Lab 8 and 9: Basics of Number System Conversion

1. Introduction

In modern precision agriculture, farm machinery relies on various electronic control units (ECUs) to manage critical functions such as engine performance, hydraulic control, and automated guidance systems. These ECUs communicate through protocols like RS-232, CAN Bus, and ISOBUS to share sensor data and control signals. **Understanding number system conversions** is crucial for diagnosing faults in these communication networks. Many agricultural diagnostic tools interface with machines via RS-232 or CAN Bus, and interpreting **binary**, **decimal**, and **hexadecimal** values is essential for troubleshooting electronic systems in tractors, harvesters, and other precision agriculture equipment.

i. **Diagnostic Tool Understanding**: Agricultural diagnostic systems frequently use RS-232 or CAN Bus to extract real-time machine data. By understanding number conversions, students can interpret raw sensor values and diagnostic trouble codes.
ii. **ECU Communication Analysis**: Electronic components in farm machinery send and receive data using binary and hexadecimal formats. Converting these values into decimals allows technicians to identify faults.

Fig. 1 Tractor ECU

iii. **Data Transmission Principles**: Simulating RS-232 communication provides students with hands-on experience with serial data transmission, helping them understand how telemetry and remote monitoring systems work in agriculture.

This lab will lay the foundation for analyzing and troubleshooting digital communication networks in farm machinery, helping you to gain practical knowledge relevant to precision agriculture and smart farming.

A number system is a way of representing numbers using a consistent set of symbols. The most common number systems used in computing and digital electronics include:

i. **Decimal (Base-10)** – The standard system we use daily, consisting of digits 0-9.
ii. **Binary (Base-2)** – Used in digital electronics and computing, consisting of only 0s and 1s.
iii. **Hexadecimal (Base-16)** – A compact representation used in computing, consisting of digits 0-9 and letters A-F.

Binary to Decimal Conversion

Each digit in a binary number represents a power of 2. To convert binary to decimal:
Example:
Binary: 1011
Decimal: $(1 \times 2^3) + (0 \times 2^2) + (1 \times 2^1) + (1 \times 2^0) = 8 + 0 + 2 + 1 = 11$

Decimal to Binary Conversion

Use repeated division by 2, recording remainders from bottom to top.
Example:
Convert 25 to binary:
25 / 2 = 12 remainder 1
12 / 2 = 6 remainder 0
6 / 2 = 3 remainder 0
3 / 2 = 1 remainder 1
1 / 2 = 0 remainder 1
Binary: 11001

Decimal to Hexadecimal Conversion

Use repeated division by 16, converting remainders to hex digits (0-9, A-F).
Example:
Convert 250 to hexadecimal:
250 / 16 = 15 remainder 10 (A in hex)
15 / 16 = 0 remainder 15 (F in hex)
Hexadecimal: FA

2. Materials and Methods

2.1 Materials Required

- Pen and paper for hand calculations.
- MATLAB/Simulink software for model verification.
- Computer with Simulink installed

2.2 Methods

1. You will first solve number conversion problems manually.

2. Then you will verify your results using Simulink models.

3. Experiments

3.1 Handwritten Number Conversion Practice

Solve the following:
1. Convert 110101 (binary) to decimal.
2. Convert 45 (decimal) to binary.
3. Convert 198 (decimal) to hexadecimal.
4. Convert A3 (hexadecimal) to decimal.

