AR19

CODE: 19MVL1009 **SET-1**

ADITYA INSTITUTE OF TECHNOLOGY AND MANAGEMENT, TEKKALI
(AUTONOMOUS)

I M.Tech II Semester Supplementary Examinations, July, 2025

MIXED SIGNAL IC DESIGN
(VLSI System Design)

Time: 3 Hours Max Marks: 60

Answer any FIVE questions
All questions carry EQUAL marks

1.	a)	Draw the basic circuit of a switched capacitor, its equivalent circuit, explain its operation and derive its equivalent resistor value.	6
	b)	If C1 = C2 = C, find the value of C that will evaluate a 1MΩ resistor if the clock frequency is 200 KHz.	6
2.	a)	What are biquad filters? Explain about the two switched capacitor biquad realizations.	6
	b)	Derive an integrator using switched capacitor circuit.	6
3.	a)	Discuss about non ideal effects in PLL	6
	b)	Draw the block diagram of a charge pump PLL and explain the functions of each block.	6
4.	a)	What are the specifications of a DAC? Explain any four in detail.	6
	b)	Explain about deterministic approach and statistic approach of quantization noise in data converters	6
5.	a)	Draw the circuit diagram of a pipeline ADC and explain its operation.	10
	b)	What is a flash converter?	2
6.	a)	Distinguish oversampling without noise shaping and with noise shaping.	8
	b)	Discuss about Delta-Sigma ADC.	4
7.	a)	Discuss the working of a 3-bit flash A/D Converter.	8
	b)	Write about hybrid converters.	4
8.	a)	What is time interleaving? Explain the operation of a time interleaved ADC.	10
	b)	Define the clock feedthrough	2

AR22

CODE: 22MCM1007 **SET-1**

ADITYA INSTITUTE OF TECHNOLOGY AND MANAGEMENT, TEKKALI
(AUTONOMOUS)
I M.Tech II Semester Regular Examinations, July,2025

Product Design and Development

(COMPUTER INTEGRATED MANUFACTURING)

Time: 3 Hours **Max Marks:60**

Answer any FIVE questions
All questions carry EQUAL marks

1.	a)	Discuss the characteristics of successful product development	6M
	b)	Explain the essential factors of product design	6M
2.		Explain the slot, bus, and sectional Modular architectures	12M
3.	a)	Why is industrial design important in today's era?	6M
	b)	Discuss the terms Preliminary design and detail design	6M
4.		Explain the steps involved in the DFM	12M
5.	a)	Discuss the various aspects of the aesthetic design	6M
	b)	Explain the term Anthropometry	6M
6.	a)	Discuss the need of Concurrent Engineering	6M
	b)	Explain the major factors to be considered for maintainability	6M
7.	a)	Write a short note on Ergonomics	6M
	b)	Explain the significance of Man-Machine interaction	6M
8.	a)	Discuss the need of Rapid Prototyping techniques	6M
	b)	Explain the various types of intellectual property	6M

AR22

CODE: 22MSE1009 **SET-1**

ADITYA INSTITUTE OF TECHNOLOGY AND MANAGEMENT, TEKKALI
(AUTONOMOUS)
I M.Tech. II Semester Regular & Supplementary Examinations, July,2025

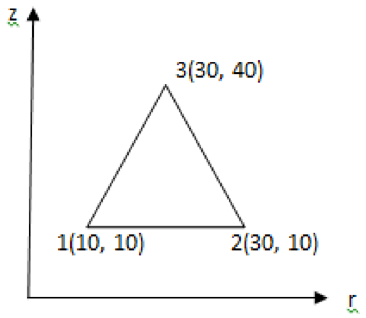
FINITE ELEMENT ANALYSIS

(STRUCTURAL ENGINEERING)

Time: 3 Hours **Max Marks:60**

Answer any FIVE questions
All questions carry EQUAL marks

1.	a)	Explain in detail about the brief history of FEM	4M
	b)	Obtain equilibrium equations for 2-D body.	8M
2.		Briefly discuss element load vector and matrix boundary conditions	12M
3.		Obtain safe function for Four noded rectangular element.	12M
4.	a)	Describe briefly the types of elements	6M
	b)	Explain in detail about isoparametric formulations	6M
5.	a)	Explain briefly about CST element	6M
	b)	Describe briefly the procedure involved in plane stress problems	6M
6.		Nodal coordinates for an Axi-Symmetric element are given below. Evaluate Stiffness Matrix. $E=2x10^5$ N/mm^2, $\upsilon = 0.25$.	12M



