**Controllers:**

* Analog Dial
  + Outputs an analog value determined by how far the dial has been rotated. Can be used to configure settings on certain logic components.
* Button
  + Outputs a signal (from both ends) for as long as the button is held down
* Joystick
  + Outputs signals to any of the four outputs depending on where the joystick is relative to each output. Dual signals can be sent by positioning the joystick near the middle of two outputs. The outputted signal(s) will contain both an analog value and a 'Bits' value (relative to the extent to which the joystick is tilted or pushed toward an output).
* Keypad
  + A security device which outputs a signal when the correct keycode is entered. The default code is '0000' until changed. Instructions for changing code:
    - [1]: Connect a Cable to the cable socket on the bottom of the keypad
    - [2]: Connect the other end of the cable to the top of a second keypad (via Cable Adapter)
    - [3]: Type in the current keycode for the first keypad using the first keypad.
    - [4]: Quickly type in your new 4-digit code on the second keypad.
* Micro Button
  + A miniature variant of the standard Button. Outputs a signal for as long as the button is held down.
* Micro Switch
  + A miniature variant of the standard Switch. Outputs a signal for as long as the Switch remains on.
* Number Pad
  + A handy device which allows you to type in a custom value and will output a 'Bits' or analog signal depending on the current mode. To switch between analog/bits, simply flip the switch on the bottom of the device. When in analog mode, the device will function just like the Analog Dial/Module and can be used to configure certain logic components.
* Switch
  + Outputs a signal (from both ends) for as long as the Switch remains on.

**Gates:**

* AND Gate
  + Outputs a signal if both inputs are true
* BLINK Gate
  + Pulses a signal once if the input is true. Will reset once the input is no longer true.
* NAND Gate
  + Outputs a signal if either/neither input is true, but not if both are true at the same time.
* NOR Gate
  + Outputs a signal if neither input is true.
* NOT Gate
  + Outputs a signal unless the input is true.
* OR Gate
  + Outputs a signal if either or both inputs are true.
* SWITCH Gate
  + Function like a switch - toggles on/off each time the input is true.
* TRIGGER Gate
  + Functionally similar to the Blink Gate, this gate outputs a single pulse when the primary input is true; however, it will not pulse again until the secondary input (located on the left side of the gate) receives a signal, effectively acting as a reset.
    - Trigger modes:
      * [Pulse Once]: (Default) Functions like a Blink Gate, outputting a single pulse regardless of whether the primary input remains true in the next signal tick. (However, if the secondary input unlocks the gate within the same tick, it will pulse again.)
      * [Until False]: Continuously outputs a signal while the primary input remains true. Once the input turns false, the gate will not accept another signal until it is reset by the secondary input
* VALVE Gate
  + Can be used as an alternative to the AND Gate to prevent data interference between two signals (assuming they are carrying any). When input 'A' is true, it can only pass if input 'B' (located on the side of the gate) is also true. Input 'B' only controls the state of input 'A', so any data that input 'B' may be carrying will be ignored and will not affect the output.
* XNOR Gate
  + Outputs a signal if both or neither input are true.
* XOR Gate
  + Outputs a signal if either input are true, but not if both are true at the same time.
* Buffer
  + Syncs a signal to the logic timeline by yielding its input for a quarter of a logic tick before outputting. Only necessary for advanced circuits where timing is paramount to its function.
* Delay
  + Outputs a signal after a set amount of time if the input is true. Delay-time can be configured using an Analog Dial/Module, Cable, and Cable Adapter.
* Queue
  + Functionally similar to the Delay Gate with the combined functionality of outputting signal data from each tick in the order they were received, effectively acting as a chain of Buffer Gates.
  + Delay-time can be configured using an Analog Dial/Module, Cable, and Cable Adapter.
* Repeater
  + Pulses a signal at intervals of the configured time if the input is true. Time between intervals can be configured using an Analog Dial/Module, Cable, and Cable Adapter.
* Sustainer
  + Maintains a signal for the configured amount of time once the input is no longer true.
  + Sustain-time can be configured using an Analog Dial/Module, Cable, and Cable Adapter.

