Design and Implementation of 3-bit High Speed Flash ADC

eSim Research Migration Project

Boddu Ajay
Rajiv Gandhi University of Knowledge Technologies, Nuzvid

Abstract

The Design and Implementation of a 3-bit High-Speed Flash ADC focuses on developing an ultra-fast analog-to-digital converter (ADC) suitable for high-frequency applications. The Flash ADC is a mixed-signal circuit, integrating both analog and digital components to achieve rapid and accurate signal conversion. The design employs a resistor-ladder network for reference voltage generation, a comparator array for parallel analog signal comparison, and a priority encoder to convert the thermometer code into a 3-bit binary output. This architecture ensures minimal latency and high-speed performance, making it ideal for applications like real-time data acquisition, communication systems, and signal processing. The proposed implementation is optimized for speed and efficiency, leveraging parallel processing techniques to enhance throughput while maintaining circuit simplicity.

Working Principle

The operation of a 3-bit Flash ADC is based on the simultaneous comparison of an analog input voltage with multiple reference voltages. The circuit follows these steps:

Resistor Ladder Operation

- A voltage divider generates seven reference voltages from a fixed V_{ref}.
- These reference voltages are applied to a set of comparators.

Comparator Operation

- Each comparator compares the input voltage (V_{in}) with its respective reference voltage.
- If V_{in} is higher than the reference voltage, the comparator outputs logic '1'; otherwise, it outputs logic '0'.
- This results in a thermometer code output.

Priority Encoding

- The thermometer code is fed into a priority encoder, which converts it into a 3-bit binary value.

- The highest comparator that outputs ‘1’ determines the corresponding digital output.

Since all comparisons happen simultaneously, Flash ADCs exhibit ultra-fast conversion speeds. However, this comes at the cost of increased hardware complexity due to the large number of comparators required for higher-bit resolutions.