7.	a)	Discuss in detail about potential energy principle	6M
	b)	Describe briefly Galerkin Finite Element Method	6M
8.		Obtain shape function for 3 – dimensional hexahedron element	12M

Scheme of evaluation
AR22

CODE: 22MCS1009 **SET-1**

ADITYA INSTITUTE OF TECHNOLOGY AND MANAGEMENT, TEKKALI
(AUTONOMOUS)
I M.Tech. II Semester Regular & Supplementary Examinations, July,2025

Machine Learning

(COMPUTER SCIENCE AND ENGINEERING)

Time: 3 Hours **Max Marks:60**

Answer any FIVE questions
All questions carry EQUAL marks

1. a) Explain Find - S algorithm with an example. 6M

Explanation with examples 6m

What is Find-S?

Find-S is a concept learning algorithm introduced by Tom Mitchell.
It's used to find the most specific hypothesis that fits all the positive examples in a dataset.

- **S** stands for "Specific" → it searches for the most specific hypothesis.
- It works only with positive examples (ignores negative examples).
- It assumes the target concept is conjunctive (AND of attribute constraints).

Algorithm Steps (Find-S)

Given: a set of training examples, each described by attributes.

1. Initialize the hypothesis `h` to the most specific hypothesis possible (often represented as all "Ø" or the most specific values).
2 For each **positive** example:

- If the attribute value in `h` is different from the example, generalize it (replace specific value with `?`).
  3 Ignore negative examples.
  4 At the end, `h` is the most specific hypothesis consistent with all positive examples.

b) What is Machine learning ?why it is important 6M

What is Machine learning 2m

Machine learning is a field of computer science and artificial intelligence (AI) that focuses on creating systems that can **learn from data**, **identify patterns**, and **make decisions** with minimal human intervention.

In other words:

Instead of programming explicit rules to solve a problem, we train a model using data so it can learn the rules by itself.

why it is important 4m

Handles large and complex data

In today's world, huge amounts of data are generated every second (e.g., social media, sensors, medical devices).
ML can process and learn from this data in ways that would be impossible for humans or traditional programming.

---

2 Automates tasks and decisions

ML enables systems to automatically detect spam, recommend products, translate text, or recognize faces — all without needing hand-crafted rules for every possible situation.

This automation:

- Saves time
- Reduces human error
- Increases efficiency

---

3 Improves with experience

Unlike traditional software that stays the same unless updated by programmers, ML systems **can get better over time** as they see more data.

For example:

- A recommendation engine can become more accurate as it learns about your preferences.
- A medical diagnostic model can improve as it's trained on more patient data.

---

4 Solves complex real-world problems

ML can handle problems where:

- The rules are too complex to describe (e.g., speech recognition)
- The environment is constantly changing (e.g., stock market prediction)
- There's a need for personalization (e.g., adaptive learning systems)

---

5 Drives innovation

ML is a core part of technologies shaping the future:

- Self-driving cars
- Personal assistants
- Smart healthcare (early disease detection, personalized treatment)
- Financial fraud detection
- Robotics

2. a) Explain the convergence algorithm with an example. 6M

Algorithm explanation 6m

T he **Candidate-Elimination Algorithm**, sometimes also called the **Convergence Algorithm** in concept learning.

The Candidate-Elimination Algorithm is a machine learning method used to find all hypotheses that are **consistent** with the observed training examples.

It keeps track of:

- **S**: the most **Specific** hypotheses (most narrow/generalized only as needed)
- **G**: the most **General** hypotheses (most broad/specialized only as needed)

Together, **S and G represent the version space**:

The set of all hypotheses consistent with the training data.

The algorithm **converges** toward the target concept by progressively narrowing this version space as it processes new examples.