**Arduino:**

* Analog Module
  + Outputs an analog value by taking any 'Bits' that an input is carrying and dividing them by the configurable denominator. Will also output the rounded percent total in the form of 'Bits'. This gate can be used to throttle certain components or as an alternative to the 'Analog Dial' to allow for autonomous configuration of certain logic components.
* Analog Gauge
  + A measurement device that can be used to visualize an analog value (or calculate and display an analog value via two 'Bits' values). If input 'B' (to the right of the dial) receives a 'Bits' value, that will become the reference denominator for any 'Bits' sent to input 'A' (beneath the dial) and the gauge will display A/B. If input 'B' is not receiving a signal, signals sent to input 'A' must already be in analog format in order for the dial to display anything (custom analog values can be achieved by using the 'Analog Module' or by sending 'Bits' to input 'B').
* Angle Module
  + Outputs the item's upwards orientation in relation to the ground in the form of 'Bits'.
    - Additional features:
      * If the input receives a signal from a Distance Module, the outputted value will be the angle between the upwards orientation of the distance module relative to the upwards orientation of the angle module.
      * For additional compatibility features, refer to the description of the Coordinate Module.
* Bit Display (S, M, L)
  + A modular digital display segment that can display a single digit in an input's 'Bits' value. Multiple segments can be linked together to allow for larger numbers to be displayed. If multiple displays are used, only the first segment in the chain (furthest to the right) will accept an input (the rest will be synced automatically).
* Bit Memory Module
  + A compact flash-memory storage device which is capable of storing up to 20 'Bits' values, each accessible by inputting their respective index.
    - Writing to an index:
      * To store a value, start by sending a signal with your desired index (in the form of 'Bits') to input 'A' (highlighted in yellow). Next, send the 'Bits' value you wish to assign to that index to input 'B' (highlighted in blue). Your value will be forever saved to that index until overwritten with a new value or wiped. Refer to the following section for how to retrieve a value.
    - Retrieving a stored value:
      * To retrieve a stored value, simply send a signal with the index you wish to retrieve (in the form of 'Bits') to input 'A' (highlighted in yellow). If there is a value assigned to the index, it will be assigned to the signal and carried to the primary output (located on the back of the module). If there isn't a value assigned to the index, the signal (along with all of its original data) will be carried to the secondary output (located on the left side of the module). Using the secondary output is entirely optional; it is just there to make it easier for those who wish to make advanced sorting systems and whatnot.
    - Wiping and index:
      * To wipe a value from an index, press and quickly release the 'Wipe' button (located on the right side of the module). To wipe every index at once, press and hold the button until it begins to flash, and continue to hold until it beeps and flashes green twice. (Releasing at any point before then will cancel it.)
    - Additional information:
      * If no index is sent to input 'A' when retrieving a value, the module will use the last index that was sent.
      * You can write to an index regardless of whether or not input 'A' is actively receiving a signal. Similar to above, the module will use the last index sent.
      * Input 'B' will automatically yield for a quarter of a logic tick before writing to an index. This is a safeguard for situations where you are consecutively writing to indexes every tick, which ensures that it writes to the latest index.
      * Here's a cookie if you read this far: 🍪
* Bit Module
  + Compares 'Bits' from inputs 'A' & 'B' and outputs signal 'A' if the operator statement is true. Input 'B' will save the number of bits from its most recent signal. To reset it, activate input 'A' and send a blank signal (one that has not passed through another module or node) to input 'B'. The operator can be configured by linking a Cable Adapter, Cable, and Analog Dial/Module to input 'A'. If using a relational operator, output 'B' (located on the side of the module) can be used as an 'else' output and will output signal 'A' if the operator statement is false.
* Bit Node
  + Outputs a signal with a set amount of 'Bits' added to its input signal. If the input signal is hot carrying any Bits, it will be given the number that is displayed on the node. This number can be configured using an Analog Dial/Module, Cable, and 2 Cable Adapters. To change the number to negative or positive, it's the same setup as if you were changing the number, but use a Switch instead of an Analog Dial/Module.
* Bit Splitter Module
