```
% Initialize parameters
clc;
clear;

% Define hyperparameters
learning_rate = 0.5; % Learning rate
N = 10000; % Maximum number of iterations
soft_high = 1; % Softened value for 1
soft_low = 0;  % Softened value for 0

input_patterns = [soft_high, soft_low, soft_low, soft_low;
                  soft_low, soft_high, soft_low, soft_low;
                  soft_low, soft_low, soft_high, soft_low;
                  soft_low, soft_low, soft_low, soft_high];
targets = input_patterns;

% Initialize weights
W1 = rand(4, 2) - 0.5; % Input-to-hidden weights
W2 = rand(2, 4) - 0.5; % Hidden-to-output weights


 % Weight matrix from hidden to output layer, randomly initialized

% Store error values
errors = zeros(1, N);

% Sigmoid activation function and its derivative
sigmoid = @(x) 1 ./ (1 + exp(-x));
sigmoid_derivative = @(y) y .* (1 - y);

% Training process
for n = 1:N
    total_error = 0; % Total error for the current iteration

    for d = 1:4
        % Forward propagation
        input = input_patterns(d, :)'; % Current input pattern
        target = targets(d, :)'; % Current target output

        % Hidden layer computations
        net_hidden = W1' * input;
        output_hidden = sigmoid(net_hidden);

        % Output layer computations
        net_output = W2' * output_hidden;
        output = sigmoid(net_output);

        % Compute error
        error = target - output;
        total_error = total_error + sum(error .^ 2);

        % Backpropagation
        delta_output = error .* sigmoid_derivative(output);
        delta_hidden = (W2 * delta_output) .* sigmoid_derivative(output_hidden);

        % Update weights
        W2 = W2 + learning_rate * (output_hidden * delta_output');
        W1 = W1 + learning_rate * (input * delta_hidden');
    end
```

```
    % Record the error
    errors(n) = total_error;
end

% Plot error curve
figure;
plot(1:N, errors);
xlabel('Number of Iterations');
ylabel('Total Error');
title('Error vs. Number of Iterations');
```