A_{1max} > A_{1min}$, which is achievable when:
   $\frac{\gamma}{m_1} < \frac{n_1+n_2}{2}$ (S5.16.2)

We now continue with the first condition in Eq. S5.15.1, we assume that:

$-\frac{\gamma}{m_1} - b \cdot \log(m_1) < \frac{\gamma}{m_1} - b \cdot \log(m_1) - \max(n_1, n_2)$

⇨ $2 \cdot \gamma > \max(n_1, n_2) \cdot m_1$ (S5.17.1)

Under these conditions, we get:

$\rightarrow \boldsymbol{A_{1max}} = -\frac{\gamma}{m_1} - b \cdot \log(m_1)$ (S5.17.2)

According to the second condition in Eq. S5.15.2, we should require:

$\frac{\gamma}{m_1} - b \cdot \log(m_1) - n_1 - n_2 > -\frac{\gamma}{m_1} - b \cdot \log(m_1) - \min(n_1, n_2)$

⇨ $2 \cdot \gamma > (n_1 + n_2 - \min(n_1, n_2)) \cdot m_1$ (S5.17.3)

The conditions in Eq. S5.18.1 and Eq. S5.18.3 are similar
Under these conditions we get:

$\rightarrow \boldsymbol{A_{1min}} = \frac{\gamma}{m_1} - b \cdot \log(m_1) - n_1 - n_2$ (S5.17.4)

As a summary, we get:

$\frac{\gamma}{m_1} - b \cdot \log(m_1) - n_1 - n_2 < A_1 < -\frac{\gamma}{m_1} - b \cdot \log(m_1)$ (S5.18)

Summary; The design conditions of A_5 are set by:

I. $A_5 < -m_3 \cdot b_1$
II. $A_5 < -m_2 \cdot n_3 - m_3 \cdot b_1$
III. $A_5 < -m_3 \log(z_1)$
IV. $A_5 \geq 0$
V. $A_5 \geq -m_2 \cdot n_3$
VI. $A_5 \geq -m_2 \cdot n_3 - m_3 \log(z_1)$

Therefore $A_{5min} < A_5 < A_{5max}$ (S5.19.1)
where

$A_{5max} = min\{-b_1, -m_2 \cdot n_3 - b_1, -m_3 \log(z_1)\}$
$A_{5min} = max\{0, -m_2 \cdot n_3, -m_2 \cdot n_3 - m_3 \log(z_1)\}$

The Basal level of promoter often is $\beta_1 < 1$, and therefore $b_1 < 0$, and $\log(z_1) \leq 0$, therefor:

$0 < A_5 < -m_2 \cdot n_3 - b_1$ (S5.19.2)

The simulation results as in Fig. S5.4A show that for $m_1 = 2$, a 3-input majority function can be implemented by a two-layer perceptgene network. Parameters used: $b = 0.73, \alpha = 10, n_1 = 1, n_2 = 1.5, n_3 = 0.7, m_4 = 1, m_2 = 1.5, m_3 = 1$ (T7 RNA polymerase is a single subunit), $B_1 = 0.05 (0.006 < B_1 < 0.2$, margin of two orders), $B_5 = 1.25$, $\beta_1 = 0.001, \beta_2 = 0.002$. We used a Michaelis-Menten model to model the activity of z_1. The simulation results also show that for low m_1 (e.g. $m_1 = 1$), it is challenging to implement a 3-input majority function using a two-layer peceptgene network, because the design parameters do not satisfy the conditions in Eq. S5.18-Eq. S5.19 ($B_1 = 0.05, 0.03 < \beta_1 < 0.1$, with a very small margin).

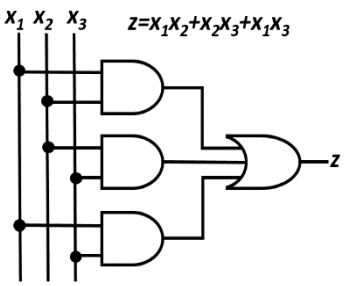
Compared with a one-layer network, a two-layer perceptgene network design has the following advantages:

- **Use less number of parts compared to digital design. In our design we used 7 proteins and 8 promoters, total of 15 parts to implement a 3-input majority function in *E. coli*. In digital design, the same circuit has been implemented in *E. coli* using 10 proteins and 12 promoters (total of 22 parts)** (*6*)**.**
- **Sigmoid functions can have benefits. For example, the state [1,1,0], which displays a "1" logic state in the output, requires that the hidden layer acts as AND logic gate with a very low value of B_1. Simultaneously, the state [1,0,1], which displays a "1" logic state in the output, requires that the hidden layer acts as OR logic gate with a high value of B_1. By contrast, sigmoid functions can solve such conditions very smoothly.**

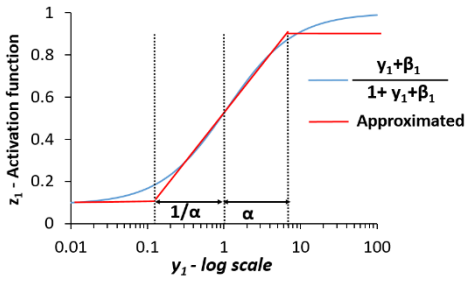
(A)

x_1	x_2	x_3	z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

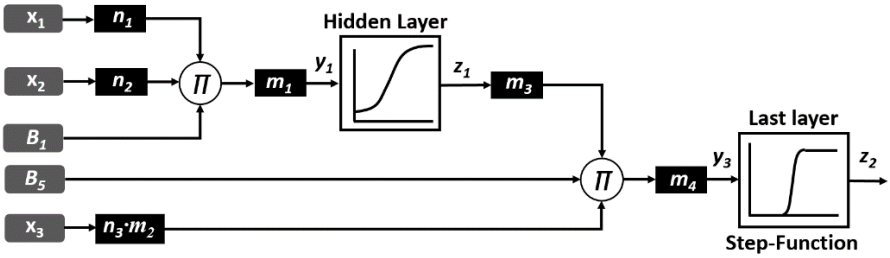
(B)



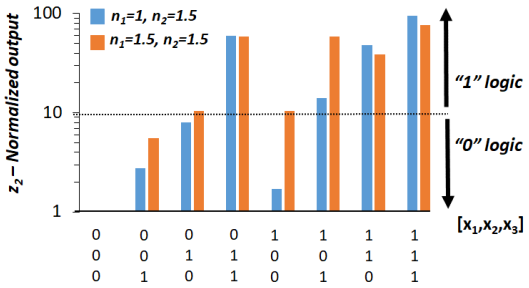
(C)



(D)



(E)



For X_i=[1,16], n_3=0.7, m_1=1.75, m_2=1.5, m_3=1, m_4=1, B_1=4.5×10^{-3}, B_5=1, β_1=0.001

Fig. S5.3. (A) Truth table of 3-input majority function. (B) The implementation of 3-input majority function by digital design. (C) The approximation of Michaelis-Menten model by a linear function in the log-scale. (D) A simplified abstract model based on two-layer perceptgene to implement a 3-input majority function. (E) Simulation results for majority function based on Fig. S5.3D. The error function for the asymmetric weights is 12% and for the symmetric weights is 11%. The other parameters are shown at right side. We used a quadric Error function at the log-domain: $E = (\log(Z_2) - \log(Z_D))^2/2$. Z_D is the expected data and equal to $Z_{DL} = 1$, and $Z_{DH} = 100$.

(A) **(B)**

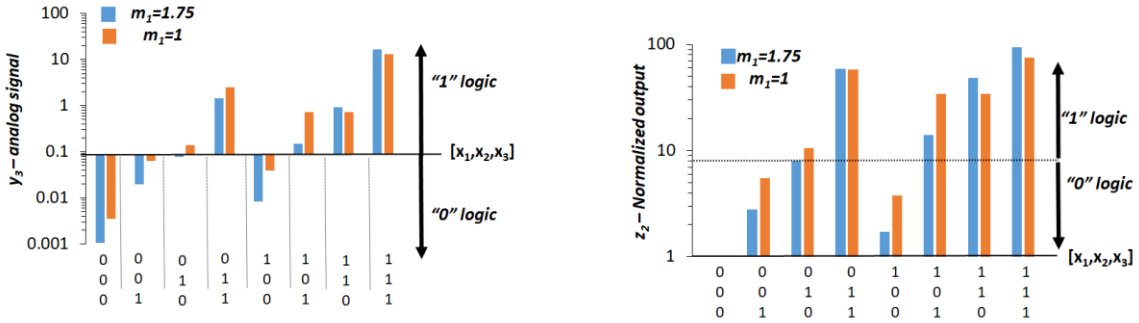


Fig. S5.4. (A) Simulation results for the analog output (y_3) of 3-input majority function based on the two-layer perceptgene network. The threshold was calculated as $10^{-\log(\beta_2)/2}$ (B) Simulation results for the output (z_2) of 3-input majority function based on perceptgene network. The simulation were performed with X_i =[1-16]

5.2. Experimental results of 3-input majority circuit

First, we showed the experimental results of P_{luxM56}-based APF, P_{lacO1}-based ANF loops and $P_{lux/lacO}$ -based combinatorial promoter circuit and fitted the data to power-law and multiplication function ($\log(GFP) = c + n_9 \cdot \log(AHL) + n_{10} \cdot \log(IPTG)$). This circuit (Fig. S5.5) is used as the majority function's first layer (Fig. S5.1).

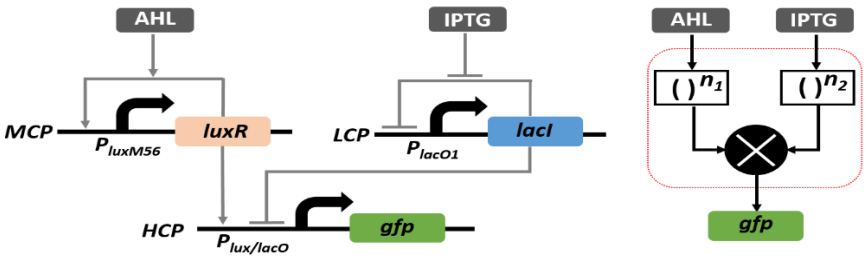


Fig.S5.5. P_{luxM56}-based APF, P_{lacO1}-based ANF loops and $P_{lux/lacO}$-based combinatorial promoter circuit.

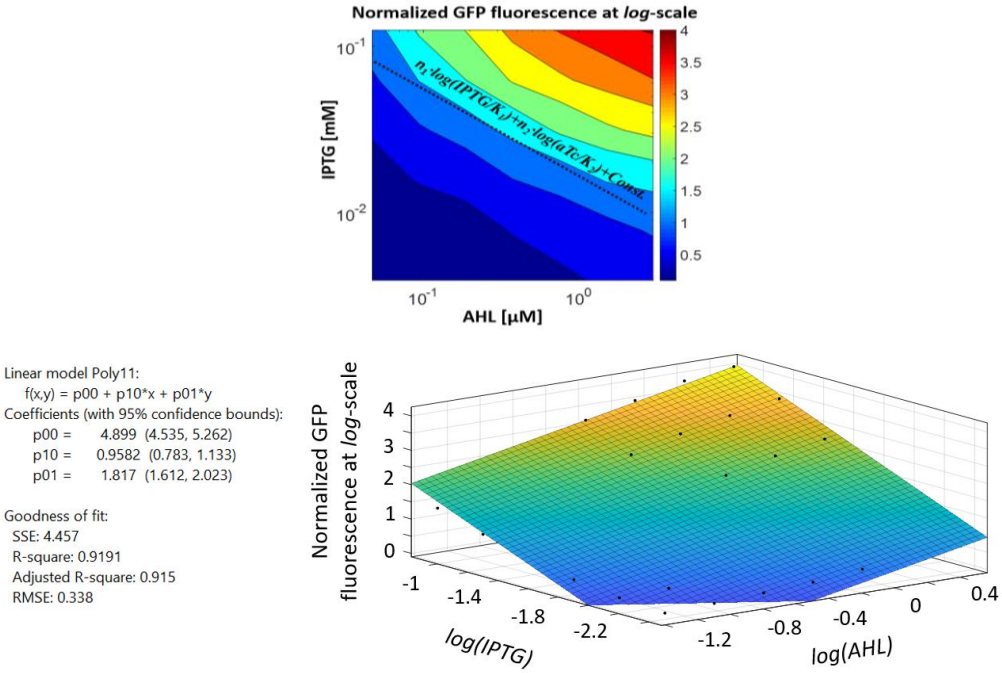


Fig. S5.7. Matlab surface fits the experimental results of APF (P_{luxM56}) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function.

Second, we showed the experimental results of P_{tetO}-based ANF loop and fitted the data to power-law function ($\log(GFP) = c + n_{11} \cdot \log(aTc)$). This circuit (Fig. S5.8) is used to regulate the third inputs aTc of the majority function.

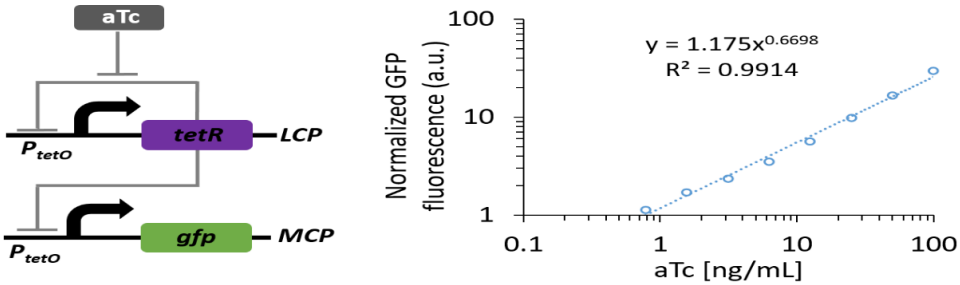


Fig. S5.8. Experimental results of P_{tetO}-based ANF loop circuit, that fits to a power-law function.

Each input (e.g., inducer-transcription factor) has its input dynamic range (*IDR*). However, occasionally when multiple inputs are aggregated at the same computational node, transcription factor binding interference can effectively reduce the dynamic range for neuromorphic computation. Since this disturbance affects the final computation, we characterize every synthetic part separately and combine it with other inputs. For example, we characterized TetR alone and with hybrid promoters TetR/LacI and TetR/LuxR. The *IDR* for TetR was observed to be 2.1 orders of magnitude (Fig. S5.8), remained the same for TetR/LacI (Fig. S2.1, Fig. S2.6), and was reduced to 1.8 orders of magnitude for TetR/LuxR (Fig. S2.15).

The 3-input majority function accepts AHL [0.1875-0.3 μM], IPTG [7.8125-125 μM] and aTc [1.5625-25 ng/mL]. The three inputs have a dynamic input range from 1 to 16. The simulation results of the two-layer perceptgene network are shown in Fig. S5.4B. We used a Michaelis-Menten model as an activation function to calculate the activities of z_1 (the hidden layer) and z_2 (the final layer). We used a consistent set of model parameters as in Fig. S5.4A and Fig. S5.4B except that B_1 is changed to 0.0025 and B_5 is changed to 1. We normalized each measurement by the minimum activity of [0,0,0] state. The experimental results of the 3-input majority circuit (Fig. S5.1 and Fig. S5.9). To keep B_1 very low, we located AraC on a low-copy-number plasmid and added an ssrA degradation tag(*15*) (LAA) to AraC. To keep B_5 very low, a ribosome binding sequence with a low binding affinity (BBa_B0031(*28*)) was used to regulate the T7 RNA polymerase. A low Arabinose concentration was set as 0.03125 mM, and a high Arabinose was set as 0.25 mM. Based on biochemical reactions described in Eq. S5.7-Eq. S5.11, our model could capture well the experimental results.

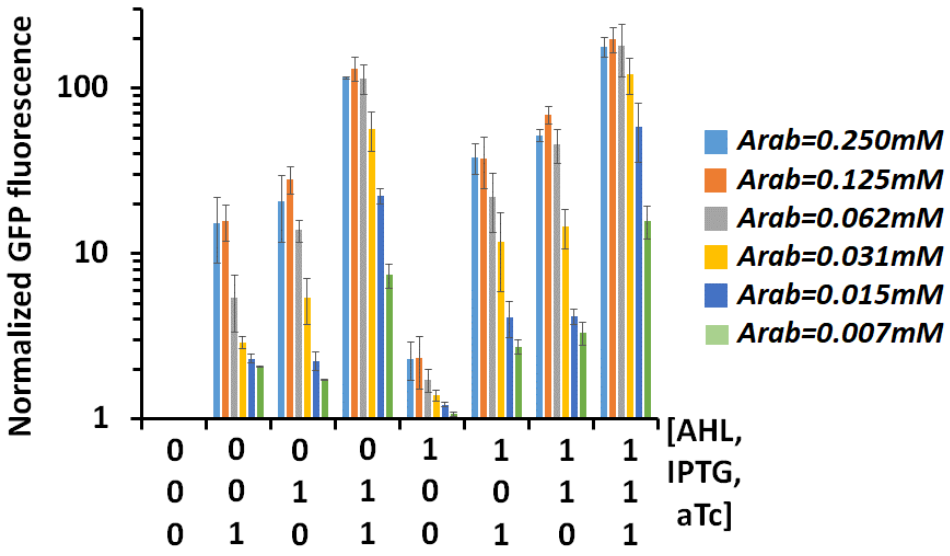
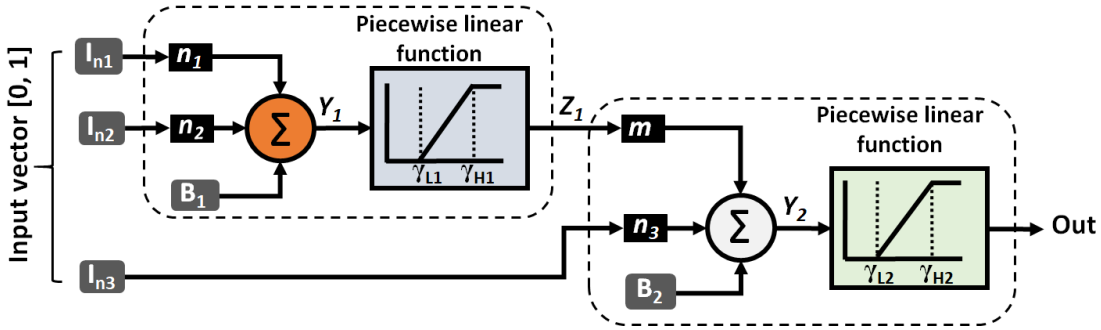


Fig. S5.9. Experimental results of majority circuit for various Arabinose concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM).

Table S5.1: Truth table of the linear-domain perceptron-based 3-input majority function and evaluation of constraints on the design parameters. B_1 and B_1 are the biases of the first layer and second layer perceptgenes, respectively. The three input weights are n_1, n_2, and n_3, while m is the weight of the first layer perceptegene output (Z_1) that serves as an input to the second layer perceptgene. γ_{L1}, γ_{L2} and γ_{H1} and γ_{H2} are the low and high thresholds of the piecewise-linear first and the second activation functions f_{A1} and f_{A2}. The γ_{H2}- γ_{L2}, and γ_{H1}- γ_{L1} are defined as the input dynamic ranges of the activation functions.



I_{n1} I_{n2} I_{n3}	Out	Y_1	Z_1	Y_2	Constraints on Design parameters
0 0 0	0	Design constraints subsumed by 001 case			
0 0 1	0	B_1	0	B_2+n_3	$B_1 < \gamma_{L1}$ $B_2+n_3 < \gamma_{L2}$
0 1 0	0	B_1+n_2	$0 \le f_{A1}(B_1+n_2)<1$	$B_2+m\times f_{A1}(B_1+n_2)$	$B_2+m\times f_{A1}(B_1+n_2) < \gamma_{L2}$
0 1 1	1	B_1+n_2	$0<f_{A1}(B_1+n_2)\le 1$	$B_2+n_3+m\times f_{A1}(B_1+n_2)$	$B_1+n_2 > \gamma_{L1}$ $B_2+n_3+m\times f_{A1}(B_1+n_2) > \gamma_{H2}$
1 0 0	0	B_1+n_1	$0 \le f_{A1}(B_1+n_1)<1$	$B_2+m\times f_{A1}(B_1+n_1)$	$B_2+m\times f_{A1}(B_1+n_1) < \gamma_{L2}$
1 0 1	1	B_1+n_1	$0<f_{A1}(B_1+n_1) \le 1$	$B_2+n_3+m\times f_{A1}(B_1+n_1)$	$B_1+n_1 > \gamma_{L1}$ $B_2+n_3+m\times f_{A1}(B_1+n_1) > \gamma_{H2}$
1 1 0	1	$B_1+n_1+n_2$	1	B_2+m	$B_1+n_1+n_2 > \gamma_{H1}$ $B_2+m > \gamma_{22}$
1 1 1	1	Design constraints subsumed by 110, 101, 011 cases			

Majority analysis for the first activation function:

To satisfy state [001], we require $B_1 < \gamma_{L1}$

To satisfy state [011], we require $B_1+n_2 > \gamma_{L1} \rightarrow B_1 > \gamma_{L1}- n_2$

To satisfy state [101], we require $B_1+n_1 > \gamma_{L1} \rightarrow B_1 > \gamma_{L1}- n_1$

The last three conditions yield to $\gamma_{L1}-\min(n_1,n_2)<B_1 < \gamma_{L1}$

To satisfy state [110], we require $B_1+n_1+n_2 > \gamma_{H1} \rightarrow B_1 > \gamma_{H1}- n_1-n_2$

The last and first conditions yield $\gamma_{H1}-n_1-n_2<B_1 < \gamma_{L1}$

Thus, the input dynamic range of the first activation function should be $\gamma_{H1}-\gamma_{L1}<n_1+n_2$

Majority analysis for the second activation function:

To satisfy state [110], we require $B_2+m > \gamma_{H2} \rightarrow B_2 > \gamma_{H2}-m$

To satisfy state [011], we require $B_2+n_3+m\times f_{A1}(B_1+n_2) > \gamma_{H2} \rightarrow B_2 > \gamma_{H2}-n_3-m\times f_{A1}(B_1+n_2)$

To satisfy state [101], we require $B_2+n_3+m\times f_{A1}(B_1+n_1) > \gamma_{H2} \rightarrow B_2 > \gamma_{H2}-n_3-m\times f_{A1}(B_1+n_1)$

The last three conditions yield to $B_2 > \gamma_{H2}-\min(m,n_3+ m\times f_{A1}(B_1+n_2), n_3+ m\times f_{A1}(B_1+n_1))$

To satisfy state [001], we require $B_2+n_3 < \gamma_{L2} \rightarrow B_2 < \gamma_{L2}- n_3$

To satisfy state [010], we require $B_2+m\times f_{A1}(B_1+n_2) < \gamma_{12} \rightarrow B_2 < \gamma_{12}-m\times f_{A1}(B_1+n_2)$

To satisfy state [101], we require $B_2+m\times f_{A1}(B_1+n_1) < \gamma_{12} \rightarrow B_2 < \gamma_{12}-m\times f_{A1}(B_1+n_1)$

The last three conditions yield to:

$$\gamma_{H2}- \min(m,n_3+ m\times f_{A1}(B_1+n_2), n_3+ m\times f_{A1}(B_1+n_1)) < B_2 < \gamma_{L2}-\max(n_3, m\times f_{A1}(B_1+n_2), m\times f_{A1}(B_1+n_1))$$

Thus, the input dynamic range of the second activation function should be

$$\gamma_{H2}- \gamma_{L2} < \min(m,n_3+ m\times f_{A1}(B_1+n_2), n_3+ m\times f_{A1}(B_1+n_1))-\max(n_3, m\times f_{A1}(B_1+n_2), m\times f_{A1}(B_1+n_1))$$

The analysis of the last condition yields that: $m>n_3$

There are three cases: (for simplicity, we assumed that $n_1<n_2$):

1. When $m<n_3$, the last equation yields that $\gamma_{H2}- \gamma_{L2} <m-n_3<0$
2. When $m=n_3$, the last equation yields that $\gamma_{H2}- \gamma_{L2} <0$
3. When $m>n_3$, the last equation yields that $\gamma_{H2}- \gamma_{L2} <m-n_3$, or $\gamma_{H2}- \gamma_{L2} < m\times(1-f_{A1}(B_1+n_2))$

The condition of [001] and [010] yields for $n_3>\gamma_{H2}- \gamma_{L2}$; thus we obtain $m>n_3>\gamma_{H2}-\gamma_{L2}$.

Table S5.2: Truth table of perceptgene-based majority function and evaluation of the design parameters. D.C. = Don't care.

$m = m_3 \times m_4 = 1 \times 1 = 1$

$n'_1 = n_1 \times m_1 = 1 \times 2 = 2$

$n'_2 = n_2 \times m_1 = 1.5 \times 2 = 3$

$n'_3 = n_3 \times m_2 \times m_4 = 0.7 \times 1.75 \times 1 = 1.25$

$B_1 \times 4.5 \times 10^{-3}, = B_2 \times 1, \gamma_1 = -1, \gamma_2 = 1$

I_{n1} I_{n2} I_{n3}	Out	Y_1	Z_1	Y_2	Constraints on Design parameters
log(1) log(1) log(1)	"1"	B_1 $\log(4.5\times10^{-3})$=-2.3	D.C.	$B_2+m\times Z_1$ log(1)+1×log(Z_1)	$B_2+m\times Z_1 < \gamma_1$ log(1)+1×log(Z_1)< -1 log(Z_1)<-1 Z_1<0.79
log(1) log(1) log(10)	"0"	B_1 $\log(4.5\times10^{-3})$=-2.3	0	B_2+n_3' log(1)+1.25	(1) $B_1 < \gamma_1$ $\log(4.5\times10^{-3})$<1 (2) $B_2+n_3'+m\times\log(Z_1) < \gamma_1$ log(1)+1.25+1×log(Z_1) < -1 log(Z_1)<-2.25 Z_1<0.005
log(1) log(10) log(1)	"0"	B_1+n_2' $\log(4.5\times10^{-3})$+3 -2.3+3=0.7	D.C	$B_2+m\times Z_1$ log(1)+1×log(Z_1)	$B_2+m\times Z_1 < \gamma_1$ log(1)+1×log(Z_1)< -1 log(Z_1)<-1 Z_1<0.79

log(1) log(10) log(10)	"1"	B_1+n_2' $\log(4.5\times10^{-3})+3$ -2.3+1.5=0.7	Interme d	$B_2+n_3'+m$ log(1)+1.25+1	(1) $B_1+n_2' > \gamma_1$ $1\log(4.5\times10^{-3})+3 > -1$ (2) $B_2+n_3'+m\times\log(Z_1) > \gamma_2$ $1\log(1)+1.25+ \log(Z_1) >1$ $Z_1>0.56$
log(10) log(1) log(1)	0	B_1+n_2' $\log(4.5\times10^{-3})+3$ -2.3+3=0.7	D.C.	$B_2+m\times Z_1$ $\log(1)+1\times\log(Z_1)$	$B_2+m\times Z_1< \gamma_1$ $\log(1)+1\times\log(Z_1)< -1$ $\log(Z_1)<-1$ $Z_1<0.79$
log(10) log(1) log(10)	"1"	B_1+n_1' $\log(4.5\times10^{-3})+2$ -2.3+2=-0.3	Interme d	$B_2+n_3'+m$ log(1)+1.25+1	(1) $B_1+n_1' > \gamma_1$ $1\log(0.004)+2 >-1$ (2) $B_2+n_3'+m\times\log(Z_1) > \gamma_2$ $1\log(1)+1.25+\log(Z_1)>1$ $Z_1>0.56$
log(10) log(10) log(1)	"1"	$B_1+n_1'+n_2'$ $\log(4.5\times10^{-3})+2+3$ -2.3+5=2.7	1	B_2+m log(1)+1	(1) $B_1+n_1'+n_2' > \gamma_2$ $\log(4.5\times10^{-3})+2+3>1$ (2) $B_2+m > \gamma_2$ $\log(1)+1>1$
log(10) log(10) log(10)	"1"	$B_1+n_1'+n_2'$ $\log(4.5\times10^{-3})+2+3$ -2.3+5=2.7	1	$B_2+n_3'+m$ log(1)+1.25+1	(1) $B_1+n_1'+n_2' > \gamma_2$ $\log(4.5\times10^{-3})+2+3>1$ (2) $B_2+n_3'+m > \gamma_2$ $\log(1)+1.25+1>1$

Table S5.3 List of parameters used in this section

Symbol	Description
y_i	The power law and multiplication signal (analog signal)
Y_{mi}	Fitting parameter that has concentration units
K_{mi}	Dissociation constant of binding x_i to Y_i
n_i, m_i	Hill coefficient
z_i	The expression level of the output protein proportional to the promoter activity
z_{mi}	The maximum expression level of the output protein
β_i	The basal level of the promoter
K_i	Dissociation constant of complex binding to promoter
a	Dissociation constant of binding Arabinose–AraC complex to P_{BAD}
b	Fitting parameter to the effective dissosciation constant of binding Arabinose–AraC complex to P_{BAD}
B_i	Bias
I_{ni}	Normalized input

Table S5.4 List of abbreviations used in this section

Symbol	Description
ANF	Auto-negative feedback
APF	Auto-positive feedback
IDR	Input Dynamic range
LTU	linear threshold unit
AHL	Free N-(β-Ketocaproyl)-L-homoserine Lactone $3OC_6HSL$ concentration
IPTG	Free Isopropyl1-β-D-1-thiogalactopyranside concentration
aTc	Free anhydrotetracycline
$AraC$	$AraC$ protein
LAA	ssrA degradation tag
$T7_{RNA}$	T7 RNA Polymerase
tRNA $supD$	Amber suppressor tRNA
P_{luxM56}	Mutated LuxR promoter is activated by the $LuxR$ when it is induced by AHL
$P_{lux/lacO}$	Combinatorial promoter
P_{T7}	T7 promoter
P_{TetO}	TetR promoter is activated by the $TetR - aTc$
$P_{luxM56/tetO}$	Combinatorial promoter
$P_{luxM56/lacO1}$	Combinatorial promoter
$P_{lux/tetO}$	Combinatorial promoter

6. Gradient descent and backpropagation algorithms in living cells

The perceptron weights can be adjusted in small steps through iterations, following a first-order optimization algorithm known as a gradient descent (*31*). This process converges to the global minimum of the gradient descent of the mean square error metric function or cost function (C). Typically, this ensures that a high resolution of weight adjustments in tradeoff with training time and number of samples (*32*). Likewise, we developed a perceptgene-based rule that minimizes the output error, following log-linear domain's gradient descent. We defined a logarithmically quadratic cost function for each state or sample i as:

$$C_i = \frac{1}{2}(\log(Z_{Di}) - \log(Z_i))^2 \quad \text{(S6.1)}$$

Z_{Di} is the desired output of every state, and Z_i is the actual network output for every state. The cost function of network is the average of cost functions over individual samples:

$$< C >= \frac{1}{N}\sum_{i=1}^{N} C_i \quad \text{(S6.2)}$$

N is the number of samples and is also called the batch size. Substituting Eq. S6.1 into Eq. S6.2, we get:

$$< C >= \frac{1}{2\cdot N}\sum_{i}^{N}(\log(Z_{Di}/Z_i))^2 \quad \text{(S6.3)}$$

We call C the logarithmically (average) *quadratic* cost function of the network, and it is a function of the weights and biases. We can see that $< C >$ function is non-negative, since every term in the sum is non-negative. Following the gradient descent, at every iteration, the network output is adjusted toward the desired value and accordingly, the average-cost function decreases (the cost function becomes small when the output is approximately equal to the desired value for all the samples i ($< C >\approx 0\ when\ z_i \approx z_{Di}$). Fig. S6.1 shows the average-cost function of a perceptgene network for two inputs and one output. In this case, there are four input states. We attempted to find an algorithm that minimizes the cost function on average with respect to the weights ($\partial < C >/\partial m_i$ or $\partial < C >/\partial n_i$ (m_i and n_i are network weights). In this work, we optimized the m_1 of the majority circuit, which consists of two layers (Fig. S5.1 and Fig. S5.2). Therefore, we used a chain-rule in addition to gradient-descent to update the m_1.In particular at every iteration, we calculated the average error or cost at the output and distributed it back through the network layers which is essentially backpropagation algorithm (*7*):

$$\frac{\partial <C>}{\partial m_1} = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial C_i}{\partial m_1} \quad \text{(S6.4)}$$

We can write the set of equations that describe the majority circuit as: (we assumed that, the Basal levels are much lower than 1 ($\beta_{1,2} << 1$):

$$y_1 = \left(B_1 \cdot \left(\frac{AHL}{K_{m1}}\right)^{n_1} \cdot \left(\frac{IPTG}{K_{m2}}\right)^{n_2}\right)^{m_1} \quad \text{(S6.5)}$$

$$z_1 \cong \frac{y_1}{1+y_1} + \beta_1 \quad \text{(S6.6)}$$

$$y_2 = \left(B_2 \cdot \left(\frac{x_3}{K_{m3}}\right)^{n_3}\right)^{m_2} \quad \text{(S6.7)}$$

$$y_3 = (B_4 \cdot y_2 \cdot (B_3 \cdot z_1)^{m_3})^{m_4} \quad \text{(S6.8)}$$

$$z_2 \cong \frac{y_3}{1+y_3} + \beta_2 \quad \text{(S6.9)}$$

Where $B_4 \equiv \frac{Y_{m3}}{K_5}\cdot\frac{K_3}{K_4}$, $B_3 \equiv \frac{Z_{m1}}{K_3}$, $B_2 \equiv \frac{Y_{m2}}{K_2}$, $B_1 \equiv \frac{Y_{m1}}{a}\cdot {m_1}^b$

Using a chain rule, for every state we can get:

$$\frac{\partial C_i}{\partial m_1} = \frac{\partial C_i}{\partial z_2}\cdot\frac{\partial z_2}{\partial y_3}\cdot\frac{\partial y_3}{\partial z_1}\cdot\frac{\partial z_1}{\partial y_1}\cdot\frac{\partial y_1}{\partial m_1} \quad \text{(S6.10)}$$

$$\frac{\partial C_i}{\partial z_2} = \log(\frac{z_D}{z_2}) \cdot \frac{1}{z_2} \quad \text{(S6.11)}$$

$$\frac{\partial z_2}{\partial y_3} \cong (z_2 - \beta_2) \cdot (1 - z_2) \cdot \frac{1}{y_3} \quad \text{(S6.12)}$$

$$\frac{\partial y_3}{\partial z_1} = y_3 \cdot \frac{m_3 \cdot m_4}{z_1} \quad \text{(S6.13)}$$

$$\frac{\partial z_1}{\partial y_1} \cong (z_1 - \beta_1) \cdot (1 - z_1) \cdot \frac{1}{y_1} \quad \text{(S6.14)}$$

$$\frac{\partial y_1}{\partial m_1} = y_1 \cdot \left(\log\left(\frac{Y_{m1}}{a} \cdot {x_1}^{n_1} \cdot {x_2}^{n_2}\right) + b \cdot (\log(m_1) + 1)\right) \quad \text{(S6.15)}$$

The partial derivative of the average-cost function of every sample or input state with respect to m_1 is:

$$\frac{\partial C_i}{\partial m_1} = m_3 \cdot m_4 \cdot \log(\frac{z_2}{z_D}) \cdot \left(\frac{z_2 - \beta_2}{z_2}\right) \cdot (1 - z_2) \cdot \left(\frac{z_1 - \beta_1}{z_1}\right) \cdot (1 - z_1) \cdot \left(\log\left(\frac{Y_{m1}}{a} \cdot {x_1}^{n_1} \cdot {x_2}^{n_2}\right) + b \cdot (\log(m_1) + 1)\right) \quad \text{(S6.16.1)}$$

We normalized the partial derivative by $\log(\frac{z_{max2}}{z_{min2}})$ term, because the output dynamic range depends on the Arabinose level:

$$\frac{\partial C_i}{\partial m_1} = m_3 \cdot m_4 \cdot \frac{\log(\frac{z_2}{z_D})}{\log(\frac{z_{max2}}{z_{min2}})} \cdot \left(\frac{z_2 - \beta_2}{z_2}\right) \cdot (1 - z_2) \cdot \left(\frac{z_1 - \beta_1}{z_1}\right) \cdot (1 - z_1) \cdot \left(\log\left(\frac{Y_{m1}}{a} \cdot {x_1}^{n_1} \cdot {x_2}^{n_2}\right) + b \cdot (\log(m_1) + 1)\right) \quad \text{(S6.16.2)}$$

The weights adjustment in m_1 to minimize the average-cost function is:

$$\Delta m_1 = -\zeta \cdot \frac{1}{N} \sum_{i=1}^{N} \frac{\partial C_i}{\partial m_1} \quad \text{(S6.17)}$$

Where ζ is a scalar and it determines the rate of m_1 being updated (also known as learning/training rate). The direction of the update is opposite to the partial derivative of a cost function, which guarantees that the weights adjustment is in the direction of a minimum, not a maximum, of the average-cost function. This technique calculates the average error at the output and distributes it back through the network layers. Therefore, it is also called "backward propagation of errors".

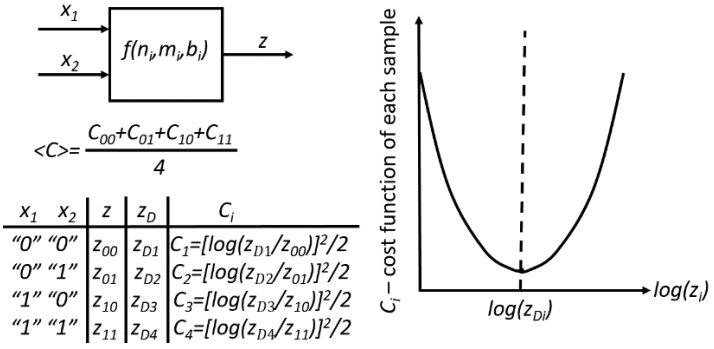


Fig. S6.1. Logarithmically *quadratic* cost function of the perceptgene-based network.

Eq. S6.16 and Eq. S6.17 have two important implications:

- The cost function of the network can be written as an average over cost functions for individual samples or input states.

- The cost function can be written as a function of the outputs of each layer independent on the analog signals of the network.

6.1. Calculation the experimental average (normalized) cost function

First, we calculated the average-cost function of the majority circuit based on Eq. S6.1-Eq. S6.3 using experimental results and compared the outcome of Eq. S6.16-Eq. S6.17. Subsequently, we changed m_1 by inducing the circuit with varying Arabinose concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM). The relation between arabinose concentration and the weigh m_1 , was calculated based on the experiment in Fig. S2.9A, and is shown in Fig. S6.2A. Since the maximum level is measured strongly depends on the arabinose concertation (Fig. S6.2B**)**, we defined a normalized average cost function. In this way, we can compare the average cost functions for the different Arabinose concentrations compatible. The process that describing the calculation of average normalized cost function is given by:

1. Given and eights measured outputs: $Z_{xxx} = \{Z_{000}, Z_{001}, Z_{010}, Z_{011}, Z_{100}, Z_{101}, Z_{110}, Z_{111}\}$. Here each element of Z_{xxx} is average of three experiments.
2. Find the minimum between $Z_{min} = minimum\ \{Z_{000}, Z_{001}, Z_{010}, Z_{011}, Z_{100}, Z_{101}, Z_{110}, Z_{111}\}$.
3. Normalized the measured outputs by Z_{min} $Z_{nxxx} = \{\frac{Z_{000}}{Z_{min}}, \frac{Z_{001}}{Z_{min}}, \frac{Z_{010}}{Z_{min}}, \frac{Z_{011}}{Z_{min}}, \frac{Z_{100}}{Z_{min}}, \frac{Z_{101}}{Z_{min}}, \frac{Z_{110}}{Z_{min}}, \frac{Z_{111}}{Z_{min}}\}$.
4. The lowest desired value $Z_{nmin} = minimum\{Z_{nxxx}\} = 1$.
5. The highest desired value $Z_{nmax} = \text{maximum}\{Z_{nxxx}\}$.
6. Define normalized signal in the linear domain as $\frac{\log\left(\frac{Z_{xxx}}{Z_{min}}\right)}{Z_{nmax}}$.
7. $< C_n >= \frac{1}{2 \cdot N} \sum_i^N \left(Z_{Di} - \frac{\log\left(\frac{Z_{xxx}}{Z_{min}}\right)}{Z_{nmax}} \right)^2$, $Z_{Di} = \{0,1\}$. (S6.18)

Then, we simulated the process of updating m_1 by applying the backpropagation algorithm (Eq. S6.16-Eq. S6.17), as shown in Fig. S6.3. We used desired values (z_{Di}) that are similar to experimental results: "0" =1 a.u., "1" = maximum normalized GFP (z_{D111}) for each Arabinose concentration. Therefore, we fitted the maximum normalized GFP (z_{D111}) to polynomial function as shown in Fig. S6.2B**.** Hypothetically the relationship between maximum normalized GFP and Arabinose affects the partial derivative of the cost function ($\partial C / \partial m_1$).

The cost function in Fig. 3 in the main text are based on the experimental results in Fig. S5.9 Eq. S6.18.

(A) **(B)**

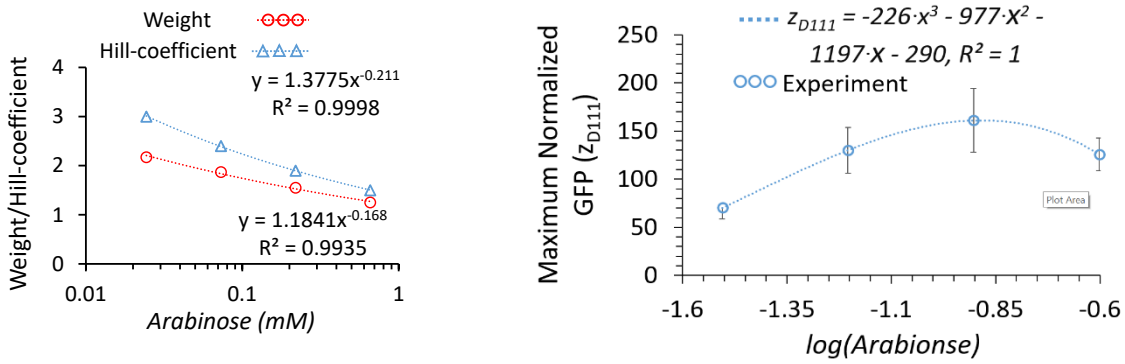


Fig. S6.2. (A) The relation between Arabinose concentration and the weigh m_1, and which was calculated with new Arabinose values based on the experiment Fig. S2.9A**.** (B) The maximum normalized signal achieved for each Arabinose concentration, was used as the "1" logic desired value for calculating the cost function.

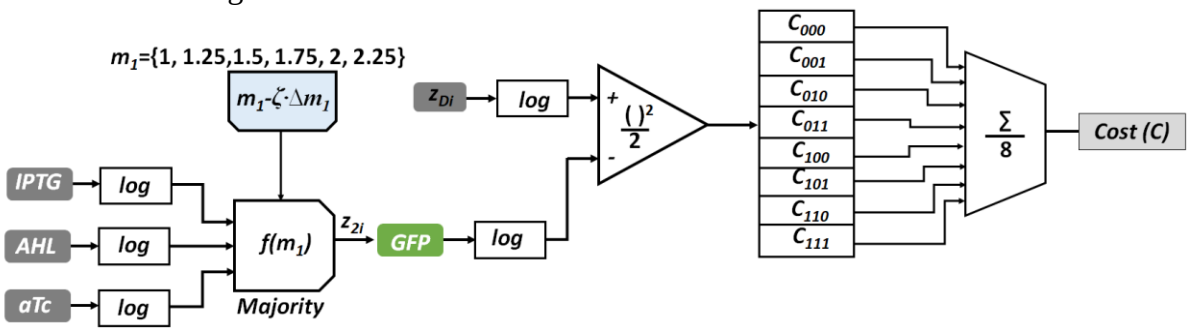


Fig. S6.3. An algorithm for estimating and simulating cost function.

6.2. Backpropagation algorithm for two weights

Our next step is to use the backpropagation algorithm for programming two weights (m_1 for P_{BAD}/AraC and n_1 for P_{lux}/LuxR) within the majority function. To control the P_{lux}/LuxR weight, we introduce random mutations to the operator sequence of a LuxRtranscription factor. Fig. S8.8 describes seven modulations of transcription factor LuxR's DNA binding affinity via Lux operator sequence changes. Here, our backpropagation described in the following equations shows that it was enough to use less than four mutations TCTA, GTTG, GAGC and TGGG (PluxM56) for the APF loop of the first layer (Fig. S6.4) to reach a majority function.

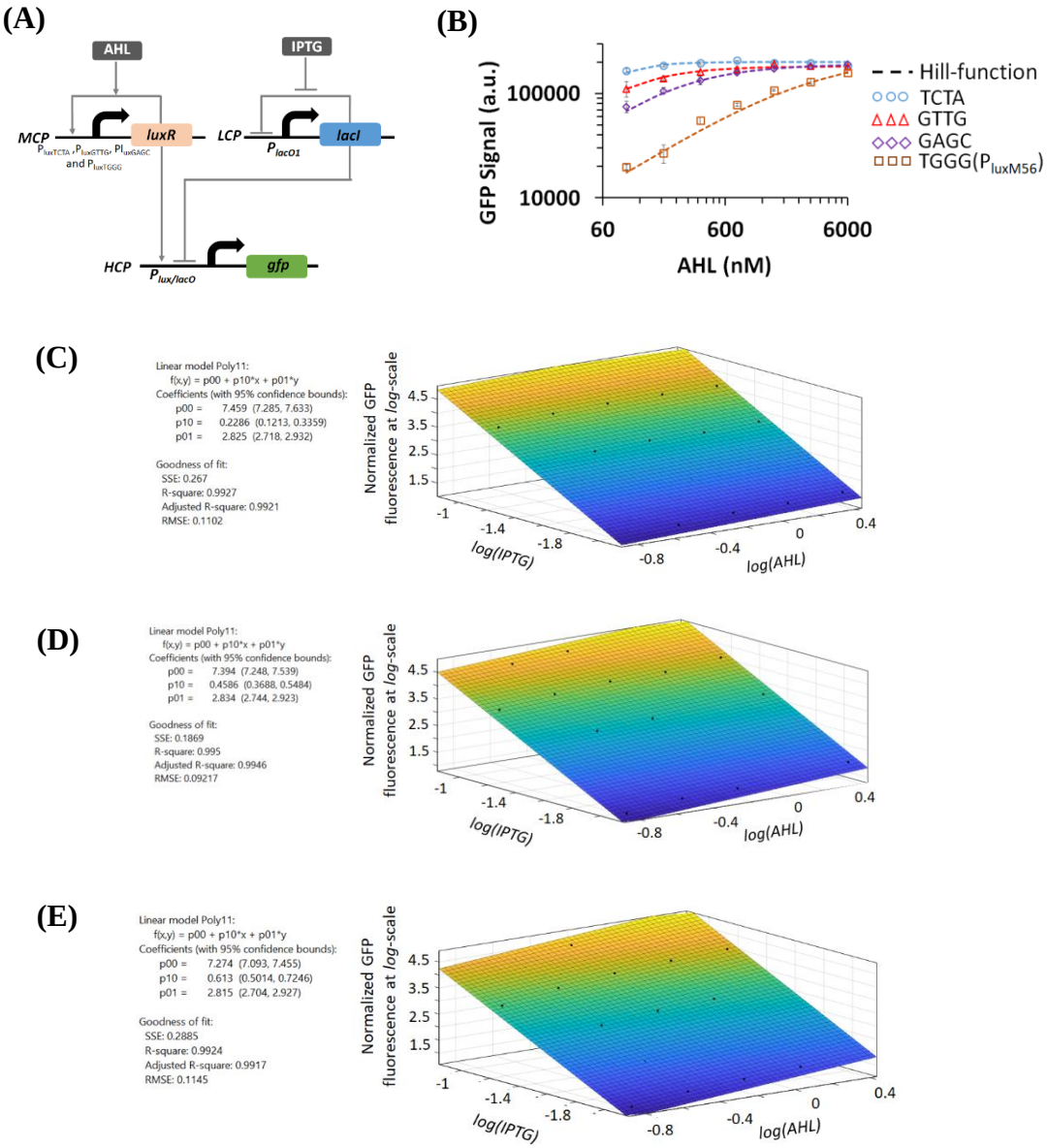
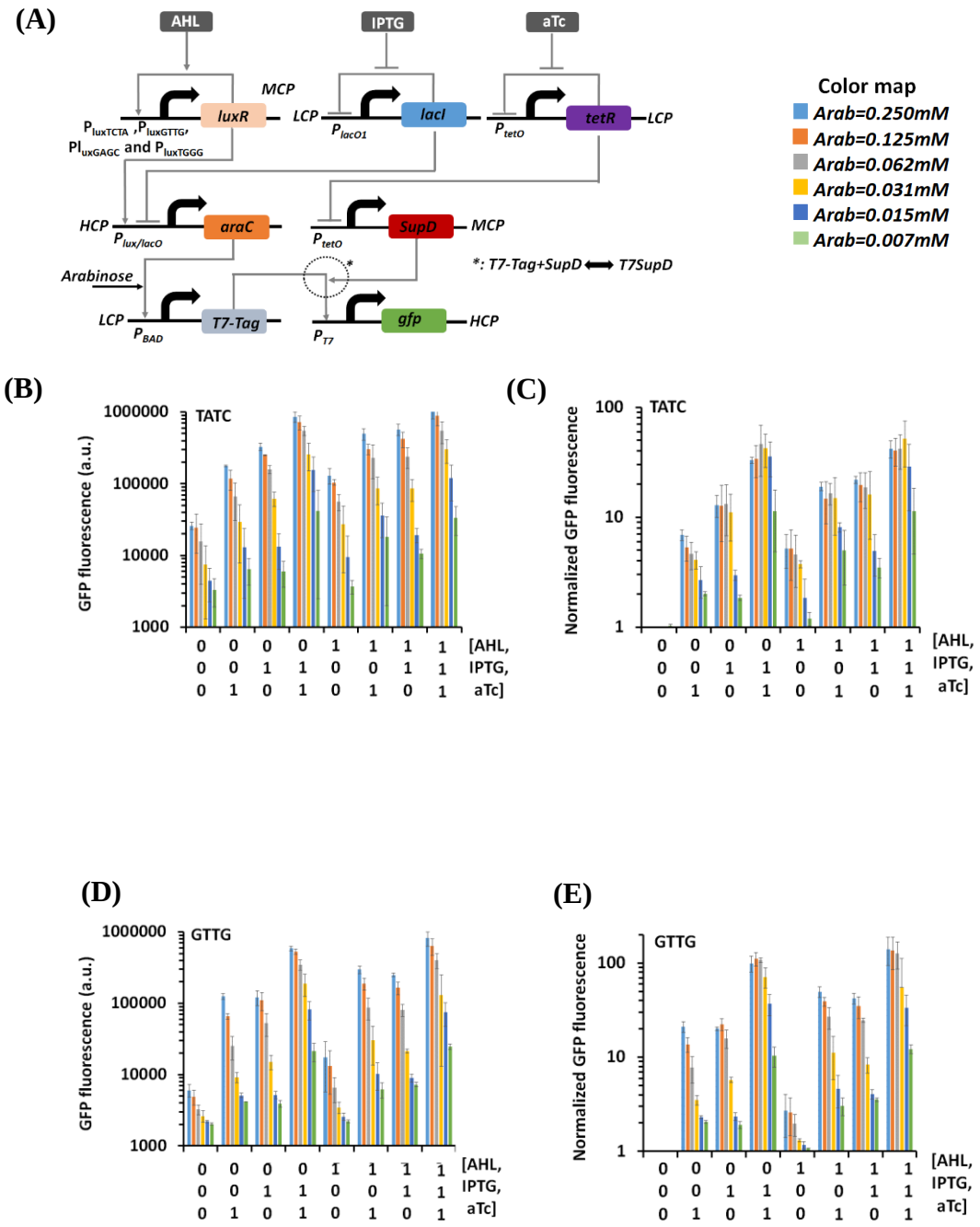


Fig. S6.4. (A) $P_{luxNNNN}$ -based APF, P_{lacO1} -based ANF loops, and $P_{lux/lacO}$ -based combinatorial promoter circuit, similar to circuit in Fig. S5.5. (B) AHL − GFP transfer function for four different mutations (TCTA, GTTG, GAGC, and TGGG (P_{luxM56})) within P_{lux} promoter regulating LuxR by APF loop. IPTG = 0.125mM. (C) Matlab surface fits the experimental results of APF ($P_{luxTCTA}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function,the weight of AHL is 0.228, and the weight of IPTG is 2.8. (D) Matlab surface fits the experimental results of APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function, the weight of AHL is 0.45, and the weight of IPTG is 2.8. (E) Matlab surface fits the experimental results of APF ($P_{luxGAGC}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function, the weight of AHL is 0.61, and the weight of IPTG is

2.8. The results of P_{luxM56} and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) are presented in Fig. S5.7.

Next, we modified the three-input perceptgene network from Fig. S5.1 in the APF loop by changing the first four nucleutides in the P_{lux} promoter sequence as followed; TCTA ,GTTG, GAGC (Fig. S6.5). Then, we measured the GFP signal for all the four different circuits including P_{luxM56} across (Fig. S5.1) the eight states of the inputs [AHL, IPTG, aTc]=[0,0,0], [0,0,1], [0,1,0], [0,1,1], [1,0,0], [1,0,1], [1,1,0], [1,1,1].The measured signals are presented in their absolute values (without normalization) and normalized values (Fig. S6.5).



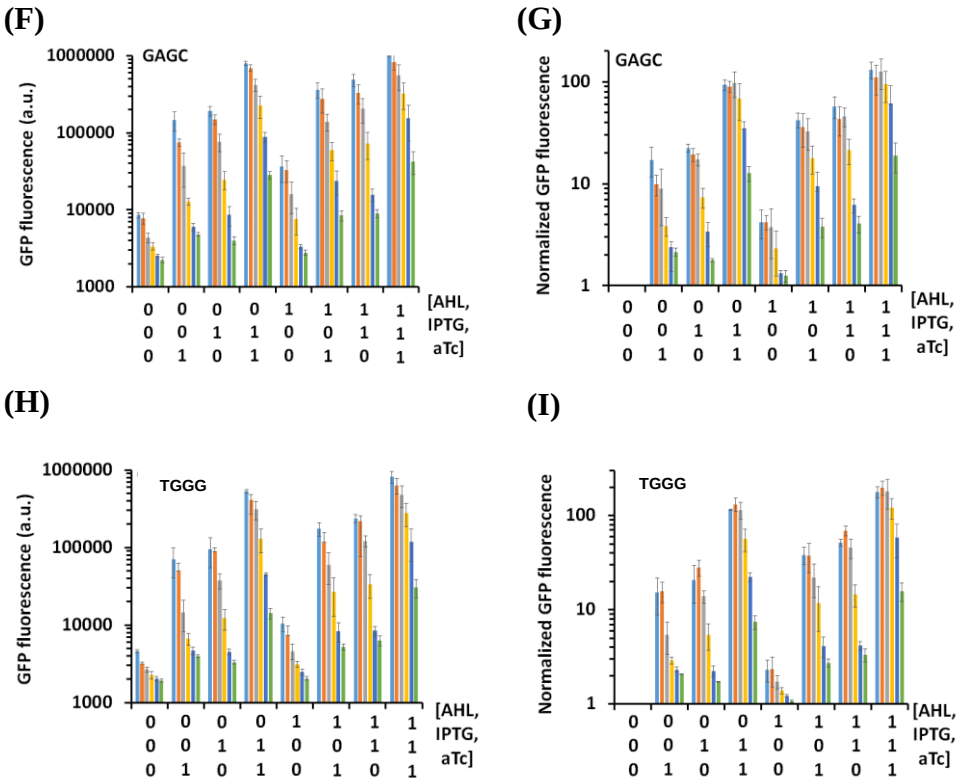


Fig. S6.5. (A) Three-input perceptgene network accepts three analog inputs (AHL, IPTG and aTc) including four P_{lux} mutations (TCTA, GTTG, GAGC, TGGG (P_{luxM56})) within the APF in similar to Fig. S5.1. (B) and (C) Experimental results of 3-input perceptgene network with TCTA for various Arabinose concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM). The data is presented by absolute signals as measured by the Flow analyzer (B) and normalized signals (C). (D) and (E) Experimental results of 3-input circuit with GTTG for various Arabinose concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM). The data is presented by absolute signals as measured by the Flow analyzer (D) and normalized signals (E). (F) and (G) Experimental results of 3-input circuit with GAGC for various Arabinose concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM). The data is presented by absolute signals as measured by the Flow analyzer (F) and normalized signals (G). (H) and (I) Experimental results of 3-input circuit with TGGG for various Arabinose concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM). The data is presented by absolute signals as measured by the Flow analyzer (H) and normalized signals (I).

The $\mathrm{AHL}-\mathrm{GFP}$ transfer functions for the four P_{lux} mutations show that each modification has its slope (i.e., weight) and bias. Thus, there is a disturb between weight programming and bias levels. The AHL-GFP transfer functions for the four mutations can be written as:

$$GFP \propto B \cdot {n_1}^{-d} \cdot {x_1}^{n_1} \tag{S6.19}$$

Where $= 3.7$. Fig. S6.6 shows the dependency of the bias on the weight based on the data from Fig. S6.4B. The next step is to calculate the derivative of Eq. S6.19:

$$\frac{dGFP}{dn_1} \propto B \cdot {n_1}^{-d} \cdot {x_1}^{n_1} \cdot \left(log(x_1) - \frac{d}{n_1}\right) \quad \text{(S6.20)}$$

Eq. S.620 shows that update in the weight also changes the bias. Based on that, we modified Eq. S6.5 to $B_1 = \frac{Y_{m1}}{a} \cdot {m_1}^{b} \cdot {n_1}^{-d}$. The partial derivative of average-cost function of every sample or input state with respect to n_1 is:

$$\frac{\partial <C>}{\partial n_1} = \frac{1}{N}\sum_{i=1}^{N} \frac{\partial C_i}{\partial n_1}$$

⇨

$$\frac{\partial C_i}{\partial n_1} = -m_3 \cdot m_4 \cdot \frac{\log(\frac{z_2}{z_D})}{\log(\frac{z_{max2}}{z_{min2}})} \cdot \left(\frac{z_2-\beta_2}{z_2}\right) \cdot (1-z_2) \cdot \left(\frac{z_1-\beta_1}{z_1}\right) \cdot (1-z_1) \cdot m_1 \cdot \left(log(x_1) - \frac{d}{n_1}\right) \quad \text{(S6.21)}$$

In Eq. S6.21, we normalized the error $\log(\frac{z_{max2}}{z_{min2}})$, by output dynamic range $\log(\frac{z_{max2}}{z_{min2}})$, and this is because the output dynamic range depends on the Arabinose level. The partial derivative of average-cost function of every sample or input state with respect to m_1 is similar to Eq. S.6.18 with a one modification; we included the $-d \cdot \log(n_1)$:

$$\frac{\partial C_i}{\partial m_1} = -m_3 \cdot m_4 \cdot \frac{\log(\frac{z_2}{z_D})}{\log(\frac{z_{max2}}{z_{min2}})} \cdot \left(\frac{z_2-\beta_2}{z_2}\right) \cdot (1-z_2) \cdot \left(\frac{z_1-\beta_1}{z_1}\right) \cdot (1-z_1) \cdot \Big(b \cdot \log(m_1) + n_2 \cdot \log(x_2) + \log\left(\frac{Y_{m1}}{a}\right) + n_1 \cdot (-d \cdot \log(n_1) + \log(x_1)) + b\Big) \quad \text{(S6.22)}$$

To calculate the update weights, based on Eq. S6.21 ad Eq. S6.22, we measured the output signals of the first layer (Z_1) as shown in Fig. S6.7 and the output signals of the **Three-input perceptgene network** (Z_2) as shown in Fig. S6.5. We also measured the basal level of each activation function (β_1 and β_2) by measuring Z_1 and Z_2 when no inducers (AHL, IPTG and aTc) were added to the networks (Table S6.1). The parameters (b=*0.37*, $\frac{Y_{m1}}{a} = 0.0001$) are estimated from the data shown in Fig. S2.9, $d = 3.7$ is estimated from Fig. S.6.6, n_1 and n_2 are estimated from Fig. S6.4 and Fig. S5.7 (TCTA: 0.23, 2.8, GTTG: 0,45, 2.8, GAGC: 0.65, 2.8, TGGG: 0.95, 1.85), $m_3 = m_4 = 1$. The first layer is the perceptgene of two inputs (AHL and IPTG) with four P_{lux} mutations of the APF loop; TCTA, GTTG, GAGC and TGGG (P_{luxM56}) (Fig. S6.7A). The experimental results of backpropagation algorithms based on Eq. S6.21 and Eq. S6.22 are presented in Table S6.2. Based on these results, we built an optimized pathway to reach the best majority results. We first update m_1 and then n_1, which reach a minimum cost function. We marked the optimized pathway in Table S6.2.

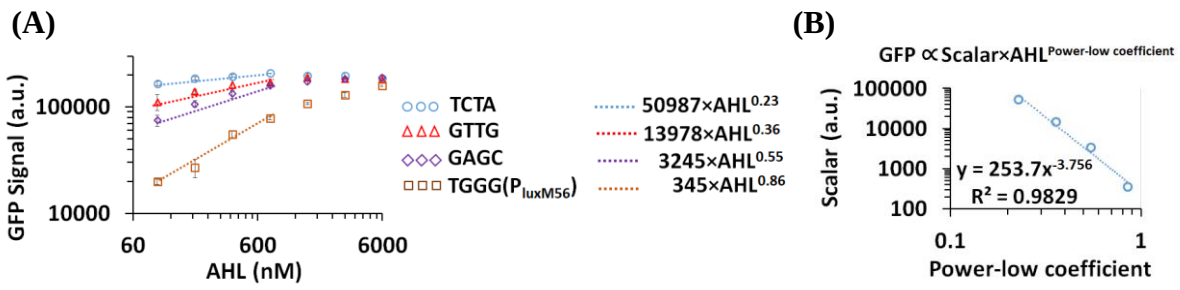
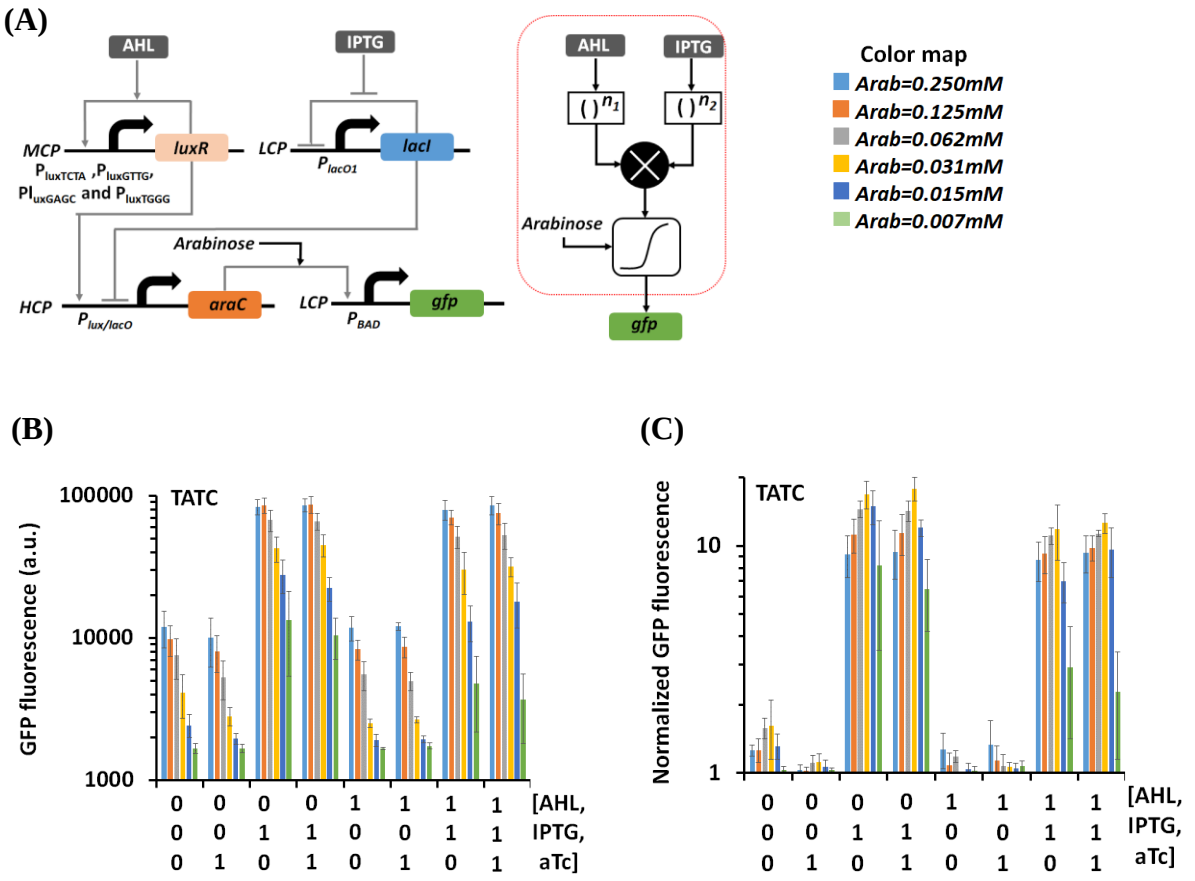


Fig. S6.6. (A) Experimental results of $P_{luxNNNN}$-based APF, P_{lacO1}-based ANF loops, and $P_{lux/lacO}$-based combinatorial promoter circuit fit power-law function for $IPTG = 0.125\,mM$. (B) The scalar number in the power-law fitting from (A), which is proportional to bias, is also a function of the power-law coefficient. The power-law coefficient is proportional to weight.



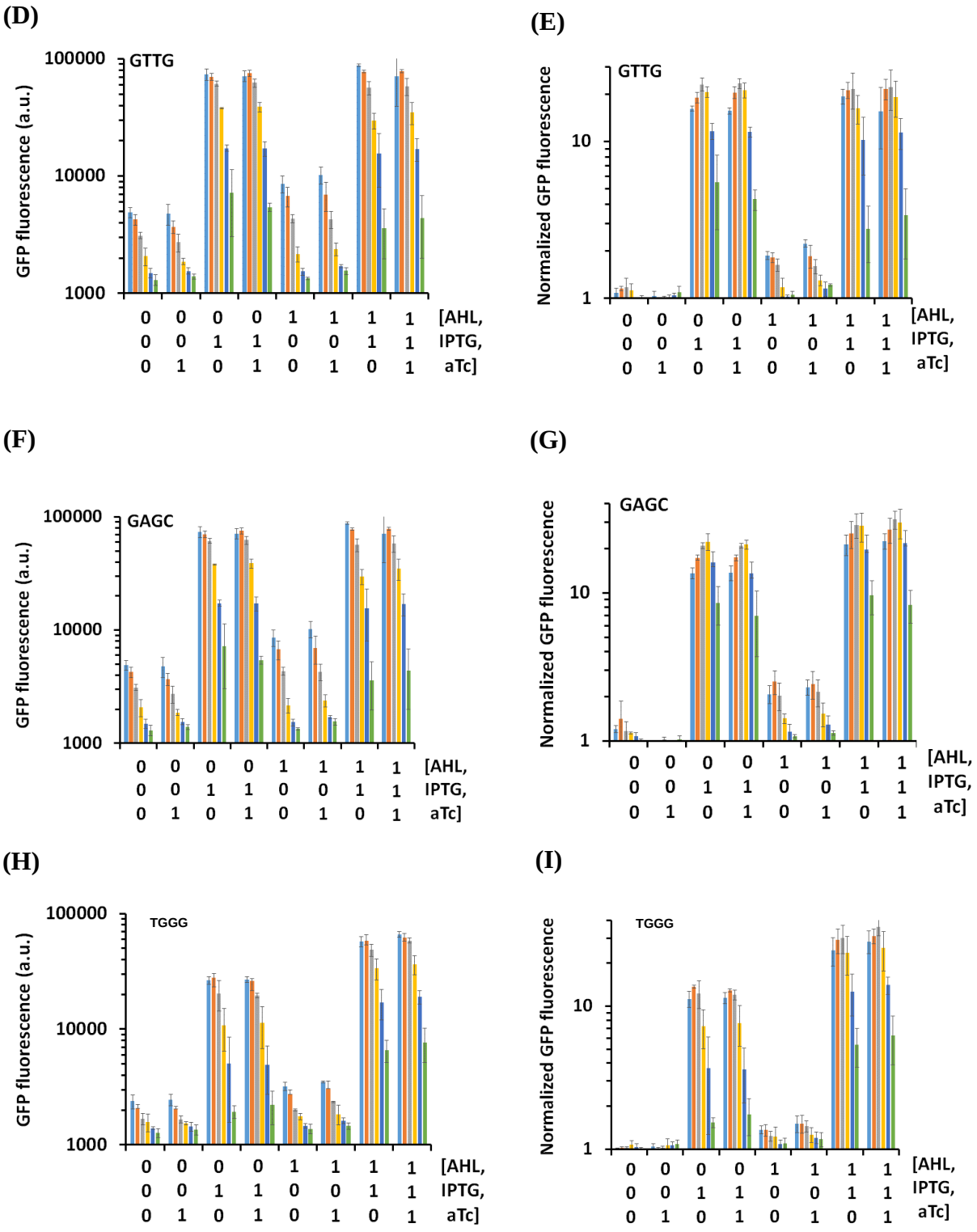


Fig. S6.7. (A) The first perceptgene layer from the three-input perceptgene network (Fig. S6.5). The layer accepts two analog inputs (AHL, IPTG) including four P_{lux} mutations (TCTA, GTTG, GAGC, TGGG (P_{luxM56})) within the APF. (B) and (C) Experimental results of 2-input circuit with TCTA for various Arabinose concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM). The data is presented by absolute signals as measured by the Flow analyzer (B) and normalized signals (C). (D) and (E) Experimental results of 2-input circuit with GTTG for various Arabinose

concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM). The data is presented by absolute signals as measured by the Flow analyzer (D) and normalized signals (E).
(F) and (G) Experimental results of 2-input circuit with GAGC for various Arabinose concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM). The data is presented by absolute signals as measured by the Flow analyzer (F) and normalized signals (G). (H) and (I) Experimental results of 2-input circuit with TGGG for various Arabinose concentrations (0.250, 0.125, 0.062, 0.031, 0.015, 0.007 mM). The data is presented by absolute signals as measured by the Flow analyzer (H) and normalized signals (I).

			Arabinose(mM)					
			0.2500	0.1250	0.0625	0.0313	0.0156	0.0078
TCTA	β_2	Average	1573	1739	1506	1651	1484	1603
		STDEV	97	78	37	49	76	47
	β_1	Average	2403	2003	1722	1453	1629	2512
		STDEV	33	430	97	159	311	1490
GTTG	β_2	Average	1469	1435	1491	1488	1530	1499
		STDEV	1	53	75	17	6	3
	β_1	Average	2185	1886	1643	1645	1378	1314
		STDEV	122	161	12	40	0	131
GAGC	β_2	Average	1443	1453	1448	1501	1504	1581
		STDEV	108	174	95	70	114	134
	β_1	Average	2317	2105	1743	1507	1505	1362
		STDEV	24	300	169	13	66	1
TGGG	β_2	Average	1466	1444	1912	1538	1522	1563
		STDEV	26	68	641	35	129	141
	β_1	Average	1609	1481	1184	1457	1264	1187
		STDEV	26	68	641	35	129	141

Table S6.1. Measured values of β_1 (basal level of the first layer, Fig. S6.7A) and β_2 (basal level of the second layer, Fig. S6.5A) for the four mutations. The measurements were performed when no inducers (AHL, IPTG, aTc) were added.

TCTA mutation, n_1 =0.23 weight					
Arabinose (mM)	m_1 weight	Experimental cost function	Digital cost function	$\frac{\partial < C >}{\partial m_1}$	$\frac{\partial < C >}{\partial n_1}$
(1) 0.2500	1	0.061	0.1250	-0.36	-7.34
(2) 0.1250	1.25	0.057	0.0625	-0.31	-6.91
0.0625	1.5	0.051	0.0625	-0.31	-7.03
0.0312	1.75	0.051	0.0625	-0.27	-3.21
0.0156	2	0.047	0.0625	-0.68	6.13
0.0078	2.25	0.033	0.0625	-0.50	5.48

GTTG mutation, n_1 =0.45 weight					
Arabinose (mM)	m_1 weight	Experimental cost function	Digital cost function	$\frac{\partial < C >}{\partial m_1}$	$\frac{\partial < C >}{\partial n_1}$
0.2500	1	0.056	0.1250	-0.50	-2.90
(3) 0.1250	1.25	0.054	0.1250	-0.36	-2.65
(4) 0.0625	1.5	0.046	0.0625	-0.20	-0.74
0.0312	1.75	0.044	0.0625	-0.11	1.70
0.0156	2	0.051	0.1250	-0.02	1.27
0.0078	2.25	0.044	0.0625	-0.18	2.53

GTTG mutation, n_1 =0.65 weight					
Arabinose (mM)	m_1 weight	Experimental cost function	Digital cost function	$\frac{\partial < C >}{\partial m_1}$	$\frac{\partial < C >}{\partial n_1}$
0.2500	1	0.058	0.1250	-0.66	-2.70
0.1250	1.25	0.052	0.0625	-0.51	-2.87
(5) 0.0625	1.5	0.046	0.0625	-0.28	-2.55
(6) 0.0312	1.75	0.035	0	0.03	-0.75
0.0156	2	0.042	0.0625	0.08	0.60
0.0078	2.25	0.045	0.1250	0.05	0.38

TGGG mutation, n_1 =0.95 weight					
Arabinose (mM)	m_1 weight	Experimental cost function	Digital cost function	$\frac{\partial < C >}{\partial m_1}$	$\frac{\partial < C >}{\partial n_1}$
0.2500	1	0.052	0.1250	-0.66	-1.38
0.1250	1.25	0.052	0.1250	-0.50	-1.54
0.0625	1.5	0.038	0.0625	-0.22	-1.24
0.0312	1.75	0.039	0	-0.003	-0.42
0.0156	2	0.061	0.1250	0.12	0.68
0.0078	2.25	0.056	0.1250	0.10	0.75

Table S6.2. The experimental results of backpropagation algorithms based on Eq. S6.21 and Eq. S22.

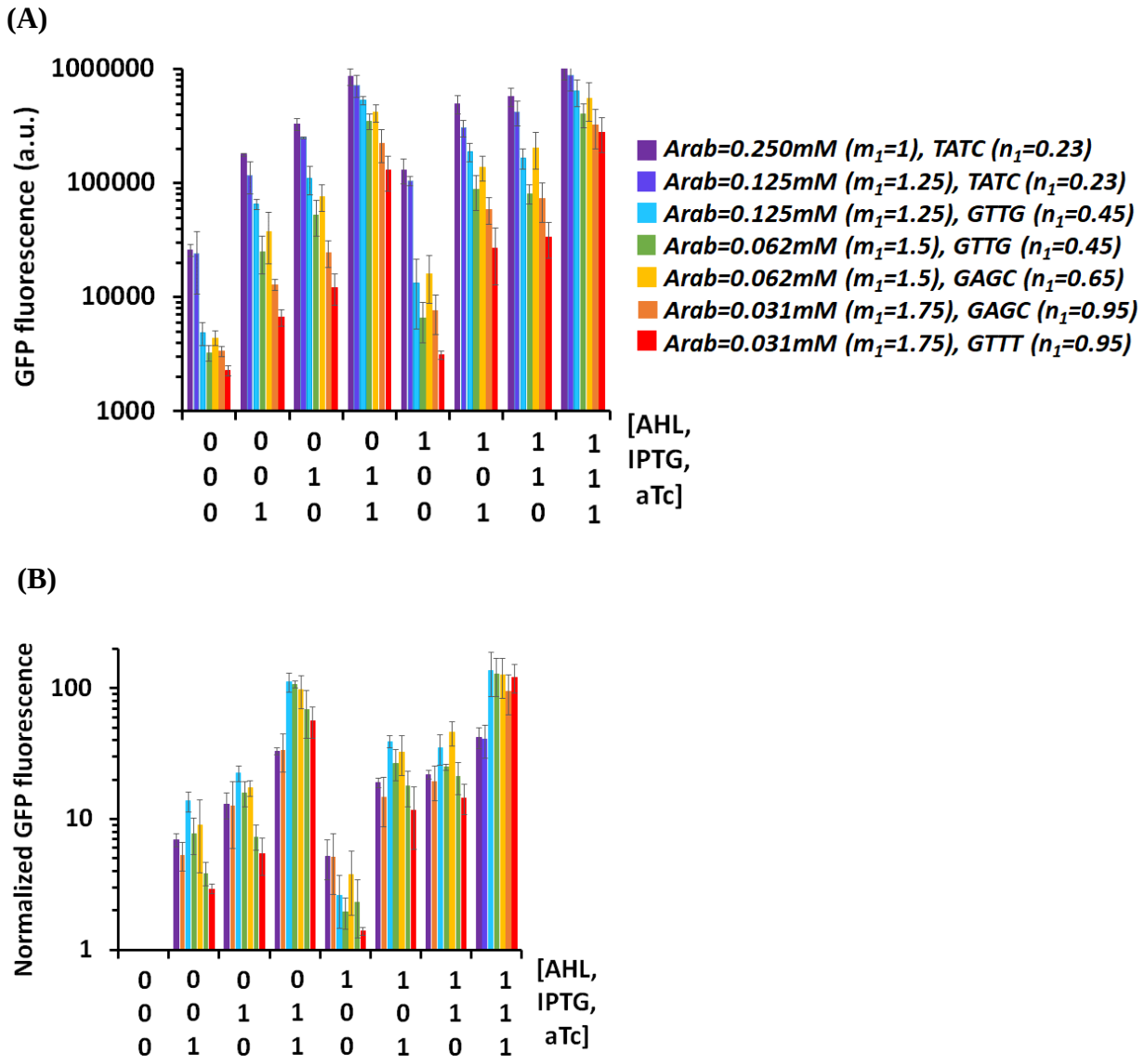


Fig. S6.8. (A) The experimental results of the three-input perceptgene network accept three analog inputs (AHL, IPTG and aTc) to reach a minimum cost function. This data is based on Fig. S6.5. (B) Normalized.

Table S6.3 List of parameters used in this section

Symbol	Description
C	Cost function of network
C_i	logarithmically quadratic cost function for each state or sample i
m_i	network weights
n_i	network weights
β	Basal level
ζ	Scalar that determines the rate of m_1 being updated (also known as learning/training rate)
Z_i	the network actual output for every state
Z_{Di}	the desired output of every state
K_{mi}	Effective dissociation constant
z_1, z_2	the output of every perceptgene layer in the biophysical model
y_i	the analog signal of every perceptgene layer in the biophysical model
N	the number of samples and is also called the batch size

7. Synthetic Data converters

In this work, we designed and built two types of data converters that operate in logarithmic domain:

1. Analog-to-Digital converter (ADC) which converts analog signals on a logarithmic scale to digital outputs, as shown in Fig. S7.1A. Namely, each decade is encoded to one discrete level (*33*).
2. Analog-to-multilevel (Fuzzy) converter which converts analog signals on a logarithmic scale to multi-discrete levels as shown in Fig. S7.1B. Specifically, we built a ternary converter.

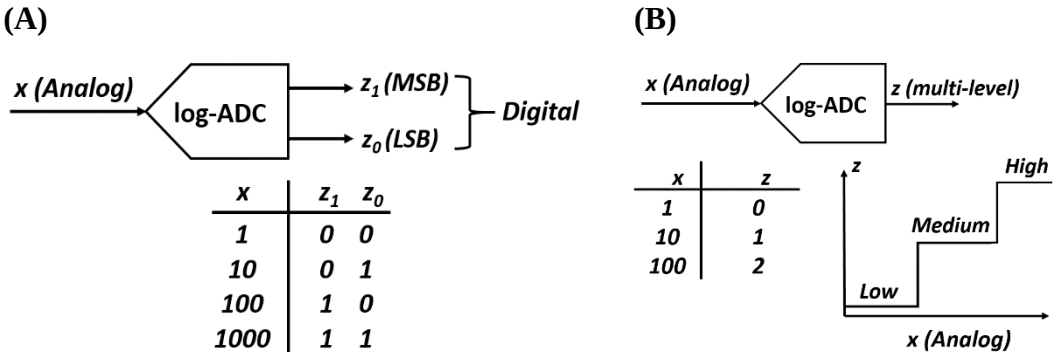


Fig. S7.1. A 2-bit data converters operating on the logarithmic domain: (A) ADC-Analog-to-Digital converter. (B) Analog-to-multilevel (Fuzzy) converter. LSB: Least Significant Bit, MSB: Most Significant Bit.

7.1. Design I: Design and implementation of 2-bit log-ADC

There are several architectures and concepts to design ADC systems (*34*). we used design principles of feed-forward neural networks (*35*, *36*) that can provide reliable results with a minimal number of synthetic parts (Fig. S7.2A). First, we designed a 2-bit ADC in the linear domain (equivalent to a perceptron model), and then we transformed it to the logarithmic domain. In the proposed design, a bit comparison is equivalent to neural activation in the proposed design, and each reference scale during the successive binary search algorithm is equivalent to a binary-weighted synapse. For simplicity, we approximated the activation function as a step function:

$$y_1 = n_1 \cdot x + A_1 \tag{S7.1}$$

$$Z_1 = \begin{cases} 1 & y_1 \geq 0 \\ 0 & y_1 < 0 \end{cases} \tag{S7.2}$$

$$y_0 = n_0 \cdot x + m_0 \cdot z_1 + A_0 \tag{S7.3}$$

$$Z_0 = \begin{cases} 1 & y_2 \geq 0 \\ 0 & y_2 < 0 \end{cases} \tag{S7.4}$$

Fig. S7.2B shows the output signals of 2-bit ADC in the linear domain. We divided the dynamic input range (IDR) to $2^2 = 4$ intervals. Therefore, the set of equations that describes the design of 2-bit ADC based on perceptron (Fig. S7.2A) is given by:

Interval 1: $0 \leq x < \frac{IDR}{4}$ $- Z_1 = 0, Z_0 = 0$:

$$y_1 = n_1 \cdot \frac{IDR}{4} + A_1 \leq 0 \quad \rightarrow A_1 \leq -n_1 \cdot \frac{IDR}{4} \tag{S7.5}$$

$$y_0 = n_0 \cdot \frac{IDR}{4} + A_0 \leq 0 \quad \rightarrow A_0 \leq -n_0 \cdot \frac{IDR}{4} \tag{S7.6}$$

Interval 2: $\frac{IDR}{4} \leq x < \frac{IDR}{2} - Z_1 = 0, Z_0 = 1$:

$y_1 = n_1 \cdot \frac{IDR}{2} + A_1 < 0$ ➔ $A_1 < -n_1 \cdot \frac{IDR}{2}$ (S7.7)

$y_0 = n_0 \cdot \frac{IDR}{2} + A_0 > 0$ ➔ $A_0 > -n_0 \cdot \frac{IDR}{2}$ (S7.8)

Interval 3: $\frac{IDR}{2} \leq x < \frac{3 \cdot IDR}{4} - Z_1 = 1, Z_0 = 0$:

$y_1 = n_1 \cdot \frac{3 \cdot IDR}{4} + A_1 > 0$ ➔ $A_1 > -n_1 \cdot \frac{3 \cdot IDR}{4}$ (S7.9)

$y_0 = n_0 \cdot \frac{3 \cdot IDR}{4} + m_0 + A_0 < 0$ ➔ $A_0 < -n_0 \cdot \frac{3 \cdot IDR}{4} - m_0$ (S7.10)

Interval 4: $\frac{3 \cdot IDR}{4} \leq x \leq IDR - Z_1 = 1, Z_0 = 1$:

$y_1 = n_1 \cdot IDR + A_1 \geq 0$ ➔ $A_1 \geq -n_1 \cdot IDR$ (S7.11)

$y_0 = n_0 \cdot IDR + m_0 + A_0 \geq 0$ ➔ $A_0 \geq -n_0 \cdot IDR - m_0$ (S7.12)

In summary, the conditions on the weights and biases to implement a 2-bit ADC in the linear domain:

$$-n_1 \cdot \frac{3 \cdot IDR}{4} < A_1 < -n_1 \cdot \frac{IDR}{2} \quad \text{(S7.13)}$$

$$-n_0 \cdot \frac{IDR}{2} < A_0 \leq -n_0 \cdot \frac{IDR}{4} \quad \text{(S7.14)}$$

$$-n_0 \cdot IDR - A_0 \leq m_0 < -n_0 \cdot \frac{3 \cdot IDR}{4} - A_0 \ \text{➔} \ m_0 = -n_0 \cdot \frac{IDR}{2} \quad \text{(S7.15)}$$

The simulation results of Fig. S7.2C suggest that 2-bit ADC can be implemented by a feedforward neural network when the design parameters (weights and biases) satisfy Eq. S7.13-Eq. S7.15. The simulation parameters are $IDR = 5$, $n_0 = n_1 = 4, m_0 = -10, A_1 = -12.5, A_0 = -7.5$. In our simulation, we used a sigmoid function to calculate z_i ($z_i = \frac{1}{1+e^{-y_i}}$, the output of each perceptron), instead of step function as was done in the analysis, where all the outputs above 0.5 as a "1" logic and those below 0.5 as "0" logic. To implement such a design in living cells, we must convert the design parameters to the logarithmic domain. However, these parameters cannot be achieved in living cells directly, and thus the ADC design needs to be modified. In the proposed design, the MSB dynamically controls the LSB threshold through a negative regulation (inhibitory weight) (Fig. S7.2D). The molecular 2-Bit ADC consists of two reactions (Z_0 and Z_1) which are regulated by the same input (x) (Fig. S7.3A). Each reaction represents a digital bit, and Z_1 enhances the reverse reaction of Z_0. So that the amount of Z_1* changes and accordingly the amount of Z_0 is affected. The set of biochemical reactions that describe the reaction network is given by:

$$\frac{dZ_1^*}{dt} = k_{f1} \cdot x^{n_1} \cdot Z_1 - k_{r1} \cdot Z_1^* \quad \text{(S7.16)}$$

$$\frac{dZ_0^*}{dt} = k_{f0} \cdot x^{n_0} \cdot Z_0 - k_{10} \cdot {Z_1^*}^{m_0} \cdot Z_0^* - k_{r0} \cdot Z_0^* \quad \text{(S7.17)}$$

$$Z_{T1} = Z_1^* + Z_1 \quad \text{(S7.18)}$$

$$Z_{T0} = Z_0^* + Z_0 \quad \text{(S7.19)}$$

Where Z_0, and $Z_1$are the product concentrations of biochemical reactions. k_{f0} and k_{f1} are the rates for the forward reactions from Z_0 to Z_0 and Z_1 to Z_1, respectively. Likewise, k_{r0} and k_{r1} are the rates for the corresponding backward reactions. The rate k_{10} describes the regulation of Z_1 on the activation of Z_0. Z_{T0} and Z_{T1} are the total concentration of molecules Z_0 and Z_1. At the steady-state:

$$Z_1^* = Z_{T1} \cdot \frac{\left(\frac{x}{K_{n1}}\right)^{n_1}}{1+\left(\frac{x}{K_{n1}}\right)^{n_1}} \quad \text{(S7.20)}$$

$$Z_0{}^* = Z_{T0} \cdot \frac{\left(\frac{x}{K_{n0}}\right)^{n_0}}{1+\left(\frac{x}{K_{n0}}\right)^{n_0}+\left(\frac{Z_1{}^*}{K_{m0}}\right)^{m_0}} \quad \text{(S7.21)}$$

Where n_0, n_1 and m_0 are Hill Coefficients. K_{n0}, K_{n1} *and* K_{m0} are the dissociation constants ($K_{n0} = \left(k_{r0}/k_{f0}\right)^{1/n_0}, K_{n1} = \left(k_{r1}/k_{f1}\right)^{1/n1}, K_{m0} = (k_{r0}/k_{10})^{1/m_0}$). The reaction activity is defined as ratio of the product and the total concentration of molecules. Thus, we can rewrite Eq. S7.20 and Eq. S7.21 as:

$$P_1 = \frac{Z_1{}^*}{Z_{T1}} = \frac{\left(\frac{x}{K_{n1}}\right)^{n_1}}{1+\left(\frac{x}{K_{n1}}\right)^{n_1}} \quad \text{(S7.22)}$$

$$P_0 = \frac{Z_0{}^*}{Z_{T0}} = \frac{\left(\frac{x}{K_{n0}}\right)^{n_0}}{1+\left(\frac{x}{K_{n0}}\right)^{n_0}+\left(\frac{P_1}{K_{m0}/Z_{T1}}\right)^{m_0}} \quad \text{(S7.23)}$$

$$P_0 = \frac{Z_0{}^*}{Z_{T0}} = \frac{\left(\frac{x}{K_{n0}}\right)^{n_0}\cdot\left(\frac{P_1}{K_{m0}/Z_{T1}}\right)^{-m_0}}{\left(\frac{P_1}{K_{m0}/Z_{T1}}\right)^{-m_0}+\left(\frac{x}{K_{n0}}\right)^{n_0}\cdot\left(\frac{P_1}{K_{m0}/Z_{T1}}\right)^{-m_0}+1} \quad \text{(S7.24)}$$

In case that $Z_{T1} >> K_{m0}$, we can approximate Eq. S7.23:

$$P_0 \approx \frac{\left(\frac{x}{K_{n0}}\right)^{n_0}\cdot\left(\frac{P_1}{K_{m0}/Z_{T1}}\right)^{-m_0}}{\left(\frac{x}{K_{n0}}\right)^{n_0}\cdot\left(\frac{P_1}{K_{m0}/Z_{T1}}\right)^{-m_0}+1} \quad \text{(S7.25)}$$

Fig. S7.3B shows a schematic model of the set of reactions based on Eq. S7.24 and Eq. S7.25. The schematic model consists of two perceptgenes that are connected in feedforward neural networks through a negative weight, similar to the ADC design (Fig. S7.2A), where:

$$B_1 = {K_{n1}}^{-n_1} \quad \text{(S7.26)}$$

$$B_0 = {K_{n0}}^{-n_0} \cdot \left(\frac{K_{m0}}{Z_{T1}}\right)^{m_0} \quad \text{(S7.27)}$$

$$B_1 = 10^{A_1} \quad \text{(S7.28)}$$

$$B_0 = 10^{A_2} \quad \text{(S7.29)}$$

To estimate the weights and biases parameters, which operates in the log-domain, we transformed Eq. S7.13-Eq. S7.15 from the linear scale to the logarithmic scale using Eq. S7.26-Eq. S7.29:

(1) $$-n_1 \cdot \frac{3\cdot IDR}{4} < \log(B_1) < -n_1 \cdot \frac{IDR}{2}$$

$$-\frac{3\cdot IDR}{4} < -\log(K_{n1}) < -\frac{IDR}{2} \quad \text{(S7.30)}$$

(2) $$-n_0 \cdot \frac{IDR}{2} < \log(B_0) \leq -n_0 \cdot \frac{IDR}{4}$$

$$-\frac{IDR}{2} < -\log(K_{n0}) + \frac{m_0}{n_0} \cdot \log\left(\frac{K_{m0}}{Z_{T1}}\right) \leq -\frac{IDR}{4} \quad \text{(S7.31)}$$

(m_0>0)

(3) $$-n_0 \cdot IDR - \log(B_0) \leq -m_0 < -n_0 \cdot \frac{3\cdot IDR}{4} - \log(B_0)$$

$$-IDR \leq -\log(K_{n0}) + \frac{m_0}{n_0} \cdot \log\left(\frac{K_{m0}}{Z_{T1}}\right) - \frac{m_0}{n_0} < -\frac{3\cdot IDR}{4} \quad \text{(S7.32)}$$

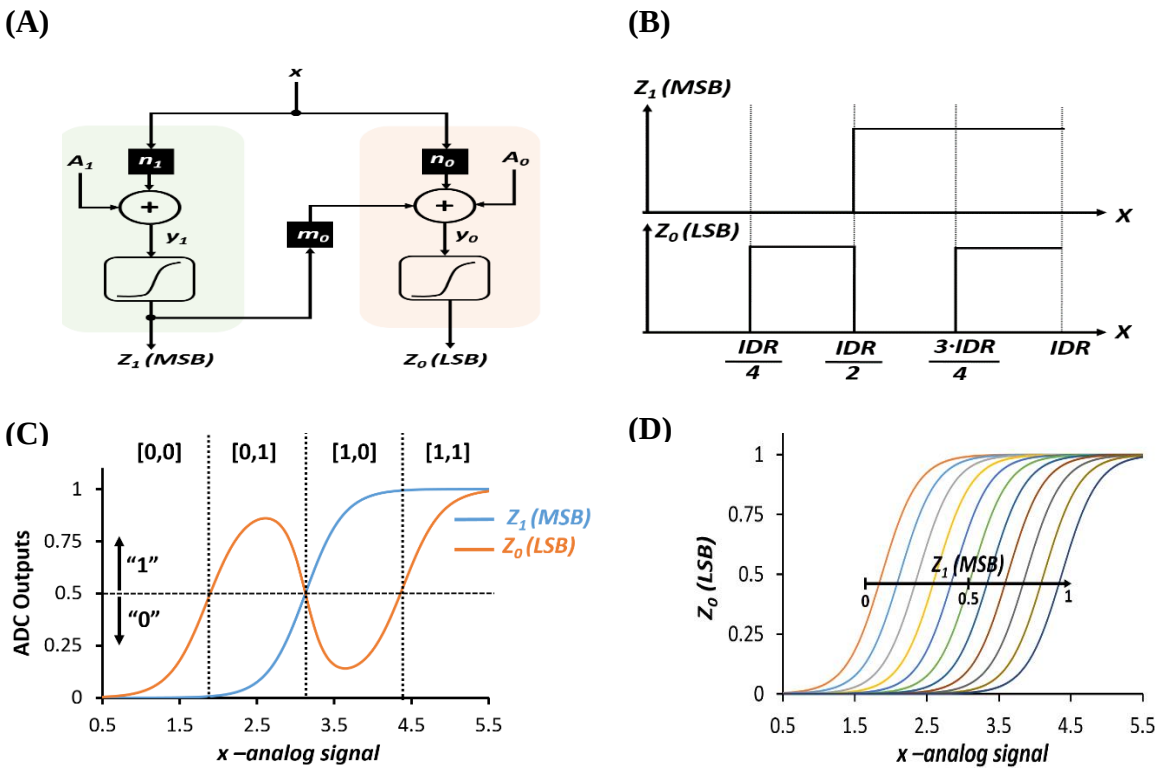


Fig. S7.2. (A) Design of 2-bit ADC based on feedforward neural networks. (B) Digital outputs of 2-bit ADC operate in the linear domain. (C) Simulation results of 2-bit neural-network ADC design. (D) Most Significant Bit (MSB) dynamically controls the Least Significant Bit (LSB) of 2-bit ADC via varying the value of m_0.

The simulation results as shown in Fig. S7.3C suggest that a 2-bit molecular ADC that operates in the log domain can be implemented using a feedforward neural network when the parameters (weights and biases) satisfy Eq. S7.30-Eq. S7.32. In the particular simulation, $IDR = 5, n_0 = n_1 = 2, m_0 = 4, K_{n0} = 25, K_{n1} = 800, K_{m0} = 40, Z_{T1} = 800, B_0 = 2.4 \times 10^{-11}, B_1 = 1.56 \times 10^{-6}, A_0 = -10.5, A_1 = -5.8$.Notably, operating in the log domain allows parameter values to be compressed, comparing with the ADCs in the linear domain with similar features ($IDR = 5, 2\ bits$). That means the design parameters required to implement molecular ADC in living cells are achievable by contrast to linear ADC. Furthermore, varying m_0 (Hill coefficient) affects the LSB (Fig. S7.3D), in particular, the behavior of LSB is qualitatively changed. As in Fig. S7.3E when $n_0 \approx m_0$, LSB acts as ternary logic **(**$IDR = 5, n_0 = n_1 = 1.5$, $K_{n0} = 25, K_{n1} = 500$, $K_{m0} = 40$, $Z_{T1} = 700, m_0 = 1.5$ (for ternary logic, the blue curve**)** and $m_0 = 2.5$ (for quaternary logic, the red curve).

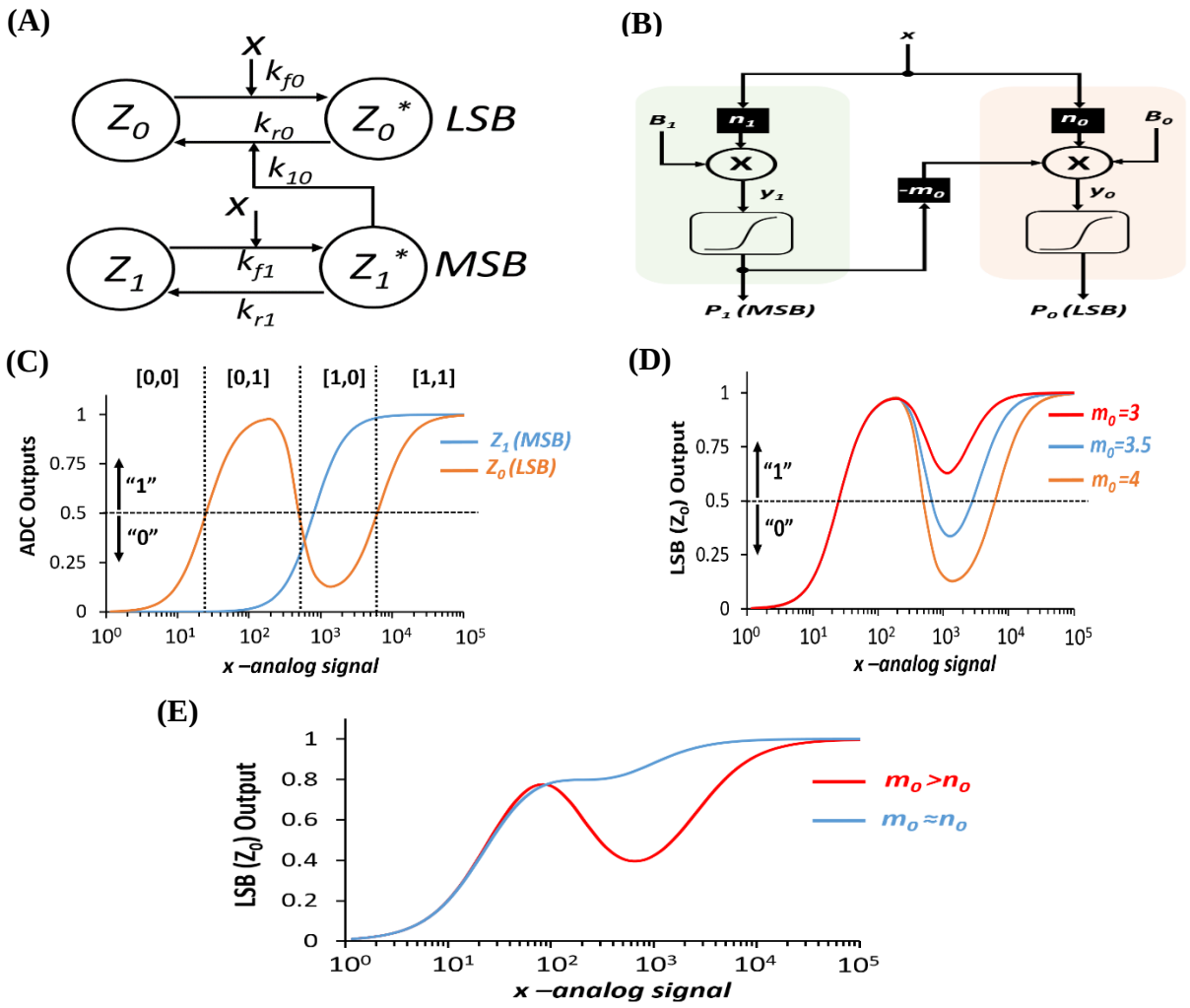


Fig. S7.3. (A) Design of 2-bit molecular ADC by controlling LSB via MSB. (B) An abstract model of 2-bit molecular ADC based on the perceptgene feedforward network. (C) Simulation results of 2-bit molecular ADC. (D) The influence of m_0 on the LSB behavior. (E) 2-bit Molecular ADC displays ternary logic by programming the m_0 weight.

7.1.1. Optimization process of 2-bit ADC

In this process, we changed the original mathematical model (Fig. S7.3) to be suitable for genetic networks. Fig. S7.4A is built based on Fig. S7.3B, including:

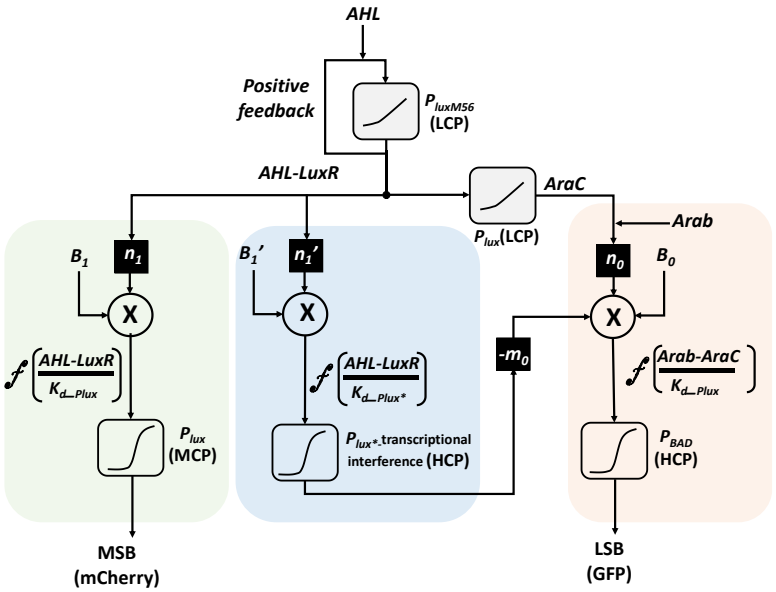
1. Positive feedback loop for linearization.
2. AraC as a new wire to implement the LSB.
3. Transcriptional interference promoter to implement the subtraction.

Fig. S7.4B is built based on Fig. S7.4A, including:

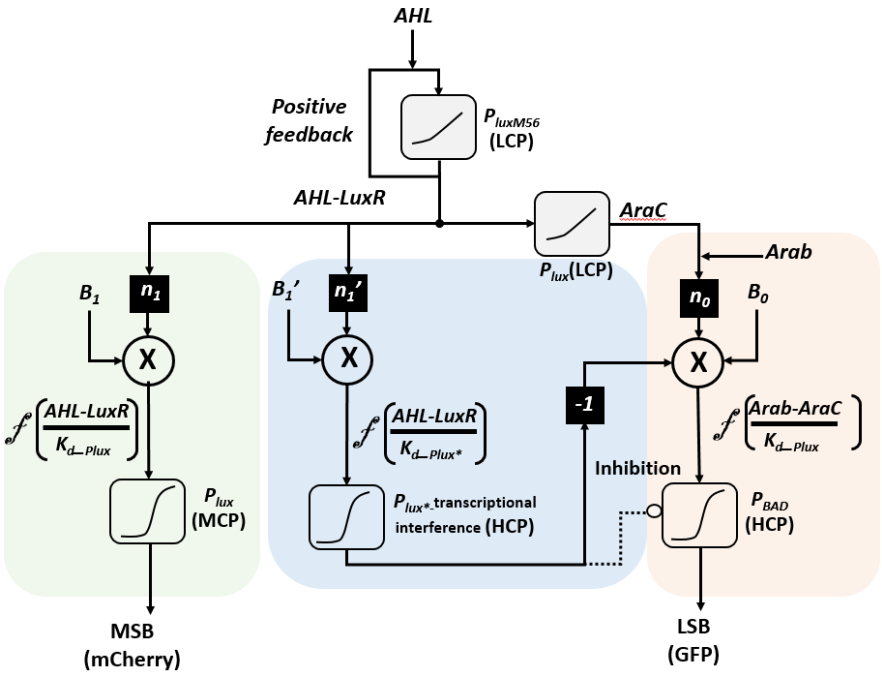
1. $m_0 = -1$, since there is only one binding site of LuxR in the transcriptional interference promoter.
2. Inhibition to P_{BAD}, which is achieved by the transcriptional interference of P_{lux} promoter.

Fig. S7.4C is the final construct which is built based on Fig. S7.4B, including the TetR repressor to reduce the disturb of the inhibition.

(A)



(B)



(C)

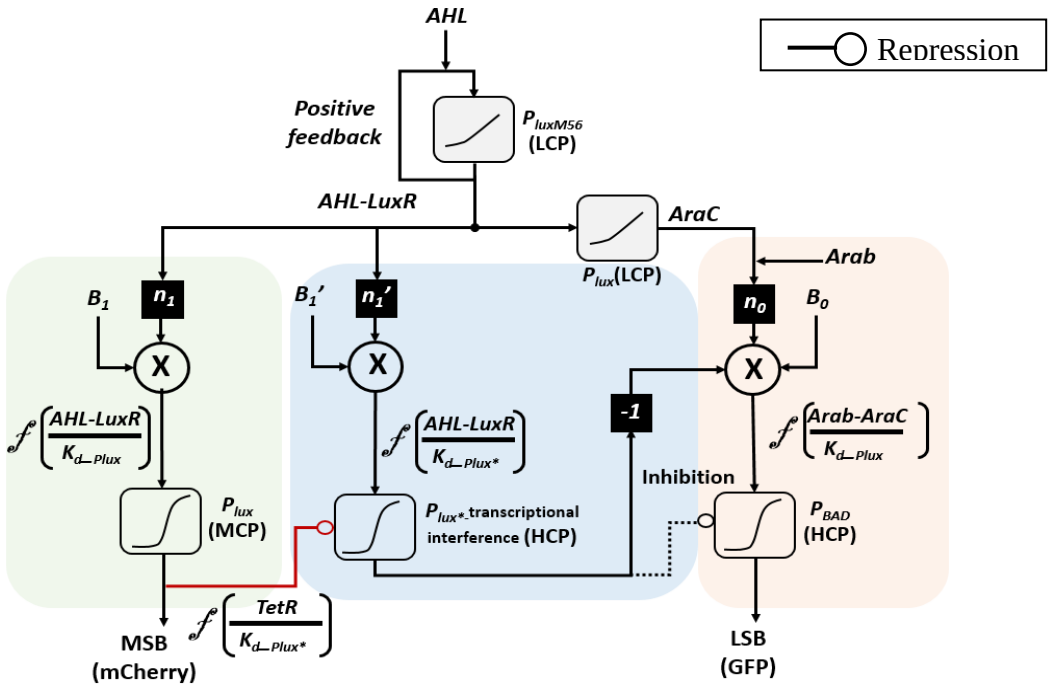


Fig. S7.4. Process optimization of 2-bit ADC.

To implement the 2-bit Molecular ADC in living cells, we first constructed a genetic circuit with an effective tunable threshold. To that end, we utilized two competitive promoters (P_{BAD} vs P_{lux}) that are located in opposite orientation to each other, which P_{lux} produces transcriptional interference with P_{BAD} (Fig. S7.5A). For P_{BAD} we call forward promoter and for P_{lux} we call reverse promoters. Oppositely oriented promoters relative to a gene have been reported in previous studies (*37–39*) to tune gene expression (*40*), control the input threshold of genetic switches (*41*, *42*), and reduce the leaky expression of toxic proteins (*43*). A RiboJ (*44*) was used to cleave the *5′*-UTR of GFP mRNA, a computationally designed RBS, the GFP -coding sequence and a transcriptional terminator. A reverse complementary terminator was cloned upstream to P_{BAD} to disturb the activity of RNA polymerase for P_{lux}. Adetailed biophysical model was developed (*40*) to describe such systems, in our work, for simplicity. Because the two opposite promoters are located close to each other, we treat the system as one statistical thermodynamic model (Fig. S7.5B). The model describes 5 different statistical states, (1) promoters are empty, (2) RNA polymerases (RNAp) with transcription factors are bound on both promoters, leading to a basal level, (3) the complex activator (Y_1)-RNAp is bound to the forward promoter, leading to active the output signal, (4) the complex activator (Y_2)-RNAp is bound to the reverse promoter, leading to inhibit the output signal, and (5) the complex activator (Y_1)-RNAp is bound to the forward promoter, and the complex activator (Y_2)-RNAp is bound to the reverse promoter. In our model, we also assume that the collision interference is large and thus the probability that the forward and the reverse RNA polymerases can simultaneously bind to the DNA is very low ($\theta << 1$). Therefore, the level of gene expression is proportional to the probability (P) that RNA polymerase is bound to the forward promoter at the equilibrium:

$$P = \frac{\left(\frac{Y_1}{K_{d1}}\right)^{n_1} + \beta}{1 + \left(\frac{Y_1}{K_{d1}}\right)^{n_1} + \left(\frac{Y_2}{K_{d2}}\right)^{n_2}} \quad \text{(S7.33)}$$

Y_1 is the concentration of Arab-$AraC$ complex. Y_2 is the concentration of AHL -LuxR complex. These variables are given by the following set of equations:

$$g(x_i) = \frac{\left(\frac{x_i}{K_{mi}}\right)^{m_i}}{1 + \left(\frac{x_i}{K_{mi}}\right)^{m_i}} \quad \text{(S7.34)}$$

$$Y_1 = AraC_T \cdot g(Arab) \quad \text{(S7.35)}$$

$$Y_2 = LuxR_T \cdot g(AHL) \quad \text{(S7.36)}$$

We can rewrite the promoter activity (Eq. S7.33) as follows:

$$P = \frac{\frac{AraC_T \cdot g(Arab)}{K_{deff}} + \beta_{eff}}{1 + \frac{AraC_T \cdot g(Arab)}{K_{deff}}} \quad \text{(S7.37)}$$

$$\beta_{eff} = \frac{\beta}{1 + \frac{LuxR_T \cdot g(AHL)}{K_{d2}}} \quad \text{(S7.38)}$$

$$K_{deff} = K_{d1} \cdot \left(1 + \frac{LuxR_T \cdot g(AHL)}{K_{d2}}\right) \quad \text{(S7.39)}$$

Where we assume that $n_1 = 1$, $n_2 = 1$. Fig. S7.5C shows the experimental and simulation results of the normalized signals for the forward (P_{BAD}) promoter and reverse promoter (P_{lux}) using the set of Eq. S7.37- Eq. S7.39. For each AHL concentration, we normalized the measured GFP by the maximum achieved GFP level. Assuming that Arabinose>>K_{m1}, the fold-change of GFP can be given by:

$$\frac{P_{max}}{P_{min}} = \frac{AraC_T}{K_{deff} + AraC_T} \cdot \frac{1}{\beta_{eff}} \quad \text{(S7.40)}$$

The experimental results show that varying the concentration of AHL affects the Arabinose-to-GFP transfer function. When AHL concentration increases, the fold-change of GFP decreases, which is consistent with Eq. S7.40. The experimental results can be well captured by the simulation results. Also, our experimental and simulation results imply that there is interference between the P_{BAD} promoter and P_{lux}, leading to a shift in the threshold or effective dissociation constant by 1.5 magnitudes of orders. Parameters that were used in the simulation are $K_{m1} = 1000, m_1 = 1.5, K_{m2} = 0.3, m_2 = 1, AraC_T/K_{d1} = 11, LuxR_T/K_{d2} = 10, \beta = 0.0055$. To fit the absolute GFP signals to our simulation results, we modified Eq. S7.37 by including a repression term:

$$GFP = GFP_{max} \cdot \frac{\frac{AraC_T \cdot g(Arab)}{K_{deff}} + \beta_{eff}}{1 + \frac{AraC_T \cdot g(Arab)}{K_{deff}}} \cdot \left(\frac{1}{1 + \rho \cdot \frac{LuxR_T \cdot g(AHL)}{K_{d2}}} + \beta_2\right) \quad \text{(S7.41)}$$

Fig. S7.5D shows that Eq. S7.41 captures well the behavior of the absolute GFP signals ($GFP_{max} = 5.5 \times 10^5, \beta_2 = 0.04, \rho = 0.8$). The repression term takes effect when $\theta > 0$ in Eq. S7.33. In conclusion, the competition between the two promoters increases the threshold, and decreases the promoter activity (signal).