1. Initialize:
   - `S` = most specific hypothesis possible.
   - `G` = most general hypothesis possible (all attributes are "?").
2. For each training example:
   - If the example is **positive**:
     - Remove from `G` any hypothesis that doesn't classify it as positive.
     - Generalize `S` minimally so it still covers the new positive example.
   - If the example is **negative**:
     - Remove from `S` any hypothesis that incorrectly classifies it as positive.
     - Specialize `G` minimally to exclude the negative example.
3. Repeat until all examples are processed.

b) Describe the Decision Tree Learning algorithm with an Example 6M

What is Decision Tree Learning algorithm 1m

Explanation with examples 5m

A decision tree is a supervised learning method used for **classification and regression** tasks.
It splits data into subsets based on the **value of input attributes**, creating a tree where each internal node represents a test on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label (or a value).

Step-by-step process:

1. **Select the best attribute** to split data based on some measure like:
   - Information Gain
   - Gain Ratio
   - Gini Index
2. **Create a decision node** that splits on this attribute.
3. **Repeat recursively** on each branch/subset:
   - If all instances in the subset belong to the same class → create a leaf node.
   - If there are no remaining attributes → create a leaf node with the most common class.
4. **Stop** when stopping criteria are met (like minimum number of instances, max depth, etc.).

3. a) Derive the backpropagation rule for a multilayer neural network and discuss its convergence. 6M

Derive the backpropagation rule 6m

Backpropagation is a learning algorithm for training multilayer feedforward neural networks.
It works by computing the error at the output layer and propagating it backward to update weights.
The goal is to minimize the error function, often defined as: $E = \frac{1}{2} \Sigma (t - y)^2$, where t is the target and y is the output.
Using the chain rule, the gradient of the error w.r.t each weight is computed.
The weights are then updated using: $w\_new = w\_old - \eta * \partial E/\partial w$, where η is the learning rate.
Properly tuning η ensures convergence: too large may overshoot, too small may slow down learning.
Backpropagation can converge to a local minimum of the error surface.
Over many epochs, it helps neural networks learn complex, nonlinear mappings.

b) Explain about Gradient descent and the Delta rule 6M

Gradient descent and the Delta rule Explanation 6m

Gradient descent is an **optimization algorithm** used to minimize a loss (error) function by iteratively adjusting model parameters (like weights in neural networks).

Key idea:

- Imagine the loss function as a landscape (surface).
- The gradient (vector of partial derivatives) shows the direction of steepest increase.

- To reduce error, we move **against the gradient** — "descending the hill."

How it works:

1. Initialize weights randomly.
2. Compute the error/loss EEE over the dataset.
3. Calculate the gradient.
4. Update weights: step in the direction that reduces error.
5. Repeat until:
   - Convergence (very small gradient),
   - Error stops decreasing,
   - Or after fixed number of epochs.

Delta Rule

The **Delta rule** is a special case of gradient descent, typically used for single-layer perceptrons trained on linearly separable data.

It tells us **how to update the weight** based on the error between actual and target output.

Explanation:

- If the actual output yyy is too low (compared to ttt): t−y>0t - y > 0t−y>0 → w increases.
- If yyy is too high: t−y<0t - y < 0t−y<0 → w decreases.
- It adjusts weights **proportional to the input and the error**.

4. a) Explain boosting algorithm with one example 6M

What is Boosting 1m

Boosting is an **ensemble learning technique** that combines **many weak learners** (models that perform slightly better than random guessing) to form a **strong learner** with high accuracy.

Key idea:

- Train weak models sequentially.
- Each new model focuses more on **examples that previous models got wrong**.
- Combine them (usually by weighted majority voting or sum).

Boosting reduces **bias** and often improves predictive power.

Explain boosting algorithm with one example 6m

- Start by training a weak learner (e.g., a shallow decision tree) on the data.
- Identify which examples were misclassified.

- Increase the weights (importance) of those misclassified examples.
- Train the next weak learner on the updated dataset.
- Repeat for a set number of iterations.
- Combine all weak learners to produce the final prediction.

  - Boosting combines several weak models to get a powerful final model.
  - Each new model corrects errors from previous ones.
  - Example: AdaBoost, Gradient Boosting, XGBoost.