0.1 Circuit Diagram

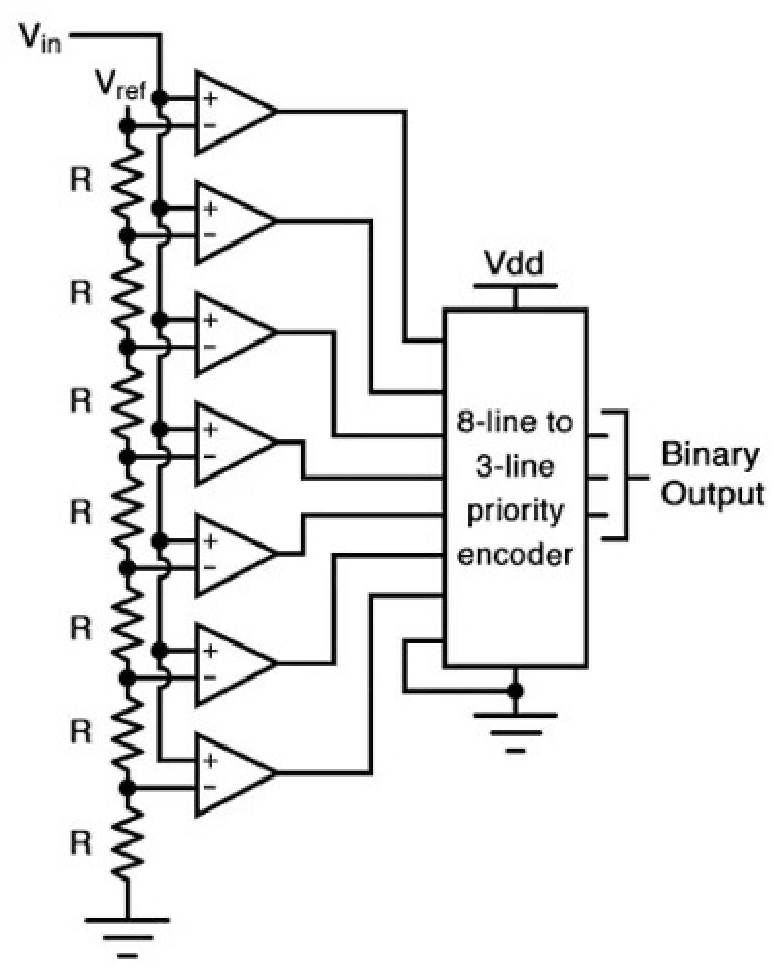


Figure 1: 3-Bit Flash ADC

0.2 Circuit Schematic in eSim

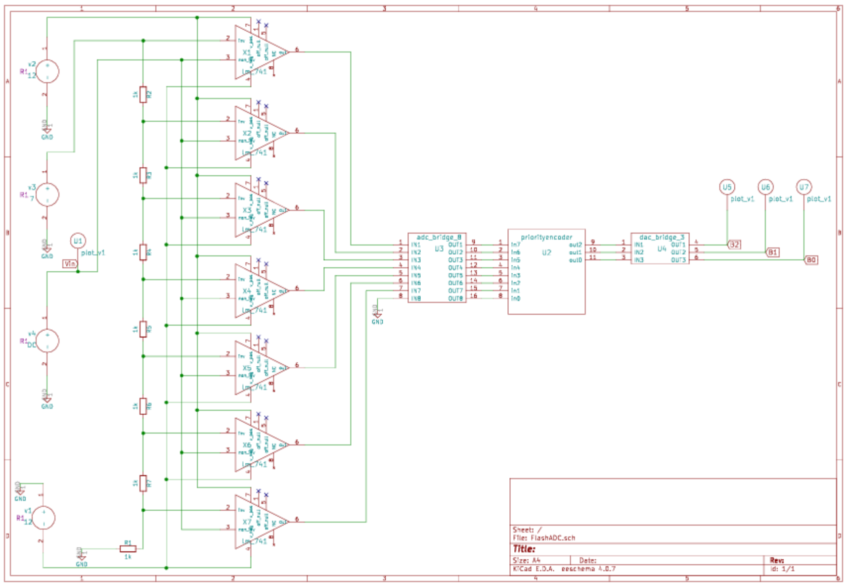


Figure 2: 3-Bit Flash ADC Schematic in eSim

0.3 Digital Implementation of Priority Encoder

In the design of the 3-bit Flash ADC, the priority encoder plays a crucial role in converting the thermometer code generated by the comparator array into a 3-bit binary output. Since multiple comparators may produce a high output (1), the priority encoder ensures that only the highest active comparator determines the final digital output.

The priority encoder was implemented using Verilog, ensuring a fast and efficient conversion of the thermometer code to binary. The logic prioritizes the most significant bit (MSB), ignoring lower active bits.

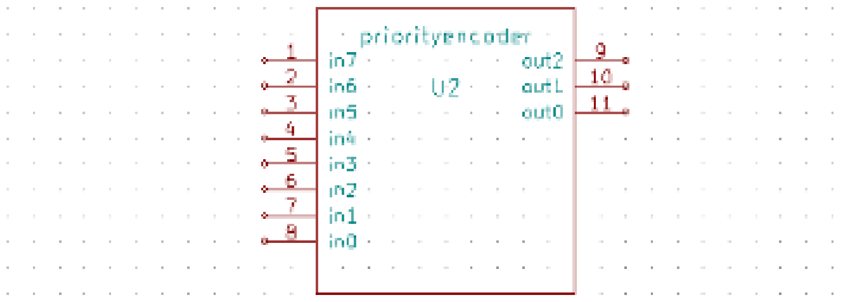


Figure 3: Priority Encoder

0.4 Simulation Results of Priority Encoder

The priority encoder designed for the 3-bit Flash ADC was simulated using Makerchip, an online platform for designing and verifying digital circuits. Makerchip provides an interactive environment for developing TL-Verilog and Verilog-based designs, enabling real-time simulation and debugging.

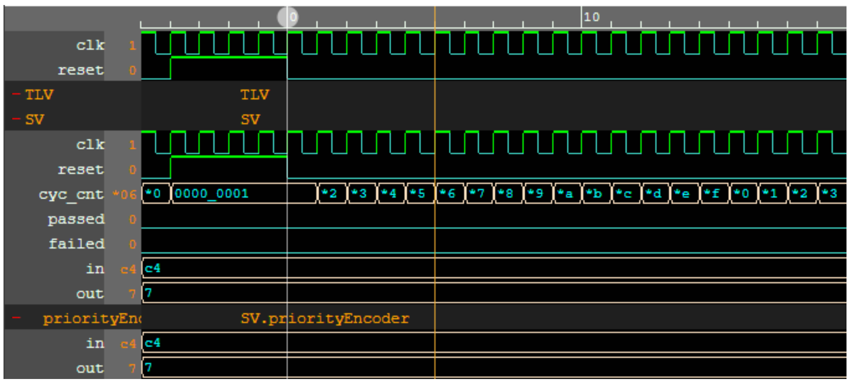
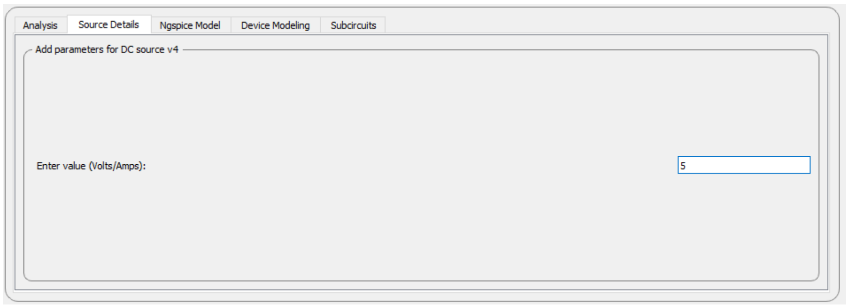