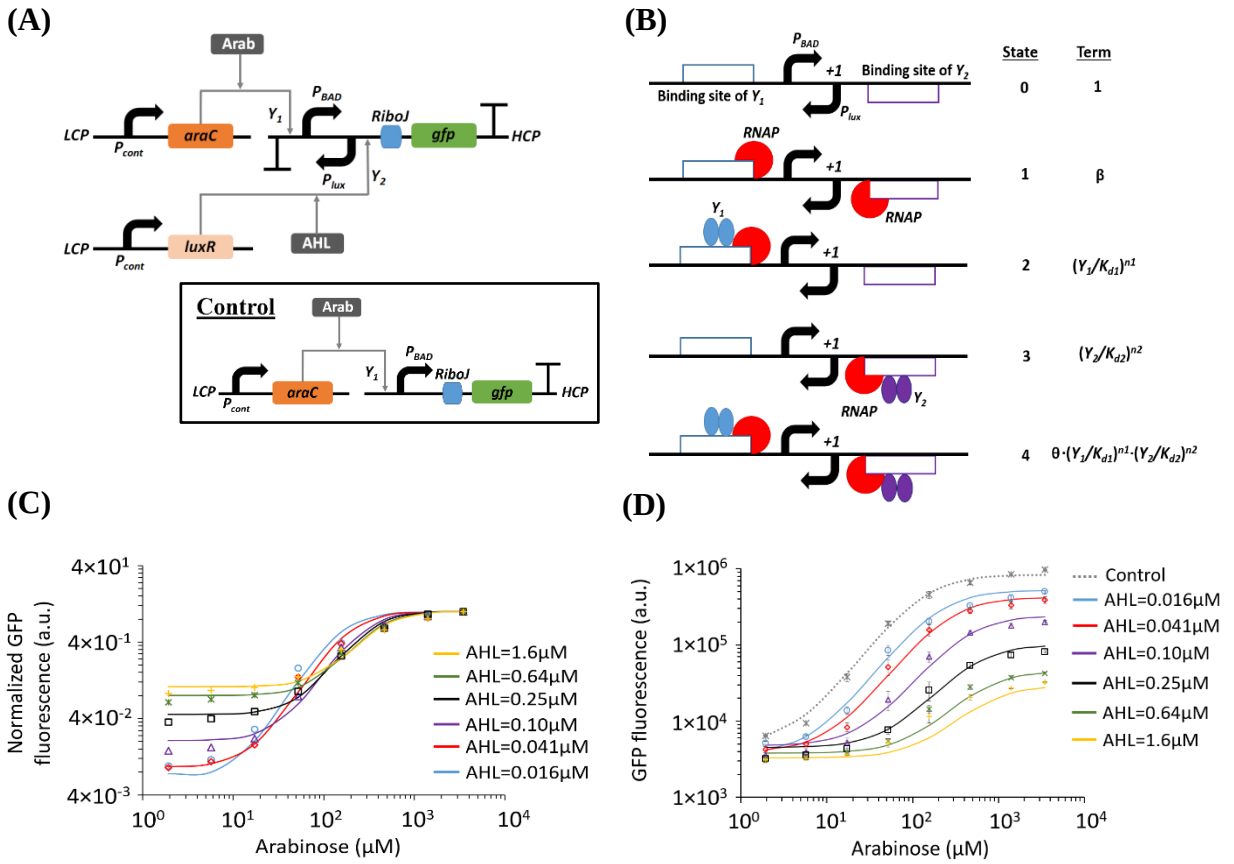


Fig. S7.5. (A) Genetic circuit with a tunable effective threshold based on a forward (P_{BAD}) promoter and a reverse (P_{lux}) promoter that is oriented in opposite direction to P_{BAD}. The $P_{lux}$is having a transcriptional interference with P_{BAD}. (B) The binding states of the forward promoter and the reverse P_{lux}. (C) Experimental and simulation (based on Eq. S7.37-Eq. S7.39) results of normalized measured GFP signal. (D) Experimental and simulation (Eq. S7.38-Eq. S7.39, Eq. S7.41) results of absolute measured GFP signal. The control experiment is shown in Fig. S7.5A inset and the experimental results are shown in Fig. S7.5D.

Second approach: The optimization process of implementing a 2-bit Molecular ADC in living cells (Fig. S7.6) consists of three steps:

1. Open loop circuit: AHL is the input signal. The AHL -LuxR complex binds to the mutated promoter P_{luxM56}. The latter promoter regulates AraC and *A* (another activator) in analog fashion. AraC affects the LSB by binding to P_{BAD}, and *A* affects the MSB by binding to P_1. The transcription factor *B*, which is regulated by P_1, regulates the P_{BAD} promoter.
2. PF circuit: we replaced the open loop with a positive feedback circuit.
3. We replaced the transcription factor *B* with LuxR. We used mutations of P_{lux} that has a different strength.

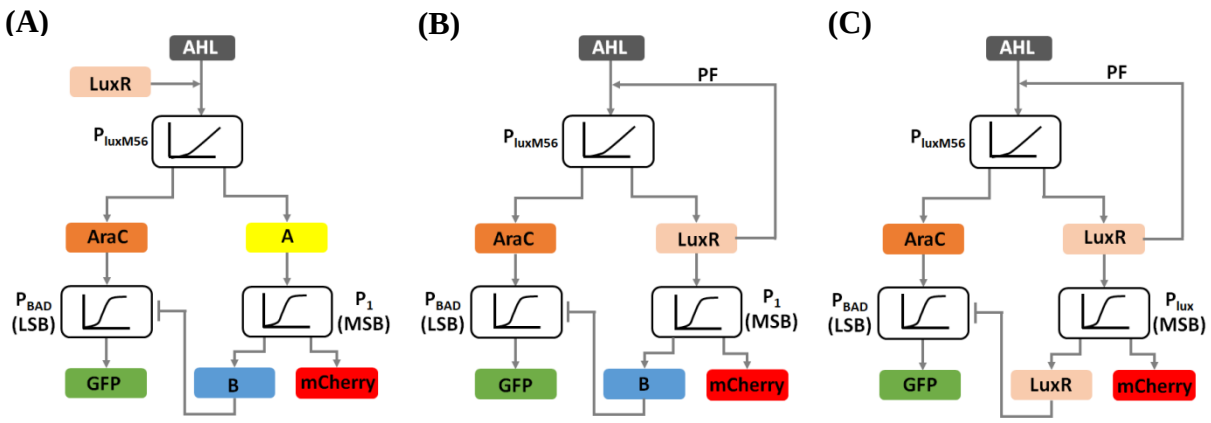


Fig. S7.6. Optimization process of implementation 2-bit molecular ADC in living cells.

Based on our optimization, we created two circuits that implement LSB and MSB (Fig. S7.7A and B), and another two control circuits (Fig. S7.7C and D). The LSB and MSB circuits accept AHL as an analog input and include a graded PF (P_{luxM56}), which regulates LuxR in an analog fashion. The P_{lux} of LSB circuit is located on LCP and regulates AraC in an analog manner (Fig. S2.13). In particular, we kept a low expression level of AraC by altering the binding between RNA polymerase and the promoter P_{lux} (*45*). Subsequently, the Arabinose- AraC complex binds to the forward P_{BAD} the promoter, while the LuxR − AHL complex binds to the reverse P_{lux} promoter. The binding reaction of LuxR − AHL complex, causes the RNA polymerase to reverse P_{lux} dynamically increases the P_{BAD} threshold and decreases the GFP expression (Fig. S7.8A). The MSB circuit that locates on MCP (Fig. S7.7B), regulates the output signal in a digital fashion. Following the previous Eq. S7.34 to Eq. S7.41, a set of empirical models are used to describe the LSB and MSB signals:

$$MSB \propto \frac{\left(\frac{AHL}{K_1}\right)^{r_1}+\beta_1}{1+\left(\frac{AHL}{K_1}\right)^{r_1}} \tag{S7.42}$$

$$LSB \propto \frac{\left(\frac{AHL}{K_0}\right)^{r_0}+\beta_0}{1+\left(\frac{AHL}{K_0}\right)^{r_0}+\alpha\cdot\left(\frac{MSB}{K_2}\right)^{r_2}} \cdot \left(\frac{1}{1+\left(\frac{AHL}{K_3}\right)^{r_3}}\right) + \beta_2 \tag{S7.43}$$

To find the model parameters, we first fitted the control experimental results (Fig. S7.8A) to a Hill-function (Eq. S7.42). Both control circuits contain a graded PF (P_{luxM56}) and P_{BAD} promoter. As a side note, control 1 appears similar to the circuit in Fig. S2.9, except that a degradation tag was added to AraC. In control 2 circuit, the transcriptional interference is controlled by a constitutive promoter P_{const} to P_{BAD}, which reduced the basal level and increased the threshold of P_{BAD} activation (Fig. S7.8A). The data was normalized by the maximum achieved level. The circuits were modeled as follows:

Control 1: $r_1 = 0.5, K_1 = 20$

Control 2: $r_1 = 0.7, K_1 = 50$

MSB: $r_1 = 1.5, K_1 = 500, \beta_1 = 0.1$

LSB: $r_0 = 0.8, K_0 = 40, \beta_0 = 0.001, \alpha = 200, r_2 = 1.5, K_2 = 10, r_3 = 1, K_3 = 500, \beta_2 = 0.04$

As in Fig. S7.8B, the proposed LSB and MSB circuits successfully converted the dynamic range of AHL concentration to [0,0], [0,1] and [1,0] logic states, while failed to achieve the [1,1] state. This circuit acts a 1.5-bit ADC.

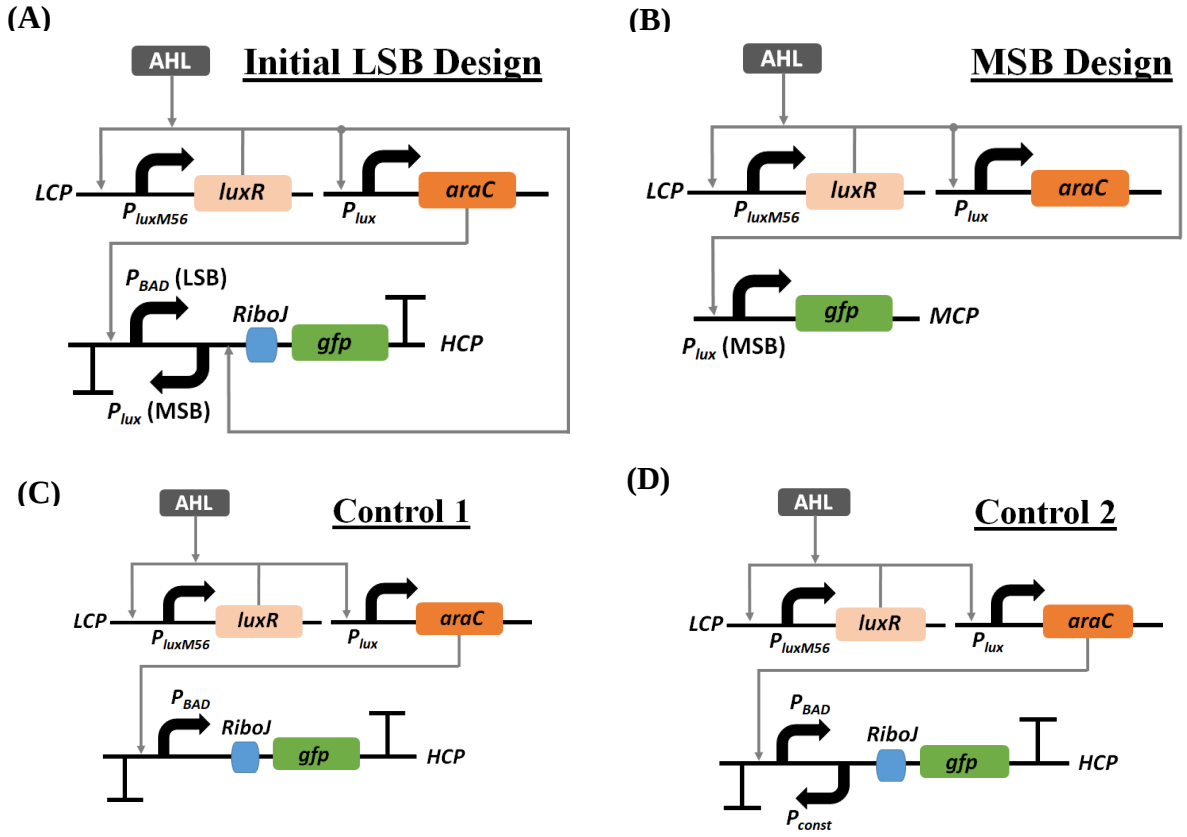


Fig. S7.7. (A) The implementation of LSB genetic circuit using a forward P_{BAD} promoter and reverse P_{lux} promoter which produces a transcriptional interference. (B) The implementation of MSB. (C) Control 1 circuit. (D) Control 2 circuit using a constitutive promoter to produce the transcriptional interference.

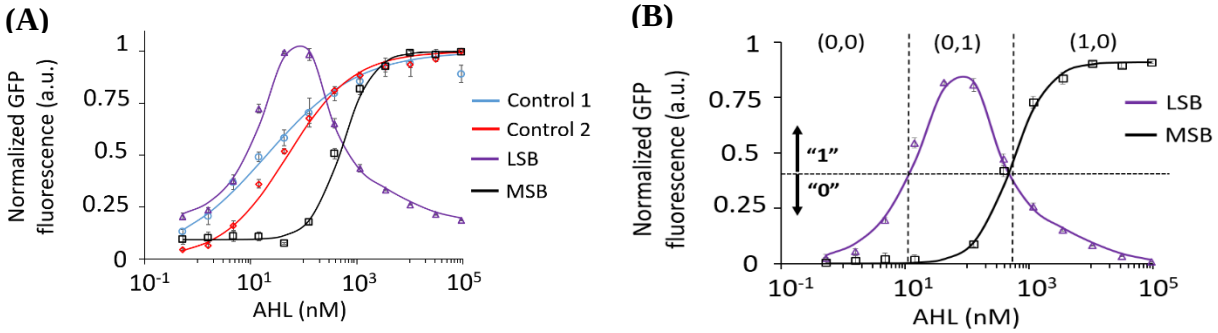


Fig. S7.8. (A) Experimental results of LSB, MSB and control circuits. Solid lines indicate modelling results based on the empirical model (Eq. S7.42, Eq. S7.43). We used Arabinose of 0.05 M for all the circuits. (B) The normalized GFP signals of LSB and MSB referred to each basal level (the data was subtracted by the basal levels of LSB and MSB).

The failure of the proposed LSB and MSB circuits to achieve the [1,1] logic state is possibly due to the repression of the P_{lux}transcriptional interference with P_{BAD} promoter in the presence of high levels of AHL. To solve this issue, a repressor (TetR) was added to the MSB (Fig. S7.9A). The TetR indirectly inhibits the activity of the P_{lux} transcriptional interference in the presence of high levels of AHL, through binding to the combinatorial $P_{lux/tetO}$ promoter (Fig. S7.9B). To implement, we constructed a hybrid promoter that consists of a forward P_{BAD} promoter and a combinatorial $P_{lux/tetO}$ promoter which is oriented in opposite direction to P_{BAD} (Fig. S7.9B). To understand the mechanism of $P_{lux/tetO}$, we created two control circuits (control 3 and 4) that are regulated by AHL and a graded PF and includes a forward P_{BAD} and a reverse combinatorial $P_{lux/tetO}$ as shown in Fig. S7.9C. In control 3, TetR is regulated by a constitutive promoter, and in control 4 there is no expression of TetR. The experimental results of control circuits 3 and 4 (Fig. S7.10) indicate that when TetR binds to the $P_{lux/tetO}$ promoter, there is no transcriptional interference with P_{BAD} promoter. Therefore, the control 3 circuit achieves a high GFP signal for high AHL levels. The results of control 3 are similar to those of control 1. The results of control 4 are similar to those of LSB circuit (Fig. S7.8A). The data of control 3 and 4 are well fitted by our empirical model (Eq. S7.42-Eq. S7.43), with the parameters as follows:

Control 3: $r_1 = 0.7, K_1 = 200, \beta_1 = 0.07$

Control 4: $r_0 = 0.8, K_0 = 40, \beta_0 = 0.001, \alpha = 200, r_2 = 1.5, K_2 = 100, r_3 = 1, K_3 = 500, \beta_2 = 0.005$.

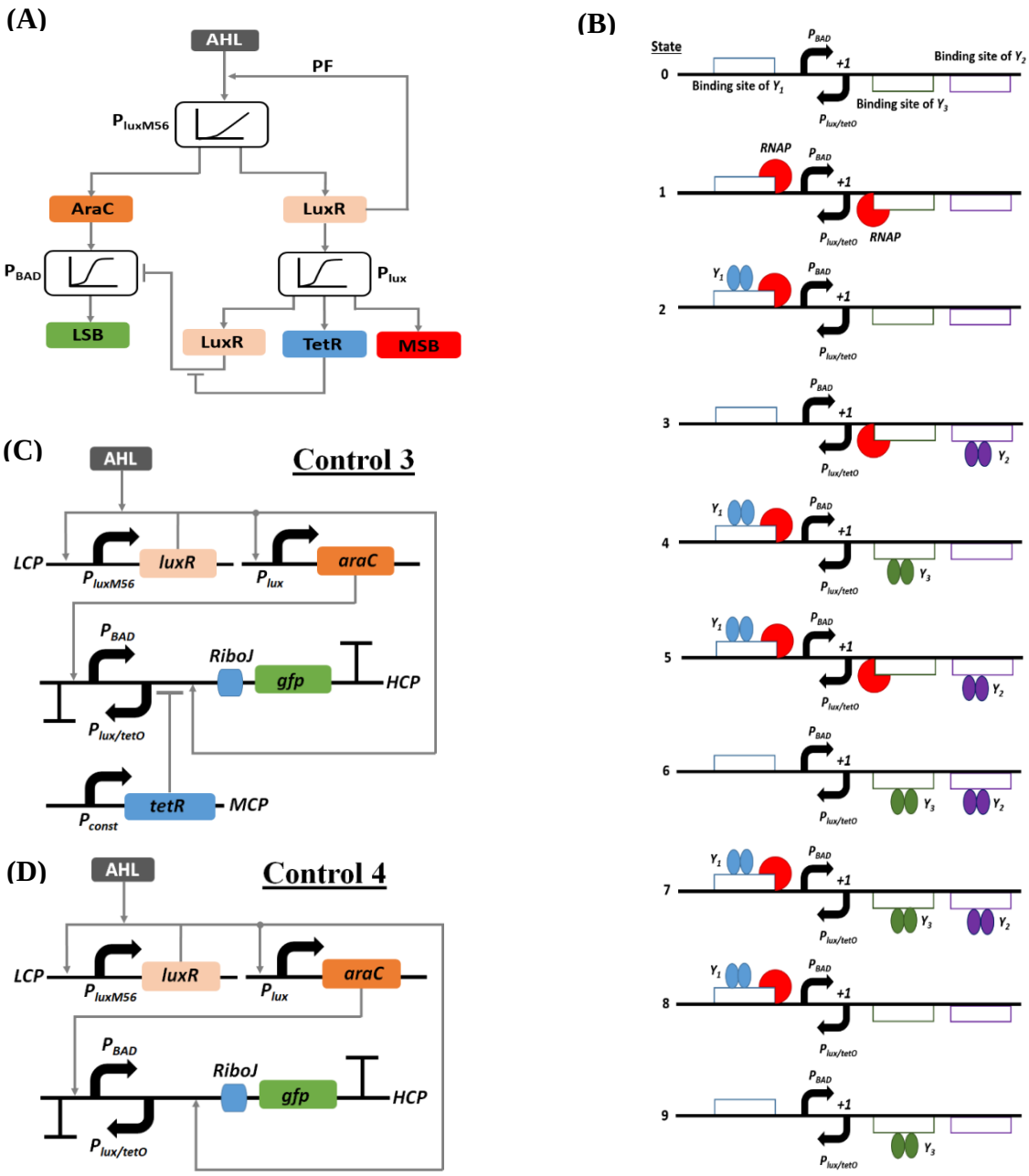


Fig. S7.9. (A) A design of 2-bit ADC includes the TetR to inhibit the regulation on LSB by MSB for high AHL. (B) The binding states of forward P_{BAD} promoter and a combinatorial $P_{lux/tetO}$ promoter which is oriented in opposite direction and produces a transcriptional interference with P_{BAD}. (C) A control circuit consists of a graded PF that regulates P_{BAD} promoter and a combinatorial $P_{lux/tetO}$, while $TetR$ repressor is constitutively expressed. (D) A control circuit consists of a graded PF that regulates P_{BAD} promoter and a combinatorial $P_{lux/tetO}$, when there is no expression of TetR repressor.

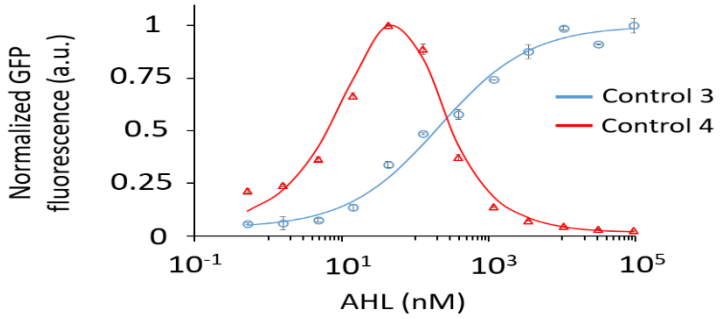


Fig. S7.10. Experimental results of control circuits 3 and 4. Solid lines represent modelling results from the empirical models (Eq. S7.42, Eq. S7.43). We induced the circuits with Arabinose of 0.6 mM.

Based on our control experimental results, we modified the LSB circuit to allow TetR being dynamically regulated by P_{lux} promoter **(Fig. S7.11A shows the construction of 2-bit ADC)**. Empirical models for the new circuit are given by:

$$MSB \propto \frac{\left(\frac{AHL}{K_1}\right)^{r_1}+\beta_1}{1+\left(\frac{AHL}{K_1}\right)^{r_1}} \tag{S7.44}$$

$$LSB \propto \frac{\left(\frac{AHL}{K_0}\right)^{r_0}+\beta_0}{1+\left(\frac{AHL}{K_0}\right)^{r_0}+\alpha\cdot\left(\frac{MSB}{K_2}\right)^{r_2}} \cdot \left(\frac{1}{1+f(AHL)}\right) + \beta_2 \tag{S7.45}$$

Where $f(AHL)$ represents the amount of GFP as a function as AHL. For a high concentration of TetR $f(AHL) \to 0$, and for a low concentration of TetR, Eq. S7.45 tends to converge to Eq. S7.43 therefore, an empirical model of $f(AHL)$ can be given by:

$$f(AHL) = \frac{LuxR-AHL}{1+TetR} \tag{S7.46}$$

For simplicity, we assumed that $f(AHL) \propto MSB$, then:

$$LSB \propto \frac{\left(\frac{AHL}{K_0}\right)^{r_0}+\beta_0}{1+\left(\frac{AHL}{K_0}\right)^{r_0}+\alpha\cdot\left(\frac{MSB}{K_2}\right)^{r_2}} \cdot \left(\frac{1}{1+\alpha\cdot\left(\frac{MSB}{K_3}\right)^{r_3}}\right) + \beta_2 \tag{S7.47}$$

Fig. S7.11B shows the experimental results of the modified LSB circuit using the new design, when TetR is regulated by MSB. These results demonstrate that 2-bit ADC can be achieved using such a design (Fig. S7.11C). The data of 2-bit ADC is well fitted by our empirical models (Eq. S7.44, Eq. S7.47), with a set of model parameters:

MSB: $r_1 = 1.5\,, K_1 = 500, \beta_1 = 0.015$

LSB: $r_0 = 1.45, K_0 = 20, \beta_0 = 0.03, \alpha = 300, r_2 = 1.2, K_2 = 1.5, r_3 = 1.3, K_3 = 85, \beta_2 = 0.04$

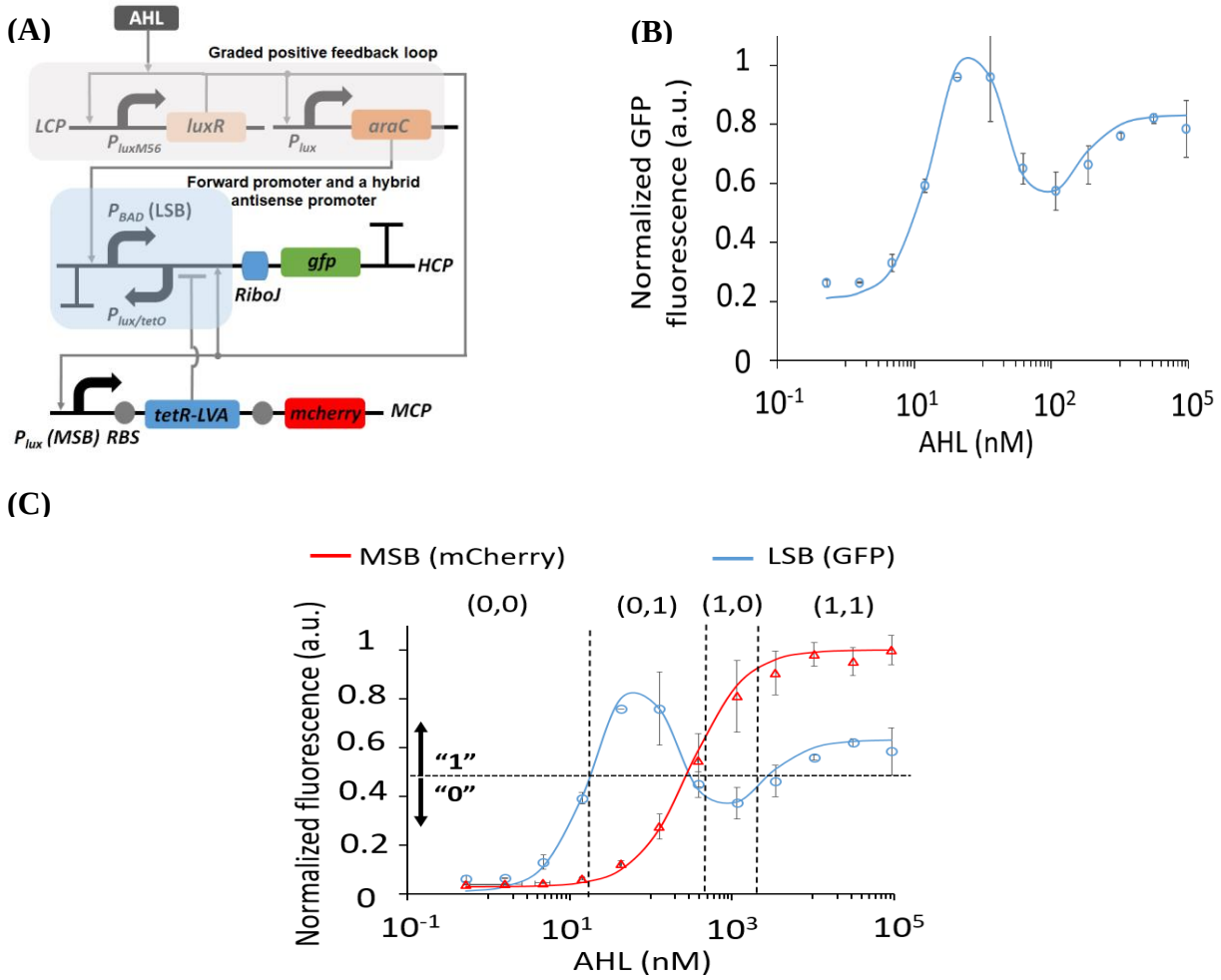


Fig. S7.11. (A) Implementation of a 2-bit ADC, where the LSB circuit is modified to allow TetR to be regulated by MSB. (B) Experimental and simulation (Eq. S7.47) results of the modified LSB circuit. (C) The normalized GFP and mCherry signals of the 2-bit ADC. The signals referred to each basal level (The data was subtracted by the basal levels of GFP and mCherry). Solid lines indicate the modeling results of the empirical models (Eq. S7.44, Eq. S7.47). We induced the circuits with Arabinose of 0.4 mM.

7.2. Design II: Design and implementation of 2-bit log-ADC

ADC systems have significant applications in biotechnology and medicine (*46*), for example, ADC can be used to regulate several genes with all logic combinations using only a single inducer. To improve the performance of our 2-bit ADC, we combined principles of neural-network and pipelined ADC (*47*) design (Fig. S7.12A). Pipelined ADC consists of several cascaded stages; every stage is built from comparators with each one has its own linear threshold (*47*). In the new design, we added a third perceptgene which receives AHL and acts as comparator for very high levels of AHL concentration. In this case, we require that $B_2 > B_1$, and $n_2 \approx n_1$. The design of hybrid ADC is shown in Fig. S7.12B. The LSB (GFP) signal is regulated by two parts: (1) a forward P_{BAD} promoter and an reverse P_{lux} promoter, which is oriented in opposite direction to P_{BAD} (Fig. S7.7A); (2) a quorum sensing P_{rhlR} promoter that interacts with AHL inducer (Fig.

S7.13) (*3*). The MSB is implemented by a P_{lux} promoter and regulates mCherry. By fitting the activity of P_{rhlR} to Eq. S7.44, we obtained that $K_1 = 20 \times 10^3$ nM which is larger than K_1 (500 nM) of P_{lux}, satisfying $B_2 > B_1$.

The 2-bit hybrid ADC was constructed as shown in Fig. S7.14. The circuit receives AHL as an analog input and contains a graded PF (P_{luxM56}) which regulates LuxR. The LSB circuit is built from two GFP signals: (1) Forward P_{BAD} promoter reverse P_{lux} promoter and (2) the P_{rhlR} promoter. The AraC regulated by P_{lux} is located on LCP (Fig. S7.14A). We altered the binding efficiency of RNA polymerase (*45*) to ensure a low expression level of AraC. Meanwhile, the AraC and LuxR – AHL complex binds to the forward P_{BAD} and reverse P_{lux} promoter, respectively, which increases the threshold of P_{BAD} and decreases the expression level of GFP. The P_{rhlR} promoter located on HCP is activated by LuxR – AHLcomplex and regulates the GFP signal for high AHL concentrations. To achieve similar GFP levels in the two parts, an ssrA degradation tag (*15*) (LVA) was added on HCP. The P_{lux} of MSB circuit which is located on MCP, regulates the output mCherry signal. Fig. S7.14B shows the experimental results of 2-bit hybrid ADC with distinct four logic states. The results show that there is a narrow region (marked in gray color) that the ADC has irregular behavior.

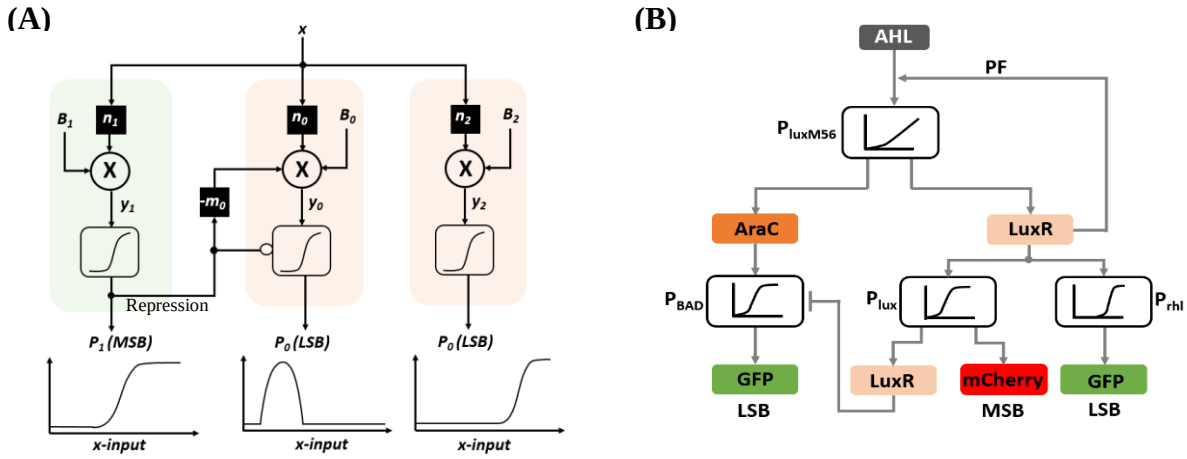


Fig. S7.12. (A) A 2-bit hybrid ADC combined with neural networks and Pipelined ADC. (B) A 2-bit hybrid ADC using a third comparator, which activates P_{rhlR} promoter only for a high AHL concentration.

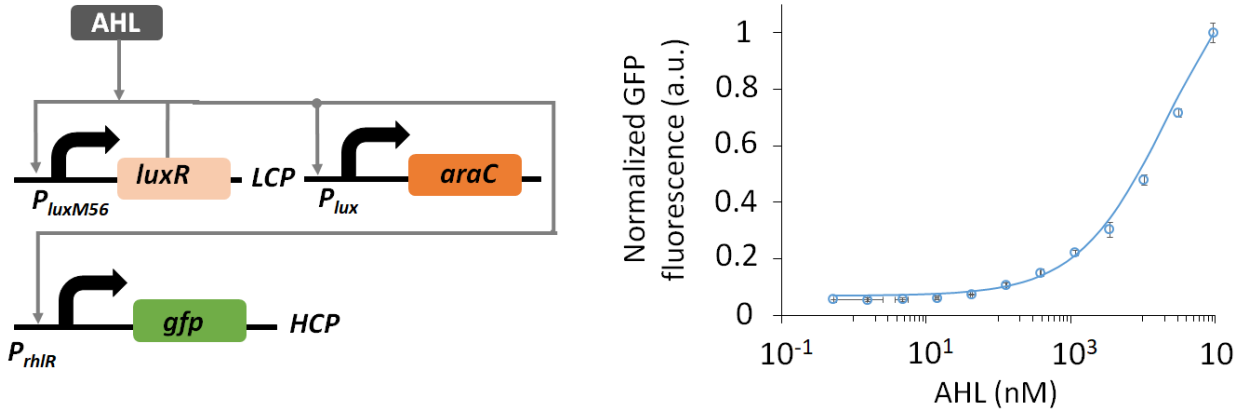


Fig. S7.13. AHL -GFP transfer function of P_{rhlR} promoter. Solid line indicates the fitting results of the empirical models (Eq. S7.44).

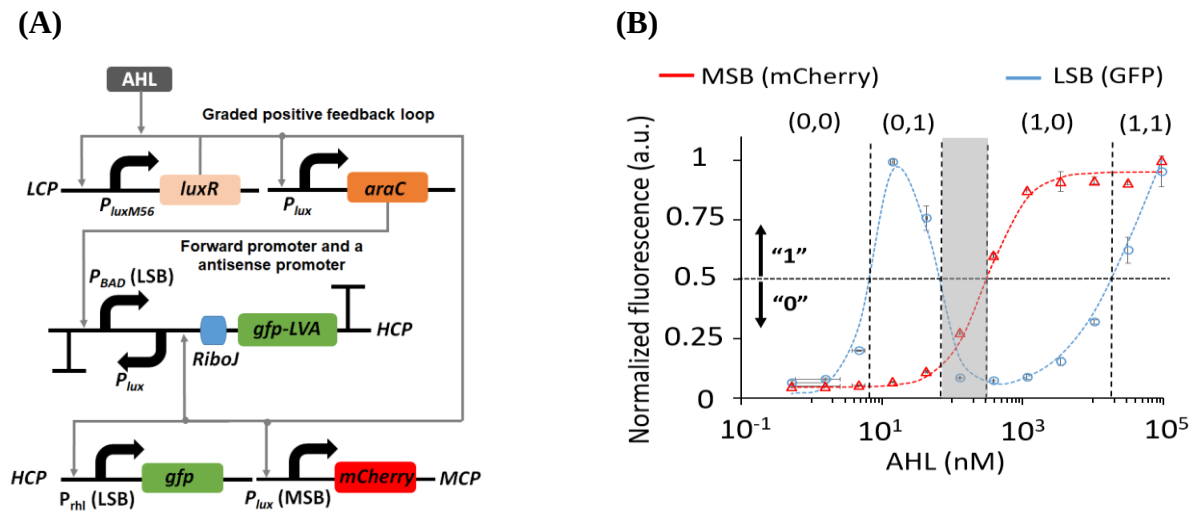


Fig. S7.14. (A) Implementation of 2-bit hybrid ADC. (B) the normalized GFP and mCherry signals of 2-bit ADC. Solid lines indicate fitting to the empirical models (Eq. S7.44, Eq. S7.47). We induced the circuits with Arabinose of 0.06 mM.

7.3.Programmable a simple logic gates based on perceptgene

Our simulation models show that by changing the bias of a single perceptgene, we can achieve different logic gates. As was shown in Section 2, the perceptgene simulations consists of two parts:

1. The power-law and multiplication function: The simulations are based on Eq. S2.18, Eq. S2.19, and Eq. S2.20. Parameters that were used in simulations:

Based on P_{lacO1} and P_{tetO} within ANF loop, and combinatorial promoter ($P_{lacO/tetO}$) – Fig. 1F

$$K_{m1} = 0.8, K_{m2} = 1, K_{d1} = 90, K_{d2} = 6, K_{d1h} = 45, K_{d2h} = 4, h_1 = 1, h_2 = 1.4, R_{max1} = 2000, R_{max2} = 3000, \boldsymbol{n_1} = \mathbf{1}, n_2 = 2, n_{1h} = 1, n_{2h} = 1, \theta = 1, \beta = 0.001$$

2. Activation function: here we used a similar activation function to P_{BAD}/ AraC (Eq. S2.21, Section 2.3), but without a repression element. We also assumed that there are several binding sites of AraC in the promoter P_{BAD}. Therefore, the P_{BAD} activation function can be described as:

$$P = \frac{\left(\frac{AraC_c}{K_{d3}}\right)^{n_3} + \beta_4}{1 + \beta_4 + \left(\frac{AraC_c}{K_{d3}}\right)^{n_3}} \quad \text{(S7.48)}$$

Where: $\text{AraC}_T = 30,\ K_{m3} = 0.09, \beta_4 = 0.002, n_3 = 1.5,$
For the OR gate, we used: $K_{d3} = 2,$
For the OR gate, we used: $K_{d3} = 30,$

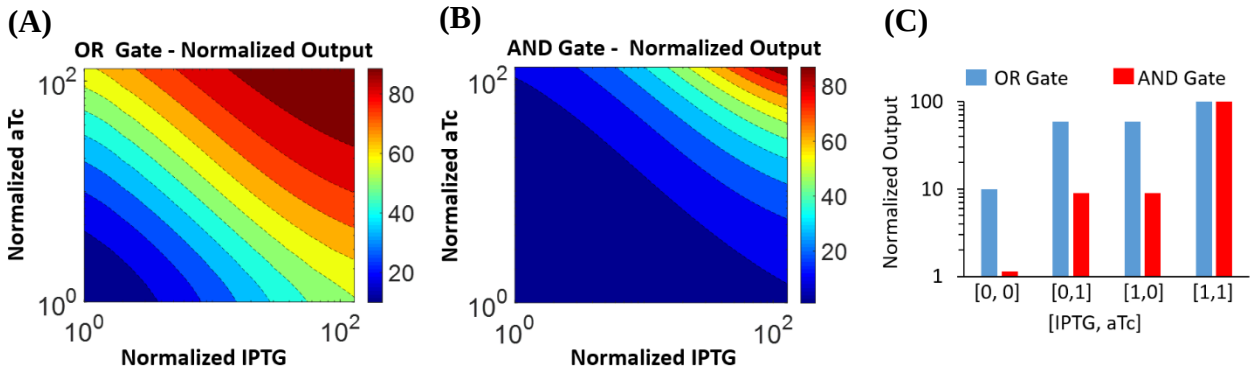


Fig. S7.15. Simulation results of a single perceptgene for simple logic gates (A) OR logic gate: $K_{d3} = 2$, (B) AND logic gate $K_{d3} = 30$. (C) shows the data as logic states, "0" ➔ Normalized inducer level=1, "1" ➔ Normalized inducer level=128.

7.4 Design and implementation of ternary data converter (switch)

From a dynamic system point of view, changing model parameters can lead to qualitatively different patterns of steady states. We therefore explored the model parameters (Eq. S7.47), by varying the ratio between repression and thresholding terms in Eq. S7.47 (K_2 Vs K_3), or by controlling the weights (Fig. S7.3E). Interestingly we obtained a new behavior of the LSB circuit (Fig. S7.16). We demonstrated this behavior by controlling different levels of Arabinose. The Arabinose concentration controls the repression as shown in Eq. S2.21, meaningly, , In high Arabinose concentrations, the AraC acts only as an activator without repression. Because AraC is regulated by P_{lux}, we can assume that AraC $\propto$ MSB. The experimental resulted AHL -GFP transfer function using the new design is a three-valued logic (ternary logic) as shown in Fig. S7.17. These results demonstrate that ternary was achieved using neural networks (Fig. S7.3E). The data of ternary data converter is well fitted by our empirical models (Eq. S7.44, Eq. S7.47), with a set of model parameters:
MSB: $r_1 = 1.5, K_1 = 450, \beta_1 = 0.005$
LSB: $r_0 = 1.6, K_0 = 28,\ \beta_0 = 0.02,\ \alpha = 200, r_2 = 1.2, K_2 = 1, r_3 = 1.4, K_3 = 300, \beta_2 = 0.02$
Ternary genetic circuits, converts analog signals to fuzzy levels, may find new applications in biotechnology, such as allowing engineers to tune the expression level of toxic proteins, enzymes in metabolic pathways in a reliable way. Furthermore, such systems can useit in building biosensors, which able to report in three states: low, medium and high. By contrast, the digital circuits can report only in two states.

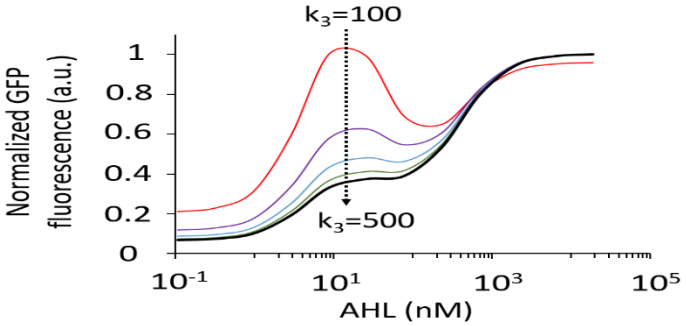


Fig. S7.16. Simulation results of the influence of repression K_3 versus thresholding k_2 on LSB circuit. Modeling parameters: MSB: $r_1 = 1.5, K_1 = 450, \beta_1 = 0.005$, LSB: $r_0 = 1.6, K_0 = 28, \beta_0 = 0.03, \alpha = 160, r_2 = 1.2, K_2 = 1, r_3 = 1, \beta_2 = 0.04$.

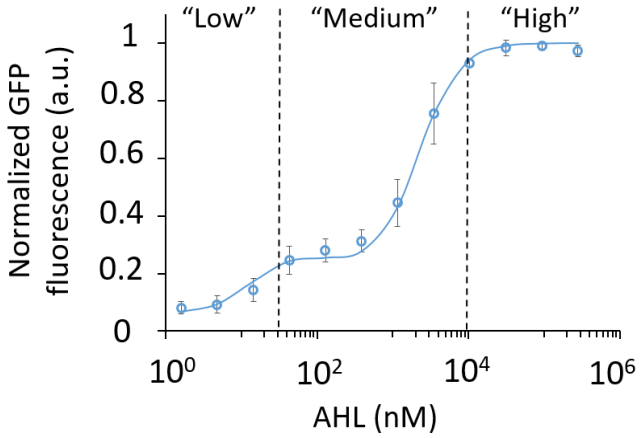


Fig. S7.17. Implementation of ternary data converter, based on the regulation of repression versus thresholding. Experimental and modeling (Eq. S7.47) results of ternary circuit. Solid lines indicate modelling results of the empirical models (Eq. S7.44, Eq. S7.47). We induced the circuits with high Arabinose of 50 mM compared to Fig. S7.13 with 0.4 mM.

7.5. Reconfigurable perceptgene-based logic networks

To demonstrate the computational efficiency of perceptgene design, we modified the 3-input majority circuit by replacing the P_{tetO} promoter with $P_{luxM56/tetO}$ combinatorial promoter as shown in Fig. S7.18A. The effect of AHL input on the GFP signal is collectively integrated by the $P_{luxM56/tetO}$ and $P_{luxM56/lacO1}$prompters. Otherwise, the $P_{luxM56/tetO}$ combinatorial promoter acts as a logical conjunction operation rather than an integrative operation, which means it is active only if the AHL and IPTG are "1". The new network architecture allows AHL to exert more reliable effect on GFP by affecting both the AraC branch and the SupD branch of the network. The biophysical model that describes the new genetic circuit is based on the equation set Eq. S5.7-Eq. S5.11, with a modification of y_2:

$$y_1 = \left(B_1 \cdot \left(\frac{AHL}{K_{m1}}\right)^{n_1} \cdot \left(\frac{IPTG}{K_{m2}}\right)^{n_2}\right)^{m_1} \quad \text{(S7.49.1)}$$

$$z_1 = \frac{y_1+\beta_1}{1+\beta_1+y_1} \quad \text{(S7.49.2)}$$

$$y_2 = \left(B_2 \cdot \left(\frac{aTc}{K_{m3}}\right)^{n_3} \left(\frac{AHL}{K_{m1}}\right)^{n_1\prime}\right)^{m_2} \quad \text{(S7.49.3)}$$

$$y_3 = (B_4 \cdot y_2 \cdot (B_3 \cdot z_1)^{m_3})^{m_4} \quad \text{(S7.49.4)}$$

$$z_2 = \frac{y_3+\beta_2}{1+\beta_2+y_3} \quad \text{(S7.49.5)}$$

Where $B_4 \equiv \frac{y_{m3}}{K_5} \cdot \frac{K_3}{K_4}$, $B_3 \equiv \frac{z_{m1}}{K_3}$, $B_2 \equiv \frac{y_{m2}}{K_2}$, $B_1 \equiv \frac{y}{a} \cdot {m_1}^b$, $B_5 \equiv B_4 \cdot {B_2}^{m_2} \cdot {B_3}^{m_3}$

Eq. S7.49.4 and Eq. S7.49.5 show that the network consists of two layers (Fig. S7.18B). We used parameters consistent with the previous majority function model, except that $n_3 = 0.6$ (compared with $n_3 = 0.7$, because the effective Hill-coefficient of aTc on P_{tetO} is slightly different from $P_{lux/tetO}$ promoter). Our model accurately captures the behavior of our new circuit (Fig. S7.18C). We used different Hill-coefficients of AHL -LuxR for $P_{lux/tetO}$ and $P_{lux/lacO}$; ${n_1}' = 0.65$, $n_1 = 1$. The results of the new circuit show that the [1,0,0] state gave a "1", by contrast to 3-input majority function that gave "0". Because the AHL was collectively integrated by $P_{luxM56/tetO}$ and $P_{luxM56/lacO1}$ prompters.

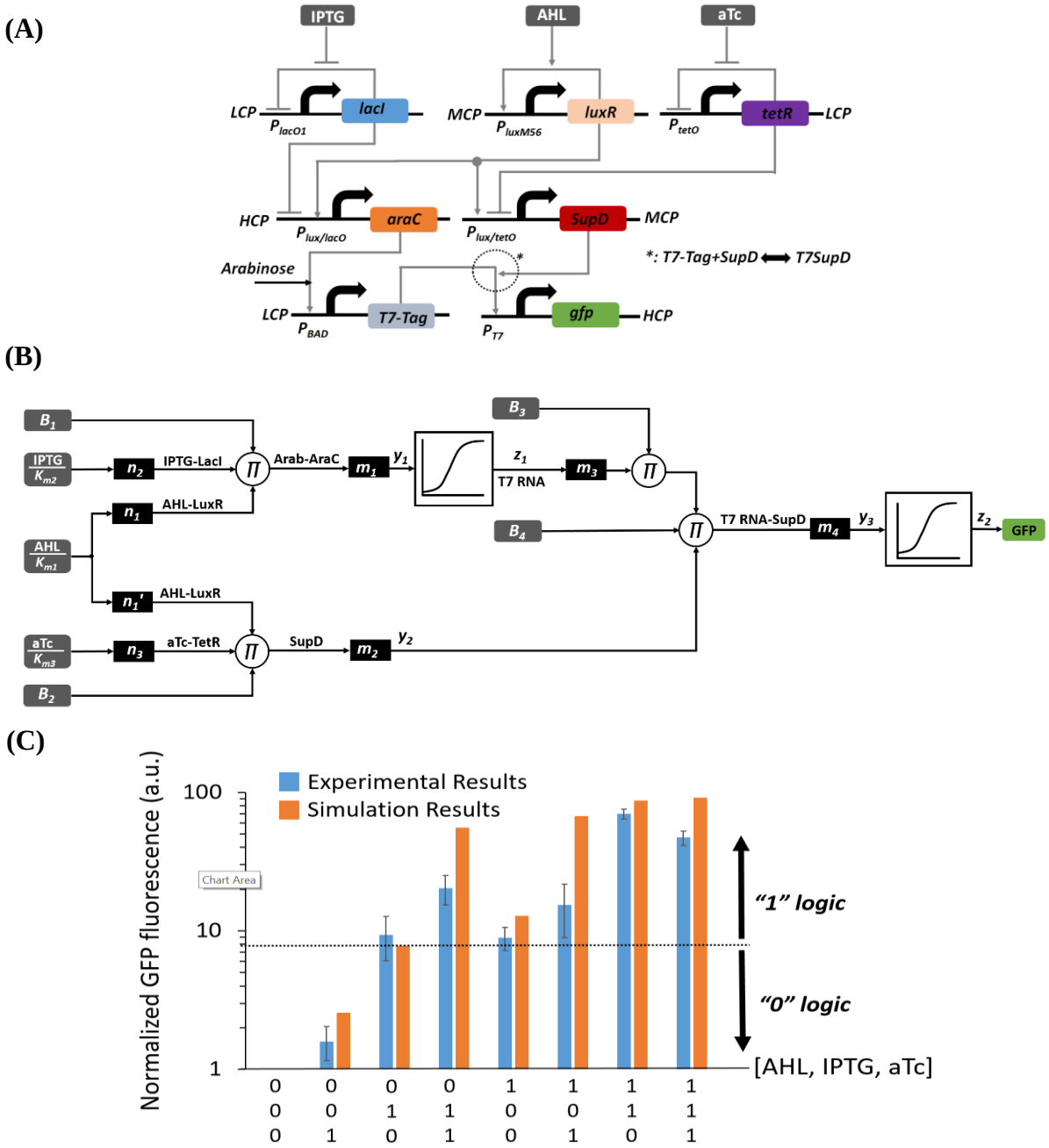


Fig. S7.18. (A) Multilayer perceptgene displays a new logic function for three inputs (AHL, IPTG and aTc). (B) Abstract model of the new multilayer perceptgene network. (C) Experimental and simulation results.

Table S7.1 List of parameters used in this section

Symbol	Description
X	Input
Z_i	Output
y_i	Analog signals
n_i	Weights or Hill Coefficients
A_i	Biases
k_{f0}	The rates for the forward reactions from Z_0 to Z_0*
k_{f1}	The rates for the forward reactions from Z_1 to Z_1*
k_{r0}	The rates for the corresponding backward reactions Z_0* to Z_0
k_{r1}	The rates for the corresponding backward reactions Z_1* to Z_1
k_{10}	The regulation of Z_1* on the activation of Z_0
Z_{T0}	The total concentration of molecules Z_0
Z_{T1}	The total concentration of molecules Z_1
K_{n0}, K_{n1} *and* K_{m1}	Dissociation constants
P_i	The reaction activity is defined as ratio of the product and the total concentration of molecules
P	Probability that RNA polymerase is bound to the forward promoter at the equilibrium
β	Basal level
Y_i	The concentration of inducer-TF complex
K_{di}	Dissociation constants
θ	Interference
Arab	Free arabinose concentration
$AraC_T$	The total concentration of $AraC$
AHL	Free N-(β-Ketocaproyl)-L-homoserine Lactone $3OC_6HSL$ concentration
$LuxR_T$	The total concentration of $LuxR$
β_{eff}	Effective basal constant
K_{deff}	Effective dissociation constant
ρ	Fitting parameter
A	Activator
B	Transcription factor
r_i	Hill Coefficients
K_i	Effective dissociation constant
β_i	Promoter basal level
α	Fitting parameter

Table S7.2 List of abbreviations used in this section

Symbol	Description
ADC	Analog to digital convertor
DAC	Digital to analog convertor
LSB	Last Significant Bit
MSB	Most Significant Bit
IDR	Input dynamic range
Arab	Free arabinose concentration
AHL	Free N-(β-Ketocaproyl)-L-homoserine Lactone $3OC_6HSL$ concentration
P_{BAD}	AraC promoter is activated by the $AraC$ when it is induced by arabinose (Arab)
P_{luxM56}	Mutated LuxR promoter is activated by the $LuxR$ when it is induced by AHL
P_{const}	Constitutive promoter
P_{lux}	LuxR promoter is activated by the $LuxR$ when it is induced by AHL
TetR	Concentration of TetR
$P_{lux/tetO}$	Combinatorial promoter
P_{rhlR}	Quorum sensing promoter that interacts with AHL inducer
LVA	ssrA degradation tag

8. Design principles of neuromorphic gene circuits

Weights and biases in neuromorphic circuits are determined by several factors, including Hill coefficients of small molecule inducers that serve as perceptgene inputs, the number and sequence of transcription factor binding sites, regulation of negative feedback strength (Fig.1-2, Fig. S8.3), regulation of incoherent feedforward strength (Fig. S8.5), transcription factor sequestration via protein-protein interactions (Fig. S8.6), transcription factors that competitively inhibit expression via steric hinderance (Fig. S8.7), operator sequence that controls binding affinity of transcription factor in open loop and positive feedback (Fig. S8.8-Fig. S.8.10), activation via RNA-protein interactions (Fig. 3), and protein structure (*e.g.*, dimerization and cooperativity), and circuit topology. Of particular importance, we demonstrate modulation of activation function weight m_1 for the majority function via administration of various Arabinose levels (Fig. S6.2A and Fig. 3E, F). Specifically, we induce the system with eight different Arabinose concentrations and obtain fine-grain control of AraC-Arabinose weight, allowing continuous control of the system. The process of affecting weights and biases begins with a hypothesis of modulating the dosage response (e.g. transfer function) of a regulatory element. This is inspired by what has already been demonstrated in the literature, by a new approach that builds upon existing knowledge, or with completely new innovative methods. After implementing the circuit modifications, the new transfer functions are evaluated to determine the resultant weight and bias.

The first step toward the design of neuromorphic gene circuits is to understand the nature of the input molecules and determine their computing weights. In neuromorphic gene circuits, the computing weights of small molecules can be controlled by modifying the log domain slope of a regulated promoter's dosage response curve and can be characterized with Hill coefficients (e.g., the number of identical inducers that bind to transcription factors, and cooperativity of transcription factors). Regulatory topologies such as a negative feedback loop and an incoherent feedforward loop provide additional strategies that can be used to program the computing weights of small molecules and proteins. A mathematical model for the open-loop circuit, shown in Fig. S8.1A, describes an input (In) that inhibits the activity of repressor R, which in turn represses the output. The production of R is constitutive and can be expressed as follows

$$Out = \frac{\alpha \cdot \tau}{1+\left(\frac{R}{K_d}\right)^n} \quad \text{(S.8.1.1)}$$

$$R = \frac{R_T}{1+\left(\frac{In}{K_m}\right)^h} \quad \text{(S.8.1.2)}$$

Where α is output production rate, τ is protein half-life, K_d is binding dissociation constant of the repressor (R) to the output, R_T is the total concentration level of R, K_m is binding dissociation constant of input (In) to R, n and m are Hill-coefficients of In and R. While the repressor R level is constant in the open-loop circuit, it is regulated by the output protein in the negative feedback circuit. A mathematical model for the auto-negative feedback loop circuit (Fig. S8.1B) is given by:

$$Out_T = \frac{\alpha \cdot \tau}{1+\left(\frac{Out}{K_d}\right)^n} \quad \text{(S8.2.1)}$$

$$Out = \frac{Out_T}{1+\left(\frac{In}{K_m}\right)^h} \quad \text{(S8.2.2)}$$

Simulation results that compare the characteristic and computing weights of the open loop and auto-negative feedback circuits are shown in Fig. S8.1C. By programming the strength of the auto-negative feedback loop, one can obtain fine grain control over the input weights (Fig. S8.1D). In this work, we varied the number of binding sites for transcription factors in the promoter to control the strength of auto-negative feedback (Fig. 1).

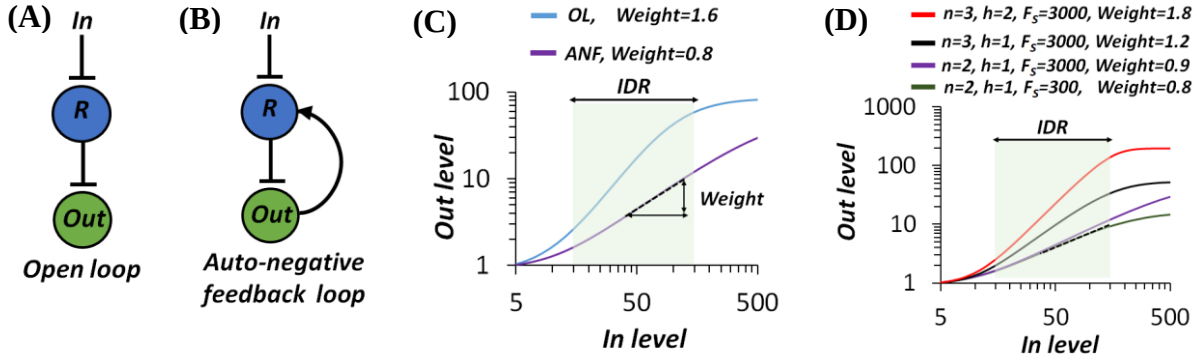


Fig. S8.1. (A) Open loop design. (B) Auto-negative feedback design. (C) Simulation results for open loop and auto-negative feedback loop circuits. Simulation parameters: $K_m = 10, h = 2, K_d = 1, R_T = 100, \alpha \times \tau = 3000$. (D) Simulation for the auto-negative feedback circuit, where the feedback loop strength $F_s = \alpha \times \tau / K_d$.

In order to program the input weights continuously within a range, we split the auto-negative feedback loop into two reactions, one is the feedforward ($R \rightarrow Out$) loop and second is the negative feedback loop that is controlled by small molecule inducer x ($A \rightarrow R$). A mathematical model describing such a system is given by:

$$A_T = \frac{\alpha_1 \cdot \tau}{1+\left(\frac{R}{K_{d1}}\right)^{n_1}} \quad \text{(S8.3.1)}$$

$$R = \frac{R_T}{1+\left(\frac{In}{K_{m1}}\right)^{h}} \quad \text{(S8.3.2)}$$

$$R_T = \alpha_2 \cdot \tau \frac{\left(\frac{AX}{K_{d2}}\right)^{n_2}}{1+\left(\frac{AX}{K_{d2}}\right)^{n_2}} \quad \text{(S.8.3.3)}$$

$$AX = A_T \frac{\left(\frac{x}{K_{m2}}\right)^{m}}{1+\left(\frac{x}{K_{m2}}\right)^{m}} \quad \text{(S.8.3.4)}$$

The simulation results show that the level of inducer (x) can control the input weight by regulating the strength of the negative feedback (Fig. S8.2).

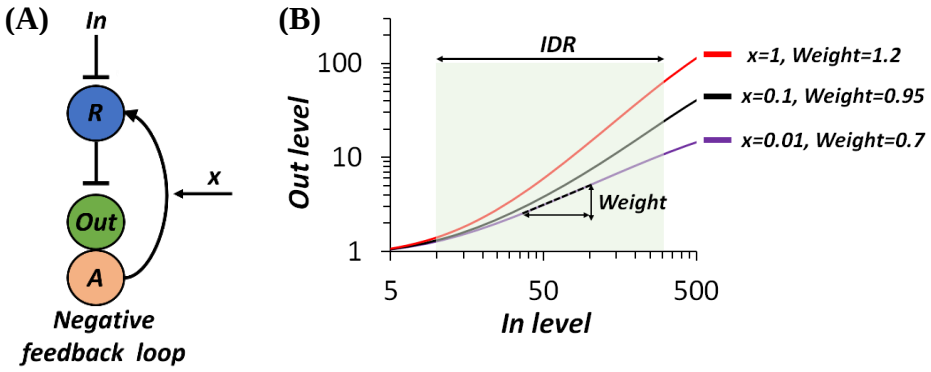
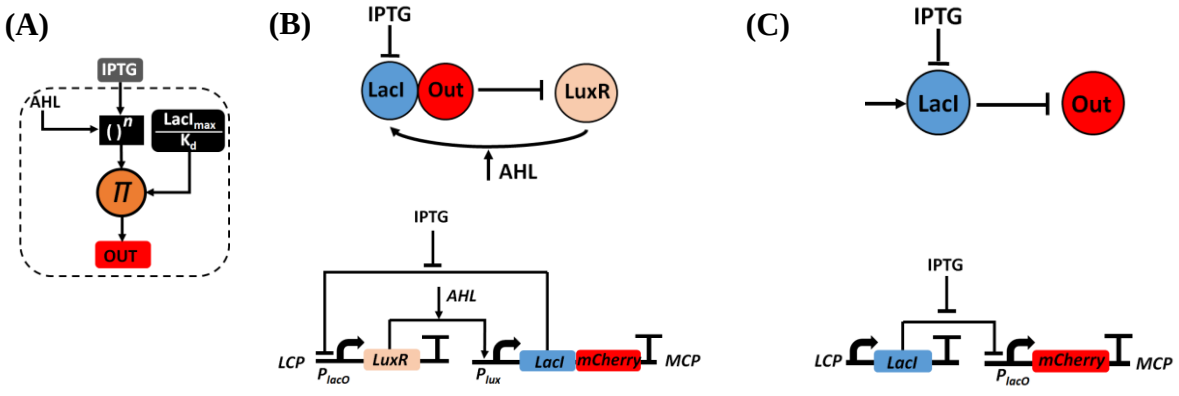


Fig. S8.2. (A) Negative feedback design based on splitting the feedforward and feedback by using different proteins. (B) simulation results for the negative feedback loop. Simulation parameters: $K_{m1} = 10, h = 1.5, K_{d1} = 1, \alpha_2 \times \tau = 1000, n_1 = 1, K_{m2} = 1, m = 1.5, K_{d2} = 10, \alpha_1 \times \tau = 1000, n_2 = 1.$

In Fig. S8.3 we describe small molecule control of negative feedback regulation via transcription factor activation of repressor. This design allows us to continuously program the weight of IPTG by changing the level of AHL. First, we compared the negative feedback circuit (Fig. S8.3B) with an open-loop circuit (Fig. S8.3C). The open loop circuit includes regulation of mCherry by P_{lacO} promoter that is induced by IPTG. While LacI is constitutively expressed in the open loop circuit, it is regulated by the P_{lux} promoter in the negative feedback circuit. Input IPTG regulates the activity of promoter P_{lacO} and expression of LuxR. The LuxR/AHL complex regulates LacI levels, which represses promoter P_{lacO}, creating a negative feedback loop. Fig. S8.3D shows experimental results of open loop and negative loop circuits, in agreement with our theoretical results (Fig. S8.1). Furthermore, this negative feedback loop design allows us to continuously program the weight of IPTG by changing the level of AHL (Fig. S8.3E). The negative feedback loop strength, which is controlled by AHL, determines the IPTG input weight (Fig. S8.3F). These experimental results are consistent with our simulation results (Fig. S8.2).



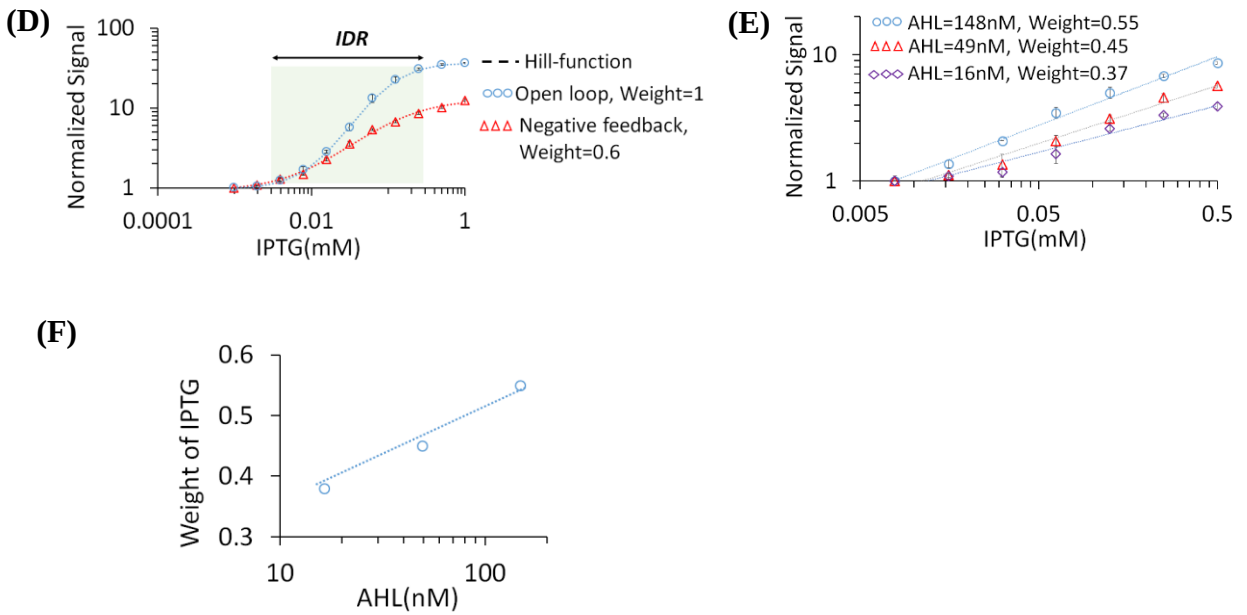


Fig. S8.3. (A) Programmable perceptgene input weight. IPTG is the input and AHL regulates IPTG weight. (B) High level circuit diagram and genetic circuit implementation of programmable perceptgene input weight based on a negative feedback. AHL binds LuxR, forming a complex that controls the strength of the negative feedback loop. When IPTG binds LacI, it induces promoter P_{lacO} activity, increasing LuxR levels. AHL binds LuxR, forming a complex that regulates expression of LacI. LuxR and GFP are regulated by P_{lacO} promoter. P_{lux} is encoded on MCP, while P_{lacO} is encoded on LCP. (C) High level circuit diagram and genetic circuit implementation of open loop circuit. (D) Experimentally measured IPTG/GFP transfer function of open loop and negative feedback circuits (AHL = 0.1 mM). (E) Experimentally measured IPTG/GFP transfer function under three different AHL concentrations. (F) IPTG input weight is shown as a function of AHL concentration.

Incoherent feedforward loops can also be used to program the weights of small molecules and proteins. In these networks, the upstream regulator (A) directly activates the target gene (Out) and indirectly represses it by activating repressor (R) of the target gene (Fig. S8.4A). In our design, we assumed that the upstream regulator (A) is induced by the input (In). A mathematical model for such a system is given by:

$$InA = \alpha_1 \cdot \tau \frac{\left(\frac{In}{K_m}\right)^h}{1+\left(\frac{In}{K_m}\right)^h} \quad \text{(S8.4.1)}$$

$$R_T = \alpha_2 \cdot \tau \cdot \frac{\left(\frac{InA}{K_{ar}}\right)^n + \beta}{1+\left(\frac{InA}{K_{ar}}\right)^n} \quad \text{(S8.4.2)}$$

$$Out = \alpha_3 \cdot \tau \cdot \frac{\left(\frac{InA}{K_a}\right)^n + \beta}{1+\left(\frac{AX}{K_a}\right)^n} \cdot \frac{1}{1+\left(\frac{R}{K_r}\right)^m} \quad \text{(S8.4.3)}$$

Where α_i are protein production rate, τ is protein half-life, K_a, K_{ar} and K_r are binding dissociation constants of the regulator (A) and repressor (R) to the output, K_m is binding dissociation constant of input (In) to A, n and m are Hill-coefficients and β is the basal level. The simulation results of

incoherent feedforward circuit are shown in Fig. S.8.4B. These results indicate finely tunable weight with positive and negative values.

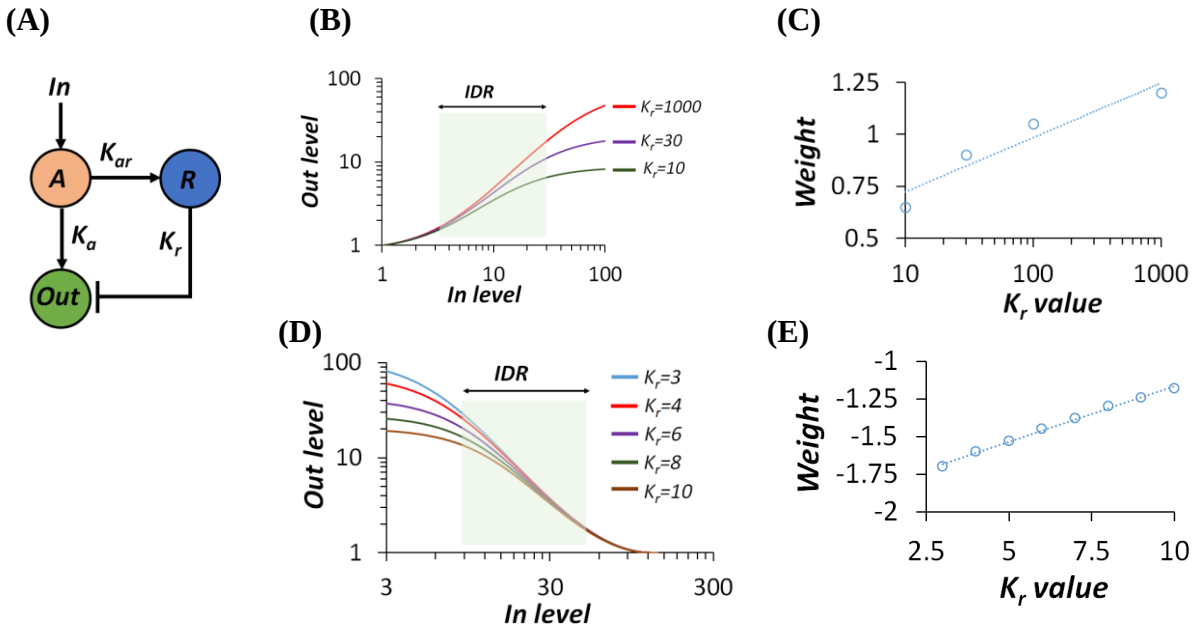
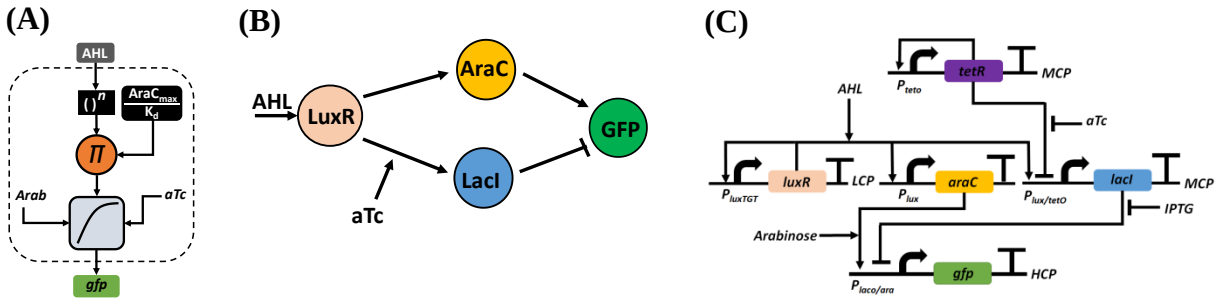


Fig. S8.4. (A) Design of incoherent feedforward loop. (B) Design of programmable perceptgene with a positive weight, simulation results with parameters: $K_{m1} = 100, h = 1.5, K_a = 10, K_{ar} = 10, \alpha_1 \times \tau = 100, n = 1, \beta = 0.01, m = 1, \alpha_2 \times \tau = 100, K_r = 10, 30, 1000$ (C) Design of programmable perceptgene with a negative weight, simulation results with parameters: $K_{m1} = 100, h = 1.5, K_a = 100, K_{ar} = 1, \alpha_1 \times \tau = 10, n = 1, \beta = 0.005, m = 1.5, \alpha_2 \times \tau = 150, K_r = 3 - 10$.

In Fig. S8.5 we demonstrate experimentally small molecule control of transcription factor competitive inhibition via binding to an output promoter. This design allows us to modulate the weight of input AHL continuously by changing the aTc level (Fig. S8.5E). The input AHL binds LuxR and forms a complex that induces expression of activator (AraC) and repressor (LacI), which combine to regulate GFP output, resulting in an incoherent feed-forward loop. Small molecule aTc controls LacI expression via de-repression of TetR, which in turn affects the overall AHL-GFP transfer function. We show the resulting input weight as a function of aTc relevant for the input dynamic range. Our incoherent feed-forward circuit provides negative weights.



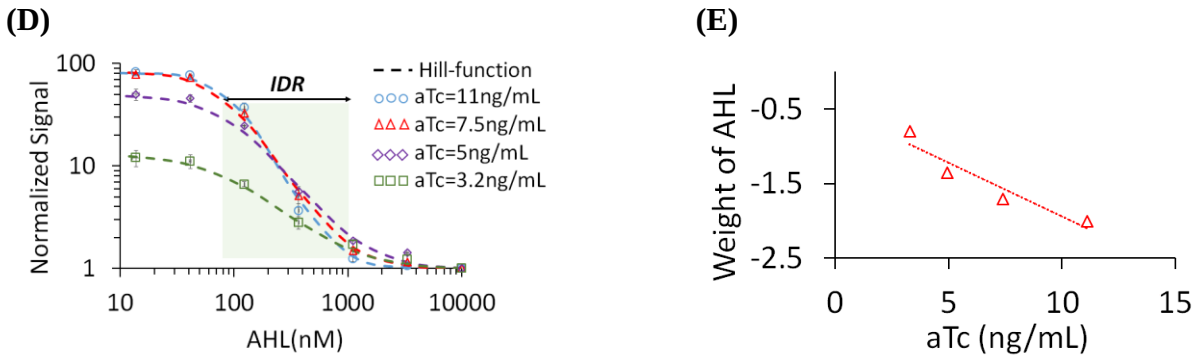


Fig. S8.5. (A) A programmable percentgene with single input weight. aTc is used to modulate the weight of AHL continuously. (B) High level circuit design. Input AHL binds LuxR and forms a complex that regulates AraC and LacI. The AraC transcription factor activates GFP output expression, while the LacI transcription factor represses GFP expression. The activation function is determined by the AraC/LacI interaction, where LacI expression is controlled by aTc, and hence impacts input weight. (C) Genetic circuit implementation. P_{tetO} promoter is regulated by TetR through an auto-negative feedback loop and induced by aTc. P_{TGT} and P_{lux} promoters are regulated by LuxR through a positive feedback loop and induced by AHL. AraC is regulated by P_{lux} promoter, and LacI is regulated by combinatorial $P_{lux/tetO}$ promoter. A ssrA degradation tag (LVA) was added to LacI to reduce the maximum protein level. GFP is regulated by AraC/LacI through activation/repression of combinatorial $P_{ara/lacO}$ promoter. The feedback loops in this circuit increase the input dynamic ranges of AHL and aTc. The P_{tetO} and combinatorial $P_{lux/tetO}$ promoters are encoded on a medium-copy-number plasmid (MCP). The combinatorial $P_{ara/lacO}$ promoter is encoded on a high-copy-number plasmid (HCP). The P_{lux} and P_{TGT} promoters are encoded on a low-copy-number plasmid (LCP). (D) Measured AHL - GFP transfer function where aTc is varied ($\text{aTc} = 11, 7.5, 5, 3.2\ ng/_{ml}$, $\text{Arabinose} = 50\ \text{mM}, \text{IPTG} = 1\ \text{mM}$). The dotted lines are Hill-function fitting with $GFP \propto \frac{\left(\frac{AHL}{K_{eff}}\right)^{h_{eff}} + \beta}{1+\left(\frac{AHL}{K_{eff}}\right)^{h_{eff}}}$:

(1) $\text{aTc} = 11\ ng/_{ml}$, $h_{eff} = 2.2$, $K_{eff} = 100\ mM$, $\beta = 0.013$,
(2) $\text{aTc} = 7.5\ ng/_{ml}$, $h_{eff} = 1.9$, $K_{eff} = 85\ mM$, $\beta = 0.012$,
(3) $\text{aTc} = 5\ ng/_{ml}$, $h_{eff} = 1.6$, $K_{eff} = 100\ mM$, $\beta = 0.02$,
(4) $\text{aTc} = 3.2\ ng/_{ml}$, $h_{eff} = 1.3$, $K_{eff} = 100\ mM$, $\beta = 0.08$,

(E) AHL weight based on the experimental results as a function of aTc.

The second step in the design of neuromorphic gene circuits is to aggregate the multiple inputs to one node in order to implement the multiplication function, which serves as a collective analog node. There are several biological mechanisms that can be used to accomplish such a function. For example, in this work combinatorial promoters ($P_{lacO/tetO}$, $P_{lux/lacO}$, $P_{lux/tetO}$) in Figures 1 and 2, and mRNA-protein interaction in Figure 3 was used to aggregate the analog weighted inputs and implement multiplication function. In Figure 4, we showed that transcriptional interference can also be used to aggregate inputs acting as division with negative weights.

The third step in the design of neuromorphic gene circuits is to add an activation function that converts the analog pattern of the multiple inputs into a non-linear function for performing analog

classification. This can be achieved by wiring the output of the multiplication circuit with an activator and promoter to regulate the perceptgene output.
In our neuromorphic genetic circuits, controlling the bias is perhaps easier than controlling the weights. The bias is determined by the ratio between the maximum protein expression level of the power-law/multiplication circuit output and the dissociation constant of transcription factor binding to DNA. The maximum protein expression level is determined by transcription rate, translation rate, mRNA and protein half-lives, and is given by:

$$\text{Bias} = \frac{\text{transcription rate}\times\text{mRNA half life}\times\text{translation rate}\times\text{protein half life}}{\text{dissociation constant of transcription factor binding to DNA}} \quad \text{(S8.5)}$$

In this study, we use different methods to control the maximum protein level such as promoter strength, ribosome binding site strength, ssrA degradation tag and plasmid copy number. We now show experimentally that it is possible to readily increase the dissociation constant of a transcription factor by controlling expression of a second biological element that competitively inhibits the transcription factor. To this end, we use two biological systems: (1) dCas9 regulation (Fig. S8.6), and (2) protein sequestration (Fig. S8.7).
In Fig. S8.6, we show competitive inhibition of gene activation via steric hinderance binding of DNA that is tuned by the DNA binding location of the dCas9/single guide RNA (sgRNA) complex. This design allows us to program bias continuously by choosing different sgRNA sequences. The dCas9 regulation system is built from two parts; a P_{BAD} promoter that activates the target gene by binding the Arabinose-AraC complex. The second part is the complex sgRNA-dCas9 which binds the AraC operator, and prevents the Arabinose-AraC complex from activating promoter P_{BAD}. The affinity of dCas9-sgRNA complex to its binding site and ability to sterically hinder transcription factor the promoter, control the binding dissociation constant of Arabinose-AraC complex to P_{BAD}. These factors, and hence perceptgene bias, can be readily controlled by building a library of sgRNA sequences (Fig. S8.6D-E). We used Hill-function (Eq. S8.6) to estimate the effective dissociation constant of Arabinose (K_{eff}):

$$GFP \propto \frac{\left(\frac{In}{K_{eff}}\right)^{h_{eff}}}{1+\left(\frac{In}{K_{eff}}\right)^{h_{eff}}} + GFP_0 \quad \text{(S8.6)}$$

Where In is the Arabinose concentration. In order to evaluate the bias based on the changes of dissociation constant, we fit a model that includes induction and activation to our experimental results. Such model can be given by:

$$P = \frac{\left(\frac{InA}{K_d}\right)^{n}+\beta}{1+\left(\frac{InA}{K_d}\right)^{n}} \quad \text{(S8.7.1)}$$

$$InA = A_{max} \cdot \frac{\left(\frac{In}{K_m}\right)^{h}}{1+\left(\frac{In}{K_m}\right)^{h}} \quad \text{(S8.7.2)}$$

Where K_d is the binding dissociation constant of inducer-activator (Arabinose-AraC) complex (InA) to promoter, β is the promoter basal level, K_m is the binding dissociation constant of inducer (Arabinose) to activator (AraC), A_{max} is the activator maximum level achieved by promoter, and h is Hill-coefficient of inducer. In this simple model, the bias is defined as $B = A_{max}/K_d$.

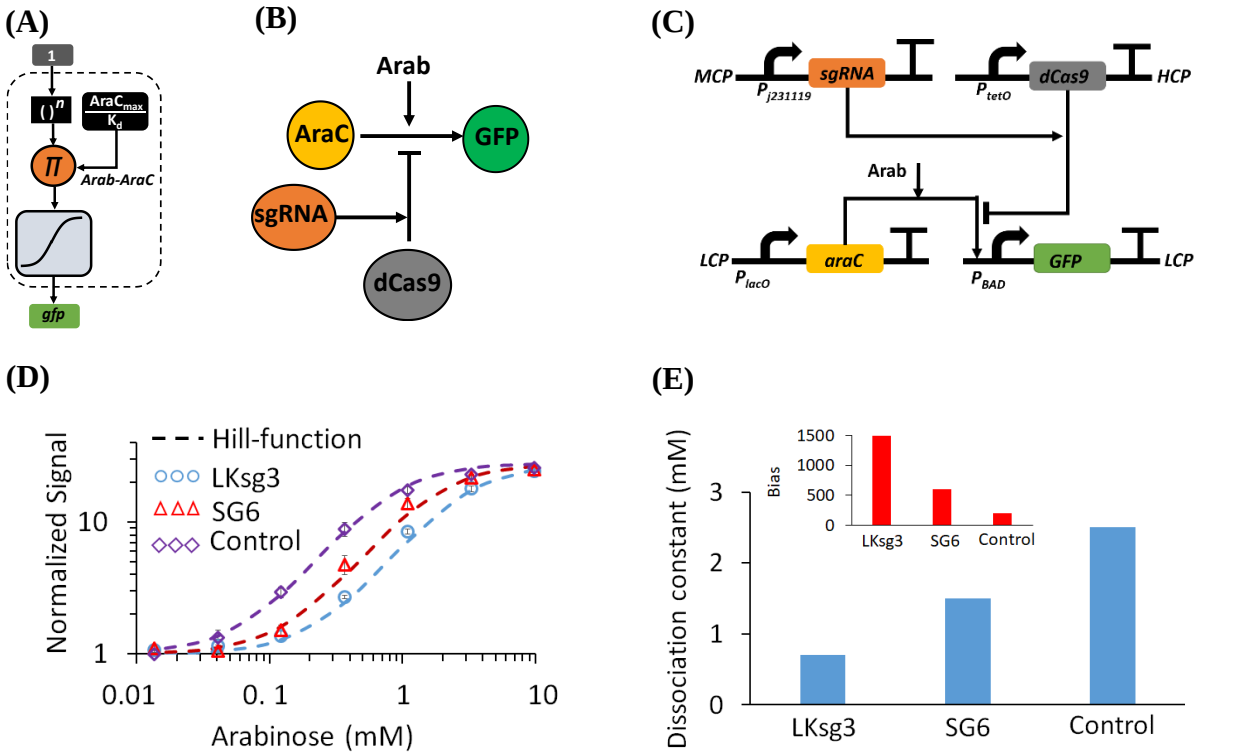


Fig. S8.6. (A) A percentgene with a constant input value of 1, allowing analysis of the activation function's programmable bias. P_{BAD} promoter serves as the activation function, and the AraC/Arabinose complex is an analog signal that modulates bias. (B) High level circuit schematics. The design is based on competitive inhibition of gene expression via a tunable dCas9/sgRNA complex. The sgRNA sequence determines the affinity of dCas9/sgRNA binding to AraC operator, and hence can modulate bias by preventing AraC/Arabinose activation of P_{BAD}, which results in an increase of the AraC/Arabinose dissociation constant. (C) Genetic circuit implementation. AraC, dCas9 and sgRNA are constitutively expressed by P_{lacO}, P_{tetO} and $P_{J231119}$ promoters. P_{BAD} promoter is encoded on a low-copy-number plasmid (LCP), dCas9 is encoded on a high-copy-number plasmid (HCP), and sgRNA is encoded on a medium-copy-number plasmid (MCP). (D) Experimentally measured transfer functions for three circuit variants encoding two different sgRNA sequences and a control (purple; without sgRNA and dCas9). SG6 targets the middle of P_{BAD} promoter, while LKsg3's target is at the end of the promoter. SG6: GACGCTTTTTATCGCAACTC; LKsg3: TTTTTTGGGCTAGCGAATT. The dotted lines are Hill-function fittings. $K_m = 90\,mM$, $h = 1.5$, $\beta = 0.035$, $n = 1$. (E) Arabinose dissociation constant and bias (inset) for all three circuit variants. $Bias = AraC_{max}/K_d$, where K_d is the dissociation constant of AraC/Arabinose complex, $AraC_{max}$ is the maximum AraC produced.

In Fig. S8.7, we demonstrate small molecule control of transcription factor sequestration (*48*) via protein-protein interactions. This design allows us to continuously program perceptgene bias using two heterologous proteins (ExsA and ExsD) where ExsA transcriptional activator is sequestered by ExsD into an inactive complex. Arabinose-AraC regulates the expression of ExsA that activates GFP output expression. aTc induces expression of anti-activator ExsD, which inhibits ExsA gene activation. Hence, the extent of the ExsA/ExsD protein-protein interaction and resultant perceptgene bias is controlled by aTc. We used Hill-function (Eq. S8.6) to estimate the effective dissociation constant of Arabinose and set of equations Eq. S8.7.1-2 to estimate the bias.

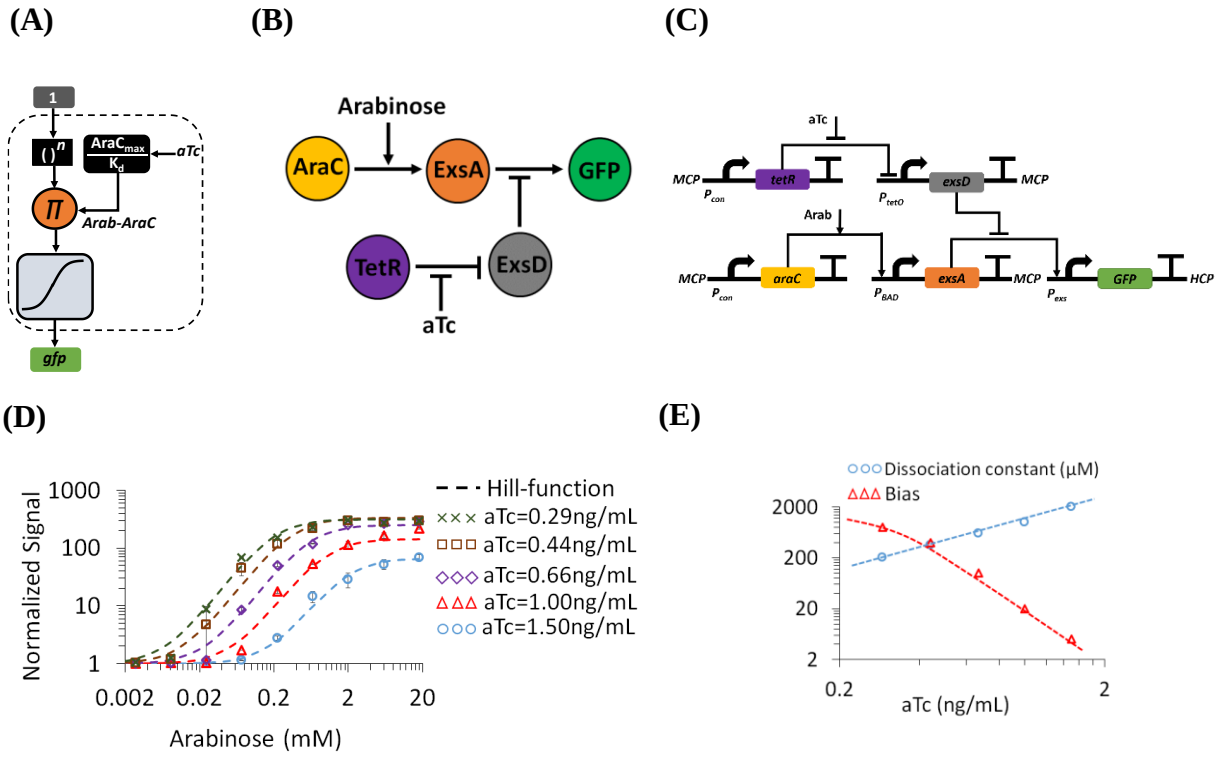


Fig. S.8.7. (A) A percentgene with a constant input value of 1, allowing analysis of the activation function's programmable bias. P_{exs} promoter serves as the activation function and the AraC/Arabinose complex is the analog signal. aTc level controls the activation function bias. (B) High level circuit diagram. The design is based on protein sequestration where ExsD shunts ExsA from activating GFP expression. This sequestration increases the dissociation constant of ExsA promoter binding and hence modulates bias. (C) Genetic circuit implementation. The ExsD- ExsA interaction that is used to regulate the activation function bias and is controlled via aTc. The AraC/Arabinose complex regulates expression of the ExsA activator. The TetR/aTc complex regulates expression level of anti-activator ExsD, which binds ExsA and inhibits its activation of P_{exs} promoter. AraC and TetR are constitutively expressed. P_{exs} promoter is encoded on HCP while the other promoters are encoded on MCP. (D) Experimentally measured Arabinose transfer functions under different aTc conditions. The dotted lines are Hill-function fittings, $K_m = 9\ mM$, $h = 1.5$, $\beta = 0.035$, $n = 1$. (E) The Arabinose dissociation constant and relative bias (inset) as a function of aTc.

Introducing random mutations to operator sequence of transcription factor can also be used to control the weights and bias. Fig. S8.8 describes modulations of transcription factor LuxR's DNA binding affinity via changes in Lux operator sequence. We introduced 7 random mutations into the first four nucleotides of the LuxR binding site. In order to precisely calculate the Hill-coefficient and effective dissociation constant, we simultaneously measured the activity of the P_{lux} promoters using GFP signal and measured the activity of the host cell using constitutive mCherry signal for each AHL level (Fig. S8.8A). With our open loop circuit topology, we experimentally measured

Hill-coefficient values ranging essentially continuously between 0.4 and 1 with a step of 0.1 (Fig. S8.8B and C). The AHL input weight is derived from the Hill-coefficient. Indeed, the input weight is the same as the Hill-coefficient when the basal level is very low, because both are equal to the slope at the log-log scale.

As we showed above, while the negative feedback can reduce the Hill-coefficient, here we show that positive feedback can increase the Hill-coefficient. This result matches other efforts that utilized auto-positive feedback (APF) to produce a sharp threshold in the response of inducer-promoter activity (*49*). Fig. S8.9 represent the experimental results of open and auto-positive feedback loops, from Fig. S2.13 at the logarithmic scale. The Hill-coefficient values were doubled when auto-positive feedback was used compared to open loop. We then incorporated other six Lux operator mutants into a Lux response circuit with positive feedback regulation, and obtained Hill-coefficient values ranging between 1.1 and 2 and computed weights between 0.75 and 1.7 (Fig. S8.10).

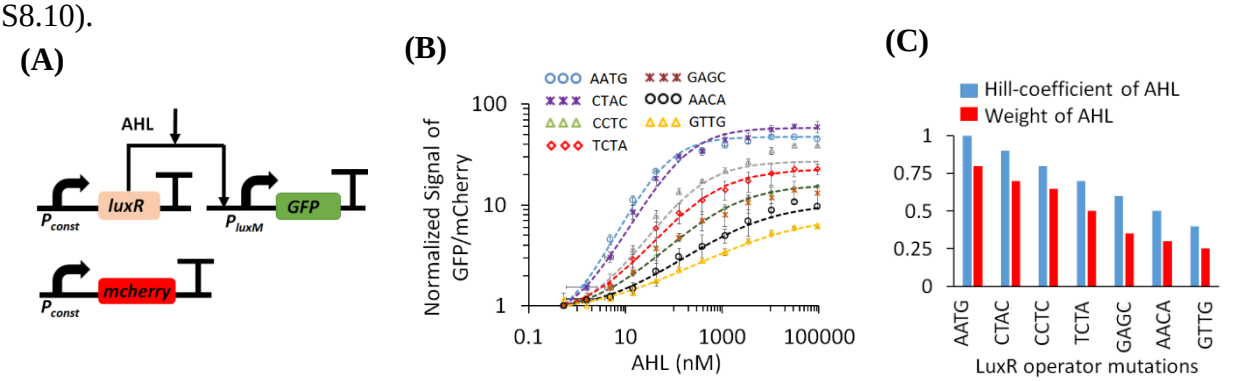


Fig. S8.8. (A) Circuit design for open loop followed AHL induction. LuxR is constitutively produced. LuxR/AHL binds Lux operator mutants within P_{lux} promoter and activates GFP expression. (B) Experimentally measured AHL-GFP transfer functions of the Lux operator mutants used to determine Hill coefficients of the AHL input. (C) Experimental data shows that random mutations in the first four bases of the Lux operator result in an essentially continuous range of Hill-coefficients throw AHL input and AHL input weights. The dotted lines are Hill-function fittings.

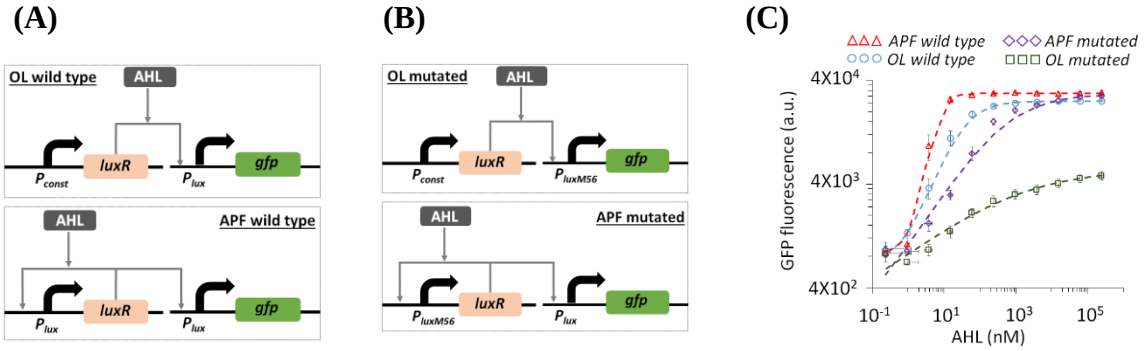


Fig. S8.9. Hill coefficients for a modified circuit that encodes auto-positive feedback regulation (APF), where LuxR is expressed by the same mutated P_{lux} promoters. (A) The construction of open loop (OL) and APF circuits based on P_{lux} promoter. (B) The construction of OL and APF circuits based on mutated P_{lux} promoter (P_{luxM56}). (C) Measured transfer functions of multiple circuits, dots are experimental data, and dashed-line is a Hill function fitting with the below parameters (See Fig. S2.13):

OL circuit – Wild type P_{lux}: $K = 30, m_{eff} = 1, a = 25 \times 10^3, b = 600$

APF circuit – Wild type P_{lux}: $K = 7, m_{eff} = 2, a = 30 \times 10^3, b = 800$

OL circuit – Mutated P_{luxM56}: $K = 500, m_{eff} = 0.3, a = 5 \times 10^3, b = 100$

APF circuit – Mutated P_{luxM56}: $K = 500, m_{eff} = 0.5, a = 30 \times 10^3, b = 100$

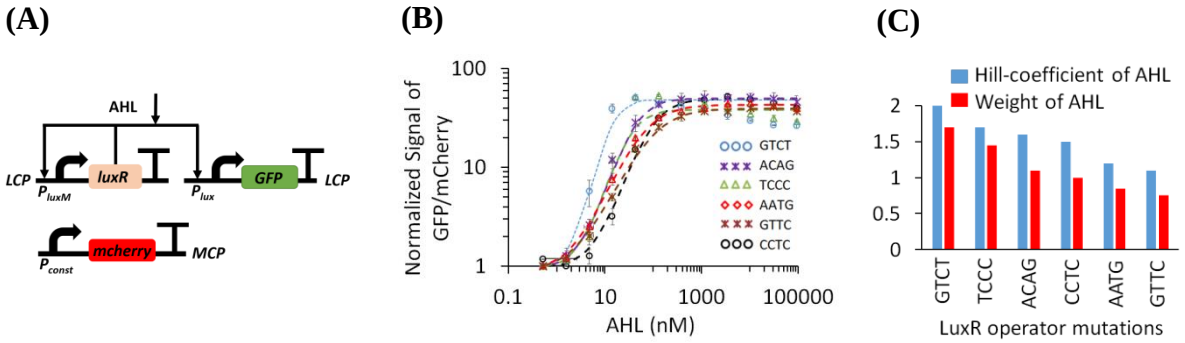
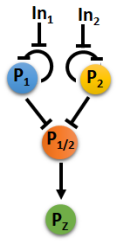
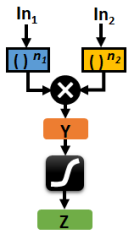
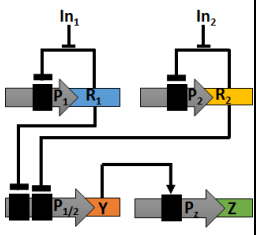
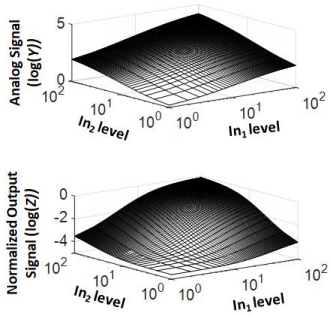


Fig. S8.10. (A) Circuit design for positive feedback AHL. LuxR is regulated by the mutated P_{luxM} promoter. LuxR/AHL binds mutant Lux operators within promoter P_{luxM} and activates LuxR expression, and also LuxR/AHL binds wild type Lux operators within promoter P_{lux} and activates GFP expression. (B) Experimentally measured AHL-GFP transfer functions of the Lux operator mutants used to determine the AHL input Hill coefficients. (C) Experimental data shows that random mutations in the first four bases of the Lux operator, resulting in an essentially continuous range of AHL input Hill-coefficients and AHL input weights. The dotted lines are Hill-function fitting.

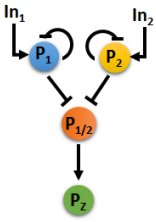
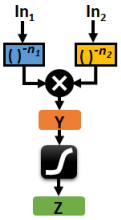
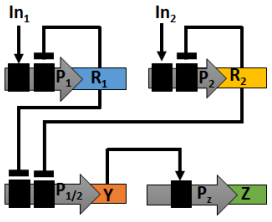
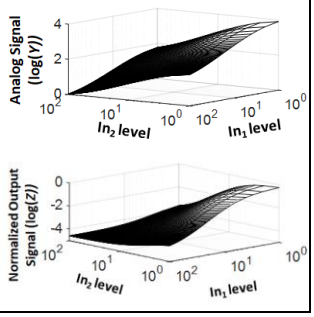
So far, we showed theoretically and experimentally that biological factors and design topologies determine weights and biases in our neuromorphic circuits. To provide a better view of design principles for neuromorphic gene circuits, we summarize and show below other examples (Table S8.1). Remarkably, we conclude that the same biological mechanism can be used to tune the bias and also to implement power law circuit. The table S8.1 starts with auto-negative feedback (ANF) loops (P_1, P_2), a dual repression node ($P_{1/2}$), and feedforward loop (P_z). The analog signal (Y) is represented by the activity of the dual repression node. The elements P_1 and P_2 are self-regulated and negatively induced by the inputs In_1 and In_2 respectively. The analog signal (Y) is combined with a nonlinear function to produce the output Z. This design has positive weights and is experimentally implemented using genetic components, as shown in Fig. 1. The design includes ANF loops consist of promoters that are regulated by repressors (R_1, R_2), (2) the inputs In_1 and In_2 are small molecules that inhibit the repressors activity, (3) the dual repression node is implemented by combinatorial promoter, and is also regulated by R_1 and R_2 and (4) the nonlinear activation function is realized by the regulation of the activator Y to P_z promoter. Our simulation results (Table S8.1A) show that the output of the combinatorial promoter can be described by the analog pattern at the logarithmic domain ($log(Y) = n_1 \cdot log(In_1) + n_2 \cdot log(In_2) + Cons$), and the activator Y with P_z promoter convert this analog behavior to non-linear pattern with two states, asymptotically. Essentially, the circuit makes a decision based on collective interaction of transcription factors with analog behavior through their binding to a combinatorial promoter. Table S8.1B shows our design to implement a perceptgene with negative weights. This design is slightly similar to the previous one, where P_1 and $P_2$are replaced by activation-repression (hybrid) nodes. Such nodes are directly activated by the inputs (In_1 and In_2) and self-repressed. We can implement

the activation-repression nodes in living cells using combinatorial promoters that are regulated by activators and repressors. The inputs can be small molecules or transcription factors. Table S8.1C shows our third design and it implements a perceptgene with negative and positive weights. This design is based on the previous design. Specifically, the dual repression node that regulates the collective analog signal (*Y*)was replaced by an activation-repression (hybrid) node. Perceptgenes can also be implemented by other biological mechanisms. For example, Table S8.1D shows that a perceptgene with negative and positive weights was implemented by protein sequestration, where activator and anti-activator pair is involved. Other examples are shown in Table S8.1E where a binding interaction between two sub-proteins can occur (*45*), and Table S8.1F where phosphorylation and dephosphorylation reactions in two-component signaling system (*50*) are involved. Lastly, antisense transcription (*40*), which occurs counter to gene orientation, can also be applied to implement a power-law function with a positive and negative weights (Table S8.1G).

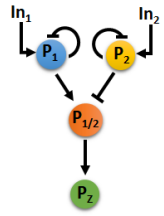
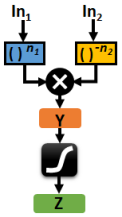
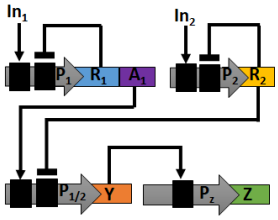
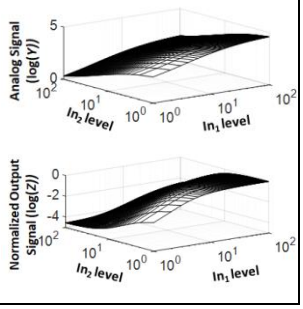
(A) Positive- weight perceptgene based dual repression system

Schematic design	Schematic block	Genetic design	Mathematical model at steady state	Simulation results
			$R_i = \frac{R_{Ti}}{\left(1+\left(\frac{In_i}{K_{mi}}\right)^{h_i}\right)}$ $R_{Ti} = \frac{\alpha_R \cdot \tau}{1+\left(\frac{R_i}{K_{di}}\right)^{n_i}}$ $Y = \frac{\alpha_Y \cdot \tau}{1+\left(\frac{R_1}{K_{d1}}\right)^{n_1}+\left(\frac{R_2}{K_{d2}}\right)^{n_2}+\left(\frac{R_1}{K_{d1}}\right)^{n_1}\cdot\left(\frac{R_2}{K_{d2}}\right)^{n_2}}$ $Z = \frac{\alpha_z \cdot \tau \cdot \left(\left(\frac{Y}{K_d}\right)^m+\beta\right)}{1+\left(\frac{Y}{K_d}\right)^m}$ $K_{mi}=1,\ h_i=1.5, \alpha_R \cdot \tau=300,\ K_{di}=1,$ $n_i=1,\ m=2.5,\ \alpha_Y \cdot \tau=100,\ K_d=30,\ \beta=0.01$	

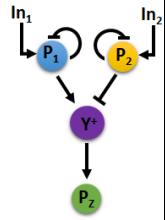
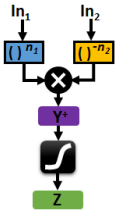
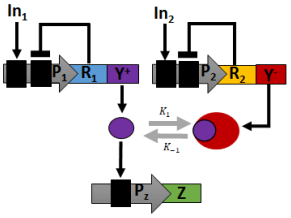
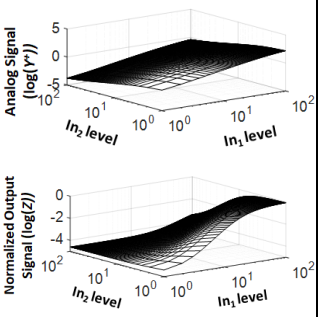
(B) Negative-weight perceptgene based dual repression

Schematic design	Schematic block	Genetic design	Mathematical model at steady state	Simulation results
			$R_i = \frac{\alpha_R \cdot \tau}{1+\left(\frac{R_i}{K_{di}}\right)^{n_i}} \cdot \frac{\left(\left(\frac{In_i}{K_{mi}}\right)^{h_i}+\beta_i\right)}{1+\left(\frac{In_i}{K_{mi}}\right)^{h_i}}$ $Y = \frac{\alpha_Y \cdot \tau}{1+\left(\frac{R_1}{K_{d1}}\right)^{n_1}+\left(\frac{R_2}{K_{d2}}\right)^{n_2}+\left(\frac{R_1}{K_{d1}}\right)^{n_1}\cdot\left(\frac{R_2}{K_{d2}}\right)^{n_2}}$ $Z = \frac{\alpha_z \cdot \tau \cdot \left(\left(\frac{Y}{K_d}\right)^m+\beta\right)}{1+\left(\frac{Y}{K_d}\right)^m}$ $K_{mi}=100,\ h_i=1.5, \alpha_R \cdot \tau=300,\ K_{di}=1,$ $n_i=1,\ m=2.5,\ \alpha_Y \cdot \tau=100,\ K_d=30,\ \beta=0.01$	

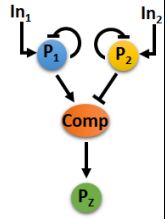
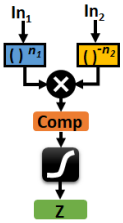
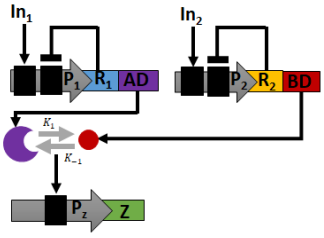
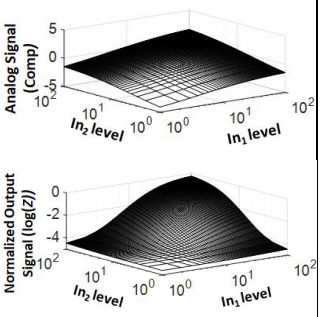
(C) Perceptgene based hybrid activation-repression system

Schematic design	Schematic block	Genetic design	Mathematical model at steady state	Simulation results
			$R_i = \frac{\alpha_R \cdot \tau}{1+\left(\frac{R_1}{K_{di}}\right)^{n_i}} \cdot \frac{\left(\left(\frac{In_i}{K_{mi}}\right)^{h_i}+\beta_i\right)}{1+\left(\frac{In_i}{K_{mi}}\right)^{h_i}} \quad A_1 = g_1 \cdot R_1$ $Y = \frac{\alpha_Y \cdot \tau \cdot \left(\left(\frac{A_1}{K_{da1}}\right)^{n_1}+\beta\right)}{1+\left(\frac{A_1}{K_{da1}}\right)^{n_1}+\left(\frac{R_2}{K_{d2}}\right)^{n_2}+\left(\frac{A_1}{K_{da1}}\right)^{n_1}\cdot\left(\frac{R_2}{K_{d2}}\right)^{n_2}}$ $Z = \frac{\alpha_Z \cdot \tau \cdot \left(\left(\frac{Y}{K_d}\right)^{m}+\beta\right)}{1+\left(\frac{Y}{K_d}\right)^{m}}$ $K_{mi} = 100,\ h_i = 1.5, \alpha_R \cdot \tau$=300, g_1=1, $K_{di} = 1$ $n_i = 1,\ m = 2.5,\ \alpha_Y \cdot \tau$=200, K_{da1}=5, K_d=30, β =0.01	

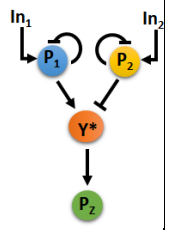
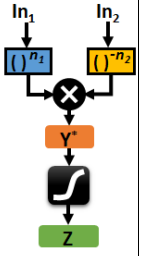
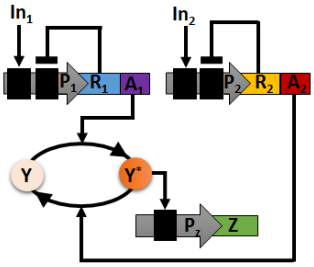
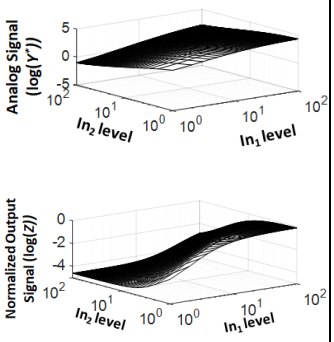
(D) Perceptgene based protein sequestration system

Schematic design	Schematic block	Genetic design	Mathematical model at steady state	Simulation results
			$R_i = \frac{\alpha_R \cdot \tau}{1+\left(\frac{R_1}{K_{di}}\right)^{n_i}} \cdot \frac{\left(\left(\frac{In_i}{K_{mi}}\right)^{h_i}+\beta_i\right)}{1+\left(\frac{In_i}{K_{mi}}\right)^{h_i}} \quad Y_T^{+} = g_1 \cdot R_1 \quad Y_T^{-} = g_2 \cdot R_2$ $C = \frac{K_{mc}+Y_T^{+}+Y_T^{-}}{2} - \frac{\sqrt{(K_{mc}+Y_T^{+}+Y_T^{-})^2 - 4 \cdot Y_T^{+} \cdot Y_T^{-}}}{2}$ $Y^{+} = Y_T^{+}$-C $\quad K_{mc} = K_{-1}/K_1$ $Z = \frac{\alpha_Z \cdot \tau \cdot \left(\left(\frac{Y^{+}}{K_d}\right)^{m}+\beta\right)}{1+\left(\frac{Y^{+}}{K_d}\right)^{m}}$ $K_{mi} = 100,\ h_i = 1.5, \alpha_R \cdot \tau$=300, g_1=1, g_2=15 $K_{di} = 1,\ n_i = 1, K_{mc} =$ 10, $m = 2.5$, K_d=3, β =0.01	

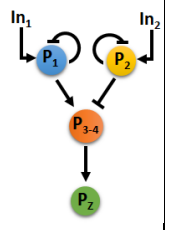
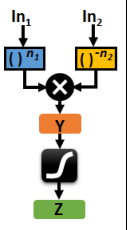
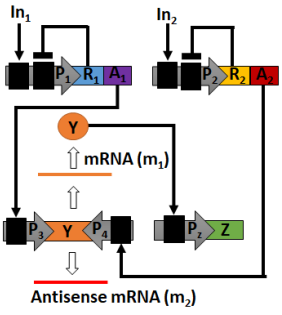
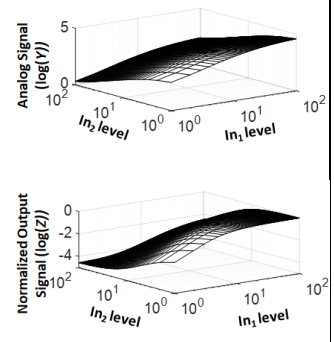
(E) Perceptgene based fusion protein system

Schematic design	Schematic block	Genetic design	Mathematical model at steady state	Simulation results
			$R_i = \frac{\alpha_R \cdot \tau}{1+\left(\frac{R_1}{K_{di}}\right)^{n_i}} \cdot \frac{\left(\left(\frac{In_i}{K_{mi}}\right)^{h_i}+\beta_i\right)}{1+\left(\frac{In_i}{K_{mi}}\right)^{h_i}}$ $AD_T = g_1 \cdot R_1 \quad BD_T = g_2 \cdot R_2$ $Comp = \frac{K_m+AD_T+BD_T}{2} - \frac{\sqrt{(K_{mc}+AD_T+BD_T)^2 - 4 \cdot AD_i}}{2}$ $K_{mc} = K_{-1}/K_1$ $Z = \frac{\alpha_Z \cdot \tau \cdot \left(\left(\frac{C}{K_d}\right)^{m}+\beta\right)}{1+\left(\frac{C}{K_d}\right)^{m}}$ $K_{mi} = 100,\ h_i = 1.5, \alpha_R \cdot \tau$=300, g_1=1, g_2=1, $K_{di} = 1,\ n_i = 1,$ $K_{mc} = 10,\ m = 2.5,\ K_d$=3, β =0.01	

(F) Perceptgene based two-component regulatory system

Schematic design	Schematic block	Genetic design	Mathematical model at steady state	Simulation results
			$R_i = \frac{\alpha_R \cdot \tau}{1+\left(\frac{R_i}{K_{di}}\right)^{n_i}} \cdot \frac{\left(\left(\frac{In_i}{K_{mi}}\right)^{h_i}+\beta_i\right)}{1+\left(\frac{In_i}{K_{mi}}\right)^{h_i}}$ $A_i = g_i \cdot R_i$ $Y^* = Y_T \frac{\frac{A_1}{A_2 \cdot K_m}}{\frac{A_1}{A_2 \cdot K_m}+1}$ $K_{mc} = K_{-1}/K_1$ $Z = \frac{\alpha_Z \cdot \tau \cdot \left(\left(\frac{Y^*}{K_d}\right)^m + \beta\right)}{1+\left(\frac{Y^*}{K_d}\right)^m}$ $K_{mi} = 100,\ h_i = 1.5, \alpha_R \cdot \tau = 300,$ $g_1 = 1,\ g_2 = 1,\ K_{di} = 1,\ n_i = 1,\ Y_T = 100,\ K_{mc} = 10,\ m = 2.5,$ $K_d = 30,\ \beta = 0.01$	

(G) Perceptgene based antisense transcriptional regulatory system

Schematic design	Schematic block	Genetic design	Mathematical model at steady state	Simulation results
			$R_i = \frac{\alpha_R \cdot \tau}{1+\left(\frac{R_i}{K_{di}}\right)^{n_i}} \cdot \frac{\left(\left(\frac{In_i}{K_{mi}}\right)^{h_i}+\beta_i\right)}{1+\left(\frac{In_i}{K_{mi}}\right)^{h_i}}$ $m_{Ti} = g_i \cdot R_i$ $C = \frac{(m_{T1}+m_{T2}+K_{mc})}{2} - \frac{\sqrt{(m_{T1}+m_{T2}+K_{mc})^2 - 4 \cdot m_{T1} \cdot m_{T2}}}{2}$ $Y = m_{T1} - C$ $Z = \frac{\alpha_Z \cdot \tau \cdot \left(\left(\frac{Y}{K_d}\right)^m + \beta\right)}{1+\left(\frac{Y}{K_d}\right)^m}$ $K_{mi} = 100,\ h_i = 1.5, \alpha_R \cdot \tau = 300,$ $g_1 = 1,\ g_2 = 1,\ K_{di} = 1,\ n_i = 1,$ $K_{mc} = 10,\ m = 2.5,\ K_d = 3,\ \beta = 0.01$	

Tables S8.1. Examples for neuromorphic gene circuits. The schematic design describes the regulatory elements, the schematic block shows the mathematical operations, the genetic design shows the genetic implementation and biological regulatory components.