Boosting is widely used in practice because it often gives **high accuracy** on complex datasets.

b) What is model overfitting, and how can it be addressed during model selection? 6M

What is model overfitting 2m

Overfitting is a common problem in machine learning where a model:

- Learns the **training data too well**, including noise and random fluctuations.
- Fits the training data almost perfectly,
- But **fails to generalize** to new, unseen data.

In other words, the model becomes too complex: it memorizes the data instead of learning the true underlying patterns.

and how can it be addressed during model selection 4m

During model selection, the goal is to **choose a model that generalizes well** (good performance on unseen data), not just the one that fits training data perfectly.

Here are common strategies:

---

1. Cross-validation

- Use techniques like **k-fold cross-validation**.
- Instead of relying on training accuracy, evaluate models on **validation data**.
- This helps detect overfitting: a model that scores much higher on training than validation is probably overfitting.

---

2. Regularization

- Add a penalty term to the loss function to discourage overly complex models.
- Examples:
  - **L1 regularization** (Lasso): encourages sparsity.
  - **L2 regularization** (Ridge): discourages large weights.
- Regularization keeps the model simpler, reducing overfitting.

3. Choose simpler models

- Start with simpler models (fewer parameters).
- Only increase complexity if validation performance improves.
- For example: prefer a shallow decision tree over a very deep one unless deeper trees are truly better on validation data.

4. Pruning (for decision trees)

- Grow a complex tree first, then **prune** branches that don't improve validation accuracy.
- This helps avoid trees that perfectly memorize training data.

5. Early stopping

- Especially in neural networks: monitor validation loss during training.
- Stop training when validation loss stops improving, instead of training until the model overfits.

🔷 6. Use more data (if possible)

- More data reduces the chance the model will memorize noise.
- Can also use **data augmentation** (e.g., image transformations) to synthetically increase dataset size.

5. a) State and explain Bayes Theorem. How is it applied in concept learning? 6M

State and explain Bayes Theorem 3m

Bayes Theorem describes how to update the probability of a hypothesis H given some observed evidence E:

(H|E)=P(E)P(E|H)·P(H)

- P(H|E): **Posterior probability** — probability of hypothesis H after seeing evidence E.
- P(E|H)P(E | H)P(E|H): **Likelihood** — probability of observing evidence E assuming hypothesis H is true.
- P(H)P(H)P(H): **Prior probability** — initial belief about hypothesis H before seeing evidence.
- P(E)P(E)P(E): **Evidence probability** — overall probability of observing E under all possible hypotheses.

How is it applied in concept learning 3m

In concept learning, we want to learn a target concept or function that correctly classifies data.

We have:

- Hypotheses H: different possible classifiers/models.
- Training data D: examples labeled with their true classes.

How it is used:

1. **Start with prior beliefs** about which hypotheses are more likely.
2. **Compute likelihood**: for each hypothesis, how likely it is to produce the observed data.
3. **Update**: use Bayes Theorem to get posterior probabilities P(h|D)P(h|D)P(h|D).
4. **Select a hypothesis**:
    - o Choose hypothesis with highest posterior (MAP hypothesis).
    - o Or combine predictions from all hypotheses weighted by their (P(h|D)

b) Describe the k-Nearest Neighbour (k-NN) algorithm. How does it use the instance-based learning approach? 6M

Describe the k-Nearest Neighbour (k-NN) algorithm 6m

The **k-Nearest Neighbour (k-NN)** algorithm is a simple, intuitive, and widely used method for **classification** and **regression**.
It makes predictions based on the **k** closest training examples (neighbors) in the feature space.

How it works (classification):

1 Choose a value of **k** (e.g., k=3).

2 For a new input point xxx:

- Compute the distance between xxx and **all** training points.
- Common distance measures: Euclidean, Manhattan, etc.

3 Find the **k** training points closest to xxx.
These are the "nearest neighbors."

4 Predict the class label by:

- **Majority vote** among these neighbors (the most common class wins).

Instance-based learning (also called **memory-based** or **lazy learning**) is a type of machine learning where:

- The algorithm **stores the training examples (instances)**.
- It **doesn't build an explicit abstract model** during training.