  + Compares 'Bits' from inputs 'A' & 'B' and outputs signal 'A' if the operator statement is true. Input 'B' will save the number of bits from its most recent signal. To reset it, activate input 'A' and send a blank signal (one that has not passed through another module or node) to input 'B'. The operator can be configured by linking a Cable Adapter, Cable, and Analog Dial/Module to input 'A'. If using a relational operator, output 'B' (located on the side of the module) can be used as an 'else' output and will output signal 'A' if the operator statement is false.
* Booster Antenna
  + An attachment mod for the Control Box which will extend its range to 1000 studs.
* Coordinate Module
  + Outputs the item's current coordinate of the configurable axis in the form of 'Bits'. The measured axis can be configured using an Analog Dial/Module, Cable, and 2 Cable Adapters.
    - Additional features:
      * S If the input receives a signal from a Distance Module, the outputted value will be the difference between the distance module's ray hit position and the coordinate module's position on the configured axis (i.e local/object space).
      * If the input receives a signal from an Angle Module, the outputted value will be the angle module's world-space rotation on the configured axis.
      * If the input receives a signal from a Player Sensor, the outputted value will be the difference between the player's position and the player sensor's position on the configured axis (i.e local/object space).
    - Warning: When using sequence (Angle Module> Coordinate Module) or (Player Sensor > Coordinate Module), the coordinate module's position/orientation will not influence the outputted value; you'd only be using this module to select which axis of the value you wish to read.
* Clock Module
  + Outputs the current in-game clock time as a preformatted (HH|MM) 'Bits' value. If the module is inactive (not receiving an input signal), it will pulse a single signal at sundown, and another at sunrise.
* Contact Receiver
  + Used to bind components to Control Boxes. Must be attached to one end of a button.
* Control Box (4, 6, 10 Components can be linked)
  + Controls components remotely-must be configured in advance. Maximum control range is 250 studs, or 1000 studs with a Booster Antenna.
    - Instructions for binding:
      * [1]: Attach a Contact Receiver to one end of a Button
      * [2]: Using a Cable Adapter, attach one end of a Cable to the other end of the button, then attach the other end of the cable to the cable socket on this component
      * [3]: Touch the Contact Receiver to the component you wish to bind and press the button.
      * [4]: The Contact Receiver will flash green once bound.
      * [5]: To unbind, simply repeat steps 3-4, and the Contact Receiver will flash red once unbounded. (It may also flash red if you attempt to bind a new item and the device is already full.)
    - When a component is bound to the device, a light will appear on the top of the device, with each light representing a separate component. The light will change colors depending on the status of the component bind.
    - Indicator light legend:
      * [Black]: Empty/unbounded slot
      * [Blue]: Idle/inactive signal
      * [Green]: Active and stable signal
      * [Orange]: Out of maximum control range
      * [Red]: No permissions (component is within non whitelisted zone)
* Distance Module
  + Outputs the distance between the bottom of the module and whatever surface is beneath it in the form of 'Bits'. If the input signal is already carrying Bits, they will be overwritten by the outputted value.
  + For additional compatibility features, refer to the descriptions of Angle Module & Coordinate Module.
* Keybind Relay
  + Outputs a signal to any input that it's attached to. Must be bound to a Control Box or Control Seat.
* Laser Receiver
  + Outputs a signal if hit by a Laser Transmitter.
* Laser Transmitter
  + Outputs a laser that can trigger Laser Receivers. Range can be configured using an Analog Dial/Module, Cable, and 2 Cable Adapters.
* Micro Gyro
  + A single-axis gyroscope that will attempt to keep itself oriented upright while activated. Automatically adjusts its force depending on the weight of the assembly it is attached to.
* Mini 7-Segment
  + Can be used to display numbers and symbols. Each segment can be illuminated by its corresponding input behind it.
* Note Module
  + Plays one of the 61 configurable piano notes when the input receives a signal. The longer the input remains active, the longer the note will be sustained (up to 6.5 seconds). Multiple modules can be played in sequence to make music. The active note can be configured using an Analog Dial/Module, Cable, and 2 Cable Adapters.
* Phase Analyzer
  + Can be used to precisely measure the difference in arrival time between two signals, useful for debugging time-sensitive circuits.
* Pixel Panel