In addition to our own experimental data and simulation results, previous articles have also demonstrated the ability to modulate various properties of engineered gene circuits that are relevant to our neuromorphic circuit engineering efforts. The engineered libraries of genetic device variants described briefly below could be used in our neuromorphic approach to obtain essentially continuous modulation of weights and biases:

- The Ribosome Binding Site Calculator is a tool that predicts the affinity to RNA polymerase of synthetic ribosome binding sites in *Escherichia coli*, and as such enables rational control over protein expression levels (*51*). This tool can be useful for programming bias, which is directly affected by the translation rate.
- The Anderson synthetic promoter library includes more than 30 characterized promoters with variable strength of approximately 100 fold between the weakest and strongest (http://parts.igem.org/Promoters/Catalog/Anderson). This tool can also be useful for

programming bias, which is directly affected by the transcription rate. Another synthetic promoter library was also published around the same time (*52*).

- The Weiss TALER library includes 26 programmed transcriptional repressors that bind synthetic combinatorial promoters in mammalian cells (*53*). With TALE modular protein construction, any DNA sequence can be targeted, leading to an essentially limitless design search space (with a usable length of anywhere between 14 and 26 DNA bases for TALER binding). The library elements have an approximately 2 orders of magnitude difference in repression folds from around 20 to greater than 10^3 leading to different Hill coefficients and hence different input weights. Promoter engineering by inclusion of two versus four TALER binding sites increased fold repression by five and ten fold for TALER21 and TALER14 respectively.
- The Voigt repressor library includes 16 orthogonal TetR-family repressors and their cognate promoters. Each repressor/promoter pair's transfer function has been characterized. The measured Hill coefficients range between 1.5 and 6.5, with fold changes between 1-2 orders of magnitudes (*54*). This tool can be useful for programming input weights.
- A recent effort in *Escherichia coli* has demonstrated several inducible synthetic promoters with varying ligand-promoter activity transfer functions. The synthetic promoters are regulated by TtgR, PmeR and NalC and are induced by phloretin, Naringenin, and PCP, receptively (*55*). This tool can be useful for programming input weights similar to Fig. S8.8.
- The Riboswitch Binding Sequence Calculator predicts ligand induced gene activation of riboswitch sequences using a physics based model. Then, computational design with this tool is used to create a library of 62 different synthetic riboswitches with activation fold of up to 383x (*56*). This tool can be useful for programming weight and bias.
- A library of LuxR transcription factors were developed (*57*). AHL -dependent transcriptional activation can be selected to meet design specifications. This tool can be useful for programming input weight similar to Fig. S8.8.
- A library with 238 member of tunable control for protein degradation in bacteria were developed (*58*). This tool can be useful for programming bias, which is directly affected by the protein half-life.
- A library of antisense constitutive promoters was developed (*40*) . Every member of the library includes a target gene that is regulated by a repressor and by another promoter that is oriented opposite to the target gene. The library includes 5,668 terminator–promoter combinations that was used to control the expression of three repressors (PhlF, SrpR, and TarA). Such design can be used reliably to tune gene expression level and control small molecules' dissociation constant. This tool can be useful for programming bias.

Other methods to alter the dosage response curves of genetic regulation elements have also been published, and these could also be used to modulate weight and bias in neuromorphic circuits:

- Landry *et al.* 2018 developed a two-component signaling system that can dynamically tune the dissociation constant of small molecules. This system can be used to control bias (*59*).

- Segall-Shapiro *et al.* 2014 split T7 proteins into several parts and changed cooperativity. This method can be used to control the weights for inputs and activation functions (*22*).
- Morel *et al.* 2016 introduced extra binding sites into promoters and changed cooperativity. This method can be used to control input weight (*60*).

We also analyzed the properties of common synthetic biological parts, including weights for some of the parts used in this manuscript (Fig. S8.11A) and Hill coefficients for devices that were previously published (Fig. S8.11B) providing another source of parts with desired weights and Hill-coefficients for small molecules and transcription factors.

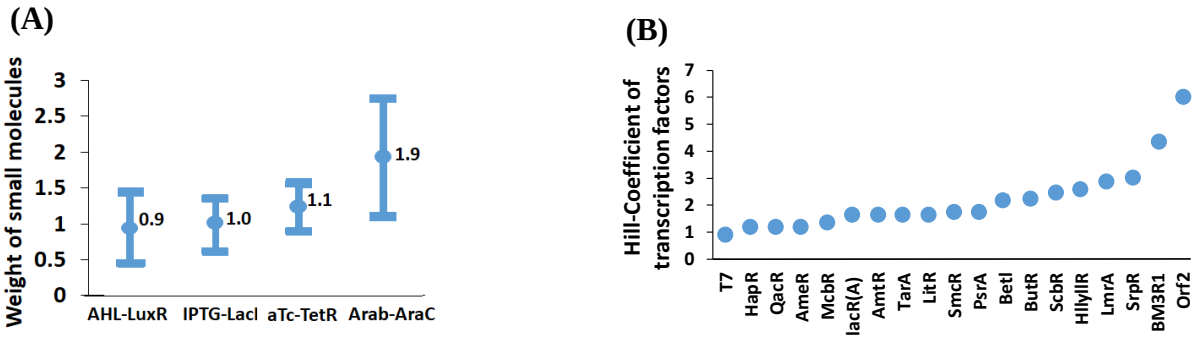
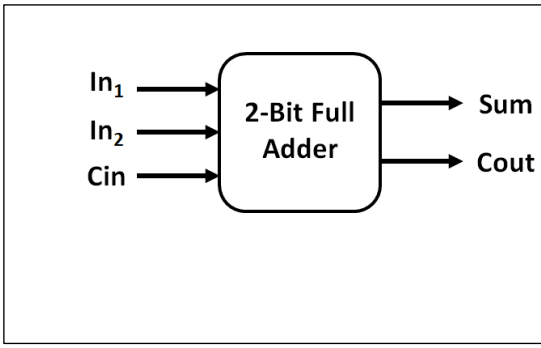


Fig. S8.11. (A) Hill coefficient values of small molecules that are used in this study. (B) Hill coefficient values of transcription factors that are used in Stanton et al. 2014 (*54*).

In summary, there are many methods to control the weights and biases, these include transcription factor binding sites, operator mutations, and T7 RNA polymerase mutations. With respect to LacI regulation of a promoter via the number of binding sites, there is a practical limit on the number of binding sites that can be used in a single promoter. As such, the number of binding sites in a single promoter only represents one coarse grain 'knob' for tweaking weights. The example of T7 RNA polymerase is also coarse-grained. The power and flexibility come from combining such coarse-grain approaches with others that provide more fine-grain tuning (e.g., operator sequence mutations). For the operator mutations, we show experimentally seven different weights with good coverage of the desired range and support the feasibility of obtainingnear-continuous control (Fig. S8.8). Importantly, we show experimentally that replacing the open-loop control with closed loop feedback control shifted the range of weights from 0.25-0.80 to 0.75-1.70 (Fig. S8.10). This is an example of how coarse grain and fine grain control can be used synergistically to obtain desired weights. In terms of additional control, other synergistic approaches mentioned above include Hill coefficients of small molecule inducers that serves as perceptgene inputs, transcription factors that competitively inhibit expression via steric hinderance, regulation of negative feedback strength, transcription factor sequestration via protein-protein interactions, protein structure (*e.g.*, dimerization and cooperativity), and circuit topology. As such, optimization and reconfiguration of neuronal circuit function is not solely dependent on the success or failure of any particular approach. These approaches can be mixed and matched, and the impact on the cost function can be then observed in order to further refine neuronal circuit behavior. Clearly, at the moment, these modulations are not as easy to manipulate as, for example, modifying weights in a computer simulation of neuronal circuits.

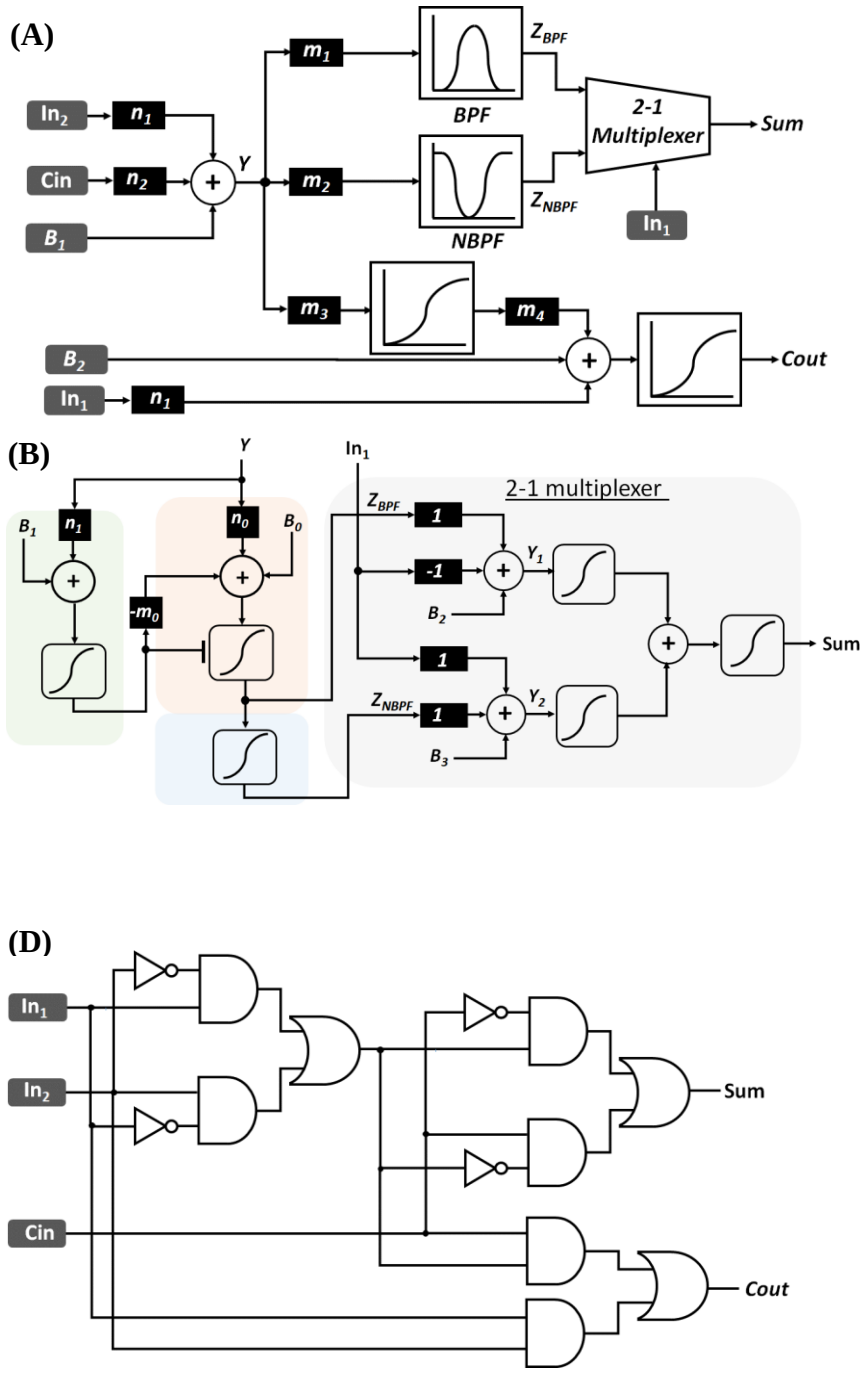
Scaling and optimization of neuromorphic circuit based on using design principles from analog and digital computation (Supplementary Information, Sections 5 and 7), fuzzy computation (Supplementary Information, Sections 3 and 4), algorithms from ANNs as gradient descent and backpropagation (Supplementary Information, Section 6), and modules that were developed in the neuromorphic field such as Hopfield networks (*61*). Here we provide another example to design 2-Bit Full adder based on neuromorphic design. We start by presenting the truth table of our circuit (Table S8.2):



In_1	In_2	C_{in}	In_2+C_{in}	Sum	Cout
0	0	0	0	0	0
0	0	1	1	1	0
0	1	0	1	1	0
0	1	1	2	0	1
1	0	0	0	1	0
1	0	1	1	0	1
1	1	0	1	0	1
1	1	1	2	1	1

Table S8.2. Truth table of 2-bit Full Adder.

As shown in the Table S8.2, the *Cout* output displays "1" if and only if the majority of the inputs are "1", and displays "0" if and only if the majority of the inputs are "0". Such function is called 3-input majority and is implemented in this study (Figure 3, Supplementary Information, Section 5). The implementation of *Sum* output of 2-bit Full adder is more complex depending on the design roles (e.g., the input numbers of single perceptgene). According to the truth Table S8.2, when the input In_1 is "0", the summation of In_2 and C_{in} inputs can be encoded to band-pass filter circuit (BPF), and when the input In_1 is "1" the summation of In_2 and C_{in} inputs can be encoded to NOT-BPF (NBPF). Our 2-bit Full adder comprises BPF, NBPF and 2-1 multiplexer (Fig. S8.12A). A NBPF circuit is an inverted BPF circuit, which means, a high output results if and only if the input level is very low/high, and low output results for intermediate levels of input. The implementation of BPF is shown in Supplementary Information, Section 7, and it includes two cascaded perceptgenes that one inhibits the other (Fig. S8.12B). By wiring the output of BPF with an inhibitor, one can simply implement the NBPF. The 2-1 multiplexer selects between the BPF and NBPF output signals and forwards it to Sum output (Fig. S8.12C). In our design, the In_1 acts as a selector and the outputs of BPF and NBPF are the data signals which are forwarded to the Sum output. The operation of Multiplexer can be described as $u\{Z_{BPF} \cdot (1 - In_1)\} + u\{Z_{NBPF} \cdot In_1\}$, where u is the sigmoid function. Fig. S8.12D shows a digital design of 2-bit Full adder which include 6 AND gates, 3 OR gates, and 2 NOT gates. According to the assumption that each 2-input logic gate can be implemented only by 2 transcription factors, we get that neuromorphic design requires 9 transcription factors while the digital design requires 20 transcription factors. Notably, that 2-bit Full adder can be implemented with other biological parts (*62*) than transcription factors, which might require fewer components.



(C)

In 0	Z_{BPF}	Z_{NBPF}	Sum
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Fig. S8.12. (A) Design of 2-bit Full adder. (B) Design of band-pass (BFP) filter circuit BPF, NOT-BPF (NBPF) and 2-bit Multiplexer. (C) Truth table of 2-bit Multiplexer. (D) Digital design of 2-bit Full adder.

9. Dynamic Measurements of Neuromorphic Genetic Circuits

We characterized the dynamics of perceptgenes implementing the power-law and multiplication function and the average function, as well as the multi-layered 2-bit ADC. These experiments monitor the progress of circuit output at multiple time points (4.5, 7, 8.5 and 10 hours). In general, circuit output is quite stable across all of these time points (Fig. S9.1, Fig. S9.3, and Fig. S9.4). The maximum and minimum levels of the LSB and MSB circuit in the multi-layered 2-bit ADC across all input dosages reach approximately 2/3 of their highest values after 4.5 hours (Fig. S9.5). These levels gradually increase until they reach their peak at 8.5 hours and then decreases back to about 2/3 of the maximum at 10 hours. Importantly, the input levels where LSB and MSB outputs transition between low and high levels are consistent across all time points (Fig. S9.6). Therefore, at all time points measured the ADC continues to properly convert AHL input concentration levels to the appropriate four output states [0,0], [0,1], [1,0] and [1,1]. Similar dynamic behavior is observed for the average circuit and power-law and multiplication circuit (Fig. S9.1 and Fig. S9.3). The dynamics of our circuits are mainly determined by the characteristics of the synthetic parts and the regulatory topologies. The synthetic parts we use are based on parts that have been extensively characterized in the literature. The regulatory topologies that govern the behavior of our circuits include cascades, feed-forward, and feedback motifs – again, motifs that frequently occur in synthetic biology. by definition, we expect that the dynamics of our neuromorphic circuits are roughly the same as existing digital and analog circuits using similar synthetic parts and motifs (*4*, *6*, *16*), e.g., response times in few hours.

Time-course experiments were performed on perceptgene for computing power-law and multiplication function (Fig. 1B and 1D), perceptgene for computing an average function (Fig. 2G and 2I) and on ADC circuits (Fig. 4F, and 4G). *E. coli* strains were picked from LB agar plates and grown overnight at 37°C, 300 r.p.m. in 5 mL of LB medium with appropriate antibiotics and inducers (Carbenicillin (50 µg/ml), Kanamycin (30 µg/ml), Chloramphenicol (34 µg/ml)). Overnight cultures were diluted 1:100 into 5 mL of LB medium with added antibiotics and were then incubated at 37°C, 300 r.p.m. for 30 min. 200 µl of culture was then moved into a 96-well plate, combined with inducers (Arabinose and AHL 3OC6HSL), and incubated in a VWR microplate shaker at 37°C, 500 r.p.m. Once the diluted cultures grew to an OD_{600} of ~0.5 (~4 hours and 30 min), 120 µl of culture was taken to a FACS machine for measurement. Simultaneously, we performed two steps:

1. 40 µl of culture was moved into a new 96-well plate containing 200 µl of media, antibiotics, and inducers and then incubated in a VWR microplate shaker at 37°C, 500 r.p.m. At OD_{600} ~0.5 (after 2.5 hours), 200 µl of culture was taken to a FACS machine for measurement and 40 µl of culture was moved into a new 96-well plate containing 200 µl of media, antibiotics, and inducers and then incubated in a VWR microplate shaker at 37°C, 500 r.p.m. This iterative dilution, growth, and measurement process was repeated and resulted the dynamics after 7 hours and 10 hours.
2. 20 µl of culture was moved into a new 96-well plate containing 200 µl of media, antibiotics and inducers, then incubated in a VWR microplate shaker at 37°C, 500 r.p.m. At OD_{600} ~0.5 (4 hours and 20 min), 200 µl of culture was taken to a FACS machine for measurement and resulted the dynamics after 8.30 hours.

The experimental results of power-law, multiplication function, and average circuits corresponding to different times are shown in Fig. S9.1 and Fig. S9.3 below shown by fitting our experimental results to surface, the weight values are consistent over time (Fig. S9.2). The average circuit continues to operate correctly and compute the average between two analog numbers (Fig. S9.3).

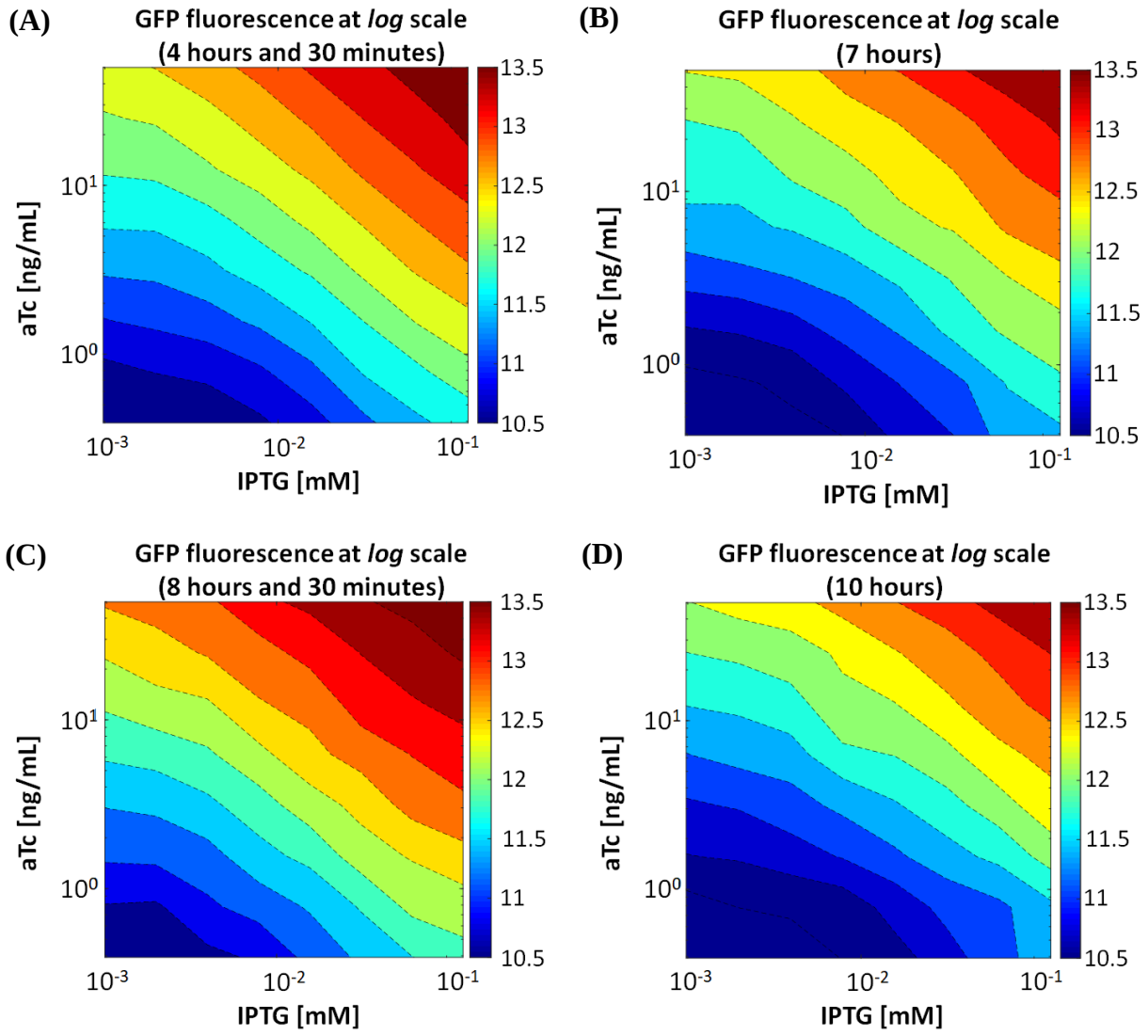


Fig. S9.1. Raw data of time-course experiments (4.5 hours, 7 hours, 8.5 hours and 10 hours) for perceptgene computes power-law and multiplication function (Matching Fig. 1B with P_{lacO}/P_{tetO})

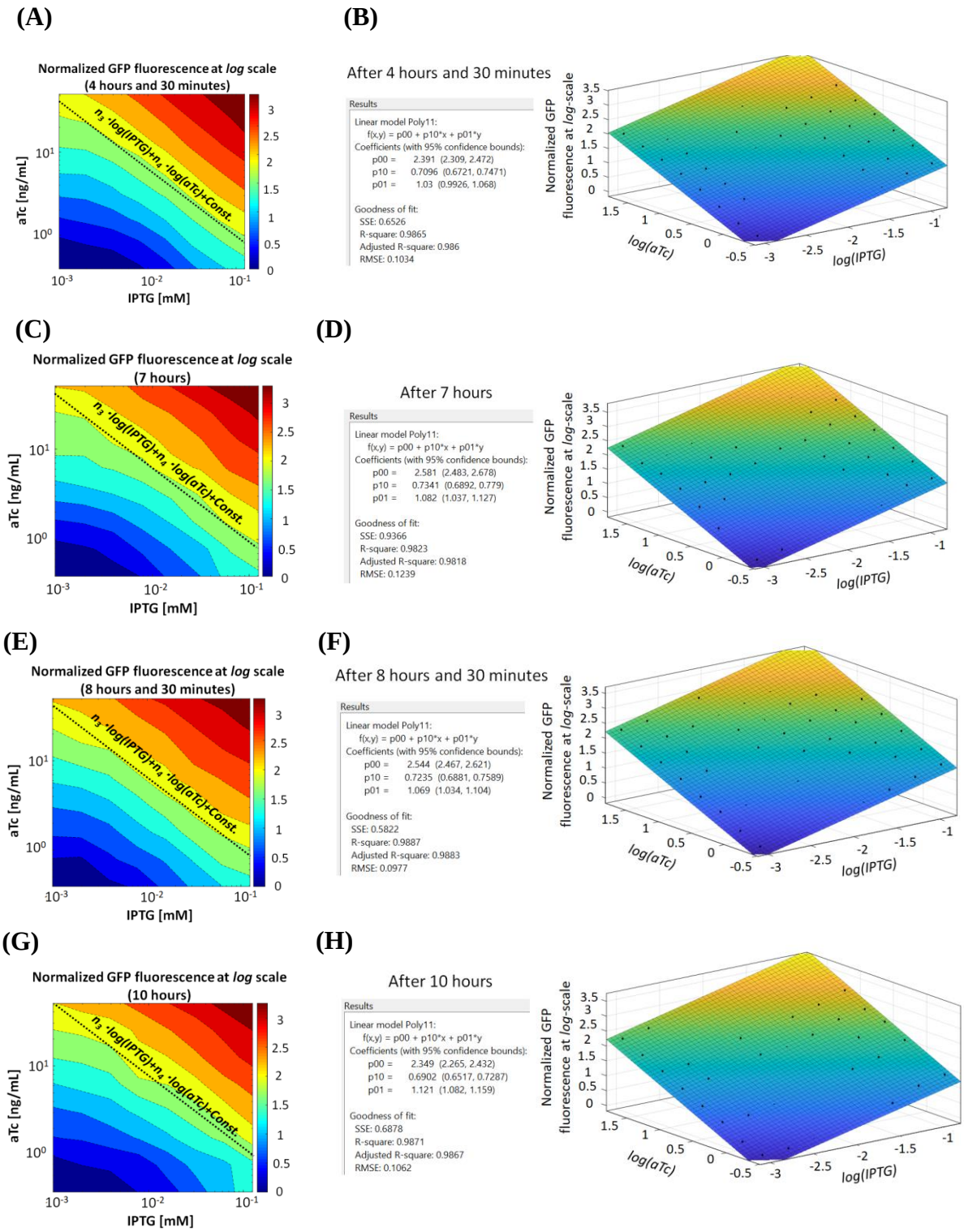


Fig. S9.2. Normalized data of Time-course experiments (4.5 hours, 7 hours, 8.5 hours and 10 hours) for perceptgene that computes power-law and multiplication function (A, C, E, and G). Matlab surface fits (B, D, F, and H) the experimental results to power-law and multiplication function $log(GFP) = c + n_3 \cdot log(IPTG) + n_4 \cdot log(aTc)$.

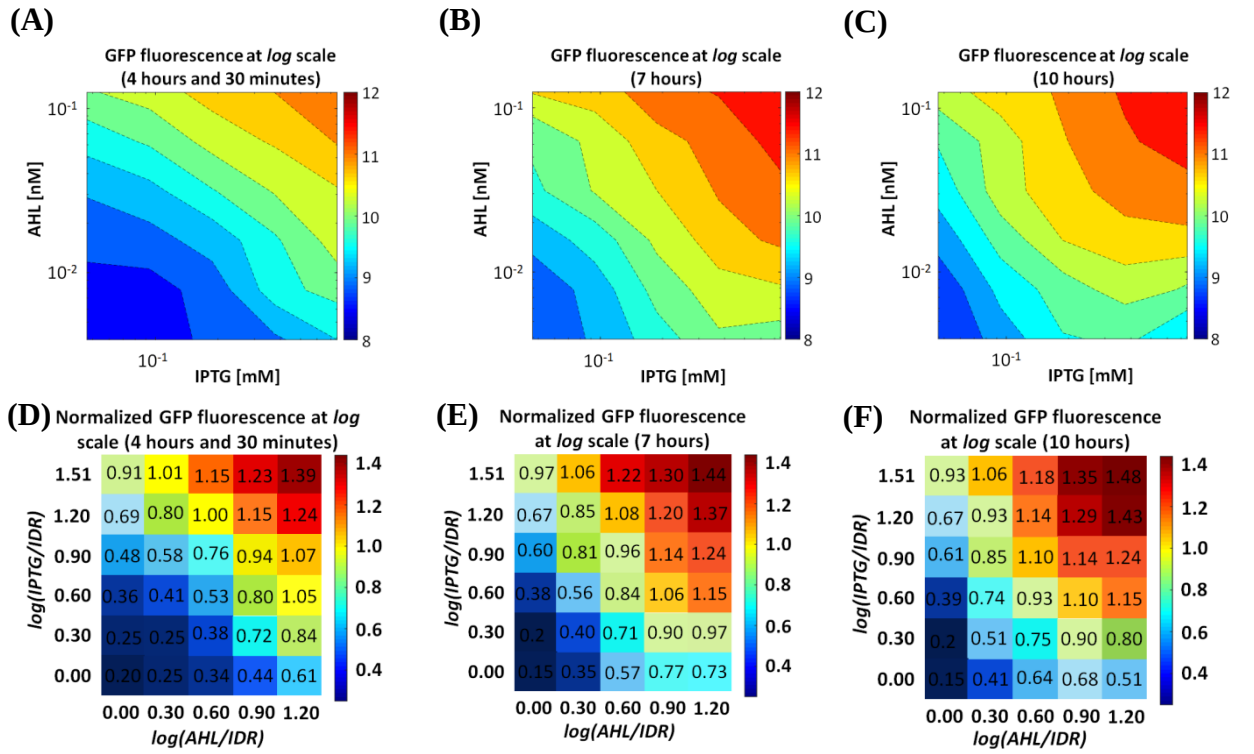


Fig. S9.3. Dynamics of average computing circuit. (A)-(C) Raw data of time-course experiments (4.5 hours, 7 hours and 10 hours) for perceptgene that computes the average between two analog signals. (D)-(F) Normalized data of Time-course experiments (4.5 hours, 7 hours, and 10 hours) for perceptgene that computes average function between two analog inputs.

The experimental results of ADC corresponding to different times are shown in Fig. S9.4 below. The GFP signal of the ADC circuit represents the LSB output, and the mCherry signal of the ADC circuit represents the MSB output. In the fours time points (4.5 hours, 7 hours, 8.5 hours and 10 hours) our ADC continues to operate properly and convert the AHL concentration level to four states [0,0], [0,1], [1,0] and [1,1] (Fig. S9.5) We also compared the minimum and maximum expression levels of GFP and mCherry at different time points (Fig. S9.6). Furthermore, we fitted the data to an empirical model, and we found that the fitting parameters change slightly across time (Table S9.1). The empirical model is based on Section 7 and is given by:

$$LSB_{low} = \frac{\left(\frac{AHL}{K_{m1}}\right)^{h1}}{1+\left(\frac{AHL}{K_{m1}}\right)^{h1}} \cdot \frac{1}{1+\left(\frac{AHL}{K_{m2}}\right)^{h2}} + b_1 \quad \text{(S9.1)}$$

$$LSB_{high} = \frac{\left(\frac{AHL}{K_{m3}}\right)^{h3}}{1+\left(\frac{AHL}{K_{m3}}\right)^{h3}} + b_2 \quad \text{(S9.2)}$$

$$LSB = \frac{LSB_{low}}{Max(LSB_{low})} + \frac{LSB_{high}}{Max(LSB_{high})} \quad \text{(S9.3)}$$

$$MSB = \frac{\left(\frac{AHL}{K_{m4}}\right)^{h4}}{1+\left(\frac{AHL}{K_{m4}}\right)^{h4}} + b_3 \quad \text{(S9.4)}$$

Where K_{mi} are dissociation constants, h_i are Hill-coefficients, and b_i are basal levels.

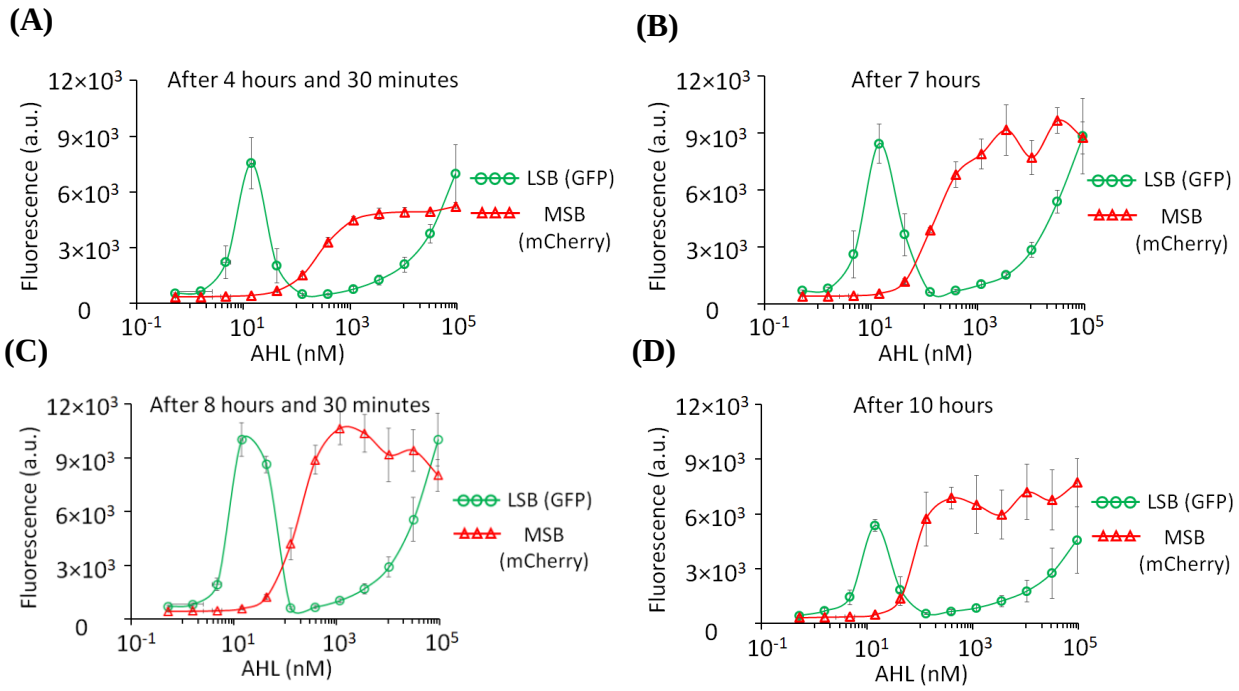


Fig. S9.4. Raw data of time-course experiments (4.5 hours, 7 hours, 8.5 hours and 10 hours) for the ADC circuit. To enable a suitable comparison between the GFP and mCherry signals, we scaled the measured mCherry signal by 23.

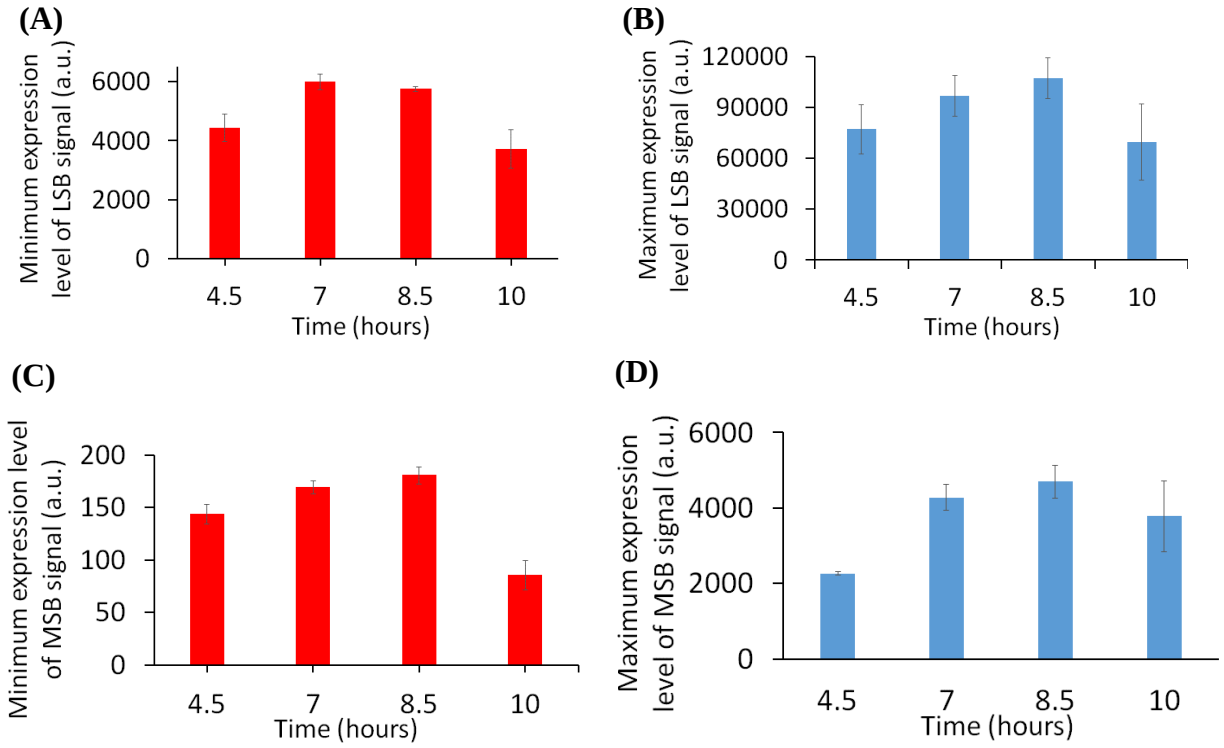


Fig. S9.5. Time course of expression levels for the ADC circuit. (A) Minimum level of GFP (LSB). (B) Maximum level of GFP (LSB). (C) Minimum level of mCherry (MSB). (D) Maximum level of mCherry (MSB).

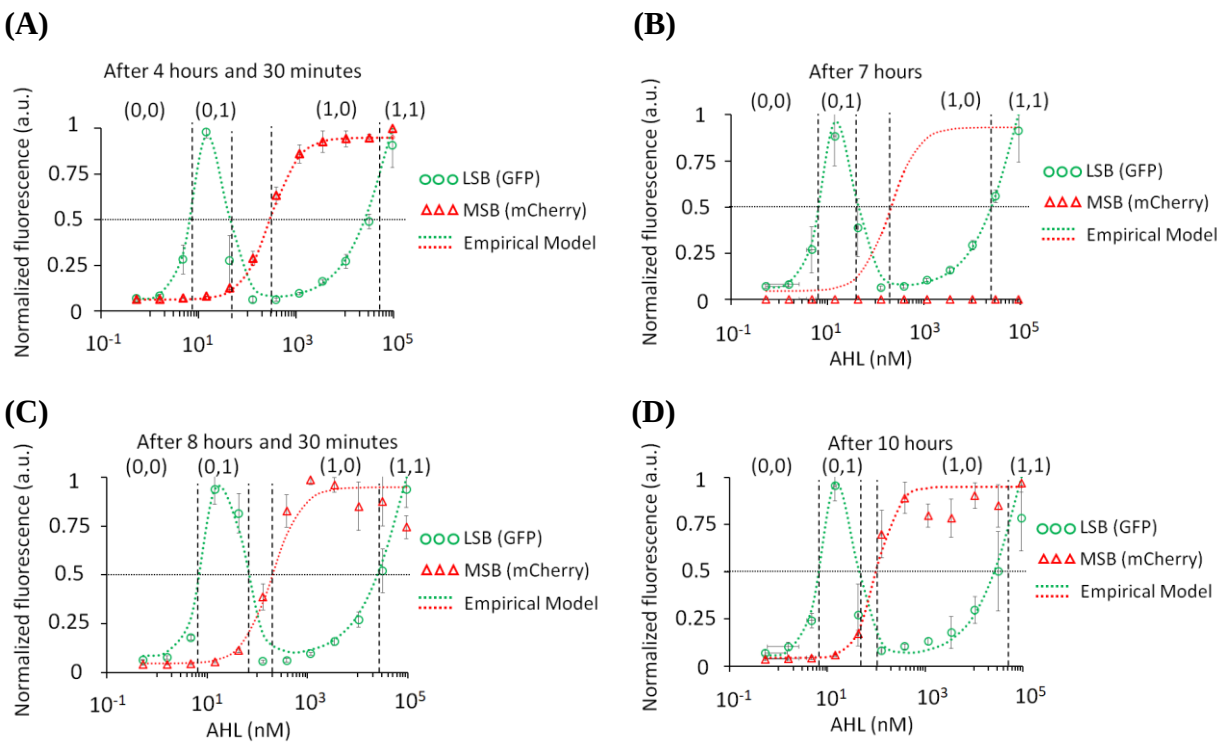


Fig. S9.6. Normalized data for Time-course experiments (4.5 hours, 7 hours, 8.5 hours and 10 hours) of the ADC circuit. The dotted line corresponds to a set of equations; Eq. S9.1-4. The average signals and standard divisions were calculated for the normalized signals. Therefore, there is a difference in the standard deviation values compared to Fig. S9.4.

Table S9.1. Shows the time course fitting parameters (Eq. S9.1-.4) for the ADC circuit. The empirical model is based on Section 8.

Parameter	Time=4.5 hours	Time=7 hours	Time=8.5 hours	Time=10 hours
K_{m1}	20nM	20nM	**30nM**	**20nM**
K_{m2}	15nM	15nM	15nM	15nM
K_{m3}	10^5nM	10^5nM	10^5nM	10^5nM
K_{m4}	300nM	**200nM**	200nM	**100nM**
h_1	2	2	2	2
h_2	2	2	2	2
h_3	0.9	0.9	0.9	0.9
h_4	1.5	1.5	1.5	1.5
b_1	0.01	0.01	**0.008**	0.008
b_2	0.005	0.005	0.005	0.005
b_3	0.07	**0.05**	0.05	**0.03**

10. Noise Analysis in Neuromorphic Circuits

In this section, we perform the Signal-to-Noise ratio (SNR). The sensitivity analysis from Section 1 and SNR parameters can provide quantitative information for the precision and reliability of circuits. We quantified the noise of neuromorphic circuits by further analysis of single-cell FACS experimental data. Specifically, for each single layer perceptgene circuit (from Figures 1 and 2) we quantified the signal-to-noise ratio for each input dosage that we tested and graphed a corresponding SNR histogram (Fig. S10.1). These histograms show the distribution of SNR exhibited by each of the circuits. The general observation is that the power law and multiplication circuits that use only auto-negative feedback for the inputs (LacI/IPTG and TetR/aTc) generally tend to have higher SNRs than circuits that include auto-positive feedback (LuxI/AHL with either TetR or LacI). Another observation is that the activation function (AraC) addition tends to coalesce the SNR distributions of all three circuits to roughly the same values. Thus, activation functions utilized in two circuits were able to increase the SNR.

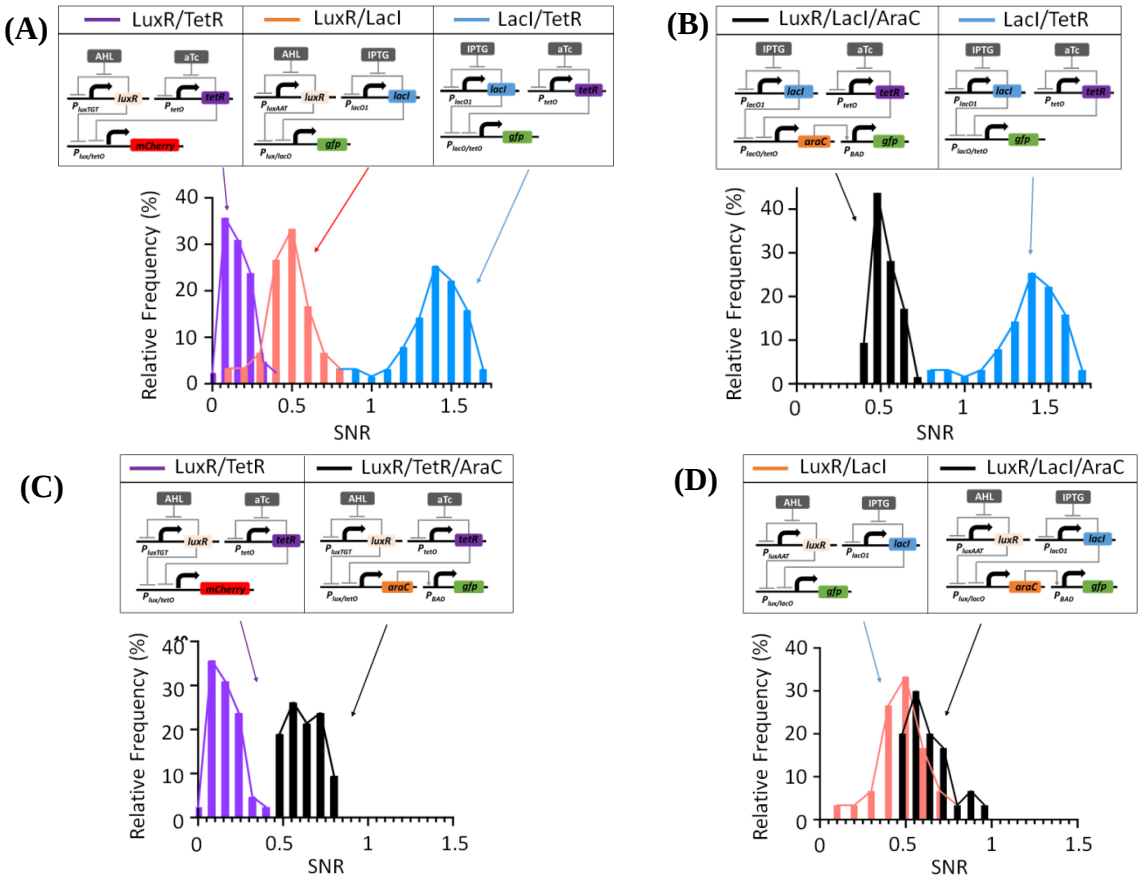


Fig. S10.1. Signal-to-Noise ratio (SNR) analysis of perceptgene units (Fig. 1 and 2 in the main text). (A) SNR analysis for power-law and multiplication circuits. The circuit with only negative feedback (via LacI and TetR) exhibits improved SNR over the other two circuits that contain a positive feedback motif (via LuxR). (B), (C) and (D) SNR analysis for perceptgene with activation functions computes smooth minimum, maximum and average functions, respectively.

11. Benefits of log-based ANNs computing

In electronics, ANNs outperform the conventional computing paradigms (e.g., digital and analog) in a variety of settings (such as classification and signal processing (*63–67*)**) owing to their collective resilient properties. ANNs perform efficient execution of complex functions by utilizing a low number of components similar to analog design and producing reliable results similar to digital circuits (e.g. majority function, supplementary information, Section 5, ADC, Supplementary information, Section 7). Implementing ANNs in a biological setting requires an important change to obtain the same performance benefits. Specifically, using logarithmic rather than linear functions is often more appropriate for describing biochemical reactions in gene regulation (e.g., Hill Functions that describe dosage response curves, Weber's law** (*4, 49, 68*)**). The use of logarithmic functions for ANNs places particular requirements on properties of gene regulatory elements in terms of their Hill coefficients, basal and maximal protein expression levels, and transcription factor dissociation constants. These requirements can be met 'easily'. Furthermore, we have shown that common biochemical reactions can be simply converted to perceptgene units (Design principles of neuromorphic circuits, Supplementary Information, Section 8).**

In comparison, a linear genetic implementation would have been based on less reliable parts, for example, we have shown in Section 1 that computation based on the perceptron requires an activation function with a very high Hill-coefficients (>2.5). Such values are very challenging to obtain in synthetic and natural biological systems. Therefore, linear-based ANNs computing would place much more stringent requirements on gene regulatory elements' properties and necessitated a more complex design to achieve the same performance. Furthermore, the logarithmic domain is also more appropriate for attenuating the effects of typical fluctuations in protein expression levels. Subsequently, it provides a more resistant platform for neuromorphic computing in a gene regulation context. The essence here is the reliance on fold-change regulation, as opposed to absolute-change regulation, with the former being more appropriate for genetic circuits (as previously articulated in the community, e.g., by Uri Alon (*69*)**).**

12. Comparing neuromorphic computing with digital and analog computing

Our selection of circuits to design and build was based on two guidelines:

1. ***1.*** To prove that, for the first time, neuromorphic computing principles can be achieved in single living cells by transforming concepts from neural networks to genetic regulatory networks.
2. ***2.*** To construct synthetic gene circuits that perform complex computation with minimal requirements in computational devices and host cell resources.

Three of the circuits we decided to build, min/max/avg, are fundamental building blocks for neuromorphic computing. The other three, majority / 2-bit ADC / ternary switch, demonstrate multi-layered neuronal circuits.

In terms of appreciating the min/max/avg functions, it is essential to recognize that to the best of our knowledge, only "hard" (i.e., discrete) minimum and maximum functions have been demonstrated in synthetic biology. Complicated functions operate using binary AND/OR logic, where each bit has two logic states. In terms of these complicated functions, AND implements binary min, while OR implements binary max. The average function cannot be implemented with an individual single-bit binary logic gate but would instead require digitization of input and output signals and very complex multi-device logic. In sharp contrast, soft functions operate in the analog domain. Our single perceptgenes implement single-device analog min/max/avg computations whereby the single devices transform analog input signals into output values that remain in the analog domain. These operations have not been demonstrated in synthetic biology!

Our multi-layer functions also represent significant progress over existing efforts in synthetic biology. Our majority function (1) demonstrates neuromorphic modularity because the three-input majority function is built from two-layer perceptron and (2) allows us to compare the properties of neuromorphic design with digital design. Our three-input majority function has two main advantages over the previous digital design (*6*): (1) We use fewer synthetic parts; our three-bit majority function comprises 15 biological parts (i.e., promoters and genes) in comparison to 22 parts, (2) the neuromorphic circuit is reconfigurable and trainable via learning algorithms that optimize desired behavior efficiently (e.g., reduce error). With this neuromorphic architecture, we minimized error by modulating the weight of $P_{BAD}/AraC$ in a manner similar to backpropagation algorithm. This optimization approach could not have been performed for the existing digital circuit design.

Our other two multi-layer neuromorphic circuits also provide innovation beyond existing approaches. To the best of our knowledge, we are the first to demonstrate a 2-bit analog-to-digital converter (ADC). In general, analog-to-digital converters take as input a graded signal, partition the analog input into several consecutive ranges that cover the entire input range, and assign a digital value to these ranges in a sequential manner. Representing this digital value requires multiple bits if more than two regions are specified. A 1-bit ADC partitions the input range into two, and the output is then a single bit with a value of either 0 or 1. A 2-bit ADC partitions the input range into four, with an output that requires two bits representing each of the four consecutive ranges, namely 00, 01, 10 and 11.

Two recent synthetic biology publications have discussed the notion of analog-to-digital converters. In one publication (*70*), 1-bit analog-to-digital conversion was used to quantize extracellular inputs (including dihydrojasmone and eugenol) each into single-bit values, and then these were combined into several 2-input logic functions (AND, OR, NOR) still operating with single bit output. In another recent publication (*46*), a single analog input (H_2O_2) was partitioned

into three consecutive ranges, and three separate 1-bit outputs (GFP, RFP, and BFP) were used to indicate which of three ranges was detected. Hence, one of these digital outputs is high for a given analog input value while the other two are low. In conventional ADC circuit design, these three 1-bit outputs are then combined via a second stage digital logic circuit (comprising three 2-input logic gates: one XOR and two AND gates) to create a 2-bit digital representation of the analog input signal. Hence, this work represents only the first stage of a 2-bit ADC, but not the second stage. In terms of biological circuit elements, they used seven transcription units. We estimate that it would require 6-8 additional transcription units to implement their second stage of the 2-bit ADC, which would require a total of 15 transcription units if it was built. In comparison, ours is a fully functional 2-bit ADC implemented using only five transcription units. Besides minimizing the size of the circuits, our perceptgene networks also operate with low expression levels, mainly in order to maintain low bias levels. In contrast, digital systems often attempt to operate with significant noise margins, and hence high expression levels for ON values. This latter point is further elaborated on in the main narrative.

The implementation of our third multi-layered neuromorphic circuit, the ternary switch, demonstrated the ease of converting one neuromorphic computing to another. Specifically, we started with the 2-bit ADC circuit and increased the LSB perceptgene activation using higher Arabinose, which corresponds to increasing the MSB input weight into the LSB computation. Such ease in changing neuronal network parameters and achieving new functions is an important component for ultimately implementing learning algorithms using gene circuits.

13. Potential Applications of Synthetic Neuromorphic Circuits

For example, a three-input majority function can be fitted to any logic gate with up to three inputs, including the two-input AND gate (*71*, *72*), and optimized in applications currently suffering from a trade-off between specificity and sensitivity. Typically, synthetic gene circuits for disease treatment must be highly sensitive to detect biomarkers and deliver the produced therapeutic proteins to target cells and must be precisely specified to protect surrounding healthy cells (*60*). Ternary converters may also be helpful in engineering cells whose therapeutic outputs are connected in a closed-loop and are regulated by quantitative levels of disease biomarkers. While circuits behave either in an analog manner, showing insensitivity to disease biomarkers, or in a digital manner, in which they are quantized to a single level of therapeutic proteins, ternary converters with feedback loops can settle at two saturated levels and can precisely adjust the production of the therapeutic proteins (e.g., adiponectin which attenuates insulin-resistance syndrome (*73*)) to the level required for disease management. Furthermore, data converters may find applications in biotechnology either by coordinating the expression of several genes, using a single inducer or by improving the production yield of the desired biomass in synthetic pathways, using a three-state genetic switch. For example, engineered cells that produce quorum-sensing signals (AHL) and contain a ternary converter, could have multilevel phases, dictated by accumulated AHL levels in the bioreactor. These phases could efficiently optimize the production rate versus the cell growth rate compared to the two-state switch (*74*).

Another example is that the next-generation therapeutic-based synthetic gene circuits can be self-controlled once administered, replacing the need for exhaustive manipulation by a manually customized, trial-and-error clinical design. Recently, the design cycle of bioproducts has driven the set-up of laboratory automation, foundries (e.g., robotics) and information infrastructures (*75*) using ‘design, build, test, learn and correct’ heuristics. We expect that adaptive genetic circuits will significantly standardize this cycle, drastically reduce time-to-market and cost, through a generic methodology, using training algorithms suitable for general purpose applications.

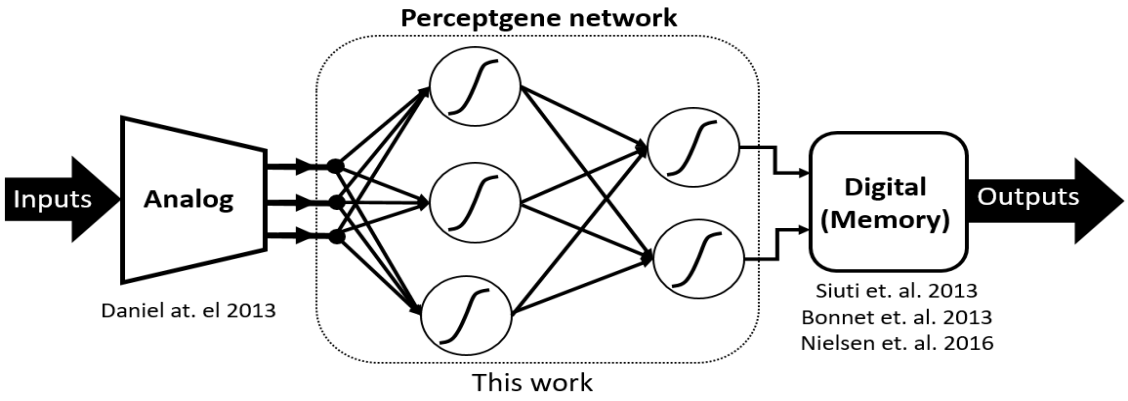


Fig. S13. proposed an efficient and reliable computing platform, which combines analog, a perceptgene network, and digital memory for sorting.

Fig. S13 shows the proposed platform is compatible with digital and analog computing platforms using data converters. This complementary strategy can leverage the advantages of the three platforms to achieve an efficient and accurate computational approach for scaling the architecture of robust genetic networks in living cells. For instance, analog computing can be applied for front-end calculations (e.g., ratiometric for sensory systems), perceptgene networks can be applied for processing and computation, and digital circuits can be applied back-end data storage (memory) with clear ON and OFF states.

14. FACS Data

All fluorescence intensities were smoothed using Matlab.

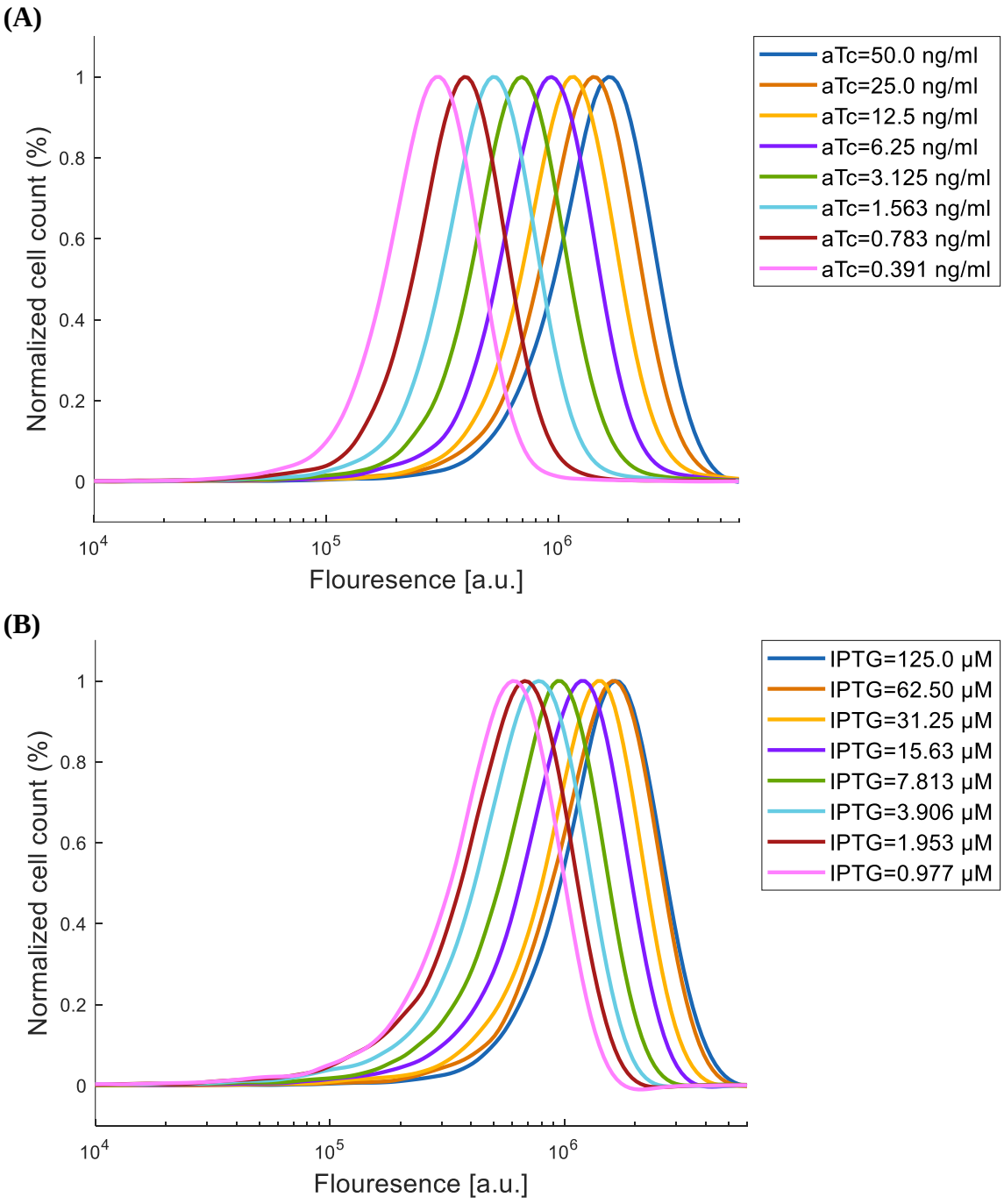


Fig. S14.1. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops (Fig. 1C). (A) IPTG was held constant at 125 μM, and aTc was varied. (B) aTc was held constant at 50 ng/ml, and IPTG was varied.

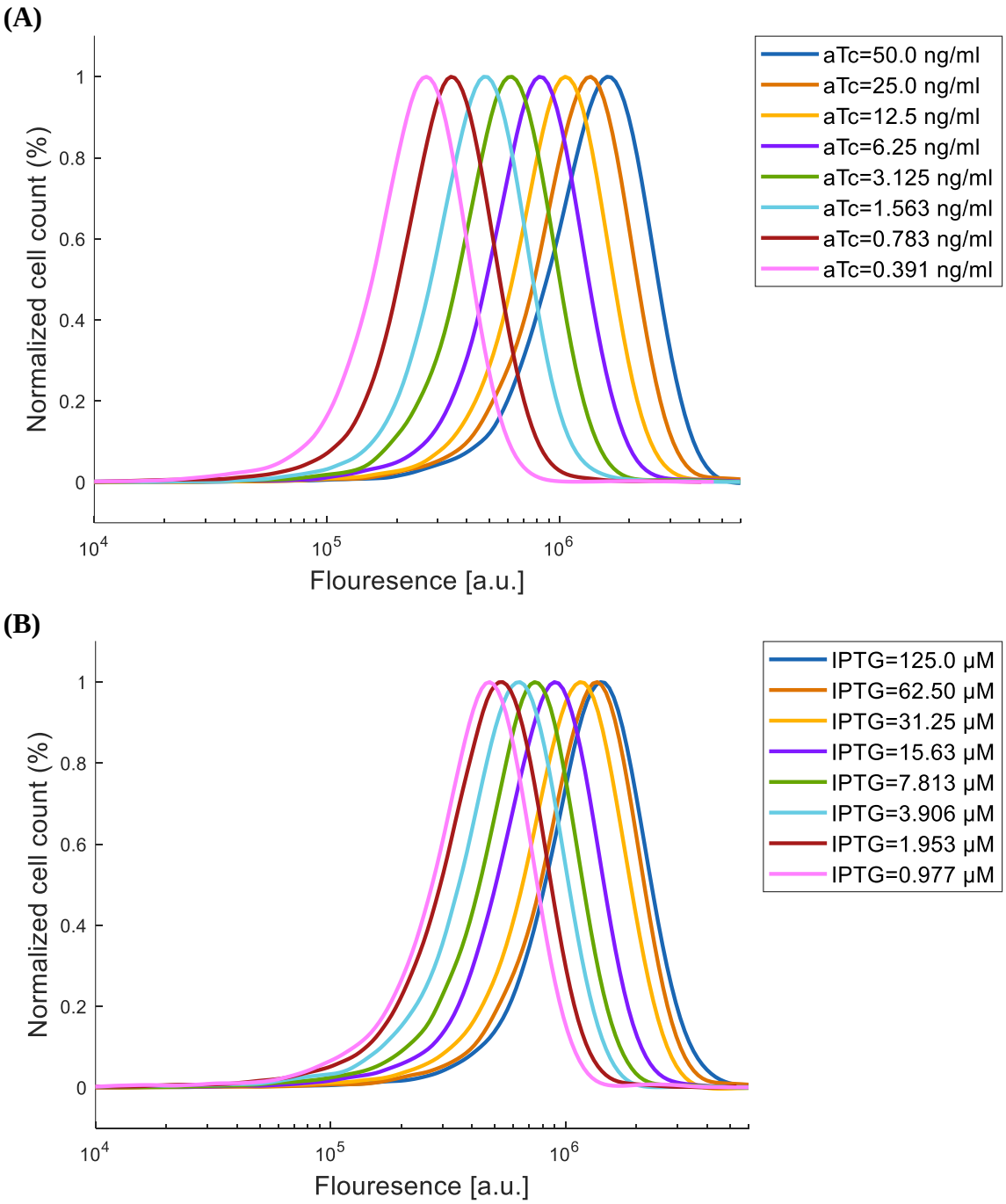


Fig. S14.2. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops (Fig. 1C). (A) IPTG was held constant at 62.5 μM, and aTc was varied. (B) aTc was held constant at 25 ng/ml, and IPTG was varied.

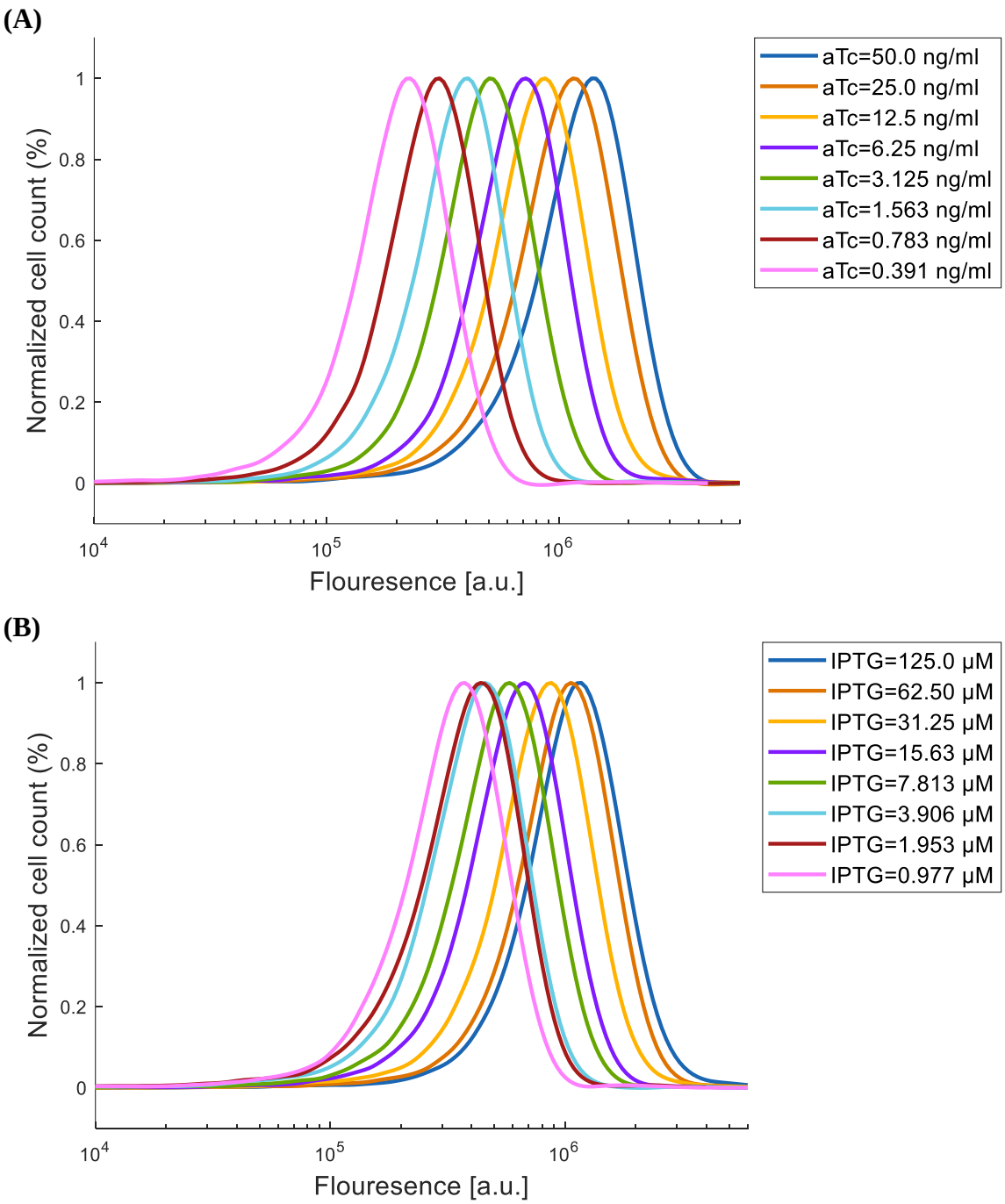


Fig. S14.3. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops (Fig. 1C). (A) IPTG was held constant at 31.25 μM, and aTc was varied. (B) aTc was held constant at 12.5 ng/ml, and IPTG was varied.

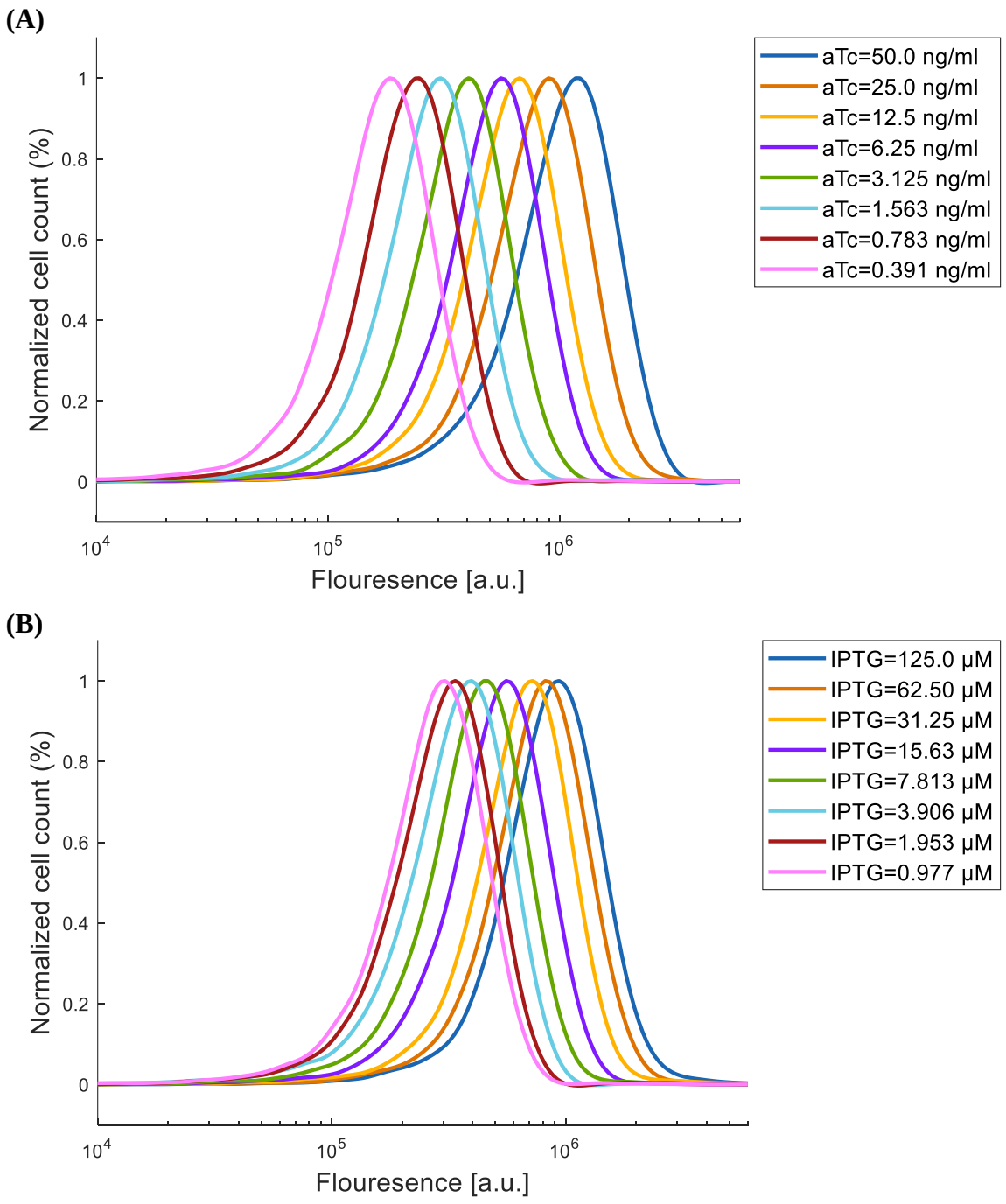


Fig. S14.4. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops (Fig. 1C). (A) IPTG was held constant at 15.63 μM, and aTc was varied. (B) aTc was held constant at 6.25 ng/ml, and IPTG was varied.

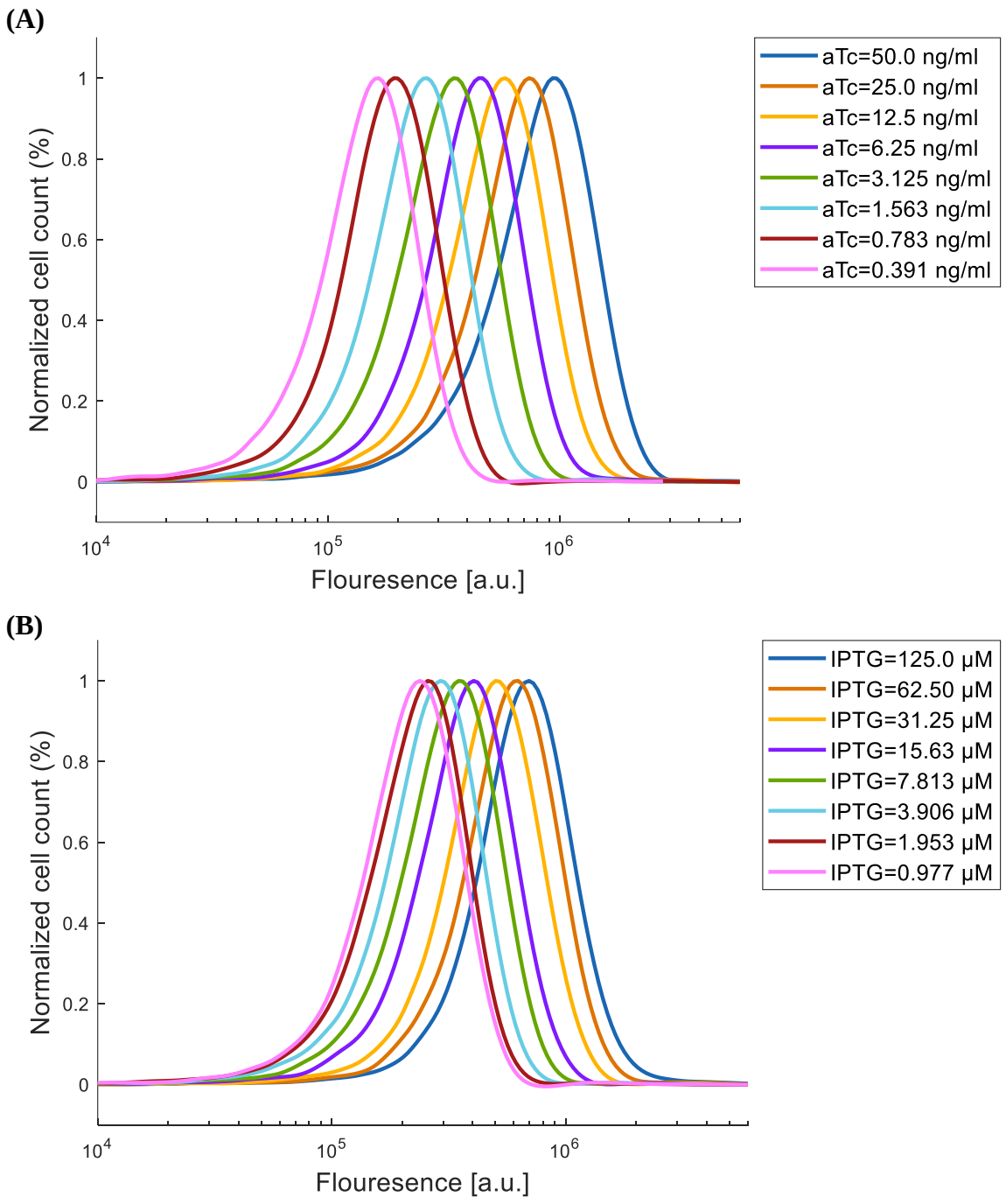


Fig. S14.5. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops (Fig. 1C). (A) IPTG was held constant at 7.813 μM, and aTc was varied. (B) aTc was held constant at 3.125 ng/ml, and IPTG was varied.

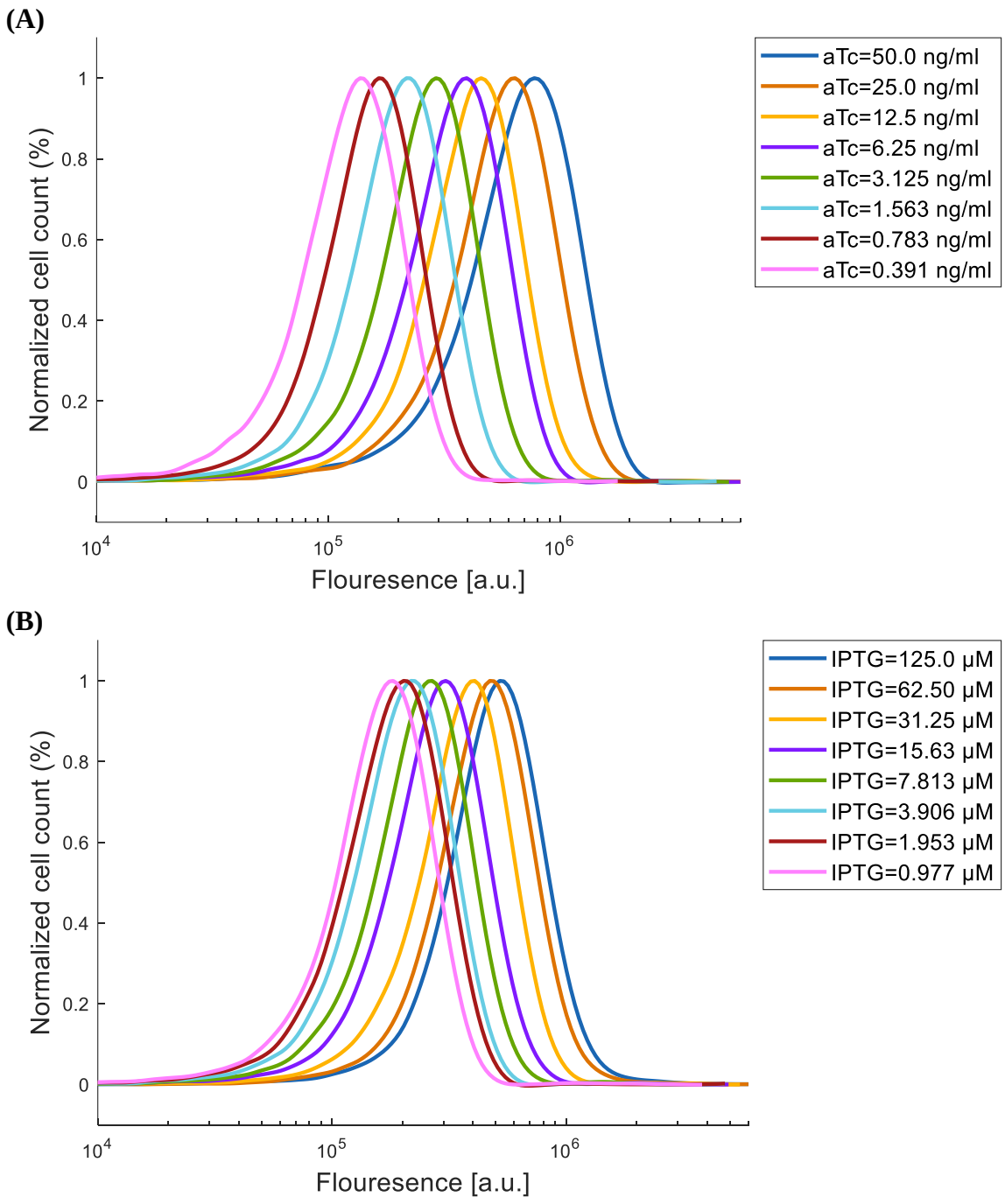


Fig. S14.6. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops (Fig. 1C). (A) IPTG was held constant at 3.906 μM and, aTc was varied. (B) aTc was held constant at 1.563 ng/ml, and IPTG was varied.

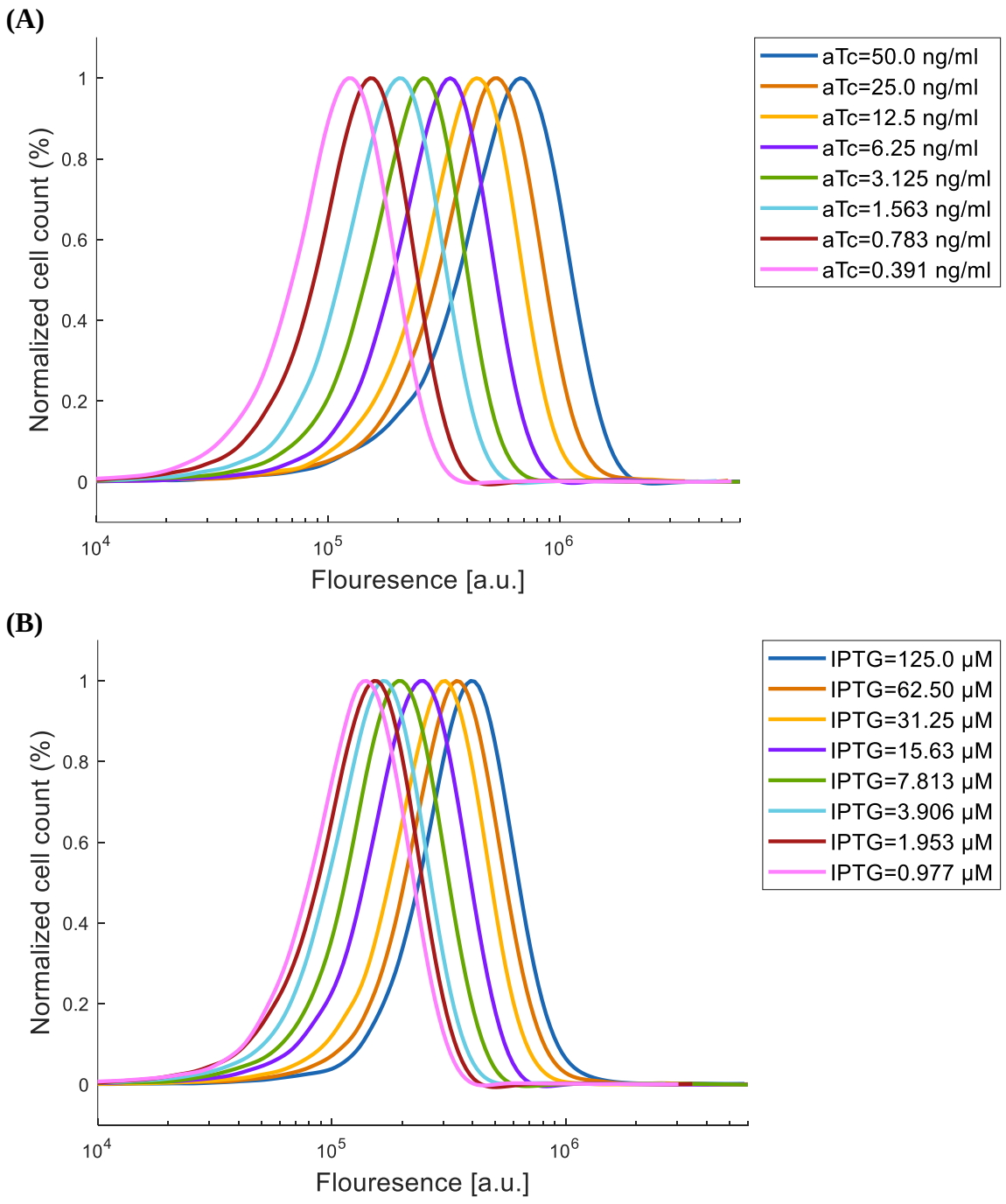


Fig. S14.7. GFP flow cytometry data for a population cells containing the synthetic perceptgene based on ANF loops (Fig. 1C). (A) IPTG was held constant at 1.953 μM and aTc was varied. (B) aTc was held constant at 0.783 ng/ml and IPTG was varied.

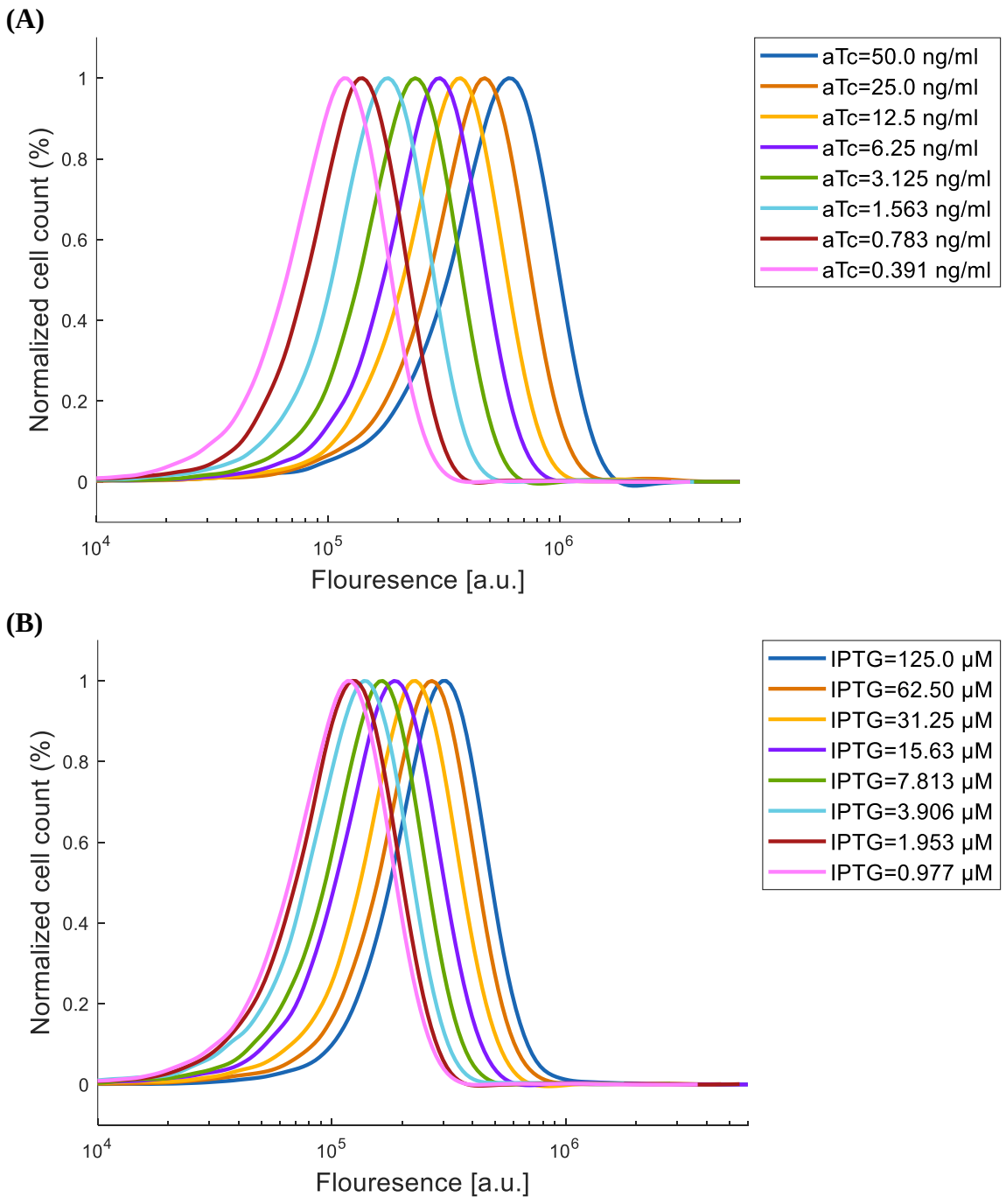


Fig. S14.8. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops (Fig. 1C). (a) IPTG was held constant at 0.977 μM, and aTc was varied. (B) aTc was held constant at 0.391 ng/ml, and IPTG was varied.

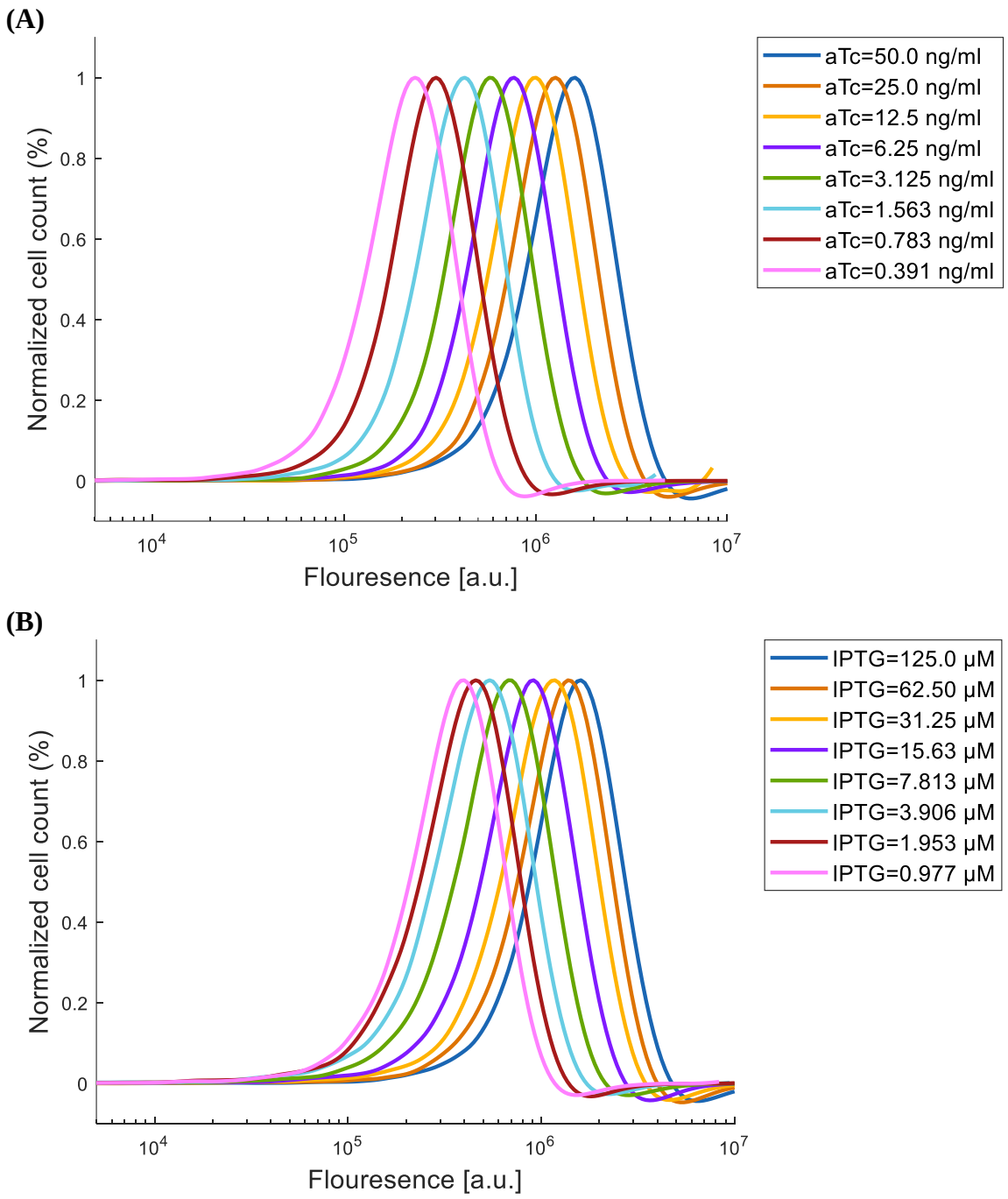


Fig. S14.9. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} (Fig. 1D). (A) IPTG was held constant at 125 μM, and aTc was varied. (B) aTc was held constant at 50 ng/ml, and IPTG was varied.

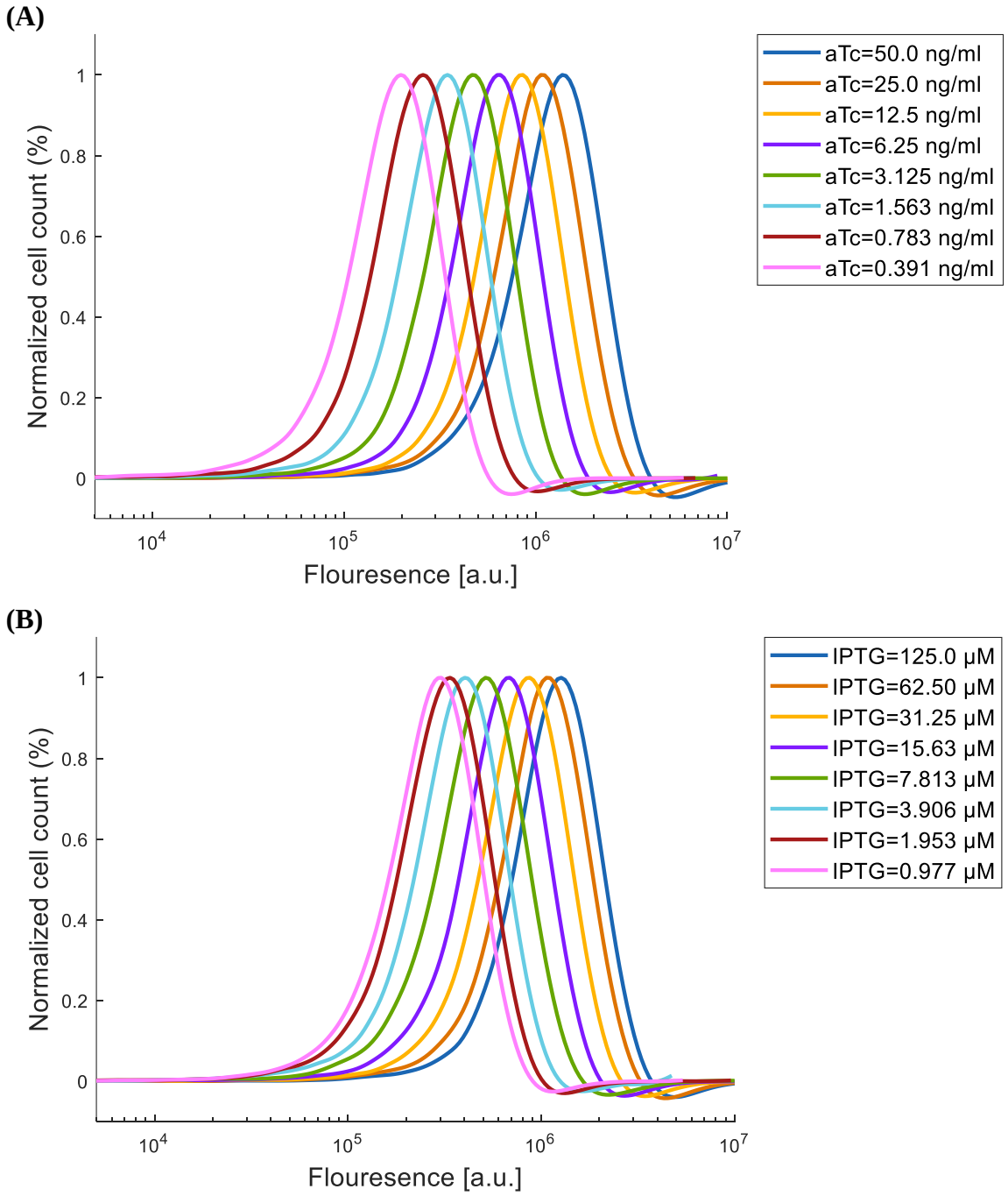


Fig. S14.10. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} (Fig. 1D). (A) IPTG was held constant at 62.5 μM, and aTc was varied. (B) aTc was held constant at 25 ng/ml, and IPTG was varied.

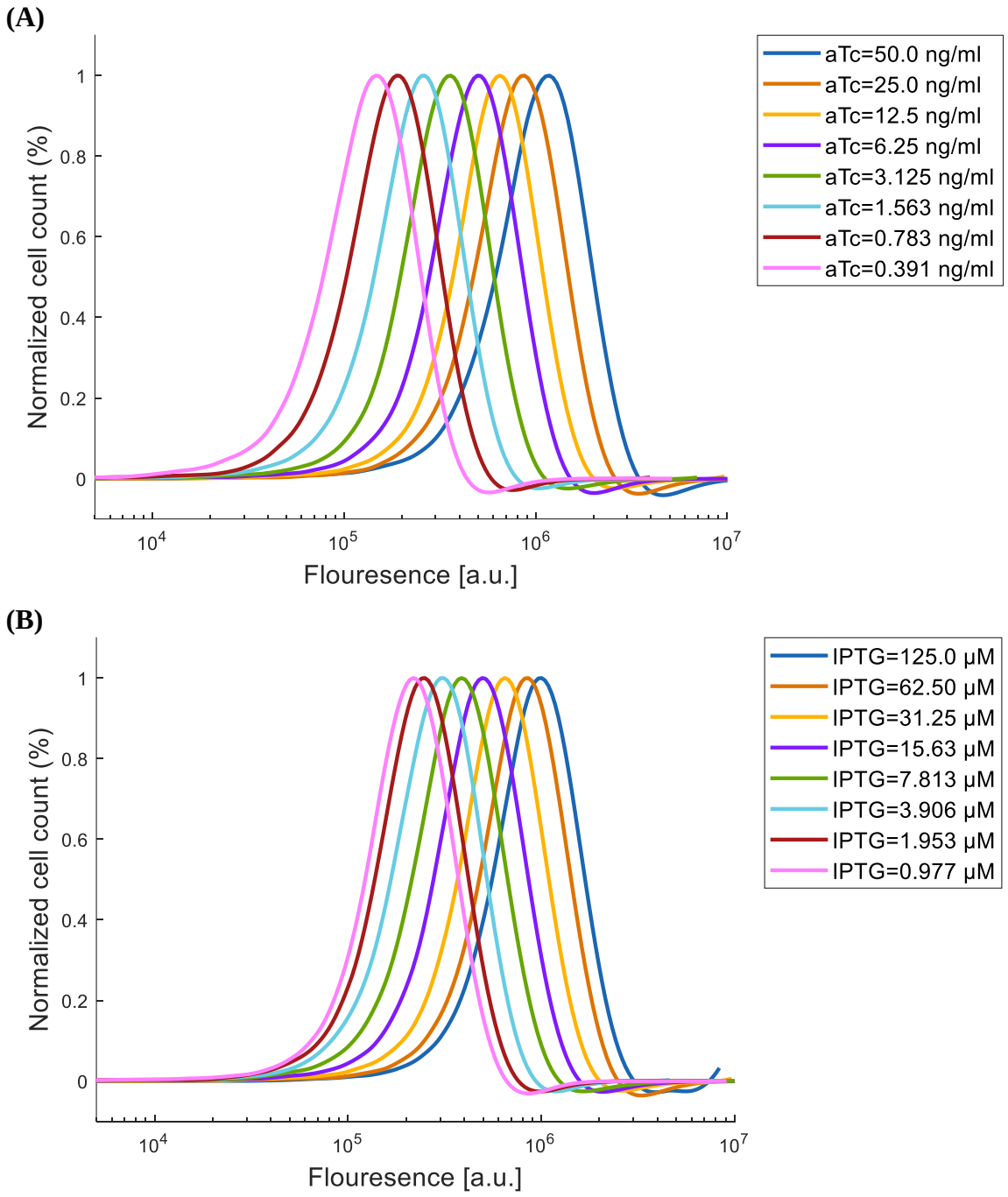


Fig. S14.11. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} (Fig. 1D). (A) IPTG was held constant at 31.25 μM, and aTc was varied. (B) aTc was held constant at 12.5 ng/ml, and IPTG was varied.

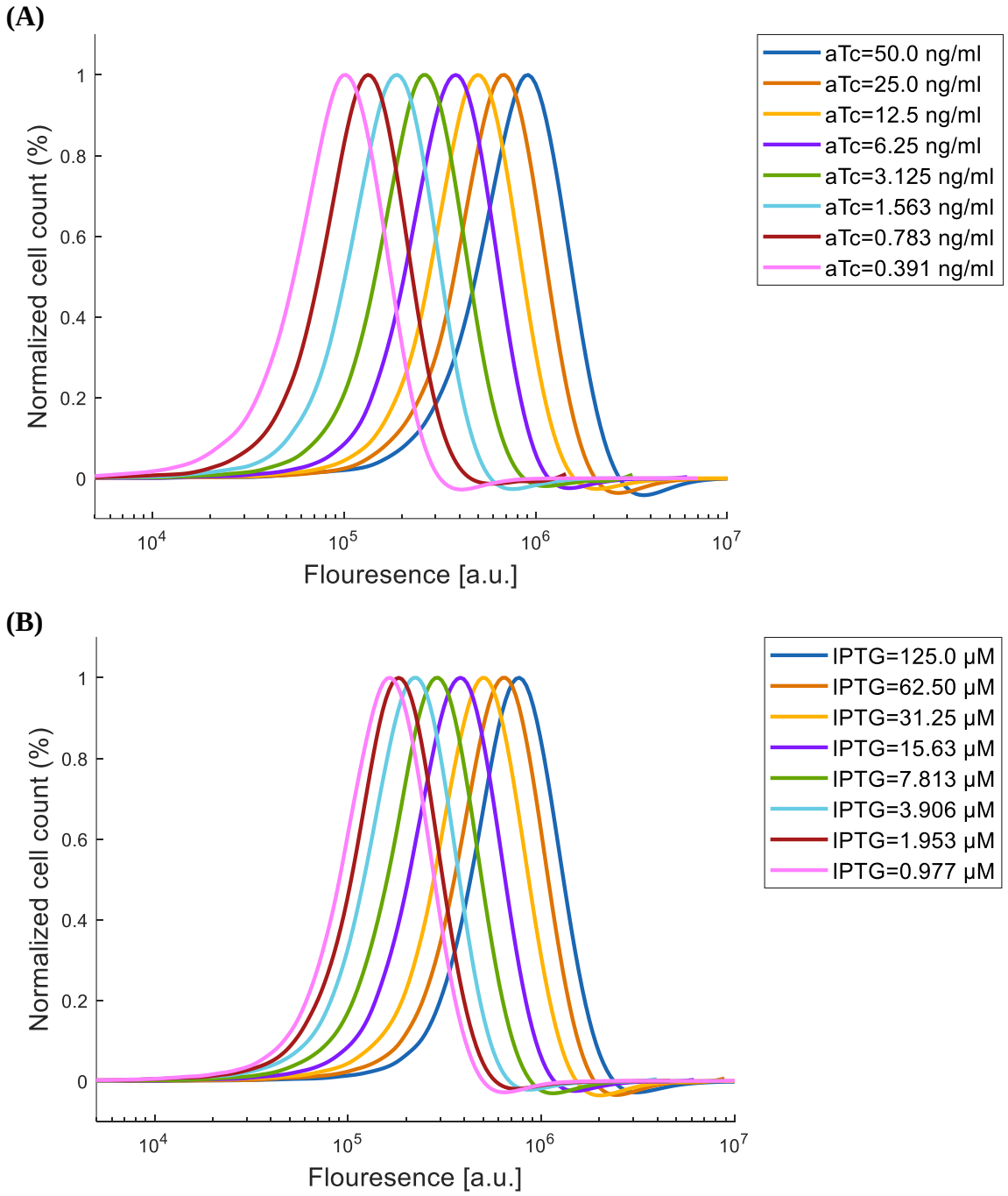


Fig. S14.12. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} (Fig. 1D). (A) IPTG was held constant at 15.63 μM, and aTc was varied. (B) aTc was held constant at 6.25 ng/ml,and IPTG was varied.

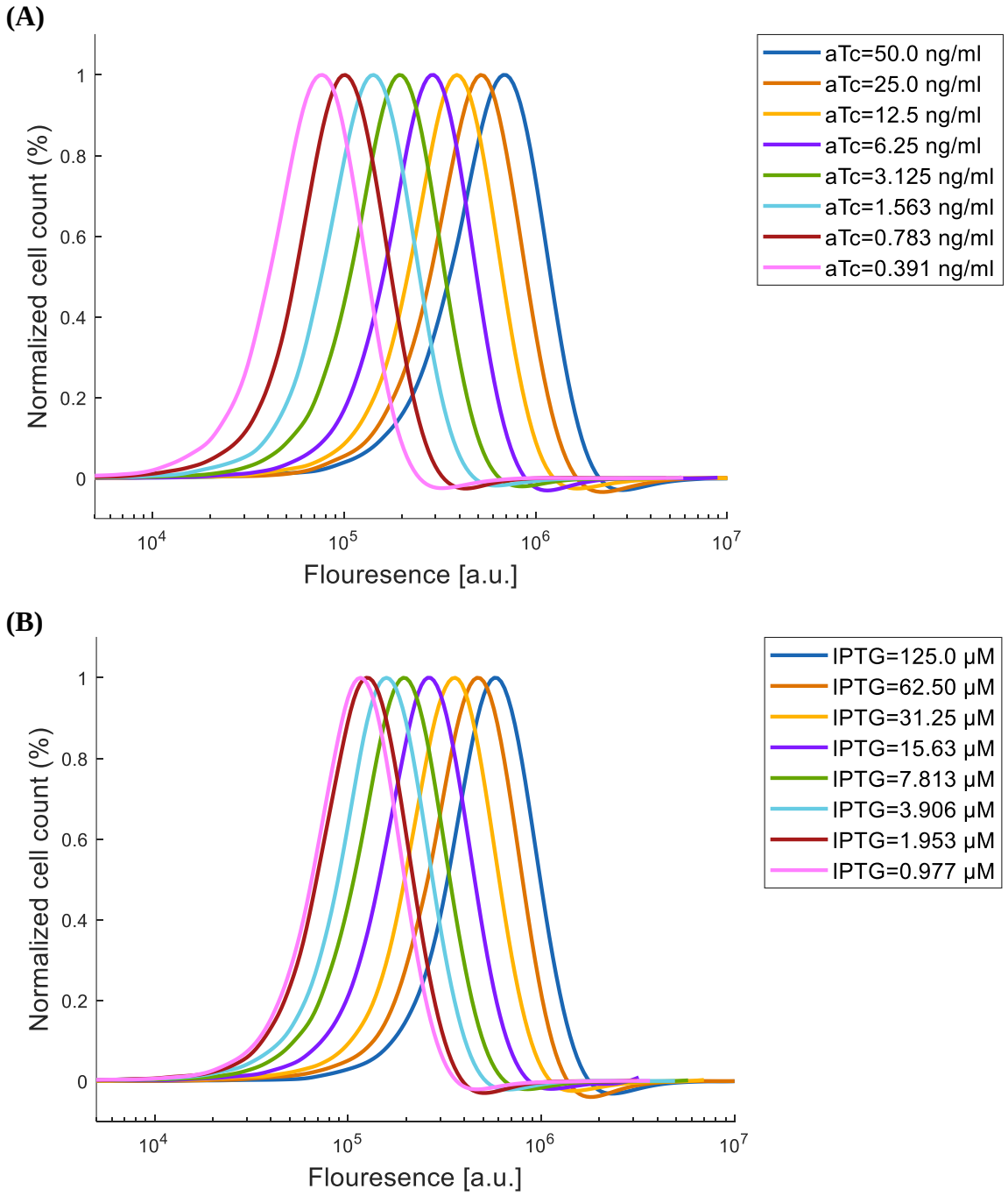


Fig. S14.13. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} (Fig. 1D). (A) IPTG was held constant at 7.813 μM and aTc was varied. (B) aTc was held constant at 3.125 ng/ml and IPTG was varied.

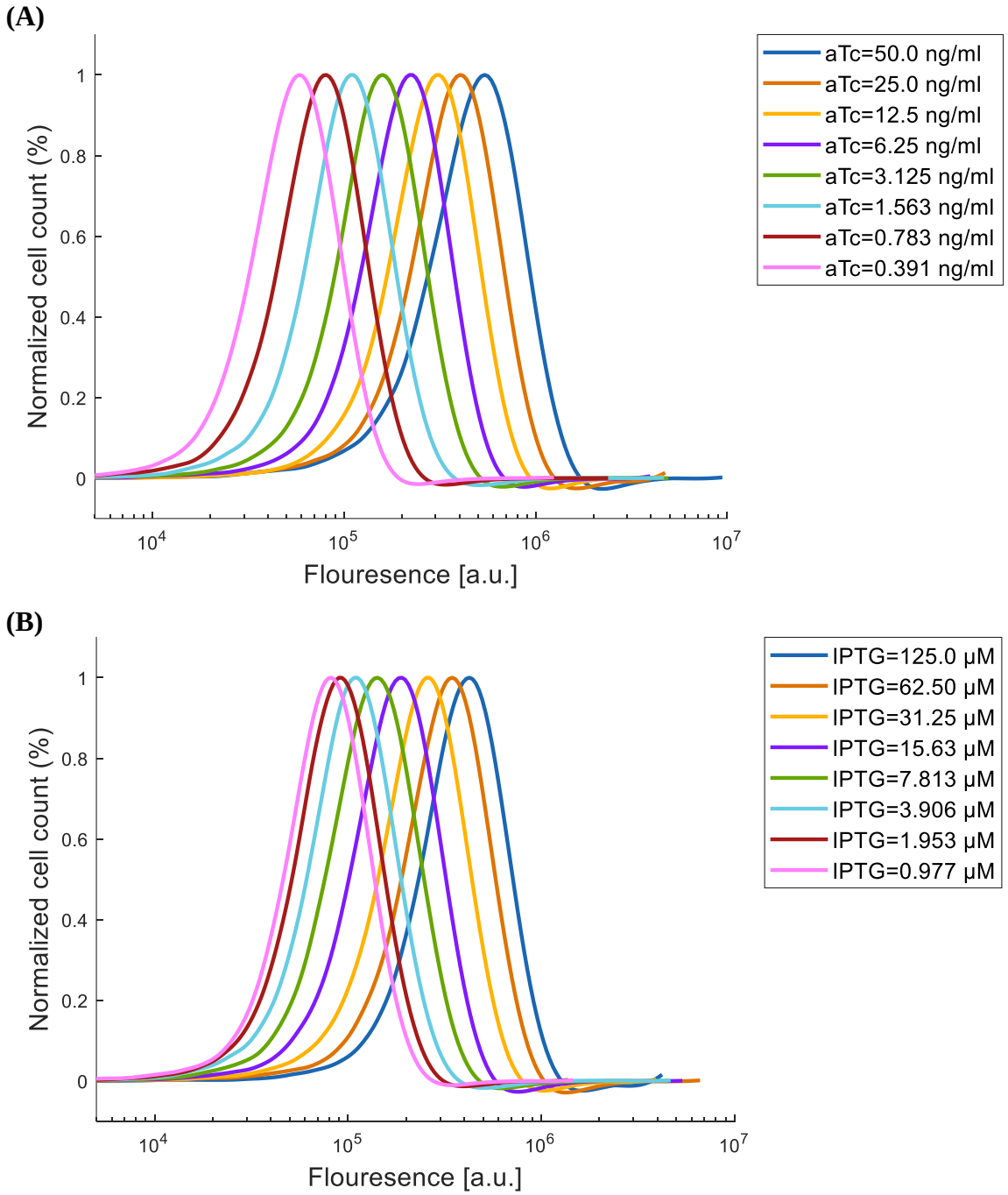


Fig. S14.14. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} (Fig. 1D). (A) IPTG was held constant at 3.906 μM, and aTc was varied. (B) aTc was held constant at 1.563 ng/ml, and IPTG was varied.