Figure 4: Priority Encoder-Simulation Result

0.5 Analysis and Results of Flash ADC in Ngspice

The designed 3-bit Flash ADC was simulated in Ngspice, an open-source mixed-signal circuit simulator, to analyze its performance. The key aspects of the analysis includes transient simulations to verify the correct operation of the ADC under different input conditions.

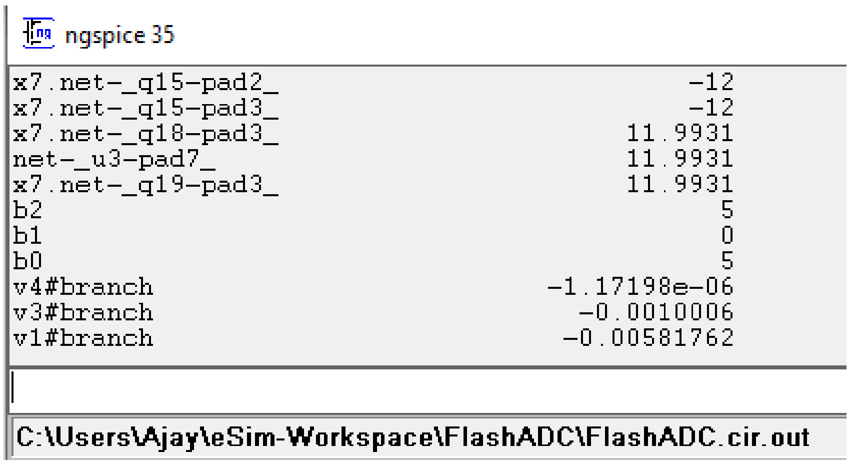
0.5.1 Test Case - 1



Add parameters for DC source v4

Enter value (Volts/Amps): 5

Figure 5: Input Vin



```
x7.net-_q15-pad2_                          -12
x7.net-_q15-pad3_                          -12
x7.net-_q18-pad3_                      11.9931
net-_u3-pad7_                          11.9931
x7.net-_q19-pad3_                      11.9931
b2                                           5
b1                                           0
b0                                           5
v4#branch                         -1.17198e-06
v3#branch                           -0.0010006
v1#branch                          -0.00581762
```

C:\Users\Ajay\eSim-Workspace\FlashADC\FlashADC.cir.out

Figure 6: Digtal Output analyzed in NgSpice

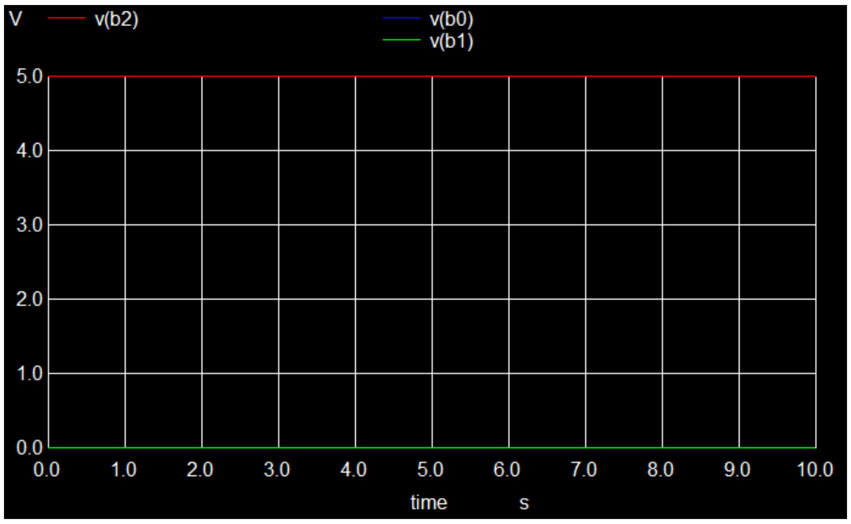
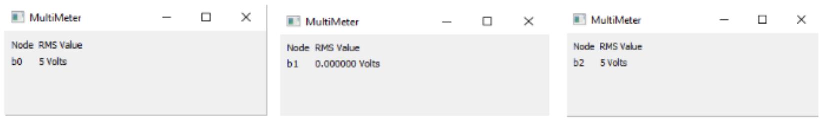