3.2 Simulink Models for Number System Conversion

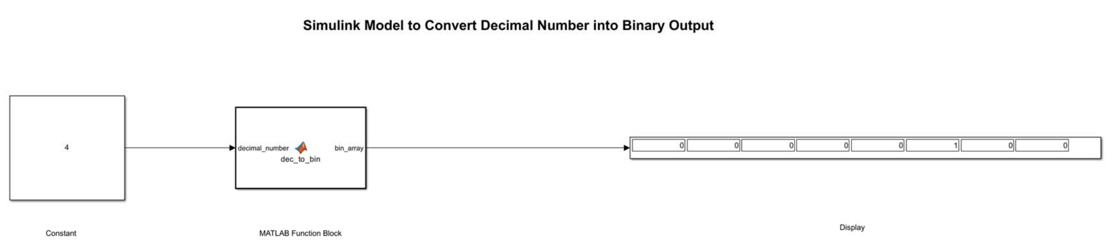

Fig. 2 Simulink model for number system conversion from decimal to binary

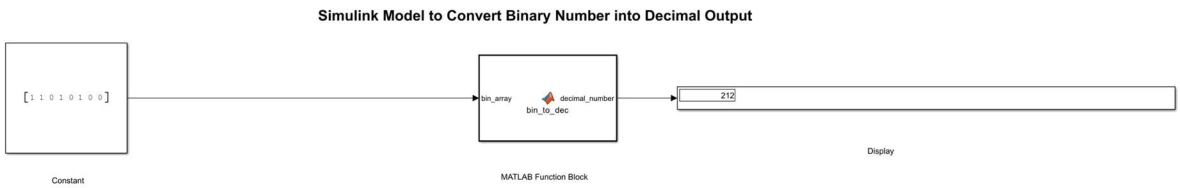

Fig. 3 Simulink model for number system conversion from binary to decimal

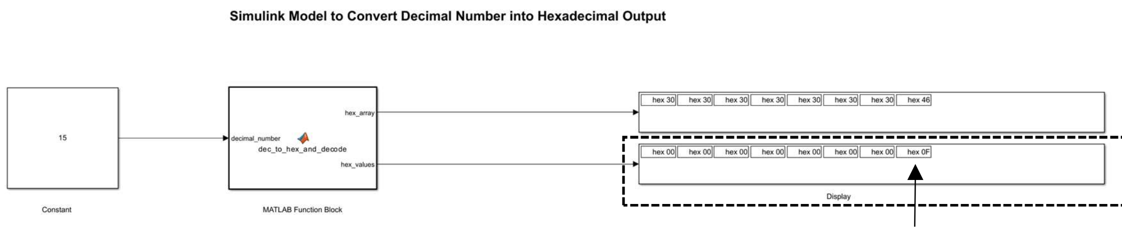

Fig. 4 Simulink model for number system conversion from decimal to hexadecimal

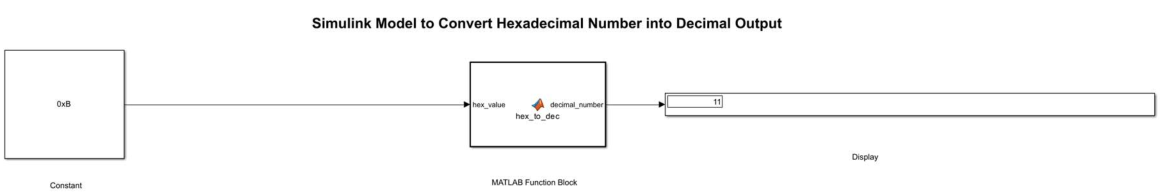

Fig. 5 Simulink model for number system conversion from hexadecimal to decimal

Modify the above constant value to check your above calculation of decimal number 45 to binary. Make sure to show it to the lab TAs before moving to the next step. Now using similar approach build Simulink models for the rest of the conversions and validate your calculations. You can use the given MATLAB functions in your model for reference. **Please note that except for the**

decimal to hexadecimal model, the display block setting will be "short" while for this will be "hex(stored integer)".

MATLAB function for the MATLAB function block [decimal to binary]

```
function bin_array = dec_to_bin(decimal_number)
    % Converts a decimal number to a 16-bit binary array (uint8 format)

    % Define a fixed-size binary output of 16 bits
    bin_array = zeros(1, 8, 'uint8');

    % Convert decimal to binary string
    bin_str = dec2bin(decimal_number);

    % Convert to numeric uint8 array
    bin_num = uint8(bin_str - '0');

    % Fill the fixed-size output with binary digits
    bin_array(end - length(bin_num) + 1:end) = bin_num;
end
```