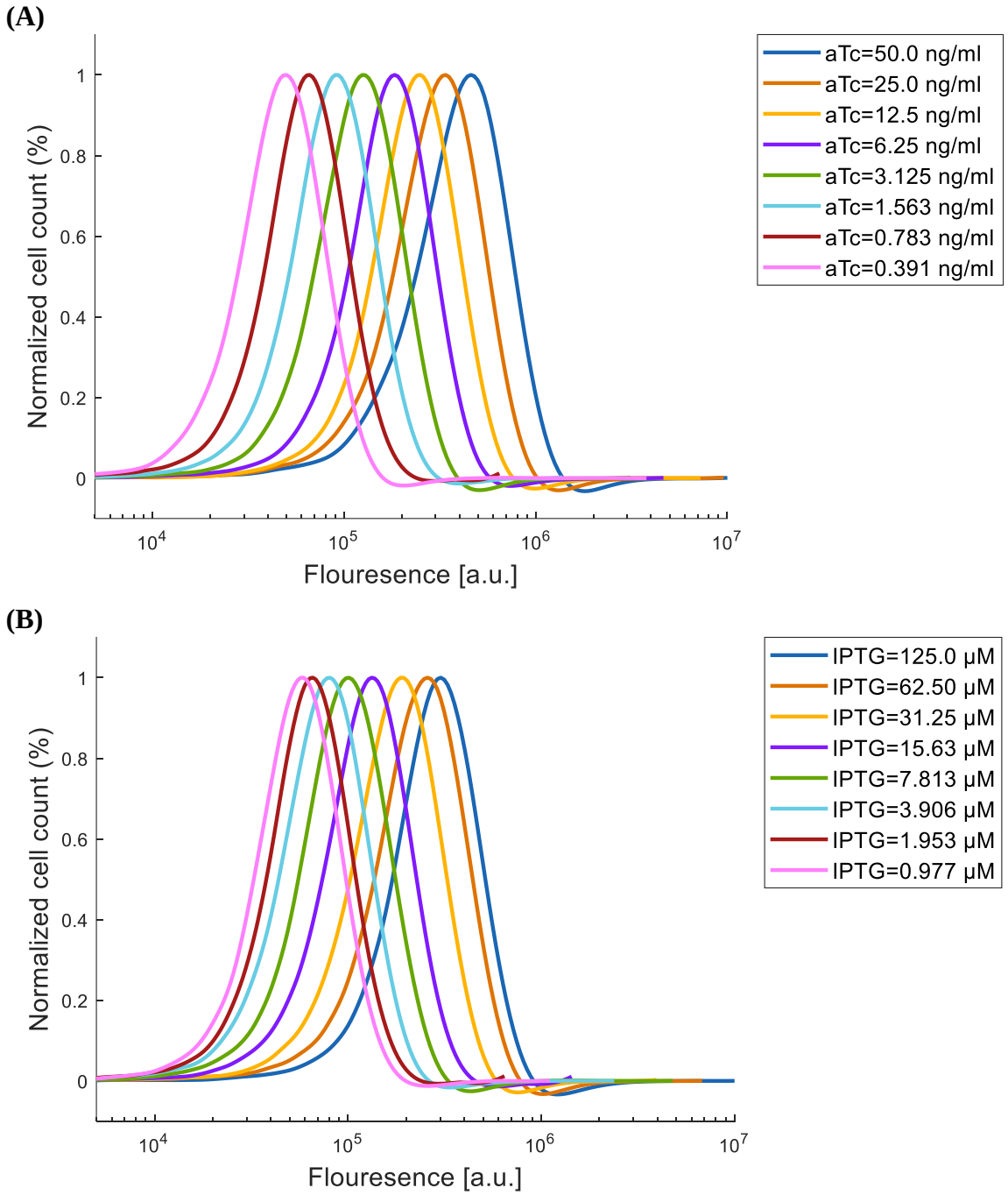


Fig. S14.15. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} (Fig. 1D). (A) IPTG was held constant at 1.953 μM, and aTc was varied. (B) aTc was held constant at 0.783 ng/ml, and IPTG was varied.

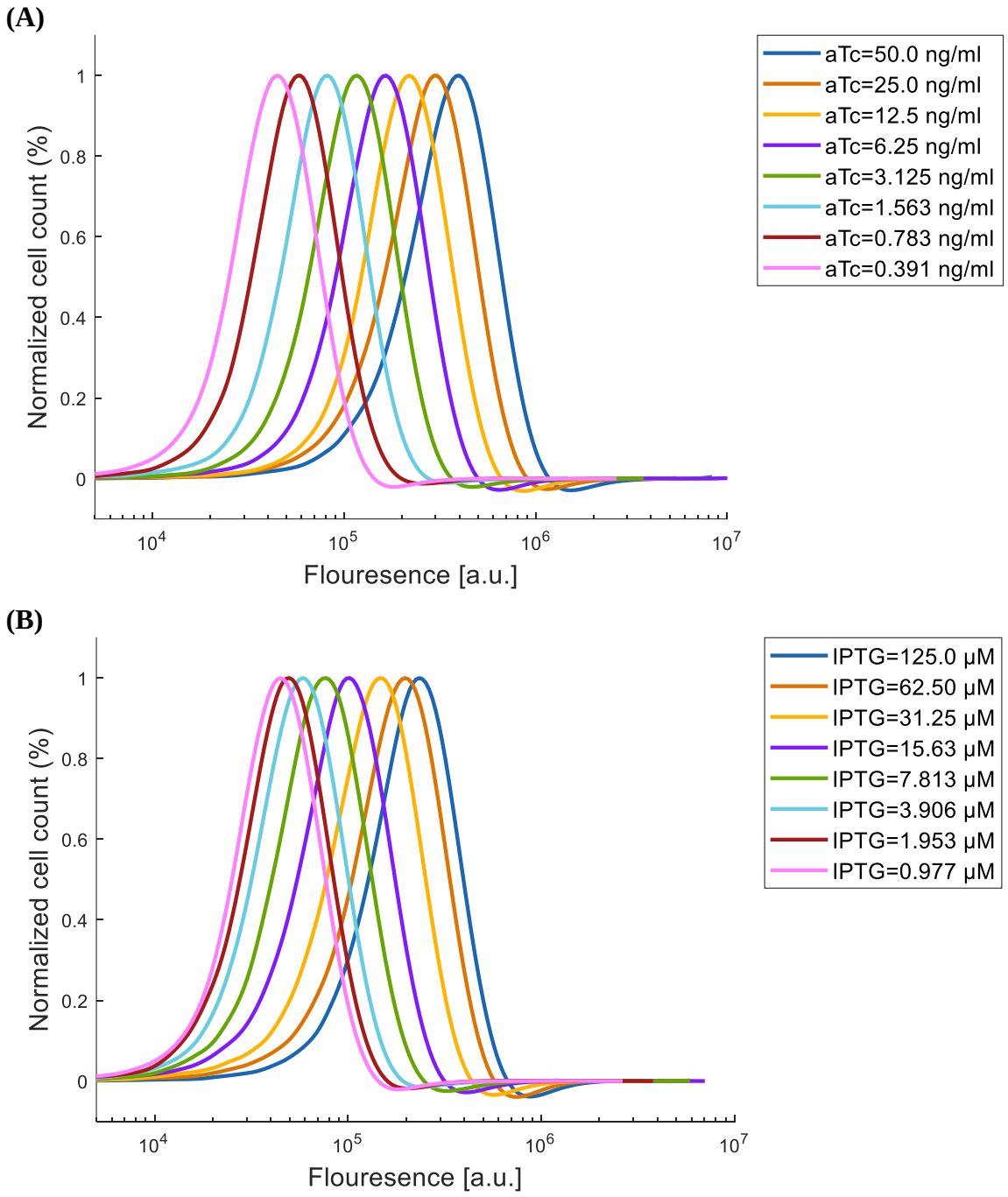


Fig. S14.16. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} (Fig. 1D). (A) IPTG was held constant at 0.977 μM, and aTc was varied. (B) aTc was held constant at 0.391 ng/ml, and IPTG was varied.

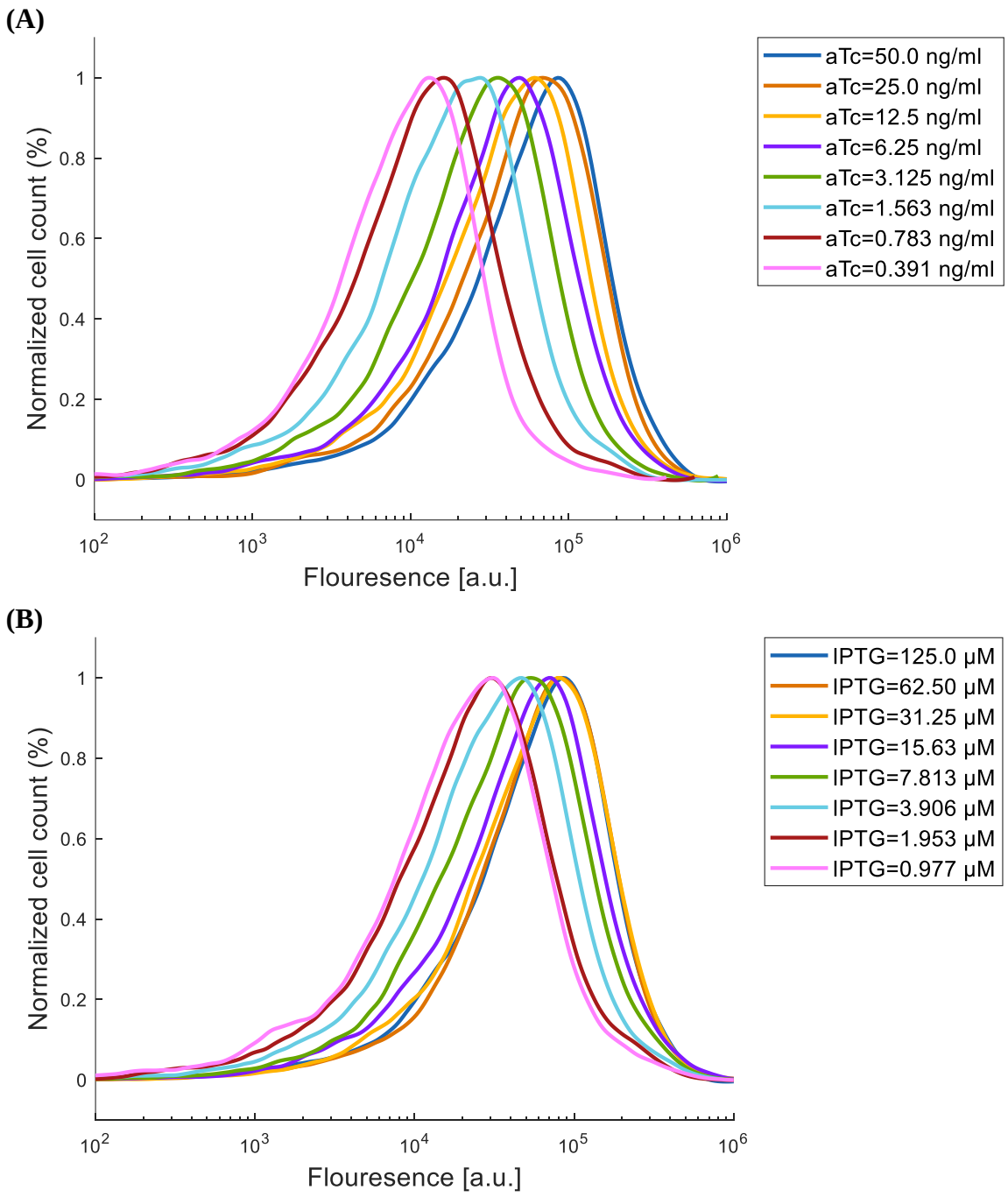


Fig. S14.17. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO}within the ANF was replaced by P_{lacO1} and AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. 1H). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 125 μM, and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 50 ng/ml, and IPTG was varied.

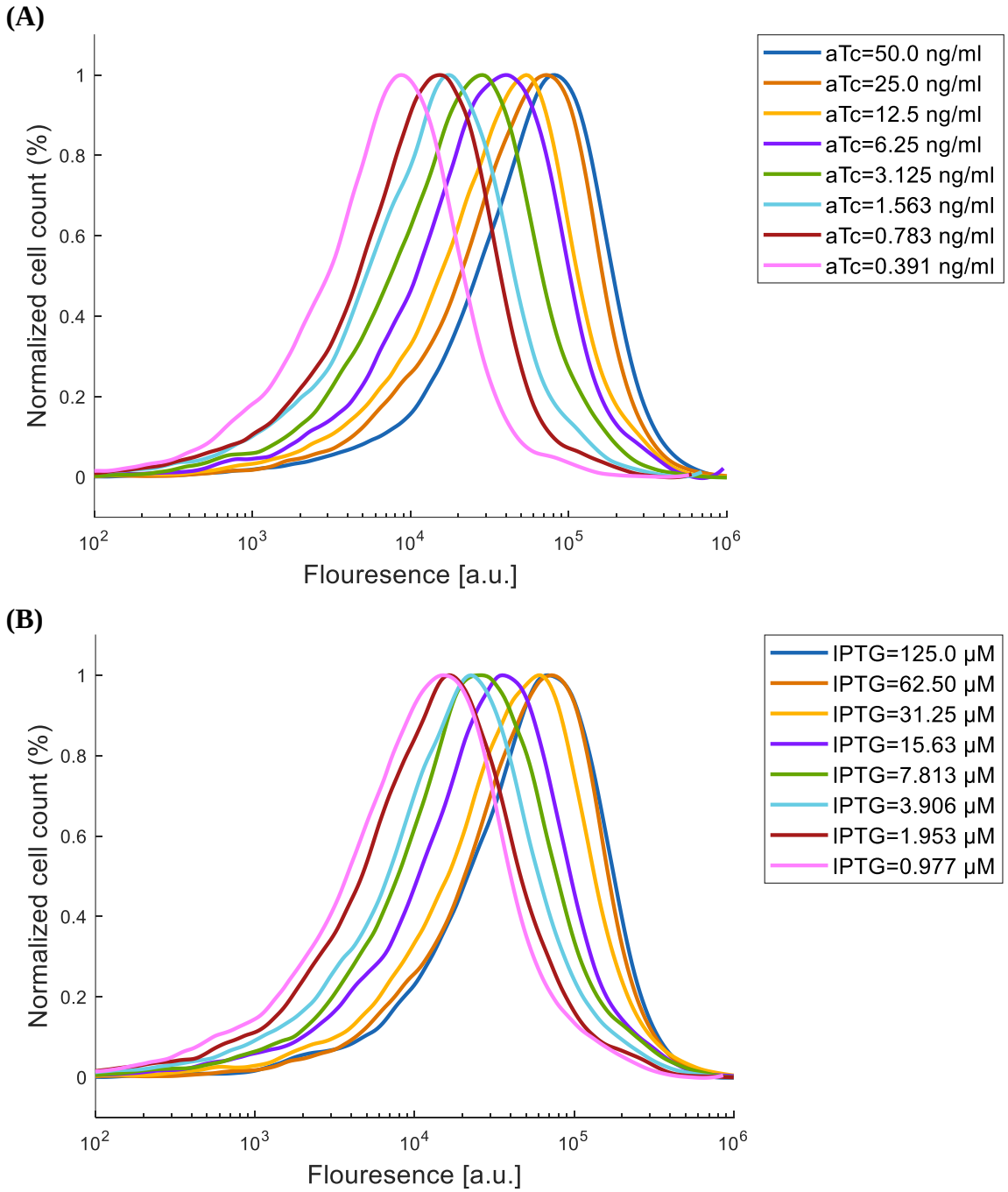


Fig. S14.18. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} and AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. 1H). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 62.5 µM, and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 25 ng/ml, and IPTG was varied.

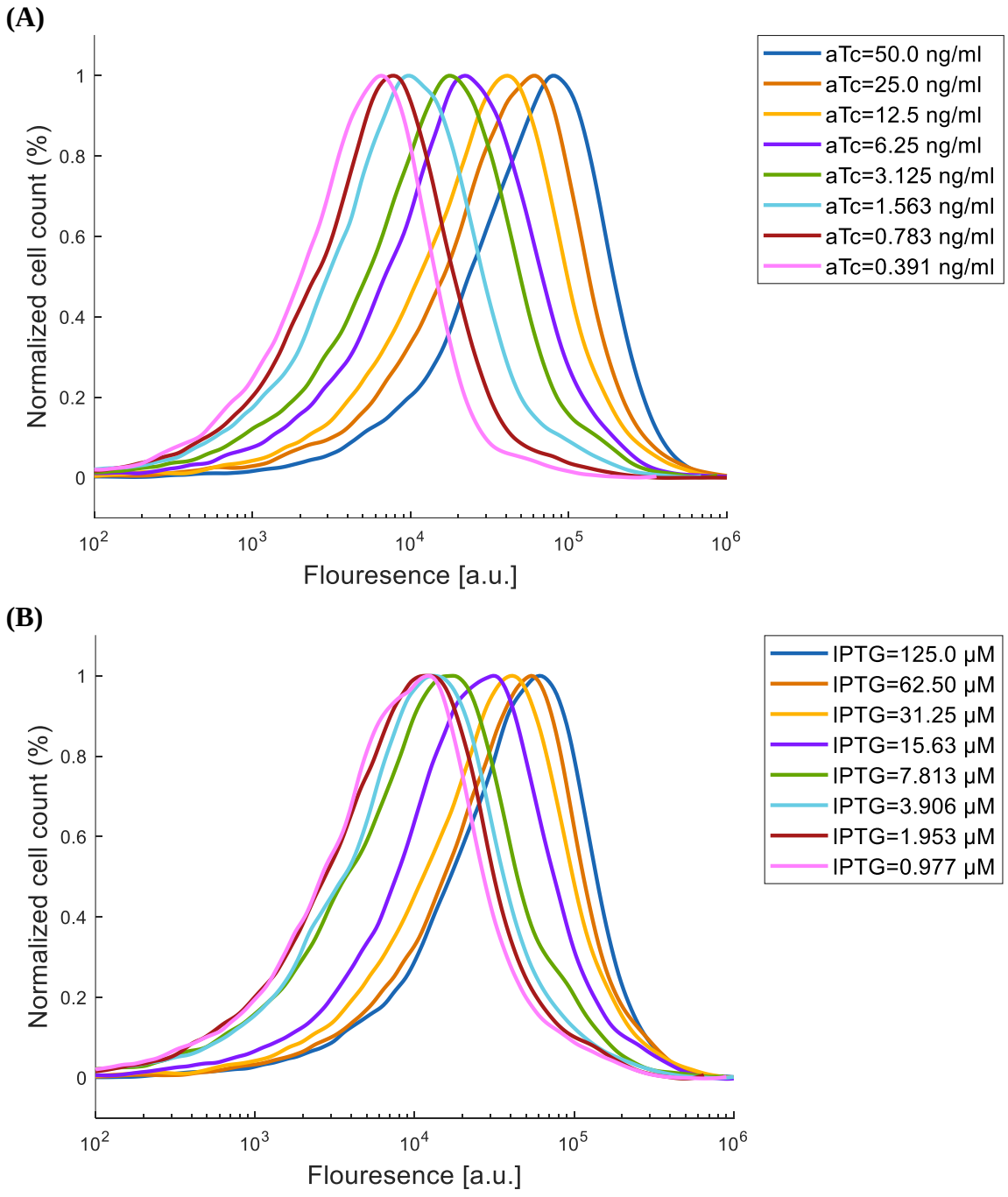


Fig. S14.19. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} and AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. 1H). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 31.25 µM and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 12.5 ng/ml and IPTG was varied.

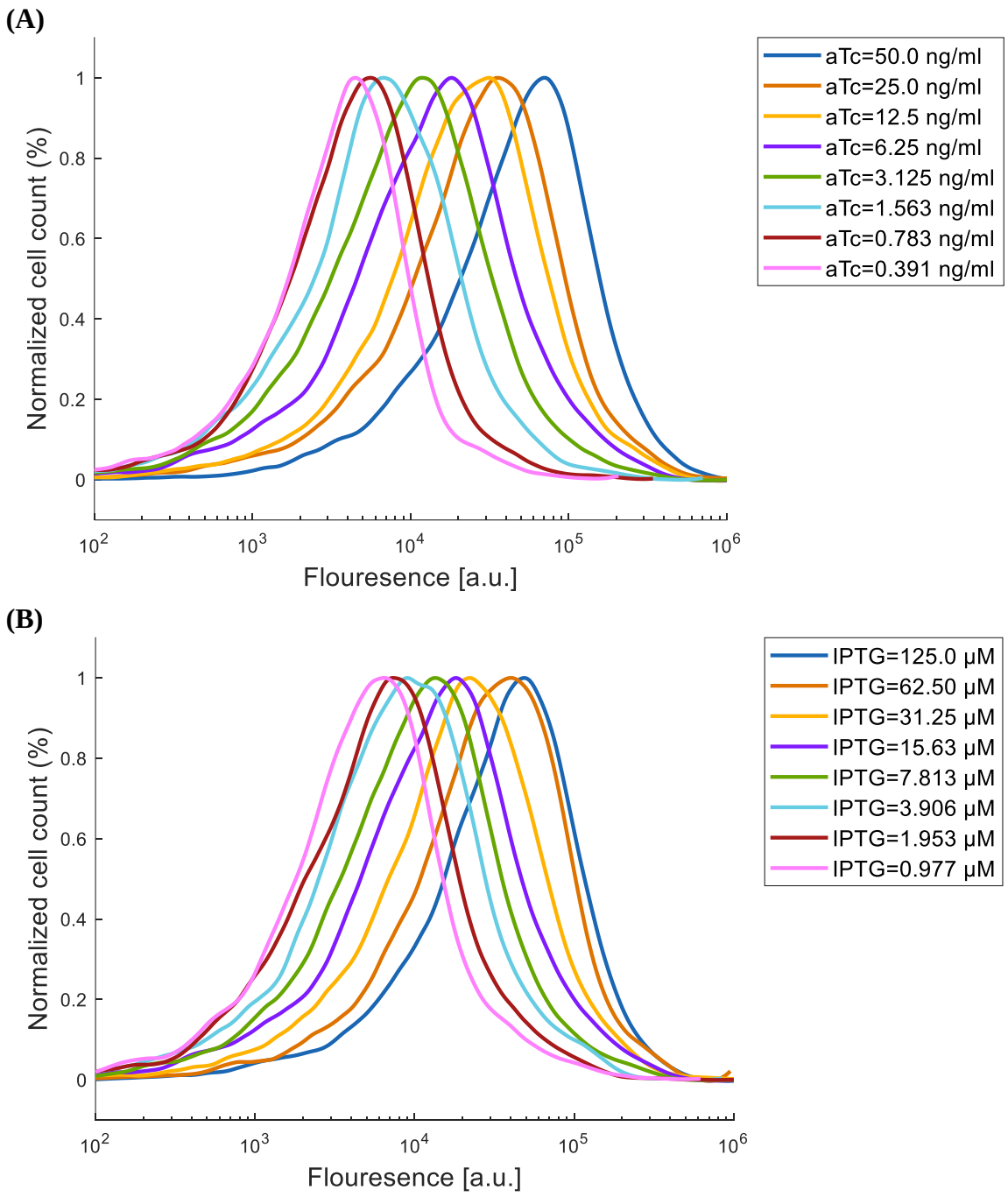


Fig. S14.20. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} and AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. 1H). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 15.63 μM, and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 6.25 ng/ml, and IPTG was varied.

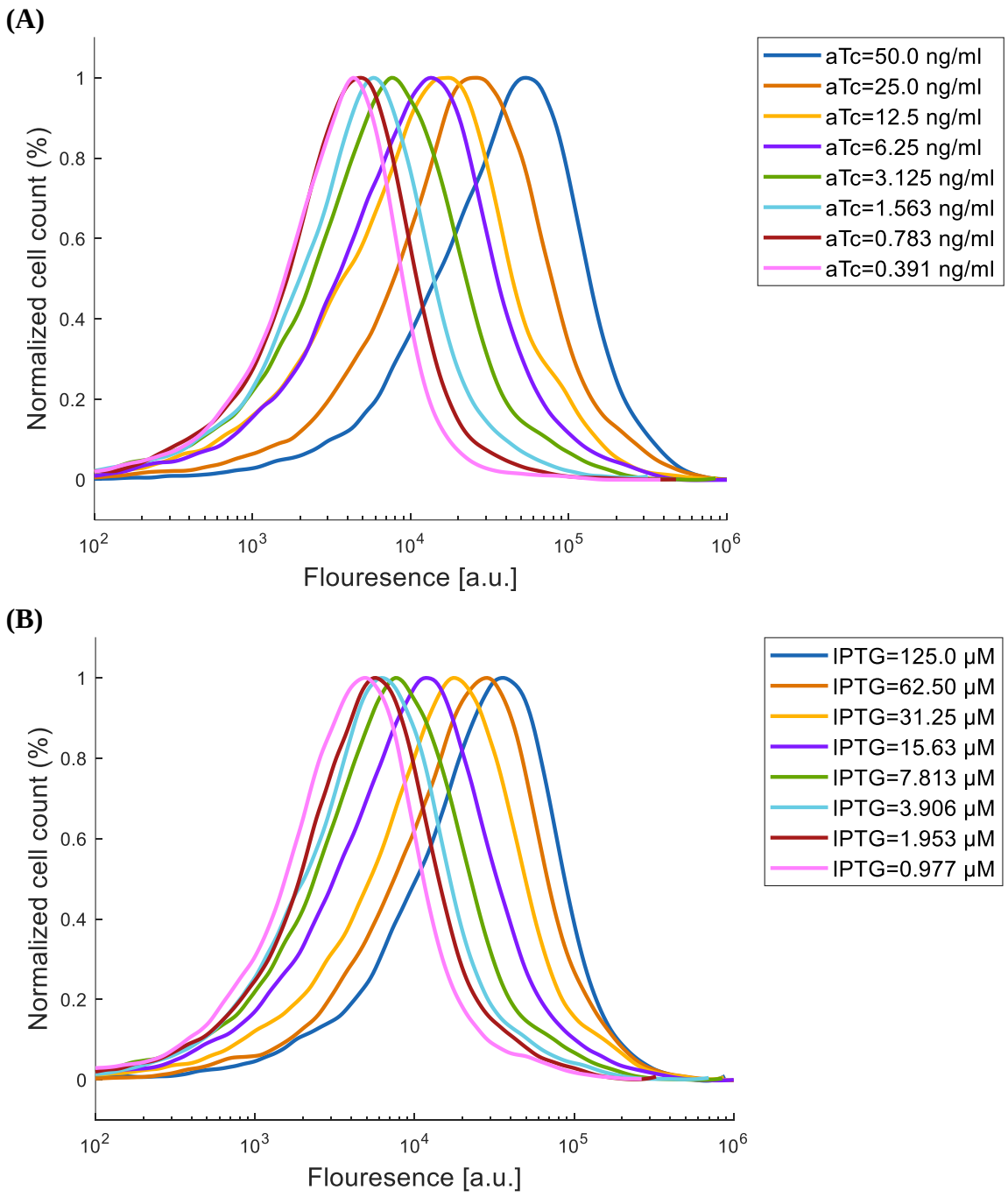


Fig. S14.21. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} and AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. 1H). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 7.813 μM, and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 3.125 ng/ml, and IPTG was varied.

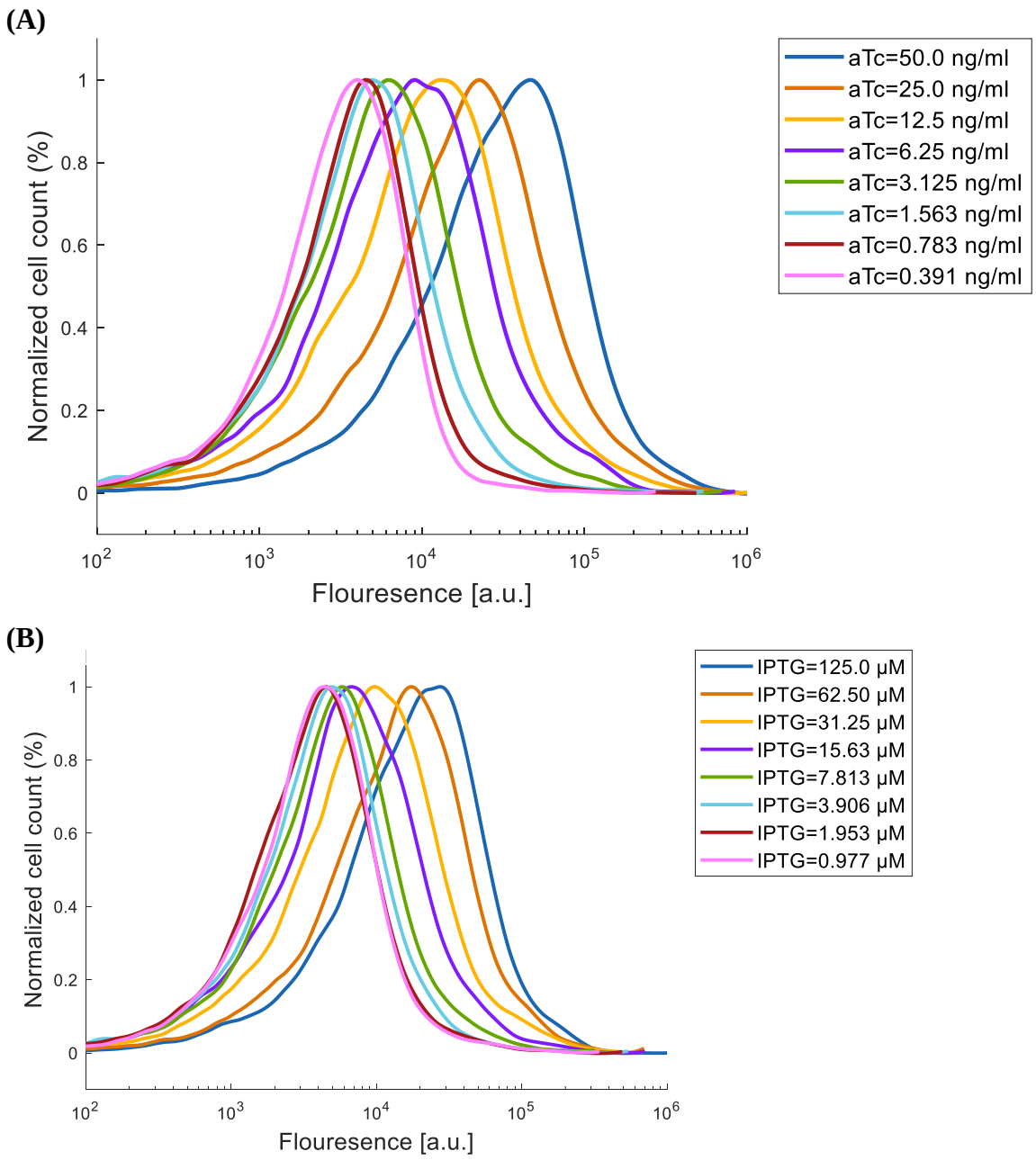


Fig. S14.22. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} and AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. 1H). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 3.906 µM, and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 1.563 ng/ml, and IPTG was varied.

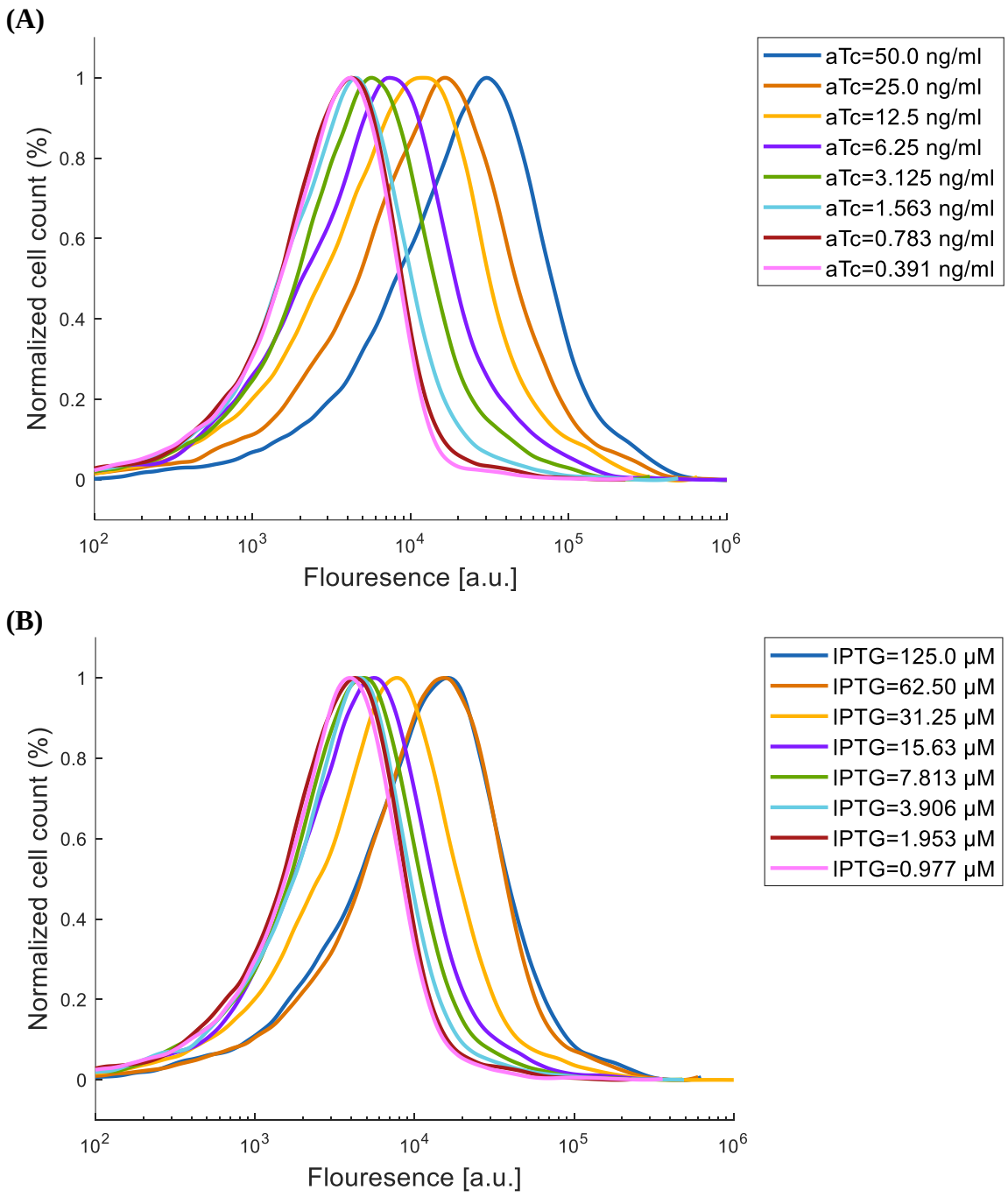


Fig. S14.23. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} and AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. 1H). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 1.953 μM, and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 0.783 ng/ml, and IPTG was varied.

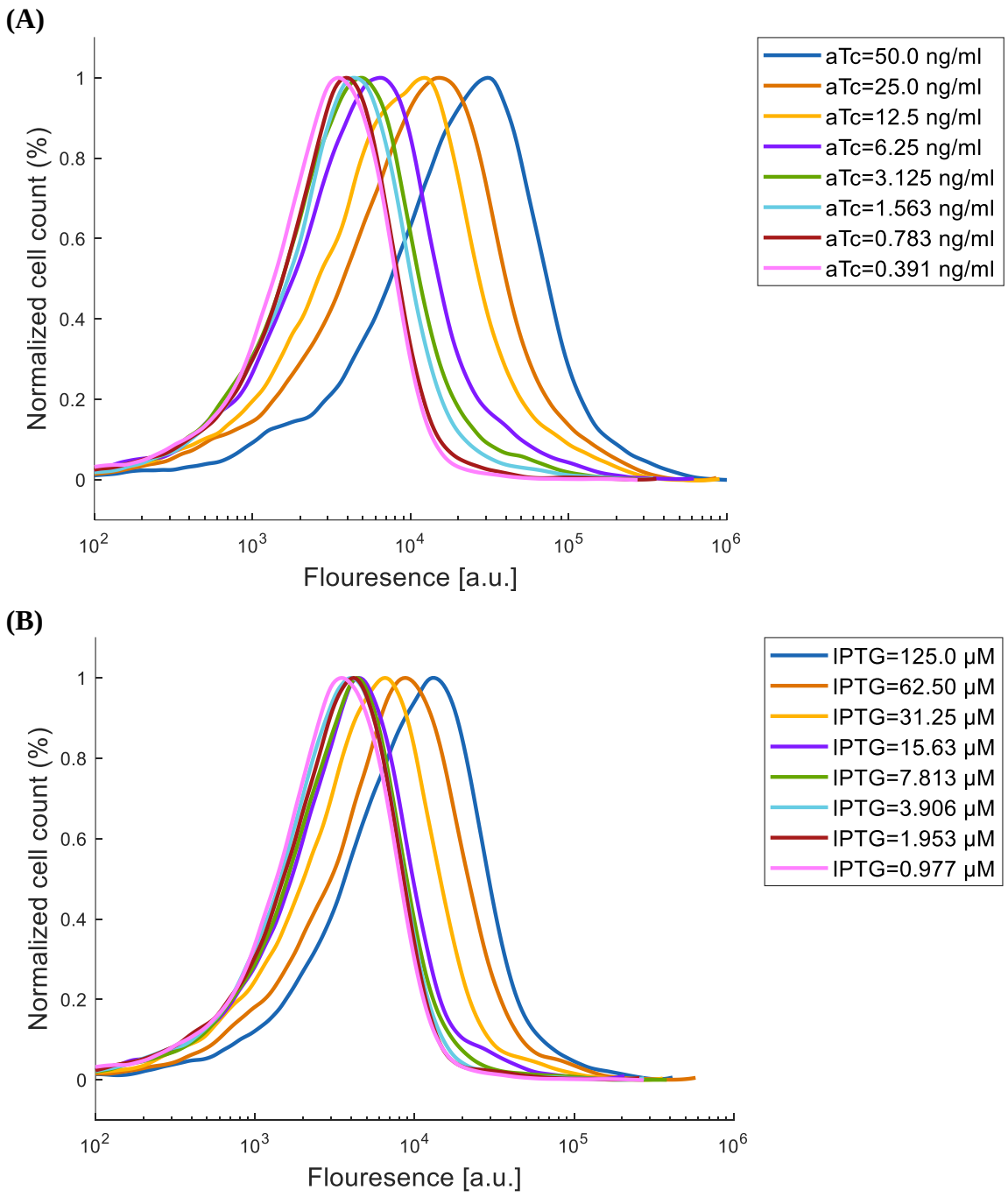


Fig. S14.24. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF loops. In this circuit, P_{lacO} within the ANF was replaced by P_{lacO1} and AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. 1H). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 0.977 μM, and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 0.391 ng/ml, and IPTG was varied.

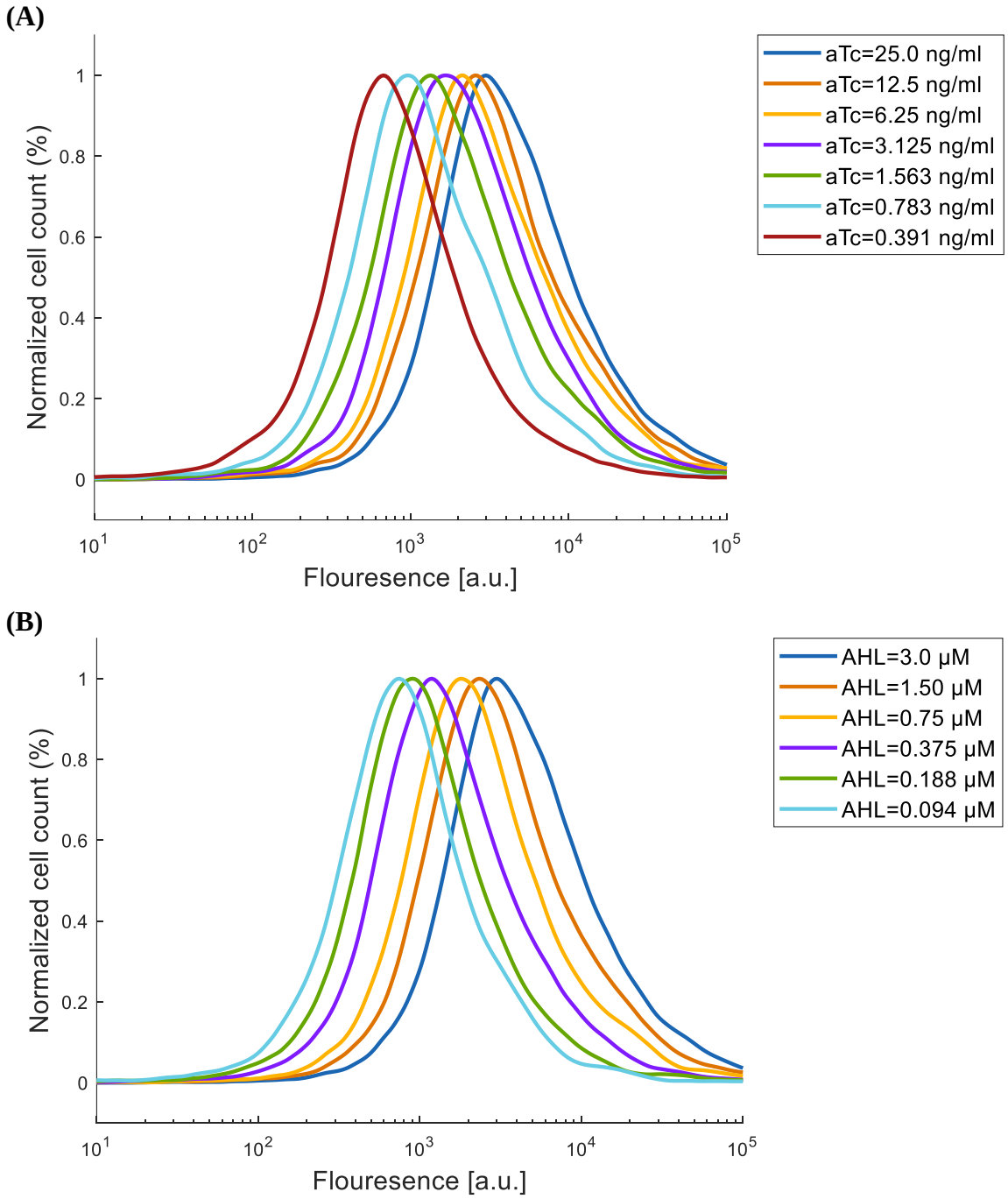


Fig. S14.25. mCherry flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop (Fig. 2B). (A) AHL was held constant at 3.0 μM and aTc was varied. (b) aTc was held constant at 25 ng/ml and AHL was varied.

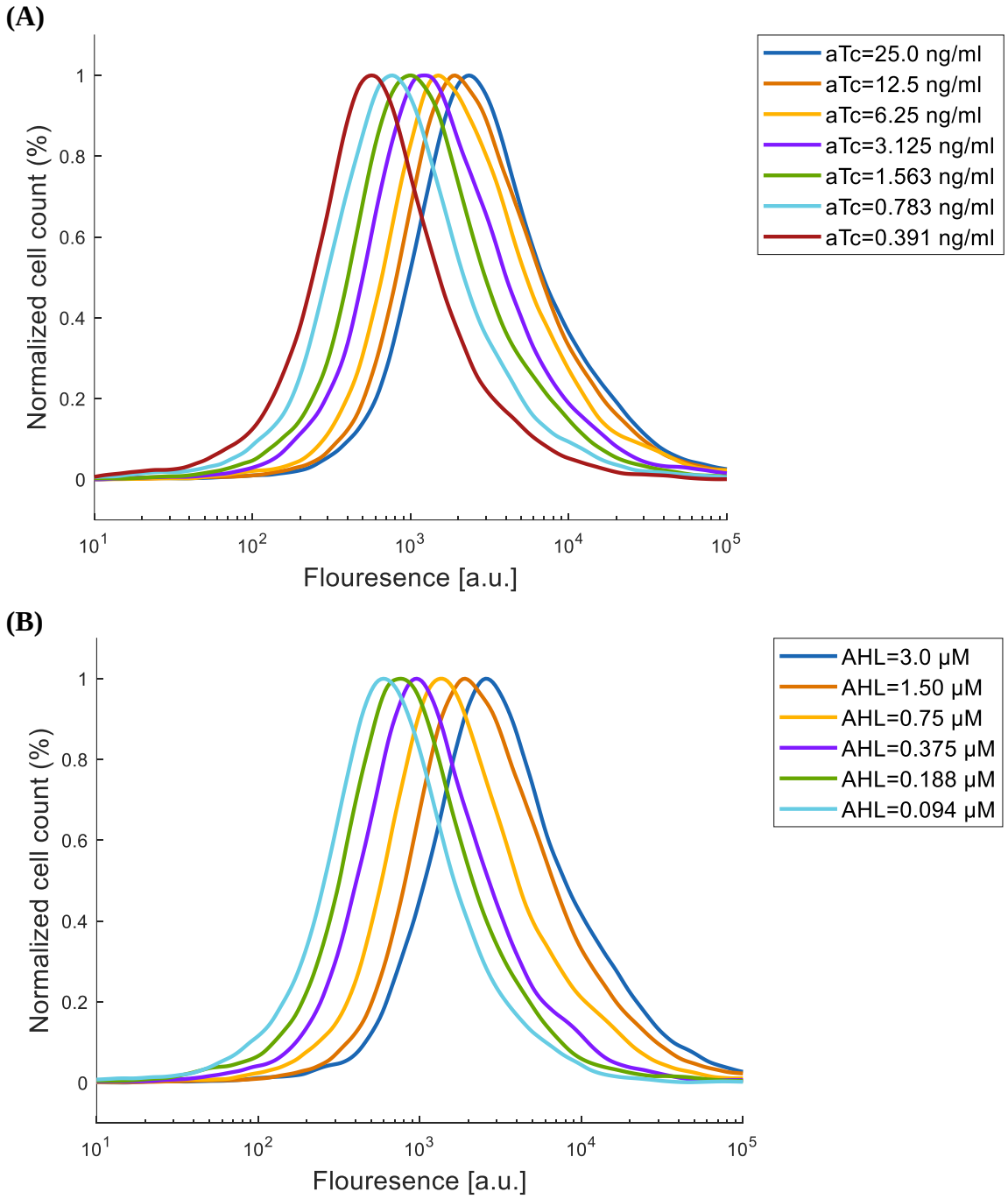


Fig. S14.26. mCherry flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop (Fig. 2B). (A) AHL was held constant at 1.50 μM and aTc was varied. (B) aTc was held constant at 12.5 ng/ml and AHL was varied.

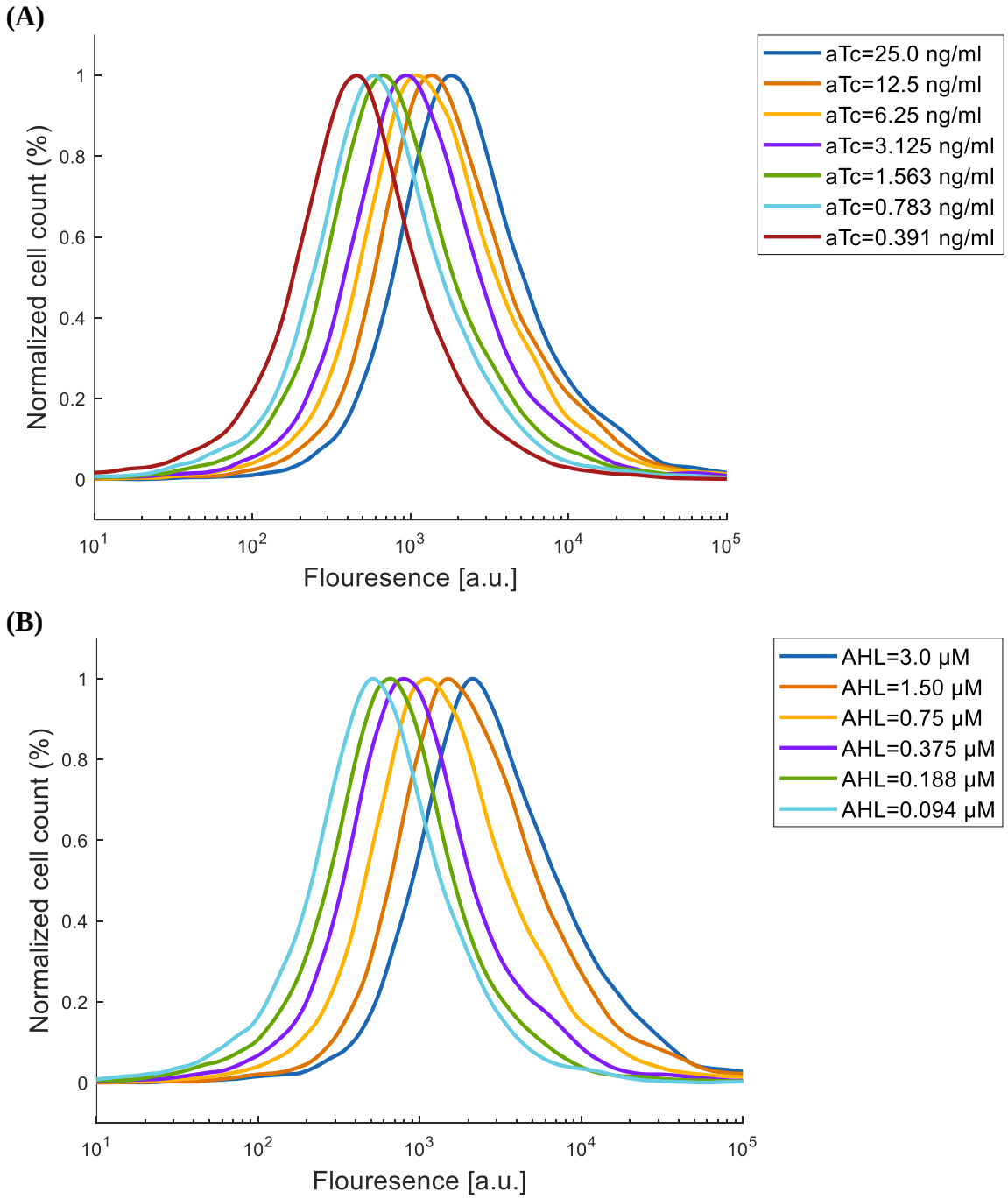


Fig. S14.27. mCherry flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop (Fig. 2B). (A) AHL was held constant at 0.75 μM and aTc was varied. (B) aTc was held constant at 6.25 ng/ml and AHL was varied

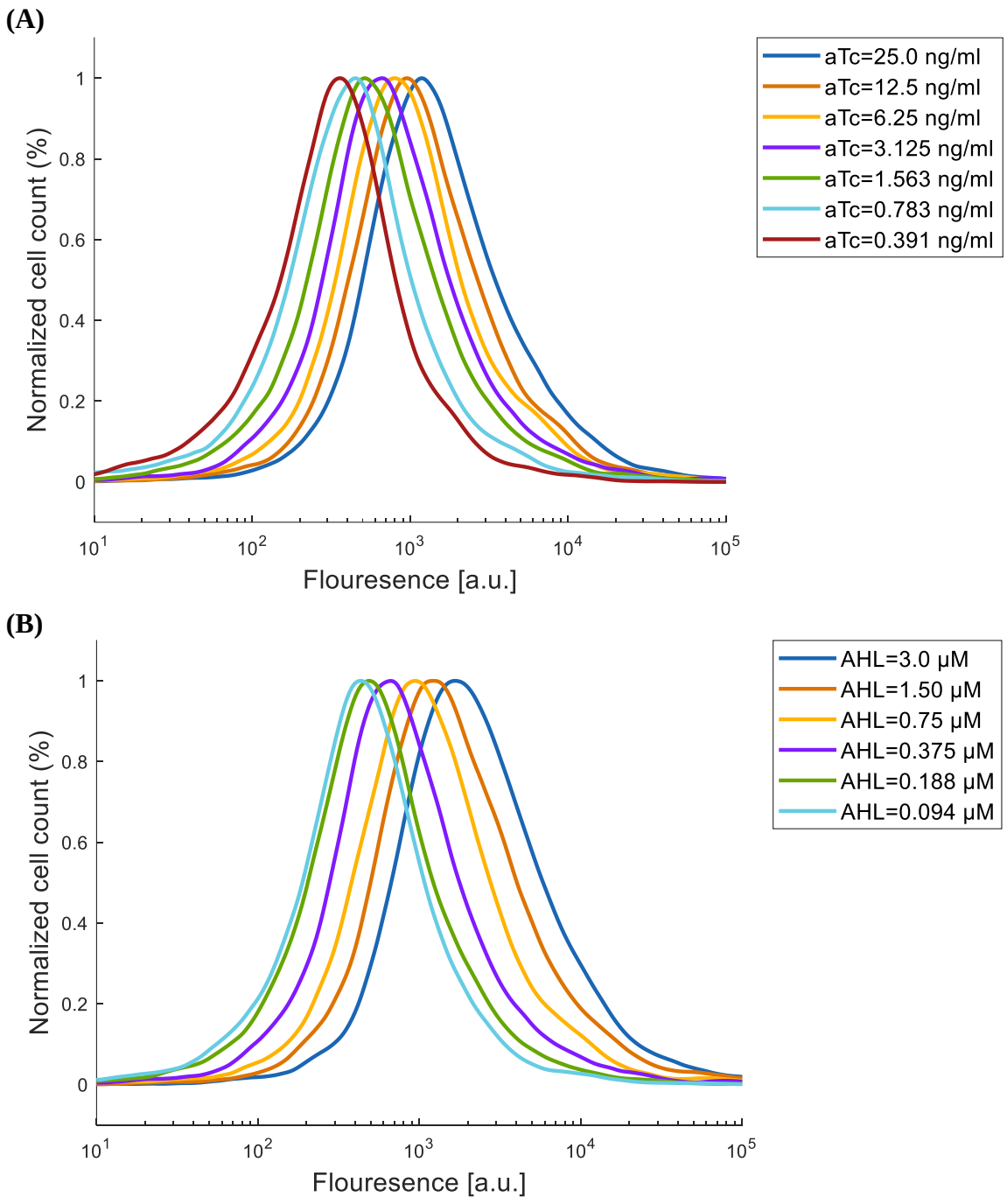


Fig. S14.28. mCherry flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop (Fig. 2B). (A) AHL was held constant at 0.375 μM and aTc was varied. (B) aTc was held constant at 3.125 ng/ml and AHL was varied.

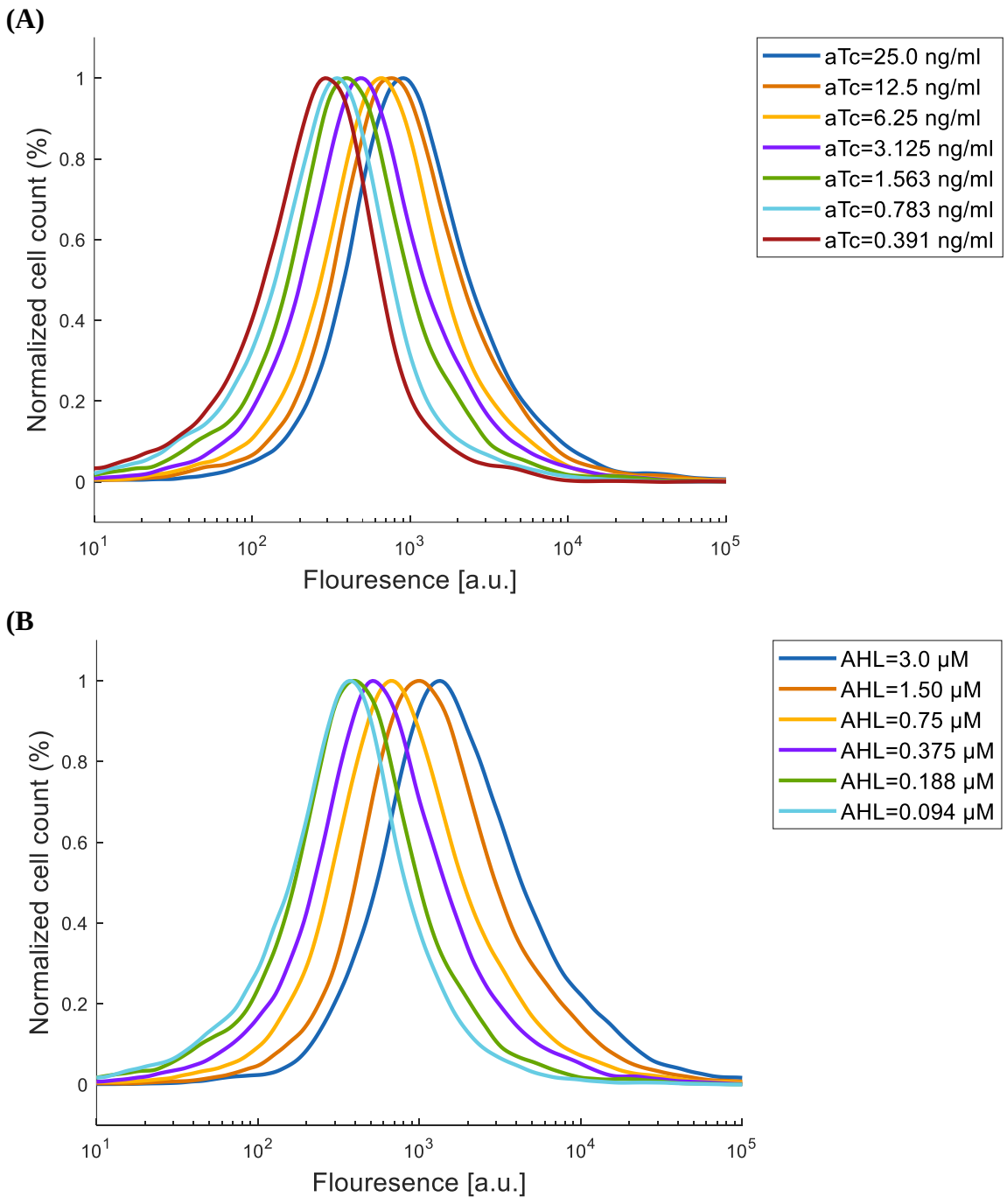


Fig. S14.29. mCherry flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop (Fig. 2B). (A) AHL was held constant at 0.188 μM and aTc was varied. (B) aTc was held constant at 1.563 ng/ml and AHL was varied.

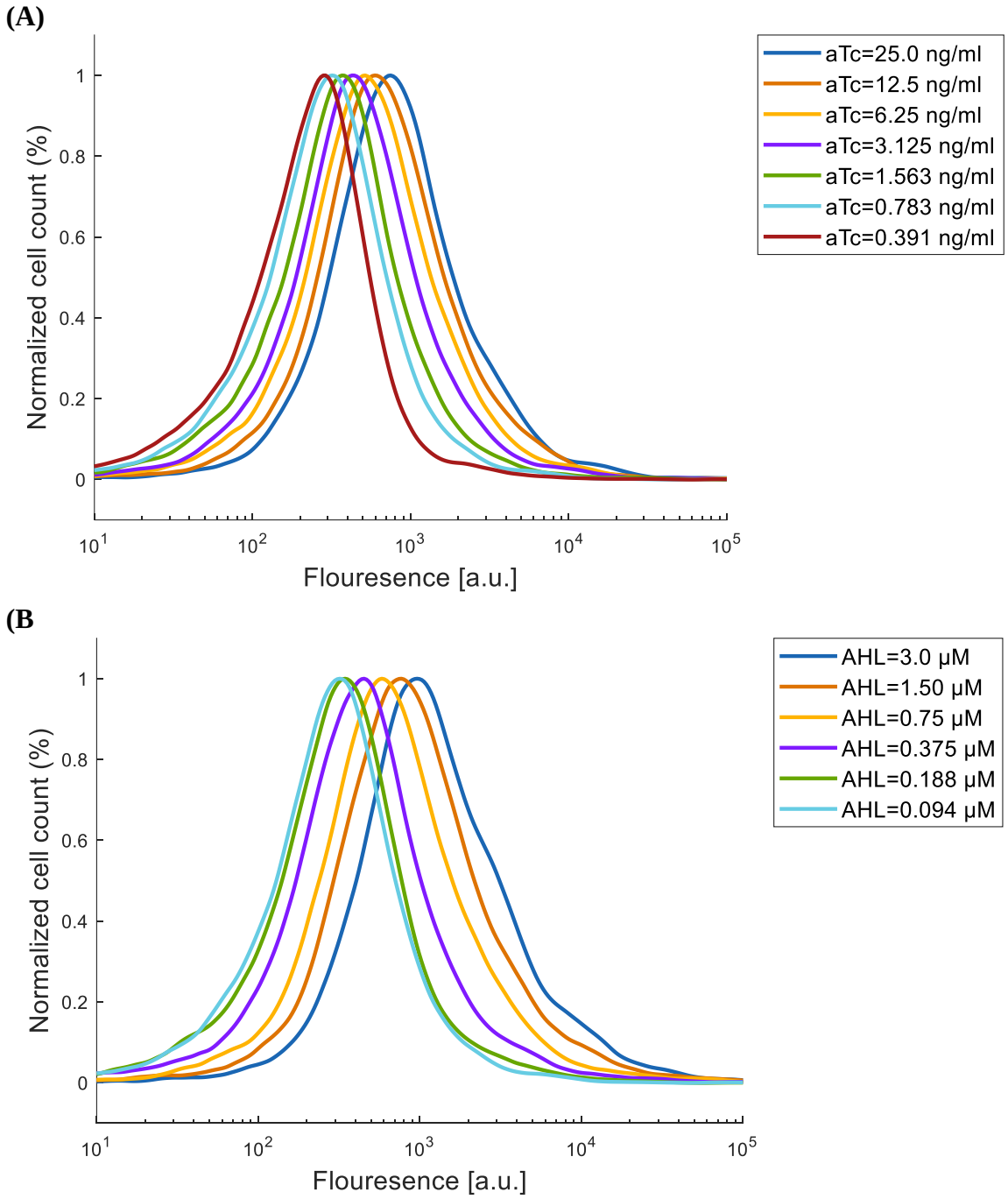


Fig. S14.30. mCherry flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop (Fig. 2B). (A) AHL was held constant at 0.094 μM and aTc was varied. (B) aTc was held constant at 0.783 ng/ml and AHL was varied.

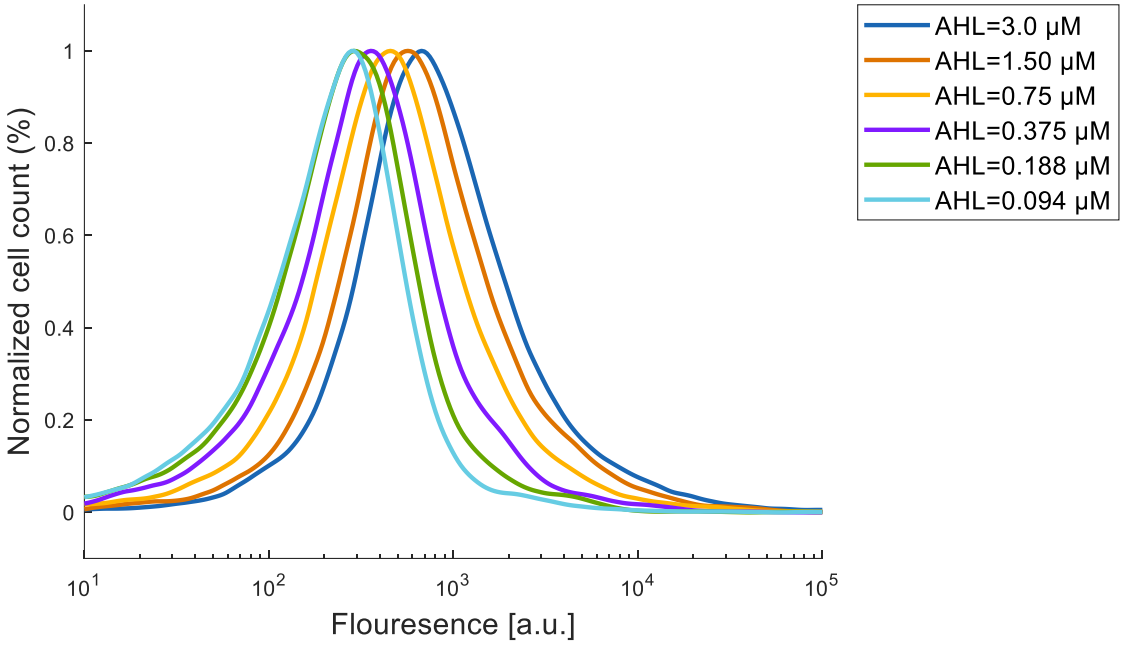


Fig. S14.31. mCherry flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop (Fig. 2B). aTc was held constant at 0.391 ng/ml and AHL was varied.

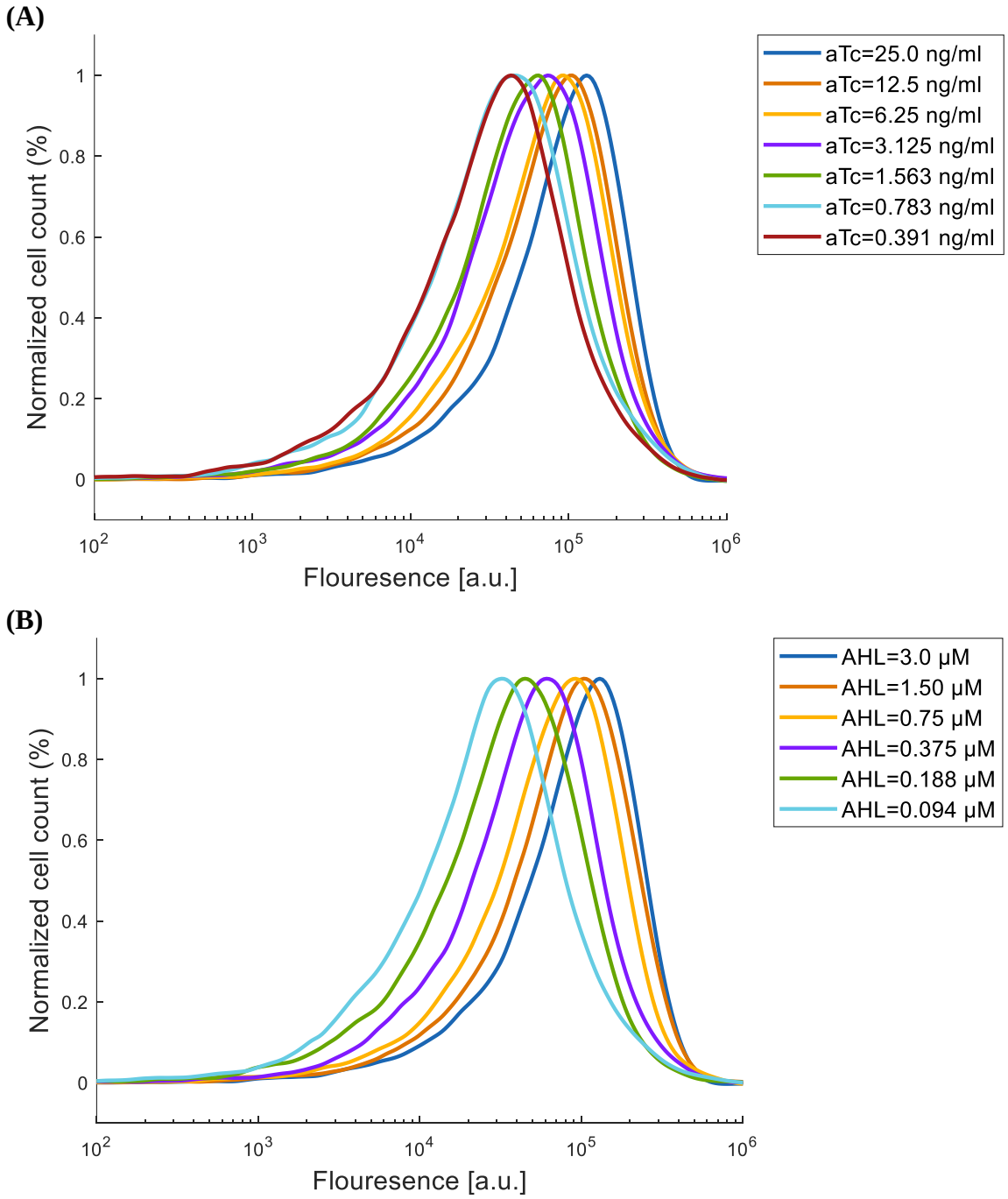


Fig. S14.32. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop. The output of the power-law and multiplication function was replaced by AraC activator, which regulate P_{BAD} promote (Fig. 2E). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 3.0 μM and aTc was varied. (B) Arabinose was held constant at 0.5 mM, aTc was held constant at 25 ng/ml and AHL was varied.

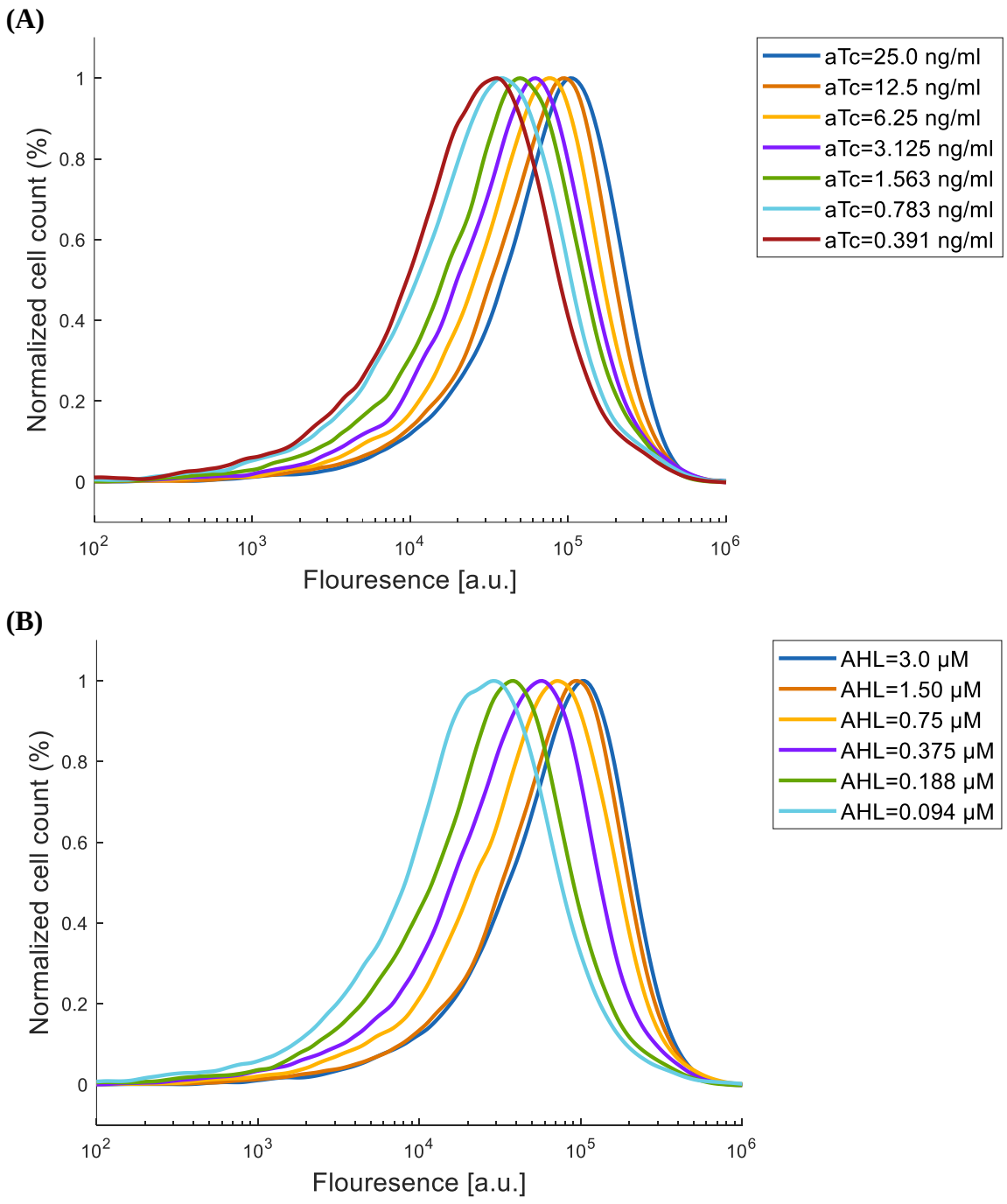


Fig. S14.33. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop. The output of the power-law and multiplication function was replaced by AraC activator, which regulate P_{BAD} promoter (Fig. 2E). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 1.5 μM and aTc was varied. (B) Arabinose was held constant at 0.5 mM, aTc was held constant at 12.5 ng/ml and AHL was varied.

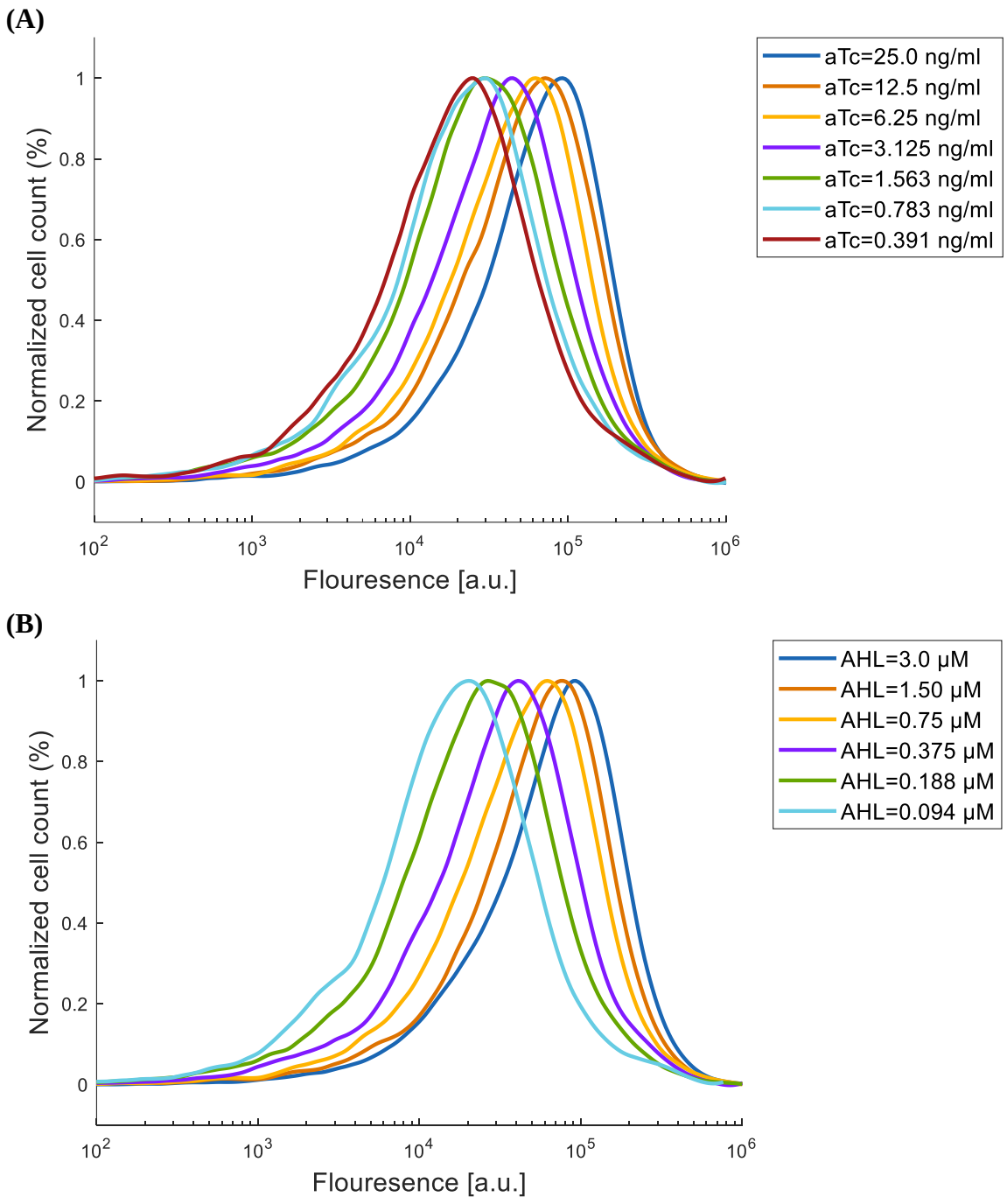


Fig. S14.34. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop. The output of the power-law and multiplication function was replaced by AraC activator, which regulate P_{BAD} promoter (Fig. 2E). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.75 μM and aTc was varied. (B) Arabinose was held constant at 0.5 mM, aTc was held constant at 6.25 ng/ml and AHL was varied.

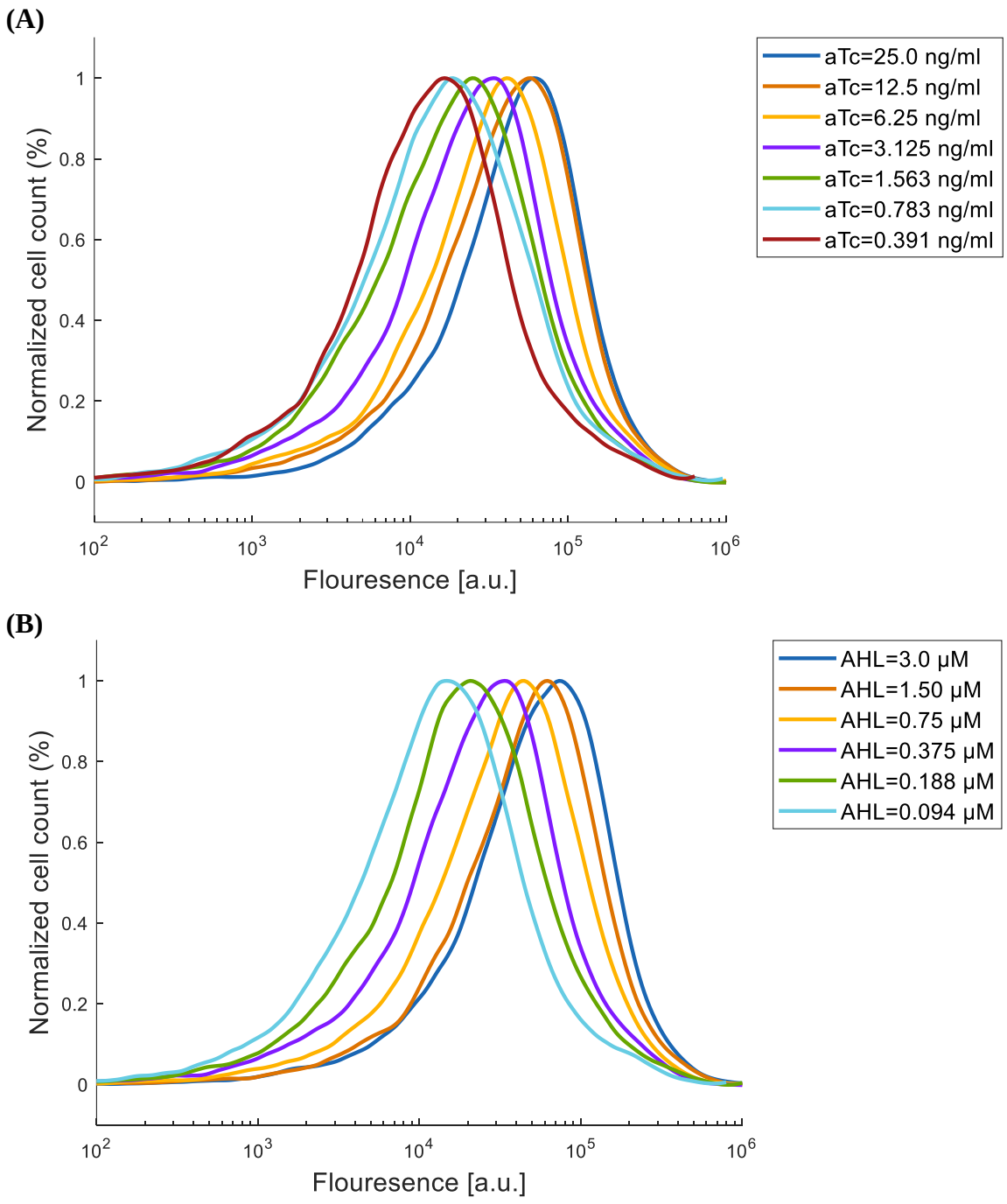


Fig. S14.35. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop. The output of the power-law and multiplication function was replaced by AraC activator, which regulate P_{BAD} promoter (Fig. 2E). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.375 μM and aTc was varied. (B) Arabinose was held constant at 0.5 mM, aTc was held constant at 3.125 ng/ml and AHL was varied.

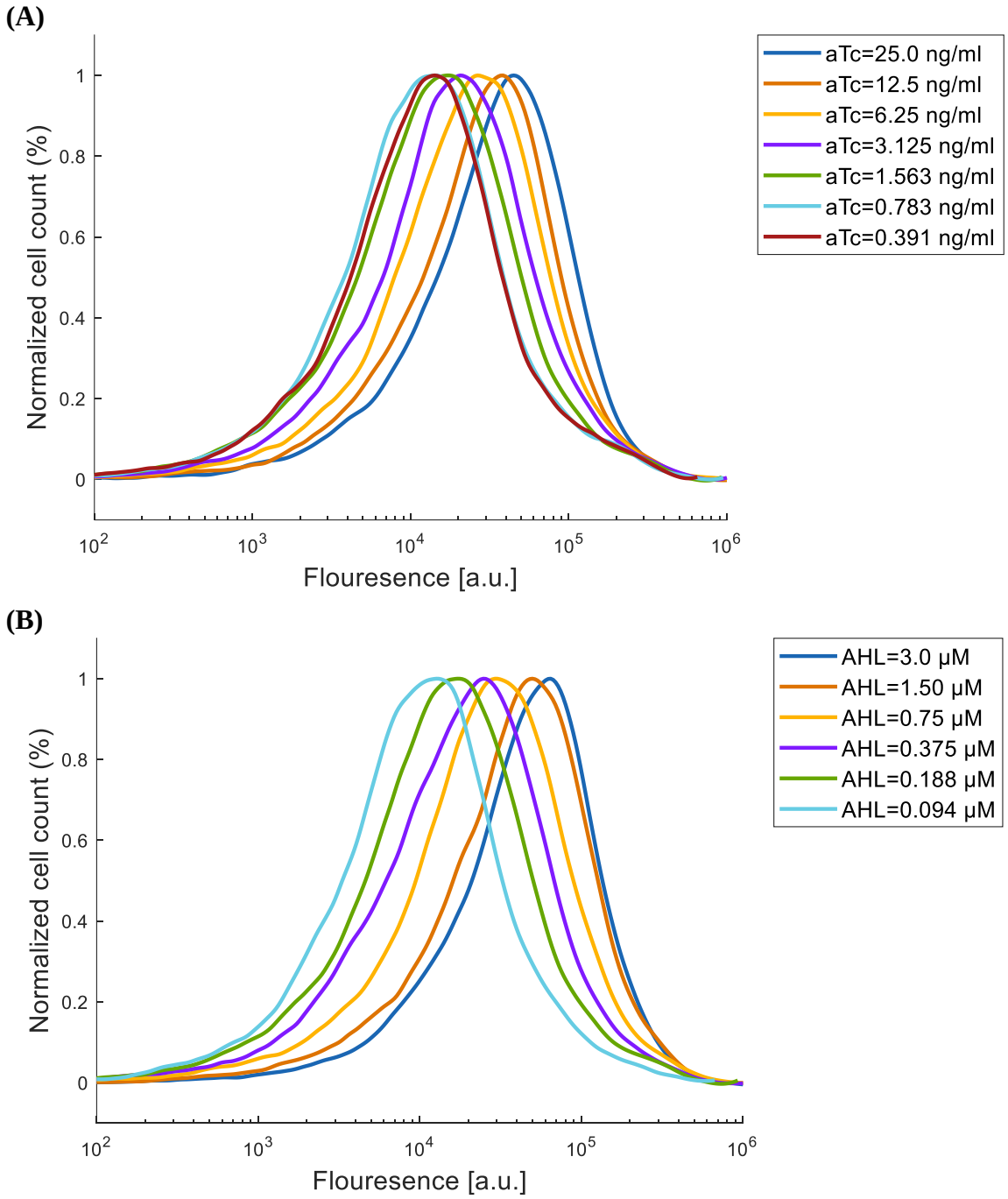


Fig. S14.36. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop. The output of the power-law and multiplication function was replaced by AraC activator, which regulate P_{BAD} promoter (Fig. 2E). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.188 μM and aTc was varied. (B) Arabinose was held constant at 0.5 mM, aTc was held constant at 1.563 ng/ml and AHL was varied.

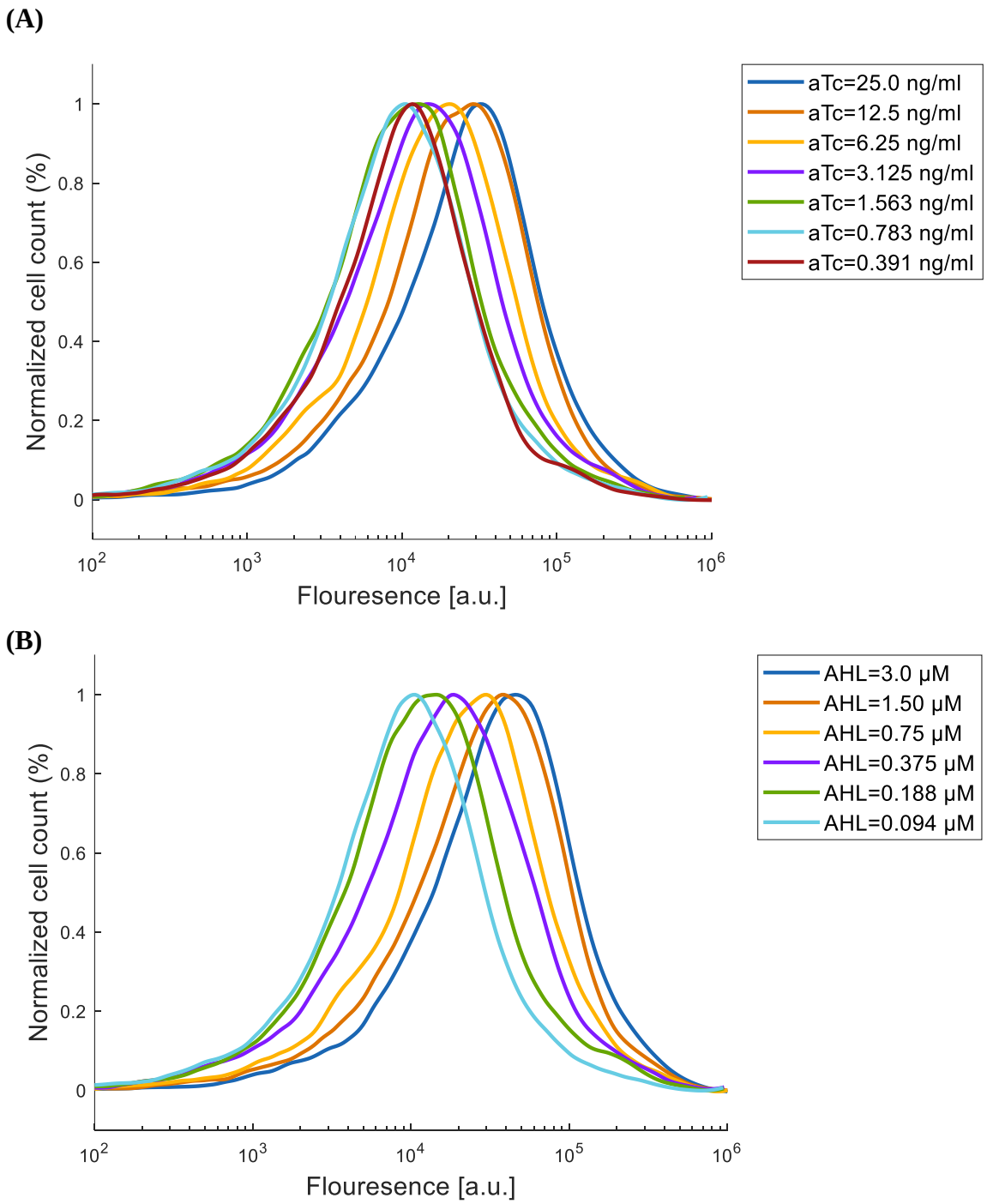


Fig. S14.37. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop. The output of the power-law and multiplication function was replaced by AraC activator, which regulate P_{BAD} promoter (Fig. 2E). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.094 μM and aTc was varied. (B) Arabinose was held constant at 0.5 mM, aTc was held constant at 0.783 ng/ml and AHL was varied

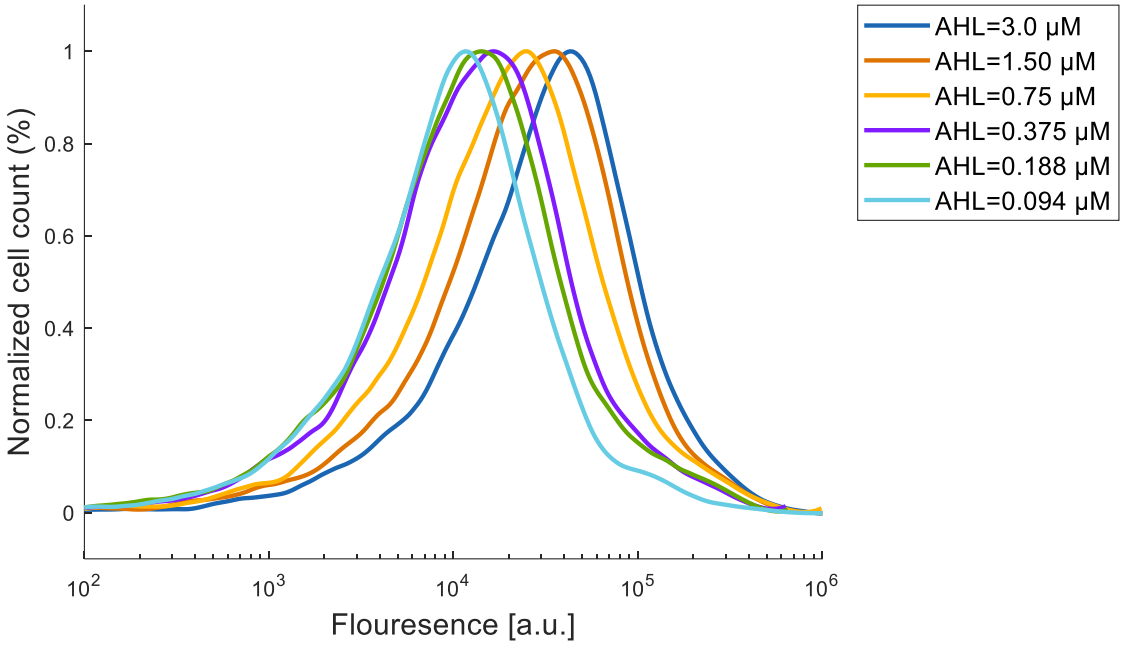


Fig. S14.38. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{tetO} promoter was regulated by TetR through ANF loop and mutated P_{luxTGT} promoter was regulated by LuxR through APF loop. The output of the power-law and multiplication function was replaced by AraC activator, which regulate P_{BAD}D promoter (Fig. 2E). Arabinose was held constant at 0.5 mM, aTc was held constant at 0.391 ng/ml and AHL was varied

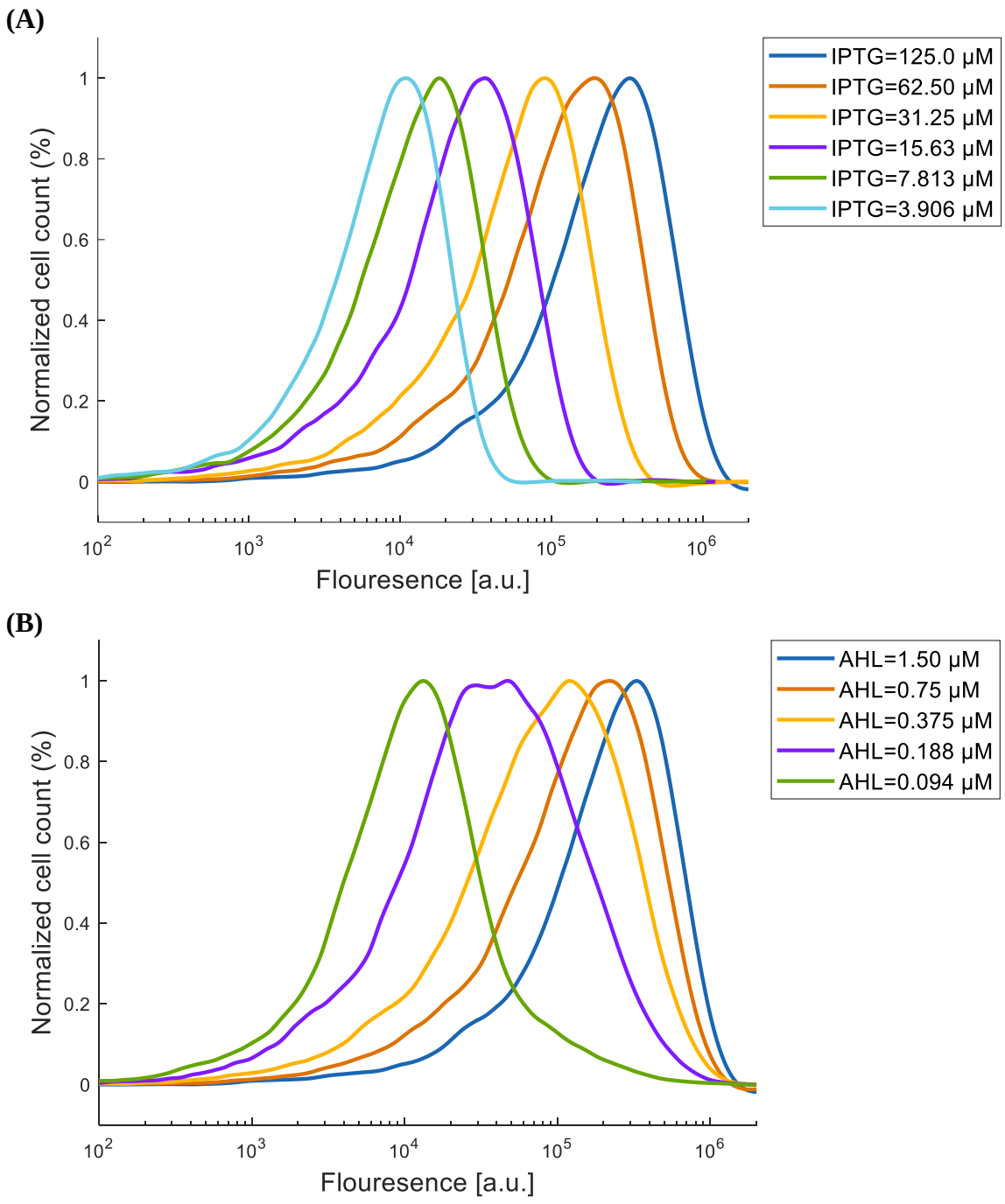


Fig. S14.39. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{lacO1} promoter was regulated by LacI through ANF loop and mutated P_{luxAAT} promoter was regulated by LuxR through APF loop (Fig. S3.4, Fig. S3.6). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 1.5 μM and IPTG was varied. (B) Arabinose was held constant at 0.5 mM, IPTG was held constant at 125 μM and AHL was varied.

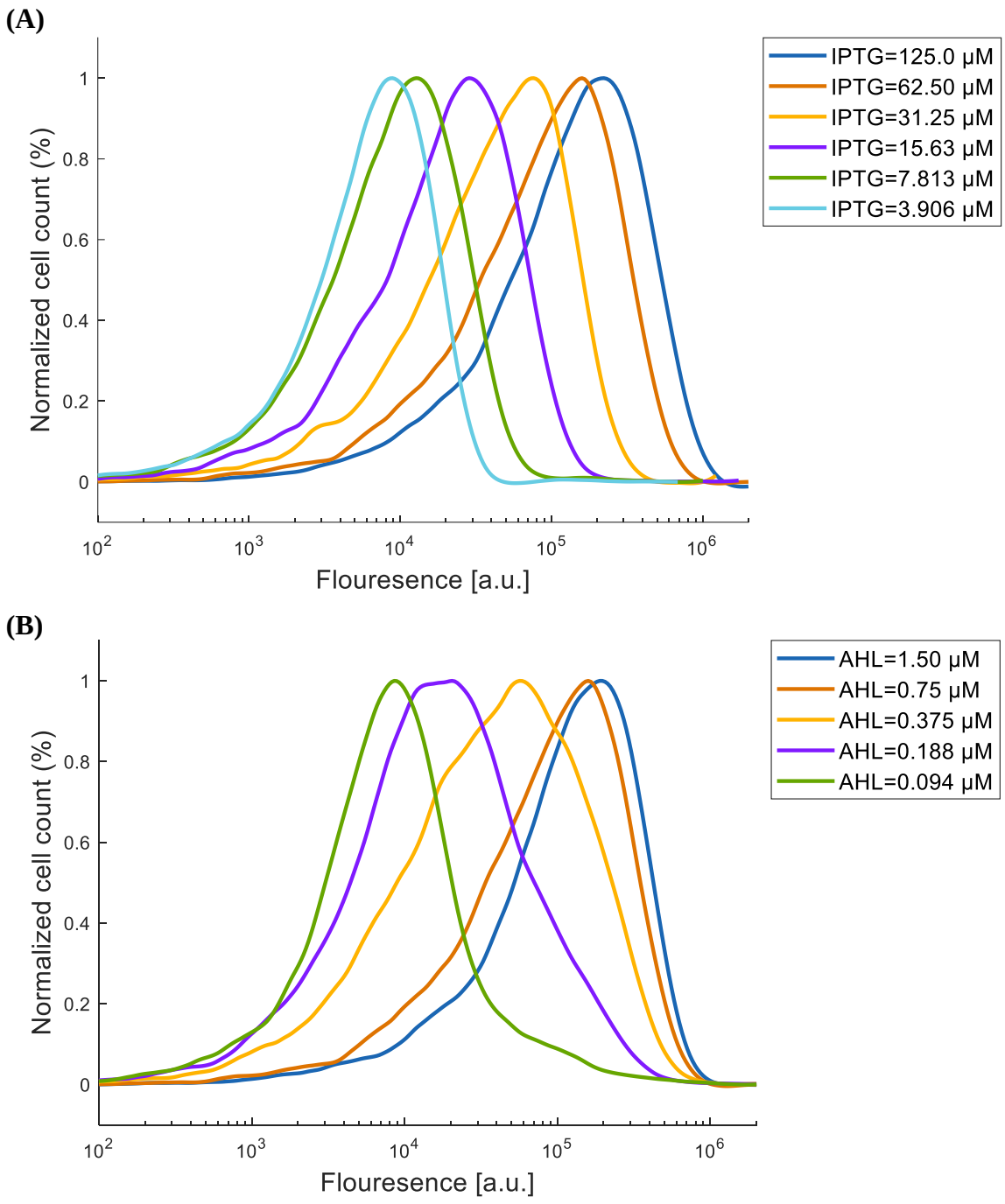


Fig. S14.40. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{lacO1} promoter was regulated by LacI through ANF loop and mutated P_{luxAAT} promoter was regulated by LuxR through APF loop (Fig. S3.4, Fig. S3.6). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.75 μM and IPTG was varied. (B) Arabinose was held constant at 0.5 mM, IPTG was held constant at 62.5 μM and AHL was varied.

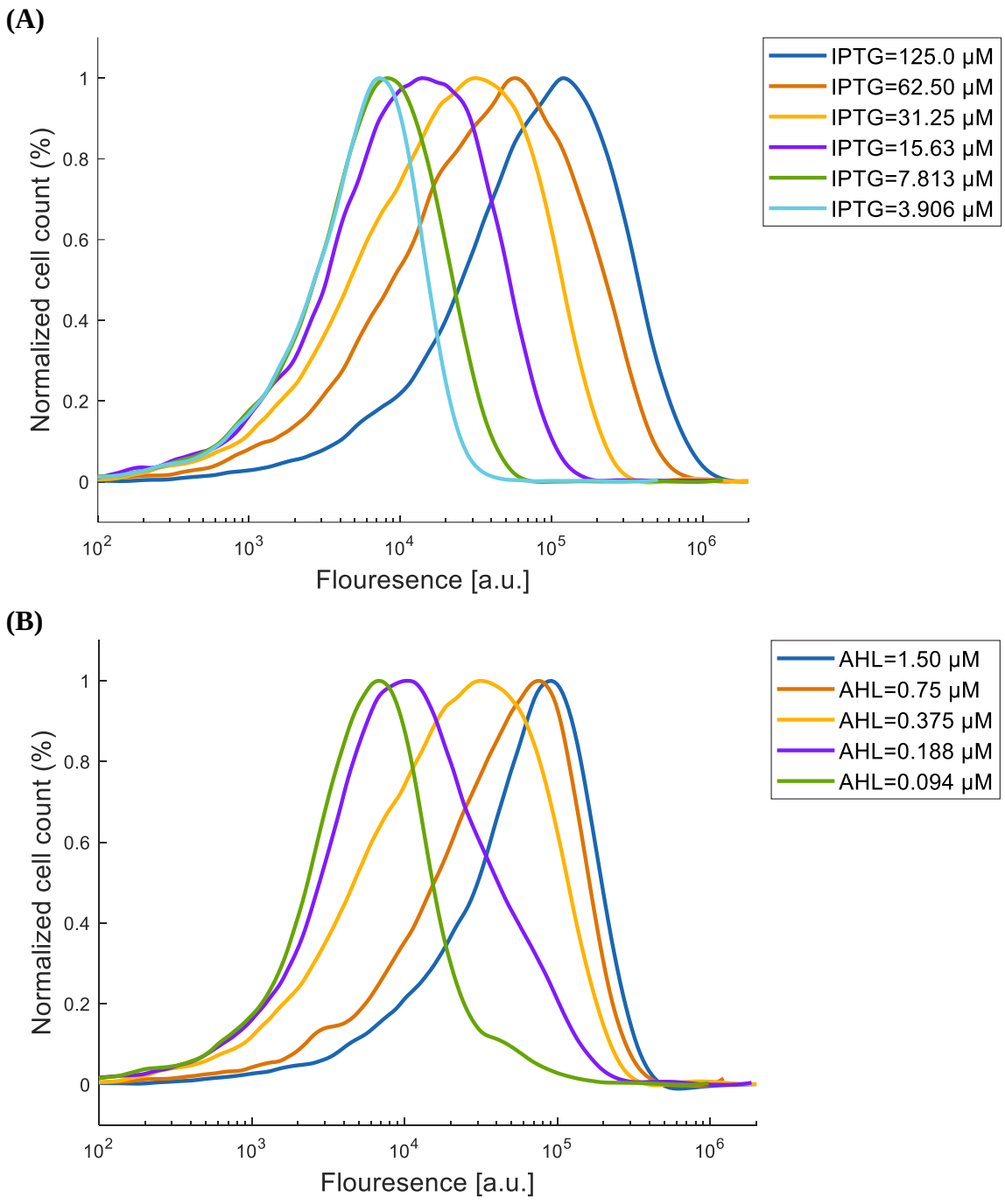


Fig. S14.41. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{lacO1} promoter was regulated by LacI through ANF loop and mutated P_{luxAAT} promoter was regulated by LuxR through APF loop (Fig. S3.4, Fig. S3.6). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.375 μM and IPTG was varied. (B) Arabinose was held constant at 0.5 mM, IPTG was held constant at 31.25 μM and AHL was varied.

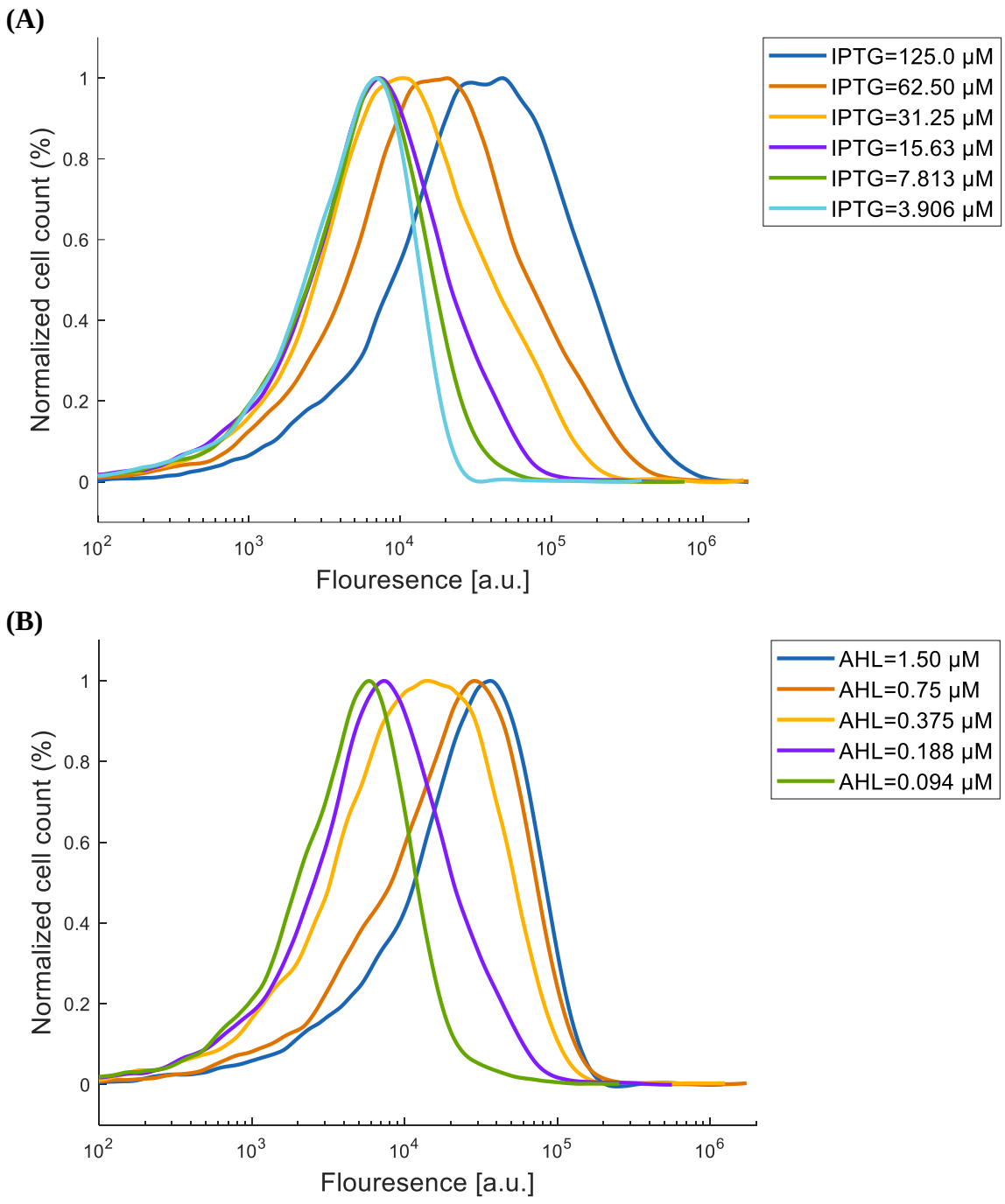


Fig. S14.42. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{lacO1} promoter was regulated by LacI through ANF loop and mutated P_{luxAAT} promoter was regulated by LuxR through APF loop (Fig. S3.4, Fig. S3.6). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.188 μM and IPTG was varied. (B) Arabinose was held constant at 0.5 mM, IPTG was held constant at 15.63 μM and AHL was varied.

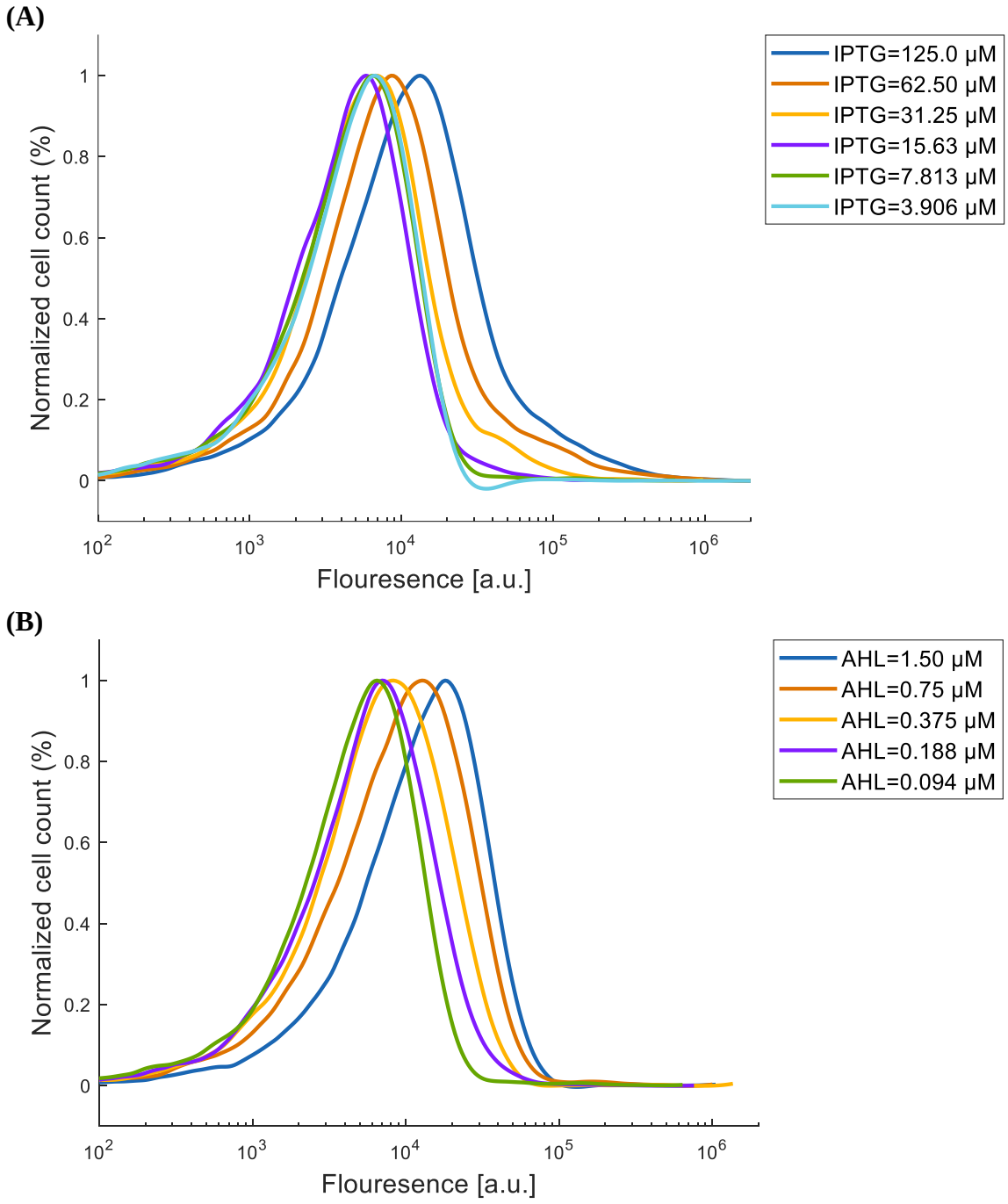


Fig. S14.43. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{lacO1} promoter was regulated by LacI through ANF loop and mutated P_{luxAAT} promoter was regulated by LuxR through APF loop (Fig. S3.4, Fig. S3.6). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.094 μM and IPTG was varied. (B) Arabinose was held constant at 0.5 mM, IPTG was held constant at 7.813 μM and AHL was varied.

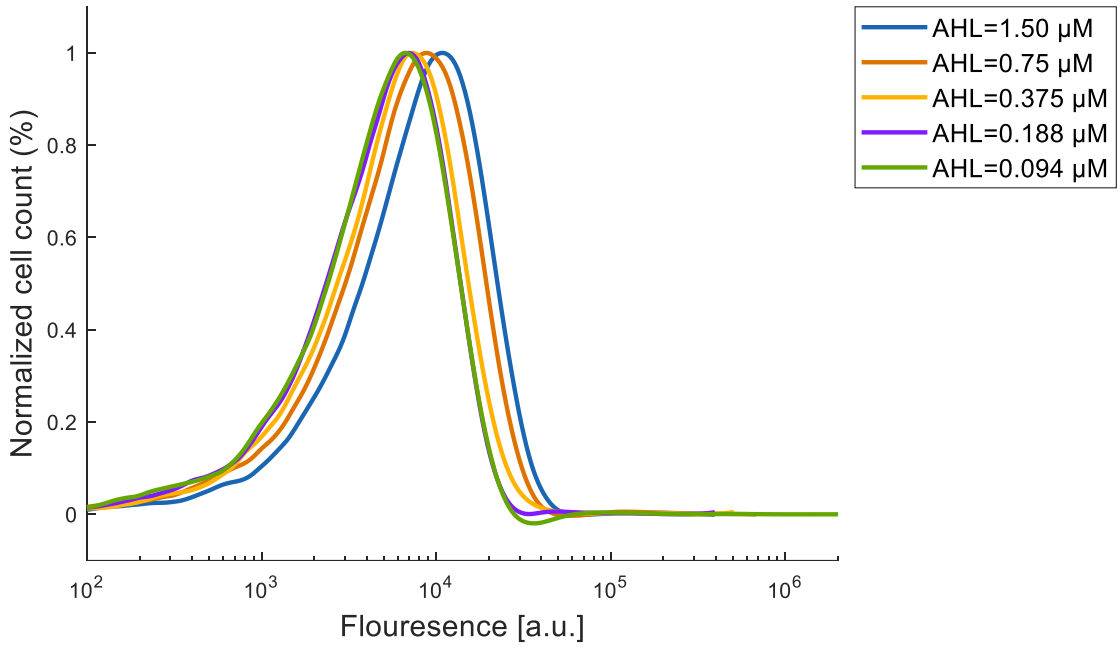


Fig. S14.44. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, P_{lacO1} promoter was regulated by LacI through ANF loop and mutated P_{luxAAT} promoter was regulated by LuxR through APF loop (Fig. S3.4, Fig. S3.6). Arabinose was held constant at 0.5 mM, IPTG was held constant at 3.906 μM and AHL was varied.

