



National University of Technology (NUTECH)

CEN4119 – Embedded System Design

Category: Depth Elective

Complex Engineering Problem (CEP) / Activity (CEA)

Course Title	: Embedded System Design	Course code	: CEN4119/CEN4120
Batch	: 2020	Credit Hrs	: 3/1
Instructor	: Dr. Abdul Rehman Buzdar	Weightage	: 10%
Semester	: Spring 2024 (8th)	Time	: 6 weeks

CEP/CEA Title: Embedded System Design of ECG Classification for continuous Cardiac monitoring on wearable devices

CEP/CEA Statement:

The objective of this project is to **develop** a portable ECG classification system tailored for the convenience of individuals with heart conditions. This endeavor focuses on devising an embedded system design leveraging the wavelet transform technique alongside multiple long short-term memory (LSTM) recurrent neural networks. The system architecture comprises two main components: a feature extraction (FE) module and a classification module.

The FE module is tasked with executing a four-level Daubechies discrete wavelet transform to align with the physiological bands of the electrocardiogram (ECG or EKG) signal. It then proceeds to extract time-frequency domain features that reflect the nonstationary properties of the signal.

On the other hand, the classification module integrates several LSTM recurrent neural networks, fully connected layers, and a multilayer perceptron (MLP) to carry out the classification tasks efficiently. Through this integrated approach, the system aims to accurately differentiate between normal and abnormal ECG patterns, facilitating real-time monitoring of cardiac health.

The problem of ECG classification has no unique solution and requires the study and application of suitable ECG classification algorithm. The proposed solution should be based on a reasonable **assumption**. Also, apply the proposed solution on an actual public data set to **justify** your solution.

The CEP is mapped on the following attributes:

WP1	Preamble
WP2	Range of conflicting requirements
WP3	Depth of analysis required
WP4	Depth of knowledge required
WP5	Familiarity of issues
WP6	Extent of applicable codes

WP1:

The project involves developing a portable ECG classification system that leverages advanced signal processing and machine learning techniques. It aims to facilitate real-time monitoring of cardiac health for individuals with heart conditions by accurately differentiating between normal and abnormal ECG patterns. The system will use a four-level Daubechies discrete wavelet transform for feature extraction and integrate LSTM recurrent neural networks, fully connected layers, and an MLP for classification.

WP2:

- **Accuracy vs. Portability:** Ensuring high accuracy in classification while maintaining the system's portability.
- **Real-time Processing vs. Power Consumption:** Achieving real-time processing of ECG signals without excessive power consumption.
- **Complexity vs. Usability:** Balancing the complexity of the algorithms and hardware with the ease of use for end users.
- **Robustness vs. Cost:** Developing a robust and reliable system without significantly increasing the cost.

WP3:

- **Signal Processing:** Detailed analysis of the ECG signal's characteristics and the effectiveness of the four-level Daubechies discrete wavelet transform in feature extraction.
- **Machine Learning:** In-depth analysis of LSTM recurrent neural networks and their ability to classify ECG patterns accurately.
- **System Integration:** Comprehensive analysis of integrating the FE module with the classification module to ensure seamless operation.

WP4:

- **Electrophysiology:** Understanding of ECG signal properties and cardiac physiology.
- **Wavelet Transform:** Expertise in discrete wavelet transform, particularly the Daubechies wavelet, for feature extraction.
- **Deep Learning:** Proficiency in LSTM recurrent neural networks, fully connected layers, and MLP for classification tasks.
- **Embedded Systems:** Knowledge of embedded system design for implementing the ECG classification system in a portable format.

WP5:

- **Signal Noise:** Challenges associated with noise and artifacts in ECG signals and their impact on classification accuracy.
- **Data Variability:** Dealing with the variability in ECG signals due to different patients, conditions, and recording environments.
- **Real-time Constraints:** Issues related to real-time processing capabilities and ensuring low latency in classification.

WP6:

- **Medical Device Regulations:** Compliance with relevant medical device regulations and standards for ECG monitoring devices (e.g., FDA, CE mark).
- **Data Privacy and Security:** Adherence to data privacy and security standards, particularly when dealing with sensitive health information (e.g., HIPAA, GDPR).
- **Software Development Standards:** Following best practices in software development, including code quality, testing, and documentation.