  + A modular 4×4 array of pixels that converts a provided 'Bit value into visible pixels. Please read the entirety of this description to learn how to correctly use this component.
    - Instructions for usage:
    - [1]: If this is your first time using, begin by inputting a 'Bit' value of 8 and verify that the display is placed upright. This will make the following steps easier to understand. (An indicator is located on the back of the display to help you identify which way is up.)
    - [2]: This item accepts (up to) a 16-digit 'Bits' value, where each digit's position in the string corresponds to the respective pixel on the display. Facing the front of the display, locate the pixel at top left corner. This is the starting pixel, which the first digit in your 'Bits' value represents. The display is read left to right, so every group of 4 digits in your 'Bits' value is a new line of pixels. to see it in action, continue adding another 8 to your example string (8, 88,888, 888, and so on...) until the display fills up.
    - [3]: This display uses a 3-bit tonal scale, so the value of each digit in your 'Bits' value will control the brightness of that pixel: Compatible range for each digit: 1(Darkest) – 8(Brightest).
* Player Sensor Module
  + Outputs the nearest player's distance from the item (within a maximum radius of 100 studs) in the form of 'Bits'. Inputting a 'Bit' value of -1 will exclude your player from being detected, and inputting a value of 1 will exclude all other players except yours from being detected. If no 'Bits' are provided, all players will be included. (Will always exclude dead players or players protected by a spawn field.)
  + For additional compatibility features, refer to the description of the Coordinate Module.
* Pressure Plate
  + Outputs a signal for as long as an object is on top of the plate's surface. Only responds to parts, items, or players.
* Remote Camera
  + Attaches your camera to its lens when activated. Can either be a key-bound to a Control Box and activated via a Keybind Relay. Will not function if you are not seated.
* Remote Socket
  + Can be welded to a component and used as a socket to control it. (If you wish for a wireless control, consider using the Control Box instead.)
* Super Antenna
  + An attachment mod for the Control Box which will extend its range to 5000 studs.
* Signal Indicator
  + Illuminates when the input is true. Used for testing and debugging circuits.
* Tripwire Sensor
  + An invisible laser that outputs a signal when an object obstructs its path. Range can be configured using an Analog Dial/Module, Cable, and 2 Cable adapters.
* Velocity Module
  + Outputs the item's current velocity (studs/second) in the form of 'Bits'. If the input signal is already carrying Bits, they will be overwritten by the outputted value.
* 7-Segment Display
  + Can be used to display number and symbols. Each segment can be illuminated by its corresponding input behind it.
* Signal Radio
  + A wireless communication device capable of transmitting and receiving signals, provided that at least one or more radios are tuned to the same channel.
    - Configuring a channel:
      * To configure the device's channel, send a signal (in the form of 'Bits') to input 'C' (highlighted in purple) located on the back of the module. The value you provide will be stored as the active channel. Any radios set to the same channel will automatically be linked together, forming a shared communication line.
    - Sending a signal:
      * To transmit a signal, send a signal to input 'A' (highlighted in green) located on the left side of the device. The signal will then be broadcasted to all radios that are tuned to the same channel.
        + Note: The transmitting device will never receive its own broadcast, even if it shares the same channel. This prevents feedback loops and echoing signals.
    - Receiving a signal:
      * Any signals broadcasted on the channel will be output through input 'B' (highlighted in blue) located on the right side of the device. Only transmissions from other radios configured to the same channel will be received. If no other radios are connected, the incoming port will remain idle.
    - Channel protection:
      * The switch located on the front of the device toggles channel protection. When enabled, only signals sent by the player or players they have whitelisted will be accepted. This prevents interference from untrusted sources. When disabled, any player may broadcast signals to the device's channel, regardless of whitelist status. Signal protection is enabled by default.
  + Additional information:
    - Each radio functions as both a transmitter and receiver, allowing for two-way communication.
    - Radios cannot self-transmit; outgoing signals will never be echoed back to the same device.
    - Multiple radios can share the same channel, making it possible to construct wide-area networks or coordinate multi-point systems.
    - Channel protection can be toggled at any time, providing quick security when needed.