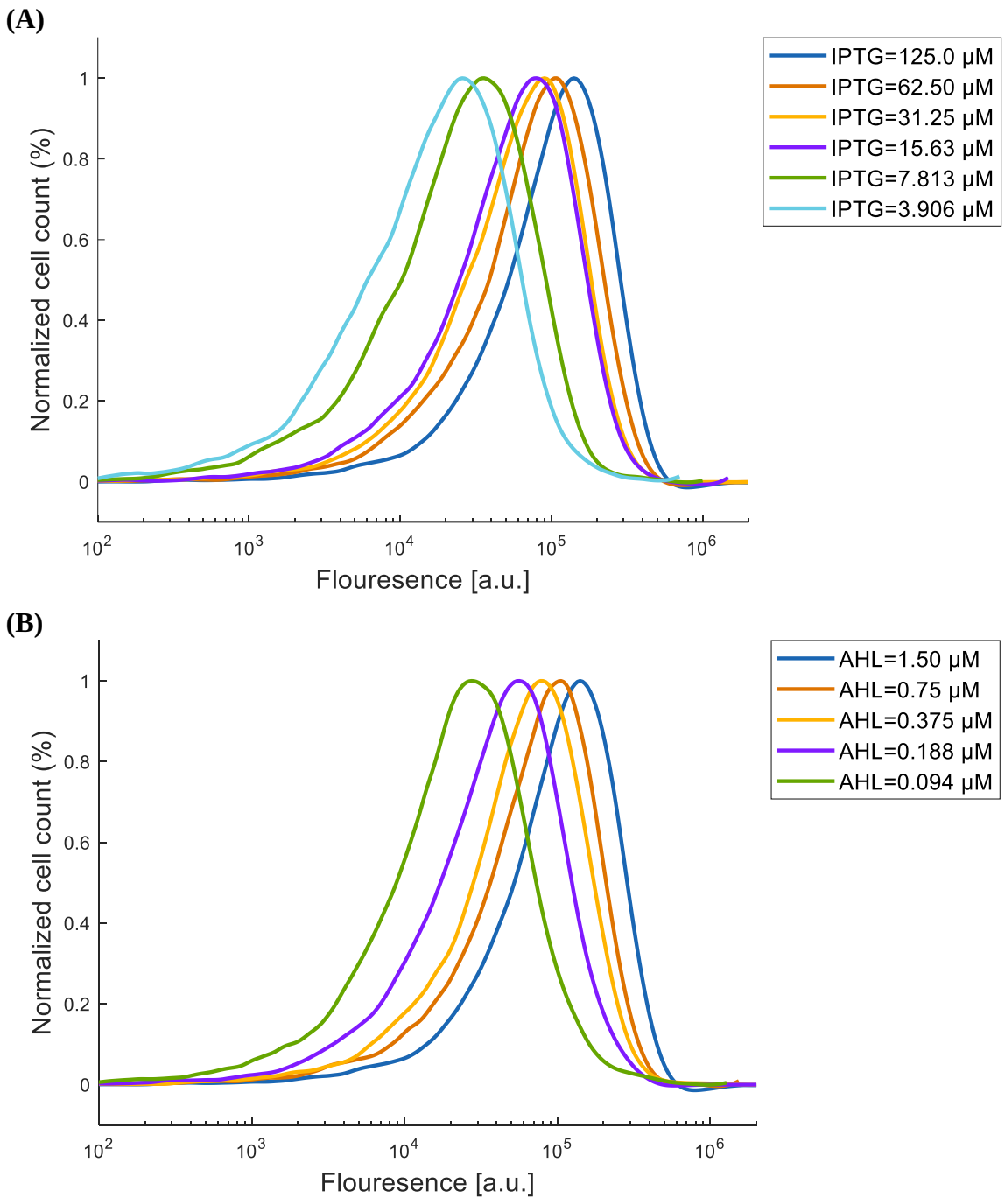


Fig. S14.45. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, synthetic average-meter based on perceptgene model was calculated using P_{lacO1} promoter, mutated P_{luxAAT} promoter and a combinatorial promoter ($P_{lux/tetO}$). P_{BAD} was used to set the logistic curve of the analog inputs (Fig. 2H). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 1.5 μM and IPTG was varied. (B) Arabinose was held constant at 0.5 mM, IPTG was held constant at 125 μM and AHL was varied.

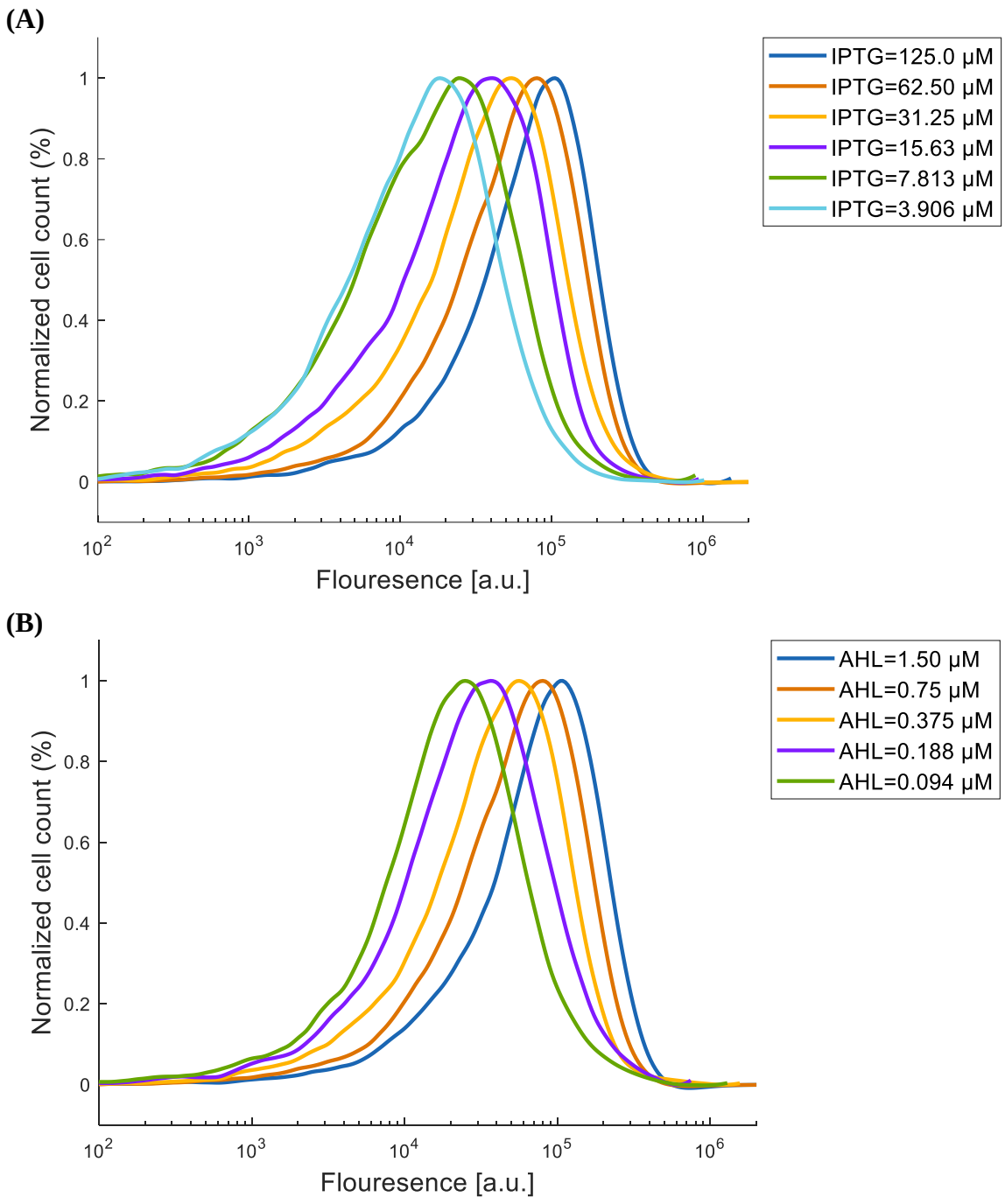


Fig. S14.46. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, synthetic average-meter based on perceptgene model was calculated using P_{lacO1} promoter, mutated P_{luxAAT} promoter and a combinatorial promoter ($P_{lux/tetO}$). P_{BAD} was used to set the logistic curve of the analog inputs (Fig. 2H). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.75 μM and IPTG was varied. (B) Arabinose was held constant at 0.5 mM, IPTG was held constant at 62.5 μM and AHL was varied.

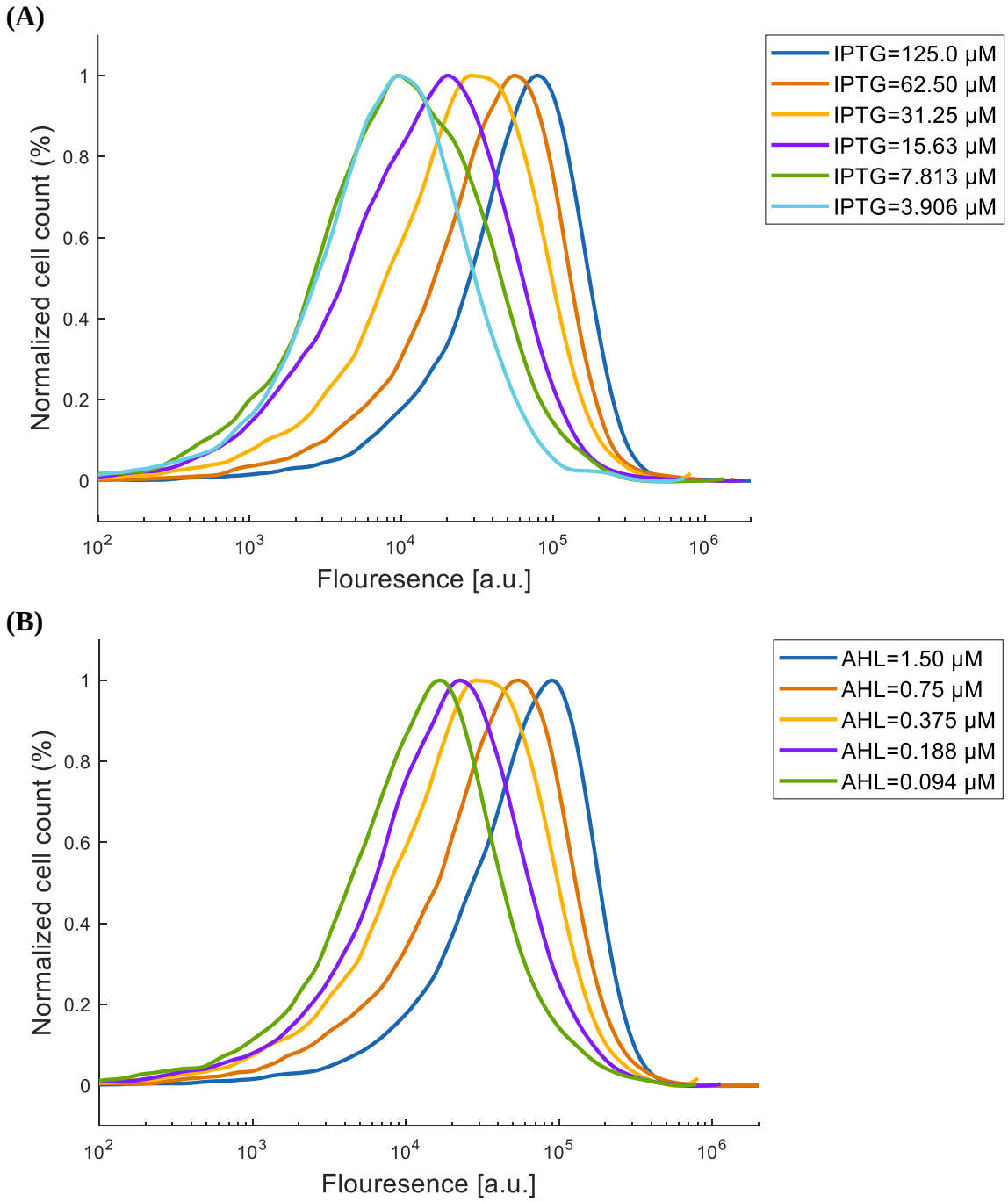


Fig. S14.47. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, synthetic average-meter based on perceptgene model was calculated using P_{lacO1} promoter, mutated P_{luxAAT} promoter and a combinatorial promoter ($P_{lux/tetO}$). P_{BAD} was used to set the logistic curve of the analog inputs (Fig. 2H). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.375 μM and IPTG was varied. (b) Arabinose was held constant at 0.5 mM, IPTG was held constant at 31.25 μM and AHL was varied.

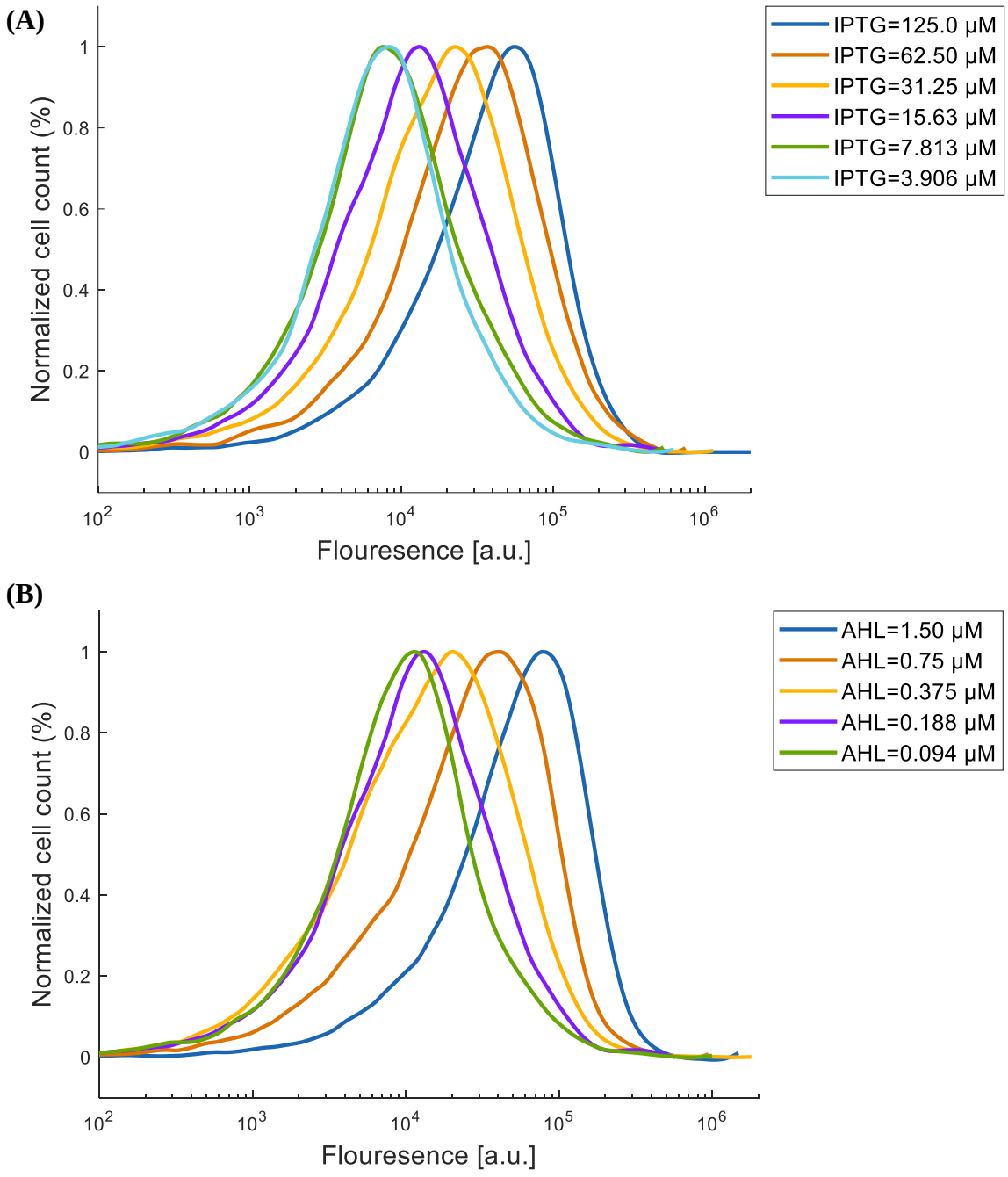


Fig. S14.48. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, synthetic average-meter based on perceptgene model was calculated using P_{lacO1} promoter, mutated P_{luxAAT} promoter and a combinatorial promoter ($P_{lux/tetO}$). P_{BAD} was used to set the logistic curve of the analog inputs (Fig. 2H). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.188 μM and IPTG was varied. (b) Arabinose was held constant at 0.5 mM, IPTG was held constant at 15.63 μM and AHL was varied.

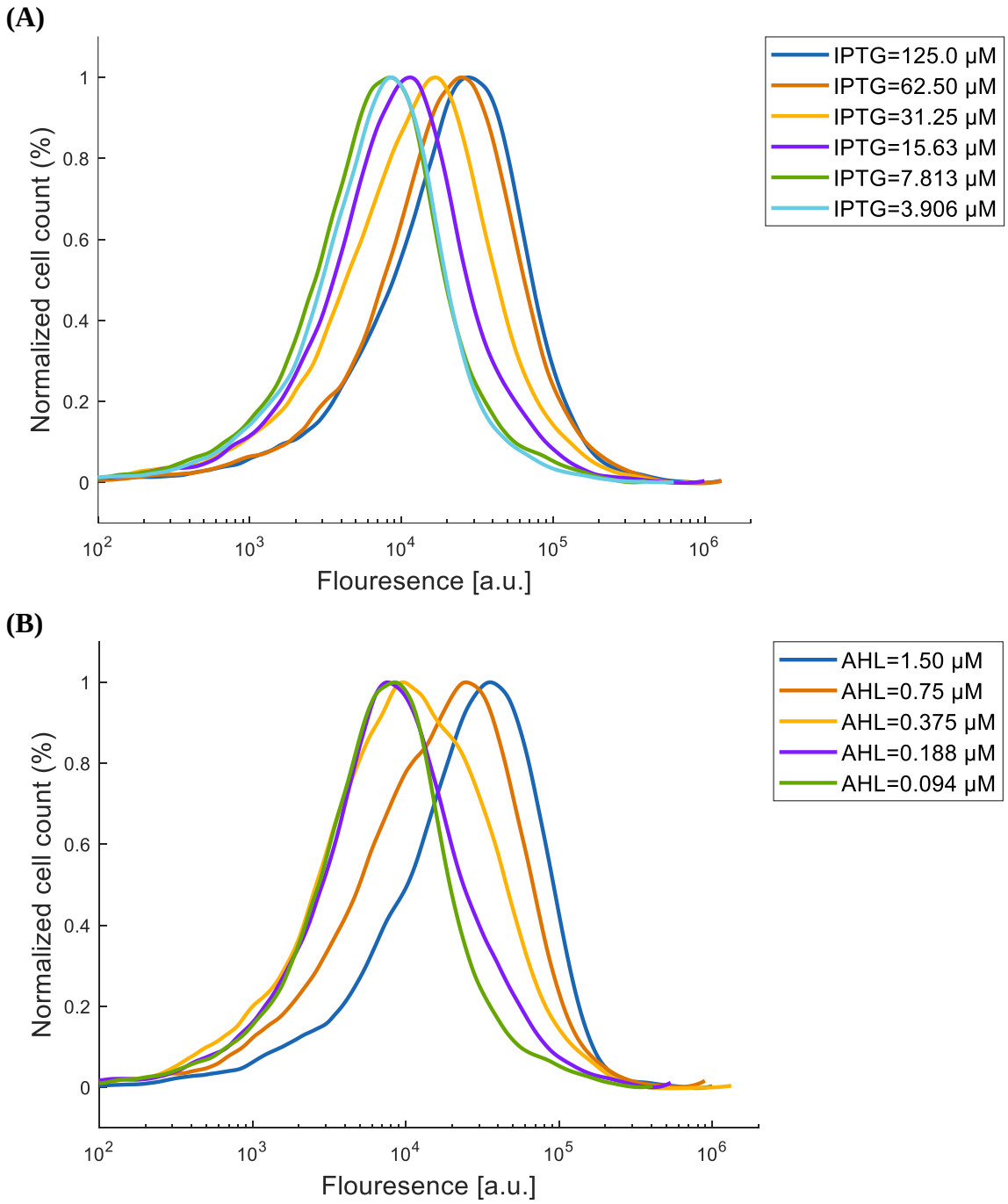


Fig. S14.49. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, synthetic average-meter based on perceptgene model was calculated using P_{lacO1} promoter, mutated P_{luxAAT} promoter and a combinatorial promoter ($P_{lux/tetO}$). P_{BAD} was used to set the logistic curve of the analog inputs (Fig. 2H). (A) Arabinose was held constant at 0.5 mM, AHL was held constant at 0.094 μM and IPTG was varied. (B) Arabinose was held constant at 0.5 mM, IPTG was held constant at 7.813 μM and AHL was varied.

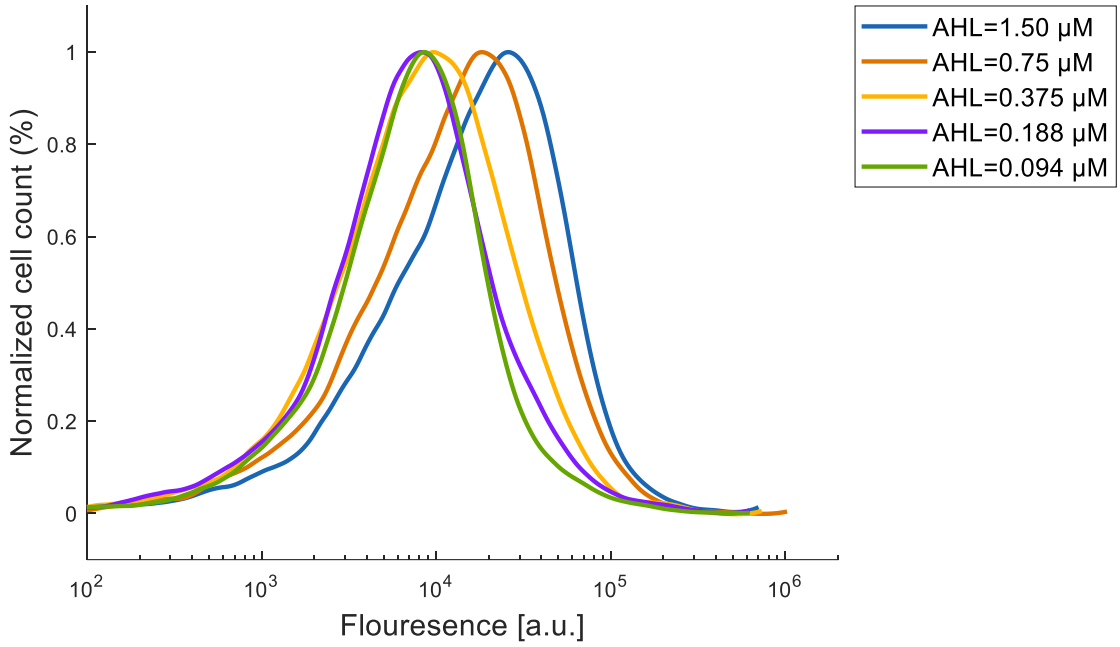


Fig. S14.50. GFP flow cytometry data for a population of cells containing the synthetic perceptgene based on ANF and APF loops. In this circuit, synthetic average-meter based on perceptgene model was calculated using P_{lacO1} promoter, mutated P_{luxAAT} promoter and a combinatorial promoter ($P_{lux/tetO}$). P_{BAD} was used to set the logistic curve of the analog inputs (Fig. 2H). Arabinose was held constant at 0.5 mM, IPTG was held constant at 3.906 μM and AHL was varied.

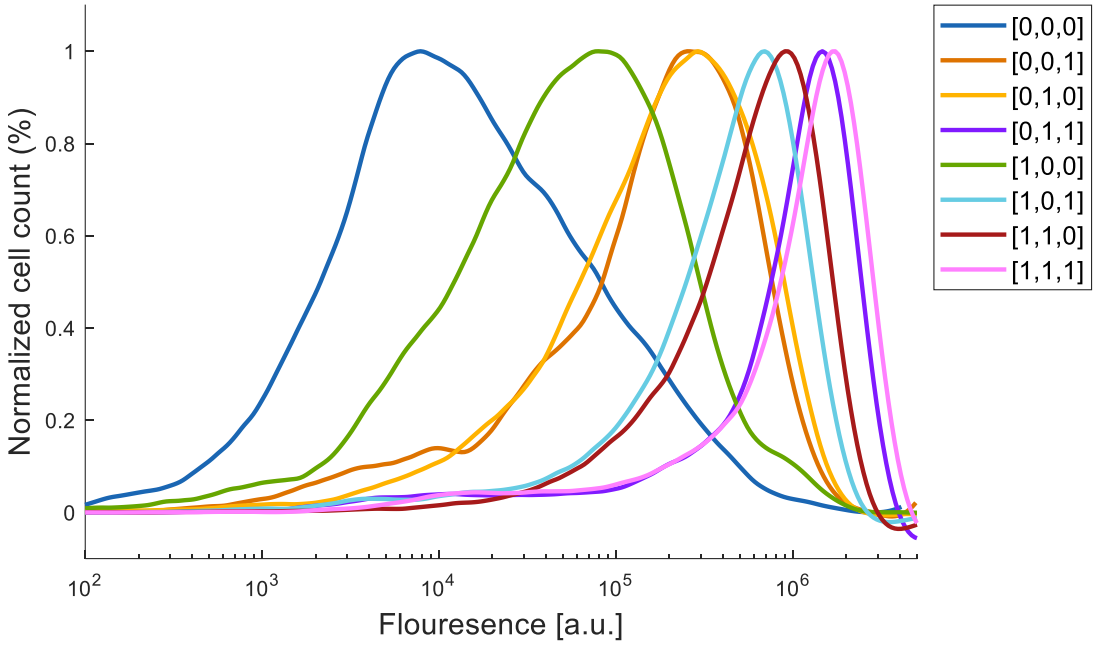


Fig. S14.51. GFP flow cytometry data for a population of cells containing the synthetic multilayer perceptgene network (Fig. 3C). Measured response of majority circuit. AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.25mM].

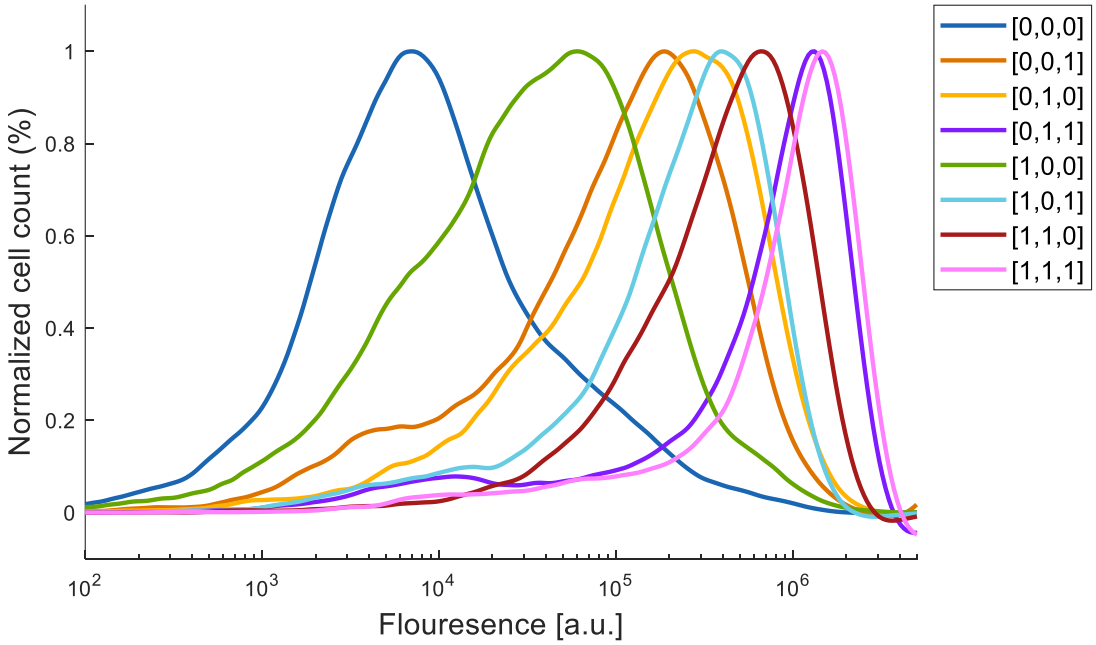


Fig. S14.52. GFP flow cytometry data for a population of cells containing the synthetic multilayer perceptgene network (Fig. 3C). Measured response of majority circuit. AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.125 mM].

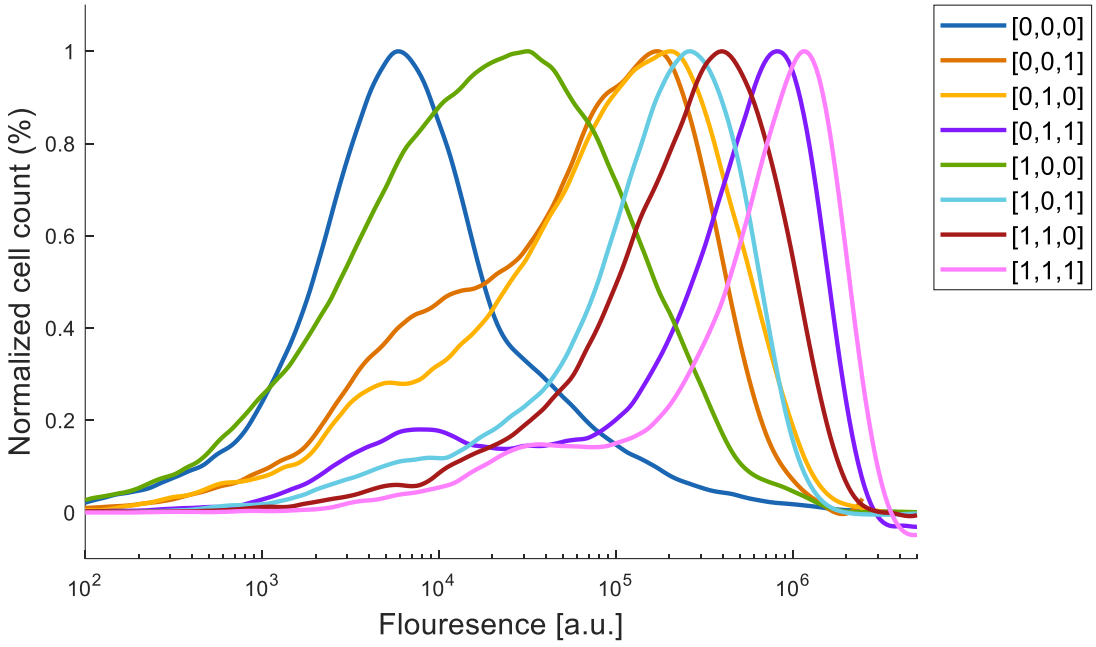


Fig. S14.53. GFP flow cytometry data for a population of cells containing the synthetic multilayer perceptgene network (Fig. 3C). Measured response of majority circuit. AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0625 mM].

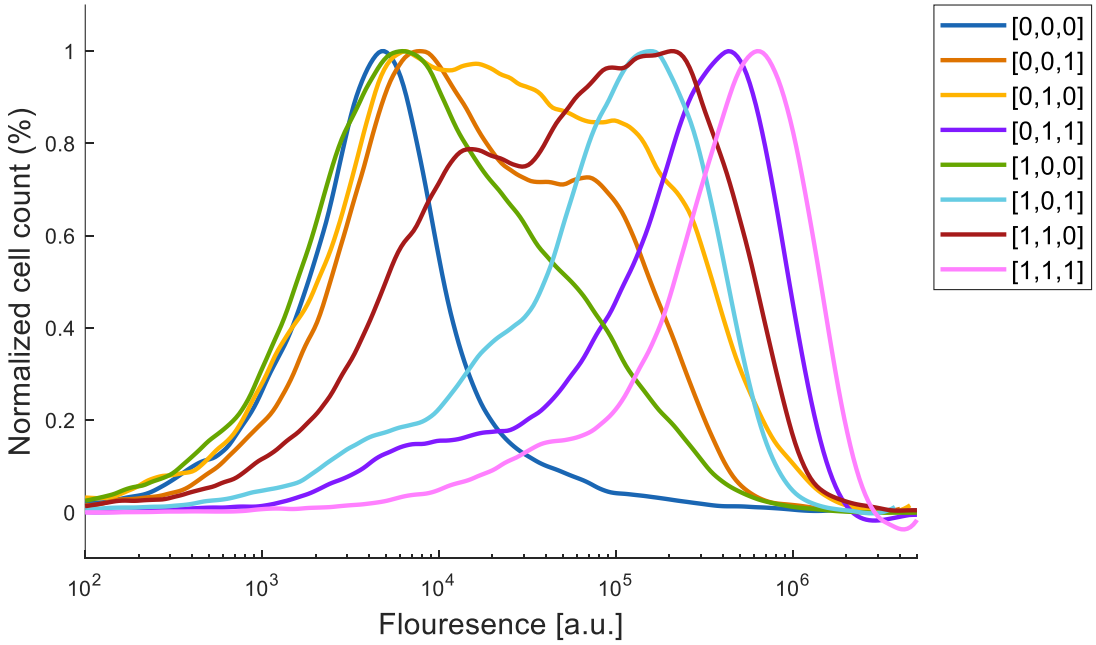


Fig. S14.54. GFP flow cytometry data for a population of cells containing the synthetic multilayer perceptgene network (Fig. 3C). Measured response of majority circuit. AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125 mM].

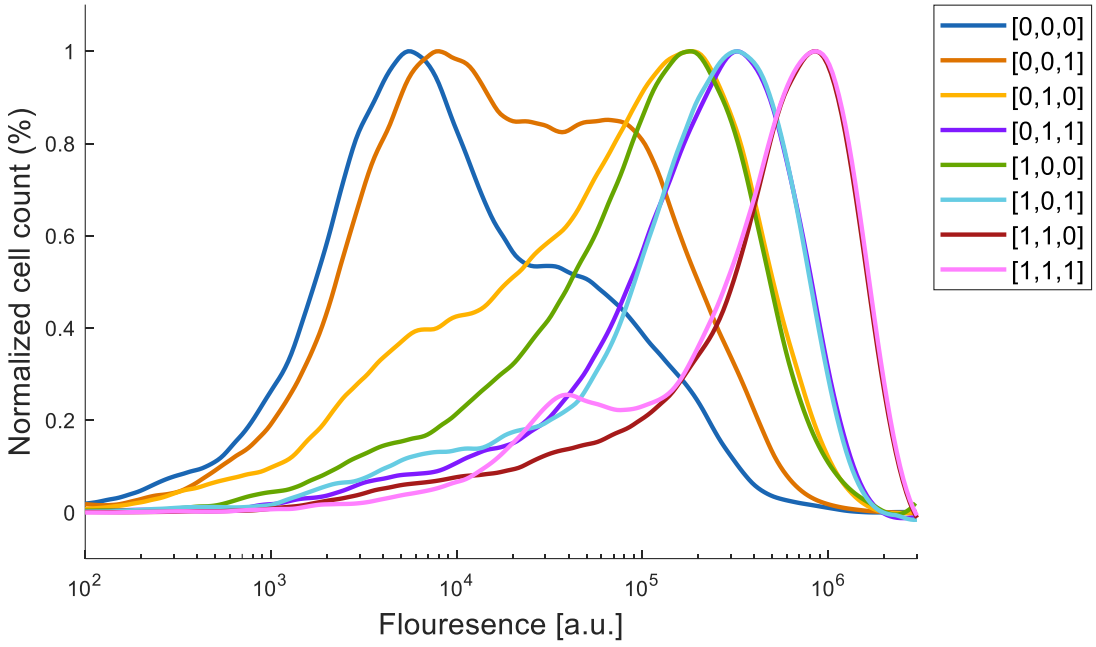


Fig. S14.55. GFP flow cytometry data for a population of cells containing the synthetic multilayer perceptgene displays a new logic function for three input (AHL, IPTG and aTc) (Fig. 7.18). Measured response of majority circuit. AHL [0.1875, 0.3µM], IPTG [7.8125, 125µM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125 mM].

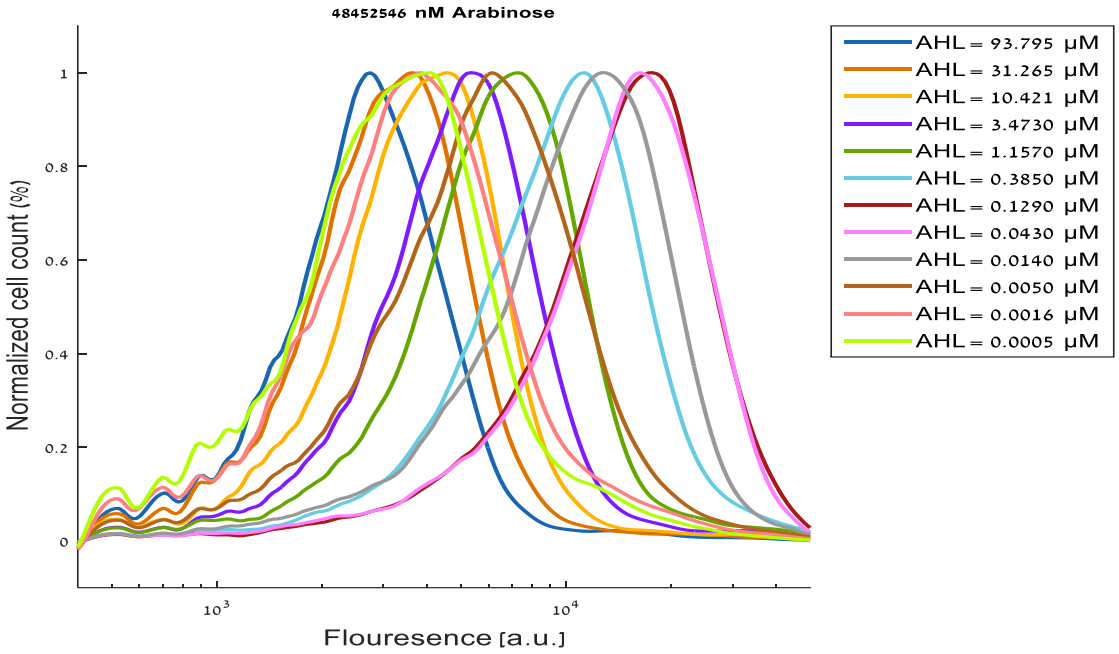


Fig. S14.56. GFP flow cytometry data for a population of cells containing the genetic circuit to implement LSB using a forward P_{BAD} promoter and an antisense P_{ux} promoter (Fig. S7.7A, Fig. S7.8).

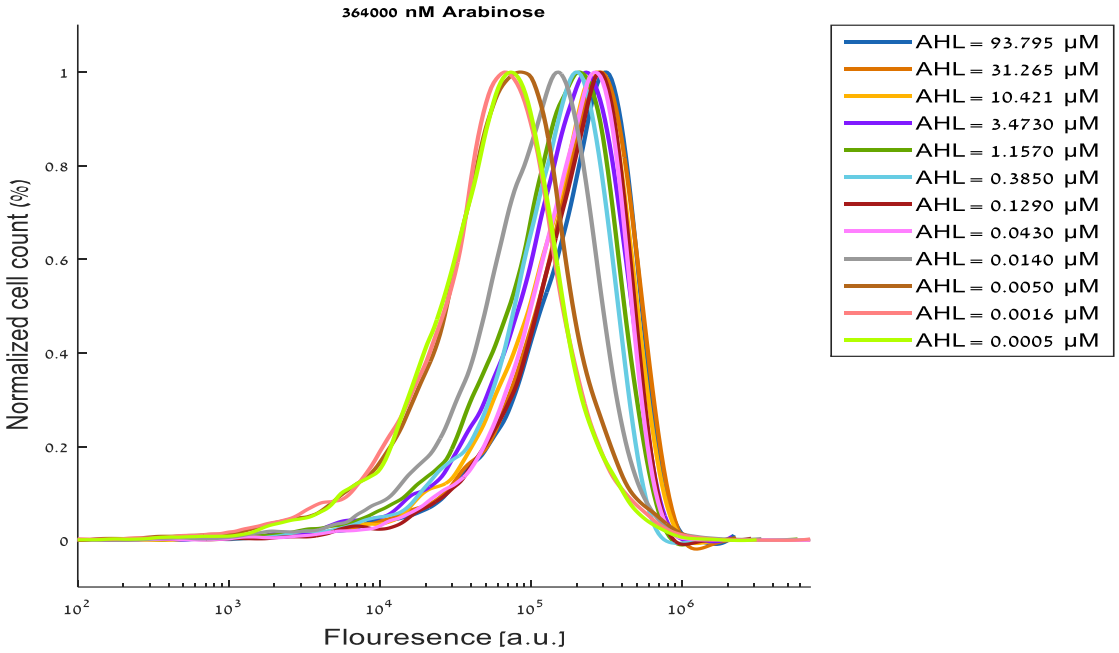


Fig. S14.57. GFP flow cytometry data for a population of cells containing the genetic circuit to implement 2-bit ADC, using a graded PF that regulates P_{BAD} promoter and a combinatorial antisense $P_{lux/tet0}$, while $TetR$ repressor is regulated by MSB Circuit (Fig. 4C blue plot).

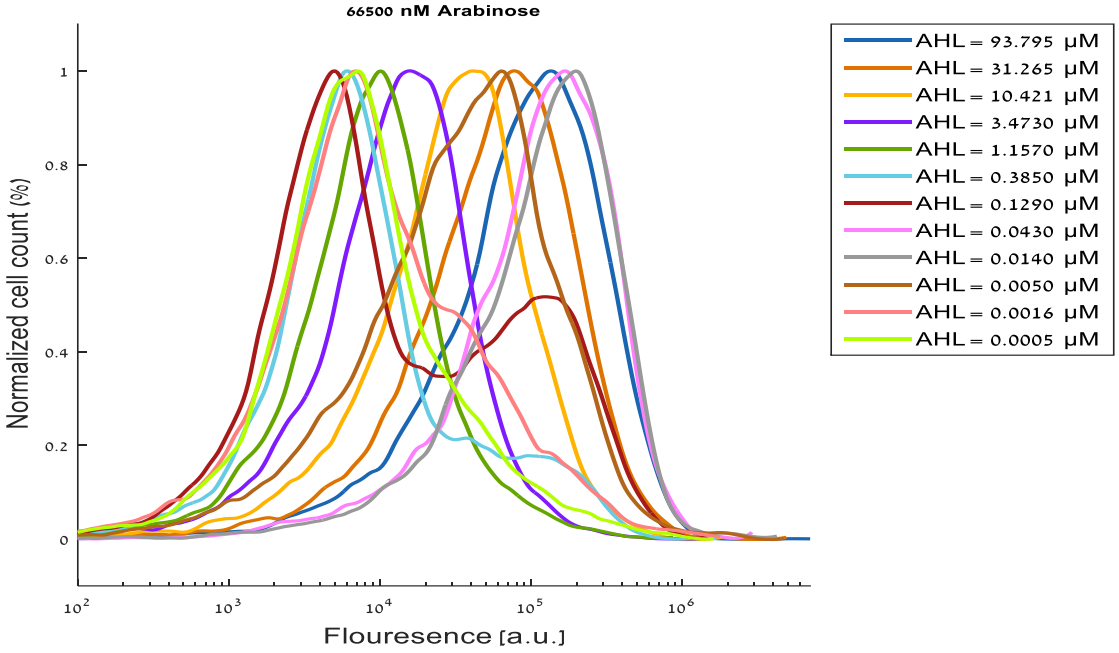


Fig. S14.58. GFP flow cytometry data for a population of cells containing the genetic circuit to implement 2-bit hybrid ADC, where LSB circuit is built from two GFP signals: (1) Forward P_{BAD} promoter with antisense P_{lux} promoter and (2) the P_{rhlR} promoter (Fig. 4F).

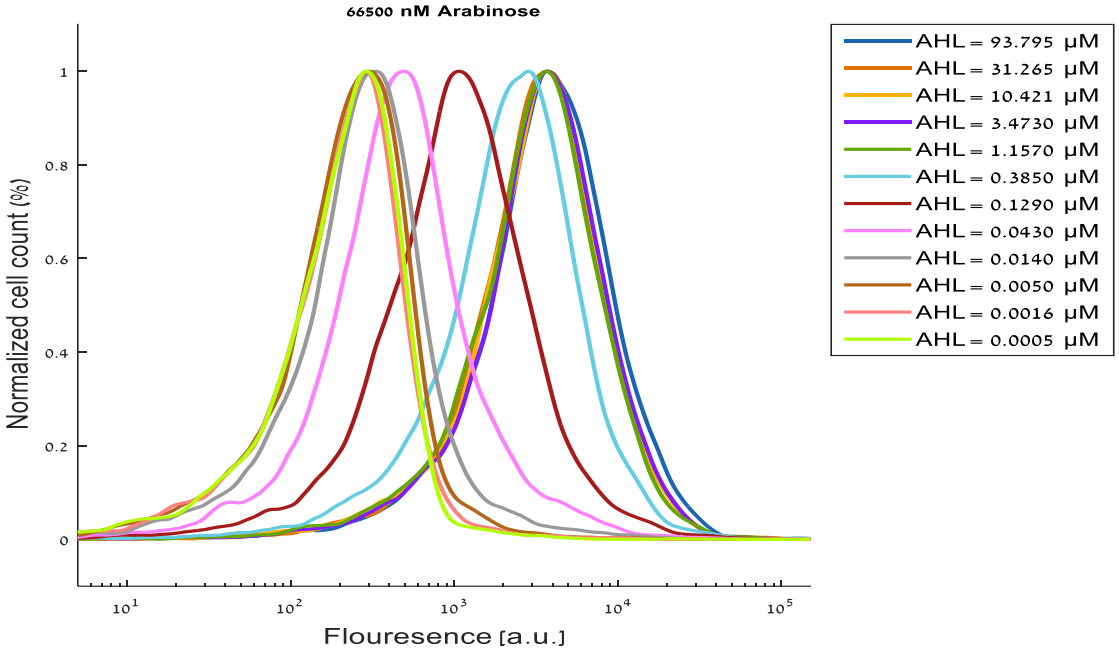


Fig. S14.59. mCherry flow cytometry data for a population of cells containing the genetic circuit to implement 2-bit hybrid ADC. Where the P_{lux} of MSB circuit which is located on MCP, regulates the output mCherry signal (Fig. 4F).

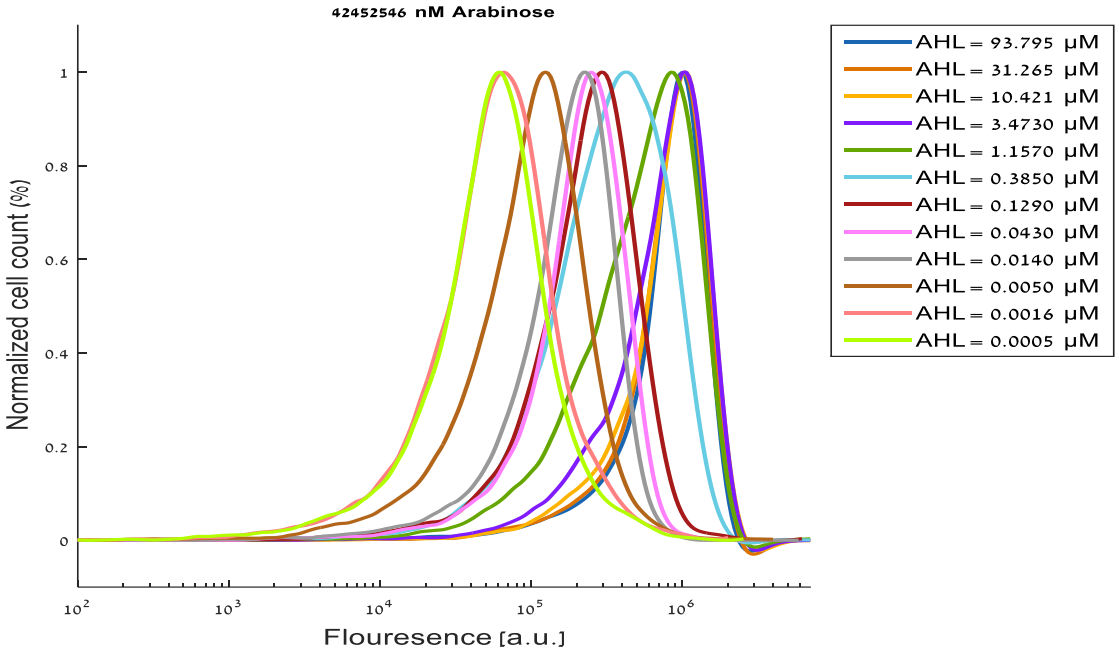


Fig. S14.60. GFP flow cytometry data for a population of cells containing the genetic circuit to implement ternary data converter, based on the regulation of $TetR$ by MSB (Fig. 4G).

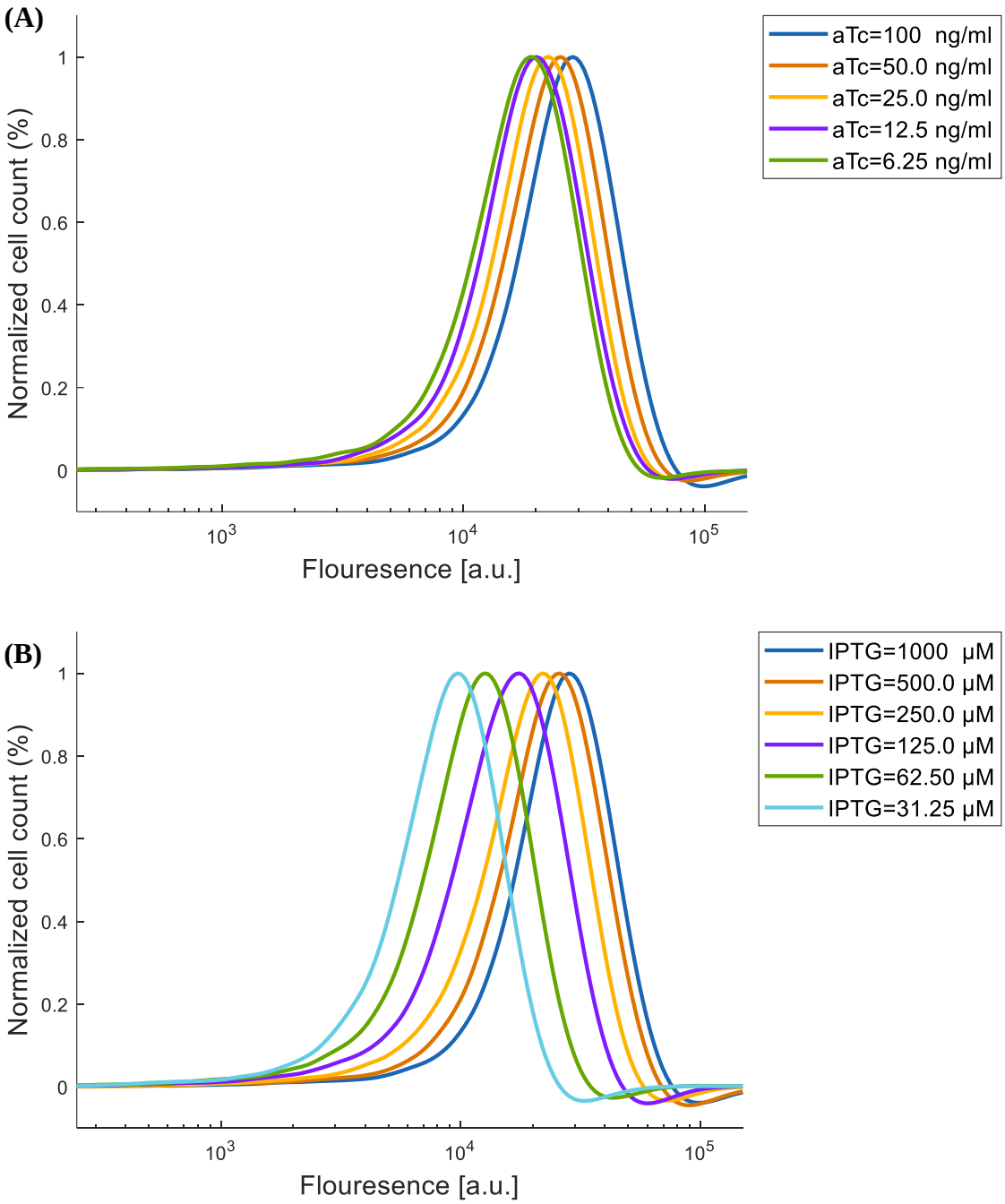


Fig. S14.61. GFP flow cytometry data for a population of cells containing the linear summation using ANF (Fig. S2.20). (A) IPTG was held constant at 1000 µM and aTc was varied. (B) aTc was held constant at 100 ng/ml and IPTG was varied.

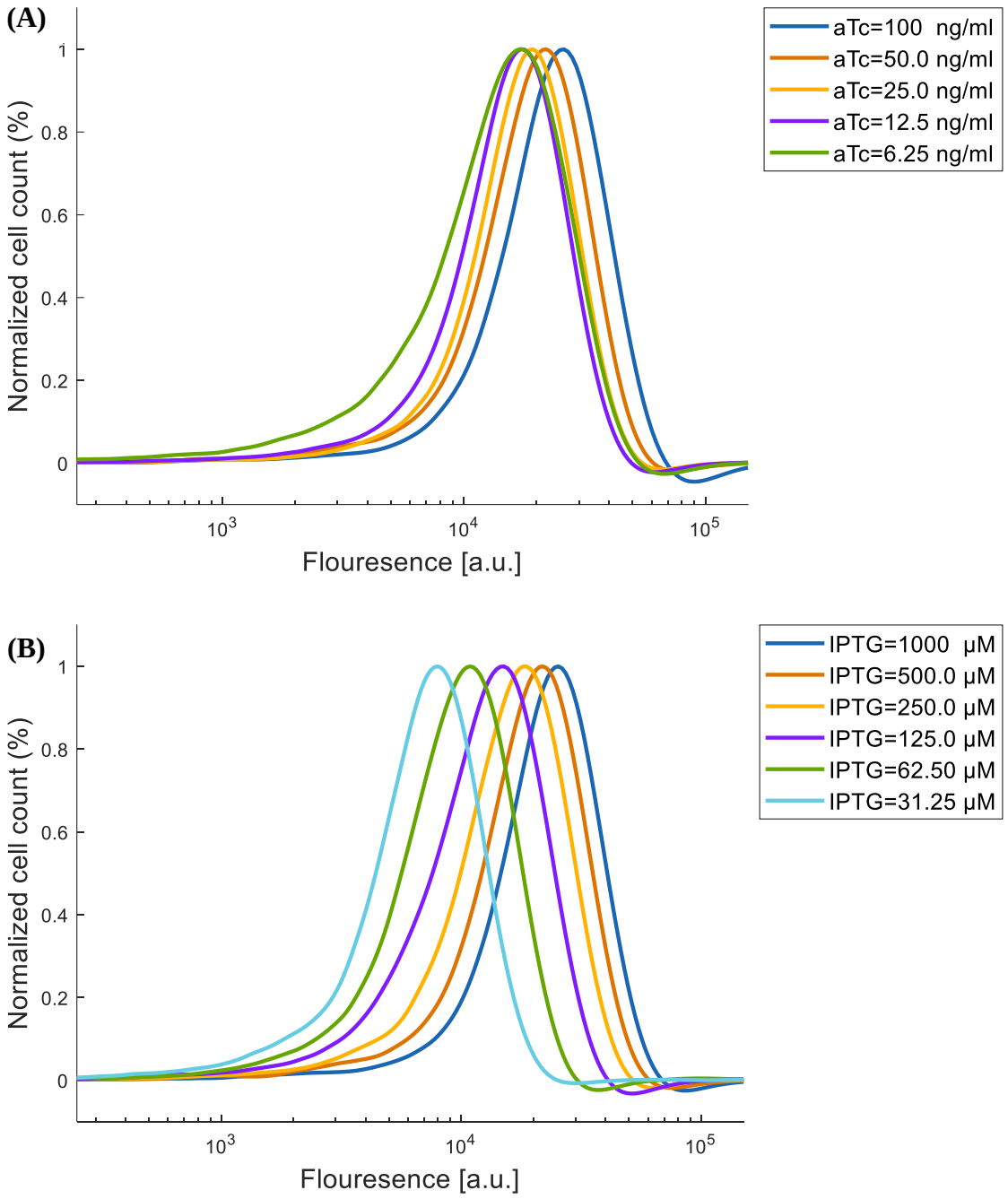


Fig. S14.62. GFP flow cytometry data for a population of cells containing the linear summation using ANF (Fig. S2.20). (A) IPTG was held constant at 500 μM and aTc was varied. (B) aTc was held constant at 50 ng/ml and IPTG was varied.

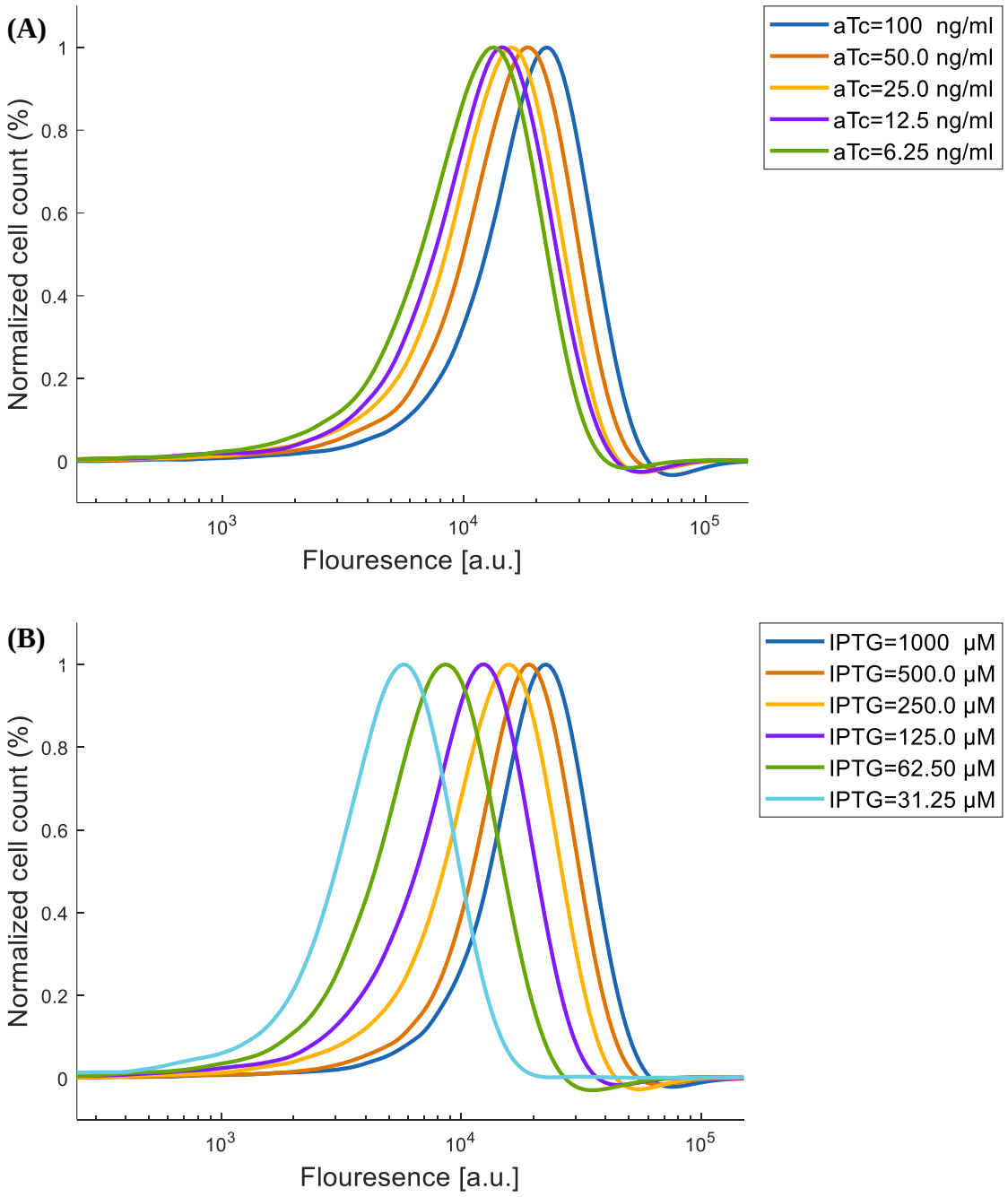


Fig. S14.63. GFP flow cytometry data for a population of cells containing the linear summation using ANF (Fig. S2.20). (A) IPTG was held constant at 250 μM and aTc was varied. (B) aTc was held constant at 25 ng/ml and IPTG was varied.

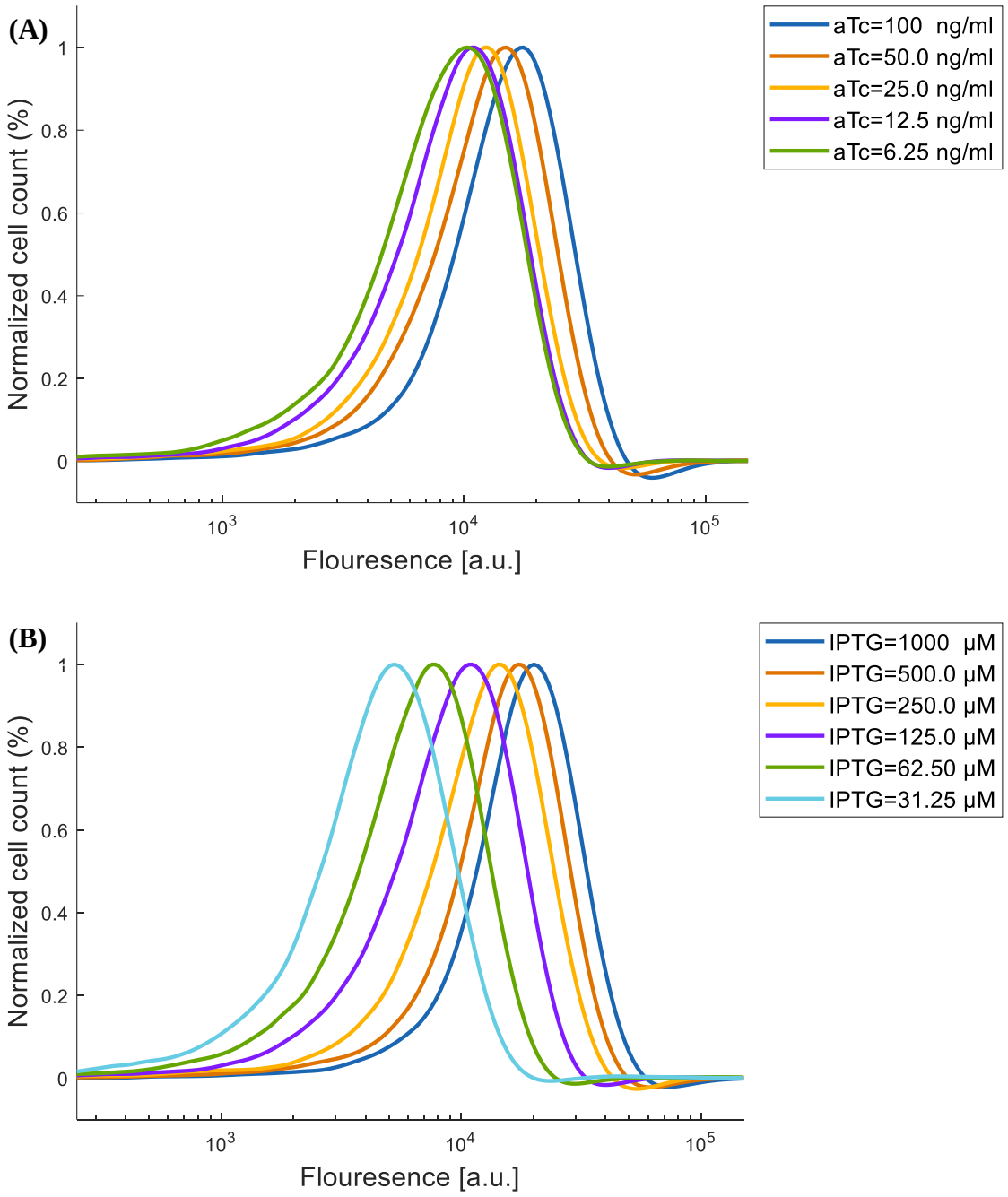


Fig. S14.64. GFP flow cytometry data for a population of cells containing the linear summation using ANF (Fig. S2.20). (A) IPTG was held constant at 125 μM and aTc was varied. (B) aTc was held constant at 12.5 ng/ml and IPTG was varied.

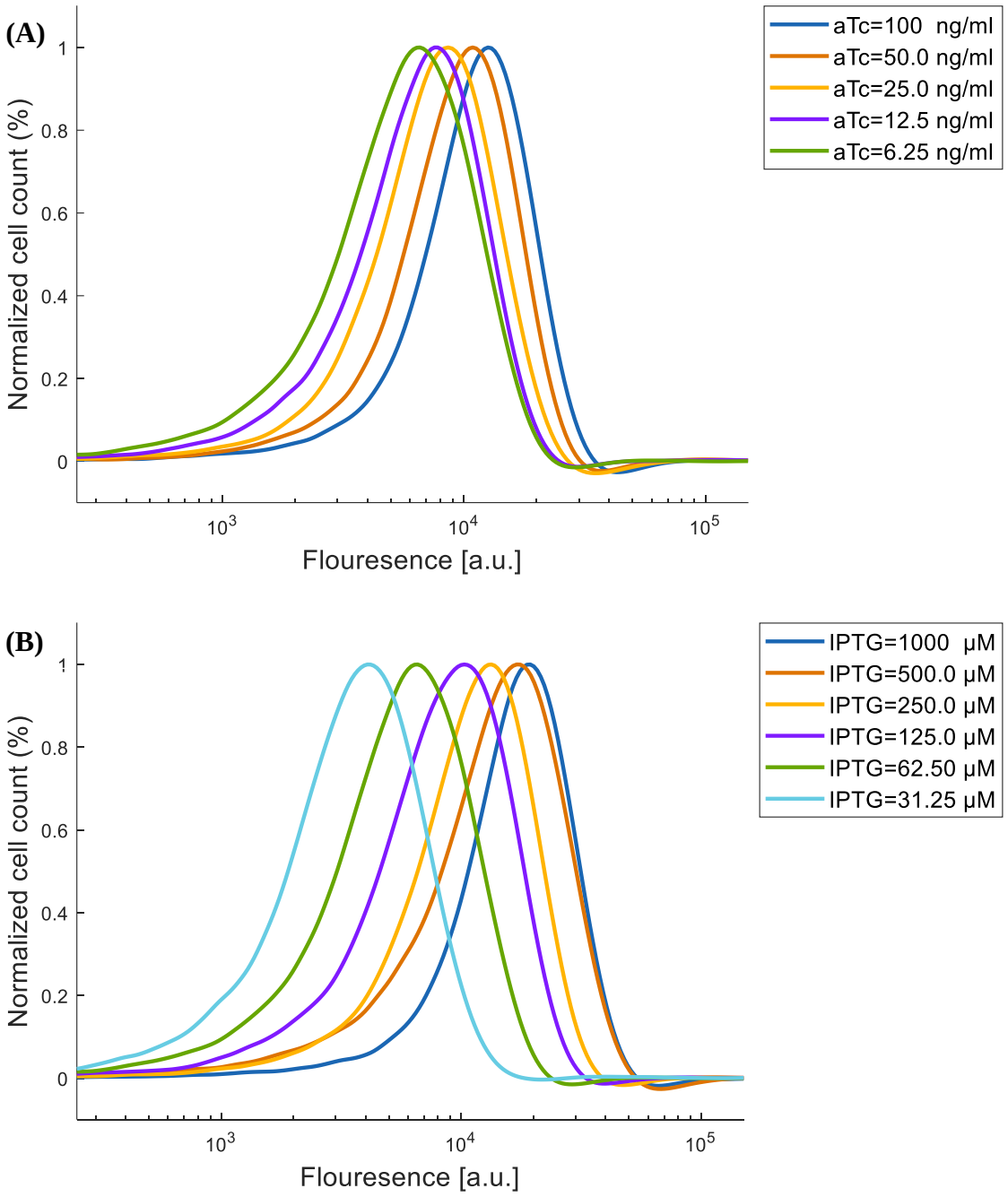


Fig. S14.65. GFP flow cytometry data for a population of cells containing the the linear summation using ANF (Fig. S2.20). (A) IPTG was held constant at 62.5 µM and aTc was varied. (B) aTc was held constant at 6.25 ng/ml and IPTG was varied.

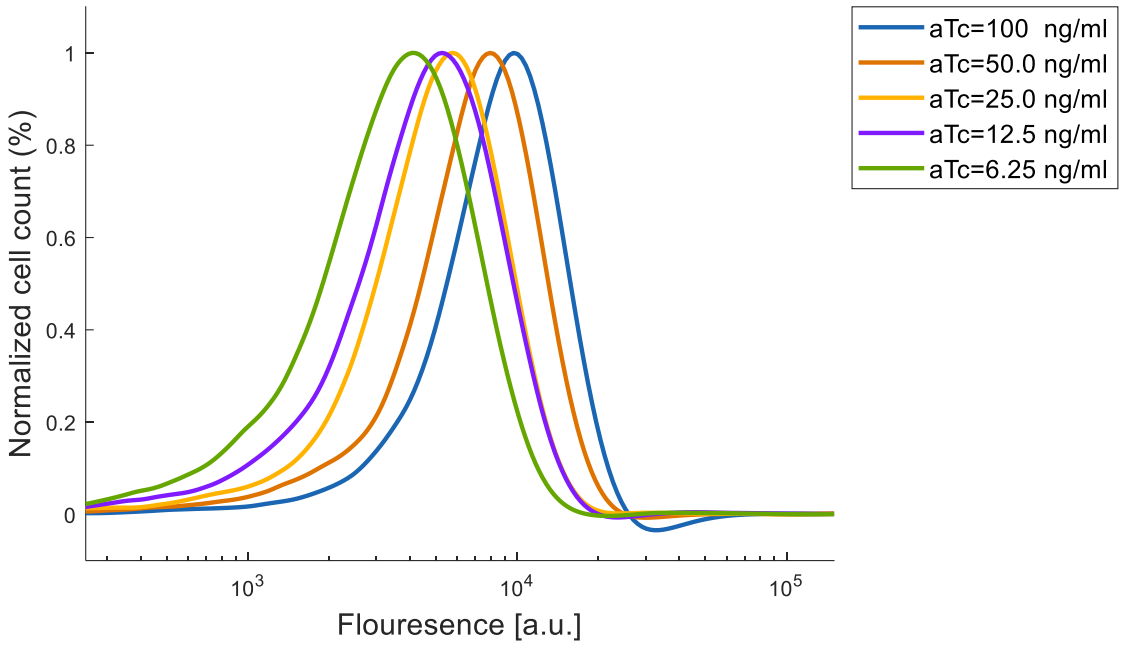


Fig. S14.66. GFP flow cytometry data for a population of cells containing the the linear summation using ANF (Fig. S2.20). IPTG was held constant at 31.25 μM and aTc was varied.

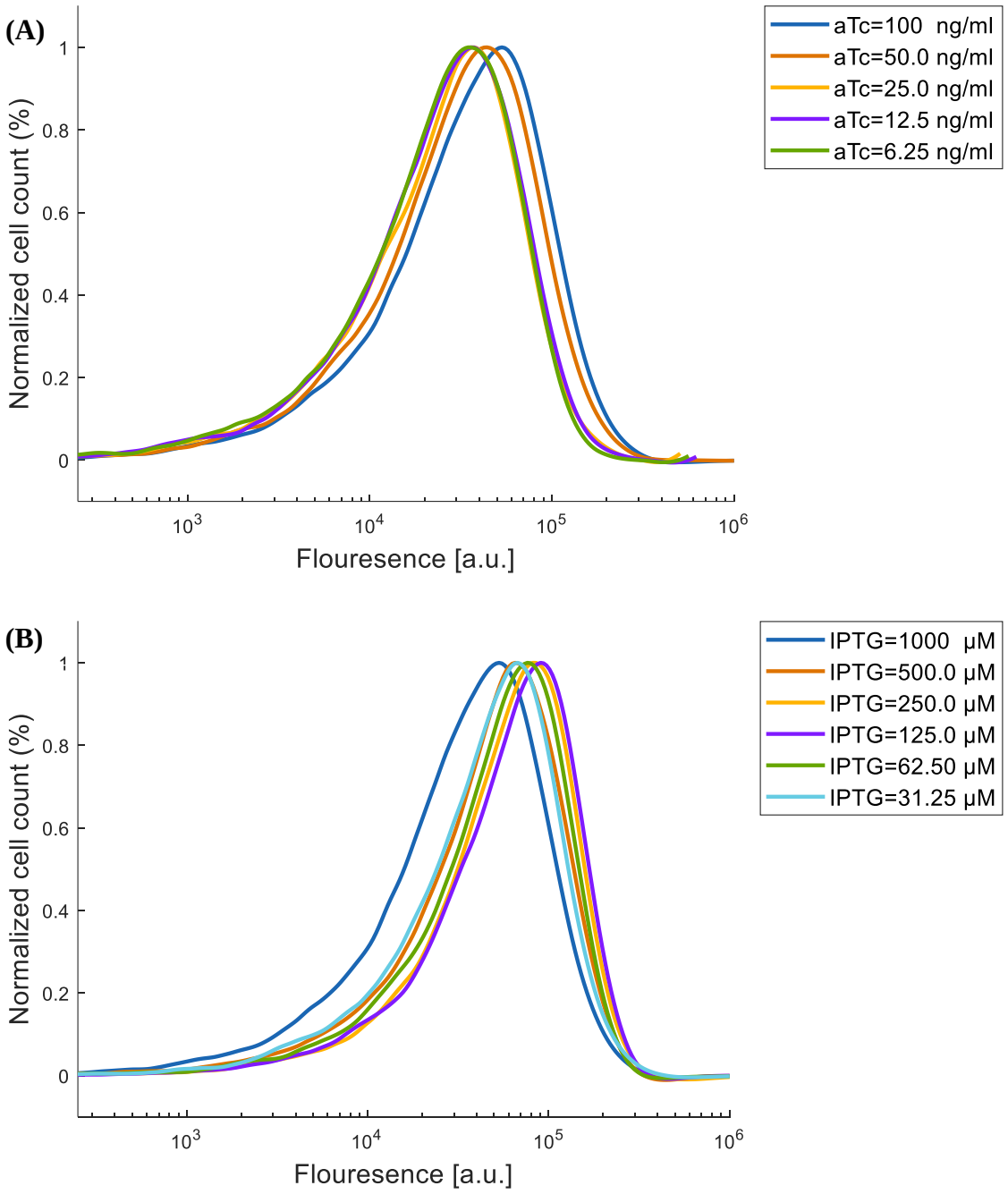


Fig. S14.67. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21A). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 1000 µM and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 100 ng/ml and IPTG was varied.

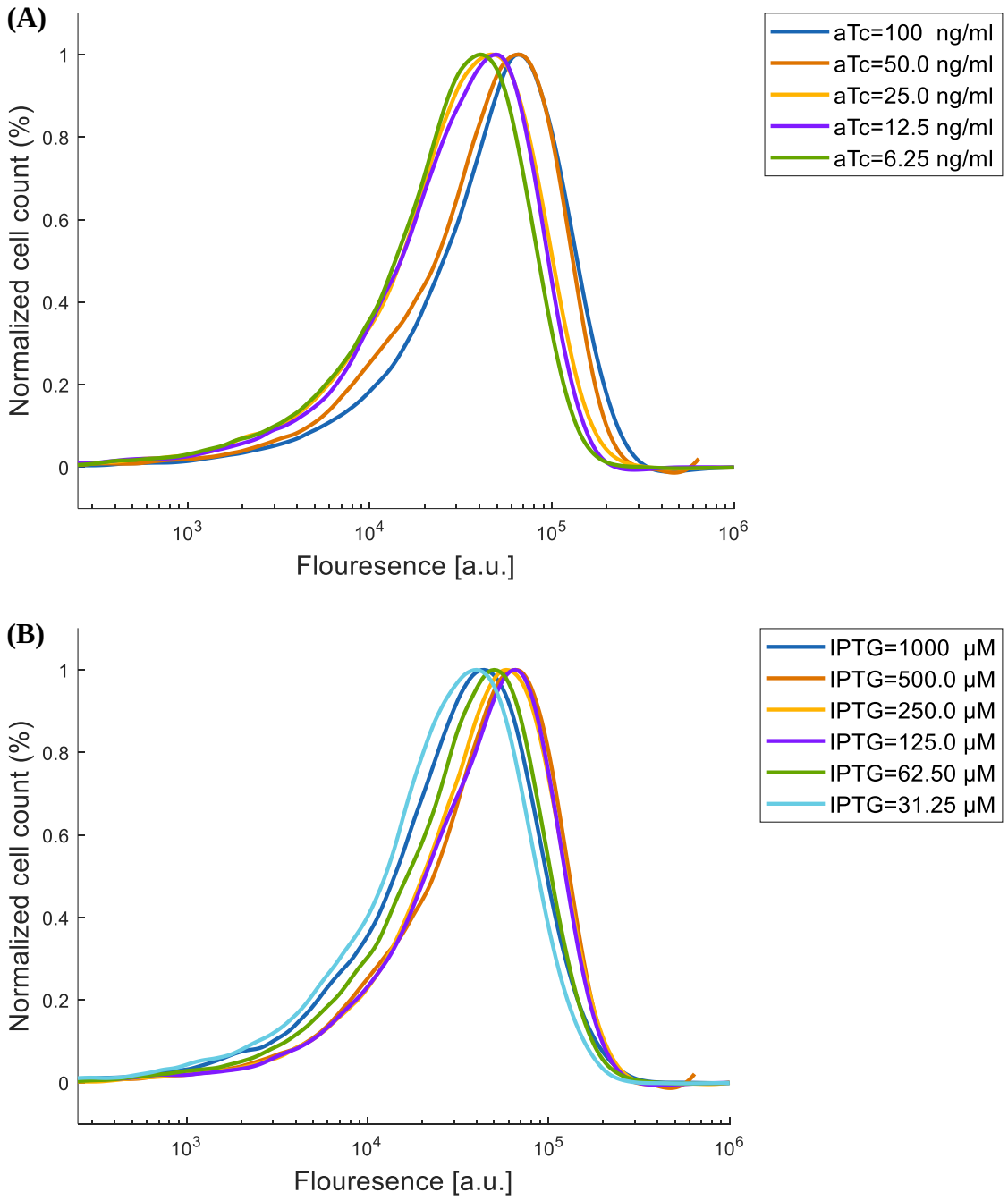


Fig. S14.68. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21A). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 500 μM and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 50 ng/ml and IPTG was varied.

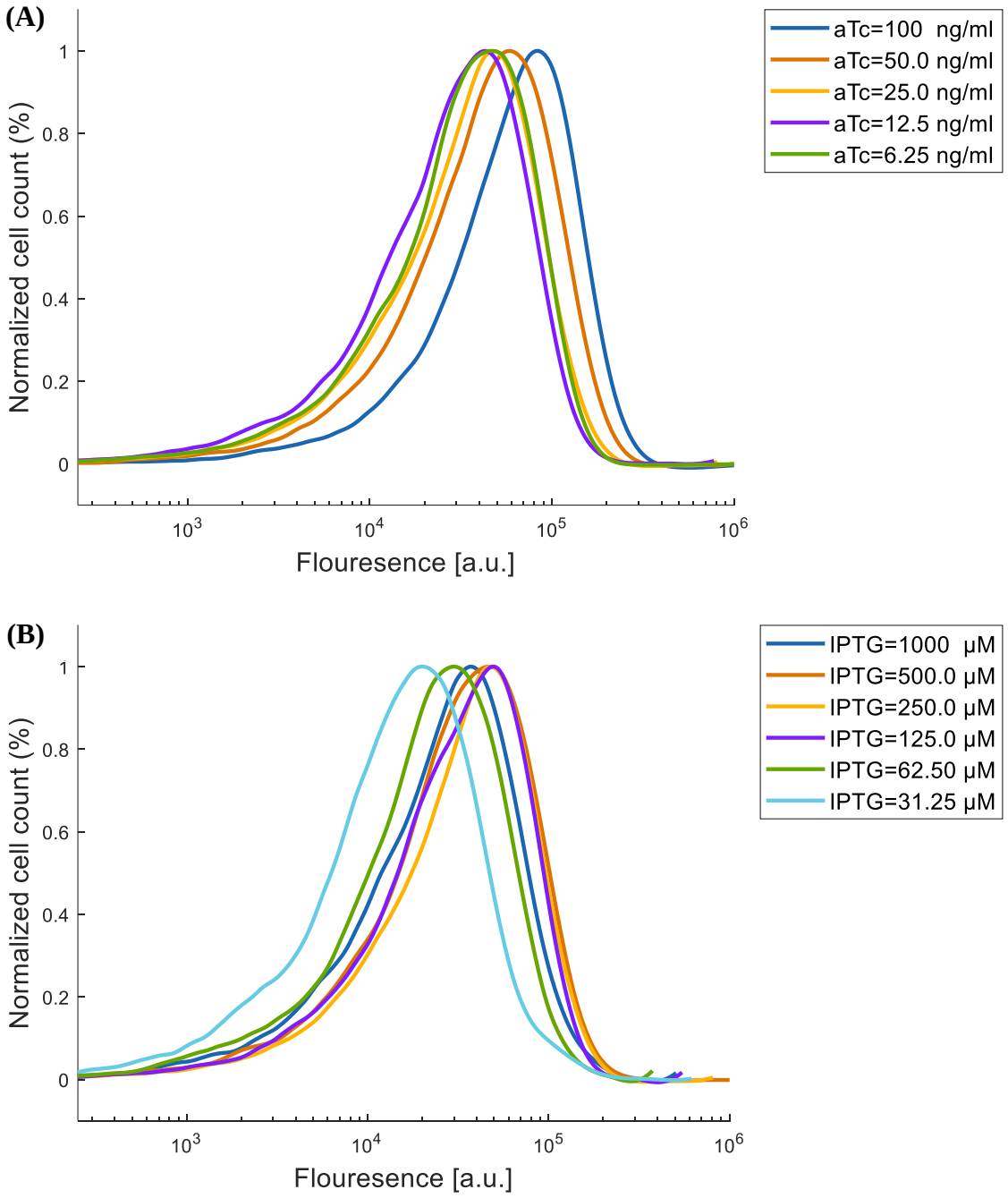


Fig. S14.69. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21A). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 250 μM and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 25 ng/ml and IPTG was varied.

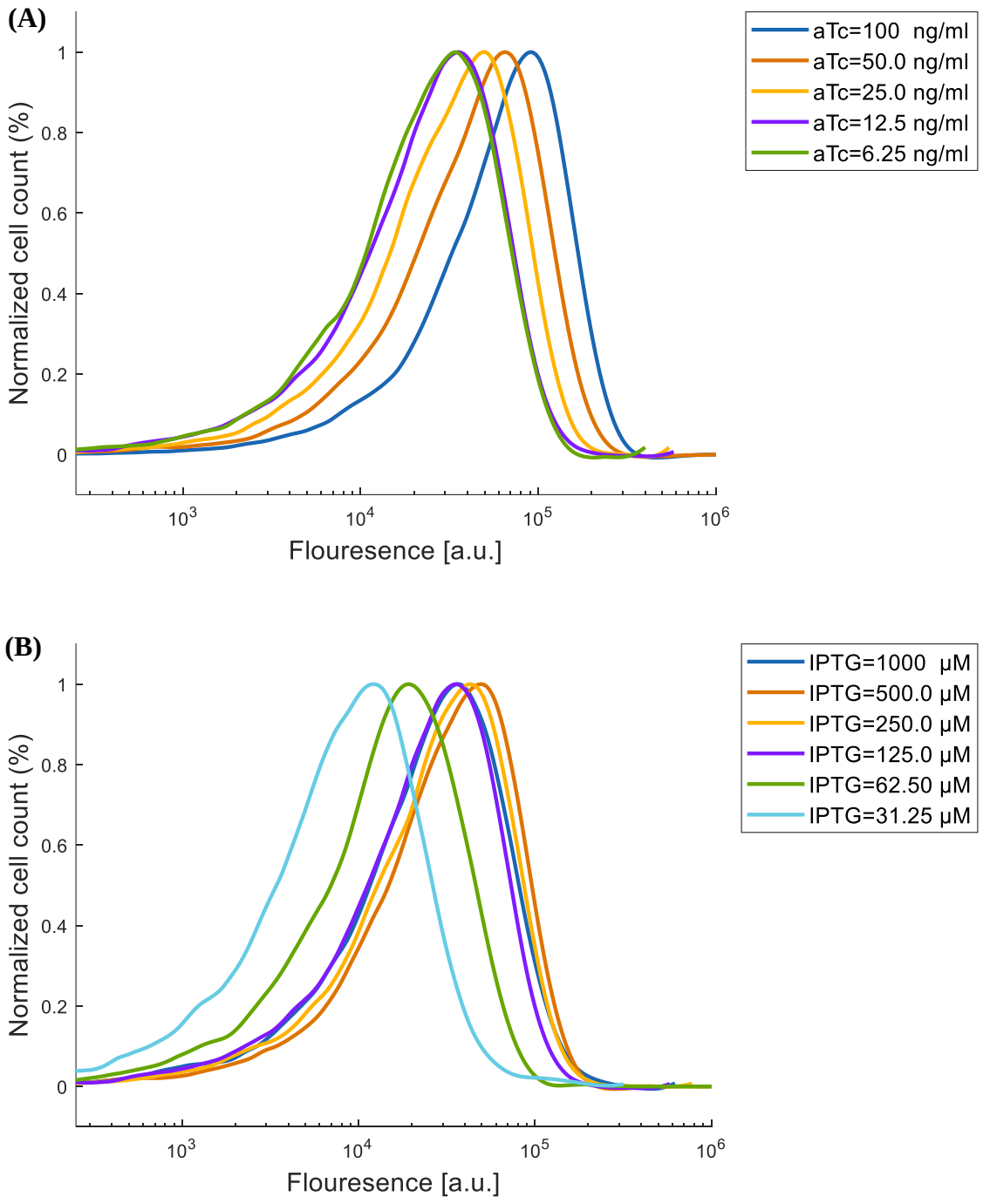


Fig. S14.70. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21A). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 125 μM and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 12.5 ng/ml and IPTG was varied.

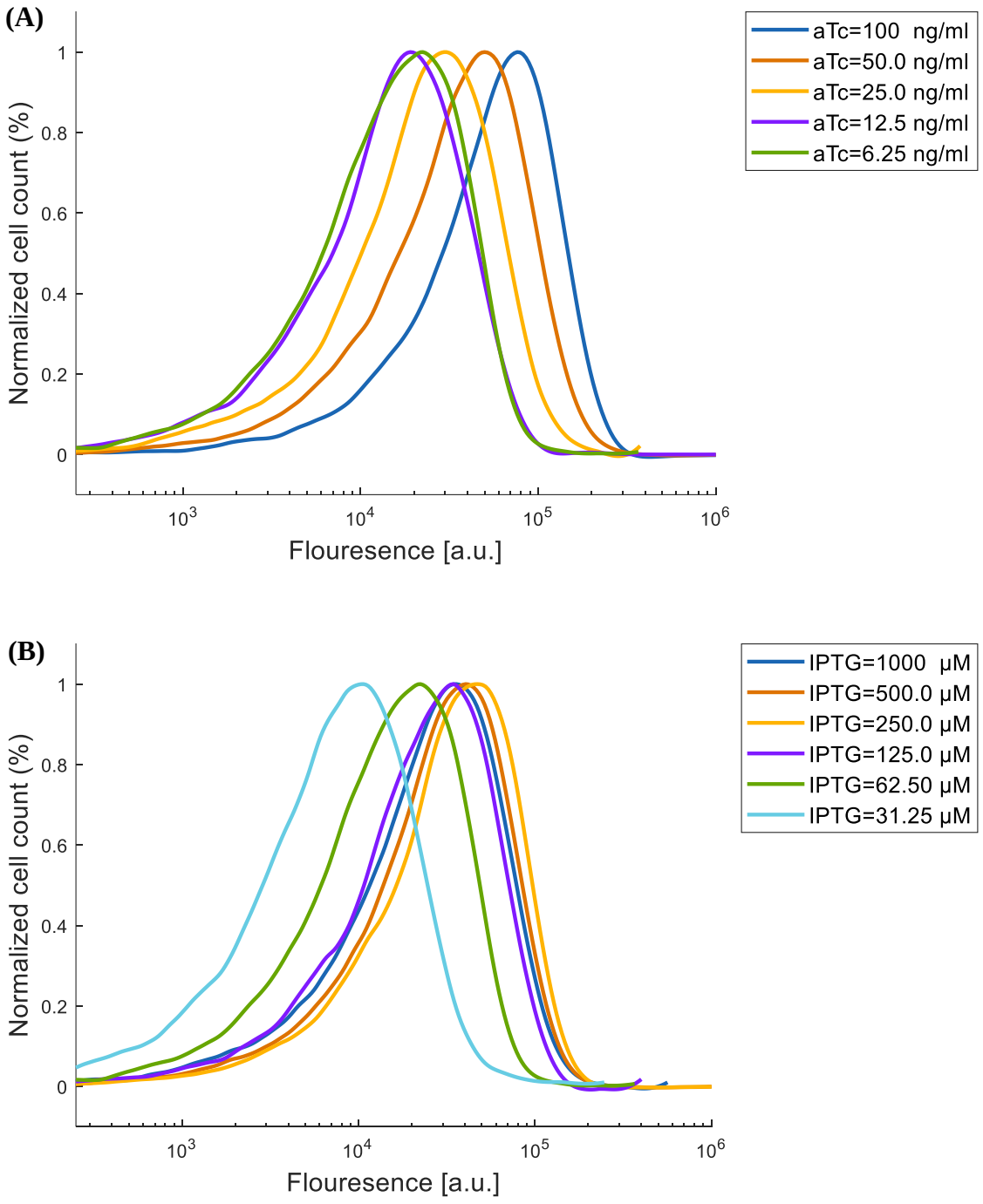


Fig. S14.71. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21A). (A) Arabinose was held constant at 0.04 mM, IPTG was held constant at 62.5 µM and aTc was varied. (B) Arabinose was held constant at 0.04 mM, aTc was held constant at 6.25 ng/ml and IPTG was varied.

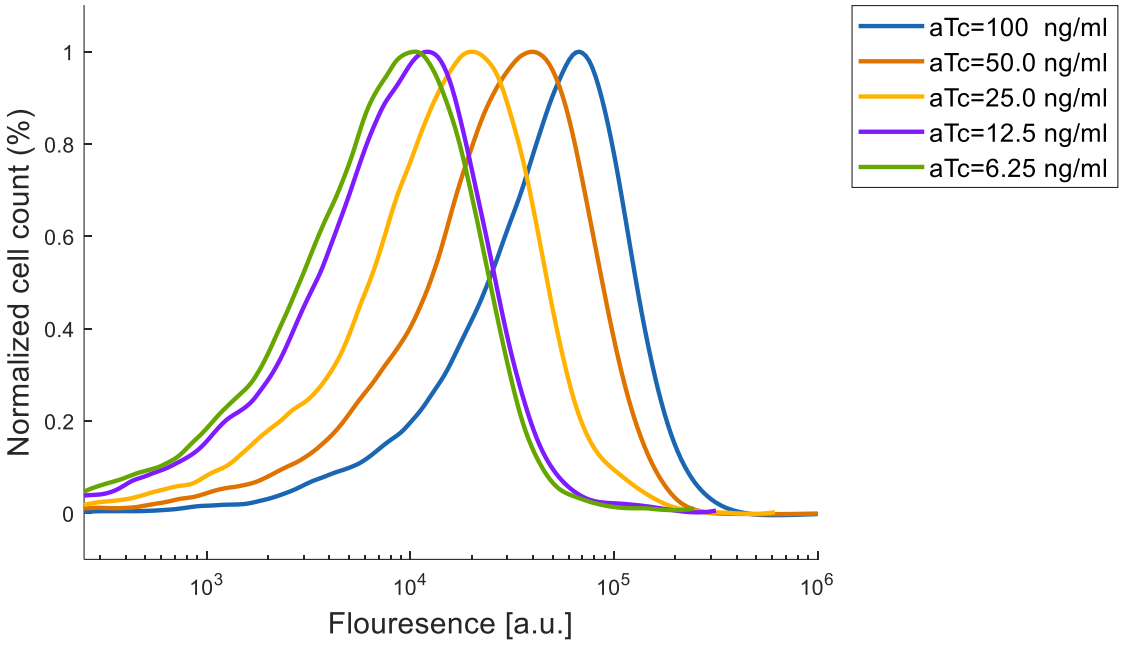


Fig. S14.72. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21A).. Arabinose was held constant at 0.04 mM, IPTG was held constant at 31.25 μM and aTc was varied.

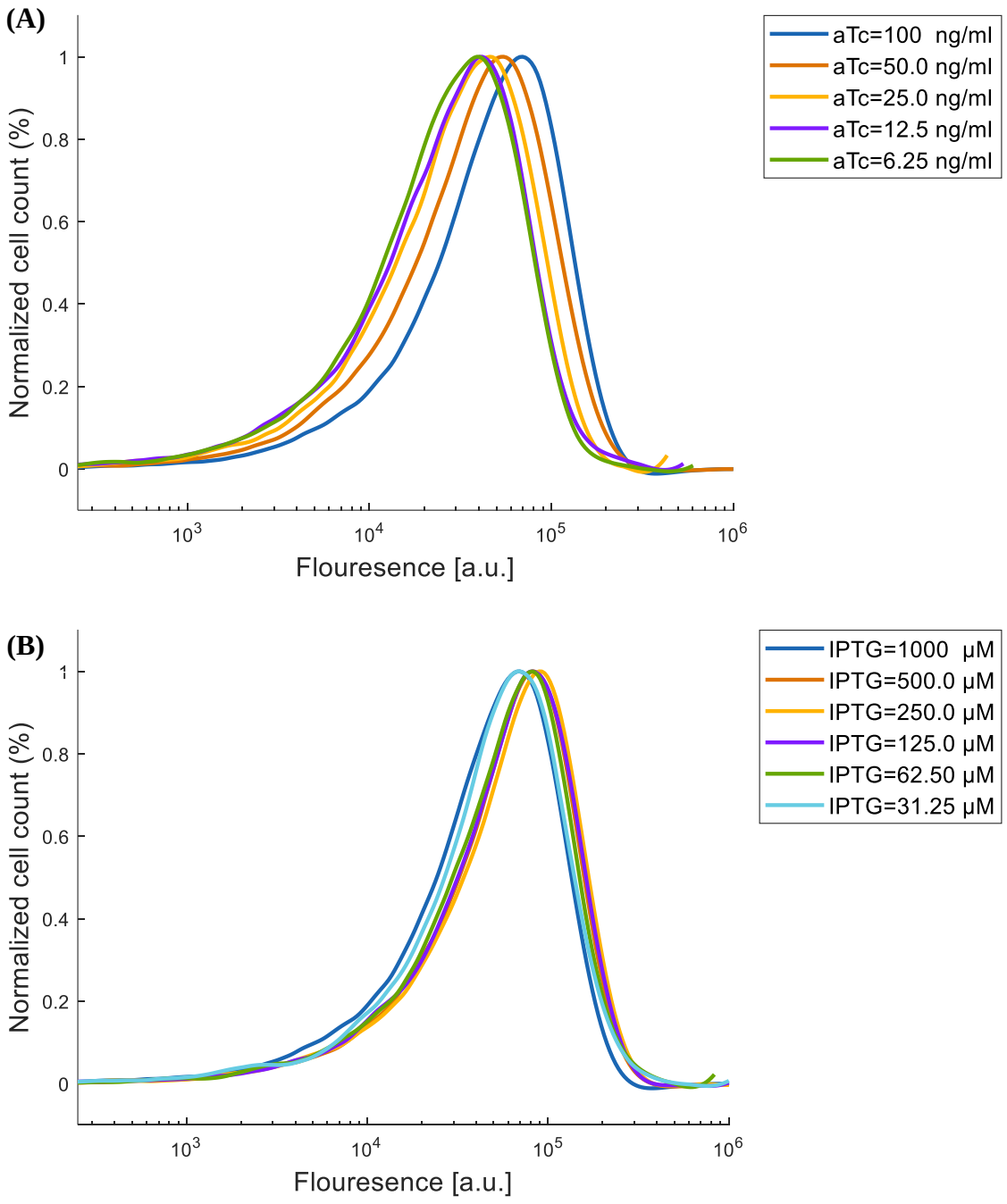


Fig. S14.73. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21C). (A) Arabinose was held constant at 0.08 mM, IPTG was held constant at 1000 µM and aTc was varied. (B) Arabinose was held constant at 0.08 mM, aTc was held constant at 100 ng/ml and IPTG was varied.

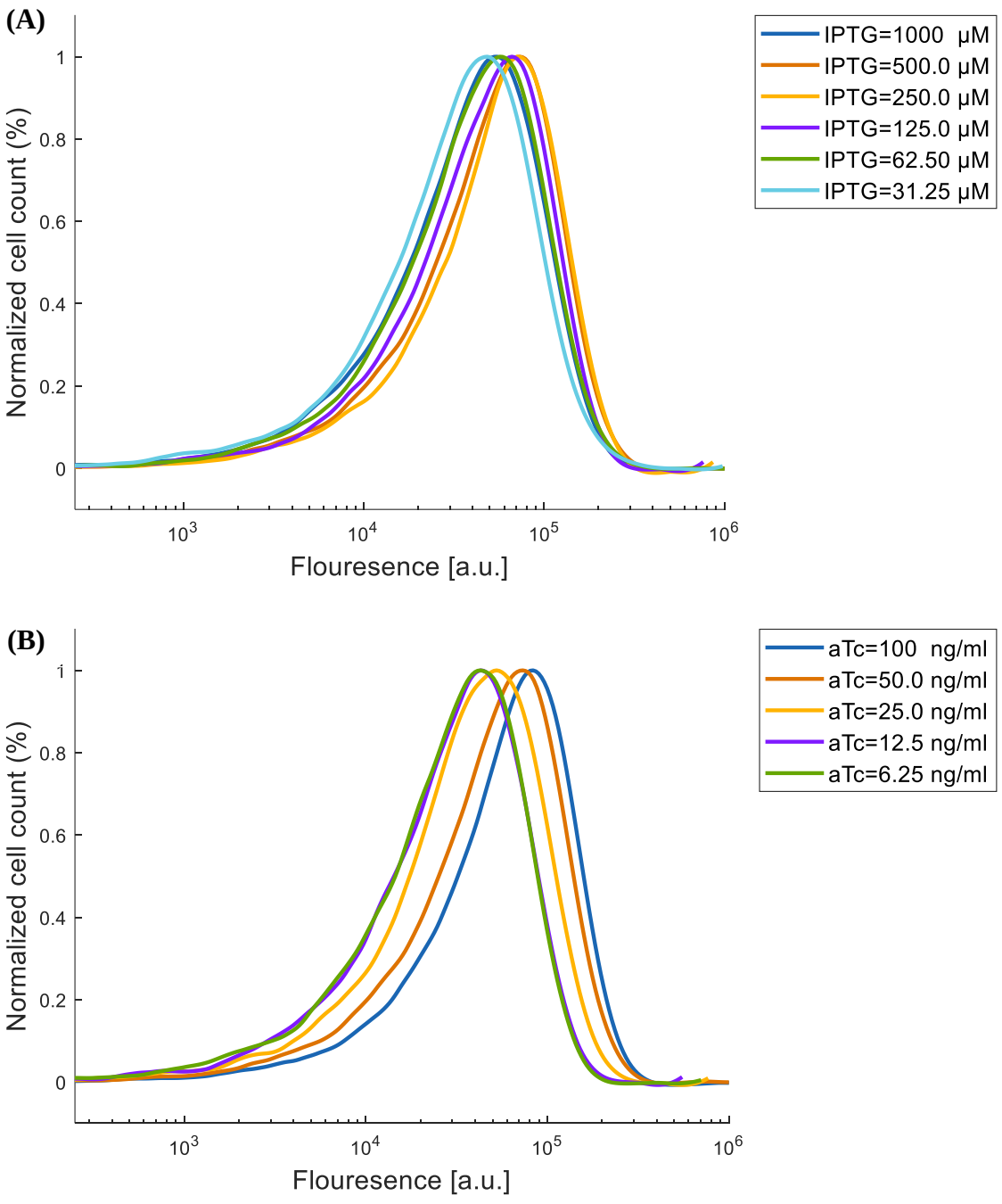


Fig. S14.74. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21C). (A) Arabinose was held constant at 0.08 mM, IPTG was held constant at 500 μM and aTc was varied. (B) Arabinose was held constant at 0.08 mM, aTc was held constant at 50 ng/ml and IPTG was varied.

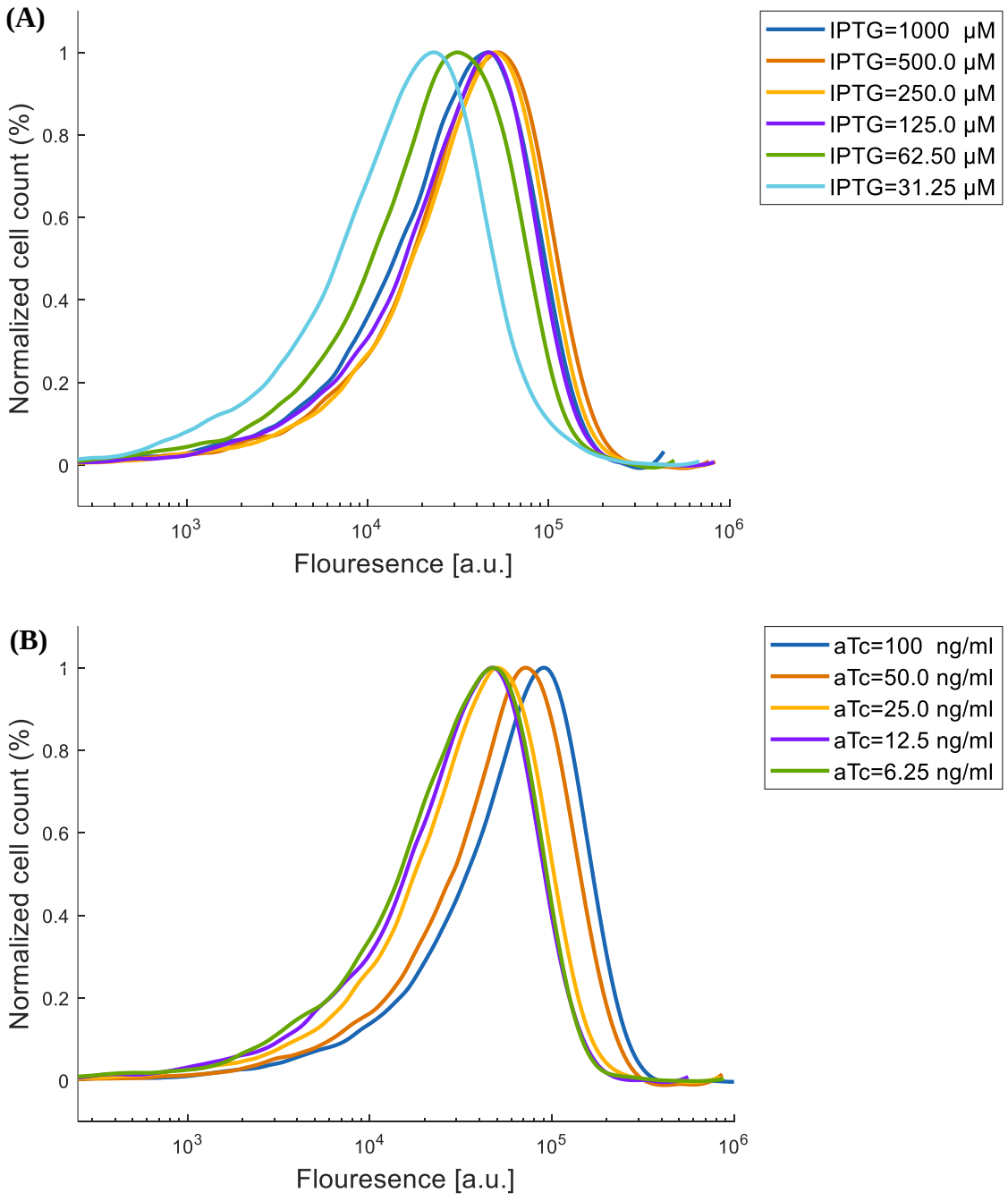


Fig. S14.75. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21C). (A) Arabinose was held constant at 0.08 mM, IPTG was held constant at 250 μM and aTc was varied. (B) Arabinose was held constant at 0.08 mM, aTc was held constant at 25 ng/ml and IPTG was varied.

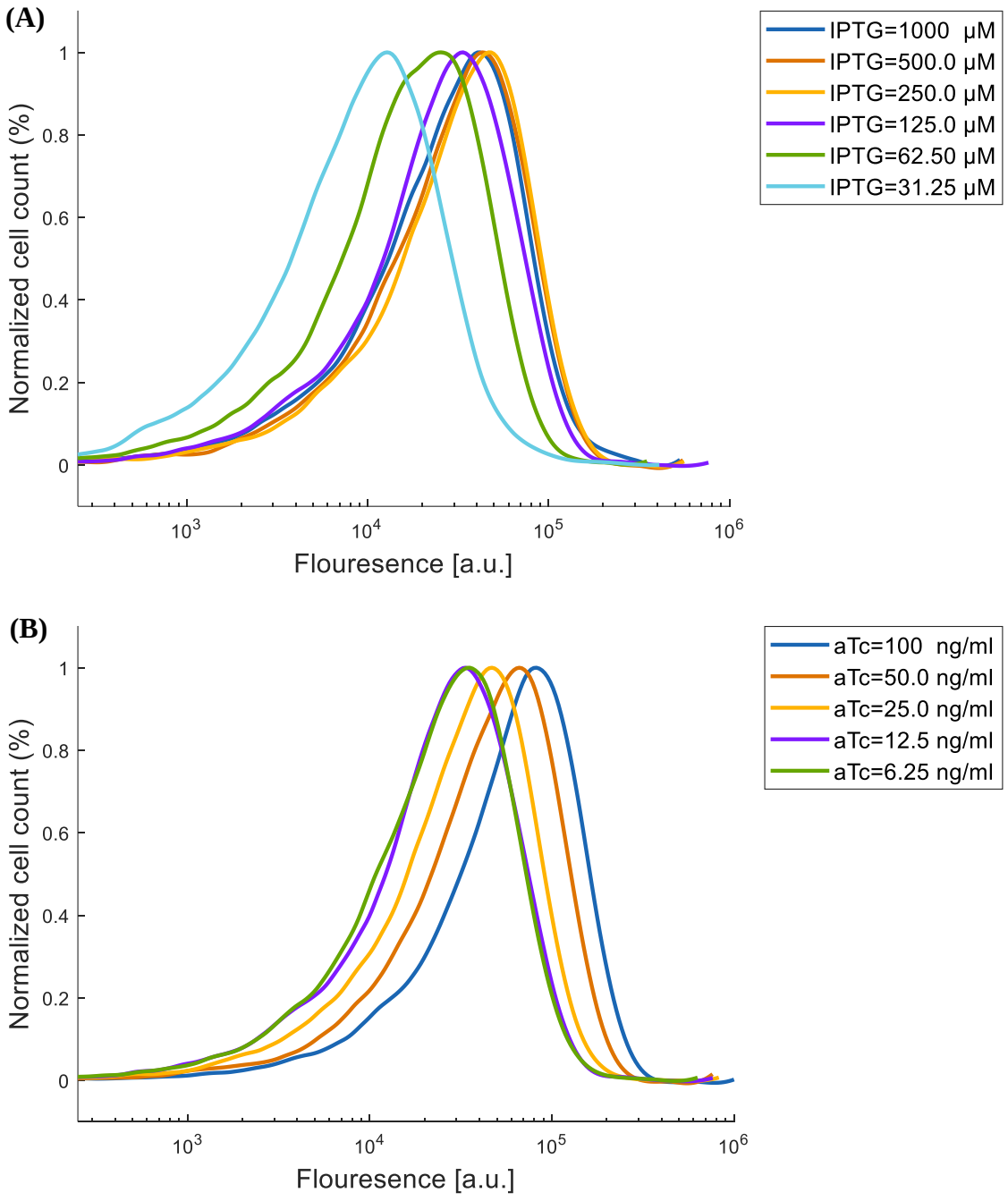


Fig. S14.76. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21C). (A) Arabinose was held constant at 0.08 mM, IPTG was held constant at 125 μM and aTc was varied. (B) Arabinose was held constant at 0.08 mM, aTc was held constant at 12.5 ng/ml and IPTG was varied.

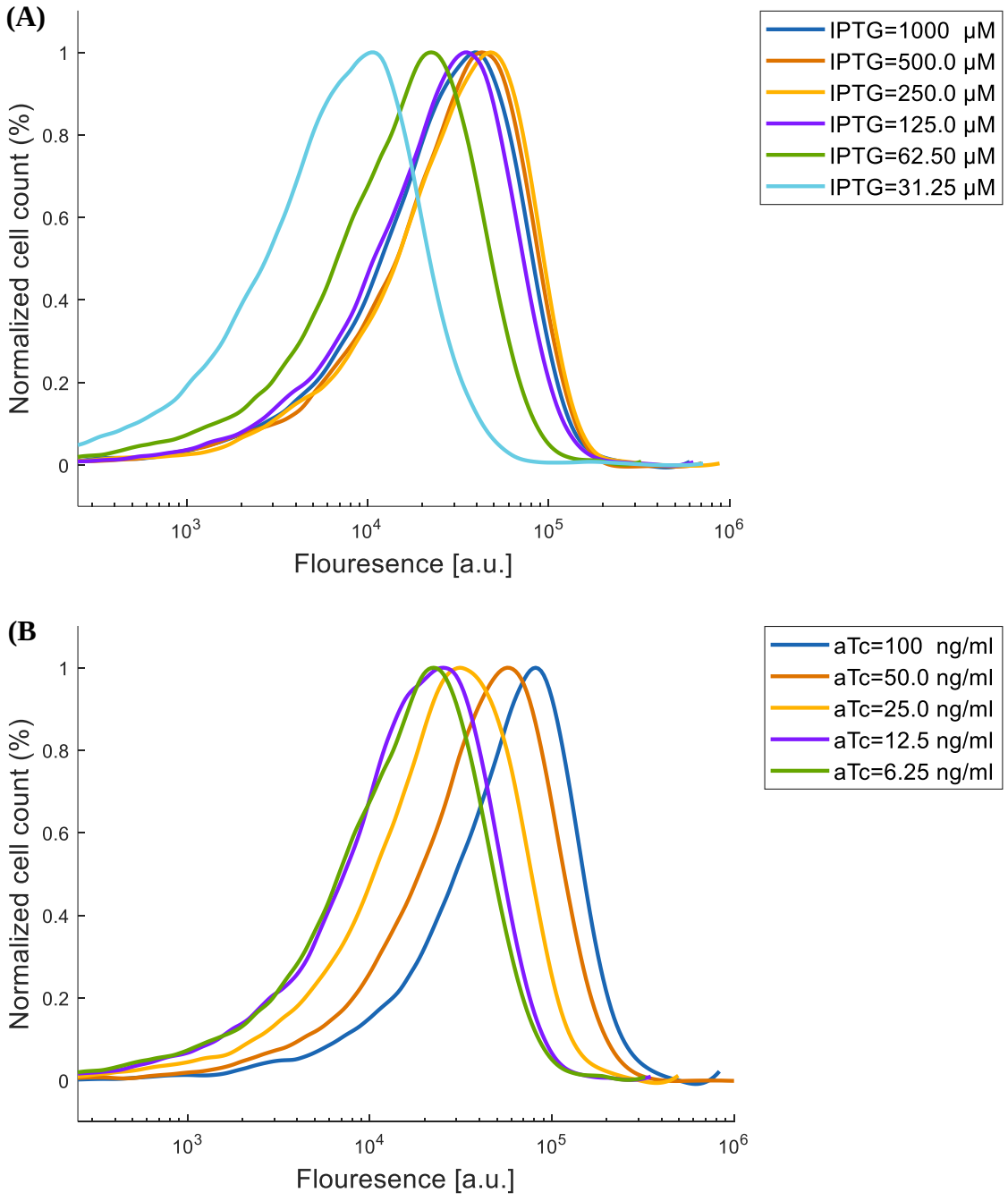


Fig. S14.77. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21C). (A) Arabinose was held constant at 0.08 mM, IPTG was held constant at 62.5 μM and aTc was varied. (B) Arabinose was held constant at 0.08 mM, aTc was held constant at 6.25 ng/ml and IPTG was varied.