The CEA is mapped on the following attributes:

EA1	Preamble
EA 2	Range of resources
EA 3	Level of Interaction
EA 6	Familiarity

EA1:

This project aims to create a portable ECG classification system for individuals with heart conditions. It involves developing an embedded system that uses wavelet transform techniques and multiple LSTM recurrent neural networks to analyze and classify ECG signals. The system includes a feature extraction module and a classification module to facilitate real-time monitoring of cardiac health by distinguishing between normal and abnormal ECG patterns.

EA2:

- **Hardware Resources:** Embedded systems platform (e.g., microcontroller or FPGA), ECG sensors, portable power supply (e.g., battery).
- **Software Resources:** Development environment for embedded systems, libraries for wavelet transform and machine learning (e.g., TensorFlow, PyTorch), data processing tools.
- **Data Resources:** Public ECG datasets for training and validating the classification model (e.g., MIT-BIH Arrhythmia Database, PhysioNet databases).
- **Human Resources:** Expertise in electrophysiology, signal processing, machine learning, embedded systems, and software development.
- **Financial Resources:** Budget for hardware components, software licenses, and potential costs for regulatory compliance and testing.

EA3:

- **Interdisciplinary Collaboration:** The project requires collaboration among experts in medical sciences (cardiology), signal processing, machine learning, and embedded systems engineering.
- **User Interaction:** Regular feedback from end users (patients and healthcare professionals) to ensure the system meets usability and functionality requirements.
- **Stakeholder Communication:** Engagement with stakeholders such as healthcare providers, regulatory bodies, and investors to align project objectives with practical needs and compliance standards.

- **Team Coordination:** Effective communication and coordination within the project team to integrate different system components and ensure seamless operation.

EA6:

- **Signal Processing and Wavelet Transform:** Familiarity with discrete wavelet transform, particularly the Daubechies wavelet, for feature extraction from ECG signals.
- **Machine Learning and LSTM Networks:** Proficiency in LSTM recurrent neural networks, including their training and optimization for time-series data like ECG signals.
- **Embedded Systems Development:** Experience in designing and implementing embedded systems, including hardware-software integration for portable medical devices.
- **Regulatory and Compliance:** Understanding of medical device regulations, data privacy, and security standards relevant to ECG monitoring systems.
- **Data Variability and Real-time Processing:** Knowledge of handling variability in ECG data and ensuring the system can process and classify signals in real-time.

Deliverables

1. Selecting an appropriate ECG classification algorithm from the literature
2. Hardware/Software Design of ECG classification embedded system and its testing for accurate operation using actual public data set.
3. Write a report containing the details of your project design (hardware, software), testing of your complete hardware solution on an actual public data set.

Through the course of this engineering activity, students should be able to partially attain some or all of the following graduate attributes which are being covered in Embedded System Design Theory and Lab.

GA2: Problem Analysis:

GA3: Design/Development of Solutions

GA4: Investigation

GA5: Modern Tool usage

GA11: Project Management

In this theory course, this activity maps onto the following CLOs as mentioned in course plan and your performance in this project will play an important role in the attainment of this course learning outcomes and will be evaluated against **PLO-4**.

S#	CLO	Learning Domain	Taxonomy Level	PLO
1	Analyze different embedded system design technologies, explain the various metrics or challenges in designing an embedded system.	Cognitive	4	2
2	Design an embedded system by integrating an embedded processor, memory, buses, peripheral components, and establishing their interconnections.	Cognitive	6	3
3	Evaluate how architectural and implementation design decisions influence performance, area and power dissipation in embedded systems.	Cognitive	5	4

In lab course, this activity maps onto the following CLOs as mentioned in course plan and your performance in this project will play an important role in the attainment of this course learning outcomes and will be evaluated against **PLO-11**.