Figure 7: Digital Output for Analog Input Vin



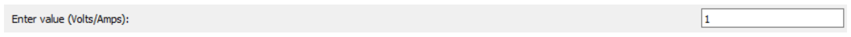
```
MultiMeter
Node  RMS Value
b0    5 Volts
```

```
MultiMeter
Node  RMS Value
b1    0.000000 Volts
```

```
MultiMeter
Node  RMS Value
b2    5 Volts
```

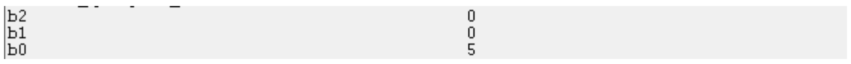
Figure 8: Multimeter Readings of digital Output from Python Plot

0.5.2 Test Case - 2



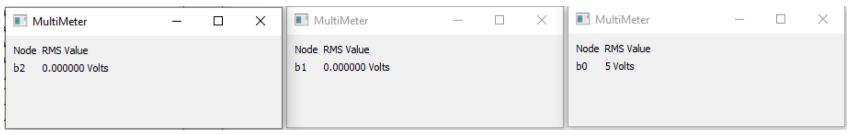
Enter value (Volts/Amps): 1

Figure 9: Input Vin



```
b2                                        0
b1                                        0
b0                                        5
```

Figure 10: Digtal Output analyzed in NgSpice



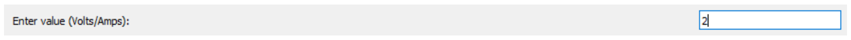
```
MultiMeter
Node  RMS Value
b2    0.000000 Volts
```

```
MultiMeter
Node  RMS Value
b1    0.000000 Volts
```

```
MultiMeter
Node  RMS Value
b0    5 Volts
```

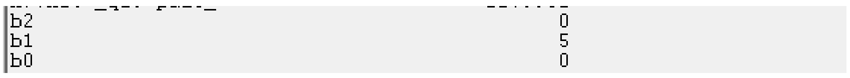
Figure 11: Multimeter Readings of digital Output from Python Plot

0.5.3 Test Case - 3



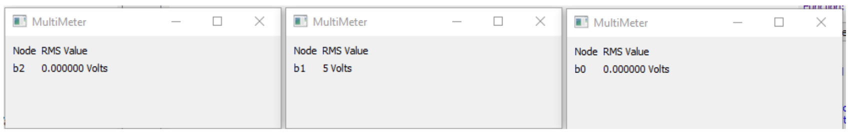
Enter value (Volts/Amps): 2

Figure 12: Input Vin



```
b2                                        0
b1                                        5
b0                                        0
```

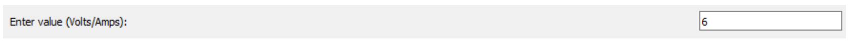
Figure 13: Digtal Output analyzed in NgSpice



```
MultiMeter                MultiMeter            MultiMeter
Node  RMS Value           Node  RMS Value       Node  RMS Value
b2    0.000000 Volts      b1    5 Volts         b0    0.000000 Volts
```

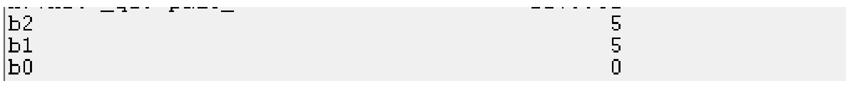
Figure 14: Multimeter Readings of digital Output from Python Plot

0.5.4 Test Case - 4



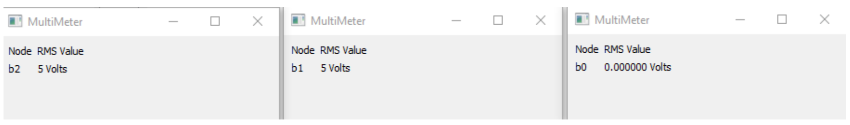
Enter value (Volts/Amps): 6

Figure 15: Input Vin



```
b2                                        5
b1                                        5
b0                                        0
```