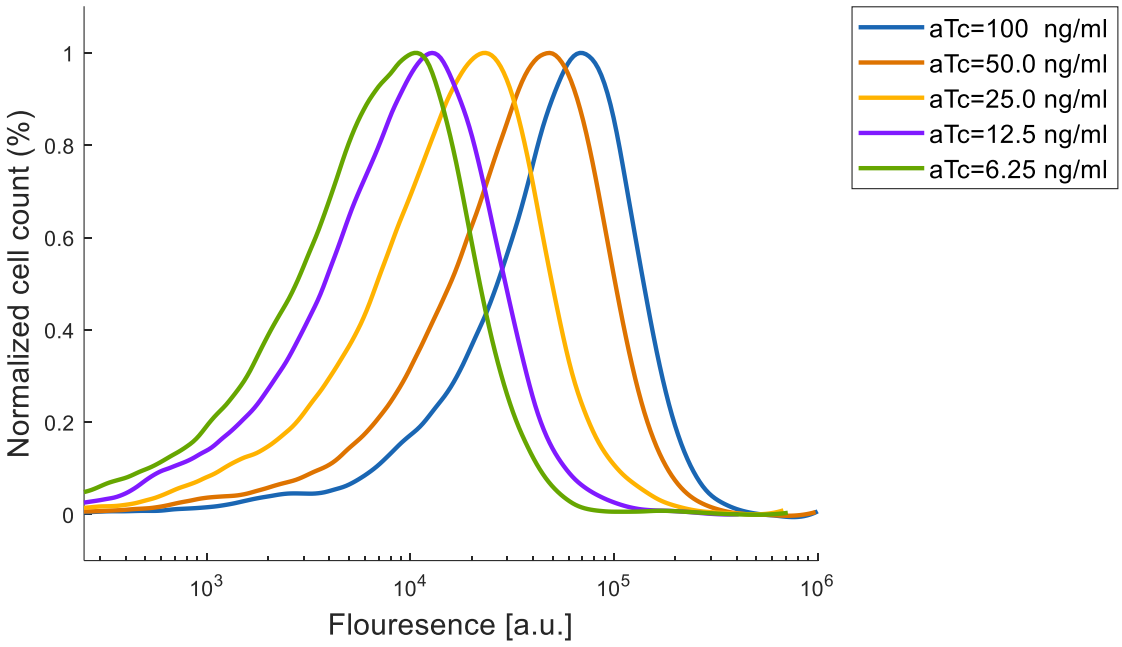


Fig. S14.78. GFP flow cytometry data for a population of cells containing the genetic perceptron in the linear domain. AraC truncated was used to improve the compatibility of Arabinose and IPTG (Fig. S2.21C). Arabinose was held constant at 0.04 mM, IPTG was held constant at 31.25 μM and aTc was varied.

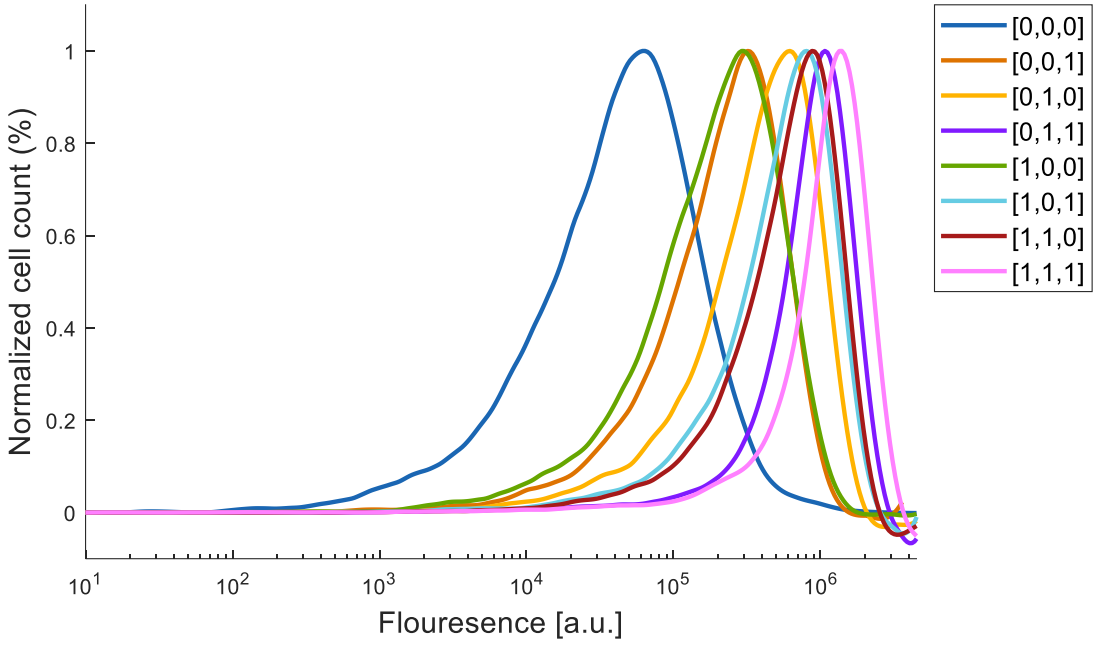


Fig. S14.79. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TCTA Mutation (Fig. 3G and Fig. S6.5B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.25mM].

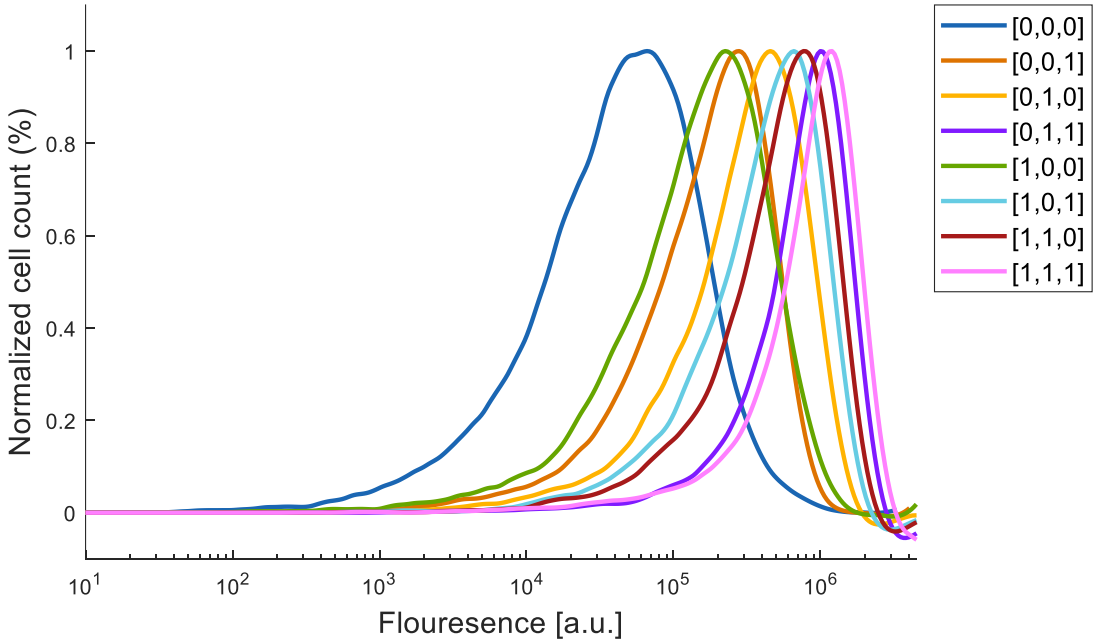


Fig. S14.80. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TCTA Mutation (Fig. 3G and Fig. S6.5B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.125mM].

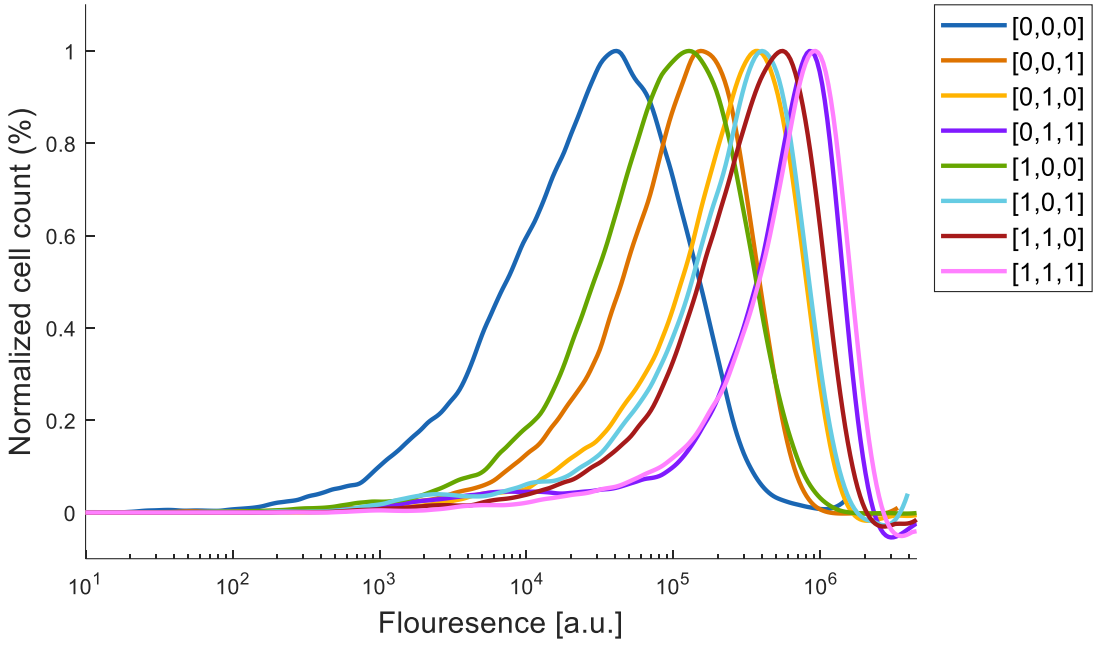


Fig. S14.81. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TCTA Mutation (Fig. 3G and Fig. S6.5B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0625mM].

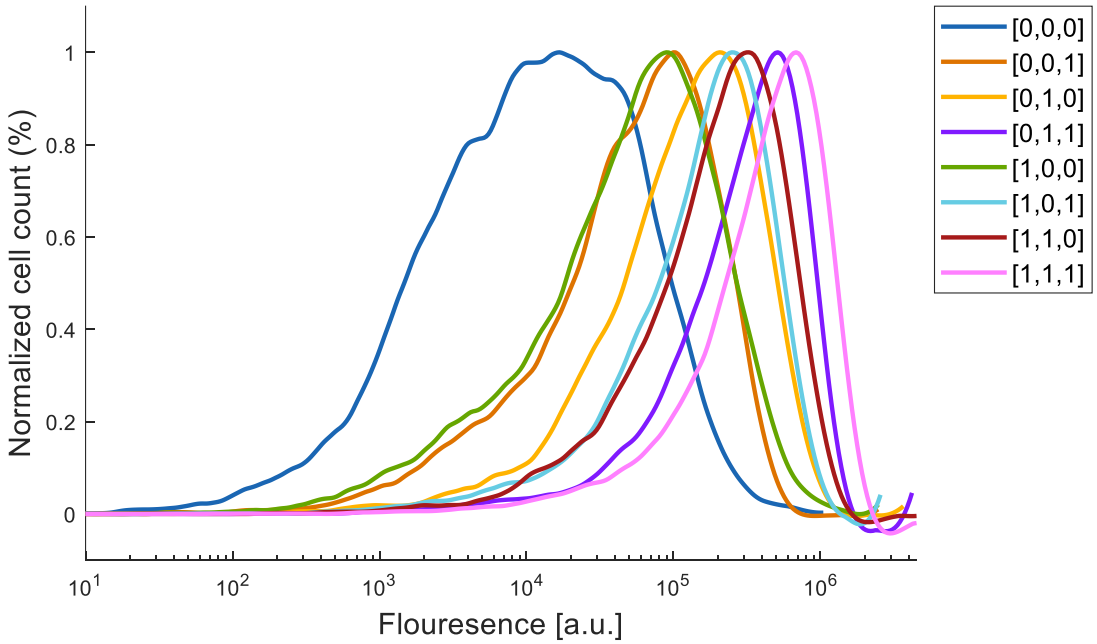


Fig. S14.82. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TCTA Mutation (Fig. 3G and Fig. S6.5B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125 mM].

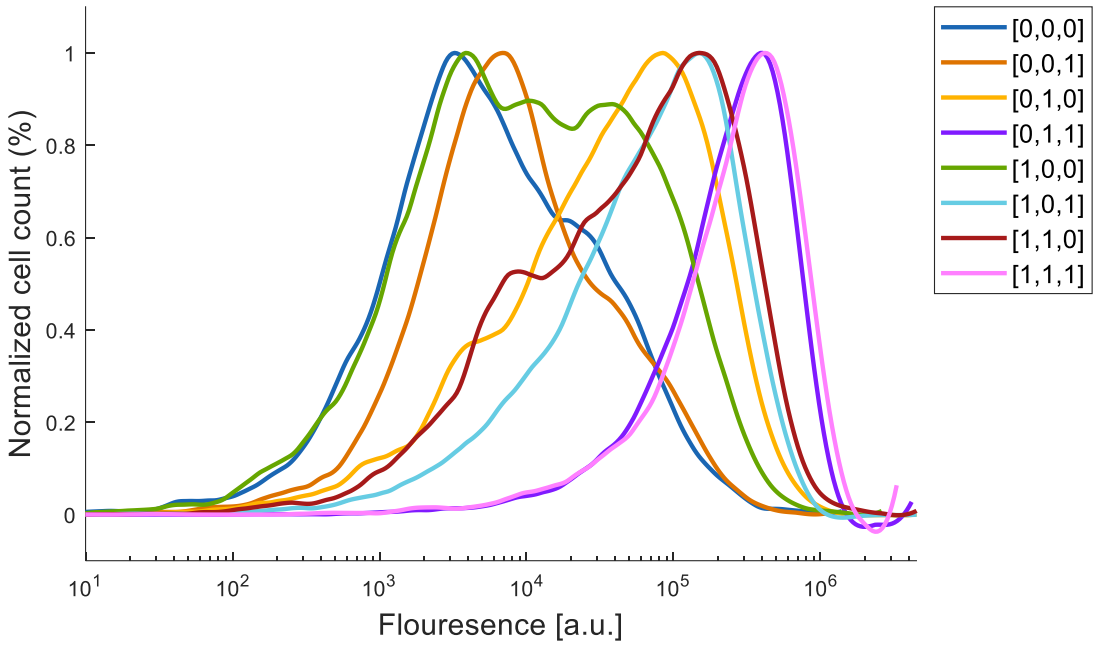


Fig. S14.83. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TCTA Mutation (Fig. 3G and Fig. S6.5B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.015625 mM].

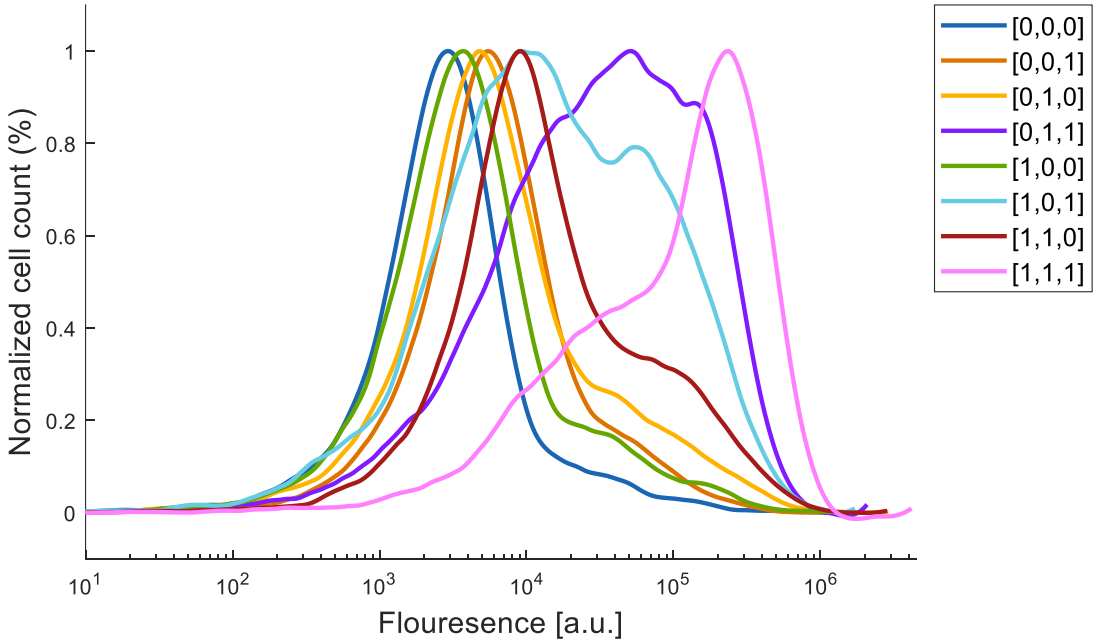


Fig. S14.84. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TCTA Mutation (Fig. 3G and Fig. S6.5B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0078125mM].

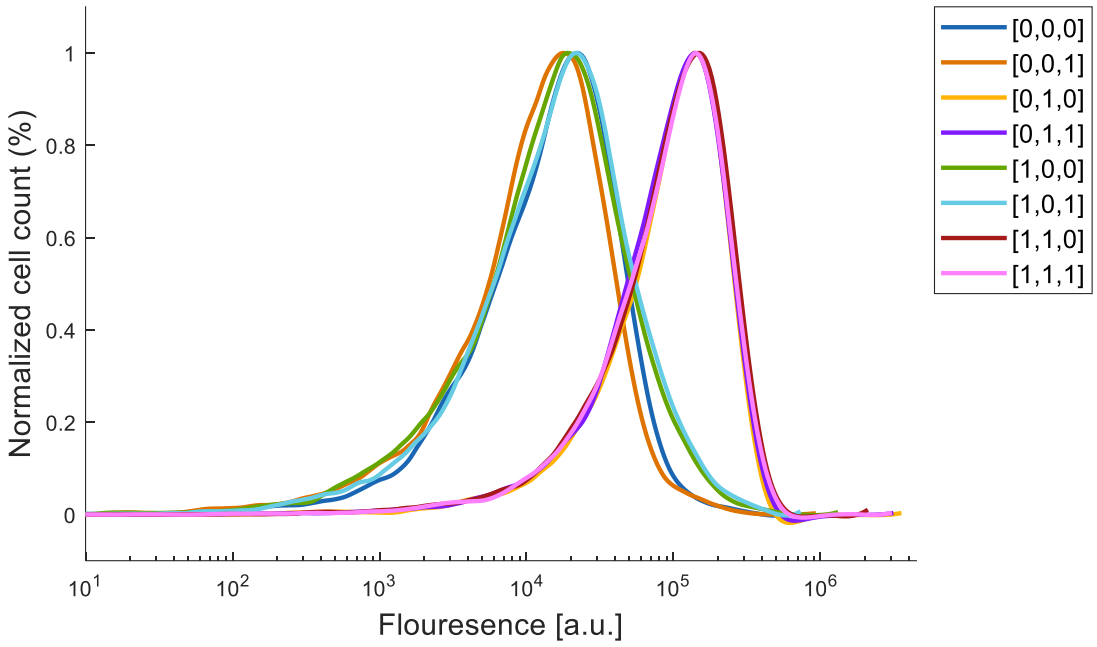


Fig. S14.85. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TCTA Mutation (Fig. S6.7B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.25mM].

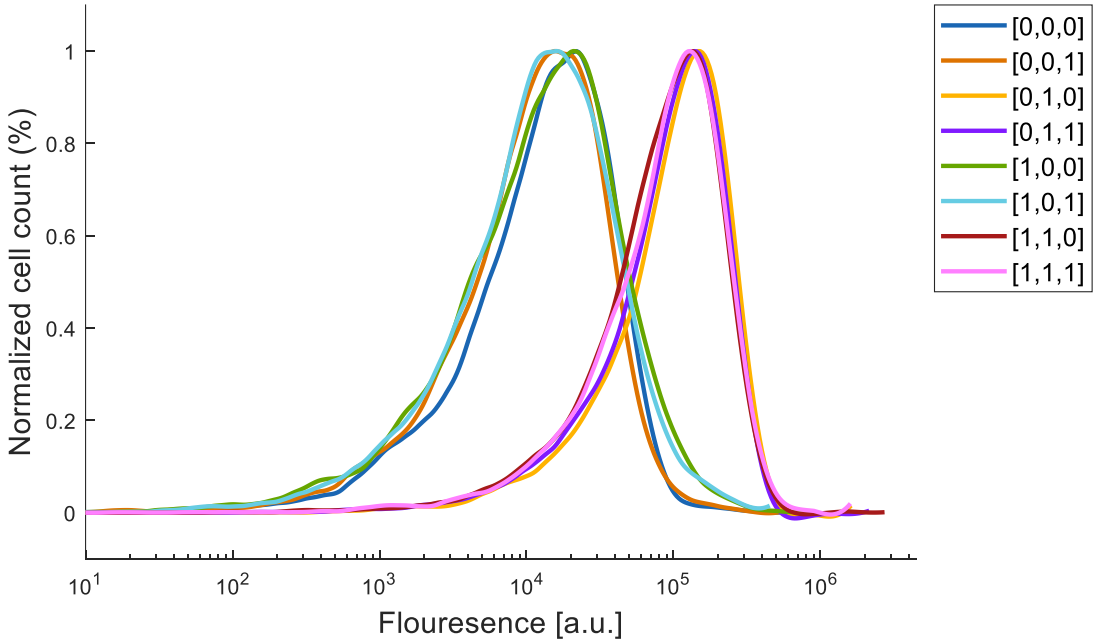


Fig. S14.86. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TCTA Mutation (Fig. S6.7B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.125mM].

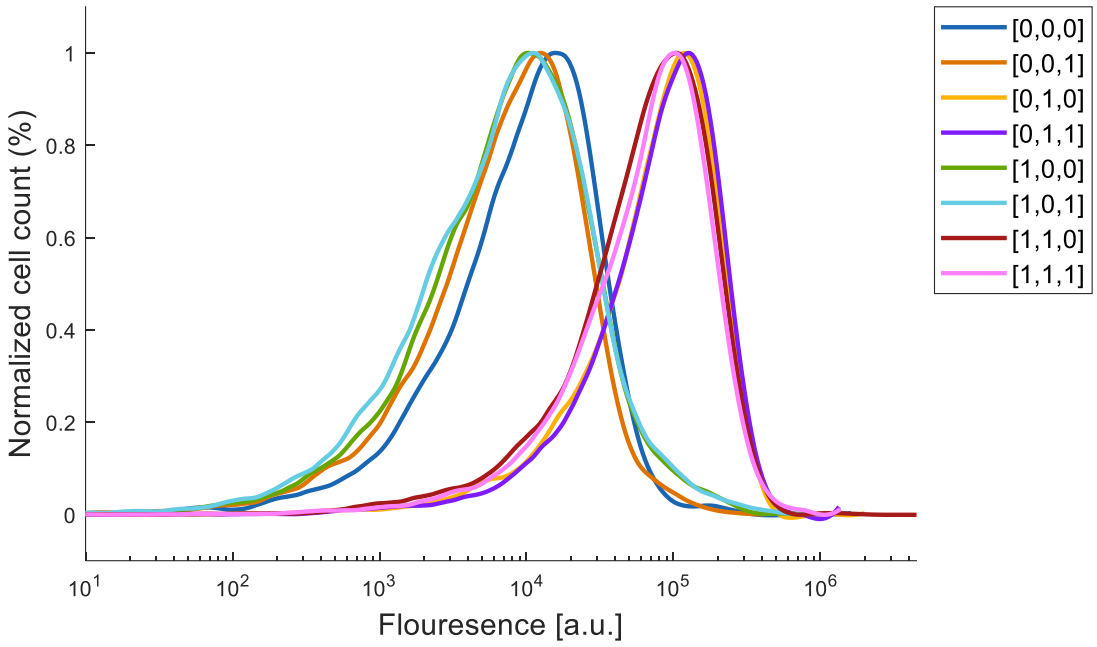


Fig. S14.87. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TCTA Mutation (Fig. S6.7B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0625mM].

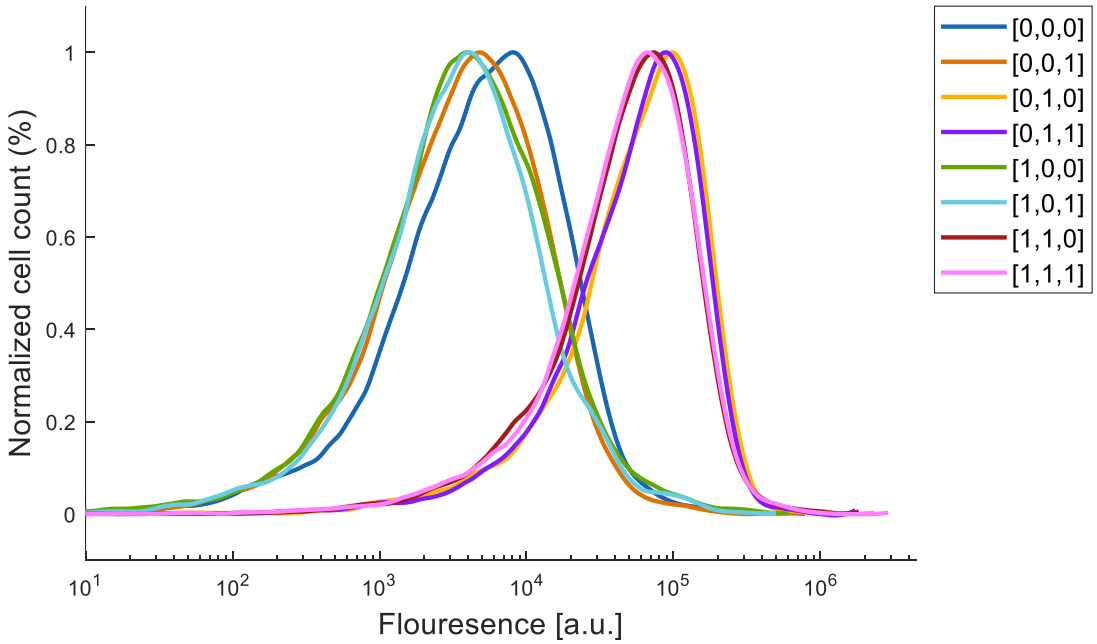


Fig. S14.88. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TCTA Mutation (Fig. S6.7B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125mM].

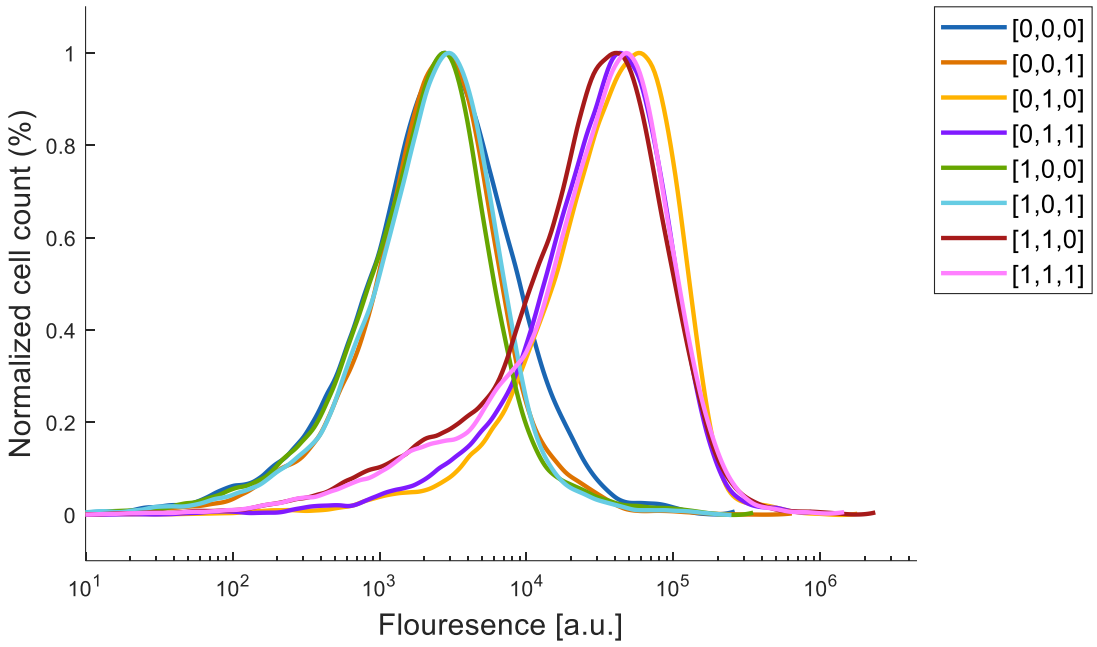


Fig. S14.89. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TCTA Mutation (Fig. S6.7B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.015625mM].

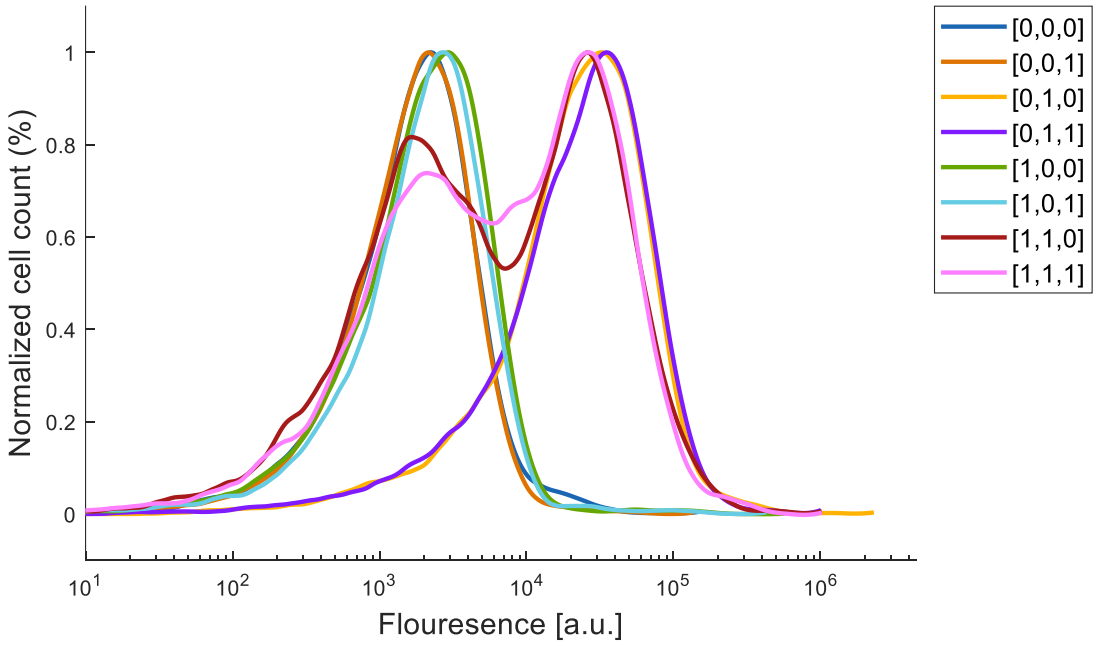


Fig. S14.90. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TCTA Mutation (Fig. S6.7B). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0078125mM].

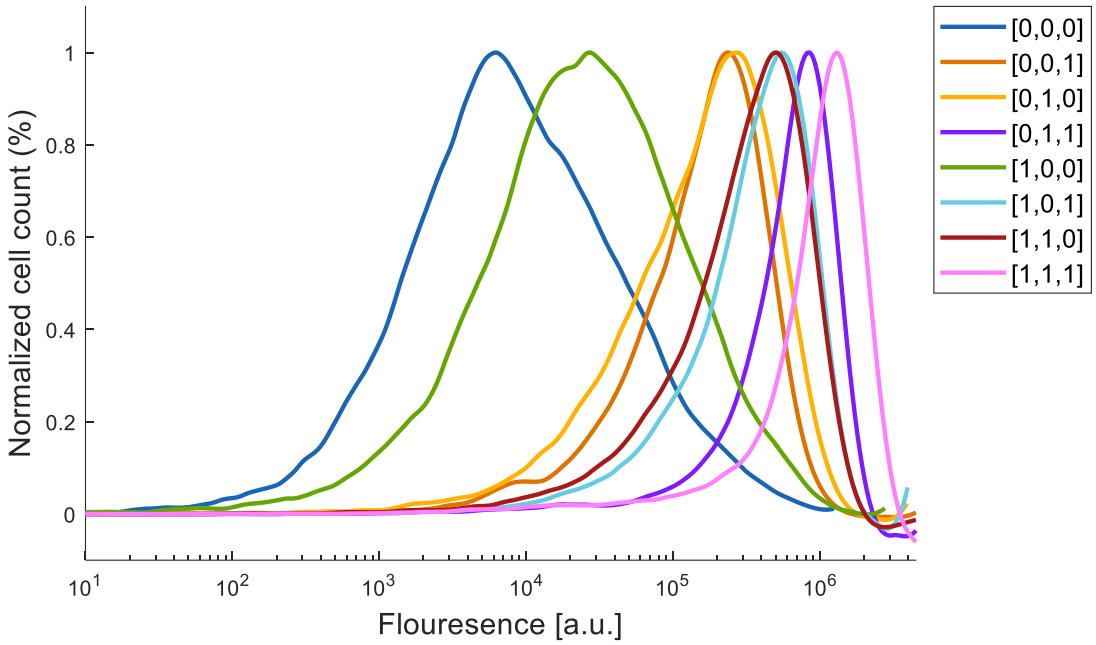


Fig. S14.91. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GTTG Mutation (Fig. 3G and Fig. S6.5D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.25mM].

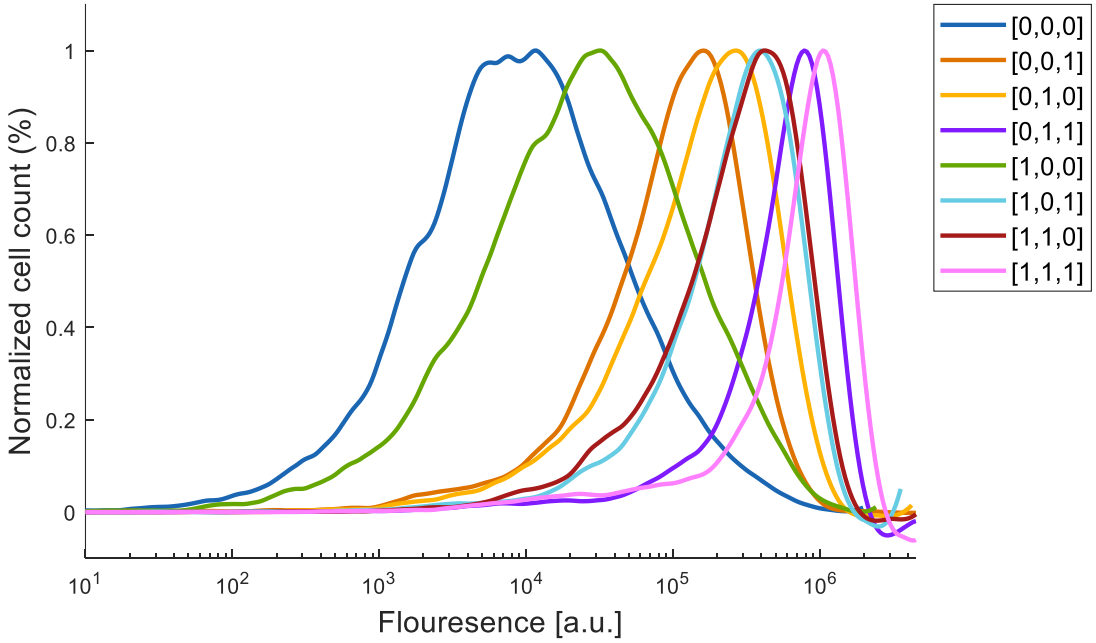


Fig. S14.92. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GTTG Mutation (Fig. 3G and Fig. S6.5D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.125mM].

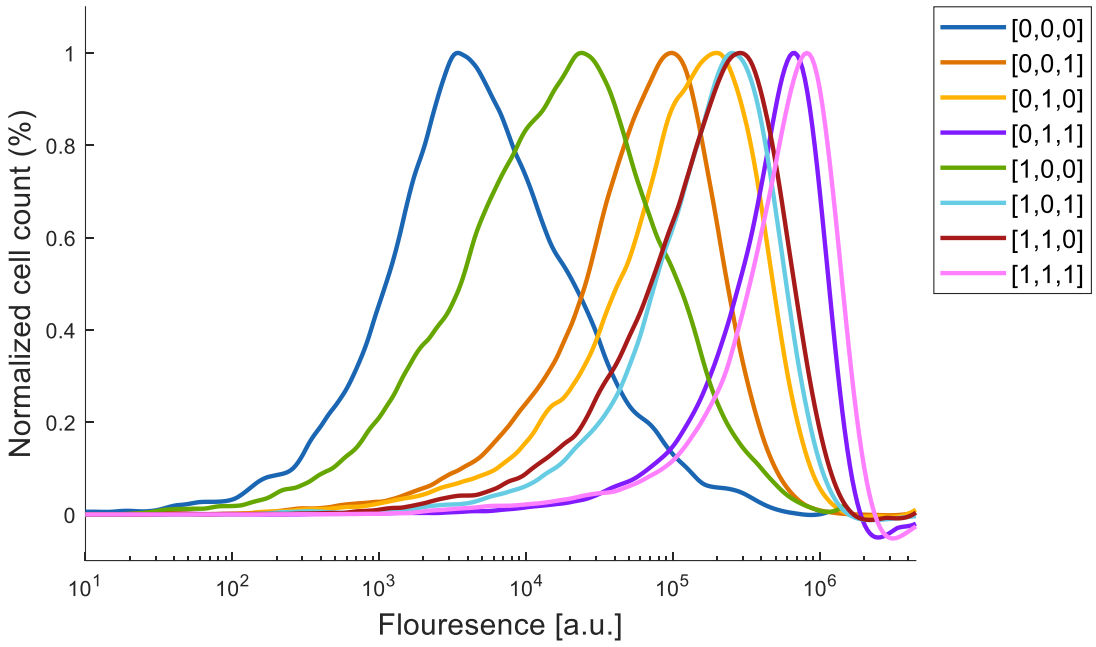


Fig. S14.93. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GTTG Mutation (Fig. 3G and Fig. S6.5D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0625mM].

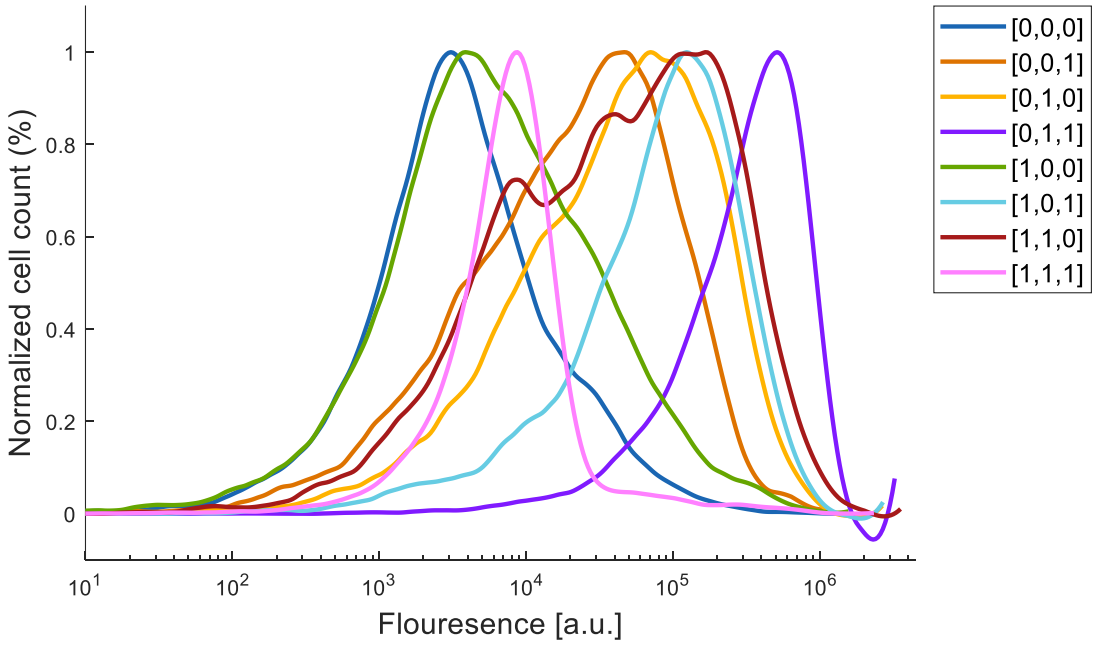


Fig. S14.94. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GTTG Mutation (Fig. 3G and Fig. S6.5D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125 mM].

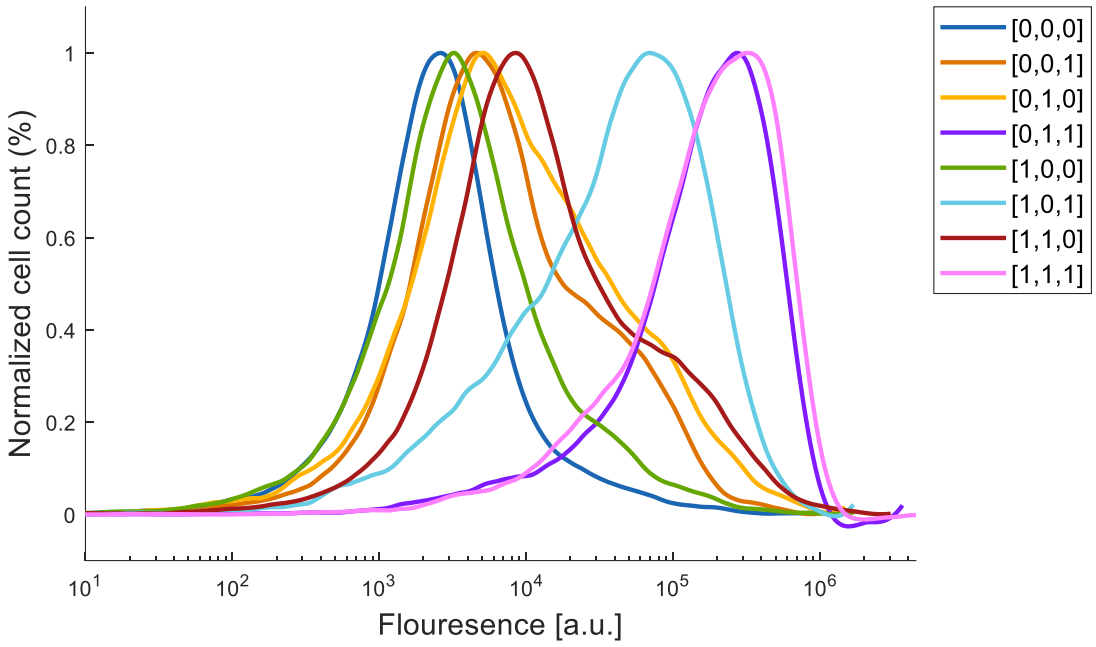


Fig. S14.95. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GTTG Mutation (Fig. 3G and Fig. S6.5D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.015625 mM].

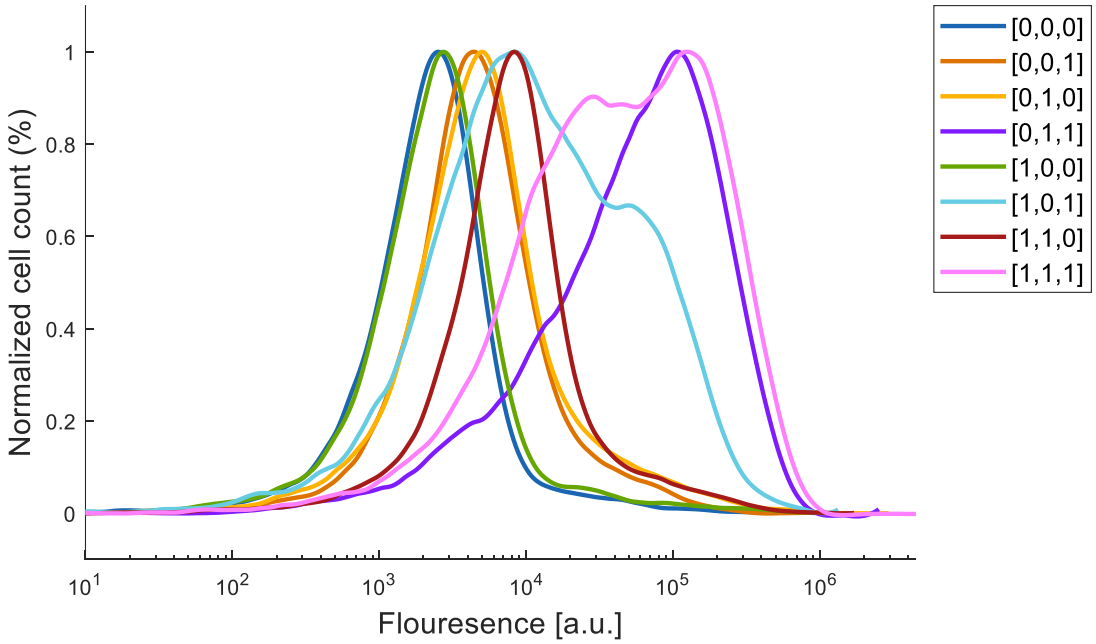


Fig. S14.96. GFP flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GTTG Mutation (Fig. 3G and Fig. S6.5D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0078125mM].

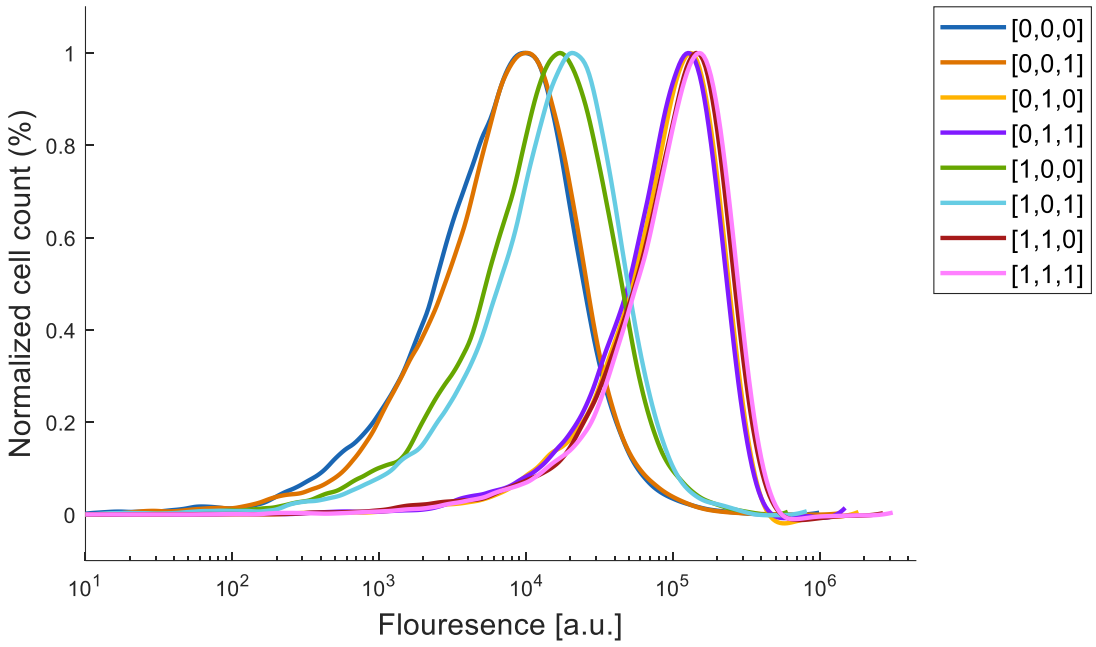


Fig. S14.97. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GTTG Mutation (Fig. S6.7D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.25mM].

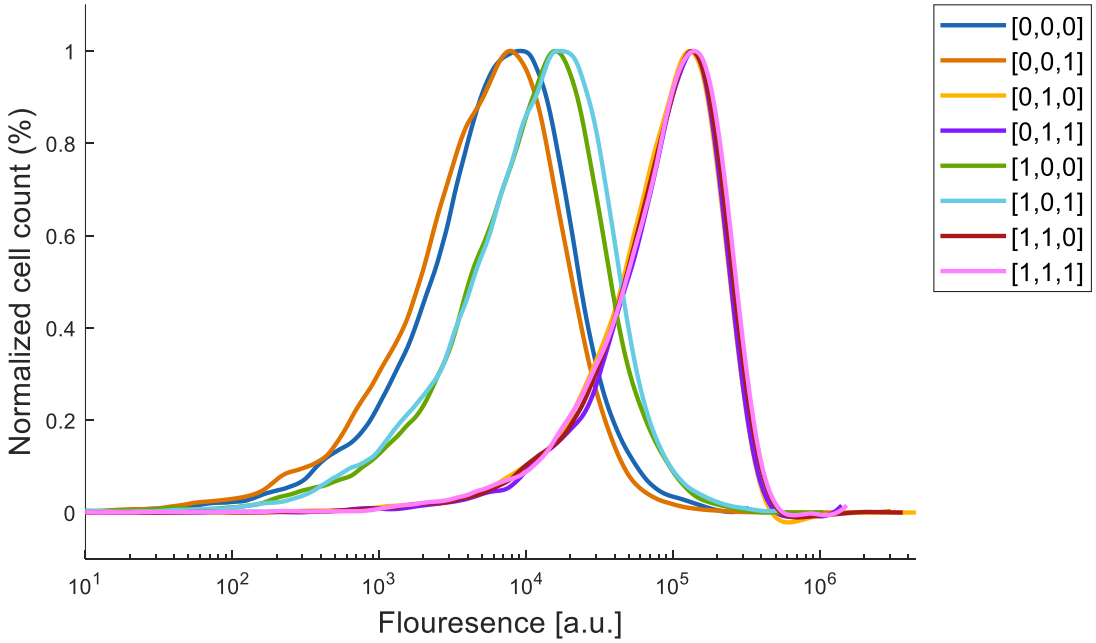


Fig. S14.98. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GTTG Mutation (Fig. S6.7D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.125mM].

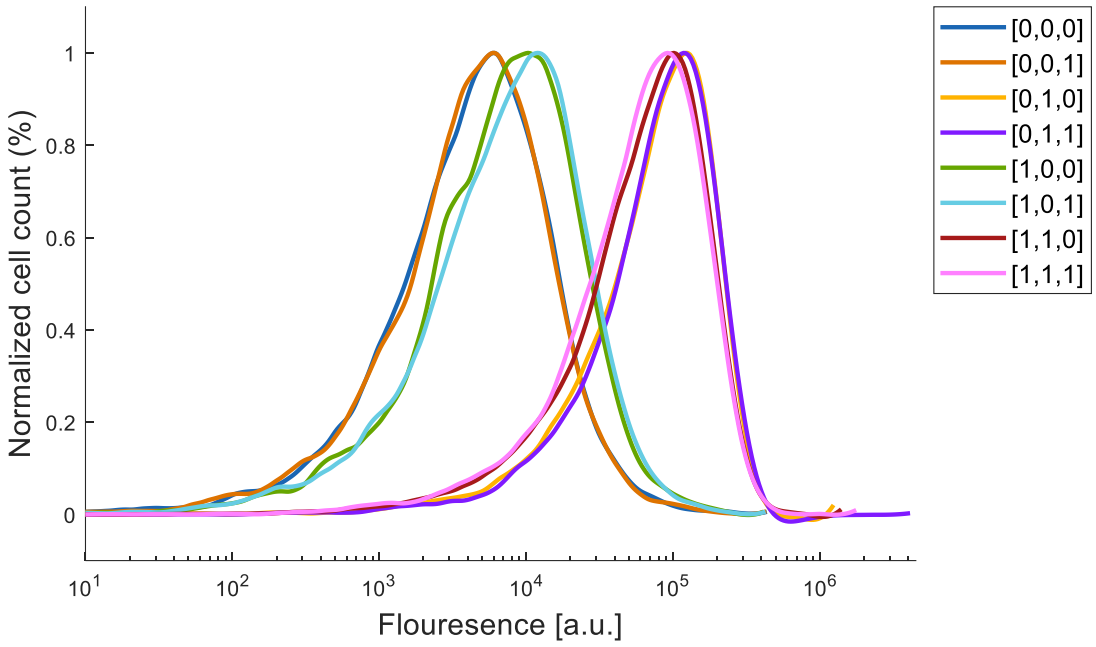


Fig. S14.99. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GTTG Mutation (Fig. S6.7D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0625mM].

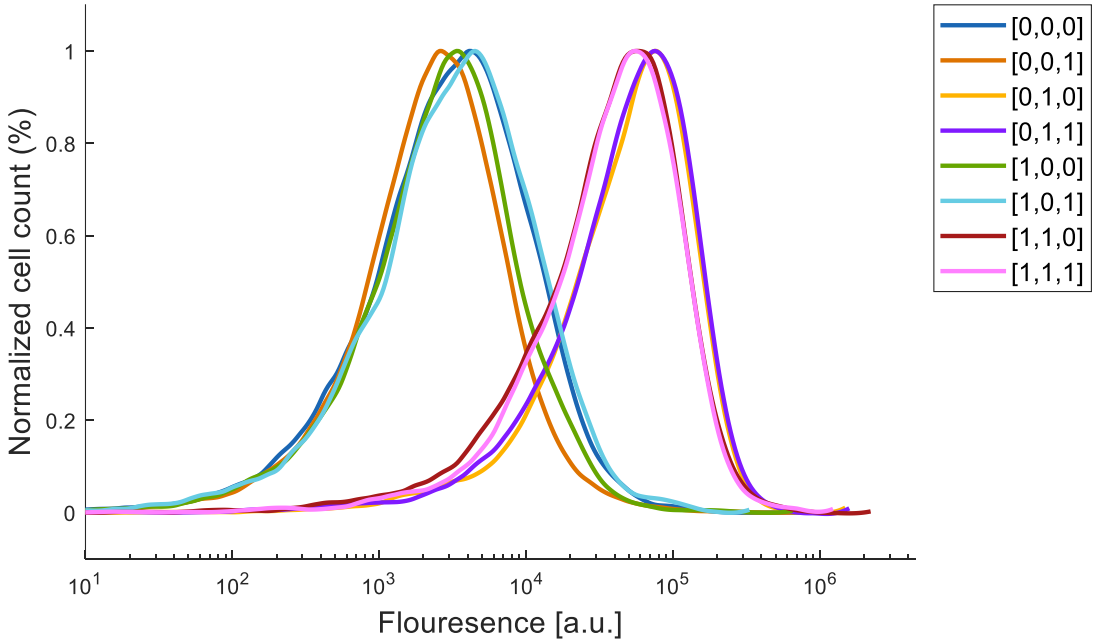


Fig. S14.100. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GTTG Mutation (Fig. S6.7D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125mM].

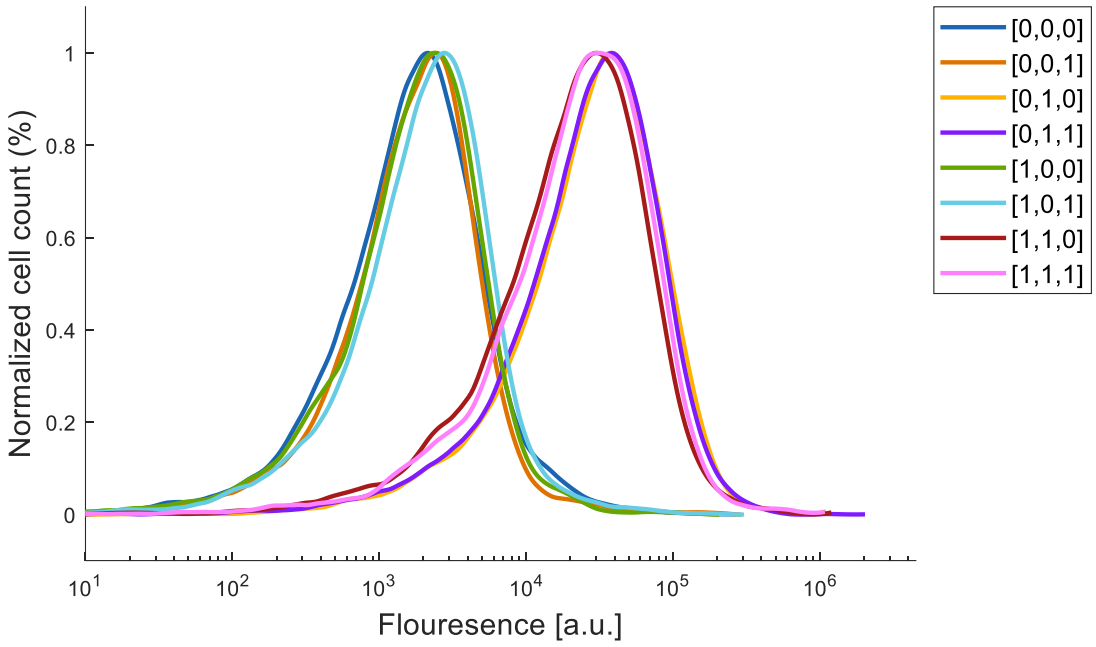


Fig. S14.101. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GTTG Mutation (Fig. S6.7D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.015625mM].

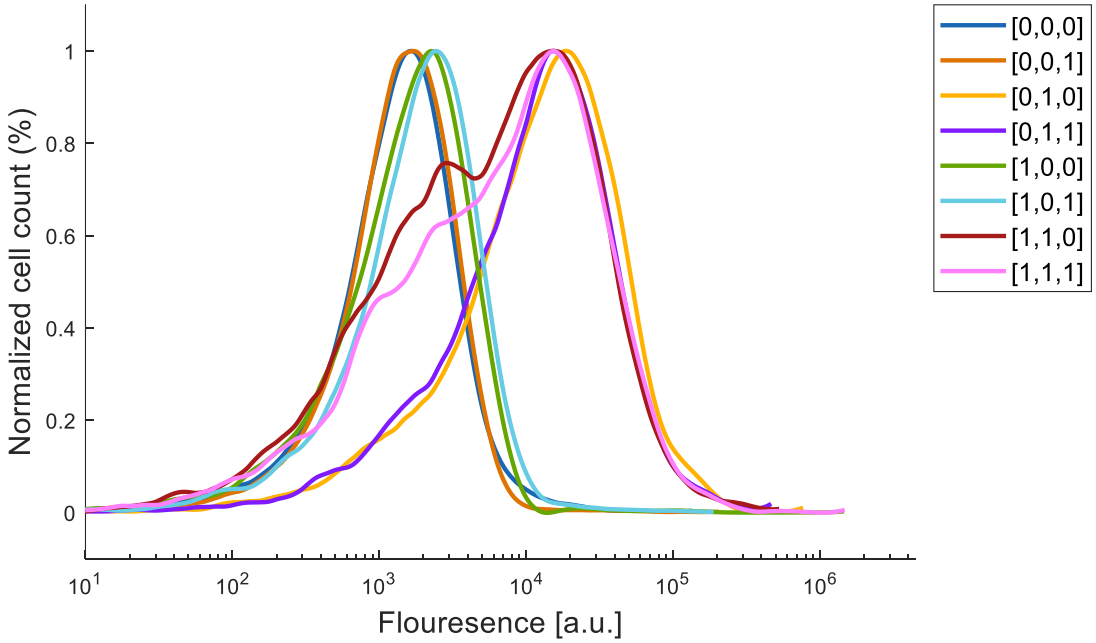


Fig. S14.102. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GTTG Mutation (Fig. S6.7D). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0078125].

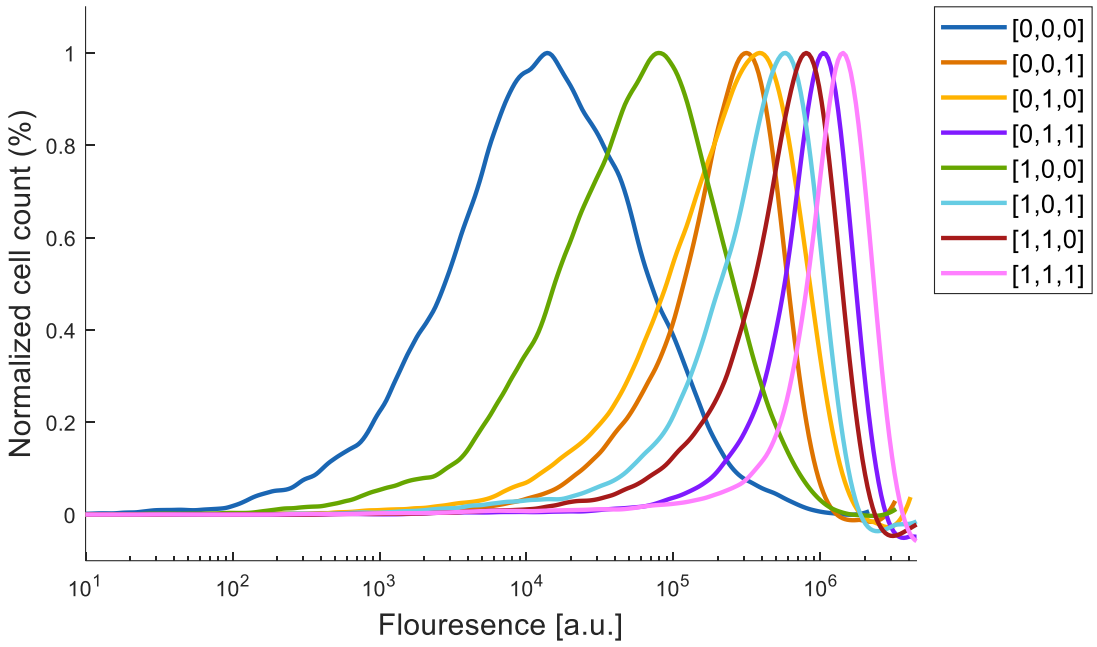


Fig. S14.103. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GAGC Mutation (Fig. 3G and Fig. S6.5F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.25mM].

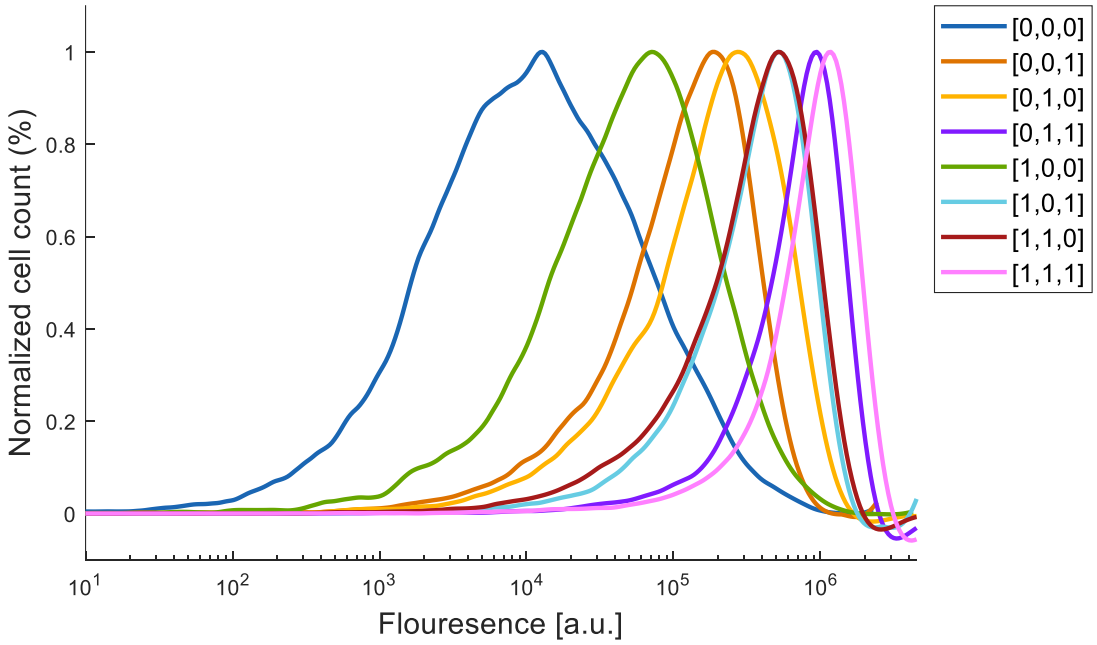


Fig. S14.104. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GAGC Mutation (Fig. 3G and Fig. S6.5F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.125mM].

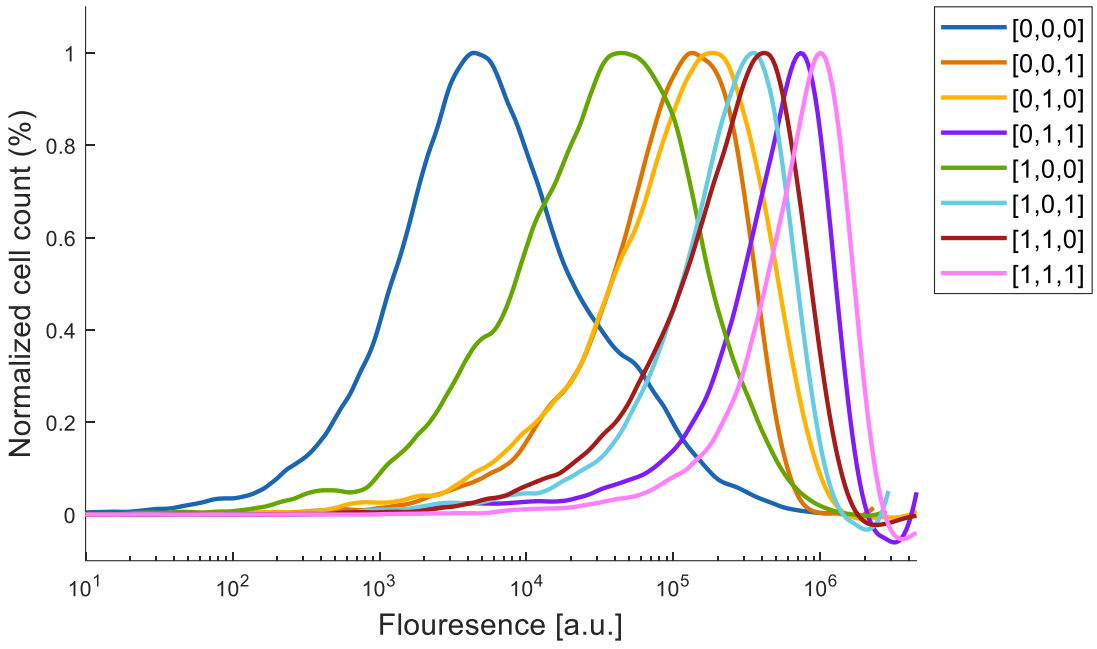


Fig. S14.105. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GAGC Mutation (Fig. 3G and Fig. S6.5F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0625mM].

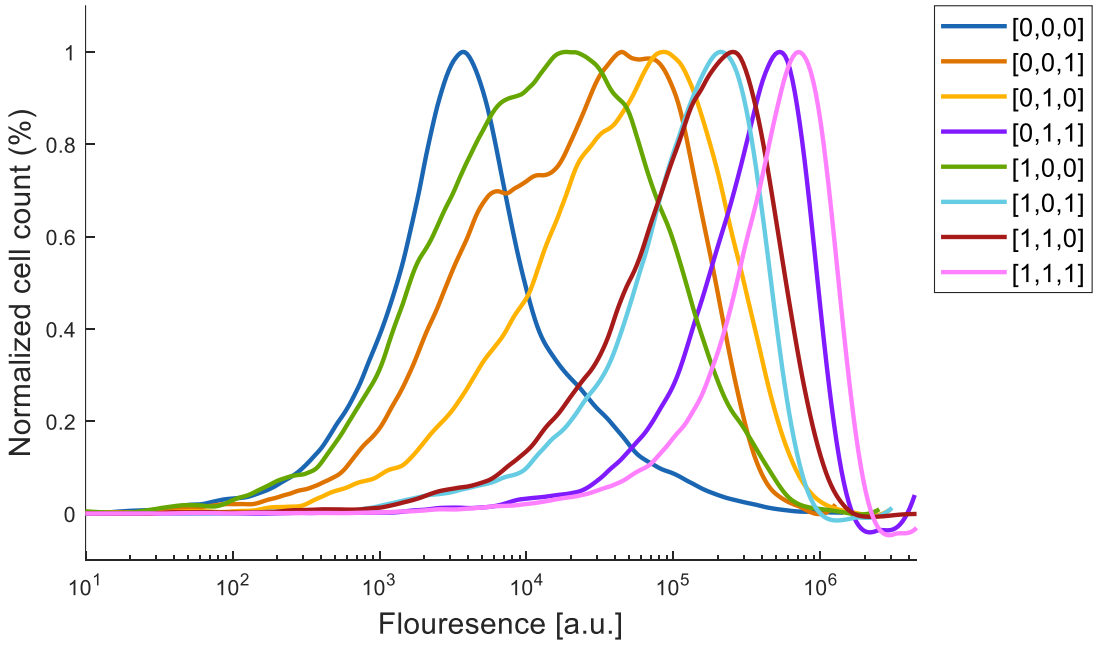


Fig. S14.106. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GAGC Mutation (Fig. 3G and Fig. S6.5F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125 mM].

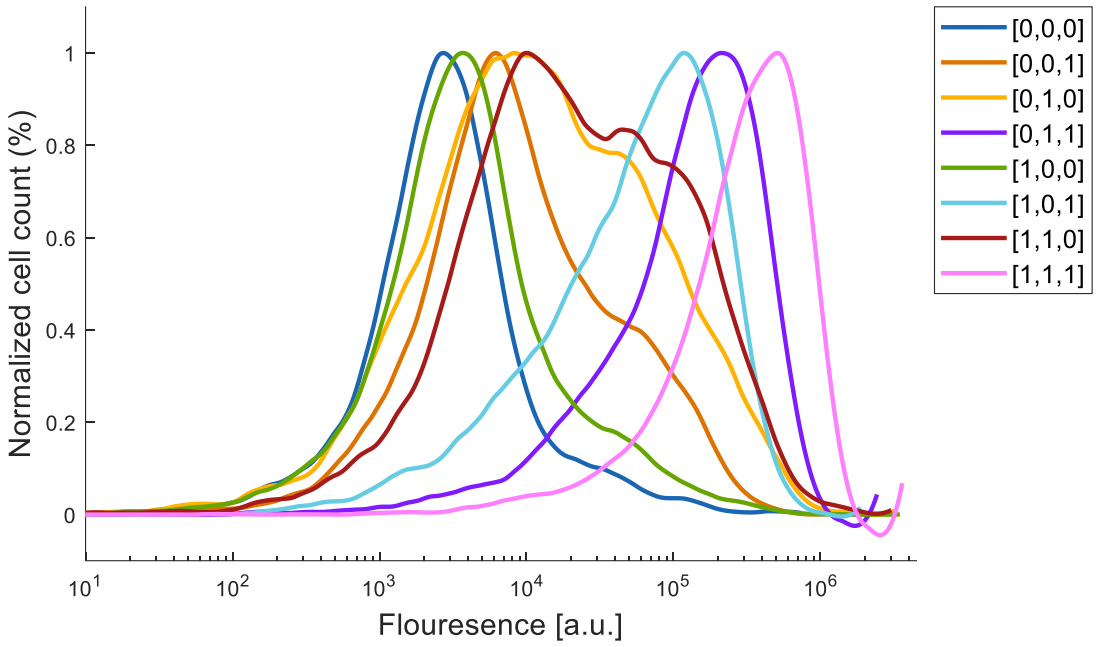


Fig. S14.107. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GAGC Mutation (Fig. 3G and Fig. S6.5F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.015625 mM].

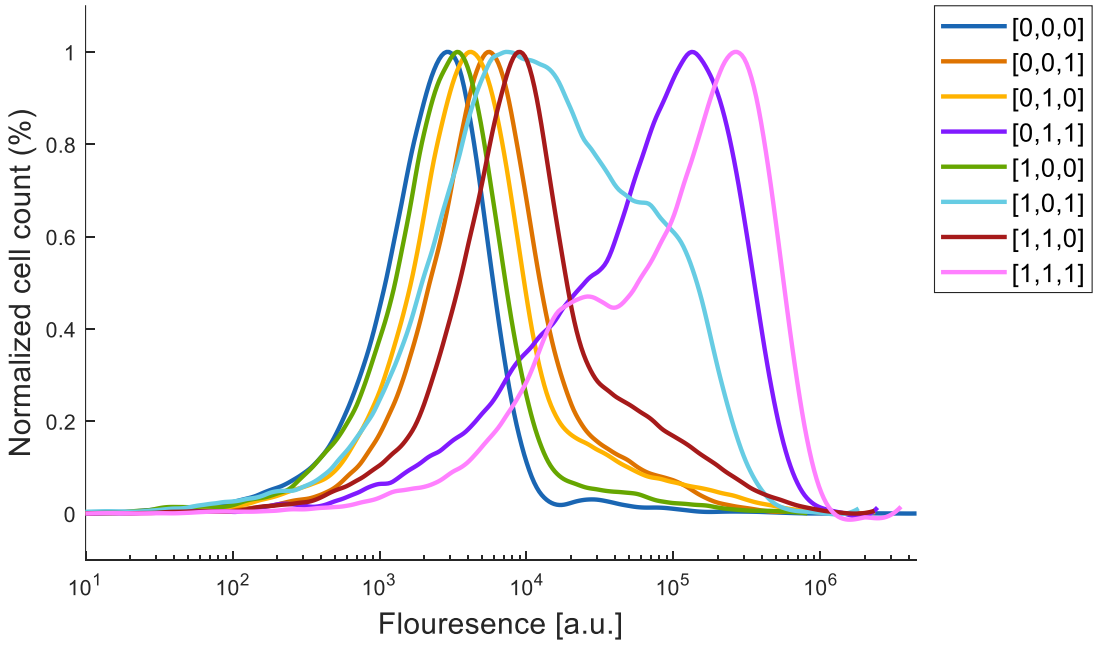


Fig. S14.108. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with GAGC Mutation (Fig. 3G and Fig. S6.5F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0078125mM].

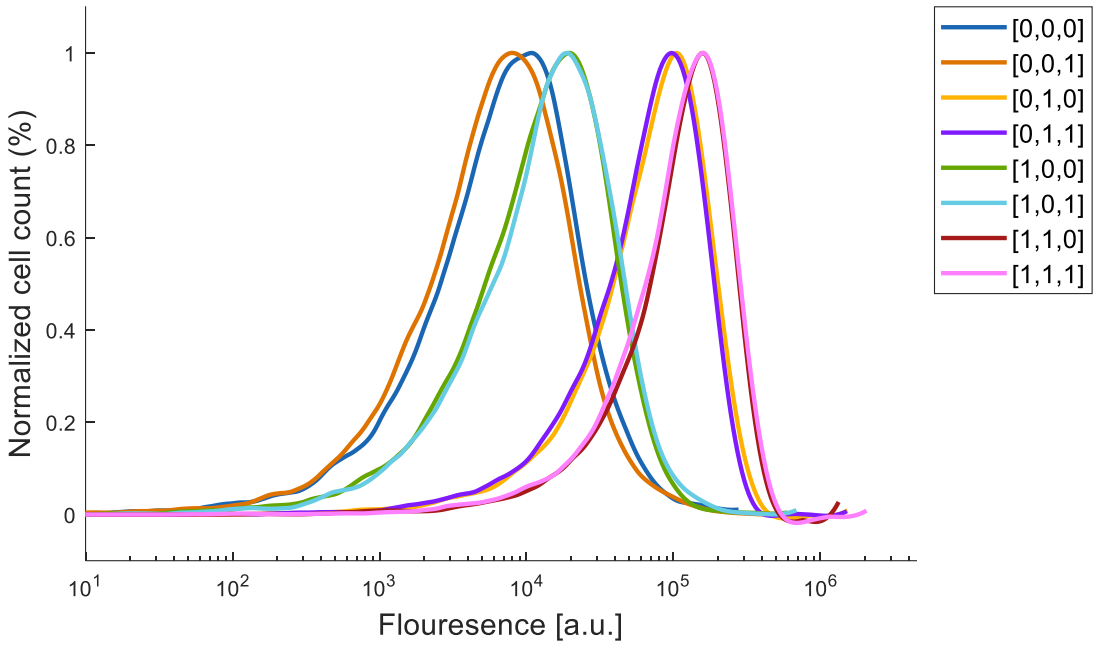


Fig. S14.109. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GAGC Mutation (Fig. S6.7F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.25mM].

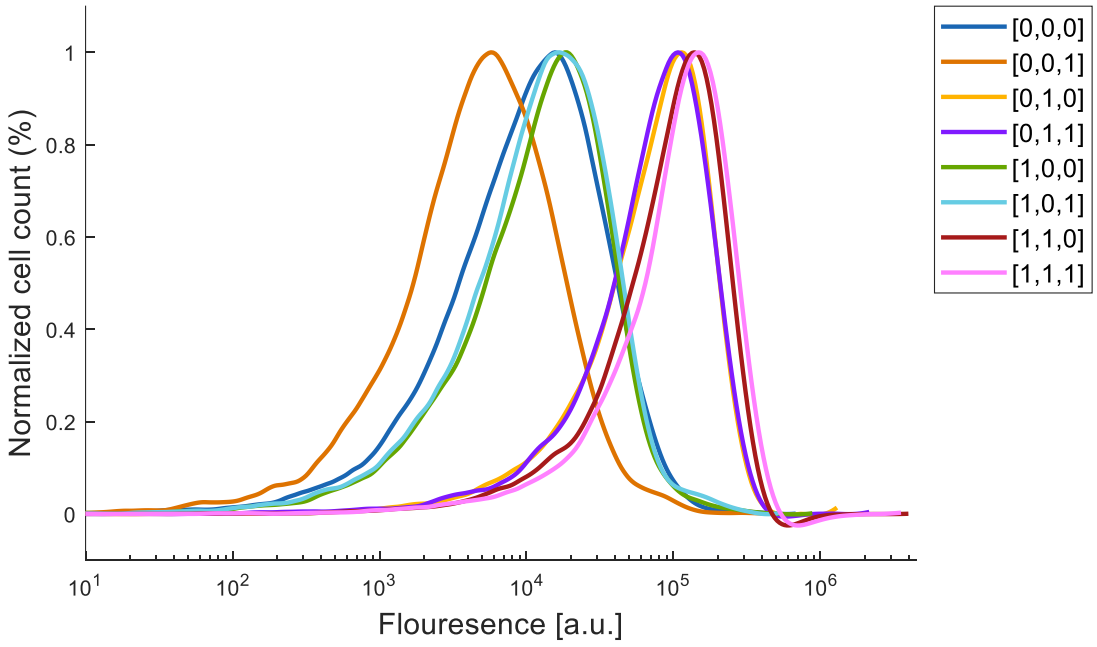


Fig. S14.110. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GAGC Mutation (Fig. S6.7F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.125mM].

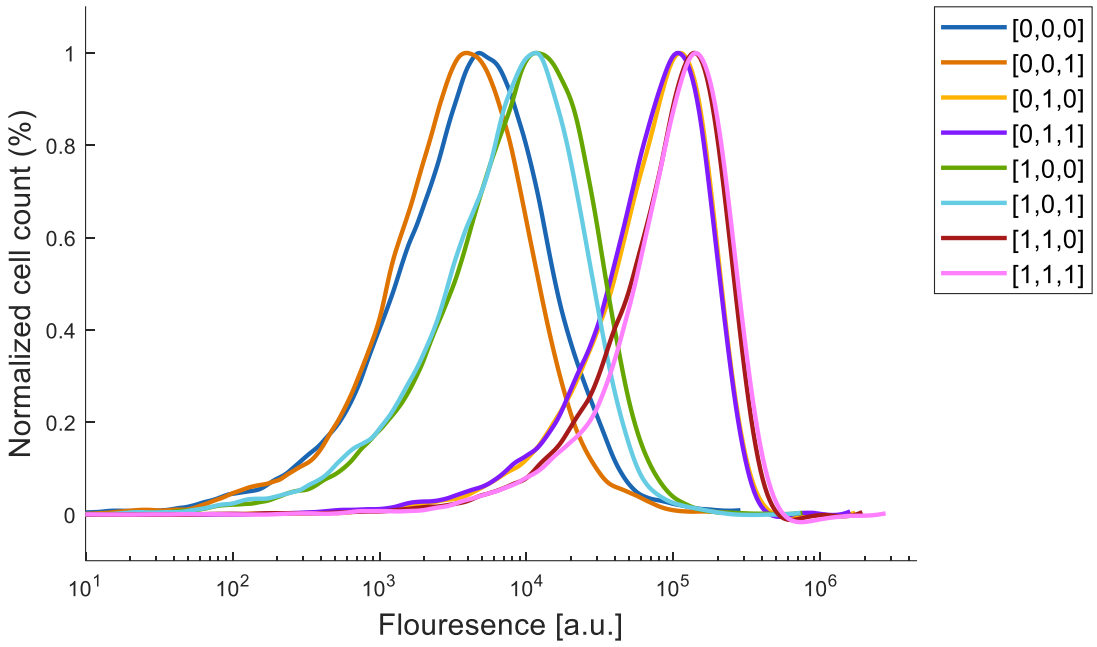


Fig. S14.111. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GAGC Mutation (Fig. S6.7F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0625mM].

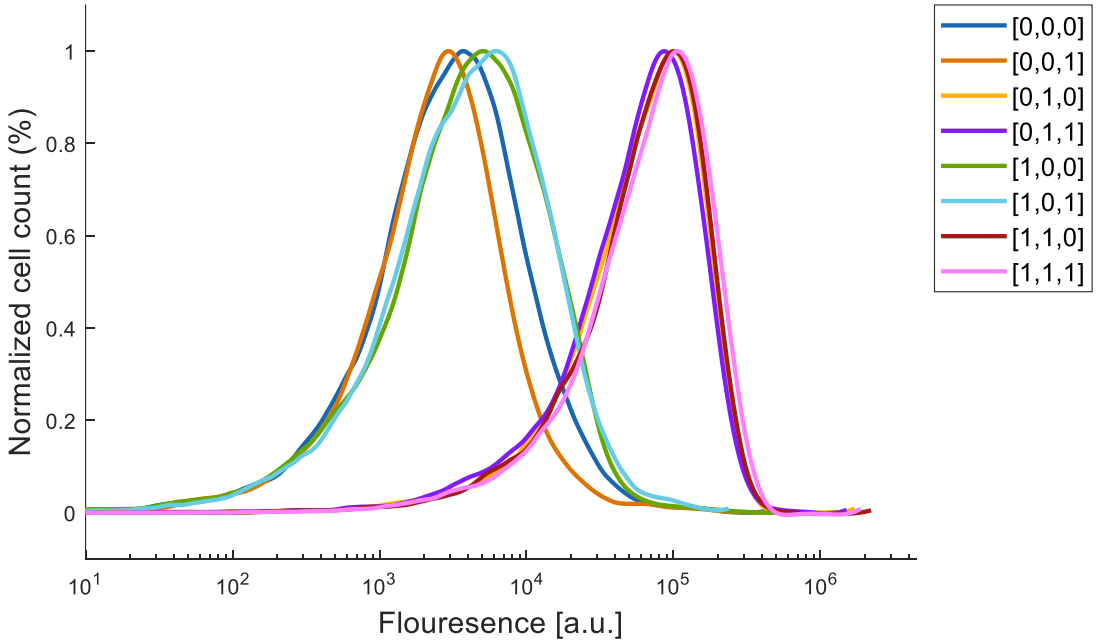


Fig. S14.112. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GAGC Mutation (Fig. S6.7F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125mM].

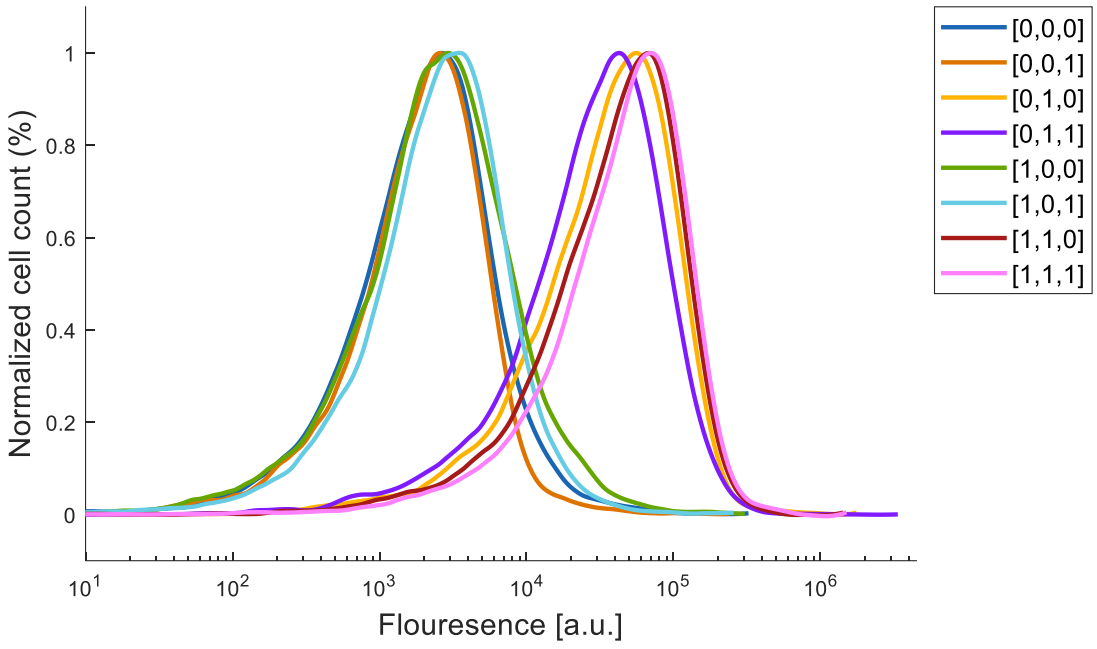


Fig. S14.113. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GAGC Mutation (Fig. S6.7F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.015625mM].

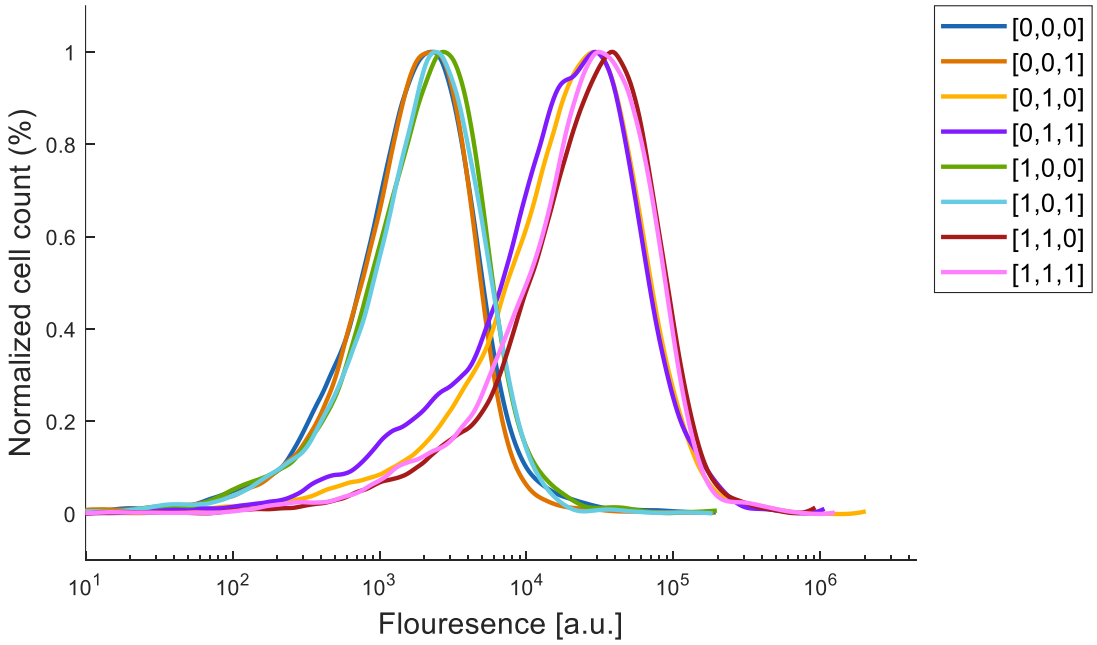


Fig. S14.1114. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with GAGC Mutation (Fig. S6.7F). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0078125].

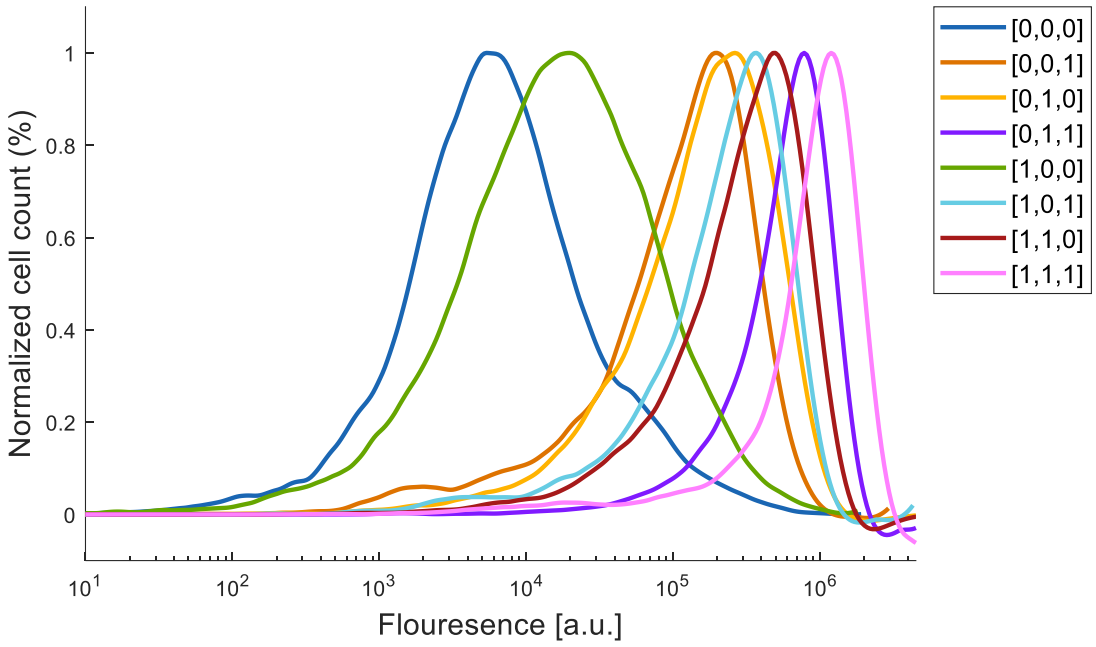


Fig. S14.115. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TGGG Mutation (Fig. 3G and Fig. S6.5H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.25mM].

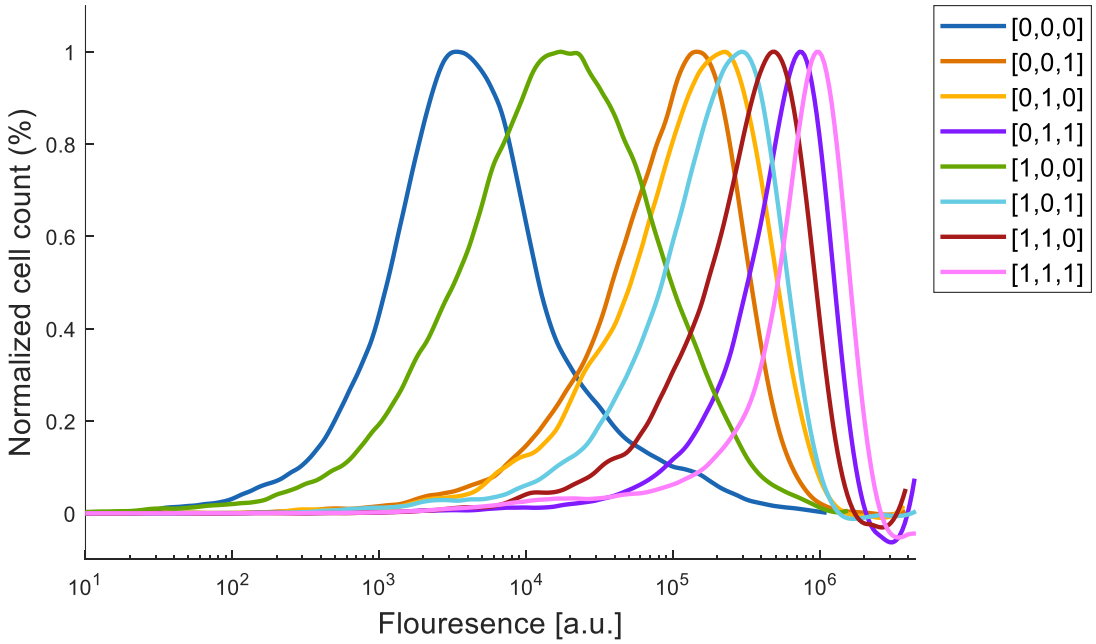


Fig. S14.116. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TGGG Mutation (Fig. 3G and Fig. S6.5H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.125mM].

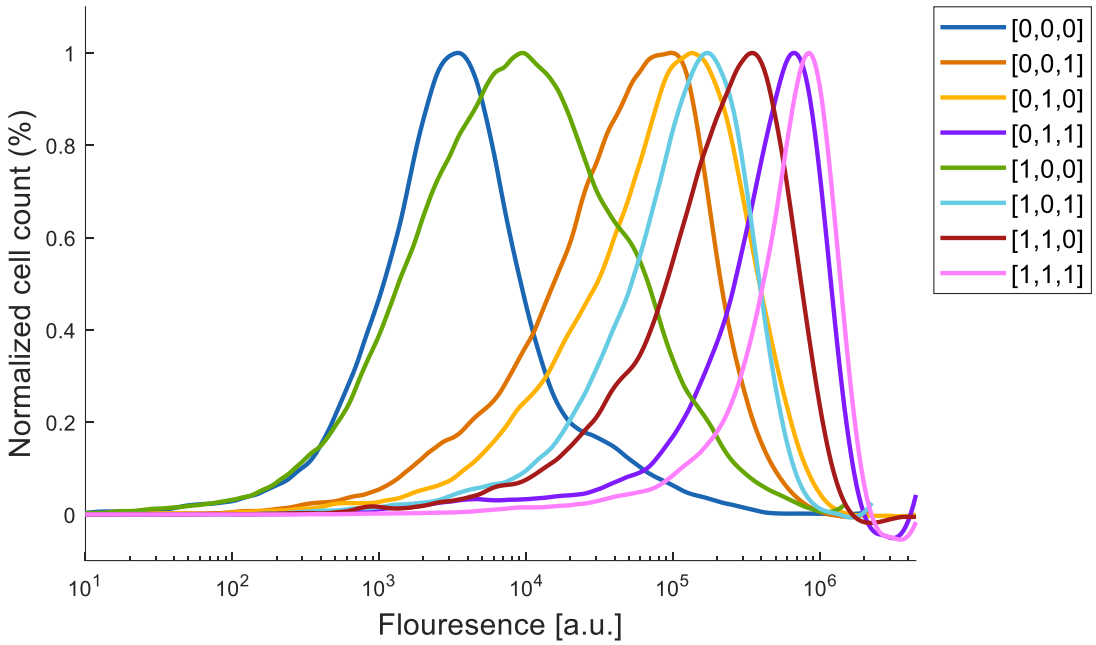


Fig. S14.117. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TGGG Mutation (Fig. 3G and Fig. S6.5H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0625mM].

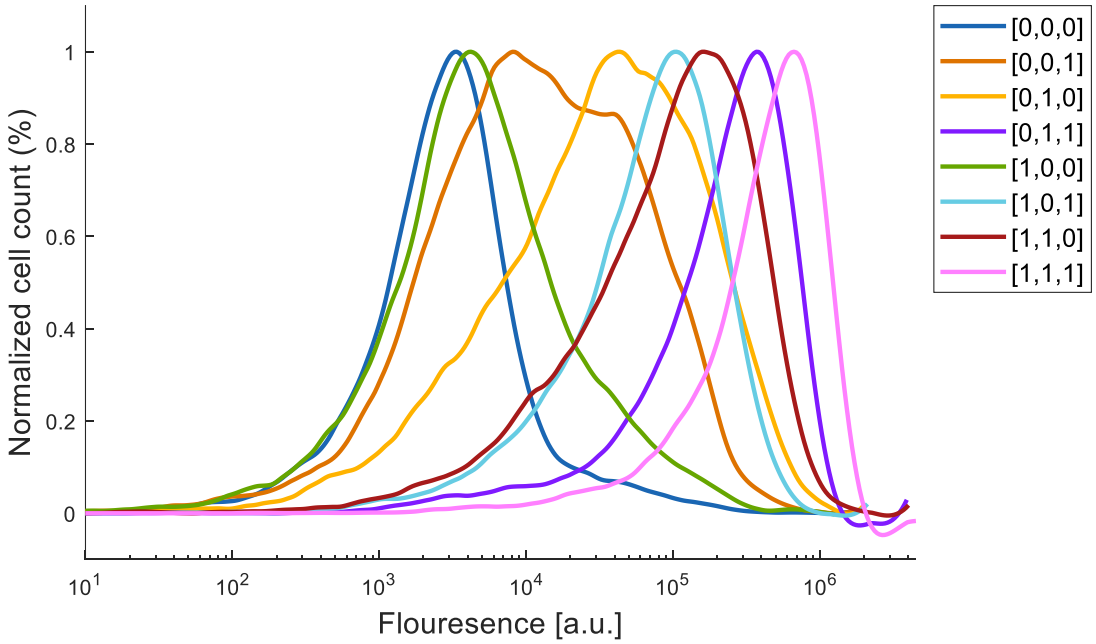


Fig. S14.118. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TGGG Mutation (Fig. 3G and Fig. S6.5H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125 mM].

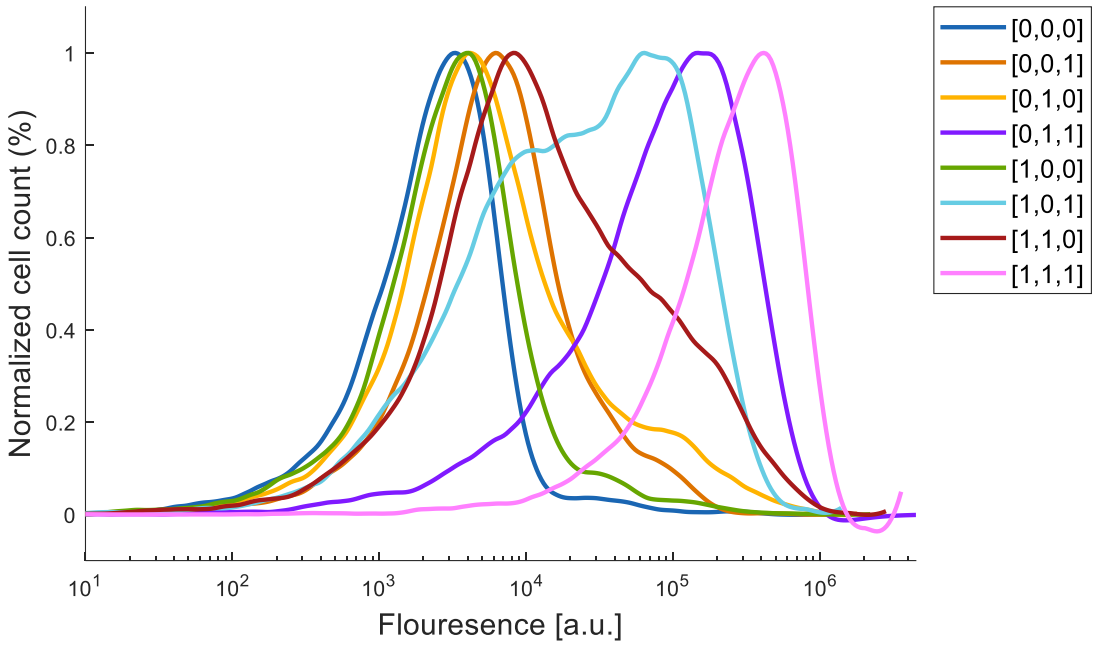


Fig. S14.119. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TGGG Mutation (Fig. 3G and Fig. S6.5H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.015625 mM].

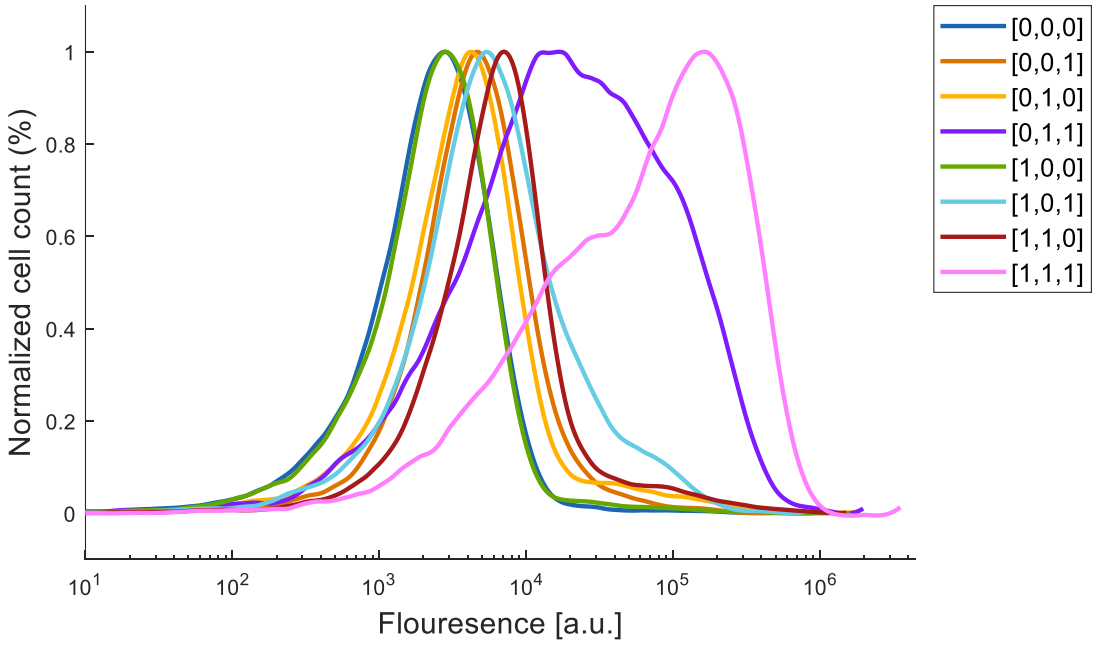


Fig. S14.120. GFP flow cytometry data for a population of cells containing flow cytometry data for a population of cells containing a 3-input perceptgene network and back propagation algorithm with TGGG Mutation (Fig. 3G and Fig. S6.5H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0078125mM].

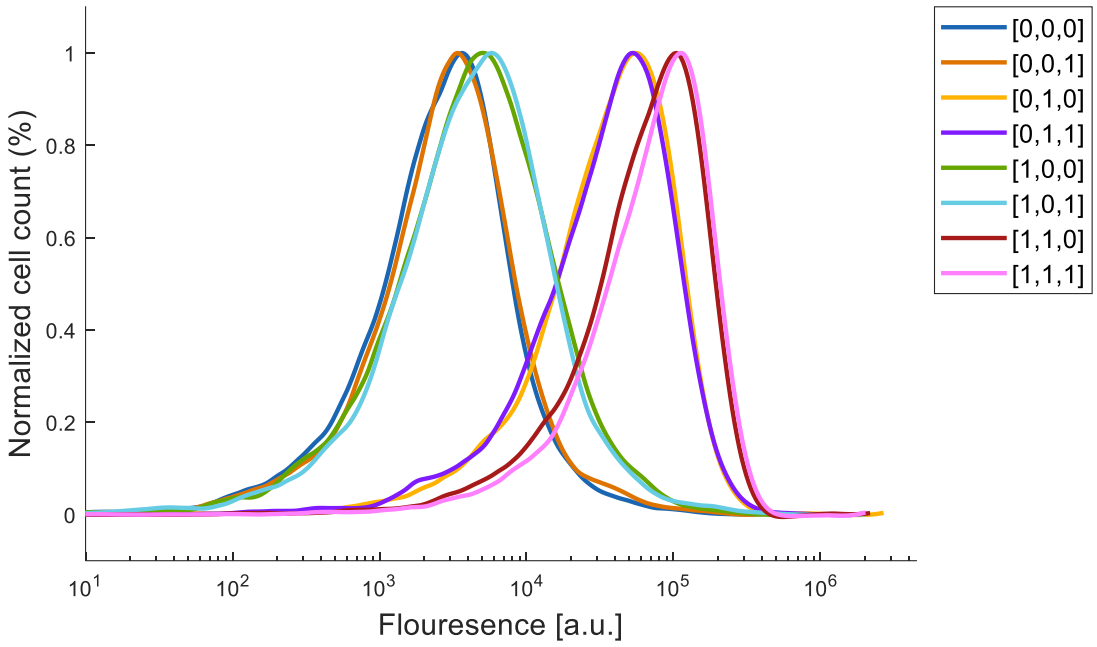


Fig. S14.121. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TGGG Mutation (Fig. S6.7H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.25mM].

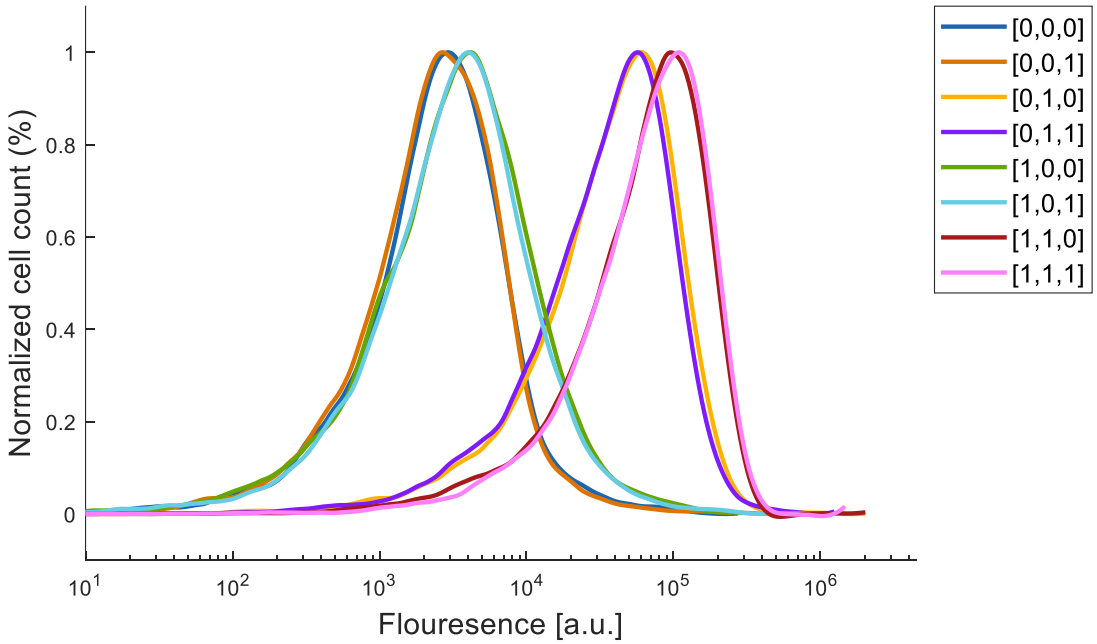


Fig. S14.122. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TGGG Mutation (Fig. S6.7H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.125mM].

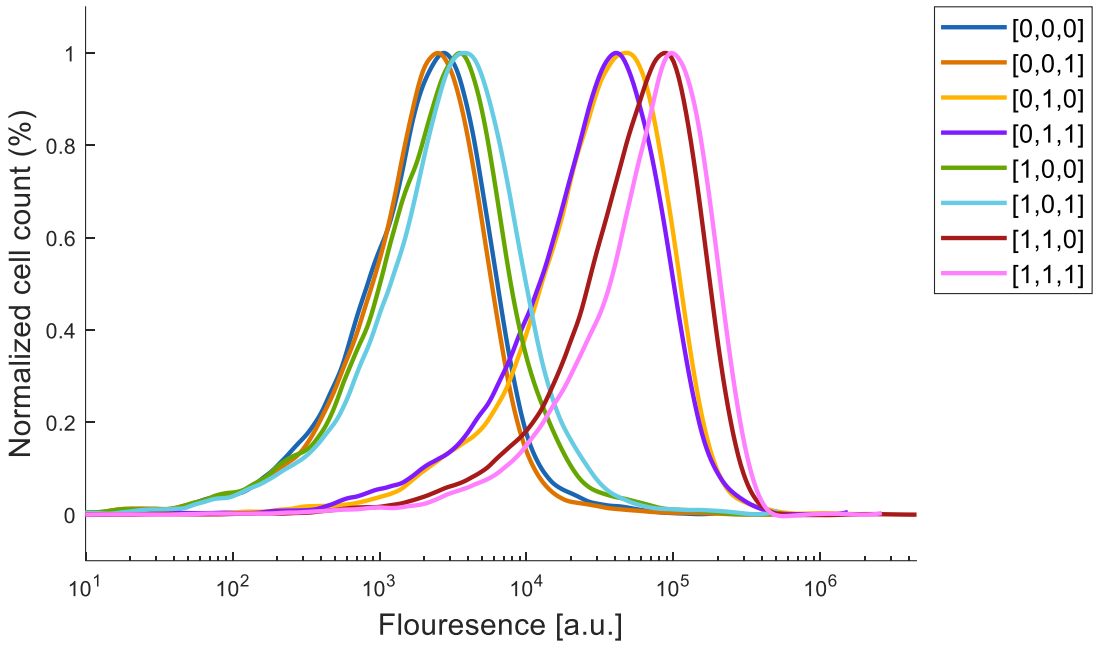


Fig. S14.123. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TGGG Mutation (Fig. S6.7H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0625mM].

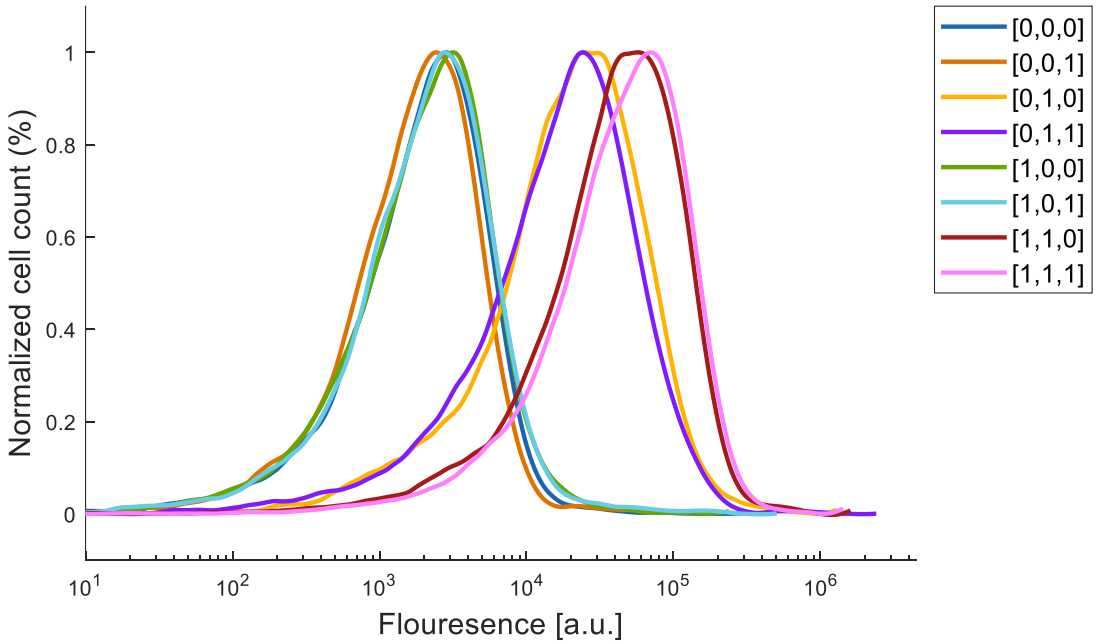


Fig. S14.124. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TGGG Mutation (Fig. S6.7H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.03125mM].