S#	CLO	Learning Domain	Taxonomy Level	PLO
1	Perform experiments in the laboratory to acquire technical skills in using software/hardware tools for embedded system design	Psychomotor	4	5
2	To design an embedded system using Microcontrollers/SoC and Practice as per a well-defined project plan adhering to deadlines and resource constraints.	Affective	4	11

Project Description:

A portable ECG classification system is very convenient for heart patients to carry. This project aims to implement an embedded system design based on the wavelet transform and multiple long short-term memory (LSTM) recurrent neural networks. The design may consist of a feature extraction (FE) module and classification module. The FE module performs the four-level Daubechies discrete wavelet transform to fit the physiological bands of the electrocardiogram (ECG or EKG) signal and extracts the time-frequency domain features reflecting the nonstationary signal properties. The classification module integrates multiple LSTM recurrent neural networks, fully connected layers, and a multilayer perceptron (MLP).

ECG Classification

The view of the suggested algorithm is shown in Fig. 1. At the beginning, in the segmentation phase, the approaching ECG samples are fragmented into heartbeats and their RR interval features and then, the wavelet features are extracted. After that, this ECG signal in conjunction with its extracted features are pass to the RNN network that performs classification for every heartbeat signal and results in the form of two outputs. At that point these two outputs are blended to create a final classification for each heartbeat.

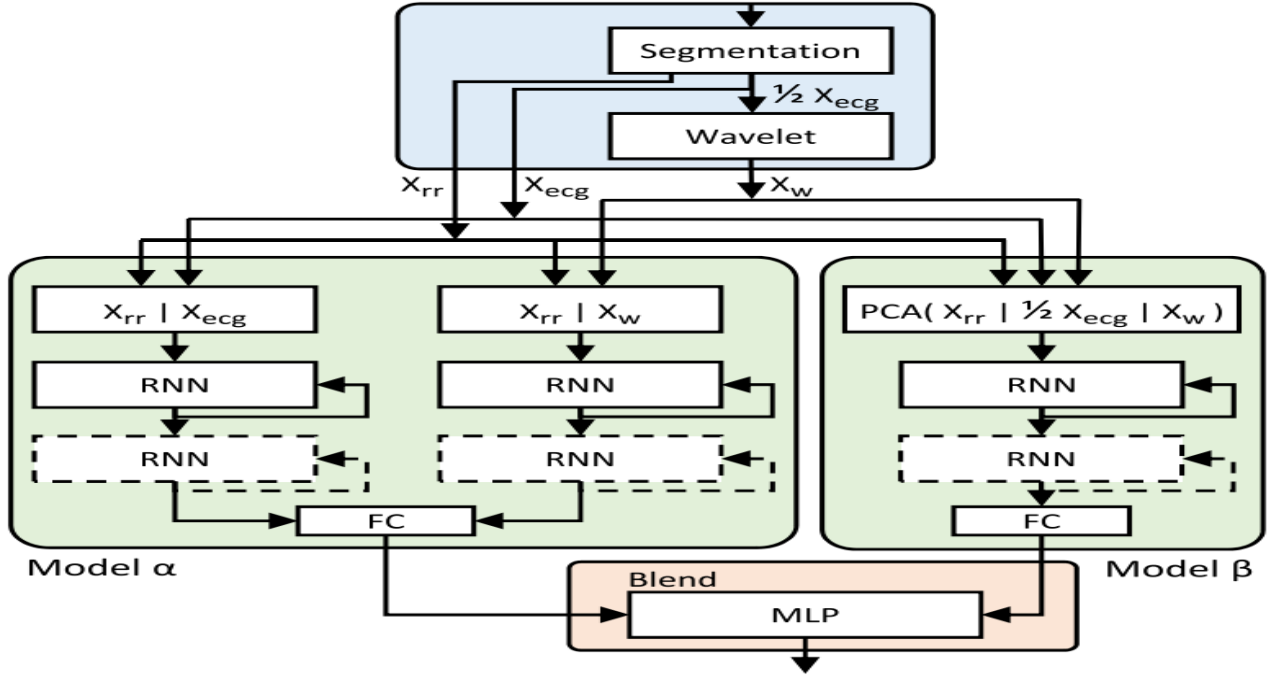


Figure 1: LSTM-Based ECG Classification Algorithm

a) Segmentation and RR interval Features

After the digitization of the ECG signals, they are segmented into a series of heartbeats. It is carried out on the basis of R peaks. Each segment consists of 0.45 sec of the ECG signal after the R peak and 0.25 sec before it and every segment has fixed length. It is denoted by X_{ecg} as shown in Fig. 1.