- Instead, when it needs to make a prediction, it **directly uses these stored examples**.

6. a) Explain about Evaluating Machine Learning Algorithms 6M

Explain about Evaluating Machine Learning Algorithms6m

What is evaluation?

Evaluating a machine learning algorithm means:

- Measuring **how well** the model performs on data **it hasn't seen before**.
- The goal is to check if the model **generalizes well** (predicts correctly on new data), rather than just memorizing training data.

Why it is important:

- Helps detect problems like **overfitting** or **underfitting**.
- Lets us compare different models and choose the best one.
- Ensures that our model will work well in the real world.

How do we evaluate?

1. Splitting the data

- **Training set**: used to train the model.
- **Test set**: used only to evaluate final performance.
- Sometimes also use a **validation set** to tune hyperparameters.

---

2. Cross-validation

- A technique like **k-fold cross-validation** helps get a more reliable estimate.
- Process:
  - Split data into k equal parts (folds).
  - Train the model on k−1 folds, test on the remaining fold.
  - Repeat k times, each time with a different fold as test set.
  - Average the results.

Benefits:

- Reduces bias from unlucky splits.
- Makes full use of available data.

---

3. Evaluation metrics

Different tasks use different metrics:

For classification:

- **Accuracy**: % of correctly classified instances.
- **Precision & Recall**: for imbalanced classes.
- **F1 score**: harmonic mean of precision and recall.
- **ROC curve & AUC**: measure trade-off between true positive rate and false positive rate.

For regression:

- **Mean Squared Error (MSE)**.
- **Root Mean Squared Error (RMSE)**.
- **Mean Absolute Error (MAE)**.
- **R^2 score** (coefficient of determination).

⚖️ 4. Other considerations:

- **Confusion matrix**: shows true positives, false positives, false negatives, and true negatives.
- **Bias-variance trade-off**: understanding whether errors come from overly simple or overly complex models.
- **Domain-specific metrics**: sometimes accuracy isn't enough — e.g., in medical diagnosis, false negatives might be more costly.

b) Explain how Genetic Algorithms work using an illustrative example. What are selection, crossover, and mutation? 6M

Explain how Genetic Algorithms work using an illustrative example 6m

A Genetic Algorithm is a **search and optimization technique** inspired by the process of natural evolution.
It uses mechanisms like **selection**, **crossover**, and **mutation** to evolve better solutions over time.

They are useful for complex problems where traditional methods struggle.

Basic steps of a Genetic Algorithm:

1. **Initialization**: Generate an initial population of candidate solutions randomly.
2. **Evaluation**: Calculate the fitness of each solution (how good it is).
3. **Selection**: Choose the better solutions to become parents.
4. **Crossover (Recombination)**: Combine pairs of parents to produce new offspring.
5. **Mutation**: Randomly alter offspring slightly to introduce variation.
6. **Replacement**: Form a new generation, often by keeping the best solutions

Repeat steps 2–6 until a stopping criterion is met (e.g., enough generations, or a satisfactory fitness level).

Selection:

- Chooses which solutions become parents.
- Favors solutions with higher fitness → simulates survival of the fittest.

Crossover (Recombination):

- Combines parts of two parents to create new offspring.
- Helps explore new regions in the solution space.

Mutation:

- Randomly changes small parts (e.g., flips bits).
- Maintains genetic diversity → prevents getting stuck in local optima.

7. a) Discuss about explanation-based learning of search control knowledge 6M

Discuss about explanation-based learning 6m

Explanation-Based Learning is a type of **analytical learning** where:

- The system **analyzes** why a particular example is an instance of a target concept.
- It creates a **general rule** from this explanation.
- Instead of learning purely from statistical patterns in many examples, EBL relies on **understanding** a single or few examples deeply.

In simple words:

The system "explains" to itself why a training example works, and then learns a rule that can be reused.

EBL for search control knowledge:

In problem solving, especially in AI planning or search:

- The system often needs to explore many possible choices (large search space).
- **Search control knowledge** helps **guide the search**, focusing only on promising paths.