MATLAB function for the MATLAB function block [decimal to hexadecimal]

```
function [hex_array, hex_values] = dec_to_hex_and_decode(decimal_number)
    %#codegen
    % Converts a decimal number to an 8-character fixed-size hexadecimal array
    % and then decodes it back to actual hex values.

    % Ensure input is uint32 (prevents type mismatch error)
    decimal_number = uint32(decimal_number);

    % Convert decimal to hexadecimal string (fixed width of 8)
    hex_str = sprintf('%08X', decimal_number);

    % Convert hex string to ASCII-encoded numeric values
    hex_array = uint8(zeros(1, 8));  % Define fixed-size output
    hex_num = uint8(hex_str);  % Convert "0000000F" to ASCII values
    hex_array(end - length(hex_num) + 1:end) = hex_num;  % Right-align

    % Decode ASCII-encoded hex back to actual hex values
    decoded_hex_str = char(hex_array);  % Convert ASCII back to string
    num_values = sscanf(decoded_hex_str, '%2X').';  % Convert hex string to numbers

    % Define a fixed-size output for decoded hex values
    hex_values = uint8(zeros(1, 8));
    hex_values(end - length(num_values) + 1:end) = num_values;
end
```

MATLAB function for the MATLAB function block [binary to decimal]

```
function decimal_number = bin_to_dec(bin_array)
    % Converts a binary array (uint8) to a decimal integer

    % Convert uint8 array to a character string of '0' and '1'
    bin_str = char(bin_array(:)' + '0');  % Ensure row vector

    % Convert binary string to decimal
    decimal_number = double(bin2dec(bin_str)); % Ensure scalar output
end
```

MATLAB function for the MATLAB function block [hexadecimal to decimal]

```
function decimal_number = hex_to_dec(hex_value)
    %#codegen
    % Converts a hexadecimal number (given in 0x notation) to decimal.

    % Ensure input is treated as an integer (uint32 for large values)
    decimal_number = uint32(hex_value);
end
```

Expected Results

In your Display block you will see output corresponding to the values you provide in the constant block.

3. Discussion

In this lab, you explored the fundamental concepts of number system conversions, which are essential in understanding how different parts of farm machinery system communicate. Through Simulink models and MATLAB function blocks, various conversions between binary, decimal, and hexadecimal representations were implemented and analyzed. The number system conversion exercises provided an understanding of how numerical data is represented in computing systems and how different formats are used in digital logic and microcontroller programming. The binary-to-decimal, decimal-to-binary, and hexadecimal conversions helped you develop skills that are crucial when working with embedded systems like ECU, digital circuits, and machine automation. The use of fixed-size arrays ensured compatibility with Simulink, allowing for structured data processing. A key challenge in the implementation was handling ASCII-encoded hexadecimal values and ensuring proper data representation. By developing MATLAB functions for decimal-to-hexadecimal conversion and decoding hexadecimal ASCII values, you were able to properly format and interpret numerical data, reinforcing your understanding of how computers process and store information. This lab is highly relevant to electrical diagnostics in farm machinery, where digital systems are used to monitor sensor data, control actuators, and process information. Understanding how numbers are converted and interpreted in different formats is critical for working with microcontrollers, programmable logic controllers (PLCs), and embedded systems that rely on digital signals and numerical data processing.

4. Conclusion

This lab successfully demonstrated the importance of number system conversions in digital computing and control systems. By implementing Simulink models and MATLAB functions, you gained hands-on experience in converting numerical values between binary, decimal, and hexadecimal formats, which are widely used in digital electronics, embedded computing and communication systems. The ability to efficiently convert between number systems ensures accurate data processing, which is fundamental in designing and troubleshooting electronic control units (ECUs), sensor interfaces, and automation systems. Overall, this lab enhanced your understanding of fundamental computing principles and their applications in modern agricultural technology. The knowledge gained will be directly applicable to diagnosing and maintaining digital control systems in farm machinery, making it a valuable learning experience in the field of precision agriculture and electrical diagnostics of farm machinery systems

5. Lab Submission

- Submit a short report summarizing **your observations** and **screenshots** of the outputs.