The R peak algorithms are highly accurate and well established. We used PanTompkin's algorithm for our segmentation process. Moreover, the time span between the consecutive R peaks is calculated, as it's a part of Pan-Tompkin's algorithm. Let the time span from R peak $i - 1$ to R peak i is denoted by RR_i . So, by these four different features are extracted for heartbeat i : I) RR_i as the past RR interval, II) RR_{i+1} as the next RR interval, III) $\frac{1}{10} \sum_{k=i-4}^{i+5} RR_k$ as the local average of the five past and the five next RR intervals, and IV) the average duration of the RR intervals in each person's train data. These four features form a feature vector X_{rr} and are referred to as RR interval features as in Fig. 1.

Note that heartbeat features are required by the feature number II and III. However, the feature information is not available in our setting as opposed to preprocessing previously stored ECG signals. The reason of this is that the proposed algorithm is designed for persistent monitoring. So to overcome this problem we use a concept to buffer the ECG signal in real time inside the first-in-first-out memory. As the buffer is designed in the software so it must be small. The heartbeats that fall within the center of the buffer, by classifying them, made it possible to access the near future and near past information.

The last feature is the average duration of the RR intervals in each person's train data. This feature differs between peoples, because of the diverse average heart rates. For instance, because of athlete's lower heart rates, it is larger.

Apart from the minimal computation and accurately extraction of the above RR interval features, the hand-crafted morphological features like those that are dependent on T, S or Q are not employ by the proposed algorithm. The reason for this is that to represent the characteristics of underlying signals, such features are not optimal. Secondly, they didn't be able to effectively represent the variation among the arrhythmia classes, as they are fixed under all circumstances for all patients. So, by using RNN and wavelet, we let the features to be extracted automatically.

b) Wavelet Features

The characteristics of ECG signals are non-satisfactory. So, in every heartbeat sample, the discrete wavelet transform is implemented to digitize ECG samples to capture both the frequency and time domain knowledge. In particular, due to the resemblance with the ECG signal Daubechies wavelet

family is selected. High-order Daubechies have low time resolution and high frequency resolution, while low-level have low frequency resolution and high time resolution. Types 1 to 4 was mostly employed by the previous works. We employ type T = 2, i.e., levels of decomposition L = 4, and db2, which drop somewhat in the middle of this range. Therefore, the final wavelet coefficients list is consisting of $X_\omega = (D1, D2, D3, D4, A4)$.

The discrete wavelet transform's computational complexity of an input array of size N with Daubechies, L levels of decomposition and type T is

$$N \times T \times \left(1 + \frac{1}{2} + \dots + \frac{1}{2^{L-1}}\right) \quad (1)$$

Before applying the wavelet transform the wavelet input X_{ecg} as shown in the Figure is down sample by the factor of 2, because the input size is proportional to the computational requirement. For every heartbeat signal because of the down sampling, it cuts the length of the wavelet output to about half, which results in the reduction of the computational requirement.

c) RNN-Based Models

For each heartbeat signal, two different RNN-based models named as model α and β are provided by the help of input ECG samples (X_{ecg}) in company with wavelet features (X_{rr}, X_ω) and the extracted RR interval features. The view of our proposed algorithm is shown in Figure. From this, it is clear that the two models α and β are making separate arrhythmia predictions and then the final prediction for every heartbeat is form by the blending of these predictions.

RNN models with the help of X_{rr} and X_ω along with X_{ecg} , captures the ECG signal patterns more efficiently. Refined information is provided to the RNN model by the help of additional features. Thus, the accurate results can be reached with smaller and hence faster RNNs. Without even significantly increase in the computational cost, increase in the accuracy is achieved, by using the multiple smaller RNNs in parallel instead of one large RNN.

In model α , the X_{ecg} and X_ω are first processed individually and then combine the outputs. But in model β , X_{ecg} and X_ω are first combine and then processed. The detail about this is discussed below.

Model α : Model α is composed of two branches as shown in Figure. One or two RNNs are there in each branch and every RNN comprises of several hidden units. $X^{\alpha 1}$ is the input to the left branch by and is made by concatenating X_{ecg} and X_{rr} . The RNN cells process the array $X^{\alpha 1}$ and extract the feature $N_h^{\alpha 1}$. Likewise, the $N_h^{\alpha 2}$ feature is extracted in the right branch, by concatenates the X_ω and X_{rr} into array $X^{\alpha 2}$ and then process it.