**Carriers:**

* Cable
  + Transfers information between logic components - primarily used for configuration. Does not send a power signal. Illuminates blue if transfer was successful.
* Cable Adapter
  + Can be plugged into a normal socket to convert it into a cable socket. Primarily used for configuring certain logic components.
* Cable Coupling
  + A rigid version of the standard Cable. For more information, refer to the Cable's description.
* Cable Extender
  + A two-way transistor that can connect two cables for additional length.
* Contact Socket
  + Carries or outputs a signal when two or more contacts are touching each other. If a Contact Socket is receiving a signal through a wire, it will carry that signal to any other Contact Sockets it is touching.
* Signal Converter
  + Outputs a negative or positive signal depending on which input is true. Primarily used with the Control Box or Remote Socket for controlling a component's direction (e.g., clockwise, anti-clockwise).
* Signal Splitter
  + Duplicates a signal, along with its data (if any), into two separate signals.
* Signal Sweeper
  + A two-way micro-gate that wipes a signal of any data it may be carrying. This can be useful for preventing unwanted "special" data types from lingering onto a signal and influencing the output of certain module/sensors further down the signal's path.
* Wire
  + Carries signals to and from logic components - the fundamental building block for all logic creations. Illuminates yellow when active.
* Wire Coupling
  + A rigid version of the standard Wire. For more information, refer to the Wire’s description.
* Wire Extender
  + A one-way transistor that can connect 2 wires for additional length.

**Bit Module Operators Explained**

'A' and 'B' represent the left and right inputs on the module. If a signal is not carrying bits, the input value will default to 0. It's important to remember that B's bits will always be saved to the module, so if you're using a fixed value, there's no need to supply a constant signal to the input.

Relational Operators

These operators will compare bits sent to A's input to B's bits and output A's signal (along with any data the signal is carrying) if the statement is true, otherwise, nothing will be outputted.

Greater than - If A is greater than B then output A.

Greater than or equal to - If A is greater or equal to B then output A.

Less than - If A is less than B then output A.

Less than or equal to - If A is less than or equal to B then output A.

Equal to - If A is equal to B, then output A.

Not equal to - If A is not equal to B, then output A.

Arithmetic Operators

Outputs the value of the equation between A and B.

Addition (+) - Outputs A + B.

Subtraction (-) - Outputs A - B.

Division (÷) - Outputs the floored (rounded down) value of A ÷ B.

Multiplication (×) - Outputs A × B.

Power (^) - Outputs A ^ B.

Function Operators

These are built in functions that output the value between A and B.

Memory (Floppy Disk with Sync Arrows) - Overwrites the bits in A's signal with the stored bits from B. If A is not carrying any bits, they will be assigned to the signal.

Clamp (Padlock with Vertical Bars) - Clamps A's bits to B. If B's stored value is greater than or equal to 0 (positive), the outputted value will never be above B. So, if A is 3 and B is 5, the outputted value will still be 3. If A is 7 and B is 5, the outputted value will be 5. If B's stored value is less than 0 (negative), the outputted value will never be below B.

Random (Shuffle icon with Sync Arrows) - The outputted value will be a random number between A and B. So, if A is 1 and B is 10, the outputted value would be a random number between 1 and 10.

Modulus (X with Vertical Bars- Outputs the absolute value of A. So, if A is -4, the outputted value would be 4. (B is ignored in this function.)

Modulo (% with Sync Arrows) - Outputs the signed remainder of A ÷ B. So, if A is 24 and B is 10, the outputted value would be 4 because 24 ÷ 10 has a quotient of 2 and a remainder of 4 (10 can go into 24 two times and 4 is the leftover of 24 - 10(2)).

Increment (Approximate Equal icon) - Outputs the rounded value of A in increments of B. So, if A is 12 and B is 8, the outputted value would be 16 because that is the closest number to 12 that is divisible by 8. (If B is 0, it will be defaulted to 1 instead so that nan is not produced.)