EBL helps by:

- Observing successful solutions.
- Explaining why these solutions work.
- Learning general rules (control rules) to **prune** irrelevant parts of the search

space next time.

Illustrative example:

Imagine a robot trying to stack blocks.

- It solves the task: stack Block A on Block B.
- The robot analyzes **why** the steps it took led to success.
- It sees that it must first clear Block B, then pick up Block A, and finally place it on Block B.

EBL generalizes:

- Learns: "To stack X on Y, ensure Y is clear before picking X."
- This becomes **search control knowledge**: next time, the robot won't waste time trying to pick X before Y is clear.

b) Explain about Analytical Learning 6M

Explain about Analytical Learning 6m

What is Analytical Learning?

Analytical learning is a type of machine learning that:

- Uses **prior domain knowledge** (like rules, theories, or logic) to help learn.
- Learns by **analyzing** why a particular example belongs to a concept.
- Focuses on understanding and reasoning rather than purely relying on patterns in large datasets.

It contrasts with **inductive learning**, which generalizes from many examples without needing deep domain knowledge.

How it works (key idea):

- Given:
  - A **domain theory** (e.g., rules about how the world works).
  - A **specific training example** (e.g., an object that belongs to a concept).
- The system explains why the example fits the concept.
- Then, it **generalizes** this explanation into a rule that applies to new instances.

In simple terms:

"If I know *why* this example is positive, I can find what makes it positive in general."

Example:

Suppose we want to learn what makes a good chess move.

- The system knows domain knowledge about chess rules and tactics.
- It observes that moving the queen to a certain square led to checkmate.
- By explaining *why* this worked (e.g., the opponent's king was trapped), it can generalize:
    - "A move that attacks a piece and simultaneously threatens checkmate is likely good."

It learns this rule **without needing thousands of games** — just by reasoning about one example.

8. a) Explain Q-learning. How is it applied in environments with deterministic rewards and actions? 6M

Explain Q-learning. How is it applied in environments with deterministic rewards and actions? 6m

What is Q-learning 2m

Q-learning is a type of **model-free reinforcement learning algorithm**.
Its goal is to learn the **optimal action-selection policy**: what action to take in each state to maximize total future reward.

It does this by learning a function called the **Q-function**:

$$Q(s, a)$$

which estimates the **expected total reward** of taking action aaa in state sss and then following the best policy afterward.

Explanation 4m

How Q-learning works (step by step):

1. Initialize the Q-table arbitrarily:

    Q(s,a)←0Q(s, a)

2. At each time step, in current state sss:

- Choose an action aaa (often using ε-greedy policy: mostly best action, sometimes random for exploration).

3. Take action aaa; observe:

- Reward r
- New state s′

4 . Update Q-value using the update rule:

Q(s,a)←Q(s,a)+α[r+γa′maxQ(s′,a′)−Q(s,a)]

where:

- α\alphaα: learning rate
- γ\gammaγ: discount factor (importance of future rewards)

5. Repeat until convergence.

Environments with deterministic rewards and actions:

In deterministic environments:

- The next state s′s's′ and reward rrr are **always the same** for a given state-action pair.
- No randomness in transitions or rewards.

Q-learning still works, but:

- Converges faster because:
  - Each action has a predictable effect.
  - The learned Q-values stabilize quickly.
- The Q-table becomes an accurate reflection of the true optimal policy after fewer episodes.

Example:

- Moving right from (2,2) always goes to (2,3) and always gets the same reward.
- Q-learning will quickly learn the optimal path because every experience is consistent.

b) What is Temporal difference? Explain. 6M

What is Temporal difference 2m

Temporal Difference (TD) learning is a class of **model-free reinforcement learning methods**.
It learns by combining:

- **Ideas from dynamic programming** (bootstrapping: updating estimates based partly on other learned estimates)
- **Ideas from Monte Carlo methods** (learning from raw experience, without knowing the environment's model)

In TD learning, we update predictions **partially based on the difference between successive predictions** — this difference is called the **temporal difference error**.

Explanation 4m

Suppose an agent is in state s, takes action a, receives reward rrr, and moves to new state s′.