To build the probability of all the N_y output arrhythmia classes the outputs of the two branches are concatenated and pass into a fully connected neural network layer. $(N_h^{\alpha 1} + N_h^{\alpha 2}) \times N_y$ is the dimension of this fully connected layer. The maximum

probability determines the arrhythmia class that is predicted by model α .

Model β : Model β is composed of only one branches as shown in Figure. Instead of processing the X_ω and X_{ecg} by using the two independent RNN branches and then blending their outputs, here in this the inputs are merge. In particular, by joining the down sampled version of X_ω with X_{ecg} and X_{rr} , followed by applying PCA on the joined array, the X^β array is formed. Then N_h^β features are extracted by process the

X^β with the help of RNNs in this model. With dimension $N_h^\beta \times N_y$ these features are pass into a fully connected NN layer.

Number of hyper-parameters, namely, the number of hidden units in every RNN, the number of RNNs in every branch, and the RNN cell types are include in Model α and β . Different RNN cell types are discussed below.

Input vector at time t is named as x_t . State vectors c_t and h_t are carried from time t - 1 to t, and hence, they are encoding the previous information so act as a memory. h_t is also considered as the cell output. N_h also known as the number of hidden units is the size of vectors c and h. The cell works based on the following equations.

$$m_t[j] = \tanh\left(\sum_{k \in [1, N_x]} \omega_{xm}[j, k]x_t[k] + \sum_{k \in [1, N_h]} \omega_{hm}[j, k]h_{t-1}[k] + b[j]\right) \quad (2)$$

$$c_t[j] = c_{t-1}[j] + m_t[j] \quad (3)$$

$$h_t[j] = \tanh(c_t[j]) \quad (4)$$

As shown in (2), on the linear combination of h_{t-1} and x_t i.e., previous output and current input, respectively, by applying the tanh activation function, an intermediate vector is formed named as m_t . $j \in [1, N_h]$. During the training phase the bias vector b and the weight matrices ω_{xm} and ω_{hm} are determined. By collecting the m_t over time, the state vector named as c_t is formed, as shown in (3). Likewise, by applying the tanh activation function on state vector c_t , the output vector names as h_t are formed. It can be seen that the output is related to all previous inputs.

LSTM (Long Short-Term Memory): In the above simple RNN cell the effect of all past information is collected in the internal state vector. When temporal dependencies get too long, the Gradient-based algorithms may fail because the gradient values may decrease or increase exponentially.

By allowing to forge according to the actual dependencies which exist in the problem LSTM solves this issue. Based on the data the dependencies are automatically extracted. This is achieved through forget, input and output gates. Fig. 7 shows the LSTM cell. Based on h_{t-1} and x_t the gate signals are formed as shown below.

$$f_t[j] = \sigma \times \left(\sum_{k \in [1, N_x]} \omega_{xf}[j, k] x_t[k] + \sum_{k \in [1, N_h]} \omega_{hf}[j, k] h_{t-1}[k] + b_f[j] \right) \quad (5)$$

$$i_t[j] = \sigma \times \left(\sum_{k \in [1, N_x]} \omega_{xi}[j, k] x_t[k] + \sum_{k \in [1, N_h]} \omega_{hi}[j, k] h_{t-1}[k] + b_i[j] \right) \quad (6)$$

$$o_t[j] = \sigma \times \left(\sum_{k \in [1, N_x]} \omega_{xo}[j, k] x_t[k] + \sum_{k \in [1, N_h]} \omega_{ho}[j, k] h_{t-1}[k] + b_o[j] \right) \quad (7)$$

In the above equations, the sigmoid activation function is denoted by σ , and $j \in [1, N_h]$. m_t is already computed in the LSTM Cell in (2), but have to modified the equations (3) and (4) based on the forget, output and input gate signals as the following.

$$c_t[j] = f_t[j] \times c_{t-1}[j] + i_t[j] \times m_t[j] \quad (8)$$

$$h_t[j] = o_t[j] \times \tanh(c_t[j]) \quad (9)$$

The forget gate f_t at time $t-1$ to t controls the carrying of state vector c , as shown in the equation (8). The accumulation of m_t in c_t is adjusted by the input gate. By applying activation function named as tanh on c_t , and then adjusted by the output gate o_t , the output h_t is formed as shown in the equation (9).

