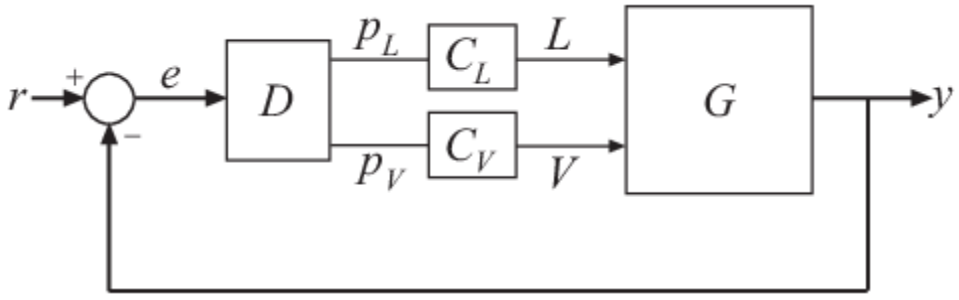
MIMO CONTROL SYSTEM

MIMO control system description:

A system of inputs and outputs can be described as one of four types: SISO (single input, single output), SIMO (single input, multiple output), MISO (multiple input, single output), or MIMO (multiple input, multiple output).

Multiple input, multiple output (MIMO) systems describe processes with more than one input and more than one output which require multiple control loops. Examples of MIMO systems include heat exchangers, chemical reactors, and distillation columns. These systems can be complicated through loop interactions that result in variables with unexpected effects. Decoupling the variables of that system will improve the control of that process.

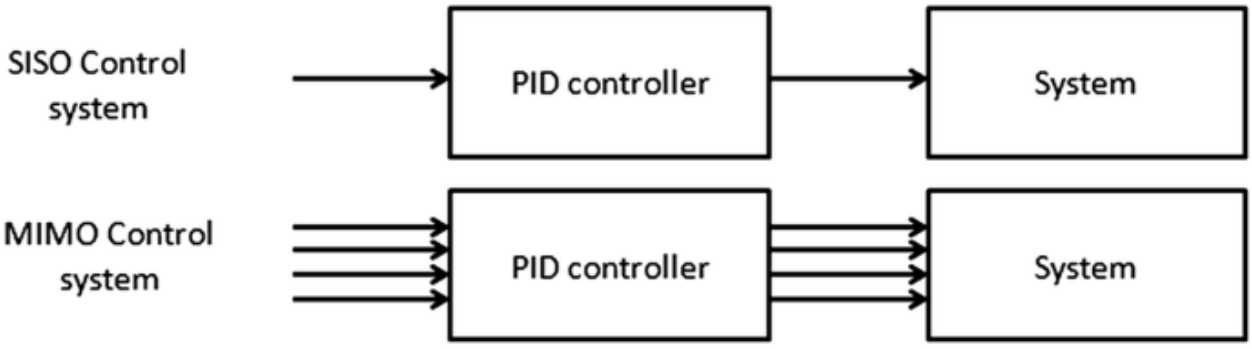
MIMO systems that are lumped and linear can be described easily with state-space equations. To represent multiple inputs, we expand the input *u(t)* into a vector *U*(t) with the desired number of inputs. Likewise, to represent a system with multiple outputs, we expand *y(t)* into *Y*(t), which is a vector of all the outputs. For this method to work, the outputs must be linearly dependent on the input vector and the state vector.



MIMO VS SISO:

Single variable Input or Single variable Output (SISO) control schemes are just one type of control scheme that engineers in industry use to control their process. They may also use MIMO, which is a Multi-Input-Multi-Output control scheme. In MIMO, one or more manipulated variables can affect the interactions of controlled variables in a specific loop or all other control loops. A MIMO control

scheme is important in systems that have multiple dependencies and multiple interactions between different variables.



The common types of input used in the control system are **SISO** (Single Input Single Output) and **MIMO** (Multiple Input and Multiple Output). SISO means that the system produces single output for the single input, while MIMO produces multiple outputs for the multiple inputs. The reference input in a control system is also known as the **set-point**, the desired value. It acts as the basis for error-controlled regulation using negative feedback for error control.

Single-input single-output PID controllers have been used for multiple-input multiple-output (MIMO) plants for many years. This is generally carried out by pairing inputs (actuators) and outputs (sensors) and connecting them with SISO PID controllers. These SISO PID controllers can be tuned one at a time (in 'successive loop closure') using standard SISO PID tuning rules. For MIMO plants that are already reasonably well-decoupled, multi-loop SISO PID design can work well. Unlike SISO PID design, however, MIMO PID design is more complex; the SISO loops must be chosen carefully, and then tuned the correct way in the correct order.

When to use MIMO system?

An alternative approach is to use control theory that combines the designer intuition with rigorous methodologies to generate reliable and optimal adaptation controllers. Specifically, we need what is called as Multiple Input Multiple Output (MIMO) control. In this method, the designer specifies the information he/she has about the design such as the priorities of the output or the overheads of changing inputs. The underlying algorithms generate a controller that meets multiple objectives by actuating on multiple inputs simultaneously. The interactions

between the inputs and outputs are represented in a structured manner and the final controller can take better decisions, being aware of all these interactions. Unfortunately, there has not been prior work that interfaces MIMO control theory with computer architecture. We address this issue by describing MIMO controller design and the architectural insights into this process. This would help the architect to use this powerful tool to create efficient adaptation controllers. The purpose of [1] was to shed light on the MIMO controller design process from the architecture side. It does not describe in detail the specific design methods and analysis one must perform to design a MIMO controller for architecture.

A MIMO controller:

An LQG (Linear Quadratic Gaussian) controller is a type of MIMO controller to minimize tracking errors in the outputs. The controller first infers the state of the system being controlled from the system outputs and then produces system inputs to keep the system outputs close to their desired values. Moreover, the inputs are generated such that a designer specified cost function is minimized. This cost function is the sum of two costs that respectively capture the penalties of not meeting the output targets and the penalties of rapidly changing the inputs. These costs have weights that the designer can specify. This cost function with the weights is shown in Equation 1.

$$J = (\Delta y^T \times Q \times \Delta y) + (\Delta u^T \times R \times \Delta u) \quad [1]$$

In Equation 1, Δy denotes the difference between the values of outputs and the desired reference values for these outputs. Δu is the difference between the current input and the proposed new value of the input. Q is a diagonal matrix with positive entries that indicate the relative importance of meeting different output targets. R is also a diagonal matrix that specifies the relative preference for changing inputs. For example, when we have inputs with different levels of quantization, we would want to apply small changes to the input with many levels, taking advantage of the fine-grained quantization. In another scenario, we might have an input that has a large overhead to change compared to another, such as power gating a core vs resizing the reorder buffer. We would want to prefer changing the low overhead input more often than the higher overhead input. Through the Q and R matrices in the cost function, the architect can convey the design goals to the LQG methodology. LQG control also allows a system to be

characterized as the combination of a deterministic part and an unknown part that follows a Gaussian distribution. This stochastic component accounts for unpredictable effects, program behavior, sensor noise and other effects on the outputs that are not caused by the inputs. This makes the controller more reliable and useful in a wide variety of scenarios. Since we rely on a model for LQG controller design, we need to check that the controller we design is robust to modeling errors. Complicated systems such as a processor cannot always be accurately described using models and model errors are very likely to occur. The controller we design should be able to work correctly and provide stability guarantees even when the true system deviates from this model. This is achieved through Robust Stability. It is a type of analysis that lets the designer specify a model confidence level or guard band for the model accuracy. Then, it checks that the designed controller is robustly stable for all conditions that do not consistently deviate the system from the model by more than the specified guard band.