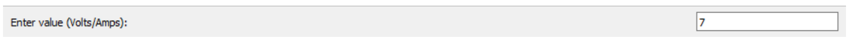
Figure 16: Digtal Output analyzed in NgSpice



```
MultiMeter                MultiMeter            MultiMeter
Node  RMS Value           Node  RMS Value       Node  RMS Value
b2    5 Volts             b1    5 Volts         b0    0.000000 Volts
```

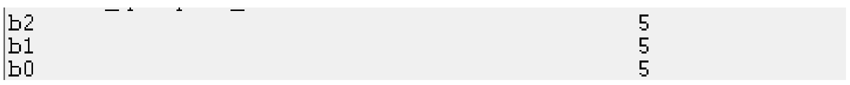
Figure 17: Multimeter Readings of digital Output from Python Plot

0.5.5 Test Case - 5



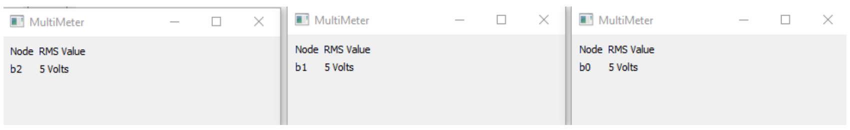
Enter value (Volts/Amps): 7

Figure 18: Input Vin



```
b2                                        5
b1                                        5
b0                                        5
```

Figure 19: Digtal Output analyzed in NgSpice



```
MultiMeter                MultiMeter            MultiMeter
Node  RMS Value           Node  RMS Value       Node  RMS Value
b2    5 Volts             b1    5 Volts         b0    5 Volts
```

Figure 20: Multimeter Readings of digital Output from Python Plot

0.6 Response of Flash ADC to a Sine Wave Input

A sine wave signal was applied to the 3-bit Flash ADC to analyze its performance. The comparators continuously compared the input voltage with reference levels, generating a thermometer code. The priority encoder then converted this into a 3-bit digital output. The ADC output followed a stepwise approximation of the sine wave, demonstrating successful analog-to-digital conversion with expected quantization effects.