The LSTM output still depends on all the past inputs, as shown in the above equations. Previous information is neither completely carried over to the current state nor completely discarded. Rather, by the help of the gate signals, the effect of the past knowledge on the present state is carefully controlled. LSTM with peepholes: by adding the additional connections from the internal state vector to the output, input and forget gates the LSTM cell can be extended. By the linear combination of the h_{t-1} , x_t and now also c_{t-1} the gate signals are formed.

GRU (Gated Recurrent Unit): this is a simplified version of the LSTM cell which employs the different gating strategy and also blends the two state vectors into one. Here, c_t is formed as

$$c_t[j] = (1 - z_t[j]) \times c_{t-1}[j] + z_t[j] \times \tanh \left(\sum_{k \in [1, N_x]} \omega[j, k] x_t[k] + \sum_{k \in [1, N_h]} v[j, k] r_t[k] c_{t-1}[k] + b[j] \right) \quad (10)$$

Where, r_t and z_t are update and reset gate signals, and are formed same as the LSTM gate signals as linear combinations of c_{t-1} and x_t .

Complexity Analysis: Several matrix and vector operations are performed by the above RNN cells. For instance, the LSTM cell requires 4 matrix vector multiplications of size $N_h \times N_h$, 4 matrix vector multiplications of size $N_h \times N_x$, and several vector operations of size N_h . So, the total computational complexity for every RNN cell execution is therefore equal to

Where a, b, c and d are dependent on the type of cell. From the above equation, w.r.t the number of hidden units i.e., N_h , the computational complexity of a RNN cell has a quadratic growth. Hence, one larger RNN have high computational cost in total as compared to multiple smaller RNNs.

To speed up the classification accuracy, ensemble methods such as blending are designed to blend the predictions made by multiple learning models. In order to keep the computational cost as minimum as possible, in our algorithm only two models are blended as shown in the Figure. First, the probability of all the N_y outputs of arrhythmia classes is independently compute by both the RNN-based models β and α . Then to form the final probability of the N_y output of arrhythmia classes these two outputs are blended. By using the MLP with two hidden layers the blend model is implemented. The output and input layers have N_y and $2 \times N_y$ neurons, respectively.

Implementation Decision:

You must determine the implementation approach for the ECG classification system. Whether to deploy it entirely in software on a microcontroller, or opt for a hardware-centric solution by leveraging FPGA/ASIC or Zynq SoC using a hardware-software approach. You can either utilize the provided algorithm or adapt some other ECG classification algorithm. Assess the impact of architectural and implementation design choices on the performance, area and power consumption of this embedded system.


Additionally, you need to choose between a Bare-Metal system or a real-time operating system (RTOS) for the ECG classification system.

Rubrics for Assessment of the CEP/CEA

<u>I Hardware</u>	Unacceptable (0-10)	Just acceptable (11-15)	Basic (16-20)	Good (21-25)	Excellent (26-30)
	Not Prepared	Incomplete	All hardware settings and signals are readable.	Hardware Results are completely working.	Hardware has a product shape with fine working.
<u>II. Demonstration</u>	(0-3)	(4-6)	(7-9)	(10-12)	(13-15)
	No Understanding	Couldn't demonstrate well / Little bit understanding	Understand Project well but couldn't demonstrate well	Understand well and demonstrate well.	Have a complete understanding and can relate project to real life problems
<u>III. Presentation</u>	(0-5)	(6-10)	(11-20)	(20-25)	(26-30)
	Not Prepared	Incomplete presentation	Prepared but results are incorrectly explained.	Prepared well and could explain little bit.	Complete results are included (samples, testing etc.) and Present Diligently.
<u>IV. Project Report</u>	(0-5)	(6-10)	(11-15)	(16-20)	(21-25)
	No report /Report was not prepared in time	The report submitted but incomplete	The requirements of report are not properly addressed	Report meets all prescribed requirements with minimum mistakes	Report meets all requirements and prepared in original and creative way.