In TD(0) (the simplest TD method), we update the value of s:

$$V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$$

where:

- α\alphaα: learning rate
- γ\gammaγ: discount factor

The term:

$$\delta = r + \gamma V(s') - V(s)$$

s the **temporal difference error** — the difference between:

- The current estimate V(s)V(s)V(s)
- A better estimate: r+γV(s′)r + \gamma V(s')r+γV(s′)

AR22

CODE: 22MVL1009 **SET-1**

ADITYA INSTITUTE OF TECHNOLOGY AND MANAGEMENT, TEKKALI
(AUTONOMOUS)
I M.Tech. II Semester Supplementary Examinations, July,2025

MIXED SIGNAL IC DESIGN
(VLSID)

Time: 3 Hours **Max Marks:60**

Answer any FIVE questions
All questions carry EQUAL marks

1.	a)	Discuss key technical challenges and market challenges before mixed signal IC design engineers and also discus some important manufacturers leading the market in the field of Mixed signal ICs.	6M
	b)	Explain charge injection and clock-feedthrough in MOS switch.	6M
2.	a)	Draw the circuit diagram and explain the working of a Switched Capacitor integrator.	6M
	b)	Write short notes on Switch sharing in Switched capacitor circuits.	6M
3.	a)	Design a switched capacitor realization for a first order, high pass circuit with a high frequency gain of -10 and a -3dB frequency of 1 kHz using a clock of 100kHz.	6M
	b)	Discuss the non-ideal effects in switched capacitor circuits.	6M
4.	a)	Explain the dynamics of a simple PLL Circuit.	6M
	b)	Explain the Jitter in PLLs and delay locked loops.	6M
5.	a)	Draw the block diagram of a charge pump PLL and explain the functions of each block.	6M
	b)	Differentiate the working of analog PLL circuits from digital PLL Circuits.	6M
6.	a)	Discuss the DC and dynamic specifications of Data Converters.	6M
	b)	Explain about Decoder based converters.	6M
7.	a)	What is a flash converter? Explain the function of a 3 bit flash ADC.	6M
	b)	Explain about Interpolating A/D converters.	6M
8.	a)	Explain about Noise shaping modulators.	6M
	b)	Describe about Decimating filters in detail.	6M

AR22

CODE: 22MPE1009 **SET-1**

ADITYA INSTITUTE OF TECHNOLOGY AND MANAGEMENT, TEKKALI
(AUTONOMOUS)
I M.Tech. II Semester Supplementary Examinations, July, 2025

POWER ELECTRONIC CONTROL OF AC DRIVES
(POWER ELECTRONIC DRIVES)

Time: 3 Hours **Max Marks: 60**

Answer any FIVE questions
All questions carry EQUAL marks

1.	a)	Compare the volt/Hz control of voltage fed inverter drive with current fed inverter drive	6M
	b)	Explain the speed torque characteristics of induction motor drive with slip regulation and with torque control.	6M
2.	a)	Explain static Kramer drive with neat diagram.	6M
	b)	Briefly discuss the operation of static Scherbius's drive.	6M
3.	a)	Describe vector control scheme with neat diagram.	6M
	b)	Explain induction motor characteristics in constant torque and field weakening regions.	6M
4.	a)	Discuss different control strategies of synchronous motor with reference to its characteristics.	6M
	b)	Illustrate the operation of the field weakening controller with a diagram.	6M
5.	a)	Draw the block diagram of the PMSM drive control strategy and explain the function of each block.	6M
	b)	Discuss the working of a BLDC motor drive when fed from voltage source inverter on closed loop operation. Draw neat circuit diagram.	6M
6.	a)	Draw neat drive characteristics of Variable reluctance motor drive and also discuss about the torque production in this drive.	6M
	b)	Explain the operation of current controlled variable reluctance drives.	6M
7.	a)	Explain field weakening operation in permanent magnet synchronous motors.	6M
	b)	Draw the circuit diagram of a three phase brushless DC (BLDC) motor drive and discuss its operations.	6M
8.	a)	Explain speed torque characteristics with variable voltage operation, variable frequency operation, constant v/f operation and variable stator current operation in induction motor.	6M
	b)	Explain load commutated inverter controlled synchronous motor drive.	6M