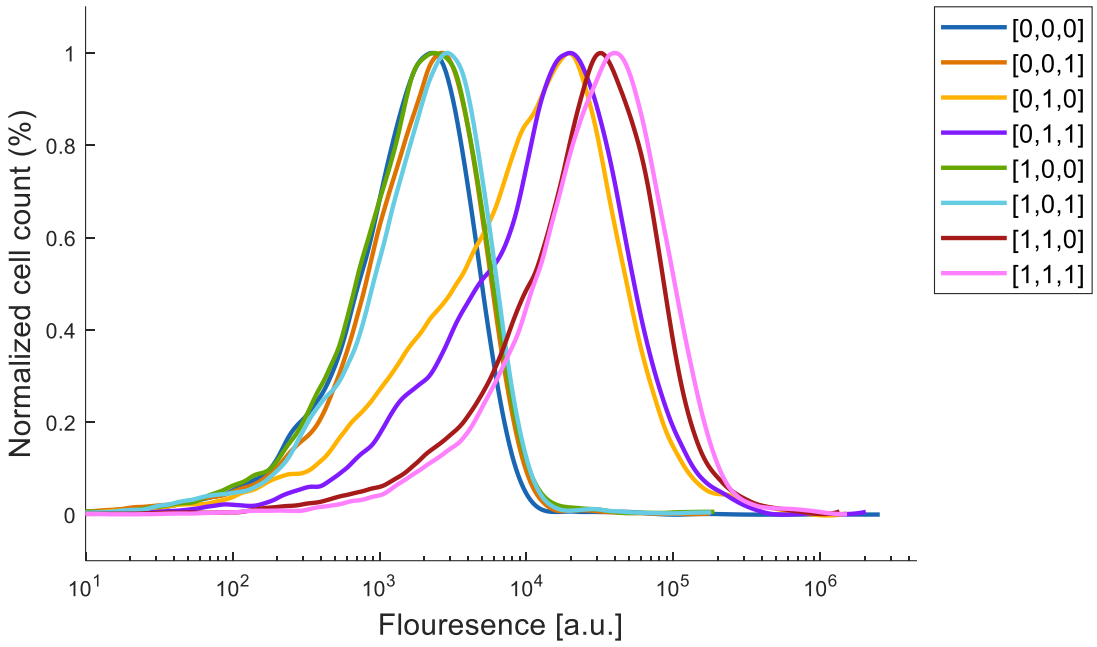


Fig. S14.125. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TGGG Mutation (Fig. S6.7H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.015625mM].

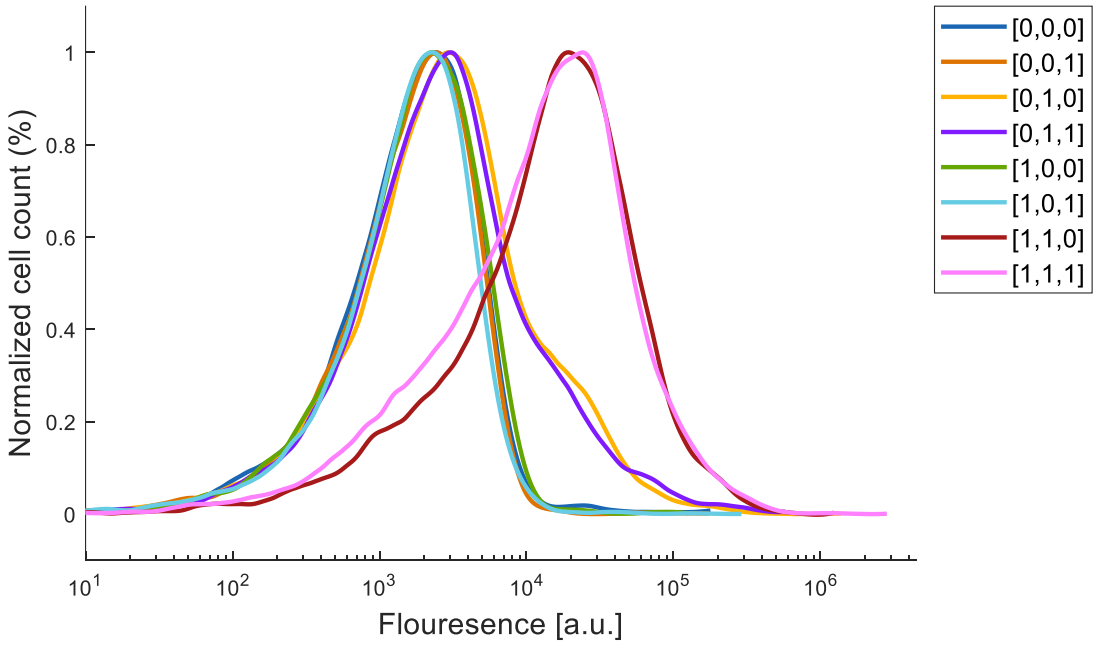


Fig. S14.126. GFP flow cytometry data for a population of cells containing the first perceptgene layer with 2-input from the 3-input network with TGGG Mutation (Fig. S6.7H). AHL [0.1875, 0.3μM], IPTG [7.8125, 125μM], aTc [1.5625, 25ng/mL] and Arabinose [0.0078125].

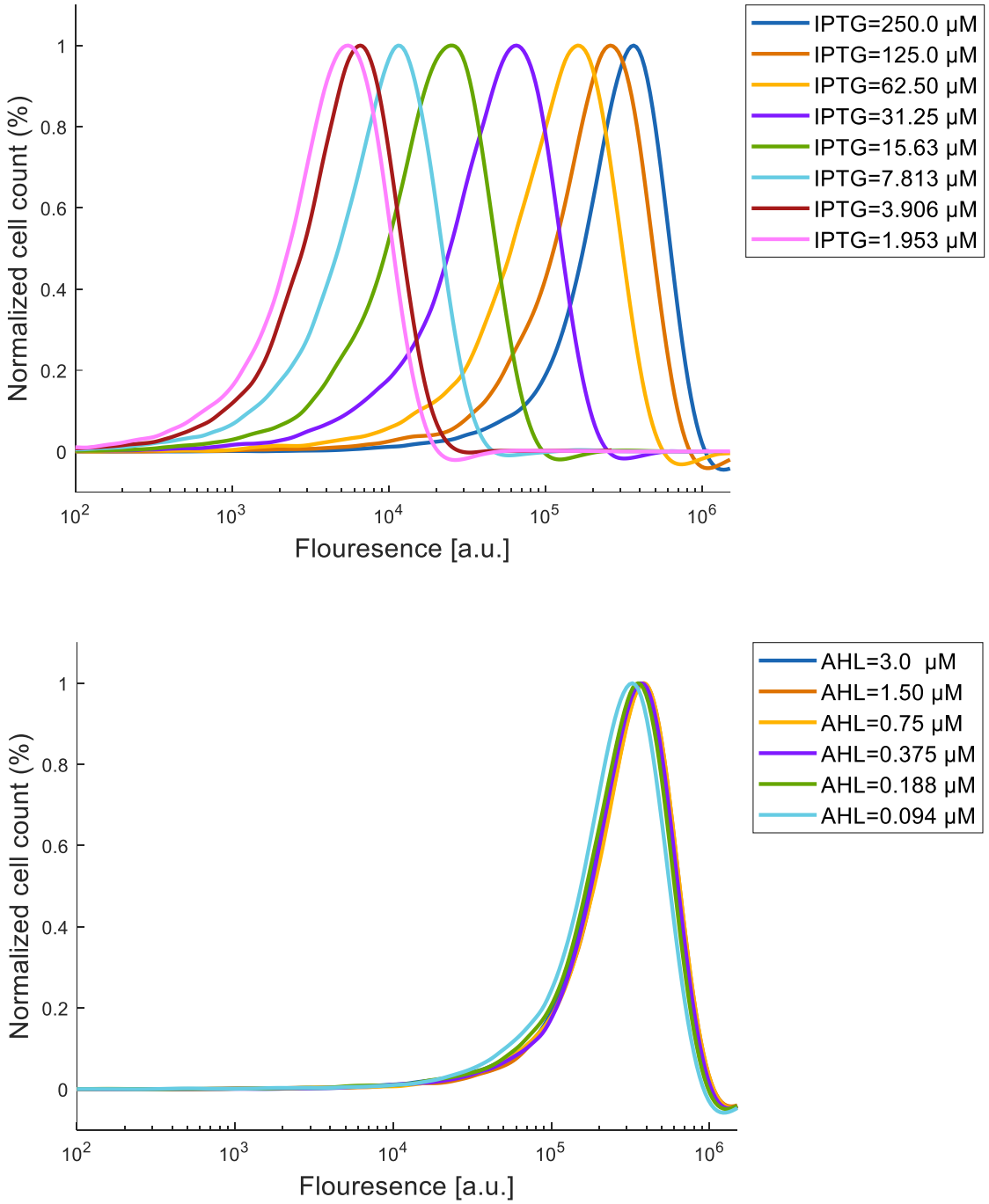


Fig. S14.127. GFP flow cytometry data for a population of cells containing APF ($P_{luxTCTA}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4C). (A) AHL was held constant at 3.0 μM and IPTG was varied. (B) IPTG was held constant at 250 μM and AHL was varied.

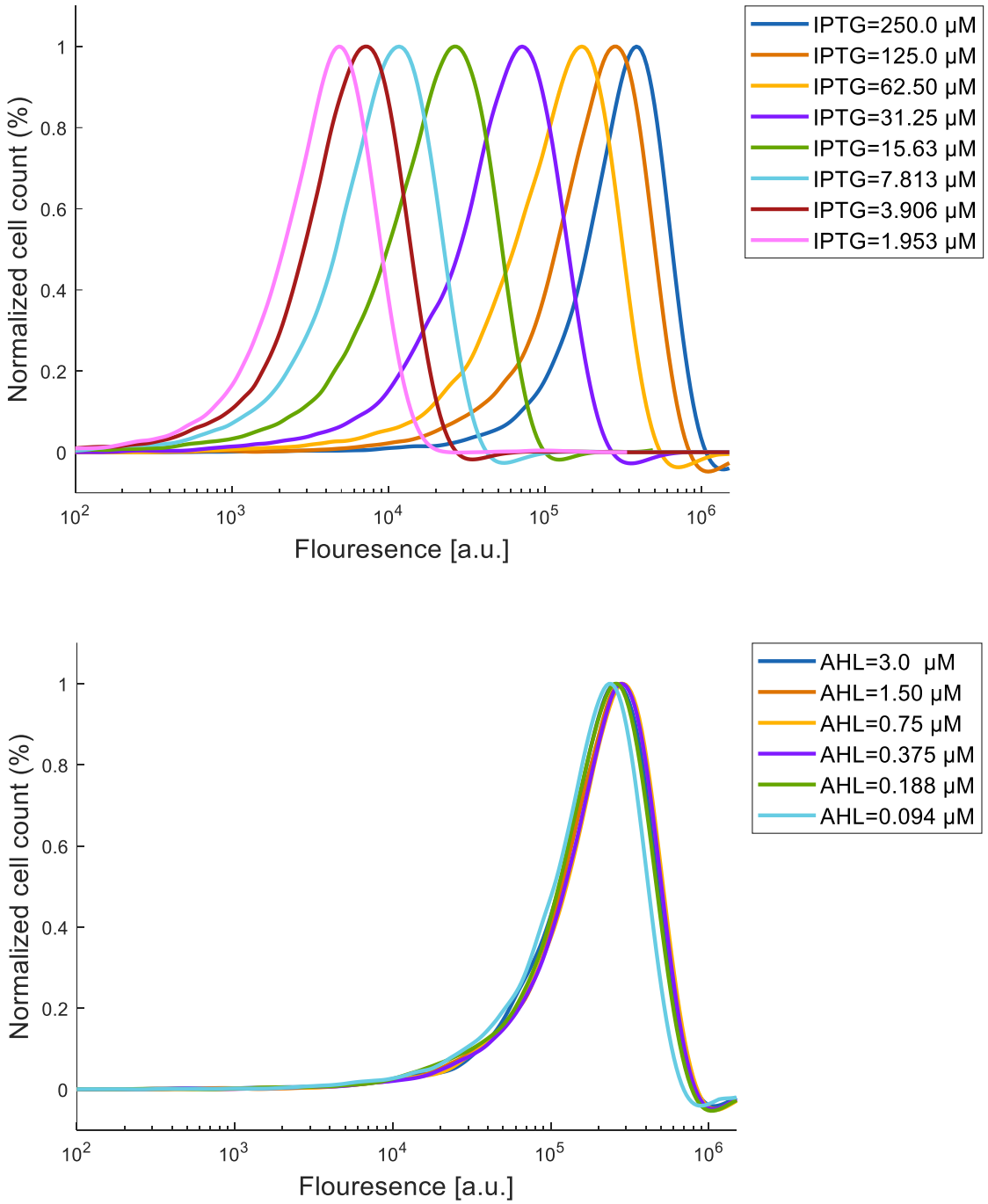


Fig. S14.128. GFP flow cytometry data for a population of cells containing APF ($P_{luxTCTA}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4C). (A) AHL was held constant at 1.5 μM and IPTG was varied. (B) IPTG was held constant at 125 μM and AHL was varied.

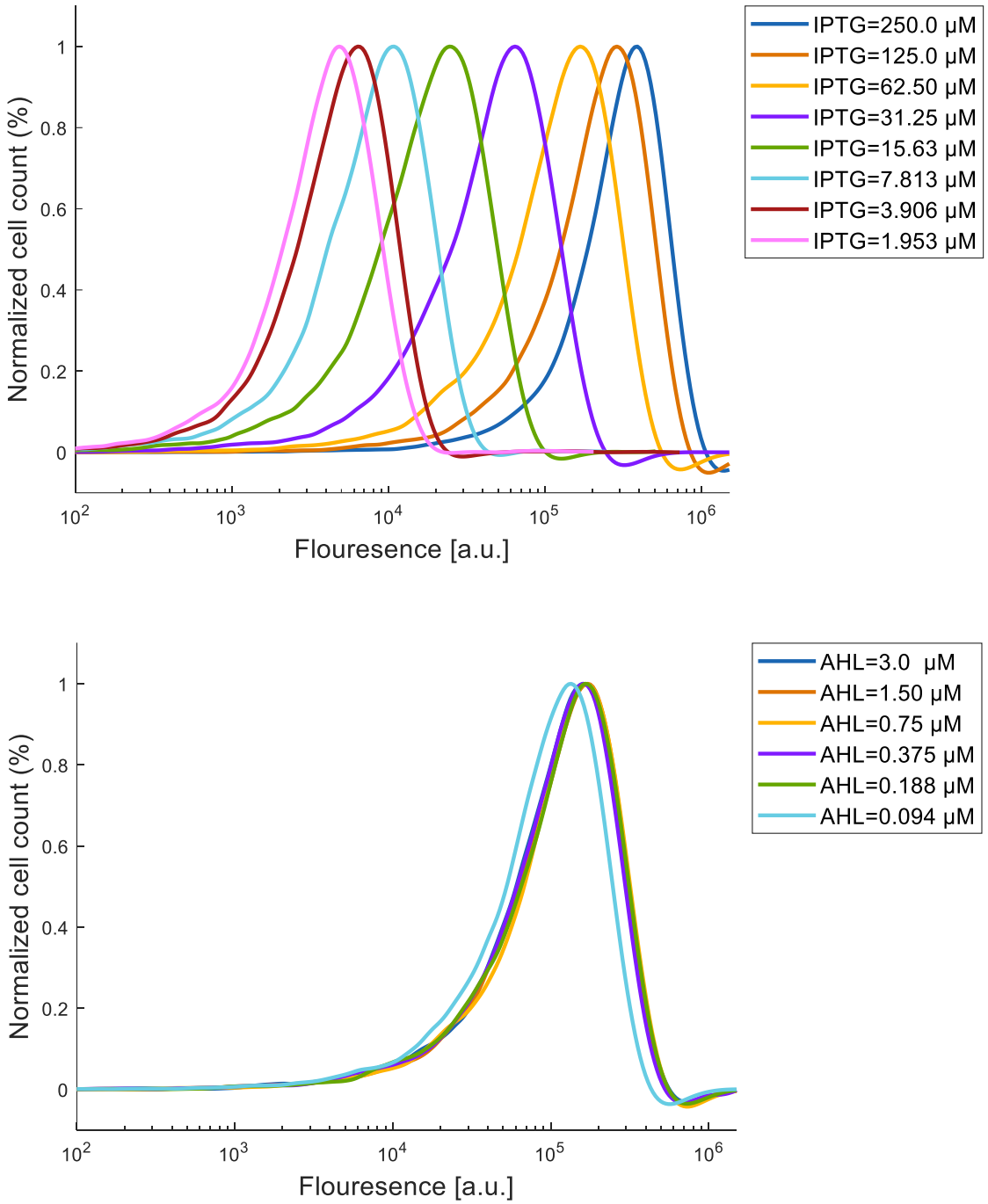


Fig. S14.129. GFP flow cytometry data for a population of cells containing APF ($P_{luxTCTA}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4C). (A) AHL was held constant at 0.75 μM and IPTG was varied. (B) IPTG was held constant at 62.5 μM and AHL was varied.

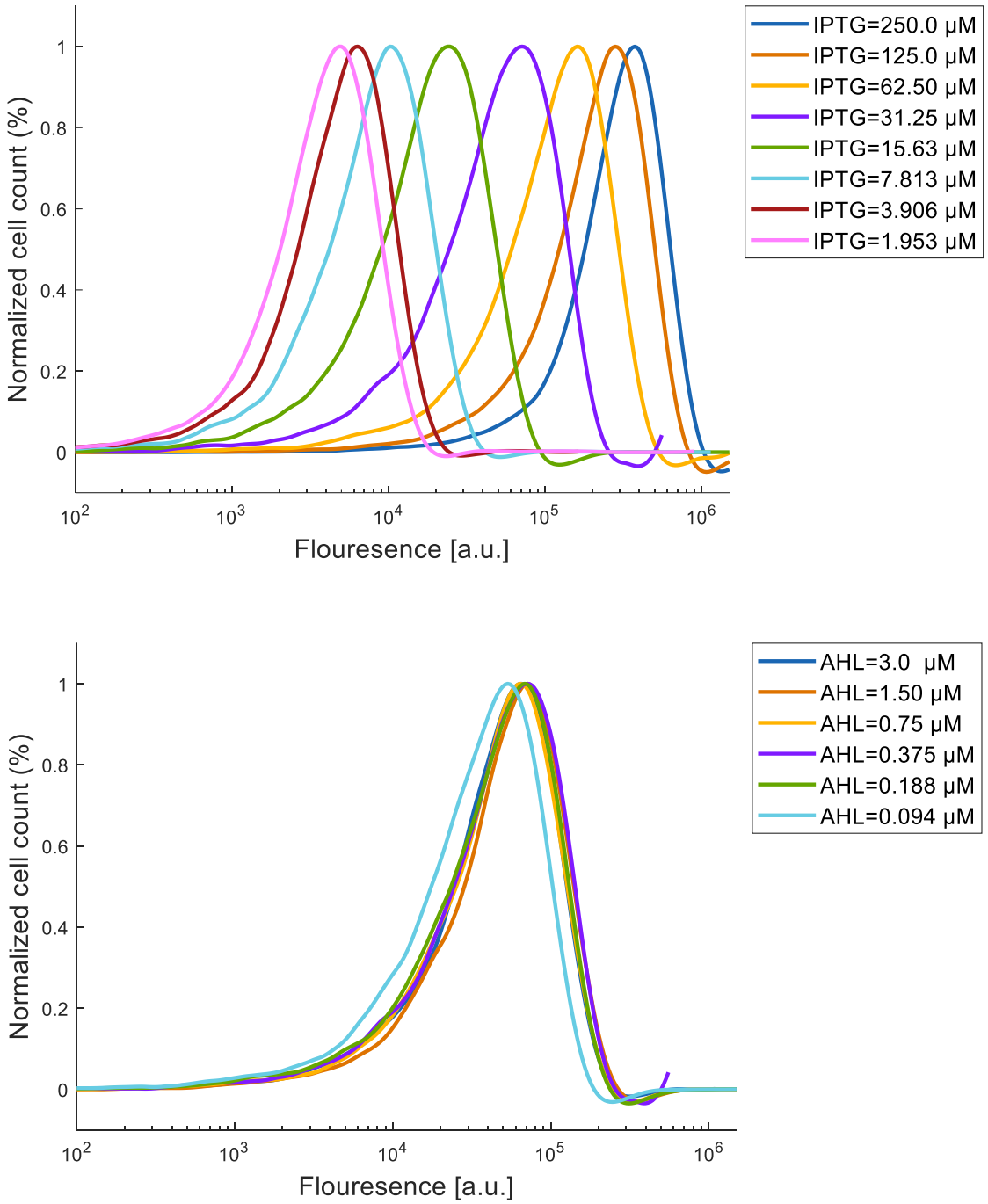


Fig. S14.130. GFP flow cytometry data for a population of cells containing APF ($P_{luxTCTA}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4C). (A) AHL was held constant at 0.375 μM and IPTG was varied. (B) IPTG was held constant at 31.25 μM and AHL was varied.

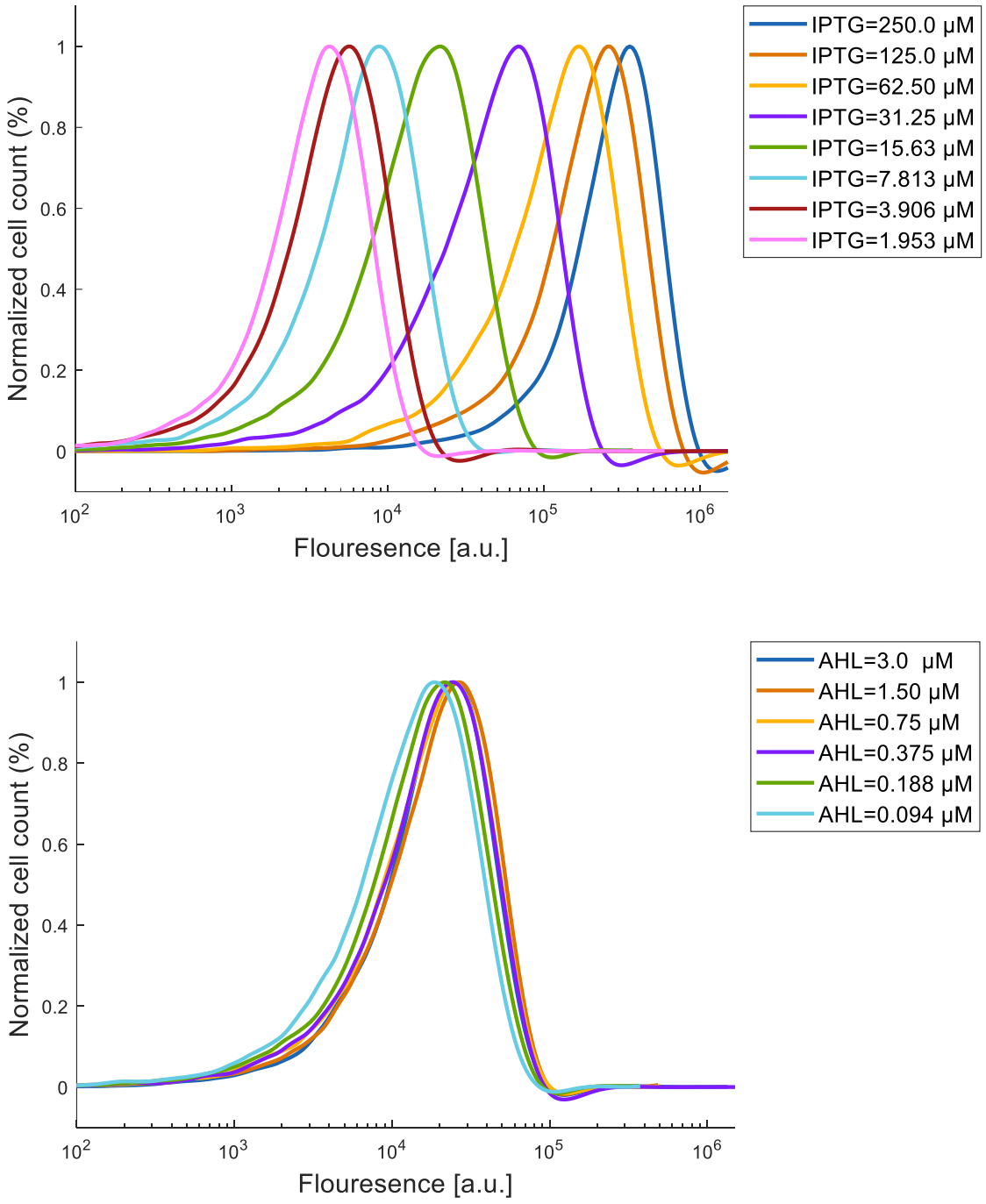
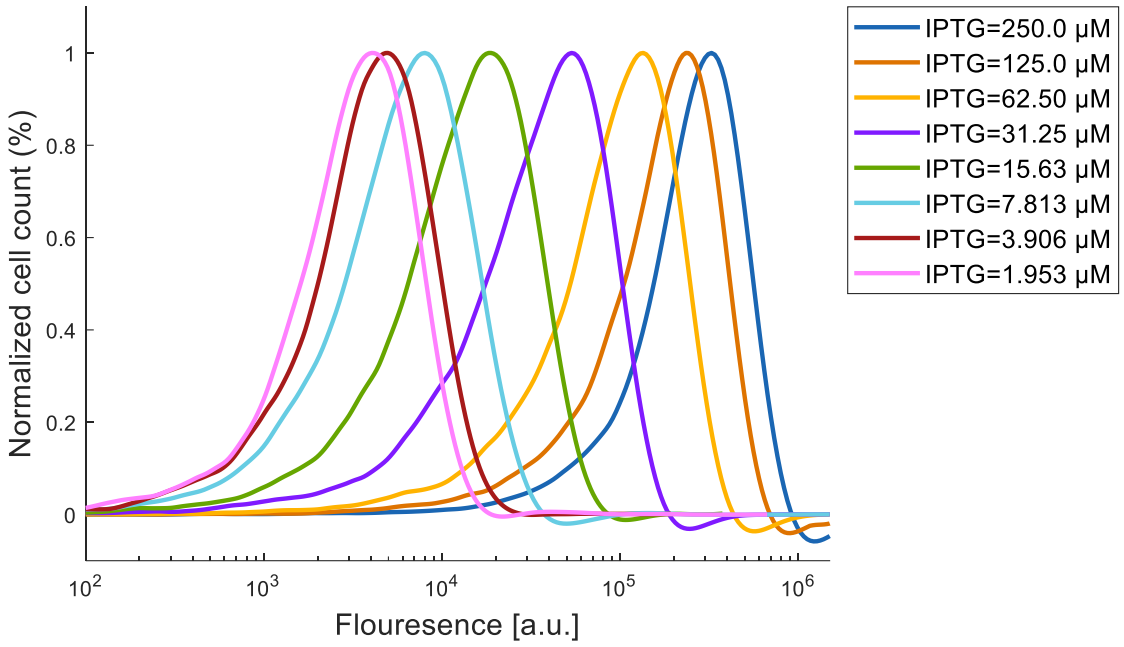


Fig. S14.131. GFP flow cytometry data for a population of cells containing APF ($P_{luxTCTA}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4C). (A) AHL was held constant at 0.188 μM and IPTG was varied. (B) IPTG was held constant at 15.63 μM and AHL was varied.



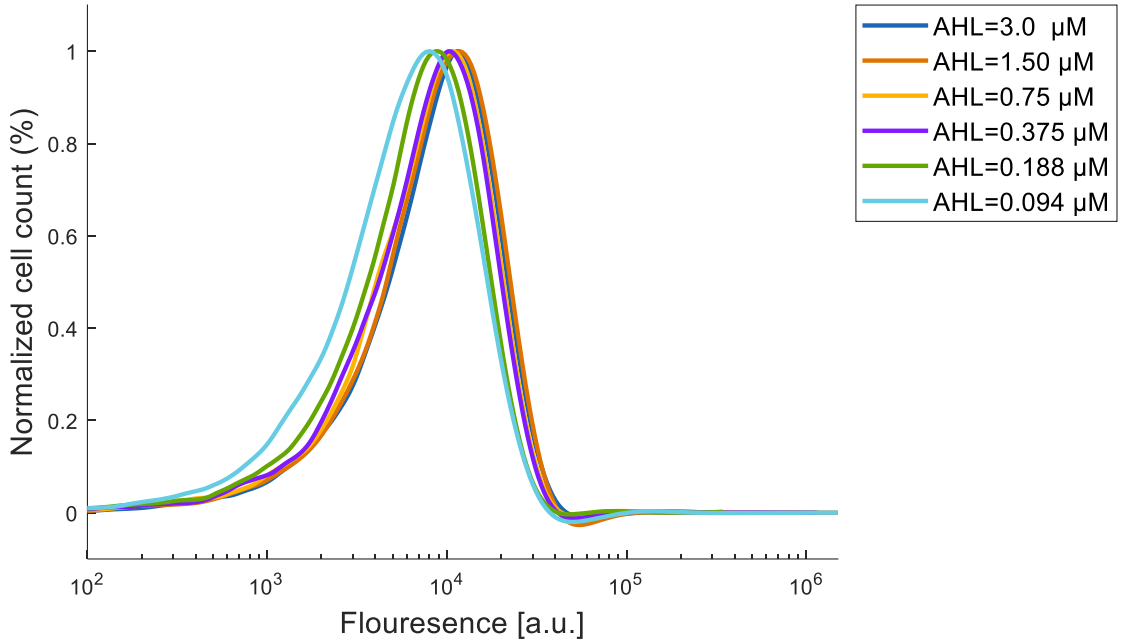


Fig. S14.132. GFP flow cytometry data for a population of cells containing APF ($P_{luxTCTA}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4C). (A) AHL was held constant at 0.094 μM and IPTG was varied. (B) IPTG was held constant at 7.813 μM and AHL was varied.

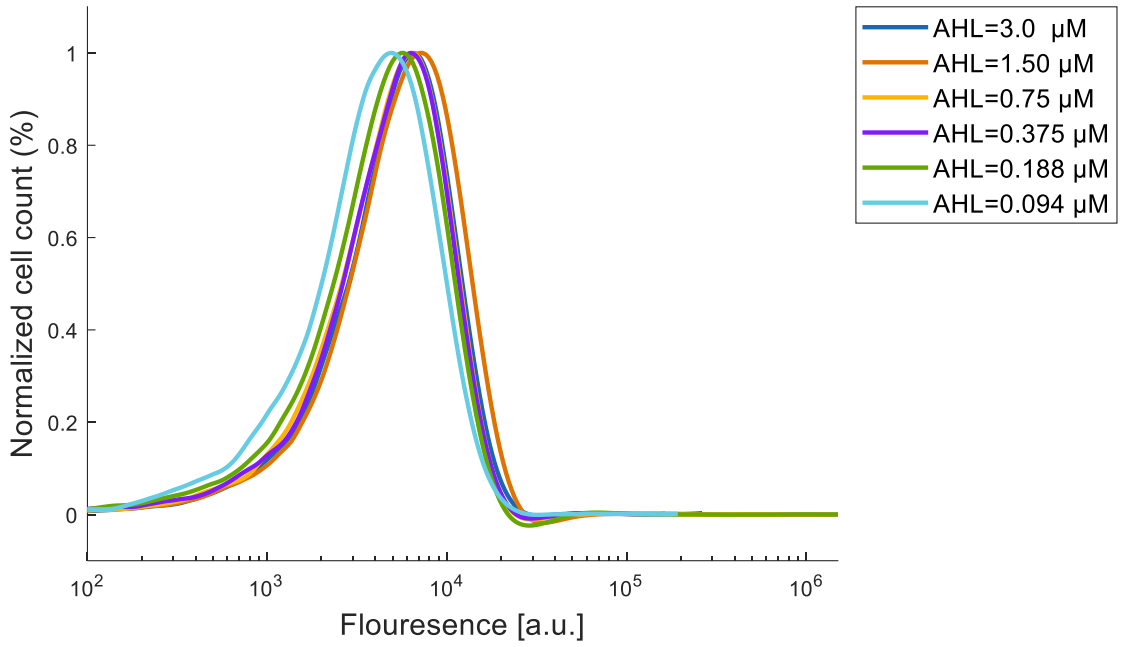


Fig. S8.133. GFP flow cytometry data for a population of cells containing APF ($P_{luxTCTA}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4C). IPTG was held constant at 3.906 μM and AHL was varied.

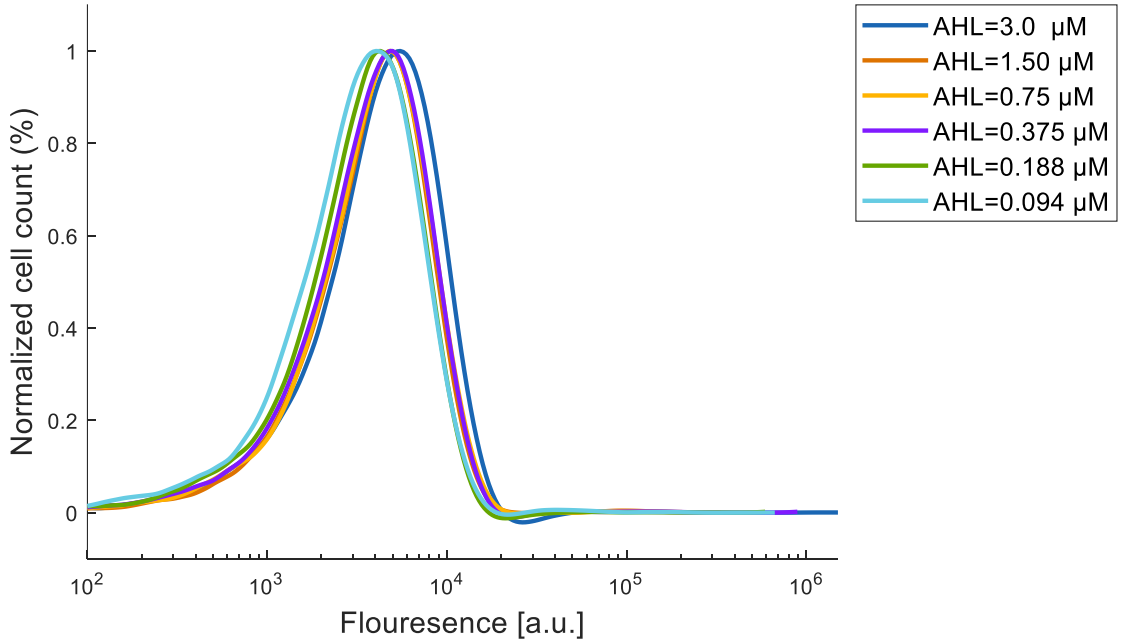


Fig. S14.134. GFP flow cytometry data for a population of cells containing APF ($P_{luxTCTA}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4C). IPTG was held constant at 1.953 μM and AHL was varied.

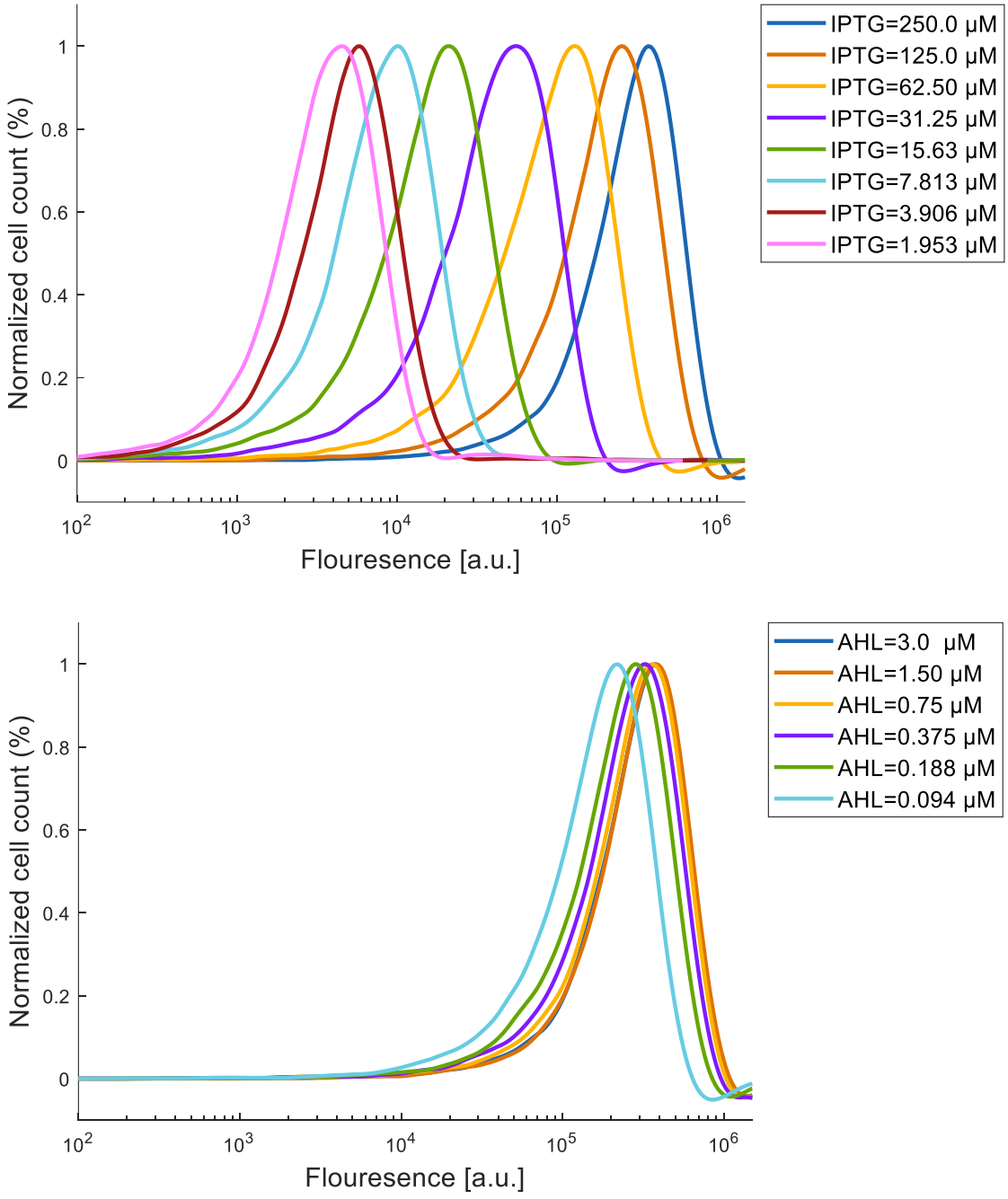


Fig. S14.135. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 3.0 μM and IPTG was varied. (B) IPTG was held constant at 250 μM and AHL was varied.

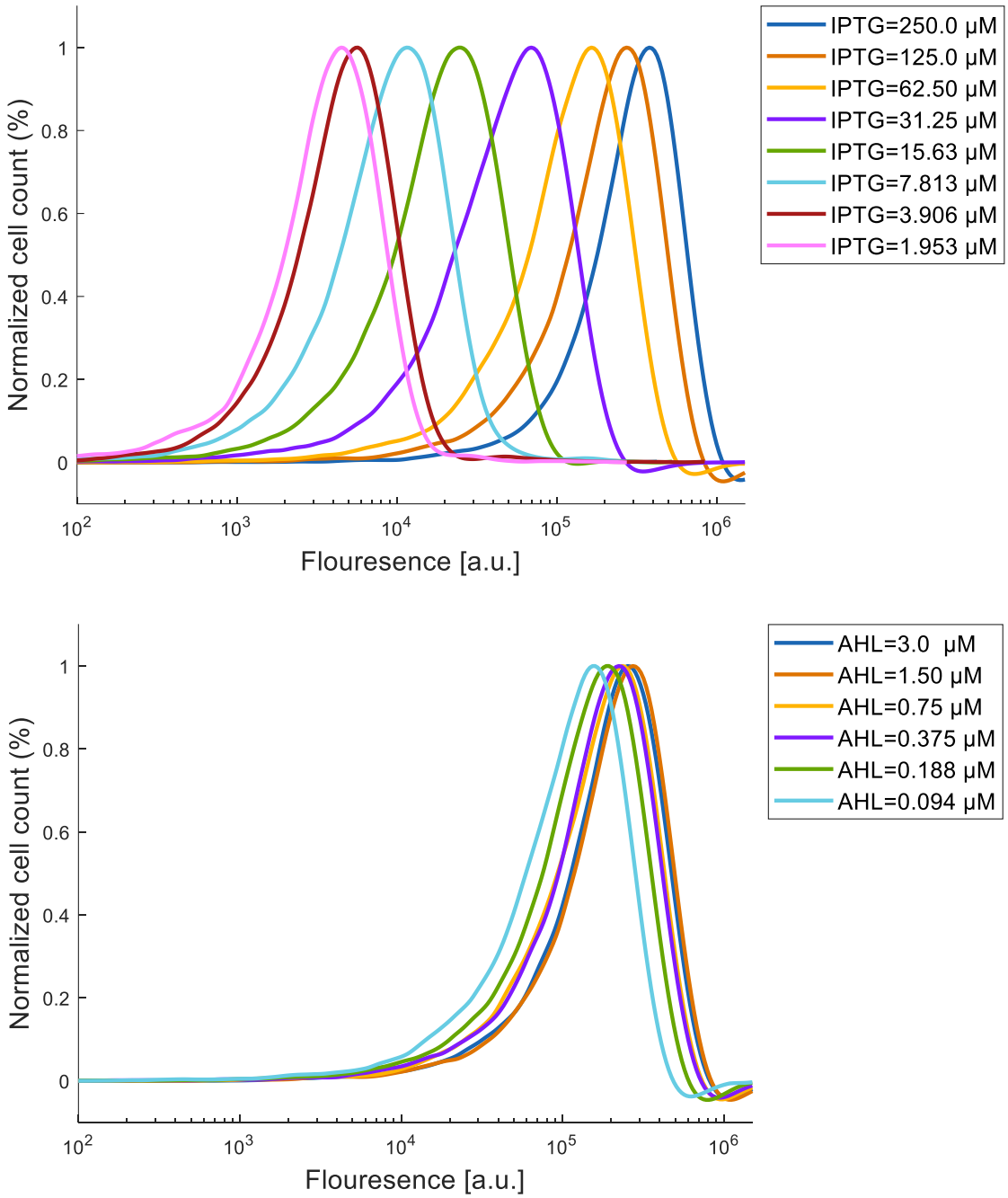


Fig. S14.136. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 1.5 μM and IPTG was varied. (B) IPTG was held constant at 125 μM and AHL was varied.

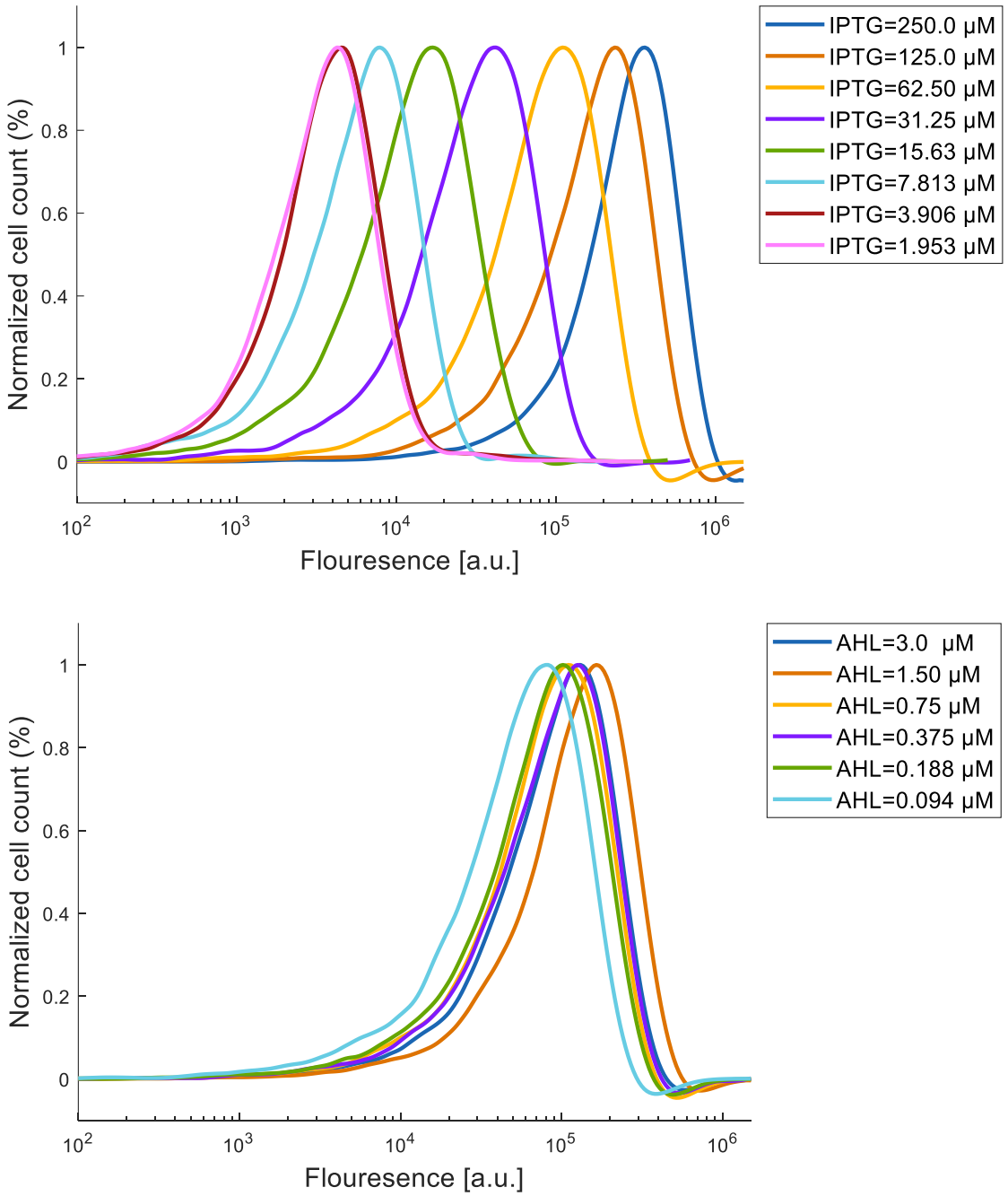


Fig. S14.137. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 0.75 μM and IPTG was varied. (B) IPTG was held constant at 62.5 μM and AHL was varied.

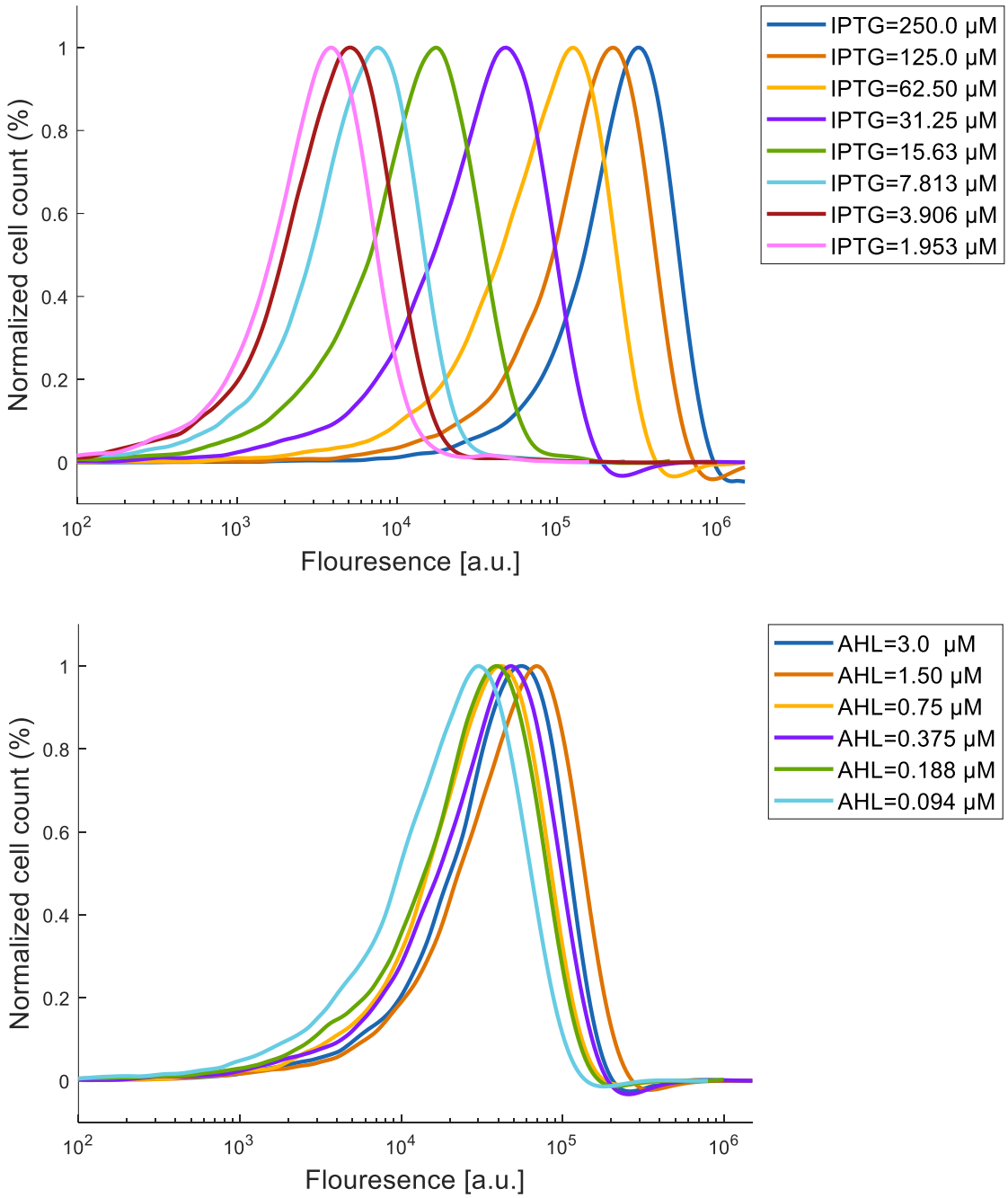


Fig. S14.138. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 0.375 μM and IPTG was varied. (B) IPTG was held constant at 31.25 μM and AHL was varied.

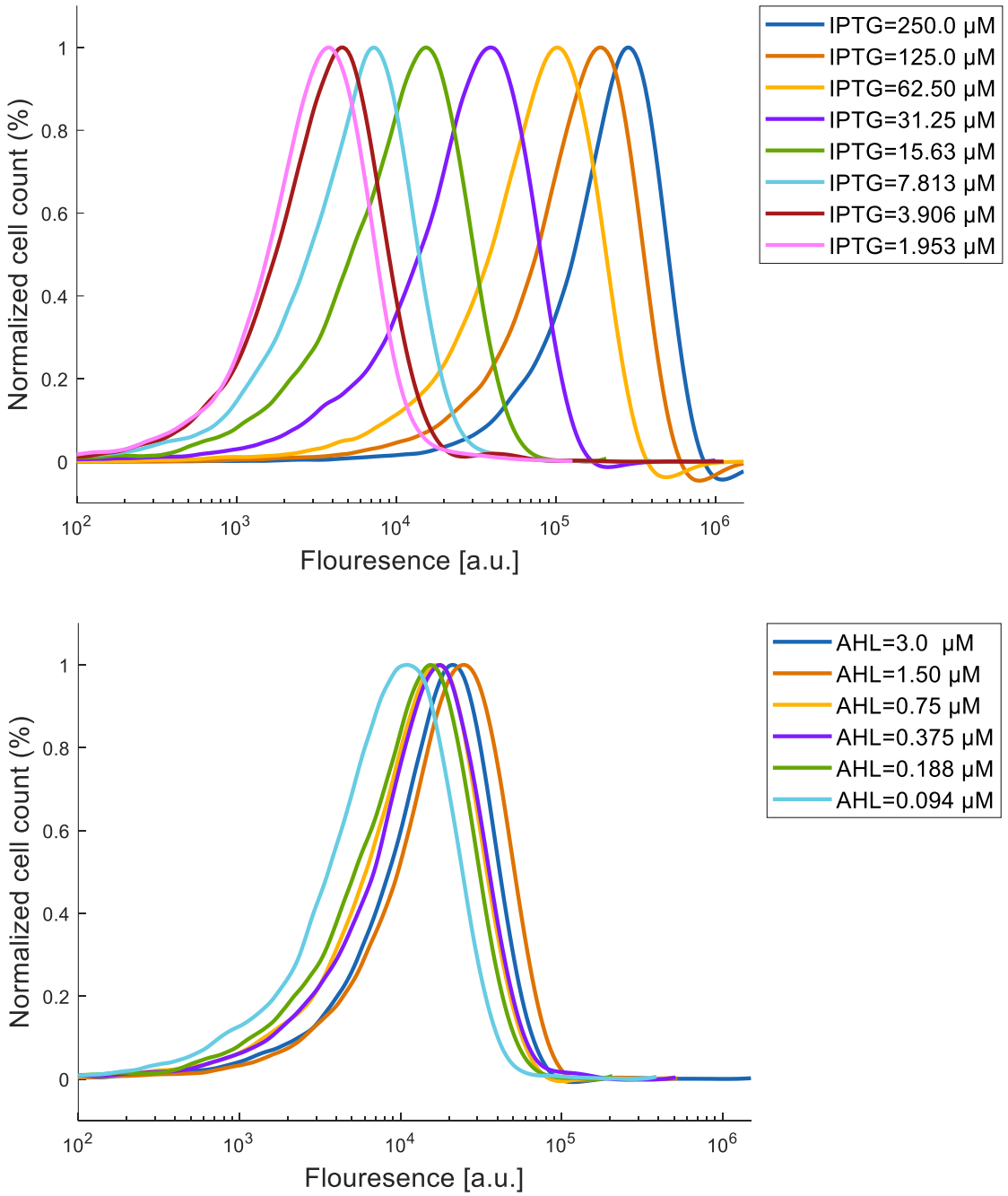
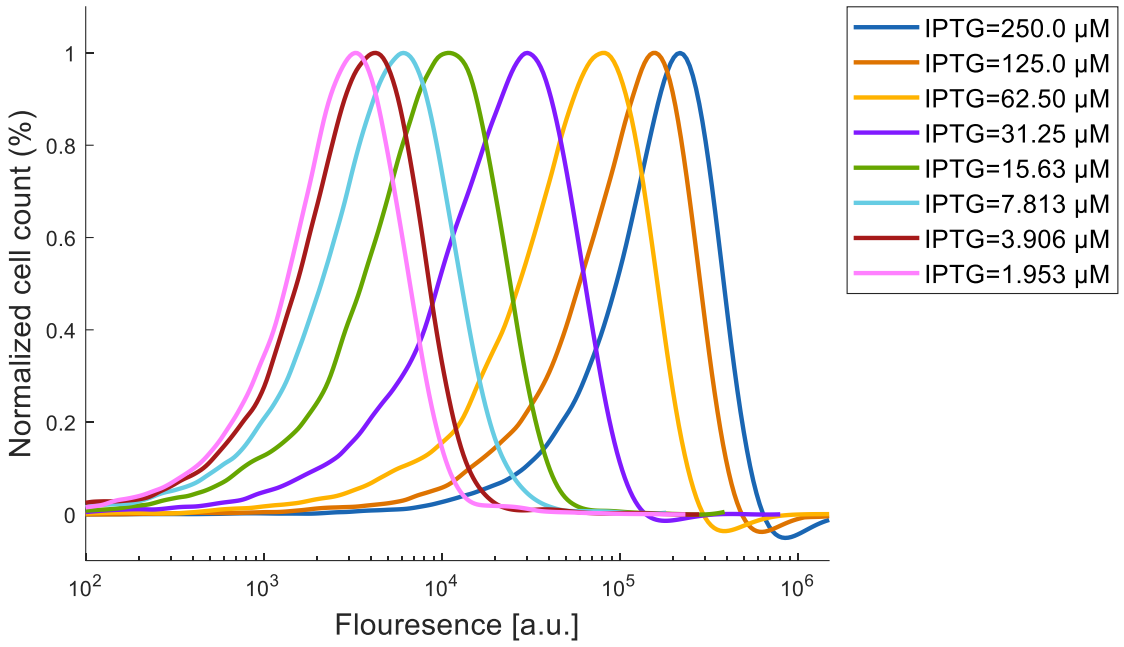


Fig. S14.139. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 0.188 μM and IPTG was varied. (B) IPTG was held constant at 15.63 μM and AHL was varied.



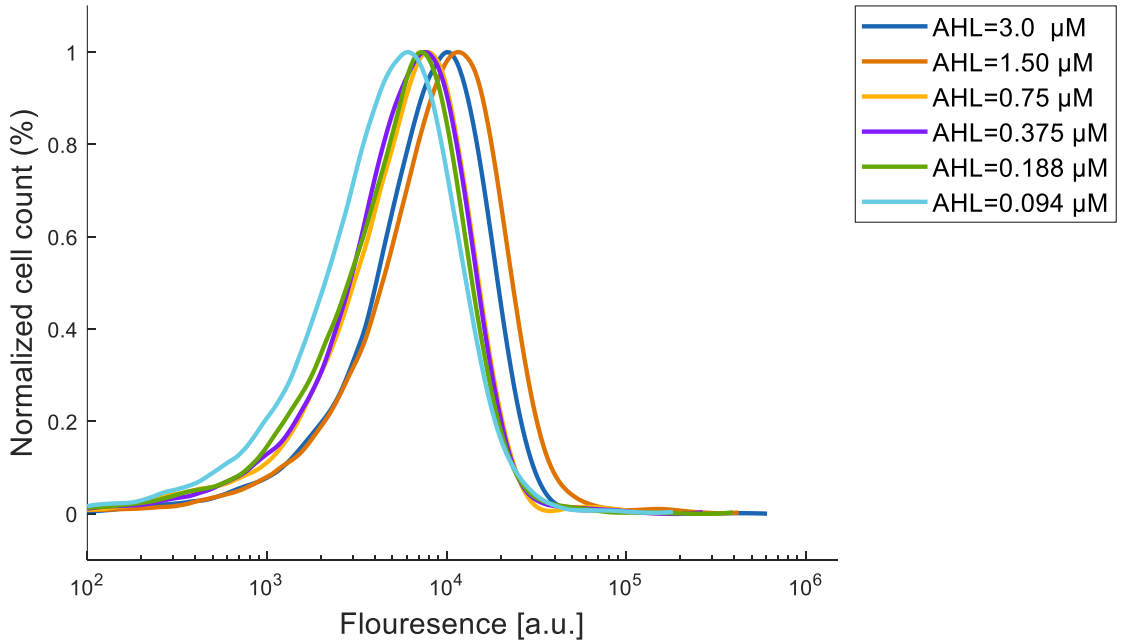


Fig. S14.40 GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 0.094 μM and IPTG was varied. (B) IPTG was held constant at 7.813 μM and AHL was varied.

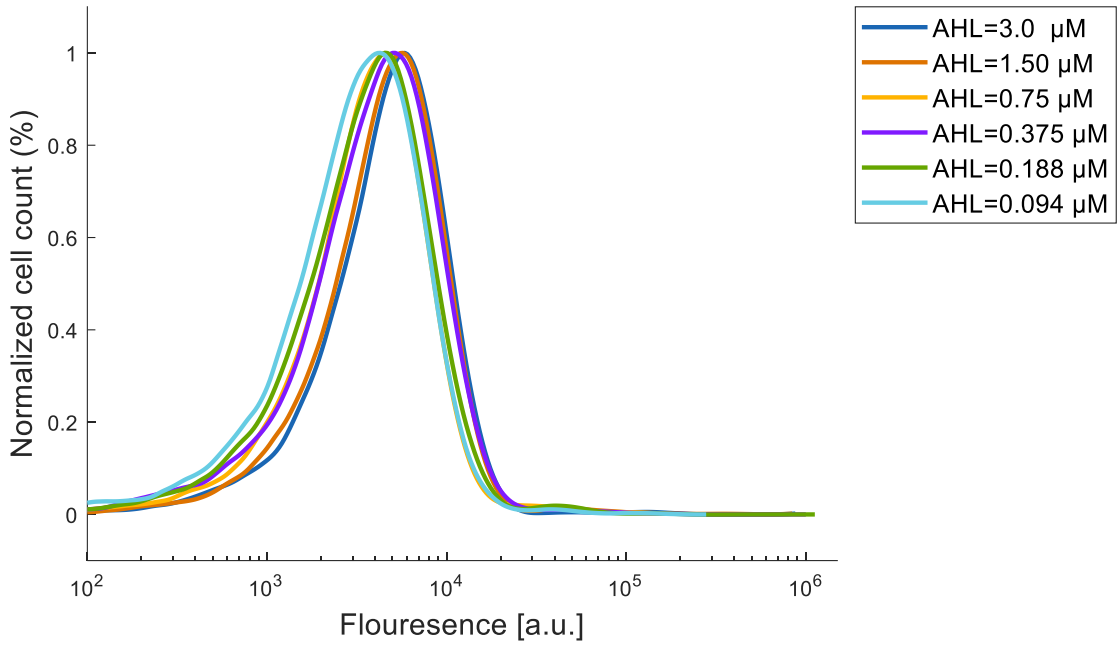


Fig. S14.141. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). IPTG was held constant at 3.906 μM and AHL was varied.

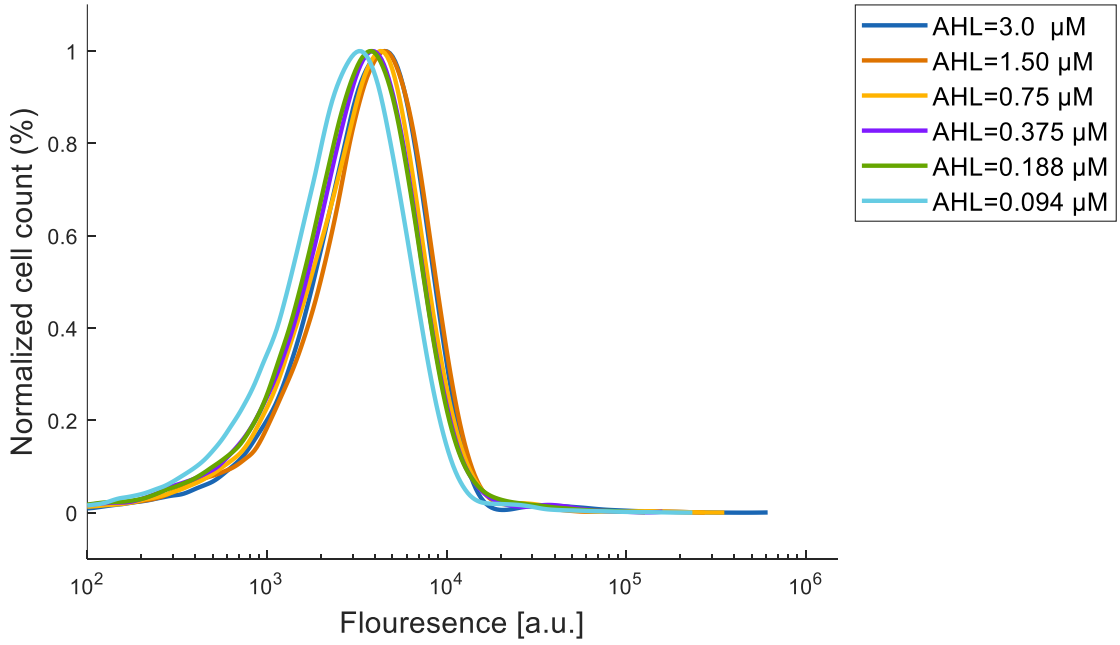
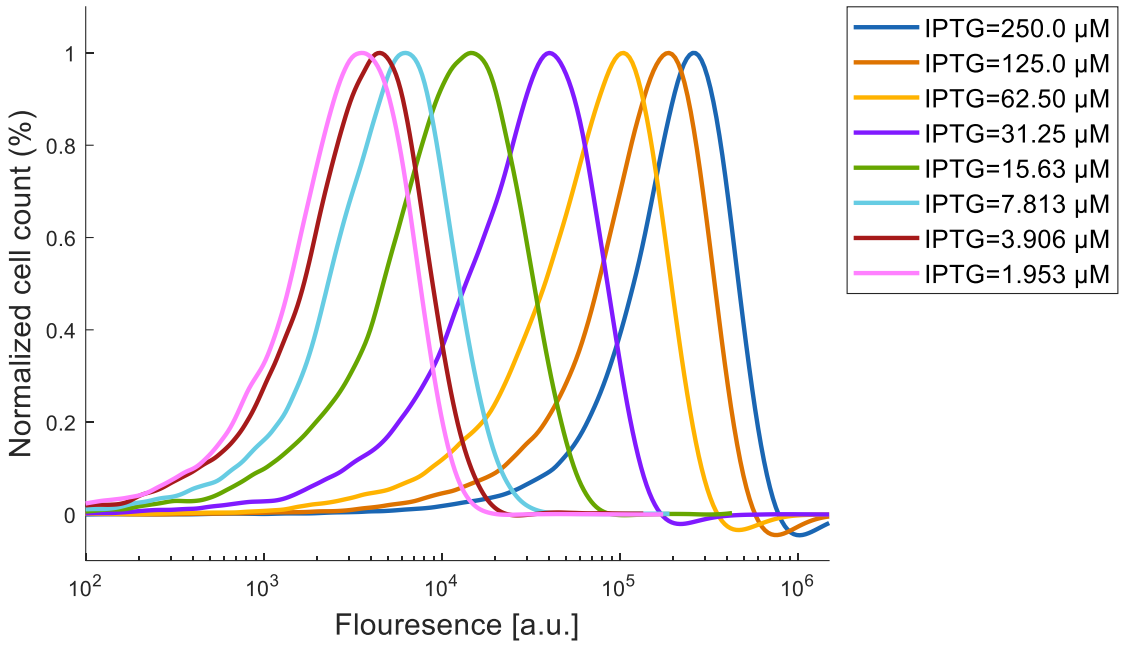


Fig. S14.142. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). IPTG was held constant at 1.953 μM and AHL was varied.



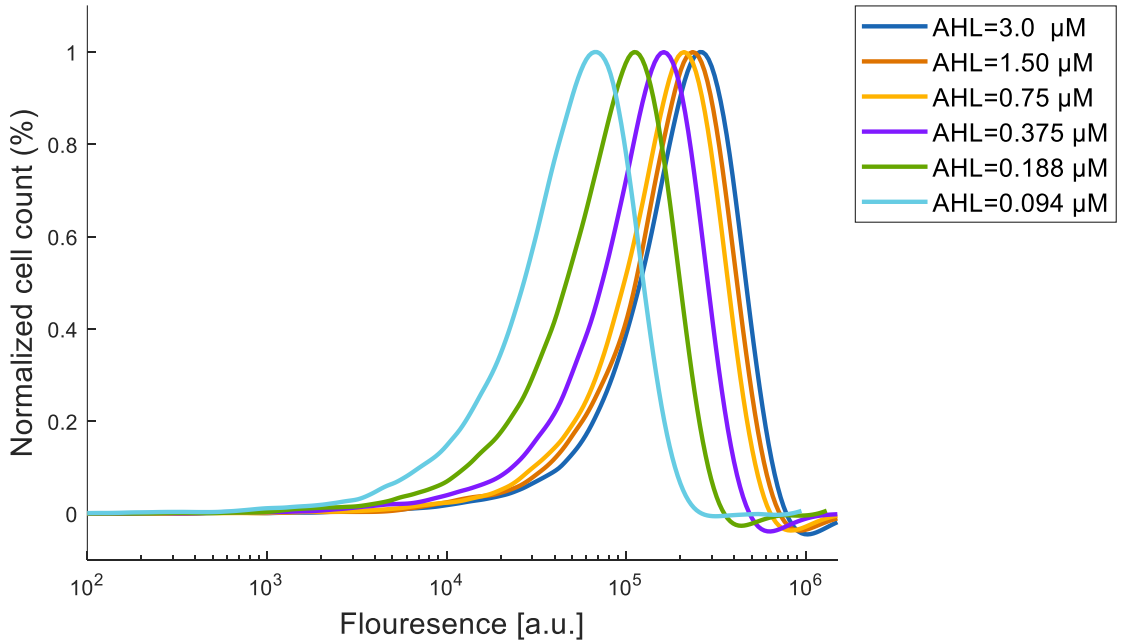


Fig. S14.143. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 3.0 μM and IPTG was varied. (B) IPTG was held constant at 250 μM and AHL was varied.

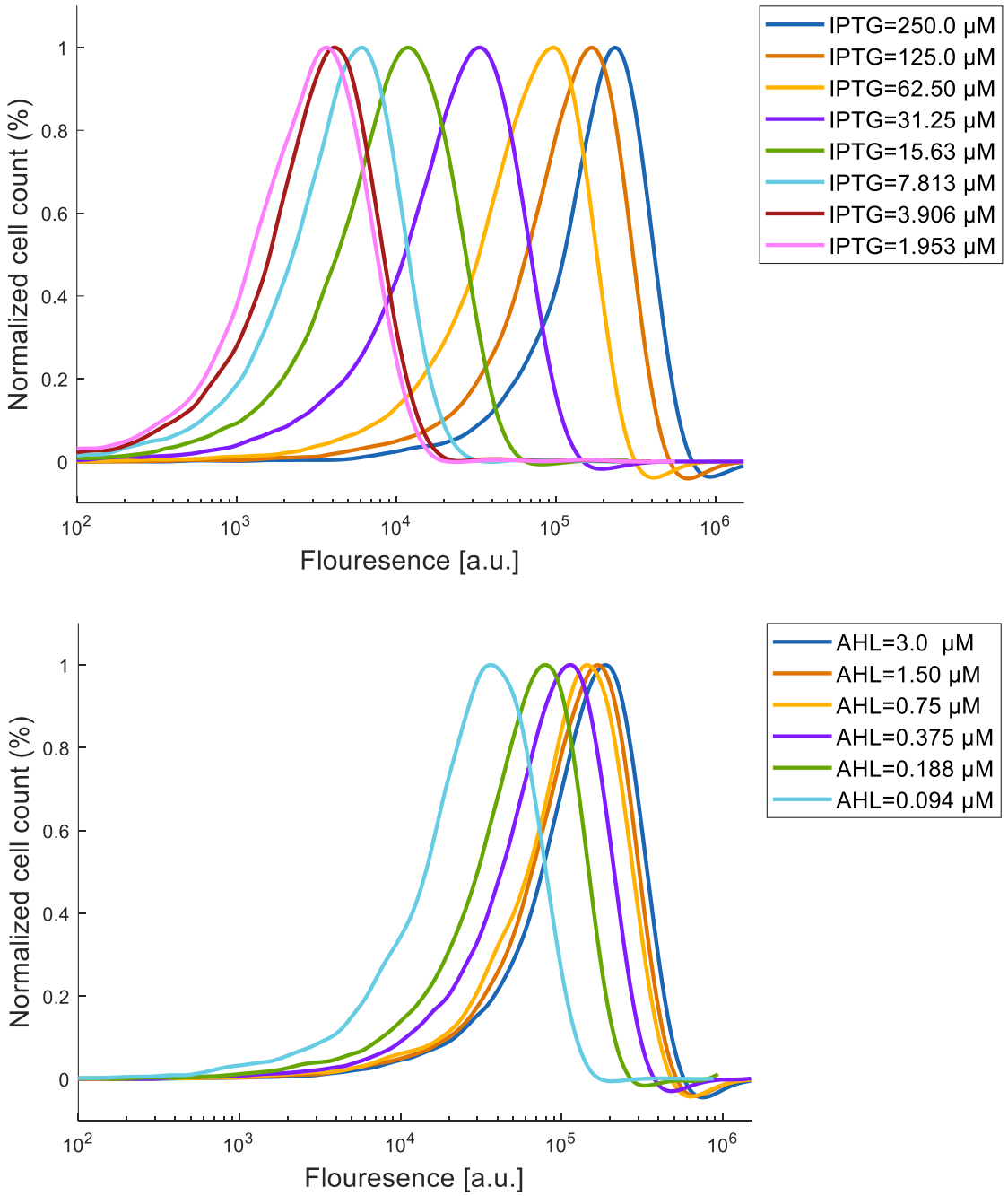


Fig. S14.144. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 1.5 μM and IPTG was varied. (B) IPTG was held constant at 125 μM and AHL was varied.

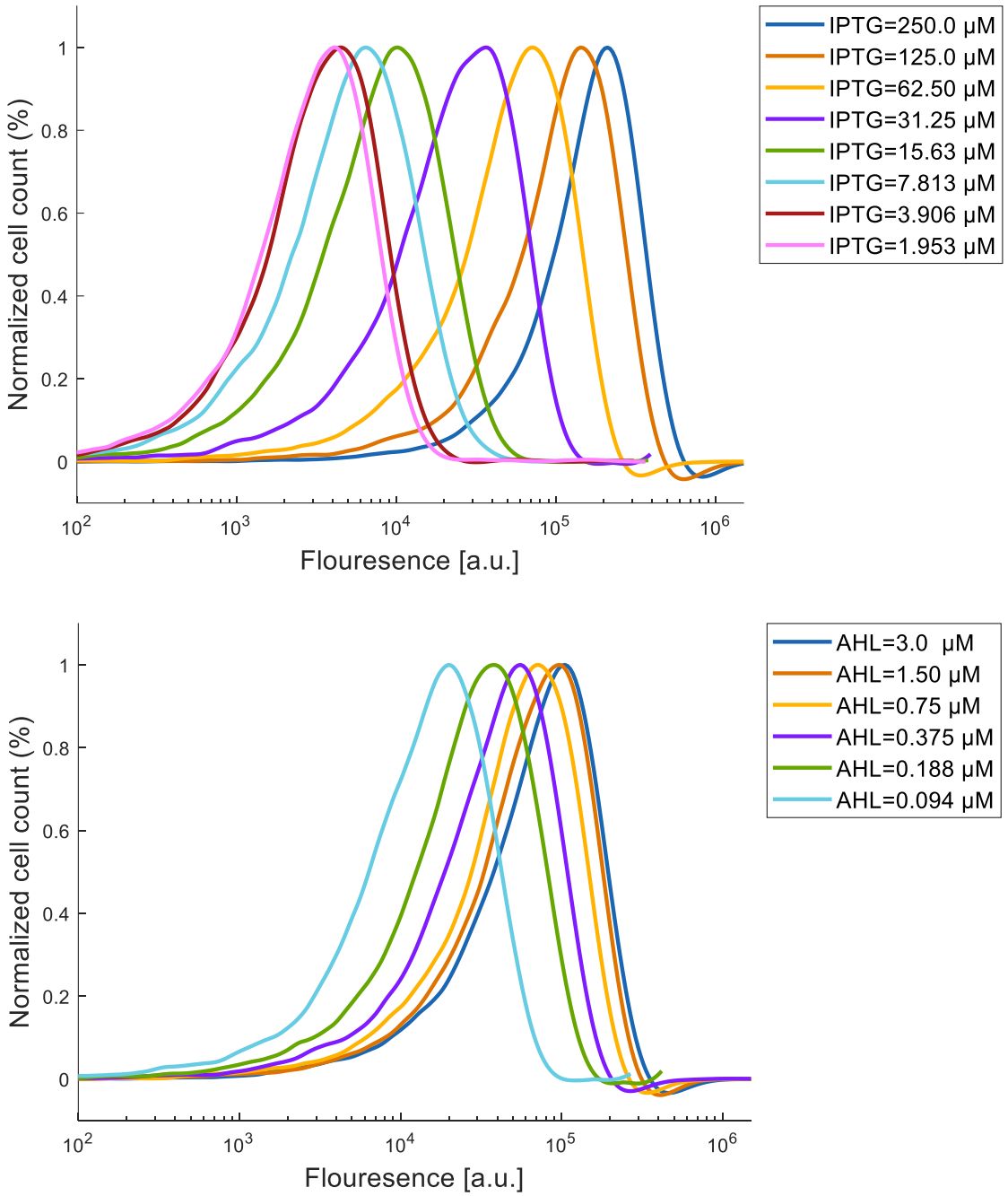


Fig. S14.145. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 0.75 μM and IPTG was varied. (B) IPTG was held constant at 62.5 μM and AHL was varied.

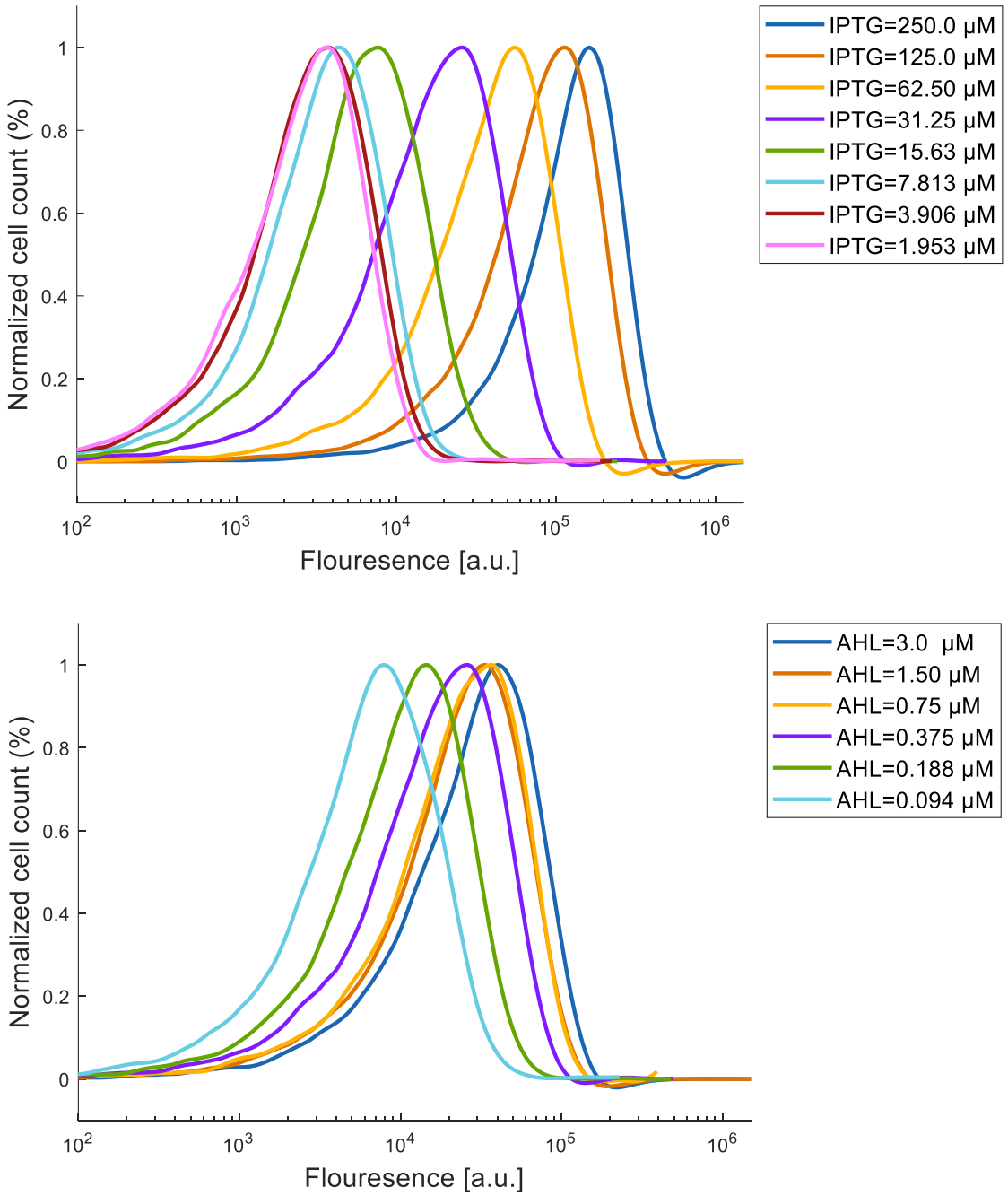
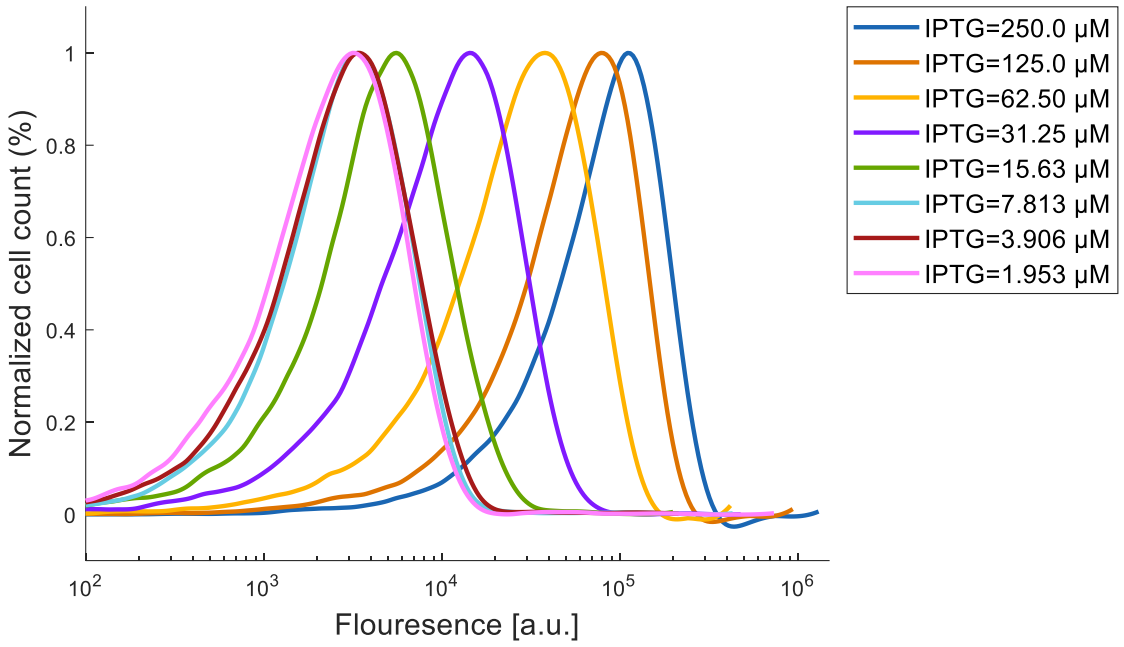


Fig. S14.146. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 0.375 μM and IPTG was varied. (B) IPTG was held constant at 31.25 μM and AHL was varied.



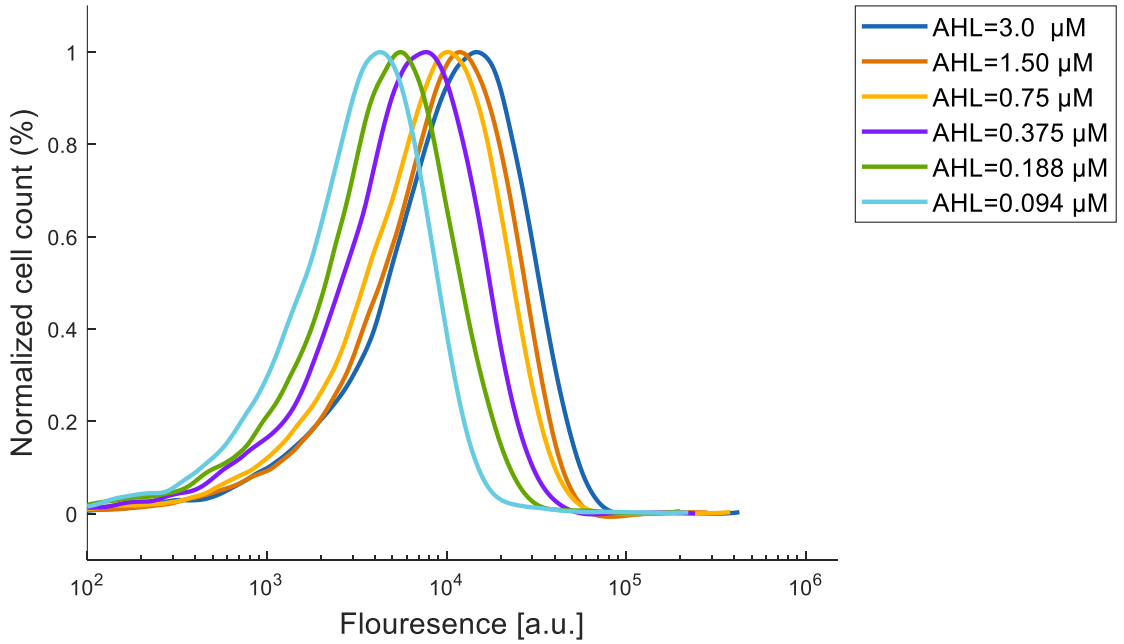


Fig. S14.147. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 0.188 μM and IPTG was varied. (B) IPTG was held constant at 15.63 μM and AHL was varied.

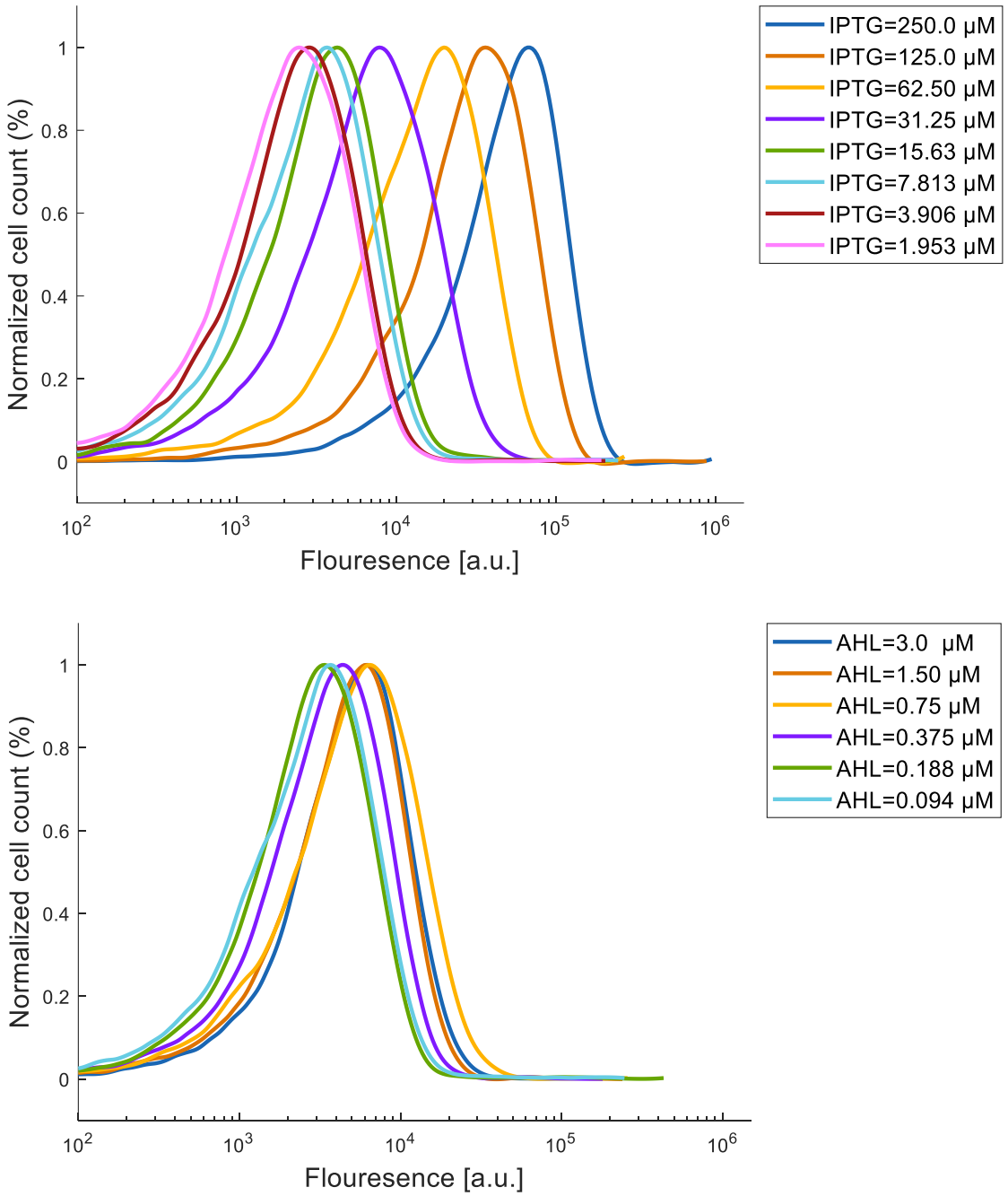


Fig. S14.148. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). (A) AHL was held constant at 0.094 μM and IPTG was varied. (B) IPTG was held constant at 7.813 μM and AHL was varied.

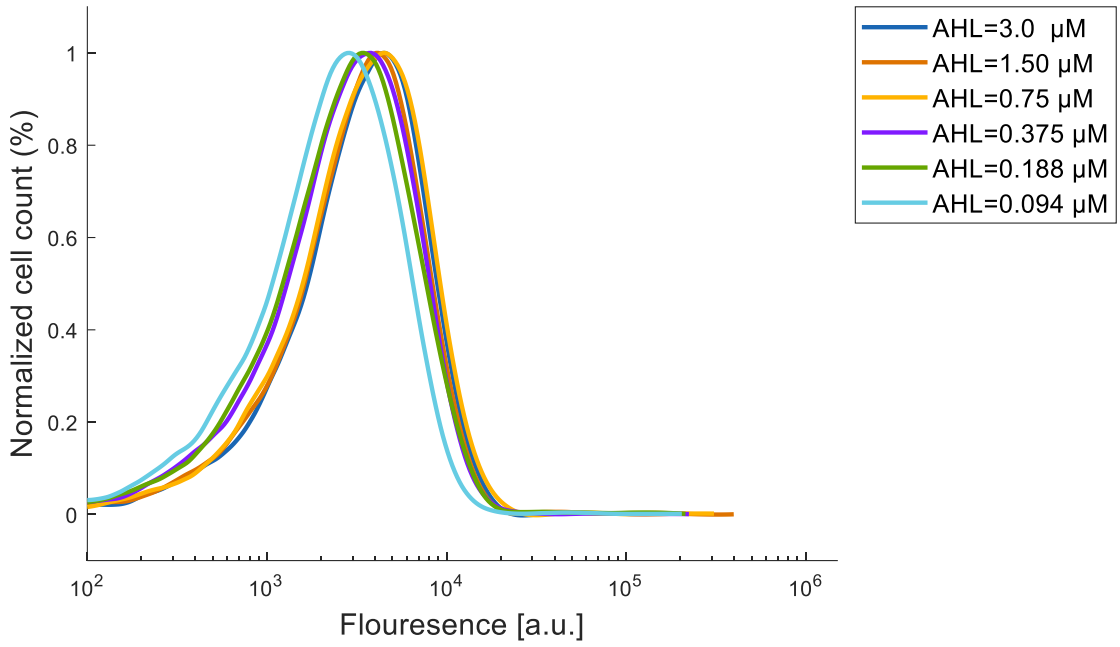


Fig. S14.149. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). IPTG was held constant at 3.906 μM and AHL was varied.

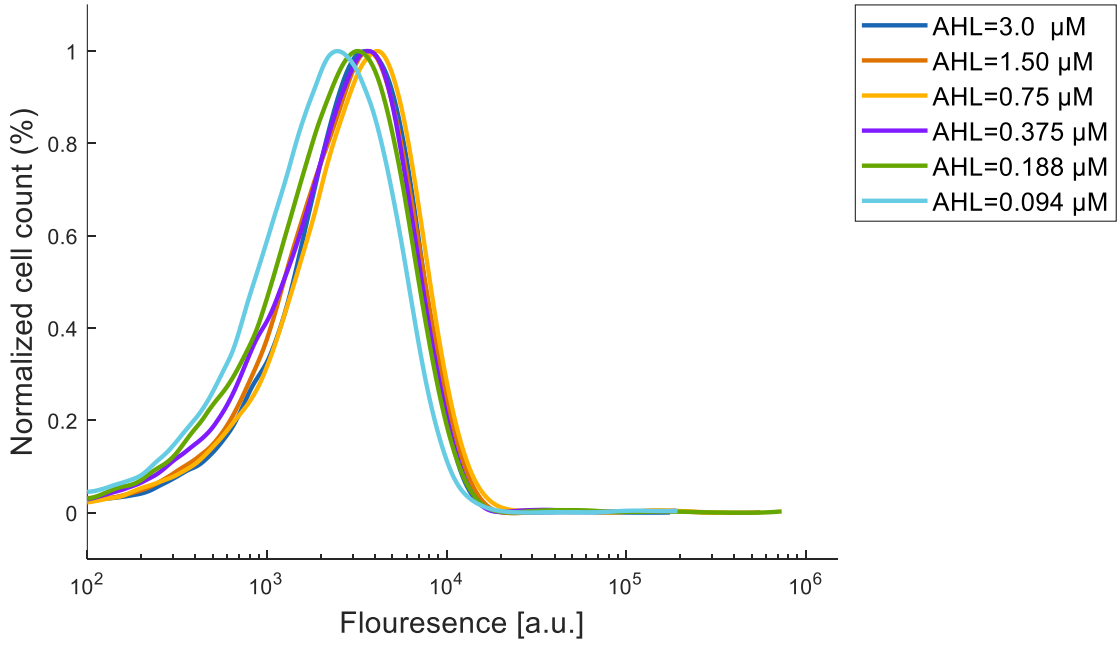


Fig. S14.150. GFP flow cytometry data for a population of cells containing APF ($P_{luxGTTG}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4D). IPTG was held constant at 1.953 μM and AHL was varied.

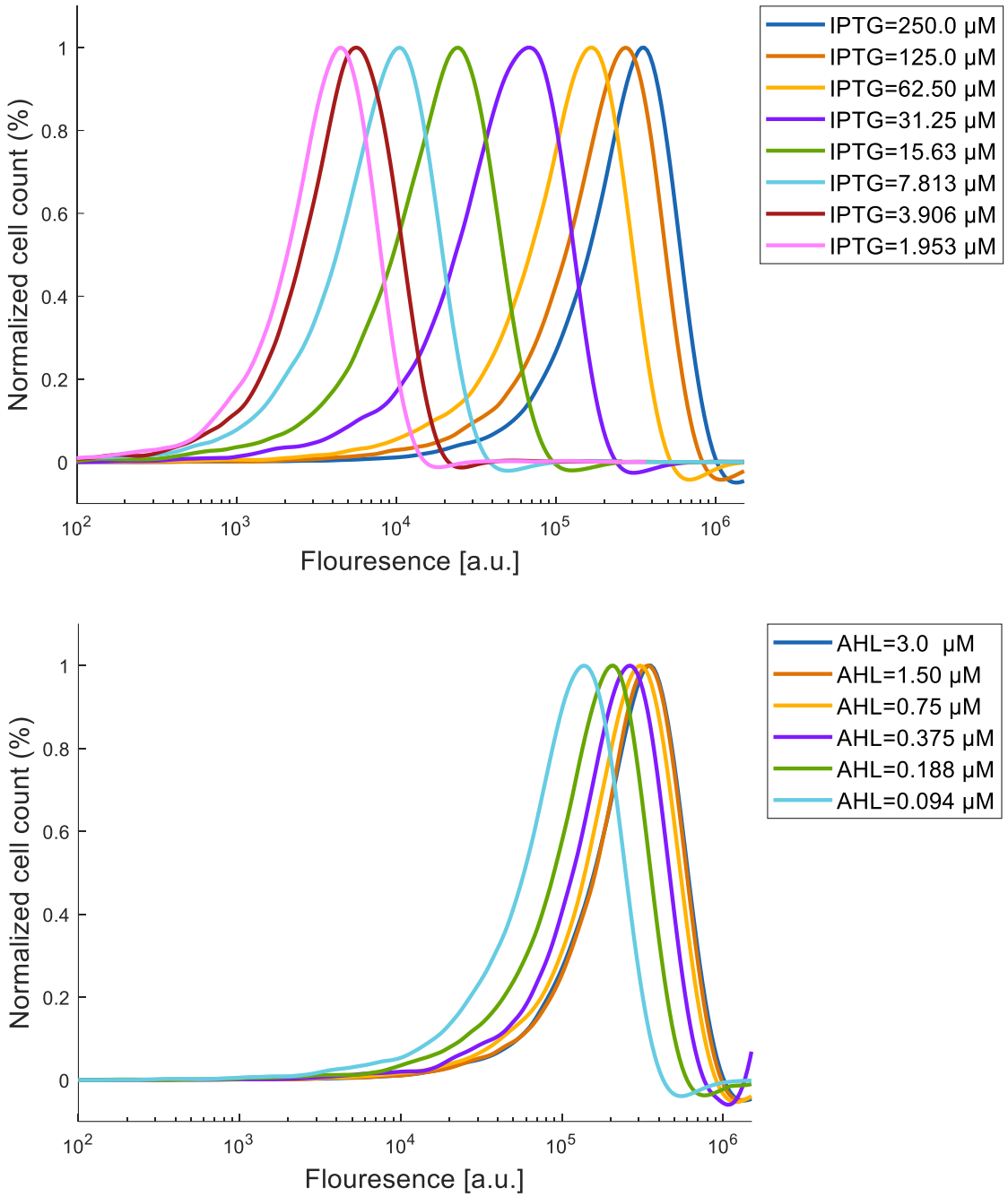
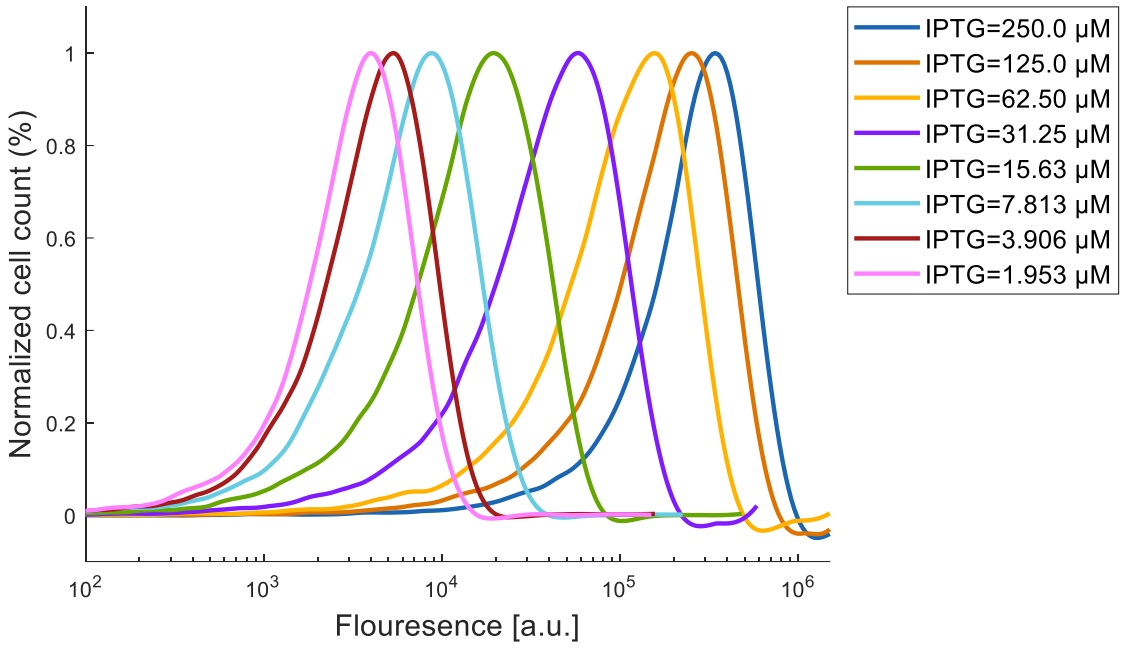


Fig. S14.151. GFP flow cytometry data for a population of cells containing APF ($P_{luxGAGC}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4E). (A) AHL was held constant at 3.0 μM and IPTG was varied. (B) IPTG was held constant at 250 μM and AHL was varied.



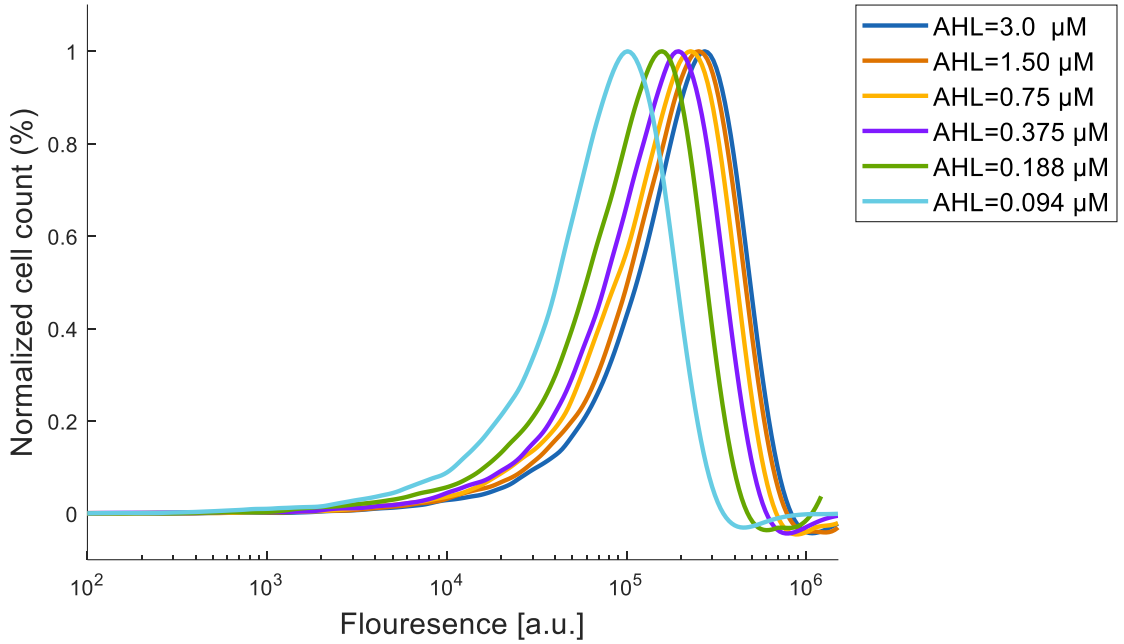


Fig. S14.152. GFP flow cytometry data for a population of cells containing APF ($P_{luxGAGC}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4E). (A) AHL was held constant at 1.5 μM and IPTG was varied. (B) IPTG was held constant at 125 μM and AHL was varied.

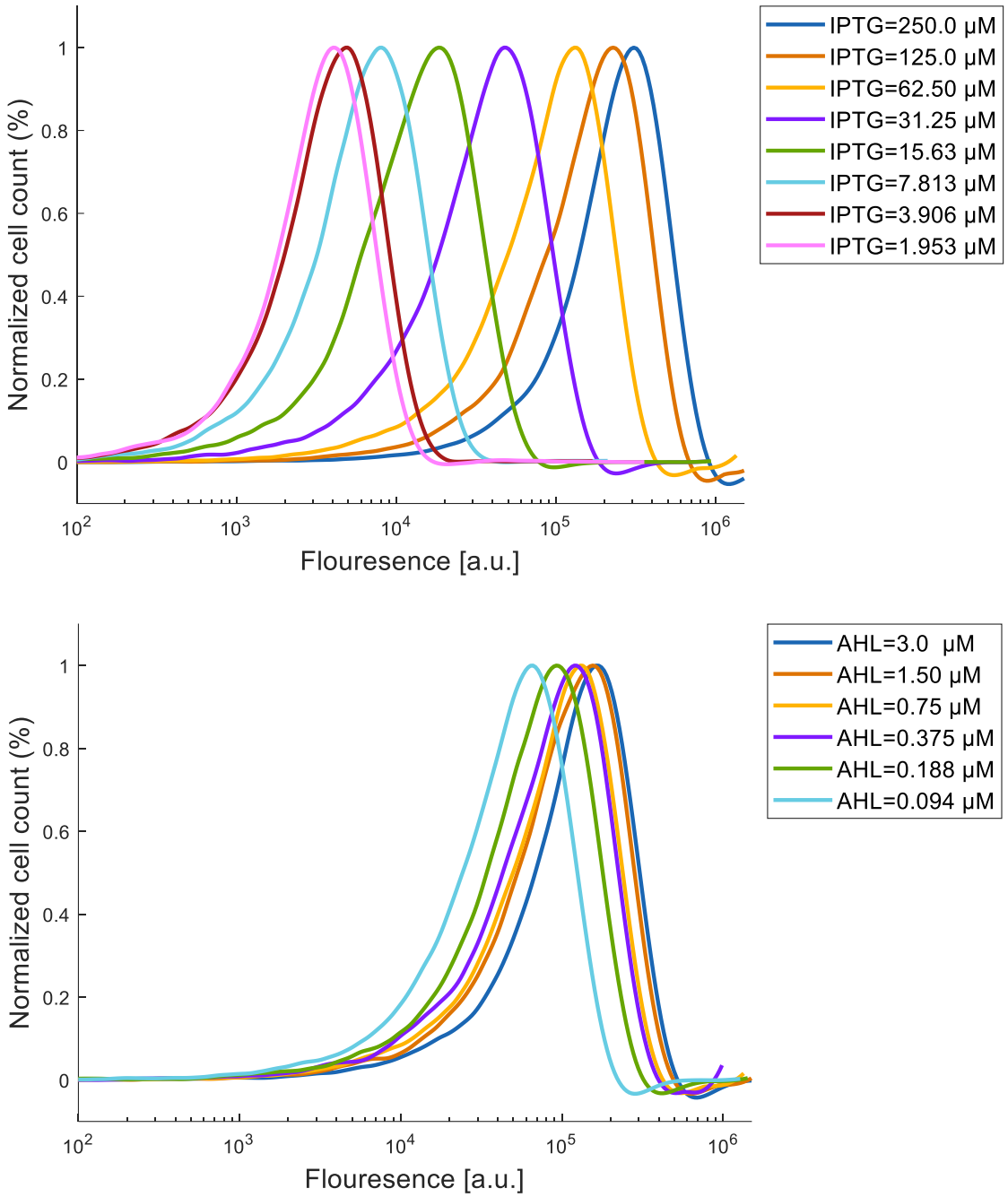


Fig. S14.153. GFP flow cytometry data for a population of cells containing APF ($P_{luxGAGC}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4E). (A) AHL was held constant at 0.75 μM and IPTG was varied. (B) IPTG was held constant at 62.5 μM and AHL was varied.

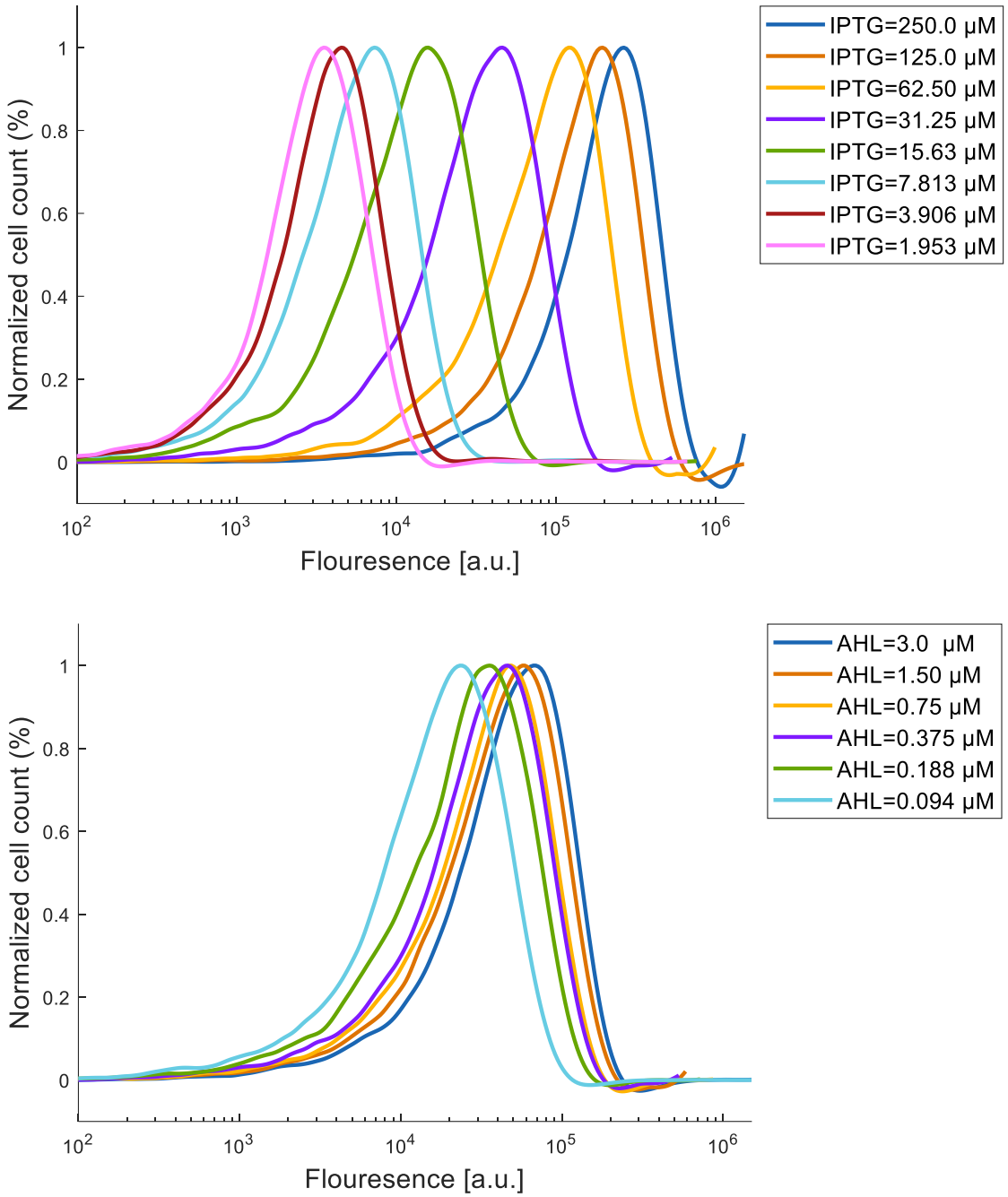


Fig. S14.154. GFP flow cytometry data for a population of cells containing APF ($P_{luxGAGC}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4E). (A) AHL was held constant at 0.375 μM and IPTG was varied. (B) IPTG was held constant at 31.25 μM and AHL was varied.

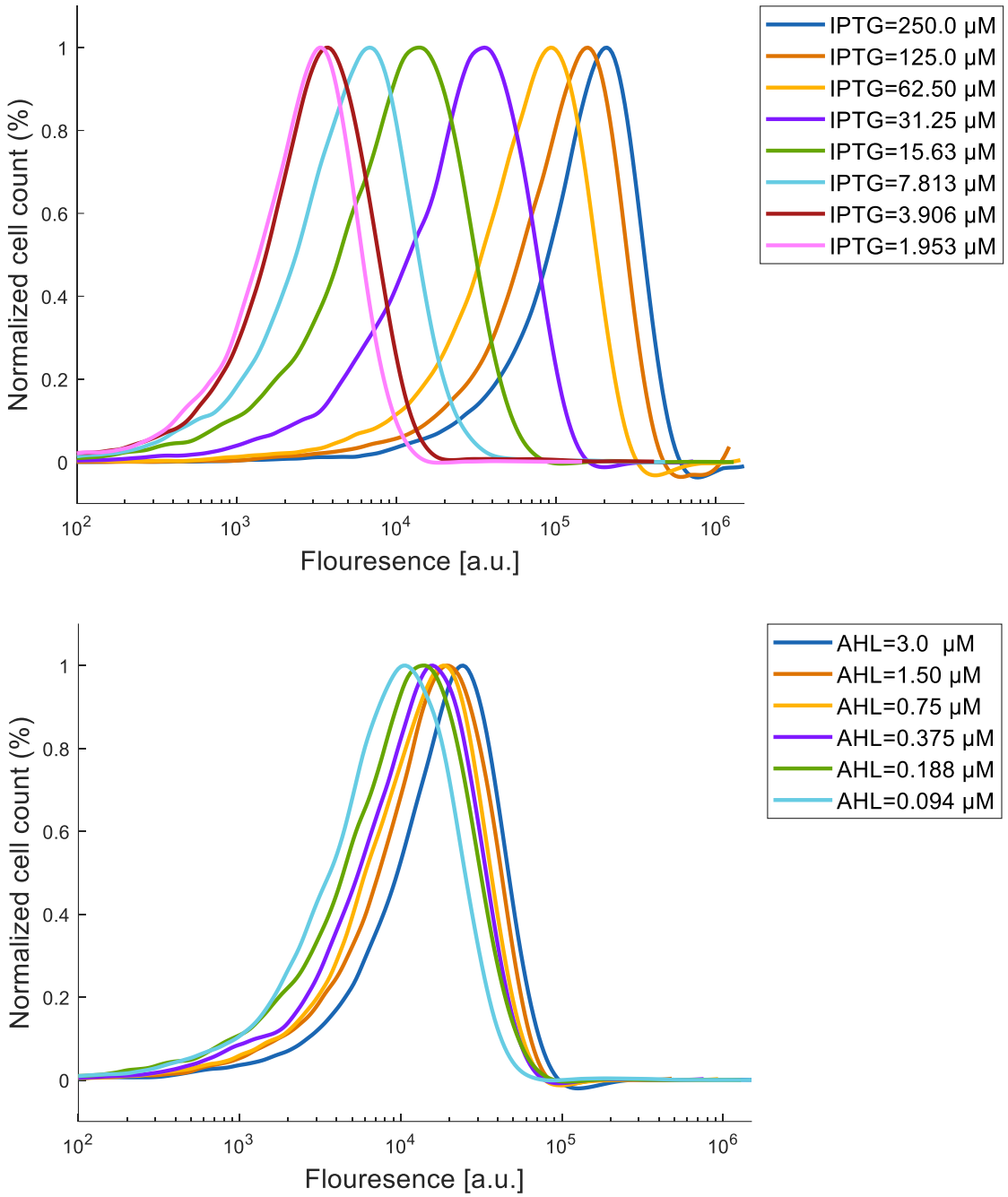


Fig. S14.155. GFP flow cytometry data for a population of cells containing APF ($P_{luxGAGC}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4E). (A) AHL was held constant at 0.188 μM and IPTG was varied. (B) IPTG was held constant at 15.63 μM and AHL was varied.