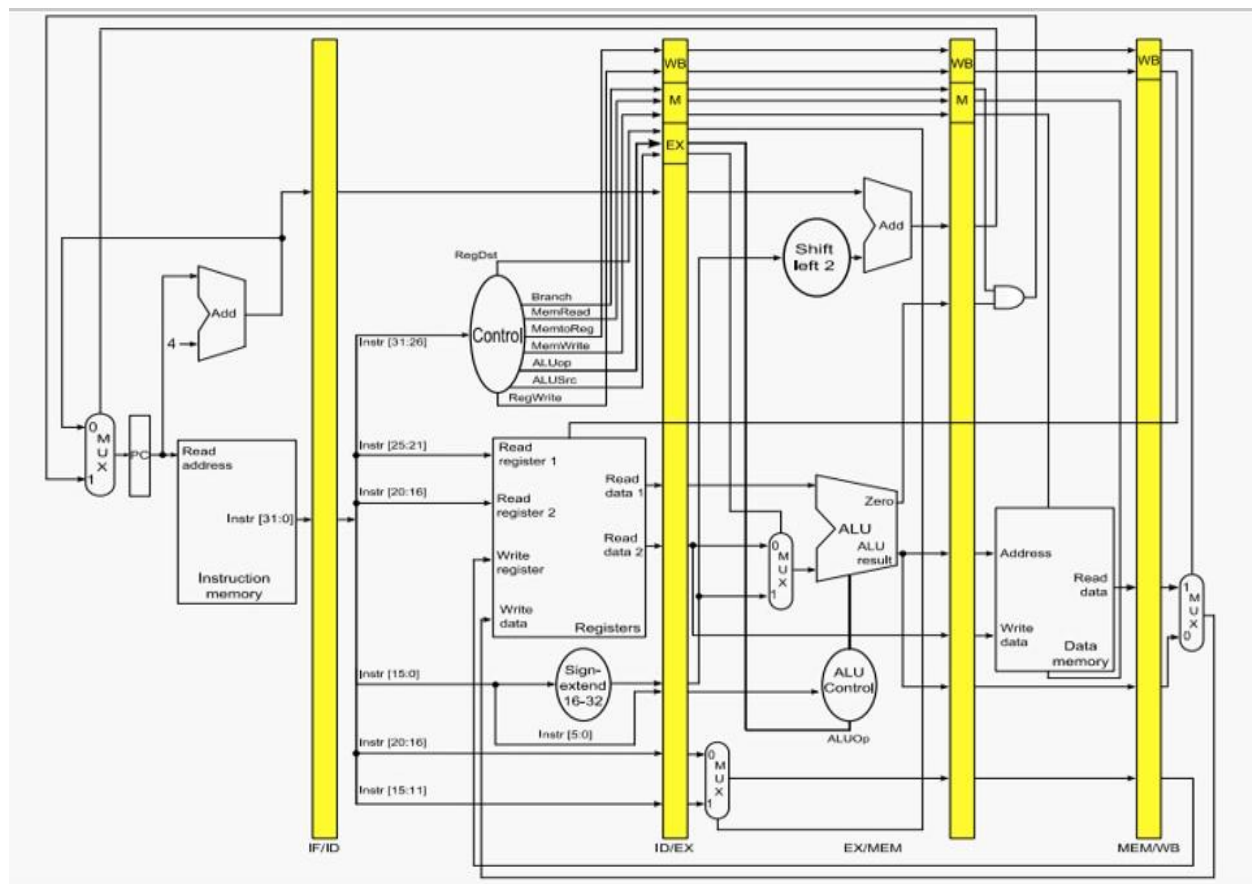
Sustainable Development Goals Mapping of CEP/CEA

Sr	SDGs	Mapping
1	No Poverty	
2	Zero Hunger	
3	Good Health and Wellbeing	✓
4	Quality Education	✓
5	Gender Equality	
6	Clean Water and Sanitation	
7	Affordable and Clean Energy	
8	Decency Work and Economic Growth	
9	Industry, Innovation and Infrastructure	✓
10	Reduced Inequities	
11	Sustainable Cities and Communities	
12	Responsible Consumption and Production	
13	Climate Action	
14	Life Below Water	
15	Life on Land	
16	Peace, Justice and Strong Institutions	
17	Partnerships for the Goals	✓

	National University of Technology	<u>CLO 3</u> <u>PLO 3</u>
	<u>Problem Based Learning</u>	
Batch 2021	Computer Engineering Department	
Semester 4th	Course: CEN 2014 Computer Organization and Architecture	
Name & ID	Weightage 10%	

Problem Statement:

Implement the Datapath and Control of MIPS processor architecture shown in the figure below using Verilog HDL. Evaluate how architectural and implementation design decisions influence performance and power dissipation of your processor design.