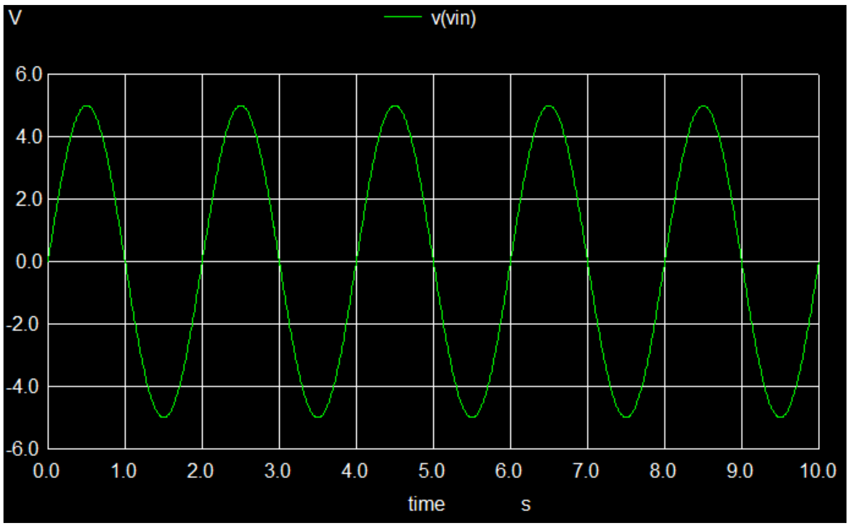


Figure 21: Sine wave as Input

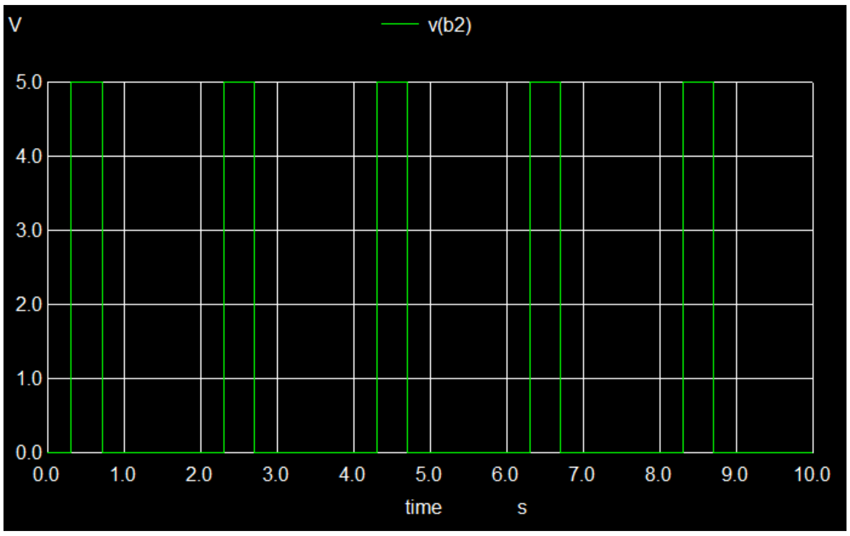


Figure 22: Digital Output 'B2'

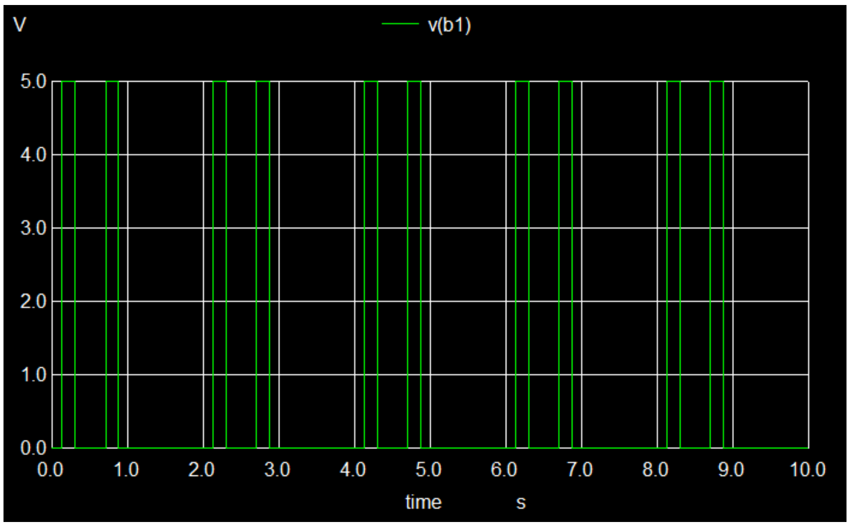


Figure 23: Digital Output ’B1’

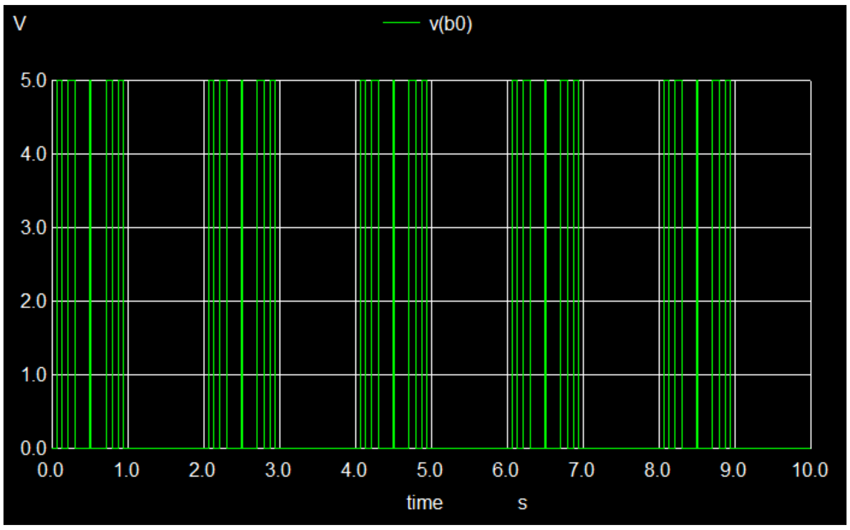


Figure 24: Digital Output ’B0’

References

[1] Mirza Nemath Ali Baig, Rakesh Ranjan, Design Impementation of 3-Bit High Speed Flash ADC for Wireless LAN Applications, International Journal of Advanced Research in Computer and Communication Engineering, 2017, Available at : `https://www.researchgate.net/publication/318286256_Design_Implementation_of_3-Bit_High_Speed_Flash_ADC_for_Wireless_LAN_Application`)