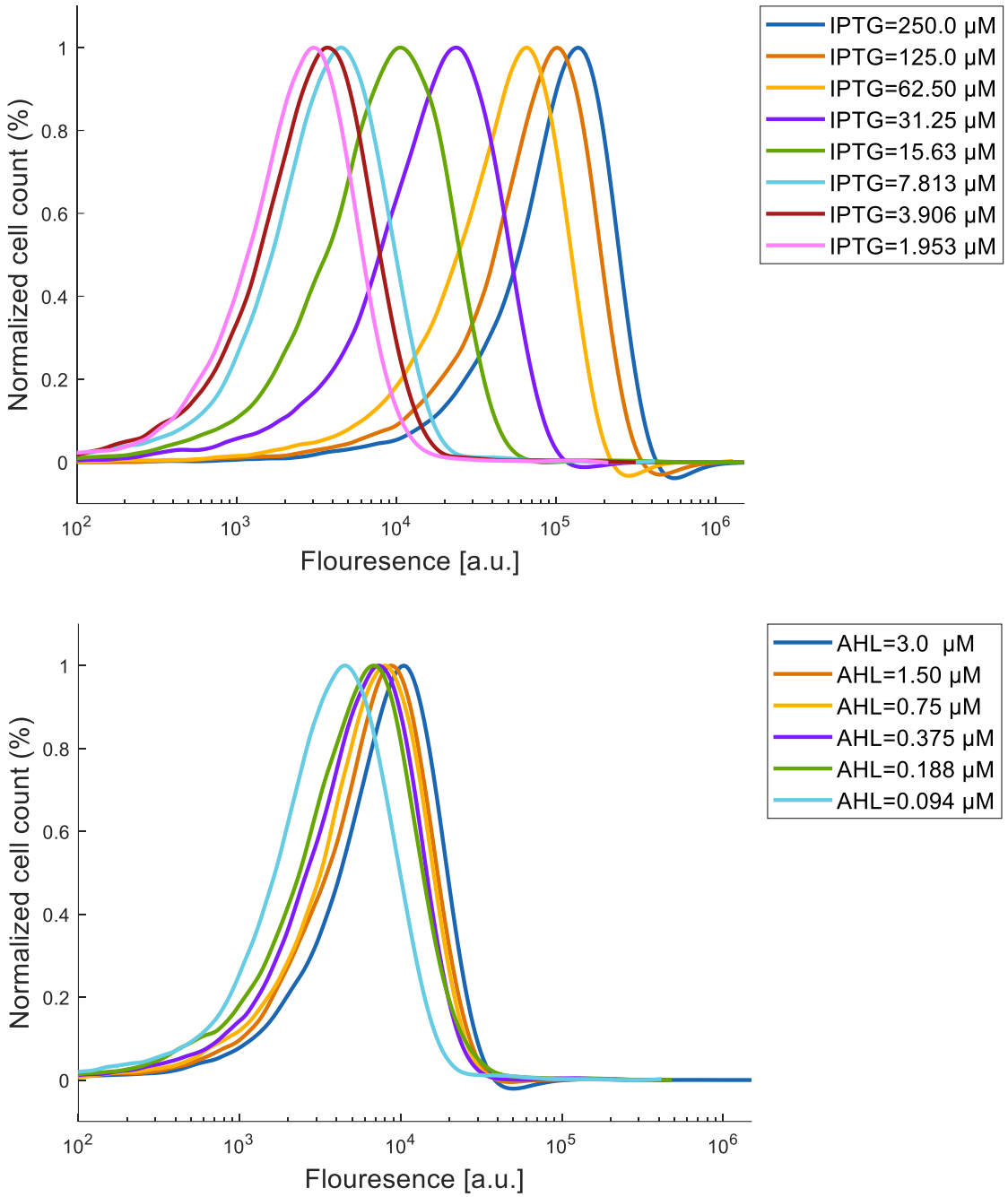


Fig. S14.156. GFP flow cytometry data for a population of cells containing APF ($P_{luxGAGC}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4E). (A) AHL was held constant at 0.094 μM and IPTG was varied. (B) IPTG was held constant at 7.813 μM and AHL was varied.

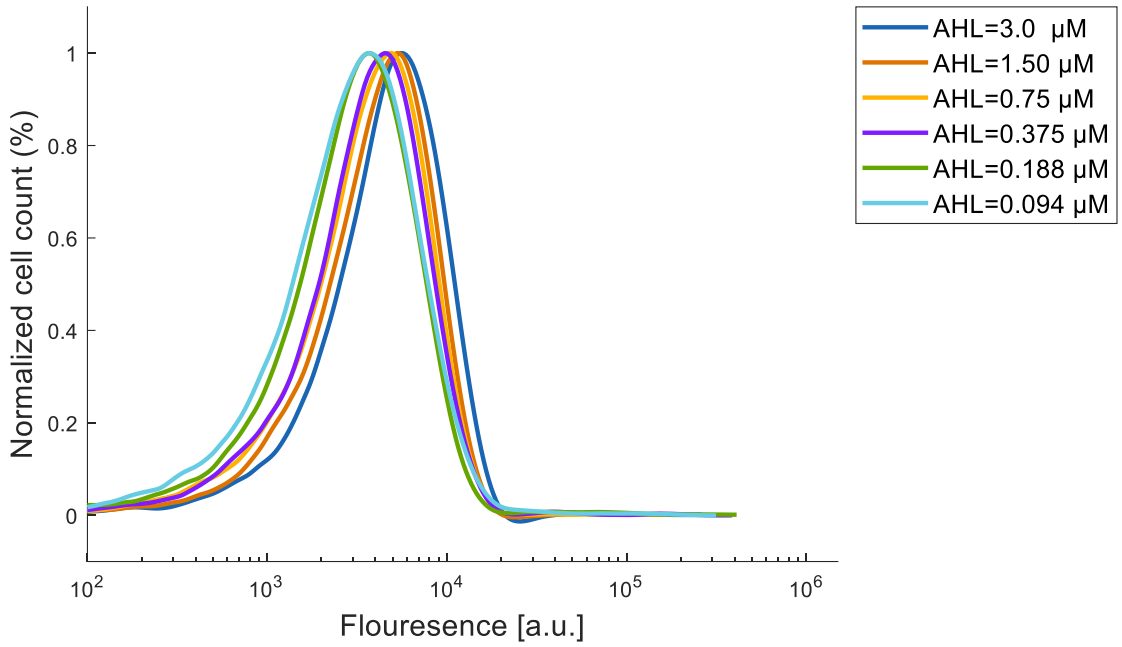


Fig. S14.157. GFP flow cytometry data for a population of cells containing APF ($P_{luxGAGC}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4E). IPTG was held constant at 3.906 μM and AHL was varied.

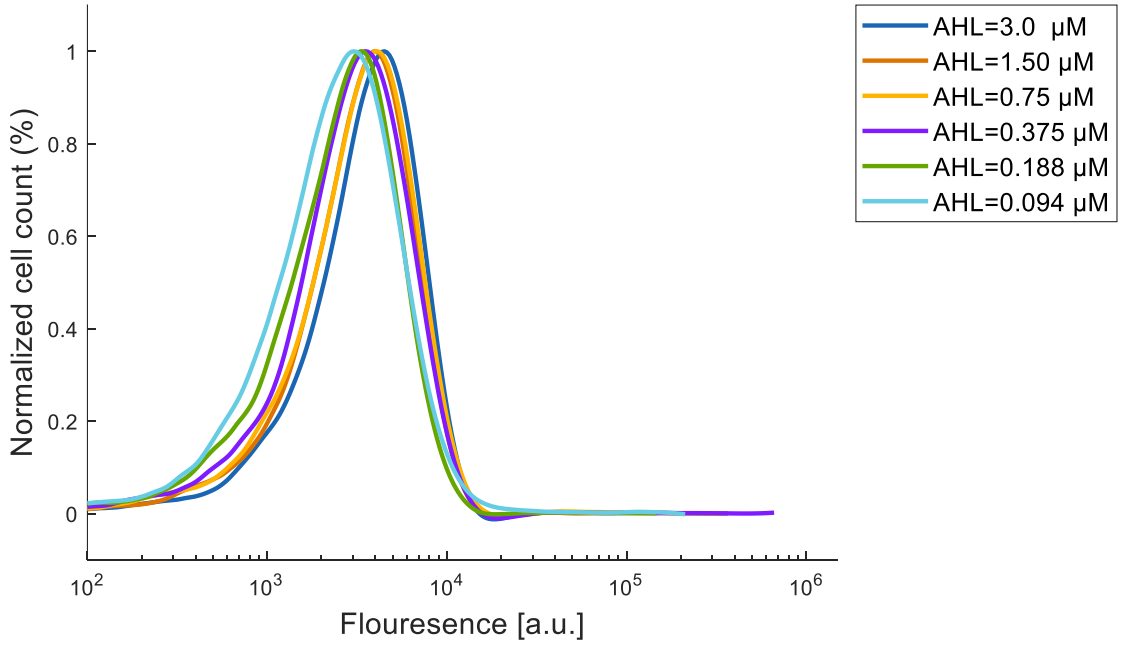


Fig. S14.158. GFP flow cytometry data for a population of cells containing APF ($P_{luxGAGC}$) and ANF (P_{lacO1}) loops and combinatorial promoter ($P_{lux/lacO}$-GFP) to power-law and multiplication function (Fig. S6.4A and S6.4E). IPTG was held constant at 1.953 μM and AHL was varied.

15. List of biological parts used in this study

Part	Description and source	DNA sequence	Mutation done in this study
araC	AraC coding sequence(*14*)	atggctgaagcgcaaaatgatcccctgctgccgggatactcgtttaacgcccat ctggtggcgggtttaacgccgattgaggccaacggttatctcgatttttttatcga ccgaccgctgggaatgaaaggttatattctcaatctcaccattcgcggtcaggg ggtggtgaaaaatcagggacgagaatttgtctgccgaccgggtgatattttgctg ttcccgccaggagagattcatcactacggtcgtcatccggaggctcgcgaatg gtatcaccagtgggtttactttcgtccgcgcgcctactggcatgaatggcttaact ggccgtcaatatttgccaatacgggtttctttcgcccggatgaagcgcaccagc cgcatttcagcgacctgtttgggcaaatcattaacgccgggcaaggggaaggg cgctattcggagctgctggcgataaatctgcttgagcaattgttactgcggcgca tggaagcgattaacgagtcgctccatccaccgatggataatcgggtacgcgag gcttgtcagtacatcagcgatcacctggcagacagcaattttgatatcgccagcg tcgcacagcatgtttgcttgtcgccgtcgcgtctgtcacatcttttccgccagcag ttagggattagcgtcttaagctggcgcgaggaccaacgcattagtcaggcgaa gctgcttttgagcactacccggatgcctatcgccaccgtcggtcgcaatgttggt tttgacgatcaactctatttctcgcgagtatttaaaaaatgcaccggggccagccc gagcgagtttcgtgccggttgtgaagaaaaagtgaatgatgtagccgtcaagtt gtcataa	
araC (Truncated)		atggctgaagcgcaaaatgatcccctgctgccgggatactcgtttaacgcccat ctggtggcgggtttaacgccgattgaggccaacggttatctcgatttttttatcga ccgaccgctgggaatgaaaggttatattctcaatctcaccattcgcggtcaggg ggtggtgaaaaatcagggacgagaatttgtctgccgaccgggtgatattttgctg ttcccgccaggagagattcatcactacggtcgtcatccggaggctcgcgaatg gtatcaccagtgggtttactttcgtccgcgcgcctactggcatgaatggcttaact ggccgtcaatatttgccaatacgggtttctttcgcccggatgaagcgcaccagc cgcatttcagcgacctgtttgggcaaatcattaacgccgggcaaggggaaggg cgctattcggagctgctggcgataaatctgcttgagcaattgttactgcggcgca tggaagcgattaacgagtcgctccatccaccgatggataatcgggtacgcgag gcttgtcagtacatcagcgatcacctggcagacagcaattttgatatcgccagcg tcgcacagcatgtttgcttgtcgccgtcgcgtctgtcacatcttttccgccagcag ttagggattagcgtcttaagctggcgcgaggaccaacgcattagtcaggcgaa gctgcttttgagcactacccggatgcctatcgccaccgtcggtcgcaatgttggt tttgacgatcaactctatttctcgcgagtatttaaaaaatgcaccggggccagccc gagcgagtttcgtgccggttaa	
GFP	Enhanced Green Fluorescent Protein coding sequence (*76*)	atg**a**gtaaaggagaagaacttttcactggagttgtcccaattcttgttgaattagat ggtgatgttaatgggcacaaattttctgtcagtggagagggtgaaggtgatgca acatacggaaaacttacccttaaatttatttgcactactggaaaactacctgttcca tggccaacacttgtcactactttcggttatggtgttcaatgctttgcgagatactcca gatcatatgaaacagcatgactttttcaagagtgccatgcccgaaggttatgtac aggaaagaactatatttttcaaagatgacgggaactacaagacacgtgctgaag tcaagtttgaaggtgataccttgttaatagaatcgagttaaaaggtattgattttaa agaagatggaaacattcttggacacaaattggaatacaactataactcacacaat gtatacatcatggcagacaaacaaaagaatggaatcaaagttaacttcaaaatta gacacaacattgaagatggaagcgttcaactagcagaccattatcaacaaaata ctccaattggcgatggccctgtccttttaccagacaaccattacctgtccacacaa tctgccctttcgaaagatcccaacgaaaagagagaccacatggtccttcttgagt ttgtaacagctgctgggattacacatggcatggatgaactatacaaataa	Either a or c
LacI	LacI coding sequence (*2*)	gtgaaaccagtaacgttatacgatgtcgcagagtatgccggtgtctcttatcaga ccgtttcccgcgtggtgaaccaggccagccacgtttctgcgaaaacgcgggaa aaagtggaagcggcgatggcggagctgaattacattcccaaccgcgtggcac	

		aacaactggcgggcaaacagtcgttgctgattggcgttgccacctccagtctgg ccctgcacgcgccgtcgcaaattgtcgcggcgattaaatctcgcgccgatcaa ctgggtgccagcgtggtggtgtcgatggtagaacgaagcggcgtcgaagcct gtaaagcggcggtgcacaatcttctcgcgcaacgcgtcagtgggctgatcatta actatccgctggatgaccaggatgccattgctgtggaagctgcctgcactaatgt tccggcgttatttcttgatgtctctgaccagacacccatcaacagtattattttctcc catgaagacggtacgcgactgggcgtggagcatctggtcgcattgggtcacca gcaaatcgcgctgttagcgggcccattaagttctgtctcggcgcgtctgcgtctg gctggctggcataaatatctcactcgcaatcaaattcagccgatagcggaacgg gaaggcgactggagtgccatgtccggttttcaacaaaccatgcaaatgctgaat gagggcatcgttcccactgcgatgctggttgccaacgatcagatggcgctggg cgcaatgcgcgccattaccgagtccgggctgcgcgttggtgcggatatctcgg tagtgggatacgacgataccgaagacagctcatgttatatcccgccgttaacca ccatcaaacaggattttcgcctgctggggcaaaccagcgtggaccgcttgctgc aactctctcagggccaggcggtgaagggcaatcagctgttgcccgtctcactg gtgaaaagaaaaaccaccctggcgcccaatacgcaaaccgcctctccccgcg cgttggccgattcattaatgcagctggcacgacaggtttcccgactggaaagcg ggcagtga	
LuxR	LuxR coding sequence (BBa_C0062) (*77*), induced by AHL (3OC6HSL)	atgaaaaacataaatgccgacgacacatacagaataattaataaaattaaagctt gtagaagcaataatgatattaatcaatgcttatctgatatgactaaaatggtacatt gtgaatattatttactcgcgatcatttatcctcattctatggttaaatctgatatttcaat cctagataattaccctaaaaaatggaggcaatattatgatgacgctaatttaataa aatatgatcctatagtagattattctaactccaatcattcaccaattaattggaatata tttgaaaacaatgctgtaaataaaaaatctccaaatgtaattaaagaagcgaaaa catcaggtcttatcactgggtttagtttccctattcatacggctaacaatggcttcg gaatgcttagttttgcacattcagaaaaagacaactatatagatagtttatttttacat gcgtgtatgaacataccattaattgttccttctctagttgataattatcgaaaaataa atatagcaaataataaatcaaacaacgatttaaccaaaagagaaaaagaatgttt agcgtgggcatgcgaaggaaaaagctcttgggatatttcaaaaatattaggttgc agtgagcgtactgtcactttccatttaaccaatgcgcaaatgaaactcaatacaac aaaccgctgccaaagtatttctaaagcaattttaacaggagcaattgattgcccat actttaaaaattaataa	
mCherry	Red Fluorescent Protein coding sequence (*76*)	atggtgagcaagggcgaagaagataacatggccatcatcaaggagttcatgcg cttcaaggtgcacatggagggctccgtgaacggccacgagttcgagatcgag ggcgagggcgagggccgcccctacgagggcacccagaccgccaagctgaa ggtgaccaagggtggccccctgcccttcgcctgggacatcctgtcccctcagtt catgtacggctccaaggcctacgtgaagcaccccgccgacatccccgactact tgaagctgtccttccccgagggcttcaagtgggagcgcgtgatgaacttcgag gacggcggcgtggtgaccgtgacccaggactcctccctgcaggacggcgag ttcatctacaaggtgaagctgcgcggcaccaacttcccctccgacggccccgta atgcagaagaagaccatgggctgggaggcctcctccgagcggatgtaccccg aggacggcgccctgaagggcgagatcaagcagaggctgaagctgaaggac ggcggccactacgacgctgaggtcaagaccacctacaaggccaagaagccc gtgcagctgcccggcgcctacaacgtcaacatcaagttggacatcacctccca caacgaggactacaccatcgtggaacagtacgaacgcgccgagggccgcca ctccaccggcggcatggacgagctgtacaagtaa	
tetR	tetR coding sequence (*2*)	atgtccagattagataaaagtaaagtgattaacagcgcattagagctgcttaatg aggtcggaatcgaaggtttaacaacccgtaaactcgcccagaagctaggtgta gagcagcctacattgtattggcatgtaaaaaataagcgggctttgctcgacgcct tagccattgagatgttagataggcaccatactcacttttgcccttagaaggggaa agctggcaagatttttacgtaataacgctaaaagttttagatgtgctttactaagtc atcgcgatggagcaaaagtacatttaggtacacggcctacagaaaaacagtat gaaactctcgaaaatcaattagccttttatgccaacaaggtttttcactagagaat gcattatatgcactcagcgctgtggggcattttactttaggttgcgtattggaagat caagagcatcaagtcgctaaagaagaaagggaaacacctactactgatagtat gccgccattattacgacaagctatcgaattatttgatcaccaaggtgcagagcca	

		gccttcttattcggccttgaattgatcatatgcggattagaaaaacaacttaaatgt gaaagtgggtcctaa	
T7ptag	T7 Tag coding sequence (*26*)	atgattaccgtgcactagaataccattaacattgctaagaacgacttctctgacat cgaactggctgctatcccgttcaacactctggctgaccattacggtgagcgttta gctcgcgaacagttggcccttgagcatgagtcttacgagatgggtgaagcacg cttccgcaagatgtttgagcgtcaacttaaagctggtgaggttgcggataacgct gccgccaagcctctcatcactaccctactccctaagatgattgcacgcatcaac gactggtttgaggaagtgaaagctaagcgcggcaagcgcccgacagccttcc agttcctgtaggaaatcaagccggaagccgtagcgtacatcaccattaagacca ctctggcttgcctaaccagtgctgacaatacaaccgttcaggctgtagcaagcg caatcggtcgggccattgaggacgaggctcgcttcggtcgtatccgtgaccttg aagctaagcacttcaagaaaaacgttgaggaacaactcaacaagcgcgtagg gcacgtctacaagaaagcatttatgcaagttgtcgaggctgacatgctctctaag ggtctactcggtggcgaggcgtggtcttcgtggcataaggaagactctattcatg taggagtacgctgcatcgagatgctcattgagtcaaccggaatggttagcttaca ccgccaaaatgctggcgtagtaggtcaagactctgagactatcgaactcgcac ctgaatacgctgaggctatcgcaacccgtgcaggtgcgctggctggcatctctc cgatgttccaaccttgcgtagttcctcctaagccgtggactggcattactggtggt ggctattgggctaacggtcgtcgtcctctggcgctggtgcgtactcacagtaag aaagcactgatgcgctacgaagacgtttacatgcctgaggtgtacaaagcgatt aacattgcgcaaaacaccgcatggaaaatcaacaagaaagtcctagcggtcgc caacgtaatcaccaagtggaagcattgtccggtcgaggacatccctgcgattga gcgtgaagaactcccgatgaaaccggaagacatcgacatgaatcctgaggctc tcaccgcgtggaaacgtgctgccgctgctgtgtaccgcaaggacaaggctcgc aagtctcgccgtatcagccttgagttcatgcttgagcaagccaataagtttgctaa ccataaggccatctggttcccttacaacatggactggcgcggtcgtgtttacgct gtgtcaatgttcaacccgcaaggtaacgatatgaccaaaggactgcttacgctg gcgaaaggtaaaccaatcggtaaggaaggttactactggctgaaaatccacgg tgcaaactgtgcgggtgtcgataaggttccgttccctgagcgcatcaagttcatt gaggaaaaccacgagaacatcatggcttgcgctaagtctccactggagaacac ttggtgggctgagcaagattctccgttctgcttccttgcgttctgctttgagtacgct ggggtacagcaccacggcctgagctataactgctcccttccgctggcgtttgac gggtcttgctctggcatccagcacttctccgcgatgctccgagatgaggtaggt ggtcgcgcggttaacttgcttcctagtgaaaccgttcaggacatctacgggattg ttgctaagaaagtcaacgagattctacaagcagacgcaatcaatgggaccgata acgaagtagttaccgtgaccgatgagaacactggtgaaatctctgagaaagtca agctgggcactaaggcactggctggtcaatggctggcttacggtgttactcgca gtgtgactaagcgttcagtcatgacgctggcttacgggtccaaagagttcggctt ccgtcaacaagtgctggaagataccattcagccagctattgattccggcaaggg tctgatgttcactcagccgaatcaggctgctggatacatggctaagctgatttgg gaatctgtgagcgtgacggtggtagctgcggttgaagcaatgaactggcttaag tctgctgctaagctgctggctgctgaggtcaaagataagaagactggagagatt cttcgcaagcgttgcgctgtgcattgggtaactcctgatggtttccctgtgtggca ggaatacaagaagcctattcagacgcgcttgaacctgatgttcctcggtcagttc cgcttacagcctaccattaacaccaacaaagatagcgagattgatgcacacaaa caggagtctggtatcgctcctaactttgtacacagccaagacggtagccaccttc gtaagactgtagtgtgggcacacgagaagtacggaatcgaatcttttgcactgat tcacgactccttcggtaccattccggctgacgctgcgaacctgttcaaagcagtg cgcgaaactatggttgacacatatgagtcttgtgatgtactggctgatttctacga ccagttcgctgaccagttgcacgagtctcaattggacaaaatgccagcacttcc ggctaaaggtaacttgaacctccgtgacatcttagagtcggacttcgcgttcgca taa	
SupD	SupD-tRNA coding sequence	caattcggagagatgccggagcggctgaacggaccggtctctaaaaccggag taggggcaactctaccgggggttcaaatccccctctctccgccactacagatcc ttagcgaaagctaaggatttttttaagct	

	(BBa_K228001) (*26*)		
LAA + stop codon		aggcctgcagcaaacgacgaaaactacgctttagcagcttaa	
LVA + stop codon		aggcctgctgcaaacgacgaaaactacgctttagtagcttaa	
P_{BAD}	*araBAD* promoter (*14*)	aagaaaccaattgtccatattgcatcagacattgccgtcactgcgtcttttactgg ctcttctcgctaaccaaaccggtaaccccgcttattaaaagcattctgtaacaaag cgggaccaaagccatgacaaaaacgcgtaacaaaagtgtctataatcacggca gaaaagtccacattgattatttgcacggcgtcacactttgctatgccatagcatttttt atccataagattagcggatcctacctgacgctttttatcgcaactctctactgtttct ccat	
$P_{J231119}$		TTGACAGCTAGCTCAGTCCTAGGTATAATACTA GT	
LKsg3		aattcgctagcccaaaaaaa	
SG6		gagttgcgataaaaagcgtc	
gRNA		GTTTTAGAGCTAGAAATAGCAAgttaaaataagGCTA GTCCGTTATCAACTTGAAAAAGTGGCACCGAGT CGGTGC	
dCas9		ATGGATAAGAAATACTCAATAGGCTTAGCTATC GGCACAAATAGCGTCGGATGGGCGGTGATCACT GATGAATATAAGGTTCCGTCTAAAAAGTTCAAG GTTCTGGGAAATACAGACCGCCACAGTATCAAA AAAAATCTTATAGGGGCTCTTTTATTTGACAGTG GAGAGACAGCGGAAGCGACTCGTCTCAAACGG ACAGCTCGTAGAAGGTATACACGTCGGAAGAAT CGTATTTGTTATCTACAGGAGATTTTTTCAAATG AGATGGCGAAAGTAGATGATAGTTTCTTTCATC GACTTGAAGAGTCTTTTTTGGTGGAAGAAGACA AGAAGCATGAACGTCATCCTATTTTTGGAAATA TAGTAGATGAAGTTGCTTATCATGAGAAATATC CAACTATCTATCATCTGCGAAAAAAATTGGTAG ATTCTACTGATAAAGCGGATTTGCGCTTAATCTA TTTGGCCTTAGCGCATATGATTAAGTTTCGTGGT CATTTTTTGATTGAGGGAGATTTAAATCCTGATA ATAGTGATGTGGACAAACTATTTATCCAGTTGGT ACAAACCTACAATCAATTATTTGAAGAAAACCC TATTAACGCAAGTGGAGTAGATGCTAAAGCGAT TCTTTCTGCACGATTGAGTAAATCAAGACGATTA GAAAATCTCATTGCTCAGCTCCCCGGTGAGAAG AAAAATGGCTTATTTGGGAATCTCATTGCTTTGT CATTGGGTTTGACCCCTAATTTTAAATCAAATTT TGATTTGGCAGAAGATGCTAAATTACAGCTTTC AAAAGATACTTACGATGATGATTTAGATAATTT ATTGGCGCAAATTGGAGATCAATATGCTGATTT GTTTTTGGCAGCTAAGAATTTATCAGATGCTATT TTACTTTCAGATATCCTAAGAGTAAATACTGAA ATAACTAAGGCTCCCCTATCAGCTTCAATGATTA AACGCTACGATGAACATCATCAAGACTTGACTC TTTTAAAAGCTTTAGTTCGACAACAACTTCCAGA AAAGTATAAAGAAATCTTTTTTGATCAATCAAA AAACGGATATGCAGGTTATATTGATGGGGGAGC TAGCCAAGAAGAATTTTATAAATTTATCAAACC AATTTTAGAAAAAATGGATGGTACTGAGGAATT ATTGGTGAAACTAAATCGTGAAGATTTGCTGCG	

		CAAGCAACGGACCTTTGACAACGGCTCTATTCC CCATCAAATTCACTTGGGTGAGCTGCATGCTATT TTGAGAAGACAAGAAGACTTTTATCCATTTTTAA AAGACAATCGTGAGAAGATTGAAAAAATCTTGA CTTTTCGAATTCCTTATTATGTTGGTCCATTGGC GCGTGGCAATAGTCGTTTTGCATGGATGACTCG GAAGTCTGAAGAAACAATTACCCCATGGAATTT TGAAGAAGTTGTCGATAAAGGTGCTTCAGCTCA ATCATTTATTGAACGCATGACAAACTTTGATAA AAATCTTCCAAATGAAAAAGTACTACCAAAACA TAGTTTGCTTTATGAGTATTTTACGGTTTATAAC GAATTGACAAAGGTCAAATATGTTACTGAAGGA ATGCGAAAACCAGCATTTCTTTCAGGTGAACAG AAGAAAGCCATTGTTGATTTACTCTTCAAAACA AATCGAAAAGTAACCGTTAAGCAATTAAAAGAA GATTATTTCAAAAAAATAGAATGTTTTGATAGT GTTGAAATTTCAGGAGTTGAAGATAGATTTAAT GCTTCATTAGGTACCTACCATGATTTGCTAAAAA TTATTAAAGATAAAGATTTTTTGGATAATGAAG AAAATGAAGATATCTTAGAGGATATTGTTTTAA CATTGACCTTATTTGAAGATAGGGAGATGATTG AGGAAAGACTTAAAACATATGCTCACCTCTTTG ATGATAAGGTGATGAAACAGCTTAAACGTCGCC GTTATACTGGTTGGGGACGTTTGTCTCGAAAATT GATTAATGGTATTAGGGATAAGCAATCTGGCAA AACAATATTAGATTTTTTGAAATCAGATGGTTTT GCCAATCGCAATTTTATGCAGCTGATCCATGATG ATAGTTTGACATTTAAAGAAGACATTCAAAAAG CACAAGTGTCTGGACAAGGCGATAGTTTACATG AACATATTGCAAATTTAGCTGGTAGCCCTGCTAT TAAAAAAGGTATTTTACAGACTGTAAAAGTTGT TGATGAATTGGTCAAAGTAATGGGGCGGCATAA GCCAGAAAATATCGTTATTGAAATGGCACGTGA AAATCAGACAACTCAAAAGGGCCAGAAAAATTC GCGAGAGCGTATGAAACGAATCGAAGAAGGTA TCAAAGAATTAGGAAGTCAGATTCTTAAAGAGC ATCCTGTTGAAAATACTCAATTGCAAAATGAAA AGCTCTATCTCTATTATCTCCAAAATGGAAGAG ACATGTATGTGGACCAAGAATTAGATATTAATC GTTTAAGTGATTATGATGTCGATGCCATTGTTCC ACAAAGTTTCCTTAAAGACGATTCAATAGACAA TAAGGTCTTAACGCGTTCTGATAAAAATCGTGG TAAATCGGATAACGTTCCAAGTGAAGAAGTAGT CAAAAAGATGAAAAACTATTGGAGACAACTTCT AAACGCCAAGTTAATCACTCAACGTAAGTTTGA TAATTTAACGAAAGCTGAACGTGGAGGTTTGAG TGAACTTGATAAAGCTGGTTTTATCAAACGCCA ATTGGTTGAAACTCGCCAAATCACTAAGCATGT GGCACAAATTTTGGATAGTCGCATGAATACTAA ATACGATGAAAATGATAAACTTATTCGAGAGGT TAAAGTGATTACCTTAAAATCTAAATTAGTTTCT GACTTCCGAAAAGATTTCCAATTCTATAAAGTA CGTGAGATTAACAATTACCATCATGCCCATGAT GCGTATCTAAATGCCGTCGTTGGAACTGCTTTGA TTAAGAAATATCCAAAACTTGAATCGGAGTTTG TCTATGGTGATTATAAAGTTTATGATGTTCGTAA	

		AATGATTGCTAAGTCTGAGCAAGAAATAGGCAA AGCAACCGCAAAATATTTCTTTTACTCTAATATC ATGAACTTCTTCAAAACAGAAATTACACTTGCA AATGGAGAGATTCGCAAACGCCCTCTAATCGAA ACTAATGGGGAAACTGGAGAAATTGTCTGGGAT AAAGGGCGAGATTTTGCCACAGTGCGCAAAGTA TTGTCCATGCCCCAAGTCAATATTGTCAAGAAA ACAGAAGTACAGACAGGCGGATTCTCCAAGGAG TCAATTTTACCAAAAAGAAATTCGGACAAGCTT ATTGCTCGTAAAAAAGACTGGGATCCAAAAAAA TATGGTGGTTTTGATAGTCCAACGGTAGCTTATT CAGTCCTAGTGGTTGCTAAGGTGGAAAAAGGGA AATCGAAGAAGTTAAAATCCGTTAAAGAGTTAC TAGGGATCACAATTATGGAAAGAAGTTCCTTTG AAAAAAATCCGATTGACTTTTTAGAAGCTAAAG GATATAAGGAAGTTAAAAAAGACTTAATCATTA AACTACCTAAATATAGTCTTTTTGAGTTAGAAAA CGGTCGTAAACGGATGCTGGCTAGTGCCGGAGA ATTACAAAAAGGAAATGAGCTGGCTCTGCCAAG CAAATATGTGAATTTTTTATATTTAGCTAGTCAT TATGAAAAGTTGAAGGGTAGTCCAGAAGATAAC GAACAAAAACAATTGTTTGTGGAGCAGCATAAG CATTATTTAGATGAGATTATTGAGCAAATCAGT GAATTTTCTAAGCGTGTTATTTTAGCAGATGCCA ATTTAGATAAAGTTCTTAGTGCATATAACAAAC ATAGAGACAAACCAATACGTGAACAAGCAGAA AATATTATTCATTTATTTACGTTGACGAATCTTG GAGCTCCCGCTGCTTTTAAATATTTTGATACAAC AATTGATCGTAAACGATATACGTCTACAAAAGA AGTTTTAGATGCCACTCTTATCCATCAATCCATC ACTGGTCTTTATGAAACACGCATTGATTTGAGTC AGCTAGGAGGTGAC	
P_{BAD_RD1}		atagcatttttatccataagattagcggatcctacctgacgctttttatcgcaactct ctactgtttctccataccgtttttttgggctagc	
ExsA	ExsA gene (*48*)	atgcaaggagccaaatctcttggccgaaagcagataacgtcttgtcattggaac attccaactttcgaatacagggtaaacaaggaagagggcgtatatgttctgctcg agggcgaactgaccgtccaggacatcgattccacttttgcctggcgcctggcg agttgcttttcgtccgccgcggaagctatgtcgtaagtaccaagggaaaggaca gccgaatactctggattccattatctgcccagtttctacaaggcttcgtccagcgc ttcggcgcgctgttgagtgaagtcgagcgttgcgacgagcccgtgccgggcat catcgcgttcgctgccacgcctctgctggccggttgcgtcaaggggttgaagg aattgcttgtgcatgagcatccgccgatgctcgcctgcctgaagatcgaggagtt gctgatgctcttcgcgttcagtccgcaggggccgctgctgatgtcggtcctgcg gcaactgagcaaccggcatgtcgagcgtctgcagctattcatggagaagcact acctcaacgagtggaagctgtccgacttctcccgcgagttcggcatggggctg accaccttcaaggagctgttcggcagtgtctatggggtttcgccgcgcgcctgg atcagcgagcggagaatcctctatgcccatcagttgctgctcaacagcgacatg agcatcgtcgacatcgccatggaggcgggcttttccagtcagtcctatttcaccc agagctatcgccgccgtttcggctgcacgccgagccgctcgcggcaggggaa ggacgaatgccgggctaaaaataactga	
ExsD	ExsD gene (*48*)	atggagcaggaagacgataagcagtactcccgagaagcggtgttcgctggca ggcgggtatccgtggtgggctcggacgcccgctcgcggggtcgggtgccgg gttacgcatcgagcagtttgtatcgtgagtccggaatcatcagtgcgcggcaact ggcgttgctgcagcggatgctgccgcgcctgcggctggagcaactgttccgct gcgagtggttgcagcagcgcctggcgcgcggcctggcgctggggcgcgaa	

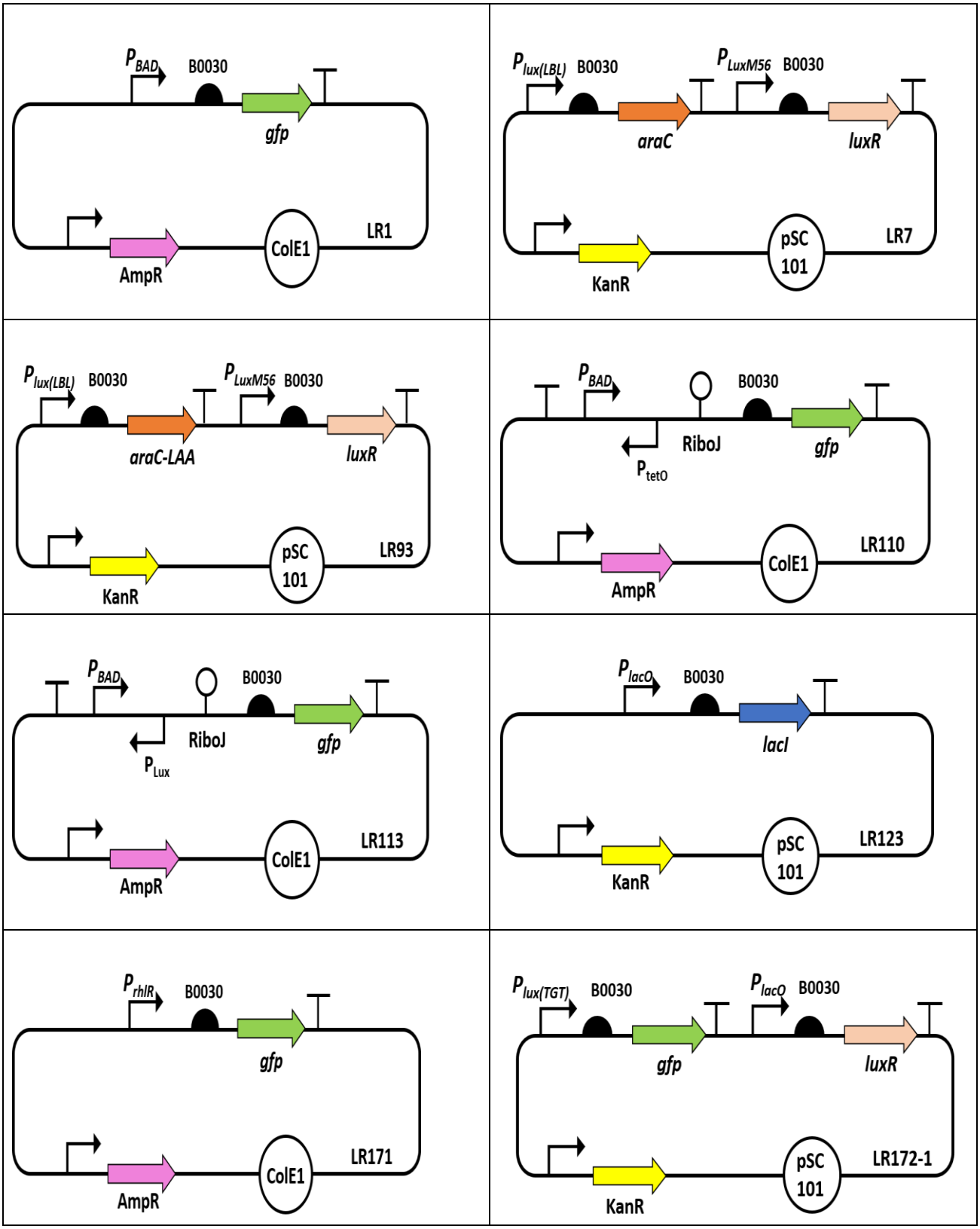
		gaggtgcggcagattctcctctgcgcggcgcaggacgacgacggctggtgct ccgaactgggcgaccgggtcaacctcgccgtgccgcagtcgatgatcgactg ggtcctgctgccggtctatggctggtgggaaagcctgctcgaccaggcgatcc ccggctggcgcctgtcgctggtggagctggagacccagtcccggcaactgcg agtcaagtccgaattctggtcccgcgtggccgagctggagccggagcaggcc cgcgaggaactggccagggtcgccaagtgccaggcgcgcacccaggaaca ggtggccgaactggccggcaagctggagacggcttcggcactggcgaagag cgcctggccgaactggcagcggggcatggcgacgctgctcgccagcggcg ggctggccggcttcgagccgatccccgaggtcctcgaatgcctctggcaacct ctctgccggctggacgacgacgtcggcgcggcggacgccgtccaggcctgg ctgcacgaacgcaacctgtgccaggcacaggatcacttctactggcagagctg a	
pexsD	Promoter (*48*)	gaaggacgaatgccgggctaaaaataactgacgtttttgaaagcccggtagc ggctgcatgagtagaatcggcccaaat	
P_{lacO}	PLlacO-1 promoter (*2*)	**aattgtgagcggataacaa**ttgacattgtgagcggataacaagatactgagca catcagcaggacgcactgacc	
$P_{laco/teto}$		tacaacgtcgtgttaaattgtgagcggataacaatttagttgacatttatgcttccg gctcgtataattccacccctatcagtgatagagagcgttacccaac	
P_{lacO1}		ttgacattgtgagcggataacaagatactgagcacatcagcaggacgcactga cc	PLlacO-1_Deleting 1 binding site
P_{lux}	Lux promoter, BBa_R0062 (*77*)	**acct**gtaggatcgtacaggtttacgcaagaaaatggtttgtta**tag**tcgaataaa	
$P_{lux(AAT)}$		**aatt**gtaggatcgtacaggtttacgcaagaaaatggtttgttatagtcgaataaa	The "**acct**" of *Plux* (in bold) was mutated to "**aatt**" by site directed Mutagenesis.
$P_{lux(TGT)}$		**tgtt**gtaggatcgtacaggtttacgcaagaaaatggtttgttatagtcgaataaa	The "**acct**" of *Plux* (in bold) was mutated to "**tgtt**" by site directed Mutagenesis.
$P_{lux(LBL)}$		acctgtaggatcgtacaggtttacgcaagaaaatggtttgtta**ctt**tcgaataaa	The "**tag**" of *Plux* (in bold) was mutated to "**ctt**" by site directed Mutagenesis.
P_{luxM56}		**tggg**gtaggatcgtacaggtttacgcaagaaaatggtttgttatagtcgaataaa	The "**acct**" of *Plux* (in bold) was mutated to "**TGGG**" by site directed Mutagenesis.
$p_{lux/laco}$		acctgtaggatcgtacaggtttacttgtgagcggataacaatatagtgtgtggaat tgtgagcggataacaatt	
$p_{lux/teto}$		acctgtaggatcgtacaggtttacgcaagaaaatggtttgttatagtcgaatatcc ctatcagtgatagaga	
P_{rhlR}		tcctgtgaaatctggcagttaccgttagctttcgaattggctaaaaagtgttc	

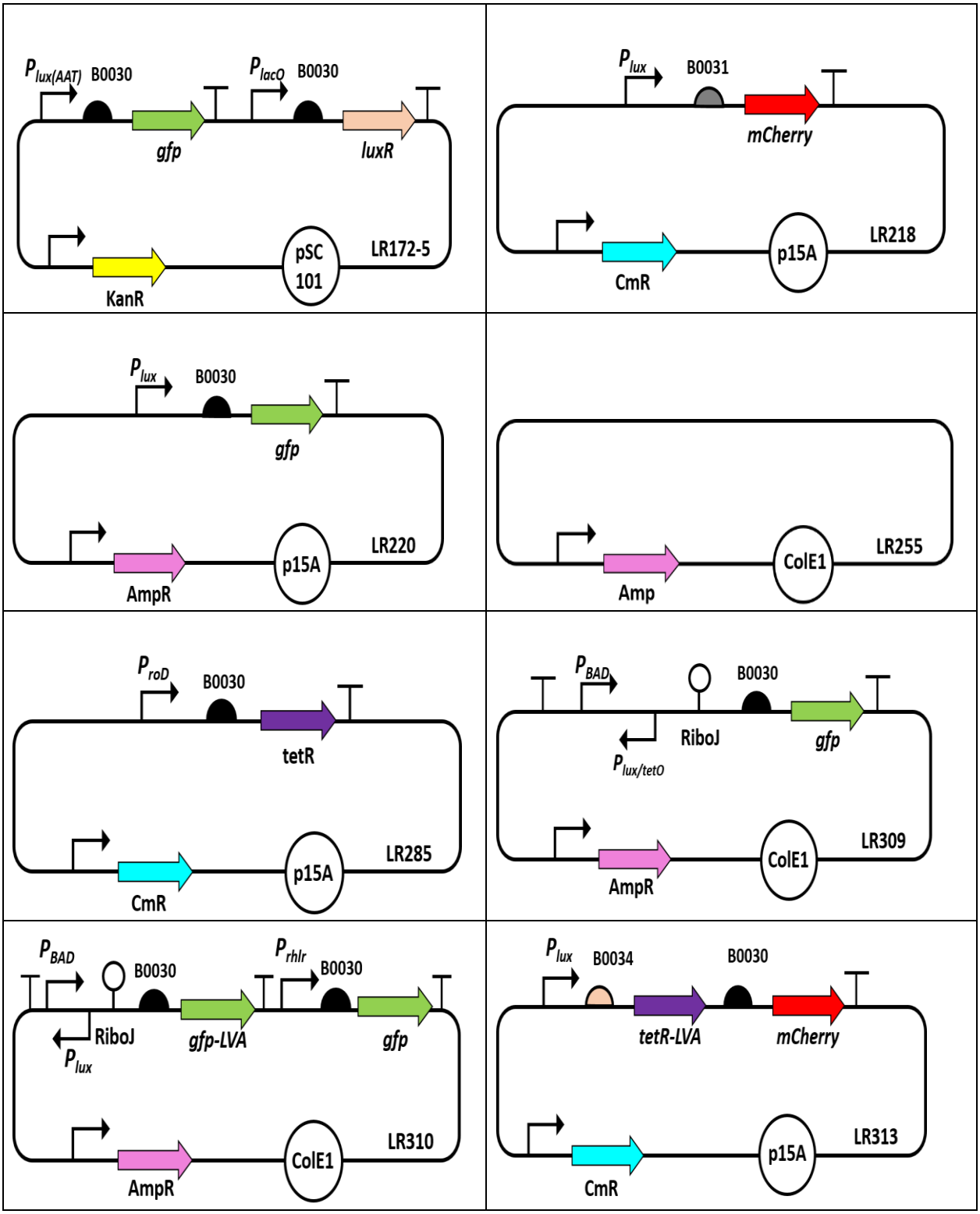
P_{teto}	PLtetO-1 promoter (*2*)	tccctatcagtgatagagattgacatccctatcagtgatagagatactgagcacat cagcaggacgcactgacc	
P_{teto*}	PLtetO* promoter (*48*)	ttttcagcaggacgcactgacctccctatcagtgatagagattgacatccctatca gtgatagagatactgagcacatat	
PT7		taatacgactcactatagggaga	
RBS1	BBa_B0030	attaaagaggagaaa	
RBS2	BBa_B0031	tcacacaggaaacc	
RBS3	BBa_B0034	aaagaggagaaa	
RiboJ	sTRSV HHRz	agctgtcaccggatgtgctttccggtctgatgagtccgtgaggacgaaacagcc tctacaaataattttgtttaa	
rrnB T1 terminator	transcription terminator T1 from the *E.coli* rrnB gene	caaataaaacgaaaggctcagtcgaaagactgggcctttcgttttatctgttgttt gtcggtgaacgctctcctgagtaggacaaat	
p15A	medium-copy-number p15A origin of replication	cggtcgttcgactgcggcgagcggaaatggcttacgaacggggcggagattt cctggaagatgccaggaagatacttaacagggaagtgagagggccgcggca aagccgtttttccataggctccgccccctgacaagcatcacgaaatctgacgct caaatcagtggtggcgaaacccgacaggactataaagataccaggcgtttccc cctggcggctccctcgtgcgctctcctgttcctgcctttcggtttaccggtgtcatt ccgctgttatggccgcgtttgtctcattccacgcctgacactcagttccgggtag gcagttcgctccaagctggactgtatgcacgaacccccccgttcagtccgaccg ctgcgccttatccggtaactatcgtcttgagtccaacccggaaagacatgcaaa agcaccactggcagcagccactggtaattgatttagaggagttagtcttgaagtc atgcgccggttaaggctaaactgaaaggacaagttttggtgactgcgctcctcc aagccagttacctcggttcaaagagttggtagctcagagaaccttcgaaaaacc gccctgcaaggcggtttttcgttttcagagcaagagattacgcgcagaccaaa acgatctcaagaagatcatcttattaatcagataaaatatttctagatttcagtgcaa tttatctcttcaaatgtagcacctgaagtcagccccatacgatataagttgtt	
pSC101	Low-copy replication origin	gtacgggttttgctgcccgcaaacgggctgttctggtgttgctagtttgttatcaga atcgcagatccggcttcaggtttgccggctgaaagcgctatttcttccagaattgc catgatttttttccccacgggaggcgtcactggctcccgtgttgtcggcagctttga ttcgataagcagcatcgcctgtttcaggctgtctatgtgtgactgttgagctgtaac aagttgtctcaggtgttcaatttcatgttctagttgctttgttttactggtttcacctgtt ctattaggtgttacatgctgttcatctgttacattgtcgatctgttcatggtgaacag ctttaaatgcaccaaaaactcgtaaaagctctgatgtatctatctttttacaccgttt tcatctgtgcatatggacagttttcccttgatatctaacggtgaacagttgttctact tttgtttgttagtcttgatgcttcactgatagatacaagagccataagaacctcaga tccttccgtatttagccagtatgttctctagtgtggttcgttgtttttgcgtgagccat gagaacgaaccattgagatcatgcttactttgcatgtcactcaaaaattttgcctca aaactggtgagctgaatttttgcagttaaagcatcgtgtagtgtttttcttagtccgtt acgtaggtaggaatctgatgtaatggttgttggtattttgtcaccattcatttttatct ggttgttctcaagttcggttacgagatccatttgtctatctagttcaacttggaaaat caacgtatcagtcgggcggcctcgcttatcaaccaccaatttcatattgctgtaag tgtttaaatctttacttattggtttcaaaacccattggttaagccttttaaactcatggt agttatttcaagcattaacatgaacttaaattcatcaaggctaatctctatatttgcc ttgtgagttttcttttgtgttagttcttttaataaccactcataaatcctcatagagtattt gttttcaaaagacttaacatgttccagattatattttatgaatttttttaactggaaaag ataaggcaatatctcttcactaaaaactaattctaatttttcgcttgagaacttggca tagtttgtccactggaaaatctcaaagcctttaaccaaaggattcctgatttccaca gttctcgtcatcagctctctggttgctttagctaatacaccataagcattttccctact gatgttcatcatctgagcgtattggttataagtgaacgataccgtccgttctttccttt gtagggttttcaatcgtggggttgagtagtgccacacagcataaaattagcttggt ttcatgctccgttaagtcatagcgactaatcgctagttcatttgctttgaaaacaact	

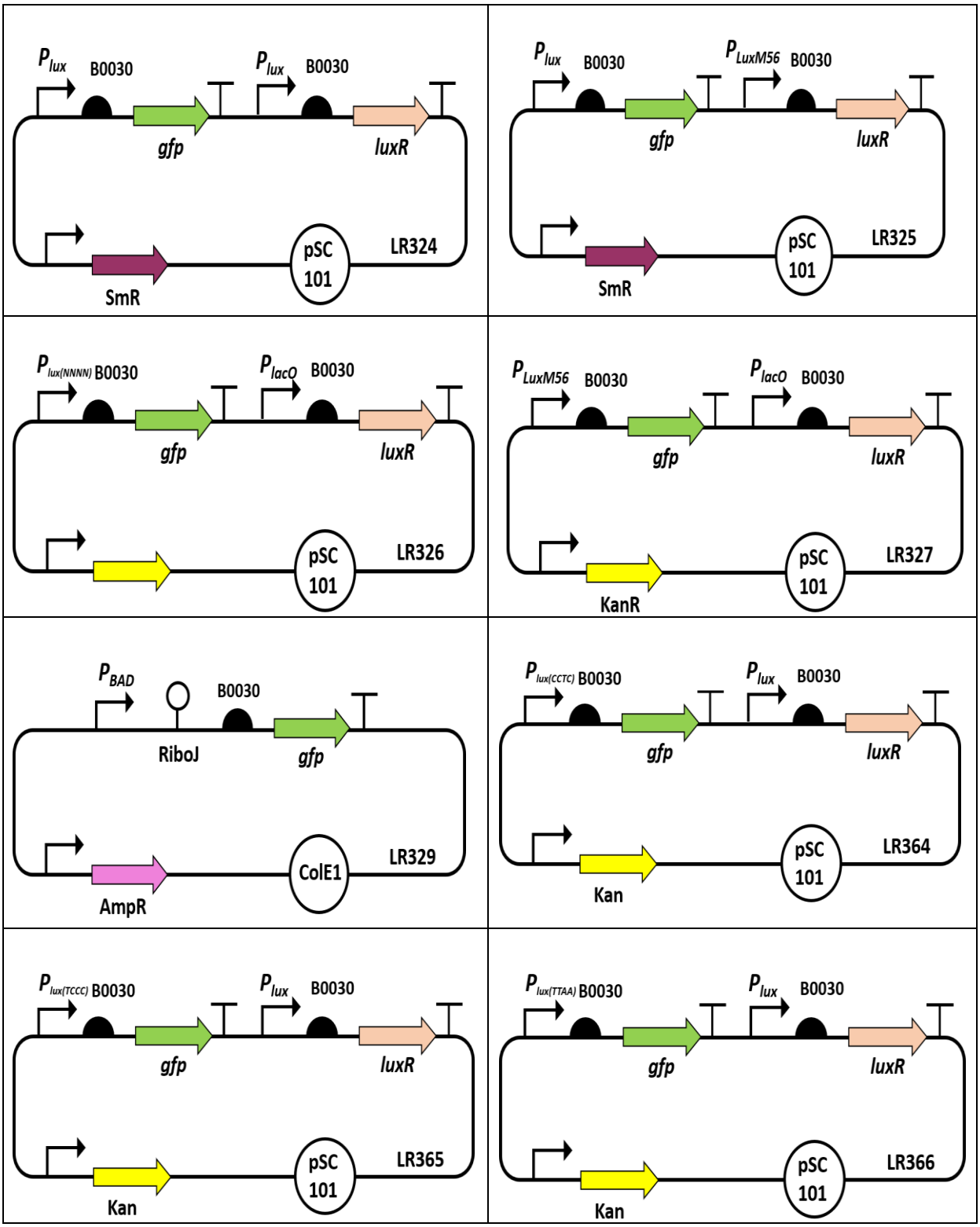
		aattcagacatacatctcaattggtctaggtgattttaatcactataccaattgagat gggctagtcaatgataattactagtccttttcctttgagttgtgggtatctgtaaattc tgctagacctttgctggaaaacttgtaaattctgctagaccctctgtaaattccgct agacctttgtgtgttttttttgtttatattcaagtggttataatttatagaataaagaaa gaataaaaaaagataaaaagaatagatcccagccctgtgtataactcactacttt agtcagttccgcagtattacaaaaggatgtcgcaaacgctgtttgctcctctacaa aacagaccttaaaaccctaaaggcttaagtagcaccctcgcaagctcgggcaa atcgctgaatattccttttgtctccgaccatcaggcacctgagtcgctgtctttttcg tgacattcagttcgctgcgctcacggctctggcagtgaatgggggtaaatggca ctacaggcgccttttatggattcatgcaaggaaactacccataatacaagaaaag cccgtcacgggcttctcagggcgttttatggcgggtctgctatgtggtgctatctg actttttgctgttcagcagttcctgccctctgattttccagtctgaccacttcggatta tcccgtgacaggtcattcagactggctaatgcacccagtaaggcagcggtatca tcaacaggcttacccgtcttactgtccctagt	
ColE1	High-copy replication origin	cgttcggctgcggcgagcggtatcagctcactcaaaggcggtaatacggttatc cacagaatcaggggataacgcaggaaagaacatgtgagcaaaaggccagca aaaggccaggaaccgtaaaaaggccgcgttgctggcgtttttccataggctcc gccccccctgacgagcatcacaaaaatcgacgctcaagtcagaggtggcgaaa cccgacaggactataaagataccaggcgtttcccccctggaagctccctcgtgc gctctcctgttccgaccctgccgcttaccggatacctgtccgcctttctcccttcg ggaagcgtggcgctttctcaatgctcacgctgtaggtatctcagttcggtgtagg tcgttcgctccaagctgggctgtgtgcacgaaccccccgttcagcccgaccgct gcgccttatccggtaactatcgtcttgagtccaacccggtaagacacgacttatc gccactggcagcagccactggtaacaggattagcagagcgaggtatgtaggc ggtgctacagagttcttgaagtggtggcctaactacggctacactagaaggaca gtatttggtatctgcgctctgctgaagccagttaccttcggaaaaagagttggtag ctcttgatccggcaaacaaaccaccgctggtagcggtggttttttgtttgcaagc agcagattacgcgcagaaaaaaaggatctcaagaagatcctttgatcttttctac ggggtctgacgctcagtggaacgaaaactcacgttaagggattttggtcatg	
cmR		aaattacgccccgccctgccactcatcgcagtactgttgtaattcattaagcattct gccgacatggaagccatcacaaacggcatgatgaacctgaatcgccagcggc atcagcaccttgtcgccttgcgtataatatttgcccatggtgaaaacgggggcga agaagttgtccatattggccacgtttaaatcaaaactggtgaaactcacccaggg attggctgagacgaaaaacatattctcaataaaccctttagggaaataggccag gttttcaccgtaacacgccacatcttgcgaatatatgtgtagaaactgccggaaat cgtcgtggtattcactccagagcgatgaaaacgtttcagtttgctcatggaaaac ggtgtaacaagggtgaacactatcccatatcaccagctcaccgtctttcattgcc atacgaaattccggatgagcattcatcaggcgggcaagaatgtgaataaaggc cggataaaacttgtgcttatttttctttacggtctttaaaaaggccgtaatatccagc tgaacggtctggttataggtacattgagcaactgactgaaatgcctcaaaatgttc tttacgatgccattgggatatatcaacggtggtatatccagtgatttttttctccatttt agcttccttagctcctgaaaatctcgataactcaaaaaatacgcccggtagtgat cttatttcattatggtgaaagttggaacctcttacgtgcccgatcaa	
ampR		gtaaacttggtctgacagttaccaatgcttaatcagtgaggcacctatctcagcg atctgtctatttcgttcatccatagttgcctgactccccgtcgtgtagataactacga tacgggagggcttaccatctggccccagtgctgcaatgataccgcgagaccca cgctcaccggctccagatttatcagcaataaaccagccagccggaagggccg agcgcagaagtggtcctgcaactttatccgcctccatccagtctattaattgttgc cgggaagctagagtaagtagttcgccagttaatagtttgcgcaacgttgttgcca ttgctacaggcatcgtggtgtcacgctcgtcgtttggtatggcttcattcagctcc ggttcccaacgatcaaggcgagttacatgatcccccatgttgtgcaaaaaagcg gttagctccttcggtcctccgatcgttgtcagaagtaagttggccgcagtgttatc actcatggttatggcagcactgcataattctcttactgtcatgccatccgtaagatg cttttctgtgactggtgagtactcaaccaagtcattctgagaatagtgtatgcggc gaccgagttgctcttgcccggcgtcaatacgggataataccgcgccacatagc	

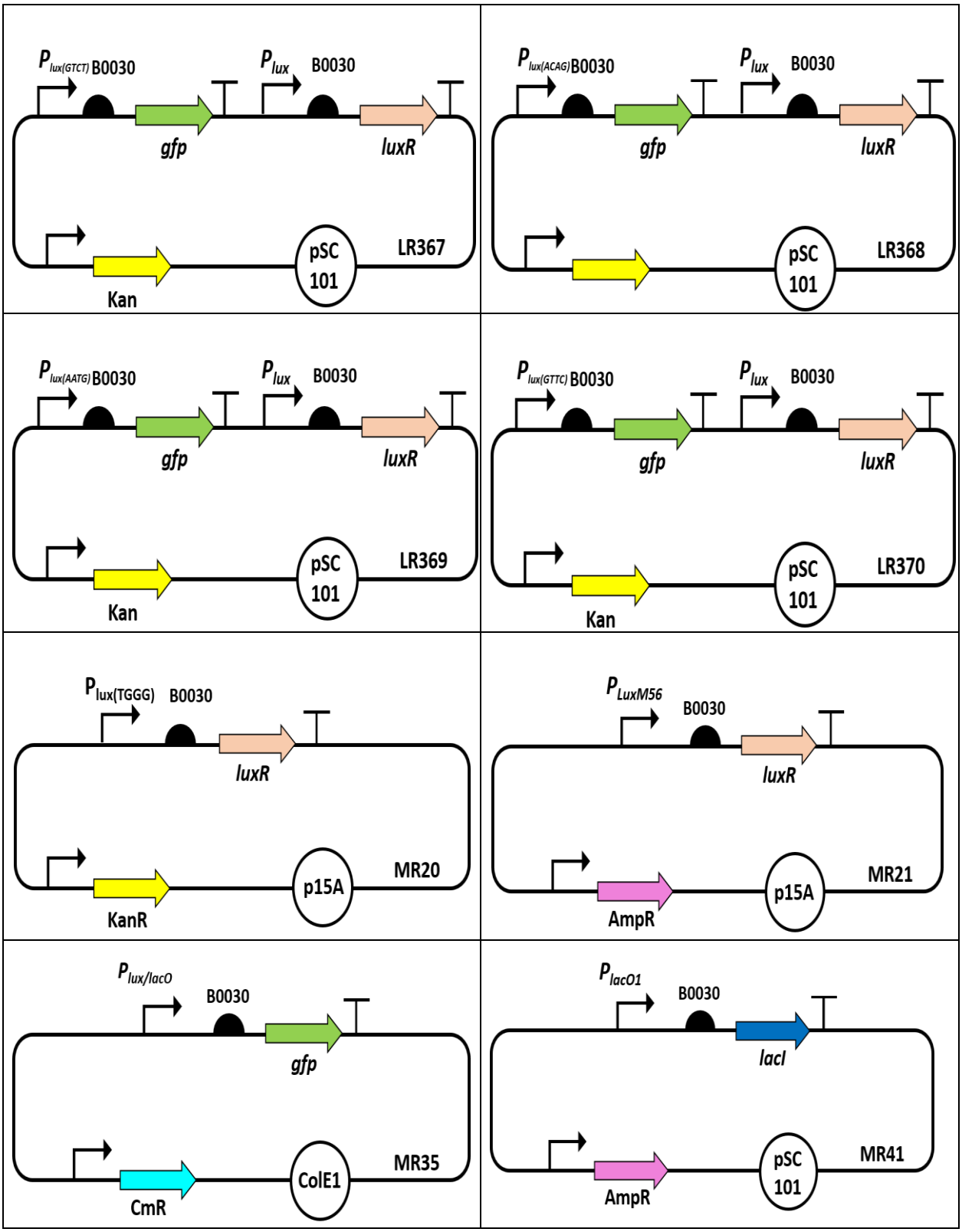
		agaactttaaaagtgctcatcattggaaaacgttcttcggggcgaaaactctcaa ggatcttaccgctgttgagatccagttcgatgtaacccactcgtgcacccaactg atcttcagcatcttttactttcaccagcgtttctgggtgagcaaaaacaggaaggc aaaatgccgcaaaaaagggaataagggcgacacggaaatgttgaatactcata ctcttccttttttcaatattattgaagcatttatcagggttattgtctcatgagcggata catatttgaatgtatttagaaaaataaacaaatagggttccgcgcacatttcccc gaaaagtgccacct	
kanR		tcgaaccccagagtcccgctcagaagaactcgtcaagaaggcgatagaaggc gatgcgctgcgaatcgggagcggcgataccgtaaagcacgaggaagcggtc agcccattcgccgccaagctcttcagcaatatcacgggtagccaacgctatgtc ctgatagcggtccgccacacccagccggccacagtcgatgaatccagaaaag cggccattttccaccatgatattcggcaagcaggcatcgccatgggtcacgacg agatcctcgccgtcgggcatgcgcgccttgagcctggcgaacagttcggctgg cgcgagcccctgatgctcttcgtccagatcatcctgatcgacaagaccggcttc catccgagtacgtgctcgctcgatgcgatgtttcgcttggtggtcgaatgggcag gtagccggatcaagcgtatgcagccgccgcattgcatcagccatgatggatact ttctcggcaggagcaaggtgagatgacaggagatcctgccccggcacttcgcc caatagcagccagtcccttcccgcttcagtgacaacgtcgagcacagctgcgc aaggaacgcccgtcgtggccagccacgatagccgcgctgcctcgtcctgcag ttcattcagggcaccggacaggtcggtcttgacaaaaagaaccgggcgcccct gcgctgacagccggaacacggcggcatcagagcagccgattgtctgttgtgc ccagtcatagccgaatagcctctccacccaagcggccggagaacctgcgtgc aatccatcttgttcaatcatgcgaaacgatcctcatcctgtctcttgatcagatcttg atcccctgcgccatcagatccttggcggcaagaaagccatccagtttactttgca gggcttcccaaccttaccagagggcgccccagctggcaattcc	

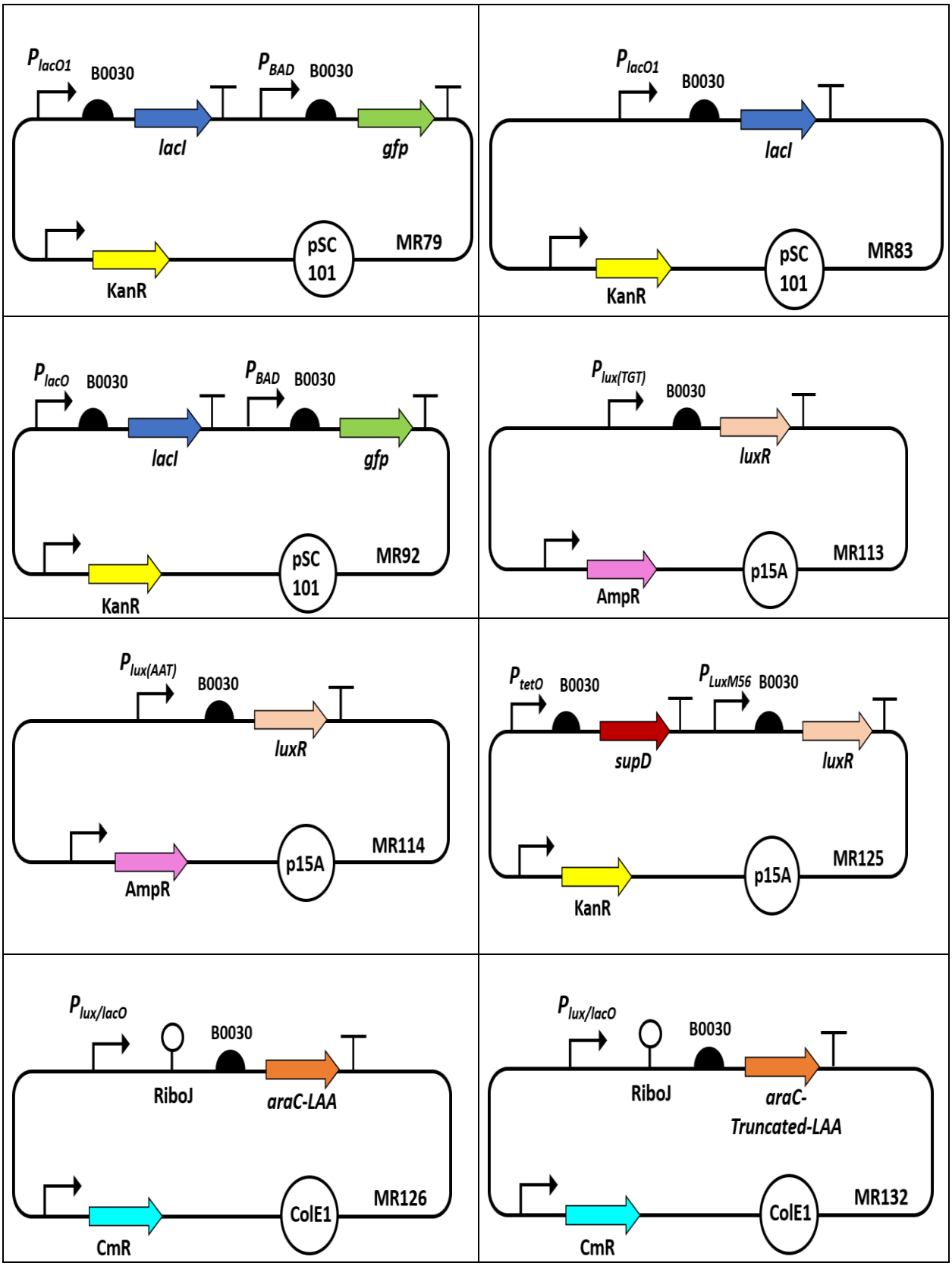
16. Plasmid Maps

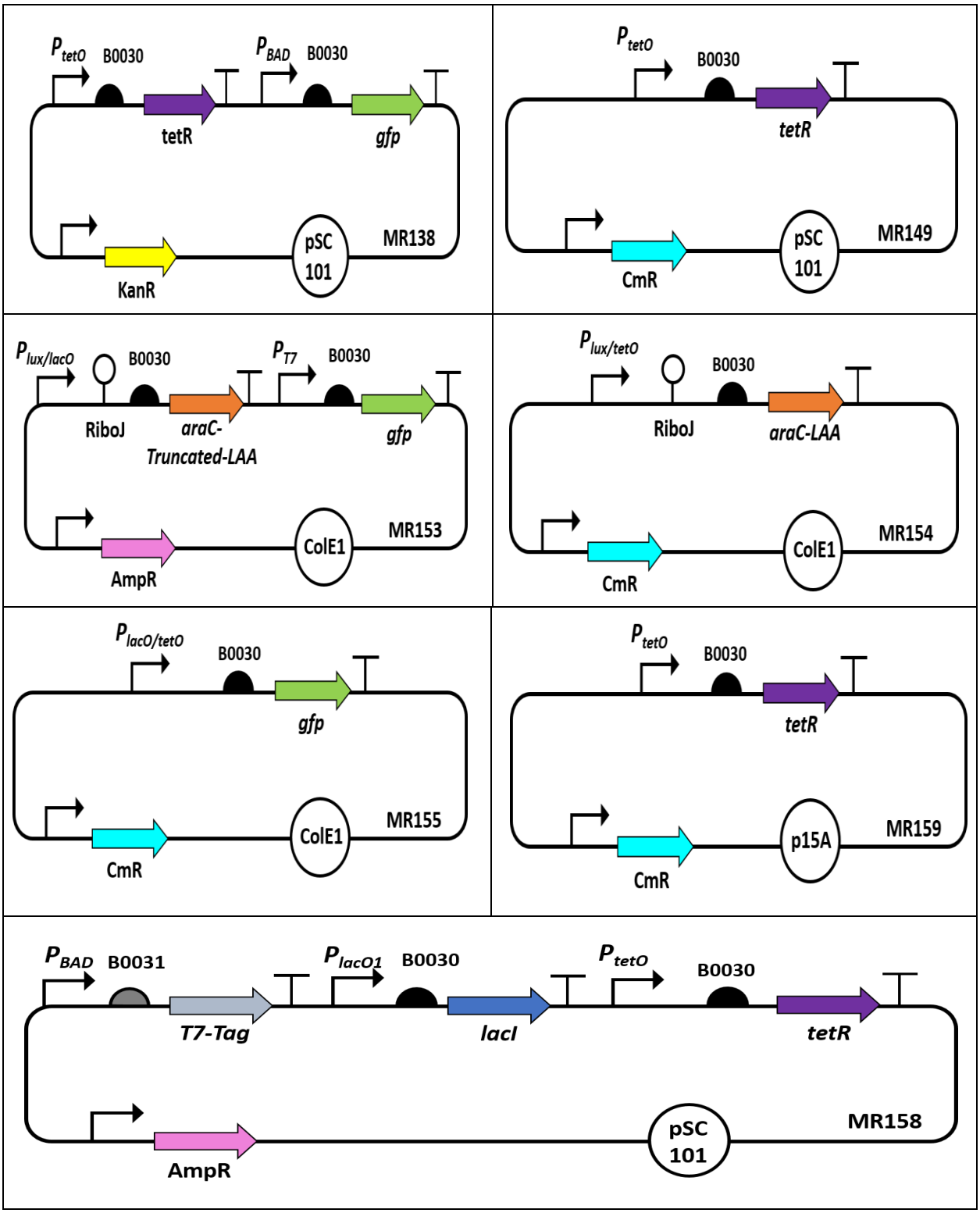


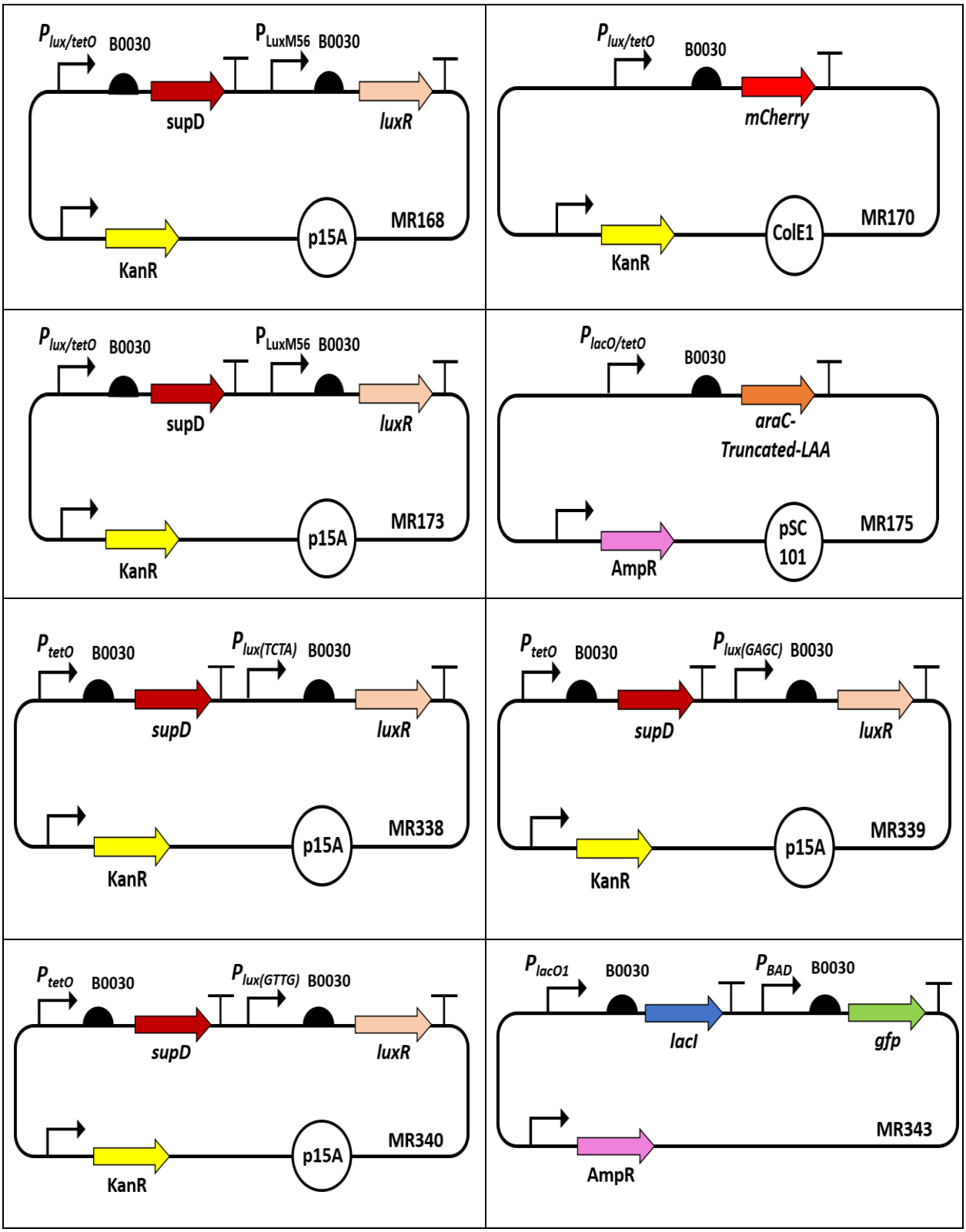

P lux(AAT)
B0030
P lacO
B0030
gfp
luxR
KanR
pSC 101
LR172-5
P lux
B0031
mCherry
CmR
p15A
LR218
P lux
B0030
gfp
AmpR
p15A
LR220
Amp
ColE1
LR255
P roD
B0030
tetR
CmR
p15A
LR285
P BAD
B0030
RiboJ
gfp
P lux/tetO
AmpR
ColE1
LR309
P BAD
B0030
P rhlr
B0030
RiboJ
gfp-LVA
gfp
P lux
AmpR
ColE1
LR310
P lux
B0034
B0030
tetR-LVA
mCherry
CmR
p15A
LR313

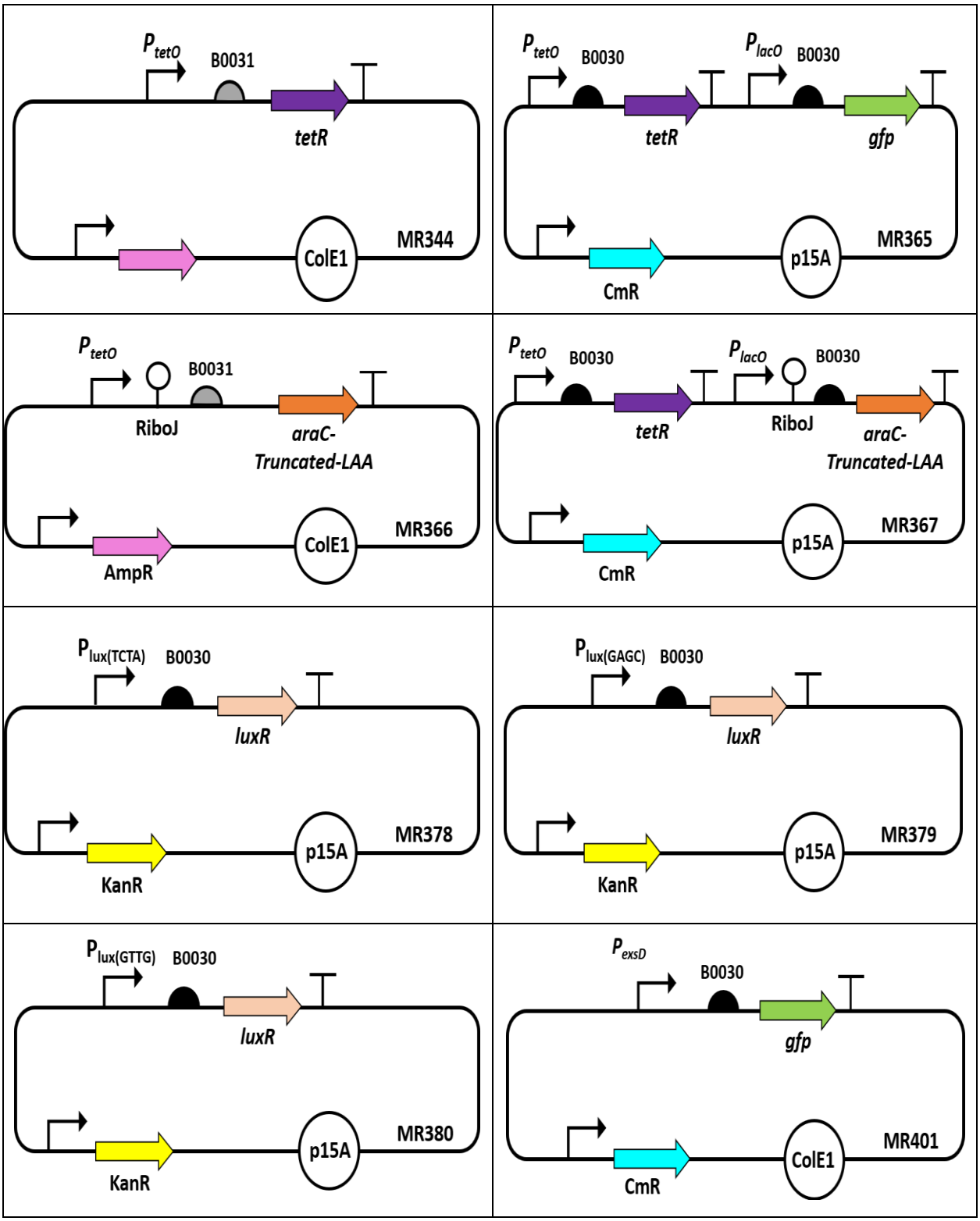

P lux
B0030
P lux
B0030
gfp
luxR
SmR
pSC 101
LR324
P lux
B0030
P LuxM56
B0030
gfp
luxR
SmR
pSC 101
LR325
P lux(NNNN)
B0030
P lacO
B0030
gfp
luxR
pSC 101
LR326
P LuxM56
B0030
P lacO
B0030
gfp
luxR
KanR
pSC 101
LR327
P BAD
B0030
RiboJ
gfp
AmpR
ColE1
LR329
P lux(CCTC)
B0030
P lux
B0030
gfp
luxR
Kan
pSC 101
LR364
P lux(TCCC)
B0030
P lux
B0030
gfp
luxR
Kan
pSC 101
LR365
P lux(TTAA)
B0030
P lux
B0030
gfp
luxR
Kan
pSC 101
LR366

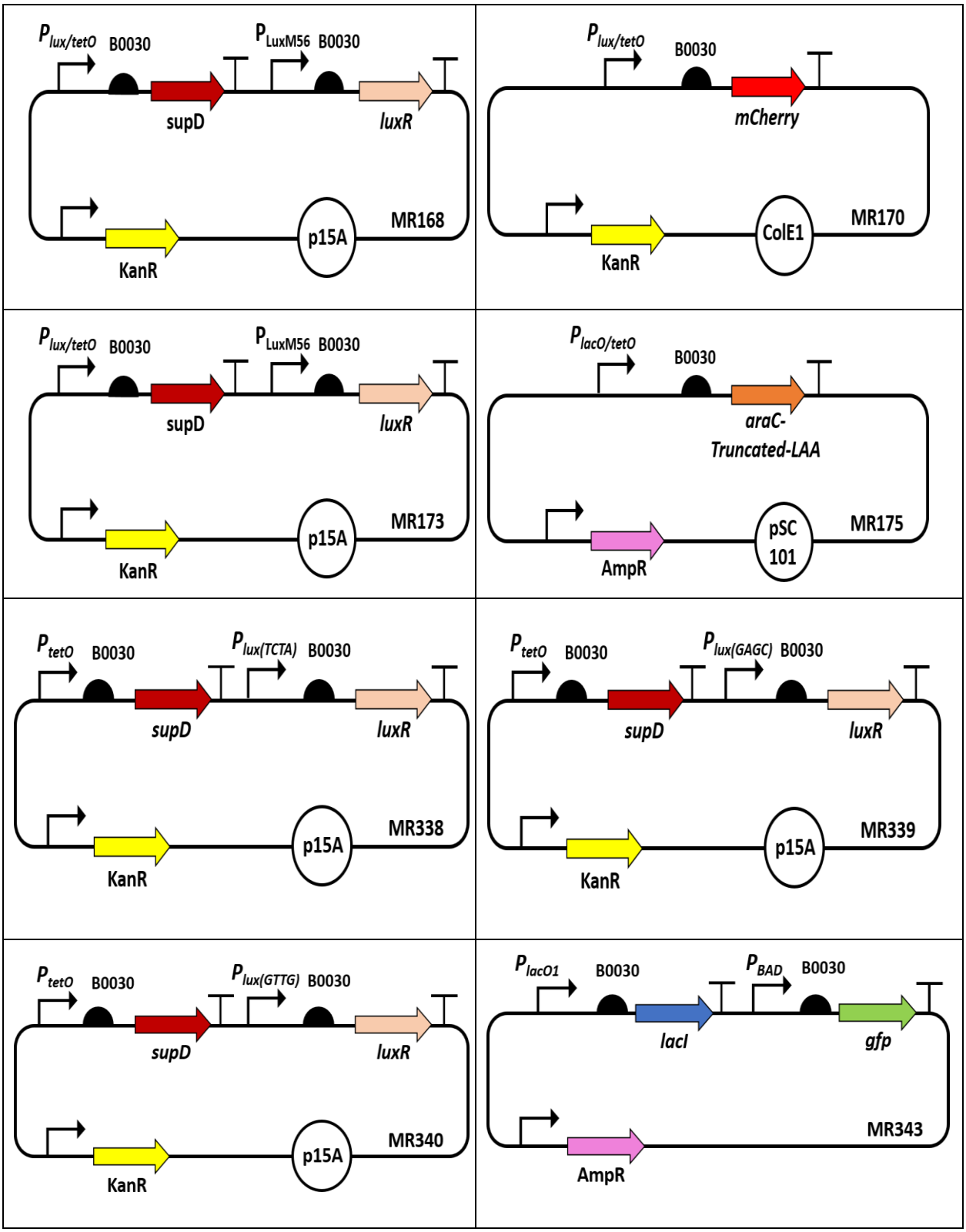

P lux(GTCT) B0030
P lux B0030
gfp
luxR
Kan
pSC 101
LR367
P lux(ACAG) B0030
P lux B0030
gfp
luxR
pSC 101
LR368
P lux(AATG) B0030
P lux B0030
gfp
luxR
Kan
pSC 101
LR369
P lux(GTTC) B0030
P lux B0030
gfp
luxR
Kan
pSC 101
LR370
P lux(TGGG) B0030
luxR
KanR
p15A
MR20
P LuxM56
B0030
luxR
AmpR
p15A
MR21
P lux/lacO
B0030
gfp
CmR
ColE1
MR35
P lacO1
B0030
lacI
AmpR
pSC 101
MR41

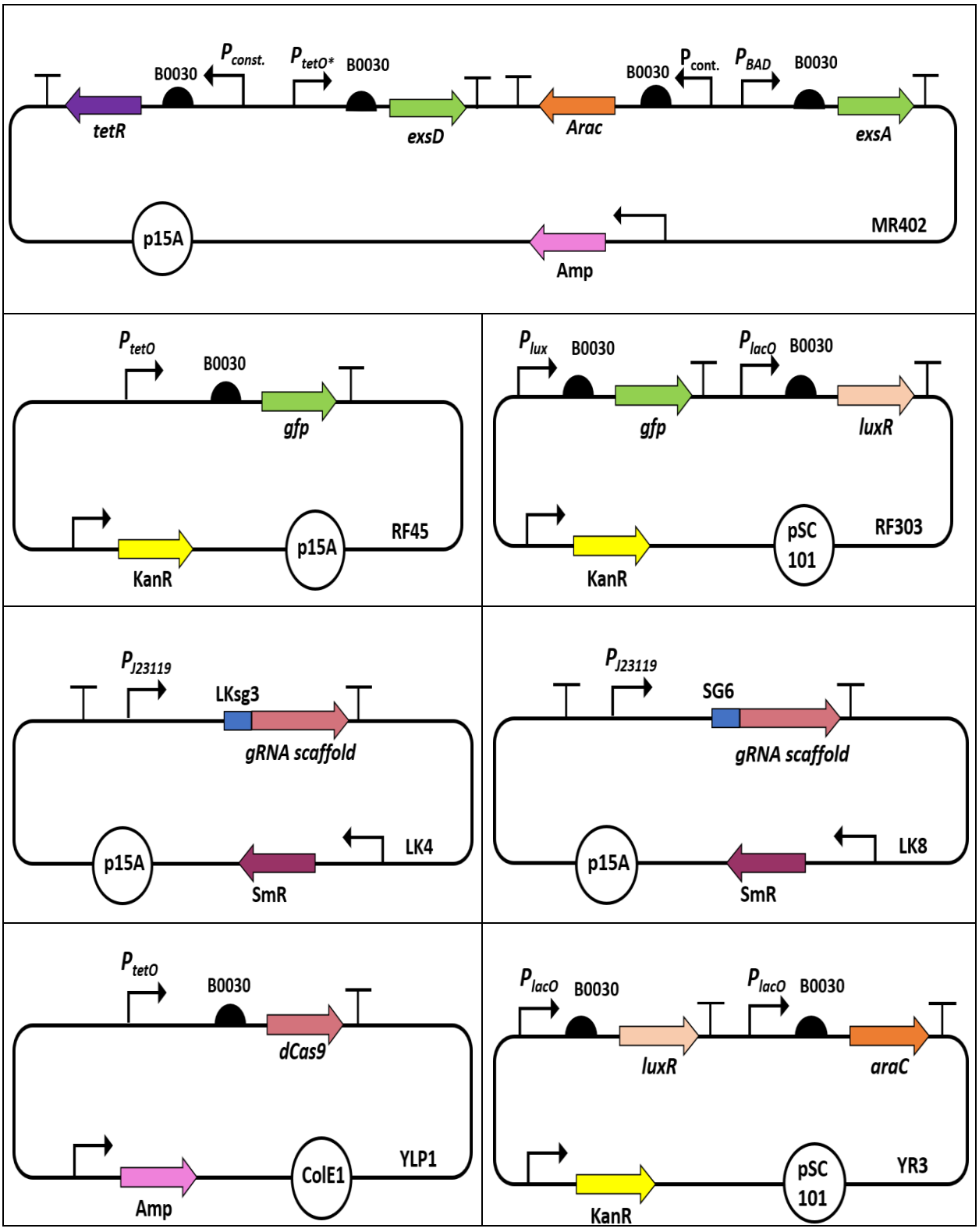

P lacO1
B0030
lacI
P BAD
B0030
gfp
KanR
pSC 101
MR79
P lacO1
B0030
lacI
KanR
pSC 101
MR83
P lacO
B0030
lacI
P BAD
B0030
gfp
KanR
pSC 101
MR92
P lux(TGT)
B0030
luxR
AmpR
p15A
MR113
P lux(AAT)
B0030
luxR
AmpR
p15A
MR114
P tetO
B0030
supD
P LuxM56
B0030
luxR
KanR
p15A
MR125
P lux/lacO
B0030
RiboJ
araC-LAA
CmR
ColE1
MR126
P lux/lacO
B0030
RiboJ
araC-
Truncated-LAA
CmR
ColE1
MR132

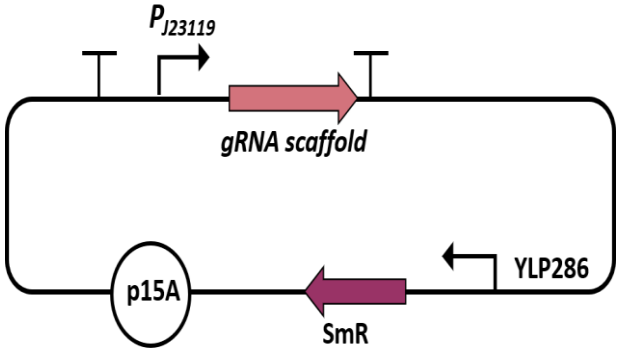

P tetO
B0030
P BAD
B0030
tetR
gfp
KanR
pSC 101
MR138
P tetO
B0030
tetR
CmR
pSC 101
MR149
P lux/lacO
B0030
P T7
B0030
RiboJ
araC-
Truncated-LAA
gfp
AmpR
ColE1
MR153
P lux/tetO
B0030
RiboJ
araC-LAA
CmR
ColE1
MR154
P lacO/tetO
B0030
gfp
CmR
ColE1
MR155
P tetO
B0030
tetR
CmR
p15A
MR159
P BAD
B0031
T7-Tag
P lacO1
B0030
lacI
P tetO
B0030
tetR
AmpR
pSC 101
MR158


P lux/tetO
B0030
P LuxM56
B0030
supD
luxR
KanR
p15A
MR168
P lux/tetO
B0030
mCherry
KanR
ColE1
MR170
P lux/tetO
B0030
P LuxM56
B0030
supD
luxR
KanR
p15A
MR173
P lacO/tetO
B0030
araC-
Truncated-LAA
AmpR
pSC
101
MR175
P tetO
B0030
P lux(TCTA)
B0030
supD
luxR
KanR
p15A
MR338
P tetO
B0030
P lux(GAGC)
B0030
supD
luxR
KanR
p15A
MR339
P tetO
B0030
P lux(GTTG)
B0030
supD
luxR
KanR
p15A
MR340
P lacO1
B0030
P BAD
B0030
lacI
gfp
AmpR
MR343


P tetO
B0031
tetR
ColE1
MR344
P tetO
B0030
P lacO
B0030
tetR
gfp
CmR
p15A
MR365
P tetO
B0031
RiboJ
araC-
Truncated-LAA
AmpR
ColE1
MR366
P tetO
B0030
P lacO
B0030
tetR
RiboJ
araC-
Truncated-LAA
CmR
p15A
MR367
P lux(TCTA)
B0030
luxR
KanR
p15A
MR378
P lux(GAGC)
B0030
luxR
KanR
p15A
MR379
P lux(GTTG)
B0030
luxR
KanR
p15A
MR380
P exsD
B0030
gfp
CmR
ColE1
MR401


P lux/tetO
B0030
P LuxM56
B0030
supD
luxR
KanR
p15A
MR168
P lux/tetO
B0030
mCherry
KanR
ColE1
MR170
P lux/tetO
B0030
P LuxM56
B0030
supD
luxR
KanR
p15A
MR173
P lacO/tetO
B0030
araC-
Truncated-LAA
AmpR
pSC
101
MR175
P tetO
B0030
P lux(TCTA)
B0030
supD
luxR
KanR
p15A
MR338
P tetO
B0030
P lux(GAGC)
B0030
supD
luxR
KanR
p15A
MR339
P tetO
B0030
P lux(GTTG)
B0030
supD
luxR
KanR
p15A
MR340
P lacO1
B0030
P BAD
B0030
lacI
gfp
MR343
AmpR


P const.
P tetO*
B0030
B0030
P cont.
P BAD
B0030
tetR
exsD
Arac
exsA
p15A
MR402
Amp
P tetO
B0030
gfp
KanR
p15A
RF45
P lux
B0030
P lacO
B0030
gfp
luxR
KanR
pSC 101
RF303
P J23119
LKsg3
gRNA scaffold
p15A
LK4
SmR
P J23119
SG6
gRNA scaffold
p15A
LK8
SmR
P tetO
B0030
dCas9
Amp
ColE1
YLP1
P lacO
B0030
P lacO
B0030
luxR
araC
KanR
pSC 101
YR3



17. List of strains used in this study

Plasmids	Figure
LR123 + MR155 + MR159	Fig. 1B, Fig. S2.1, Fig. S2.5A
MR92 + MR159 + MR175	Fig. 1G
MR113 + MR149 + MR170	Fig. 2A
MR113 + MR138 + MR154	Fig. 2D
MR79 + MR114 + MR126	Fig. 2G, Fig. S3.4B
MR125 + MR153 + MR158	Fig. 3C, Fig. S5.1
MR153 + MR158 + MR338	Fig. 3G, Fig.S6.5B
MR153 + MR158 + MR340	Fig. 3G, Fig.S6.5D
MR153 + MR158 + MR339	Fig. 3G, Fig.S6.5F
MR153 + MR158 + MR125	Fig. 3G, Fig.S6.5H
LR7 + LR309 + LR313	Fig. 4C, Fig. S7.11A
LR7 + LR218 + LR310	Fig. 4F, Fig S7.14A, Fig. S9.4, Fig. S9.6
MR83 + MR155 + MR159	Fig. S2.5B
LR1 + LR93	Fig. S2.9A
RF303	Fig. S2.13A - OL Wild type, Fig. S2.14
LR324	Fig. S2.13A - APF Wild type
LR327	Fig. S2.13B - OL mutated
LR325	Fig. S2.13B - APF mutated
LR172-5	Fig. S2.14 - P_{luxAAT}
LR172-1	Fig. S2.14 - P_{luxTGT}
MR21 + MR35 + MR83	Fig. S5.5
MR149 + RF45	Fig. S5.8
MR35 + MR41 + MR378	Fig.S6.4A, C
MR35 + MR41 + MR380	Fig.S6.4A, D
MR35 + MR41 + MR379	Fig.S6.4A, F
MR132 + MR343 + MR378	Fig.S6.7B
MR132 + MR343 + MR380	Fig.S6.7D
MR132 + MR343 + MR379	Fig.S6.7F
MR132 + MR343 + MR20	Fig.S6.7H
LR113 + YR3	Fig. S7.5A

LR329 + YR3	Fig. S7.5A - Control
LR7+LR113	Fig. S7.7A
LR7+LR220	Fig. S7.7B
LR7+LR329	Fig. S7.7C
LR7+LR110	Fig. S7.7D
LR7+LR285+LR309	Fig. S7.9C
LR7+LR309	Fig. S7.9D
LR7+LR171	Fig. S7.13
MR153 + MR158 + MR168	Fig. S7.18A
MR401 + MR402	Fig. S.8.7
LKsg3: LK4 +RF42 + YLP1	Fig. S8.6
SG6: LK8 +RF42 + YLP1	Fig. S8.6
Control: YLP286 +RF42 + LR255	Fig. S8.6
Control: RF303+LR319 LR326+LR319	Fig.S8.8
Control: LR324+LR319 LR364+LR319 LR365+LR319 LR366+LR319 LR367+LR319 LR368+LR319 LR369+LR319 LR370+LR319	Fig.S8.10

18. Supplementary References

1. T. Sambrook, J., Fritsch, E. F., and Maniatis, *Molecular Cloning: A Laboratory Manual* (Cold Spring Harbor Laboratory Press, Plainview, New York, ed. 2, 1989).
2. R. Lutz, 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 Res.* **25**, 1203–1210 (1997).
3. G. Medina, K. Juárez, B. Valderrama, G. Soberón-Chávez, Mechanism of Pseudomonas aeruginosa RhlR Transcriptional Regulation of the rhlAB Promoter. *J. Bacteriol.* **185**, 5976–5983 (2003).
4. R. Daniel, J. R. Rubens, R. Sarpeshkar, T. K. Lu, Synthetic analog computation in living cells. *Nature*. **497**, 619–623 (2013).
5. D. Madar, E. Dekel, A. Bren, U. Alon, Negative auto-regulation increases the input dynamic-range of the arabinose system of Escherichia coli. *BMC Syst. Biol.* **5**, 111 (2011).
6. A. A. K. Nielsen, B. S. Der, J. Shin, P. Vaidyanathan, V. Paralanov, E. A. Strychalski, D. Ross, D. Densmore, C. A. Voigt, Genetic circuit design automation. *Science*. **352** (2016), doi:10.1126/science.aac7341.
7. M. B. Elowitz, A. J. Levine, E. D. Siggia, P. S. Swain, Stochastic gene expression in a single cell. *Science (80-.).* **297** (2002), doi:10.1126/science.1070919.
8. P. S. Swain, M. B. Elowitz, E. D. Siggia, Intrinsic and extrinsic contributions to stochasticity in gene expression. *Proc. Natl. Acad. Sci. U. S. A.* **99** (2002), doi:10.1073/pnas.162041399.
9. E. M. Ozbudak, M. Thattai, I. Kurtser, A. D. Grossman, A. Van Oudenaarden, Regulation of noise in the expression of a single gene. *Nat. Genet.* **31** (2002), doi:10.1038/ng869.
10. P. R. Milo Ron, *Biology by the Numbers* (2008).
11. G. K. Ackers, A. D. Johnsontt, M. A. Shea, Quantitative model for gene regulation by A phage repressor. *Proc. NatL Acad. Sci. USA*. **79**, 1129–1133 (1982).
12. L. Bintu, N. E. Buchler, H. G. Garcia, U. Gerland, T. Hwa, J. Kondev, R. Phillips, Transcriptional regulation by the numbers: models. *Curr Opin Genet Dev*. **15**, 116–124 (2005).
13. R. S. Cox, M. G. Surette, M. B. Elowitz, M. B. Elowitz, Programming gene expression with combinatorial promoters. *Mol. Syst. Biol.* **3** (2007), doi:10.1038/msb4100187.
14. N. L. Lee, W. 0 Gielow, R. G. Wallace, Mechanism of araC autoregulation and the domains of two overlapping promoters, Pc and PBAD' in the L-arabinose regulatory region ofEscherichia coli. *Biochemistry*. **78**, 752–756 (1981).
15. J. B. Andersen, C. Sternberg, L. K. Poulsen, S. P. Bjørn, M. Givskov, S. Molin, New unstable variants of green fluorescent protein for studies of transient gene expression in bacteria. *Appl. Environ. Microbiol.* **64**, 2240–2246 (1998).
16. A. Tamsir, J. J. Tabor, C. A. Voigt, Robust multicellular computing using genetically encoded NOR gates and chemical "wiresg." *Nature*. **469** (2011), doi:10.1038/nature09565.
17. R. Daniel, T. Lu, J. Rubens, Front-End Analog Signal Processing For Cellular Computation (2017), pp. 1–12.
18. K. Jarrett, K. Kavukcuoglu, M. A. Ranzato, Y. LeCun, in *IEEE 12th International Conference on Computer Vision* (IEEE, 2009), pp. 2146–2153.

19. V. Nair, G. E. Hinton, in *ICML 27th International Conference on Machine Learning* (ICML, 2010).
20. Smooth maximum - Wikipedia, (available at https://en.wikipedia.org/wiki/Smooth_maximum).
21. T. S. Gardner, C. R. Cantor, J. J. Collins, Construction of a genetic toggle switch in Escherichia coli. *Nature*. **403**, 339–342 (2000).
22. T. H. Segall-Shapiro, A. J. Meyer, A. D. Ellington, E. D. Sontag, C. A. Voigt, A "resource allocator" for transcription based on a highly fragmented T7 RNA polymerase. *Mol. Syst. Biol.* (2014), doi:10.15252/msb.20145299.
23. A. A. K. Nielsen, C. A. Voigt, Multi-input CRISPR/Cas genetic circuits that interface host regulatory networks. *Mol Syst Biol* (2014), doi:10.15252/msb.20145735.
24. D. E. Paschon, M. Ostermeier, Construction of Protein Fragment Complementation Libraries Using Incremental Truncation. *Methods Enzymol.* **388**, 103–116 (2004).
25. M. Kanwar, R. C. Wright, A. Date, J. Tullman, M. Ostermeier, Protein Switch Engineering by Domain Insertion. *Methods Enzymol.* **523**, 369–388 (2013).
26. J. C. Anderson, C. A. Voigt, A. P. Arkin, Environmental signal integration by a modular and gate. *Mol. Syst. Biol.* **3** (2007), doi:10.1038/msb4100173.
27. D. Jeruzalmi, T. A. Steitz, Structure of T7 RNA polymerase complexed to the transcriptional inhibitor T7 lysozyme. *EMBO J.* **17**, 4101–4113 (1998).
28. Part:BBa B0031 - parts.igem.org, (available at http://parts.igem.org/Part:BBa_B0031).
29. D. E. Knuth, in *The Art of Computer Programming* (Addison-Wesley Professional, 2008), pp. 64–74.
30. J. A. Anderson, *An introduction to neural networks* (MIT Press, 1995).
31. J. A. Snyman, *PRACTICAL MATHEMATICAL OPTIMIZATION: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms* (Springer, Boston, MA, 2005).
32. N. Qian, On the momentum term in gradient descent learning algorithms. *Neural Networks*. **12**, 145–151 (1999).
33. R. Daniel, L. Rizik, L. Daniel, in *NanoCom '17 Proceedings of the 4th ACM International Conference on Nanoscale Computing and Communication* (Washington, D.C., 2017).
34. R. H. Walden, Analog-to-digital converter survey and analysis. *IEEE J. Sel. Areas Commun.* **17**, 539–550 (1999).
35. G. Avitabile, M. Forti, S. Manetti, M. Marini, On a class of nonsymmetrical neural networks with application to ADC. *IEEE Trans. Circuits Syst.* **38**, 202–209 (1991).
36. L. Danial, N. Wainstein, S. Kraus, S. Kvatinsky, Breaking Through the Speed-Power-Accuracy Tradeoff in ADCs using a Memristive Neuromorphic Architecture. *IEEE Trans. Emerg. Top. Comput. Intell.* **2**, 396–409 (2018).
37. D. W. Selinger, K. J. Cheung, R. Mei, E. M. Johansson, C. S. Richmond, F. R. Blattner, D. J. Lockhart, G. M. Church, RNA expression analysis using a 30 base pair resolution Escherichia coli genome array. *Nat. Biotechnol.* **18**, 1262–1268 (2000).
38. J. Georg, B. rn Voß, I. Scholz, J. Mitschke, A. Wilde, W. R. Hess, Evidence for a major role of antisense RNAs in cyanobacterial gene regulation. *Mol. Syst. Biol.* (2009), doi:10.1038/msb.2009.63.
39. M. Güell, V. van Noort, E. Yus, W.-H. Chen, J. Leigh-Bell, K. Michalodimitrakis, T. Yamada,

M. Arumugam, T. Doerks, S. Kühner, M. Rode, M. Suyama, S. Schmidt, A.-C. Gavin, P. Bork, L. Serrano, Transcriptome complexity in a genome-reduced bacterium. *Science*. **326**, 1268–1271 (2009).
40. J. A. N. Brophy, C. A. Voigt, Antisense transcription as a tool to tune gene expression. *Mol. Syst. Biol.* (2016), doi:10.15252/MSB.20156540.
41. A. Chatterjee, L. Drews, S. Mehra, E. Takano, Y. N. Kaznessis, Convergent Transcription in the Butyrolactone Regulon in Streptomyces coelicolor Confers a Bistable Genetic Switch for Antibiotic Biosynthesis. *PLoS One*. **6**, 21974 (2011).
42. A. Chatterjee, C. M. Johnson, C.-C. Shu, Y. N. Kaznessis, D. Ramkrishna, G. M. Dunny, W.-S. Hu, Convergent transcription confers a bistable switch in Enterococcus faecalis conjugation. *Proc. Natl. Acad. Sci.* **108**, 9721–9726 (2011).
43. E. M. Fozo, M. R. Hemm, G. Storz, Small toxic proteins and the antisense RNAs that repress them. *Microbiol. Mol. Biol. Rev.* **72**, 579–589 (2008).
44. C. Lou, B. Stanton, Y.-J. Chen, B. Munsky, C. A. Voigt, N. B. Author, Ribozyme-based insulator parts buffer synthetic circuits from genetic context. *Nat Biotechnol*. **30**, 1137–1142 (2012).
45. T. S. Moon, C. Lou, A. Tamsir, B. C. Stanton, C. A. Voigt, Genetic programs constructed from layered logic gates in single cells. *Nature*. **491**, 249–53 (2012).
46. J. R. Rubens, G. Selvaggio, T. K. Lu, Synthetic mixed-signal computation in living cells. *Nat. Commun.* **7** (2016), doi:10.1038/ncomms11658.
47. B. Murmann, B. E. Boser, A 12-bit 75-MS/s Pipelined ADC Using Open-Loop Residue Amplification. *IEEE J. Solid-State Circuits*. **38**, 2040–2050 (2003).
48. T. Shopera, W. R. Henson, A. Ng, Y. J. Lee, K. Ng, T. S. Moon, Robust, tunable genetic memory from protein sequestration combined with positive feedback. *Nucleic Acids Res.* **43**, 9086–9094 (2015).
49. J. E. Ferrell, Signaling Motifs and Weber's Law. *Mol. Cell*. **36**, 724–727 (2009).
50. K. S. Nilgiriwala, J. Joséjiméjoséjiménez, P. M. Rivera, D. Del Vecchio, Synthetic Tunable Amplifying Buffer Circuit in E. coli (2014), doi:10.1021/sb5002533.
51. H. M. Salis, E. A. Mirsky, C. A. Voigt, Automated design of synthetic ribosome binding sites to control protein expression. *Nat. Biotechnol.* **27**, 946–950 (2009).
52. H. Alper, C. Fischer, E. Nevoigt, G. Stephanopoulos, Tuning genetic control through promoter engineering. *Proc. Natl. Acad. Sci. U. S. A.* **102**, 12678–12683 (2005).
53. Y. Li, Y. Jiang, H. Chen, W. Liao, Z. Li, R. Weiss, Z. Xie, Modular construction of mammalian gene circuits using TALE transcriptional repressors. *Nat. Chem. Biol.* **11**, 207–213 (2015).
54. B. C. Stanton, A. A. K. Nielsen, A. Tamsir, K. Clancy, T. Peterson, C. A. Voigt, Genomic mining of prokaryotic repressors for orthogonal logic gates. *Nat. Chem. Biol.* **10**, 99–105 (2014).
55. X. Liu, S. T. P. Gupta, D. Bhimsaria, J. L. Reed, J. Jos´, J. A. Rodríguez-Martínez, A. Z. Ansari, S. Raman, De novo design of programmable inducible promoters. *Nucleic Acids Res.* **47**, 10452–10463 (2019).
56. A. Espah Borujeni, D. M. Mishler, J. Wang, W. Huso, H. M. Salis, Automated physics-based design of synthetic riboswitches from diverse RNA aptamers. *Nucleic Acids Res.* **44**, 1–13 (2016).
57. S. Zucca, L. Pasotti, N. Politi, M. Casanova, G. Mazzini, M. G. Cusella De Angelis, P. Magni,

Multi-Faceted Characterization of a Novel LuxR-Repressible Promoter Library for Escherichia coli. *PLoS One*. **10**, e0126264 (2015).
58. D. E. Cameron, J. J. Collins, Tunable protein degradation in bacteria. *Nat. Biotechnol.* **32**, 1276–1281 (2014).
59. B. P. Landry, R. Palanki, N. Dyulgyarov, L. A. Hartsough, J. J. Tabor, Phosphatase activity tunes two-component system sensor detection threshold. *Nat. Commun.* **9**, 1–10 (2018).
60. M. Morel, R. Shtrahman, V. Rotter, L. Nissim, R. H. Bar-Ziv, Cellular heterogeneity mediates inherent sensitivity-specificity tradeoff in cancer targeting by synthetic circuits. *Proc. Natl. Acad. Sci. U. S. A.* **113**, 8133–8138 (2016).
61. J. J. Hopfield, D. W. Tank, Neural computation of decisions in optimization problems. *Biol. Cybern.* **52**, 141–152 (1985).
62. B. H. Weinberg, N. T. H. Pham, L. D. Caraballo, T. Lozanoski, A. Engel, S. Bhatia, W. W. Wong, Large-scale design of robust genetic circuits with multiple inputs and outputs for mammalian cells. *Nat. Biotechnol.* **35**, 453–462 (2017).
63. S. Haykin, *Neural Networks: A Comprehensive Foundation* (Pearson Education, Indian, Bangladesh, ed. 2, 2004), vol. 19.
64. M. Prezioso, F. Merrikh-Bayat, B. D. Hoskins, G. C. Adam, K. K. Likharev, D. B. Strukov, Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature*. **521**, 61–64 (2015).
65. E. Neftci, J. Binas, U. Rutishauser, E. Chicca, G. Indiveri, R. J. Douglas, Synthesizing cognition in neuromorphic electronic systems. *Proc. Natl. Acad. Sci. U. S. A.* **110**, E3468--76 (2013).
66. L. Qian, E. Winfree, J. Bruck, Neural network computation with DNA strand displacement cascades. *Nature* (2011), doi:10.1038/nature10262.
67. J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities. *Proc. NatL Acad. Sci. USA*. **79**, 2554–2558 (1982).
68. R. Sarpeshkar, Analog synthetic biology. *Philos. Trans. A. Math. Phys. Eng. Sci.* **372** (2014), doi:10.1098/rsta.2013.0110.
69. M. Adler, U. Alon, Fold-change detection in biological systems. *Curr. Opin. Syst. Biol.* (2018), , doi:10.1016/j.coisb.2017.12.005.
70. M. Müller, S. Ausländer, A. Spinnler, D. Ausländer, J. Sikorski, M. Folcher, M. Fussenegger, Designed cell consortia as fragrance-programmable analog-to-digital converters. *Nat. Chem. Biol.*, 309–316 (2017).
71. J. W. Kotula, S. Jordan Kerns, L. A. Shaket, L. Siraj, J. J. Collins, J. C. Way, P. A. Silver, by D. Richard Kolodner, Programmable bacteria detect and record an environmental signal in the mammalian gut. *Proc Natl Acad Sci*. **111**, 4838–4843 (2014).
72. L. Schukur, B. Geering, G. Charpin-El Hamri, M. Fussenegger, Implantable synthetic cytokine converter cells with AND-gate logic treat experimental psoriasis. *Sci. Transl. Med*. **7** (2015), doi:10.1126/scitranslmed.aac4964.
73. H. Ye, M. Xie, S. Xue, G. Charpin-El Hamri, J. Yin, H. Zulewski, M. Fussenegger, Self-adjusting synthetic gene circuit for correcting insulin resistance. *Nat. Publ. Gr.* **1**, 5 (2016).
74. Y. Soma, K. Tsuruno, M. Wada, A. Yokota, T. Hanai, Metabolic flux redirection from a central metabolic pathway toward a synthetic pathway using a metabolic toggle switch

(2014), doi:10.1016/j.ymben.2014.02.008.
75. R. I. Kitney, P. S. Freemont, Engineering biology: a key driver of the bio-economy. *Eng. Biol.* **1**, 3–6 (2017).
76. Introduction to Fluorescent Proteins | MicroscopyU, (available at https://www.microscopyu.com/techniques/fluorescence/introduction-to-fluorescent-proteins).
77. B. Canton, A. Labno, D. Endy, Refinement and standardization of synthetic biological parts and devices. *Nat. Biotechnol.* **26**, 787–793 (2008).