The tasks of the project are:

1. Design of building blocks of microprocessor, which consist of; ALU, Program Memory, Data Memory, Register File, Program Counter etc.
2. Design of Instruction Formats.
3. Design of Data-path, connecting microprocessor blocks as per the designed instruction formats.
4. Design of the Control Unit.
5. Construction of test program.

The PBL is mapped on the following attributes:

WP1	Preamble
WP2	Range of conflicting requirements
WP3	Depth of analysis required
WP4	Depth of knowledge required
WP5	Familiarity of issues
WP6	Extent of applicable codes

WP1: This involves understanding the MIPS architecture, setting project objectives, and planning the overall approach. It includes the initial research phase where students familiarize themselves with the MIPS instruction set and the fundamentals of processor design.

WP2: This attribute covers the balancing of various design requirements, such as performance, area, and power efficiency. For example, optimizing the ALU for speed might increase power consumption, so trade-offs must be carefully considered.

WP3: This involves a detailed analysis of the MIPS architecture, understanding the data flow, and control signal generation. It also includes analyzing how different instructions affect the datapath and control signals.

WP4: A deep understanding of digital design principles, Verilog HDL, and computer architecture is necessary. Students need to comprehend how high-level instructions translate to low-level hardware operations.

WP5: This attribute assesses the students' familiarity with common issues in processor design, such as hazards (data, control, and structural), timing issues, and resource contention. Familiarity with these issues helps in designing robust and efficient processors.

WP6: This involves adhering to best practices in Verilog coding, following design conventions, and ensuring the code is synthesizable and efficient. It also includes understanding and applying any relevant coding standards or guidelines specific to digital design and FPGA implementation.

Project Deliverables:

1. Verilog HDL code of MIPS Processor Datapath and Control.
2. Create a report outlining the project's details, such as the flowchart, objectives, and output.
3. Prepare the presentation.

Rubrics for Assessment of the PBL

<u>I Hardware</u>	Unacceptable (0-10)	Just acceptable (11-15)	Basic (16-20)	Good (21-25)	Excellent (26-30)
	Not Prepared	Incomplete	All hardware settings and signals are readable.	Hardware Results are completely working.	Hardware has a product shape with fine working.
<u>II. Demonstration</u>	(0-3)	(4-6)	(7-9)	(10-12)	(13-15)
	No Understanding	Couldn't demonstrate well / Little bit understanding	Understand Project well but couldn't demonstrate well	Understand well and Demonstrate well.	Have a complete understanding and can relate project to real life problems
<u>III. Presentation</u>	(0-5)	(6-10)	(11-20)	(20-25)	(26-30)
	Not Prepared	Incomplete presentation	Prepared but results are incorrectly explained.	Prepared well and could explain little bit.	Complete results are included (samples, testing etc.) and Present Diligently.
<u>IV. Project Report</u>	(0-5)	(6-10)	(11-15)	(16-20)	(21-25)
	No report /Report was not prepared in time	The report submitted but incomplete	The requirements of report are not properly addressed	Report meets all prescribed requirements with minimum mistakes	Report meets all requirements and prepared in original and creative way.

Sustainable Development Goals Mapping

Sr	SDGs	Mapping
1	No Poverty	
2	Zero Hunger	
3	Good Health and Wellbeing	
4	Quality Education	✓
5	Gender Equality	
6	Clean Water and Sanitation	
7	Affordable and Clean Energy	
8	Decency Work and Economic Growth	
9	Industry, Innovation and Infrastructure	✓
10	Reduced Inequities	
11	Sustainable Cities and Communities	
12	Responsible Consumption and Production	
13	Climate Action	
14	Life Below Water	
15	Life on Land	
16	Peace, Justice and Strong Institutions	
17	Partnerships for the Goals	✓